BAB 2
TINJAUAN PUSTAKA

2.1 Mata Mengantuk

Mengantuk merupakan kondisi ketika tubuh membutuhkan istirahat atau tidur dan didefinisikan sebagai kecenderungan untuk tidur. Mengantuk dapat disebabkan oleh kelelahan melakukan pekerjaan yang berulang-ulang seperti survey monitor ataupun mengendarai kendaraan ketika dalam perjalanan jauh. Kantuk dan lelah memiliki banyak efek yang sama. Pada kejadian ini, pada saat ketika kelopak mata mulai terasa berat dan seketika menutup 100%, pandangan mulai kabur dan tiba-tiba saja kelopak mata tersebut sudah menutup 100% ,sehingga mata tanpa bisa diajak kompromi lagi. Padahal dalam pikiran, merasa masih terjaga. Hal itulah yang menjadi tanda akan seseorang tersebut mengantuk. Oleh karena itu dalam penelitian kali ini akan mendeteksi mata mengantuk, menggunakan objek mata dalam keadaan tertutup 100% [9]. Mata dalam keadaan terbuka dapat dilihat pada Gambar 2.1 dan mata dalam keadaan tertutup dapat dilihat pada Gambar 2.2



Gambar 2.1 Mata dalam keadaan terbuka



Gambar 2.2 Mata dalam keadaan tertutup 100%

2.2 Parameter Mengantuk

Menurut penelitian Tecce (1992) [9], frekuensi kedipan dapat dipengaruhi faktor yang berbeda seperti : kondisi dan perintah. Dalam keadaan normal atau bebas dari stres rata-rata kedipan mata adalah 15 sampai 20 kali permenit. Frekuensi ini menurun sampai 3 kali permenit ketika membaca. Frekuensi tersebut meningkat dalam keaadaan stress, tertekan ataupun ketika menutupnya mata saat dibutuhkan.

Indikator untuk mengetahui seseorang sedang mengantuk dapat dideskripsikan ketika kondisi normal (tidak mengantuk) posisi kelopak mata membuka lebar sebelum menutup. Ketika menutupmemiliki interfal waktu yang cepat (kurang dari satu detik). Ketika seseorang mulai lelah dan mengantuk, jarak antara kedua kelopak mata semakin menyempit dan frekuensi kedipan semakin menurun hingga tertidur.

Untuk memodelkan pengemudi yang sedang mengantuk dapat diindikasikan bahwa terdapat parameter-parameter sebagai berikut:

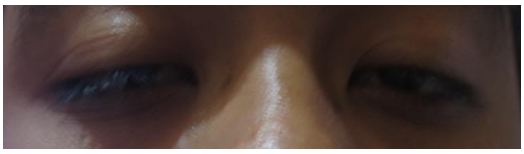
1. Menurunnya interest interest terhadap lingkungan .
2. Meningkatnya kantuk atau kecenderungan untuk tidur, yaitu ditandai dengan meningkatnya durasi kedipan mata untuk menutup.

Menurut studi yang dilakukan oleh Phillip .P. Caffier [10], mengelompokkan tingkatan kantuk berdasarkan durasi kedipan mata. Umumnya durasi kedipan rata-rata adalah kurang dari 400 Ms dan 75 Ms untuk minimum. Berdasarkan alasan ini, maka digunakan 400 Ms sebagai waktu kantuk (T_kantuk) dan 800 Ms sebagai waktu telah tertidur (T_tidur).

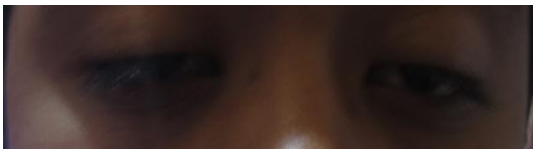
Tabel 2.1 Deteksi kantuk berdasarkan durasi kedipan

Level Kantuk	Deskripsi
Normal (Terbangun)	Durasi kedipan < T_kantuk
Mengantuk	Durasi kedipan > T_kantuk dan Durasi kedipan < T_tidur
Tidur	Durasu kedipan >= T_tidur

Macam – macam gambar mata mengantuk



Gambar 2.3 Mengantuk 1



Gambar 2.4 Mengantuk 2



Gambar 2.5 Mengantuk 3

2.3 Pengolahan Citra *(Image Processing)*

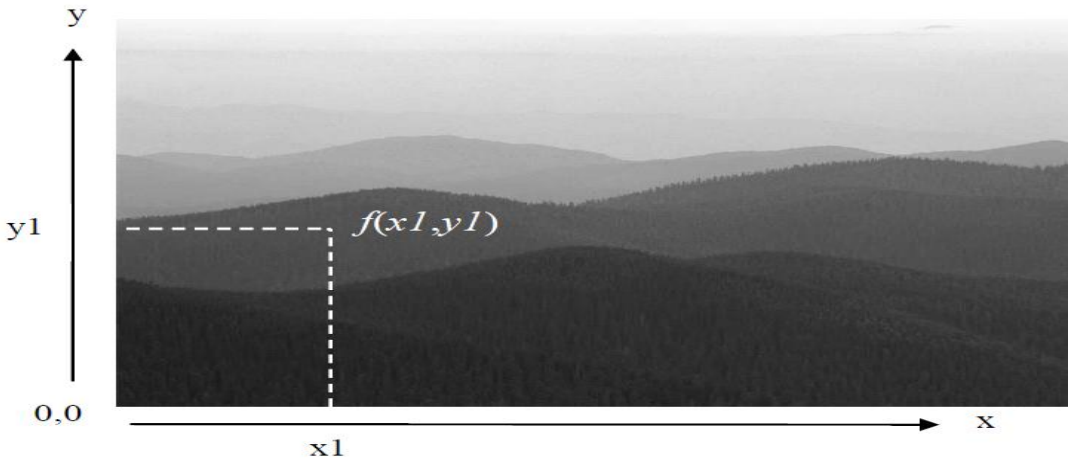
Pengolahan citra (*image processing*) adalah pengolahan suatu citra (gambar) dengan menggunakan komputer secara khusus, untuk menghasilkan suatu citra yang lain. Sesuai dengan perkembangan komputer itu sendiri, pengolahan citra mempunyai dua tujuan utama[6], yaitu sebagai berikut:

1. Memperbaiki kualitas citra, dimana citra yang dihasilkan dapat menampilkan informasi secara jelas. Hal ini berarti manusia sebagai pengolah informasi *(human perception).*
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. Oleh sebab itu uncul 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.

2.4 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 2.6 Citra Digital

a. Gambar *Grayscale*

Citra yang terdiri dari satu layer warna dengann derajat keabuan tertentu. Dinyatakan dalam suatu fungsi : $f(x,y) \Sigma [0\ldots255]$ ..... (2.1)

b. Gambar Biner

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

$f(x,y) \Sigma \{0,1\}$

.....(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 :

$$f_R(x,y)\ \Sigma\ [0\ldots255] \quad \text{.....(2.3)}$$
$$f_G(x,y)\ \Sigma\ [0\ldots255]$$
$$f_B(x,y)\ \Sigma\ [0\ldots255]$$

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*).

2.5 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 objek dengan indera penglihatan (mata) kemudian objek citra diteruskan ke otak untuk diinterpretasi sehingga manusia mengerti objek apa yang tampak dalam pandangan matanya. Hasil interpretasi ini mungkin digunakan untuk mengambil suatu keputusan [6].

Sebagaimana layaknya mata dan otak, *computer vision* adalah merupakan proses otomatis yang mengintegrasikan sejumlah besar proses untuk persepsi visual, seperti akuisisi citra, pengolahan citra, pengenalan dan membuat keputusan. *Computer vision* mencoba meniru cara kerja sistem visual manusia *(human vision)* yang sesungguhnya sangat kompleks, setelah data objek yang bersangkutan dimasukkan dalam bentuk citra (*image*) [6].

Manusia melihat dengan objek dengan indera penglihatan (mata), lalu citra objek diteruskan ke otak untuk diinterpretasi sehingga manusia mengerti objek apa yang tampak dalam pandangan mata. Hasil interpretasi ini digunakan untuk pengambilan keputusan (misal ketika mata pengemudi mengantuk, *alarm system* bekerja).

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

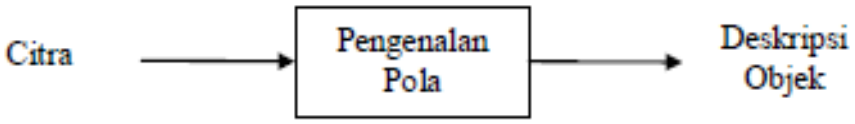
1. Memperoleh atau mengakuisisi citra digital.
2. Melakukan teknik komputasi untuk memproses atau memodifikasi data citra.
3. Menganalisis dan menginterpretasi citra menggunakan hasil pemrosesan untuk tujuan tertentu, misalnya memandu robot, mengontrol peralatan, memantau manufaktur dan lain-lain.

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.

Pengolahan citra merupakan proses awal pada computer vision, pengenalan pola merupakan proses untuk menginterpretasi citra.

2.6 Pengenalan Pola

Pengenalan pola adalah mengelompokkan data numerik dan simbolik (termasuk citra) secara otomatis oleh mesin (komputer).Tujuan pengelompokkan 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 yang dicoba ditiru oleh mesin. Komputer menerima masukan berupa citra objek yang akan diidentifikasi, memproses citra tersebut dan memberikan keluaran berupa informasi/deskripsi objek di dalam citra.[5]



Gambar 2.7 Pola pengenalan[5]

Contoh pengenalan pola misalnya citra pada Gambar 5adalah citra mata yang digunakansebagai data masukan untuk mengenali mata. Dengan menggunakan suatu algoritmapengenalan pola, diharapkan komputer dapat mengenali bahwa citra tersebut adalah mata.

2.7 Deteksi Mata *(Eye Detection)*

Untuk deteksi mata, pada penelitian ini menggunakan metode yang dibangun oleh *EmguCV*. Deteksi objek oleh *EmguCV* diajukan oleh Paul Viola dan dikembangkan oleh Rainer Lienhart.*EmguCV* menggunakan tipe deteksi objek yang disebut *Haar Cascade Classifier*. Dengan memberikan gambar yang berasal dari *file* maupun *live video*, *detector* ini menguji tiap lokasi gambar dan mengklasifikasi sebagai objek atau bukan objek (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 *EmguCV* sudah termasuk didalamnya data XML, untuk deteksi objek.Dalam aplikasinya cukup memberitahu *software*, dimana dapat menemukan *file* data yang ingin digunakan [3].

2.8 Haar Cascade Classifier

Proses deteksi adanya citra mata dalam sebuah gambar pada OpenCV, menggunakan sebuah metode yang dipublikasikan oleh Paul Viola dan Michael Jones tahun 2001. Umumnya disebut metode *Haar Classifier*.Metode ini merupakan metode yang menggunakan *statistical model* (*classifier*)[7]. Pendekatan untuk mendeteksi objek dalam gambar menggabungkan empat konsep utama:

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

4. Pengklasifikasi bertingkat (*Cascade classifier*) untuk menghubungkan banyak fitur secara efisien

2.8.1 *Training* Data pada Haar

Metode ini memerlukan 2 tipe gambar objek dalam proses *training* yang dilakukan[7], yaitu :

1. *Positive samples*

Berisi gambar objek yang ingin di deteksi, apabila ingin mendeteksi mata maka *positive samples* ini berisi gambar mata, begitu juga objek lain yang ingin dikenali.

2. *Negative samples*

Berisi gambar objek selain objek yang ingin dikenali, umumnya berupa gambar *background* (tembok, pemandangan, lantai, dan gambar lainnya). Resolusi untuk sampel negatif disarankan untuk memiliki resolusi yang sama dengan resolusi kamera.

Training dari Haar menggunakan dua tipe sampel diatas.Informasi dari hasil training ini lalu dikonversi menjadi sebuah parameter model statistik.

2.8.2 Sistem kerja Algoritma *Haar Cascade Classifier*

Algoritma Haar menggunakan metode statistikal dalam melakukan pengenalan mata. Metode ini menggunakan *simple haar-like features* dan juga *AdaBoost Classifier. Classifier* ini menggunakan gambar berukuran tetap. Cara kerja dari Haar dalam mendeteksi mata adalah menggunakan teknik *resize* pada gambar dan mencari apakah terdapat bagian dari gambar yang berbentuk seperti mata atau tidak. Haar juga memiliki kemampuan untuk melakukan *scaling* 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.

2.9 Persiapan DataSet data training

Dataset terdiri dari 2 buah sample. Sample positif adalah gambar yang mengandung obyek yang akan dideteksi. Jika kita menginginkan mata mengantuk untuk dideteksi maka sample positif berisi gambar – gambar mata tertutup.

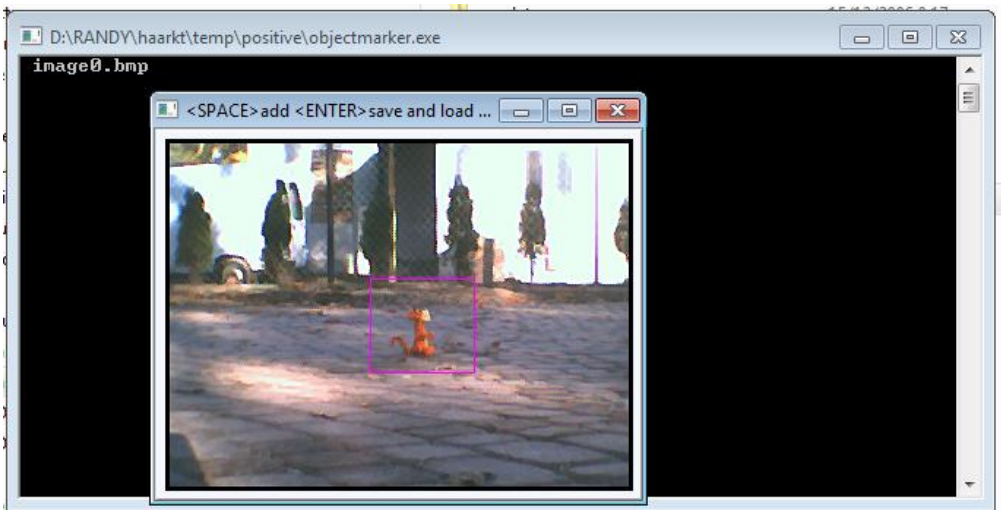
Sample negatif adalah gambar yang tidak mengandung obyek yang akan dideteksi. Seperti gambar pegunungan, mobil dsb. Masukkan sample positif pada 1 direktori, misalnya positiveSample/rawdata. Sedangkan sample negatif, dimasukkan pada /negativeSample. Catatan: file gambar harus file *.bmp.

2.9.1 Membuat infofile.txt untuk Sample Negative

Gunakan create_list.bat pada folder /negativeSample untuk mencatatkan nama file sample negative pada infofile.txt.

2.9.2 Membuat info.txt untuk Sample Positive

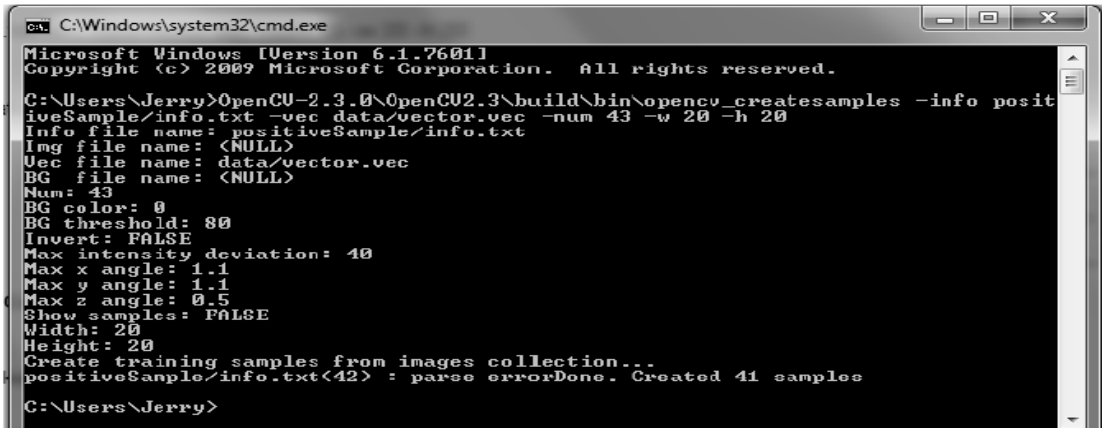
Kemudian jalankan program objectmarker.exe pada folder positiveSample. Ketika program ini dijalankan, maka akan muncul satu per satu file dari sample positiveSample. Kemudian tandai obyek yang dimaksud dari gambar tersebut dengan menggerakkan kursor mouse membentuk sebuah box persegi panjang. Kemudian tekan spasi untuk menambahkan box tersebut, lalu tekan enter untuk beralih pada file gambar berikutnya. Kalau berhasil, maka info.txt akan berisi data gambar



Gambar 2.8 Objectmarker.exe

2.9.3 Membuat file vector.vec dari Sample Positive

Lalu kita gunakan tool createsamples.exe untuk mengubah obyek gambar ke file vec. Jalankan perintah berikut pada dos command.



Gambar 2.9 tool createsamples.exe pada dos command

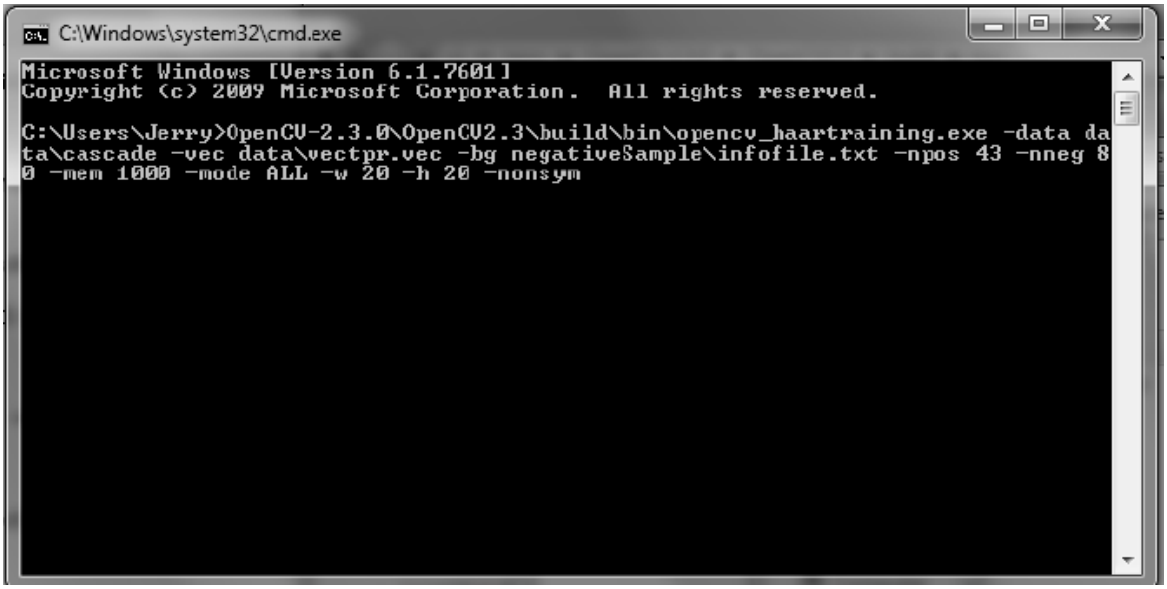
Tabel 2.2 Keterangan parameter:

Parameter	Explanation
Info<collection of filename>	Lokasi tempat image berada
Num<sample_width>	Jumlah positive image yang di training
W<sample_width>	Panjang image
H<sample_height>	Lebar dari image
Vec<vec_file_name>	Binary file yang menampung hasil olahan dari positive image

Maka bila hal ini telah berhasil dijalankan maka akan muncul pada folder /data, file vector.vec

2.9.4 Memulai HaarTraining

Setelah kita punya file vector.vec maka kita mulai haartraining. Jalankan program haartraining.exe di dos command:



Gambar 2.10 Haartraining.exe di dos command

Tabel 2.3 Keterangan parameter:

Parameter	Explanation
Data <dir_name>	Tempat folder cascade of classifier akan disimpan
Voo <voo_filename>	Informasi nama image
Bg <background_filename>	Mengandung informasi negatif image
Nneg <number_negativ_samples>	Menunjukan banyaknya jumlah negatif image>
Npos <number_positive_samples>	Menunjukan banyaknya positive image
Men <memory_in_MB>	Banyaknya memory yang dipakai selama proses pembuatan cascade of classifier
Nonsym	Untuk memastikan datanya bukan simetrik
Mode ALL	Untuk memastikan parameter tidak ditulis dalam kondisi default
W <sample_width>	Tinggi dan lebar image
H <sample_height>	

lalu pada folder tools/temp/data/cascade maka akan muncul folder mulai dari 0 sampai N. Kemudian copy semua folder tersebut pada tools/cascade2xml/data.

2.9.5 Membuat file *.xml

Jalankan haarconv.exe pada folder /cascade2xml di dos command sebagai berikut:

Jika berhasil, maka akan muncul file output.xml pada folder /cascade2xml.

```
<opencv_storage>
<haarcascade_eye type_id="opencv-haar-classifier">
  <size>
      24 24</size>
  <stages>
      <_>
        <!-- stage 0 -->
        <trees>
          <_>
            <!-- tree 0 -->
            <_>
              <!-- root node -->
              <feature>
                <rects>
                  <_>
                    8 12 3 8 -1.</_>
                  <_>
                    8 16 3 4 2.</_></rects>
              <tilted>0</tilted></feature>
            <threshold>0.0273259896785021</threshold>
            <left_val>-0.9060062170028687</left_val>
            <right_node>1</right_node></_>
            <_>
              <!-- node 1 -->
              <feature>
                <rects>
                  <_>
                    5 11 8 9 -1.</_>
                  <_>
                    7 11 4 9 2.</_></rects>
              <tilted>0</tilted></feature>
            <threshold>-7.056845817714929 6e-03</threshold>
            <left_val>0.9338570833206177</left_val>
            <right_val>-0.4585995972156525</right_val></_></_>
```

Tabel 2.4 Keterangan

Threshold	Merupakan nilai amabang batas suatu objek apakah objek mata tertutup (mengantuk) atau tidak
Left_val	Nilai dari ambang batas sebelah kiri (atas)
Right_val	Nilai dari ambang batas sebelah kanan (bawah)
Stage_threshold	Nilai yang dipilih untuk ambang batas mata tertutup
Size	Dengan size untuk image 24x24

2.10 DFD (Data Flow Diagram)

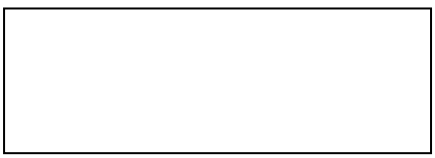
Data Flow Diagram (DFD) adalah sebuah diagram yang menggunakan notasi untuk menggambarkan arus data dari sistem. Penggunaan notasi ini sangat membantu dalam komunikasi dengan pemakai sistem untuk memahami sistem secara logika. Penggunaan DFD sebagai modelling tools dipopulerkan oleh Tom De Marco (1978) dan Gane & Sarson (1979) dengan menggunakan pendekatan metoda analisis sistem terstruktur (*structured system analysis methode*) [3].

2.9.1 Simbol-Simbol Pada DFD

Berikut ini merupakan beberapa simbol yang digunakan untuk pemodelan menggunakan DFD [3]:

1. *Eksternal Entity*

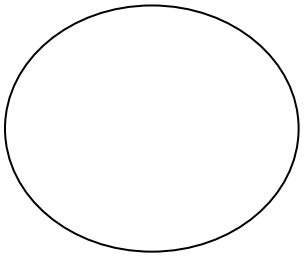
Entity bisa berupa orang atau organisasi yang berada diluar sistem yang memberikan data kepada sistem atau yang menerima informasi dari sistem. Berikut ini adalah gambar eksternal entity pada DFD :



Gambar 2.11 Eksternal Entity

2. Proses

Menggambarkan apa yang dilakukan oleh sistem. Berfungsi mentransformasikan satu atau beberapa data masukan menjadi satu atau beberapadata keluaran sesuai dengan spesifikasi yang diinginkan. Berikut ini adalah gambar proses pada DFD :



Gambar 2.12 Proses

3. *Data Flow*

Data Flow menggambarkan aliran data dari suatu entity ke entity lainnya. Arah panah menggambarkan aliran data. Aliran data bisa terjadi antara :

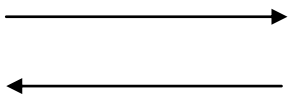
a. Dua proses yang berurutan

b. Dari data store ke proses dan sebaliknya.

c. Dari source ke proses

d. Dari proses ke link.

Berikut ini adalah gambar *data flow* pada DFD :



Gambar 2.13 Data Flow

4. *Data Store*

Menggambarkan tempat penyimpanan data. Proses dapat mengambil data dari atau memberi data ke store. Berikut ini adalah gambar *data store* pada DFD :



Gambar 2.14 Data Store

2.11 Pengujian

Ada dua macam pengujian yang akan dilakukan, yaitu pengujian performansi dan black box testing. Pengujian performansi yaitu dengan melakukan percobaan-percobaan dalam kondisi-kondisi tertentu seperti pengaruh posisi wajah pada saat pendeteksian, jarak wajah terhadap webcam, dan pengaruh pencahayaan terhadap deteksi mata. Pengujian pada black box testing yaitu menemukan kesalahan yang terdapat pada program.

2.11.1 Pengujian Performansi

Pengujian performansi dilakukan dengan serangkaian percobaan-percobaan dalam kondisi-kondisi tertentu yang dapat mempengaruhi keefektifan kinerja sistem pendeteksian mata mengantuk.

2.11.2 Pengujian Black Box

Black Box Testing ini bertujuan untuk menunjukkan fungsi perangkat lunak tentang cara beroperasinya, apakah pemasukan data keluaran telah berjalan sebagaimana yang diharapkan dan apakah informasi yang disimpan secara eksternal selalu dijaga kemutakhirannya. Tehnik pengujian black-box berfokus pada domain informasi dari perangkat lunak, dengan melakukan test case dengan menpartisi domain input dari suatu program dengan cara yang memberikan cakupan pengujian yang mendalam. Metode pengujian graph-based mengeksplorasi hubungan antara dan tingkah laku objek-objek program. Partisi ekivalensi membagi domain input ke dalam kelas data yang mungkin untuk melakukan fungsi perangkat lunak tertentu. Analisis nilai batas memeriksaa kemampuan program untuk menangani data pada batas yang dapat diterima. Metode pengujian yang terspesialisasi meliputi sejumlah luas kemampuan perangkat lunak dan area aplikasi. GUI, arsitektur client/ server, dokumentasi dan

fasilitas help dan sistem real time masing-masing membutuhkan pedoman dan tehnik khusus untuk pengujian perangkat lunak [9].

2.12 Software Pendukung

2.12.1 Bahasa Pemrograman C#

Bahasa C# adalah sebuah bahasa pemrograman modern yang bersifat general-purpose, berorientasi objek, yang dapat digunakan untuk membuat program di atas arsitektur Microsoft .NET Framework. Bahasa C# ini memiliki kemiripan dengan bahasa Java, C dan C++ .

Bahasa pemrograman ini dikembangkan oleh sebuah tim pengembang di Microsoft yang dipimpin oleh Anders Hejlsberg, seorang yang telah lama malang melintang di dunia pengembangan bahasa pemrograman karena memang ialah yang membuat Borland Turbo Pascal, Borland Delphi, dan juga Microsoft J++.

Kini, C# telah distandarisasi oleh European Computer Manufacturer Association (ECMA) dan juga International Organization for Standardization (ISO) yang mendukung beberapa fitur baru semacam Language Integrated Query (LINQ) dan lain-lainnya.

2.12.2 Microsoft Visual Studio 2010

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).

Microsoft kini merilis Microsoft Visual Studio 2010 dan Microsoft .NET Framework 4.0.Dua *tool* yang paling banyak digunakan untuk mengembangkan program di atas Windows, Windows Mobile, Web (ASP.NET), Silverlight, dan beberapa *platform* lainnya.

2.12.3 OpenCV 2.3.1

OpenCV 2.3.1 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.

Semua kode program ditulis dalam C/C++/C# bahasa dan di-*compile* dengan gcc/g++/gsharp.Suatu pengetahuan yang umum tentang C *programming* adalah penting untuk memahami metode*programming* digunakan di OpenCV [8].