

**NASKAH PUBLIKASI**

**IMPLEMENTASI K-NEAREST NEIGHBOR DAN EUCLIDEAN  
DISTANCE UNTUK APLIKASI PENGENALAN CITRA RAMBU LALU  
LINTAS**

Program Studi Teknik Informatika



Disusun oleh:

RUSMA EKO FIDDY RIZARTA

5140411081

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI DAN ELEKTRO  
UNIVERSITAS TEKNOLOGI YOGYAKARTA  
2019**

**NASKAH PUBLIKASI**

**IMPLEMENTASI K-NEAREST NEIGHBOR DAN EUCLIDEAN DISTANCE UNTUK  
APLIKASI PENGENALAN CITRA RAMBU LALU LINTAS**

Disusun oleh:

Rusma Eko Fiddy Rizarta  
5140411081



Pembimbing

**Donny Avianto, ST., MT.**

Tanggal : .....  
9/2/19

# IMPLEMENTASI K-NEAREST NEIGHBOR DAN EUCLIDEAN DISTANCE UNTUK APLIKASI PENGENALAN CITRA RAMBU LALU LINTAS

**Rusma Eko Fiddy Rizarta**

Program Studi Teknik Informatika, Fakultas Teknologi Informasi dan Elektro  
Universitas Teknologi Yogyakarta  
Jln. Siliwangi (Ringroad Utara) Jombor, Sleman, D.I Yogyakarta, 55285, Indonesia  
E-mail: [fiddy.root@gmail.com](mailto:fiddy.root@gmail.com)

## **Abstrak**

Rambu lalu lintas merupakan salah satu alat perlengkapan jalan dalam bentuk tertentu yang memuat lambang, huruf, angka, kalimat yang digunakan untuk memberikan perintah, larangan, peringatan dan petunjuk bagi pengguna jalan agar tertib berlalu lintas. Namun, banyak di antara pengguna jalan yang belum mengetahui arti dari setiap rambu lalu lintas yang terpasang. Pada penelitian ini, dibuatlah suatu aplikasi yang mampu melakukan klasifikasi citra rambu ke dalam 3 kelas yaitu: peringatan simpang empat prioritas, larangan parkir dan perintah memasuki jalur atau lajur yang ditunjuk. Mula-mula sistem akan melakukan pra-pemrosesan seperti seperti: grayscalling, histogram equalization, dan segmentasi pada citra input. Selanjutnya, tahap ekstraksi ciri akan dilakukan pada citra hasil pra-pemrosesan. Adapun metode ekstraksi ciri yang digunakan pada penelitian kali ini adalah ekstraksi fitur momen spasial dan pusat ternormalisasi (momen) dan ekstraksi fitur statistika warna (warna). Terakhir, nilai fitur yang dihasilkan oleh kedua metode tersebut akan diklasifikasi menggunakan K-Nearest Neighbor. Hasil uji coba menunjukkan bahwa metode ekstraksi fitur gabungan momen-warna memberikan hasil yang menjanjikan. Dari 15 citra uji yang digunakan, sistem mampu mengenali rambu dengan tepat 100% pada  $K=3$ , 86,6% pada  $K=5$ , dan 86,6% pada  $K=7$ .

**Kata Kunci** : Ekstraksi Fitur Momen-Warna, K-Nearest Neighbor, Rambu Lalu Lintas, Citra

## **1. PENDAHULUAN**

Pemerintah melalui Kementerian Perhubungan memasang berbagai rambu lalu lintas di jalan raya untuk memberikan informasi terkait kondisi jalan kepada setiap pengguna jalan yang melintas. Berdasarkan Peraturan Menteri Perhubungan Nomor PM 13 tahun 2014, yang dimaksud dengan rambu lalu lintas adalah bagian perlengkapan Jalan yang berupa lambang, huruf, angka, kalimat, dan atau perpaduan yang berfungsi sebagai peringatan, larangan, perintah, atau petunjuk bagi pengguna jalan. Rambu lalu lintas memiliki peranan penting dalam menjaga keselamatan dan ketertiban para pengguna jalan. Pemahaman terhadap arti dari setiap rambu lalu lintas merupakan suatu keharusan bagi seluruh pengguna jalan. Dengan memahami dan mematuhi rambu lalu lintas yang terpasang, diharapkan para pengguna jalan akan terhindar dari hal-hal yang merugikan seperti kecelakaan lalu lintas dan juga akan mengurangi tingkat permasalahan lalu lintas seperti kemacetan.

Namun, seperti yang kita ketahui bersama, tidak semua pengguna jalan memahami arti di balik rambu-rambu lalu lintas yang terpasang. Data yang dirilis oleh Polda Metro Jaya pada agustus tahun 2017, menunjukkan bahwa jenis pelanggaran lalu lintas yang paling banyak dilakukan oleh pengguna jalan adalah pelanggaran terhadap rambu lalu lintas dengan jumlah mencapai 15.133 kasus pelanggaran. Sebesar penyebab kasus pelanggaran tersebut bersumber dari ketidaktahuan pengguna jalan terhadap arti dari rambu lalu lintas yang ada. Dengan kata lain, jumlah tersebut dapat menjadi indikasi banyaknya pengguna jalan yang belum memahami arti rambu-rambu lalu lintas.

Di sisi lain, saat ini mulai banyak penelitian-penelitian tentang *auto-pilot*, *smart car* atau *autonomous vehicle* yang memungkinkan kendaraan dapat berjalan tanpa pengemudi manusia. Salah satu modul penting yang harus dimiliki pada *autonomous vehicle* adalah pengenalan rambu-rambu lalu lintas secara otomatis[3]. Modul ini biasanya diterapkan dengan menambahkan sensor atau kamera pada mobil yang mampu menangkap citra keadaan jalan raya termasuk didalamnya adalah rambu lalu

lintas. Citra rambu ini kemudian akan dikenali secara otomatis untuk mengatur perilaku kendaraan (kecepatan, arah melaju) atau diteruskan sebagai informasi bagi penumpang. Dengan modul ini, *autonomous vehicle* dapat melintas di jalan raya dengan aman dan selamat didasarkan pada pengamatan rambu lalu lintas.

Untuk mengurangi tingkat permasalahan berlalu lintas karena pengguna jalan kurang mengetahui jenis-jenis rambu lalu lintas yang tersebar dan melihat data dari permasalahan tentang pengetahuan rambu lalu lintas maka peneliti memiliki ide untuk membangun sebuah aplikasi pengenalan citra dari rambu lalu lintas dengan mengimplementasikan metode *K-Nearest Neighbor* dan *Euclidean Distance* untuk mengenali citra atau gambar dari rambu lalu lintas yang terpasang. Aplikasi yang dibangun diharapkan dapat memberikan pengetahuan kepada masyarakat tentang arti dari rambu lalu lintas yang memiliki bentuk bermacam-macam di jalan raya hanya dengan memasukkan citra yang terpasang di jalan raya serta penerapan teknologi *computer vision* pada kehidupan sehari-hari.

Beberapa penelitian yang menjadi referensi dalam penelitian ini diantaranya : Referensi pertama menggunakan metode (GLCM) dan *Euclidean Distance* untuk menentukan selisih kemiripan citra uji batik dan citra *training*. Hasil dari penelitian mendapatkan akurasi 77% pada *recall* 37.75% dan akurasi 82% pada *recall* 30.26% [6]. Referensi kedua yaitu penelitian dengan menerapkan metode jarak *City-Block* dan ekstraksi fitur vektor untuk pengenalan citra rambu lalu lintas berbentuk lingkaran. Hasil dari penelitian menunjukkan tingkat akurasi tertinggi yaitu 88% pada ukuran citra 50x50 piksel [7]. Referensi yang ketiga mengacu pada penelitian yang mengimplementasikan metode *K-Nearest Neighbor* dan *Gray Level Coocurrence Matrix* (GLCM). Citra diubah menjadi *grayscale* dan hasil data *grayscale discaling* menjadi 6 level dan dicari GLCM dengan derajat searah 0°, 45°, 90°, 135° dan jarak piksel 1,2,3,4,5. Ekstraksi fitur yang digunakan yaitu kontras, korelasi, energi dan homogenitas. Hasil dari penelitian ini didapat akurasi tertinggi pada sudut 45° yaitu 57,74% dan akurasi tertinggi menggunakan jarak piksel 3 yaitu 52,17% dengan jumlah data *training* dan *testing* 150 dan 60 [8]. Referensi keempat membahas tentang pendeteksi rambu lalu lintas berdasarkan bentuk dengan menggunakan operator *sobel* dan metode *hamming* untuk mencari kemiripan citra yang terdapat dalam *database*. Berdasarkan hasil uji coba menggunakan 10 citra uji yang berbeda didapat akurasi sebesar 100% dan disimpulkan operator *sobel* dan metode *hamming* dapat mendeteksi bentuk rambu lalu lintas [10].

## 2. METODOLOGI PENELITIAN

### 2.1 Metodologi Penelitian

#### a. Studi Pustaka

Studi pustaka dilakukan untuk mendapatkan teori acuan, informasi dan referensi dalam kegiatan penelitian. Sehingga dapat digunakan sebagai pengetahuan dasar dalam pengolahan citra rambu lalu lintas, algoritma metode *K-Nearest Neighbor* dan perhitungan jarak *Euclidean Distance*.

#### b. Kegiatan pengumpulan data rambu lalu lintas untuk menentukan hasil keluaran program didapat dari Peraturan Menteri Perhubungan Nomor PM 13 tahun 2014 di Dinas Perhubungan Kabupaten Sleman. Proses pengumpulan citra rambu lalu lintas dilakukan dengan mengambil foto secara langsung. Terdapat dua cara yang dilakukan dalam pengumpulan citra rambu lalu lintas yaitu :

- Citra rambu lalu lintas yang digunakan untuk data latih (*training*) atau data sampel dibuat dari cetakan kertas kemudian di foto dengan latar belakang yang berbeda.
- Citra rambu lalu lintas yang digunakan untuk pengujian aplikasi diambil langsung fotonya pada rambu yang terpasang di jalan pada kondisi siang hari. Pengambilan citra uji langsung dilapangan merupakan tujuan utama dari aplikasi bahwa mampu mengenali atau mengidentifikasi jenis rambu yang telah terpasang di jalan umum.

#### c. Perancangan Sistem

Rancangan sistem pada penelitian ini meliputi beberapa tahapan proses, yaitu rancangan terhadap alur kerja program atau *flowchart*, DAD, ERD , rancangan terhadap *preprocessing* citra, pengolahan citra/segmentasi, ekstraksi fitur citra, rancangan antarmuka data latih citra dan pengujian rambu lalu lintas menggunakan *K-Nearest Neighbor* dan perhitungan jarak *Euclidean Distance*.

d. Implementasi

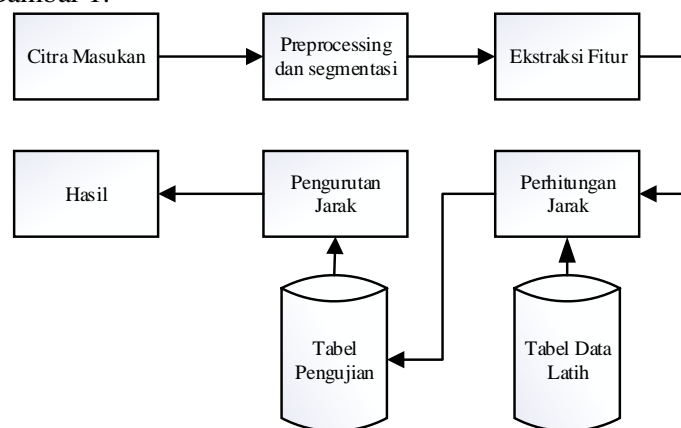
Implementasi dilakukan dengan menggunakan bahasa pemrograman Java dan basis data MySQL. Bahasa pemrograman Java dikenal sebagai *Object Oriented Programming* (OOP).

e. Pengujian

Tahapan ini dilakukan untuk mengetahui apakah hasil analisis telah sesuai dengan yang diharapkan atau belum. Pengujian juga dilakukan untuk mengetahui ada tidaknya kesalahan atau *error* dari sistem yang dibuat dengan mengamati hasil eksekusi melalui data uji dan memeriksa fungsionalitas perangkat lunak atau hasil perhitungan. Pengujian sistem menggunakan citra rambu lalu lintas sebanyak 36 buah, dengan masing-masing rambu memiliki 7 citra latih dan 5 citra uji.

## 2.2 Sistem Yang Diusulkan

Sistem yang diusulkan pada penelitian kali ini bertujuan untuk mengenali citra rambu lalu lintas dengan mengimplementasikan metode K-Nearest Neighbor dan perhitungan jarak Euclidean Distance. Rambu lalu lintas yang diproses merupakan hasil pengambilan gambar dari handphone / smartphone yang kemudian dimasukkan ke dalam aplikasi desktop. Secara umum, jalannya sistem yang diusulkan dapat dilihat pada Gambar 1.



Gambar 1 Prinsip Pengenalan Citra

Berdasarkan Gambar 1 di atas, tahapan pertama dari sistem yang diusulkan adalah pra-pemrosesan terhadap citra input. Pada tahap ini mula-mula citra input akan diubah ukurannya (*resizing*) menjadi 256x256 piksel dengan tujuan agar proses pengolahan citra lebih cepat, selanjutnya citra di konversi menjadi bentuk *grayscale*, ekualisasi histogram, penajaman citra (*sharpening*), konversi citra *biner* dan segmentasi dilasi dan erosi untuk mendapatkan nilai hasil ekstraksi fitur momen spasial dan momen pusat ternormalisasi. Sedangkan untuk ekstraksi statistika warna masih menggunakan citra asli yang telah di *resize*. Nilai fitur yang sudah diperoleh akan disimpan ke basis data. Pada pengujian citra, fitur yang diperoleh akan dibandingkan dengan semua objek data latih yang sudah disimpan dan dilakukan perhitungan jarak. Hasil jarak akan diberikan ranking atau peringkat. Seluruh peringkat akan diurutkan dari nilai terkecil ke nilai terbesar. Objek yang memiliki peringkat paling rendah/kecil merupakan citra yang mirip dan dijadikan hasil berupa informasi dari data rambu lalu lintas seperti kategori tabel rambu, nomor rambu, nama rambu, nama tipe dan arti rambu.

## 3. LANDASAN TEORI

### 3.1 K-Nearest Neighbor

*K-Nearest Neighbor* (KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap suatu objek berdasarkan data pembelajaran yang memiliki jarak paling dekat dengan objek tersebut. Data pembelajaran alam diproyeksikan ke dalam K ruang berdimensi banyak, yang masing-masing dimensi merepresentasikan fitur dari data. Ruang ini akan dibagi-bagi menjadi suatu bagian berdasarkan klasifikasi yang telah dilakukan terhadap data pembelajaran[9].

Langkah-langkah yang dilakukan untuk pada metode *K-Nearest Neighbor* adalah sebagai berikut

:

- Generate* data sampel yang akan digunakan sebagai data *training*.
- Inialisasi K titik sebagai titik pusat (*centroid*) awal.
- Hitung jarak setiap objek data set dengan data *training* dengan rumus perhitungan *Euclidean Distance*.

*Classifier* atau klasifikasi berasumsi jarak catatan dari satu dengan yang lainnya sebagai kriteria untuk kedekatan mereka dan memilih catatan paling mirip. Ada banyak metode untuk menghitung jarak seperti fungsi jarak *Euclidean*, *Manhattan*, dll.

### 3.2 Euclidean Distance

*Euclidean Space* diperkenalkan oleh Euclid, seorang matematikawan dari Yunani sekitar tahun 300 B.C.E. *Euclidean Distance* adalah jarak antara dua titik yang dapat diukur dan dihasilkan oleh formula *pythagoras* dan untuk mempelajari hubungan antara sudut dan jarak. *Euclidean vector* atau sering hanya disebut dengan *vector* adalah objek geometri yang memiliki panjang (*magnitude*) dan arah (*direction*). Sedangkan ruang vektor adalah sebuah struktur matematika yang dibentuk oleh sekumpulan vektor. Vektor-vektor tersebut dapat ditambahkan, dikalikan dengan bilangan real dan lain-lain. Gambar merupakan contoh dari ruang vektor, pada ruang vector tersebut terdapat 2 vektor yaitu vektor A dan vektor B. Untuk menghitung jarak antara vektor A dan vektor B digunakan persamaan *euclidean distance*.

Metode *Euclidean Distance* adalah suatu metode pencarian kedekatan nilai jarak dari 2 buah variabel atau lebih, selain mudah metode ini juga tidak memakan waktu dan proses yang cepat. Karena berkaitan dengan teori *Pythagoras* biasanya *Euclidean* diterapkan pada 1, 2 dan 3 dimensi. Tapi dapat diterapkan pada dimensi yang lebih tinggi. Berikut persamaan perhitungan dari *Euclidean Distance* yang digunakan dalam penelitian:

$$jarak(i) = \sqrt{(X_i - X_{uji})^2 + (Y_i - Y_{uji})^2} \quad (1)$$

Keterangan :

i = data latih ke-i

uji= data uji

X = vektor atau ekstraksi fitur X yaitu momen spasial dan momen pusat ternormalisasi

Y = vektor atau ekstraksi fitur Y yaitu statistika warna

Jarak(i) = tingkat perbedaan atau jarak

### 3.3 Preprocessing

*Preprocessing* adalah proses awal dilakukannya perbaikan citra sebelum dilakukan proses lebih lanjut terhadap citra masukan. *Preprocessing* bertujuan supaya meningkatkan kemungkinan keberhasilan pada tahap pengolahan lebih lanjut pada sebuah citra. *Preprocessing* merupakan suatu proses untuk menghilangkan bagian-bagian yang tidak diperlukan pada gambar input untuk proses selanjutnya[2]. Dalam pengolahan citra, citra berwarna seringkali harus dikonversi ke dalam bentuk citra berskala keabuan (*Grayscale*) atau citra biner (hitam-putih). Mengingat kedua citra ini yang mudah diinterpretasikan[5]. Berikut beberapa *preprocessing* yang dilakukan:

#### 3.3.1 Resize

*Resize* adalah proses mengubah resolusi atau mengubah ukuran horizontal dan vertikal citra.

#### 3.3.2 Grayscale

Untuk mengubah citra berwarna *Red* (R), *Green* (G) dan *Blue* (B) menjadi citra skala abu-abu (*Grayscale*) menggunakan persamaan sebagai berikut [5] :

$$CG = (R \times 0,2989) + (G \times 0,5870) + (B \times 0,1141) \quad (2)$$

Keterangan:

CG = Citra *Grayscale*

R = nilai piksel *red* (merah)

G = nilai piksel *green* (hijau)

B = nilai piksel *blue* (biru)

### 3.3.3 Ekualisasi Histogram (Histogram Equalization)

Tujuan dari perataan histogram adalah untuk memperoleh penyebaran aras keabuan yang merata, sehingga setiap derajat keabuan memiliki jumlah piksel yang relatif sama [1]. Akumulasi histogram untuk piksel beraras  $k$  dinyatakan pada persamaan 2.4 dengan  $i$  bernilai  $0, 1, 2, 3, \dots, L-1$  dengan  $L$  menyatakan jumlah aras keabuan.

$$c[k + 1] = \sum_{i=1}^k \text{hist}[k + 1] \quad (3)$$

Selanjutnya aras keabuan  $k$  akan diganti dengan  $a$  melalui persamaan sebagai berikut dengan  $N$  menyatakan jumlah piksel citra.

$$a_k = \text{round} (L - 1) \frac{c[k+1]}{N} \quad (4)$$

### 3.3.4 Thresholding

*Thresholding* adalah proses mengubah citra berderajat keabuan (*grayscale*) menjadi citra biner atau hitam-putih sehingga dapat diketahui daerah mana yang termasuk objek dan *background* dari citra secara jelas [4].

$$B(i) = \begin{cases} 0, & i \geq T \\ 1, & i < T \end{cases} \quad (5)$$

Keterangan :

$B(i)$  = citra biner

$i$  = Intensitas citra

$T$  = Threshold

## 3.4 Ekstraksi Fitur

Fitur suatu objek merupakan karakteristik yang melekat pada objek. Fitur bentuk merupakan suatu fitur yang diperoleh melalui bentuk objek dan dapat dinyatakan melalui kontur, area dan transformasi. Fitur bentuk biasa digunakan untuk kepentingan identifikasi objek [5].

### 3.4.1 Statistika Warna

Fitur warna dapat diperoleh dari perhitungan statistis seperti rerata dan deviasi standar. Perhitungan dikenakan pada setiap komponen R, G dan B.

Rerata memberikan ukuran mengenai distribusi dan dihitung dengan :

$$\mu = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N P_{ij} \quad (6)$$

Keterangan:

$\mu$  = rata-rata

$N$  = Baris piksel

$M$  = kolom piksel

$P$  = intensitas komponen RGB

$MN$  = jumlah piksel

Varians menyatakan luas sebaran distribusi. Akar kuadrat varians dinamakan sebagai standar deviasi. Standar deviasi terdapat pada persamaan 7.

$$\sigma = \sqrt{\frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (P_{ij} - \mu)^2} \quad (7)$$

Keterangan:

$\sigma$  = varians

$\mu$  = rata-rata

$N$  = Baris piksel

$M$  = kolom piksel

$P$  = intensitas komponen RGB

$MN$  = jumlah piksel

### 3.4.2 Momen Spasial dan Momen Pusat Ternormalisasi

Persamaan momen spasial yaitu :

$$M_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^p y^q i(x, y) \quad (8)$$

Keterangan :

$p, q = 0, 1, 2, \dots$  dengan  $p, q$  menyatakan orde momen adalah *integer*  
 $M$  menyatakan jumlah kolom citra  
 $N$  menyatakan jumlah baris citra  
 $x$  adalah ordinat piksel  
 $y$  adalah absis piksel  
 $i(x, y)$  menyatakan intensitas aras keabuan piksel  $(x, y)$

Momen pusat adalah momen spasial yang dihitung relatif terhadap pusat massa. Persamaan momen pusat yaitu :

$$\mu_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (x - \bar{x})^p (y - \bar{y})^q i(x, y) \quad (9)$$

Momen diatas bersifat *invariant* (tidak terpengaruhi) terhadap translasi. Dalam hal ini,  $\bar{x}$  dan  $\bar{y}$  diperoleh melalui :

$$\bar{x} = \frac{M_{10}}{M_{00}}, \bar{y} = \frac{M_{01}}{M_{00}} \quad (10)$$

Agar momen pusat bersifat bebas terhadap translasi, penyekalan, dan rotasi, maka momen perlu dinormalisasi. Persamaan momen ternormalisasi yaitu :

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\frac{p+q+2}{2}}}, \gamma = \frac{p+q+2}{2} \quad (11)$$

### 3.5 Perhitungan Nilai Akurasi

Persamaan untuk mencari tingkat akurasi hasil pengenalan citra uji yaitu :

$$Akurasi = \frac{Jumlah\ Citra\ Dikenali}{Jumlah\ Citra\ Uji} \times 100 \quad (12)$$

## 4. HASIL DAN PEMBAHASAN

### 4.1 Hasil

Aplikasi yang dibangun digunakan untuk mengenali citra rambu lalu lintas dengan mengimplementasikan metode *K-Nearest Neighbor* dan perhitungan jarak *Euclidean Distance*. Rambu lalu lintas yang diproses merupakan hasil pengambilan gambar dari handphone / *smartphone* yang kemudian dimasukkan ke dalam aplikasi desktop.

#### 4.1.1 Kebutuhan Sistem

Berikut beberapa proses yang dilakukan untuk memenuhi kebutuhan pengguna aplikasi yaitu :

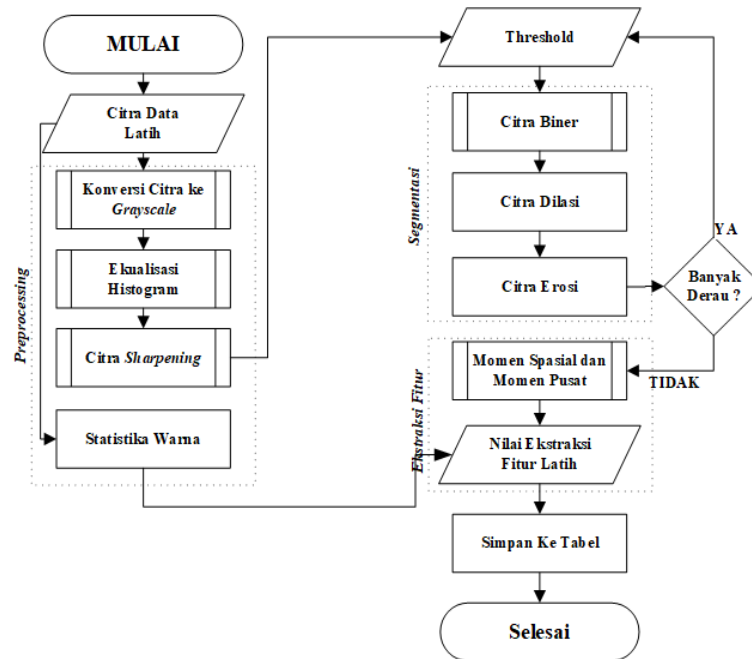
- Menyimpan data rambu lalu lintas yang memuat informasi arti rambu lalu lintas, nomor rambu, tabel rambu, nama tipe dan gambar atau simbol rambu lalu lintas.
- Melakukan pengolahan citra dari data latih atau data uji yang dimasukkan.
- Dapat mengatur *threshold* jika citra hasil atau citra biner yang akan di ekstraksi kurang maksimal (mengandung derau).
- Melakukan atau menghitung ekstraksi fitur dari data latih atau data uji.
- Menyimpan hasil ekstraksi fitur ke basis data.
- Menentukan nilai  $K$  yaitu 3, 5 dan 7 saat pengujian citra.
- Menghitung jarak data uji dengan data latih menggunakan *Euclidean Distance* dan melakukan pengurutan jarak nilai terkecil ke terbesar untuk mendapatkan peringkat.
- Menentukan keputusan klasifikasi dan memberikan informasi hasil rambu yang dapat dikenali sistem.
- Menambahkan petunjuk atau bantuan kepada pengguna bagaimana citra hasil biner yang sesuai atau citra yang seharusnya memperlihatkan objek rambu

#### 4.1.2 Flowchart

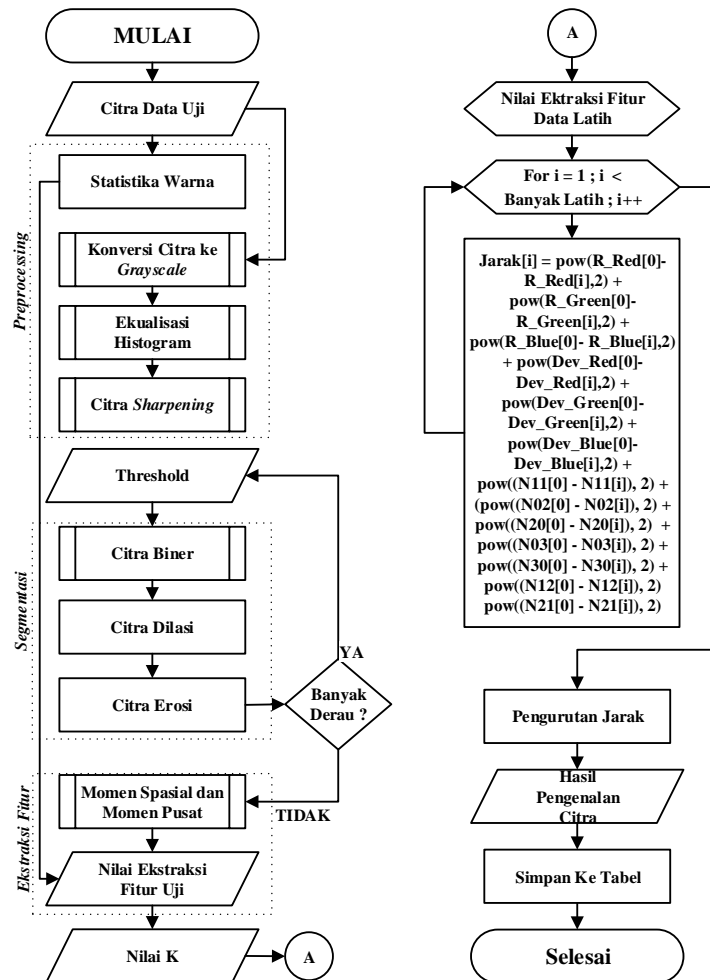
Pada aplikasi pengenalan citra rambu lalu lintas ini untuk proses pengolahan masukkan citra berupa gambar yang akan dilakukan dengan cara-cara sesuai teori yang telah dipelajari, sedangkan



untuk proses pengenalan akan diberikan nilai K dan perhitungan *Euclidean Distance* yang akan disajikan pada *flowchart*.



Gambar 2 *Flowchart* Data Latih



Gambar 3 *Flowchart* Data Uji

## 4.2 Pembahasan

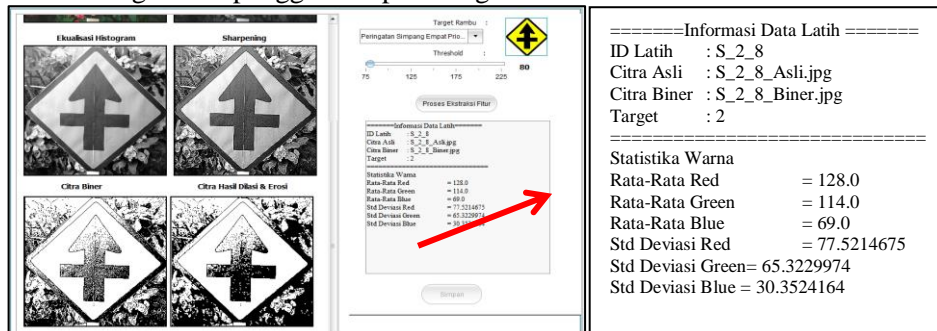
Aplikasi bertujuan untuk mengenali salah satu rambu lalu lintas dari tiga rambu lalu lintas yang digunakan yaitu peringatan simpang empat prioritas, larangan parkir, dan perintah memasuki jalur atau lajur yang ditunjuk yang terpasang di jalan. Proses pengenalan citra diawali dengan memproses semua data citra data latih yang masing-masing jenis rambu memiliki tujuh buah citra data latih yang telah dimasukkan ke dalam sistem, sehingga total citra latih yaitu 21. Citra-citra yang dimasukkan tersebut diolah dengan menggunakan metode pengolahan citra yang terdiri dari konversi ke bentuk *grayscale*, ekualisasi histogram, konversi ke bentuk *biner*, proses dilasi dan erosi. Hasil akhir dari pemrosesan gambar, selanjutnya digunakan untuk mendapatkan nilai ekstraksi fitur statistika warna, momen pusat dan momen spasial dan disimpan ke dalam basis data.

### 4.2.1 Implementasi

Implementasi pada aplikasi yaitu memasukkan data rambu lalu lintas sebagai informasi hasil pengenalan citra, implementasi data latih, implementasi pengujian citra dan melihat hasil akurasi dari pengenalan citra uji. Selain itu implementasi tampilan dibuat cukup sederhana yang bertujuan pengguna aplikasi tidak kesulitan ketika akan melakukan pengujian citra rambu lalu lintas.

#### a. Tampilan Antar Muka Data Latih

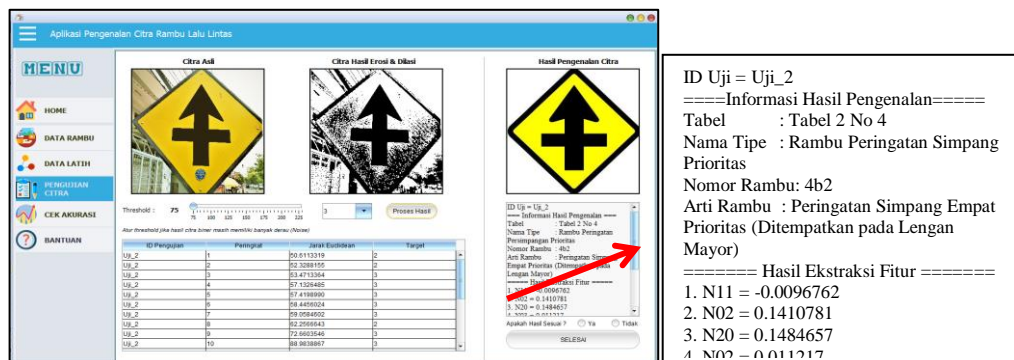
Tampilan data latih merupakan menu untuk memasukkan citra latih hingga proses akhir yaitu nilai ekstraksi fitur. Selain itu, data ekstraksi fitur citra latih akan digunakan untuk perhitungan jarak *Euclidean Distance* dengan citra uji. Pengguna memasukkan citra melalui kotak dialog buka gambar, kemudian menentukan target rambu lalu lintas, proses ekstraksi fitur dan simpan nilai. Saat pengguna memasukkan citra dengan klik *icon* buka gambar, nilai *threshold* akan otomatis terisi berdasarkan perhitungan statistika, namun jika hasil citra biner dirasa kurang sesuai pengguna dapat mengatur *threshold* sendiri.



Gambar 4 Tampilan Antar Muka Data Latih

#### b. Tampilan Antar Muka Pengujian Citra

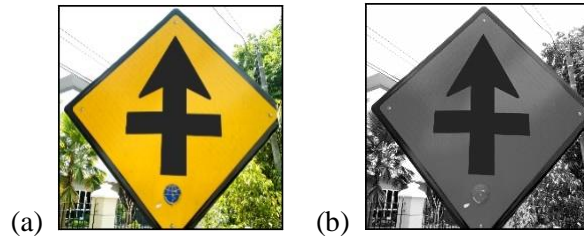
Tampilan menu pengujian citra pada gambar 6 merupakan menu yang digunakan oleh pengguna untuk melakukan pengenalan citra uji. Pengguna diminta untuk memasukkan citra uji melalui *icon* buka gambar. Citra hasil pengolahan yaitu citra *biner* akan otomatis tampil di sebelah kanan *icon* buka gambar, jika hasil kurang sesuai pengguna dapat mengatur *threshold* sendiri. Secara garis besar, proses pengolahan citra pada menu pengujian sama seperti pada data latih.



Gambar 5 Tampilan Antar Muka Pengujian Citra

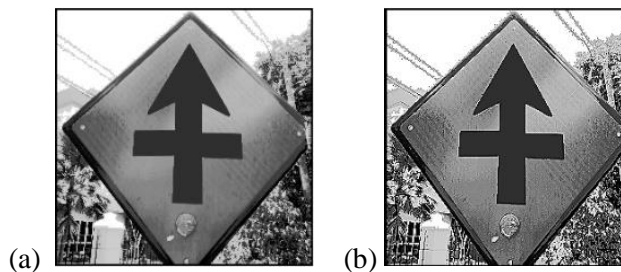
#### 4.2.2 Pembahasan Flowchart

Pembahasan ini meliputi pengolahan citra dan mendapatkan nilai-nilai ekstraksi fitur pada aplikasi. Berikut merupakan tahapan awal yaitu proses pengolahan citra berdasarkan *flowchart*, mula mula masukkan citra asli ke dalam aplikasi, kemudian akan dilakukan proses *resizing* atau mengubah ukuran menjadi 256x256 piksel untuk selanjutnya di konversi ke citra *grayscale* yang ditunjukkan pada gambar 6. Citra asli pada tahap awal akan dilakukan ekstraksi fitur statistika warna, karena setiap pengolahan citra disimpan pada variabel yang berbeda maka citra lain nilainya tidak akan terpengaruh.



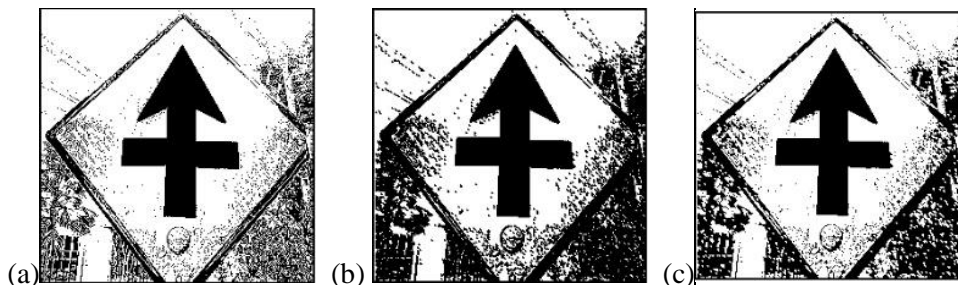
**Gambar 6** Citra Asli (a) dan Citra Hasil Grayscale

Selanjutnya, dari citra *grayscale* akan diproses ekualisasi histogram untuk mendapatkan sebaran intensitas yang merata. Dari citra ekualisasi histogram akan dilakukan proses penajaman citra (*sharpening*) yang bertujuan untuk mempertajam setiap tepi dari objek, perbedaan hasil pada penajaman citra terlihat pada tepian tanda panah atau simbol rambu lalu lintas terdapat garis putih. Hasil dari proses tersebut ditunjukkan pada gambar 7.



**Gambar 7** Citra Hasil Ekualisasi Histogram (a) dan Hasil *Sharpening* (b)

Karena ekstraksi fitur yang digunakan adalah momen spasial dan momen pusat maka perlu dilakukan segmentasi citra, dari citra hasil *sharpening* akan di proses menjadi citra biner. Selanjutnya citra biner akan dilakukan proses dilasi dan erosi untuk mengurangi derau (*noise*) yang terdapat pada citra. Citra hasil erosi merupakan citra yang akan diambil ekstraksi fiturnya. Berikut hasil dari pengolahan ditunjukkan pada gambar 8.



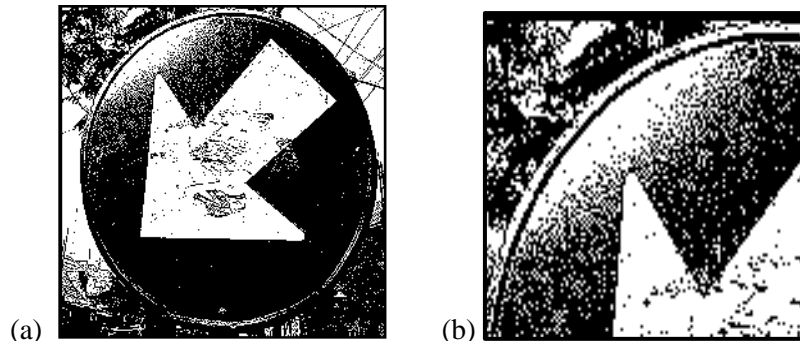
**Gambar 8** Citra Biner (a), Citra Hasil Dilasi (b) dan Citra Hasil Erosi (c)

Jika semua pengolahan citra dan ekstraksi fitur pada data latih selesai, maka program akan menyimpan nilai-nilai ke basis data. Untuk proses pengujian citra, hasil nilai ekstraksi fitur statistika warna, momen spasial dan pusat dari citra uji akan dihitung jaraknya dengan ekstraksi fitur data latih yang telah tersimpan ke basis data menggunakan persamaan 1 *euclidean distance*.

### 4.2.3 Pengujian

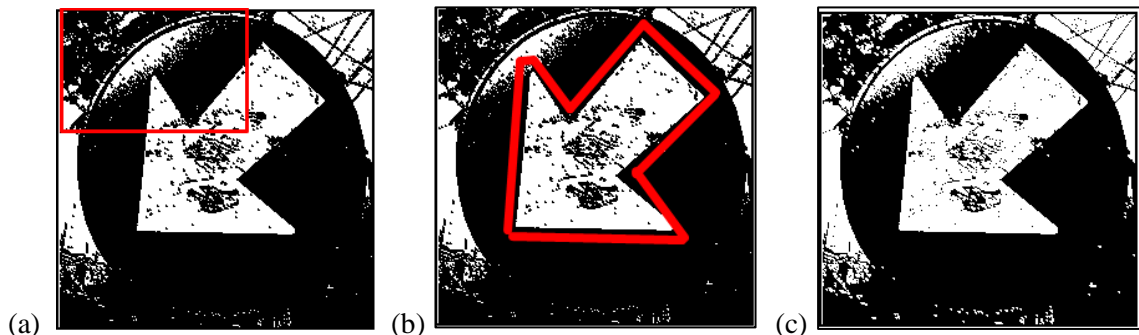
#### 4.2.3.1 Pengujian Segmentasi Citra

Pengujian segmentasi dilakukan untuk mendapatkan hasil citra biner yang baik dalam memisahkan objek citra dari latar belakang yang dapat mengganggu proses ekstraksi fitur. Karena proses konversi citra *grayscale* ke citra *biner* saja kurang cukup, maka peneliti menambahkan proses dilasi dan erosi pada citra. *Threshold* awal yang diberikan bernilai 160, berikut hasil citra *biner* pada gambar 9 (a) dan terlihat pada pojok atas citra terdapat derau (*noise*) yang mengganggu objek citra gambar 9 (b).



Gambar 9 Citra Biner (a) dan Derau Pada Citra Biner (b)

Karena hasil konversi citra *biner* dengan *threshold* 160 masih terdapat derau, maka dilakukan proses dilasi untuk memperlebar aras citra bernilai 1. Berikut hasil yang didapat ditunjukkan pada gambar 10 (a) yang mana bercak atau derau warna putih yang lebar tadi sudah tertutup oleh warna hitam. Kemudian citra hasil dilasi juga masih terdapat warna hitam yang menutupi objek citra gambar 10 (b) atau warna hitam yang masih lebar penyebarannya, maka selanjutnya diberikan proses penipisan (erosi) citra berwarna hitam sehingga bercak derau menjadi tipis penyebarannya di dalam objek dengan hasil pada gambar 10 (c).



Gambar 10 Perbandingan Hasil Citra Dilasi (a), Derau pada objek (b) Dan Hasil Erosi (c)

#### 4.2.3.2 Pengujian Ekstraksi Fitur Statistika Warna, Momen Spasial-Pusat

Tahapan ini bertujuan untuk melihat metode pengaruh metode ekstraksi fitur yang digunakan terhadap kemampuan sistem dalam mengenali objek rambu lalu lintas. Terdapat 2 metode ekstraksi fitur yang terlibat pada tahap ini, yaitu ekstraksi fitur statistika warna (SW) dan ekstraksi fitur momen spasial-pusat (MSP). Kedua metode tersebut diterapkan pada seluruh citra uji. Citra uji yang digunakan berjumlah 15 citra, terdiri dari 5 citra rambu peringatan simpang empat prioritas, 5 citra rambu larangan parkir, dan 5 citra rambu perintah memasuki jalur yang ditunjuk. Setelah melakukan ekstraksi fitur terhadap citra uji, sistem akan melanjutkan ke tahap klasifikasi menggunakan metode KNN. Pada penelitian ini, digunakan 3 nilai K yaitu K=3, K=5, dan K=7 dalam rangka mencari nilai akurasi yang tertinggi. Hasil dari tahap pengujian ini dapat terangkum pada Tabel 1-Tabel 2.

**Tabel 1** Presentase Pengenalan Citra

Citra Uji ke-	Threshold	Metode Ekstraksi Fitur								
		MSP			SW			MSP + SW		
		K=3	K=5	K=7	K=3	K=5	K=7	K=3	K=5	K=7
1	100	x	x	✓	✓	✓	✓	✓	✓	✓
2	75	✓	✓	✓	✓	x	x	✓	x	x
3	75	x	x	✓	✓	✓	✓	✓	✓	✓
4	75	x	x	✓	✓	x	x	✓	x	x
5	80	x	x	✓	✓	✓	✓	✓	✓	✓
6	170	✓	✓	✓	✓	✓	✓	✓	✓	✓
7	170	✓	x	x	✓	✓	✓	✓	✓	✓
8	170	✓	x	x	✓	✓	✓	✓	✓	✓
9	170	✓	x	x	✓	✓	✓	✓	✓	✓
10	160	✓	✓	✓	✓	✓	✓	✓	✓	✓
11	120	x	x	x	✓	✓	✓	✓	✓	✓
12	160	x	x	x	✓	✓	✓	✓	✓	✓
13	148	✓	x	✓	✓	✓	✓	✓	✓	✓
14	160	x	✓	x	✓	✓	✓	✓	✓	✓
15	130	x	✓	x	✓	✓	✓	✓	✓	✓
Rata-rata akurasi		46,7%	33,3%	53,3%	100%	86,6%	86,6%	100%	86,6%	86,6%

Peneliti melakukan pengujian lain menggunakan nilai *threshold* yang sama untuk semua citra uji, nilai *threshold* yang digunakan yaitu 133 didapat dari rata-rata hasil dari pengenalan pada tabel 1.

**Tabel 2** Presentase Pengenalan Threshold 133

Citra Uji ke-	Threshold	Metode Ekstraksi Fitur								
		MSP			SW			MSP + SW		
		K=3	K=5	K=7	K=3	K=5	K=7	K=3	K=5	K=7
1	133	✓	✓	✓	✓	✓	✓	✓	✓	✓
2	133	x	x	x	✓	x	x	✓	x	x
3	133	✓	✓	✓	✓	✓	✓	✓	✓	✓
4	133	x	x	x	✓	x	x	✓	x	x
5	133	x	x	✓	✓	✓	✓	✓	✓	✓
6	133	x	x	x	✓	✓	✓	✓	✓	✓
7	133	x	x	x	✓	✓	✓	✓	✓	✓
8	133	x	x	x	✓	✓	✓	✓	✓	✓
9	133	x	x	x	✓	✓	✓	✓	✓	✓
10	133	x	x	x	✓	✓	✓	✓	✓	✓
11	133	x	x	x	✓	✓	✓	✓	✓	✓
12	133	✓	x	x	✓	✓	✓	✓	✓	✓
13	133	✓	x	✓	✓	✓	✓	✓	✓	✓
14	133	x	x	x	✓	✓	✓	✓	✓	✓
15	133	x	x	x	✓	✓	✓	✓	✓	✓
Rata-rata akurasi		26,7%	13,3%	26,7%	100%	86,6%	86,6%	100%	86,6%	86,6%

Karena nilai *threshold* berpengaruh ke konversi citra biner, maka dari hasil pada tabel 2 dapat disimpulkan bahwa dengan statistika warna dapat digunakan untuk mengklasifikasi citra rambu lalu lintas berdasarkan warna untuk mencari rambu peringatan, larangan atau perintah. Setelah itu, momen spasial dan momen pusat digunakan untuk pencarian bentuk atau objek dari citra yang dimasukkan ke aplikasi. Terlihat pada tabel 2 jika dibandingkan dengan tabel 1, hasil pengenalan lebih bagus pada tabel 1 karena pada pengujiannya menggunakan *threshold* yang menyesuaikan seleksi objek pada rambu lalu lintas.

#### 4.2.3.3 Pengujian Metode K-Nearest Neighbor dan Euclidean Distance

Pengujian metode *K-Nearest Neighbor* merupakan simulasi perhitungan dari ekstraksi fitur data latih dengan data uji. Pengujian menggunakan citra data latih sebanyak 21 buah dan satu citra uji. Pada pengujian simulasi perhitungan ini data uji menggunakan *threshold* 75 menghasilkan ekstraksi fitur pada tabel 3 dan citra hasil biner pada gambar 11.



Langkah-langkah pengujian citra dengan metode *K-Nearest Neighbor* dilakukan dengan urutan sebagai berikut :

1. Masukkan Citra

Masukkan citra uji dan berikan *threshold* bernilai 100 sehingga menghasilkan citra *biner* pada gambar 11 (kanan).



**Gambar 11** Citra Asli (kiri) dan Citra *Biner* (kanan)

2. Tentukan Nilai K

Pada simulasi perhitungan ini peneliti menggunakan nilai  $K = 5$ .

3. Hitung Ekstraksi Fitur

Tekan tombol proses maka akan menghitung ekstraksi fitur. Nilai ekstraksi fitur citra uji yang dihitung tertera pada tabel 3.

**Tabel 3** Ekstraksi Data Uji

R_Red	R_Green	R_Blue	Dev_R	Dev_G	Dev_B
130.0	112.0	36.0	89.9966617	77.3127549	34.3286388
N11	N02	N20	N03	N30	N12
0.0151709	0.1338387	0.1244211	0.0020316	0.0077015	-0.0028548
N21					
-0.0005377					

Keterangan :

- **N11, N02, N20, N03, N30, N12** dan **N21** merupakan kolom ekstraksi fitur momen spasial dan momen pusat ternormalisasi.
- **R\_Red, R\_Green, R\_Blue** merupakan rata-rata dari sebaran intensitas RGB setiap piksel dihitung dari persamaan 2.8 statistika warna.
- **Dev\_R, Dev\_G, Dev\_B** merupakan standar deviasi RGB yang dihitung dari nilai rerata dari persamaan 2,9 statistika warna.

4. Hitung Nilai Jarak Euclidean

Menghitung hasil ekstraksi fitur citra uji dengan ekstraksi data latih menggunakan persamaan 1 dan simpan nilai ke basis data namun kolom peringkat masih kosong. Hasil perhitungan dapat dilihat pada tabel 4.

**Tabel 4** Jarak *Euclidean*

ID_det	ID_Uji	Euclidean	Target	Peringkat
1	Uji_1	116,6455231	1	
2	Uji_1	124,2898102	1	
3	Uji_1	138,5784454	1	
4	Uji_1	114,6457596	1	
5	Uji_1	137,7447815	1	
6	Uji_1	100,3209915	1	

7	Uji_1	37,5792084	2	
8	Uji_1	47,2186432	2	
9	Uji_1	71,6607513	2	
10	Uji_1	29,4930973	2	
11	Uji_1	43,4338989	2	
12	Uji_1	29,0763969	2	
13	Uji_1	113,4697952	3	
14	Uji_1	100,1833878	3	
15	Uji_1	99,8655777	3	
16	Uji_1	113,9651184	3	
17	Uji_1	82,9838715	3	
18	Uji_1	105,4908524	3	
19	Uji_1	53,1637726	2	
20	Uji_1	99,5513611	3	
21	Uji_1	135,3760223	1	

5. Urutkan Nilai Perhitungan Jarak

Berdasarkan jarak *euclidean* yang telah dihitung, kolom nilai *euclidean* terkecil akan diurutkan dan urutan peringkat diberikan seperti pada tabel 4.

**Tabel 5** Urutan Jarak Euclidean

ID_detail_Uji	ID_Uji	Euclidean	Target	Peringkat
12	Uji_1	29,0763969	2	1
10	Uji_1	29,4930973	2	2
7	Uji_1	37,5792084	2	3
11	Uji_1	43,4338989	2	4
8	Uji_1	47,2186432	2	5
19	Uji_1	53,1637726	2	6
9	Uji_1	71,6607513	2	7
17	Uji_1	82,9838715	3	8
20	Uji_1	99,5513611	3	9
15	Uji_1	99,8655777	3	10
14	Uji_1	100,1833878	3	11
6	Uji_1	100,3209915	1	12
18	Uji_1	105,4908524	3	13
13	Uji_1	113,4697952	3	14
16	Uji_1	113,9651184	3	15
4	Uji_1	114,6457596	1	16
1	Uji_1	116,6455231	1	17
2	Uji_1	124,2898102	1	18
21	Uji_1	135,3760223	1	19
5	Uji_1	137,7447815	1	20
3	Uji_1	138,5784454	1	21

6. Ambil Data Klasifikasi Berdasarkan Nilai K

Dengan menggunakan nilai  $K = 5$ , maka didapatkan klasifikasi anggota nilai  $K$  yang dapat dilihat pada tabel 5.

**Tabel 6** Klasifikasi Nilai K

ID_detail_Uji	Euclidean	Target	Peringkat
12	29.0763969	2	1
10	29.4930973	2	2
7	37.5792084	2	3
11	43.4338989	2	4
8	47.2186432	2	5

7. Kesimpulan Target Terbanyak Berdasarkan Klasifikasi Nilai K

Tabel 5 menunjukkan bahwa didapat target terbanyak yaitu nilai 2 sebanyak 5 pada klasifikasi nilai  $K=5$ , maka selanjutnya program akan mengambil rambu lalu lintas yang memiliki target atau  $ID\_R = 2$  (gambar 12) . Karena proses pengujian antara citra uji dengan citra hasil pengenalan luarannya sama, pengujian ini dapat dinyatakan sebagai “**Dikenali**” atau dengan mengkonfirmasi *radiogrup* “YA”.



**Gambar 12** Rambu Peringatan Simpang Empat Prioritas

## 5. PENUTUP

Berdasarkan hasil penelitian tentang pengenalan citra rambu lalu lintas menggunakan metode *K-Nearest Neighbor* dan *Euclidean Distance* dapat diambil kesimpulan yaitu aplikasi dapat menampilkan hasil pengenalan citra salah satu dari tiga jenis rambu lalu lintas yang dipakai lengkap dengan informasi tentang rambu lalu lintas seperti arti rambu, tipe rambu dan lain sebagainya. Faktor yang dapat mempengaruhi pengenalan citra yaitu adalah intensitas cahaya, komposisi warna, kontras, debu, bayangan objek lain seperti daun dan posisi saat gambar di foto. Nilai *threshold* menjadi faktor yang mempengaruhi hasil konversi biner yang digunakan untuk ekstraksi fitur momen spasial dan momen pusat ternormalisasi. Kemudian berdasarkan jumlah data citra latih sebanyak 21 citra dan data citra uji sebanyak 15 menggunakan ekstraksi fitur statistika warna, momen spasial dan momen pusat ternormalisasi dengan pengujian nilai  $K = 3, 5$  dan  $7$ , didapat akurasi tertinggi pada nilai  $K=3$  yaitu sebesar 100%, 86,6% untuk nilai  $K=5$ , 86,6% untuk nilai  $K=7$  dan 91,1% untuk total keseluruhan citra uji sebanyak 15 citra.



## UCAPAN TERIMA KASIH

Terima kasih saya ucapkan kepada pembimbing penulis dalam penelitian ini dan kepada dosen program studi Informatika Universitas Teknologi Yogyakarta yang sudah membantu dalam proses penelitian dan perkuliahan penulis. Terima kasih kepada bapak ibu segenap keluarga dan teman-teman yang sudah memberikan berbagai bentuk semangat selama penulis melakukan penelitian ini.

## DAFTAR PUSTAKA

- [1] Munir, R. (2004), *Pengolahan Citra Digital dengan Pendekatan Algoritmik*, Cetakan Pe Bandung: Informatika Bandung.
- [2] Bahri, R.. dan Maliki, I. (2012), *Perbandingan Algoritma Template Matching dan Feature Extraction pada Optical Character Recognition*, *Jurnal Komputer dan Informatika*, 1, 29–35.
- [3] Wirjaputra, A. (2012), *Mengungkap Teknologi “Google Autonomous Car,”* Diambil dari <http://comp-eng.binus.ac.id/files/2012/06/Mengungkap-Teknologi-Google-Autonomous-Car-Andrew-W.pdf> (akses 10 Desember 2018).
- [4] Dewi, M.. Kesiman, W.. dan Sunarya, I.M.. (2014), *Aplikasi Pembelajaran Pengenal Aksara Bali*, *Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI)*, 41–50.
- [5] Kadir, A. dan Susanto, A. (2013), *Teori dan Aplikasi Pengolahan Citra*, D. Hardjono, Ed. Andi Offset.
- [6] Karimah, F.U. (2014), *Rancang Bangun Aplikasi Pencarian Citra Batik Besurek Berbasis Tekstur Dengan Metode Gray Level Co-Occurence Matrix dan Euclidean Distance*, Universitas Bengkulu.
- [7] Romadhon, G. dan Murinto (2014), *Aplikasi Pengenalan Citra Rambu Lalu Lintas Berbentuk Lingkaran Menggunakan Metode Jarak City-Block*, *Jurnal Sarjana Teknik Informatika*, 2(2).
- [8] Listyanto, S.R. (2015), *Implementasi K-Nearest Neighbor Untuk Mengenali Pola Citra Dalam Mendeteksi Penyakit Kulit*, Universitas Dian Nuswantoro Semarang.
- [9] Rosyidi, A. Ginardi, H. dan Munif, A. (2017), *Implementasi Metode K-Nearest Neighbor Untuk Penentuan Lokasi Pos Hujan Terdekat Dengan Titik Rute Perjalanan Pada Aplikasi Clearroute*, *Jurnal Teknik ITS*, 6(2).
- [10] Swedia, E.R. Septian, M.R.D. dan Cahyanti, M. (2017), *Aplikasi Pendeteksi Rambu Lalu-Lintas Menggunakan Operator Sobel dan Metode Hamming*, Universitas Gunadarma.