BAB 2

TINJAUAN PUSTAKA

2.1 Mata

Mata adalah indra terbaik yang dimiliki oleh manusia sehingga citra (gambar) memegang peranan penting dalam perspektif manusia. Namun mata manusia memeliki keterbatasan dalam menangkap sinyal elektromagnetik [7].

Mengedipkan mata adalah suatu proses penutupan dan pembukaan dari kelopak mata dalam waktu yang cepat. Proses tersebut merupakan suatu fungsi yang penting untuk memberikan air mata pada bagian-bagian mata dan untuk memindahkan benda yang tidak diinginkan dari permukaan kornea mata dan selaput mata.

2.2 Pengolahan Citra (*Image Processing*)

Pengolahan citra (*image processing*) adalah pengolahan suatu citra (gambar) dengan menggunakan komputer secara khusus, untuk menghasilkan suatu citra yang lain [4].

Sesuai dengan perkembangan komputer itu sendiri, pengolahan citra mempunyai dua tujuan utama, yaitu sebagai berikut:

1. Memperbaiki kualitas citra, dimana citra yang dihasilkan dapat menampilkan informasi secara jelas. Hal ini berarti manusia sebagai pengolah informasi.

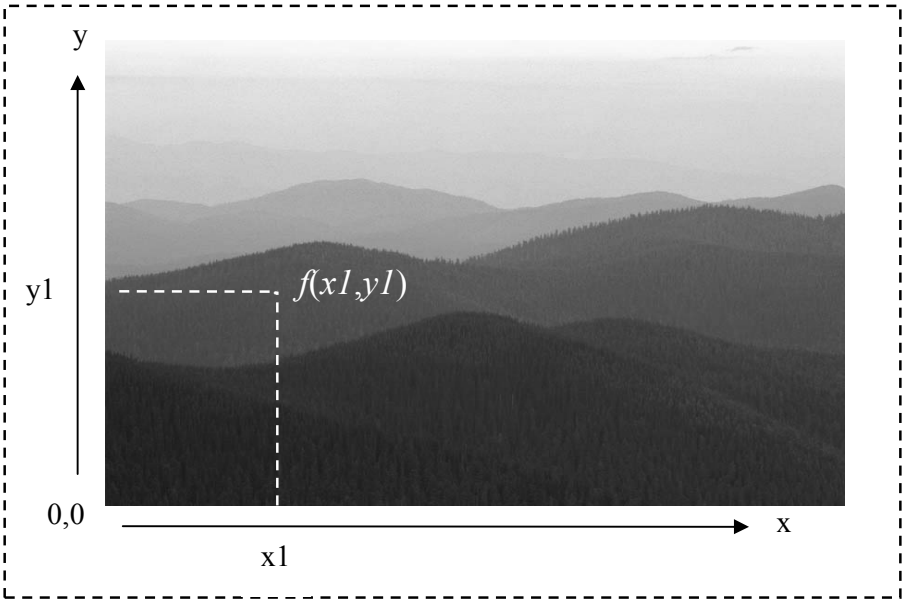
2. Mengekstraksi informasi ciri yang menonjol pada suatu citra, dimana hasilnya adalah informasi citra dimana manusia mendapatkan informasi ciri dari citra secara numerik atau dengan kata lain *computer* (mesin) melakukan interprestasi terhadap informasi yang ada pada citra melalui besaran-besaran data yang dapat dibedakan secara jelas (berupa besaran numerik).

Dalam perkembangan lebih lanjut, *image processing* dan *computer vision* digunakan sebagai pengganti mata manusia dengan perangkat *input image capture* seperti kamera dan scanner dijadikan sebagai mata dan mesin komputer dijadikan sebagai otak yang mengolah informasi. Sehingga muncul beberapa pecahan bidang yang menjadi penting dalam *computer vision* antara lain: *pattern recognition* (pengenalan pola), *biometric* (pengenalan identifikasi manusia berdasarkan ciri-ciri biologis yang tampak pada badan manusia), *content based image and video retrieval* (mendapatkan kembali citra atau video dengan informasi tertentu), *video editing* dan lain-lain[4].

2.2.1 Dasar – Dasar Pengolahan Citra Digital

Citra digital diasumsikan dengan persamaan f(x,y) dimana x menyatakan nomor baris, y menyatakan nomor kolom, dan f menyatakan nilai derajat keabuan dari citra. Sehingga (x,y) adalah posisi dari piksel dan f adalah nilai derajat keabuan pada titik (x,y). Kecerahan setiap citra disimpan dengan cara pemberian nomor pada setiap piksel. Makin tinggi nomor piksel maka makin gelap (hitam) piksel tersebut. Begitu juga sebaliknya makin rendah nilai piksel tersebut maka makin terang. Sistem yang umum memiliki

256 tingkat kecerahan untuk setiap piksel, yang paling terang adalah 255 dan yang paling gelap adalah 0.



Gambar 2. 1 Citra Digital

Citra atau gambar terbagi dalam tiga tipe adalah sebagai berikut: [7]

a. Gambar *Grayscale*

Citra yang terdiri dari satu layer warna dengann derajat keabuan tertentu. Dinyatakan dalam suatu fungsi :

f (x,y) Σ [0…255] **(2.1)**

b. Gambar Biner

Citra yang hanya memiliki dua nilai yaitu 1 dan 0. Dinyatakan dalam suatu fungsi :

f(x,y) Σ {0,1} **(2.2)**

c. Gambar berwarna

Citra yang terdiri dari tiga layer warna yaitu RGB (*Red-Green-Blue*) dimana *R-layer* adalah matrik yang menyatakan derajat kecerahan untuk warna merah, *G-layer* adalah matrik yang menyatakan derajat kecerahan untuk warna hijau, dan *B-layer* adalah matrik yang menyatakan derajat kecerahan untuk warna biru. Representasi dalam citra digital dinyatakan dalam persamaan :

$$\mathbf{f_R(x,y)\ \Sigma\ [0\ldots255]}$$

$$\mathbf{f_G(x,y)\ \Sigma\ [0\ldots255]} \qquad \mathbf{(2.3)}$$

$$\mathbf{f_B(x,y)\ \Sigma\ [0\ldots255]}$$

Proses pengolahan citra digital dengan menggunakan komputer digital adalah terlebih dahulu mentransformasikan citra ke dalam bentuk besaran-besaran diskrit dari nilai tingkat keabuan pada titik-titik elemen citra. Bentuk citra ini disebut citra digital. Elemen-elemen citra digital apabila ditampilkan dalam layar monitor akan menempati sebuah ruang yang disebut dengan piksel (*picture elemen/pixel*) [7].

2.2.2 Model Citra Digital

Citra ada dua macam yaitu citra kontinu dan citra diskrit(citra digital).Citra kontinu dihasilkan sistem optik yang menerima sinyal analog, misalnya mata manusia dan kamera analog. Sedangkan citra diskrit dihasilkan melalui proses digitalisasi sehingga mampu menghasilkan citra diskrit (citra digital) misalnya kamera digital dan *scanner*. Citra merupakan fungsi

kontinu dari intensitas cahaya pada bidang dua dimensi, dinyatakan dalam bentuk fungsi ***f(x ,y)*** dimana :[11]

(x, y) : koordinat pada bidang dua dimensi

f(x, y) : intensitas cahaya (*brightness*) pada titik *(x, y),*

besarnya $0 < f(x,y) < \infty$

Intensitas cahaya, *f*(*x, y*), sebenarnya merupakan hasil kali antara jumlah cahaya yang berasal dari sumbernya (*illumination*) dengan derajat kemampuan obyek memantulkan cahaya (*reflectance*), dinyatakan dalam bentuk persamaan berikut :

$$\mathbf{f(x,y) = i(x,y).r(x,y)} \qquad \mathbf{(2.4)}$$

dimana :

i(*x, y*) : *illumination*, besarnya $0 < i(x, y) < \infty$

r(x, y) : *reflectance*, besarnya $0 < r(x, y) < 1$

Nilai *i*(*x, y*) ditentukan oleh sumber cahaya. Contoh nilai *i*(*x, y*) :

1. Pada hari cerah, matahari menghasilkan $i(x, y) \approx 9000$ *foot candles*
2. Pada hari mendung, matahari menghasilkan $i(x, y) \approx 1000$ *foot candles*
3. Pada malam bulan purnama, sinar bulan menghasilkan $i(x, y) \approx 0.01$ *foot candles*

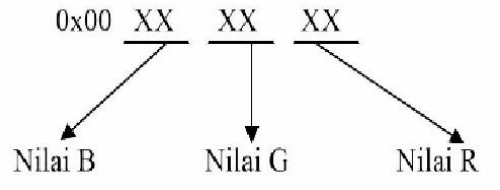
Nilai *r*(*x, y*) ditentukan oleh karakteristik obyek di dalam gambar.

Contoh nilai $r(x, y)$:

1. Benda hitam mempunyai $r(x, y) = 0.01$
2. Dinding putih mempunyai $r(x, y) = 0.8$
3. Benda logam *stainless steel* mempunyai $r(x, y) = 0.65$

2.2.3 Model Warna RGB

Dasar dari pengolahan citra adalah pengolahan warna RGB pada posisi tertentu. Dalam pengolahan citra, warna dipresentasikan dengan nilai hexadesimal dari 0x00000000 sampai 0x00ffffff. Warna hitam adalah 0x00000000 dan warna putih adalah 0x00ffffff. Definisi nilai 8 warna di atas seperti Gambar **2.2**, variabel 0x00 menyatakan angka dibelakangnya adalah hexadesimal.



Gambar 2. 2 Nilai Warna RGB dalam Heksadesimal

Pengkodean warna RGB dapat ditulis dalam angka hexadesimal (basis 16) untuk masing-masing komponen R, G atau B. Misalkan:

1. Untuk Hitam murni akan ditulis dengan kode #000000 (R=00, G=00, B=00)
2. Untuk Putih sempurna akan ditulis dengan kode #FFFFFF (R=FF, G=FF, B=FF)

3. Untuk Biru murni akan ditulis dengan kode #0000FF (R=00, G=00, B=FF)

Model Warna RGB adalah sebuah model warna *additif* dimana pancaran warna r*ed* (merah), *green* (hijau), dan *blue* (biru) ditambahkan bersama dengan cara yang bervariasi untuk mereproduksi susunan warna yang lebar. Warna *additif* digunakan untuk *lighting*, video, dan monitor. Monitor sebagai contoh, menciptakan warna dengan memancarkan cahaya melalui merah, hijau dan biru fosfor.

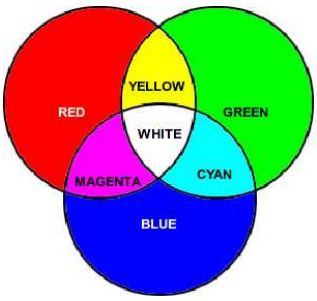
Tujuan utama dari model warna RGB adalah untuk mempresentasikan ulang, dan menampilkan gambar dalam sistem elektronik, misalnya dalam televisi dan komputer. Model warna RGB juga digunakan dalam fotografi konvensional.

RGB adalah sebuah ruang warna yang sifatnya bergantung kepada perangkat. Perangkat yang berbeda akan mendeteksi atau mereproduksi nilai RGB secara berbeda. Untuk membentuk warna dengan RGB, tiga cahaya berwarna balok (satu merah, satu hijau, dan satu biru) harus ditumpangkan (misalnya dengan emisi dari layar hitam, atau dengan refleksi dari layar putih). Masing-masing dari tiga balok disebut sebagai komponen warna, dan masing-masing dapat memiliki intensitas yang berbeda.

RGB sering digunakan dalam perangkat input seperti: TV berwarna dan kamera video, *scanner*, dan kamera digital. Perangkat output seperti: TV dalam berbagai teknologi (CRT, LCD, plasma, dan lain-lain), komputer dan

layar HP, video proyektor, layar LED multiwarna, dan layar lebar seperti JumboTron, dan lain-lain.

RGB juga umumnya digunakan dalam pembuatan desain sebuah *website.* Awalnya kedalaman warna yang terbatas pada sebagian besar perangkat keras video memicu keterbatasan palet warna menjadi 216 warna RGB, hal itu ditetapkan oleh *Netscape Color Cube*. Namun, dengan keunggulan menampilkan 24-bit, penggunaan penuh dari 16.7 juta warna dari kode warna RGB HTML tidak lagi menimbulkan masalah bagi sebagian besar pengunjung website[3].



Gambar 2. 3 Model warna RGB

2.2.4 Pengenalan Pola (*Pattern Recognition*)

Pengenalan pola (*Pattern Recognition*) adalah suatu ilmu untuk mengklasifikasikan atau menggambarkan sesuatu berdasarkan pengukuran kuantitatif fitur (ciri) atau sifat utama dari suatu objek[7].

Pengenalan pola mengelompokkan data numeric dan simbolik (termasuk citra) secara otomatis oleh mesin (dalam hal ini computer). Tujuan pengelompokan adalah untuk mengenali suatu objek di dalam citra. Manusia bisa mengenali objek yang dilihatnya karena otak manusia telah belajar

mengklasifikasi objek-objek di alam sehingga mampu membedakan suatu objek dengan objek lainnya. Kemampuan sistem visual manusia inilah yang dicoba ditiru oleh mesin. Komputer menerima masukan berupa citra objek yang akan diidentifikasi, memproses citra tersebut, dan memberikan keluaran berupa deskripsi objek didalam citra.



Gambar 2. 4 Pengenalan Pola

Contoh pengenalan pola misalnya citra pada Gambar **2.4** adalah citra mata yang digunakan sebagai data masukan untuk mengenali mata. Dengan menggunakan suatu algoritma pengenalan pola, diharapkan komputer dapat mengenali bahwa citra tersebut adalah mata[7].

2.2.5 Penskalaan Citra (Scaling)

Penskalaan citra (scaling) adalah sebuah operasi geometri yang memberikan efek memperbesar atau memperkecil ukuran citra *input* sesuai dengan variabel penskalaan citranya.

Penskalaan citra (*scaling*) digunakan untuk memperbesar (*zoom-in*) atau memperkecil (*zoom-out*) citra. Rumus yang digunakan untuk penskalaan citra adalah :

$$x' = Sh\ x$$
$$y' = Sv\ y \qquad \textbf{(2.5)}$$

Keterangan : Sh = faktor skala horizontal

Sv = faktor skala vertical



400 x 200 *pixel*



200 x 100 *pixel*

Gambar 2. 5 Penskalaan Citra

2.2.6 Grayscale

Citra *Grayscale* adalah citra yang hanya memiliki satu nilai kanal pada setiap pixelnya, dengan kata lain nilai bagian *Red* = *Green* = *Blue.* Nilai tersebut digunakan untuk menunjukkan tingkat intensitas. Warna yang dimiliki adalah warna dari hitam, keabuan dan putih. Tingkat keabuan di sini merupakan warna abu dengan berbagai tingkatan dari hitam hingga mendekati putih[7].

Pengubahan dari citra berwarna ke bentuk *grayscale* mengikuti aturan sebagai berikut :

$$(x,y,f(x,y)) \rightarrow (x,y,I(x,y)) \qquad \textbf{(2.6)}$$

$$I(i,j)= \frac{R(i,j)+G(i,j)+B(i,j)}{3} \qquad \textbf{(2.7)}$$

Contoh gambar hasil *grayscale* :

Gambar asli ***Gambar grayscale***



Gambar 2. 6 Citra *Grayscale*

2.2.7 Tresholding

Tresholding digunakan untuk mengatur jumlah derajat keabuan yang ada pada citra. Dengan menggunakan *tresholding* maka derajat keabuan bisa diubah sesuai keinginan, misalkan diinginkan menggunakan derajat keabuan 16, maka tinggal membagi nilai derajat keabuan dengan 16. Proses *tresholding* ini pada dasarnya adalah proses pengubahan kuantisasi pada citra, sehingga untuk melakukan *tresholding* dengan derajat keabuan dapat digunakan rumus:

$$x = \frac{\mathrm{w}}{\mathrm{b}} \qquad \textbf{(2.8)}$$

Keterangan :

w = Nilai derajat keabuan sebelum thresholding

b = Jumlah keabuan yang diinginkan { b = (256/a)}

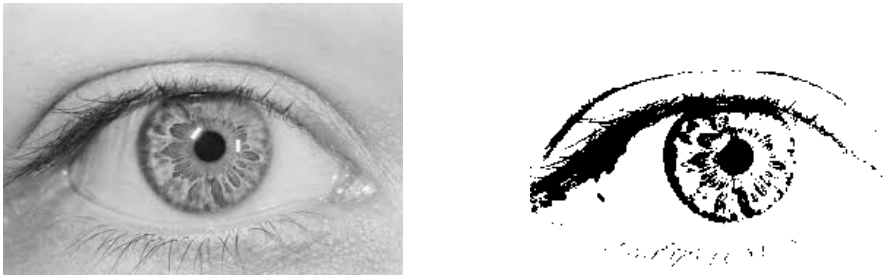
x = Nilai derajat keabuan setelah *thresholding*

Proses *tresholding* akan menghasilkan citra biner, yaitu citra yang memiliki dia nilai tingkat keabuan yaitu hitam dan putih. Secara umum proses

tresholding citra *grayscale* untuk menghasilkan citra biner adalah sebagai berikut :

$$g(x,y) = \begin{cases} 1 \text{ if } f(x,y) \geq T \\ 0 \text{ if } f(x,y) < T \end{cases} \quad (2.9)$$

Berikut ini contoh thresholding untuk mengubah citra grayscale menjadi 2 warna yaitu hitam dan putih :



Gambar 2. 7 Citra *Thresholding*

2.3 Computer Vision

Terminologi lain yang berkaitan erat dengan pengolahan citra digital adalah *computer vision* atau *machine computer*. Pada hakikatnya, *computer vision* mencoba meniru cara kerja visual manusia (*Human Vision*). *Human Vision* sesungguhnya sangat kompleks, manusia melihat obyek dengan indera penglihatan (mata) lalu obyek citra diteruskan ke otak untuk diinterpretasi sehingga manusia mengerti obyek apa yang tampak dalam pandangan matanya. Hasil interpretasi ini mungkin digunakan untuk mengambil suatu keputusan[6].

Sebagaimana layaknya mata dan otak, *computer vision* adalah suatu sistem yang mempunyai kemampuan untuk menganalisis obyek secara visual,

setelah data obyek yang bersangkutan dimasukkan dalam bentuk citra (*image*)[3].

Proses-proses dalam *computer vision* dibagi dalam 3 (tiga) aktifitas :

1. Memperoleh atau mengakuisisi citra digital.
2. Melakukan teknik komputasi untuk memproses atau memodifikasi data citra.

   (operasi-operasi pengolahan citra).
3. Menganalisis dan menginterpretasi citra menggunakan hasil pemrosesan untuk tujuan tertentu, misalnya memandu robot, mengontrol peralatan, memantau manufaktur dan lain-lain.

Pengolahan citra merupakan proses awal (*preprocessing*) pada *computer vision*, sedangkan pengenalan pola merupakan proses untuk menginterpretasi citra. Teknik-teknik di dalam pengenalan pola memainkan peranan penting dalam *computer vision* untuk mengenali objek.

2.3.1 Elemen-Elemen Computer Vision

Gambar **2.8** di bawah ini adalah struktur yang mendasari elemen-elemen suatu mesin vision[6].

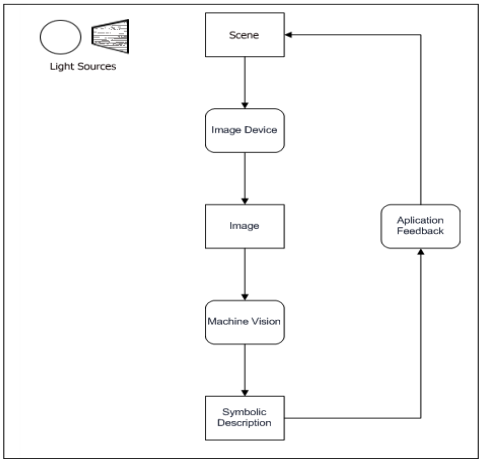
a. *Light sources*, merupakan sumber cahaya yang digunakan sebagai sumber untuk aplikasi seperti layar laser, sistem robotika dan sebagainya.
b. *Scence*, merupakan kumpulan objek.
c. *Image Device*, merupakan alat yang digunakan untuk mengubah gambar menjadi suatu yang dimengerti oleh mesin.

d. *Image*, merupakan gambar dari suatu objek yang merupakan representasi dari keadaan sesungguhnya.

e. *Machine vision*, merupakan mesin yang menginterpretasikan gambar yang berkenaan dengan cirri-ciri pola maupun objek yang dapat ditelusuri oleh sistem.

f. *Symbolic description*, merupakan sistem yang dapat digunakan untuk menganalogikan struktur kerja sistem ke simbol-simbol tertentu yang dapat mimengerti oleh sistem.

g. *Aplication feedback*, merupakan suatu keadaan yang dapat memberikan respon untuk menerima gambar dari suatu sistem penglihatan.



Gambar 2. 8 Struktur *computer vision* secara umum

2.4 Deteksi Mata (*Eye Detection*)

Untuk deteksi mata, pada penelitian ini menggunakan metode yang dibangun oleh *OpenCV* yaitu *cvHaar*. *cvHaar* adalah teknik modern yang mem-verifikasi roman wajah atau obyek yang lainnya. Selain untuk mendeteksi wajah, metode ini juga bisa digunakan untuk mendeteksi lokasi mata. Deteksi obyek oleh *OpenCV* diajukan oleh Paul Viola dan dikembangkan oleh Rainer Lienhart. *OpenCV* menggunakan tipe deteksi wajah disebut *Haar Cascade Classifier*. Dengan memberikan gambar yang berasal dari file maupun *live video*, detector ini menguji tiap lokasi gambar dan mengklasifikasi sebagai wajah atau bukan wajah (dalam penelitian ini mata atau bukan mata). Klasifikasi dilakukan dengan menggunakan data yang disimpan dalam file XML, dimana berfungsi untuk memutuskan klasifikasi tiap lokasi gambar. Instaler *OpenCV* sudah termasuk didalamnya data XML, untuk deteksi wajah atau mata. Dalam aplikasinya cukup memberitahu software, dimana dapat menemukan file data yang ingin digunakan[5].

2.4.1 Integral Proyeksi

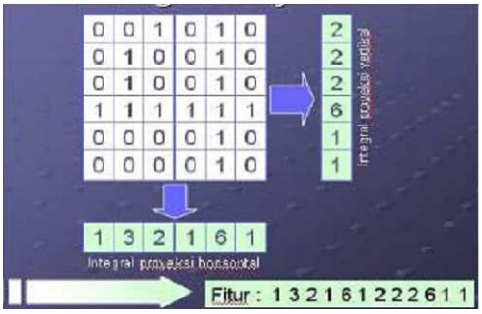
Integral proyeksi adalah suatu metode yang digunakan untuk mencari daerah lokasi dari obyek. Metode ini dapat kita gunakan untuk mendeteksi batas dari daerah gambar yang berbeda sehingga kita bisa mencari daerah lokasi kepala, tangan, kaki dan fitur – fitur lainnya. Integral ini juga di sebut dengan integral baris dan kolom dari pixel, karena integral ini menjumlahkan pixel per baris dan pixel per kolom. Dari metode ini akan dengan mudah untuk menemukan daerah lokasi objek yang diperlukan.

Rumus dari Integral Proyeksi adalah sebagai berikut :

$$h(j) = \sum_{i=1}^{Nbaris} x(i, j) \qquad \textbf{(2.10)}$$

$$h(i) = \sum_{j=1}^{Nkolom} x(i, j)$$

Contoh Penghitungan dalam integral proyeksi :



Gambar 2. 9 Contoh perhitungan *integral* proyeksi

2.4.2 Metode Haar Cascade Classifier

Proses deteksi adanya citra mata dalam sebuah gambar pada OpenCV, menggunakan sebuah metode yang umumnya disebut metode Haar Classifier. Metode ini merupakan metode yang menggunakan *statistical model* (*classifier*). Pendekatan untuk mendeteksi objek dalam gambar menggabungkan empat konsep utama : [1]

1. Fitur segi empat sederhana yang disebut fitur Haar.
2. Integral image untuk pendeteksian fitur secara cepat.

3. Metoda machine learning AdaBoost yang digunakan untuk meningkatkan kinerja algoritma classifier[14].
4. Pengklasifikasi bertingkat (Cascade classifier) untuk menghubungkan banyak fitur secara efisien.

2.4.2.1 Sistem Kerja Algoritma Haar Cascade Classifier

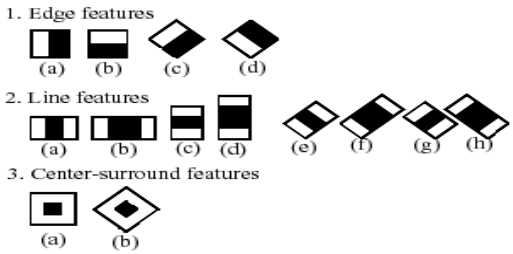
Algoritma Haar menggunakan metode statistikal dalam melakukan pengenalan mata. Metode ini menggunakan *simple haar-like features* dan juga *a cascade of boosted tree classifier.* Classifier ini menggunakan gambar berukuran tetap. Cara kerja dari haar dalam mendeteksi mata adalah menggunakan teknik *sliding window* pada keseluruhan gambar dan mencari apakah terdapat bagian dari gambar yang berbentuk seperti mata atau tidak. Haar juga memiliki kemampuan untuk melakukan *scalling* sehingga dapat mendeteksi adanya mata yang berukuran lebih besar ataupun lebih kecil dari gambar pada *classifier*.

Tiap *feature* dari *haar-like feature* didefinisikan pada bentuk dari *feature*, diantaranya koordinat dari *feature* dan juga ukuran dari *feature* tersebut.

2.4.2.2 Haar Feature

Haar Feature adalah fitur yang digunakan oleh Viola dan Jones didasarkan pada Wavelet Haar. Wavelet Haar adalah gelombang tunggal bujur sangkar (satu interval tinggi dan satu interval rendah). Untuk dua dimensi, satu terang dan satu gelap. Selanjutnya kombinasi-kombinasi kotak yang digunakan untuk pendeteksian objek visual yang lebih baik

Setiap *Haar-like feature* terdiri dari gabungan kotak - kotak hitam dan putih :



Gambar 2. 10 *haar Feature*

Tiga tipe kotak(rectangular) feature:

Tipe *two-rectangle feature* (horisontal/vertikal)

Tipe *three-rectangle feature*

Tipe *four-rectangle feature*

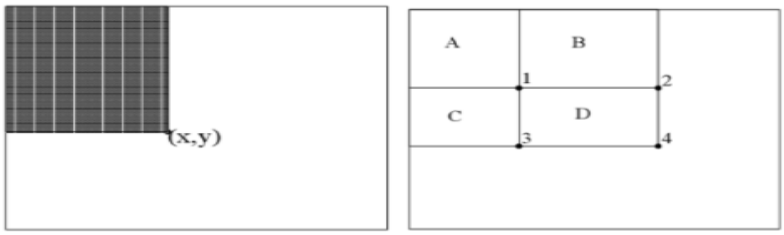
Adanya fitur Haar ditentukan dengan cara mengurangi rata-rata piksel pada daerah gelap dari rata-rata piksel pada daerah terang. Jika nilai perbedaannya itu diatas nilai ambang atau treshold, maka dapat dikatakan bahwa fitur tersebut ada.

Nilai dari *Haar-like feature* adalah perbedaan selisih antara jumlah nilai-nilai piksel *gray level* dalam daerah kotak hitam dan daerah kotak putih. Kotak Haar-like feature dapat dihitung secara cepat menggunakan *integral image*.

2.4.2.3 Integral Image

Integral Image digunakan untuk menentukan ada atau tidaknya dari ratusan fitur Haar pada sebuah gambar dan pada skala yang berbeda secara

efisien. Pada umumnya, pengintegrasian tersebut berarti menambahkan unit-unit kecil secara bersamaan. Dalam hal ini unit-unit kecil tersebut adalah nilai-nilai piksel. Nilai integral untuk masing-masing piksel adalah jumlah dari semua piksel-piksel dari atas sampai bawah. Dimulai dari kiri atas sampai kanan bawah, keseluruhan gambar itu dapat dijumlahkan dengan beberapa operasi bilangan bulat per piksel.



(a) **(b)**

Gambar 2. 11 ***integral image***

Seperti yang ditunjukkan oleh gambar **2.6(a)** di atas setelah pengintegrasian, nilai pada lokasi piksel (x,y) berisi jumlah dari semua piksel di dalam daerah segiempat dari kiri atas sampai pada lokasi (x,y) atau daerah yang diarsir. Untuk menentukan nilai rata-rata piksel pada area segiempat (daerah yang diarsir) ini dapat dilakukan hanya dengan membagi nilai pada (x,y) oleh area segiempat.

$$\mathbf{P(x, y) = \sum i(x', y')} \qquad \mathbf{(2.11)}$$

Untuk mengetahui nilai piksel untuk beberapa segiempat yang lain missal, seperti segiempat D pada gambar **2.6(b)** di atas dapat dilakukan dengan cara menggabungkan jumlah piksel pada area segiempat A+B+C+D,

dikurangi jumlah dalam segiempat A+B dan A+C, ditambah jumlah piksel di dalam A. Dengan, A+B+C+D adalah nilai dari integral image pada lokasi 4, A+B adalah nilai pada lokasi 2, A+C adalah nilai pada lokasi 3, dan A pada lokasi 1.

Sehingga hasil dari D dapat dikomputasikan :

$$\mathbf{D = (A+B+C+D)-(A+B)-(A+C)+A} \qquad \mathbf{(2.12)}$$

Contoh Integral Image :

Image

0	**8**	**6**	**1**
1	**5**	**9**	**0**
0	**7**	**5**	**0**
2	**8**	**9**	**2**



Integral Image

0	**8**	**14**	**15**
1	**14**	**29**	**30**
1	**21**	**41**	**42**
3	**31**	**60**	**63**

Gambar 2. 12 Contoh perhitungan *integral image*

2.4.2.4 Adaptive Boosting (AdaBoost)

Stage classifier dibangun dengan menggunakan algoritma adaptive-boost (*AdaBoost*). Algoritma tersebut mengkombinasikan performance banyak weak classifier untuk menghasilkan *strong* classifier. *Weak* classifier dalam hal ini adalah nilai dari haar-like feature. Jenis *AdaBoost* yang digunakan adalah *Gentle AdaBoost*[14].

Diberikan :

training set: $(x_1, y_1),\ldots,(x_m, y_m)$ dimana

$x_i \sum X,\ y_i \sum Y = \{-1, 1\}$

inisialisasi $D_1(i) = \frac{1}{m}$, i = 1,…,m.

For t = 1 ,…, T

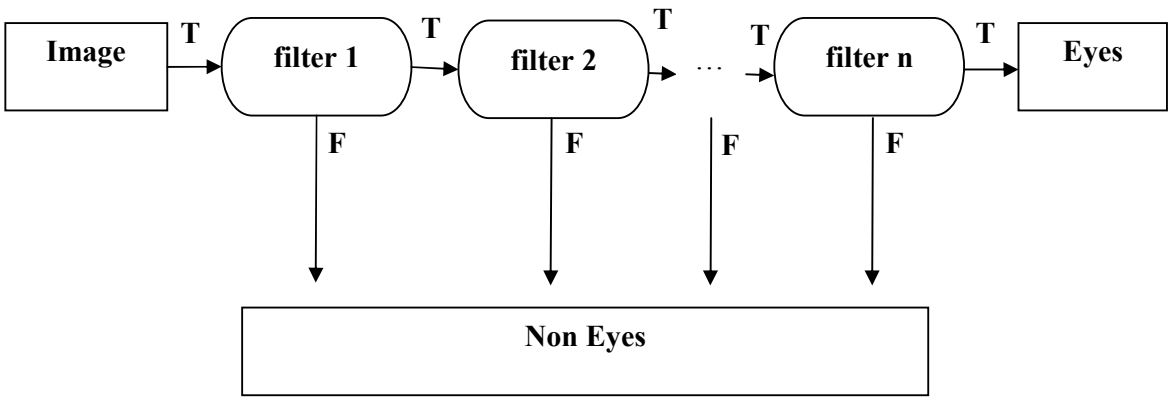
Dari kumpulan *classifiers* $\mathcal{H}$, carilah *classifier* H_t yang memaksimalkan nilai yang dibedakan dari error rate (E_t) dari nilai 0.5 yang berhubungan dengan nilai distribusi D_t :

$h_t = \text{argmax} \mid 0.5 - E_t \mid$

Dimana nilai E_t diperoleh dari $E_t = \sum_{i=1}^{m} D_t(i)\, I(y_i \neq h_t(x_i))$

2.4.2.5 *Cascade Classifier*

Cascade classifier adalah sebuah rantai *stage classifier*, dimana setiap *stage classifier* digunakan untuk mendeteksi apakah didalam image *sub window* terdapat obyek yang diinginkan (*object of interest*)[1].



Gambar 2. 13 *Cascade Classifier*

2.5 Metode Fisher's Linear Discriminant Analisys

Metode ini ditemukan oleh Robert Fisher pada tahun 1936 untuk klasifikasi taksonomi dan menjadi salah satu tehnik yang banyak digunakan dalam pengenalan pola (*pattern recognition*) [13].

FLD merupakan salah satu contoh metode *class spesific*, karena metode ini berusaha untuk membentuk jarak (*scatter*) antar kelas dan intra kelas sehingga dapat menghasilkan klasifikasi yang lebih baik.

Jika terdapat 20 sampel (N = 20) yang terbagi atas 2 kelas sampel. Dimensi sampel, n = 2 dan dimensi reduksi, m = 1. Jadi, sampel akan ditransformasikan dari ruang sampel dimensi-2 ke ruang ciri dimensi-1.

Fisher's Linear Discriminant, yang menjadi dasar dari metode *Fisherface* yang akan digunakan untuk mengenali pola pada gambar, memilih matriks transformasi *W* yang dapat memaksimalkan rasio antara determinan *between-class scatter* dengan *within-class scatter* dari vektor-vektor ciri. Pernyataan tersebut dapat diekspresikan sebagai berikut :

$$W_{opt} = \arg\max_{W} \frac{\left|WS_B W^T\right|}{\left|WS_W W^T\right|} \qquad \textbf{(2.13)}$$

$$= [w_1; w_2; \ldots; w_m]$$

dimana w_i, i = 1...m, merupakan m buah vektor eigen dari rasio antara S_B dengan S_W (dalam bentuk vektor baris), yang bersesuaian dengan m buah nilai eigen terbesar.

Jika w_i adalah vektor eigen dari rasio antara matriks S_B dengan matriks S_W dan d_i merupakan nilai eigen yang bersesuaian, maka :

$$S_B w_i^T = d_i S_W w_i^T$$

dimana i = 1...m dan $d_1 > d_2 > \ldots > d_m$.

Jika x_i, i = 1...N, adalah vektor citra dimensi-n dan masing-masing vektor citra merupakan anggota salah satu dari C kelas citra wajah $\{X_1, X_2, \ldots, X_C\}$,

vektor μ adalah rata-rata vektor citra yang dapat diperoleh dari persamaan (2.1), maka matriks *between-class scatter*, S_B, dan matriks *within-class scatter*, S_W, dapat diperoleh melalui persamaan berikut :

$$S_B = \sum_{i=1}^{C} N_i (\mu_i - \mu)^T (\mu_i - \mu) \qquad \textbf{(2.14)}$$

$$S_W = \sum_{i=1}^{C} \sum_{j=1, in_j \in X_i}^{N_i} (x_j - \mu_i)^T (x_j - \mu_i) \qquad \textbf{(2.15)}$$

dimana N_i adalah jumlah anggota kelas X_i dan μ_i adalah rata-rata citra anggota kelas X_i, i = 1…C.

Pada pengenalan wajah, matriks *within-class scatter*, S_W, yang dihasilkan selalu singular. Hal ini disebabkan oleh rank maksimum dari matriks S_W adalah (N-C) dan pada umumnya, jumlah citra masukan, N, jauh lebih kecil dari dimensi citra, n [12].

Untuk mengatasi hal ini, vektor-vektor citra terlebih dahulu ditransformasikan ke ruang ciri berdimensi-(N-C) dengan metode PCA, kemudian menerapkan persamaan (2.9) dengan vektor input baru,yaitu x'_i, i = 1..N, berdimensi (N-C). Inilah yang disebut sebagai metode *Fisherface*.

2.6 *Tracking Mata* (Eye Tracking)

Tracking mata dilakukan menggunakan algoritma *template machine*. Citra mata dari *webcam* akan dibandingkan dengan citra mata yang sudah disimpan sebagai *database*, dalam hal ini disebut metode prinsip pencocokan bentuk atau metode *matching template*. Dalam *OpenCV* digunakan *cvnorm* yaitu mencari nilai perubahan bentuk membandingkan citra mata dalam bentuk *ray* yang di-*capture* dari *webcam* dengan citra mata dalam bentuk gray yang disimpan sebagai

database, sehingga didapatkan arah posisi pandangan mata yaitu arah mata ke kanan, ke kiri, ke atas, ke bawah.

2.6.1 *Template Machine*

Template machine adalah proses mencari suatu obyek (*template*) pada keseluruhan obyek yang berada dalam suatu citra. *Template* dibandingkan dengan keseluruhan obyek tersebut dan bila *template* cocok (cukup dekat) dengan suatu obyek yang belum diketahui pada citra tersebut maka obyek tersebut ditandai sebagai *template*[7].

Perbandingan antara *template* dengan keseluruhan obyek pada citra dapat dilakukan dengan menghitung selisih jaraknya, seperti berikut :

$$\mathbf{D(m, n) = \sum \sum [f(\mathit{j, k}) - T(\mathit{j-m,k-n})]} \qquad \mathbf{(2.16)}$$

Dengan *f(j,k)*menyatukan citra tempat obyek yang akan dibandingkan dengan *template* T(j,k) sedangkan D(m,n) menyatakan jarak antara *template* dengan obyek pada citra. Pada umumnya ukuran *template* jauh lebih kecil dari ukuran citra.

Secara ideal *template* dikatakan cocok dengan obyek pada citra bila D(m,n)=0, namun kondisi seperti ini sulit dipenuhi apalagi bila *template* merupakan suatu citra *grayscale.*

Oleh karena itu, aturan yang digunakan untuk menyatakan *template* cocok dengan obyek adalah bila :

$$D(m,n) < L_D(m,n) \quad \textbf{(2.17)}$$

Dengan $L_D(m,n)$ merupakan nilai *treshold*.

2.7 Deteksi Kedipan (*Blink Detection*)

Berdasarkan konstren geometri wajah, mata terletak pada setengah bagian atas wajah. Karena pada umumnya, etika seseorang berkedip kedua matanya bergerak bersamaan, maka dalam deteksi kedipan mata ini, sistem megidentifikasi kedipan masing – masing mata, baik mata sebelah kiri ataupun kanan.

Dari hasil segmentasi daerah mata yang diperoleh pada tahap sebelumnya, dapat diketahui posisi mata. Untuk itu citra warna pada daerah mata diubah menjadi citra keabuan (*grayscale*), yaitu dengan menghitung nilai keabuan tiap piksel (x,y). Dimana gray (x,y) adalah nilai intensitas keabuan pada piksel (x,y), sedangkan RGB adalah komponen citra warna. Proses selanjutnya adalah melakukan *equalisasi histogram* pada citra yang bertujuan untuk mengatasi masalah pencahayaan.

Tahap berikutnya yang penting adalah operasi morfologi. Dalam penelitian ini operasi morfologi digunakan untuk menghilangkan *noise* yang ada. Sebelum dilakukan operasi morfologi, dipilih daerah tertentu saja (daerah yang di-bounding) yang diproses atau biasa disebut ROI (*Region of Interest*) untuk mempercepat proses komputasi. Setelah menetapkan daerah ROI (pada penelitian ini ROI adalah daerah mata), pada daerah ersebut dilakukan proses *tresholding*, dalam penelitian ini dipilih nilai ambang batas 5. Bila nilai intensitas keabuan piksel dibawah atau sama dengan 5, hasil deteksinya ditandai dengan warna putih,

sedangkan bila nilai intensitas keabuannya di atas 5, hasil deteksinya ditandai dengan warna hitam[6].

2.8 *Sistem* Real Time

Sistem *real time* merupakan suatu sistem yang menangkap citra, memindahkan bingkai ke dalam memori computer, melakukan analisis dan perhitungan, dan menghasilkan citra lain atau lebih lagi sering digunakan untuk melakukan aksi, misalnya memberi predikat pada obyek yang diambil citranya seperti pada sistem sortasi, atau menggerakkan manipulator untuk memetik buah pada robot pemanen buah, dan sebagainya[3].

Menurut kamus "*Oxford Dictionary of Computing*", Real-time system dapat didefinisikan sebagai: "Sistem apapun dalam hal waktu dimana suatu keluaran dihasilkan adalah penting. Hal ini biasanya dikarenakan suatu masukan yang berhubungan dengan suatu pergerakan dalam dunia fisik, dan keluarannya harus tetap memiliki hubungan dengan pergerakan tersebut. Keterlambatan dari waktu masuk sampai waktu keluar harus cukup kecil dan memenuhi batasan waktu yang dapat diterima".

Secara umum sistem *real time* merupakan suatu ukuran kinerja suatu sistem tertentu yang mana menyangkut batasan kinerja sistem dan tahapan perancangan sistem. Dalam sebuah sistem *real time* yang mana memiliki permasalahan waktu harus nyata, maksudnya harus memiliki persyaratan pewaktuan yang sesuai dengan keadaan nyata yang dijadikan parameter dalam sistem itu.

2.9 ROI (Region of Interest)

Region of Interest (ROI) merupakan salah satu fitur yang tersedia dalam JPEG2000. ROI memungkinkan dilakukannya pengkodean secara berbeda pada area tertentu dari citra dijital, sehingga mempunyai kualitas yang lebih baik dari area sekitarnya (*background*). Fitur ini menjadi sangat penting, bila terdapat bagian tertentu dari citra dijital yang dirasakan lebih penting dari bagian yang lainnya[15].

2.10 Open CV

OpenCV adalah suatu *library* gratis yang dikembangkan oleh *Intel Corporation*. *Library* ini terdiri fungsi-fungsi *computer vision* dan *image processing* tingkat tinggi. OpenCV diusulkan kepada para programmer untuk dapat menciptakan aplikasi yang handal, kuat dibidang digital vision. Karena *library* ini bersifat cuma-cuma dan terbuka, OpenCV tidak dipesan khusus untuk pengguna arsitektur Intel, tetapi dapat dibangun hampir semua arsitektur.

Anda harus menambahkan .cv. untuk menggunakan fungsi apapun dari OpenCV. OpenCV juga menyediakan fungsi-fungsi GUI yang disebut highGUI (*high graphical user interface*). Fungsi-fungsi GUI ini adalah sangat penting sebab dibutuhkan untuk menampilkan gambar yang akan dan sudah diproses sesuai keinginan kita[5].

2.11 Konsep Dasar Analisa Sistem

Analisis sistem sangat bergantung pada teori sistem umum sebagai sebuah landasan konseptual. Tujuannya adalah untuk memperbaiki berbagai fungsi didalam sistem agar menjadi lebih efisien, mengubah sasaran sistem yang sedang

berjalan, merancang/mengganti output yang sedang digunakan untuk mencapai tujuan yang sama dengan seperangkat input yang lain (bisa jadi lebih sederhana dan lebih interaktif) atau melakukan beberapa perbaikan serupa. Istilah analisis sistem ini memang tepat, karena memang itulah yang dikerjakan oleh analis sistem dalam tahap ini, yaitu mengalisis sistem untuk menemukan kelemahan-kelemahannya sehingga dapat diusulkan perbaikannya.

2.11.1 Diagram Konteks

Diagram konteks adalah diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem. Diagram konteks merupakan level tertinggi dari DFD (Data Flow Diagram) yang menggambarkan seluruh input ke sistem atau output dari sistem. Ia akan memberi gambaran tentang keseluruhan sistem. Sistem dibatasi oleh boundary (dapat digambarkan dengan garis putus). Dalam diagram konteks hanya ada satu proses. Tidak boleh ada store dalam diagram konteks.

2.11.2 Data Flow Diagram (DFD)

Diagram alir data (data flow diagram) merupakan model dari sistem untuk menggambarkan pembagian sistem ke modul yang lebih kecil. Salah satu keuntungan menggunakan diagram alir data ialah memudahkan pemakai atau user yang kurang menguasai bisang komputer untuk mengerti sistem yang akan dikerjakan.

2.12 *Software* Pendukung

2.12.1 *Microsoft Visual* C#

C# (dibaca: C sharp) merupakan sebuah bahasa pemrograman yang berorientasi objek yang dikembangkan oleh Microsoft sebagai bagian dari inisiatif kerangka .NET Framework. Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek-aspek ataupun fitur bahasa yang terdapat pada bahasa-bahasa pemrograman lainnya seperti Java, Delphi, Visual Basic, dan lain-lain) dengan beberapa penyederhanaan. Menurut standar *ECMA-334 C# Language Specification*, nama C# terdiri atas sebuah huruf Latin C (U+0043) yang diikuti oleh tanda pagar yang menandakan angka # (U+0023). Tanda pagar # yang digunakan memang bukan tanda kres dalam seni musik (U+266F), dan tanda pagar # (U+0023) tersebut digunakan karena karakter kres dalam seni musik tidak terdapat di dalam *keyboard* standar.Pointer telah dihilangkan dari C#. Operasi-operasi yang 'tidak aman' seperti misalnya akses memori langsung sudah tidak diizinkan. Tidak ada lagi penggunaan operator "::" atau "->" seperti pada pendahulunya. Dukungan tipe data primitif yang lebih banyak seperti misalnya Integer, Floats, dan sebagainya. Nilai integer 0 dan 1 tidak diterima lagi sebagai alternatif nilai Boolean. Nilai Boolean adalah murni *true* atau *false*.

2.12.2 Microsoft Visual Studio 2010

Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap (suite) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi console, aplikasi Windows, ataupun aplikasi Web. Visual Studio

mencakup kompiler, SDK, Integrated Development Environment (IDE), dan dokumentasi (umumnya berupa MSDN Library). Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual J#, Visual FoxPro, dan Visual SourceSafe. Microsoft Visual Studio dapat digunakan untuk mengembangkan aplikasi dalam native code (dalam bentuk bahasa mesin yang berjalan di atas Windows) ataupun managed code (dalam bentuk Microsoft Intermediate Language di atas .NET Framework). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi Silverlight, aplikasi Windows Mobile (yang berjalan di atas .NET Compact Framework).

2.12.3 Microsoft Visio 2007

Microsoft Visio 2007 adalah *software* yang memiliki perangkat-perangkat pemodelan secara visual untuk membangun suatu solusi dalam rekayasa *software* dan pemodelan bisnis. *Software* ini memudahkan pengguna untuk mengoneksikan diagram-diagram buatan mereka terhadap beberapa sumber data dan juga menampilkan informasi secara visual dengan menggunakan grafik.

2.13 Hardware *Pendukung*

2.13.1 Web Camera (*WebCam*)

Kamera digital digunakan sebagai sarana masukan untuk memberikan data objek gambar ke komputer. Sebuah kamera digital dapat menjangkau hingga resolusi yang cukup. Biasanya resolusi yang mendukung mencapai 320x460 sampai 768x1024 pixel. Kemampuan suatu kamera untuk

mengambil banyak gambar utuh (frame) dalam satu detiknya menjadi penting.

Untuk menghasilkan video yang memiliki pergerakan yang lancar, suatu kamera setidaknya harus memiliki kemampuan mengambil gambar utuh sebanyak 30 frame setiap detiknya. Semakin banyak gambar utuh yang diambil setiap detiknya makin baik gambar yang dihasilkan. Ada dua jenis kamera yang bisa digunakan, pertama adalah tipe WebCam yang dihubungkan melalui USB. Untuk tipe ini, citra yang dihasilkan kurang bagus. Tipe kedua adalah tipe VCR (Video Camera Recorder) atau kamera video digital yang dihubungkan melalui port fireware. Untuk jenis kedua, citra yang dihasilkan baik, dengan resolusi tinggi.

2.13.2 Koneksi Kamera

Ada banyak cara untuk melakukan koneksi kamera pada pemograman Visual C++. Pada proyek akhir ini, kami menggunakan video OCX sebagai penghubung antara program dengan web kamera. Dimana didalam Video OCX tersebut terdapat bermacam-macam metode-metode pengolahan citra seperti *difference* dan *tresholding* yang dipakai dalam tugas akhir ini, seperti:

Tabel 2. 1 Berbagai macam metode koneksi program ke kamera

METHOD	DESKRIPSI
m_Vid.Init()	Inisialisasi dan koneksi kamera
m_Vid.SetPreview(TRUE)	Mengaktifkan video preview dari kamera

	pada kontrol window
m_Vid.Start()	Menyiapkan kamera untuk proses capture
m_Vid.Capture()	Menangkap image ataui frame
m_Vid.GetColorImageHandle()	Menampung image dalam format RGB
m_Vid.GetGrayImageHandle	Menampung image dalam format gray scale
m_Vid.SaveBMP(long imghandle, LPCSTR filename)	Menyimpan image dalam file format BMP
m_Vid.SaveJPEG(long imghandle, LPCSTR filename)	Menyimpan image dalam file format JPEG
m_VidTool.Difference(long imghandle1, long imghandle2, long imghandleTarget)	Menampilkan perbedaan antar frame dalam format gray scale
m_VidTool.Threshold(long imghandleSrc, int th, long imghandleTarget)	Merubah format image dari grayscale ke format binary
m_Vid.Stop()	Menghentikan kamera untuk proses capture
m_Vid.Close()	Menutup koneksi kamera