BAB II

TINJAUAN PUSTAKA

2.1. *Optical Character Recognition*

2.1.1. Pengertian

Menurut Priyatma, dkk (2004) pengenalan karakter dengan menggunakan alat optik (*optical character recognition*) adalah sebuah sistem komputer yang dapat membaca karakter termasuk (huruf), baik yang berasal dari sebuah alat pencetak (*printer* atau mesin ketik) maupun yang berasal dari tulisan tangan. Adanya sistem pengenal huruf ini akan meningkatkan fleksibilitas ataupun kemampuan dan kecerdasan sistem komputer.

Adanya sistem pengenal huruf yang cerdas akan sangat membantu usaha besar-besaran yang saat ini dilakukan banyak pihak yakni usaha digitalisasi informasi dan pengetahuan, misalnya dalam pembuatan koleksi pustaka digital, koleksi sastra kuno digital, dll. Dengan adanya pengenal huruf juga akan memudahkan penanganan pekerjaan yang memakai input tulisan seperti penyortiran surat di kantor pos, pemasukan data buku di perpustakaan, dll. Selain itu adanya sistem pengenal huruf otomatis maka *user* dapat lebih leluasa memasukkan

data karena *user* tidak harus memakai papan ketik tetapi bisa menggunakan pena elektronik untuk menulis sebagaimana *user* menulis di kertas. (Priyatma dkk, 2004)

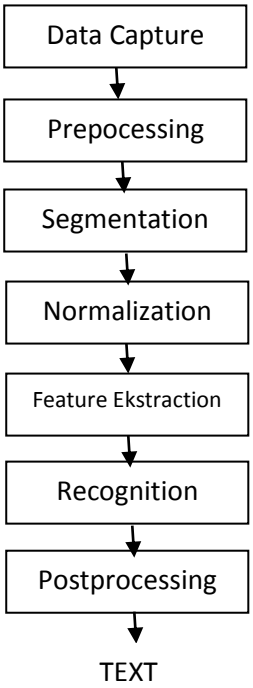
Dalam pengenal pola otomatis, sistem pengenal pola mencoba mengenali apakah citra masukan yang diterima cocok dengan salah satu citra yang telah ditentukan. Sistem ini misalnya dipakai untuk mendeteksi sidik jari, tanda tangan, bahkan wajah seseorang. Menurut Sukmawan (2008) OCR dapat dipandang sebagai bagian dari pengenal otomatis yang lebih luas yakni pengenal pola otomatis (*automatic pattern recognition*). Ada banyak pendekatan yang dapat dipakai untuk mengembangkan pembuatan pengenal pola otomatis antara lain memakai pendekatan numerik, statistik, sintaktik, neural dan, aturan produksi (*rule-based*). Secara umum metode-metode tersebut dapat digolongkan menjadi dua kelompok metode yakni metode berbasis statistik dan metode berbasis struktur.

Dalam metode yang berbasis statistik, setiap pola ditransformasi ke dalam vektor yang memakai ukuran dan karakteristik tertentu. Karakteristik ini seringkali lebih bersifat statistik misalnya distribusi pixel ataupun jarak piksel. Sedang dalam metode yang berbasis struktur, setiap pola yang diproses dinyatakan sebagai gabungan beberapa struktur elementer. Pengenalan selanjutnya dilakukan dengan mencocokkan komposisi struktur elementer dengan struktur

yang sudah disimpan memakai aturan tertentu misalnya memakai pendekatan teori bahasa formal dan automata. (Sukmawan, 2008)

2.1.2 Cara Kerja

Ilustrasi dari proses OCR



Gambar 2.1
Ilustrasi Proses OCR

Prinsip kerja dari aplikasi OCR menurut Manik (2010) adalah sebagai berikut:

1. Mengambil objek berupa teks menggunakan kamera sehingga didapat sebuah *file* citra.
2. *File* citra tersebut diproses menggunakan perangkat lunak aplikasi pengenalan teks, di mana perangkat ini melakukan proses pengenalan terhadap karakter-karakter yang ada pada *file* citra tersebut.

3. Keluaran dari perangkat lunak aplikasi pengenalan teks ini berupa file teks yang berisi karakter-karakter yang telah dikenali dan siap untuk diolah lebih lanjut.

Tingkat keberhasilan dari perangkat lunak pada aplikasi pengenalan teks ini sangat bergantung dari sejumlah faktor berikut:

1. Kualitas gambar teks yang ada pada dokumen yang dibaca serta tingkat kompleksitasnya (ukuran, format, teks, warna, latar belakang).
2. Kualitas perangkat lunak aplikasi pengenalan teks itu sendiri.
3. Kualitas alat optik yang dipakai *(kamera)*.

(Manik, 2010)

Proses yang dilakukan oleh OCR secara berurutan adalah *data capture, prepocessing, segmentation normalization, feature ekstraction, recognition* dan *postprocessing*.

1. Sistem *Preprocessing*

Perancangan sistem *preprocessing* menjelaskan alur kerja dalam OCR dalam meyiapkan *image* sebagai sebagai *input* gambar yang siap diolah lebih lanjut oleh sistem.

Menurut Sukamto (2010) proses *preprocessing* meliputi proses-proses berikut:

- *Grey Scalling/Thesholding*

  Proses *grey scaling* mengubah citra berwarna menjadi hitam putih dengan mengubah warna setiap komponen RGB gambar menjadi bernilai sama. Proses *threshold* digunakan untuk mengekstrak *foreground* (tinta) dari *background* (kertas) dan mengubah menjadi citra biner. Proses *Thresholding* mengubah warna gambar menjadi citra biner *(binary image)* dimana ditentukan sebuah nilai *level threshold* kemudian piksel yang memiliki nilai di bawah *level threshold* diset menjadi warna putih (0 pada nilai biner) dan nilai di atas nilai *threshold* diset menjadi warna hitam (1 pada nilai biner).

- *Smoothing*

  Proses *smoothing* menggunakan metode *Stentiford Boundary* untuk menghilangkan detail dan noise dengan memeriksa keterikatan dengan piksel bernilai 0 dengan piksel yang bernilai 1 dan menghitung tetangga piksel yang bernilai 0. Jika piksel yang sedang dicek memenuhi syarat tidak boleh dihapus, maka piksel ditandai sebagai piksel yang tidak boleh dihapus. *Smoothing* digunakan untuk meminimalisir derau (*noise).*

- *Scaling*

  Menskala citra karakter tulisan tangan menjadi lebih kecil agar proses lainnya lebih cepat dan tidak terpengaruh besar citra karakter (pada bagian inti karakter)

- *Stroke Thinning*

  Proses ini menggunkan metode *hybrid* di mana menggunakan campuran tiga buah metode yang saling mendukung untuk proses *thinning* yaitu algoritma *thinning* Zhang-Suen yang mengecek ketetenggaan dan konektivitas 8-arah pikase, Stentiford Acute Angel Emphasis untuk menandai piksel pada bagian luar yang tidak boleh dihapus, dan algoritma Holt untuk menghapus hasil *thinning* yang tidak diperlukan (percabangan yang tidak perlu).

2. Sistem *Segmentation*

Tujuan dari proses *segmentasi* adalah untuk mempermudah atau menyederhanakan representasi dari citra menjadi sesuatu yang lebih berarti dan mudah dianalisa. (Sukamto: 2010).

3. Normalization

Normalization adalah proses merubah dimensi region tiap karakter dan ketebalan karakter. Dalam OCR, algoritma yang digunakan pada proses ini adalah algoritma *Scaling* dan *Thinning.* (Sukamto: 2010).

4. Feature Extarction

Menurut Sukamto (2010) *Feature Extraction* adalah proses transformasi data masukan menjadi kumpulan fitur untuk mengambil representasi minimal dari data masukan. Fitur citra menyediakan cara untuk mendeskripsikan *property* citra. Fitur geometris merupakan kunci untuk mendeskripsikan struktur citra.

- *Ascenders dan Descenders*

  Merupakan pambagian tulisan menjadi tiga buah area yaitu bagian atas *(ascenders)*, bagian tengah *(main body)*, bagian bawah *(descenders)*. Kemudian setiap area diambil fiturnya dengan menggunakan histogram untuk membedakan karakter yang akan dikenali.

- Ukuran rata-rata tinggi dan lebar karakter

  Ukuran diambil dari perata-rataan setiap karakter yang dimasukan sebagai pembelajaran.

- Permodelan *stroke* menggunakan rangkaian *stroke* (garis tulisan) untuk mengenali karakter. Rangkaian *stroke* merupakan kumpulan titik-titik yang diberi label angka berdasarkan arah titik tetangga berikutnya yang disimpan di dalam *list* yang kemudian di periksa polanya. Label yang diberikan adalah sebagai berikut:

  ✓ Angka 1 untuk arah ke atas atau ke bawah (garis vertikal)

- ✓ Angka 2 untuk arah ke samping (garis horizontal)
- ✓ Angka 3 untuk arah ke kanan atas atau bawah (garis miring hadap kanan)
- ✓ Angka 4 untuk arah ke kiri bawah atau atas (garis miring hadap kiri)

Rangkaian label dapat lebih dari satu untuk menggambarkan fitur geometrinya.

5. **Sistem *Recognition***

Recognition merupakan proses untuk mengenali karakter yang diamati dengan cara membandingkan ciri-ciri karakter yang diperoleh dengan ciri-ciri karakter yang ada pada *database*.

(Sukamto: 2008)

6. Postprocessing

Pada umumnya proses yang dilakukan pada tahap ini adalah proses koreksi ejaan sesuai dengan bahasa yang digunakan

(Sukamto, 2008)

2.2. **Android**

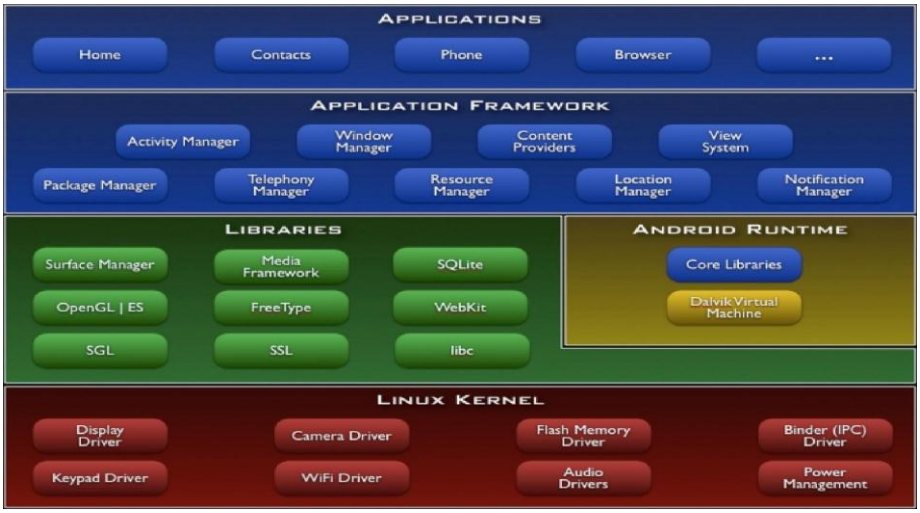
2.2.1. Pengertian

Android adalah software untuk perangkat *mobile* yang mencakup sistem operasi, *middleware* dan *key application*. SDK Android menyediakan *tools* dan API diperlukan untuk mengembangkan

aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. (Developer.android.com.2012.)

2.2.2. Arsitektur Android

Gambar 2.2 menunjukkan komponen utama dari sistem operasi Android.



Gambar 2.2
Arsitektur Android
(http://developer.android.com/guide/basics/what-is-android.html)

2.2.2.1. *Application*

Android merupakan suatu set aplikasi inti termasuk email client, program SMS, kalender, peta, *browser*, kontak, dan lainnya. Semua aplikasi dibuat menggunakan bahasa pemrograman Java. (Developer.android.com: 2012)

2.2.2.2. Libraries

Android menggunakan beberapa paket *library* yang terdapat pada C/C++ dengan standar *Berkeley Software Distribution* (BSD) hanya setengah dari yang aslinya yang tertanam pada kernel Linux. Beberapa *library* yang tertanam pada kernel Linux adalah:

- *Media Library* untuk memutar dan merekam berbagai macam format Audio dan video.
- *Surface Manager* untuk mengatur hak akses layer dari berbagai aplikasi.
- *Graphic Library* termasuk didalamnya *SGL* dan *OpenGL*, untuk Tampilan 2D dan 3D.
- *SQLite* untuk mengatur relasi *database* yang digunakan pada aplikasi.
- *SSl* dan *WebKit* untuk browser dan keamanan internet.

(Developer.android.com: 2012)

2.2.2.3. **Android *Runtime***

Android merangkum seperangkat *library* inti yang menyediakan sebagian besar fungsi yang tersedia di *library* inti dari bahasa pemrograman java. (Developer.android.com: 2012)

Setiap aplikasi android berjalan dalam prosesnya sendiri dengan contoh dari mesin virtual Dalvik. Dalvik telah ditulis agar perangkat dapat menjalankan multiple VMs secara efisien. VM Dalvik megeksekusi *file* dalam Dalvik excutable (.Dex) format

yang dioptimalkan untuk jejak memori minimal. VM adalah *register-based* dan menjalankan kelas dikompilasi oleh *compiler* bahasa Java yang telah ditransformasikan ke dalam format .dex dengan menggunakan dex *tools*. (Developer.android.com: 2012)

VM Dalvik bergantung pada kernel Linux untuk fungsi dasar seperti manajemen memori *threading* dan tingkat rendah.

(Developer.android.com: 2012)

2.2.2.4. Application Framework

Dengan menyediakan *platform* pengembangan aplikasi yang *open source*, Android menawarkan kepada pengembang untuk membangun aplikasi yang sangat kaya dan inovatif. Pengembang memiliki akses penuh ke API yang sama pada *framework* digunakan oleh aplikasi inti.

Bagian terpenting dalam kerangka aplikasi Android adalah sebagai berikut:

1. *Activity Manager*, berfungsi untuk mengontrol siklus hidup aplikasi dan menjaga keadaan ”*Backstack“* untuk navigasi penggunaan.
2. *A rich* dan *extensible* dari *Views* yang dapat digunakan untuk membangun aplikasi, termasuk *list*, *grids*, *text boxes*, *botton*, dan sebuah *browser web embeddable*

3. *Content Providers* yang memungkinkan aplikasi untuk mengakses data dari aplikasi lain (seperti Kontak), atau berbagi data mereka sendiri
4. *Resuource Manager* menyediakan akses sumber daya diluar kode program, seperti karakter, grafik, dan *file* layout.
5. *Notification Manager*, mencakup berbagai macam peringatan seperti, pesan masuk, janji, dan lain sebagainya yang akan ditampilkan pada *status bar.*

(Developer.android.com: 2012)

2.2.2.5. *Liux Kernel*

Android bergantung pada Linux versi 2,6 untuk layanan sistem inti seperti keamanan, manajemen memori, manajemen proses, stack jaringan, dan driver model. Kernel juga bertindak sebagai lapisan abstraksi antara perangkat keras dan seluruh *software stack*. (Developer.android.com: 2012)

2.3 *Image atau Citra*

2.3.1 **Pengertian**

Citra atau gambar pada dua dimensi dengan bentuk segi empat berformat *horizontal* dan *vertical* yang memiliki warna dan representasi *digital.* Citra sebagai fungsi menerus dari intensitas cahaya pada bidang dwimatra. Menurut Jain, dkk (1995) *Image* atau citra merupakan istilah lain dari gambar yang merupakan informasi berbentuk *visual. "A picture is more than a thousand words"* artinya "sebuah gambar bermakna lebih

dari seribu kata" maksudnya sebuah gambar akan memberikan informasi lebih banyak daripada informasi yang disajikan dalam bentuk kata-kata.

Ditinjau dari sudut pandang matematis, citra merupakan fungsi *continue* dari intensitas cahaya pada bidang dwitarma. Sumber cahaya, objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pemantulan cahaya ini ditangkap oleh alat-alat optik, misalnya mata manusia, kamera, pemindai (*scanner*), dsb, sehingga bayangan objek citra tersebut terekam. Definisi citra menurut kamus Webster adalah suatu representasi, kemiripan, atau imitasi dari suatu benda, secara harfiah, citra (*image*) adalah gambar pada bidang dwimatra (dua dimensi). (Jain, dkk: 1995)

Ada dua bagian pada proses pembentukan gambar, yaitu:

1. Geometri pembentukan citra yang menentukan proyeksi titik koordinat akan ditempatkan pada bidang gambar.
2. Cahaya fisik yang menentukan kecerahan titik pada bidang gambar sebagai fungsi dari iluminasi atau pencahayaan

(Jain dkk, 1995)

2.3.2 Elemen-elemen citra

A. Kecerahan dan Kontras

1. Kecerahan

Yang dimaksud kecerahan (*brightness*) adalah intensitas yang terjadi pada satu titik citra. Umumnya pada sebuah citra kecerahan ini merupakan kecerahan rata-rata dari suatu daerah lokal. (Suprianto, Agung: 2009).

2. Kontras

Untuk menentukan kepekaan kontras (*contras sensitivity*) pada mata manusia. (Suprianto, Agung: 2009).

B. ***Acuity***

Yang dimaksud *acuity* adalah kemampuan mata manusia untuk merinci secara detail bagian-bagian pada suatu citra (pada sumbu visual). (Suprianto, Agung: 2009).

C. ***Kontur***

Adalah keadaan pada citra dimana terjadi perubahan intensitas dari satu titik ke titik tetangganya. Dengan perubahan intensitas inilah mata seorang sanggup mendeteksi pinggiran atau kontur suatu benda. (Suprianto, Agung: 2009).

D. **Warna**

Adalah reaksi yang dirasakan oleh sistem visual mata manusia terhadap perubahan panjang gelombang cahaya. Setiap warna mempunyai panjang gelombang yang berbeda-beda. Warna merah memiliki panjang gelombang paling tinggi, sedangkan warna violet mempunyai panjang gelombang paling rendah. (Suprianto, Agung: 2009).

E. **bentuk**

Pada umumnya citra yang dibentuk oleh mata merupakan citra 2 dimensi, sedangkan objek yang diamati adalah 3 dimensi. (Suprianto, Agung: 2009).

F. **Tekstur**

Pada hakikatnya sistem *visual* manusia tidak menerima informasi citra secara terpisah pada setiap titik, tetapi suatu citra dianggap sebagai satu kesatuan. Dua buah citra tidak dapat disamakan hanya degan satu parameter saja. Hal ini tampak nyata dalam bentuk tektur *(texture)*. Pada daerah yang berdekatan tekstur dua buah citra mudah dibedakan, namun bila letaknya berjauhan tekstur kedua citra tersebut sukar dibedakan. (Suprianto, Agung: 2009).

G. **Waktu dan Pergerakan**

Respon suatu *visual* tidak hanya berlaku pada faktor ruang, tetapi juga faktor waktu. Sebagai contoh, bila citra-citra diam ditampilkan secara cepat, akan berkesan melihat citra bergerak. (Suprianto, Agung: 2009).

H. Deteksi dan Pengenalan

Dalam pendeteksian dan megenali suatu citra ternyata tidak hanya sistem *visual* manusia saja yang bekerja, tetapi juga ikut melibatkan ingatan dan daya pikir manusia. (Suprianto, Agung: 2009)

2.3.3 Pengolahan Citra

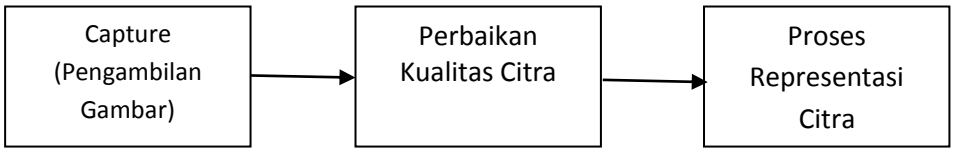
Menurut (Prima, RS: 2010) pengolahan citra digital adalah pemrosesan citra menjadi citra yang lain dengan kualitas yang lebih baik, yaitu pemrosesan pada usaha untuk memanipulasi. Citra yang telah menjadi gambar lain menggunakan algoritma atau teknik tertentu. Pengolahan citra mempunyai tujuan yaitu:

1. Proses memperbaiki kualitas citra agar mudah diinterpretasikan oleh manusia atau komputer.
2. Teknik pengolahan citra dengan mentrasformasikan citra menjadi citra lain.
3. Pengolahan citra merupakan proses awal dari komputer visi.

Ada beberapa hal yang penting di dalam pengolahan citra *digital*, antara lain teknik-teknik pengambilan citra, model citra *digital*,

sampling dan kuantitasi, *histogram*, proses filtering, perbaikan citra sampai pada pengolahan citra digital yang lebih lanjut seperti *segmentasi*, *image clustering* dan ekstrasi ciri. (Prima, RS: 2010).

Proses pengolahan citra secara diagram yaitu proses dimulai dari pengambilan citra, perbaikan citra sampai dengan pernyataan representatif citra digambarkan dengan gambar 2.2: (Prima, RS: 2010)



Gambar 2.3
proses pengolahan citra

Menurut (Prima, RS: 2010) secara umum teknik pengolahan citra digital dibagi menjadi 3 tingkat pengolahan, yakni:

1. Tahap 1 yang dinamakan dengan *Low-Level Processing* (pengolahan tingkat rendah). Pengolahan ini *operasional-operasional* dasar dalam pengolahan citra, seperti pengurangan *noise* (*noise redution*), perbaikan citra (*image enhancement*) dan restorasi citra (*image restoration*).
2. Tahap 2 yang dinamakan dengan *Mid-Level Processing* (pengolahan tingkat menengah). Pengolahan ini meliputi segmentasi pada citra, deskripsi objek dan klasifikasi objek secara terpisah.

3. Tahap 3 yang dinamakan dengan *High-Level Processing* (pengolahan tingkat tinggi), yang meliputi analisis citra

2.3.4 Jenis-Jenis Gambar

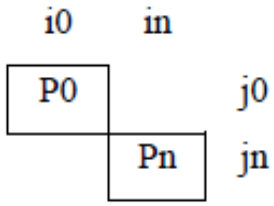
2.3.4.1 Gambar Biner

Gambar biner *(binery image)* adalah gambar dimana piksel-pikselnya hanya memiliki dua buah nilai intensitas, biasanya 0 dan 1 dimana 0 menyatakan warna latar belakang *(background)* dan 1 meyatakan warna tinta/objek *(foreground)* atau dalam bentuk angka 0 untuk wara hitam dan angka 255 untuk warna putih. *Background* merupakan kumpulan komponen yag terkoneksi yang bernilai 0 pada gambar biner sedangkan *foreground* merupakan kumpulan komponen terkoneksi atau tidak dan bernilai 1 pada gambar biner. (Sukamto, AR: 2008)

2.3.4.2 Piksel

Piksel biasanya direpresentasikan dengan titik-titik yang membentuk sebuah citra. Pada citra *digital* sebuah piksel adalah bagian terkecil informasi dari sebuah citra. Piksel-piksel yang terhubung *(connected pixels)* atau piksel-piksel ketetanggaan dapat diilustrasikan sebagai berikut, sebuah pixel P0 pada koordinat i0,j0 dikatakan

terkoneksi dengan piksel Pn pada koordinat in,jn jika dan hanya jika ada jalur antara P0 dan Pn yang merupakan urutan titik, seperti pada gambar 2.3: (Sukamto, AR: 2008)



Gambar 2.4
piksel terkoneksi (connected pixels)

Menurut Sukamto, AR (2008) ada beberapa jenis piksel ketetanggaan, antara lain 4-ketetanggaan, 6-ketetanggaan, dan 8-ketetanggaan, namun yang banyak digunakan adalah 4-ketetanggaan dan 8-ketetanggaan. Kumpulan piksel yang dikatakan sebagai 8-ketetanggaan (8-connected) adalah ketika sebuah piksel pada koordinat I,j memiliki piksel tetengga yaitu sebagai berikut:

tetangga i-1, j-1	tetangga i, j-1	tetangga i+1,j-1
tetangga	i,j	tetangga

Gambar 2.5
piksel 8-ketetanggaan

i-1, j		i+1, j
tetangga i-1, j+1	tetangga i, j+1	tetangga i+1, j+1

Gambar 2.6
piksel-8 ketetanggaan (2)

Relasi ketetanggaan dari sebuah piksel yang telah didefinisikan dapat dijadikan represetasi geometri secara diskrit.

2.4 Java

2.4.1 Pengertian Java

Java adalah sebuah bahasa pemrograman yang diciptakan oleh James Gosling, seorang developer dari Sun Microsystem pada tahun 1991. Selanjutnya Java dikembangkan Sun Microsystem dan banyak digunakan untuk menciptakan Executable Content yang dapat didistribusikan melalui network. (Nyura, Yusni: 2010)

Java dapat melakukan banyak hal dalam melakukan pemrograman, seperti membuat animasi halaman *web*, pemrograman Java untuk Ponsel dan aplikasi interaktif. Java adalah bahasa pemrograman *Object-Oriented* dengan unsur-unsur seperti bahasa C++ dan bahasa-bahasa lainnya yang memiliki *libraries* yang cocok untuk lingkungan internet. Java juga dapat digunakan untuk *handphone*, internet dan lain-lain. (Nyura, Yusni: 2010)

2.4.2 **Karakteristik Java**

a. Sederhana

Bahasa pemrograman Java menggunakan sintaks yang mirip dengan bahasa C++ namun sintaks pada Java telah banyak diperbaiki, terutama dengan menghilangkan *pointer* yang rumit dan *multiple*

inheritance. Java juga menggunakan *automatic memory allocation* dan *garbage collection*. (Nyura, Yusni: 2010)

b. Berorientasi Objek

Java merupakan bahasa pemrograman berorientasi objek yang memungkinkan program untuk dibuat secara modular dan digunakan kembali. (Nyura, Yusni: 2010)

c. Terdistribusi

Java dibuat untuk memudahkan distribusi aplikasi dengan adanya *networking libraries* yang terintegrasi dalam Java. (Nyura, Yusni: 2010)

d. *Interpreted*

Program Java dijalankan menggunakan program *Interpreter*, yaitu Java *Virtual Machine* (JVM). Hal ini menyebabkan source code Java yang telah dikompilasi menjadi *bytecodes* dapat dijalankan pada berbagai *platform*. (Nyura, Yusni: 2010)

e. *Robust*

Java mempunyai *reliabilitas* yang tinggi. *Kompiler* pada Java mempunyai kemampuan mendeteksi *error* yang lebih baik dibandingkan bahasa pemrograman yang lain. Java mempunyai

Runtime Exception Handling untuk membantu mengatasi *error* pada pemrograman. (Nyura, Yusni: 2010).

f. *Secure*

Sebagai bahasa pemrograman aplikasi internet dan terdistribusi, Java memiliki beberapa mekanisme keamanan untuk menjaga agar aplikasi tidak digunakan untuk merusak sistem komputer yang menjalankan aplikasi tersebut. (Nyura, Yusni: 2010).

2.4.3 Cara Kerja

Lingkungan pemrograman Java menggunakan *kompiler* sekaligus *interpreter* agar dapat berjalan pada *platform* yang berbeda. *Kompiler* Java akan mentransformasikan kode-kode dalam bahasa Java ke dalam suatu *bytecode*. *Bytecode* adalah sekumpulan perintah hasil kompilasi yang kemudian dapat dieksekusi melalui sebuah mesin komputer abstrak yang disebut dengan JVM. JVM juga sering dinamakan sebagai *interpreter*, karena sifatnya yang selalu menerjemahkan kode-kode yang tersimpan dalam *bytecode* dengan cara baris demi baris. (Nyura, Yusni: 2010)

2.5. Library OCR

2.5.1. Pengertian

Tesseract merupakan sebuah *library open source* untuk mengenali karakter OCR *(Optical Character recognition)* yang pada awalnya di

kembangkan oleh Hawlet-Packard antara tahun 1985 dan 1995. *Tesseract* tidak pernah dikomersialkan. Keakuratannya diuji di The Fourt Annual Test of OCR Accurasi yang diselenggarakan pada tahun 1995 di Universitas Nevada Las Vegas' Information Science Research Institute. Namun pada saat itu pengembangan *tesseract* telah berhenti. (Saltcymru.org: 2012)

2.5.2. Cara Kerja

Gambaran mengenai cara kerja Tesseract secara singkat adalah:

1. Garis-garis dianalisis dan disimpan
2. Garis-garis berkumpul bersama sebagai himpunan
3. Himpunan tersebut akan disusun dalam baris teks
4. Baris teks akan dipecah menjadi kata-kata
5. Langkah pertama proses pengenalan mencoba untuk mengenali tiap kata pada urutannya
6. Kata yang tepat harus melewati proses penyesuaian oleh *adaptive trainer*
7. Pembelajaran oleh adaptive trainer berlangsung pada tahap kedua yang mencoba mengenali kata-kata yang tidak dikenali secara tepat pada tahap pertama
8. proses fuzzy diselesaikan dan teks diperiksa *small caps*
9. Teks digital dikeluarkan

(Saltcymru.org: 2012)

Menurut (Saltcymru.org: 2012), selama proses tersebut berlangsung, *tesseract* menggunakan:

- Algoritma untuk mendeteksi *skewed pages.*
- Algoritma untuk mendeteksi kata-kata proporsional dan non proporsional (proporsional merupakan sebuah kata di mana semua huruf yang lebar yang sama).
- Algoritma untuk mencacah karakter bergabung dan untuk menghubungkan karakter yang rusak.
- Linguistik analisis untuk mengidentifikasi kata yang paling mungkin dibentuk oleh sekelompok karakter.
- Dua karakter pengklasifikasi: classifier statis, dan sebuah classifier yang adaptif menggunakan data pelatihan, dan yang lebih baik dalam membedakan antara atas dan huruf kecil.

Tesseract memiliki akurasi yang tinggi jika teks yang diterjemahkan berasal dari bahasa yang *support*.

Untuk bahasa Inggris, 8 komponen yang digunakan:

1. *General Words Wordlist* (`tessdata/eng.word-dawg`).
2. *Frequent Word Wordlist* (`tessdata/eng.freq-dawg`).
3. *User Wordlist* (`tessdata/eng.user-words`).
4. *Index for Character Set* (`tessdata/eng.inttemp`).
5. Box file for use in locating characters in the training file (`tessdata/eng.normproto`).
6. *Box file for use in locating characters in the training file*

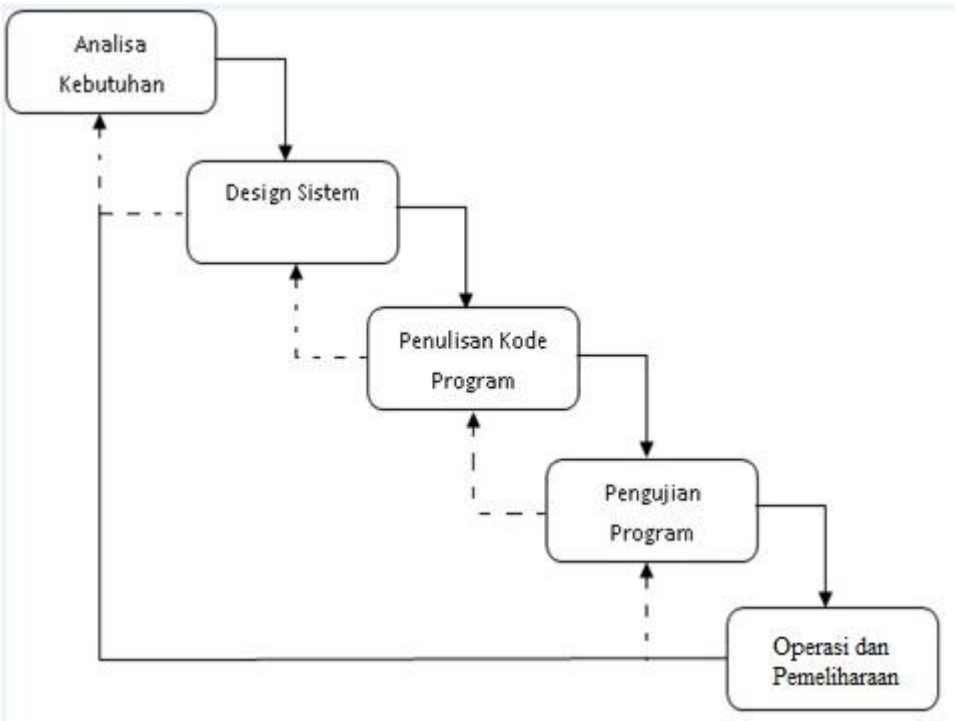
(`tessdata/eng.pffmtable`).

7. *Language's Character Set* (`tessdata/eng.unicharset`).
8. *Character Cluster Disambiguator for 'm' and 'rn', for instance.* (`tessdata/eng.DangAmbigs`)

(Saltcymru.org: 2012)

2.6. Metode Waterfall

Menurut Samantha, DP (2011) *Waterfall model* merupakan salah satu model proses perangkat lunak yang mengambil kegiatan proses dasar seperti spesifikasi, pengembangan, validasi, dan evolusi, dan merepresentasikannya sebagai fase-fase proses yang berbeda seperti analisis dan definisi persyaratan, perancangan perangkat lunak, implementasi, pengujian unit, integrasi sistem , pengujian sistem, operasi dan pemeliharaan.



Gambar 2.7

Metode waterfall

1. Analisa Kebutuhan yaitu Proses mengumpulkan informasi kebutuhan sistem/perangkat lunak melalui konsultasi dengan *user system*. Proses ini mendefinisikan secara rinci mengenai fungsi-fungsi, batasan dan tujuan dari perangkat lunak sebagai spesifikasi sistem yang akan dibuat. (Samantha,DP : 2011).

2. Design Sistem merupakan proses yang difokuskan pada empat atribut, yaitu struktur data, arsitektur perangkat lunak, representasi antarmuka, dan detail (algoritma) prosedural. Yang dimaksud struktur data adalah representasi dari hubungan logis antara elemen-elemen data individual.

   (Samantha,DP : 2011)

3. Penulisan Kode program merupakan tahap dimana perancangan perangkat lunak direalisasikan sebagai serangkaian program atau unit program. Kemudian pengujian unit melibatkan verifikasi bahwa setiap unit program telah memenuhi spesifikasinya. (Samantha,DP : 2011).

4. Pengujian Program yaitu unit program/program individual diintegrasikan menjadi sebuah kesatuan sistem dan kemudian dilakukan pengujian. Dengan kata lain, pengujian ini ditujukan untuk menguji keterhubungan dari tiap-tiap fungsi perangkat lunak untuk menjamin bahwa persyaratan sistem telah terpenuhi. Setelah pengujian sistem selesai dilakukan, perangkat lunak dikirim ke pelanggan/*user*. (Samantha,DP : 2011).

5. Operasi dan Pemeliharaan yaitu tahap ini biasanya memerlukan waktu yang paling lama. Sistem diterapkan (di-*install*) dan dipakai. Pemeliharaan mencakup koreksi dari beberapa kesalahan yang tidak diketemukan pada tahapan sebelumnya, perbaikan atas implementasi unit sistem dan pengembangan pelayanan sistem, sementara persyaratan-persyaratan baru ditambahkan. (Samantha,DP : 2011).