BAB 2
LANDASAN TEORI

2.1. Definisi Wayang Kulit

Wayang kulit adalah seni tradisional Indonesia yang terutama berkembang di Jawa. Wayang berasal dari kata *'Ma Hyang*' yang artinya menuju kepada roh spiritual, dewa, atau Tuhan Yang Maha Esa. Ada juga yang mengartikan wayang adalah istilah bahasa Jawa yang bermakna 'bayangan', hal ini disebabkan karena penonton juga bisa menonton wayang dari belakang kelir atau hanya bayangannya saja. Wayang kulit dimainkan oleh seorang dalang yang juga menjadi narator dialog tokoh-tokoh wayang, dengan diiringi oleh musik gamelan yang dimainkan sekelompok nayaga dan tembang yang dinyanyikan oleh para pesinden. Dalang memainkan wayang kulit di balik kelir, yaitu layar yang terbuat dari kain putih, sementara di belakangnya disorotkan lampu listrik atau lampu minyak (blencong), sehingga para penonton yang berada di sisi lain dari layar dapat melihat bayangan wayang yang jatuh ke kelir. Untuk dapat memahami cerita wayang (lakon), penonton harus memiliki pengetahuan akan tokoh-tokoh wayang yang bayangannya tampil di layar[10].

2.1.1. Pandawa Lima

Pandawa adalah sebuah kata dari bahasa Sanskerta, yang secara harfiah berarti anak Pandu, yaitu salah satu Raja Hastinapura dalam wiracarita Mahabharata. Dengan demikian, maka Pandawa merupakan putra mahkota kerajaan tersebut. Dalam wiracarita Mahabharata, para Pandawa adalah protagonis sedangkan antagonis adalah para Korawa, yaitu putera Dretarastra, saudara ayah mereka (Pandu). Menurut susastra Hindu (Mahabharata), setiap anggota Pandawa merupakan penjelmaan (penitisan) dari Dewa tertentu, dan setiap anggota Pandawa memiliki nama lain tertentu. Misalkan nama "Werkodara" arti harfiahnya adalah "perut serigala". Kelima Pandawa menikah dengan Dropadi yang diperebutkan

dalam sebuah sayembara di Kerajaan Panchala, dan memiliki (masing-masing) seorang putera darinya[10]. Berikut anggota dari Pandawa Lima :

a. Yudistira

   Yudistira merupakan saudara para Pandawa yang paling tua. Ia merupakan penjelmaan dari Dewa Yama dan lahir dari Kunti. Sifatnya sangat bijaksana, tidak memiliki musuh, dan hampir tak pernah berdusta seumur hidupnya. Memiliki moral yang sangat tinggi dan suka memaֹafkan serta suka mengampuni musuh yang sudah menyerah. Memiliki julukan Dhramasuta (putera Dharma), Ajathasatru (yang tidak memiliki musuh), dan Bhārata (keturunan Maharaja Bharata). Ia menjadi seorang Maharaja dunia setelah perang akbar di Kurukshetra berakhir dan mengadakan upacara Aswamedha demi menyatukan kerajaan-kerajaan India Kuno agar berada di bawah pengaruhnya. Setelah pensiun, ia melakukan perjalanan suci ke gunung Himalaya bersama dengan saudara-saudaranya yang lain sebagai tujuan akhir kehidupan mereka. Setelah menempuh perjalanan panjang, ia mendapatkan surge[10].

b. Bima

   Bima merupakan putra kedua Kunti dengan Pandu. Nama bhimā dalam bahasa Sanskerta memiliki arti "mengerikan". Ia merupakan penjelmaan dari Dewa Bayu sehingga memiliki nama julukan Bayusutha. Bima sangat kuat, lengannya panjang, tubuhnya tinggi, dan berwajah paling sangar di antara saudara-saudaranya. Meskipun demikian, ia memiliki hati yang baik. Pandai memainkan senjata gada. Senjata gadanya bernama Rujakpala dan pandai memasak. Bima juga gemar makan sehingga dijuluki Werkodara. Kemahirannya dalam berperang sangat dibutuhkan oleh para Pandawa agar mereka mampu memperoleh kemenangan dalam pertempuran akbar di Kurukshetra. Ia memiliki seorang putera dari ras rakshasa bernama Gatotkaca, turut serta membantu ayahnya berperang, namun gugur. Akhirnya Bima

memenangkan peperangan dan menyerahkan tahta kepada kakaknya, Yudistira. Menjelang akhir hidupnya, ia melakukan perjalanan suci bersama para Pandawa ke gunung Himalaya. Di sana ia meninggal dan mendapatkan surga. Dalam pewayangan Jawa, dua putranya yang lain selain Gatotkaca ialah Antareja dan Antasena[10].

c. Arjuna

Arjuna merupakan putra bungsu Kunti dengan Pandu. Namanya (dalam bahasa Sanskerta) memiliki arti "yang bersinar", "yang bercahaya". Ia merupakan penjelmaan dari Dewa Indra, Sang Dewa perang. Arjuna memiliki kemahiran dalam ilmu memanah dan dianggap sebagai ksatria terbaik oleh Drona. Kemahirannnya dalam ilmu peperangan menjadikannya sebagai tumpuan para Pandawa agar mampu memperoleh kemenangan saat pertempuran akbar di Kurukshetra. Arjuna memiliki banyak nama panggilan, seperti misalnya Dhananjaya (perebut kekayaan – karena ia berhasil mengumpulkan upeti saat upacara Rajasuya yang diselenggarakan Yudistira); Kirti (yang bermahkota indah – karena ia diberi mahkota indah oleh Dewa Indra saat berada di surga); Partha (putera Kunti – karena ia merupakan putra Perta alias Kunti). Dalam pertempuran di Kurukshetra, ia berhasil memperoleh kemenangan dan Yudistira diangkat menjadi raja. Setelah Yudistira mangkat, ia melakukan perjalanan suci ke gunung Himalaya bersama para Pandawa dan melepaskan segala kehidupan duniawai. Di sana ia meninggal dalam perjalanan dan mencapai surge[10].

d. Nakula

Nakula merupakan salah satu putera kembar pasangan Madri dan Pandu. Ia merupakan penjelmaan Dewa kembar bernama Aswin, Sang Dewa pengobatan. Saudara kembarnya bernama Sadewa, yang lebih kecil darinya, dan merupakan penjelmaan Dewa Aswin juga. Setelah kedua orangtuanya meninggal, ia bersama adiknya diasuh oleh Kunti, istri

Pandu yang lain. Nakula pandai memainkan senjata pedang. Dropadi berkata bahwa Nakula merupakan pria yang paling tampan di dunia dan merupakan seorang ksatria berpedang yang tangguh. Ia giat bekerja dan senang melayani kakak-kakaknya. Dalam masa pengasingan di hutan, Nakula dan tiga Pandawa yang lainnya sempat meninggal karena minum racun, namun ia hidup kembali atas permohonan Yudistira. Dalam penyamaran di Kerajaan Matsya yang dipimpin oleh Raja Wirata, ia berperan sebagai pengasuh kuda. Menjelang akhir hidupnya, ia mengikuti pejalanan suci ke gunung Himalaya bersama kakak-kakaknya. Di sana ia meninggal dalam perjalanan dan arwahnya mencapai surge[10].

e. Sadewa

Sadewa merupakan salah satu putera kembar pasangan Madri dan Pandu. Ia merupakan penjelmaan Dewa kembar bernama Aswin, Sang Dewa pengobatan. Saudara kembarnya bernama Nakula merupakan penjelmaan Dewa Aswin juga. Setelah kedua orangtuanya meninggal, ia bersama kakaknya diasuh oleh Kunti, istri Pandu yang lain. Sadewa adalah orang yang sangat rajin dan bijaksana. Sadewa juga merupakan seseorang yang ahli dalam ilmu astronomi. Yudistira pernah berkata bahwa Sadewa merupakan pria yang bijaksana, setara dengan Brihaspati, guru para Dewa. Ia giat bekerja dan senang melayani kakak-kakaknya. Dalam penyamaran di Kerajaan Matsya yang dipimpin oleh Raja Wirata, ia berperan sebagai pengembala sapi. Menjelang akhir hidupnya, ia mengikuti pejalanan suci ke gunung Himalaya bersama kakak-kakaknya. Di sana ia meninggal dalam perjalanan dan arwahnya mencapai surge[10].

2.2. Citra Digital

Citra atau gambar yang didefinisikan sebagai sebuah fungsi dua dimensi, f(x,y), dimana x dan y adalah koordinat bidang datar, dan harga fungsi f disetiap pasangan koordinat (x,y) disebut intensitas atau level keabuan (*grey level*) dari gambar dititik itu. Jika x,y dan f semuanya berhingga (*finite*) dan nilainya diskrit maka gambarnya disebut citra digital[6].

2.2.1. Citra

Secara harfiah, citra (*image*) adalah gambar pada bidang dwimatra (dua dimensi) Gambar 2.1 adalah citra gadis model yang bernama lena. Ditinjau dari sudut pandang matematis, citra merupakan fungsi terus menerus (*continue*) dari intensitas bidang dwimatra. Sumber cahaya menerangi objek, objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pantulan cahaya ini ditangkap oleh alat – alat optik, misalnya mata pada manusia, kamera, pemindai (*scanner*), dan sebagainya, sehingga bayangan objek yang disebut citra tersebut terekam.

Citra sebagai keluaran dari suatu sistem perekaman data dapat bersifat : optik berupa foto, analog berupa sinyal video seperti gambar pada monitor televisi, digital yang dapat langsung disimpan pada suatu pita magnetik. Citra mempunyai dua jenis kategori yaitu : citra diam (*still image*) adalah citra tunggal yang tidak bergerak. Gambar 2.1 adalah contoh dari citra diam. Citra bergerak (*moving images*) adalah rangkaian citra diam yang ditampilkan secara beruntun (sekuensial) sehingga memberi kesan pada mata kita sebagai gambar yang bergerak. Setiap citra di dalam rangkaian itu disebut *frame*. Gambar-gambar yang tampak pada film layar lebar atau televisi pada hakikatnya terdiri atas ratusan sampai ribuan *frame* [6].



Gambar 2.1 Lena [6]

2.2.2. Definisi Pengelolahan Citra

Meskipun sebuah citra kaya informasi, namun seringkali citra yang dimiliki mengalami penurunan mutu (degradasi) misalnya mengandung cacat atau derau (*noise*), warnanya terlalu kontras, kurang tajam, kabur (*blurring*), dan sebagainya. Tentu saja citra semacam ini menjadi lebih sulit diinterpretasi karena informasi yang disampaikan oleh citra tersebut menjadi berkurang. Agar citra yang mengalami gangguan mudah diinterpretasi (baik oleh manusia maupun mesin), maka citra tersebut perlu dimanipulasi menjadi citra lain yang kualitasnya lebih baik. Umumnya, operasi – operasi pada pengolahan citra diterapkan pada citra bila[6] :

1. Perbaikan atau memodifikasi citra perlu dilakukan untuk meningkatkan kualitas penampakan atau untuk menonjolkan beberapa aspek informasi yang terkandung di dalam citra.
2. Elemen di dalam citra perlu dikelompokan, dicocokan atau diukur.
3. Sebagian citra perlu digabung dengan bagian citra yang lain.

2.2.3. Model Citra

Citra ada dua macam yaitu citra kontinu dan citra diskrit. Citra kontinu dihasilkan dari sistem optik yang menerima sinyal analog, misalnya mata manusia dan kamera analog. Citra diskrit dihasilkan melalui proses digitalisasi terhadap citra kontinu. Beberapa sistem optik dilengkapi dengan fungsi digitalisasi sehingga mampu menghasilkan citra diskrit, misalnya kamera digital dan *scanner*. Citra diskrit disebut juga citra digital. Komputer digital saat ini hanya dapat mengelola citra digital. Citra merupakan fungsi malar (kontinu) dari intensitas cahaya pada bidang dwimatra. Secara matematis fungsi intensitas cahaya pada bidang dwimatra disimbolkan dengan f (x,y) yang dalam hal ini :

(x,y) : koordinat pada bidang dwimatra.

f (x,y) : intensitas cahaya (*brightness*) pada bidang dwimatra.

Pada Gambar 2.2 memperlihatkan posisi koordinat pada bidang citra. Sistem koordinat yang diacu adalah sistem koordinat kartesian, yang dalam hal ini sumbu mendatar menyatakan sumbu x, dan sumbu tegak menyatakan sumbu y.



Gambar 2.2 Cara Menentukan Koordinat Titik di Dalam Citra[6]

Karena cahaya merupakan bentuk energi, maka intensitas cahaya bernilai antara 0 sampai tidak berhingga, sehingga di dapatkan persamaan seperti pada persamaan (2.1).

$$0 \leq f\ (x,y) < \infty \quad \text{..... (2.1)}$$

Nilai f (x,y) sebenarnya adalah hasil kali dari :

1. i (x,y) = jumlah cahaya yang berasal dari sumbernya (*illumination*), nilainya antara 0 sampai tidak berhingga.
2. r (x,y) = derajat kemampuan objek memantulkan cahaya (*reflection*), nilainya antara 0 dan 1.

Gambar 2.3 memperlihatkan proses pembentukan intensitas cahaya. Sumber cahaya menyinari permukaan objek. Jumlah pancaran (iluminasi) cahaya yang diterima objek pada koordinat (x,y) adalah i (x,y). Objek memantulkan cahaya yang diterimanya dengan derajat pantulan r (x,y). Hasil kali antara i (x,y) dan r (x,y) menyatakan intensitas cahaya pada koordinat (x,y) yang ditangkap oleh sensor visual pada sistem optik seperti pada persamaan (2.2) kemudian penjelasannya penguraian rumus berasal dari persamaan (2.3) dan (2.4) sehingga penggabungan keduanya akan menjadi persamaan (2.5).

$$f(x,y) = i(x,y).r(x,y) \quad \text{.....(2.2)}$$

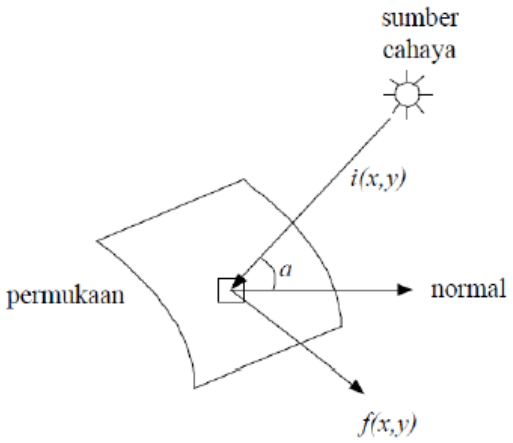
yang dalam hal ini

$$0 \leq i(x,y) < \infty \quad \text{..... (2.3)}$$

$$0 \leq r(x,y) \leq 1 \quad \text{..... (2.4)}$$

Sehingga

$$0 \leq f\ (x,y) < \infty \quad \text{.....(2.5)}$$



Gambar 2.3 Pembentukan Citra[6]

Nilai $i(x, y)$ ditentukan oleh sumber cahaya, sedangkan $r(x, y)$ ditentukan oleh karakteristik objek di dalam gambar. Nilai $r(x,y) = 0$ mengindikasikan penerapan total, sedangkan $r(x,y) = 1$ menyatakan pemantulan total. Jika permukaan mempunyai derajat pemantulan nol, maka fungsi intensitas cahaya, $f(x, y)$, juga nol. Sebaliknya, jika permukaan mempunyai derajat pemantulan 1, maka fungsi intensitas cahaya sama dengan iluminasi yang diterima oleh permukaan tersebut. Intensitas f dari gambar hitam putih pada titik (x, y) disebut *grey level* (derajat keabuan), yang dalam hal ini derajat keabuannya bergerak dari hitam ke putih, sedangkan citranya disebut *greyscale image* (citra hitam putih) atau *monochrome image* (citra *monochrome*). Derajat keabuan memiliki rentang nilai dari *lmin* sampai *l*max atau *lmin* < *f* < *l*max. Selang (*lmin, l*max) disebut skala keabuan.

Biasanya selang (l_{min}, l_{max}) sering digeser untuk alasan-alasan praktis menjadi selang $[0, L]$, yang dalam hal ini nilai intensitas 0 menyatakan hitam, nilai intensitas L menyatakan putih, sedangkan nilai intensitas antara 0 sampai L bergeser dari hitam ke putih. Sebagai contoh, citra hitam-putih dengan 256 *level* artinya mempunyai skala abu dari 0 sampai 255 atau [0, 255], yang dalam hal ini nilai intensitas 0 menyatakan hitam, nilai intensitas 255 menyatakan putih, dan nilai antara 0 sampai 255 menyatakan warna keabuan yang terletak antara hitam dan

putih. Citra hitam-putih disebut juga citra satu kanal, karena warnanya hanya ditentukan oleh satu fungsi intensitas saja. Citra berwarna (*color images*) dikenal dengan nama citra spektral, karena warna pada citra disusun oleh tiga komponen warna yang disebut komponen *RGB*, yaitu merah (*red*), hijau (*green*), dan biru (*blue*). Intensitas suatu titik pada citra berwarna merupakan kombinasi dari tiga intensitas: derajat keabuan merah ($f_{merah}(x,y)$), hijau ($f_{hijau}(x,y)$), dan biru ($f_{biru}(x,y)$) [6].

2.2.4. Digitalisasi Citra

Agar dapat diolah dengan komputer digital, maka suatu citra harus direpresentasikan secara numerik dengan nilai-nilai *diskrit*. Representasi citra dari fungsi malar (kontinu) menjadi nilai-nilai diskrit disebut *digitalisasi* . Citra yang dihasilkan inilah yang disebut citra digital (*digital image*). Pada umumnya citra digital berbentuk empat persegi panjang, dan dimensi ukurannya dinyatakan sebagai tinggi x lebar (atau lebar x panjang).

Citra digital yang tingginya *N*, lebarnya *M*, dan memiliki *L* derajat keabuan dapat dianggap sebagai fungsi seperti pada persamaan (2.6).

$$f(x,y)\begin{cases}0 \le x \le M\\ 0 \le y \le N\\ 0 \le f \le L\end{cases} \quad \text{.....(2.6)}$$

Citra digital yang berukuran N x M lazim dinyatakan dengan matriks yang berukuran N baris dan M kolom seperti pada persamaan (2.7).

$$f(x,y) \approx \begin{bmatrix} f(0,0) & \cdots & f(0,M) \\ \vdots & \ddots & \vdots \\ f(N-1,0) & \cdots & f(N-1,M-1) \end{bmatrix} \quad \text{.....(2.7)}$$

Indeks baris (*i*) dan indeks kolom (*j*) menyatakan suatu koordinat titik pada citra, sedangkan *f* (*i*, *j*) merupakan intensitas (derajat keabuan) pada titik (*i*, *j*). Masing-masing elemen pada citra digital (berarti elemen matriks) disebut *image element*, *picture element* atau *Pixel* atau *pel*. Jadi, citra yang berukuran *N* x *M*

mempunyai *NM* buah *Pixel*. Sebagai contoh, misalkan sebuah berukuran 256 x 256 *Pixel* dan direpresentasikan secara numerik dengan matriks yang terdiri dari 256 buah baris (di-indeks dari 0 sampai 255) dan 256 buah kolom (di-indeks dari 0 sampai 255) seperti contoh berikut pada persamaan (2.8).

$$\begin{bmatrix} 0 & 134 & 231 \\ \vdots & \ddots & \vdots \\ 221 & \cdots & 156 \end{bmatrix} \quad \text{.....(2.8)}$$

Pada matriks persamaan 2.7 *Pixel* pertama pada koordinat (0, 0) mempunyai nilai intensitas 0 yang berarti warna *Pixel* tersebut hitam, *Pixel* kedua pada koordinat (0, 1) mempunyai intensitas 134 yang berarti warnanya antara hitam dan putih, dan seterusnya. Proses digitalisasi citra ada dua macam:

1. Digitalisasi spasial (*x*, *y*), sering disebut sebagai penerokan (*sampling*). *Sampling* menyatakan besarnya kotak-kotak yang disusun dalam baris dan kolom. Dengan kata lain sampling pada citra menyatakan besar kecilnya ukuran piksel pada citra. Untuk memudahkan implementasi, jumlah sampling biasanya diasumsikan perpangkatan dari dua seperti pada persamaan (2.9).

   $$N = 2^n \text{..... (2.9)}$$

   Dimana :

   N = jumlah sampling pada suatu baris atau kolom

   n = bilangan bulat positif

   Pembagian gambar menjadi ukuran tertentu menentukan resolusi spasial yang diperoleh. Semakin tinggi resolusinya, yang berarti semakin kecil ukuran piksel (atau semakin banyak jumlah pikselnya), semakin halus gambar yang diperoleh karena informasi yang hilang akibat pengelompokkan derajat keabuan pada pen-*sampling*-an semakin kecil.

2. Digitalisasi intensitas *f* (*x*, *y*), sering disebut sebagai *kuantisasi*. Setelah proses *sampling* pada citra maka proses selanjutnya adalah kuantisasi. Kuantisasi menyatakan besarnya nilai tingkat kecerahan yang dinyatakan dalam nilai tingkat keabuan (*Grayscale*) sesuai dengan jumlah *bit biner* yang digunakan,

dengan kata lain kuantisasi pada citra menyatakan jumlah warna yang ada pada citra. Proses kuantisasi membagi skala keabuan (0, L) menjadi G buah level yang dinyatakan dengan suatu harga bilangan bulat (integer), biasanya G diambil perpangkatan dari 2, seperti pada persamaan (2.10). Pada **tabel 2.1** ditunjukkan hasil kuantisasi citra dengan skala keabuan yang berbeda-beda.

$$G = 2^m \quad \text{.....(2.10)}$$

Dimana,

G = derajat keabuan

m = bilangan bulat positif

Tabel 2.1 Tabel Kuantisasi Citra Dengan Skala Keabuan Yang Berbeda [6]

Skal Keabuan	Rentang Nilai Keabuan	*Pixel* Depth
2^1(2 nilai)	0,1	1 bit
2^2(4 nilai)	0 sampai 3	2 bit
2^4(16 nilai)	0 sampai 15	4 bit
2^8(256 nilai)	0 sampai 255	8 bit

Jumlah *bit* yang dibutuhkan untuk mempresentasikan nilai keabuan piksel disebut kedalaman piksel (*Pixel depth*). Citra sering diasosiasikan dengan kedalaman pikselnya. Jadi, citra dengan kedalaman 8 *bit* disebut juga citra 8-*bit* (atau citra 256 warna, $G = 256 = 2^8$). Semakin banyak jumlah derajat keabuan (berarti jumlah bit kuantisasinya makin banyak), semakin bagus gambar yang diperoleh [6].

2.2.5. Kategori Pengolahan Citra

Pengolahan citra dapat dibagi ke dalam tiga kategori yakni, kategori rendah, menengah, dan tinggi.

1. Kategori rendah, melibatkan operasi-operasi sederhana seperti prapengolahan citra untuk mengurangi derau, pengaturan kontras, dan pengaturan ketajaman citra. Pengolahan kategori rendah ini memiliki *Input* dan *Output* berupa citra.
2. Kategori menengah, melibatkan operasi-operasi seperti segmentasi dan klasifikasi citra. Proses pengolahan citra menengah ini melibatkan *Input* berupa citra dan *Output* berupa atribut (fitur) citra yang dipisahkan dari citra *Input*.
3. Kategori tinggi, melibatkan proses pengenalan dan deskripsi citra [6].

2.2.6. Pengolahan Citra Digital

Pengolahan citra adalah kegiatan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia/mesin (komputer). Data masukan adalah citra dan keluarannya juga citra tapi dengan kualitas lebih baik dari pada citra masukan. Proses pengolahan citra yang digunakan pada penelitian ini terdiri dari proses *Scaling, Grayscale, edge detection (Canny).*

2.2.6.1. *Scaling*

Scaling merupakan proses pengubahan ukuran gambar digital, hal ini perlu dilakukan agar semua gambar digital data masukan memiliki ukuran yang sama dengan memperkecil citra[6]. Menghitung skala perubahan panjang dan lebar dengan terlebih dahulu menentukan skala baru dihitung menggunakan persamaan (2.11) dan (2.12).

$$nwidthfactor = \frac{width}{nwidth} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.11)$$

$$nheightfactor = \frac{height}{nheight} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.12)$$

Menentukan koordinat *Pixel* di sekitar matriks yang sedang diulang, x dan y dengan menggunakan persamaan (2.13) – (2.20).

$$fr_x = x * nwidthfactor \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.13)$$

$$fr_y = y * nheightfactor \ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.14)$$

$$C_x = fr_x + 1 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.15)$$

$$C_y = fr_y + 1 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.16)$$

$$f_x = x * nwidthfactor - fr_x \ldots\ldots\ldots\ldots\ldots\ldots (2.17)$$

$$f_y = y * nheightfactor - fr_y \ldots\ldots\ldots\ldots\ldots\ldots (2.18)$$

$$n_x = 1{,}0 - f_x \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.19)$$

$$n_y = 1{,}0 - f_y \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.20)$$

Mengambil nilai *Pixel* pada 4 koordinat menggunakan persamaan (2.21) – (2.24) .

$$C_1 = pixel\ (fr_x, fr_y) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.21)$$

$$C_2 = pixel\ (C_x, fr_y) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.22)$$

$$C_3 = pixel\ (fr_x, C_y) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.23)$$

$$C_4 = pixel\ (C_x, C_y) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.24)$$

Menghitung nilai RGB baru dengan menggunakan nilai RGB dari keempat koordinat yang sudah diambil dengan menggunakan persamaan (2.25) – (2.33).

$$b_1 = (n_x * C_1.R + f_x * C_2.R) \quad (2.25)$$

$$b_2 = (n_x * C_3.R + f_x * C_4.R) \quad (2.26)$$

$$R = (n_x * b_1 + f_y * b_2) \quad (2.27)$$

$$b_1 = (n_x * C_1.G + f_x * C_2.G) \quad (2.28)$$

$$b_2 = (n_x * C_3.G + f_x * C_4.G) \quad (2.29)$$

$$G = (n_x * b_1 + f_y * b_2) \quad (2.30)$$

$$b_1 = (n_x * C_1.B + f_x * C_2.B) \quad (2.31)$$

$$b_2 = (n_x * C_3.B + f_x * C_4.B) \quad (2.32)$$

$$B = (n_x * b_1 + f_y * b_2) \quad (2.33)$$

2.2.6.2. *Grayscale*

Citra *Grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap *Pixel* nya, 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 kebuan disini merupakan warna abu dengan berbagai dari hitam hingga mendekati putih. Citra *Grayscale* memiliki kedalaman warna 8 *bit* (256 kombinasi warna keabuan). Rentang nilai warna – warna pada RGB untuk *Red* (merah) rentangnya (255,0,0), *Green* (hijau) rentangnya (0,255,0), *Blue* (biru) rentangnya (0,0,255), dan Medium *Gray* (abu) rentangnya (127,127,127). Misalkan nilai RGB warna adalah (r, g, b), dimana r, g, dan b adalah bilangan bulat antara 0 dan 255. Rata-rata tertimbang *Grayscale*, x, diberikan oleh persamaan (2.34).

$$x = (R + G + B) / 3 \quad (2.34)$$

Contoh, sebuah bayangan gelap ungu memiliki nilai RGB (100, 0, 150). Rata-rata tertimbang adalah x = (100 + 0 +150)/3 = 83 , Jadi (83,83,83) hasilnya adalah abu – abu.

Tabel 2.2 Tabel Warna RGB [6]

Colorname	HEX	RGB
Black	000000	0,0,0
Blue	0000FF	0,0,255
Green	008000	0,128,0
Red	FF0000	255,0,0

2.2.6.3. *Edge Detection* (*Canny*)

Salah satu algoritma deteksi tepi modern adalah deteksi tepi dengan menggunakan metode *Canny* [16]. Algoritma deteksi tepi *Canny* dikenal sebagai detektor tepi yang optimal. Deteksi tepi *Canny* ditemukan oleh Marr dan Hilderth yang meneliti pemodelan persepsi visual manusia. Ada beberapa kriteria pendeteksian tepi yang optimum yang dapat dipenuhi oleh algoritma *Canny* [16]:

1. Mendeteksi dengan baik (kriteria deteksi)

   Kemampuan untuk meletakkan dan menandai semua tepi yang sesuai dengan pemilihan parameter-parameter konvolusi yang dilakukan. Sekaligus juga memberikan fleksibilitas yang sangat tinggi dalam hal menentukan tingkat deteksi ketebalan tepi suatu yang diinginkan.
2. Melokasikan dengan baik (kriteria lokalisasi)

   Dengan *Canny* dimungkinkan menghasilkan jarak yang minimum antara tepi dideteksikan tepi yang asli.
3. Respon yang jelas (kriteria respon)

   Hanya ada satu respon untuk tiap tepi, sehingga mudah untuk dideteksi dan tidak menimbulkan kerancuan pada pengolahan citra selanjutnya.

Metode *Canny* akan mendeteksi tepi dengan mencari nilai *gradient* maksimal local dari sebuah citra I. *Gradient* tersebut dihitung dengan menggunakan turunan dari *Gaussian filter*. Metode *Canny* mengguanakan dua *thearholds*, yang berguna untuk mendeteksi tepian yang terlihat jelas dan tepian

yang jelas. Metode *Canny* lebih utama dan mendeteksi tepian yang kurang jelas, yang tidak dapat diperoleh dengan mengguanakan metode lain.

Deteksi tepi *Canny* dapat mendeteksi tepian yang sebenarnya dengan tingkat kesalahan minimum. Dengan kata lain, operator *Canny* didesain untuk menghasilkan citra tepian yang optimal. Berikut adalah langkah-langkah dalam melakukan deteksi tepi *Canny* :

1. Menghilangkan derau yang ada pada citra dengan mengimplementasikan tapis *Gaussian*. Proses ini akan menghasilkan citra yang tampak sedikit buram. Hal ini dimaksud untuk mendapatkan tepian citra yang sebenarnya. Bila tidak dilakukan maka garis garis halus juga akan dideteksi sebagai tepian. Berikut ini adalah salah satu contoh tapis *Gaussian* dengan σ =1.4 (2.35).

$$\frac{1}{115}\begin{pmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{pmatrix} \quad \text{..... (2.35)}$$

2. Melakukan deteksi tepi menggunakan *Sobel* dengan melakukan pencarian secara horizontal (Gx) dan secara vertikal (Gy).

Tabel 2.3 Tapis Gx *Sobel* (Horizontal)

-1	0	1
-2	0	2
-1	0	1

Tabel 2.4 Tapis Gy *Sobel* (Vertical)

1	2	1
0	0	0
-1	-2	-1

3. Menentukan arah tepian yang ditemukan dengan menggunakan persamaan sebagai berikut (2.36). Kemudian membuat semua arah menjadi positif menggunakan persamaan (2.37). Lalu mendekatkan

arahnya ke nilai 0, 45, 90 atau 135 derajat. Kemudian menghitung *magnitude* dengan persamaan (2.39)

θ=(Gx,Gy) ..... (2.36)

θ=θ∗180/π..... (2.37)

(a,b)= 360+ (a,b) ..... (2.38)

$\theta(a,b)$= 0 jika ($\theta(a,b) \geq 0$ dan $\theta(a,b) < 22.5$) $atau$ ($\theta(a,b) \geq 157.5$ dan $\theta(a,b) < 202.5$) $atau$ ($\theta(a,b) \geq 337.5$ dan $\theta(a,b) < 360$)

$\theta(a,b)$= 45 jika ($\theta(a,b) \geq 22.5$ dan $\theta(a,b) < 67.5$) $atau$ ($\theta(a,b) \geq 202.5$ dan $\theta(a,b) < 247.5$)

$\theta(a,b)$= 90 jika ($\theta(a,b) \geq 67.5$ dan $\theta(a,b) < 112.5$) $atau$ ($\theta(a,b) \geq 247.5$ dan $\theta(a,b) < 292.5$)

$\theta(a,b)$= 135 jika ($\theta(a,b) \geq 112.5$ dan $\theta(a,b) < 157.5$) $atau$ ($\theta(a,b) \geq 292.5$ dan $\theta(a,b) < 337.5$)

$$m = \sqrt{Gx^2 + Gy^2} \quad \text{..... (2.39)}$$

4. Memperkecil garis tepi yang muncul dengan menerapkan *nonmaximum suppression* sehingga menghasilkan garis tepian yang lebih ramping.

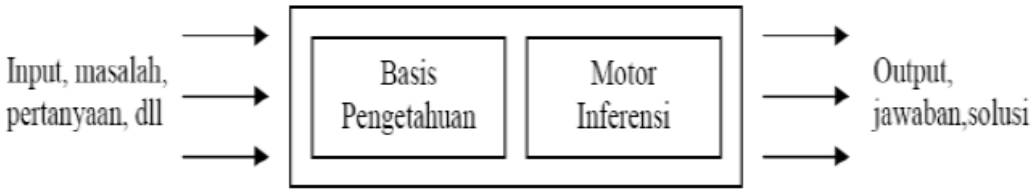
   $Jika$ (a,)=0, $maka\ nilai\ f(a,b)$
   =max($m[a,b]$, $m[a,b+1]$,$m[a,b-1]$)
   $Jika$ (a,)=45, $maka\ nilai\ f(a,b)$
   =max($m[a,b]$,$m[a,b-1]$,$m[a,b+1]$)
   $Jika$ (a,)=90,$maka\ nilai\ f(a,b)$
   =max($m[a,b]$,$m[a+1,b]$,$m[a-1,b]$)
   $Jika$ (a,)=135,$maka\ nilai\ f(a,b)$
   =max($m[a,b]$,$m[a-1,b]$,$m[a+1,b]$)

5. Langkah terakhir adalah menerapkan dua buah *Tresholding* atau sering disebut *Hysteresis Tresholding*. Dengan nilai Tmin 0.075 dan Tmax 0.175

2.3 Kecerdasan Buatan

Kecerdasan buatan *(Artificial Intelligence)* merupakan salah satu ilmu komputer yang membuat agar mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia. Kecerdasan buatan mempunyai dua bagian utama yang sangat dibutuhkan dalam melakukan aplikasi, seperti terlihat pada **gambar 2.4** komponen yang dibutuhkan adalah :

a. Basis Pengetahuan *(Knowledge Base)*, berisi fakta-fakta, teori, pemikiran dan hubungan antara satu dengan lainnya.
b. Motor Inferensi *(Inference Engine),* yaitu kemampuan menarik kesimpulan berdasarkan pengalaman.



Gambar 2.4 Penerapan Kecerdasan Buatan [11]

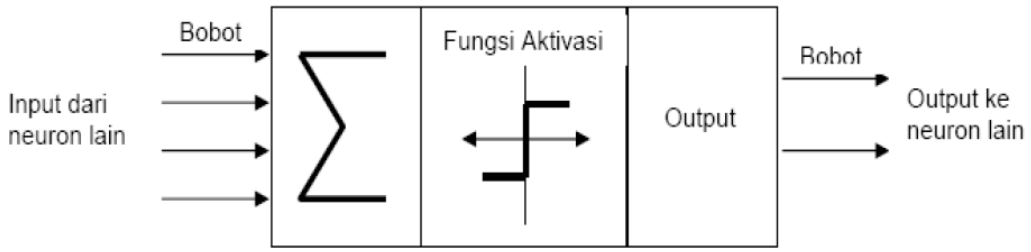
Kecerdasan buatan terdiri dari beberapa kajian diantaranya adalah sistem pakar, jaringan syaraf tiruan, *computer vision*, simulasi *crowd*, logika *fuzzy*, algoritma genetika, dokumen *minning.*

2.4 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan bisa dibayangkan seperti otak buatan di dalam cerita-cerita fiksi ilmiah. Otak buatan ini dapat berpikir seperti manusia, dan juga sepandai manusia dalam menyimpulkan sesuatu dari potongan – potongan informasi yang diterima. Khayalan manusia tersebut mendorong para peneliti untuk mewujudkannya. Komputer diusahakan agar bisa berpikir sama seperti cara berpikir manusia. Caranya adalah dengan melakukan peniruan terhadap aktivitas – aktivitas yang terjadi di dalam sebuah jaringan saraf biologis [11].

Komponen JST terdiri dari neuron – neuron dan neuron – neuron tersebut saling berhubungan. Neuron – neuron tersebut akan mentransformasikan informasi yang diterima melalui sambungan keluarnya menuju neuron – neuron yang lain,

pada JST hubungan neuron - neuron tersebut dikenal dengan nama bobot. Komponen JST seperti terlihat pada **Gambar 2.5** terdiri dari *Input* (informasi) bobot (nilai- nilai tertentu), fungsi aktivasi berfungsi apabila *Input* yang dimasukan sesuai dengan nilai ambang yang ditentukan jika tidak sesuai maka fungsi aktivasi tidak diaktifkan, dan apabila neuron-neuron tersebut diaktifkan maka neuron akan mengirimkan *Output* melalui bobot-bobot.



Gambar 2.5 Komponen Jaringan Syaraf Tiruan [11]

Jaringan Syaraf tiruan mempunyai dua macam proses pembelajaran yaitu pembelajaran terawasi dan pembelajaran tidak terawasi. Pembelajaran terawasi adalah jika *Output* yang diharapkan telah diketahui sebelumnya, sedangkan pembelajaran tidak terawasi adalah proses pembelajaran yang tidak memerlukan target *Output*.

1. Pembelajaran Terawasi (*Supervised Learning*)

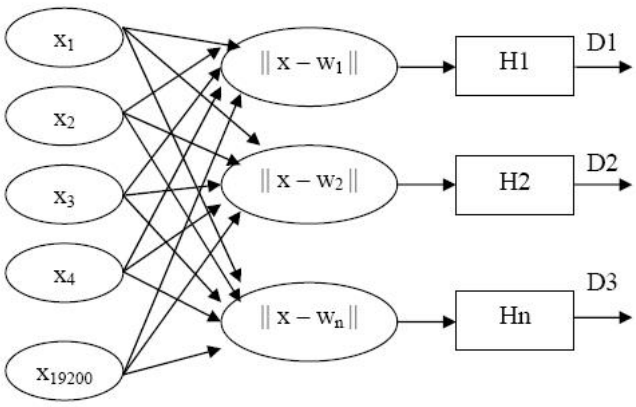
   Metode pembelajaran pada jaringan syaraf disebut terawasi jika *Output* yang diharapkan telah diketahui sebelumnya. Pada proses pembelajaran, satu pola *Input* akan diberikan ke satu neuron pada lapisan *Input*. Pola ini akan dirambatkan di sepanjang jaringan syaraf hingga sampai ke neuron pada lapisan *Output*. Lapisan *Output* ini akan membangkitkan pola *Output* yang nantinya akan dicocokkan dengan pola *Output* targetnya. Apabila terjadi perbedaan antara pola *Output* hasil pembelajaran dengan pola target, maka disini akan muncul *error*. Apabila nilai *error* ini masih cukup besar, mengindikasikan bahwa masih perlu dilakukan lebih banyak pembelajaran lagi. Metode pembelajaran terawasi diantaranya *Hebb Rule, Perceptron, Delta Rule,*

Backpropagation, Learning Vektor Quantization, Heteroassociative Memory.

2. Pembelajaran Tak Terawasi (*Unsupervised Learning*) pada metode pembelajaran yang tak terawasi ini tidak memerlukan target *Output*. Pada metode ini, tidak dapat ditentukan hasil yang seperti apakah yang diharapkan selama proses pembelajaran. Metode pembelajaran tidak terawasi salah satunya adalah jaringan kohonen.

2.5 *Learning Vector Quantization*

*Learning Vector Quantization (*LVQ*)* merupakan salah satu bentuk Algoritma Jaringan Syaraf Tiruan (*Neural Network*) dengan metode pembelajaran pada lapisan kompetitif yang terawasi [12]. Suatu lapisan kompetitif akan secara otomatis belajar untuk mengklasifikasikan vektor-vektor *Input*. Kelas-kelas yang didapatkan sebagai hasil dari lapisan kompetitif ini hanya tergantung pada jarak antara vektor-vektor *Input*. Jika 2 vektor *Input* mendekati sama, maka lapisan kompetitif akan melakukan kedua vektor *Input* tersebut ke dalam kelas yang sama. Untuk melihat arsitektur *learning vector quantization* bisa dilihat pada Gambar 2.6.



Gambar 2.6 Arsitektur LVQ [12]

Keterangan :

a. x1 sampai dengan xm = nilai *Input*
b. || x – w1 || sampai dengan || x – wn || = jarak bobot
c. H1 sampai dengan Hn = lapisan *Output*

d. D1 sampai dengan Dn = nilai *Output*
e. m : jumlah elemen *Input*
f. n = jumlah kelas

Berikut ini algoritma pembelajaran metode *learning vector quantization*:

Langkah 0 : Inisialisasikan vektor referensi dan *learning* rate(α).

Langkah 1 : Selama kondisi berhenti bernilai salah, kerjakan langkah 2 dan 3. (Kondisi berhenti jika perulangan mencapai maksimal perulangan dan perubahan rasio lebih kecil dari minimal *error*)

Langkah 2 : Untuk masing-masing pelatihan vektor masukan , kerjakan:

a. Temukan J sehingga ||*x* – *Wj*|| bernilai minimum.

$$C_j = \sqrt{\sum_{i=1}^{n}(x_i - w_{ij})^2} \quad \text{.....(2.40)}$$

b. Perbaiki Wj dengan :

i. Jika T=Cj maka

Wj(baru) = Wj(lama) + α [x - Wj(lama)] .....(2.41)

ii. Jika T≠Cj maka

Wj(baru) = Wj(lama) – α [x – Wj(lama)].....(2.42)

Langkah 3 : Kurangi *learning* rate. $\alpha = \alpha * pengurang\ rasio$

Langkah 4: Tes kondisi berhenti.

Keterangan rumus :

T = Target
J = Jumlah selisih data dan bobot
C = Kelas selisih bobot terkecil
W = Bobot
A = Pembelajaran
x = Data

2.6 Pemrograman Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara matematis. Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana perangkat lunak di bangunmelalui pendekatan objek secara sistematis [13]. Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut :

1. Meningkatkan produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih bisa dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).
2. Kecepatan pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan dalam pengkodean.
3. Kemudahan pemeliharaan

   Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.
4. Adanya konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analis, perancangan maupun pengkodean.
5. Meningkatkan kualitas perangkat lunak

   Karena pendekatan pembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Berikut ini adalah beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek :

a. *Class*

   Class adalah kumpulan objek – objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama mungkin lahir atau diciptakan dengan kelas tersebut. Sebuah kelas mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (*relationship*) dan arti.

b. Objek

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, suatu organisasi, tempat, kejadian, struktur, status atau hal–hal lain yang bersifat abstrak.

c. Metode

   Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi.

d. Atribut

   Atribut adalah sebuah kelas variabel global yang dimiliki sebuah kelas, atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek.

e. Abstraksi

   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

f. Enkapsulasi

   Pembungkusan atribut data dan layanan yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

g. Pewarisan

   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.

h. Antarmuka

Antarmuka sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi.

i. *Reusability*

Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

j. Generalisasi dan Spesialisasi

Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus. Misalnya kelas yang lebih umum (generalisasi) adalah kendaraan darat dan kelas khususnya (spesialisasi) adalah mobil, motor, dan kereta.

k. Komunikasi Antar Objek

Komunikasi antar – objek dilakukan lewat pesan yang dikirim dari satu objek ke objek lainnya.

l. *Polimorfisme*

Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

m. *Package*

Package adalah sebuah komentar atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam *package* yang berbeda.

2.7 UML (*Unified Modelling Language*)

UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram teks-teks pendukung. Berbagai diagram pada UML antara lain [13] :

a. *Class Diagram*

   Diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

b. *Object Diagram*

   Diagram object menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem.

c. *Component Diagram*

   Diagram komponen dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem.

d. *Package Diagram*

   Menyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram UML. Hampir semua diagram dalam uml dapat dikelompokkan menggunakan *package* diagram.

e. *Deployment Diagram*

   Diagram ini menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi.

f. *Use Case Diagram*

   Merupakan pemodelan kelakuan sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat.

g. *Activity Diagram*

   Diagram aktivitas menggambarkan *workflow* atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak.

h. *Sequence Diagram*

   Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

2.8 IDE yang Digunakan

IDE atau singkatan dari (*Integrated Development Environment* adalah program komputer yang memiliki beberapa fasilitas yang diperlukan dalam pembangunan perangkat lunak. Tujuan dari IDE adalah untuk menyediakan semua utilitas yang diperlukan dalam membangun perangkat lunak[14].

Sebuah IDE, atau secara bebas dapat diterjemahkan sebagai Lingkungan Pengembangan Terpadu, setidaknya memiliki fasilitas:

a. Editor, yaitu fasilitas untuk menuliskan kode sumber dari perangkat lunak.
b. *Compiler*, yaitu fasilitas untuk mengecek sintaks dari kode sumber kemudian mengubah dalam bentuk binari yang sesuai dengan bahasa mesin.
c. *Linker*, yaitu fasilitas untuk menyatukan data binari yang beberapa kode sumber yang dihasilkan compiler sehingga data-data binari tersebut menjadi satu kesatuan dan menjadi suatu program komputer yang siap dieksekusi.
d. *Debuger*, yaitu fasilitas untuk mengetes jalannya program, untuk mencari bug/kesalahan yang terdapat dalam program.

Sampai tahap tertentu IDE modern dapat membantu memberikan saran yang mempercepat penulisan. Pada saat penulisan kode, IDE juga dapat menunjukan bagian-bagian yang jelas mengandung kesalahan atau keraguan.

2.8.1 Matlab (*Matrix Labolatory*)

Matlab singkatan dari (*Matrix Labolatory*) adalah suatu program untuk analisis dan komputasi numerik, merupakan suatu bahasa pemrograman matematika lanjutan yang dibentuk dengan dasar pemikiran menggunakan sifat dan bentuk matriks. Pada awalnya, program ini merupakan interface untuk koleksi rutin-rutin numerik proyek *linpack* dan *eispack*, dikembangkan dengan menggunakan bahasa fortran. Namun sekarang, program ini merupakan produk komersial dari perusahaan Mathworks, Inc yang dalam perkembangan selanjutnya dikembangkan menggunakan bahasa C++ dan *assembler* (terutama untuk fungsi-fungsi dasar matlab)[15].

Matlab telah dikembangkan menjadi sebuah *environment* pemograman yang canggih dan berisi fungsi-fungsi *built-in* unuk melakukan tugas pengolahan sinyal, aljabar linier, dan kalkulasi matematis lainnya. *Matlab* juga berisi *toolbox* yang berisi fungsi-fungsi tambahan untuk aplikasi khusus. *Matlab* bersifat *extensible*, dalam arti bahwa seorang pengguna dapat menulis fungsi baru untuk ditambahkan di *library* jika fungsi *built-in* yang tersedia tidak dapat melakukan tugas tertentu[15].