

BAB II

LANDASAN TEORI

Dalam bagian ini, akan dijelaskan landasan teori pada penelitian implementasi HOG dan SVM untuk sistem identifikasi mobil pada area ruang henti khusus sepeda motor (RHK).

2.1 Citra Digital

Citra digital merupakan suatu *array* yang berisi nilai-nilai *real* maupun kompleks serta direpresentasikan intensitasnya melalui suatu nilai. Citra digital disusun oleh fungsi $f(x,y)$ yang memiliki ukuran M baris dan N kolom. Nilai x dan y sendiri merupakan nilai koordinat spasial, sementara tiap-tiap piksel pada titik koordinat memiliki amplitude bernilai f yang merupakan tingkatan intensitas dari piksel tersebut (Mirah, 2018).

Sementara menurut (Kadir & Adhi, 2013), secara umum jenis citra yang dapat digunakan dalam pengolahan citra digital, terbagi menjadi 3 jenis, yaitu citra biner, citra *grayscale*, serta citra warna. Ketiganya memiliki kriteria tersendiri yaitu:

1. Citra Berwarna (RGB)

Citra berwarna adalah citra yang tiap-tiap pikselnya diwakili oleh 3 komponen yaitu komponen warna R (merah), G(hijau), dan B(biru). Tiap komponen, ataupun kanal warna tersebut menggunakan nilai berbasis delapan bit yang memiliki nilai dengan kisaran 0 hingga 255. Sehingga, warna yang dihasilkan adalah $255 \times 255 \times 255$ atau setara dengan 16.581.375 warna.

2. Citra Grayscale

Citra grayscale merupakan citra yang intensitasnya diwakili oleh nilai keabuan tiap-tiap piksel dan memiliki nilai dengan kisaran 0 sampai 255. Nilai 0 merupakan nilai terendah dengan warna hitam, sementara nilai 255 merupakan nilai tertinggi dengan warna putih.

3. Citra Biner

Citra biner merupakan citra monokrom dan hanya memiliki 2 kemungkinan intensitas warna. Yaitu hitam (0), dan putih (1). Secara umum, citra biner

sering disebutkan dengan sebutan citra hitam putih, ataupun citra *Black and White*.

2.2 Pengolahan Citra

Pengolahan citra adalah sebuah proses pengolahan yang *inputnya* adalah citra. *Outputnya* dapat berupa citra atau sekumpulan karakteristik atau parameter yang berhubungan dengan citra. Istilah pengolahan citra digital secara umum didefinisikan sebagai pemrosesan citra dua dimensi dengan komputer. Dalam definisi yang lebih luas, pengolahan citra digital juga mencakup semua data dua dimensi. Citra digital adalah barisan bilangan nyata maupun kompleks yang diwakili oleh bit-bit tertentu. Pengolahan citra memiliki beberapa fungsi, diantaranya adalah: (Nagataries, Hardiristanto, & Purnomo, 2013)

1. Digunakan sebagai proses memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau komputer.
2. Digunakan untuk Teknik pengolahan citra dengan mentrasformasikan citra menjadi citra lain. Contoh : pemampatan citra (*image compression*) Sebagai proses awal (*preprocessing*) dari komputer visi.

2.3 Deteksi Objek

Deteksi Objek adalah suatu metode yang digunakan untuk menentukan keberadaan objek tertentu di dalam suatu citra digital. Proses deteksi dapat dilakukan dengan berbagai macam metode yang umumnya melakukan pembacaan fitur dari seluruh objek pada citra *input* (Nagataries, Hardiristanto, & Purnomo, 2013) Fitur dari objek pada citra *input* tersebut akan dibandingkan dengan fitur dari objek referensi.

2.4 Grayscale

Citra *grayscale* adalah citra yang memiliki warna yang dipakai warna hitam sebagai warna minimal (0) dan warna putih (255) sebagai warna maksimalnya, sehingga warna antaranya adalah abu-abu (Indriani, Jumaddina, & Sinaga, 2014).

Sementara, proses *grayscale* adalah suatu proses dalam mengkonversi atau merubah suatu citra kedalam warna-warna keabuan dengan hanya memperhatikan intensitas-intensitas cahaya atau warna yang dimiliki oleh tiap-tiap piksel. Proses *grayscale* mengurangi dimensi yang dimiliki oleh citra, dengan melakukan pemetaan citra tiga kanal warna menjadi hanya satu kanal warna yaitu warna keabuan (Lanora, Nguyan, & Kasampalis, 2019). Sehingga, piksel-piksel yang menyusun citra tersebut tidak lagi memiliki 3 kanal warna (merah, hijau, dan biru), melainkan hanya satu kanal warna, yaitu abu.

Tiap-tiap piksel yang menyusun citra *grayscale* dengan pola 8 bit memiliki nilai intensitas piksel yang bernilai mulai dari 0 sampai 255. Nilai 0 pada citra merepresentasikan warna hitam, sementara nilai 255 merepresentasikan warna putih. (Jayaraman, Esakirajan, & Veerakumar, 2009). Terdapat berbagai cara dalam melakukan proses *grayscale*, diantaranya adalah 2 buah rumus yang ditunjukkan pada Persamaan (1) dan (2) berikut :

$$\boxed{grayscale = 0.299R + 0.587G + 0.114B} \dots\dots\dots(1)$$

$$\boxed{grayscale = 0.333R + 0.333G + 0.333B} \dots\dots\dots(2)$$

Keterangan :

R = Intensitas piksel berwarna merah

G = Intensitas piksel berwarna hijau

B = Intensitas piksel berwarna biru

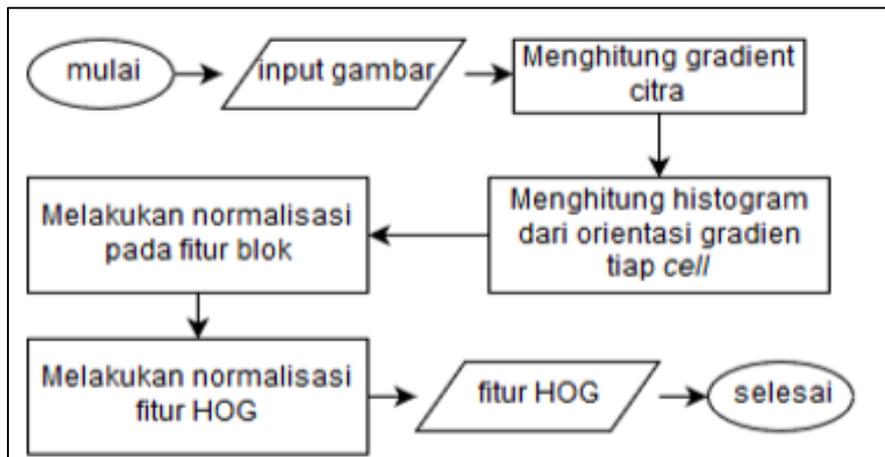
Dengan demikian, konsep itu diubah dengan mengubah 3 komponen menjadi 1 komponen matrik *greyscale* dan hasilnya adalah citra *greyscale* seperti pada **Error! Reference source not found.2.1** .



Gambar 2. 1 Citra RGB- Greyscale

2.5 *Histogram of Oriented Gradients (HOG)*

Histogram of Oriented Gradients (HOG) merupakan salah satu teknik pengambilan fitur yang bertujuan untuk mengambil informasi penting dari sebuah citra. Cara kerja metode ini, yaitu dengan mengevaluasi histogram lokal yang sudah ternormalisasi secara baik dari distribusi gradien citra dalam grid yang padat (N & B, 2005). Teknik mengekstrak fitur untuk metode ini yaitu dari distribusi lokal dari intensitas gradien tiap piksel yang terdapat pada sebuah objek citra. Dalam metode *Histogram of Oriented Gradients* pada pengenalan wajah manusia, ukuran sel berupa kumpulan atau gabungan *piksel* dan blok berupa kumpulan atau gabungan sel beserta jumlah *orientation* bin yang merupakan tempat untuk menampung hasil arah dan besar gradien akan mempengaruhi hasil keluaran fitur vektor yang dihasilkan dan juga akurasi yang didapat (S Chang & et. al, 2011). Algoritma pada proses HOG dapat dilihat pada Gambar 3.



Gambar 2. 2 Algoritma HOG

(Sumber : (Randa, Suciati, & Navastara, 2016)

Dari Gambar 2.2, tahap awal dari metode HOG adalah menghitung nilai gradien citra dihitung menggunakan Persamaan (3).

$$|G| = \sqrt{I_x^2 + I_y^2} \dots\dots\dots(3)$$

Keterangan :

I = Citra graylevel

I_x = Matrik terhadap sumbu-x

I_y = Matrik terhadap sumbu-y

I_x dan I_y dapat dihitung dengan Persamaan (4).

$$I_x = I * S_x, I_y = I * S_y \dots\dots\dots(4)$$

S_y dan S_x didapatkan dari metode untuk mencari nilai gradient yaitu dengan metode sobel :

$$S(x) = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ dan } S(y) = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Gambar 2. 3 kernel sobel

Masing-masing dihitung dengan cara konvolusi. Kemudian gradien ditransformasi ke dalam kordinat sumbu dengan sudut diantara 0° sampai 180° yang disebut orientasi gradien. Orientasi gradien (θ) dapat dihitung dengan Persamaan (5).

$$\theta = \arctan \left(\frac{lx}{ly} \right) \dots\dots\dots(5)$$

Tahap selanjutnya adalah melakukan perhitungan histogram dari orientasi gradien tiap *cell*. Setiap *piksel* dalam sebuah *cell* mempunyai nilai histogram sendiri-sendiri berdasarkan nilai yang dihasilkan dalam perhitungan gradien yang kemudian dilakukan normalisasi pada setiap blok.

Nilai normalisasi fitur blok didapat dari Persamaan (6). Fitur blok dinormalisasi untuk mengurangi efek perubahan kecerahan obyek pada satu blok.

$$b = \frac{b}{\sqrt{b^2+e}} \dots\dots\dots (6)$$

Keterangan :

b = nilai blok fitur

e = 2,71828 (konstanta)

Nilai normalisasi tiap blok digabungkan menjadi satu vektor menjadi fitur vektor HOG. Kemudian fitur vektor HOG dilakukan normalisasi. Normalisasi dilakukan melalui Persamaan (7).

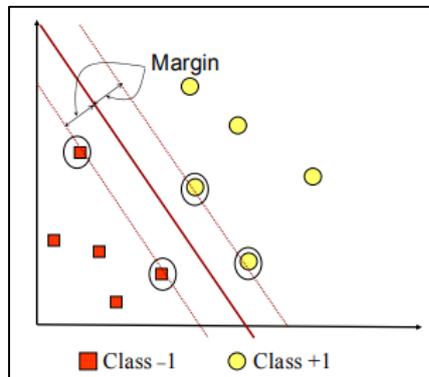
$$h = \frac{h}{\sqrt{|h|^2+e}} \dots\dots\dots(7)$$

Keterangan :

h = nilai fitur HOG

e = 2,71828 (konstanta)

2.6 Support Vector Machine (SVM)



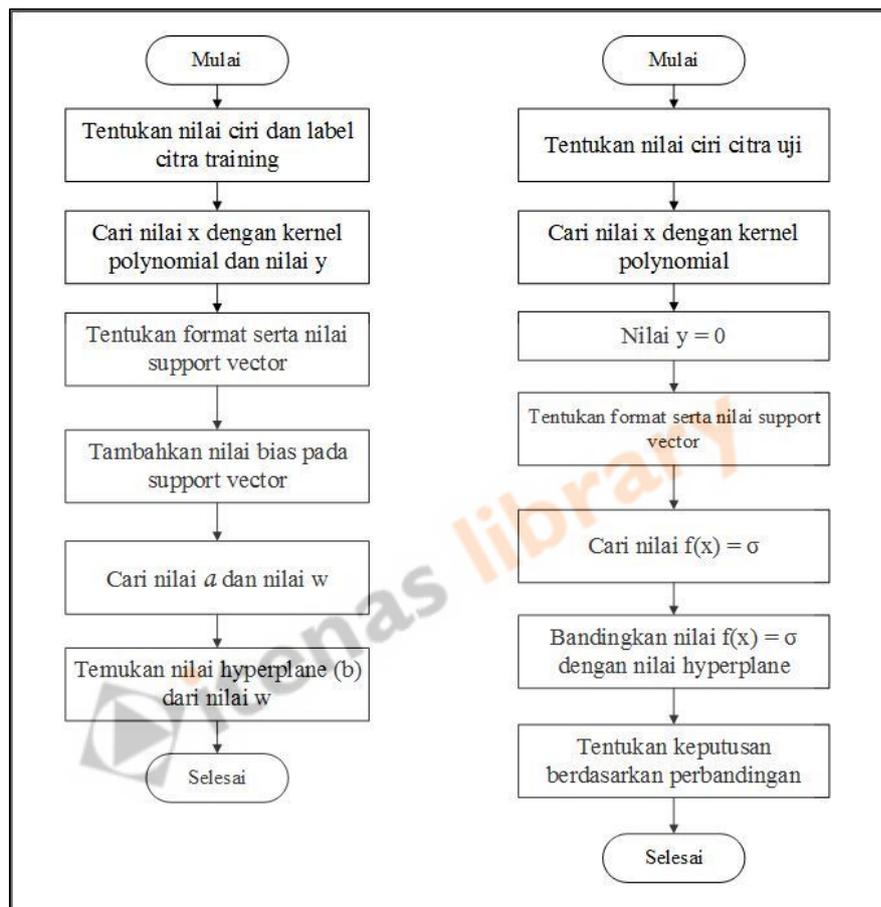
Gambar 2. 4 Hyperplane pada Support Vector Machine
(Neneng, Adi, & Isnanto, 2016)

SVM merupakan sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah ruang ciri (*feature space*) berdimensi tinggi, dilatih dengan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan bias pembelajaran yang berasal dari teori pembelajaran statistik (Neneng, Adi, & Isnanto, 2016). Maka, secara sederhana, konsep *SVM* adalah suatu cara guna mencari suatu *hyperplane* terbaik yang berfungsi untuk pemisah dua buah class pada *input space* serta sebagai jawaban dari problema klasifikasi.

Contoh pengimplementasian *hyperplane* terbaik terlihat pada Gambar 6, pada Gambar tersebut terlihat bahwa terdapat garis *hyperplane* yang terletak tepat pada tengah-tengah kedua *class*. Terdapat dua jenis *pattern* pada Gambar yaitu lingkaran kuning dan kotak merah, yang menggambarkan ciri dari tiap class. *Pattern* kotak merah dan lingkaran kuning yang ditandai dengan lingkaran hitam merupakan *support vector* dari masing-masing *class*, yaitu class -1 dan class +1. Serangkaian proses tersebut, yang diakhiri dengan proses pencarian *hyperplane* adalah inti dari proses *training* pada *Support Vector Machine*.

Pada dasarnya, *SVM* dikembangkan hanya untuk pengklasifikasian dua kelas data, namun pada perkembangannya, *SVM* dikembangkan untuk melakukan pengklasifikasian banyak kelas dengan menggunakan *Multiclass SVM*. Dalam proses klasifikasi *multiclass SVM*, *hyperplane* yang terbentuk adalah lebih dari satu.

Salah satu metode *multiclass SVM* adalah metode one vs all. Klasifikasi one vs all menghasilkan sebanyak $n-1$ *hyperplane* dari n kelas. Pada pendekatan klasifikasi one vs all, semua data dari salah satu kelas bernilai 1, sementara kelas lainnya bernilai -1. Proses tersebut berulang untuk semua kelas.



Gambar 2. 5 Flowchart SVM

(a)

(b)

(a)Flowchart proses training; (b)Flowchart proses klasifikasi

Tahapan-tahapan pada proses training *Support Vector Machine* ditunjukkan pada Gambar 2.5a. Sementara, proses klasifikasi pada *Support Vector Machine* ditunjukkan pada Gambar 2.5b. *Support Vector Machine* sebenarnya adalah suatu teknik klasifikasi yang digunakan untuk memisahkan data secara linear. Sementara untuk data yang distribusi kelasnya tidak linear, maka dibutuhkan suatu pendekatan kernel pada data latih. Kernel dapat didefinisikan sebagai fungsi yang memetakan

fitur data dari dimensi awal (rendah) ke fitur lain yang berdimensi lebih tinggi (Prasetyo, 2012). Salah satu kernel yang dapat digunakan adalah kernel *polynomial* dengan Persamaan 8 berikut.

$$K(x, y) = (x \cdot y + c)^d \dots\dots\dots(8)$$

(Prasetyo, 2012)

x dan y adalah pasangan dua data dari semua data latih, sementara c dan d adalah konstanta yang nilainya harus > 0 . Nilai tersebut kemudian membentuk suatu matriks yang berisi nilai-nilai data latih baru pada dimensi lebih tinggi. Nilai-nilai pada matriks tersebut kemudian digunakan untuk mencari x dan y pada *support vector*. Rumus untuk mencari nilai x ditunjukkan pada Persamaan 9 berikut.

$$\sum_{i=1, j=1}^n (x_i^T x_j + c)^d \quad (i, j = 1, \dots, n) \dots\dots\dots(9)$$

Sementara, Rumus untuk mencari nilai y ditunjukkan pada Persamaan 10 berikut.

$$\sum_{i=1, j=1}^n y_i^T y_j \quad (i, j = 1, \dots, n) \dots\dots\dots(10)$$

Nilai x dan y tersebut kemudian digunakan untuk mencari nilai *support vector* (φ) dengan cara memasukkan nilai tersebut ke dalam Persamaan 11 berikut

$$\varphi \begin{bmatrix} x \\ y \end{bmatrix} = \begin{cases} \begin{bmatrix} \sqrt{x_n^2 + y_n^2} - x + |x - y| \\ \sqrt{x_n^2 + y_n^2} - x + |x - y| \end{bmatrix}, & \text{if } \sqrt{x_n^2 + y_n^2} > 2 \\ \begin{bmatrix} x \\ y \end{bmatrix}, & \text{if } \sqrt{x_n^2 + y_n^2} \leq 2 \end{cases} \dots\dots\dots(11)$$

Setelah *support vector* didapatkan, nilai tersebut ditambahkan suatu nilai bias yaitu bernilai 1. Maka selanjutnya adalah mencari nilai a_i , nilai tersebut didapatkan dengan mengalikan tiap-tiap data training menggunakan Persamaan 12 berikut.

$$\sum_{i=1, j=1}^n a_i T_i^T T_j \dots\dots\dots(12)$$

$$\sum_{i=1, j=1}^n a_i T_i^T T_j = y_i \dots\dots\dots(13)$$

Setelah mendapatkan nilai a_i , langkah selanjutnya adalah mencari nilai w , dengan menggunakan Persamaan 14 berikut.

$$w = \sum_{i=1}^n a_i T_i \dots\dots\dots(14)$$

Dikarenakan pada perhitungan nilai *support vector*, diberikan suatu nilai bias, maka nilai w yang didapatkan telah mengandung nilai b sebagai *hyperplane*. Nilai w dan nilai b tersebut menjadi nilai penting dalam proses *training* maupun klasifikasi.

Sedangkan, pada proses klasifikasi *SVM*, proses tersebut berjalan dengan memanfaatkan matrik proyeksi yang dihasilkan pada proses ekstraksi fitur yang kemudian dikalikan dengan data uji (sampel pengujian) sebagai masukan. Pengklasifikasian *SVM* untuk multiclass One Against All akan membangun sejumlah k *SVM* biner (k adalah jumlah kelas). Nilai klasifikasi dihitung dengan menggunakan Persamaan 15 berikut (Ventura, 2009).

$$f(x) = \sigma \left(\sum a_i \cdot \varphi_{i-train} \cdot \varphi_{testing} \right) \dots\dots\dots(15)$$

Fungsi keputusan yang mempunyai nilai maksimal menunjukkan bahwa data x_d merupakan anggota dari kelas fungsi keputusan tersebut. (Damayanti, Arifin, & Soelaiman, 2010).

2.7 Pengujian Kinerja Sistem

Pengujian kinerja sistem dilakukan dengan menggunakan 3 metode yaitu *Precision*, *Recall*, dan *Accuracy*. Pengujian *precision* dilakukan untuk menunjukkan ketepatan antara informasi yang diminta oleh pengguna dan jawaban yang dihasilkan oleh sistem. Pengujian untuk mendapatkan nilai *Recall* digunakan untuk mengetahui tingkat keberhasilan sistem dalam menemukan suatu informasi. Sedangkan nilai *Accuracy* adalah nilai seberapa besar tingkat kedekatan antara nilai prediksi dan nilai sebenarnya (Anugrah, 2019). Nilai presisi pada pengujian *multiclass* dilakukan dengan mengambil nilai rata-rata presisi yang didapatkan dari tiap-tiap kelas.

$$\textit{precision kelas} = \frac{TP}{TP + FP} \dots\dots\dots(16)$$

$$\textit{recall kelas} = \frac{TP}{TP + FN} \dots\dots\dots(17)$$

$$\textit{accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \dots\dots\dots(18)$$

$$f1 - \textit{score} = \frac{2 * (\textit{precision} * \textit{recall})}{(\textit{precision} + \textit{recall})} \dots\dots\dots(19)$$

dimana :

TP = banyaknya data positif yang terklasifikasi dengan benar oleh sistem

TN = banyaknya data negative yang terklasifikasi dengan benar oleh sistem

FN = banyaknya data negative yang terklasifikasi salah oleh sistem

FP = banyaknya data positif yang terklasifikasi salah oleh sistem

Data positif sendiri merupakan data yang sesuai dengan kelas yang sedang diuji, sedangkan data negative merupakan data dari citra yang tidak sedang diuji.