BAB II
LANDASAN TEORI

2.1 Citra Digital

Citra adalah gambar pada bidang dua dimensi. Ditinjau dari sudut pandang matematis, citra merupakan fungsi yang bersifat terus-menerus (*continue*) dari intensitas cahaya pada bidang dua dimensi. Citra digital merupakan susunan *pixel* yang membentuk sebuah gambar, dimana tiap *pixel* merepresentasikan warna (gambar warna/RGB) dan tingkat keabuan (gambar hitam putih/*greyscale*). Citra digital merupakan sebuah larik (*array*) yang berisi nilai-nilai *real* maupun *complex* yang direpresentasikan dengan deretan bit tertentu [Rinaldi Munir, Pengolahan Citra Digital, Bandung, Informatika, 2004].

2.1.1 Tipe Citra Digital

Pada dasarnya pengolahan citra digital merupakan citra dengan fungsi *continue* yang diubah ke dalam bentuk diskrit untuk menyatakan intensitas cahaya dan juga informasi dari citra warna dalam bidang dua dimensi (x dan y). Bagian yang terkecil pada citra disebut piksel (*pixel/px*) yang dapat digunakan pada pengolahan citra digital untuk proses lebih lanjut. Adapun citra digital dibagi dalam beberapa tipe diantaranya:

1. Citra RGB

   Setiap piksel akan terdiri dari tiga warna dasar, yaitu *Red* (merah), *Green* (hijau) dan *Blue* (biru). RGB (*red, green, blue*) merupakan citra 24 bit dengan komponen dari merah, hijau dan biru yang dimana biasanya setiap warna akan memiliki nilai antara 0-255 (8 bit). Kombinasi pada setiap nilainya nanti akan menghasilkan warna yang berbeda-beda.

2. Citra Greyscale

   Citra *greyscale* adalah citra yang hanya memiliki satu nilai kanal pada setiap pikselnya (*red=green=blue*). Tersimpan dalam format 8 bit untuk setiap piksel

dan memungkinkan sebanyak 256 nilai intesitasnya. Tingkat kecerahan yang paling rendah yaitu 0 untuk warna hitam dan putih bernilai 255.

2.1.2 Format Citra Digital

Citra Digital memiliki beberapa format umum diantaranya Bitmap (BMP), Joint Photographic Group Exepert (JPEG/JPG), Graphics Interchange Format (GIF), dan Purchase Ordentable Network Graphics (PNG).

1. Bitmap (BMP)

   Format baku citra pada sistem operasi *windows* dan IBM OS/2 adalah bitmap. Citra berformat Bitmap (BMP) merupakan citra yang tidak terkompresi, sehingga pada umumnya citra berformat BMP memiliki ukuran relative lebih besar dibandingkan dengan format citra yang lainnya. Intensiras pada pixel dari citra berformat BMP dipetakan ke sejumlah bit tertentu. Panjang setiap *pixel* pada bitmap adalah 4 bit, 8 bit, hingga 24 bit yang merepresentasikan nilai intensitas *pixel*. Maka dari itu ada sebanyak 28=256 derajat keabuan, mulai dari 0 sampai dengan 255.

2. Joint Photographic Group Expert (JPEG/JPG)

   Joint Photographic Group Expert atau yang biasa kita kenal JPEG/ JPG merupakan format standar kompresi file yang dikembangkan oleh Group Joint Photographic Expert dengan menggunakan kombinasi DCT dengan pengkodean Huffman untuk mengkompresikan citra. Citra dengan format JPEG merupalam citra terkompresi yang bersifat lossy, artinya dimana citra tidak dapat dikembalikan ke bentuk aslinya. Citra ini memiliki ukuran yang relative lebih kecil dibandingkan dengan citra berformat BMP karena telah terkompresi.

3. Graphics Interchange Format (GIF)

   Graphics Interchange Format (GIF) merupalam format citra terkompresi. Ukuran dari citra berformat GIF adalah yang terkecil dari semua format citra digital. Kombinasi warna yang tersedia adalah sebanyak 256 warna. Citra pada jenis ini biasanya banyak digunakan untuk keperluan *website*, untuk membuat sebuah

gambar, *icon*, untuk membuat animasi dan keperluan lainnya yang mengharuskan menggunakan citra dengan ukuran yang sangat kecil.

4. Purchase Ordentable Network Graphics (PNG)

   Purchase Ordentable Network Graphics atau lebih dikenal dengan PNG merupakan salah satu format penyimpanan citra yang menggunakan metode kompresi yang tidak menghilangkan bagian dari citra tersebut (*lossless compression*). Biasanya citra berformat PNG merupakan salah satu format yang baik untuk digunakan pengolahan citra karena format ini tidak menghilangkan bagian dari citra yang sedang diolah.

2.2 Pengolahan Citra

Pengolahan citra atau yang sering disebut dengan *image processing* merupakan suatu sistem dimana proses dilakukan dengan cara masukan (*input*) berupa citra (*image*) dan hasilnya (*output*) juga berupa citra. Pengolahan citra digital dimulai sekitar awal tahun 1960-an, pada saat itu diluncurkan komputer yang mampu melakukan pengolahan citra digital. Pada tahun 1964 terjadi proses pengolahan citra berupa perbaikan kualitas citra bulan dari distorsi di laboratorim Jet Propulsion. Citra bulan kemudian dikirim oleh pesawat ulang-alik Ranger 7. Komputer tersebut adalah pemicu cepatnya perkembangan teknologi pengolahan citra komputer. Pada awalnya pengolahan citra ini dilakukan untuk dapat memperbaiki kualitas citra namun dengan berkembangnya dunia komputasi yang diiringi semakin meningkatnya kapasitas dan kecepatan proses computer, serta munculnya ilmu-ilmu komputasi yang ditandai dengan semakin meningkatnya kapasitas dan kecepatan proses komputer, serta dengan munculnya ilmu-ilmu komputasi yang memungkinkan manusia dapat mengambil informasi dari suatu citra, maka pengolahan citra tida dapat dilepaskan dengan bidang *computer vision* (sistem visual).

Setelah semakin pesatnya perkembangan komputasi yang terjadi, pengolahan citra memiliki dua tujuan utama yaitu memperbaiki kualitas citra yang dimana citra yang dihasilkan dapat menampilkan informasi secara jelas atau dengan kata lain kita

sebagai manusia dapat melihat informasi yang diharapkan dengan menginterpretasikan citra yang ada dan mengekstrasi informasi ciri yang biasanya menonjol pada suatu citra, hasilnya merupakan informasi citra dimana manusia mendapatkan informasi ciri dari citra secara numerik atau dengan kata lain komputer melakukan interpretasi terhadap informasi yang ada melalui besaran-besaran data yang dapat dibedakan secara jelas.

2.3 Operasi Pengolahan Citra

Dalam pengolahan citra banyak operasi-operasi yang dilakukan, umumnya operasi pengolahan citra dapat diklasifikasikan dalam beberapa jenis diantaranya:

1. Pemugaran Citra (Image Restoration)

   Pemugaran citra atau *image restoration* adalah operasi citra yang bertujuan untuk menghilangkan atau meminimalkan cacat pada citra. Adapun tujuan dari pemugaran citra ini hampir sama dengan operasi perbaikan citra hanya saja perbedaanya pada pemugaran citra adalah penyebab degradasi gambit yang diketahui. Contoh dari operasi pemugaran citra adalah penghilang kesamaran (*debluring*) dan penghilang derau (*noise*).

2. Segmentasi Citra (Image Segmentation)

   Segmentasi Citra atau image segmentation merupakan operasi pengolahan citra yang bertujuan untuk memecah suatu citra kedalam beberapa segmen dengan suatu kriteria tertentu. Biasanya jenis operasi ini berkaitan erat dengan pengolahan pola.

3. Perbaikan Kualitas Citra (*Image Enhancement)*

   Operasi cira pada jenis ini bertujuan untuk memperbaiki kualitas citra dengan cara memanipulasi parameter-parameter citra, ciri khusus yang terdapat didalam citra lebih ditonjolkan. Adapun beberapa contoh operasi citra perbaikan kualitas citra diantaranya adalah perbaikan kontras gelap/terang, perbaikan tepian objek (*edge enhancement)*, Penajaman warna semu (*Sharpening),* dan penapisan derau (*noise filtering)*.

4. Pemampatan Citra (*Image Compression*)

   Pemampatan citra atau *image compression* merupakan jenis operasi yang bertujuan untuk citra yang dipresentasikan dalam bentuk yang lebih kompak sehingga memerlukan memori yang lebih sedikit. Adapun hal penting yang harus diperhatikan dalam pemampatan citra adalah citra yang telah dimampatan harus tetap memiliki kualitas gambar yang baik.

5. Analisis Citra (*Image Analysis)*

   Jenis operasi ini bertujuan untuk menghitung besaran kuantitatif gari sebuah citra untuk menghasilkan deskripsinya. Untuk Analisis citra atau *image analysis* memiliki tekniknya sendiri yaitu dengan mengekstraksi ciri-ciri tertentu yang membantu dalam identifikasi objek. Untuk proses segmentasi seringkali diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya. Beberapa contoh Analisis citra adalah pendeteksian tepi objek (*edge detection*), representasi daerah (*region*), Ekstraksi batas (*boundary*), Rekontruksi citra (*image reconstruction*).

6. Perubahan Model Warna

   Setiap warna memiliki panjang gelombang yang berbeda, karena warna merupakan presepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh suatu objek. Warna merah misalnya memiliki panjang gelombang yang paling tinggi, sedangkan ungu mempunyai panjang gelombang yang rendah. Warna akan diterima oleh mata karena merupakan hasil kombinasi cahaya dengan panjang gelombang yang berbeda. Penelitian memperlihatkan kombinasi warna yang paling lebar adalah merah/*red*(R), hijau/*green*(G), dan biru/*blue*(B).

7. Klasifikasi Citra

   Klasifikasi citra merupakan proses yang berusaha mengelompokkan seluruh pixel pada suatu citra ke dalam sejumlah class (kelas), sedemikian hingga tiap class merepresentasikan suatu entitas dengan properti yang spesifik (Chein-I Chang dan H.Ren, 2000).

8. Rekontruksi Citra (*Image Reconstruction*)

   Rekontruksi citra atau *image reconstruction* merupakan operasi citra yang bertujuan untuk membentuk ulang objek dari beberapa citra hasil proyeksi. Operasi rekontruksi citra banyak digunakan dalam bidang medis. Contohnya adalah rontgen dengan sinar X yang digunakan untuk membentuk ulang gambar organ tubuh.

2.4 Deteksi Tepi

Deteksi tepi atau *Edge detection* pada suatu citra merupakan suatu proses yang menghasilkan tepi-tepi dari objek-objek citra, tujuan dari deteksi tepi ini adalah untuk menandai bagian yang menjadi detail citra dan untuk memperbaiki detail dan citra yang kabur yang terjadi karena *error* atau adanya efek dari proses akusisi citra.

2.4.1 Metode *Canny*

Metode *canny* merupakan salah satu algoritma deteksi tepi modern. Deteksi tepi *canny* awalnya ditemukan oleh Marr dan Hilderth yang meneliti pemodelan persepsi visual manusia. Ada beberapa kriteria pendeteksi tepian paling optimum yang dapat dipenuhi oleh algoritma *canny* diantaranya:

1. Melokalisasi dengan baik (kriteria lokalisasi)

   Metode *canny* akan melokalisasi sebuah citra dengan baik, dengan *canny* dimungkinkan dihasilkan jarak yang minimum antara tepi yang dideteksi dengan tepi yang asli.

2. Mendeteksi dengan baik (kriteria deteksi)

   Metode *canny* dapat mendeteksi citra dengan baik, kemampuan untuk meletakkan dan juga menandai seluruh tepi yang ada sesuai dengan pemilihan seluruh parameter konvolusi yang dilakukan. Sekaligus juga memberikan fleksibilitas yang sangat tinggi pada hal menentukan tingkat deteksi ketebalan tepi sesuai dengan yang diinginkan.

3. Respon yang jelas (kriteria respon)

   Biasanya hanya ada satu respon untuk tiap tepi, hingga sangat mudah dideteksi dan tidak akan menimbulkan kerancuan pada pengolahan citra selanjutnya. Pemilihan parameter deteksi tepi *canny* akan sangat mempengaruhi hasil dari tepian yang dihasilkan. Beberapa parameter tersebut diantaranya adalah nilai *standard* deviasi Gaussian dan nilai ambang.

Metode *canny* akan mendeteksi tepi dengan mencari nilai *gradient* maksimal dari sebuah citra. *Gradient* tersebut dihitung dengan menggunakan turunan *Gaussian Filter*. Metode *canny* secara keseluruhan mendeteksi tepian yang kurang jelas, dan yang tidak dapat dideteksi oleh metode lain. Deteksi *canny* dapat mendeteksi tepian yang sebenarnya dengan tingkat kesalahan minimum.

2.5 Region of Interest

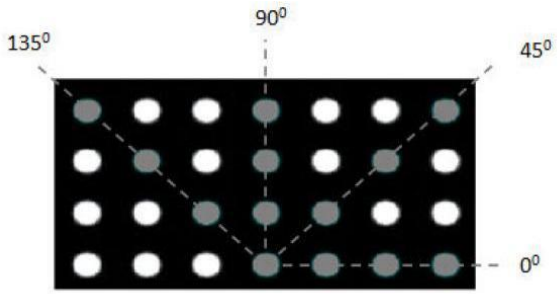
Region of Interest (ROI) adalah suatu bagian dari citra yang dipilih untuk dilakukan proses pengolahan citra. ROI memungkinkan untuk mengakses bagian tertentu dari sebuah citra untuk diolah secara berbeda. Fitur ini menjadi sangat penting apabila terdapat bagian atau area tertentu dari sebuah citra yang dianggap lebih penting dari area lainnya.

2.6 Ekstraksi Ciri GLCM (Grey-Level Co-occurrence Matrix)

GLCM (*Grey-Level Co-occurrence Matrix*) merupakan sebuah matriks yang menyimpan nilai frekuensi kombinasi mengenai perbedaan tingkat kecerahan antara satu piksel dengan piksel yang ada disekitarnya yang terjadi pada sebuah citra. GLCM biasanya akan menyimpan informasi mengenai jejak intensitas keabuan antara dua buah piksel yang dipisahkan dengan jarak *a* dan arah *g*. pada GLCM arah *g* dinyatakan dengan sudut θ (0°, 45°, 90° dan 135°) sedangkan jarak *a* menyatakan sebarapa jauh jarak antara dua buah piksel dengan piksel yang ditinjaunya.

GLCM didefinisikan sebagai tabulasi dari data piksel citra dimana digambarkan seberapa sering kombinasi yang berada pada nilai keabuan yang muncul pada citra. Arah piksel tetangga untuk mewakili jarak dapat dipilih, misalnya 135^{o}, 90^{o}, 45^{o}, 0^{o}

atau, seperti yang diilustrasikan pada Gambar 2.1. Sedangkan Gambar 2.2 menggambarkan bagaimana untuk menghasilkan matriks menggunakan arah 0^o dan dengan jarak 1 piksel.



Gambar 2.1 Arah hitung GLCM

p	0	1	2	3
0	1	1	0	0
1	2	1	0	0
2	0	2	2	0
3	0	0	3	3

Matriks awal

p	0	1	2	3
0	0,0	0,1	0,2	0,3
1	1,0	1,1	1,2	1,3
2	2,0	2,1	2,2	2,3
3	3,0	3,1	3,2	3,3

Framework GLCM

p	0	1	2	3
0	3	0	1	1
1	2	1	0	0
2	1	1	1	0
3	0	0	0	1

Matriks GLCM

Gambar 2.2 Langkah mendapatkan Matriks GLCM pada sudut 0^o

Adapun langkah-langkah metode GLCM adalah sebagai berikut:

1. Quantization

   Quantization atau kuantisasi merupakan konversi nilai *greyscale* (256 nilai keabuan) citra kedalam rentang nilai tertentu. Tujuan dari langkah ini adalah

mengurangi angka perhitungan dan meringankan proses komputasi. Untuk melakukan kuantisasi digunakan persamaan:

Xb = int(X*(2n - 1)/255)

Dimana:

Xb adalah Nilai greyscale baru

X adalah Nilai greyscale lama

2n adalah nilai derajat keabuan baru

Berikut contoh proses kuantisasi pada matriks awal 4x4 diubah dengan nilai keabuan 4 (0-3).

Tabel 2.1 Matriks awal

P	0	1	2	3
0	86	142	139	82
1	138	103	99	138
2	134	102	98	138
3	73	142	137	61

Tabel 2.2 Matriks hasil kuantisasi

P	0	1	2	3
0	1	2	2	1
1	2	1	1	2
2	2	1	1	2
3	1	2	2	2

2. Co-occurrence

Co-occurrence atau kookurensi artinya adalah kejadian bersama atau jumlah kejadian pada satu level nilai intensitas piksel yang bertetangga dengan satu level intensitas piksel lain dalam jarak (*distance*) dan orientasi sudut (θ) tertentu. Tahapan ini membentuk matriks *Co-occurrence* berdasarkan *framework* yang telah ditentukan berdasarkan nilai derajat keabuan dari proses kuantisasi. Pada contoh berikut dibentuk matriks *Co-occurrence* dengan nilai derajat keabuan 4 pada sudut 0^0 dan jarak d=1.

Tabel 2.3 Matriks Framework GLCM

P	0	1	2	3
0	0,0	0,1	0,2	0,3
1	1,0	1,1	1,2	1,3
2	2,0	2,1	2,2	2,3
3	3,0	3,1	3,2	3,3

Tabel 2 4 Matriks hasil kuantisasi

P	0	1	2	3
0	1	2	2	0
1	2	1	1	2
2	2	1	1	2
3	0	2	2	0

Tabel 2.5 Matriks GLCM

P	0	1	2	3
0	0	0	1	0
1	0	2	3	0
2	2	2	2	0
3	0	0	0	0

3. Symetric

Symetric atau matriks simetris disini dapat diartikan sebagai kemunculan posisi piksel yang sama. Misal terdapat piksel (1,3) maka secara orientasi horizontal piksel (1,3) sama dengan piksel (3,1). Oleh karena itu, matriks simetris adalah matriks hasil penjumlahan matriks kookurensi dengan matrik transposenya sendiri. Berikut contoh pembentukan matriks simetris.

Tabel 2.6 Penjumlahan Matriks GLCM dengan matriks Transpose

P	0	1	2	3
0	0	0	1	0
1	0	2	3	0
2	2	2	2	0
3	0	0	0	0

+

P	0	1	2	3
0	0	0	2	0
1	0	2	2	0
2	1	3	2	0
3	0	0	0	0

Tabel 2.7 Matriks Simetris

P	0	1	2	3
0	0	0	3	0
1	0	4	5	0
2	3	5	4	0
3	0	0	0	0

4. Normalization

   Normalization atau normalisasi merupakan langkah untuk menghitung probabilitas matriks. Normalisasi berperan sebagai pembeda antara piksel yang mirip dengan cara membagi jumlah kemunculan piksel tersebut dengan jumlah kemungkinan yang muncul. Maka normalisasi adalah membagi jumlah kemunculan ketetanggaan piksel tertentu dengan jumlah seluruh piksel ketetanggan yang mungkin muncul.

Tabel 2.8 Proses Normalisasi

$$\begin{bmatrix} \frac{0}{24} & \frac{0}{24} & \frac{3}{24} & \frac{0}{24} \\ \frac{0}{24} & \frac{4}{24} & \frac{5}{24} & \frac{0}{24} \\ \frac{3}{24} & \frac{5}{24} & \frac{4}{24} & \frac{0}{24} \\ \frac{0}{24} & \frac{0}{24} & \frac{0}{24} & \frac{0}{24} \end{bmatrix}$$

5. Feature Extraction

   Berdasarkan penelitian yang dilakukan oleh Harlick dkk terdapat 14 jenis ciri tekstural, hanya saja tidak semua ciri diterapkan, karena diantara ciri tekstural tersebut ada beberapa ciri yang bisa menjadi ciri pengacu terhadap objek yang diteliti. Beberapa ciri yang biasa digunakan adalah sebagai berikut:

   a. Momen Angular Kedua (*Angular Second Moment*) / Energi (*Energy*)

      Angular Second Moment atau juga sering disebut Energi menyatakan ukuran konsentrasi pasangan dengan intensitas keabuan tertentu pada matriks, dimana

(i, j) menyatakan nilai pada baris i dan kolom j pada matriks kookurensi. ASM dihitung dengan menggunakan persamaan berikut:

$$\sum_{i,j=0}^{n-1} P(i,j)^2 \qquad (2.1)$$

b. Entropi (*Entropy*)

Entropi dapat menunjukkan ketidakteraturan ukuran bentuk, jika nilai Entropi-nya besar untuk citra dengan transisi derajat keabuan merata dan bernilai kecil jika struktur citra tidak teratur (bervariasi). Entropi dihitung dengan menggunakan persamaan berikut:

$$\sum_{i,j=0}^{n-1} -(P_{\mathrm{i,j}}) * (\log(P_{\mathrm{i,j}})) \qquad (2.2)$$

c. Kontras (*Contrast*)

Kontras menunjukkan ukuran penyebaran (momen inersia) elemen-elemen matriks citra. Jika letaknya jauh dari diagonal utama, nilai kekontrasan besar. Secara visual, nilai kekontrasan adalah ukuran variasi antar derajat keabuan suatu daerah citra. Hasil perhitungan kontras berkaitan dengan jumlah keberagaman intensitas keabuan dalam citra. Kontras dihitung dengan menggunakan persamaan berikut:

$$\sum_{i,j=0}^{n-1} P_{i,j}\,(i-j)^2 \qquad (2.3)$$

d. Disimilaritas (*Dissimilarity*)

Disimilaritas menyatakan ukuran ketidaksamaan linear derajat keabuan citra sehingga dapat memberikan petunjuk adanya struktur linear dalam citra. Disimilaritas dihitung dengan menggunakan persamaan berikut:

$$\sum_{i,j=0}^{n-1} P(i,j).|i-j| \qquad (2.4)$$

e. Momentum Selisih Terbalik (*Inverse Difference Moment/Homogenity*)

Momentum selisih terbalik disebut juga homogenitas yang menunjukkan kehomogenan citra berderajat keabuan sejenis. Citra homogen akan memiliki harga IDM yang besar. IDM dihitung dengan menggunakan persamaan berikut:

$$\sum_{i,j=0}^{n-1} \frac{P_{\mathrm{i,j}}}{1+(i-j)^2} \qquad (2.5)$$

2.7 Klasifikasi

Klasifikasi merupakan suatu proses yang melakukan penilaian terhadap suatu obyek data untuk masuk ke dalam suatu kelas tertentu dari sejumlah kelas yang tersedia. Terdapat 2 proses utama pada klasifikasi, yaitu Pelatihan (*Training*) dan Pengujian (*Testing*).

2.7.1 Support Vector Machine (SVM)

Support Vector Machine (SVM) adalah suatu teknik untuk menemukan *hyperplane* yang bisa memisahkan dua set data dari dua kelas yang berbeda. SVM juga merupakan salah satu algoritma *Machine Learning* yang paling popular untuk klasifikasi dan regresi, SVM memiliki tingkat keberhasilan yang tinggi saat diterapkan di berbagai bidang. Sehingga banyak dari kalangan komunitas *Machine Learning*

berminat untuk mempelajari dan mengembangkan SVM karena kinerjanya sangat baik dalam berbagai masalah pembelajaran.

SVM dikembangkan oleh Boser, Guyon, Vapnik pada tahun 1992. SVM merupakan suatu teknik untuk menemukan hyperplane yang bisa memisahkan dua set data dari dua kelas yang berbeda (Vapnik, 1999). *Hyperplane* adalah garis batas pemisah data antar kelas, sedangkan *margin* adalah jarak antara hyperplane dengan data terdekat pada masing-masing kelas. Adapun data terdekat dengan *hyperplane* pada masing-masing kelas inilah yang disebut *support vector*. Karena konsep awal SVM adalah untuk mengatasi masalah klasifikasi dua kelas, maka ada dua metode diusulkan agar SVM bisa digunakan untuk klasifikasi *multi-class* dengan pendekatan mengombinasikan beberapa *binary classifier* atau menggabungkan semua data yang terdiri dari beberapa kelas ke dalam permasalahan optimasi.

Dalam linear Support Vector Machine pemisah merupakan fungsi linear. Data latih dinyatakan oleh (xi, yi) dan xi = { x1, x2, ... xi} merupakan atribut (fitur) set untuk data latih kelas ke-i. Untuk yi € {-1,1} menyatakan label kelas. Pendefinisian persamaan suatu *hyperplane* pemisah yang dituliskan dengan:

$$w * xi + b = 0 \qquad (2.6)$$

Data xi yang terbagi ke dalam dua kelas, yang termasuk kelas -1 (sampel negatif) didefinisikan sebagai vektor yang memenuhi pertidaksamaan (2.7) berikut.

$$w * xi + b < 0 \text{ untuk } yi = -1 \qquad (2.7)$$

Sedangkan yang termasuk kelas +1 (sampel positif) memenuhi pertidaksamaan (2.8) berikut.

$$w * xi + b > 0 \text{ untuk } yi = +1 \qquad (2.8)$$

Dimana:

xi = data input

yi = label yang diberikan

w = nilai dari bidang normal

b = posisi bidang relatif terhadap pusat koordinat

Parameter *w* dan *b* adalah parameter yang akan dicari nilainya. Bila label data y_i = -1, maka pembatas menjadi persamaan (2.9) berikut:

$$w * xi + b \leq -1 \quad (2.9)$$

Bila label data yi = +1, maka pembatas menjadi persamaan (2.10) berikut:

$$w * xi + b \geq +1 \quad (2.10)$$

Margin terbesar dapat dicari dengan cara memaksimalkan jarak antar bidang pembatas kedua kelas dan titik terdekatnya, yaitu 2/|w|. Hal ini dirumuskan sebagai permasalahan *Quadratic Programming* (QP) *Problem* yaitu mencari titik minimal persamaan (2.11) dengan memperhatikan persamaan (2.12) berikut:

$$min\ \tau(w) = 1/2\ ||w||2 \quad (2.11)$$

$$yi(w * xi + b) - 1 \geq 0,\ (i = 1,..,n) \quad (2.12)$$

Permasalahan ini ini dapat dipecahkan dengan berbagi teknik komputasi. Lebih mudah diselesaikan dengan mengubah persamaan (2.11) ke dalam fungsi Lagrangian pada persamaan (2.13), dan menyederhanakannya menjadi persamaan (2.14) berikut:

$$L(w, b, a) = \frac{1}{2}\ \|w\|^2 - \sum_{i=1}^{n} a_i \left(y_i\big((w^T x_i + b) - 1\big)\right) \quad (2.13)$$

$$L(w, b, a) = \frac{1}{2}\ \|w\|^2 - \sum_{i=1}^{n} a_i\ y_i(w^T x_i + b) + \sum_{i=1}^{n} a_i \quad (2.14)$$

Dimana a_i adalah lagrange multiplier yang bernilai nol atau positif ($ai \geq 0$). Nilai optimal dari persamaan (2.14) dapat dihitung dengan meminimalkan L terhadap *w*, *b* dan *a*. Dapat dilihat pada persamaan (2.15) sampai (2.17) berikut:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} a_i y_i x_i = 0 \quad (2.15)$$

$$\frac{\partial L}{\partial w} = \sum_{i=1}^{n} a_i y_i = 0 \quad (2.16)$$

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} a_i y_i (w^T x_i + b) - \sum_{i=1}^{n} a_i = 0 \qquad (2.17)$$

Maka masalah Lagrange untuk klasifikasi dapat dinyatakan pada persamaan (2.18) berikut:

$$Min\ L(w, b, a) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} a_i y_i (w^T x_i + b) - \sum_{i=1}^{n} a_i \qquad (2.18)$$

Dengan memperhatikan persamaan (2.19) dan (2.20) berikut:

$$w - \sum_{i=1}^{n} a_i y_i x_i = 0 \qquad (2.19)$$

$$\sum_{i=1}^{n} a_i y_i = 0 \qquad (2.20)$$

Model persamaan (2.18) diatas merupakan model *primal Lagrange*. Sedangkan dengan memaksimalkan L terhadap a_i, persamaannya menjadi persamaan (2.21) berikut:

$$Max \sum_{i=1}^{n} a_i - \sum_{i=1, j=1}^{n} a_i a_j y_i y_j^T\ x_i x_j^T \qquad (2.21)$$

Dengan memperhatikan persamaan (2.22) berikut:

$$\sum_{i=1}^{n} a_i y_i = 0, a_i \geq 0\ (i, j = 1, \dots, n) \qquad (2.22)$$

Untuk mencari nilai x_i dan y_i dapat dilakukan ketika sudah didapatkan nilai tiap ciri dari hasil ekstraksi ciri. Hasil dari ekstraksi ciri diubah ke dalam bentuk format data SVM, sedangkan data kelas (positif dan negatif) menjadi label data SVM.

Untuk mendapatkan nilai a_i, langkah pertama adalah mengubah setiap niliai ciri menjadi nilai vektor (*support vector*) = ($\frac{x}{y}$) .Kemudian nilai vektor dari setiap nilai ciri dimasukkan ke persamaan (2.23) kernel *trick phi* berikut.

$$\phi\begin{bmatrix}x\\y\end{bmatrix} = \begin{cases}\sqrt{x_n^2 + y_n^2} > 2\, maka \begin{bmatrix}\sqrt{x_n^2 + y_n^2} - x + |x-y|\\ \sqrt{x_n^2 + y_n^2} - y + |x-y|\end{bmatrix}\\ \sqrt{x_n^2 + y_n^2} \leq 2\, maka \begin{bmatrix}x\\y\end{bmatrix}\end{cases} \quad (2.23)$$

Nilai x didapatkan dari persamaan (2.24) kernel linear untuk x berikut:

$$\sum_{i=1,j=1}^{n} x_i x_j^T \,, (i,j = 1, \dots, n) \quad (2.24)$$

Nilai y didapatkan dari persamaan (2.25) kernel linear untuk y berikut:

$$\sum_{i=1,j=1}^{n} y_i y_j^T \,, (i,j = 1, \dots, n) \quad (2.25)$$

Untuk mendapatkan jarak tegak lurus yang optimal dengan mempertimbangkan vektor positif, maka hasil perhitungan dari substitusi nilai *x* dan nilai *y* ke persamaan (2.22) diberi nilai bias = 1. Kemudian cari parameter a_i, dengan terlebih dahulu mencari nilai fungsi setiap ciri menggunakan persamaan (2.26), lalu mencari nilai a_i pada persamaan linear menggunakan persamaan (2.27) dengan memperhatikan i, j = 1, . . , n berikut:

$$\sum_{i=1,j=1}^{n} a_i S_i^T S_j \quad (2.26)$$

$$\sum_{i=1,j=1}^{n} a_i S_i^T S_j = y_i \quad (2.27)$$

Setelah parameter ai didapatkan, kemudian masukkan ke persamaan (2.28) berikut:

$$\widetilde{W} = \sum_{i=1}^{n} a_i S_i \quad (2.28)$$

Hasil yang didapatkan menggunakan persamaan (2.28), selanjutnya digunakan persamaan (2.29) untuk mendapatkan nilai w dan b:

$$y = wx + b \qquad (2.29)$$

Dengan demikian didapatkanlah nilai w dan b atau *hyperplane* untuk mengklasifikasikan dua kelas.

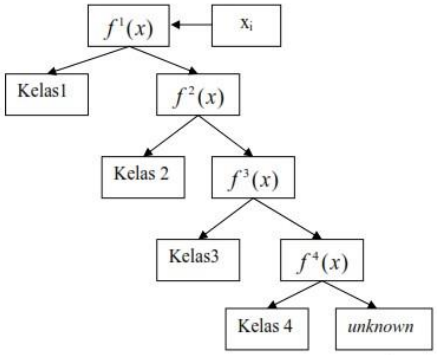
2.7.2 Multiclass SVM dengan Metode ”One-Against-All”

SVM saat pertama kali diperkenalkan oleh Vapnik, hanya dapat mengklasifikasikan data ke dalam dua kelas (klasifikasi biner). Namun, penelitian lebih lanjut untuk mengembangkan SVM sehingga bisa mengklasifikasi data yang memiliki lebih dari dua kelas, terus dilakukan. Ada dua pilihan untuk mengimplementasikan *multi-class* SVM yaitu dengan menggabungkan beberapa SVM biner atau menggabungkan semua data yang terdiri dari beberapa kelas ke dalam sebuah bentuk permasalah optimasi. Namun, pada pendekatan yang kedua permasalahan optimasi yang harus diselesaikan jauh lebih rumit. Berikut ini adalah salah satu metode yang umum digunakan untuk mengimplementasikan *multiclass* SVM yaitu dengan pendekatan *”One-Against-All”.*

Dengan menggunakan metode ini, dibangun k buah model SVM biner (k adalah jumlah kelas). Contohnya, terdapat permasalahan klasifikasi dengan 4 buah kelas. Untuk pelatihan digunakan 4 buah SVM biner seperti pada tabel 2.9 dan penggunannya dalam mengklasifikasi data baru dapat dilihat pada tabel 2.9 [8].

Tabel 2.9 Klasifikasi 4 kelas dengan metode multiclass “One-Against-All”

$y\ i$=1	$y\ i$=-1	Hipotesis
Kelas 1	Bukan Kelas 1	$f^1(x) = (w^1)\ x + b^1$
Kelas 2	Bukan Kelas 2	$f^2(x) = (w^2)\ x + b^2$
Kelas 3	Bukan Kelas 3	$f^3(x) = (w^3)\ x + b^3$
Kelas 4	Bukan Kelas 4	$F^4(x) = (w^4)\ x + b^4$



Gambar 2.3 Diagram keputusan SVM Multiclass "One-Against-All"

2.8 Metode Perancangan Sistem

Metode perancangan yang digunakan adalah metode perancangan terstruktur. Metode ini diperkenalkan pada tahun 1970, yang merupakan hasil turunan dari pemrograman terstruktur. Pemrograman Terstruktur adalah suatu proses untuk mengimplementasikan urutan langkah untuk menyelesaikan suatu masalah dalam bentuk program. Metode pengembangan dengan metode terstruktur ini terus diperbaiki sampai akhirnya dapat digunakan dalam dunia nyata.

Perancangan ini bertujuan untuk membuat model solusi terhadap problem yang sudah dimodelkan secara lengkap pada tahap analisis terstruktur. Prinsip dari pemrograman terstruktur adalah jika suatu proses telah sampai pada suatu titik/langkah tertentu, maka proses selanjutnya tidak boleh mengeksekusi langkah sebelumnya/kembali lagi ke baris sebelumnya, kecuali pada langkah-langkah untuk proses berulang (*Loop*).

2.8.1 DFD (Data Flow Diagram)

Data Flow Diagram (DFD) atau Diagram Arus Data adalah suatu model logika data atau proses yang dibuat untuk menggambarkan dari mana asal data dan kemana tujuan data yang keluar dari sistem, dimana data disimpan, dimana data dihapus, proses apa yang menghasilkan data tersebut dan interaksi antara data yang tersimpan dan proses yang dikarenakan pada data tersebut.

DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir atau dimana data tersebut disimpan.

DFD merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur. Kelebihan utama pendekatan aliran data, yaitu :

1. Kebebasan dari menjalankan implementasi teknis sistem.
2. Pemahaman lebih jauh mengenal keterkaitan satu sama lain dalam sistem dan subsistem.
3. Mengkomunikasikan pengetahuan sistem yang ada dengan pengguna melalui diagram aliran data.
4. Menganalisis sistem yang diajukan untuk menentukan apakah data-data dan proses yang diperlukan sudah ditetapkan.

Dibawah ini terdapat istilah-istilah untuk DFD yaitu:

1. *Entity*: *Terminator* atau *Source* atau *Destination* atau dikenal juga dengan *External Entity*, berupa orang, organisasi atau sistem lain yang berada diluar batas sistem yang berinteraksi dengan sistem yang sedang dikembangkan.
2. *Proses*: Suatu proses adalah kegiatan atau kerja yang dilakukan oleh orang, prosedur atau alat yang digunakan untuk mentransformasikan data.
3. *Data Flow*: (Arus Data), data yang mengalir dengan arah tertentu dari asal ke tujuan. Data yang mengalir dapat berupa dokumen, surat atau bentuk lainnya.
4. Data *Store*: (Penyimpanan Data), digunakan untuk menyimpan dan mengambil data oleh proses. Data yang disimpan dapat berupa data yang terkomputerisasi maupun tidak terkomputerisasi.

2.8.2 Komponen Terminator/Entitas

Terminator mewakili entitas eksternal yang berkomunikasi dengan sistem yang sedang dikembangkan. Biasanya terminator dikenal dengan nama entitas luar (*external entity*). Sesuatu yang berada diluar system, tapi memberikan data kedalam sistem atau memberikan data dari sistem. Disimbolkan dengan notasi kotak [9].

2.8.3 Komponen Proses

Komponen proses merupakan apa yang dikerjakan oleh system, proses dapat mengolah data atau aliran data masuk menjadi aliran data keluar. Proses berfungsi mentransformasikan satu atau beberapa data masukan menjadi satu atau beberapa data keluaran sesuai dengan spesifikasi data yang dinginkan [9].

2.8.4 Komponen Data Flow/Alur Data

Arus data merupakan tempat mengalirnya informasi dan digambarkan dengan garis yang menghubungkan komponen dari sistem [9]. Arus data ditunjukan dengan arus panah dan garis diberi nama atas arus data yang mengalir. Arus data ini mengalir diantara proses, data store dan menunjukan arus data dari data yang berupa masukan untuk sistem atau hasil proses sistem.

2.9 CT-Scan

Computer Tomographu Scanner (CT-Scan) ditemukan oleh Allan Cormack dan Geoffrey Hounsfiled pada tahun 1970. Penggunaan CT-Scan pada bidang radiologi telah mengalami kemajuan yang sangat pesat. Adapun asal muasal teknik ini digunakan untuk pemeriksaan otak. Suatu tabung sinar X tergandeng dengan dua detektor memancar dua potongan yang berdekatan dengan kepala menggunakan gerakan translasi. Dampak dari penggunaan sinar-X adalah untuk menghitamkan film negative dengan warna yang lebih terang daripada benda-benda yang menyerap sinar lebih sedikit.

2.10 MatLab

Matrix laboratory atau MatLab adalah sebuah program untuk menganalisis dan mengkomputasi data numerik. MatLab juga merupakan suatu bahasa pemrograman matematika lanjutan, yang dibentuk dengan dasar pemikiran yang menggunakan sifat dan bentuk matriks. Pada awalnya program aplikasi MATLAB ini merupakan suatu intergace untuk koleksi rutin-rutin numeric dari proyek LINPACK dan EISPACK dan dikembangkan dengan bahasa FORTRAN, namun sekarang ini MATLAB merupakan produk komersial perusaan Mathwords.

Matlab telah berkembang menjadi sebuah environtment pemrograman yang canggih yang berisi fungsi-fungsi *built-in* untuk melakukan tugas pengelolaan sinyal, aljabar linear dan kalkulasi matematis lainnya. Matlab juga bersifat ekstensibel, yang artinya seorang pengguna dapat menulis fungsi baru untuk menambahkan pada library. Ketika fungsi-fungsi built-in yang tersedia tidak dapat melakukan tugas tertentu. Maka kemampuan program yang dibutuhkan tidak terlalu sulit bila kita telah memiliki pengalaman dalam pemrograman bahasa lain seperti C, PASCAL dan FORTRAN.

Dalam pengoperasiannya, Matlab memiliki *toolbox* yang dapat digunakan untuk mempermudah pengguna, termasuk membantu pengguna dalam mengimplementasikan metode atau algoritma tertentu. *Toolbox* Matlab yang digunakan untuk membantu penelitian ini adalah:

1. Canny Edge Detection Toolbox

   Canny Edge Detection Toolbox adalah *toolbox* Matlab yang dikembangkan oleh Peter Kovesi [7] yang dapat digunakan untuk melakukan deteksi tepi citra dengan metode *canny*. Untuk menggunakan fungsi ini, nilai dari parameter *sigma, rad* (*radius*), *T1* (*Upper-threshold*), dan *T2* (*Lower-threshold*) harus ditentukan sesuai kebutuhan sistem.

2. SVMTrain

   SVMTrain adalah fungsi yang digunakan untuk melakukan pelatihan SVM. Data input yang dilatih dengan fungsi ini akan menghasilkan data SVM berupa *support vector* yang dapat digunakan untuk melakukan pengujian klasifikasi SVM. Secara *default* SVMTrain menggunakan fungsi kernel *Linier*.

3. SVMClassiffy

   SVMClassiffy adalah fungsi klasifikasi dengan menggunakan metode SVM. Data *support vector* dari hasil SVMTrain diuji menggunakan fungsi SVMClassiffy untuk mendapatkan kelas untuk data uji.

2.11 Teknik Pengujian

Salah satu teknik pengujian perangkat lunak adalah pengujian *Blackbox*. Pengujian *blackbox* digunakan untuk menguji fungsi-fungsi khusus dari perangkat lunak yang dirancang. Pada teknik ini, kebenaran perangkat lunak yang diuji hanya dilihat berdasarkan keluaran yang dihasilkan dari data atau kondisi masukan yang diberikan untuk fungsi yang ada tanpa melihat bagaimana proses untuk mendapatkan keluaran tersebut. Dari keluaran yang dihasilkan, kemampuan program dalam memenuhi kebutuhan pemakai dapat diukur sekaligus dapat diketahui kesalahan-kesalahannya. Beberapa jenis kesalahan yang dapat diidentifikasi:

1. Fungsi tidak benar atau hilang.
2. Kesalahan antar muka.
3. Kesalahan pada struktur data (pengaksesan basis data).
4. Kesalahan inisialisasi dan akhir program.
5. Kesalahan performasi.

Walaupun sulit untuk menelusuri kesalahan yang mungkin didapat, teknik pengujian *blackbox* lebih sering dipilih untuk menguji perangkat lunak karena kemudahan dalam pelaksanaannya.

2.11.1 Pengujian Akurasi

Pengujian akurasi dilakukan dengan *confusion matrix* yaitu sebuah tabel yang menyatakan jumlah data uji yang benar diklasifikasikan dan jumlah data uji yang salah diklasifikasikan. Contoh *confusion matrix* untuk klasifikasi biner ditunjukkan pada tabel dibawah ini.

Tabel 2.10 Confusion Matrix

		Kelas Prediksi	
		1	0
Kelas Sebenarnya	1	TP	FN
	0	FP	TN

Keterangan untuk tabel 2.10 dinyatakan sebagai berikut:

True Posstive (TP), yaitu jumlah dokumen dari kelas 1 yang benar dan diklasifikan sebagai kelas 1.

True Negative (TN), yaitu jumlah dokumen dari kelas 0 yang benar diklasifikasikan sebagai kelas 0.

False Positive (FP), yaitu jumlah dokumen dari kelas 0 yang salah diklasifikasikan sebagai kelas 1.

False Negative (FN) yaitu jumlah dokumen dari kelas 1 yang salah diklasifikasikan sebagai kelas 0.

Perhitungan akurasi dinyatakan dengan rumus

Akurasi = $\frac{TP+TN}{TP+FN+FP+TN}$ * 100%