BAB 2
LANDASAN TEORI

2.1 Pengertian Umum Kamera

Kamera adalah alat paling populer dalam aktivitas fotografi. nama ini didapat dari camera *obscura*, bahasa Latin untuk "ruang gelap", mekanisme awal untuk memproyeksikan tampilan di mana suatu ruangan berfungsi seperti cara kerja kamera fotografis yang modern, kecuali tidak ada cara pada waktu itu untuk mencatat tampilan gambarnya selain secara manual mengikuti jejaknya. Dalam dunia fotografi, kamera merupakan suatu peranti untuk membentuk dan merekam suatu bayangan potret pada lembaran film. Pada kamera televisi, sistem lensa membentuk gambar pada sebuah lempeng yang peka cahaya. Lempeng ini akan memancarkan elektron ke lempeng sasaran bila terkena cahaya. Selanjutnya, pancaran elektron itu diperlakukan secara elektronik. Ada banyak jenis kamera dan masing-masing fungsinya yang akan dijelaskan pada subbab ini [1].

2.1.1 Kamera Digital

Kamera jenis ini merupakan kamera yang dapat bekerja tanpa menggunakan film. Si pemotret dapat dengan mudah menangkap suatu objek tanpa harus susah-susah membidiknya melalui jendela pandang karena kamera digital sebagian besar memang tidak memilikinya. Sebagai gantinya, kamera digital menggunakan sebuah layar LCD yang terpasang di belakang kamera. Lebar layar LCD pada setiap kamera digital berbeda-beda. Sebagai media penyimpanan, kamera digital menggunakan *internal memory* ataupun *external memory* yang menggunakan *memory card*.

2.1.2 Kamera CCTV

CCTV (*Closed Circuit Television*) memiliki kamera yang akan mentransmisikan image video ketempat yang spesifik dan jumlah televisi yang terbatas. Perbedaannya dengan bentuk televidi CCTV tidak dapat menerima monitor lain, bahkan jika diarea yang sama sekalipun, kecuali monitor tersebut telah

masuk ke dalam area CCTV. Sistem CCTV biasanya digunakan alasan keamanan atau komersial ketika orang memerlukannya bila berada di lingkungan yang berbahaya.

CCTV pefiama kali dibuat oleh Waller Brunch, dan di-install disebuah area peluncuran roket di Jerman. Oleh karena peluncuran tersebut dirasa berbahaya, dan banyak orang yang ingin menyaksikannya, maka dibuatlah CCTV sehingga dapat digambarkan secara detail mengenai peluncurannya. Teknologi CCTV masih digunakan untuk melihat peluncuran roket, namun rneluas fungsinya ke keamanan bank, institusi militer dan tempat lain yang membutuhkan pengamanan yang tinggi. Ditahun 1990 dan 2000, camera CCTV mulai dipakai di *area public*, seperti di sudut jalan di negara lnggris.

Teknologi CCTV telah mernbuat evolusi jalan keamanan disektor publik dan *private*. CCTV juga diperbolehkan oleh lingkungan hukum untuk menyelesaikan kriminalitas di area, dimana camera CCTV dipasang. Sekarang ini, camera CCTV mudah diidentifikasikan oleh setiap orang. Banyak kamera CCTV yang dipasang di langit-lagit rumah, dinding atau atap bangunan. Kamera CCTV memiliki lensa dibagian depan, dan untuk CCTV model baru berwarna hitam dan bentuk kecil, juga dapat melakukan manuver petaran 360 derajat.

2.1.3 Kamera Webcam

Webcam atau *web camera* adalah sebuah kamera video digital kecil yang dihubungan dengan komputer melalui *port* USB atau *port* COM istilah web camera merujuk pada teknologi secara umumnya, sehingga kata web terkadang diganti dengan kata lain yang mendeskripsikan pemandangan yang ditampilkan di kamera. Beberapa contoh pemakaian webcam yaitu *TruffiCam* yang digunakan untuk memonitor keadaan jalan raya, cuaca dengan *WeatherCam* bahkan dalam memonitoring gunung berapa digunakan *VolcanoCam*. Sebuah *web camera* yang sederhana terdiri dari sebuah lensa standar, dipasang disebuah papan sirkuit untuk menangkap sinyal gambar, *casing* (penutup), termasuk *casing* depan dan *casing* samping untuk menutupi lensa standar dan memiliki sebuah lubang lensa di-*casing* depan yang berguna untuk memasukkan gambar, kabel *support*, yang sibuat dari

bahan yang fleksibel, salah satu ujungnya dihubungkan dengan papan sirkuit dan ujung satu lagi memiliki *connector*, kabel ini dikontrol untuk menyesuaikan ketinggian, arah dan sudut pandang *web camera*. Sebuah *web camera* biasanya dilengkapi dengan *software, software* ini mengambil gambar-gambar dari camera digital secara *real-time* ataupun dalam interval waktu tertentu dan menyiarkannya melalui koneksi internet. Ada beberapa metode penyiaran, metode yang paling umum digunakan adalah aplikasi yang merubah gambar ke dalam bentuk file JPEG dan mengunggahnya ke *webserver* menggunakan *File Transfer Protocol* (FTP). *Frmae rate* mengindikasikan jumlah gambar sebuah *software* dapat diambil dan transfer dalam satu detik. Sebuah *web camera* tidak harus selalu terhubung dengan komputer, ada *web camera* yang memiliki *software web camera* dan *web server bulit-in,* sehingga yang diperlukan hanyalah koneksi internet. *Web camera* seperti ini dinamakan "*network camera*".

Sekarang ini *web camera* yang ada di pasaran pada umurnnya terbagi ke dalam dua tipe yang penama *web camera* permanen (*fixed*) dan *revolving web camera*. Pada *web camera* permanen terdapat pengapit untuk mengapit lensa standar diposisi yang diinginkan untuk menangkap gambar pengguna. Sedangkan pada *revolving web camera* terdapat landasan dan lensa standar dipasang dilandasan tersebut sehingga dapat disesuaikan ke sudut pandang yang terbaik untuk menangkap gambar pengguna. *Web camera* memiliki fitur-fitur dan pengaturan yang bermacam- macam, diantaranya adalah:

1. *Motion Sensing*

   Web camera akan mengambil gamar ketika kamera menndeteksi gerakan.
2. *Image archiving*

   Penggunaan dapat membuat sebua archiving yang menyimpan semua gambar dari *web camera* atau hanya gambar-gambartertentu saat *interval pre-set.*
3. *Video messaging*

   Beberapa *program messaging* mendukung fitur ini.
4. *Advanced connecting*

Menyambungkan perangkat *home theater* ke *web camera* dengan kabel meupun nirkabel.

5. *Antomotion*

   Camera robotic yang memungkinkan pengmbilan gambar sesuai *setting program* pengambilan *frame* berdasarkan posisi kamera.

6. *Streaming med*ia

   Aplikasi profesional, *setup web camera* dapat menggunakan kompresi MPEG4 untuk mendapatkan *streaming audio* dan *video* yang sesungguhnya.

7. *Custom coding*

   Mengimport kode komputer pengunaan untuk memberitahukan *web camera* apa yang harus dilakukan (misal : *automatically refresh*)

2.2 Teori Dasar Pengolahan Citra Digital

Istilah citra digital sangat populer pada masa sekarang. Banyak peralatan elektronik, misalnya scanner, kamera digital, mikroskop digital, dan *fingerprint reader* (pembaca sidik jari), yang menghasilkan citra digital. Perangkat lunak untuk mengolah citra digital juga sangat populer digunakan oleh pengguna untuk mengolah foto atau untuk berbagai keperluan lain. Sebagai contoh, Adobe Photoshop dan GIMP (*GNU Image Manipulation Program*) menyajikan berbagai fitur untuk memanipulasi citra digital [1].

Secara umum, istilah pengolahan citra digital menyatakan "pemrosesan gambar berdimensi-dua melalui komputer digital" (Jain, 1989). Menurut Efford (2000), pengolahan citra adalah istilah umum untuk berbagai teknik yang keberadaannya untuk memanipulasi dan memodifikasi citra dengan berbagai cara. Foto adalah contoh gambar berdimensi dua yang dapat diolah dengan mudah. Setiap foto dalam bentuk citra digital (misalnya berasal dari kamera digital) dapat diolah melalui perangkat lunak tertentu. Sebagai contoh, apabila hasil bidikan kamera terlihat agak gelap, citra dapat diolah menjadi lebih terang. Dimungkinkan pula untuk memisahkan foto orang dari latar belakangnya. Gambaran tersebut menunjukkan hal sederhana yang dapat dilakukan melalui pengolahan citra digital.

Tentu saja, banyak hal pelik lain yang dapat dilakukan melalui pengolahan citra digital.

Tujuan umum dari pengolahan citra digital yaitu memperbaiki kualitas gambar dilihat dari aspek radiometrik (peningkatan kontras, transformasi warna, restorasi citra) dan dari aspek geometri (rotasi, translasi, skala, tranformasi geometrik. Melakukan proses penarikan informasi atau deskripsi objek atau pengenalan objek yang terkandung pada citra. Selain itu juga pengolahan digital sebagai kompresi atau reduksi data untuk tujuan penyimpanan data, transmisi data dan waktu proses data.

2.2.1 Definifisi Citra Digital

Citra atau gambar dapat didefinisikan sebagai sebuah fungsi dua dimensi f(x, y), dimana x dan y adalah koordinat bidang datar dan harga fungsi f di setiap pasangan koordinat (x,y) disebut intensitas atau level keabuan (*grey scale*) dari gambar di titik itu. Jika x, y dan f semuanya berhingga (*finite*), dan nilainya diskrit, maka gambarnya disebut citra digital (gambar digital). Sebuah citra digital terdiri dari sejumlah elemen yang berhingga, di mana masing-masing mempunyai lokasi dan nilai tertentu. Elemen-elemen ini disebut sebagai *picture element, image element, pels* atau *pixel*.

Bidang digital image processing meliputi pengolahan digital image dari suatu komputer digital. Gambar dihasilkan dari seluruh spektrum elektromagnetik mulai dari gamma sampai gelombang radio. Ada tiga tipe pengolahan citra yaitu diantaranya :

1. Low-level process

   Low-level process meliputi operasi dasar seperti *image preprocessing* :

   a. Reduce Noise
   b. Contrast enhancement
   c. Image sharpennig

   Pada level ini baik input maupun output adalah berupa gambar.
2. Mid-level process

Mid-level process meliputi segmentasi (membagi sebuah gambar dalam regional atai *object*), mendeskripsikan objek tersebut untuk mereduksi dalam bentuk yang diinginkan dan klasifikasi (*recognation*) dari objek tersebut. Input dari proses ini berupa gambar dan output-nya berupa atribut yang diambil dari gambar tersebut(misal : *edge, contrast* dan identitas dari objek tertentu).

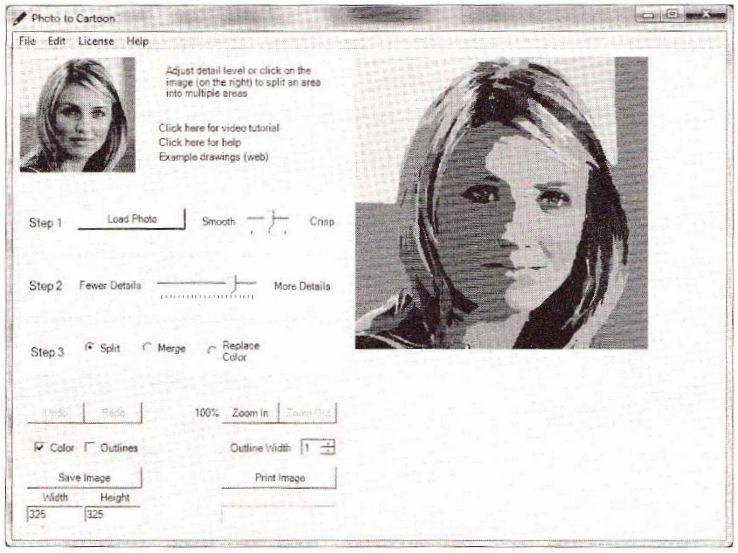
3. High-level process

   High-level process meliputi pemberian arti dari suatu rangkaian objek-objek yang dikenali dan akhirnya menampilkan fungsi-fungsi kognitif secara normal sehubungan dengan penglihatan.

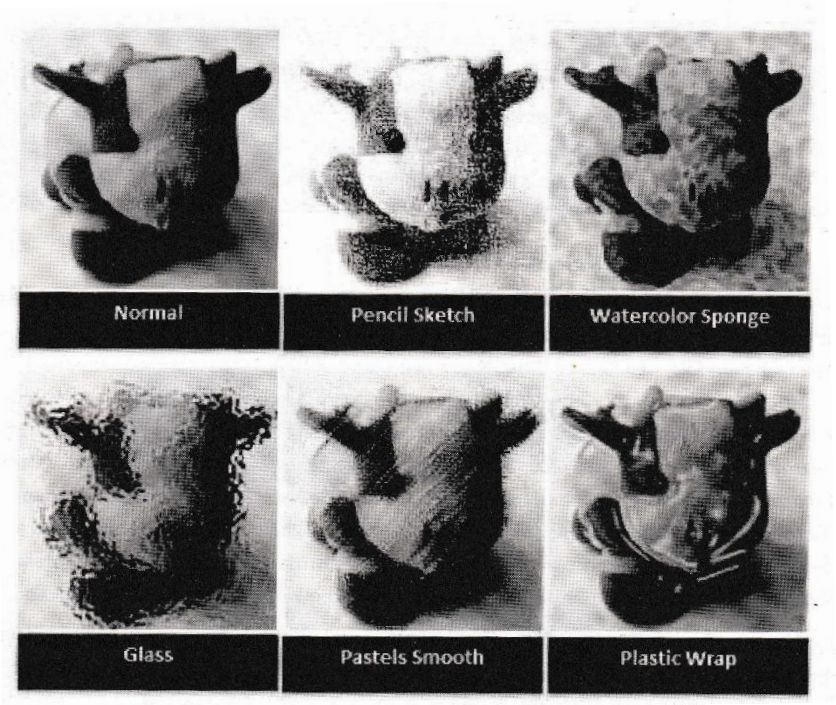
2.3 Aplikasi Pengolahan Citra

Pengertian citra merupakan bagian penting yang mendasari berbagai aplikasi nyata, seperti pengenalan pola, penginderaan jarak jauh melalui satelit atau pesawat udara, dan machine vision. Pada pengenalan pola, pengolahan citra antara lain berperan memisahkan objek dari latar belakang secara otomatis. Selanjutnya, objek akan diproses oleh pengklasifikasi pola. Sebagai contoh, sebuah objek buah dapat dikenali sebagai jeruk, apel, atau pepaya. Pada penginderaan jarak jauh, tekstur atau warna pada citra dapat dipakai untuk mengidentifikasi objek-objek yang terdapat didalam citra. Pada *machine vision* (sistem yang dapat "melihat" dan "memahami" yang dilihatnya), pengolahan citra berperan untuk mengenali bentuk-bentuk khusus yang dilihat oleh mesin. Penggunaan kamera pemantau ruangan merupakan contoh bagian aplikasi pemrosesan citra perubahan gerakan yang ditangkap melalui citra dapat menjadi dasar untuk melakukan pelaporan situasi yang terekam.

Pengolahan citra juga memungkinkan wajah seseorang dikartunkan. Sebagai contoh ditunjukan digambar Gambar 2.1. aplikasi seperti itu memungkinkan pembuatan kartun yang didasarkan pada objek-objek nyata. Perangkat lunak seperti Microsoft Word 2010 menyertakannya fasilitas pengolahan gambar yang memungkinkan setiap gambar seperti goresan pensil, kapur, pastel, dan lain-lain, sebagaimanan ditunjukan di Gambar 2.1 [1].



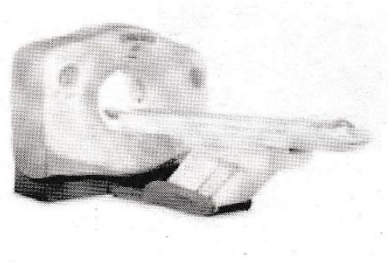
Gambar 2.1 Aplikasi Pengolhana citra untuk mengartunkan wajah orang



Gambar 2.2 Penggunaan efek gambar pada Microsoft World 2010 memanfaatkan pengolahan citra

Aplikasi pengolahan citra memang makin meluas. Di dunia kedokteran, pengolahan citra mempunyai peran sangat besar. CT scan (*Computed Tomography Scan*) atau kadang disebut CAT scan (*Computerized Axial Tomography Scan*)

merupakan suatu contoh apliaksi pengolahan citra, yang dapat dipakai untuk potongan atau penampang bagian tubuh manusia. Tomografi adalah proses untuk menghasilakan citra berdimensi dua dari potongan objek berdimensi tiga dari sejumlah hasil pemindaian satu dimensi. Gambar 2.3 memperlihatkan contoh CT Scan dan hasilnya.



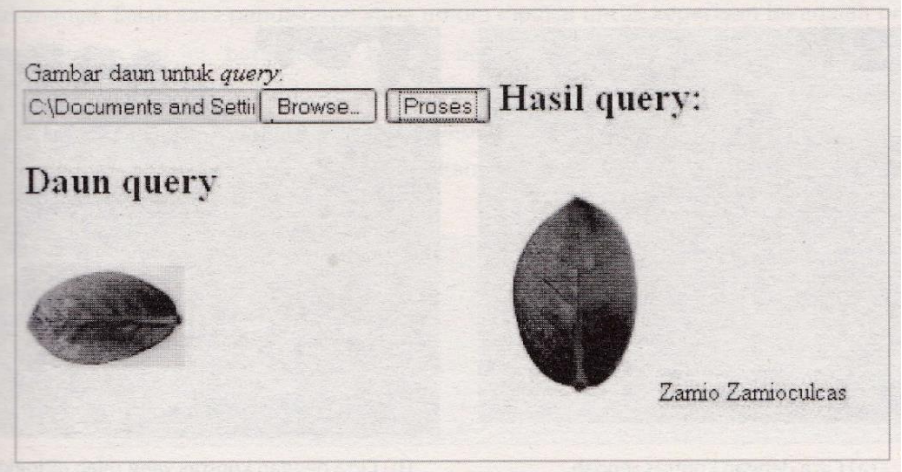
(a) CT Scan (sumber : http://www.dxhealthcorp.com)



(b) contoh hasil CT Scan (sumber : www.thirdage.com)

Gambar 2.3 CT Scan

Pengolahan citra juga dapat dimanfaatkan, misalnya untuk kepentingan penentuan jenis tanaman hias melalui ciri-ciri citra daun. Seseorang yang ingin tahu mengenai suatu tanaman cukup memasukan citra daun yang dimiliki dan kemusian memunggahkan kesistem berbasis web. Selanjutnya, sistem web dapat mencarikan informasi yang sesuai dengan citra tersebut. Gambar 2.3 memperlihatkan contoh hasil pencarian jenis tanaman hias.



Gambar 2.4 Masukan citra dapat digunakan sebagai dasar pencarian jenis tanaman hias

Berbagai aplikasi pengolahan citra juga telah dilakukan di indonesi. Beberapa contoh ditunjukan berikut ini.

1. Identifikasi sidik jari (Isnanto, dkk.,2007).
2. Pencarian database orang melalui foto orang (Aribowo,2009).
3. Identifikasi kematangan buah tomat (Noor dan Hariadi, 2009).
4. Identifikasi penyakit diabetes mellitus melalui citra kelopak mata (Rachmad, 2009).
5. Ekstrasi fitur motif batik (Mualaab, 2010).
6. Identifikasi telapak tangan (Putra dan Erdiawan, 2010).

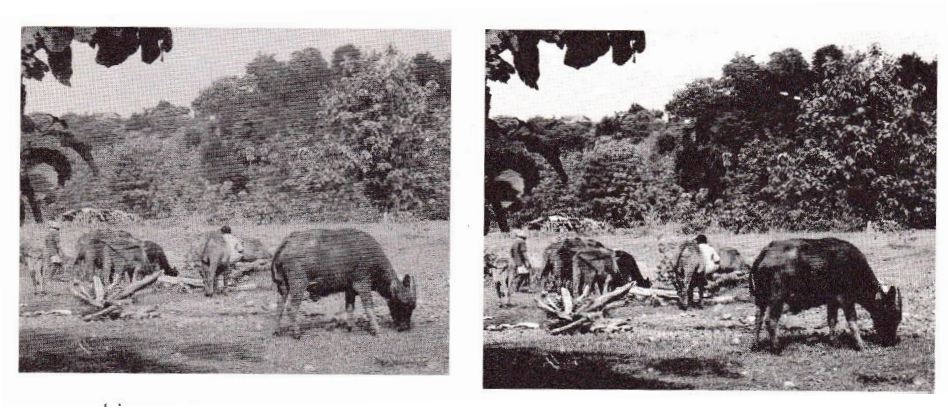
2.4 Prinsip Dasar Dalam Pengolahan

Hal-hal yang telah dijelaskan di Gambar 2.4 merupakan contoh-contoh aplikasi kegiatan pengolahan citra. Aplikasi-aplikasi seperti itu sesungguhnya menggunakan prinsip dasar dalam pengolahan citra seperti penignkatan keceranahan dan kontras, penglihatan derau pada citra, dan pencarian bentuk objek. Beberapa contoh diberikan berikut ini [1].

2.4.1 Peningkatan Kecerahan dan Kontras

Seringkali dijumpai citra yang tidak jelas diakibatkan kekurangan sinar ketika objek dibidik melalui kamera digital. Dengan menggunakan pengolahan citra, citra seperti itu dapat diperbaiki melalui peningkatan kecerahan dan kontras.

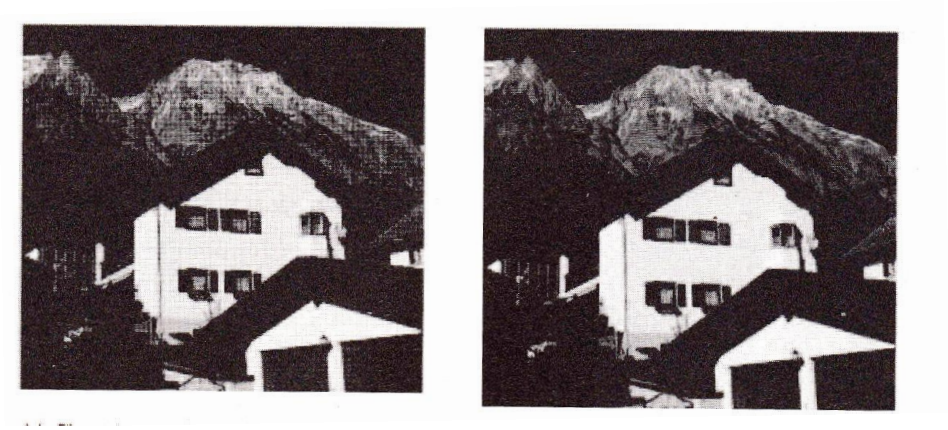
Gambar 2.5 menunjukan contoh citra yang kurang cerah dan kurang kontras menjadi citra yang lebih kontras.



Gambar 2.5 Citra dengan kontras rendah, Citra dengan kontras yang telah ditinggikan

2.4.2 Penghilangan Derau

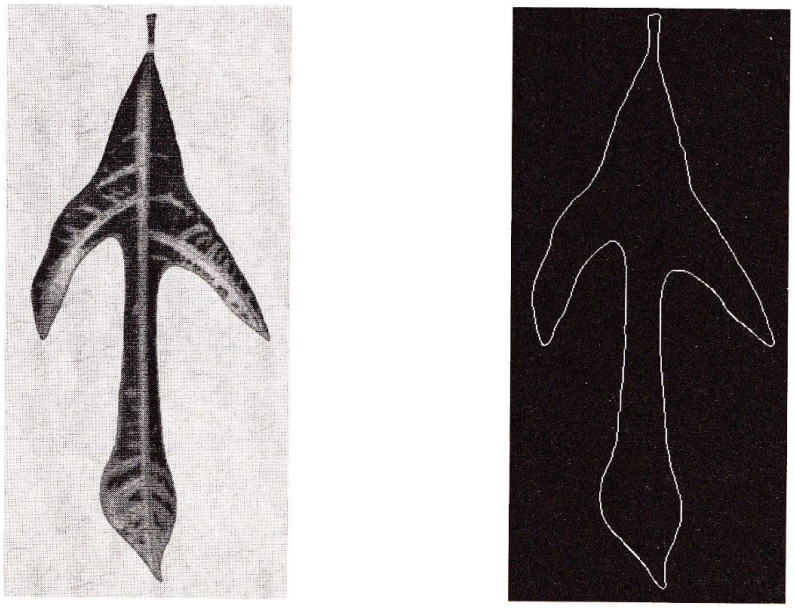
Citra yang akan diproses seringkali dalam keadaan tersidtorsi atau mengandung derau. Untuk kepintangan tertentu, derau tersebut perlu dibersihkan terlebih dahulu. Dalam pengolahan citra terdapat beberapa metode yang dapat dipakai untuk keperluan tersebut. Salah satu cara dilaksanakan melalui filter notch. Efek filter tersebut dapat dilihat di Gambar 2.5.



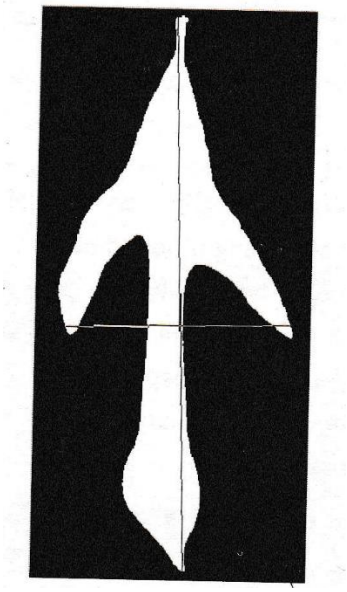
Gambar 2.6 Citra dengan derau berbentuk kotak-kotak, citra dengan derau telah dihilangkan

2.4.3 Pencarian Bentuk Objek

Untuk kepentingan mengenali suatu objek didalam citra, objek perlu dipisahkan terlebih dulu dari latar belakangnya. Salah satu pendekatan yang umum dipakai untuk keperluan ini adalah penemuan batas objek. Dalam hal ini, batas objek berupa bagian tepi objek. Setelah objek diketahui, encarian ciri terhadap objek dapat dilaksanakan, misalnya berdasar perbandingan panjang dan lebar daun. Objek daun dan batas daun yang didapatkan melalui pengolahan citra diperlihatkan di Gambar 2.6. adapun Gambar 2.7 memperlihatkan tahapan penentuan panjang dan lebar daun diperoleh melalui komputasi oleh komputer [2].



Gambar 2.7 Citra asli, Citra dengan hanya bagian luar yang digambar



Gambar 2.8 Penentuan panjang dan lebar berdasarkan tepi daun

2.5 Segmentasi Citra

Segmentasi merupakan proses membagi suatu citra ke dalam komponen-komponen region atau objek. Algoritma segmentasi secara umum didasarkan pada salah satu dari sifat dasar nilai intensitas, yaitu :

a. *Discontinuity* ialah pendekatan dengan membagi citra berdasarkan perubahan besar pada nilai intensitasnya, seperti tepi citra.
b. *Similarity* ialah pendekatan dengna membagi citra kedalam region-region yang serupa sesuai dengan kriteria awal yang diberikan. Contoh pendekatan ini ialah *Treshold ing, region growing* dan region splitting and merging

2.5.1 Deteksi *Discontinuity*

Ada tiga tipe diskontynuitas *gray-level*, yaitu :

1. Titik
2. Garis
3. Tepi

Salah satu cara yang paling banyak digunakan untuk melihat diskontinyuitas adalah menjalankan seuah mask atau filter pada citra seperti pada Gambar 2.9 berikut [2] :

W1	W2	W3
W4	W5	W6
W7	W8	W9

Gambar 2.9 Contoh filter citra

Operasi yang dilakukan adalah *sum of product* dari koefesien-koefesien dalam mask dengan *grey-level* yang ada didaerah dimana mask itu berada. Respon dari mask pada suatu titik dalam citra diberikan oleh persamaan :

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9 = \sum w_1 z_1 \quad \textbf{(2.1)}$$

2.5.1.1 Deteksi Titik

Deteksi dari titik yang terisolasi dalam sebuah citra cukup mudah. Dengan menggunakan mask pada Gambar 2.10 didapatkan bahwa suatu titik dapat dideteksi pada pusat lokasi dimana mask tersebut berada jika

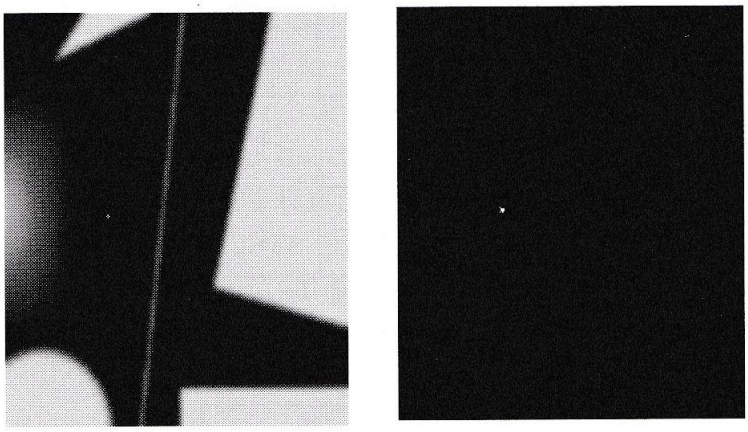
$$|R| \geq T \quad \textbf{(2.2)}$$

Dimana T merpakan suatu nilai threshold yang tidak negatif.

-1	-1	-1
-1	8	-1
-1	-1	-1

Gambar 2.10 *Mask threshold*

Isolated point adalah sebuah titik yang *grey-level*-nya sangat berbeda dari *background*-nya dan yang ada didaerah yang homogen hampir homogen. Jumlah koefesien mask adalah nol, mengindikasikan bahwa respons mask akam bernilai nol pada area *gray-level* yang konstan.



Gambar 2.11 (a) Citra Asli, (b) Hasil deteksi titik dgn T=90% dari nilai intensitas terbesar

2.5.1.2 Deteksi Garis

Pada umumnya deteksi garis atau *line detection* merupakan perbandingan dalam menentukan sebuah garis yang ada pada citra digital yang sudah di threshold, untuk mencapai respon pada suatu garis yang kuat pada citra dibutuhkan ketebalan suatu piksel. Setelah itu dapat di peroleh hasil maksimum dalam menentukan garis yang ada pada citra. Untuk mendeteksi garis horisontal, vertikal maupun miring yaitu dengan menentukan *mask* yang digunakan. *Mask* diperlukan sebagai *background* yang dilewati citra garis .

2.5.1.3 Deteksi Tepi

Tepi adalah sebuah himpunan dari piksel-piksel yang terhubung yang berada pada batas (*boundary*) diantara dua *region*. Definisi tepi membutuhkan kemampuan untuk mengukur transisi *gray-level* dengan cara yang tepat. Suatu model tepi yang ideal merupakan suatu himpunan connected pixel yang masing-masing berada pada suatu perubahan langkah tegak lurus (*orthogonal*) dalam *gray-level*. Tapi kenyataannya, hasil sampling atau akuisisi citra belum tentu menghasilkan citra yang sesempurna itu, misalkan citra lebih kabur dan sebagainya. Akibatnya tepi lebih mendekati model '*ramplike*'. Kemiringan (*slope*) dari ramp sesuai dengan tingkat kekaburan dari tepi.

Ketebalan dari tepi ditentukan oleh panjang dari ramp, yang merupakan transisi dari *gray-level* awal sampai *gray-level* akhir. Panjang ramp ditentukan oleh *slope*, yang ditentukan juga oleh tingkat kekaburan. Berarti bahwa tepi yang kabur cenderung tebal dan tepi yang tajam cenderung tipis. *Magnitude* dari turunan pertama dapat digunakan untuk mendeteksi adanya suatu tepi pada sebuah titik dalam citra (yang berarti menentukan apakah titik tersebut ada di ramp).

Turunan pertama dalam pengolahan citra diimplementasikan menggunakan besaran dari *gradient* Pada subbab sebelumnya, kita mengetahui bahwa pengaburan (*blurring*) citra pada domain spasial dilakukan dengan rata-rata piksel (*averaging*) dalam suatu window ketetanggaan. Karena *averaging* analog dengan integral, maka dapat dikatakan penajaman citra analog dengan diferensiasi spasial.

2.5.2 *Thresholding*

2.5.2.1 Pengertian Dasar

Histogram sebelah kiri mewakili citra f(x,y), yang tersusun atas objek terang di atas *background* gelap. Piksel-piksel objek dan *background* dikelompokkan menjadi dua mode yang dominan. Cara untuk mengekstrak objek dari dari *background* adalah dengan memilih *threshold* T yang memisahkan dua mode tersebut.

Sembarang titik (x,y) yang memenuhi f(x,y)>T disebut titik objek; selain itu, titik disebut titik background. Histogram sebelah kanan terbagi menjadi tiga mode. Misalkan, citra terdiri atas dua objek terang di atas *background* gelap. *Thresholding multilevel* digunakan untuk mengklasifikasikan suatu titik.

2.5.2.2 Basic Global Thresholding

Teknik *thresholding* yang paling sederhana adalah mempatisi histogram dengan menggunakan sebuah *threshold ing* global T. Segmentasi dilakukan dengan memeriksa piksel semi piksel dari citra dan melabeli setiap piksel sebagai objek atau sebagai *background*. Tergantung pada tingkat keabuan dari piksel tersebut, apakah lebih besar atau lebih kecil dari T.

Threshold ing global diharapkan bisa sukses pada lingkungan yang sangat terkontrol. *Threshold* dispesifikasikan menggunakan pendekatan *heuristic*, berdasarkan pada pengamatan visual terhadap *the histogram*

2.5.2.3 Optimal Thresholding

Threshold optimal adalah *threshold* yang menghasilkan rata-rata *error* segmentasi minimum. Salah satu metodenya adalah metode *Otsu*. Teknik Otsu memaksimalkan kemiripan (*likehood*) yang akan dapat memisahkan antara objek dengan latar belakangnya.

Gambar 2.12 berikut merupakan contoh perbandingan hasil thresholding yang menggunkana *metode basic* dan *metode otsu*. Gambar kiri merupakan citra hasil *basic thresholding* menggunakan nilai ambang = 100, sedangkan gambar kanan adalah citra hasil *otsu thresholding* dimana dengan metode *otsu* diperoleh nilai ambang optimal = 127.

$$\boldsymbol{p(i) = \frac{n_i}{N}, p(i) \geq 0, \sum_1^{256} p(i) = 1} \qquad \textbf{(2.3)}$$



Gambar 2.12 perbandingan hasil antara *basic thresholding* dan *otsu thresholding*

2.5.2.4 Basic Adapter Thressholding

Faktor *imaging* seperti pencahayaan yang tidak merata, menyebabkan histogram tidak bisa dipartisi dengan baik, sehingga citra tidak bisa disegmentasi

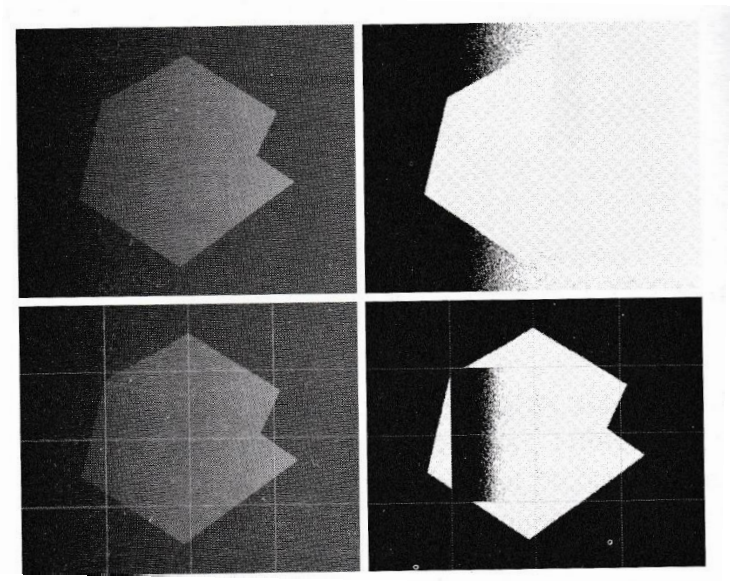
dengan menggunakan sehuah *threshold global*. Untuk mengatasi hal ini, citra asal dibagi menjadi beberapa subcitra. Pada setiap subcitra, segmentasi dilakukan dengan menggunakan *threshold* yang berbeda.

Yang menjadi isu penting adalah bagaimana cara membagi citra dan bagaimana mengestimasi *threshold* untuk Setiap subcitra. Karena *threshold* yang digunakan untuk setiap piksel tergantung pada lokasi piksel, maka *thresholding* disebut adaptif. Semua subcitra yang tidak berisi *boundary* antara objek dan *background* memiliki variance kurang dari 75. Semua subcitra yang berisi *boundary* memiliki *variance* melebihi 100.

Tiap subcitra dengan *variance* lebih besar dari 100 disegmentasi dengan *threshold* yang dihitung pada subcitra tersebut dengan algoritma iteratif yang telah dibahas. Nilai awal untuk T adalah titik tengah antara tingkat keabuan maksimum dan minimum dalam subcitra. Semua subcitra dengan *variance* kurang dari 100 diperlakukan sebagai citra komposit, yang disegmentasi menggunakan *threshold* tunggal dan diestimasi menggunakan algoritma yang sama.

2.5.2.5 Optimal dan Adaptif Thresholding

Thresholding optimal adalah *threshold* yang menghasilkan rata-rata *error* segmentasi minimum. Permasalahan pada gambar dibawah ini adalaha menemukan *boundary* dari rongga dan jantung secara otomatis pada *cardioangiograms* (citra sinar X yang telah di injeksi dengan medium kontras).



Gambar 2.13 (a) Citra asli, (b) hasil *global thresholding*, (c) citra dibagi dalam beberapa subcitra, (d) hasil *adaptive thresholding*

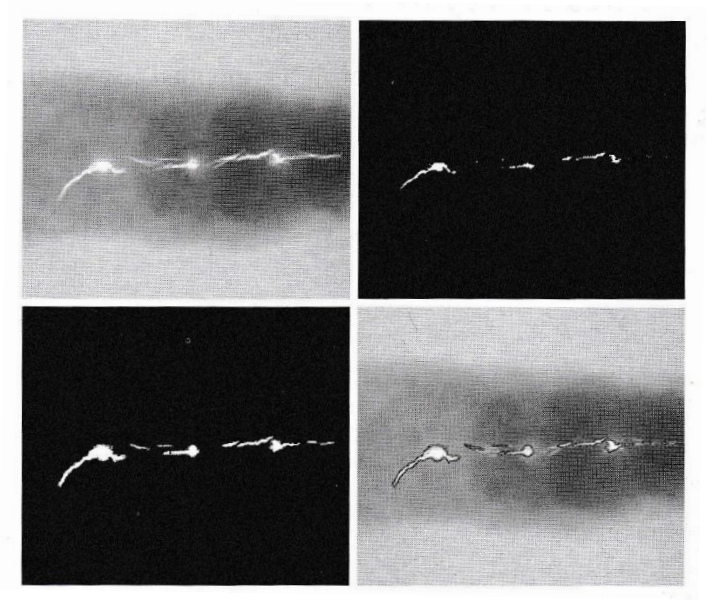
Untuk menghitung *threshold optimal*, citra dibagi menjadi 49 region dengan menempatkan *grid* berukuran 7 x 7 dengan overlap masing-masing citra sebesar 50% (ukuran citra 256 x 256 piksel). Masing-masing dari 49 *region* yang *overlap* berisi 64 x 64 piksel. Setelah 49 histogram dihitung, dilakukan tes *bimodality* untuk menghilangkan histogram-histogram yang *unimodal*.

2.5.3 Region Based Segmentation

2.5.3.1 Region Growing

Region growing merupakan sebuah prosedur dimana sekumpulan piksel atau *subregion* hingga *region* yang lebih besar berdasarkan kriteria yang sudah didefinisikan pendekatan dasar misalnya sekumpulan titik awal dan dari sini *region* tumbuh dengan menambahkan pada masing-masing benih ke piksel tetangga yang memiliki properti yang sama dengan benih tersebut. Deskriptor dapat mendapatkan hasil yang salah jika keterhubungan dan informasi piksel yang bersebelahan tidak digunakan dalam proses *growing region*. Contohnya visualisasi sebuah susunan random piksel dengan hanya tiga nilai *gray-level* yang jelas. Mengelompokan piksel dengan *level-gray* yang sama pada sebuah region yang tanpa konektivitas

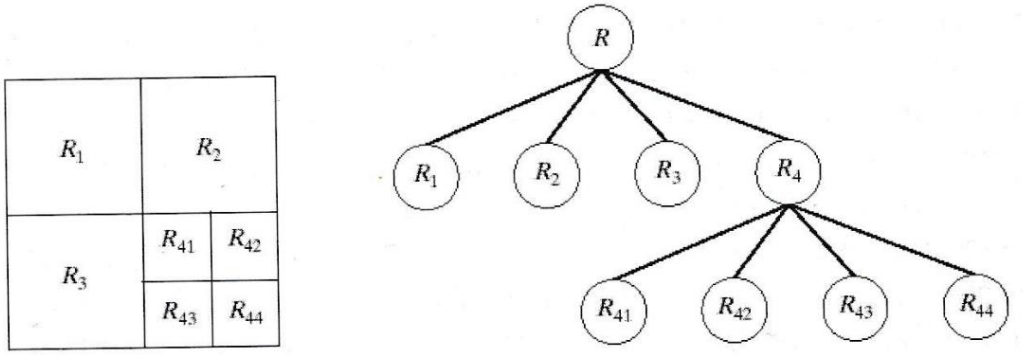
yang akna menghasilkan hasil segmentasi menjadi tidak ada artinya. Permasalahan multimodal histogram baik jika menggunakan pendekatan dasar *region*.



Gambar 2.14 (a) gambar asli, (b) titik-titik *seed*, (c) hasil *region growing*, (d) *baoundary* dari *segmented defective welds* (warna hitam)

2.5.3.2 Region Splitting and Merging

Teknik yang bertujuan untuk membagi sebuah gambar ke dalam bagian-bagian secara acak, *region* yang tidak berhubungan satu sama lain, lalu merg dan/atau *split region* sehingga memenuhi kondisi yang ditentukan.



Gambar 2.15 Proses *splitting region* dalam citra

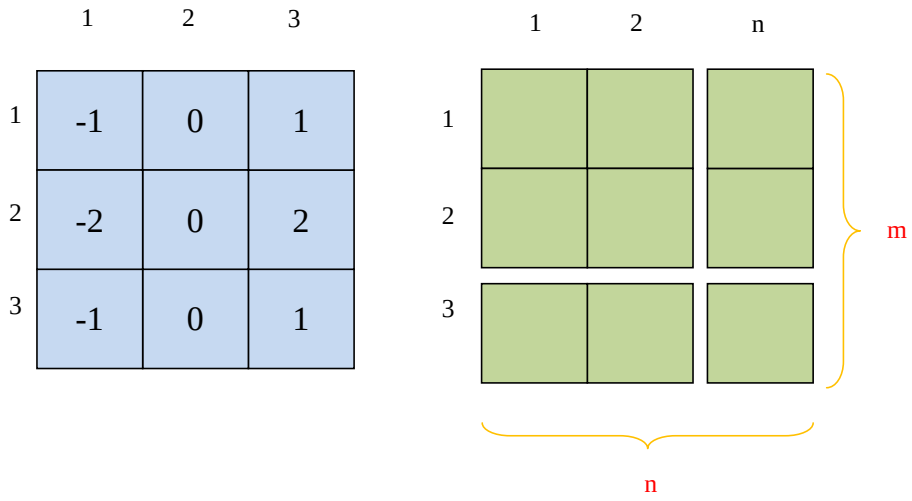
Prosedurnya :

1. Split menjadi empat *disjoint quadrants* pada region R_i di mana P(R_i) = FALSE.
2. *Merge region* yang bersebelahan R_j dan R_k dimana P(R_iU R_k) = TRUE.
3. *Stop* ketika tidak mungkin ada lagi *merge* dan *split*

Properti yang berdasarkan mean dan standard daviasi pada piksel sebuah bidang untuk menentukan teksture dari sebuah region. Konsep *texture segmentation* berdasarkan kegunaan ikuran tekstur untuk predikat P(R_i).

2.5.4 Pengertian Konvolusi

Konvolusi seringkali dilibatkan dalam operasi ketetanggaan piksel. Konvolusi pada citra sering disebut sebagai konvolusi dua-dimensi (konvolusi 2D). Konvolusi 2D didefinisikan sebagai proses untuk memperoleh suatu piksel didasarkan pada nilai piksel itu sendiri dan tetangganya, dengan melibatkan suatu matriks yang disebut kernel yang merepresentasikan pembobotan. Wujud kernel umumnya bujur sangkar, tetapi dapat pula berbentuk persegi panjang. Gambar 2.16 menunjukkan contoh kernel untuk konvolusi [3].



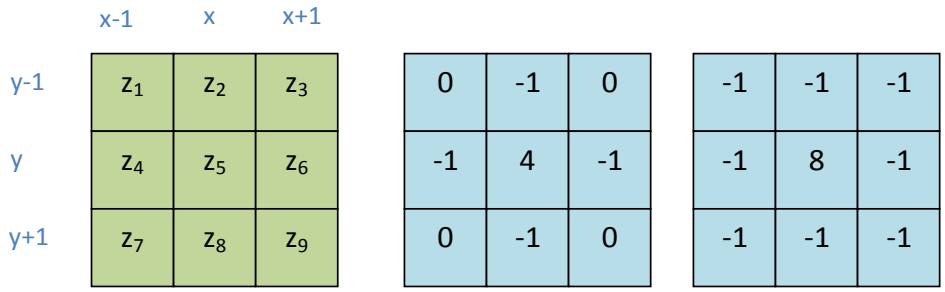
Gambar 2.16 Contoh kernel untuk konvolusi berukuran 3 x 3 dan m x n

Secara umum, proses penapisan di kawasan ruang (*space domain*), sebagai alternatif di kawasan frekuensi, dilaksanakan melalui operasi konvolusi. Operasi ini dilakukan dengan menumpangkan suatu jendela (kernel) yang berisi angka-angka

pengali pada setiap piksel yang ditimpali. Kemudian, nilai rerata diambil dari hasil-hasil kali tersebut. Khusus bila angka-angka pengali tersebut semua adalah 1, hasil yang didapat sama saja dengan filter pererataan.

2.5.5 Pengertian Laplacian

Laplacian merupakan contoh operator yang berdasarkan pada turunan kedua. Operator ini bersifat *omnidirectional*, yakni menebalkan bagian tepi ke segala arah. Namun, operator *Laplacian* memiliki kelemahan, yakni peka terhadap derau, memberikan ketebalan ganda, dan tidak mampu mendeteksi arah tepi (Gonzalez & Woods, 2002).



(a) Posisi pada citra f (b) #1 (c) #2

Gambar 2.17 Operator *Laplacian*

Berdasarkan cadar #1 pada Gambar 2.17, nilai operator *Laplacian* pada (y, x) didefinisikan sebagai :

$$l(y,x) = 4\,f(y,x) - [f(y-1,x) + f(y,x-1) + f(y,x+1) + f(y+1,x)] \quad \textbf{(2.4)}$$

Deteksi tepi orde kedua yang makin kurang sensitif terhadap derau adalah *Laplacian of Gaussian* (LoG). Hal ini disebabkan penggunaan fungsi *Gaussian* yang memuluskan citra dan berdampak pada pengurangan derau pada citra. Akibatnya, operator mereduksi jumlah tepi yang salah terdeteksi (Crane, 1997).

Menurut Fisher, dkk. (2003), operator LoG diperoleh melalui konvolusi dengan

$$\boldsymbol{LoG(y,x) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{\frac{-(x^2+y^2)}{2\sigma^2}}} \quad \textbf{(2.5)}$$

Fungsi di atas disebut sebagai filter topi Meksiko (the *Mexican hat filter*) karena bentuknya seperti topi yang biasa dikenakan orang Meksiko (lihat Gambar 10.20). Dalam hal ini, semakin besar nilai σ, semakin besar pula cadar yang diperlukan. Contoh cadar berukuran 5 x 5 yang mewakili operator LoG ditunjukkan pada Gambar 10.21 (Gonzalez & Woods, 2002).

Dalam praktik, terdapat berbagai variasi rumus yang digunakan dalam LoG. Gonzalez dan Woods (2002) menggunakan persamaan berikut:

$$\boldsymbol{LoG(y,x) = -\left[\frac{x^2+y^2-\sigma^2}{\sigma^4}\right] e^{\frac{-(x^2+y^2)}{2\sigma^2}}} \quad \textbf{(2.6)}$$

Nixon dan Aguido (2002) menggunakan persamaan:

$$\boldsymbol{LoG(y,x) = \frac{1}{\sigma^2}\left[\frac{x^2+y^2}{\sigma^2} - 2\right] e^{\frac{-(x^2+y^2)}{2\sigma^2}}} \quad \textbf{(2.7)}$$

Adapun Crane (1997) menggunakan persamaan:

$$\boldsymbol{LoG(y,x) = \frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{\frac{-(x^2+y^2)}{2\sigma^2}}} \quad \textbf{(2.8)}$$

2.5.6 Marr-Hilderth

Zero-crossing atau dikenal dengan nama lain yaitu operator *Marr-Hildreth* sebenarnya adalah operator *LoG* yang dilengkapi dengan upaya untuk menemukan *zero crossing*. *Zero-crossing* menyatakan tanda perubahan pada tepi-tepi dalam citra. Operasi inilah yang membuat operator *Marr-Hildreth* mampu menghasilkan kurva yang tertutup, yang tidak dapat dipenuhi oleh operator lain.

Ada beberapa teknik yang dapat digunakan untuk melakukan pemrosesan *zero-crossing*, misalnya seperti yang dibahas pada Crane (1997) atau pada Nixon dan Aguido (2002). Algoritma yang didasarkan pada Nixon dan Aguido dapat dilihat pada subban selannjutnya. Persamaan yang digunakan untuk menentukan nilai tepi dari setiap citra adalah sebagai berikut :

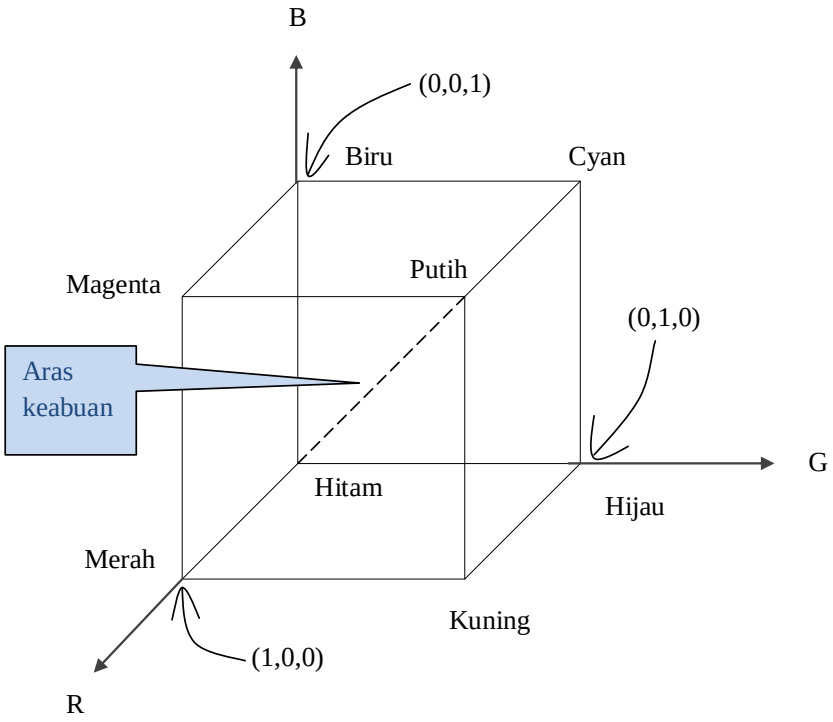
$$\boldsymbol{\frac{1}{\sqrt{2\pi(\sigma)^2}}\left(2 - \frac{(x^2)+(y^2)}{\sigma^2}\right) e^{\frac{(x^2)+(y^2)}{2(\sigma)^2}}} \quad \textbf{(2.9)}$$

Marr-Hildert Operator (Marr and Hildert, 1980) merupakan salah satu bentuk operator yang menggunakan turunan kedua. Tanda turunan kedua digunakan untuk menentukan apakah tepi tersebut ada di sisi gelap atau disisi terang dari suatu tepi. Jika negatif, berarti ada di sisi terang. Dan jika positif maka titik tersebut ada di sisi gelap. Catatan tentang sifat tambahan dari turunan kedua disekitar tepi :

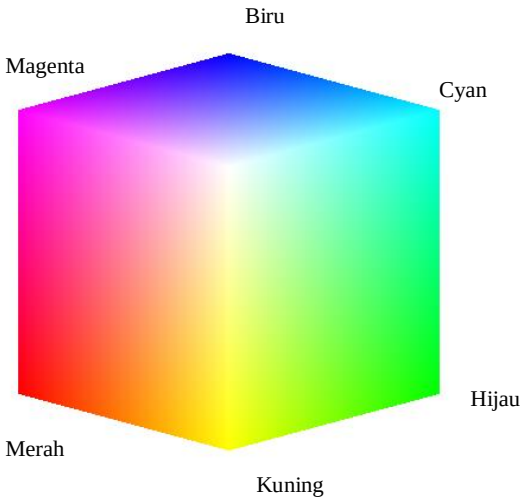
1. Turunan kedua menghasilkan dua nilai untuk tiap tepi dari sebuah citra.
2. Dapat dibayangkan bahwa sebuah garis lurus menggabungkan nilai turunan kedua ekstrem positif dan negatif yang akan memotong nol di titik tengah tepinya. Sifat *Marr-Hilderth* dari turunan kedua berguna untuk menentukan lokasi pusat dari tepi yang tebal.

2.5.7 Ruang Warna RGB

Ruang warna RGB biasa diterapkan pada monitor CRT dan kebanyakan sistem grafika komputer. Ruang warna ini menggunakan tiga komponen dasar yaitu merah (R), hijau (G), dan biru (B). Setiap piksel dibentuk oleh ketiga komponen tersebut. Model RGB biasa disajikan dalam bentuk kubus tiga dimensi, dengan warna merah, hijau, dan biru berada pada pojok sumbu (Gambar 2.18). Warna hitam berada pada titik asal dan warna putih berada di ujung kubus yang berseberangan. Gambar 2.19 memperlihatkan kubus warna secara nyata dengan resolusi 24 bit. Perlu diketahui, dengan menggunakan 24 bit, jumlah warna mencapai 16.777.216.



Gambar 2.18 Skema ruang warna RGB dalam bentuk kubus



Gambar 2.19 Kubus warna dengan 24 bit

RGB biasa digunakan karena kemudahan dalam perancangan *hardware*, tetapi sebenarnya tidak ideal untuk beberapa aplikasi. Mengingat warna merah, hijau, dan biru sesungguhnya terkorelasi erat, sangat sulit untuk beberapa algoritma pemrosesan citra (Crane, 1997). Sebagai contoh, kebutuhan untuk memperoleh

warna alamiah seperti merah dengan menggunakan RGB menjadi sangat kompleks mengingat komponen R dapat berpasangan dengan G dan B, dengan nilai berapa saja. Hal ini menjadi mudah jika menggunakan ruang warna HLS ataupun HSV.

2.5.8 Warna Keabuan (*Grayscale*)

Dalam menentukan nilai *grayscale* pada objek terlebih dahulu mencari nilai RGB (*Red Green Blue*), setelah itu nilai RGB tersebut akan dicari nilai rata-rata dengan rumus sebagai berikut :

$$Gray(i,j) = \left(\frac{(f^{RED}(i,j) + f^{GREEN}(i,j) + f^{BLUE}(i,j))}{3} \right) \quad \textbf{(2.10)}$$

Dimana :

f^{RED} : intensitas warna Red

f^{GREEN} : intensitas warna Green

f^{BLUE} : intensitas warna Blue

Untuk mencari nilai RGB pada nilai citra keluaran dapat menggunakan rumus sebagai berikut :

Nilai Red = piksel mod 256

Nilai Green = (piksel / 256) mod 256

Nilai Blue = (piksel/256/256) mod 256

2.5.9 Algoritma dan *Pseudocode*

Untuk mengetahui solusi dalam penerapan metode yang digunakan dapat dilihat dalam analisis algoritma dan *pseudocode*. Sebenarnya implementasi analisis algoritma untuk sementara dapat diketahui dengan menggunakan aplikasi yang sudah ada dengan menggunakan NetLab, sebelum melangkah lebih jauh dapat kita ketahui beberapa fungsi, variabel dan prosedur apa saja yang dibutuhkan dalam algoritma untuk memecahkan masalah.

Untuk analisis algoritma dan *pseudocode* dapat kita lihat beberapa fungsi dan prosedur berikut ini :

1. Algoritma untuk menentukan nilai konvolusi pada citra dengan asumsi kernel memiliki baris dan kolom yang ganjil.

ALGORITMA untuk konvolusi pada citra dengan mengabaikan bagian tepi

Masukan:

- f : Citra yang akan dikonvolusi
- h : kernel konvolusi

Keluaran:

- g : Citra hasil konvolusi

```
1. m2 ← floor(jumlah_baris_kernel h)
2. n2 ← floor(jumlah_lebar_kernel h)
3. FOR y ← m2+1 TO tinggi_citra_f – m2
       FOR x ←n2+1 TO lebar_citra_f – n2
           // Lakukan konvolusi
           jum ← 0;
           FOR p ← -m2 TO m2
               FOR q ← -n2 TO n2
                   jum ← jum * h(p+m2+1, q+n2+1) * f(y-p, x-p)
               END-FOR
           END-FOR
           g2(y, x) ← jum
       END-FOR
   END-FOR
4. // Salin posisi g2 ke g dengan membuang yang tidak dikonvolusi
5. FOR y ← m2+1 TO tinggi_citra_f – m2
       FOR x ← n2+1 TO lebar_citra_f – n2
           g(y-m2, x-n2) ← g2(y, x)
       END-FOR
   END-FOR
```

Berdsarkan algoritma diatas, citra hasil akan kehilangan sebesar

a. 2*m2 baris atau sama dengan jumlah kernel dikurang 1
b. 2*n2 kolom atau sama dengan jumlah kernel dikurang 1

2. Algoritma untuk menentukan nilai cadar (mask) *Laplacian of Gaussian* (LoG).

ALGORITMA untuk memperoleh mask *LoG*

Masukan:

- Ukuran : ukuran mask
- σ: Standar deviasi

Keluaran:

- h (M, N): Cadar LoG

maskLoG(ukuran, σ):

1. cx ← floor((ukuran-1) / 2)
2. cy ← cx
3. jum ← 0
4. FOR y ← 0 TO ukuran -1
   FOR x ← 0 TO ukuran -1
   nx ← x – cx
   ny ← y – cy
   nilai ← $\frac{1}{\sigma^2}\left[\frac{nx^2+ny^2}{\sigma^2}-2\right]e^{\frac{-(nx^2+ny^2)}{2\sigma^2}}$
   h(y+1, x+1) ← nilai
   jum ← jum + nilai
   END-FOR
   END-FOR
5. // Lakukan normalisasi
   FOR y ← 1 TO ukuran
   FOR y ← 1 TO ukuran
   h(y, x) ← h(y, x) / jum
   END-FOR
   END-FOR

RETURN h

Algoritmai di atas, normalisasi dilakukan agar jumlah koefesien pada masking bernilai 1

3. Algoritma untuk menentukan nilai Marr-Hilderth

ALGORITMA untuk memproses *zero crossing*

Masukan:
- g: Citra hasil pemrosesan *LoG*, berukuran MxN
- ukuran : Ukuran mask
- σ: Standar deviasi

Keluaran:
- k: citra hasil pemrosesan operator *Marr-Hildreth*

zerocross(*g, ukuran, σ*) :

```
1. // Bentuk  larik k yang seluruhnya bernilai 0
   FOR y ← 1 TO M
      FOR x ← 1 TO N
         k(y, x) ← 0
      END
   END

2. FOR baris ← 2 TO M-1
      FOR kolom ← 2 TO N-1
         bag1 ← rerata(g(x-1, y-1), g(x, y), g(x-1, y-1), g(x, y))
         bag2 ← rerata(g(x, y-1), g(x+1, y), g(x, y-1), g(x+1, y))
         bag3 ← rerata(g(x, y-1), g(x+1, y), g(x+1, y-1), g(x+1, y))
         bag4 ← rerata(g(x, y), g(x+1, y+1), g(x, y), g(x+1, y+1))

         terbesar ← max(bag)
         terkecil ← min(bag)

         IF (terbesar > 0) AND (terkecil < 0)
            k(y,x) ← 255
         END-IF
      END-FOR
   END-FOR

RETURN k
```

2.6 Java Media Framework (JMF)

Java Media Framework API adalah *library* yang digunakan untuk keperluan media baik media suara, kamera maupun animasi 3D, selain itu juga JMF API merupakan arsitektur yang menggabungkan protokol dan *interface* dalam

pemrograman java yang merekam, mentransmisi dan *playback* media. JMF 2.1.1 merupakan versi terakhir yang dikembangkan oleh Sun Microsystem, Sun's sebagai perusahaan pengembang pemrograman java berinisiatif untuk membawa pemrosesan *time-base* media ke dalam pemrograman java. *Time-base* media ialah alat untuk mengubah data yang diterima berdasarkan waktu dan mengubah *content-type*.

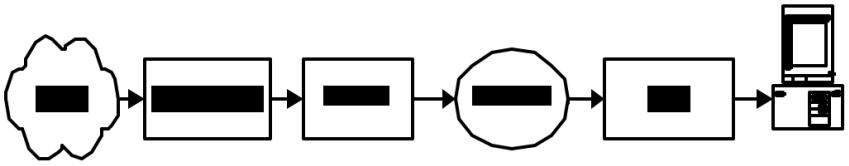
JMF API mendukung pembangunan dan pengembangan berbagai aplikasi yang bersifat multimedia, format yang di tawarkan diantaranya ialah AU, AVI, MIDI, MPEG, WAV, dan beberapa *file audio* yang didukung oleh Java. Media yang diambil juga dapat ditransmisikan ke dalam jaringan internet sebagai media *streaming* yang proses pengirimannya secara *realtime*. Arsitektur jaringan yang digunakan dalam pemrosesan media *streaming* yaitu menggunakan *Real-time Transfer Protokol*, dan *Real-time Transport Control Protokol* sebagai pendamping yang berfungsi untuk mengontrol komunikasi secara berkala pada paket transmisi yang menggunakan mekanisme distribusi paket data agar bisa digunakan oleh *receivers* untuk sinkronisasi audio dan video [3].

2.6.1 Real Time Protocol (RTP)

RTP adalah protokol yang *header format* dan kontrolnya didesain untuk mendukung aplikasi-aplikasi transmisi data *real-time* seperti audio, video, dan juga simulasi data melalui layanan jaringan. Pada TCP/IP terdapat dua protokol *transport*, yaitu: *Transmission Control Protocol* (TCP) dan *User Datagram protocol* (UDP). Pada TCP pemrograman yang berorientasi pada konektivitas (*connection oriented programming*). Protokol ini memiliki arsitektur yang standar, terbuka dan tidak bergantung pada perangkat keras atau sistem operasi. Dalam skema pengalamatannya yang umum bagi setiap *device* yang terhubung dengan jaringan menjamin bahwa semua data, dalam bentuk paket data yang dikirim oleh *server* akan diterima oleh *client.* Sedangkan UDP merupakan protokol yang bersifat *connectionless oriented*, Artinya, saat melakukan pengiriman data tidak dilakukan proses *handshaking*, tidak ada *sequencing datagram*, dan tidak ada jaminan bahwa paket data (*datagram*) yang dikirim akan tiba dengan selamat.

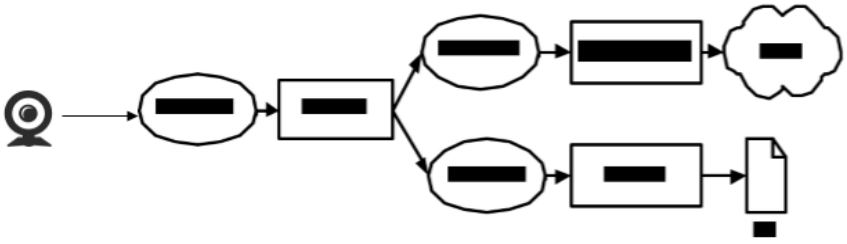
UDP sangat cocok untuk digunakan pada aplikasi yang membutuhkan *query* dan respon cepat, maka dari itu UDP sangat berguna dalam mentransmisikan file audio/video dari *server* ke *client* seperti VoIP, audio/video *streaming*. UDP sangat tidak cocok digunakan untuk mengirimkan paket data berukuran besar, karena dapat memperbesar peluang jumlah paket loss atau hilang maupun rusak.

JMF dapat mentransmisikan dan playback RTP Stream dengan API yang terdapat pada `javax.media.rtp`, `javax.media.rtp.event`, dan `javax.media.rtp.rtcp`. pada RTP *receiver/client*, dapat dilakukan *playback* atau menerima media data yang dikirimkan oleh RTP *transmeiter/server*. Seperti yang terlihat pada Gambar 2.20 diagram alur proses RTP pada bagian *receiver* yang menerima media data dari jaringan dan kemudian mempresentasikannya dengan *player*.



Gambar 2.20 RTP *Receiver*

Proses transmisi media data melalui tahapan-tahapan yang memiliki fungsi sendiri. RTP pengirim data memiliki beberapa tahapan yang dapat dilihat pada Gambar 2.21. pada RTP transmisi, input yang diperoleh data *source* berupa *capture device webcam*. Kemudian *processor* akan memprosesnya menjadi *data source* yang digunakan untuk RTP atau datasink.



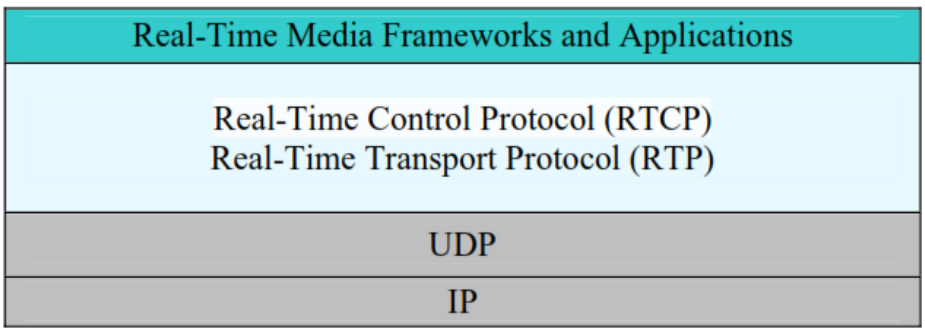
Gambar 2.21 RTP Transmisi

2.6.2 Real Time Control Protocol (RTCP)

RTCP merupakan bagian dari RTP, yang berfungsi sebagai pendamping RTP untuk mengontrol kominikasi secara berkala pada paket transmisi yang menggunakan mekanisme distribusi paket data agar bisa digunakan oleh *receivers* untuk sinkronisasi audio dan video. RTCP akan mengirimkan paket secara berkala kepada *participants* yang teridentifikasi mengenai data yang dikirim dan yang diterima. Beberapa tipe paket RTCP tersebut adalah sebgai berikut :

1. *Sender Report* ialah informasi yang akan diperoleh jika *participants* telah mengirim data paket RTP setelah laporan yang terakhir, bersamaan saat itu *receiver report* juga diinformasikan.
2. *Receiver Report* ialah sama seperti halnya *sender report*, hanya saja dilakukan pada sisi penerima.
3. *Source Deserription* ialah yang akan menunjukan identitas participants. Pada SDES, *Canonical Name* merupakan *item* yang hasur dikirimkan pada setiap paket RTCP, karena mengandung informasi identitas serta lokasi sumber pengirim.
4. *Stop* (*end*) ialah paket yang akan memberikan informasi participants yang tidak aktif atau yang akan meninggalkan session. Ketika *participants* meninggalkan session maka paket *stop* akan terkirim.

Pada Gambar 2.22 dapat dilihat arsitektur jaringan komunikasi RTP yang terdapat pada *liberary* JMF dalam mentransmisikan data maupun pengiriman data melalui RTP.



Gambar 2.22 Arsitektur RTP

2.7 Google Cloud Messagging (GCM)

Google Cloud Messaging merupakan layanan yang disediakan oleh google yang memungkinkan untuk mengirim data dari *server* ke *client* (Android-*powered*) perngkat *mobile*, dan menerima pesan dari perangkat yang terhubung dengan server yang sama. Layanan GCM menangani semua aspek antrian pesan dan pengiriman ke target aplikasi Android yang berjalan pada perangkat target. GCM disediakan oleh google dengan cuma-cuma untuk *devloper* Android dan dalam penyimpanannya pun disediakan quota dalam jumlah yang besar oleh google untuk kebutuhan yang lebih luas.

Google Cloud Messaging yang disediakan hanya untuk *platform* Android sebagai layanan gratis yang membantu pengembang dalam mengirim dan menerima data dari *server* ke aplikasi Android yang akan dibangun. Kinerja yang ditawarkan layanan GCM ini ialah menyampaikan informasi baru yang masuk ke penyimpanan *cloud*, seperti halnya ketika e-mail mendapatkan pesan baru yang masuk ke *mailbox* [4].

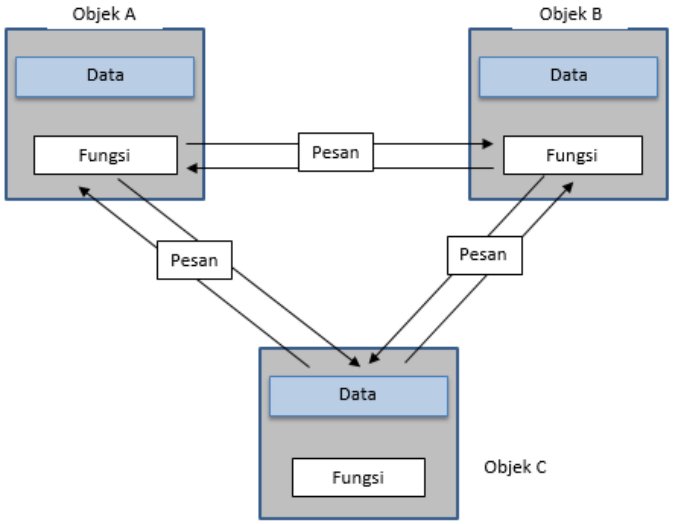
2.8 Pemodelan Berorientasi Objek

Metodologi Berorientasi Objek atau *object oriented* merupakan suatu strategi pembangunan atau paradigma baru dalam rekayasa perangkat lunak yang memandang sistem sebagai kumpulan objek-objek diskrit yang saling berinteraksi. Yang dimaksud berorientasi objek adalah bahwa mengorganisasikan perangkat lunak sebagai kumpulan obyek-obyek diskrit yang bekerja sama antara informasi atau struktur data dan perilaku (*behavior*) yang mengaturnya. Sebuah sistem yang dibangun dengan berdasarkan metode berorientasi objek adalah sebuah sistem yang komponennya dibungkus (*encapsulation*) menjadi kelompok data dan fungsi. Setiap komponen dalam sistem tersebut dapat mewarisi atribut dan sifat dan komponen lainnya serta dapat berinteraksi satu sama lainnya [5].

2.8.1 Konsep Dasar Pemodelan Berorientasi Objek

Pemrograman Berorientasi Objek adalah suatu cara baru dalam berfikir dan belogika dalam menghadapi masalah-masalah dengan bantuan komputer. Konsep

objek dalam OOP adalah konsep atau abstraksi tentang sesuatu yang memiliki arti bagi aplikasi yang akan dikembangkan. Objek biasanya adalah kata benda, namun objek dalam konteks OOP bukan hanya objek nyata yang bias diraba dan dilihat secara asat mata seperti benda yang biasa kita temukan. Dibawah ini merupakan gambar pengorganisasian data dan fungsi pada pendekatan berorientasi objek.



Gambar 2.23 Pengorganisasian Data dan Fungsi pada Pendekatan Berorientasi Objek

2.8.2 Metode Analisis Berorientasi Objek (OOA)

Untuk melakukan analisis sistem terdapat beberapa macam pendekatan, diantaranya pendekatan konvensional dan pendekatan berorientasi objek. Pendekatan konvensional terutama mengacu kepada strategi dekomposisi yang berdasar algoritma atau fungsional. Pendekatan ini telah berkembang meliputi seluruh tahap atau aktifitas proses rekayasa perangkat lunak dari mulai pemrograman dengan iterasi perbaikan, pemrograman terstruktur, ditambah dengan perancangan terstruktur kemudian analisis terstruktur dan sebagainya. Sedangkan pendekatan berorientasi objek memusatkan pada rancangan pada objek dan antar muka yang dihasilkan. Objek adalah entiti yang berisi data atau variabel dan tingkah laku. Data atau variabel yang menggambarkan sifat atau keadaan objek dalam dunia nyata (*real world*) didefiniskan sebagai attribute,

sedangkan tingkah laku yang menggambarkan aksi-aksi yang dimiliki objek didefinisikan sebagai *method*.

Unified Approach (UA) merupakan metode analisis berorientasi objek dari Ali Bahrami (1999). UA adalah suatu metodologi pengembangan sistem berbasis objek yang menggabungkan proses dan metodologi yang telah ada sebelumnya dan menggunakan UML sebagai standar pemodelannya. Proses dan tahapan yang ada dalam UA merupakan proses-proses terbaik yang diambil dari metode objek yang telah diperkenalkan oleh Booch, Rumbaugh, dan Jacobson. Tahap Analisis dalam UA ditujukan untuk mengidentifikasi kelas-kelas yang terdapat dalam sistem.

2.8.3 Pemodelan UML (Unified modelling Language)

UML (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi obyek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan system untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modelling Technique* (OMT) dan *Object Oriented Software Engeenering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented*. Metode ini menjadikan proses analisis dan design ke dalam empat tahapan iteratif yaitu identifikasi kelas-kelas dan objek-objek, identifikasi *semantic* dari hubungan objek dan kelas tersebut, perincian *interface* dan implementasi. Keunggulan metode Booch adalah pada detil dan kayanya dengan notasi dan elemen. Pemodelan OMT yang dikembangkan oleh Rumbaugh didsarkan pada analisis tersetruktur dan pemodelan *entitiy-Relationship*.

Setiap sistem yang kompleks seharusnya bisa dipandang dari sudut yang berbeda-beda sehingga kita bisa mendapatkan pemahaman secara menyeluruh. Untuk upaya tersebut UML menyediakan 9 jenis diagram yang dapat

dikelompokkan berdasarkan sifatnya statis atau dinamis. Dari 9 jenis diagram dalam UML itu diantaranya [7]:

1. ***Use Case Diagrams***, Bersifat statis. Diagram ini memperlihatkan himpunan use case dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku dalam suatu sistem yang dibutuhkan serta diharapkan pengguna.
2. ***Activity Diagram***, Bersifat dinamis. Diagram aktifitas ini adalah tipe khusus dari diagram state yang memperlihatkan aliran dari aktifitas ke aktifitas yang lainnya dalam suatu sistem. Diagram ini sangat penting terutama dalam pemodelan fungsi-fungsi dalam suatu sistem dan memberi tekanan pada aliran kendali antara objek.
3. ***Sequance Diagram***, Bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan (*message*) dalam waktu tertentu.
4. ***State Chart Diagrams***, Bersifat dinamis. *Diagram state* ini memperlihatkan state-state pada sistem, memuat state, transisi, event, serta aktifitas. Diagram ini terutama sangat penting pada pemodelan sistem-sistem reaktif.
5. **Diagram Kelas,** Bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umum dijumpai pada pemodelan sistem berorientasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas aktif.
6. **Diagram Objek,** Bersifat statis. Diagram ini memperlihatkan objek-objek serta relasi-relasi antar objek. Diagram objek memperlihatkan instansiasi statis dari segala sesuatu yang dijumpai pada diagram kelas.
7. ***Collaboration Diagram***, Bersifat dinamis. Diagram kolaborasi adalah diagram interaksi yang menekankan organisasi struktural dari objek-objek yang menerima serta mengirim pesan (*message*).
8. ***Component Diagram***, Bersifat statis. Diagram komponen ini memperlihatkan organisasi serta kebergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini

berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan ke dalam satu atau lebih kelas-kelas, antarmuka-antarmuka (*interfaces*), serta kolaborasi-kolaborasi.

9. **Deployment Diagram**, Bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (saat *run-time*). Diagram ini memuat simpul-simpul (*node*) beserta komponen-komponen yang ada di dalamnya. *Deployment diagram* berhubungan erat dengan diagram komponen dimana *deployment diagram* memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*).

2.9 Mobile Platform Android



Gambar 2.24 Logo Android

Android merupakan sistem operasi terbaru dari google yang berbasis kernel linux. Banyak kalangan umu yang sudah tidak asing lagi dengan sistem operasi ini, baik dari kalangan *low level* maupun *high level*. Sistem operasi ini memberikan fitur gratis yang disediakan oleh google yang dapat digunakan maupun di kembangkan oleh banyak user maupun devloper Android SDK.

Android SDK adalah tools API (*Aplication Programming Interface*) yang digunakan untuk mulai mengembangkan aplikasi pada *platform* android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di release oleh Google. Saat ini disediakan android SDK (*Software Development Kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi

pada platform android menggunakan bahasa pemrograman Java. Sebagai platform aplikasi netral, android memberi anda kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan hanphone/smartphone. Beberapa fitur android yang paling penting adalah [7]:

1. *Framework* aplikasi yang mendukung penggantian komponen dan *reusable.*
2. *Mesin Virtual Dalvik* dioptimalkan untuk perangkat mobile.
3. *Integrated browser* berdasarkan *engine open source webkit*.
4. Grafis yang dioptimalkan dan didukung oleh *libraries grafis 2D, grafis 3D* berdasarkan spesifikasi openGL ES 1.0 (Opsional Akselerasi *Hardware*).
5. SQLite untuk penyimpanan data.
6. *Media Support* yang mendukung *audio, video* dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), *GSM Telephony* (tergantung *hardware*).
7. Bluetooth, EDGE, 3G, WiFi (tergantung *hardware*).
8. Kamera, GPS, kompas dan *Accelerator* (tergantung *hardware*).
9. Lingkungan *development* yang lengkap dan kaya, termasuk perangkat *emulator*, *tools* untuk *debuging*, profil dan kinerja memori, dan *plug in* untuk IDE Eclipse.