Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 3, Maret 2018, hlm. 1265-1274 http://j-ptiik.ub.ac.id

Perancangan Sistem Deteksi dan Pengenalan Rambu Peringatan Menggunakan Metode *Template Matching*

Thomas Oddy Chrisdwianto[1], Hurriyatul Fitriyah[2], Edita Rosana Widasari [3]

Fakultas Ilmu Komputer, Universitas Brawijaya
E-mail: [1]thomasoddyc@gmail.com, [2]hfitriyah@ub.ac.id, [3]editarosanaw@ub.ac.id

Abstrak

Pengolahan citra digital (PCD) berguna untuk memperbaiki kualitas citra sehingga dapat mengenali suatu objek dan memberikan informasi yang kita butuhkan. Langkah awal yang dilakukan PCD adalah dengan *data acquisition.* Tahap ini merupakan proses untuk mendapatkan data gambar yang akan kita digunakan. Setelah itu proses segmentasi atau pengelompokan gambar. Langkah terakhir adalah penyeleksian dan pengekstrakan, tahap ini gambar yang telah diolah dapat dianalisis dan menampilkan informasi yang terdapat pada citra. Salah satu penggunaan PCD dapat diaplikasikan ke dalam sistem kendaraan yaitu sebagai *driver assistant.* Dengan itu sistem dapat mengurangi tingkat kecelakaan yang disebabkan oleh faktor manusia. Sistem *driver assistant* yang bekerja secara otomatis ini juga dapat membantu pengemudi, mulai dari sistem parkir, berkendara secara *auto pilot,* dapat mendeteksi rambu lalu lintas dan membantu pengendara dalam mengambil tindakan secara tepat. Dalam penelitian ini rambu yang dideteksi adalah rambu peringatan. Rambu peringatan memiliki peran penting bagi pengendara yaitu untuk memperingatkan adanya bahaya yang akan dihadapi. Salah satu metode PCD yang sederhana dan mudah digunakan dalam mengenali citra adalah dengan menggunakan metode *Template Matching*. Metode tersebut bekerja dengan mencocokkan gambar yang diambil dengan *template*. Metode ini memiliki tingkat keberhasilan yang tinggi. Hasil pengujian tingkat keberhasilan program dalam mengenali rambu peringatan mencapai 88%.

Kata kunci: *Pengolahan Citra Digital, Driver Assistant, Rambu Peringatan, Template Matching*

Abstract

Digital image processing is useful for fixing image quality, so it we can recognize objects and provide the information we need. The first step of digital image processing is data acquisition. This process is to get the data that we need from the image. After that is the process of segmentation by grouping the image. The final step is the selection and extracting process, from this process we can analyze and display the information that contained in the image. One of the use of digital image processing is that it can be applied to the vehicle system as automatic driver assistant. With that system we can reduce the rate of accidents caused by human factors. Automatic driver assistant driver system can also help the driver for parking, auto pilot driving, detect traffic signs and assist the driver in performing the appropriate action. In this study, the detected traffic signs was a warning sign. Warning signs have an important role to warn the dangers that drivers must be faced. One of the simplest and easiest methods to recognize images is by using the Template Matching method. This method works by matching the images taken with the template. This method has a high success rate. The results of the program success rate test in recognizing the warning signals reached 88%.

Keywords: *Digital Image Processing, Driver Assistant, Warning Signs, Template Matching*

1. PENDAHULUAN

Pengolahan citra digital dapat didefinisikan sebagai akuisisi dan pengolahan informasi visual yang dilakukan oleh komputer. Pengolahan ini sekarang telah banyak digunakan oleh perusahaan mobil modern sebagai sistem autonomous driver assistant. Karena dapat memudahkan pengemudi dalam kegiatan berkendara, sistem ini memiliki peran penting. Dengan sistem tersebut kendaran dapat mengenali objek dan dapat mengambil tindakan yang sesuai dengan keadaan sekitarnya. Sistem

ini dapat melakukan kegiatan secara otomatis tanpa harus dikendalikan oleh pengemudi. Contohnya dapat memarkirkan kendaraanya sendiri, berjalan tanpa ada yang memegang kemudi atau biasa disebut dengan auto pilot dan bahkan dapat mengenali rambu lalu lintas agar pengemudi dapat mengenali arti rambu tersebut.

Ini juga sebagai salah satu solusi untuk mengurangi tingkat kecelakaan yang terjadi. Karena menurut data dari Korlantas Polri pada tahun 2016 tercatat sebanyak 105.374 kasus kecelakaan di Indonesia. Jumlah ini naik dari pada tahun sebelumnya yaitu 98.970 (Korlantas, 2017). Banyak faktor yang menyebabkan terjadinya kecelakaan. Mulai dari faktor alam, kondisi jalanan, pengendara serta sarana prasarana yang dapat kita temui di jalan raya. Salah satu contoh sarana prasana yang ada dijalan raya adalah rambu – rambu lalu lintas. Terdapat sekitar lebih dari 217 rambu lalu lintas yang terpasang pada jalan raya di Indonesia. Rambu lalu lintas di Indonesia sendiri dibedakan menjadi 4 yaitu: rambu perintah, rambu larangan, rambu peringatan dan rambu petunjuk. Setiap jenis rambu memiliki fungsi dan manfaat yang berguna bagi pengendara.

Dalam penelitian (Galyamichev, 2010) yang berjudul Traffic Signs Detection Method menjelaskan bahwa pengenalan rambu lalu lintas dapat dilakukan menggunakan metode edge detection. Dengan secara real – time¸ sistem ini dapat mendeteksi rambu lalu lintas. Namun pada penelitian ini hanya berfokus pada pendeteksian rambu saja, dengan tingkat keberhasilan mencapai 79%.

Pada penelitian (Mulyadi, 2002) berjudul Pengenalan Rambu Lalu lintas sederhana dengan Menggunakan Metode Template Matching, telah menjelaskan bagaimana metode template matching dapat mendeteksi rambu. Namun pada penelitian tersebut masukan yang diolah hanya gambar foto dan beberapa rambu saja. Penelitian ini juga berisi tentang analisis tingkat keberhasilan metode template matching dalam pencocokan antar rambu.

Dalam penelitian ini rambu yang diolah adalah rambu peringatan. Berbeda dengan rambu yang lainnya rambu peringatan memiliki peran penting. Sesuai dengan namanya, rambu peringatan adalah rambu yang memperingatkan adanya kondisi berbahaya dan berpotensi bahaya agar para pengemudi berhati-hati dalam menjalankan kendaraannya. Dalam penelitian ini juga sistem dapat mengolah citra secara real-time. Sehingga program ini dapat dimanfaatkan oleh pengendara saat mengemudikan kendaraannya.

Untuk mengenali rambu peringatan tersebut, diperlukan metode PCD yang fleksibel dan sederhana. Metode yang digunakan dalam laporan ini adalah metode template matching. Metode tersebut memiliki tingkat keakurasian tinggi dalam mengenali suatu objek.

2. DASAR TEORI

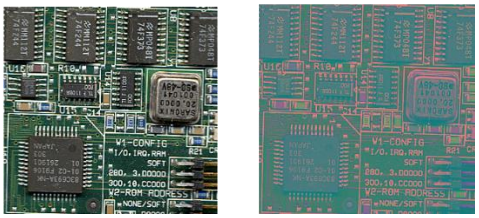
2.1 Pengolahan Citra

Pengolahan citra adalah ilmu yang mengolah sinyal yang berupa citra/gambar secara spesifik. Dalam artian yang sebenarnya, citra merupakan gambar yang dipetakan dalam dua dimensi. Dan dalam arti matematis citra itu fungsi kontinyu dari intesistas cahaya pada bidang 2 dimensi (Permadi, 2015). Manfaat dari pengolah citra adalah untuk memperbaiki bentuk citra, menganalisis dan sebagai pendeteksian citra. Terdapat jenis-jenis citra diukur dari nilai suatu piksel dalam rentang tertentu, antara rentang 0-255 (Indra, 2016).

Langkah awal dalam melakukan pengolahan citra adalah dengan *data acquisition*. Pada langkah ini merupakan proses untuk menetukan metode pengambilan citra serta data yang ingin diolah. Langkah selanjutnya *image segmentation*, ini bertujuan untuk menandai citra dengan garis tepi pada setiap sisinya. Kemudian *feature extraction and selection*, pada langkah ini bertujuan untuk mendapatkan data informative pada citra.

2.2 YCbCr

Citra YCbCr merupakan ruang warna yang sering digunakan dalam video. Citra ini terdiri dari Y, Cb, dan Cr. Y merupakan *Luminance*, Cb merupakan *Chrominance-Blue* dan Cr merupakan *Chrominance-Red* (Hendry, 2012). *Luminance* memiliki kemiripan dengan citra keabuan. Cb kuat dalam mendeteksi warna yang mengandung langit (biru), sedangkan Cr kuat dalam mendeteksi warna kemerahan pada suatu citra. Namun ke duanya lemah dalam mendeteksi warna hijau. Pada gambar 1 merupakan contoh hasil pengubahan citra RGB ke citra YCbCr.



Gambar 1. Citra YCbCr
Sumber : (Mathworks, 2017)

2.3 *Thresholding*

Adalah metode yang digunakan untuk mempertegas citra dari citra *grayscale* ke citra biner. Proses ini merupakan pengelompokan nilai derajat keabuan dari citra *grayscale* menjadi hitam dan putih. Hal yang diperlukan untuk proses *threshold* adalah memilih sebuah nilai *threshold* (T) yang nantinya dijadikan sebagai ambang batas.

$$T = \frac{fMaks+fMin}{2} \quad (1)$$

Dengan fMaks adalah nilai intensitas maksimum pada citra dan fMin adalah nilai intensitas minimum pada citra.

$$g(x,y)=\begin{cases}255\ , jika\ f(x,y) \geq T \\ 0, jika\ f(x,y) < T\end{cases} \quad (2)$$

Dimana *g(x,y)* merupakan citra biner dari *grayscale f(x,y)*. Jika nilai suatu *pixel* melebihi nilai ambang batas yang ditentukan maka akan di set menjadi warna putih, jika kurang dari nilai ambang batas maka akan di set menjadi warna hitam.

2.4 Morfologi

Morfologi adalah salah satu cabang pemrosesan citra digital yang secara spesifik untuk menganalisis bentuk dalam citra (Sianpar, 2013). Morfologi juga merupakan proses pengolahan citra yang didasarkan pada objek atau region yang terdapat dalam citra. Proses ini juga berguna untuk menganalisis bentuk di dalam citra. Morfologi memliki beberapa operasi yang digunakan untuk memproses citra antara lain dilasi, erosi, *closing*, dan *opening*.

Didalam morfologi terdapat pula *Structuring Element* (*Strel*). *Strel* digunakan sebagai parameter untuk mengatur jumlah *pixel* yang akan ditambahkan dan dikurangi. Ini merupakan sebuah himpunan kecil pada operasi morfologi yang meneliti citra yang diproses. Ada beberapa bentuk *Strel* yang biasa digunakan yaitu *rectangle*, *square*, *disk*, *linear*, dan *diamond*.

2.5 Dilasi

Operasi ini digunakan untuk memperbesar ukuran segmen objek dengan menambahakan ruang di tepi objek. Operasi ini membuat gambar menjadi lebih tebal dari gambar sebelumnya dan digunakan untuk memperjelas gambar. Terlihat pada gambar 2, tulisan pada citra tersebut menebal.

Gambar 2. Sebelum dan sesudah dilasi
Sumber: (Mathworks, 2017)

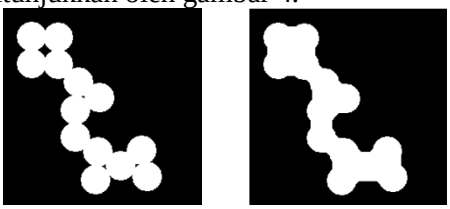
2.6 Erosi

Operasi ini adalah kebalikan dari operasi dilasi. Pada operasi ini ukuran objek akan diperkecil dengan cara mengikis tepi objek. Gambar menjadi terlihat lebih kecil dari gambar sebelumnya. Pada contoh gambar 3, operasi erosi mengurangi atau memperkecil citra kalimat tersebut.

Gambar 3. Sebelum dan sesudah erosi
Sumber: (Mathworks, 2017)

2.7 *Closing*

Operasi *Closing* adalah kombinasi dari antara operasi dilasi dan erosi. Pada proses closing citra yang asli didilasi terlebih dahulu kemudian hasilnya dilakukan operasi erosi. Pada umumnya operasi C*losing* digunakan untuk menggabungkan objek yang berdekatan. Operasi ini ditunjukkan oleh gambar 4.

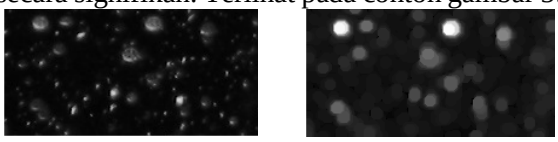


Gambar 4. Sebelum dan sesudah *closing*
Sumber: (Mathworks, 2017)

2.8 *Opening*

Operasi *Opening* juga merupakan kombinasi antara operasi dilasi dan erosi. Namun perbedaannya dengan *Closing* adalah gambar asli dilakukan operasi erosi terlebih dahulu kemudian

operasi dilasi. Operasi ini biasanya digunakan untuk menghilangan gangguan kecil citra pada tepi, seperti tonjolan. Sama seperti halnya closing yaitu memperhalus citra, operasi opening dapat menghilangkan noise dan merubah gambar secara signifikan. Terlihat pada contoh gambar 5.



Gambar 5. Sebelum dan sesudah *opening*
Sumber: (Mathworks, 2017)

2.9 *Template Matching*

Metode ini merupakan algoritma pengenalan citra yang dapat mengenali bagian – bagian dari citra. (Bahri, 2012) Prinsip yang digunakan pada metode ini adalah dengan membandingan gambar asli dengan template gambar yang telah disimpan. Proses pengenalan dilakukan dengan melihat nilai tingkat kemiripan dan nilai batas ambang pengenalan dari sebuah objek. Bila memliki jumlah nilai kemiripan yang tinggi maka akan dikategorikan sebagai objek yang kita kenali.

Kelebihan menggunakan metode ini adalah saat waktu pemrosesan hanya memakan waktu yang relatif singkat/cepat. Ini disebabkan karena proses *template matching* menggunakan proses berupa matriks, sehingga sesuai untuk diterapkan ke dalam sistem yang membutuhkan proses secara *real-time*. Tetapi metode ini juga memliki kekurangan yaitu memerlukan *template* yang banyak agar hasil pengenalan akan semakin akurat dan optimal.

Sebelum di proses image perlu diubah kedalam *grayscale*, dan image akan diolah ke dalam perhitungan dengan persamaan berikut:

$$S = 1 - \frac{\sum_{x=0}^{x-1}\sum_{y=0}^{y-1}|I(x,y)-T(x,y)|}{L\,.\,X\,.\,Y} \quad (3)$$

Dengan keterangan S adalah nilai kemiripan yang bernilai 1 – 0. Dan $I(x,y)$ adalah objek yang dan $T(x,y)$ merupakan *template* yang telah disimpan didalam sistem. X dan Y adalah dimensi dari gambar, sedangkan L adalah maksimum dari tingkat *grayscale.*

2.10 Rambu Peringatan

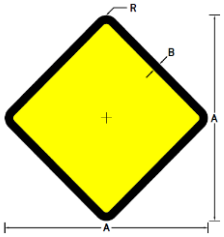
Rambu jenis ini digunakan untuk memberi peringatan kemungkinan ada bahaya atau tempat berbahaya di depan pengguna jalan. Warna dasar rambu peringatan berwarna kuning dengan lambang atau tulisan berwarna hitam. Pada gambar 2. adalah contoh rambu peringatan.

Gambar 6. Contoh rambu peringatan
Sumber: (Perhubungan, 2016)

Umumnya rambu peringatan terletak pada jalan yang memerlukan tingkat konsentrasi yang tinggi dan memiliki potensi bahaya bagi pengendara. Seperti pada persimpangan, tikungan, atau tempat yang sering terjadi bencana alam. Oleh karena itu rambu ini memliki peran penting bagi keselamatan pengendara.

Dalam peraturan pemerintah nomor 13 tahun 2014 telah mengatur tentang standart pembuatan rambu peringatan. Peraturan ini berisi ukuran, jenis huruf, dan bentuk rambu peringatan.

Jenis huruf yang dipakai dalam pembuatan rambu peringatan adalah Clearview Highway. Diatur pula untuk penulisan singakatan, yaitu penggunaan huruf kecil untuk singkatan satuan panjang sedangkan penggunaan huruf kapital untuk singkatan satuan berat.



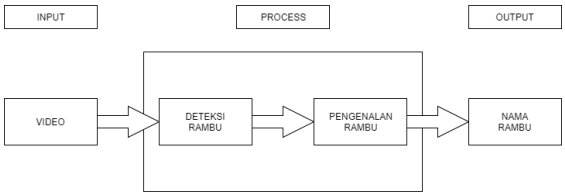
Gambar 7. Peraturan pembuatan rambu peringatan
Sumber : (Perhubungan, 2016)

Gambar 7 merupakan peraturan yang menjelaskan bentuk standart rambu peringatan. A merupakan panjang dan lebar rambu, R merupakan besar juring di setiap sisi rambu, sedangkan B adalah lebar garis hitam pada rambu.

Pemasangan jenis ukuran rambu berdasarkan batas kecepatan rencana yang ada pada jalan tersebut. Kecepatan rencana merupakan kecepatan yang dapat ditempuh oleh kendaraan di jalan yang lancer. Untuk jenis ukuran kecil dipasang pada jalan dengan kecepatan rencana 30km/jam. Untuk jenis ukuran sedang dipasang pada jalan dengan kecepatan rencana 60km/jam. Untuk jenis ukuran besar dipasang pada jalan dengan kecepatan rencana 80km/jam. Sedangkan untuk jenis ukuran sangat besar dipasang pada jalan dengan kecepatan rencana diatas 80km/jam.

3. PERANCANGAN DAN IMPLEMENTASI

3.1. Gambaran Umum Sistem



Gambar 8. Gambaran umum sistem

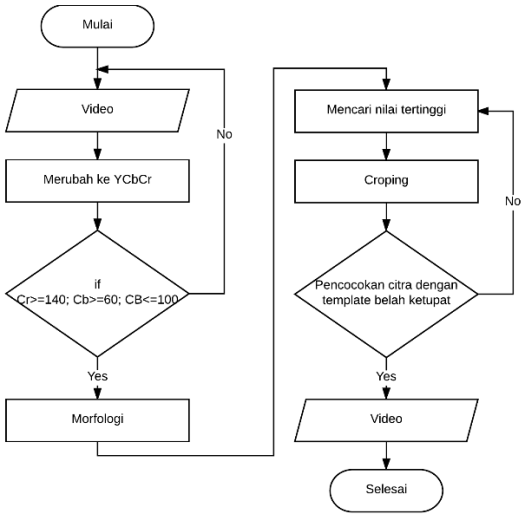
Pada gambar 8 menjelaskan secara umum sistem bekerja, ada 3 bagian utama pengolahan citra pada sistem ini. Pada bagian pertama yaitu input, citra yang dimasukkan adalah video yang tertangkap oleh kamera atau webcam. Proses pengambilan video pada sistem dilakukan secara real time. Pada bagian selanjutnya terdapat bagian proses yang terdiri dari dua pemrosesan, yaitu proses deteksi dan pengenalan. Pada proses deteksi dilakukan pencarian rambu peringatan dengan video real time. Pada proses ini juga pendeteksian dilakukan untuk membedakan mana saja yang termasuk ke dalam rambu peringatan. Kemudian proses selanjutnya adalah proses pengenalan, pada proses ini berfungsi sebagai pembeda antar rambu peringatan. Dengan menggunakan metode template matching, hasil gambar yang terdeteksi diolah dan dicocokan dengan template yang telah kita siapkan. Setelah semua proses dilakukan, sistem menghasilkan sebuah output berupa nama rambu peringatan tersebut.

3.2. Perancangan Perangkat Keras

Perangkat keras yang diperlukan dalam penelitian hanya ada dua, yaitu kamera dan laptop. Kamera digunakan untuk menangkap citra, yang digunakan sebagai input program. Sedangkan laptop digunakan sebagai perangkat yang memproses citra input menjadi sebuah output.

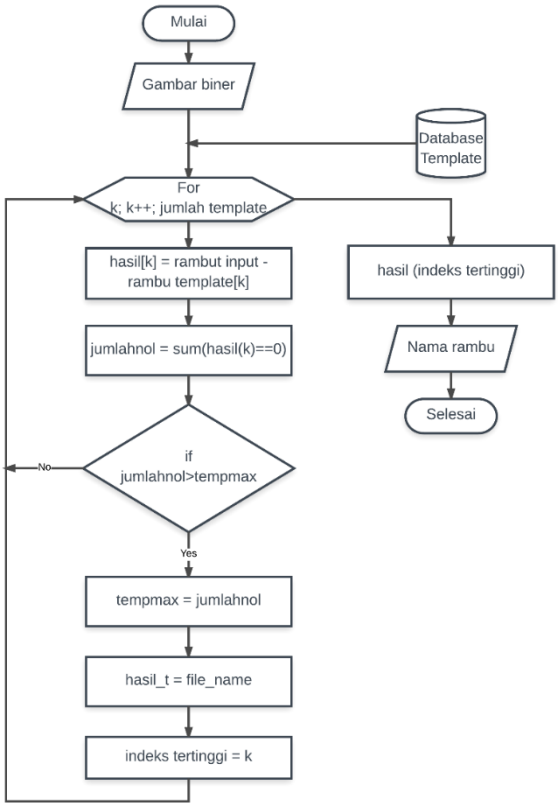
3.3. Perancangan Perangkat Lunak

Perancangan perangkat lunak berisi tentang bagaimana merancang pendeteksian dan pengenalan rambu lalu lintas khususnya rambu peringatan.



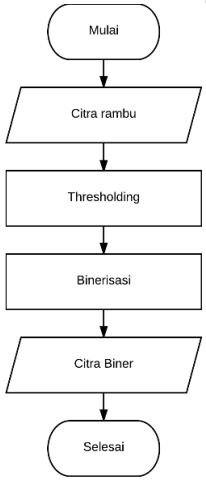
Gambar 9. Flowchart proses pendeteksian

Pada gambar 9 adalah proses pendeteksian rambu pada setiap frame video. Program akan merubah setiap frame video menjadi citra YCbCr, setelah itu diseleksi untuk mendapatkan gambar yang terdeteksi. Hasil dari seleksi akan proses melalui morfologi untuk menghilangkan derau. Setelah proses morfologi, program akan mencari nilai indeks tertinggi dari citra yang terdeteksi. Nilai indeks tertinggi digunakan untuk mencari objek yang harus diproses, dan berfungsi untuk proses cropping. Pada proses cropping dibutuhkan nilai indeks karena pada indeks tersebut terdapat nilai batas objek. Kemudian program menggunakan metode template matching untuk mencocokan objek yang terdeteksi dengan template belah ketupat yang merupakan bentuk dari rambu peringatan. Proses pembuangan background pada citra yang terdeteksi difungsikan untuk memaksimalkan proses pencarian nilai threshold. Untuk mencari nilai threshold program menggunakan metode Otsu. Terakhir setelah menemukan nilai threshold, maka diubah menjadi gambar biner. Gambar biner dapat mempermudah proses pengenalan rambu.



Gambar 10. Flowchart proses pengenalan

Terlihat pada gambar 10 citra dari hasil proses pendeteksian berbentuk gambar biner dicocokan dengan template menggunakan metode Template Matching. Untuk itu diperlukan database template yang telah disiapkan sebelumnya. Proses pencocokan dilakukan secara berulang sesuai dengan jumlah template yang telah disiapkan. Untuk setiap perulangan, citra yang terdeteksi dicocokan dengan template dan diolah. Pengolahan awal adalah dengan mencari nilai selisih antara citra terdeteksi dengan template. Setelah semua selisih diketahui program akan mencari dengan nilai selisih jumlah nilai 0 yang paling besar. Setelah semua dicocokan program menampilkan nama file yang berasal dari indeks yang memiliki nilai selisih jumlah nilai 0 terbanyak.



Gambar 11. Flowchart pembuatan template

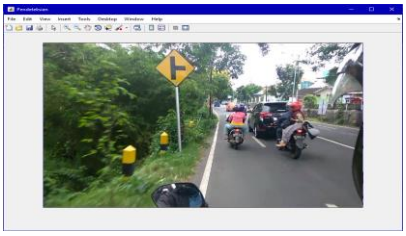
Pada flowchart yang ditampilkan gambar 11 merupakan proses pembuatan template. Proses ini terdiri dari proses pendeteksian, yang sebelumnya telah dijelaskan, kemudian proses selanjutnya adalah thresholding atau mencari nilai threshold. Setelah nilai tersebut ditemukan maka akan dilakukan proses binerisasi dan menghasilkan gambar biner yang merupakan template pengenalan rambu.

Untuk menambah tingkat keberhasilan program dalam mengenali rambu peringatan, diperlukan beberapa tambahan template rambu dengan tingkat kemiringan tertentu. Pada penelitian ini, tingkat kemiringan yang dipakai adalah 5^o ke kiri dan ke kanan. Untuk memiringkan template dilakukan fungsi imrotate.

3.4. Implementasi Perangkat Keras

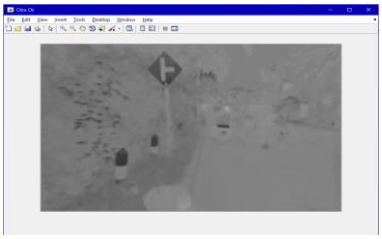
Perangkat keras berupa kamera Logitech c270h dengan resolusi 720p dipasangkan pada laptop dengan menghadap lurus kedepan. Kamera tersebut terhubung dengan laptop menggunakan port USB 2.0. Serta dibutuhkan komputer/laptop yang dapat memproses pengolahan citra.

3.5. Implementasi Perangkat Lunak



Gambar 12. Gambar video

Gambar 12 adalah potongan gambar yang diambil saat proses pengenalan rambu peringatan. Gambar tersbut memiliki tipe gambar RGB.

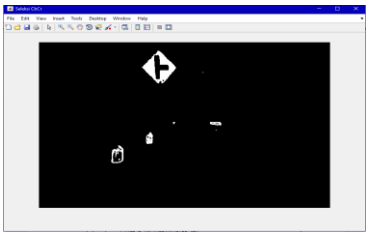


Gambar 13. Citra Cb

Gambar 13 merupakan tampilan dari gambar yang telah diubah menjadi citra Cb. Dapat terlihat jelas rambu peringatan yang terdeteksi, namun juga terdapat beberapa objek selain rambu peringatan.

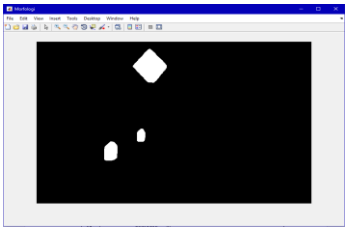
Gambar 14. Citra Cr

Gambar 14 menunjukan tampilan citra Cr. Meskipun terlihat banyak derau ketika gambar diubah menjadi citra Cr, namun rambu peringatan tetap terdeteksi.



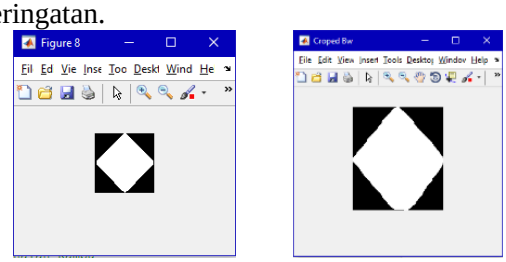
Gambar 15. Merupakan hasil seleksi Cb dan Cr

Terlihat hasil seleksi yang dilakukan terhadap Cb dan Cr pada gambar 3.8. Ini dikarenakan citra tersebut mampu mendeteksi file video. Selain itu citra Cb dan Cr digunakan untuk mengatasi kekurangan citra RGB dalam mendeteksi sensitas pencahayaan Pada gambar ini terlihat ada beberapa objek terdeteksi termasuk rambu peringatan.



Gambar 16. Hasil morfologi

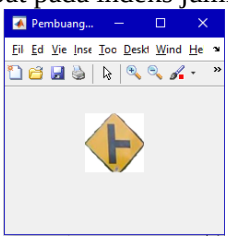
Pada gambar 16 merupakan hasil pengolahan citra setelah melewati tahap morfologi. Terlihat pada gambar tersebut derau semakin berkurang dan hanya 3 objek yang terdeteksi, termasuk diantaranya adalah rambu peringatan.



Gambar 17. Template post dan CropedBbw

Gambar 18. Hasil pencocokan objek dengan template belah ketupat

Pada gambar 17 merupakan citra post dan citra croped Bbw. Kedua citra tersebut akan dicari selisihnya. Pada gambar 18 merupakan hasil dari pencocokan dengan template. Terlihat garis putih tersebut merupakan selisih dan nilainya terdapat pada indeks jumlah cek.



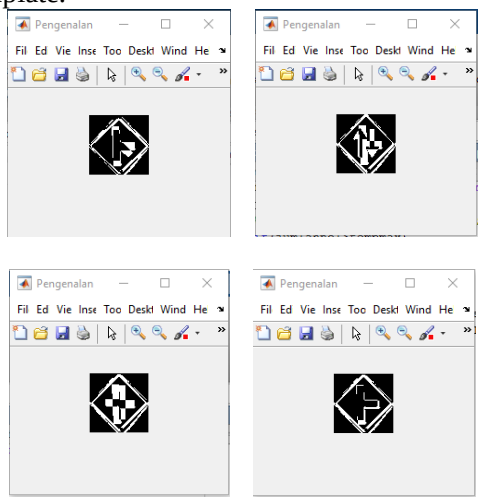
Gambar 19. Hasil pembuangan background

Gambar 19 merupakan hasil proses menghilangkan background, proses ini diperlukan untuk menyempurnakan pengolahan gambar dalam mencari nilai threshold.

Gambar 20. Hasil binerisasi

Gambar 20 merupakan hasil dari pemrosesan binerisasi terhadap rambu yang terdeteksi. Proses ini dilakukan untuk

memudahkan program dalam proses pencocokan template.



Gambar 21. Proses *template matching*

Gambar 21 merupakan proses pencocokan rambu input dan template menggunakan metode template matching. Setiap template pada database akan dicocokan dengan rambu yang terdeteksi. Terlihat pada gambar terdapat proses yang memiliki nilai selisih yang kecil. Dan hasil dari proses template matching ditampilkan pada konsol matlab.

4. PENGUJIAN DAN HASIL

4.1 Pengujian Pada Rambu Peringatan Tegas

Hasil dari pengujian program pendeteksian dan pengenalan rambu berupa nama rambu. Nama rambu tersebut ditampilkan pada konsol matlab. Pada hasil pengujian terdapat posisi pengambilan gambar, gambar yang di tangkap oleh matlab, hasil pengenalan program, dan status yang berisi berhasil atau tidak program tersebut. Berikut adalah hasil pengujian yang dilakukan.

Tabel 1. Pengujian Rambu Peringatan Tegas

NO	POSISI	GAMBAR MATLAB	HASIL DETEKSI MATLAB	STATUS
1	1		peringatantegas.	Berhasil
2	1		peringatantegas	Berhasil
3	1		peringatantegas.	Berhasil
4	2		peringatantegas	Berhasil
5	2		peringatantegas.	Berhasil
6	2		peringatantegas	Berhasil
7	3		peringatantikungangandakanan	Gagal
8	3		peringatantegas	Berhasil
9	3		peringatantegas.	Berhasil

Terlihat dari hasil pengujian pada tabel 1 bahwa program sudah dapat berjalan baik pada saat mengenali rambu peringatan tegas. Namun terlihat terdapat kesalahan citra pada saat program mendeteksi rambu pada percobaan 7. Sehingga tingkat keberhasilan program pada rambu peringatan tegas mencapai 89%. Kesalahan pendeteksian tersebut terjadi karena saat proses pengambilan gambar yang dilakukan tidak stabil. Hal ini membuat program sulit mendeteksi citra rambu peringatan tegas dan gagal dalam mengenali rambu.

4.2 Pengujian Pada Rambu Persimpangan Empat

Tabel 2. Pengujian Rambu Peringatan Persimpangan Empat

NO	POSISI	GAMBAR MATLAB	HASIL DETEKSI MATLAB	STATUS
1	1		peringatansimpangempat	Berhasil
2	1		peringatanpertigaankiri	Gagal
3	1		peringatansimpangempat	Berhasil
4	2		peringatansimpangempat	Berhasil
5	2		peringatanpertigaankiri	Gagal
6	2		peringatanpertigaankiri	Gagal
7	3		peringatansimpangempat.	Berhasil
8	3		peringatansimpangempat	Berhasil
9	3		peringatanpertigaankanan	Gagal

Hasil pengujian pada tabel 2 menunjukan bahwa program empat kali gagal dalam mengenali rambu peringatan persimpangan empat. Pada percobaan ke 2, 5 dan 6 menunjukan hasil yang sama yaitu peringatan pertigaan kiri. Kegagalan pada percobaan tersebut dikarenakan program menemukan kecocokan dan selisih yang kecil dengan template rambu peringatan pertigaan kiri. Sama halnya dengan percobaan ke 9, program menemukan kecocokan dan selisih nilai yang kecil dengan template rambu peringatan pertigaan kanan. Oleh karena itu pada pengujian ini tingkat keberhasilan program hanya mencapai 66%

4.3 Pengujian Pada Rambu Peringatan Simpang Tiga Berbentuk T

Tabel 3. Pengujian Rambu Peringatan Simpang Tiga Berbentuk T

NO	POSISI	GAMBAR MATLAB	HASIL DETEKSI MATLAB	STATUS
1	1		peringatansimpangtigaT.	Berhasil
2	1		peringatansimpangtigaT	Berhasil
3	1		peringatansimpangtigaY	Gagal
4	2		peringatansimpangtigaT	Berhasil
5	2		peringatansimpangtigaT	Berhasil
6	2		peringatansimpangtigaY	Gagal
7	3		peringatansimpangtigaT	Berhasil
8	3		peringatansimpangtigaT	Berhasil
9	3		peringatansimpangtigaT	Berhasil

Pada hasil pengujian 3 terdapat dua kali kegagalan, kegagalan tersebut terjadi pada percobaan ke 3 dan 6. Kedua percobaan tersebut memiliki hasil pengenalan yang sama yaitu rambu peringatan simpang tiga Y. Dimana rambu tersebut memiliki kemiripan bentuk dengan rambu peringatan simpang tiga T. Percobaan ini memiliki tingkat keberhasilan yang mencapai 88%.

4.4 Pengujian Pada Rambu Peringatan Simpang Tiga Berbentuk T

Tabel 4. Pengujian Rambu Peringatan Simpang Tiga Berbentuk T

NO	POSISI	GAMBAR MATLAB	HASIL DETEKSI MATLAB	STATUS
1	1		peringatanpertigaankiri	Berhasil
2	1		peringatanpertigaankiri	Berhasil
3	1		peringatanpertigaankiri	Berhasil
4	2		peringatanpertigaankiri	Berhasil
5	2		peringatanpertigaankiri	Berhasil
6	2		peringatanpertigaankiri	Berhasil
7	3		peringatanpertigaankiri	Berhasil
8	3		peringatanpertigaankiri	Berhasil
9	3		peringatanpertigaankiri	Berhasil

Pada hasil pengujian yang ditunjukan oleh tabel 4, bahwa pendeteksi dan pengenalan rambu peringatan simpang tiga kiri telah berhasil dan tidak ada kegagalan dalam mendeteksi dan mengelani rambu peringatan simpang tiga kiri. Sehingga tingkat keberhasilan pada pengujian ini adalah 100%. Ini terjadi karena gambar yang ditangkap oleh program memiliki tingkat kecocokan yang tinggi dengan template rambu peringatan pertigaan kiri.

4.5 Pengujian Pada Rambu Peringatan Alat Pemberi Isyarat Lalu Lintas

Tabel 5. Pengujian Rambu Peringatan Alat Pemberi Isyarat Lalu Lintas

NO	POSISI	GAMBAR MATLAB	HASIL DETEKSI MATLAB	STATUS
1	1		peringatantrafficlight	Berhasil
2	1		peringatantrafficlight	Berhasil
3	1		peringatantrafficlight	Berhasil
4	2		peringatantikungankiri	Gagal
5	2		peringatantrafficlight	Berhasil
6	2		peringatantrafficlight	Berhasil
7	3		peringatantrafficlight	Berhasil
8	3		peringatantrafficlight	Berhasil
9	3		peringatantrafficlight	Berhasil

Hasil pengujian pada tabel 5 menunjukan bahwa program dapat sudah cukup baik dalam mendeteksi dan mengenali rambu peringatan alat pemberi isyarat lalu lintas. Namun ada satu kegagalan sistem dalam mengenali rambu. Ini terjadi karena kesalahan sistem pada saat pendeteksian citra tersebut. Gambar YCbCr yang dihasilkan mempengaruhi proses pengenalan pada program. Dalam hal ini citra YCbCr yang dihasilkan memiliki kekurangan. Pada hasil pengujian tersebut tingkat keberhasilan program mencapai 89%.

5. KESIMPULAN

Kesimpulan ditujukan untuk menjawab rumusan masalah yang telah dibuat. Oleh karena itu dapat diambil kesimpulan sebagai berikut.

1. Kesimpulan ditujukan untuk menjawab rumusan masalah yang telah dibuat. Oleh karena itu dapat diambil kesimpulan sebagai berikut.
2. Untuk merancang sistem pendeteksian dan pengenalan rambu peringatan dengan metode template matching dapat dilakukan menggunakan aplikasi matlab.
3. Untuk mendeteksi rambu peringatan dapat dilakukan dengan mengubah citra awal berjenis RGB ke citra YCbCr lalu menyeleksinya.
4. Metode template matching bekerja dengan cara mencocokan citra yang dideteksi dan database template yang telah disiapkan.
5. Rata – rata tingkat keberhasilan metode template matching dalam mengenali rambu peringatan sudah cukup baik yaitu mencapai 88%.

6. DAFTAR PUSTAKA

Bahri, R. S., 2012. Perbandingan Algoritma Template Matching dan Feature Extraction Pada Optical Character Recognition. I Volume ed. Bandung: Jurnal Komputer dan Informatika.

Febriani dan Lussiana, 2008. Analisis Penelusuran Tepi CItra Menggunakan Detektor Tepi Sobel dan Canny. Depok: KOMMIT.

Galyamichev, P., 2010. Traffic Signs Detection Method, Lappeenranta: Lappeenranta University.

Hendry, J., 2012. Color Conversion - RGB to YCbCr, Yogjakarta: EE & IT of UGM.

Hukum, M., 2006. Peraturan Pemerintah Republik Nomor 34 Tahun 2006 Tentang Jalan. Jakarta: Pemerintah Indonesia.

Indra, F., 2016. Rancang Bangun Sistem Identifikasi Plat dan Nomor Kontainer Sebagai Pengendali Palang Pintu Depot Kontainer Menggunakan Template Matching. Malang: Universitas Brawijaya.

Mathworks, 2017. imopen. [Online] Tersedia di: https://www.mathworks.com/help/images/ref/imopen.html [Diakses 5 Juli 2017].

Mathworks, 2017. im2bw. [Online] Tersedia di: https://www.mathworks.com /help/images/ref/im2bw.html [Diakses 5 Juli 2017].

Mathworks, 2017. imclose. [Online] Tersedia di: https://www.mathworks.com/help/images/ref/imclose.html [Diakses 5 Juli 2017].

Mathworks, 2017. imdilate. [Online] Tersedia di: https://www.mathworks.com/help/images/ref/imdilate.html [Diakses 5 Juli 2017].

Mathworks, 2017. imerode. [Online] Tersedia di: https://www.mathworks.com/help/images/ref/imerode.html [Diakses 5 Juli 2017].

Mathworks, 2017. rgb2gray. [Online] Tersedia di: https://www.mathworks.com/help/matlab/ref/rgb2gray.html [Diakses 5 Juli 2017].

Mathworks, 2017. rgb2ycbcr. [Online] Tersedia di: https://www.mathworks.com/help/images/ref/rgb2ycbcr.html [Diakses 5 Juli 2017].

Mulyadi, H., 2002. Pengenalan Rambu Lalu Lintas Sederhana Dengan Menggunakan Metode Template Matching. Surabaya: Universitas Kristen Petra.

Perhubungan, M., 2016. Peraturan Menteri Perhubungan Republik Indonesia Nomor PM 13 Tentang Lalu Lintas. Jakarta: Menteri Perhubungan.

Permadi, Y., 2015. Aplikasi Pengolahan Citra Untuk Identifikasi Kematangan Mentimun Berdasarkan Tekstur Kulit Buah Menggunakan Metode Eekstraksi Ciri Statistik. Yogyakarta: JURNAL INFORMATIKA.

Sianpar, R., 2013. Pemrograman Matlab Dalam Contoh dan Terapan. Bandung: Informatika Bandung.