Aplikasi Penerjemah Bahasa Inggris – Indonesia dengan *Optical Character Recognition* Berbasis Android

Anisa Eka Utami[1)], Oky Dwi Nurhayati[2)], Kurniawan Teguh Martono[2)]
Program Studi Sistem Komputer Fakultas Teknik Universitas Diponegoro
Jalan Prof. Sudharto, Tembalang, Semarang, Indonesia
anisaekau@gmail.com

Abstrak - ***Perangkat lunak untuk pengenalan karakter yang terdapat dalam ponsel pintar khususnya berbasis Android dikembangkan dengan penekanan pada mobilitas, portabilitas, ruang penyimpanan, perangkat keras, dan keterbatasan jangkauan dapat dipecahkan. Akan tetapi, kinerja sebuah ponsel pintar berbasis Android dan komputer berbeda maka kecepatan pengenalan karakter juga akan berpengaruh. Masalah ini tampaknya akan menunjukkan suatu solusi, yaitu dengan salah satu inovasi yang diterapkankan ke dalam perangkat Android dengan teknologi OCR (Optical Character Recognition).***

Perencanaan sistem menggunakan pengembangan perangkat lunak berorientasi pemakaian ulang karena menggunakan komponen yang dapat dipakai ulang dalam pengembangannya. Sistem ini dibuat dengan memanfaatkan engine Tesseract OCR yang dikembangkan oleh Google bersifat open source. Perangkat lunak yang digunakan untuk merancang layout dan implementasi sistem, yaitu menggunakan lingkungan pengembang Android Studio yang ditulis dengan bahasa pemrograman Java dan XML.

Pengujian aplikasi penerjemah dengan OCR ini menggunakan metode white box dan menghitung akurasi pendeteksian karakter. Hasil perhitungan presentase akurasi deteksi karakter yang diberikan aplikasi terhadap keseluruhan sampel yang diuji mencapai 97,5%.

Kata Kunci : OCR, Kecerdasan buatan, Android, engine Tesseract OCR, Android Studio

I. PENDAHULUAN

BAHASA merupakan sebuah perwujudan manusia yang berkomunikasi. Definisi ini memiliki istilah yang penting, masing-masing dibagi menjadi beberapa detail, diantaranya adalah istilah sebagai sistem, tanda, vokal, konvensional, manusia, komunikasi.[1] Bahasa juga merupakan fenomena yang yang melibatkan proses-proses yang bermacam-macam seperti pengenalan bunyi, cetakan huruf, dan juga komunikasi emosional melalui irama bicara dan perubahan nada bicara.[2]

Kebanyakan orang mempelajari suatu bahasa, misalnya bahasa Inggris dengan menggunakan buku teks yang berasal dari institusi pendidikan atau ada pula secara otodidak baik seperti dengan mendengarkan lagu-lagu maupun menonton film berbahasa Inggris. Bagi sebagian orang yang baru mengenal bahasa Inggris akan mengalami kesulitan dalam mengartikan suatu kata bahasa Inggris karena terdapat banyaknya suku kata dalam bahasa Inggris.

Program – program pengenalan karakter diketahui melalui suatu citra masukan dengan sebuah alat pemindai atau kamera digital dan perangkat lunak dalam komputer yang masih terdapat masalah dalam ukuran ruang penyimpanan yang terdapat pada komputer dan alat pemindai. Untuk mengatasi keterbatasan pada komputer yang biasanya menghabiskan memori yang besar maka sebuah sistem pengenalan karakter berbasis ponsel pintar diajukan.

OCR dapat diimplementasikan pada *platform* Android tentunya ini memanfaatkan sumber dari Google yang telah menyediakan banyak sumber referensi dan salah satunya adalah menggunakan Tesseract Engine sebagai metode untuk menerapakan teknologi OCR pada platform Android. Pemanfaatan sumber yang sudah ada akan dibuat sebuah aplikasi yang dapat mengubah citra teks yang bisa diimplementasikan pada perangkat Android. Aplikasi ini nantinya akan menerjemahkan suatu objek berupa karakter teks/tulisan dalam bahasa Inggris yang tertangkap oleh kamera ke dalam bahasa Indonesia.

Pengimplementasian OCR pada perangkat Android akan dimanfaatkan sebagai penerjemah bahasa Inggris – Indonesia secara online yang terhubung dengan *library* Microsoft Translator dan Google Translator. Dalam pembuatan tugas akhir ini pembahasan masalah memiliki batasan pada permasalahan berikut :

1. Aplikasi hanya dapat memberikan hasil terjemahan bahasa Inggris ke dalam bahasa Indonesia saja.
2. Aplikasi ini dapat menerjemahkan bahasa Inggris ke dalam Indonesia dengan OCR berbasis Android ini dengan memanfaatkan Tesseract OCR Engine yang bersifat *open source project*.
3. Aplikasi ini dibangun untuk perangkat Android dengan bahasa pemrograman Java dan XML menggunakan lingkungan pengembang Android Studio 1.2.
4. Aplikasi dijalankan pada perangkat Android dengan sistem operasi minimum versi 4.1 Android Jelly Bean hingga versi 5.0 Android Lolipop.

II. LANDASAN TEORI

A. *Penelitian Terdahulu*

Penulis melakukan penelitian ini didorong oleh hasil penelitian yang sebelumnya pernah dilakukan. Penelitian-penelitian yang membahas OCR yang diterapkan pada *mobile phone* adalah sebagai berikut:

1. Penelitian dari Dwi Cahyo Nugroho, Mahmud Dwi Sulistyo, dan Bedy Purnama yang dipublikasikan dalam Seminar Nasional Sistem Informasi Indonesia Universitas Telkom pada tanggal 22 September 2014. Penelitian ini membahas *Optical Character Recognition* yang dijalankan pada *smartphone* dengan menggunakan analisis kontur dan *feature preselection* yang memanfaatkan perangkat lunak FastCV (OpenCV). Hasil keluaran penelitian ini adalah aplikasi OCR yang dijalankan pada Windows Phone untuk mendeteksi karakter yang ditangkap kamera.

2. Penelitian dari Aulia Faqih Rafa'i yang dipublikasikan dalam *Regsist Conference* UIN Sunan Kalijaga pada tanggal 22 Oktober 2013. Penelitian ini membahas bagaimana mengimplementasikan penerjemah dan penambah harakat kitab klasik/kitab kuning beserta analisisnya menggunakan perangkat Android yang memanfaatkan Tesseract OCR engine. Hasil keluaran aplikasi berupa aplikasi penerjemah dan penambah harakat pada kitab klasik/kitab kuning.

Perbedaan kedua penelitian tersebut dengan penelitian yang dilakukan oleh penulis adalah penulis membahas bagaimana cara mengimplementasikan Tesseract OCR *engine* untuk menerjemahkan kata dari bahasa Inggris ke dalam bahasa Indonesia serta pengembangannya yang dijalankan pada perangkat Android serta menganalisis tingkat akurasi dari hasil keluaran aplikasi. Hasil keluaran yang diperoleh dari aplikasi adalah menerjemahkan kata bahasa Inggris ke bahasa Indonesia.

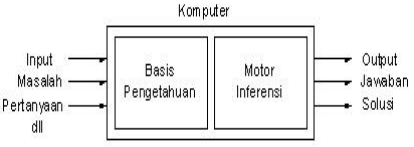
B. Kecerdasan Buatan

Kecerdasan buatan merupakan sebuah studi yang mempelajari bagaimana membuat komputer melakukan hal-hal yang pada saat ini dapat dilakukan lebih baik oleh manusia.[3] Menurut John McCarthy (1956), AI adalah ilmu untuk mengetahui dan memodelkan proses-proses berpikir manusia dan mendesain mesin agar dapat menirukan perilaku manusia. Cerdas berarti memiliki pengetahuan ditambah pengalaman, penalaran (bagaimana membuat keputusan dan mengambil tindakan), dan moral yang baik. [4]

Mesin bisa cerdas dengan diberi bekal pengetahuan sehingga mempunyai kemampuan untuk menalar. Hal-hal yang dibutuhkan untuk membuat aplikasi kecerdasan buatan terdiri dari 2 (dua) bagian atau komponen utama yaitu:

1. Basis pengetahuan (*knowledge base*) : berisi fakta-fakta, teori, pemikiran dan hubungan antar satu dengan yang lainnya.
2. Motor inferensi (*inference engine*) : kemampuan menarik kesimpulan berdasarkan pengetahuan dan pengalaman.

Penerapan konsep kecerdasan buatan pada komputer ditunjukkan seperti pada Gambar 1.



Gambar 1. Penerapan konsep kecerdasan buatan di komputer[4]

C. Pemahaman Bahasa Alami

Pemrosesan bahasa alami (*Natural Language Processing* – NLP) dapat didefinisikan sebagai kemampuan suatu komputer untuk memproses bahasa, baik lisan maupun tulisan yang digunakan manusia dalam percakapan sehari-hari. Untuk proses komputasi, bahasa harus direpresentasikan sebagai rangkaian simbol yang memenuhi aturan tertentu.[5] Banyak manfaat yang dapat dicapai dari ketersediaan aplikasi teknologi untuk bahasa, khususnya untuk Bahasa Indonesia Berikut ini adalah beberapa contoh aplikasi yang dapat dikembangkan menggunakan teknologi bahasa : [6]

1. Alat bantu membaca untuk tunanetra

Alat bantu membaca bagi tunanetra mempunya masukan berupa teks tercetak (misalnya buku) dan mempunyai keluaran berupa ucapan dari teks tercetak yang diberikan. Pada prinsipnya, ada dua komponen utama, yaitu bagian "pengenal karakter" yang menggunakan teknolgoi OCR (*Optical Character Recognition*), serta bagian TTS (*Text to Speech*).

2. Alat bantu untuk tunawicara

Alat bantu membaca bagi tunawicara mempunyai masukan posisi tangan yang dideteksi oleh suatu sensor dan unit identifikasi. Rangkaian huruf yang diidentifikasi akan disusun membentuk satu kata yang pada akhirnya akan diumpankan pada bagian TTS.

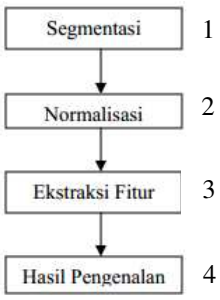
3. *Online translator*

Online translator yang dimaksud di sini adalah penerjemah yang secara otomatis dapat menerjemahkan kalimat lisan dari suatu bahasa alami (misalnya Bahasa Inggris) menjadi ucapan hasil terjemahannya dalam bahasa alami lainnya (misalnya Bahasa Indonesia). Online translator terdiri dari tiga bagian. Bagian pertama, speech recognition, berfungsi mengenali rangkaian kata dari bahasa sumber menjadi teks dalam bahasa sumber. Bagian berikutnya adalah penerjemah teks ke teks. Hasil bagian kedua ini adalah kalimat bahasa tujuan yang masih berupa teks.

D. Optical Character Recognition (OCR)

Aplikasi OCR sering digunakan untuk mengidentifikasi citra huruf untuk kemudian diubah ke dalam bentuk file tulisan. Aplikasi OCR juga digunakan dunia industri seperti industri elektronik yang sebagai pengenal label-label yang ada pada circuit board.[7] OCR biasa digunakan untuk bidang penelitian dalam pengenalan pola, kecerdasan buatan (*artificial intelligence*) dan *computer vision*. Sistem OCR memerlukan pelatihan untuk membaca font yang spesifik, versi awal harus diprogrram dengan gambar karakter masing-masing, dan bekerja pada satu font pada suatau waktu. Sistem cerdas dengan tingkat pengenalan dengan akurasi yang tinggi bagi font yang paling umum sekarang. [8]

Secara umum, proses pengenalan karakter yang dilakukan oleh OCR ditunjukkan pada gambar 2. [8]



Gambar 2. Proses pengenalan oleh OCR secara umum

Keterangan dari gambar 2 adalah sebagai berikut :

1. Proses segmentasi bertujuan untuk memisahkan wilayah objek dengan wilayah latar belakang agar objek dalam citra mudah dianalisis dalam rangka mengenali objek. Dengan demikian, citra yang besar terdiri dari objek karakter dapat disegmentasi menjadi masing-masing karakter.
2. Proses selanjutnya adalah normalisasi yang di dalamnya masih terdapat dua proses, yaitu :
   - *Scalling*, adalah fungsi untuk mengubah ukuran suatu gambar dimana *scalling* merupakan istilah yang cenderung untuk memperbesar gambar dan *shrink* untuk memperkecil gambar.
   - *Thinning*, adalah operasi morfologi yang digunakan untuk menghapus piksel *foreground* yang terpilih dari

gambar biner, biasanya digunakan untuk proses mencari rangka dari sebuah objek.

3. Langkah selanjutnya adalah ekstraksi fitur, yaitu proses analisis citra dalam mengidentifikasi sifat-sifat yang melekat dari tiap-tiap karakter atau disebut juga dengan fitur dari sebuah objek yang terdapat dalam citra. Karakteristik ini digunakan dalam mendeskripsikan sebuah objek atau atribut dari sebuah objek, kemudian fitur yang dimiliki oleh karakter dapat digunakan sebagai proses pengenalan.
4. Setelah ketiga proses tersebut selesai dilakukan maka OCR siap untuk melakukan tahap pengenalan dan akan memberikan keluaran atau hasil pengenalan karakter angka maupun huruf.

Setelah proses pengenalan maka dapat dilakukan pengujian terhadap sistem dengan OCR maka akan dilakukan perhitungan performansi sistem yaitu dengan menguji tingkat akurasi pengenalan karakater atau *Character Accuracy* (Cacc). *Character accuracy* nantinya akan digunakan untuk mengukur teks hasil pengenalan terhadap teks masukan yang sebenarnya. Untuk menghitung presentase banyaknya karakter yang dikenal dengan benar maka menggunakan rumus berikut : [9]

$$\text{acc} = \frac{\text{Jumlah karakter yang dikenal dengan tepat}}{\text{Total semua karakter masukan}} \text{ x } 100\% \quad \text{(pers 2.1)}$$

E. Tesseract OCR Engine

Tesseract merupakan mesin *optical character recognition* yang awalnya dikembangkan oleh Hewlet Packard di Bristol, Inggris dan Greelye, Colorado pada tahun 1985 hingga 1994. Banyak kode programnya yang ditulis dalam bahasa pemrograman C dan kemudian ditulis lagi ke dalam C++. Setelah itu dirilis sebagai proyek *open source* pada tahun 2005 oleh Hewlet Packard dan University of Nevada, Las Vegas (UNLV). Saat ini Tesseract dikembangkan dan disponsori oleh Google sebagai proyek *open source* sejak tahun 2006. Dengan demikian, Tesseract dapat dikembangkan dengan bebas oleh siapapun. Tesseract dianggap sebagai mesin OCR *open source* terbaik saat ini. [10]

Tesseract versi pertama hanya bisa mengenali teks bahasa Inggris. Dimulai dengan versi 2 Tesseract mampu memproses bahasa Inggris, Perancis, Italia, Jerman, Spanyol, Brasil Portugis dan Belanda. Dimulai dengan Tesseract versi 3 dapat mengenali Arab, Bulgaria, Catalan, Cina (Sederhana dan Tradisional), Kroasia, Ceko, Denmark, Belanda, Inggris, Jerman (standar dan *script* Fraktur), Yunani, Finlandia, Perancis, Ibrani, Hindi, Hungaria, Indonesia, Italia, Jepang, Korea, Latvia, Lithuania, Norwegia, Polandia, Portugis, Rumania, Rusia, Serbia, Slovakia (standar dan *script* Fraktur), Slovenia, Spanyol, Swedia, Tagalog, Tamil, Thailand, Turki, Ukraina dan Vietnam. [10]

F. Bahasa Pemrograman Java, Android, dan Android Studio

Terciptanya Java berawal ketika perusahaan Sun Microsystem ingin menciptakan sebuah bahasa pemrograman yang dapat dijalankan pada semua peralatan tanpa harus terikat pada platform yang digunakan oleh peralatan tersebut. Pada tahun 1991 saat itu dimulailah sebuah proyek dengan kode Green yang dipimpin oleh Patrick Naughton dan James Gosling. Pada awalnya, James Gosling menamakan bahasa pemrogramannya dengan "OAK". Akan tetapi karena nama OAK ini sudah dipergunakan oleh bahasa pemrograman yang telah ada sebelumnya maka mereka mengganti namanya menjadi "JAVA". [11]

Android merupakan sistem operasi yang gratis dan terbuka dari Google yang berjalan diberbagai perangkat seperti telepon, tablet bahkan televisi. Terdapat banyak perangkat yang dapat digunakan dengan satu platform. Aplikasi Android secara native dibangun menggunakan bahasa pemrogramman Java, namun pada perkembangannya kita dapat menggunakan bahasa pemrogramman berbasis Web dimana ini dimungkinkan dengan memanfaatkan Web View yang ada pada Android.[12]

Android Studio adalah sebuah perangkat lunak pengembang atau biasa disebut dengan IDE (Integrated Development Environment) untuk aplikasi Android. Android Studio merupakan pengembangkan dari Eclipse IDE, dan dibuat berdasarkan IDE Java populer, yaitu IntelliJ IDEA. Peluncuran Android Studio ini diumumkan oleh Google pada 16 Mei 2013 pada event Google I/O Conference untuk tahun 2013. Sejak saat itu, Android Studio menggantikan Eclipse sebagai IDE resmi untuk mengembangkan aplikasi Android. [13]

G. Model Pengembangan Perangkat Lunak Berorientasi Pemakaian Ulang (Re-Usable)

Rekayasa perangkat yang berbasis pemakaian ulang merupakan pendekatan terhadap pengembangan yang mencoba memaksimalisasi pemakaian ulang perangkat lunak yang ada. Unit perangkat lunak yang dipakai bisa berukuran sangat berbeda. Sebagai contoh : [14]

1. Pemakaian ulang sistem aplikasi

Seluruh sistem aplikasi dapat dipakai ulang dengan menggabungkannya tanpa perubahan dengan sistem lain atau dengan mengembangkan kerabat aplikasi yang dapat berjalan pada platform yang berbeda atau dapat dikhususkan untuk keperluan pengguna tertentu.

2. Pemakaian ulang komponen

Komponen dari suatu aplikasi yang ukurannya berkisar dari subsistem sampai satu objek tunggal dapat dipakai ulang. Sebagai contoh, sistem pengenal pola yang dikembangkan sebagai suatu bagian sistem pengolahan teks dapat dipakai ulang pada sistem manajemen database.

3. Pemakaian ulang fungsi

Komponen perangkat lunak yang mengimplementasi satu fungsi, seperti fungsi matematik, dapat dipakai ulang. Bentuk pemakaian ulang ini, berdasarkan pada *library* standar, umum dipakai selama 40 tahun terakhir ini.

Sementara tahap spesifikasi persyaratan dan tahap validasi dapat dibandingkan dengan proses lain, tahap pertengahan pada proses berorientasi pemakaian ulang ini ternyata berbeda. Tahap-tahap ini adalah : [14]

1. Analisis komponen

Jika diketahui spesifikasi persyaratan, komponen-komponen untuk implementasi spesifikasi tersebut akan dicari. Biasanya, tidak ada kesesuaian yang tepat dan komponen yang dapat dipakai hanya memberikan sebagian dari fungsionalitas yang dibutuhkan.

2. Modifikasi persyaratan

Pada tahap ini, persyaratan dianalisis dengan menggunakan informasi mengenai komponen yang telah didapat. Persyaratan kemudian dimodifikasi untuk merefleksikan komponen yang tersedia. Jika modifikasi tidak mungkin dilakukan maka kegiatan analisis komponen bisa diulang untuk mencari solusi alternatif.

3. Perancangan dengan pemakaian ulang

Pada fase ini, kerangka kerja sistem dirancang, atau kerangka kerja yang telah ada dipakai ulang. Perancang memperhitungkan komponen yang dipakai ulang dan mengatur kerangka kerja untuk menyesuaikan. Beberapa perangkat lunak

yang baru mungkin perlu dirancang jika komponen yang dapat dipakai ulang tidak tersedia.

4. Pengembangan dan integrasi

Perangkat lunak yang tidak dapat dibeli akan dikembangkan dan komponen kemudian diintegrasikan untuk membantu sistem. Integrasi sistem, pada model ini, bisa merupakan bagian dari proses pengembangan dan bukan merupakan kegiatan yang terpisah.

H. Metode Pengujian White Box

White box testing adalah pengujian yang didasarkan pada pemeriksaan terhadap detail perancangan, menggunakan struktur kontrol dari desain program secara prosedural untuk membagi pengujian ke dalam beberapa kasus pengujian. Manfaat dari dilakukan pengujian *white box* adalah untuk mengetahui cara kerja suatu perangkat lunak secara internal dan untuk menjamin operasi-operasi internal sesuai dengan spesifikasi yang telah ditetapkan dengan menggunakan struktur kontrol dari prosedur yang dirancang. Berikut ini adalah jenis-jenis dari pengujian *white box* : [15]

1. Basis *Path*

Uji coba basis path adalah teknik uji coba *white box* yg diusulkan Tom McCabe. Metode ini memungkinkan perancang *test case* mendapatkan ukuran kerumitan logika dari perancangan prosedural dan menggunakan ukuran ini sebagai petunjuk untuk mendefinisikan basis set dari jalur pengerjaan.

2. *Cyclomatic Complexity*

Metode ini merupakan pengukuran perangkat lunak yang memberikan pengukuran kuantitatif dari kompleksitas logika program. Pada metode basis *path*, nilai yang dihitung dan dalam *cyclomatic complexity* menentukan jumlah jalur-jalur yang independen dalam kumpulan basis suatu program dan memberikan jumlah tes minimal yang harus dilakukan untuk memastikan bahwa semua pernyataan telah dieksekusi. Rumus yang digunakan untuk mencari *cyclomatic complexity* :

$$V(G) = E - N + 2 \quad \text{(pers 2.2)}$$

Keterangan :

E = jumlah *edge* pada *flowgraph*

N = jumlah *node* (simpul) pada *flowgraph*

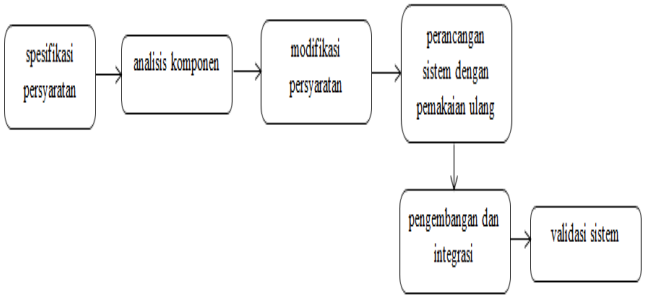
3. Graph Matrix

Metode ini berbentuk matriks persegi yang mempunyai ukuran (sejumlah baris dan kolom) yang sama dengan jumlah node pada flowgraph. Masing-masing baris dan kolom mempunyai hubungan dengan node yang telah ditentukan dan pemasukan data matriks berhubungan dengan hubungan (edge) antar node. Hubungan bobot menyediakan tambahan informasi tentang aliran kontrol. Secara sederhana hubungan bobot dapat diberi nilai 1 jika ada hubungan antara node atau nilai 0 jika tidak ada hubungan.

III. PERANCANGAN SISTEM

A. Tahap Pengembangan Sistem

Pengembangan perangkat lunak yang digunakan pada aplikasi ini menggunakan model pengembangan perangkat lunak yang berorientasi pemakaian ulang (*re-usable*) yang terdiri dari empat tahapan utama, yaitu analisis komponen, modifikasi persyaratan, perancangan dengan pemakaian ulang, serta pengembangan dan integrasi. Tahapan pengembangan perangkat lunak berorientasi pemakaian ulang ditunjukkan pada gambar 3.



Gambar 3. Tahapan pengembangan perangkat lunak dengan pemakaian ulang[14]

B. Spesifikasi Persyaratan

Pada tahapan spesifikasi persyaratan ini akan dijabarkan kebutuhan data serta koneksi, perangkat lunak pendukung terbuatnya aplikasi, perangkat keras untuk menjalankan aplikasi.

1. Perangkat Keras

Perangkat keras yang digunakan untuk merancang, dan mengembangkan aplikasi adalah notebook Asus A46CM dengan spesifikasi sebagai berikut :

a. Prosesor : Intel Core i3 3217U
b. OS : Microsoft Windows 8.1 (64-bit)
c. Memori : 4 GB SDRAM, 500 GB HDD
d. Grafis : Terintegrasi dengan Intel® HD Graphics 4000 dan NVIDIA® GeForce® GT 635M dengan 2GB DDR3 VRAM

Perangkat keras yang digunakan untuk menjalankan aplikasi adalah perangkat Android LG G2 Mini D618, berikut ini adalah spesifikasi dari perangkat tersebut :

a. CPU : Qualcomm MSM8226 Snapdragon 400 Quad-core 1,2 GHz Cortex A-7, GPU Adreno 305
b. OS : Android Lollipop 5.0.2
c. Ukuran layar : 4,7 inch
d. Tipe Layar : IPS LCD capacitive touchscreen
e. Memori internal : 8 GB, 1 GB RAM
f. Memori eksternal : 16 GB MicroSD-Card
g. Kamera belakang : 8 MP, 3264 x 2448 piksel, LED flash, geo-tagging, face detection, panorama, HDR
h. Kamera depan : 1,3 MP, 720 piksel
i. Rekaman video : 1080 piksel

2. Perangkat Lunak

Perangkat lunak pendukung yang digunakan dalam pembuatan dan pengembangan aplikasi penerjemah bahasa dengan OCR berbasis Android, yaitu :

a. Android Studio versi 1.2
b. Android SDK Tools 24.3.4
c. Android NDK R10e
d. JDK 1.7.0

3. Kebutuhan Data dan Koneksi

Pada saat menjalankan aplikasi ini, pengguna harus menghubungkan perangkat Android ke internet melalui paket data ataupun Wi-Fi. Data yang diambil untuk diuji coba pada aplikasi adalah berasal dari tulisan-tulisan bahasa Inggris yang dicetak ke dalam kertas HVS A4 80 gr. Tulisan ini nantinya akan diuji berdasarkan jumlah banyaknya kata yang terdeteksi dan dapat diterjemahkan serta bergantung intensitas cahaya dalam ruangan, yaitu dalam keadaan redup atau terang. Berikut adalah gambar contoh karakter yang akan dideteksi dan dianalisis dalam tahap pengujian pada gambar 4.

This Why Girls Wonder Taste Paradise Problems Like Finding More Occupation	Black Magic Crystall Ball Bad Blood Wildest Place Make Up Walking Straight	How Are You ? Falling In Love Good For You Make It Happen Into The Woods Waiting for Miracles
I Will Go To My Parent's House Tomorrow I'am Staring At The Sunset In North Spain Students Are Experimenting In The Laboratorium		

Gambar 4. Contoh karakter untuk pendeteksian pada aplikasi

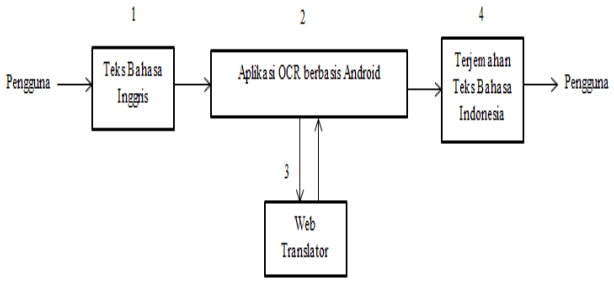
C. *Analisis Komponen*

Komponen dalam hal ini maksudnya adalah komponen yang dapat dipakai ulang dan sesuai, telah memastikan bahwa komponen akan bekerja, dan memiliki dokumentasi yang berhubungan agar dapat memahami dan mengadaptasi ke suatu pembuatan aplikasi yang baru. Ketika sistem membutuhkan layanan, sistem memanggil komponen untuk menyediakan layanan tersebut tanpa peduli di mana komponen tersebut berjalan atau bahasa pemrograman yang dipakai untuk mengembangkannya.

Komponen yang mendukung dalam pembuatan aplikasi penerjemah bahasa dengan *Optical Character Recognition* (OCR) berbasis Android adalah Tesseract v3.03 OCR *engine* yang terhubung dengan *library* Web Translator sebagai penerjemahnya. Tesseract sendiri adalah mesin pengenal karakter optik yang disebarluaskan secara gratis. Selain Tesseract v3.03 *engine* yang digunakan untuk membuat aplikasi ini, terdapat pula *library* yang dibutuhkan untuk mendukung jalannya aplikasi, yaitu Google Translator dan Microsoft Translator. Kedua *library* tersebut berfungsi sebagai penghubung dengan layanan penerjemah dari Google Cloud Platform dan Microsoft Azure Marketplace.

D. *Modifikasi Persyaratan*

Terdapat beberapa modifikasi yang ditemukan dan dilakukan oleh peneliti lain contohnya, yaitu memanfaatkan teknologi OCR ini berbasis *mobile phone* sebagai penerjemah bahasa Sunda – Indonesia – Inggris, Arab – Indonesia maka dalam hal ini penulis akan membuat aplikasi berbasis Android sebagai penerjemah bahasa Inggris – Indonesia. Rancangan diagram blok sistem yang dibuat ditunjukkan pada gambar 5.



Gambar 5. Diagram blok sistem yang dibuat

Keterangan dari diagram blok yang ditampilkan pada gambar 5 adalah sebagai berikut :

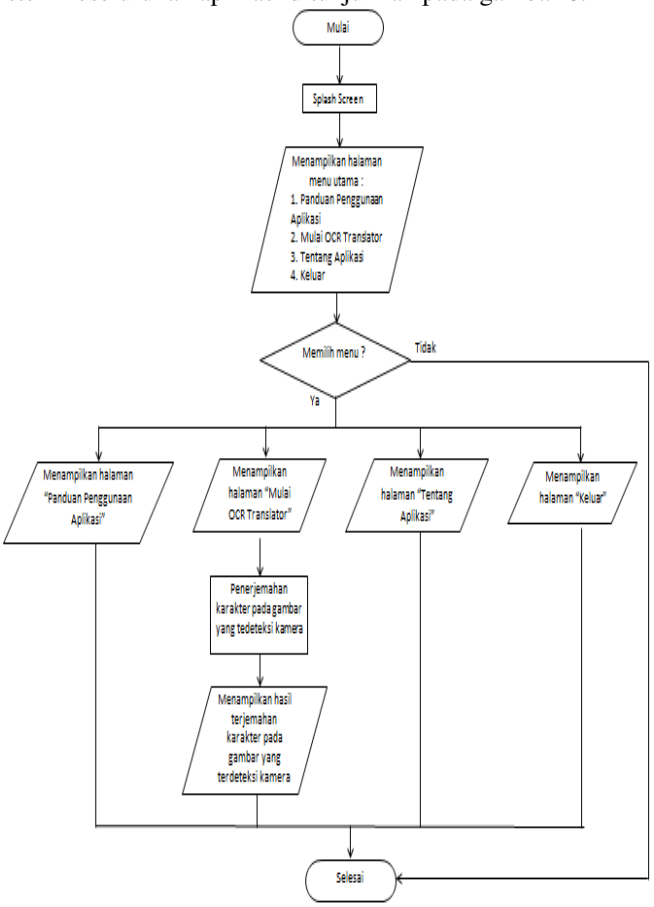
1. Pengguna aplikasi melalukan penangkapan gambar yang berupa teks bahasa Inggris melalui kamera pada *smartphone* Android, kemudian teks tersebut akan menjadi masukkan ke dalam aplikasi OCR berbasis Android.
2. Gambar teks bahasa Inggris yang ditangkap akan dikenali oleh sistem yang selanjutnya akan diterjemahkan . Teks yang terdeteksi akan ditampilkan pada layar *smartphone*, selanjutnya diterjemahkan ke dalam bahasa Indonesia.
3. Aplikasi OCR berbasis Android nantinya terdapat *library* Google Translator atau Microsoft Translator yang digunakan sebagai penghubung layanan penerjemah bahasa Inggris – Indonesia.
4. Keluaran dari sistem adalah hasil terjemahan kata bahasa Inggris ke teks bahasa Indonesia akan ditampilkan pada layar *smartphone* Android.

E. *Perancangan dengan Pemakaian Ulang*

Adanya komponen yang dapat dipakai ulang dan sesuai, telah memastikan bahwa komponen akan bekerja, dan memiliki dokumentasi yang berhubungan ini sangat penting dalam tahapan perancangan aplikasi. Tahapan perancangan dengan pemakaian ulang ini, penulisakan merancang aliran proses yang terjadi pada sistem yang akan dibuat, yaitu dalam aplikasi penerjemah bahasa dengan OCR berbasis Android menggunakan komponen yang bersifat *reusable* (dapat dipakai ulang), yaitu Tesseract *engine*. Kemudian penulismembuat dan menganalisis rancangan diagram alir kerja sistem yang sesuai dengan fungsi dan keterlibatan entitas yang terkait dalam penggunaan aplikasi nantinya.

1. Desain Diagram Alir Sistem

Diagram alir atau *flowchart* menunjukkan aliran kerja sistem secara keseluruhan. Diagram alir ini menggambarkan urutan dari prosedur-prosedur yang terdapat dalam sistem serta memperlihatkan apa yang dikerjakan oleh sistem. Diagram alir sistem keseluruhan aplikasi ditunjukkan pada gambar 6.



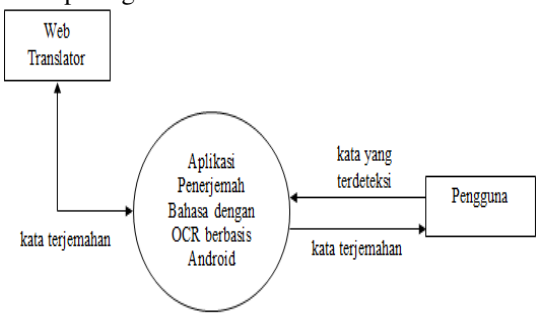
Gambar 6. Diagram alir sistem keseluruhan aplikasi

2. Diagram Konteks Sistem

Diagram konteks merupakan sebuah diagram sederhana yang menggambarkan hubungan antara entitas, masukan, dan

keluaran dari sistem. Pada aplikasi yang dibuat ini yang menjadi entitas adalah pengguna dan web translator karena keduanya mempunyai peran dalam penggunaan pada sistem. Pengguna akan memasukkan suatu objek yang berupa kata yang terdeteksi melalui kamera *handphone* ke dalam sistem.

Setelah itu, sistem akan memproses objek kata dalam bahasa Inggris tersebut untuk diterjemahkan dengan dukungan web translator, yaitu penerjemah yang terhubung melalui internet dengan dukungan *library* Microsoft dan Google Translator. Keluaran dari sistem akan menghasilkan objek kata yang sudah diterjemahkan ke dalam bahasa Indonesia. Hasil keluaran tersebut akan langsung terbaca oleh pengguna pada layar *handphone*. Diagram konteks sistem yang akan dibuat ditunjukkan pada gambar 7.

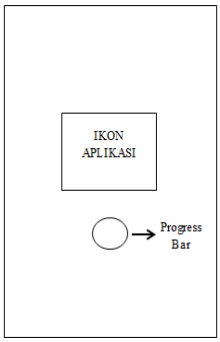


Gambar 7. Diagram Konteks Sistem

3. Desain Antarmuka Sistem

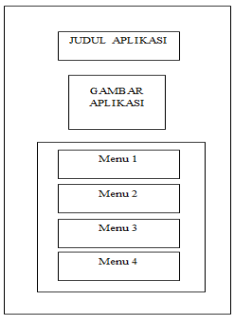
Selanjutnya adalah merancang antarmuka perangkat lunak yang bertujuan untuk memberikan gambaran secara umum dari aplikasi yang dibuat. Desain antarmuka ini berperan penting karena akan berpengaruh terhadap penggunaan aplikasi oleh pengguna nantinya yang harus dilihat dari segi kenyamanan dan kemudahan. Kesalahan ataupun adanya kerumitan dalam rancangan antarmuka aplikasi akan mengakibatkan kesulitan pada pengguna aplikasi, misal jika warna latar dengan warna huruf sama maka pengguna akan kesulitan dalam membaca suatu tulisan dalam aplikasi tersebut.

Pada gambar 7 merupakan gambaran desain dari pembuka halawan awal aplikasi yang terdiri dari komponen *textfield*, logo aplikasi, dan *progress bar*. Desain tampilan *splash screen* sebagai pembuka aplikasi ditunjukkan pada gambar 8.



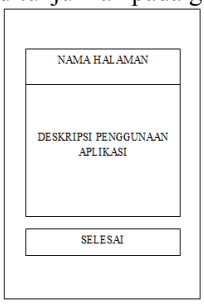
Gambar 8. Desain tampilan *splash screen* sebagai pembuka aplikasi

Setelah halaman *splash screen* berjalan maka akan muncul halaman menu utama yang terdiri dari empat pilihan untuk menuju ke fungsi masing-masing. Menu 1 adalah pilihan untuk masuk ke halaman "Mulai OCR" yang akan menjalankan aplikasi inti untuk menerjemahkan karakter tulisan bahasa Inggris ke dalam bahasa Indonesia. Menu 2 adalah pilihan untuk masuk ke halaman "Panduan Penggunaan" yang mana terdapat deskripsi tata cara penggunaan aplikasi secara menyeluruh. Menu 3 merupakan pilihan ke halaman "Tentang" yang berisi deskripsi pembuatan aplikasi. Menu 4 merupakan pilihan "Keluar" yaitu untuk mengakhiri proses aplikasi ini. Desain tampilan halaman menu utama aplikasi ditunjukkan pada gambar 9.



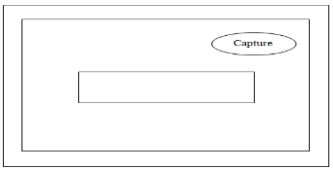
Gambar 9. Desain tampilan halaman menu utama aplikasi

Menu 1 berfungsi sebagai "Panduan Pengguna Aplikasi" yang mana akan menyajikan tentang penggunaan aplikasi secara menyeluruh kepada pengguna. Komponen yang digunakan dalam halam panduan pengguna yaitu *textfield* dan tombol. Komponen *textfield* akan diisi dengan nama halaman dan deskripsi penggunaan aplikasi. Kompoen tombol yang diberi nama "Selesai" dalam halaman ini berfungsi untuk berpindah halaman, yaitu dari halaman panduan pengguna ke halaman menu utama aplikasi. Desain tampilan halaman "Panduan Pengguna" ditunjukkan pada gambar 10.



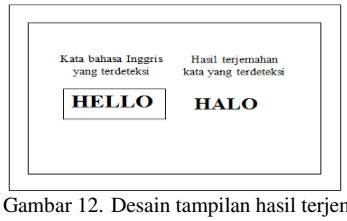
Gambar 10. Desain tampilan halaman "Panduan Pengguna"

Selanjutnya jika pengguna memilih Menu 2 yaitu "Mulai OCR Translator" maka akan masuk ke dalaman halaman inti aplikasi. Awal tampilan halaman inti ini akan memanfaatkan kamera *handphone*. Komponen yang terdapat dalam halaman ini yaitu terdapat tombol *capture.* Fungsi tombol *capture* ini dimaksudkan untuk menangkap objek yang terdeteksi pada kamera yang berupa suatu kata dalam bahasa Inggris. Setelah objek kata terdeteksi, tombol ini akan menjalankan fungsi penerjemahan kata tersebut ke dalam bahasa Indonesia. Desain tampilan aplikasi dengan tombol *capture* ditunjukkan pada gambar 11.



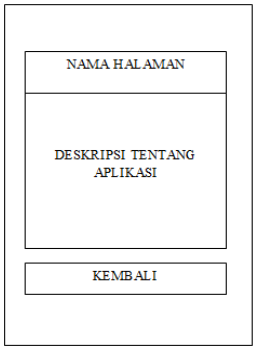
Gambar 11. *Layout* utama saat aplikasi dijalankan

Kemudian hasil tangkapan kamera pengguna akan diterjemahkan ke dalam bahasa Indonesia secara langsung pada layar *handphone.* Letak hasil terjemahan kata yang tertangkap kamera berada persis di samping kotak terdeteksinya karakter bahasa Inggris. Pengguna dapat mendeteksi kembali teks lainnya setelah hasil terjemahan ditampilkan. Desain tampilan hasil terjemahan ditunjukkan pada gambar 12.



Gambar 12. Desain tampilan hasil terjemahan

Menu 3 yang diberi nama "Tentang Aplikasi" akan memberikan informasi siapa dan aplikasi ini dikembangkan. Pada halaman tentang ini terdapat komponen *textfield* dan tombol. Komponen *textfield* berisi nama dari halaman dan deskripsi lengkap tentang aplikasi dan tombol diberikan nama "Kembali" untuk berpindah ke halaman menu utama. Desain tampilan halaman "Tentang" ditunjukkan pada gambar 13.



Gambar 13. Desain tampilan halaman "Tentang Aplikasi"

F. *Pengembangan dan Integrasi*

Pengembangan yang dilakukan pada aplikasi ini adalah menambahkan beberapa pilihan menu yang dapat digunakan oleh pengguna. Pilihan beberapa menu tersebut telah diuraikan pada bab 3 dalam makalah ini (desain antarmuka sistem). Setelah sistem selesai dirancang dan dibuat maka akan dilakukan pengujian terhadap sistem.

Keterlibatan pengguna dalam pengujian merupakan bagian penting yang memberikan pengalaman baru terhadap pengguna. Selanjutnya, sistem yang telah bekerja nantinya akan diimplementasi sesuai dengan dukungan spesifikasi persyaratan yang telah dijabarkan pada bab 3 dalam makalah ini menggunakan perangkat Android. Selanjutnya sistem akan dilakukan pengujian secara *white box*, yang artinya sistem akan diuji pada sistem kontrol dari rancangan sistem secara prosedural untuk membagi pengujian ke dalam beberapa kasus pengujian (*test-case*).

Metode pengujian pada aplikasi ini menggunakan *white box* karena berfungsi untuk memeriksa detail dari perancangan sistem dan untuk mengetahui bagaimana sistem menghasilkan keluaran yang berasal dari masukan sistem, Nantinya, sistem juga akan diuji tingkat akurasi hasil terjemahan dalam pendeteksian karakter yang dilakukan dengan kamera pada perangkat Android.

IV. IMPLEMENTASI DAN PENGUJIAN

A. *Tahap Pembuatan Sistem*

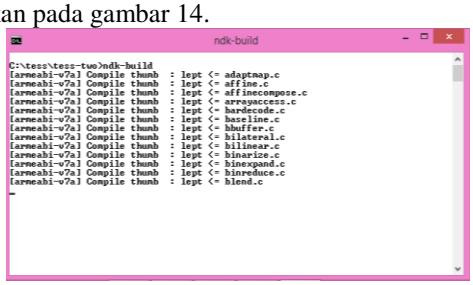
Pembuatan sistem ini dilakukan berdasarkan analisis komponen yang dibutuhkan dalam perancangan berorientasi pemakaian ulang, dimana komponen yang dapat dipakai ulang untuk dijalankan pada proses pembuatan aplikasi. Komponen tersebut yaitu Tesseract OCR versi 3.03 yang merupakan proyek *open source* dan dapat diunduh dari situs https://github.com/rmtheis/tess-two. Oleh karena itu, tahap pertama yang dilakukan adalah penerapan komponen Tesseract OCR engine ke dalam lingkungan pengembang Android Studio. Tahap berikutnya adalah membuat antarmuka sistem untuk memudahkan penggunaan sistem nantinya.

Langkah pertama yang dilakukan pada tahap penerapan Tesseract OCR engine adalah mengunduh Tesseract engine itu sendiri kemudian letakkan di dalam direktori C:\ untuk memudahkan proses ekstraksi dan building engine. Selanjutnya jika belum memiliki Android NDK maka harus diunduh terlebih dahulu dari situs https://developer.android.com/ndk/downloads/index.html.

Setelah proses mengunduh NDK selesai maka NDK harus diinstalasi ke dalam komputer, tunggu hingga prosesnya selesai dan NDK siap untuk digunakan. Sintaks yang digunakan untuk mengkompilasi engine dengan Android NDK ini adalah sebagai berikut :

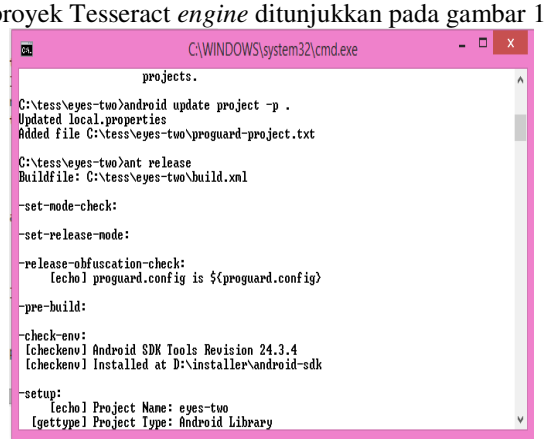
```
ndk-build
android update project --path
ant release
```

Proses *building* folder tess-two dengan Android NDK ditunjukkan pada gambar 14.



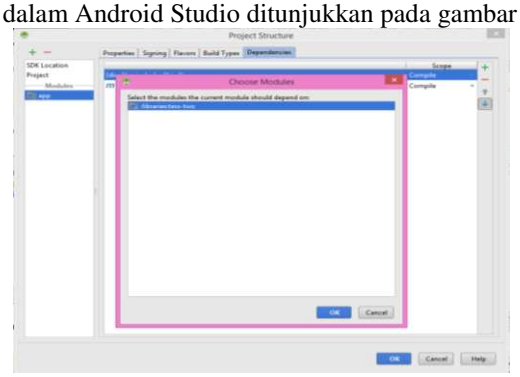
Gambar 14. Proses *building* folder tess-two dengan Android NDK

Setelah proses *building* folder tess-two selesai maka selanjutnya folder eyes-two akan dilakukan proses *building* juga dengan sintaks ndk-build. Kemudian melakukan perbaharuan pada proyek *engine* dengan sintaks android update project –p dan *ant release*. Hasil proses *update* dan release pada proyek Tesseract *engine* ditunjukkan pada gambar 15.



Gambar 15. Hasil proses *update* dan *ant release* pada proyek Tesseract *engine*

Langkah berikutnya adalah membuat proyek baru dalam Android Studio yang diberi nama Ocr_translator dengan *blank activity* dan mengatur target *running* aplikasi minimal pada Android Jelly Bean (API 16). Kemudian masukkan Tesseract *engine* yang telah melalui proses *building* ke dalam proyek Android Studio sebagai *library* dengan cara memilih menu *File* → *Project Structure* → pilih tab *Depedencies* → klik tanda + bewarna hijau pada sebelah kanan tab dan pilih tess-two sebagai *library* → OK. Memasukkan Tesseract *engine* sebagai *library* dalam Android Studio ditunjukkan pada gambar 16.

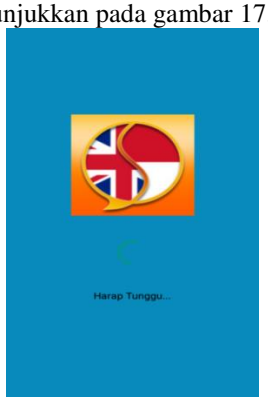


Gambar 16. Memasukkan Tesseract *engine* sebagai *library* dalam Android Studio

Selanjutnya menambahkan *library* pendukung aplikasi yang digunakan untuk menerjemahkan dengan menggunakan layanan Google Translator API dan Microsoft Translator API ke dalam Android Studio. Cara memasukkan *file* .jar ke dalam direktori aplikasi Android Studio adalah memasukkannya dari Windows Explorer dan letakkan pada direktori folder app → libs.

Tahap selanjutnya adalah mengimplementasikan rancangan antarmuka pada bab sebelumnya menjadi halaman-halaman pada aplikasi. Membuat *layout* antarmuka untuk aplikasi nantinya berguna untuk memudahkan pengguna dalam menjalankan aplikasi dalam perangkat. Terdapat tiga *layout* utama yang akan ditampilkan pada aplikasi ini. *Layout* pertama adalah *splash screen* yang berfungsi sebagai pembuka awal aplikasi sebelum ke halaman menu utama. *Layout* kedua adalah halaman menu utama. *Layout* ketiga adalah halaman inti yang menjalankan fungsi OCR untuk menerjemahkan bahasa.

Halaman pertama yang dibuat adalah halaman untuk pembuka sebelum masuk ke halaman menu utama, yaitu halaman *Splash Screen*. Halaman ini hanya muncul selama tiga detik sebagai pembuka yang kemudian langsung masuk ke dalam halaman menu utama. Pada halaman ini terdapat logo aplikasi serta *progress bar* tepat di bawahnya. Halaman *Splash Screen* aplikasi ditunjukkan pada gambar 17.

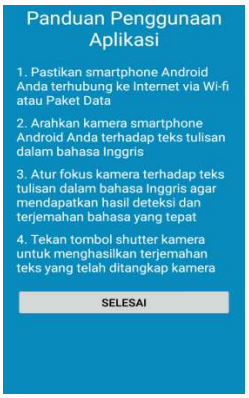


Gambar 17. Halaman *Splash Screen* aplikasi

Apabila halaman *Splash Screen* telah muncul selama tiga detik maka akan muncul halaman menu utama seperti pada gambar 18 Halaman menu utama terdapat empat tombol pilihan yaitu, "Panduan Pengguna", "Mulai OCR Tranlator", "Tentang", dan "Keluar". Halaman yang perlu diperhatikan pertama kali adalah "Panduan Pengguna" untuk mengetahui penggunaan dari aplikasi ini.

Gambar 18. Halaman menu utama aplikasi

Jika pada halaman menu utama pengguna memilih "Panduan Penggunaan Aplikasi" maka akan masuk ke halaman panduan atau tata cara penggunaan aplikasi. Pada halaman tersebut dideskripsikan tata cara penggunaan aplikasi secara rinci. Halaman panduan pengguna ditunjukkan pada gambar 19.

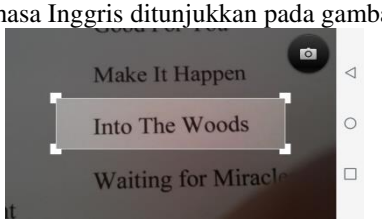


Gambar 19. Halaman panduan pengguna

Pengguna dapat melihat ke dalam halaman "Tentang Aplikasi" dengan memilih pilihan yang ada pada halaman menu utama "Tentang Aplikasi". Halaman ini berisi deskripsi informasi mengenai pembuat aplikasi dan penjelasan aplikasi seperti pada gambar 20.

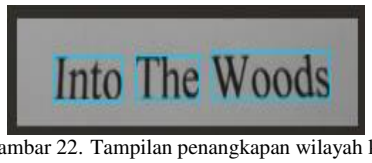
Gambar 20. Halaman tentang informasi aplikasi

Selanjutnya pengguna dapat menerjemahkan kata dalam bahasa Inggris ke bahasa Indonesia maka dari halaman menu utama dipilih "Mulai OCR Translator". Kemudian akan tampil halaman yang berfungsi sebagai penerjemah menggunakan kamera pada perangkat Android. Pada tampilan halaman OCR ini terdapat kotak yang disediakan untuk mendeteksi karakter kata dalam bahasa Inggris. Tampilan halaman penangkapan karakter dalam bahasa Inggris ditunjukkan pada gambar 21.



Gambar 21. Tampilan halaman penangkapan karakter dalam bahasa Inggris

Proses pengenalan karakter yang dilakukan oleh *engine* pada aplikasi ini yaitu pencarian garis dan kata seperti yang terlihat pada gambar 22 Kemudian *engine* melakukan proses pengenalan masing-masing kata pada gilirannya. Kata-kata yang sukses pada proses pengenalan pertama, yaitu kata-kata yang terdapat dalam kamus dan tidak ambigu kemudian diteruskan ke *adaptive classifier* sebagai data pelatihan. Data pelatihan disisipkan pada Android Studio setelah diunduh bersama *engine* dalam folder *assets* dan *file* tersebut diberi nama eng.traineddata. Setelah *adaptive classifier* memiliki sampel yang cukup, *adaptive classifier* ini dapat memberikan hasil klasifikasi bahkan pada proses pengenalan pertama seperti gambar 23.



Gambar 22. Tampilan penangkapan wilayah karakter

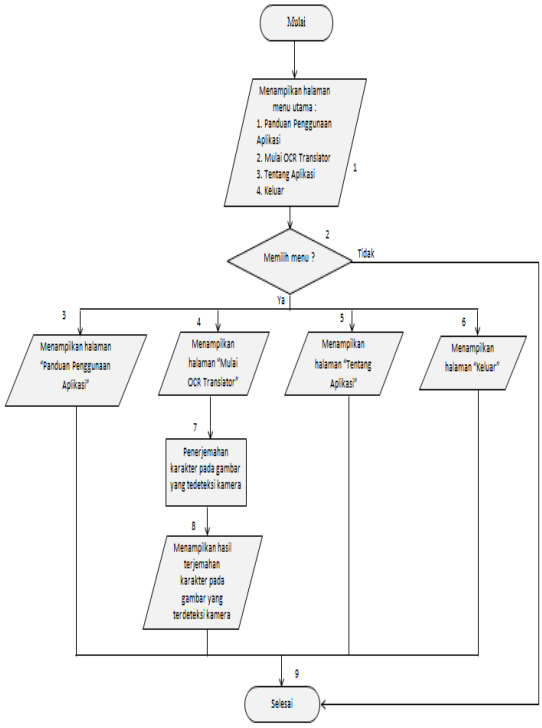
Pada proses pengenalan kedua dilakukan untuk mengenali kata-kata yang mungkin saja kurang dikenali atau terlewatkan pada proses pengenalan pertama, pada tahap ini *adaptive classifier* telah memperoleh informasi lebih dari proses pengenalan pertama. Setelah pengguna menangkap karakter dalam bahasa Inggris sesuai yang diinginkan maka hasil terjemahan dalam bahasa Indonesia akan langsung tampil pada layar perangkat Android.

Gambar 23. Hasil terjemahan bahasa Inggris – Indonesia pada aplikasi

B. Tahap Pengujian

Penelitian tugas akhir ini menggunakan metode pengujian *white box* dimana fungsi utama yang ada dalam sistem diuji secara rinci berdasarkan jalur-jalur logika yang ada. Metode pengujiannya menggunakan teknik struktur kontrol program untuk memperoleh kasus uji. Pengujian *white box* pada aplikasi ini akan memetakan *flowchart* dan *flowgraph* berdasarkan alur *source code* aplikasi untuk mempermudah menentukan *cyclomatic complexity* dan *independent path*. Jika jumlah *cyclomatic complexity* dan jumlah independent path sama maka sistem OCR dinyatakan benar, namun bila sebaliknya maka sistem masih mempunyai kesalahan dari segi logika maupun dari sisi lainnya.

Kasus pengujian pertama adalah saat aplikasi dijalankan pertama kali yang ditampilkan pada aplikasi adalah halaman *splash screen* selama tiga detik yang kemudian diikuti dengan munculnya halaman utama yang berisi menu aplikasi. Terdapat empat menu yang akan diuji. Fungsi "onClick" pada menu yang tersedia di halaman utama dibentuk menjadi sebuah *flowchart* seperti pada gambar 24.



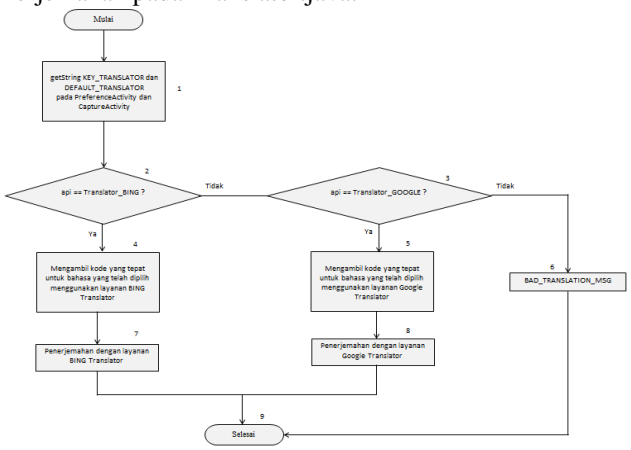
Gambar 24. *Flowchart* dari proses halaman menu utama

Cyclomatic complexity atau yang disimbolkan dengan V(G) untuk *flowgraph* dihitung dengan persamaan 2.2 dan didapatkan hasil sebagai berikut : V(G)=11–9+2=4. Nilai *cyclomatic complexity* dengan jumlah jalur pada tabel 1 sesuai dengan hasil yang didapat menggunakan persamaan 2.2 maka untuk fungsi proses menampilkan dan pemilihan menu yang diklik sudah sesuai atau benar dan hasil jalur seperti pada Tabel 1.

Tabel 1. Jalur *flowgraph* dari halaman menu utama

Jalur	Tahapan	Keterangan
1	1-2-3-9	Tampilan Panduan Penggunaan Aplikasi
2	1-2-3-4-7-8-9	Tampilan OCR Translator
3	1-2-3-4-5-9	Tampilan Tentang Pembuat Aplikasi
4	1-2-3-4-5-6-9	Keluar dari Aplikasi

Kasus pengujian berikutnya adalah ketika pengguna ingin menerjemahkan kata yang telah terdeteksi pada kamera kemudian menghasilkan gambar. Prosesnya adalah awalnya sistem akan memproses gambar yang ditangkap oleh kamera apakah gambar tersebut mengandung huruf-huruf yang membentuk suatu karakter kata. Selanjutnya memeriksa *key translator* dan *default translator* yang terdapat dalam sistem kemudian *library* penerjemah dipanggil untuk menerjemahkan gambar tersebut. Terdapat dua *library* untuk menerjemahkan yang dimasukkan ke dalam sistem, yaitu Google dan Microsoft Translator. Pada gambar 25 ditunjukkan *flowchart* dari proses penerjemahan pada Translator.java.



Gambar 25. *Flowchart* pada proses penerjemahan bahasa menggunakan *library* API Google dan Microsoft Translator

Cyclomatic complexity atau yang disimbolkan dengan V(G) untuk *flowgraph* dihitung dengan persamaan 2.2 dan didapatkan hasil sebagai berikut : V(G) = 10-9+2= 3. Nilai *cyclomatic complexity* dengan jumlah jalur pada tabel 2 sesuai dengan hasil yang didapat menggunakan persamaan 2.2 maka untuk fungsi proses menampilkan dan pemilihan menu yang diklik sudah sesuai atau benar dan hasil jalur seperti pada Tabel 2.

Tabel 2. Jalur *flowgraph* dari proses penerjemahan bahasa menggunakan *library* API Google dan Microsoft Translator

Jalur	Tahapan	Keterangan
1	1-2-4-7-9	Proses penerjemahan dengan Bing Translator
2	1-2-3-5-8-9	Proses penerjemahan dengan Google Translator
3	1-2-3-6-9	Proses penerjemahan tidak berhasil

Selanjutnya adalah pengujian pada hasil terjemahan sistem yang telah diimplementasi ke dalam perangkat Android versi 5.0.2 Lollipop. Pada tabel 3 akan ditunjukkan beberapa hasil terjemahan yang telah diujicoba berdasarkan jumlah kata-kata yang terdeteksi oleh *engine* OCR. Pengujian sistem dilakukan oleh pengguna dengan menyediakan kata-kata dalam bahasa

Inggris yang terdiri dari satu kata yang kemudian diujicoba dari satu kalimat.

Terdapat tiga kriteria hasil yang diberikan pada setiap sampel, yaitu sesuai, kurang sesuai, dan tidak sesuai. Kriteria sesuai adalah untuk hasil pengujian dengan pengenalan teks dan terjemahan yang tepat. Kriteria kurang sesuai diberikan jika pengenalan teks masih terdapat kesalahan tetapi hasil terjemahannya benar atau pengenalan teksnya dilakukan dengan baik tetapi hasil terjemahannya tidak tepat. Kemudian, kriteria tidak sesuai diberikan pada sampel pengujian yang pengenalan teksnya tidak tepat, begitupun dengan hasil terjemahannya.

Tabel 3. Pengujian hasil terjemahan pada aplikasi

No	Hasil Kata yang Terdeteksi Sistem	Hasil Terjemahan Sistem	Sesuai, Kurang, atau Tidak Sesuai
1	*Taste*	Rasa	Sesuai
2	*This*	Ini	Sesuai
3	*Girls*	Gadis	Sesuai
4	*More*	Lebih	Sesuai
5	*Problems*	Masalah	Sesuai
6	*Black Magic*	Sihir Hitam	Sesuai
7	*Bad Blood*	Darah Buruk	Sesuai
8	*Crystall Ball*	Zebra	Kurang Sesuai
9	*Walking Straight*	Berjalan tegak	Sesuai
10	*Wildest Place*	Tempat Paling alami	Sesuai
11	*Full Of Honey*	Penuh dengan madu	Sesuai
12	*How Are You ?*	Bagaimana keadaanmu?	Sesuai
13	*Belong To Me*	Milik saya	Sesuai
14	*Good for you*	Baik untuk Anda	Sesuai
15	*Falling In Love*	Jatuh cinta	Sesuai
16	*, I Will Go To My Parent; House Tomorrow*	, Aku akan pergi ke orangtua saya; Rumah besok	Kurang Sesuai
17	*Iam Staring At The Sunset In North Spain*	Iam menatap matahari terbenam di Spanyol Utara	Kurang Sesuai
18	*There Are Kids Playing In The Field*	Ada anak-anak bermain di Lapangan	Sesuai
19	*Pve Got The Recipe Of Lasagna*	Pve mendapat resep lasagna	Kurang Sesuai
20	*Trust Me, It Works*	Percayalah, bekerja	Kurang Sesuai
21	*Iam Staring At The Sunset In North Spain* *There Are Kids Playing In The Field*	Iam menatap matahari terbenam di Spanyol Utara ada adalah anak-anak bermain di Lapangan	Kurang Sesuai

Hasil keluaran yang terdiri dari kata yang terdeteksi yang memanfaatkan *engine* Tesseract menjadi kurang tepat atau sesuai jika teks masukan tidak dimaksimalkan memenuhi kotak yang disediakan untuk mengambil gambar. Jadi, gambar atau teks masukan yang ingin diterjemahkan lebih baik harus memenuhi kotak deteksi karakter. Selain itu, cahaya pada ruangan juga akan memengaruhi hasil keluaran, jika pencahayaannya kurang atau gelap maka tidak akan ada teks yang terdeteksi atau kemungkinannya terlalu kecil.

Setiap karakter terbalik atau miring harus dibetulkan terlebih dahulu atau tidak akan ada teks yang terdeteksi. Selain itu, adanya garis batas (*border*) berwarna gelap pada karakter seharusnya dihilangkan atau dihapus, atau akan disalahartikan sebagai karakter juga. Berdasarkan pada tabel 3 akan diuji tingkat akurasi pengenalan dari sampel karakter yang telah diambil, yaitu terdapat :

Jumlah sampel = 21 sampel

Jumlah seluruh kata = 80 kata

Jumlah kata yang dapat dikenali dengan baik = 78 kata dari jumlah seluruh kata

Selanjutnya, untuk mencari tingkat akurasi dari pengenalan karakter (Cacc) yang dilakukan sistem maka digunakan rumus pada persamaan 2.1 dan mendapatkan hasil sebagai berikut :

$$R = \frac{78}{80} = 0{,}975 \text{ x } 100\ \% = 97{,}5\ \%$$

Jadi, besarnya tingkat pengenalan yang dilakukan aplikasi adalah sebesar 97,5 % berdasarkan perhitungan yang telah dilakukan. Sistem mengalami kegagalan dalam mengenal karakter sebesar 2,5% dari keseluruhan sampel yang diambil, yaitu terdapat dua (2) kata yang salah dikenali oleh sistem.

V. KESIMPULAN DAN SARAN

Pada bagian ini akan dijelaskan kesimpulan dan saran dari hasil penelitian dan pembahasan.

5.1. KESIMPULAN

Kesimpulan dari penelitian tugas akhir ini adalah sebagai berikut :

1. OCR (*Optical Character Recognition*) merupakan suatu teknik pengenalan karakter teks baik itu dari tulisan tangan atau berasal dari tulisan yang telah dicetak.
2. Penggunaan metode pengembangan perangkat lunak berorientasi pemakaian ulang cocok untuk pengembangan yang membutuhkan komponen yang bersifat dari komponen yang bersifat dapat dipakai berulang-ulang dan bebas disebarluaskan untuk kepentingan pengembangan.
3. Aplikasi penerjemah bahasa Inggris –Indonesia dengan OCR telah disisipkan data pelatihan karakter yang berguna untuk mengenali karakter dalam bahasa Inggirs agar dapat diterjemahkan ke bahasa Indonesia secara *online* menggunakan Microsoft Translator atau Google Translator.
4. Faktor yang menyebabkan kurang tepatnya penerjemahan adalah kurang cukupnya cahaya, tidak sesuainya karakter dengan kotak yang disediakan sebagai pendeteksi, terlalu kecil dan terbaliknya huruf, tulisan miring, dan kurang fokusnya suatu gambar karakter masukan. Selain itu, proses terjemahan tidak akan berlangsung jika koneksi internet pada perangkat Android tidak stabil atau tidak ada sama sekali.
5. Hasil pengujian sistem dengan *white box* menunjukkan sistem tidak memiliki kesalahan dalam menjalankan fungsi-fungsi yang telah dirancang sebelumnya pada tahapan perancangan.
6. Hasil perhitungan akurasi pengenalan karakter yang dilakukan aplikasi menunujukkan 97,5% OCR dapat mengenali 78 dari 80 karakter dengan baik atau sesuai, serta 2,5% OCR masih mengalami kegagalan dalam mengenali 2 (dua) karakter yang diambil dari 21 sampel pengujian.

5.2. SARAN

Terdapat beberapa saran dari hasil penelitian yaitu sebagai berikut :

1. Aplikasi ini dapat dikembangkan lebih lanjut dengan menambahkan fitur pembalik terjemahan, misal dalam aplikasi ini hanya bahasa Inggris – Indonesia, sebaiknya dapat dibuat juga kebalikan terjemahannya, yaitu dari

Indonesia ke dalam bahasa Inggris atau juga menambahkan bahasa lainnya.

2. Memperbaiki dan memperindah tampilan antarmuka untuk menyajikan kenyamanan penggunaan aplikasi oleh pengguna.

DAFTAR PUSTAKA

[1] Algeo, John, *The Origins and Development of the English Language: Sixth Edition*, Wadsworth, Cengage Learning, 2005.

[2] Luger, George F., *Artificial Intelligence (Structure and Strategies for Complex Problem Solving): Fourth Edition*, Pearson Education Ltd, 2002.

[3] E. Rich, K. Knight, and S. B. Nair, *Artificial Intelligence Third Edition*, The McGraw-Hill Companies, 2009.

[4] M. Dahria, "Kecerdasan Buatan (Artificial Intelligence)," *Jurnal SAINTIKOM,* vol. 5, no. 2, pp. 185-197, 2008.

[5] Arhami, M., dan Desiani A, *Konsep Kecerdasan Buatan*, Yogyakarta : ANDI OFFSET, 2006.

[6] Arhami, M., *Konsep Dasar Sistem Pakar*, Penerbit Andi, Yogyakarta, 2005.

[7] Putra, Darma, *Pengolahan Citra Digital*. Yogyakarta: Andi, 2009.

[8] Mollah, Ayatullah Faruk., et al, *Design of an Optical Character Recognition System for Camera based Handheld Device*, Internatioanl Journal of Computer Science Issues, Vol-8 No-1, Jadavpur University, 2011.

[9] Sulistiyo, Mahmud Dwi, *Aplikasi Optical Character Recognition pada Perangkat Mobile Menggunakan Mixed Binarization*, DISC (*Digital Information and System Conference* 2014), Jurnal Fakultas Informatika, 2014.

[10] Wikipedia, "Tesseract (software)." [online]. Available: https://en.wikipedia.org/wiki/Tesseract_(software). [Diakses 3 Oktober 2015].

[11] Utomo, Eko Priyo, *Panduan Mudah Mengenal Bahasa Java*, Bandung : Yrama Widya, 2009.

[12] Simon, Jonathan, Ebook: *Head First Android Development*, O'Reilly Media, Inc., 2011.

[13] "Pengertian Android Studio." [online]. Available : http://www.twoh.co/2014/09/belajar-membuat-aplikasi-android-menggunakan-android-studio/. [Diakses 1 Juni 2015].

[14] Sommerville, Ian., *Rekayasa Perangkat Lunak Edisi 6 (Terjemahan)*, Jakarta : Erlangga, 2006.

[15] "Pengertian, Perbedaan White Box Dan Black Box Testing." [online]. Available: http://rivayiarifanto.blogspot.co.id/2014/03/pengertian-perbedaan-white-box-dan.html. [Diakses 3 Oktober 2015]