BAB 2

TINJAUAN PUSTAKA

2.1 Perangkat Lunak

2.1.1 Pengertian Perangkat Lunak

Perangkat Lunak adalah sekumpulan data elektronik yang disimpan dan diatur oleh komputer, data elektronik yang disimpan oleh komputer itu dapat berupa program atau instruksi yang akan menjalankan suatu perintah. melalui *software* atau perangkat lunak inilah suatu komputer dapat menjalankan suatu perintah. Perangkat lunak berdasarkan distribusinya dibedakan menjadi beberapa macam, yaitu Perangkat lunak berbayar dan Perangkat lunak gratis (*Freeware, free software, shareware*). [7]

2.1.2 Perkembangan Perangkat Lunak

Perkembangan perangkat lunak telah tumbuh dengan sangat pesat sejak beredarnya komputer personal dipasaran. Dengan semakin murahnya harga komputer, semakin banyak pula orang yang memilikinya dan pemakai komputer membutuhkan bermacam-macam perangkat lunak untuk lebih mendayagunakan komputernya. Secara kebetulan, banyak perusahaan-perusahaan perangkat lunak yang menanggapi permintaan ini, telah ribuan macam perangkat lunak yang

tersedia dilapangan, terutama perangkat lunak paket (*Package Software*) yaitu program jadi untuk aplikasi tertentu. [7]

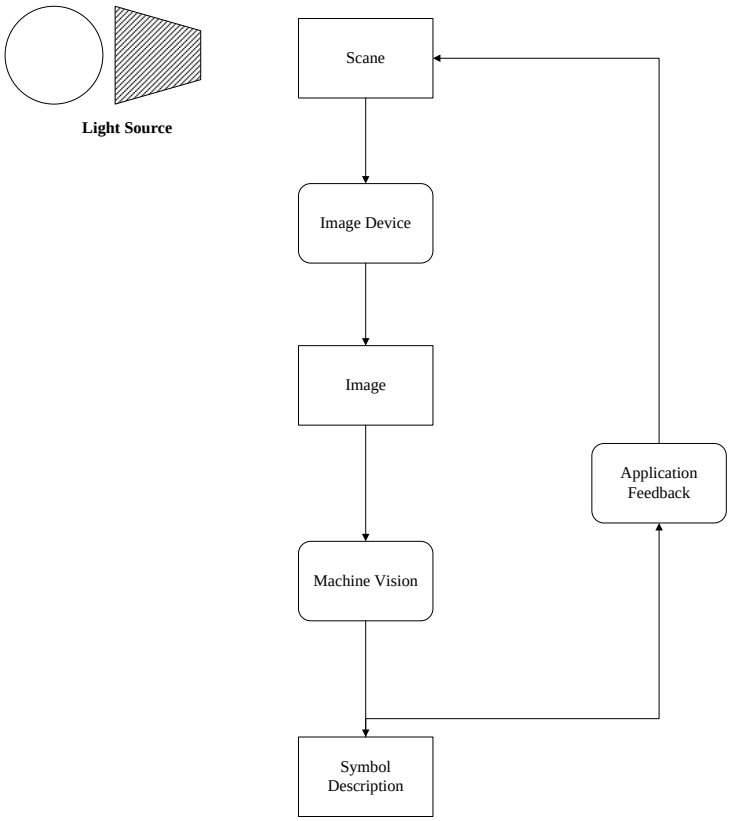
2.2 Komputer Vision

Komputer *vision* adalah sebuah disiplin ilmu yang mempelajari proses menyusun deskripsi tentang objek yang terkandung pada suatu gambar atau mengenali objek yang ada pada gambar. Komputer *vision* berusaha menterjamahkan citra menjadi deskripsi atau suatu informasi yang mempresentasikan suatu citra tersebut. [3]

Disisi lain orang-orang juga mengembangkan bagaimana suatu mesin pintar dapat memahami dan mengerti semua keadaan seperti telah dipaparkan di atas, bahkan juga dapat berkomunikasi dengan semuanya. Inilah konsep yang mendasari komputer *vision.*

2.2.1 Elemen-Elemen Komputer *Vision*

Gambar 2.1 di bawah ini adalah struktur yang mendasari elemen-elemen suatu mesin *vision.*



Gambar 2.1 Struktur Komputer *Vision*

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 ciri-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.

Dari paparan di atas ada dua elemen yang mendasari suatu sistem vision, yaitu *Image Processing,* dan *Pattern Classification*.[3]

2.3 Image Processing

Pengolahan citra atau *Image Processing* adalah bidang tersendiri yang sudah cukup berkembang sejak orang mengerti bahwa komputer tidak hanya dapat menangani data teks, tetapi juga citra. Teknik pengolahan citra biasanya digunakan untuk melakukan transformasi dari suatu citra kepada citra lain, sementara tugas perbaikan citra untu mendapatkan suatu informasi dari citra tersebut merupakan tugas manusia melalui penyususanan algoritmanya. [4]

Berikut adalah bagian proses dari *image processing*:

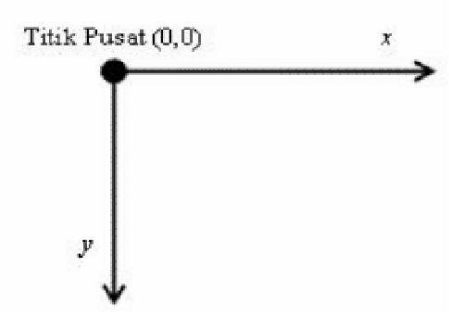


Gambar 2.2 Fase *Image Processing*

2.3.1 Pengertian Citra Digital

Secara umum, pengolahan citra digital menunjuk pada pemrosesan gambar dua dimensi menggunakan komputer. Dalam konteks yang lebih luas, pengolahan citra digital mengacu pada pemrosesan setiap data dua dimensi. Citra digital merupakan sebuah larik (*array*) yang berisi nilai-nilai *real* maupun komplek yang direpresentasikan dengan deretan bit tertentu. [1]

Umumnya citra dibentuk dari persegi empat yang teratur sehingga jarak *horizontal* dan *vertikal* antara piksel satu dengan yang lain adalah sama pada seluruh bagian citra. Indeks X bergerak ke kanan dan indeks Y bergerak ke bawah. Untuk menunjukkan koordinat (m-1,n-1) digunakan posisi kanan bawah dalam citra berukuran m x n piksel. Hal ini berlawanan untuk arah *vertikal* dan *horizontal* yang berlaku pada sistem grafik dalam matematika. Gambar 2.3 menunjukkan sistem penggambaran koordinat pada suatu citra digital.



Gambar 2.3 Koordinat Pada Citra Digital

Untuk menunjukkan tingkat intensitas cahaya suatu piksel, seringkali digunakan bilangan bulat dengan lebar selang antara 0-255, dimana 0 untuk warna hitam dan 255 untuk warna putih. Sistem visual manusia dapat membedakan ratusan ribu warna tetapi hanya dapat membedakan 100 shade keabuan. [8]

2.3.2 Teknik – Teknik Pengolahan Citra

Secara umum teknik pengolahan citra dibagi menjadi 3 tingkat pengolahan, yaitu sebagai berikut :

1. *Low-Level Processing*
2. *Mid-Level Processing*
3. *High-Level Processing*

Dari ketiga tingkat pengolahan citra diatas tersebut, dapat dinyatakan suatu gambaran mengenai teknik – teknik pengolahan citra digital dan macam – macamnya antara lain sebagai berikut :

1. *Image Enhancement*, berupa proses perbaiakan citra dengan meningkatkan kualitas citra baik kontras maupun kecerahan.
2. *Image Restoration*, proses memperbaiki model citra, biasanya berhubungan dengan bentuk citra yang sesuai.
3. *Color Image Processing*, suatu proses yang melibatkan citra berwarna, baik berupa image enhancement, image restoration atau yang lainnya.
4. *Wavelet dan Multiresolution Processing*, merupakam suatu proses yang menyatakan citra dalam beberapa resolusi.
5. *Image Compression*, merupakan proses yang digunakan untuk mengubah ukuran data pada citra.
6. *Morpholohical Processing*, proses untuk mendapatkan nformasi yang menyatakan deskripsi dari suatu bentuk pada citra.
7. *Segmentataion*, proses untuk membedakan atau memisahkan objek – objek yang ada dalam suatu citra, seperti memisahkan objek dengan latar belakangnya.
8. *Object Recognition*, suatu proses yang dialakukan untuk mengenali objek – objek apa saja yang ada dalam suatu citra.

2.3.3 Jenis – Jenis Citra

2.3.3.1Citra Biner

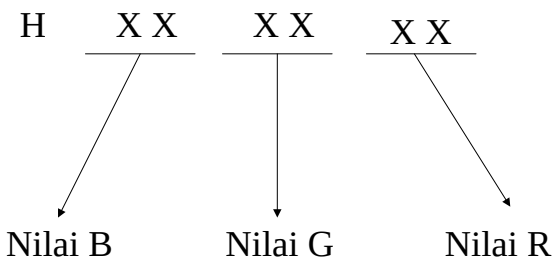
Citra *biner* adalah citra digital yang hanya memiliki dua kemungkinan nilai piksel yaitu hitam dan putih. Citra *biner* disebut juga citra B & W (*Black and White*) atau citra monokrom. Hanya dibutuhkan 1 bit untuk mewakili nilai setiap piksel dari citra *biner*. [1]



Gambar 2.4 Contoh Citra Biner

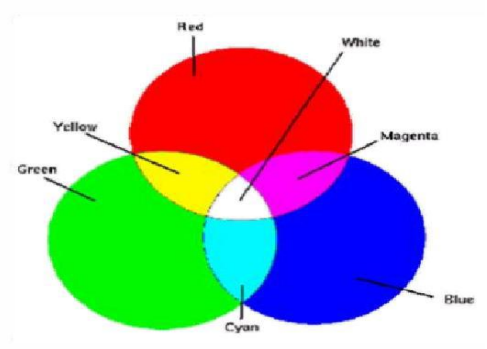
2.3.3.2 Citra Warna RGB (Red, Green, Blue)

Dalam pengolahan citra, warna dipresentasikan dalam *range* nilai H000000 sampai 0x00ffffff. Warna hitam adalah 0x00000000 dan warna putih adalah Hffffff. Definisi nilai warna di atas adalah sebagai berikut. variabel H menyatakan angka dibelakangnya adalah *hexadecimal*. [8]



Gambar 2.5 Definisi nilai warna pada citra

Dengan demikian *range* warna yang digunakan adalah (2 8) (2 8) (2 8) = 2 24 (atau yang dikenal dengan istilah *True Colour* pada *Windows*). Nilai warna yang digunakan di atas merupakan gabungan warna cahaya merah, hijau dan biru seperti yang terlihat pada Gambar 2-6.



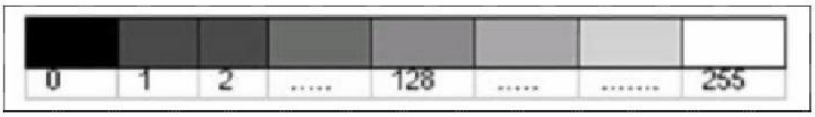
Gambar 2.6 Komposisi Warna RGB

Dengan demikian, untuk menentukan suatu warna yang bukan warna dasar digunakan gabungan skala kecerahan dari setiap warnanya.

2.3.3.3 Citra Grayscale

Citra *grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pikselnya, dengan kata lain nilai bagian *RED = GREEN = BLUE*. Nilai tersebut digunakan untuk menunjukan tingkat intensitas. Warna yang dimiliki adalah warna dari hitam, keabuan, dan putih. Tingkatan keabuan disini merupakan warna abu dengan berbagai tingkatan dari hitam hingga mendekati putih. [1]

Model warna ini menyimpan informasi warna sebesar 8 bit. Jadi warna yang bisa ditampilkan pada mode warna *grayscale* berjumlah sampai 256. Skala perhitungan numeriknya bisa dengan pendekatan nilai *brightness* yang berskala 0 sampai 255 seperti pada Gambar 2-7.



Gambar 2.7 Komposisi Warna Grayscale

Greyscaling adalah teknik yang digunakan untuk mengubah citra berwana menjadi bentuk *grayscale*. Pengubahan dari citra berwarna ke bentuk *grayscale* dapat dilakukan dengan melakukan beberapa pesamaan sebagai berikut :

a. Mengubah sistem *YUV* (system warna *NTSC*), yaitu dengan cara mengambil komponen Y (iluminasi). Komponen Y

sendiri dapat diperoleh dari sistem warna *RGB* dengan konversi :

GRAY=Y=0.299*R+0.587*G+0.114*B

…….……(2.1)

b. Dengan merata – rata setiap komponen warna pada *RGB* memiliki persamaan sebaga beriut :

$$(x, y, f(x, y)) \rightarrow (x, y, I(x, y))$$

$$I(i, j) = \frac{R(i, j) + G(i, j) + B(i, j)}{3} \quad \ldots\ldots(2.2)$$

c. Menggunaan nilai masimal dari komponen *RGB* meiliki persamaan sebagai berikut :

Gray=MAX{RGB} ….(2.3)

Cara yang sering digunakan dalam pemrosesan warna *RGB* adalah cara a dan b. berikut adalah tabel hasil konversi dari *image RGB* ke *image Grayscle* :

Tabel 2.1 hasil konversi *RGB ke Grayscale*

Source Image RGB (Data Type)	**Resultan Image (Data Type)**
8 bit per channel	Grayscale :1, 8, 16 bit per cpixel
16 bit per channel	Grayscale :1, 8, 16 bit per cpixel
32 bit per channel	Grayscale :1, 8, 16 bit per cpixel

Contoh dari Hasil dari proses *grayscaling* ditunjukkan pada Gambar 2.8 dibawah ini :

(a) (b)



Gambar 2.8 Citra Grayscale (a) dan Citra RGB (b)

2.3.4 Teknik Pengambilan Citra

Ada beberapa teknik pengambilan citra digital yang bisa dilakukan, antara lain dengan menggunakan kamera digital (*webcam*)

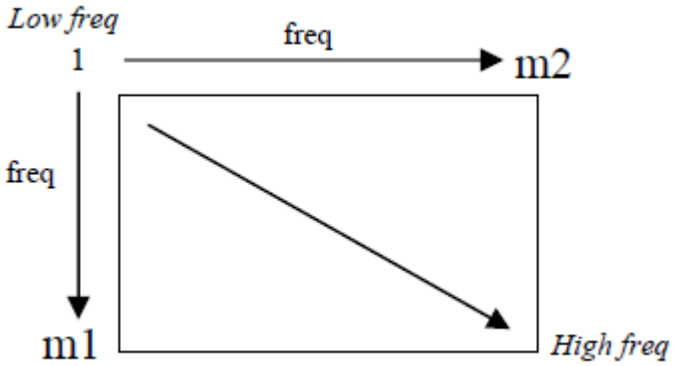
atau menggunakan *scanner*, atau *barcode scanner*. Hasil dari citra dengan menggunakan teknik kamera atau *scanner* berupa citra *raster* atau citra dengan model matriks. Teknik pengambilan citra selain membuthkan peralatan input, membutuhkan juga suatu *card* yang disebut *frame grabber* yang berupa rangkaian untuk mengolah citra secara *hardware*.

Selain itu pengambilan citra juga bisa dilakukan dengan menggunakan teknik grafika komputer, yaitu dengan membentuk citra komputer sesuai dengan teknik pengambilan citra. Hasil dari citra dengan menggunakan teknik grafika komputer ini bisa berupa citra *raster* atau bisa berupa vektor. [8]

2.3.5 Filtering

Filtering adalah suatu proses dimana diambil sebagian sinyal dari frekwensi tertentu, dan membuang sinyal pada frekwensi yang lain. *Filtering* pada citra juga menggunakan prinsip yang sama, yaitu mengambil fungsi citra pada frekwensi-frekwensi tertentu dan membuang fungsi citra pada frekwensi-frekwensi tertentu. [8]

Berdasarkan sifat *transformasi fourier* dari suatu citra dan format koordinat frekwensi seperti gambar berikut ini:



Gambar 2.9 Format Koordinat frekwensi Pada Citra

2.3.5.1 Teknik-Teknik Filtering

1. *Low Pass Filter*.

low pass filter adalah proses *filte*r yang mengambil citra dengan gradiasi intensitas yang halus dan perbedaan intensitas yang tinggi akan dikurangi atau dibuang. Hal ini banyak digunakan untuk reduksi *noise* dan proses *blur*. [8]

Ciri-ciri dari fungsi *low-pass filter* adalah sebagai berikut:

$$\sum_j \sum_i H(i,j) = 1 \qquad \text{.....(2.4)}$$

2. *High Pass Filter*.

High pass filter adalah proses *filter* yang mengambil citra dengan gradiasi intensitas yang tinggi dan perbedaan intensitas yang rendah akan dikurangi atau dibuang. Hal ini banyak digunakan untuk

menentukan garis tepi (*edge*) atau sketsa dari citra. Ciri-ciri dari fungsi *High-pass filter* adalah sebagai berikut:

$$\sum_j \quad \sum_i \; H(i,j) = 0 \qquad \ldots(2.5)$$

3. *Band Stop Filter*

Bila ingin mempertahankan gradiasi dan bentuk, dengan tetap mengurangi banyaknya bidang frekwensi (*bandwidth*) dan membuang sinyal yang tidak perlu maka frekwensi rendah dan frekwensi tinggi dipertahankan, sedangkan frekwensi tengahan dibuang atau dinamakan dengan *Band Stop Filter*.

2.4 Segmentasi Citra

Segmenatsi citra merupakan teknik untuk membagi suatu citra menjadi beberapa daerah *(region)* dimana setiap daerah memiliki kemiripan atribut.

2.4.1 Thresholding

Thresholding adalah suatu proses dalam pengolahan citra yang berfungsi untuk memisahkan *foreground* dan *background* dari suatu *image*. Proses *tresholding* dilakukan dengan cara melihat perbedaan intensitas warna dari suatu *image*. *Input* dari proses *tresholding* adalah *grayscale* image atau *color image* sedangkan *output* dari proses

tresholding adalah *binary image* dimana piksel hitam untuk mewakili background dan piksel putih mewakili *foreground*, atau sebaliknya.

Tresholding digunakan untuk mengatur jumlah derajat keabuan yang ada pada suatu citra. Dengan menggunakan *tresholding* maka derajat keabuan dapat diubah seseuai keinginan. *Tresholding* dapat juga digunakan untuk mengubah intensitas piksel menjadi salah satu dari dua nilai, a1 atau a2. Hasil dari proses *thresholding* ditentukan oleh suatu parameter yang disebut nilai *threshold.* [1]

Proses *thresholding* menggunakan fungsi sebagai berikut :

$$(x, y, f(x, y)) \rightarrow (x, y, g(x, y))$$

$$g(x,y) = \begin{cases} a_1 & , \quad f(x,y) < T \\ a_2 & , \quad f(x,y) \geq T \end{cases} \quad \ldots(2.6)$$

Keterangan :

f(x,y) = Nilai intensitas yang lama

g(x,y) = Nilai intensitas yang baru

T = Nilai *Threshold*

Jika a1=0 dan a2=1, maka proses thresholding mengubah citra *greyscale* menjadi citra *biner*.

2.5 Metode Normalized Sum-Squared Differences (NSSD)

Metode Normalized Sum-Squared Differences (NSSD) merupakan bentuk khusus dari *Sum-Squared Differences* (SSD). *Metode Normalized Sum-Squared Differences* (NSSD) didefinisikan sebagai perbedaan nilai intensitas yang dikuadratkan antara *frame* saat itu dengan *frame background* dalam suatu daerah yang ditentukan (*Windows Detections*), yang kemudian di normalisasi [9,10].

Untuk melakukan identifikasi dengan menggunakan metode *Metode Normalized Sum-Squared Differences* (NSSD), diperlukan *background image* untuk dibandingkan dengan *fram*e yang akan dideteksi. *Background image* yang digunakan merupakan *image* lajur dalam keadaan kosong atau tidak ada kendaraan yang lewat pada lajur yang akan dideteksi. *Windows Detections* atau ROI (*Region of Interest*) diset pada lajur yang akan dideteksi setelah dilakukan proses *grayscale* sebelumnya. Kemudian, semua nilai piksel yang terdapat dalam *Windows Detections* dijumlahkan.

Hal yang sama juga dilakukan pada setiap *frame* video. Maksud dari *frame* video adalah sekumpulan *image* yang ditampilkan secara berurutan sehingga membentuk suatu gambar bergerak. Sehingga proses *grayscale*, setting *Windows Detections*, dan menjumlahkan nilai *piksel* dilakukan pada masing-masing *frame*. Posisi *detection window*

pada *frame* dan pada *background image* harus sama karena pada posisi inilah yang akan dilakukan pembandingan.

Dengan adanya *Windows Detections* memudahkan proses perhitungan jumlah nilai piksel yang dilakukan oleh komputer. Bilamana tidak menggunakan *Windows Detections,* komputer akan melakukan perhitungan pada *frame* secara keseluruhan. Sedangkan dengan *Windows Detections,* komputer cukup menghitung pada *Windows Detections* saja. Setelah mendapatkan jumlah nilai piksel dari *background image* dan *frame*, nilai-nilai tersebut dicari selisihnya dengan cara dikurangkan (*subtract*). Selisih dari jumlah nilai *background image* dan *frame* dikuadratkan untuk menghindari nilai negatif. Nilai negatif terjadi bilamana warna dari obyek pada *frame* lebih gelap daripada *background image*. Sebab warna gelap memiliki nilai pixel yang kecil.

Metode NSSD dapat dirumuskan dalam persamaan berikut :

$$NSSD = \frac{\sum_{i}^{N} (I_C(i) - I_B(i))^2}{N} \qquad \ldots(2.9)$$

Keterangan:

IC(i) = intensitas *frame* pada piksel (i)

IB(i) = intensitas *background* pada piksel (i)

Q = *detection window*/ROI (*Region of Interest*)

N = jumlah piksel pada area Q

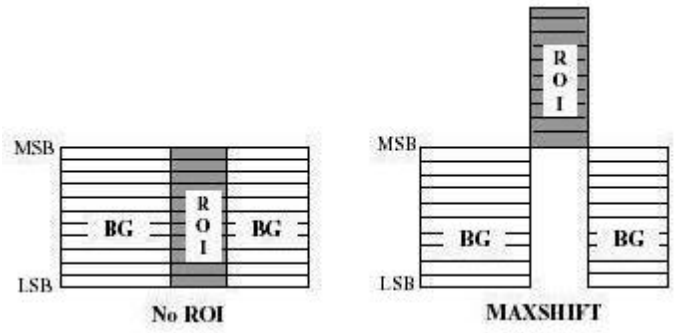
Perhitungan NSSD dilakukan pada setiap frame video yang ditampilkan, sehingga jumlah data yang didapat sesuai dengan jumlah frame yang ditampilkan.

2.5.1 Windows Detections (ROI)

Windows Detections (ROI) memungkinkan dilakukannya pengkodean secara berbeda pada area tertentu dari citra digital, sehingga mempunyai kualitas yang lebih baik dari area sekitarnya (*background*). Fitur ini menjadi sangat penting, bila terdapat bagian tertentu dari citra digital yang dirasakan lebih penting dari bagian yang lainnya.[9,10]

Prinsip utama pada pengkodean *Windows Detections* (ROI) ini adalah dengan menggeser *bitplane* dari koefisien yang dipilih sebagai *Windows Detections* (ROI), sehingga menempati posisi yang lebih tinggi daripada *bitplane* sekitarnya (*background*). Karena menggunakan metode *Maxshift*, pergeserannya dilakukan sampai batas maksimal, sehingga seluruh *bitplane* dari koefisien *Windows Detections* (ROI) berada di atas *bitplane* dari area sekitarnya. Hal ini menyebabkan, pengkodean pada *Windows Detections* (ROI) akan

menghasilkan area dengan kualitas maksimal dibandingkan area sekitarnya.



Gambar 2-10 Windows Detection (ROI)

Untuk melakukan *Windows Detections* (ROI), perlu diidentifikasi koefisien-koefisien yang termasuk dalam *Windows Detections* (ROI). Hal tersebut dimaksudkan untuk menghasilkan sebuah *Windows Detections* (ROI) *mask*, yang merupakan indikator dari koefisien yang termasuk *Windows Detections* (ROI) atau bukan. Umumnya yang digunakan sebagai penanda merupakan sebuah angka biner, yang dimiliki oleh piksel-piksel yang termasuk area *Windows Detections* (ROI).

2.6 Pemodelan Data

Pemodelan sistem memainkan peranan yang penting dalam pengembangan sistem. Pemodelan data kadang-kadang disebut pemodelan *database* karena model data kadang-kadang

diimplementasikan sebagai sebuah *database*. Pemodelan data dapat digambarkan dengan ERD (*entity Relationship Diagram*).

2.6.1 Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) merupakan suatu diagram yang digunakan untuk menghubungkan antar elemen (*relational Condition*), dimana pada tahap selanjutnya dapat diimplementasikan kedalam bentuk tabel relasi.

Entity Relationship Diagram (ERD) menggunakan sejumlah notasi dan simbol untuk menggambarkan struktur dan hubungan antar data, pada dasarnya ada 3 macam simbol yang digunakan, yaitu :

1. *Entity*

Adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai dan sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat.

2. Atribut

Elemen dari sebuah *entity* yang berfungsi mendeskripsikan karakter *entity*.

3. Hubungan

Sebagaimana halnya *entity* maka dalam hubungan pun harus dibedakan antara hubungan atau bentuk hubungan antara *entity* dengan isi dari hubungan itu sendiri.

Beberapa macam hubungan antar relasi, antara lain :

1. Satu Ke Satu (*One to One*)

   Bentuk relasi antara satu entitas dengan jumlah satu ke entitas dengan jumlah yang sama.

2. Satu Ke Banyak (*One to Many*)

   Bentuk relasi dari entitas dengan jumlah satu ke entitas lain yang berjumlah lebih dari satu (Entitas dengan banyak alternatif tujuan).

3. Banyak ke Banyak (*Many to Many*)

   Bentuk relasi yang mendeskripsikan permasalahan yang komplek yaitu hubungan antara entitas yang berjumlah lebih dari satu dengan entitas dengan jumlah yang sama.

2.6.2 Flow Map

Bagan alir atau *flow map* adalah bagan yang menunjukan alir didalam program atau prosedur sistem secara logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan dokumentasi.

Bagan alir dokumen atau sering disebut juga *Flow Map* merupakan bagan alir yang menunjukan arus dari aliran formulir termasuk tembusan-tembusannya.

Berikut adalah keterangan dari simbol-simbol yang biasa digunakan dalam pembuatan bagan alir atau *Flow map* :

1. Arah Aliran Dokumen

   Menunjukan Arah aliran dokumen antar bagian yang terkait dalam suatu sistem. Dapat dari sistem ke luar sistem, dari luar sistem ke sistem lain dan bagian luar lain.

2. Dokumen

Menunjukan dokumen *input*, *output* pada proses manual dan proses komputer.

3. Penyimpanan sementara

Menunjukan media penyimpanan sementara dari data informasi file pada proses manual. Dokumen dapat disimpan dalam lemari arsip.

4. Penyimpanan (*Database*)

Menunjukan media penyimpanan dari data informasi file. Dokumen dapat disimpan dalam *database*.

2.6.3 Diagram Konteks

Diagram konteks merupakan level tertinggi dari DFD yang menggambarkan seluruh *input* ke sistem atau *output* dari sistem. Yang digambarkan dengan lingkaran tunggal yang mewakili seluruh sistem. Sistem dibatasi oleh *boundary* (dapat digambarkan dengan garis putus). Dalam diagram konteks hanya ada satu proses.

2.6.4 Data Flow Diagram (DFD)

Data Flow Diagram atau Diagram Alir Data merupakan diagram alir yang dipresentasikan menggunakan lambang – lambang tertentu. Penulisan suatu program akan menjadi lebih mudah dalam pelaksanaan dengan adanya diagram alir data. Keuntungan menggunakan *Data Flow Diagram* (DFD) adalah supaya mempermudah pemakai (*User)* yang kurang menguasai bidang

komputer untuk lebih mengerti sistem yang akan dikembangkan atau dikerjakan.

DFD menggunakan simbol-simbol untuk menerangkannya, yaitu:

1. *External Entity* (Kesatuan Luar)

External entity (Kesatuan luar) merupakan kesatuan *(entity)* di lingkungan luar sistem yang dapat berupa orang, organisasi atau sistem lainnya yang berada dilingkungan luar sistem yang akan memberikan (*input*) atau menerima (*output*) dari sistem. Yang disimbolkan dengan kotak.

2. *Data flow* (Arus Data)

Data flow (Arus data) mengalir diantara *Process* (Proses), simpanan data *(data store)*, dan kesatuan luar *(external entity).* Arus data ini menunjukkan arus dari data yang dapat berupa masukan untuk sistem atau hasil dari proses sistem. Yang disimbolkan dengan anak panah.

3. *Process* (Proses)

Process (Proses) adalah kegiatan atau kerja yang dilakukan oleh orang, mesin, atau komputer, untuk *Physical Data flow Diagram* (PDFD). Atau merupakan suatu proses yang hanya menunjukkan proses

dari komputer, untuk *Logical Data Flow Diagram* (LDFD). Yang disimbolkan dengan lingkaran atau dengan simbol empat persegi panjang tegak dengan sudut-sudutnya yang tumpul.

4. *Data Storage* (Penyimpanan Data)

Simpan data (*data store*) merupakan tempat penyimpanan data. yang disimbolkan dengan sepasang garis *horizontal* yang tanpa tutup diujungnya.

2.6.5 Kamus Data (Data Dictionary)

kamus data merupakan sebuah daftar yang terorganisasi dari elemen data yang berhubungan dengan sistem, dengan definisi yang teliti sehingga pemakai dan analisis sistem akan memiliki pemahaman yang umum mengenai *input, output*, komponen penyimpanan serta *kalkulasi intermediate*.[14]

Pendefinisian data tersebut dilakukan dengan menggunakan notasi yang umum digunakan dalam menganalisa sistem yaitu dengan menggunakan sejumlah simbol. Kamus data biasanya dipelihara secara otomatis oleh sistem manajemen *database.*

Cara mendefinisikan kamus data yaitu :

1. Menggambarkan arti aliran data atau penyimpanan yang ditunjuk dalam DFD.
2. Menggabungkan komponen dari kumpulan data yang mengalir yaitu kumpulan komponen yang mungkin bisa dipecah lagi menjadi data *elementer*.
3. Menggambarkan data yang tersimpan.
4. Menentukan nilai dibagian elementer dari informasi yang relevan di DFD dan data *store*nya

2.7 Basis Data

2.7.1 Pengertian Basis Data

Basis Data adalah kumpulan terorganisasi dari data-data yang berhubungan sedemikian rupa sehingga mudah disimpan, dimanipulasi, serta dipanggil kembali oleh pengguna. [14]

Keuntungan Sistem basis Data:

1. Kecepatan dan Kumudahan *(Speed)*

2. Efisiensi Ruang Penyimpanan *(Space)*

3. Keakuratan *(Accuracy)*

4. Ketersediaan *(Availability)*

5. Kelengkapan *(Completeness)*

6. Keamanan *(Security)*

7. Kebersamaan *(Sharability)*

Komponen utama Basis Data (*Database*) antara lain :

1. Perangkat Keras (*Hardware*)

Yaitu Sebuah Komputer yang sudah berbentuk PC.

2. Sistem Operasi (*Software*)

Yaitu suatu bahasa pemrograman untuk melayani perintah – perintah *user.*

3. Data

Yaitu data yang bersifat terpadu dan berbagi.

4. Aplikasi

Yaitu pengolahan perangkat lunak atau sistem informasi yang bersifat opsional.

5. Pemakai (*User*)

Yaitu orang yang memakai atau menggunakan basis data.

Pengguna sistem basis data dapat melakukan berbagai operasi, antara lain :

1. Menambahkan *file* baru ke sistem basis data;

2. Mengosongkan berkas;

3. Menyisipkan data ke suatu berkas;

4. Mengambil data yang terdapat pada suatu berkas;

5. Mengubah data pada suatu berkas;

6. Menghapus data pada suatu berkas;

7. Menyajikan suatu informasi yang diambil dari sejumlah berkas.

2.7.2 Sistem Pengelola Basis Data (*Database Management System*)

Kebanyakan dari database tergantung pada *Database Management Sistem* (DBMS) untuk mengelola data yang tersimpan dalam sistem *database* dan menyiapkan data agar tersedia bagi pengguna yang ingin mengakses informasi tertentu. Sebuah *Database Management Sistem* (DBMS) terdiri atas satu perangkat *server* dan *client* yang komprehensif (meliputi banyak hal) yang mendukung berbagai macam tugas-tugas administratif dan yang berhubungan dengan data. Beberapa perangkat *Database Management Sistem* (DBMS) menyediakan beberapa tipe perangkat *client*, yang mengijinkan *user* untuk berinteraksi secara langsung dengan data yang tersimpan dalam database. *Database Management Sistem* (DBMS)

dapat menjadi alternatif penggunaan secara khusus untuk aplikasi, semisal penyimpanan data dalam *file* dan menulis kode aplikasi yang spesifik untuk pengaturannya.[14]

Penggunaan *Database Management Sistem* (DBMS) untuk suatu aplikasi tergantung pada kemampuan dan dukungan *Database Management Sistem* (DBMS) yang beroperasi secara efisien. Sehingga agar bisa menggunakan *Database Management Sistem* (DBMS) dengan baik, perlu diketahui cara kerja dari *Database Management Sistem* (DBMS) tersebut. Pendekatan yang dilakukan untuk menggunakan *Database Management Sistem* (DBMS) secara baik meliputi : implementasi *Database Management Sistem* (DBMS) dan arsitektur secara mendetail untuk dapat memahami desain dari suatu basis data

2.7.3 Bahasa Basis Data

Database Management Sistem (DBMS) merupakan perantara bagi pemakai dengan basis data dalam disk. Cara berkomunikasi atau berinteraksi antara pemakai dengan basis data tersebut diatur dalam suatu bahasa khusus yang ditetapkan oleh perusahaan pembuat *Database Management Sistem* (DBMS). Bahasa tersebut dapat kita sebut sebagai bahasa basis data yang terdiri atas sejumlah perintah yang diformulasikan oleh *user* dan diproses oleh *Database Management Sistem* (DBMS) untuk melakukan suatu aksi atau pekerjaan tertentu.[14]

Bahasa yang digunakan dalam Basis Data yaitu :

1. DDL (*Data Definition Language*)

Merupakan bahasa definisi data yang digunakan untuk membuat dan *memanage* objek *database* seperti *database*, tabel dan *view*.

2. DML (*Data Manipulation Language*)

Merupakan bahasa manipulasi data yang digunakan untuk memanipulasi data pada objek *database* seperti tabel.

3. DCL (*Data Control Language*)

Merupakan bahasa yang digunakan untuk mengendalikan pengaksesan data.

2.8 Perangkat Lunak Pembangun

2.8.1 C#

C# adalah bahasa pemrograman baru yang diciptakan *Microsoft* yang digunakan oleh banyak *developer* .NET untuk mengembangkan applikas dengan *platform* .NET. kelebihan dari bahasa pemrogeaman ini adalah sebagai berikut :

1. Sederhana

C# bersifat sederhana, karena bahasa ini didasarkan kepada bahasa C dan C++. Jika anda *familiar* dengan C dan C++ atau bahkan *Java*, anda akan menemukan aspek-aspek yang begitu *familiar*, seperti *statements, expression, operators*, dan beberapa fungsi yang diadopsi langsung dari C dan C++, tetapi dengan berbagai perbaikan yang membuat bahasanya menjadi lebih sederhana.[16]

2. Object Oriented Language

C# memenuhi syarat-syarat sebagai sebuah bahasa pemrograman yang bersifat *Object Oriented*, yaitu *encapsulation, inheritance dan polymorphism*.

3. Powerfull dan *Fleksibel*

C# bisa digunakan untuk membuat berbagai macam aplikasi, seperti aplikasi pengolah kata, grapik, *spreadsheets*, atau bahkan membuat *kompiler* untuk sebuah bahasa permrograman.

4. Efisien

C# tidak memiliki terlalu banyak *keyword*, sehingga dapat mengurangi kerumitan.

5. Modular

Kode C# ditulis dengan pembagian masing *Class-Class* (*classes*) yang terdiri dari beberapa *routines* yang disebut sebagai member *methods*. *Class-Class* dan metode-metode ini dapat digunakan kembali oleh program atau aplikasi lain. Hanya dengan memberikan informasi yang dibutuhkan oleh *Class* dan metode yang dimaksud, maka kita akan dapat membuat suata kode yang dapat digunakan oleh satu atau beberapa aplikasi dan program (*reusable code*).

2.8.2 Microsoft Access

Microsoft Access merupakan anggota dari beberapa aplikasi *Microsoft Office*, selain tentunya *Microsoft Word, Microsoft Excel, dan Microsoft PowerPoint*. Aplikasi ini menggunakan mesin basis data *Microsoft Jet Database Engine*, dan juga menggunakan tampilan grafis yang intuitif sehingga memudahkan pengguna.

Microsoft Access dapat menggunakan data yang disimpan di dalam format *Microsoft Access, Microsoft Jet Database Engine, Microsoft SQL Server, Oracle Database,* atau semua kontainer basis data yang mendukung standar ODBC. Para pengguna atau *programmer* yang mahir dapat menggunakannya untuk mengembangkan perangkat lunak aplikasi yang kompleks, sementara para programer yang kurang mahir dapat menggunakannya untuk mengembangkan perangkat lunak

aplikasi yang sederhana. *Microsoft Access* juga mendukung teknik-teknik pemrograman berorientasi objek, tetapi tidak dapat digolongkan ke dalam perangkat bantu pemrograman berorientasi objek.

2.9 Perangkat Keras Pendukung

2.9.1 Kamera (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 piksel. [11]

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 mulus gambar yang dihasilkan.

Ada dua jenis kamera yang bisa digunakan, pertama adalah tipe *Web Cam* 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.



Gambar 2-11 Webcam

2.9.2 Definisi Real Time

Suatu sistem dikatakan *real time* jika dia tidak hanya mengutamakan ketepatan pelaksanaan instruksi atau tugas, tapi juga interval waktu tugas tersebut dilakukan. Dengan kata lain, sistem *real time* adalah sistem yang menggunakan *deadline*, yaitu pekerjaan harus selesai jangka waktu tertentu. Sementara itu, sistem yang tidak *real time* adalah sistem dimana tidak ada *deadline*, walaupun tentunya respons yang cepat atau performa yang tinggi tetap diharapkan. [12]

Sistem *real time* banyak digunakan dalam bermacam-macam aplikasi. Sistem *real time* tersebut ditanam didalam alat khusus seperti dikamera, mp3 players, serta di pesawat dan mobil. Sistem *real time* bisa dijumpai pada tugas-tugas yang *mission critical*, misal sistem untuk sistem pengendali reaktor nuklir atau sistem pengendali rem

mobil. Juga sering dijumpai pada peralatan medis, peralatan pabrik, peralatan untuk riset ilmiah, dan sebagainya.

Ada dua model sistem real time, yaitu *hard real time* dan *soft real time*.

1. *Hard real time* mewajibkan proses selesai dalam kurun waktu tertentu. Jika tidak, maka gagal. Misalnya adalah alat pacu jantung. Sistem harus bisa memacu detak jantung jika detak jantung sudah terdeteksi lemah.
2. *Soft real time* menerapkan adanya prioritas dalam pelaksanaan tugas dan toleransi waktu. Misalnya adalah transmisi video. Gambar bisa sampai dalam keadaan terpatah-patah, tetapi itu bisa ditolerir karena informasi yang disampaikan masih bisa dimengerti.

Sistem *Hard real time* menjamin bahwa proses waktu nyata dapat diselesaikan dalam batas waktu yang telah ditentukan. Contoh : sistem safety-critical. Beberapa sistem waktu nyata diidentifikasi sebagai sistem safety-critical, dalam skenario ini sistem waktu nyata harus merespon kejadian dalam batas waktu yang telah ditentukan maka akan terjadi bencana. Sistem manajemen penerbangan merupakan sebuah contoh sebuah sistem waktu nyata sebagai sistem safety-critical.

Sistem *Soft real time* menyediakan prioritas untuk mendahulukan proses yang menggunakan waktu nyata dari pada proses yang tidak menggunakan waktu nyata.