Kendali Lampu Lalu Lintas dengan Deteksi Kendaraan Menggunakan Metode *Blob Detection*

Qory Hidayati[1]

***Abstract*— Traffic jam is a major traffic problem often found in big cities of Indonesia. It is because the number of vehicles increases annually. Therefore, a simulation to detect the number of vehicles in every lane of traffic is needed to monitor the traffic. Traffic control is also required in order to reduce traffic jam. This paper develops a vehicle detection and counting system using image processing. Detection is carried out using image segmentation which is processed by object filtering and blob extraction. Morphological operators are employed for blob extraction. Testing is conducted using a video obtained from the ATCS Bandung. The video is taken at the Laswi - A. Yani intersection Bandung. Software prototypes are created in C++, using Windows Forms Application as a programming library for Windows and Open CV for image processing module. The result shows that blob detection method can give good results if there is no intersection between blobs of each car. The performance is poor when this method is used for heavy traffic conditions, where the cars are close to each other. The performance level of sensitivity is 91.67%, precision is 61.11%, specificity is 80.55%, f-Measure is 73.33%, and accuracy is 83.33%. The accuracy for vehicles detection on sunny condition is 82.11% and reduced by 76.50% on rainy condition. This method works better in quiet condition, with accuracy of 83.07%, and is reduced by 67.70% in crowded condition. The average processing time is 0.042 seconds when using video, and 0.033 seconds using real camera.**

***Intisari*— Kemacetan merupakan salah satu masalah lalu lintas yang sering ditemukan di kota-kota besar Indonesia. Hal ini dikarenakan semakin meningkatnya jumlah kendaraan setiap tahunnya. Oleh karena itu, dibutuhkan simulasi untuk mendeteksi jumlah kendaraan pada setiap jalur lalu lintas dan pengendali lampu lalu lintas guna mengurangi kemacetan. Makalah ini menggunakan metode *blob detection* untuk estimasi deteksi kendaraan. Deteksi dilakukan dengan *image segmentation* yang diproses menggunakan *filtering object* dan ekstraksi *blob*. Operator morfologi digunakan dalam melakukan ekstraksi *blob*. Pengujian dilakukan menggunakan video yang diambil dari ATCS kota Bandung pada simpang empat jalan Laswi-A.Yani, Bandung. Purwarupa perangkat lunak dibuat dalam bahasa C++, menggunakan Windows Forms Application sebagai *library* untuk pemrograman Windows, dan OpenCV untuk modul pengolahan citra. Hasil menunjukkan bahwa metode *blob detection* dapat memberikan hasil yang cukup baik apabila *blob* antar kendaraan berjarak. Metode ini tidak memberikan unjuk kerja yang baik apabila digunakan pada kondisi lalu lintas yang padat dengan badan kendaraan saling menempel. Unjuk kerja yang dihasilkan adalah tingkat sensitivitas sebesar 91,67 %, kekhususan sebesar 80,55%, dan akurasi sebesar 83,33%. Tingkat akurasi untuk deteksi kendaraan pada kondisi cerah adalah sebesar 82,11% dan berkurang sebesar 76,50% pada kondisi hujan. Pada kondisi lengang menjadi lebih baik, yaitu sebesar 83,07% dan berkurang sebesar 67,70 % pada kondisi ramai.**

***Kata kunci*-- deteksi kendaraan, pengolahan citra digital, *blob detection*, morfologi, OpenCV**

I. PENDAHULUAN

Pada setiap tahun, peningkatan jumlah kendaraan semakin bertambah. Hal ini berakibat munculnya masalah lalu lintas kendaraan, misalnya kemacetan lalu lintas, kecelakaan lalu lintas, polusi udara, dan sebagainya. Kemacetan lalu lintas telah menjadi masalah signifikan. Beberapa hal yang telah dilakukan untuk menangani masalah kemacetan misalnya awal infrastruktur transportasi, lebar trotoar, dan pelebaran jalan. Namun, hal ini belum mampu mengurangi kemacetan kota, sehingga banyak peneliti tertarik melakukan penelitian terhadap sistem transportasi cerdas (*intelligent transportation systems*, ITS), seperti memprediksi arus lalu lintas pada pemantauan kegiatan di persimpangan lalu lintas untuk mendeteksi kemacetan. Dalam hal ini diperlukan peningkatan pengawasan lalu lintas pada deteksi kendaraan. Deteksi kendaraan secara otomatis dalam data *video surveillance* merupakah bagian dari *computer vision*.

Deteksi kendaraan dan penghitungan kendaraan adalah hal penting dalam komputasi kemacetan lalu lintas di jalan raya. Sistem yang diusulkan di sini bertujuan untuk memberikan data jumlah kendaraan pada persimpangan jalan raya. Tujuan utama makalah ini adalah untuk mengembangkan metode deteksi kendaraan secara otomatis dan menghitung kendaraan pada simpang empat lalu lintas. Sebuah sistem diimplementasikan untuk mendeteksi dan menghitung objek secara dinamis dan statis. Pemantauan jalan raya secara visual adalah komponen utama dalam mengembangkan sistem transportasi cerdas.

Dalam makalah ini, metode *blob detection* digunakan pada algoritme deteksi kendaraan untuk mengetahui jumlah kendaraan pada masing-masing jalur simpang empat lalu lintas. Keluaran jumlah kendaraan ditampilkan pada *user interfacing traffic monitoring*. Hasil dari keluaran jumlah kendaraan akan digunakan sebagai masukan pada pengendali lampu lalu lintas menggunakan *fuzzy* [1].

Makalah ini terdiri atas enam bagian: bagian pendahuluan, bagian literatur, bagian metode yang digunakan, bagian perancangan sistem, bagian implementasi dan hasil pengujian, dan yang terakhir bagian kesimpulan.

II. SISTEM PENGENDALI LALU LINTAS

Penelitian tentang beberapa metode dan penerapannya dalam sistem pengendali lalu lintas sudah banyak dilakukan. Penelitian yang relevan dengan makalah ini adalah dalam

[1] *Dosen Elektronika, Politeknik Negeri Balikpapan, Jl. Soekarno Hatta Km.8 Balikpapan 76125 INDONESIA (tlp: 0542-860 895; fax: 0542-861107; e-mail: qory.hidayati@poltekba.ac.id)*

sistem pengendali cerdas lalu lintas dalam menggunakan *background estimation* untuk memfilter setiap kendaraan dari *background* dan dengan menggunakan logika *fuzzy* untuk mengendalikan lampu lalu lintas [2]. Sebuah sistem *real time* menggunakan metode berbasis fitur bersama dengan oklusi digunakan untuk pelacakan kendaraan dalam kondisi lalu lintas padat [3]. Sebuah pengenalan metode obyek bergerak menggunakan *blob analysis* untuk memperbaiki model *background* [4]. Hasil yang diperoleh dari penelitian ini adalah sistem bekerja dengan baik karena adanya lingkungan yang kompleks dan pencahayaan lingkungan. Deteksi ROI dan klasifikasi untuk mobil dan sepeda menggunakan SVM juga telah dilakukan [5]. Meskipun telah banyak penelitian tentang deteksi kendaraan dan pelacakan, tetapi penelitian terkait deteksi kendaraan pada kendaraan bergerak maupun diam masih sedikit, sehingga digunakan metode *blob detection* untuk dapat menyelesaikan masalah deteksi kendaraan pada simpang empat lalu lintas.

III. DETEKSI KENDARAAN

Sistem penghitungan kendaraan otomatis ini menggunakan data video yang diperoleh dari kamera lalu lintas. Metode *blob detection* digunakan untuk mengolah serangkaian *frame* yang diperoleh dari video untuk menghitung jumlah kendaraan pada setiap jalur. Pada setiap *frame* ditentukan *region of interest* (ROI) dengan melacak objek yang terdeteksi di dalam ROI tertentu dan kemudian melakukan perhitungan. Beberapa teori yang digunakan pada deteksi kendaraan adalah sebagai berikut.

A. Region of Interest (ROI)

ROI merupakan subcitra dari suatu region. Hal ini merefleksikan kenyataan bahwa suatu citra seringkali memiliki suatu koleksi objek yang dapat menjadi suatu region tersendiri. ROI memungkinkan dilakukannya pengkodean secara berbeda pada area tertentu dari citra digital, sehingga mempunyai kualitas yang lebih baik dari area sekitarnya (*background*). Fitur ini menjadi sangat penting bila terdapat bagian tertentu dari citra digital yang dianggap lebih penting dari bagian lainnya. Metode seleksi ROI yang digunakan adalah dengan piksel karena model jalur yang digunakan lebih memungkinkan jika menggunakan metode piksel.

B. Thresholding

Thresholding digunakan untuk mengatur *gray-level* yang ada pada gambar. Misalnya pada sebuah gambar, *f*(*x*,*y*) tersusun dari objek yang terang pada sebuah *background* yang gelap. *Gray-level* milik objek dan milik *background* terkumpul menjadi dua grup yang dominan. Salah satu cara untuk mengambil objek dari *background*-nya adalah dengan memilih sebuah nilai ambang (*threshold*) *T* yang memisahkan grup yang satu dengan grup yang lain. Maka, semua piksel yang memiliki nilai > *T* disebut titik objek, sedangkan yang lain disebut titik *background*. Proses ini disebut *thresholding*. Sebuah gambar yang telah dikenai proses *thresholding g*(*x*,*y*) dapat didefinisikan sebagai berikut [6].

$$g(x,y) = \begin{cases} 1 & jika\ f(x,y) > T \\ 0 & jika\ f(x,y) \le T \end{cases} \quad (1)$$

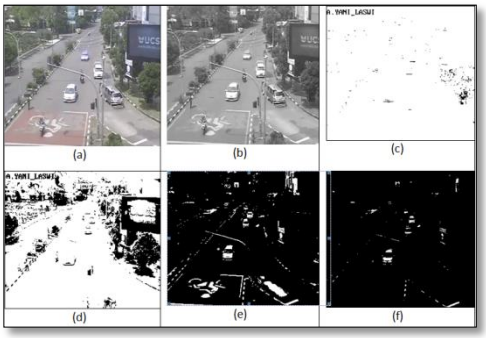
Nilai *T* dapat ditentukan melalui perhitungan rata–rata dari keseluruhan nilai karna yang ada pada gambar. Pada perhitungan ini, nilai *T* yang diperoleh tetap disimpan dalam bilangan real. Nilai *T* yang diperoleh untuk gambar yang memiliki histogram yang telah mengalami proses penyamaan (*equalization)* adalah berkisar antara 127 dan 128. Nilai maksimum *T* adalah nilai tertinggi dari sistem warna yang digunakan dan nilai minimum *T* adalah nilai terendah dari sistem warna yang digunakan. Untuk *256-graylevel* maka nilai tertinggi *T* adalah 255 dan nilai terendahnya adalah 0 [6]. Pengaruh nilai ambang ditunjukkan pada Gbr. 1.

C. Smoothing Filter

Smoothing, juga disebut *blurring*/pengaburan adalah operasi pengolahan citra sederhana dan sering digunakan. Ada banyak alasan dilakukannya *smoothing*. Pada dasarnya, *smoothing* digunakan untuk mengurangi derau. Untuk melakukan operasi *smoothing*, digunakan filter. Jenis filter yang paling umum adalah dari filter yang linear, dengan nilai keluaran piksel ($g\ (i, j)$) ditentukan sebagai jumlah bobot nilai masukan piksel (yaitu $f(i+k, j+l)$) [7].

$$g(i,j) = \sum_{k,l} f(i+k, j+l)h(k,l) \quad (2)$$

dengan $h(k,l)$ disebut kernel, yang tidak lebih dari koefisien filter. Ini membantu untuk memvisualisasikan filter sebagai jendela koefisien meluncur di gambar. Ada banyak jenis filter, di antaranya adalah Gaussian dan filter *median* [8], seperti yang ditunjukkan pada Gbr. 2.



Gbr. 1 Pengaruh nilai ambang, (a) Citra referensi, (b) Citra *gray-scale*, (c) Citra dengan TV=60. (d) Citra dengan TV=100, (e) Citra dengan TV=200 dan (f) Citra dengan TV=250.

Gbr. 2 (a) Sebelum *smoothing*, (b) Hasil *smoothing*

D. Image Morphology

Image morphology merupakan teknik atau proses yang digunakan untuk mengolah citra berdasarkan prinsip

morfologi matematika. Dalam pemrosesan citra, hasil yang diharapkan diperoleh berdasarkan bentuk atau struktur citra asal. Lebih lanjut dikatakan bahwa morfologi senantiasa berkaitan erat dengan proses ketetanggaan (*neighborhood*) yang terbentuk dari blok nilai biner, satu dan nol [9]. Lebih lanjut lagi, proses morfologi suatu citra merupakan kumpulan operasi nonlinear yang berkaitan dengan bentuk atau morfologi dalam suatu citra [10].

Hasil operasi morfologi dimanfaatkan untuk pengambilan keputusan dengan analisis lebih lanjut. Operasi ini antara lain meliputi pencarian batas/kontur, dilasi, erosi, penutupan (*closing*), pembukaan (*opening*), pengisian (*filling*), pelabelan, dan pengerangkaan (*skeletonization*) [11].

E. Dilasi

Dilasi adalah transformasi morfologi yang menggabungkan dua himpunan dengan menggunakan penjumlahan vektor elemen himpunan. Dilasi merupakan proses penggabungan titik-titik latar menjadi bagian dari objek, berdasarkan *structuring element* yang digunakan, dengan contoh seperti pada Gbr. 3. Proses ini adalah kebalikan dari erosi, yaitu mengubah latar di sekeliling objek menjadi bagian dari objek tersebut [12]. Ada dua cara untuk melakukan operasi ini, yaitu sebagai berikut.

1. Dengan mengubah semua titik latar yang bertetangga dengan titik batas menjadi titik objek.
2. Dengan mengubah semua titik di sekeliling titik batas menjadi titik objek.

Operasi morfologi dasar matematika dilasi dilakukan berdasarkan aljabar Minkowski, ditunjukkan oleh (3) [12].

$$D(A,B) = A \oplus B = \cup_{\beta \in B}(A + \beta) \quad (3)$$

F. Erosi

Erosi dapat diperoleh dengan melebarkan komplemen dari piksel hitam dan kemudian mengambil komplemen dari *set point* yang dihasilkan. Erosi merupakan proses penghapusan titik-titik batas objek menjadi bagian dari latar, berdasarkan *structuring element* yang digunakan, dengan contoh yang ditunjukkan pada Gbr.4 [12]. Pada operasi ini, ukuran objek diperkecil dengan mengikis sekeliling objek. Ada dua cara yang dapat dilakukan untuk melakukan proses erosi, yaitu sebagai berikut.

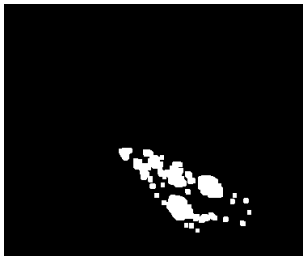
1. Dengan mengubah semua titik batas menjadi titik latar.
2. Dengan mengatur semua titik di sekeliling titik latar menjadi titik latar

Operasi morfologi dasar matematika erosi dilakukan berdasarkan aljabar Minkowski seperti pada (4) [12].

$$E(A,B) = A \ominus (-B) = \cup_{\beta \in B}(A - \beta) \quad (4)$$

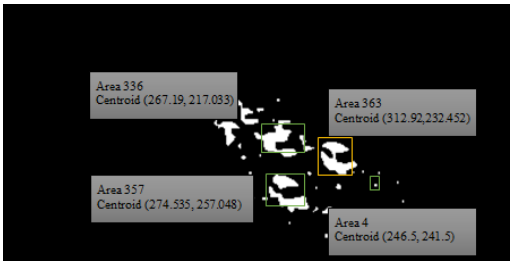
dengan $-B = \{-B | \beta \epsilon B\}$.

Himpunan A atau B dapat dianggap sebagai "citra". A biasanya dianggap sebagai citra dan B disebut *structuring element*.



Gbr. 3 Dilasi.

Gbr. 4 Erosi.



Gbr. 5 *Blob detection.*

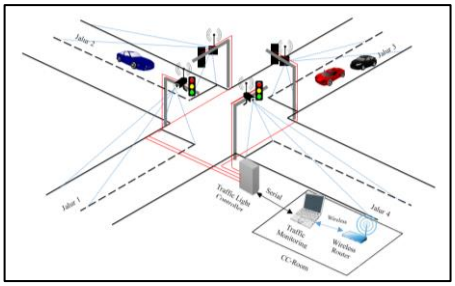
G. Blob Detection

Blob detection adalah algoritme yang digunakan untuk menentukan suatu grup dari piksel saling berhubungan satu sama lain atau tidak. Metode ini sangat berguna untuk mengidentifikasi objek yang terpisah-pisah pada suatu citra atau menghitung jumlah dari suatu objek pada suatu citra. Pada metode *blob detection*, citra harus diproses dengan metode ambang terlebih dahulu, berdasarkan warna yang akan dideteksi. Setelah itu, citra dengan warna di atas nilai ambang dapat dikategorikan sesuai dengan aturan yang telah ditentukan terlebih dahulu. Misalnya untuk piksel yang memiliki nilai lebih kecil daripada nilai ambang akan dikelompokkan sebagai komponen dari objek yang dideteksi, sedangkan yang nilainya di atas ambang dikelompokkan sebagai informasi yang tidak penting. Sedangkan untuk mendapatkan titik berat massa adalah dengan merata-ratakan lokasi tiap piksel dengan warna tertentu.

Bagi banyak pengolahan citra, mendeteksi objek *low-level* dalam sebuah gambar merupakan hal yang sangat penting. Objek dalam bentuk dua atau tiga dimensi tersebut biasa disebut *blob*. Bentuk *blob* timbul dalam cara yang berbeda, bergantung pada ukuran, dan dapat dideteksi dengan menggunakan metode sederhana dalam sebuah representasi gambar [13]. Gbr. 5 memperlihatkan *blob detection* dari video lalu lintas.

IV. PERANCANGAN SISTEM

Perancangan sistem pengendali lampu lalu lintas ini terdiri atas empat buah lampu lalu lintas, empat buah *IP Camera* nirkabel (tetapi pada penelitian ini digunakan video rekaman lalu lintas sebagai pengganti kamera), *Traffic Light Controller* menggunakan mikrokontroler [1], *Traffic monitoring* pada sebuah komputer sebagai pengolah citra video lalu lintas, antarmuka pengguna, dan *interfacing* dengan mikrokontroler. Rancangan keseluruhan pengendali lalu lintas ditunjukkan pada Gbr. 6.



Gbr. 6 Rancangan sistem keseluruhan pengendali lalu lintas.

Perancangan sistem dilakukan menggunakan OpenCV dengan bahasa pemrograman visual C++, yang ditampilkan dalam perangkat lunak *traffic monitoring*. Sistem dibuat dengan spesifikasi sebagai berikut.

1. Sistem deteksi kendaraan ini menangani jumlah kendaraan pada setiap lengan jalan.
2. Pada setiap lengan jalan terdapat sebuah video kamera yang berisi rekaman lalu lintas sebagai masukan system.

Sistem pengolahan citra dibagi menjadi tiga bagian utama, yaitu sebagai berikut.

1. *Capture* video banyaknya jumlah kendaraan ketika lampu menyala hijau dan ketika lampu menyala merah, dari masukan berupa rekaman lalu lintas.
2. Mengolah video sehingga dihasilkan keluaran berupa jumlah kendaraan.
3. Mengirim hasil jumlah kendaraan melalui komunikasi serial untuk diolah ke pengendali *fuzzy* selanjutnya, yang akan menjadi masukan dari sistem lalu lintas [1].

Dibutuhkan dua masukan untuk diolah, yaitu sebagai berikut.

1. Video dari masing-masing lengan jalan.
2. Daerah yang akan diamati dari video (ROI*).*

A. Perancangan Sistem Deteksi Kendaran

Deteksi kendaraan dirancang pada berbagai kondisi lingkungan dengan cahaya dan status lalu lintas berubah. Dalam sistem yang diusulkan, sistem menerima lalu lintas dari video dan mengonversi video menjadi *frame*, kemudian melakukan pengaturan ambang dan deteksi benda bergerak ataupun diam. Sistem ini terdiri atas empat tahap, sebagai berikut.

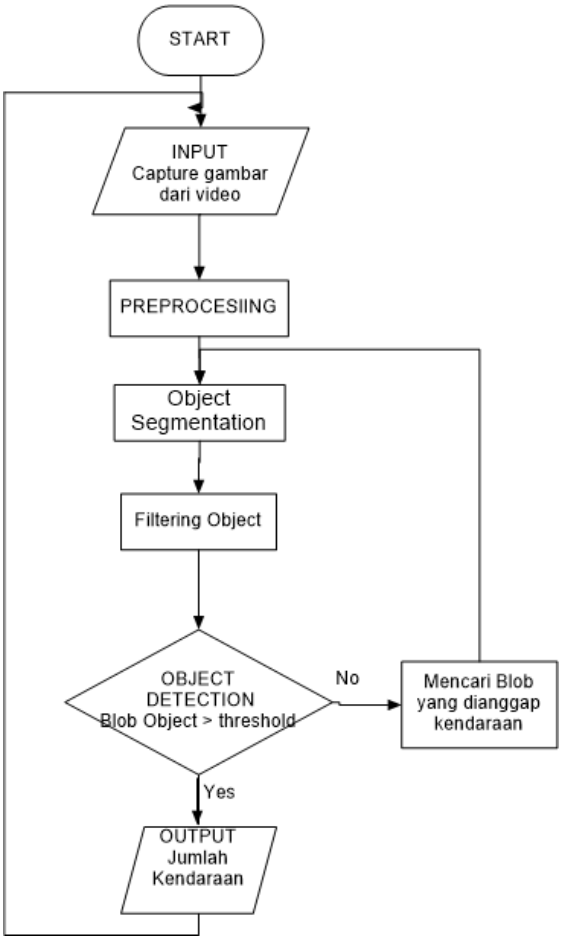
1) Sistem Pre-processing: Sistem ini bertujuan untuk mempersiapkan komponen-komponen yang akan digunakan seperti inisialisasi.

2) Segmentasi Citra: Proses ini memisahkan citra menjadi beberapa daerah atau objek. Tujuannya adalah untuk memisahkan suatu objek dengan objek lainnya dan/atau memisahkan objek dengan latar belakang. Teknik yang digunakan adalah *thresholding*.

3) Object Filtering: Terhadap objek dilakukan proses filter untuk mengurangi derau.

4) Deteksi Kendaraan dan Penghitungan: Pada tahap ini, *rendering blob* digunakan untuk menghitung jumlah kendaraan yang terdeteksi, baik kedaraan bergerak maupun diam.

Perancangan sistem perangkat lunak dapat dijelaskan melalui diagram alir seperti pada Gbr. 7.



Gbr. 7 Diagram alir perancangan sistem.

Penggunaan metode *blob detection* dilakukan dengan perintah ambang pada ROI yang telah ditentukan sebelumnya. Setelah itu akan diperoleh *blob* yang menunjukkan adanya objek pada citra tersebut. Dengan kata lain, nantinya *blob* yang diperoleh adalah warna putih. Penghitungan jumlah kendaraan dilakukan dengan melakukan filter berdasarkan ukuran *blob*. Untuk dapat mendeteksi tiap kendaraan, maka nilai ambang yang diujikan pada tiap video berbeda-beda, sesuai dengan intensitas pencahayaan video. Untuk pengujian proses penghitungan kendaraan, ditampilkan keluaran hasil perhitungan, sehingga dapat diketahui total *blob* pada masing-masing video. Filter berguna untuk menghilangkan derau dan menggabungkan piksel objek yang berdekatan satu sama lain.

Morfologi operasi dilasi dan erosi digunakan untuk kelompok tepi piksel ke dalam benda. Kemdian, tahapan akhir adalah *rendering blob* untuk melakukan boundaring deteksi kendaraan. *Rendering blob* ditunjukkan pada Gbr. 8.

Gbr. 8 *Rendering blob.*

B. Perancangan Perangkat Lunak Traffic Monitoring

Perangkat lunak *Traffic Monitoring* (antarmuka) ini dirancang menggunakan Microsoft Visual Studio. Antarmuka berguna untuk memungkinkan pengguna berinteraksi dengan sistem dan memberikan informasi jumlah deteksi kendaraan. Karena sistem berjalan pada dua mode yang berbeda, yaitu *traffic* video dan *fuzzy*, maka perlu diberikan pilihan untuk mengaktifkan *traffic*. Gbr. 9 menunjukkan antarmuka sistem yang menyediakan beberapa fungsi seperti

1. mengaktifkan ComPort untuk mengirimkan data serial,
2. mengaktifkan traffic video, dan
3. menampilkan jumlah Kendaraan terdeteksi.

Gbr. 9 Antarmuka pengguna.

V. IMPLEMENTASI DAN HASIL PENGUJIAN

A. Implementasi

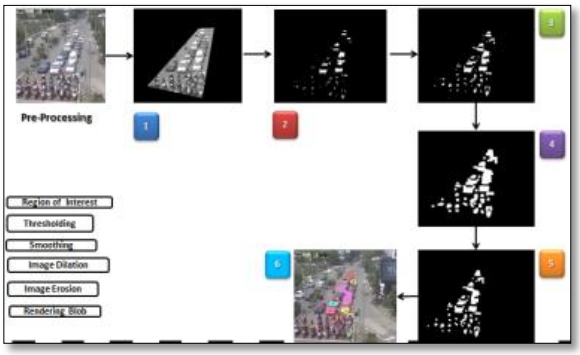
Sistem pemantau lalu lintas ini dibuat menggunakan teknologi *computer vision*. Perangkat lunak yang digunakan dalam pembuatan sistem adalah:

1. OpenCV versi 2.4.8, dan
2. Microsoft Visual Studio 2010.

Untuk mengetahui performa sistem ini, dibutuhkan pengujian dengan cara mengambil beberapa data. Pengujian dilakukan dengan menggunakan data berupa video yang berisi kondisi jalan dengan spesifikasi sebagai berikut.

1. Durasi tiap video 5 menit.
2. *Frame rate* sebesar 30 fps.
3. Ukuran video 480 x 320 piksel.

Fase pengolahan citra ditunjukkan pada Gbr. 10.



Gbr. 10 Fase pengolahan citra.

B. Hasil Pengujian

1) Tingkat Akurasi Terhadap Kondisi dan Kepadatan: Pengujian akurasi sistem dilakukan berdasarkan perbandingan perhitungan manual dengan perhitungan yang dilakukan oleh sistem. Penghitungan akurasi dilakukan dengan mencari galat (*error*) terlebih dahulu, yaitu perbandingan antara selisih jumlah kendaraan yang dideteksi sistem dan yang dilihat oleh mata manusia. Pada pembahasan ini, untuk menguji tingkat akurasi kebenaran system, dilakukan dua pengujian.

Pengujian pertama dilakukan di bawah empat kondisi lingkungan yang berbeda periode, dengan berbagai kondisi lingkungan, yaitu cerah, malam, hujan, dan subuh (berembun), dengan menggunakan video yang sama. Perhitungan ini menggunakan (5) dan (6) sebagai berikut [14].

$$Kesalahan\ Deteksi(\%) = \frac{|Selisih\ perhitungan\ simulasi\ \&\ pengukuran|}{jumlah\ kendaraan\ pengukuran} x100\% \quad (5)$$

$$Akurasi\ (\%) = 100\% - Kesalahan\ Deteksi\ (\%) \quad (6)$$

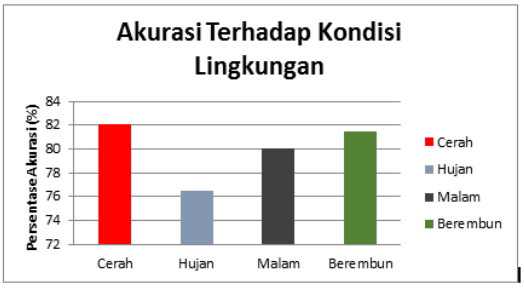
Data hasil pengujian disajikan pada Tabel I dan dalam bentuk grafik pada Gbr. 11.

TABEL I
PENGUJIAN AKURASI SISTEM TERHADAP KONDISI LINGKUNGAN

Kondisi	Jalan Laswi				
	Jumlah Kendaraan (simulasi)	Jumlah Kendaraan (pengukuran)	Selisih	Galat (%)	Akurasi (%)
Cerah	145	123	22	17,89	82,11
Hujan	52	68	16	23,50	76,50
Malam	96	120	24	20,00	80,00
Embun	63	54	9	16,67	83,33
Rata – rata				18,7	81,30

Dari nilai yang ditunjukkan pada Tabel I dan grafik pada Gbr. 11 dapat dilihat bahwa kondisi lingkungan berpengaruh terhadap akurasi sistem ini. Pada kondisi cerah, malam, dan berembuh (subuh), akurasi sistem terlihat tidak begitu berbeda.

Namun, saat kondisi hujan terlihat pengaruh perbedaan kondisi lingkungan terhadap akurasi system, sebesar 76,50 %. Hal ini dikarenakan saat kondisi hujan kamera pada perekam video mempengaruhi hasil perekaman.



Gbr. 11 Grafik pengujian akurasi sistem terhadap kondisi lingkungan.

TABEL II
PENGUJIAN AKURASI TERHADAP KONDISI KEPADATAN JALAN

Jalan	Kondisi Lengang			Kondisi Ramai		
	Simulasi	Pengukuran	Akurasi (%)	Simulasi	Pengukuran	Akurasi (%)
Laswi	2	2	100,0	20	27	74,07
	14	17	82,35	27	31	87,09
	15	12	75,00	24	47	51,06
	22	20	90,00	29	54	53,70
	17	25	68,00	45	62	72,58
	Rata-rata		83,07	Rata-rata		67,70

Pengujian kedua dilakukan dengan membandingkan kondisi kepadatan jalan terhadap akurasi pengukuran yang dilakukan oleh sistem. Pengujian dilakukan dengan memperhatikan kondisi jalan, ketika lengang dan ketika ramai, pada satu video. Data hasil pengujian disajikan pada Tabel II.

Dari hasil pengujian pada Tabel II terlihat bahwa pada kondisi lengang, metode *blob detection* bekerja lebih baik. Namun, keunggulan tersebut menurun ketika kondisi jalan semakin ramai/padat. Selisih rata-rata akurasi kondisi lengang dan ramai mencapai 15,37%.

2) Pengujian Evaluasi Performa: Pengujian ini bertujuan untuk mengevaluasi performa sensitivitas, kekhususan, presisi, *f-measure*, dan akurasi pada metode *blob detection* berdasarkan *predictive value theory*. Semua langkah ini dapat dihitung berdasarkan empat nilai, yaitu *True Positive* (TP), *False Positive* (FP), *False Negative* (FN), dan *True Negative* (TN) [10]. Nilai-nilai tersebut didefinisikan seperti dalam Tabel III.

TABEL III
KRITERIA EVALUASI

	Kategori	
Hasil Pengujian	Positif	Negatif
Positif (Kendaraan)	TP	FP
Negatif (Bukan kendaraan)	FN	TN

Dari jumlah ini, sensitivitas, presisi, kekhususan, *f-measure*, dan akurasi dihitung menggunakan (7) hingga (11) [15].

$$Sensitivitas = \frac{TP}{TP+FN} \quad (7)$$

$$Presisi = \frac{TP}{TP+FP} \quad (8)$$

$$Kekhususan = \frac{TN}{TP+FP} \quad (9)$$

$$F-Measure = \frac{2x\ Presisi\ x\ Sensitivitas}{Presisi\ +Sensitivitas} \quad (10)$$

$$Akurasi = \frac{TP+TN}{TP+FP+FN+TN} \quad (11)$$

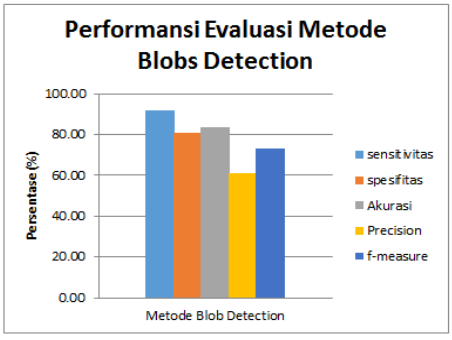
Sensitivitas, juga dikenal sebagai *recall*, adalah persentase dari semua kendaraan yang mendeteksi hasil positif. Kekhususan merupakan proporsi TN, yaitu persentase dari semua kendaraan yang mendeteksi hasil negatif. Presisi merupakan persentase dari jumlah kendaraan yang mendeteksi hasil positif dari total jumlah kendaraan yang ditemukan sistem. *F-Measure* merupakan evaluasi nilai kombinasi sensitivitas dan presisi, sedangkan akurasi adalah proporsi prediksi yang benar tanpa mempertimbangkan yang positif dan yang negatif, melainkan total keseluruhan.

Dari hasil percobaan dengan 48 sampel, diperoleh tingkatan probabilitas seperti disajikan pada Tabel IV.

TABEL IV
HASIL EKSPERIMEN

Jenis performa	Persentase (%)
Sensitivitas	91,67
Presisi	80,55
Kekhususan	83,33
F-Measure	61,11
Akurasi	73,33

Dari hasil perhitungan diketahui bahwa tingkat sensitivitas adalah sebesar 91,67%, presisi sebesar 61,11%, kekhususan sebesar 80,55%, *f-Measure* sebesar 73,33%, dan akurasi sebesar 83,33%. Maka nilai sensitivitas, presisi, kekhususan, *f-Measure*, dan akurasi perlu ditingkatkan lagi. Adapun hasil evaluasi performa digambarkan melalui grafik pada Gbr. 12.



Gbr. 12 Evaluasi performansi dengan metode *blob detection*.

Berdasarkan Gbr. 6 dapat dikatakan bahwa metode *blob detection* ini dapat digunakan sebagai metode pendeteksian kendaraan pada jalan raya. Namun, untuk letak kendaraan yang berdekatan pada citra, metode *blob detection* mendeteksi dua kendaraan yang berdekatan menjadi satu objek, sehingga akurasi nilai kepadatan berkurang. Hal ini dikarenakan metode *blob detection* mendeteksi suatu objek berdasarkan citra biner yang dihasilkan. Oleh karena itu, pada penelitian selanjutnya, perlu dilakukan pemisahan objek

kendaraan yang berdekatan, sehingga meningkatkan nilai akurasi pendeteksian.

Dari hasil eksperimen dapat disimpulkan bahwa tingkat sensitivitas metode yang digunakan adalah sebesar 91,67%, dengan tingkat presisi yang terbilang rendah, yaitu sebesar 61,11%. Namun, hal ini masih dikatakan efektif karena mencapai nilai di atas 50%. Tingkat kekhususan serta tingkat akurasi masih perlu ditingkatkan juga.

VI. KESIMPULAN

Metode *blob detection* dapat digunakan pada deteksi kendaraan dengan performa tingkat sensitivitas sebesar 91,67%, presisi sebesar 61,11%, kekhususan sebesar 80,55 %, *f-Measure* sebesar 73,33%, dan akurasi sebesar 83,33%. Walaupun nilai presisi lebih rendah dari pada nilai sensitivitas, metode yang digunakan sudah dapat dikatakan efektif karena nilianya di atas 50%.

Tingkat akurasi untuk deteksi kendaraan pada kondisi cerah adalah sebesar 82,11% dan berkurang sebesar 76,50% pada kondisi hujan. Sementara itu, deteksi kendaraan pada malam hari memberikan hasil tingkat akurasi sebesar 80,00%.

Tingkat akurasi penggunaan *blob detection* pada kondisi lengang lebih baik, yaitu sebesar 83,07%, dan berkurang sebesar 67,70% pada kondisi ramai. Kesalahan pada kondisi ramai lebih banyak disebabkan karena posisinya yang bergerombol dan berdekatan satu sama lain, sehingga sistem membaca banyak kendaraan tersebut sebagai satu kesatuan objek kendaraan dengan dimensi besar.

Pada manajemen lalu lintas, deteksi kendaraan merupakan langkah penting. Teknik berbasis *computer vision* lebih cocok digunakan karena sistem ini tidak mengganggu lalu lintas saat instalasi dan mudah untuk dimodifikasi.

Akurasi sistem sangat dipengaruhi oleh kondisi perekaman video, kondisi ruas jalan, dan pemilihan parameter pengaturan sistem yang tepat.

REFERENSI

[1] Suryatini Fitria, "Perancangan dan Implementasi Pengendali Lampu Lalu Lintas Berdasarkan Kepadatan Kendaraan Menggunakan Logika Fuzzy", Tesis, ITB, Bandung, 2015.

[2] Ahmed Bilal, "An Intelligent Traffic Controller Based on Fuzzy Logic", *International Journal of Innovation in the Digital Economy (IJIDE)*, **5** (1), pages 31-40, 2014.

[3] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, "A real-time computer vision system for measuring traffic parameters," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Puerto Rico, June 1997, pp. 496–501.

[4] K. P. Karmann and A. von Brandt, "Moving object recognition using an adaptive background memory", *Proc. Time-Varying Image Processing and Moving Object Recognition*, vol. 2, V. Capellini, Ed., 1990

[5] Rensso V. H. Mora Colque, and Guillermo, "Robust Model for Vehicle Type Identification in Video Traffic Surveillance", *SIBGRAPI-Conference Graphics, Patterns and Images*, Peru, 2013

[6] Gonzales, Rafael, C., *Digital Image Processing*, Addison-wesley publishing, 2, 760 – 783, 1992.

[7] Willey, John. Sons, *Digital Image Processing*, A Wiley-Interscience Publication, 3, 401 – 566, 2001.

[8] Willow Garage, Open Computer Vision, [Online]. Available: http:// http://docs.opencv.org/ OpenCV 2.4.11.0 documentation »

[9] Solomon, Chris. Toby, Breckon., *Fundamentals of Digital Image Processing*, Wileyblackwell Press, 1, 197-200, 2011.

[10] Intel, *Open Source Computer Vision Library*. U.S.A: Intel Corporation, 1999-2001.

[11] Ritter G.X, Wilson J. N, *Hanbook of Computer Vision Algorithms in Image Algebra*, CRC Press,Washington D.C, 2001

[12] Kaspers, Anne., "Blob Detection Biomedical Image Sciences", Image Sciences Institute, UMC Utrecht, 2011.

[13] Atkociounas, et al., "Image Processing in Road Traffic Analysis", *Nonlinear Analysis: Modelling and Control*, Vol. 10, No. 4, 315–332,2005.

[14] Sakamoto,Yoshihiro. Koichiro, Kajitani. Takeshi, Naito., "Development of The Image Processing Vehicle Detector For Intersection", *Proceedings of the 13th its world congreaa*, London. 2006

[15] Silva. R, Aires.K., Automatic Motorcycle Detection on Public Roads, *CLEI Electronic Journal*, 16 (3), Paper 04, 2013.