BAB II

LANDASAN TEORI

2.1 Tembakau

Sebagai produk yang memiliki nilai jual pada daunnya, maka perlu diperhatikan pada kesehatan daun tembakau tersebut. Penurunan kualitas daun tembakau akan mempengaruhi nilai jual daun itu sendiri. Perlu dilakukan perawatan terhadap daun tembakau agar tidak terjangkit penyakit baik yang disebabkan oleh jamur, hama ataupun virus. Penelitian ini memberikan kemudahan bagi petani tembakau ataupun masyarakat yang ingin mengetahui jenis penyakit yang menjangkit daun tembakau yang hampir memiliki gejala serupa. Metode analisis ini mengidentifikasi jenis penyakit dari daun tembakau yang memiliki gejala yang hampir serupa pada daun yang dijangkit. Penyakit tersebut ialah :

2.1.1 Penyakit Lanas

Lanas adalah penyakit tanaman tembakau yang disebabkan oleh cendawan *Phytophthora parasitica var. nicotonae*. Menurut Haryono Semangun, penyakit ini dapat timbul pada tanaman tembakau berbagai umur, baik pada pembibitan maupun yang telah ditanam di perkebunan. Gejala penyakit ini berupa antara lain :

1. Pada tanaman bibit, warna daun mula-mula berwarna hijau kelabu kotor.
2. Dapat meluas dengan cepat sehingga gejalanya seperti tersiram air panas.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

3. Pada tanaman yang lebih tua, biasanya gejala pembusukan hanya sebatas pada leher akar. Bagian yang busuk berwarna coklat kehitaman dan agak berlekuk.
4. Semua daun kemudian layu mendadak.
5. Jika bagian pangkal batang dibelah maka empulur tampak mengering dan mengamar.
6. Jika tidak segera dipetik, maka lanas akan menjalar ke bagian batang
7. Kelayuan pada tanaman akan terjadi.

Gambar 2.1. Daun yang terjangkit penyakit lanas (www.forestryimages.org)

INSTITUT BISNIS INFORMATIKA stikom SURABAYA

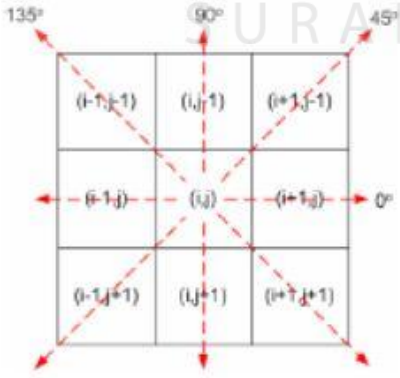
2.1.2 Bercak Karat

Bercak karat adalah penyakit daun tembakau yang disebabkan oleh jamur *Alternaria longipes*. Gejalanya timbul bercak – bercak coklat pada daunnya. Selain menyerang daun, jamur ini juga menyerang batang dan biji. Dan jamur ini selain menyerang tanaman dewasa juga menyerang tanaman persemaian. (Maulidina, 2008)

Gambar 2.2. Daun yang terjangkit bercak karat (www.padil.gov.au)

2.2 *Gray Level Co-occurrence Matrix*

Gray Level Co-Occurrence Matrix (GLCM) merupakan metode yang paling sering digunakan untuk analisis tekstur yang diperkenalkan oleh Haralick tahun 1973 (Mulkan, 2012). GLCM adalah matriks yang menggambarkan frekuensi munculnya pasangan dua piksel dengan intensitas tertentu dalam jarak d dan orientasi arah dengan sudut θ tertentu dalam citra (Hartadi, 2011). Biasanya, ada empat arah yang digunakan, yaitu 0^o, 45^o, 90^o dan 135^o yang ditunjukkan pada Gambar 2.3.



Gambar 2.3. Hubungan ketetanggaan antar piksel sebagai fungsi orientasi dan jarak spasial (Ganis, 2011)

Kookurensi berarti kejadian bersama, yaitu jumlah kejadian satu level nilai piksel bertetangga dengan satu level nilai piksel lain dalam jarak (*d*) dan

orientasi sudut (θ) tertentu . Jarak dinyatakan dalam piksel dan orientasi dinyatakan dalam derajat (Budiarso, 2010). Orientasi dibentuk dalam empat arah sudut dengan interval sudut 45°, yaitu 0°, 45°, 90°, dan 135°. Sedangkan jarak antar piksel biasanya ditetapkan sebesar 1 piksel, 2 piksel, 3 piksel dan seterusnya.

Matriks kookurensi merupakan matriks bujur sangkar dengan jumlah elemen sebanyak kuadrat jumlah level intensitas piksel pada citra. Setiap titik (*i*,*j*) pada matriks kookurensi berorientasi berisi peluang kejadian piksel bernilai *i* bertetangga dengan piksel bernilai *j* pada jarak *d* serta orientasi (180 − θ) (Pradnyana, 2015). Sebagai contoh matriks 4×4 memiliki memiliki tingkat keabuan dari 0 sampai 6. Matriks kookurensi akan dihitung dengan nilai d=1 dan θ=0^o. Jumlah frekuensi munculnya pasangan (*i*,*j*) dihitung untuk keseluruhan matriks. Jumlah kookurensi diisikan pada matriks GLCM pada posisi sel yang bersesuaian. Gambar 2.4, gambar 2.5, dan gambar 2.6 secara berurutan menunjukkan contoh proses perhitungan matriks kookurensi.

$$\begin{bmatrix} 3 & 4 & 1 & 1 \\ 6 & 0 & 1 & 1 \\ 5 & 6 & 2 & 2 \\ 2 & 2 & 3 & 3 \end{bmatrix}$$

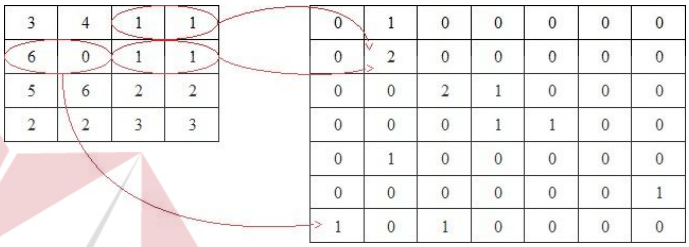
Gambar 2.4. Matriks bebas, matriks I

Karena matriks I memiliki tujuh aras keabuan, maka jumlah nilai piksel tetangga dan nilai piksel referensi pada area kerja matriks berjumlah tujuh. Berikut adalah area kerja matriks.

Nilai Piksel Referensi \ Nilai Piksel Tetangga	0	1	2	3	4	5	6
0	0,0	0,1	0,2	0,3	0,4	0,5	0,6
1	1,0	1,1	1,2	1,3	1,4	1,5	1,6
2	2,0	2,1	2,2	2,3	2,4	2,5	2,6
3	3,0	3,1	3,2	3,3	3,4	3,5	3,6
4	4,0	4,1	4,2	4,3	4,4	4,5	4,6
5	5,0	5,1	5,2	5,3	5,4	5,5	5,6
6	6,0	6,1	6,2	6,3	6,4	6,5	6,6

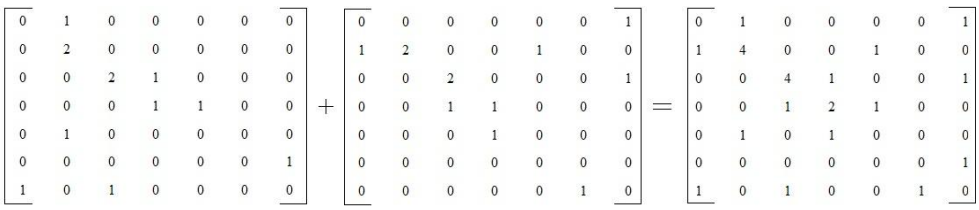
Gambar 2.5. Area kerja matriks

Hubungan spasial untuk d = 1 dan θ = 0^o pada matriks diatas dapat dituliskan dalam matriks berikut.



Gambar 2.6. Pembentukan matriks kookurensi dari matrik I

Sudut orientasi menentukan arah hubungan tetangga dari piksel-piksel referensi, orientasi θ = 0^o berarti acuan dalam arah horizontal atau sumbu x positif dari piksel-piksel referensi. Acuan sudut berlawanan arah jarum jam. Angka 2 pada (1,1) berarti jumlah hubungan pasangan (1,1) pada matriks asal berjumlah 2. Matriks kookurensi yang didapat kemudian ditambahkan dengan matriks transposenya untuk menjadikannya simetris terhadap sumbu diagonal. Berikut ini adalah (*i*, *j*) dari matriks asal ditambahkan dengan transposenya, dan hasilnya simetris, seperti pada gambar 2.7.



Gambar 2.7. GLCM simetris

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Matriks yang telah simetris selanjutnya harus dinormalisasi, elemen-elemennya dinyatakan dengan probabilitas. Nilai elemen untuk masing-masing sel dibagi dengan jumlah seluruh elemen spasial. Matriks yang telah dinormalisasi diperlihatkan pada gambar 2.8. Nilai 0,1667 pada (1,1) diperoleh dari 4 dibagi jumlah seluruh nilai piksel yaitu 24.

0.0000	0.0417	0.0000	0.0000	0.0000	0.0000	0.0417
0.0417	0.1667	0.0000	0.0000	0.0417	0.0000	0.0000
0.0000	0.0000	0.1667	0.0417	0.0000	0.0000	0.0417
0.0000	0.0000	0.0417	0.0833	0.0417	0.0000	0.0000
0.0000	0.0417	0.0000	0.0417	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0417
0.0417	0.0000	0.0417	0.0000	0.0000	0.0417	0.0000

Gambar 2.8. GLCM simetris ternormalisasi dari matriks I

2.2.1. Ekstraksi Fitur

Ekstraksi fitur merupakan langkah awal dalam melakukan klasifikasi dan interpretasi citra. Proses ini berkaitan dengan kuantisasi karakteristik citra ke dalam sekelompok nilai ciri yang sesuai. Analisis tekstur lazim dimanfaatkan sebagai proses antara untuk melakukan klasifikasi dan interpretasi citra (Wijanarko, 2013). Suatu proses klasifikasi citra berbasis analisis tekstur pada umumnya membutuhkan tahapan ekstraksi ciri, yang dapat terbagi dalam tiga macam metode berikut:

1. Metode statistik

Metode statistik menggunakan perhitungan statistik distribusi derajat keabuan (*histogram*) dengan mengukur tingkat kekontrasan, granularitas, dan kekasaran suatu daerah dari hubungan ketetanggaan antar piksel di dalam citra. Statistik ini penggunaannya tidak terbatas untuk tekstur-tekstur alami yang tidak terstruktur dari sub pola dan himpunan aturan (*mikrostruktur*).

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

2. Metode spektral

Metode spektral berdasarkan pada fungsi autokorelasi suatu daerah atau *power* distribution pada domain transformasi *Fourier* dalam mendeteksi periodisitas tekstur.

3. Metode struktural

Analisis dengan metode ini menggunakan deskripsi primitif tekstur dan aturan intaktik. Metode struktural banyak digunakan untuk pola-pola makrostruktur.

Berdasarkan orde statistiknya, analisis tekstur dapat dikategorikan menjadi 3, yaitu analisis tekstur orde satu, orde dua, dan orde tiga.

1. Statistik orde-kesatu merupakan metode pengambilan ciri yang didasarkan pada karakteristik histogram citra. Histogram menunjukkan probabilitas kemunculan nilai derajat keabuan piksel pada suatu citra, dengan mengabaikan hubungan antar piksel tetangga. Analisa tekstur orde satu lebih baik dalam merepresentasikan tekstur citra dalam parameter-parameter terukur, seperti *mean*, *skewness*, *variance*, *kurtosis* dan *Entropy* (Kusuma, 2011).

2. Statistik orde-kedua mempertimbangkan hubungan antara dua piksel (piksel yang bertetangga) pada citra. Untuk kebutuhan analisanya, analisis tekstur orde dua memerlukan bantuan matriks kookurensi (*matrix co-occurence*) untuk citra keabuan, biasanya disebut GLCM. Analisa tekstur orde dua lebih baik dalam merepresentasikan tekstur citra dalam parameter-parameter terukur, seperti kontras, korelasi, homogenitas, entropi, dan energy (Albregtsen, 2008).

3. Statistik orde-ketiga dan yang lebih tinggi, mempertimbangkan hubungan antara tiga atau lebih piksel, hal ini secara teoritis memungkinkan tetapi belum biasa diterapkan (Febrianto, 2012).

Ekstraksi ciri statistik orde kedua dilakukan dengan matriks kookurensi, yaitu suatu matriks antara yang merepresentasikan hubungan ketetanggaan antar piksel dalam citra pada berbagai arah orientasi dan jarak spasial (Albregtsen, 2008). Matriks kookurensi merupakan matriks berukuran L x L (L menyatakan banyaknya tingkat keabuan) dengan elemen $P(x_1,x_2)$ yang merupakan distribusi probabilitas bersama (*join probability distribution*) dari pasangan titik-titik dengan tingkat keabuan x_1 yang berlokasi pada koordinat (j,k) dengan x_2 yang berlokasi pada koordinat (m,n). Koordinat pasangan titik-titik tersebut berjarak *r* dengan sudut θ. Histogram tingkat kedua $P(x_1,x_2)$ dihitung dengan pendekatan sebagai berikut :

$$P(x_1,x_2) = \frac{\text{banyaknya pasangan titik-titik dengan tingkat keabuan } x_1 \text{ dan } x_2}{\text{banyaknya titik pada daerah suatu citra}} \quad \text{...(2.1)}$$

Berikut ini ketentuan untuk hubungan pasangan titik-titik dengan sudut 0^o, 45^o, 90^o, dan 135^o pada jarak *r* (Putra, 2009).

$$P_{0^o,r}(x_1,x_2) = \left| \begin{Bmatrix} ((j,k),(m,n)) \in R: \\ j-m=0, |k-n|=r, \\ f_{j,k}=x_1, f_{m,n}=x_2 \end{Bmatrix} \right| \quad \text{.....(2.2)}$$

$$P_{45^o,r}(x_1,x_2) = \left| \begin{Bmatrix} ((j,k),(m,n)) \in R: \\ (j-m=r, |k-n|=-r, \\ or\ (j-m=-r, k-n=r), \\ f_{j,k}=x_1, f_{m,n}=x_2 \end{Bmatrix} \right| \quad \text{.....(2.3)}$$

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

$$P_{90^o,r}(x_1,x_2) = \left| \begin{Bmatrix} ((j,k),(m,n)) \in R: \\ |j-m| = r, k-n = 0, \\ f_{j,k} = x_1, f_{m,n} = x_2 \end{Bmatrix} \right| \quad \text{.....(2.4)}$$

$$P_{135^o,r}(x_1,x_2) = \left| \begin{Bmatrix} ((j,k),(m,n)) \in R: \\ j-m = r, k- \ n \ = r \\ or\ (j-m = \ -r, k-n = \ -r), \\ f_{j,k} = x_1, f_{m,n} = x_2 \end{Bmatrix} \right| \quad \text{.....(2.5)}$$

dimana :

r : jarak piksel

$P(x_1,x_2)$: merupakan elemen matriks.

x_1: pasangan titik-titik dengan tingkat keabuan pada koordinat *(j, k)*.

x_2: pasangan titik-titik dengan tingkat keabuan pada koordinat (*m, n*).

GLCM adalah suatu matriks yang elemen-elemennya merupakan jumlah pasangan piksel yang memiliki tingkat kecerahan tertentu, di mana pasangan piksel itu terpisah dengan jarak *d*, dan dengan suatu sudut inklinasi *θ*. Dengan kata lain, matriks kookurensi adalah probabilitas munculnya *gray* level *i* dan *j* dari dua piksel yang terpisah pada jarak *d* dan sudut *θ*.

2.2.2. Fitur-fitur GLCM

Setelah memperoleh matriks kookurensi tersebut, selanjutnya dapat dihitung ciri statistic orde dua yang merepresentasikan citra yang diamati. Haralick dkk mengusulkan berbagai jenis ciri tekstural yang dapat diekstraksi dari matriks kookurensi (Albregtsen, 2008). Ada banyak perhitungan ciri statistik yang disediakan oleh GLCM, tetapi dalam penelitian ini digunakan perhitungan 4 ciri statistik orde dua, yaitu *Energy, Contrast, Correlation, Homogeneity.*

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

1. Contrast

Contrast menunjukkan ukuran penyebaran (momen inersia) elemen-elemen matriks citra. Jika letaknya jauh dari diagonal utama, maka nilai kekontrasannya besar. Secara visual, nilai kekontrasan adalah ukuran variasi antar derajat keabuan suatu daerah citra dan didefinisikan dengan :

$$\sum_{i,j=0}^{N-1} P_{(i,j)}(i-j)^2 \quad \text{.....(2.6)}$$

Dimana :
$P_{(i,j)}$ = nilai elemen matriks kookurensi

2. Correlation

Correlation Menunjukkan ukuran ketergantungan linear derajat keabuan citra sehingga dapat memberikan petunjuk adanya struktur linear dalam citra.

$$\sum_{i,j=0}^{N-1} P_{(i,j)} \frac{(i-\mu_x)(j-\mu_y)}{\sqrt{(\sigma_x)^2(\sigma_y)^2}} \quad \text{.....(2.7)}$$

Dimana :
μ_x adalah nilai rata-rata elemen kolom pada matriks $P_\theta(i , j)$.
μ_y adalah nilai rata-rata elemen baris pada matriks $P_\theta(i , j)$.
σ_x adalah nilai standar deviasi elemen kolom pada matriks $P_\theta(i , j)$.
σ_y adalah nilai standar deviasi elemen baris pada matriks $P_\theta(i , j)$.

3. Energy

Energy menunjukkan ukuran konsentrasi pasangan intensitas pada matriks kookurensi, dan didefinisikan dengan :

$$\sum_{i,j=0} P(i,j)^2 \quad \text{.....(2.8)}$$

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Nilai *energy* makin membesar bila pasangan piksel yang memenuhi syarat matriks intensitas kookurensi terkonsentrasi pada beberapa koordinat dan mengecil bila letaknya menyebar.

4. Homogeneity

Homogeneity menunjukkan kehomogenan variasi intensitas dalam citra. Citra homogen akan memiliki nilai *homogeneity* yang besar. Nilai *homogeneity* membesar bila variasi intensitas dalam citra mengecil dan sebaliknya

$$\sum_{i,j=0}^{N-1} \frac{P_{(i,j)}}{1+(i-j)^2} \quad \text{.....(2.9)}$$

2.3 *Support Vector Machine*

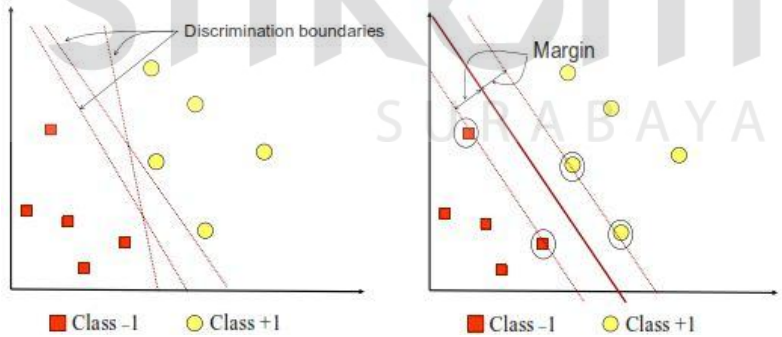
Support Vector Machine (SVM) adalah suatu teknik yang relatif baru (1995) untuk melakukan prediksi, baik dalam kasus klasifikasi maupun regresi, yang sangat populer belakangan ini. SVM berada dalam satu kelas dengan ANN dalam hal fungsi dan kondisi permasalahan yang bisa diselesaikan. Keduanya masuk dalam kelas *supervised learning*. Baik para ilmuwan maupun praktisi telah banyak menerapkan teknik ini dalam menyelesaikan masalah-masalah nyata dalam kehidupan sehari-hari (Santosa, 2005). Terbukti dalam banyak implementasi, SVM memberi hasil yang lebih baik dari ANN, terutama dalam hal solusi yang dicapai. ANN menemukan solusi berupa lokal optimal sedangkan SVM menemukan solusi yang global optimal. Tidak heran bila menjalankan ANN, solusi dari setiap *training* selalu berbeda. Hal ini disebabkan solusi lokal optimal yang dicapai tidak selalu sama. SVM selalu mencapai solusi yang sama untuk setiap *running*.

Teknik ini berusaha untuk menemukan fungsi pemisah (*klasifier*) yang optimal yang bisa memisahkan dua set data dari dua kelas yang berbeda. Teknik ini menarik dalam bidang data *mining* maupun *machine learning* karena performansinya yang meyakinkan dalam memprediksi kelas suatu data baru. Dalam hal ini fungsi pemisah yang dicari adalah fungsi linier (Santosa, 2005). Fungsi ini bisa didefinisikan sebagai

$$g(x) := sgn(f(x)) \quad \text{.....(2.10)}$$

dengan $f(x)=w^T x + b$, dimana $x, w \in R^n$ and $b \in R$

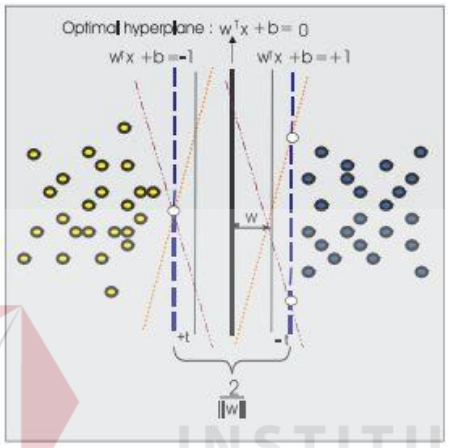
Masalah klasifikasi ini bisa dirumuskan sebagai berikut: ingin menemukan set parameter (w, b) sehingga $f(x_i) = <w, x> +b = y_i$ untuk semua i. Dalam teknik ini dicoba menemukan fungsi pemisah (*klasifier/hyperplane*) terbaik diantara fungsi yang tidak terbatas jumlahnya untuk memisahkan dua macam obyek. *Hyperplane* terbaik adalah *hyperplane* yang terletak di tengah-tengah antara dua set obyek dari dua kelas (Budi Santosa, 2005).



Gambar 2.9. SVM mencari *hyperplane* terbaik untuk memisahkan kedua *class* -1 dan +1 (Nugroho, 2003)

Mencari *hyperplane* terbaik ekuivalen dengan memaksimalkan margin atau jarak antara dua set obyek dari kelas yang berbeda. Margin adalah jarak antara *hyperplane* tersebut dengan pattern terdekat dari masing-masing *class*. Jika $wx_1 + b = +1$ adalah *hyperplane*-pendukung (*supporting hyperplane*) dari kelas +1

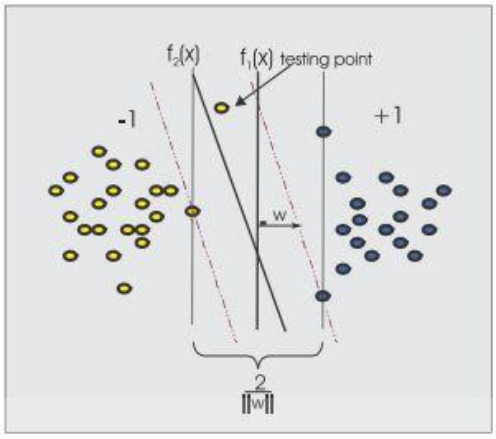
($wx_1 + b = +1$) dan $wx_2 + b = -1$ *hyperplane*-pendukung dari kelas -1($wx_2 + b = -1$), margin antara dua kelas dapat dihitung dengan mencari jarak antara kedua *hyperplane*-pendukung dari kedua kelas. Secara spesifik, margin dihitung dengan cara berikut ($wx_1 + b = +1$) - ($wx_2 + b = -1$) ⇒ $w(x_1 - x_2) = 2$ ⇒ $(\frac{w}{||w||}(x_1 - x_2)) = \frac{2}{||w||}$ (Santosa, 2005).



Gambar 2.10. SVM mencari fungsi pemisah yang optimal untuk obyek yang bisa dipisahkan secara linier (Santosa, 2005)

Gambar 2.9 memperlihatkan bagaimana SVM bekerja untuk menemukan suatu fungsi pemisah dengan margin yang maksimal. Untuk membuktikan bahwa memaksimalkan margin antara dua set obyek akan meningkatkan probabilitas pengelompokkan secara benar dari data *testing*. Pada dasarnya jumlah fungsi pemisah ini tidak terbatas banyaknya. Misalkan dari jumlah yang tidak terbatas ini diambil dua saja, yaitu $f_1(x)$ dan $f_2(x)$ (lihat gambar 2.10). Fungsi f_1 mempunyai margin yang lebih besar dari pada fungsi f_2. Setelah menemukan dua fungsi ini, sekarang suatu data baru masuk dengan keluaran -1. Harus dikelompokkan apakah data ini ada dalam kelas -1 atau +1 menggunakan fungsi pemisah yang sudah ditemukan. Dengan menggunakan f_1, akan dikelompokkan data baru ini di kelas -1 yang berarti benar mengelompokkannya. Sekarang coba menggunakan f_2 ,akan

menempatkan di kelas +1 yang berarti salah. Dari contoh sederhana ini dapat dilihat bahwa memperbesar margin bisa meningkatkan probabilitas pengelompokkan suatu data secara benar. (Santosa, 2005)



Gambar 2.11. Memperbesar margin bisa meningkatkan probabilitas pengelompokkan (Santosa, 2005)

2.3.1. Karakteristik SVM

Menurut Nugroho, A., Witarto, A., dan Handoko, D., (2003), karakteristik dari SVM antara lain :

1. Secara prinsip SVM adalah *linear classifier*.
2. *Pattern recognition* (pengenalan pola) dilakukan dengan mentransformasikan data pada *input space* ke ruang yang berdimensi lebih tinggi, dan optimisasi dilakukan pada ruang *vector* yang baru tersebut. Hal ini membedakan SVM dari solusi *pattern recognition* pada umumnya, yang melakukan optimisasi parameter pada ruang/hasil transformasi yang berdimensi lebih rendah dari dimensi *input space*.
3. Menerapkan strategi *Structural Risk Minimization* (SRM).
4. Prinsip kerja SVM pada dasarnya hanya mampu menangani klasifikasi dua *class*.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

2.3.2. Kelebihan dan Kekurangan SVM

Kelebihan-kelebihan SVM adalah (Santika, 2012) :

1. Generalisasi

Generalisasi didefinisikan sebagai kemampuan suatu metode untuk mengklasifikasikan suatu *pattern*, yang tidak termasuk data yang dipakai dalam fase pembelajaran metode tersebut.

2. *Curse of Dimensionality*

Curse of Dimensionality didefinisikan sebagai masalah yang dihadapi suatu metode *pattern recognition* dalam mengestimasikan parameter (misal jumlah *hidden neuron* pada *neural network, stopping criteria* dalam proses pembelajaran, dsb) dikarenakan jumlah sampel data yang relatif sedikit dibandingkan dimensional ruang vektor data tersebut. Semakin tinggi dimensi dari ruang *vector* informasi yang diolah, membawa konsekuensi dibutuhkannya jumlah data dalam proses pembelajaran.

3. Feasibility

SVM dapat diimplementasikan realtif mudah, karena proses penentuan *support vector* dapat dirumuskan dalam QP *problem* (*Quadratic Progamming*). Dengan demikian, jika kita memiliki *library* untuk menyelesaikan QP *problem*, dengan sendirinya SVM dapat diimplementasikan dengan mudah.

Sedangkan kekurangan SVM adalah (Santika, 2012) :

1. Sulit dipakai dalam *problem* berskala besar. Skala besar dalam hal ini dimaksudkan dengan jumlah sample yang diolah.

2. SVM secara teorik dikembangkan untuk *problem* klasifikasi dengan dua *class* atau lebih. Namun demikian, masing-masing strategi ini memiliki kelemahan, sehingga dapat dikatakan penelitian dan pengembangan SVM pada *multiclass problem* masih merupakan tema penelitian ang masih terbuka.

2.4 Formulasi Matematis

Secara matematika, formulasi problem optimisasi SVM untuk kasus klasifikasi linier di dalam *primal space* adalah

$$\min \frac{1}{2} ||w||^2 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2.11)$$

Subject to

$$Y_i (wx_i + b) \geq 1, i = 1, \ldots\ldots, l.$$

Dimana x_i adalah data input, y_1 adalah keluaran dari data x_i, w dan b adalah parameter-parameter yang dicari nilainya. Dalam formulasi di atas, diinginkan peminimalan fungsi tujuan (*obyektif function*) $\frac{1}{2} ||w||^2$ atau memaksimalkan kuantitas $||w||^2$ atau $w^T w$ dengan memperhatikan pembatas y_i(wxi + b) $\geq$ 1. Bila output data y_i = + 1, maka pembatas menjadi (wx_i + b) $\geq$ 1. Sebaliknya bila y_i = -1, pembatas menjadi (wx_i + b) $\leq$ -1. Di dalam kasus yang tidak feasible (*infeasible*) dimana beberapa data mungkin tidak bisa dikelompokkan secara benar, formulasi matematikanya menjadi berikut :

$$\min \frac{1}{2} ||w||^2 + C \sum_{i=1}^{l} t_i \ldots\ldots\ldots\ldots\ldots\ldots(2.12)$$

Subject to

$$Y_i (wx_i + b) + t_i \geq 1, t_i \geq 0, i = 1, \ldots\ldots, l.$$

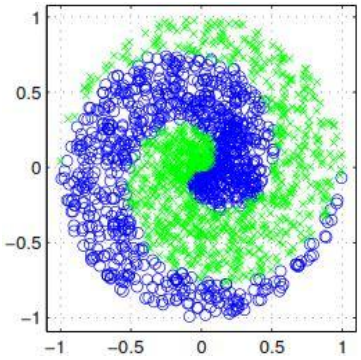
dimana t_i adalah variabel *slack*. (Santosa, 2005)

Dengan formulasi ini diinginkan pemaksimalan margin antara dua kelas dengan meminimalkan $||w||^2$. Formulasi ini berusaha meminimalkan kesalahan klasifikasi (*misclassification error*) yang dinyatakan dengan adanya variabel *slack* t_i, sementara dalam waktu yang sama memaksimalkan margin, $\frac{1}{||w||^2}$. Penggunaan variabel *slack* t_i adalah untuk mengatasi kasus ketidaklayakan (*infeasibility*) dari pembatas (*constraints*) $y_i (wx_i + b) \geq 1$ dengan cara memberi pinalti untuk data yang tidak memenuhi pembatas tersebut. Untuk meminimalkan nilai t_i ini, diberikan pinalti dengan menerapkan konstanta C. Vektor w tegak lurus terhadap fungsi pemisah: wx + b = 0. Konstanta b menentukan lokasi fungsi pemisah relatif terhadap titik asal (*origin*). (Santosa, 2005)

2.5 Metode Kernel

Banyak teknik data *mining* atau *machine learning* yang dikembangkan dengan asumsi kelinieran. Sehingga algoritma yang dihasilkan terbatas untuk kasus-kasus yang linier. Karena itu, bila suatu kasus klasifikasi memperlihatkan ketidaklinieran, algoritma seperti *perceptron* tidak bisa mengatasinya. Secara umum, kasus-kasus di dunia nyata adalah kasus yang tidak linier (Santosa, 2005). Sebagai contoh gambar 2.12, data ini sulit dipisahkan secara linier. Metoda kernel adalah salah satu untuk mengatasinya. Dengan metoda kernel suatu data x di input *space* di-*mapping* ke *feature space* F dengan dimensi yang lebih tinggi melalui map φ sebagai berikut $\varphi : x \rightarrow \varphi(x)$. Karena itu data x di input *space* menjadi φ(x) di *feature space*.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

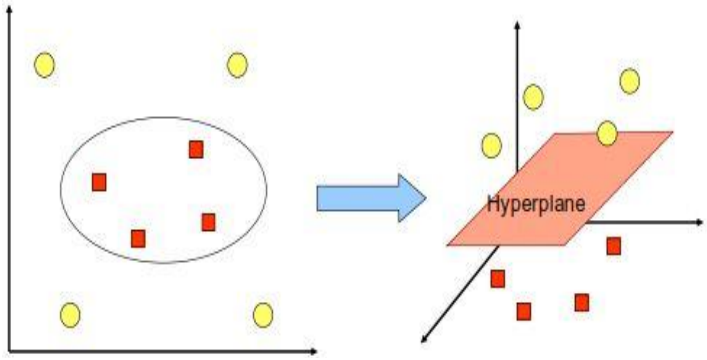


Gambar 2.12. Data spiral yang menggambarkan ketidaklinieran (Santosa, 2005)

Sering kali fungsi φ(x) tidak tersedia atau tidak bisa dihitung, tetapi *dot product* dari dua vektor dapat dihitung baik di dalam input *space* maupun di *feature space*. Dengan kata lain, sementara φ(x) mungkin tidak diketahui, *dot product* < $\varphi(x_1)$, $\varphi(x_2)$ > masih bisa dihitung di *feature space*. Untuk bisa memakai metoda kernel, pembatas (*constraint*) perlu diekspresikan dalam bentuk *dot product* dari vektor data x_i. Sebagai konsekuensi, pembatas yang menjelaskan permasalahan dalam klasifikasi harus diformulasikan kembali sehingga menjadi bentuk *dot product*.

Dalam *feature space* ini *dot product* < . > menjadi < φ(x), φ(x)′ >. Suatu fungsi kernel, k(x, x′), bisa untuk menggantikan *dot product* < φ(x), φ(x)′ >. Kemudian di *feature space*, bisa dibuat suatu fungsi pemisah yang linier yang mewakili fungsi *nonlinear* di *input space* (Santosa, 2005). Gambar 2.13 mendeskripsikan suatu contoh *feature mapping* dari ruang dua dimensi ke *feature space* dua dimensi. Dalam input *space*, data tidak bisa dipisahkan secara linier, tetapi bisa dipisahkan di *feature space*. Karena itu dengan memetakan data ke *feature space* menjadikan tugas klasifikasi menjadi lebih mudah.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA



Gambar 2.13. Suatu kernel *map* mengubah problem yang tidak linier menjadi linier dalam *space* baru untuk mencari *hyperplane* (Nugroho, 2003)

Dalam penelitian ini digunakan dua kernel untuk mencari *hyperplane* terbaik dari dua kelas penyakit yang berbeda, kelas penyakit lanas dan kelas penyakit bercak karat. Fungsi kernel tersbut adalah :

2.5.1. Kernel *Gaussian* (*Radial Basis Function*)

Fungsi radial basis yang sering digunakan adalah fungsi *gaussian* karena mempunyai sifat lokal, yaitu bila *input* dekat dengan rata-rata (pusat), maka fungsi akan menghasilkan nilai satu, sedangkan bila *input* jauh dari rata-rata, maka fungsi memberikan nilai nol (Santika, 2012). Fungsi kernel ini bisa dirumuskan dengan :

$$k(x,x') = exp\left(-\frac{1}{2\sigma^2}|||x - x_i||^2\right) \quad \text{.....(2.13)}$$

2.5.2. Kernel *Polynomial*

Fungsi kernel *polynomial* bersifat *directional*, yaitu *output* tergantung pada arah 2 vektor dalam ruang dimensi rendah. Hal ini disebabkan produksi titik di dalam kernel yang menunjukkan bentuk dua dimensi yang jumlahnya banyak. Semua vektor dengan arah yang sama akan lebih tinggi dari *output* kernelnya,

yang besar dari *output*nya juga ternyata pada besarnya vektor (Santika, 2012). Fungsi kernel *polynomial* dapat dirumuskan :

$$k(x,x') = (x^T x_i + 1)^p \quad \text{.....(2.14)}$$

2.6 MATLAB

Matrix Laboratory atau yang lebih dikenal dengan istilah MATLAB merupakan suatu bahasa pemrograman lanjutan yang dibentuk dengan dasar pemikiran menggunakan sifat dan bentuk dari matriks. Menurut Andik Mabrur dalam artikelnya yang berjudul "Pengolahan Citra Digital Menggunakan MATLAB" (2011), dalam lingkungan perguruan tinggi teknik, MATLAB merupakan perangkat standar untuk memperkenalkan dan mengembangkan penyajian materi matematika, rekayasa dan kelimuan. Di industri, MATLAB merupakan perangkat pilihan untuk penelitian dengan produktifitas yang tinggi, pengembangan dan analisanya. Penggunaan MATLAB meliputi bidang - bidang :

- Matematika dan Komputasi
- Pembentukan Algorithm
- Akusisi Data
- Pemodelan, simulasi, dan pembuatan prototipe
- Analisa data, explorasi, dan visualisasi
- Grafik Keilmuan dan bidang Rekayasa

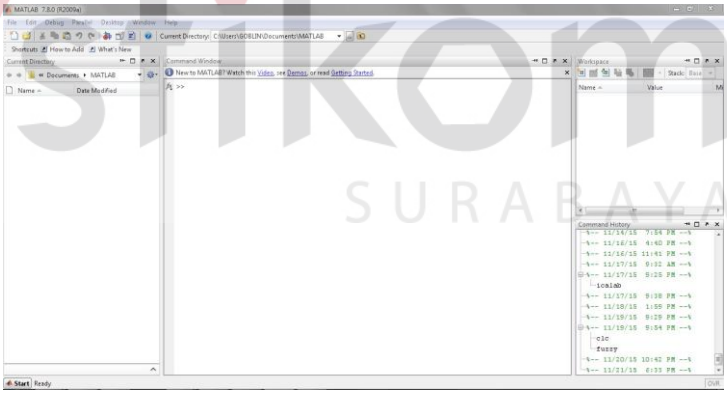
Pada intinya MATLAB merupakan sekumpulan fungsi-fungsi yang dapat dipanggil dan dieksekusi. Fungsi-fungsi tersebut dibagi-bagi berdasarkan kegunaannya yang dikelompokan didalam *toolbox-toolbox* yang ada pada MATLAB. *Toolbox* merupakan kompulan koleksi dari fungsi-fungsi MATLAB

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

(M-files) yang memperluas lingkungan MATLAB untuk memecahkan masalah-masalah tertentu. *Toolbox-toolbox* yang tersedia pada MATLAB antara lain:

- *Signal Processing Toolbox*
- *Control Systems Toolbox*
- *Neural Networks Toolbox*
- *Fuzzy Logic Toolbox*
- *Wavelets Toolbox*
- *Simulation Toolbox*
- *Image Processing Toolbox*

MATLAB juga memiliki sifat *extensible*, dalam arti bahwa pengguna dari MATLAB dapat membuat suatu fungsi baru untuk ditambahkan kedalam *library* jika fungsi-fungsi *built-in* yang tersedia tidak dapat melakukan tugas tertentu.



Gambar 2.14. Tampilan awal MATLAB

2.6.1. *Command Window*

Window ini adalah *window* utama dari MATLAB. Disini adalah tempat untuk menjalankan fungsi, mendeklarasikan variabel, menjalankan proses-proses, serta melihat isi variabel.

2.6.2. *Workspace*

Workspace berfungsi untuk menampilkan seluruh variabel-variabel yang sedang aktif pada saat pemakaian MATLAB. Apabila variabel berupa data matriks berukuran besar maka user dapat melihat isi dari seluruh data dengan melakukan double klik pada variabel tersebut. Matlab secara otomatis akan menampilkan *window* "*array editor*" yang berisikan data pada setiap variabel yang dipilih *user*.

2.6.3. *Current Directory*

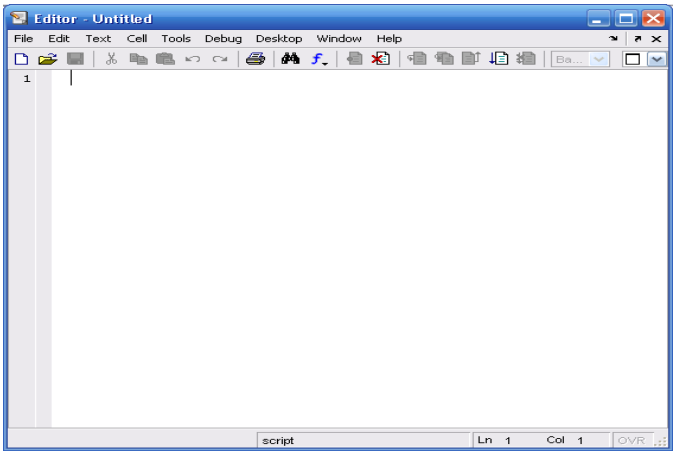
Window ini menampilkan isi dari direktori kerja saat menggunakan MATLAB. Direktori dapat diganti sesuai dengan tempat direktori kerja yang diinginkan. *Default* dari alamat direktori berada dalam folder *works* tempat program files MATLAB berada.

2.6.4. *Command History*

Window ini berfungsi untuk menyimpan perintah-perintah apa saja yang sebelumnya dilakukan oleh pengguna terhadap MATLAB.

2.6.5. *M-File*

Di dalam MATLAB, semua *script* yang akan digunakan dapat disimpan dalam file pada MATLAB dengan ekstensi *.M. *M-File* dapat dipanggil dengan memilih menu *file->new->M-File*. Contoh gambar *M-File* :



Gambar 2.15. Tampilan *M-File*.

Di dalam *M-File*, semua perintah dapat disimpan dan dijalankan dengan menekan tombol atau mengetikan nama *M-File* yang dibuat pada *Command Window.*

Di dalam *M File*, juga dapat dinuliskan fungsi-fungsi yang berisikan berbagai operasi sehingga menghasilkan data yang diinginkan. Format dasar penulisan fungsi sebagai berikut :

Function [Nilai keluaran] = *namaFungsi* (nilai masukan)

% *operasi dari fungsi*

% …

% …

2.7 Operator Dasar

Operator aritmatik dasar yang didukung oleh MATLAB ialah operator dasar perhitungan seperti pada umumnya yang terdiri dari tambah, kurang, kali, bagi dan pangkat. Hirarki operator mengikuti standar aljabar yang umum digunakan seperti :

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

1. Operator dalam kurung diselesaikan terlebih dulu.
2. Operasi pangkat, perkalian, pembagian, penjumlahan dan pengurangan.

2.8 *Variabel*

Variabel berfungsi untuk menyimpan sementara nilai baik angka maupun teks untuk digunakan kembali. MATLAB hanya memiliki dua jenis tipe data, yaitu *numeric* dan *string* (Firmansyah, 2007). Menurut Teguh Widiarsono, (2005:25) penamaan variabel mengikuti rambu-rambu berikut :

1. Menggunakan karakter alfabet (A – Z, a - z), angka (0 - 9), dan garis bawah (_), sebagai nama variabel. Besar kcilnya huruf berpengaruh pada penamaan variabel pada MATLAB, contoh :
   - Jumlah, x1, x2, S_21, H_2_in; merupakan nama variable yang valid.
   - Sinyal1, Sinyal1, SINYAL1; dianggap sebagai 3 variabel yang berbeda.
2. Jangan menggunakan spasi, titik, koma atau operator aritmatik sebagai bagian dari nama.

2.9 *Variabel* Terdefinisi MATLAB

Menurut Teguh Widiarsono (2005:26), di dalam MATLAB telah terdapat beberapa *variabel* yang telah terdefinisi, sehingga bisa langsung pergunakan tanpa perlu mendeklarasikannya lagi. *Variabel* tersebut ialah:

Table 2.1. Variabel yang Telah Terdefinisi Oleh MATLAB

Variabel	Keterangan
Ans	“answer”, digunakan untuk menyimpan hasi perhitungan terakhir
Eps	Bilangan sangat kecil mendekati nol yang merupakan batas akurasi perhitungan MATLAB

Variabel	Keterangan
Pi	Konstanta π, 3.1415926…
Inf	"infinity", bilangan positif tak terhingga, misalkan 1 / 0,2 ^ 5000, dsb.
NaN	"not a number", untuk menyatakan hasil perhitungan yang tak terdefinisi, misalkan 0 / 0 dan inf / inf.
i,j	Unit imajiner, √-1, untuk menyatakan bilangan kompleks

2.10 Fungsi Matematika

Berbagai fungsi matematika yang umum dipergunakan telah terdefinisi di MATLAB, meliputi fungsi eksponensial, logaritma, trigonometri, pembulatan dan fungsi yang berkaitan dengan bilangan kompleks. Contoh abs(x) untuk menghitung nilai absolute dari x, |x| dan sin (x), cos (x), tan (x) dsb sebagai fungsi trigonometri sinus, cosinus dan tangent.

2.11 Matriks

Terdapat tiga jenis format data di MATLAB, yaitu skalar, vektor dan matriks.

1) Skalar, ialah suatu bilangan tunggal.

2) Vektor, ialah sekelompok bilangan yang tersusun 1-dimensi. Dalam MATLAB biasanya disajikan sebagai vektor-baris atau vektor-kolom.

3) Matriks, ialah sekelompok bilangan yang tersusun dalam segi-empat 2-dimensi atau lebih. Di dalam MATLAB, matriks didefinisikan dengan jumlah baris dan kolom.

Matriks didefinisikan dengan kurung siku ([ ]) dan biasanya dituliskan baris-per-baris. Tanda koma (,) digunakan untuk memisahkan kolom, dan titik-koma (;) untuk memisahkan baris. Biasanya digunakan spasi untuk memisahkan

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

kolom dan menekan *enter* ke baris baru untuk memisahkan baris. MATLAB menyediakan berbagai *command* untuk membuat dan memanipulasi matriks secara efisien. Di antaranya ialah *command* untuk membuat matriks-matriks khusus, manipulasi indeks matriks, serta pembuatan deret.

2.11.1. Matriks Khusus

Berbagai matriks khusus yang kerap dipergunakan dalam perhitungan bisa dibuat secara efisien dengan *command* yang telah ada di MATLAB.

Tabel 2.2. Matriks Khusus

Nama matriks	Keterangan
Ones(n)	Membuat matriks satuan (semua elemen berisi 1) berukuran n x n
Ones(m,n)	Membuat matriks satuan berukuran m x n
Zeros(n)	Membuat matriks nol (semua elemen berisi 0) berukuran n x n
Zeros(m,n)	Membuat matriks no berukuran m x n
Eye(n)	Membuat matriks identitas berukuran n×n (semua elemen diagonal bernilai 1, sementara lainnya bernilai 0).
Rand(n), Rand(m,n)	Membuat matriks n×n, atau m×n, berisi bilangan random terdistribusi uniform pada selang 0 s.d. 1.
Randn(n), Rand(m,n)	Membuat matriks n×n, atau m×n, berisi bilangan random terdistribusi normal dengan mean = 0 dan varians = 1. *Command* ini kerap digunakan untuk membangkitkan derau putih gaussian.
[ ]	Matriks kosong, atau dengan kata lain matriks 0×0; biasa digunakan untuk mendefinisikan variabel yang belum diketahui ukurannya.

2.12 Citra Digital

Menurut Andik Mabrur (2011:9) sebuah citra digital adalah kumpulan piksel-piksel yang disusun dalam larik dua dimensi. Indeks baris dan kolom (x,y)

dari sebuah piksel yang dinyatakan dalam bilangan bulat dan nilai-nilai tersebut mendefinisikan suatu ukuran intensitas cahaya pada titik tersebut. Satuan atau bagian terkecil dari suatu citra disebut piksel (picture element).

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. Piksel (0,0) terletak pada sudut kiri atas pada citra, dimana 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 pixel. Hal ini berlawanan untuk arah vertical dan horizontal yang berlaku pada sistem grafik dalam matematika.

2.12.1. Membaca File Digital

Dalam MATLAB, citra digital direpresentasikan dalam matriks berukuran m x n sesuai dengan ukuran m x n ukuran citra digital dalam piksel. Jadi setiap piksel dari citra digital diwaklili oleh sebuah matriks berukuran 1x1 apabila citra tersebut berupa citra *Grayscale* atau citra *Biner*, dan 3 matriks apabila citra digital berupa citra RGB. (Andik Mabrur, 2011)

Untuk membaca file citra di MATLAB yang berada satu folder dengan file M-file nya, digunakan perintah *imread*() dan disimpan pada variabel "A". Contoh :

A = *imread*('nama_file')

Sedangkan untuk file citra yang berbeda lokasi direktorinya dengan M-file nya maka harus disertakan direktori filenya. Contoh :

A = *imread*('D:/Picture/nama_file')

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Dan untuk menampilkan file citra yang telah dipilih menggunakan perintah :

imshow(A)

2.12.2. *Grayscalling*

Grayscalling adalah teknik yang digunakan untuk mengubah citra berwana (RGB) menjadi bentuk *grayscale* atau tingkat keabuan (dari hitam ke putih). Dengan pengubahan ini, matriks penyusun citra yang sebelumnya 3 matriks akan berubah menjadi 1 matriks saja. Pengubahan dari citra berwarna ke bentuk *grayscale* biasanya mengikuti aturan sebagai berikut :

$$I_{(i,j)} = \frac{R_{(i,j)} + G_{(i,j)} + B_{(i,j)}}{3} \quad \text{.....(2.15)}$$

dimana :

$I_{(i,j)}$ = Nilai intensitas citra *grayscale*

$R_{(i,j)}$ = Nilai intensitas warna merah dari citra asal

$G_{(i,j)}$ = Nilai intensitas warna hijau dari citra asal

$B_{(i,j)}$ = Nilai intensitas warna biru dari citra asal

Untuk melakukan *grayscalling* di MATLAB, bisa menggunakan fungsi :

I = *rgb2gray*(‘variabel_citra’)

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA