Pengolahan Citra Untuk Klasifikasi dan Perhitungan Jumlah Kendaraan

Siti Mu'arifah, *Awang Hendrianto Pratomo, and Wilis Kaswidjanti

Program Studi Teknik informatika, Jurusan Teknik Informatika, Fakultas Teknik Industri UPN "Veteran" Yogyakarta, Jl. Babarsari 2 Yogyakarta 55281

Email : sitimuarifah14@gmail.com, awang@upnyk.ac.id, wilis.kas@gmail.com

***Koresponden author**: Awang Hendrianto Pratomo. email : awang@upnyk.ac.id

Abstrak :

Perhitungan kendaraan di Indonesia saat ini masih menggunakan tenaga manusia sehingga rentan terjadi kesalahan dalam perhitungan jumlah kendaraan. Sehingga diperlukan metode lain yang dapat melakukan perhitungan secara otomatis dan lebih akurat. Salah satunya yaitu dengan mendeteksi kendaraan menggunakan kamera. Deteksi kendaraan menggunakan kamera merupakan topik yang menarik untuk dijadikan suatu penelitian dan dilakukan pengembangan. Masalah yang terjadi dalam penggunaan kamera untuk deteksi kendaraan adalah minimnya intensitas cahaya dimalam hari sehingga mempengaruhi hasil akurasi dan pengenalan objek. Gerakan objek yang terlalu cepat serta *background* yang sulit dideteksi sehingga menyebabkan bertambahnya waktu pemrosesan citra juga menjadi masalah yang tidak dapat dihiraukan. Tahapan/algoritma yang digunakan untuk melakukan deteksi pada penelitian ini memiliki pengaruh yang sangat penting dalam menentukan akurasi. Penelitian ini menggunakan fungsi *background substractor* untuk mengekstraksi objek dari *background*nya, berbagai *filter* sederhana yang digunakan untuk menghilangkan *noise* serta fungsi *contour* pada OpenCV dapat membantu melakukan optimasi pada proses pendeteksian. Perhitungan akurasi yang digunakan adalah menghitung selisih jumlah kendaraan yang terdeteksi oleh sistem dengan perhitungan manual. Hasil yang diperoleh adalah akurasi apada pagi hari sebesar 43,84%, akurasi pada siang hari sebesar 41,71%, pada sore hari sebesar 45,77% dan pada malam hari sebesar -6,95 %. Kemudian dilakukan perhitungan akurasi menggunakan analisis statistik *true false* sensitivity dan specificity yang menghasilkan akurasi pada pagi hari sebesar 97,53%, pada siang hari sebesar 91,43%, pada sore hari sebesar 97,40% dan pada malam hari sebesar 38,13%. Dengan demikian penelitian ini membuktikan bahwa proses pengolahan citra sederhana ditambah fungsi *Contouring* mampu melakukan pendeteksian objek terutama kendaraan dengan akurat apabila perhitungan akurasi yang digunakan juga sesuai.

Kata Kunci : Pengolahan Citra, tracking objek, deteksi dan kasifikasi kendaraan

I. Pendahuluan

Lonjakan volume kendaraan yang dialami oleh Indonesia cukup signifikan. Tingginya volume kendaraan menyebabkan masalah lalu lintas, salah satunya adalah kemacetan. Hal ini mengakibatkan pentingnya melakukan manajemen lalu lintas. Pendataan volume lalu lintas di Indonesia dilakukan secara manual dengan menggunakan tenaga manusia sehingga rentan terjadi kesalahan dalam perhitungan jumlah kendaraan dan membutuhkan waktu yang lama. Oleh karena itu diperlukan cara lain yang dapat melakukan perhitungan secara otomatis, cepat dan lebih akurat. Salah satunya yaitu dengan mendeteksi kendaraan menggunakan kamera (Hariyanto, 2015). Deteksi kendaraan menggunakan kamera merupakan topik yang menarik untuk dijadikan penelitian dan pengembangan (Adistya and Muslim, 2016; Alamsyah, 2006).

Kendaraan akan sulit terdeteksi pada berbagai kondisi sehingga mempengaruhi hasil dari perhitungan akurasi. Kesalahan yang terjadi ketika deteksi objek menggunakan kamera adalah kesalahan dalam klasifikasi sesuai dengan jenis kendaraannya, selain itu banyak sedikitnya *noise* yang akan sulit

membedakan objek dengan latar belakangnya (Rafsyam, 2017; Saputra, 2016). Sebuah objek akan mampu di deteksi apabila memiliki algoritma deteksi objek yang sesuai.

Algoritma deteksi yang digunakan memiliki pengaruh yang sangat penting dalam menentukan akurasi. Adapun tahapan yang dilakukan pada deteksi kendaraan yaitu *tracking, training* dan deteksi (Ismail, 2012). Proses deteksi dan *tracking* menggunakan algoritma K-Means memiliki akurasi yang cukup baik (Mushawwir and Supriana, 2015).

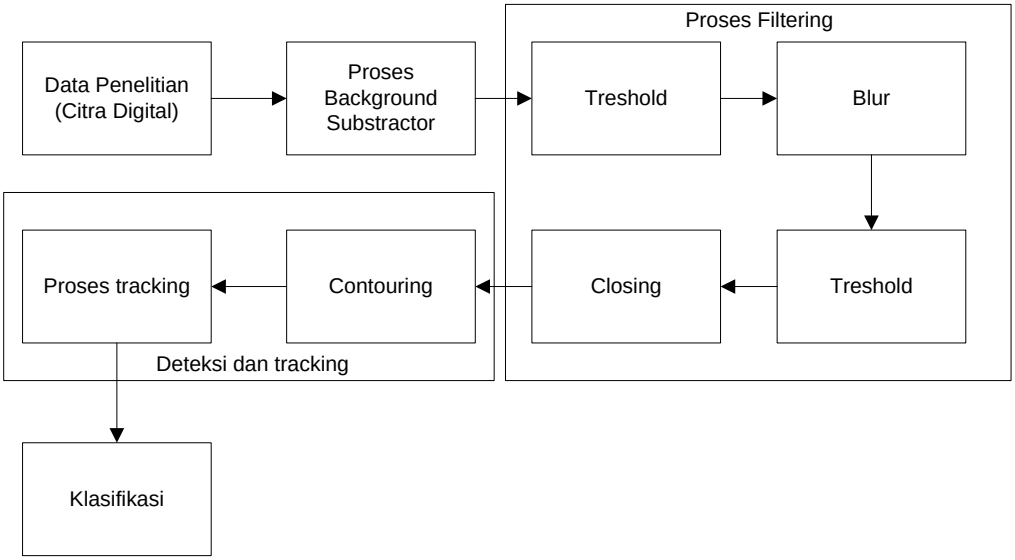
Wisnu Rizky Kurniawan (2015) menggunakan kamera web cam untuk mendeteksi objek. Tahap deteksi objek dimulai dari tahap preposisi, *background substraction*, menandai *Region Of Interest* (ROI), pelabelan, *tracking* dan klasifikasi. Algoritma yang digunakan masih memungkinkan mendeteksi objek lain dan algoritma *tracking* masih sangat sederhana tetapi memiliki akurasi yang cukup tinggi. Klasifikasi kendaraan hanya didasarkan pada jenis kendaraan motor dan mobil. Algoritma klasifikasi yang digunakan adalah menghitung lebar dan tinggi objek yang dideteksi.

Jin-Chuan Lai dkk (2010) mengatakan bahwa kendala yang dihadapi ketika melakukan deteksi kendaraan menggunakan kamera yaitu kendaraan tidak dapat terdeteksi pada suatu waktu karena terhalang kendaraan lain, kendaraan keluar dari jalur dan kendaraan yang lewat memiliki jarak yang sangat dekat satu dengan lainnya ataupun kendaraan terdeteksi melintas pada waktu yang bersamaan sehingga sulit untuk melakukan deteksi. Solusi yang ditawarkan yaitu dengan menghitung selisih waktu yang terjadi dan pemberian bobot menggunakan aspek rasio untuk menghitung dan mengklasifikasi kendaraan. Kinerja pada penelitian tersebut masih dipengaruhi oleh nilai *threshold* yang ditentukan.

Berdasarkan permasalahan yang telah disebutkan diatas, solusi yang ditawarkan adalah deteksi dan klasifikasi kendaraan menggunakan pengolahan citra. Proses klasifikasi meggunakan perbandingan ukuran data set dengan ukuran dari kendaraan yang telah terdeteksi.

II. Metodologi Penelitian

Metodologi penelitian yang digunakan pada penelitian ini ada metode kuantitatif dengan melakukan pengujian terhadap pengaruh suatu variabel terhadap variabel lain dalam penelitian (Wedianto et al., 2016). Adapun tahapan penelitian yang dilakukan ditunjukkan oleh gambar 1 :



Gambar 1. Tahapan Penelitian

A. Data Penelitian

Data penelitian berupa data video yang diambil secara *realtime*. Pengambilan data penelitian berlokasi di underpass jombor dengan pengaturan tertentu. Kamera di tempatkan pada ketinggian ± 8 - 10 meter diatas jalur underpass Jombor, menggunakan satu jalur lintasan untuk kendaraan yang sedang bergerak (tidak berhenti), diambil dalam kondisi cerah berupa data citra digital Citra digital harus melewati beberapa proses untuk kemudian dapat dikatakan suatu objek telah terdeteksi.

B. Pengolahan Citra

Pengolahan citra dapat digunakan untuk melakukan deteksi dan mengambil informasi dari suatu objek. Proses pengolahan citra yang terjadi antara lain proses *Background substractor* dilanjutkan dengan proses *filtering*.

1. *Background substractor*

Background substractor banyak digunakan dalam berbagai proyek berbasis pengolahan citra. Metode ini memisahkan objek (foreground) dengan latar belakangnya (*background*). Pada penelitian ini menggunakan *background substractor* MOG2. Setiap warna pixel pada metode ini memiliki nilai distribusi Gaussian, berbeda dengan *Background substractor* MOG yang menggunakan nilai k sebagai nilai distribusi. Hal tersebut membuat *Background substractor* MOG2 menjadi lebih adaptif dan fleksibel terhadap perubahan cahaya (Kurniawan, 2015). Pada *background substractor* MOG2 bayangan objek juga akan dapat dilihat dan berwanra abu-abu, tetapi hal ini melambatkan kecepatan pemrosesan. Berikut merupakan hasil dari aplikasi dari *background substractor* MOG2 (Opencvdoc, 2014) :

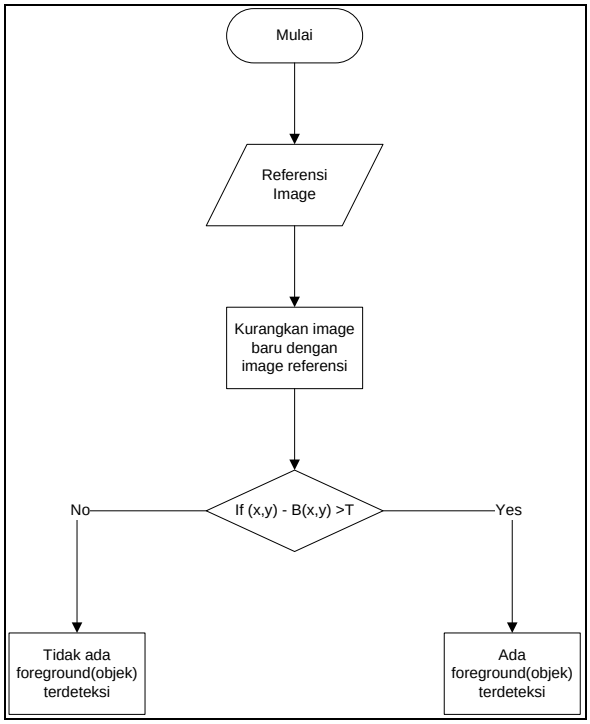
Gambar 2. Hasil *Frame Background substractor* MOG2 (Sumber:https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_video/py_bg_subtraction/py_bg_subtraction.html)

Miss Helly M Dessai dan Mr. Vaibhai Gandhi (2014) mengatakan bahwa skenario utama *background substractor* adalah jika selisih jarak antar *Frame* yang melebihi nilai ambang batas maka akan ditandai sebagai objek dan jika tidak maka akan diabaikan. Berikut persamaan untuk *background substraction* disebutkan pada persamaan 1 :

$$R_k(x,y) = f_k(x,y) - B(x,y)$$

$$D_k(x,y) = \begin{cases} 1 \; background & R_k(x,y) > T \\ 0 \; target & R_k(x,y) \le T \end{cases} \quad \text{..... (1)}$$

Flowchart background substraction menurut Miss Helly M Dessai dan Mr. Vaibhai Gandhi (2014) ditunjukkan pada gambar 3.



Gambar 3 *Flowchart* Proses *Background substraction*

Adapun alur dari *background substractor* adalah sebagai berikut :

a. Tentukan *image* yang digunakan sebagai referensi
b. Proses pengurangan nilai citra dari *image* baru dengan *image* referensi untuk menentukan selisih nilai citra pada *image* referensi dan *image* baru
c. Selisih dibandingkan dengan nilai *threshold* yang didapat dari beberapa *Frame* pertama. Jika selisih lebih besar daripada nilai *threshold* maka akan terdeteksi sebagai objek, tetapi jika selisih kurang atau sama dengan nilai *threshold* maka akan terdeteksi sebagai *background*.

Langkah- langkah yang dapat dilakukan untuk membuat *Background substraction* menjadi lebih adaptif adalah memeriksa setiap piksel baru terhadap komponen model yang sesuai dengan urutan. Komponen model yang cocok pertama akan diperbarui. Jika tidak ditemukan kecocokan, komponen Gaussian baru akan ditambahkan dengan mean pada titik tersebut dan matriks kovariansi besar dan nilai kecil parameter pembobotan.

Setiap piksel pada citra dimodelkan oleh campuran nilai distribusi K Gaussian. Kemungkinan bahwa piksel tertentu memiliki nilai xN pada saat N dapat ditulis sebagai berikut :

$$p(\mathbf{x}_N) = \sum_{j=1}^{K} w_j \eta(\mathbf{x}_N; \boldsymbol{\theta}_j) \quad \text{..... (2)}$$

Dimana wk adalah parameter berat komponen kth Gaussian. η(x; θk) adalah distribusi normal dari kth komponen yang diwakili oleh

$$\eta(\mathbf{x}; \boldsymbol{\theta}_k) = \eta(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{x}-\boldsymbol{\mu}_k)} \quad \text{..... (3)}$$

Dimana μk adalah rata-rata dan Σk = σ2k I adalah kovarian dari komponen kth. Distribusi K diurutkan berdasarkan kecocokan nilai wk/σk dan distribusi B pertama digunakan sebagai model latar belakang adegan di mana B diperkirakan sebagai

$$B = \arg\min_{b}\left(\sum_{j=1}^{b} w_j > T\right) \quad \text{..... (4)}$$

Threshold T adalah nilai minimum dari model latar belakang. Pengurangan latar belakang dilakukan dengan menandai setiap piksel foreground yang lebih dari 2,5 standar deviasi jauh dari salah satu distribusi B. Komponen Gaussian pertama yang cocok dengan nilai tes akan diperbarui oleh pembaruan persamaan berikut

$$\hat{w}_k^{N+1} = (1-\alpha)\hat{w}_k^N + \alpha\hat{p}(\omega_k \mid \mathbf{x}_{N+1})$$
$$\hat{\boldsymbol{\mu}}_k^{N+1} = (1-\alpha)\hat{\boldsymbol{\mu}}_k^N + \rho\mathbf{x}_{N+1}$$
$$\hat{\boldsymbol{\Sigma}}_k^{N+1} = (1-\alpha)\hat{\boldsymbol{\Sigma}}_k^N + \rho(\mathbf{x}_{N+1} - \hat{\boldsymbol{\mu}}_k^{N+1})(\mathbf{x}_{N+1} - \hat{\boldsymbol{\mu}}_k^{N+1})^T$$
$$\rho = \alpha\eta(\mathbf{x}_{N+1}; \hat{\boldsymbol{\mu}}_k^N, \hat{\boldsymbol{\Sigma}}_k^N)$$

$$\hat{p}(\omega_k \mid \mathbf{x}_{N+1}) = \begin{cases} 1 & ; \text{ jika } \omega_k \text{ merupakan nilai gaussian pertama yang cocok} \\ 0 & ; \text{ lainnya} \end{cases} \quad \text{..... (5)}$$

Dimana ωk adalah komponen kth Gaussian. 1/α merupakan konstanta waktu yang menentukan perubahan. Jika tidak ada distribusi K yang cocok dengan nilai piksel tersebut, komponen yang paling mungkin digantikan oleh distribusi dengan nilai saat ini sebagai rata-rata, varian awalnya tinggi, dan bobot parameter yang rendah (Kaewtrakulpong, 2001). Pada penelitian ini menggunakan *Background substractor* MOG2 karena hasil ekstraksi pada *Background substractor* MOG2 masih mengandung bayangan sehingga ukuran objek yang terdeteksi tidak jauh berbeda dari ukuran sebenarnya karena tidak terlalu banyak terjadi pengurangan atau penambahan selama dilakukannya proses ekstraksi.

2. *Filtering*

Setelah proses *Background substraction* dilakukan maka tahap selanjutnya adalah proses *filtering*. *Filtering* merupakan suatu proses untuk mengambil atau membuang frekuensi tertentu dari suatu citra (Mutiara, 2005). Beberapa proses yang terdapat di dalam proses *filtering* yaitu *thresholding*, *Bluring* dan *Closing*.

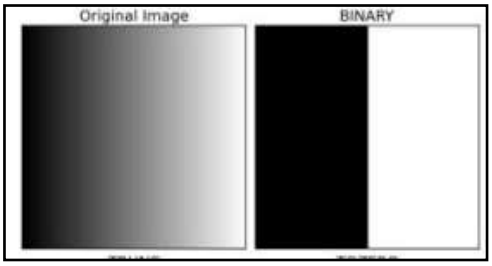
a. Threshold

Thresholding adalah proses mengubah citra yang berderajat keabuan menjadi citra biner atau hitam putih sehingga dapat diketahui antara foreground dan *background*, memisahkan nilai pixels sesuai dengan ambang batas yang telah ditentukan (Opencvdoc, 2014). Persamaan untuk *thresholding* dapat dituliskan:

$$f(x,y) \begin{cases} \text{Nilai Maks} & ; \text{Jika } f(x,y) > T \\ 0 & ; \text{Lainnya} \end{cases} \quad \text{..... (6)}$$

Berikut merupakan tahapan-tahapan yang digunakan untuk *thresholding image* :

a. Baca *image* dalam bentuk *grayscale*
b. Tentukan nilai ambang batas (nilai *threshold*) dan nilai maksimal
c. Bandingkan nilai setiap pixel pada citra dengan nilai *threshold*, jika nilai kurang atau sama dengan nilai *threshold* maka akan diubah menjadi 0 dan jika lebih dari nilai *threshold* maka akan diubah menjadi 255 (nilai maksimum citra). *Thresholding* citra ditunjukkan pada gambar 4.



Gambar 4. *Image Thresholding*

Thresholding pada kendaraan dapat dilihat pada gambar 5.

Gambar 5. Gambar *Thresholding* Kendaraan

Gambar 5 adalah gambar *thresholding* 1 sebelum dilakukan proses *Bluring*, masih terdapat banyak *noise-noise* kecil sedangkan gambar 6 adalah gambar *thresholding* setelah dilakukan proses *Bluring*.

Gambar 6. Gambar *Thresholding* Setelah Proses *Bluring*

Pada penelitian ini *thresholding* dilakukan untuk mempertegas batas dari setiap objek serta menghilangkan *noise* yang memiliki intensitas citra keabuan yang ditunjukkan seperti gambar 6.

b. *Blur*

Bluring merupakan *filter spasial-low* yang melenyapkan detail halus dari suatu citra (Murniati 2015). Menurut Hong, salah satu metode *Bluring* yang sering digunakan adalah Gaussian *Blur* (Hong, 2016). Gaussian *Blur* adalah metode yang menggunakan fungsi Gaussian untuk memperhalus *noise* pada citra (Imam et al. 2016). Penggunaan metode Gaussian sendiri lebih dapat mengurangi *noise* yang terdapat di dalam citra (Afifa 2016).

Pada penelitian ini, citra akan dijadikan lebih *smooth* menggunakan *Bluring*. Proses *Bluring* sendiri bertujuan untuk mengubah intensitas *noise* citra menjadi keabuan sehingga ketika dilakukan proses *thresholding* kembali maka *noise* citra akan lebih berkurang. Proses *Bluring* yang dilakukan pada penelitian ini menggunakan fungsi Gaussian *Blur* pada OpenCV. Dengan demikian, distribusi Gaussian dapat dilihat pada persamaan 7 dan 8(Chernenko, 2016):

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \text{..... (7)}$$

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{y^2}{2\sigma^2}} = G(x)G(y) \quad \text{..... (8)}$$

Berikut merupakan algoritma Gaussian *Blur* (Chernenko, 2016) dan *Bluring* pada citra kendaraan dapat dilihat pada gambar 7:

a. Hitung setiap 1D *Frame* dengan nilai dari G`n
b. *Filter* setiap garis gambar sebagai sinyal 1D
c. *Filter* setiap kolom yang di *Filter* sebagai sinyal 1D

Gambar 7. Gambar *Bluring*

Gambar 7 merupakan hasil dari mengaplikasikan fungsi *Bluring* terhadap citra setelah selesai dilakukannya proses *thresholding* yang pertama. Warna *noise* berubah menjadi keabuan.

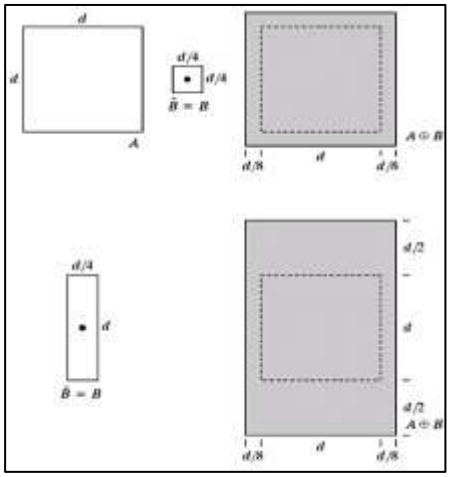
c. *Closing*

Morfologi *Closing* sendiri merupakan kebalikan dari morfologi opening. Pada morfologi *Closing* dilakukan dilasi terlebih dahulu kemudian diikuti dengan erosi. Morfologi *Closing* sendiri digunakan denga tujuan mengisi lubang kecil pada objek dan menggabungkan objek yang berdekatan. Dilasi merupakan memperbesar citra biner dengan menambah lapisan yang berada di sekeliling objek sedangkan erosi merupakan kebalikan dari dilasi yaitu mengurangi/mengikis tepi objek (Yulio, 2017).

Proses dilasi merupakan suatu penambahan piksel kedalam citra biner, proses dilasi sangat berguna untuk menggabungkan objek-objek yang terputus karena *noise*. Dilasi dapat dinyatakan A $\oplus$ B dimana

dilasi merupakan anggota dari Z^2, dilasi dapat dinyatakan dalam notasi 9 dan contoh dari proses dilasi bisa dilihat pada gambar 8 (Amin, 2014).

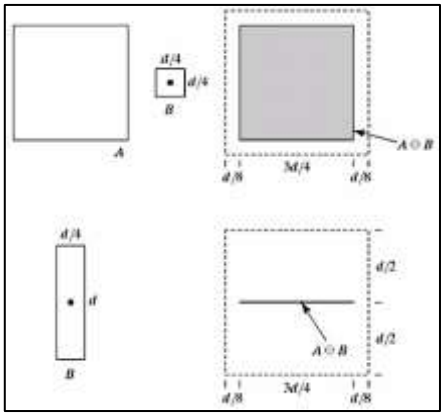
$$A \oplus B = \{z \mid [(\hat{B}_z) \cap A] \subseteq A\} \quad \text{.....(9)}$$



Gambar 8. Contoh Proses Dilasi.

Proses erosi merupakan suatu proses pengolahan citra yang digunakan untuk mengurangi tepi dari citra biner sehingga proses erosi dapat mengecilkan citra yang diproses (Yagi, 2012). Erosi dapat dinyatakan dalam notasi 10 dan contoh dari proses erosi dapat dilihat pada gambar 9 (Namboodiri, 2003).

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \quad \text{.....(10)}$$



Gambar 9. Contoh Proses Erosi.

Definisi dari *Closing* dapat dilambangkan sebagai berikut, *Closing* dari A terhadap B dapat dilambangkan sebagai A • B, dan dapat juga di definisi pada notasi 11 dan proses *Closing* dapat dilihat pada gambar 10 (Dougherty and Lotufo, 2003).

$$A \bullet B = (A \oplus B) \ominus B. \quad \text{.....(11)}$$

Gambar 10. Hasil *Closing*.

C. Deteksi dan *Tracking*

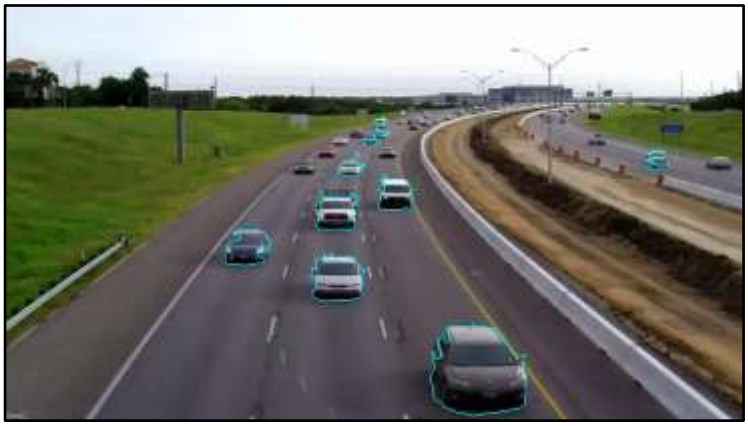
Tahap selanjutnya adalah proses deteksi dan *tracking*. Proses deteksi menggunakan fungsi *Contouring* yang terdapat pada OpenCv.

a. *Contouring*

Contour merupakan pengenalan kontur pada citra, *contour* dapat mendeteksi bentuk suatu objek dengan cara menggabungkan titik-titik yang terbentuk dari setiap lekuk pada suatu kontur. *Contour* sendiri merupakan fungsi yang sudah disediakan pada Library OpenCV, fungsi ini dapat digunakan untuk menganalisa bentuk objek, mendeteksi dan mengenali suatu objek. Pada gambar 10 dapat dilihat proses pendeteksian suatu objek (OpenCv, 2017).

Syarat yang harus dipenuhi agar dapat menjalankan fungsi *Contouring* sebagai berikut :

i. Gunakan citra biner, dengan artian citra harus di rubah kedalam *grayscale* kemudian menggunakan *threshold* untuk memperjelas objek.
ii. Objek yang akan diteksi harus berwarna putih dengan *background* hitam, karena fungsi *contour* hanya mendeteksi warna putih sebagai objek.



Gambar 10. Proses pendeteksian dengan *Contour*

b. *Tracking*

Tracking atau pelacakan merupakan suatu teknik untuk menemukan objek yang sama sehingga tidak kehilangan objek yang akan di dedeteksi, dalam hal ini *tracking* berguna untuk mengetahui arah dari pergerakan objek (Evelyn, 2013).

Pada bagian *tracking* menggunakan titik tengah objek sehingga bisa tersimpan dalam *array*, penggunaan fungsi yang ada pada OpenCV yaitu, fungsi *convexHull* untuk memperbaiki bentuk dengan mengabaikan cekungan pada tepian objek, bisa dilihat pada gambar 11. dan *boundingRect* untuk

memberikan bingkai berbentuk persegi agar dapat mengetahui berapa lebar maupun panjang objek (OpenCV, 2015).

Gambar 11. Hasil Akhir Implementasi *Convexhull*.

Gambar 12. Hasil Dari *Tracking*

Gambar 13. Garis Merah Merupakan Hasil Dari *Boundingrect*.

Untuk menentukan titik tengah dari objek dapat menggunakan hasil kembalian dari fungsi *boundingRect* yaitu berupa titik koordinat (x,y) persegi tersebut, dimana titik tersebut berada pada posisi ujung kiri atas, kemudian ada (lebar, tinggi) dari persegi tersebut. Titik tengah suatu objek dapat ditentukan dengan perhitungan 12 (OpenCV, 2015).

$$center = \left(\left(x + \left(\frac{lebar}{2}\right)\right).\left(y + \left(\frac{tinggi}{2}\right)\right)\right) \quad \text{.....(12)}$$

Pada gambar 11 terlihat convexhull sangat berguna untuk membentuk suatu objek yang dimana tepinya terdapat cekungan, pada gambar 12 menampilkan titik tengah dari objek. Setelah mendapatkan titik tengah objek, kemudian titik tengah tersebut disimpan dalam *grayscale*, sehingga didapatkan arah jalan suatu objek, dari *grayscale* titik tengah tersebut dapat menjadi referensi untuk mengetahui arap jalan kenadaraan apakah ke atas atau ke bawah.

D. Proses Klasifikasi

Proses yang terjadi setelah proses deteksi dan *tracking* adalah proses klasifikasi. Proses klasifikasi sendiri terjadi di server. Adapun tahapan proses klasifikasi adalah sebagai berikut :

1. Hitung meter per piksel (m/px) menggunakan lebar piksel jalan dan lebar jalan sebenarnya. Perhitungan meter per piksel digunakan untuk mengetahui ukuran sebenarnya setiap satu piksel mewakili berapa meter ukuran sebenarnya dengan rumus:

   Mperpiksel = lebar piksel/lebar sebenarnya ..... (13)

2. Hitung ukuran panjang dan lebar kendaraan sebenarnya berdasarkan piksel kendaraan yang telah diketahui dikalikan dengan mPerpiksel yang telah diketahui menggunakan rumus:

   Lebar Sebenarnya = lebar piksel x mperpiksel ..... (14)

3. Hitung selisih panjang dan lebar sebenarnya (diambil dari data set) dengan panjang dan lebar sebenarnya kendaraan yang terdeteksi
4. Jumlahkan hasil dari selisih perhitungan panjang pada no.5
5. Membandingkan jumlah selisih yang mempunyai nilai yang paling mendekati dengan dataset kendaraan untuk menentukan kendaraan tersebut berada pada golongan tertentu.
6. Mengulangi langkah ke 3 - ke 6 sampai semua objek terklasifikasi.

III. Hasil dan Pembahasan

Hasil dari penelitian ini ditunjukkan pada tabel 1 berikut

.

Tabel 1. Hasil Penelitian

waktu percobaan	%eror	%akurasi
pagi	56,16%	43,84%
siang	58,29%	41,71%
sore	54,23%	45,77%
malam	106,95%	-6,95%

Tabel 1 merupakan hasil rata-rata pengujian menggunakan 20 video pada berbagai kondisi yaitu pagi, siang, sore dan malam. Pada kondisi pagi memiliki rata-rata akurasi sebesar 43,84%, pada kondisi Siang memiliki akurasi sebesar 41,71%, pada kondisi sore memiliki akurasi sebesar 45,77% dan pada kondisi malam memiliki akurasi sebesar -6,95%. Perhitungan akurasi yang dilakukan untuk mendapatkan hasil tersebut menggunakan rumus:

Akurasi = 100% - %eror ..... (13)

Sedangkan persentase eror di dapatkan menggunakan rumus:

%eror = (|selisih jumlah manual dengan sistem|/jumlah manual)*100%..... (14)

Setiap kondisi memiliki akurasi yang sangat rendah karena berada dibawah 50% hal ini dikarenakan persentase akurasi di dapatkan dari rata-rata seluruh video pada setiap kondisi sementara setiap video memiliki *noise* yang berbeda-beda dan tingkat akurasi yang berbeda pula. Ada yang memiliki tingkat akurasi yang tinggi yaitu mencapai 100% dan ada yang minus. Apabila dilakukan rata-rata akurasi menggunakan rumus yang ada maka akan mempengaruhi akurasi yang sudah mencapai 100% tersebut.

Pada kondisi malam hari memiliki akurasi minus dikarenakan pendeteksian yang dilakukan oleh sistem melebihi jumlah objek yang terdeteksi yang dilakukan secara manual. Pada kasus ini kesalahan

bukan terdapat pada perhitungan yang digunakan melainkan karena sistem tidak mampu mendeteksi kendaraan pada kondisi minim cahaya sehingga menghasilkan banyak *noise* yang dianggap sebagai objek baru yang akhirnya terdeteksi.

IV. Kesimpulan

Berdasarkan hasil pengujian untuk deteksi kendaraan menggunakan tahapan/algoritma yang digunakan memiliki akurasi yang cukup baik apabila menggunakan perhitungan akurasi dengan metode analisis statistik *true false* dan memiliki hasil yang kurang bagus apabila menggunakan perbandingan jumlah objek yang terdeteksi dengan jumlah objek yang dihitung secara manual. Akurasi tertinggi pada pengujian I sebesar 45,77% pada kondisi sore hari dan pada pengujian II memiliki akurasi tertinggi sebesar 97,53% pada kondisi pagi hari. Akurasi berdasarkan hasil penelitian membuktikan bahwa tingkat kesalahan deteksi dan klasifikasi relatif kecil pada berbagai kondisi yaitu pagi, siang, sore dan malam dengan rata-rata *error* sebesar 18.26%. Dengan demikian pengolahan citra dengan tahapan/algoritma tersebut dapat digunakan untuk mendeteksi objek dengan baik dalam berbagai kondisi yaitu pagi,siang, sore dan malam. Dari hasil penelitian masih ditemukan berbagai permasalahan sebagai berikut :

1. Waktu pemrosesan masih terlalu lama.
2. Pada keadaan pencahayaan yang kurang atau pada malam hari akurasi deteksi masih rendah.

V. Daftar Pustaka

Adistya, Rama, and M Aziz Muslim. 2016. "Deteksi Dan Klasifikasi Kendaraan Menggunakan Algoritma Backpropagation Dan Sobel." *Journal of Mechanical Engineering and Mechatronics* 1 (2): 65–73.

Afifa, Zuliatul. 2016. "Implementasi Metode Gaussian Filter Untuk Penghapusan Noise Pada Citra Menggunakan GPU."

Alamsyah, Derry. 2006. "1 Universitas Indonesia," 1–21.

BIML. 2017. "Test Statistics," 1–9. http://groups.bme.gatech.edu/groups/biml/resources/useful_documents/Test_Statistics.pdf.

Hariyanto, Zamroji, and Jurusan Matematika. 2015. "Klasifikasi Jenis Kendaraan Bergerak Berbasis."

Imam, Jl, and Bonjol No. 2016. "METODE GAUSSIAN SMOOTHING UNTUK PENINGKATAN KUALITAS," no. 207: 1–6.

Ismail, Ari Taufiq. 2012. "Pengembangan Sistem Identifikasi Objek Bergerak Dengan Kamera Aktif," no. 2.

Kaewtrakulpong, P, and R Bowden. 2001. "An Improved Adaptive Background Mixture Model for Real-Time Tracking with Shadow Detection 2 Background Modelling." *In Proc. 2nd European Workshop on Advanced Video Based Surveillance Systems, Computer Vision and Distributed Processing, Kluwer Academic Publishers* AVBS01: 1–5. https://doi.org/10.1.1.12.3705.

Kurniawan, Wisnu Rizky. 2015. *PURWARUPA SISTEM KLASIFIKASI DAN PENGHITUNG JUMLAH KENDARAAN BERMOTORMENGGUNAKAN KAMERAWEBCAM BERBASIS CITRA DIGITAL Di PT. Industri Telekomunikasi Indonesia (Persero)*.

Lai, Jin Cyuan, Shih Shinh Huang, and Chien Cheng Tseng. 2010. "Image-Based Vehicle Tracking and Classification on the Highway." *1st International Conference on Green Circuits and Systems, ICGCS 2010*, 666–70. https://doi.org/10.1109/ICGCS.2010.5542980.

Murniati. 2015. "Membuat_efek_Blur_invers_crop_dan_Rotate."

Mushawwir, Luqman Abdul, and iping Supriana. 2015. "Deteksi Dan Tracking Objek Untuk Sistem Pengawasan Citra Bergerak." *Konferensi Nasional Informatika (KNIF) 2015 Deteksi* 2354–645X/ (October): 1–10.

Prof. Dr.rer.nat. Achmad Benny Mutiara, SSi, Skom. 2005. "Pengantar Pengolahan Citra." *Universitas Gunadarma*, 1–10. http://amutiara.staff.gunadarma.ac.id/Downloads/folder/0.58.

Rafsyam, Yeniwarti. 2017. "Analisis Pelacakan Objek Menggunakan Background Estimation Pada Kamera Diam Dan Bergerak" 13 (2): 124–30.

Saputra, Ari Kurniawan. 2016. "Aplikasi Deteksi Objek Menggunakan Histogram Of Oriented Gradient Untuk Modul Sistem Cerdas Pada Robot Nao." *ResearchGate*, no. January: 70. https://doi.org/10.13140/RG.2.1.3592.9207.

Wedianto, Andre, Herlina Latipa Sari, and Yanolanda Suzantri H. 2016. "Analisa Perbandingan Metode Filter Gaussian , Mean Dan Median Terhadap Reduksi Noise." *Jurnal Media Infotama* 12 (1): 21–30.