Pendeteksian Rambu Lalu Lintas beserta Kontennya dengan *Image Processing*

Azmi Muhammad Syazwana - 13519151 (*Author*)

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jalan Ganesha 10 Bandung

E-mail (gmail): azmisyazwana@gmail.com

***Abstract*— Pendeteksi rambu lalu lintas dengan *image processing* merupakan salah satu teknologi yang dapat digunakan untuk membantu pengendalian kendaraan dengan menggunakan sistem kemudi otomatis. Dengan teknologi ini, sistem kemudi otomatis dapat mengambil gambar lingkungan sekitar kendaraan menggunakan kamera, kemudian mengolah gambar tersebut dengan algoritma *image processing* untuk mengenali rambu lalu lintas yang terdapat di jalan. Teknologi ini memungkinkan sistem kemudi otomatis untuk mendeteksi dan mengenali rambu lalu lintas yang terdapat di jalan, sehingga dapat membantu pengendalian kendaraan dengan lebih aman dan efisien. Namun, teknologi ini juga memiliki beberapa keterbatasan, seperti ketergantungan terhadap kualitas gambar yang tersedia dan kemampuan algoritma *image processing* untuk mengenali rambu lalu lintas dengan tepat**

Keywords—image processing, rambu lalu lintas, deteksi

I. PENDAHULUAN

Perkembangan teknologi yang sangat pesat saat ini telah memungkinkan setiap orang untuk medapatkan informasi dengan cepat. Perkembangan teknologi juga memungkinkan banyak orang untuk menghasilkan begitu banyak media visual yang tersedia secara luas. Begitu banyak gambar dan foto yang tersedia di internet untuk berbagai keperluan dan untuk menyiapkan gambar-gambar tersebut agar dapat memenuhi kebutuhan, gambar-gambar tersebut perlu diproses terlebih dahulu.

Di era digital ini, teknologi telah berkembang sangat pesat. Dengan kemajuan teknologi ini, kehidupan manusia menjadi terasa lebih mudah karena teknologi selalu membantu seluruh kehidupan manusia. Misalnya, kemajuan teknologi ini mampu memberikan informasi dengan cepat kepada setiap orang, sistem dan teknologi komputasi dapat membantu mengerjakan dan menyelesaikan tugas manusia. Salah satu contoh signifikan tugas manusia yang mampu dikerjakan oleh teknologi yaitu, adanya sistem kemudi otomatis pada kendaraan.

Sistem kemudi otomatis merupakan teknologi yang memungkinkan kendaraan untuk secara otomatis mengendalikan kemudi, akselerasi, dan rem. Sistem ini menggunakan sensor, kamera, radar, dan GPS untuk memantau lingkungan sekitar kendaraan dan mengambil tindakan yang diperlukan sesuai dengan informasi yang diterimanya. Keuntungan dari sistem kemudi otomatis adalah membantu pengemudi dalam mengendalikan kendaraan dengan lebih mudah dan aman, terutama dalam situasi yang membutuhkan reaksi cepat seperti pada saat terjadi kecelakaan atau keadaan jalan yang tidak memungkinkan. Sistem ini juga dapat membantu kelompok yang tidak mampu berkendara seperti lansia atau orang dengan kebutuhan khusus.

Salah satu fitur krusial pada sistem kemudi otomatis yaitu, pengenalan dan pendeteksian lingkungan sekitar kendaraan dengan menggunakan kamera. Rambu lalu lintas merupakan salah satu peringatan dan pemberitahuan mengenai suatu hal untuk menjaga keamanan berkendara. Oleh karena itu, pendeteksian rambu lalu lintas akan sangat berguna pada sistem kendali otomatis pada kendaraan karena dapat membantu pengendalian kendaraan. Untuk mendeteksi rambu lalu lintas ini dapat dilakukan dengan metode *image processing*.

Teknologi pendeteksi rambu lalu lintas dengan *image processing* merupakan salah satu teknologi yang dapat digunakan dalam sistem kemudi otomatis untuk membantu pengendalian kendaraan. Teknologi ini menggunakan kamera untuk mengambil gambar lingkungan sekitar kendaraan dan mengolah gambar tersebut menggunakan algoritma *image processing* untuk mengenali rambu lalu lintas yang terdapat di jalan.

Teknologi ini memungkinkan sistem kemudi otomatis untuk mendeteksi dan mengenali rambu lalu lintas yang terdapat di sepanjang jalan raya. Dengan demikian, sistem kemudi otomatis dapat menyesuaikan kecepatan kendaraan sesuai dengan rambu lalu lintas yang terdeteksi, sehingga dapat membantu meningkatkan keamanan berkendara. Selain itu, Teknologi ini dapat membantu menjaga keamanan berkendara dengan memberikan informasi tentang rambu lalu lintas yang terdapat di jalan sehingga kendaraan dapat mengambil tindakan yang tepat untuk menghindari tabrakan dengan kendaraan lain ataupun obstacle lain yang terdapat di jalan.

Teknologi ini juga dapat membantu pengendalian kendaraan dengan memberikan informasi tentang rambu lalu lintas yang

terdapat di jalan sehingga kendaraan dapat mengambil tindakan yang tepat sesuai dengan peraturan lalu lintas yang berlaku.

Di samping itu, teknologi pendeteksi rambu lalu lintas dengan *image processing* juga dapat membantu meningkatkan efisiensi berkendara. Misalnya, dengan menyesuaikan kecepatan sesuai dengan rambu lalu lintas yang terdeteksi, maka kendaraan dapat lebih lancar dan tidak terhambat oleh rambu lalu lintas yang tidak sesuai. Selain itu, teknologi ini juga dapat membantu menghemat bahan bakar karena kendaraan dapat lebih lancar dan tidak harus terus menerus mengerem atau mengerem secara mendadak.

II. Landasan Teori

A. Citra

Citra merupakan sebuah sinyal dwimatra yang dapat diamati oleh sistem visual manusia maupun digital dan berfisat menerus (*continue*). Citra juga dapat diartikan sebagai sebuah gambar baik itu dalam bentuk digital maupun fisikal yang berada di bidang dua dimensi. Citra dapat direpresentasikan dalam bentuk matematis menjadi sebuah fungsy yang menyatakan intensitas cahaya pada suatu titik dalam bidang dua dimensi sebagai berikut.

$$f(x,y)$$

x,y : titik koordinat pada bidang dua dimensi.

$f(x,y)$: intensitas cahaya (*brightness*) pada titik (x,y)

B. Citray Grayscale

Citra grayscale adalah sebuah citra yang hanya mempunyai satu nilai kanal pada setiap pikselnya dan hanya menggunakan tingkatan warna abu-abu. Pada citra ini berarti nilai piksel pada komponen warna merah, hijau, dan biru mempunyai intensitas yang sama. Oleh karena itu, citra grayscale ini lebih mudah diproses pada setiap pengolahan citra karena pada setiap pikselnya hanya mempunyai satu nilai intensitas.

Untuk mengonversi citra berwarna menjadi sebuah citra grayscale dapat dilakukan dengan dua cara. Cara yang pertama yaitu dengan menggunakan rumus yang sederhana seperti berikut.

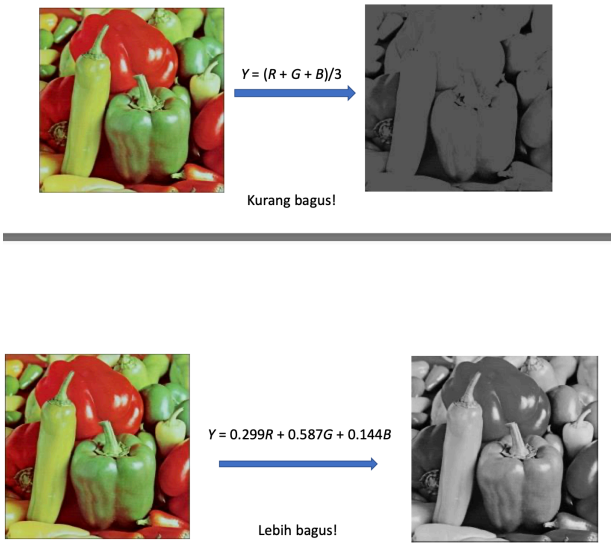
$$Y = \frac{R + G + B}{3}$$

Cara ini merupakan cara yang paling sederhana untuk mengubah citra berwarna yang memiliki tiga kanal merah, hijau, biru pada setiap pikselnya menjadi satu nilai kanal Y (*luminance)* karena hanya mencari rata-rata dari komponen merah, hijau, biru dari setiap piksel. Cara ini memiliki kekurangan yaitu, menghasilkan citra grayscale yang kurang baik.

Cara kedua merupakan cara yang lebih akurat dalam menghasilkan citra grayscale dari citra berwarna dengan rumus sebagai berikut.

$$Y = 0.299R + 0.587G + 0.144B$$

Pada gambar 1 terdapat hasil konversi dari kedua cara untuk lebih memudahkan dalam membandingkan hasil citra grayscale yang lebih baik.



Gambar 1. Hasil konversi citra berwarna menjadi citra grayscale dengan dua cara yang berbeda

Nilai intensitas setiap piksel pada citra grayscale berada diantara 0 hingga 255. Warna hitam memiliki nilai 0 sedangkan warna putih memiliki nilai 255. Namun, nilai intensitas citra grayscale tidak selalu berkisar antara 0 sampai 255. Nilai ini tergantung pada nilai kedalaman pikselnya sehingga citra grayscale memiliki beberapa derajat keabuan yang dapat dilihat pada tabel 1.

Kedalaman Piksel	Skala	Grayscale
1 bit	0 - 1	2^1
2 bit	0 - 3	2^2
4 bit	0 - 15	2^3
8 bit	0 - 255	2^4

Tabel 1. Nilai Derajat keabuan

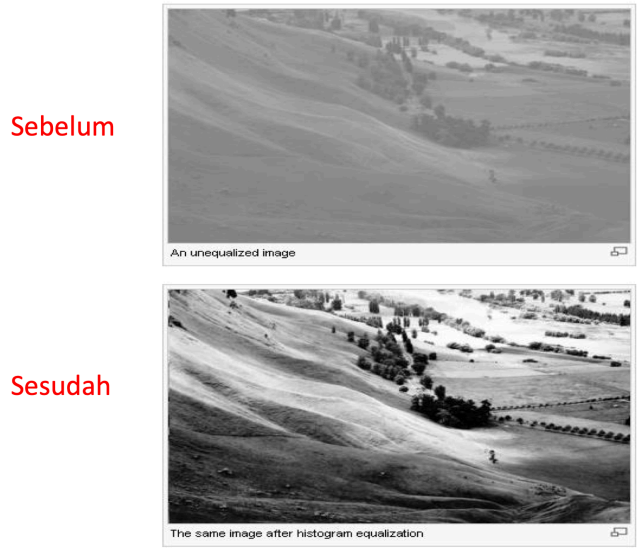
C. Perbaikan Kualitas Citra

Proses perbaikan kualitas citra ini merupakan bagian awal dari proses pngolahan citra (*preprocessing*). Perbaikan kualitas citra (*image enhancement*) bertujuan untuk meningkatkan kualitas citra agar sesuai untuk aplikasi yang lebih lanjut.

Berdasarkan cara kerjanya, metode perbaikan kualitas citra ini dapat dibagi menjadi dua kategori, yaitu perbaikan kualitas citra dalam ranah spasial dan frekuensi. Perbaikan kualitas citra dalam ranah spasial dilakukan dengan memanipulasi piksel-piksel di dalam citra, sedangkan pada ranah frekuensi dilakukan dengan mengubah citra ke ranah frekuensi terlebih dahulu, kemudian memanipulasi nilai-nilai frekuensi tersebut. Beberapa proses yang termasuk dalam perbaikan kualitas citra adalah pengubahan kecerahan citra, citra negatif, peregangan kontras, pengubahan histogram citra, pelembutan citra, penajaman tepi, pewarnaan semu, dan pengubahan geometrik.

Selain proses-proses yang telah disebutkan, terdapat juga beberapa proses lain yang termasuk dalam perbaikan kualitas citra, seperti pemrosesan citra noise reduction yang bertujuan untuk menghilangkan atau mengurangi kebisingan di dalam citra, pemrosesan citra enkripsi yang bertujuan untuk memproteksi informasi di dalam citra, pemrosesan citra restoration yang bertujuan untuk memperbaiki citra yang rusak atau terdegradasi, dan pemrosesan citra segmentation yang bertujuan untuk memisahkan objek di dalam citra menjadi segmen-segmen yang terpisah.

Proses-proses perbaikan kualitas citra dapat dilakukan dengan menggunakan beberapa teknik yang berbeda, seperti teknik aritmetik, teknik transformasi, teknik kompresi, dan teknik pemrosesan sinyal. Setiap teknik memiliki kelebihan dan kekurangan masing-masing, sehingga pemilihan teknik yang tepat akan sangat tergantung pada tujuan dan kondisi citra yang akan diproses.



Gambar 2. Perbandingan citra sebelum dan sesudah dilakukan perbaikan kualitas citra

Pada gambar 2 di atas dapat dilihat kualitas suatu citra dapat meningkat jauh dengan melakukan proses perbaikan kualitas citra.

D. Penapis Citra

Penapisan citra bertujuan untuk meningkatkan kualitas citra atau untuk mengeliminasi informasi yang tidak diinginkan dari citra. Penapisan citra dapat dilakukan untuk berbagai tujuan, seperti:

- Menghilangkan noise atau gangguan dari citra. Noise dapat muncul karena berbagai faktor, seperti kualitas kamera yang rendah, kondisi cahaya yang buruk, atau interferensi elektronik. Penapisan noise dapat memperbaiki kualitas citra dan membuatnya lebih mudah dibaca oleh manusia atau mesin.
- Meningkatkan kontras citra. Penapisan kontras dapat meningkatkan perbedaan antara area terang dan gelap dalam citra, sehingga membuat detail yang tersembunyi lebih terlihat.
- Mengeliminasi informasi yang tidak diinginkan. Penapisan ini dapat digunakan untuk menghilangkan bagian dari citra yang tidak diinginkan, seperti garis-garis atau pola yang tidak penting.
- Menyederhanakan citra. Penapisan ini dapat digunakan untuk mengurangi jumlah informasi yang ada dalam citra, sehingga membuat citra lebih mudah dipahami atau dianalisis.
- Meningkatkan kualitas citra secara keseluruhan. Penapisan citra dapat digunakan untuk meningkatkan kualitas citra secara keseluruhan, seperti memperbaiki kecerahan, warna, atau resolusi citra.

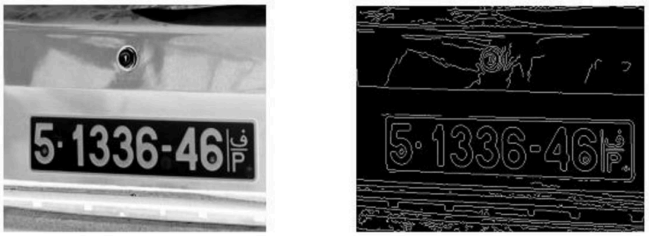
Penapisan citra dapat dilakukan dengan berbagai cara mulai dari ranah spasial hingga ranah frekuensi. Pada ranah frekuensi dapat menggunakan penapis lolos rendah atau penapis lolos tinggi. Pada penapis lolos rendah, ada tiga penapis yang utama yaitu, *ideal lowpass filter, gaussian lowpass filter*, dan *butterwoth lowpass filter*. Pada penapis lolos tinggi, ada tiga juga penapis yang utama yaitu, *ideal highpass filter, gaussian highpass filter*, dan *butterwoth highpass filter*.

E. Pendeteksian Tepi

Deteksi tepi adalah teknik pemrosesan citra yang digunakan untuk menemukan tepi atau batas dari objek atau regio dalam citra. Tepi dapat diartikan sebagai perubahan drastis dalam intensitas citra, seperti perubahan dari bagian terang ke bagian gelap atau dari latar belakang ke objek. Deteksi tepi sangat berguna untuk mengidentifikasi struktur dan detail dalam citra, seperti batas antara objek dan latar belakang, atau tepi dari objek itu sendiri.

Tujuan utama dari deteksi tepi adalah untuk mengidentifikasi tepi dalam citra dengan tepat dan membuatnya lebih mudah dianalisis atau diproses oleh mesin. Deteksi tepi juga dapat digunakan untuk meningkatkan kualitas citra dengan menghilangkan *noise* atau gangguan yang tidak diinginkan.

Deteksi tepi dapat dilakukan dengan berbagai cara, seperti menggunakan operator tepi seperti Sobel, Canny, atau Laplacian, atau dengan menggunakan teknik machine learning seperti deep learning. Setiap metode memiliki kelebihan dan kekurangan masing-masing, dan pilihan terbaik tergantung pada tujuan dan kondisi citra yang akan diproses.



Gambar 3. Contoh penerapan edge detection pada citra

Operator tepi adalah fungsi matematika yang digunakan untuk mendeteksi tepi dalam citra dengan cara mencari perubahan drastis dalam intensitas citra. Beberapa operator tepi yang sering digunakan adalah Sobel, Canny, dan Laplacian.

Sobel operator adalah operator yang digunakan untuk mendeteksi tepi horizontal dan vertikal dalam citra. Operator ini menggunakan matriks 3x3 yang disebut kernel untuk menghitung perubahan intensitas citra di sekitar pixel yang sedang diproses. Hasil dari operator Sobel adalah citra biner yang menunjukkan tepi dalam citra asli dengan warna hitam.

Canny *edge detector* adalah algoritma yang digunakan untuk mendeteksi tepi dalam citra dengan tepat dan tidak terlalu banyak noise. Algoritma ini terdiri dari beberapa tahap, seperti smoothing, gradient, dan non-maximum suppression, yang digunakan untuk menghilangkan noise dan menemukan tepi yang sebenarnya. Hasil dari Canny edge detector adalah citra biner yang menunjukkan tepi dalam citra asli dengan warna hitam.

Laplacian operator adalah operator yang digunakan untuk mendeteksi tepi dalam citra dengan cara mencari perubahan drastis dalam intensitas citra. Operator ini menggunakan matriks 3x3 yang disebut kernel untuk menghitung perubahan intensitas citra di sekitar pixel yang sedang diproses. Hasil dari Laplacian operator adalah citra grayscale yang menunjukkan tepi dalam citra asli dengan intensitas yang lebih tinggi.

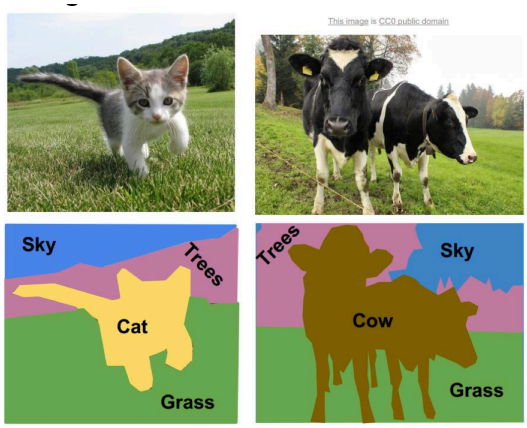
F. Segmentasi Citra

Segmentasi citra juga merupakan proses pembagian citra menjadi beberapa bagian atau wilayah yang berbeda, dengan tujuan untuk mengelompokkan objek atau fitur yang terdapat dalam citra tersebut. Segmentasi citra berguna untuk memisahkan objek atau fitur yang diinginkan dari latar belakang atau objek yang tidak diinginkan, sehingga memudahkan analisis atau pengolahan lebih lanjut.

Segmentasi citra dapat dilakukan dengan berbagai cara, tergantung pada tujuan dan kondisi citra yang akan diproses. Beberapa metode segmentasi citra yang umum digunakan adalah:

- *Thresholding*. *Thresholding* adalah metode segmentasi citra yang paling sederhana, yang memisahkan objek dari latar belakang dengan cara menetapkan nilai ambang (*threshold*) tertentu. Semua piksel yang memiliki nilai di atas ambang akan dianggap sebagai bagian dari objek, sedangkan piksel yang memiliki nilai di bawah ambang akan dianggap sebagai bagian dari latar belakang.
- *Region growing*. *Region growing* adalah metode segmentasi citra yang mengelompokkan piksel-piksel yang memiliki nilai intensitas yang hampir sama menjadi satu wilayah atau bagian. Metode ini dimulai dengan memilih satu pixel sebagai titik awal, lalu mencari piksel - piksel lain yang memiliki nilai intensitas yang hampir sama dan menambahkannya ke dalam wilayah yang sama. Proses ini terus dilakukan sampai tidak ada lagi piksel yang memenuhi kriteria.
- *Watershed*. *Watershed* adalah metode segmentasi citra yang mengelompokkan piksel-piksel yang terletak di tepi objek menjadi satu wilayah atau bagian. Metode ini dimulai dengan mencari tepi dalam citra menggunakan operator tepi, lalu mengelompokkan piksel-piksel yang terletak di tepi tersebut menjadi satu wilayah. Proses ini terus dilakukan sampai semua tepi telah terkelompokkan.
- *Clustering*. *Clustering* adalah metode segmentasi citra yang mengelompokkan piksel-piksel yang memiliki karakteristik yang sama menjadi satu wilayah atau bagian. Metode ini menggunakan algoritma clustering seperti k-means atau fuzzy c-means untuk mengelompokkan piksel-piksel tersebut berdasarkan karakteristik yang diinginkan.

Salah satu pengaplikasian dari segmentasi citra yaitu untuk melakukan pengenalan objek seperti gambar 4 di bawah ini.



Gambar 4. Contoh penerapan segmentasi citra untuk pengenalan objek

III. IMPLEMENTASI

Pada bagian ini akan dijelaskan mengenai proses pengembangan deteksi rambu lalu lintas dengan menggunakan pengolahan citra. Namun, terdapat beberapa batasan dari program yang dibuat pada makalah ini sebagai berikut:

1. Program hanya bisa membaca konten rambu lalu lintas yang mengandung huruf dan angka
2. Program hanya bisa membaca konten rambu lalu lintas jika tidak ada yang menghalangi tulisan huruf dan angka pada rambu lalu lintas.

Pada pengembangan program ini terdapat beberapa tahapan yang dilakukan sebagai berikut.

1. Memuat citra

```
image = imread(image.jpg');
image = imresize(image, [400 NaN]);
```

Perintah imread() digunakan untuk membaca citra dari file dengan nama yang diberikan. Kemudian, perintah imresize() digunakan untuk mengubah ukuran citra agar tetap sama, yaitu 400 pixel tinggi dengan rasio lebar yang sama dengan aslinya.

Berikut ini merupakan contoh citra yang dimuat.

Gambar 5. Citra yang dimuat pada program

2. Konversi citra menjadi citra grayscale

```
imageGray = rgb2gray(image);
```

Perintah rgb2gray() digunakan untuk mengkonversi citra dari format warna RGB (Red, Green, Blue) ke skala abu-abu, yang hanya memiliki satu channel intensitas warna. Hal ini dilakukan agar proses segmentasi citra yang akan dilakukan nantinya lebih mudah, karena hanya memiliki satu channel intensitas yang perlu diperhatikan.

Gambar 6. Citra grayscale hasil konversi

3. Penapisan citra

```
imageGray = medfilt2(imageGray, [3 3]);
```

Perintah medfilt2() digunakan untuk mengaplikasikan median filter pada citra, yang berguna untuk menghilangkan noise atau gangguan yang tidak diinginkan. Median filter bekerja dengan mencari nilai median dari setiap pixel dan sekitarnya, kemudian menggantinya dengan nilai tersebut. Dengan demikian, noise yang terdapat pada citra dapat dihilangkan karena nilai median yang diambil tidak terpengaruh oleh noise tersebut.

Gambar 6. Citra hasil penerapan median filter

4. Mendeteksi tepi

```
se = strel('disk', 1);
ge = imerode(imageGray, se);
gd = imdilate(imageGray, se);
gdiff = imsubtract(gd, ge);
```

Perintah strel() digunakan untuk membuat sebuah struktur elemen yang akan digunakan dalam operasi dilasi dan erosi. Struktur elemen yang digunakan adalah bentuk disk dengan radius 1 pixel. Kemudian, operasi dilasi dan erosi dilakukan menggunakan perintah imdilate() dan imerode(), dengan struktur elemen yang telah dibuat sebelumnya. Tepi dalam citra diidentifikasi dengan mengurangi hasil dari dilasi dan erosi, yaitu dengan menggunakan perintah imsubtract().

Gambar 7. Citra hasil penerapan deteksi tepi

5. Binarization

```
imgGrayDiff = mat2gray(imgGrayDiff);

imgGrayDiff = conv2(imgGrayDiff, [1 1; 1
1]);

binerImg = imbinarize(imgGrayDiff);
```

Perintah mat2gray() digunakan untuk mengubah skala nilai pixel dari citra menjadi 0 sampai 1, yang merupakan skala nilai yang dapat diterima oleh perintah imbinarize(). Kemudian, perintah conv2() digunakan untuk melakukan konvolusi pada citra, yaitu dengan mengaburkan citra dengan menggunakan filter yang diberikan. Filter yang digunakan adalah matriks 2x2 dengan semua elemennya bernilai 1, sehingga citra yang dihasilkan akan lebih halus daripada citra aslinya. Selanjutnya, citra diubah menjadi citra biner menggunakan perintah imbinarize(), yang memisahkan objek dari latar belakang dengan membuat semua pixel yang memenuhi kriteria menjadi warna hitam, sedangkan pixel yang tidak memenuhi kriteria menjadi warna putih.

Gambar 8. Citra biner

6. Penghilangan pixel yang terhubung dengan batas citra

```
binerImg = imclearborder(binerImg);
```

Perintah imclearborder() digunakan untuk menghilangkan pixel yang terhubung dengan batas citra, agar tidak mempengaruhi proses segmentasi yang akan dilakukan nantinya.

Gambar 9. Citra hasil penerapan fungsi imclearborder

7. Penghilangan garis-garis yang tidak merupakan bagian dari wilayah yang diinginkan

```
erode = imerode(binerImg, strel('line',
100, 0));

out = imsubtract(binerImg, erode);
```

Perintah imerode() digunakan untuk menyempitkan garis-garis pada citra menggunakan struktur elemen yang diberikan, yaitu struktur elemen garis dengan panjang 100 pixel dan sudut 0 derajat. Kemudian, garis-garis yang tidak merupakan bagian dari wilayah yang diinginkan dihilangkan dengan mengurangi citra asli dengan hasil operasi erosi, yaitu dengan menggunakan perintah imsubtract().

8. Pematangan citra

```
F = imfill(out, 'holes');

H = bwmorph(F, 'thin', 1);

H = imerode(H, strel('line', 3, 90));
```

Pertama, fungsi imfill digunakan untuk mengisi lubang (holes) pada citra biner. Ini akan mengisi lubang dengan nilai logika 1 (hitam) sehingga citra tidak lagi memiliki lubang. Kedua, fungsi bwmorph digunakan untuk melakukan operasi morfologi pada citra biner. Operasi morfologi merupakan teknik yang memanipulasi citra biner dengan cara mengubah bentuk objek pada citra tersebut. Ketiga, fungsi imerode digunakan untuk melakukan operasi erosi pada citra biner. Operasi erosi akan menghilangkan pixel pada tepi objek pada citra biner.

```
area = bwareaopen(H, 100);
```

Perintah bwareaopen() digunakan untuk menghilangkan wilayah dengan area pixel yang kecil, yang dapat mengganggu proses pengenalan karakter nantinya. Area

pixel yang dianggap kecil ditentukan oleh nilai yang diberikan, yaitu 100 piksel dalam kasus ini.

9. Mendapatkan bounding box dari karakter

```
BBs = GetCharactersRect(area);
```

Fungsi GetCharactersRect() digunakan untuk mendapatkan bounding box dari masing-masing karakter dalam citra. Bounding box tersebut kemudian disimpan dalam sebuah matriks dengan format [x y width height], yang menyatakan posisi dan ukuran dari setiap bounding box.

10. Pemrosesan tiap karakter pada rambu lalu lintas

```
content_traffic_sign= '';
    array_characters = ['A' 'B' 'C' 'D' 'E'
'F' 'G' 'H' 'I' 'J' 'K' 'L' 'M' 'N' 'O' 'P'
'Q' 'R' 'S' 'T' 'U' 'V' 'W' 'X' 'Y' 'Z' '1'
'2' '3' '4' '5' '6' '7' '8' '9' '0'];
    for i = 1 : size(BBs, 1)
       BB = BBs(i, :);
       characters = 'A';
       p = matchingCharacters(imcrop(final,
BB), 'A');
       l = length(array_characters);
       for j = 2 : l
           char = array_characters(j);
           cp =
matchingCharacters(imcrop(final, BB), char);
           if cp > p
               characters = char;
               p = cp;
           end
       end
       content_traffic_sign =
strcat(content_traffic_sign, characters);
    end
```

Untuk setiap karakter dalam citra, fungsi matchingCharacter() digunakan untuk menemukan kemiripan dengan masing-masing template karakter yang tersedia. Kemiripan tersebut diukur menggunakan nilai korelasi, yaitu nilai yang menunjukkan seberapa mirip dua citra yang dibandingkan. Setelah karakter teridentifikasi, maka hasil pengenalan nomor plat dihasilkan dengan menyusun semua karakter tersebut menjadi satu string.

11. Menampilkan hasil

```
disp(content_traffic_sign);
```

Perintah disp() digunakan untuk menampilkan hasil pengenalan tulisan rambu lalu lintas ke layar.

12. Menampilkan citra asli

```
imshow(image);
  hold on
  for k = 1 : size(BBs, 1)
      rectangle('Position', BBs(k, :),
'EdgeColor', 'b', 'LineWidth', 2);
  end
```

Perintah imshow() digunakan untuk menampilkan citra asli ke layar. Kemudian, perintah rectangle() digunakan untuk menambahkan bounding box pada citra, dengan menentukan posisi dan ukuran dari setiap bounding box yang telah didapatkan sebelumnya. Warna bounding box diatur menjadi merah, dengan ketebalan garis 2 pixel.

Gambar 10. Citra hasil akhir dalam mendeteksi rambu lalu lintas dan mendeteksi konten rambu lalu lintas

IV. HASIL

Pengujian program dilakukan pada beberapa citra yang dapat dilihat pada tabel di bawah ini.

Output Gambar	Output Konten
	STOP
	STOP

	80
	30

Tabel 2. Hasil pengujian

V. KESIMPULAN DAN SARAN

Pengolahan citra merupakan salah satu pembelajaran yang sangat banyak penerapan dan manfaatnya dalam kehidupan sehari-hari. Salah satunya yaitu, program pendeteksi rambu lalu lintas dan kontennya dengan menggunakan pengolahan citra. Program ini dapat berjalan cukup baik meskipun masih sangat banyak sekali batasan dalam pengimplementasiannya. Program ini dapat meningkatkan pengendalian kendaraan dan juga keamanan pada sistem kemudi otomatis dengan mengenali rambu lalu lintas di jalan raya.

Harapan dari makalah ini yaitu, dapat dikembangkan lebih lanjut lagi program ini sehingga dapat mendeteksi tidak hanya huruf dan angka pada rambu, tetapi bisa gambar seperti gambar *forbidden*, belok kanan, belok kiri, putar balik, dan sebagainya. Selain itu, diharapkan ditemukan metode yang lebih efektif dan akurat dalam mendeteksi rambu lalu lintas beserta isinya

ACKNOWLEDGMENT

Penulis mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa karena atas berkat dan rahmat-Nya penulis bisa menyelesaikan tugas makalah ini yang berjudul "Pendeteksian Rambu Lalu Lintas beserta Kontennya dengan *Image Processing*". Selain itu, tidak lupa penulis ucapkan terima kasih kepada kedua orang tua dan keluarga yang selalu memberikan dukungan.

Penulis juga mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T., selaku dosen mata kuliah IF4073 atau Interpretasi dan Pengolahan Citra. Karena tanpa beliau, penulis tidak akan mengerti tentang materi yang disampaikan pada makalah ini.

REFERENCES

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2022-2023/09-Image-Enhancement-Bagian2-2022.pdf. Diakses pada 18 Desember 2022

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2020-2021/05-Operasi-dasar-pengolahan-citra-2021.pdf. Diakses pada 18 Desember 2022

[3] https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2022-2023/12-Penapisan-Citra-dalam-Ranah-Frekuensi-2022.pdf. Diakses pada 19 Desember 2022

[4] https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2021-2022/22-Segmentasi-Citra-Bagian1-2022.pdf. Diakses pada 19 Desember 2022

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Desember 2022

Azmi Muhammad Syazwana

13519151