Sistem Perhitungan Jumlah Kendaraan Berbasis Computer Vision

Andi Asvin Mahersatillah Suradi*[1], Muhammad Furqan Rasyid[2], Nasaruddin[3]
[1,2]Manajemen Informatika, [3]Sistem Informasi
Universitas Dipa Makassar
Jalan Perintis Kemerdekaan Km. 9 Makassar, Telp. (0411) 587194 – Fax. (0411) 588284
e-mail: *[1]andiasvin@undipa.ac.id, [2]muhammad.furqan@undipa.ac.id, [3]nhas@undipa.ac.id

Abstrak

Tujuan dari penelitian ini yaitu memmbangun sebuah sistem yang dapat melakukan proses perhitungan kendaraan yang melaju pada suatu ruas jalan secara otomatis yang dapat digunakan secara real time, sehingga dengan adanya sistem ini dapat menghasilkan suatu informasi mengenai volume kendaraan pada suatu ruas jalan yang nantinya dapat menjadi sebuah data dalam pengambilan keputusan terkait masalah kemacetan yang terjadi. Adapun metode atau algoritma yang digunakan pada penelitian ini yaitu K-Nearest Neighbor dengan bantuan pustaka OpenCV, dengan melakukan pemisahan antara objek dalam hal ini kendaraan dengan backgroundnya yaitu ruas jalan. Berdasarkan hasil penelitian didapatkan bahwa sistem mampu mendeteksi kendaraan yang melaju pada ruas jalan sekaligus melakukan perhitungan dengan tingkat akurasi 78,5% pada tiga ruas jalan dan FPS sebesar 29.

Kata kunci—Pengolahan Citra, Perhitungan Kendaraan, Visi Komputer

Abstract

The purpose of this research is to build a system that can perform the calculation process of vehicles driving on a road section automatically which can be used in real-time so that this system can produce information about the volume of vehicles on a road section which can later become a data in decision making related to congestion problems that occur. The method or algorithm used in this study is K-Nearest Neighbor with the help of the OpenCV library, by separating vehicle objects from the background, namely roads. Based on the results of the study, it was found that the system was able to detect vehicles driving on one road section and at the same time perform calculations with an accuracy rate of 78.5% on three roads and 29 FPS.

Keywords—Image Processing, Vehicle Counting, Computer Vision

1. Pendahuluan

Dewasa ini peningkatan volume kendaraan atau alat transportasi dari tahun ke tahun semakin pesat khususnya di daerah perkotaan, dengan demikian tingkat kepadatan di jalan raya juga tentu semakin bertambah, sehingga dengan situasi panjang dan lebar jalan yang cenderung masih sama dan meningkatnya jumlah kendaraan, maka akan terjadi penumpukan kendaraan pada ruas jalan. Berdasarkan data Badan Pusat Statistik, peningkatan jumlah kendaraan pada tahun 2020 sebanyak 136.137.451 unit [1]. Untuk menghindari penumpukan jumlah kendaraan di lokasi tertentu dibutuhkan suatu informasi statistik tentang pertumbuhan kendaraan di daerah tersebut setiap tahunnya [sitasi]. Perkembangan kendaraan saat ini bisa dilihat dari seberapa banyak kendaraan yang melewati jalur tertentu. Dengan mengenali perkembangan kendaraan di suatu kota dapat dijadikan sebagai parameter butuh tidaknya penambahan sarana jalan raya ataupun jalur alternatif.

Tujuan dari penelitian ini yaitu untuk melakukan perhitungan jumlah kendaraan yang melintas pada suatu ruas jalan secara otomatis yang kemudian menjadi suatu data yang

digunakan untuk mengambil sebuah keputusan terkait masalah kemacetan yang terjadi pada ruas jalan tersebut. Berikut beberapa riset yang telah dikerjakan yang memiliki kesamaan dengan penelitian ini dalam hal menghitung jumlah kendaraan.

Penelitian yang dilakukan oleh Mawaddah Aynurrohmah dan Andi Suyanto yaitu sebuah perangkat lunak yang dapat menghitung jumlah mobil pada suatu ruas jalan baik yang searah maupun yang berlawanan menggunakan *webcam*. Adapun metode yang digunakan yaitu Ekualisasi Histogram, Pengambangan (*Thresholding*) untuk mengubah gambar dalam bentuk biner. Hasil perhitungan kendaraan dapat dilakukan pada jalan tertentu yang tidak memiliki banyak variasi kendaraan, seperti jalan tol [2].

Riset yang dikerjakan oleh Dufan J.P. Manajang dkk yaitu sistem perhitungan dan klasifikasi jenis kendaraan bermotor, dimana metode yang digunakan yaitu *framework Tensorflow Object Detection* yang merupakan *framework deep learning* dan juga salah satu *library* untuk *data science* yang bersifat *open-source* yang dikembangkan oleh para peneliti dari tim Google. Hasil yang diperoleh dalam menghitung kendaraan rata-rata 90,8% [3].

Penelitian yang telah dikerjakan oleh Karthik dan Kamalraj yaitu sistem perhitungan kendaraan menggunakan teknik dari *computer vision* diantaranya *thresholding* dan *the adaptive morphology*. Akurasi yang didapatkan dalam perhitungan kendaraan sebesar 96% [4].

Perbedaan penelitian ini dengan beberapa penelitian sebelumnya terletak dari pemilihan metode yang digunakan yaitu *K-Nearest Neighbor* serta implementasinya yang dapat digunakan secara *real time*.

2. Metode Penelitian

Pada penelitian ini terdapat beberapa tahapan diantaranya studi literatur mengenai sistem perhitungan jumlah kendaraan beserta teori atau teknik yang digunakan, pengumpulan data, akuisisi data, perancangan sistem, pengujian sistem, rekapitulasi hasil pengujian dan yang terakhir pembuatan laporan dalam bentuk *paper.* Adapun jenis penelitian yang digunakan yaitu penelitian eksperimental yang bersifat analisis dengan penelusuran sumber-sumber tertulis (*library research*), dipadukan dengan pengumpulan data-data faktual di lapangan dan data tersebut diolah kemudian dikembangkan menjadi sistem yang dapat menghitung secara otomatis kendaraan pada suatu ruas jalan.

2.1 Studi Literatur

2.1.1 Computer Vision

Computer vision adalah metode menggunakan komputer untuk memperoleh informasi tingkat tinggi dari gambar digital. Sebuah sistem *computer vision* mengambil gambar dan memasukkannya ke dalam komputer yang akan dieksekusi menggunakan pengolahan citra dan pengenalan pola. Algoritma *computer vision* kemudian melakukan untuk mencapai tujuan tertentu [5].

2.1.2 Image Processing

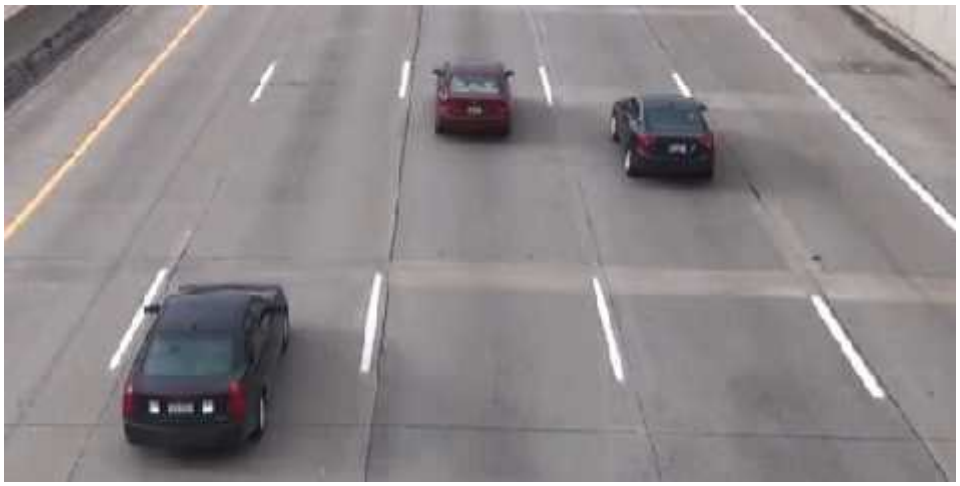
Image processing atau pengolahan citra bertujuan untuk meningkatkan kualitas citra agar mudah diinterpretasikan oleh manusia dan mesin (dalam hal ini komputer). Teknologi pemrosesan gambar mengubah suatu gambar menjadi gambar lain. Oleh karena itu, inputnya adalah gambar dan outputnya juga gambar, tetapi gambar output memiliki kualitas yang lebih tinggi daripada gambar input [6].

2.1.3 Algoritma K-Nearest Neighbor

Algoritma *K-Nearest Neighbor* (KNN) merupakan suatu prosedur dalam menerapkan klasifikasi terhadap objek baru yang bersumber pada (K) tetangga terdekatnya. KNN merupakan algoritma *supervised learning*, dimana hasil dari *query instance* yang baru, diklasifikasikan yang berdasarkan pada kebanyakan dari jenis pada KNN. Kelas yang sangat banyak yang merupakan kelas hasil klasifikasi [7].

2.2 Pengumpulan Data

Tahapan pengumpulan data terdiri dari pengumpulan data primer dan data sekunder. Data primer merupakan data berupa gambar dan video lalu lintas yang bersumber pada media online maupun yang direkam secara langsung menggunakan kamera. Adapun data sekunder yaitu literatur-literatur yang relevan dengan penelitian yang sedang dilakukan seperti jurnal maupun prosiding di situs-situs pencarian artikel ilmiah seperti IEEE, *Research Gate* dan yang semisal nya. Adapun literatur yang dicari yang berkaitan dengan sistem perhitungan jumlah kendaraan.



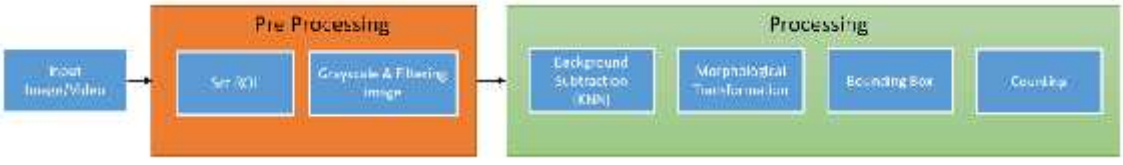
Gambar 1. Data Primer

2.3 Akuisisi Data

Akuisisi data merupakan data yang perlu disiapkan sebelum diolah oleh sistem. Akuisisi data dalam penelitian ini yaitu berupa gambar dan video jalan dari berbagai sumber online atau data yang diambil langsung di beberapa ruas jalan di kota Makassar. Data yang digunakan memiliki rasio (16:9) dan resolusi (1280 x 720) [8].

2.4 Perancangan Sistem

Rancangan atau desain sistem yang dapat dilihat pada gambar 2 merupakan gambaran yang berisi tahapan-tahapan yang akan dilakukan sistem mulai dari memasukkan data, pra pemrosesan data, sampai keluaran data yang dihasilkan.



Gambar 2. Desain Sistem

2.5 Pengujian Sistem

Proses pengujian dilakukan dengan memeriksa kondisi sebenarnya dengan hasil deteksi oleh sistem, dalam hal ini akan dilihat tingkat akurasi sistem dalam mendeteksi dan menghitung jumlah kendaraan yang melewati suatu ruas jalan seperti yang terlihat pada persamaan (1).

$$Akurasi = \frac{Kondisi\ Aktual}{Hasil\ Deteksi\ Sistem}\ x\ 100 \quad (1)$$

3. Hasil Dan Pembahasan

Data Pipeline adalah serangkaian pemrosesan data. Ada serangkaian langkah di mana setiap langkah memberikan keluaran yang merupakan masukan ke langkah berikutnya. Ini berlanjut sampai *pipeline* selesai. Proses penyelesaian *pipeline* ini menggunakan bantuan pustaka dari *OpenCV* dan *Numpy*.

3.1 Input Data

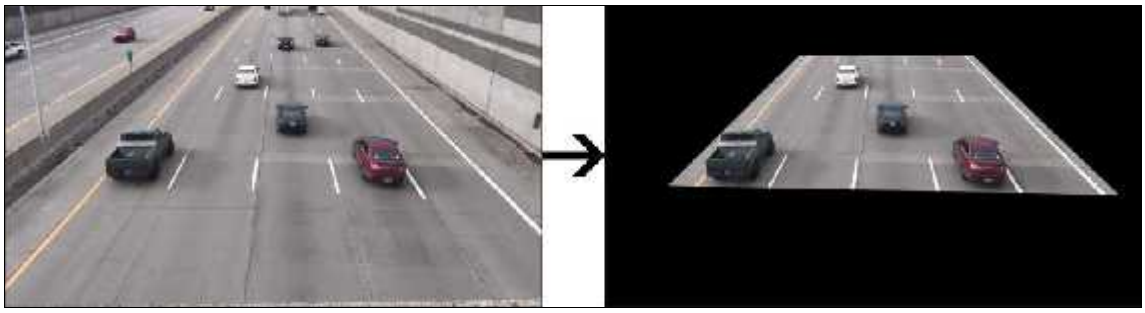
Proses pertama yang dilakukan adalah membaca data yang telah disiapkan yaitu gambar atau video. Jika sumber data diambil secara langsung menggunakan media *webcam*, maka proses *load* menggunakan parameter *index device*. Apabila datanya berupa file, cukup diketikkan nama filenya, seperti yang terlihat pada gambar 3.

```
cap = cv2.VideoCapture(0) # pengambilan secara langsung
cap = cv2.VideoCapture('video.mp4') # file
```

Gambar 3. Memuat File

3.2 Set ROI

ROI (*Region of Interest*) merupakan suatu proses yang dilakukan untuk mengatur area pemrosesan di dalam *frame*, sehingga dengan adanya ROI sistem dapat mengabaikan area-area yang tidak perlu diproses seperti yang terlihat pada gambar 4.

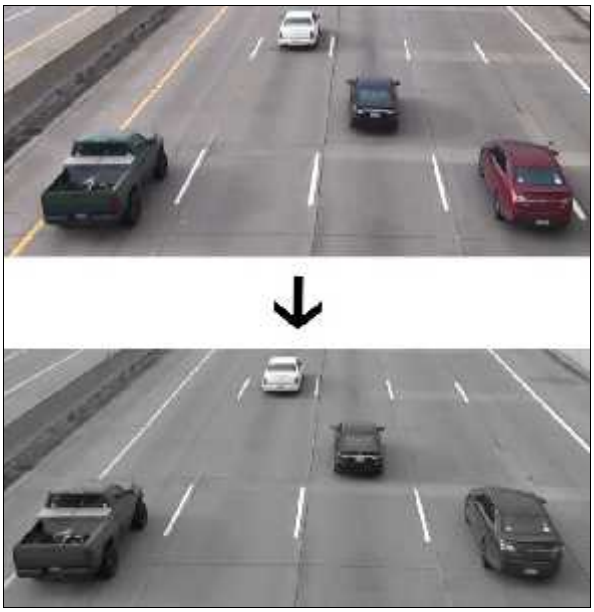


Gambar 4. Set ROI

Pada gambar 4 terlihat sistem hanya akan fokus melakukan proses pada ruas jalan di sebelah kanan, sehingga kendaraan yang melaju pada ruas jalan di sebelah kiri tidak akan terdeteksi.

3.3 Grayscale Image

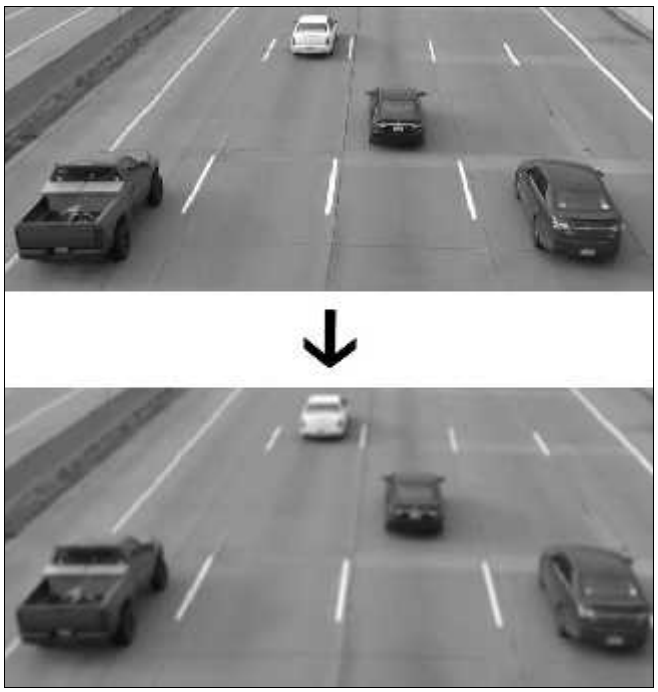
Proses selanjutnya yaitu mengubah ruang warna gambar menjadi skala keabuan, ini bertujuan untuk waktu pemrosesan lebih cepat karena hanya menggunakan *single channel* warna, selain itu gambar dengan mode *grayscale* merupakan sebuah proses awal untuk melakukan pemisahan antara objek dan latarnya atau yang dikenal dengan sebutan *Thresholding.*



Gambar 5. Grayscale Image

3.4 Image Filtering

Tahap selanjutnya yaitu meminimalisir *noise* pada gambar dengan menerapkan teknik *Gaussian Blur* dengan *kernel size* (7,7). Proses ini juga akan menghilangkan detail-detail yang tidak dibutuhkan pada saat mendeteksi sebuah objek.

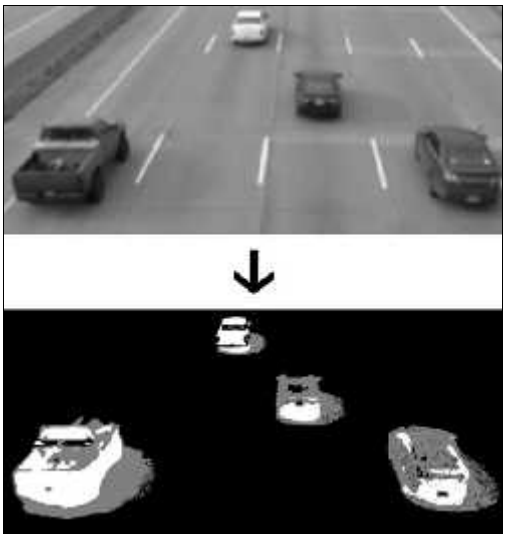


Gambar 6. Image Filtering

3.5 Background Subtraction

Proses selanjutnya yaitu proses pemisahan antara objek dan latarnya (*Thresholding*) dengan menggunakan algoritma *K-nearest neighbours.* Algoritma ini akan membuat model latar belakang dari video, dan kemudian akan mengurangi gambar dari model latar belakang untuk mendapatkan *mask foreground* objek bergerak. Algoritma ini membandingkan dua *frame* untuk

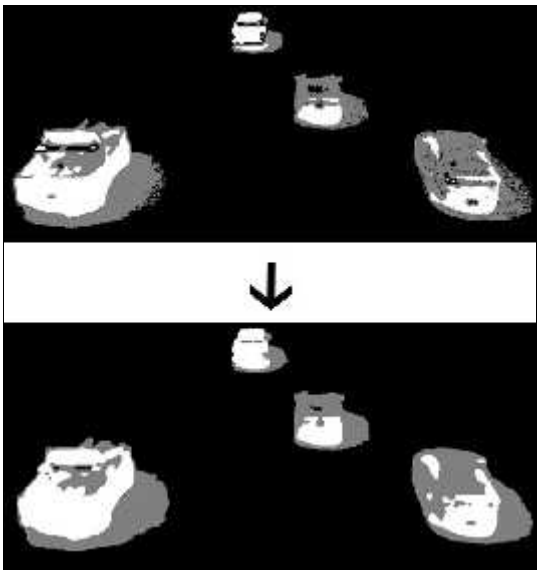
memeriksa apakah posisi piksel berubah atau tidak. Jika posisinya berubah, piksel akan ditambahkan sebagai *mask foreground*. Jadi algoritma ini hanya bisa digunakan jika kamera dalam keadaan diam dan objek bergerak seperti manusia atau kendaraan seperti yang terlihat pada gambar 8.



Gambar 7. Proses Masking

3.6 Morphological Transformations

Pada gambar 7 sistem sudah mampu mendeteksi objek yang bergerak dalam kasus ini yaitu kendaraan, proses selanjutnya adalah melebarkan piksel-piksel putih pada bagian kendaraan agar dapat mengisi bagian-bagian yang kosong dan menjadi satu kesatuan menggunakan algoritma *Morphology Image* seperti yang terlihat pada gambar 8.

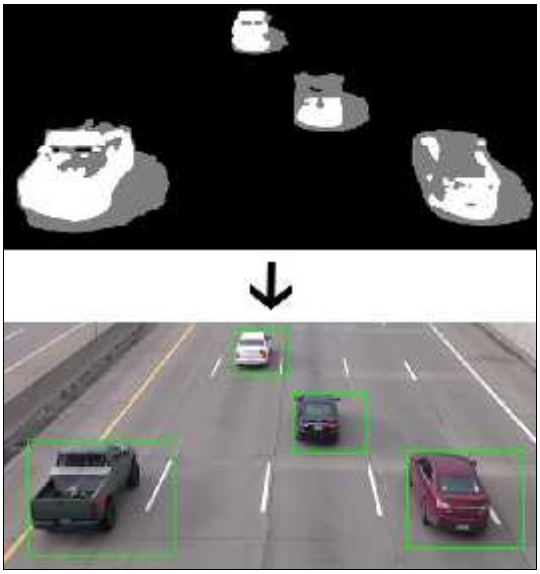


Gambar 8. Pelebaran Piksel Pada Objek

3.7 Bounding Box

Setelah mendapatkan objeknya, tahap selanjutnya yaitu lakukan proses *overlay* pada gambar semula (RGB) kemudian lakukan proses penandaan pada objek tersebut dengan memberikan sebuah kotak atau yang dikenal dengan sebutan *Bounding Box*. Setelah itu perlu

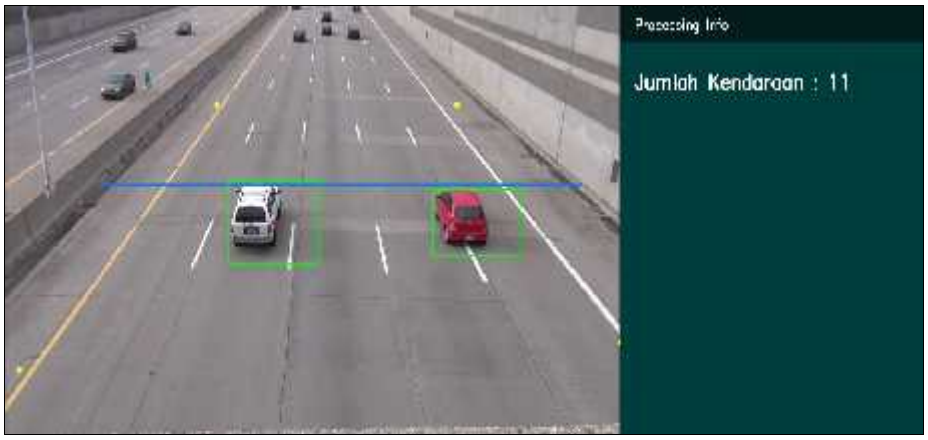
ditambahkan sebuah titik tepat di tengah objek yang berfungsi sebagai pusat dari objek tersebut yang nantinya akan digunakan dalam proses perhitungan seperti yang terlihat pada gambar 9.



Gambar 9. Penandaan Objek

3.8 Counting

Selanjutnya yaitu proses perhitungan kendaraan yang melintas. Untuk mengetahui bahwa kendaraan tersebut melintas, maka perlu adanya kondisi di dalam *frame* apabila kendaraan melewati kondisi tersebut, maka proses perhitungan kendaraan tersebut dianggap valid. Pada kasus ini dibuatkan sebuah garis horisontal pada lebar jalan yang dijadikan sebagai lokasi proses perhitungan kendaraan, sehingga kendaraan yang melintasi atau melewati garis dan sistem masih menampilkan *Bounding Box* dari kendaraan tersebut, maka kendaraan tersebut dianggap sudah terhitung dan menampilkannya ke *frame*.



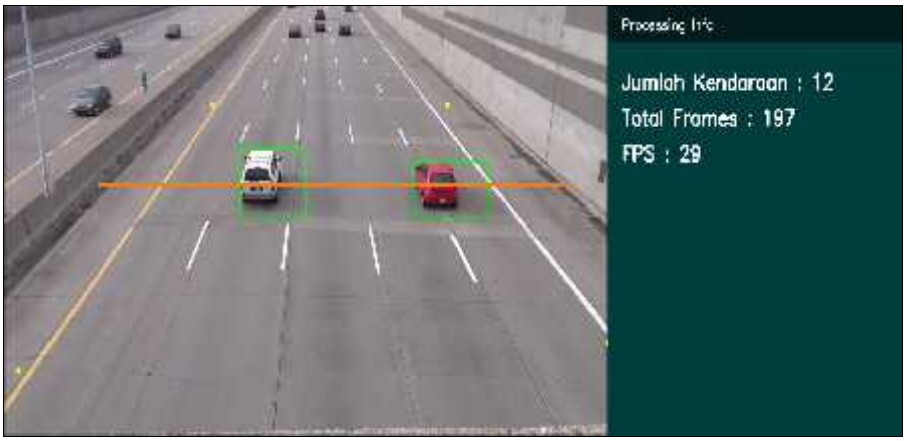
Gambar 10. Perhitungan Kendaraan

Selain dari jumlah kendaraan, sistem juga menampilkan total frame dan FPS dari waktu pemrosesan.

- Total Frames: Inisialisasikan sebuah variabel untuk menampung nilai total frame sebagai contoh beri nama “count_frame”. Selanjutnya variabel “count_frame” tambahkan nilai 1 di bagian sebelum menampilkan hasilnya di layar. Nilai dari “count_frame” akan terus bertambah selama program masih berjalan.

- FPS (Frame per Second): Untuk menghitung FPS yaitu hasil pembagian total frames dengan durasi menggunakan persamaan (2).

$$FPS = \frac{Total\ Frames}{Duration(s)} \quad (2)$$



Gambar 11. Menampilkan Total Frame dan FPS

Seperti yang terlihat pada gambar 11 FPS yang dihasilkan sistem mencapai 29, yang artinya sistem ini sangat memungkinkan jika diterapkan secara *real time.* Berikut beberapa hasil pengujian yang telah dilakukan di beberapa ruas jalan menggunakan video atau hasil rekaman seperti yang terlihat pada tabel 1.

Tabel 1. Hasil Pengujian Sistem

Sumber	Durasi (detik)	Total Frames	Kondisi Aktual	Hasil Deteksi Sistem	Akurasi
Video1.mp4	33	984	52	45	86,5%
Video2.mp4	80	1.659	36	23	63,8%
Video3.mp4	124	2.330	34	29	85,2%
Rata-rata					78,5%

4. Kesimpulan

Berdasarkan hasil penelitian yang dilakukan dapat ditarik kesimpulan bahwa sistem ini mampu melakukan perhitungan kendaraan yang melintas pada suatu ruas jalan berdasarkan hasil pengujian di beberapa ruas jalan mencapai tingkat akurasi dengan rata-rata 78,5%. Untuk mencapai akurasi yang lebih tinggi perlu adanya pendekatan-pendekatan baru agar dapat meminimalisir hasil deteksi dari bayangan kendaraan sehingga proses *Bounding Box* lebih presisi.

5. Saran

Penelitian ini tentu masih memiliki kekurangan khususnya pada bagian *Thresholding* dalam memisahkan objek dan *backgroundnya* dan juga kondisi cuaca yang gelap. Harapan peneliti kepada peneliti selanjutnya agar dapat menemukan pendekatan atau metode baru untuk menyelesaikan kekurangan dari penelitian ini.

DAFTAR PUSTAKA

[1] B. P. Statistik, "Perkembangan Jumlah Kendaraan Bermotor Menurut Jenis (Unit), 2018-2020," 2022. [Online]. Available: https://www.bps.go.id/indicator/17/57/1/jumlah-kendaraan-bermotor.html. [Accessed: 13-Jul-2022].

[2] M. Aynurrohmah and A. Sunyoto, "Penghitung Jumlah Mobil Menggunakan Pengolahan Citra Digital Dengan Input Video Digital," *Data Manaj. dan Teknol. Inf.*, vol. 12, no. 3, pp. 2–6, 2011.

[3] D. J. P. Manajang, S. R. U. A. Sompie, and A. Jacobus, "Implementasi Framework Tensorflow Object Detection API Dalam Mengklasifikasi Jenis Kendaraan Bermotor," *J. Tek. Inform.*, vol. 15, no. 3, pp. 171–178, 2020.

[4] K. S. D. S and R. Kamalraj, "Vehicle Detection and Counting of a Vehicle Using Opencv," no. 05, pp. 1610–1613, 2021.

[5] A. N. N. Afifah, Indrabayu, A. Suyuti, and Syafaruddin, "A review on image processing techniques for damage detection on photovoltaic panels," *ICIC Express Lett.*, vol. 15, no. 7, pp. 779–790, 2021, doi: 10.24507/icicel.15.07.779.

[6] Gansar Suwanto, R. Ibnu Adam, and Garno, "Identifikasi Citra Digital Jenis Beras Menggunakan Metode Anfis dan Sobel," *J. Inform. Polinema*, vol. 7, no. 2, pp. 123–128, 2021, doi: 10.33795/jip.v7i2.406.

[7] M. S. Mustafa and I. W. Simpen, "Implementasi Algoritma K-Nearest Neighbor (KNN) Untuk Memprediksi Pasien Terkena Penyakit Diabetes Pada Puskesmas Manyampa Kabupaten Bulukumba," *Pros. Semin. Ilm. Sist. Inf. Dan Teknol. Inf.*, vol. VIII, no. 1, pp. 1–10, 2019.

[8] A. A. Mahersatillah, Z. Zainuddin, and Y. Yusran, "Unstructured Road Detection and Steering Assist Based on HSV Color Space Segmentation for Autonomous Car," in *2020 3rd International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2020*, 2020, doi: 10.1109/ISRITI51436.2020.9315452.