IMPLEMENTASI ALGORITMA REGION OF INTEREST (ROI) UNTUK MENINGKATKAN PERFORMA ALGORITMA DETEKSI DAN KLASIFIKASI KENDARAAN

by Wilis Kaswidjanti

Submission date: 30-Jun-2020 11:11AM (UTC+0700)
Submission ID: 1351641319
File name: IMPLEMENTASI_ALGORITMA_ROI_-AWANG-WILIS-IFA.pdf (1.34M)
Word count: 4381
Character count: 24592

Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)
Vol. 7, No. 1, Februari 2020, hlm. 155-162
Akreditasi KEMENRISTEKDIKTI, No. 30/E/KPT/2018

DOI: 10.25126/jtiik.202071718
p-ISSN: 2355-7699
e-ISSN: 2528-6579

IMPLEMENTASI ALGORITMA REGION OF INTEREST (ROI) UNTUK MENINGKATKAN PERFORMA ALGORITMA DETEKSI DAN KLASIFIKASI KENDARAAN

Awang Hendrianto Pratomo*[1], Wilis Kaswidjanti[2], dan Siti Mu'arifah[3]

[1,2,3.] Jurusan Teknik Informatika Fakultas Teknik Industri UPN "Veteran" Yogyakarta
Email : [1]awang@upnyk.ac.id, [2]wilis.kas@gmail.com, [3]sitimuarifah14@gmail.com
*Penulis Korespondensi

(Naskah masuk: 28 Januari 2019, diterima untuk diterbitkan: 14 Januari 2020)

Abstrak

Semakin tinggi kualitas suatu citra maka semakin detail informasi yang akan di peroleh. Tetapi, tidak semua wilayah citra memungkinkan untuk dilakukan analisis dengan kecepatan proses yang tinggi. Pemilihan algoritma yang tepat berpengaruh terhadap kecepatan waktu pemrosesan. Apabila tidak ada pembatasan untuk area yang akan di proses mengakibatkan waktu pemrosesan secara *realtime* melebihi waktu pemrosesan maksimal yang seharusnya. Tingginya waktu pemrosesan yang terjadi mengakibatkan aliran data menjadi kurang cepat. Sarana/*processor* yang digunakan juga mampu mempengaruhi kecepatan pemrosesan. *Region Of Interest* (ROI) adalah cara yang tepat untuk mengurangi tingginya waktu pemrosesan tersebut. ROI mampu menandai area tertentu sehingga dapat digunakan untuk mengoptimalisasikan kinerja sistem untuk mendeteksi, menghitung dan mengklasifikasi kendaraan secara *realtime*. Tanpa adanya ROI, pemrosesan dilakukan pada seluruh piksel citra tanpa terkecuali. Terdapat beberapa tahapan yang dilakukan di dalam penelitian yaitu menganalisis masalah yang ada, penentuan wilayah ROI, aplikasi ROI sebelum proses pengolahan citra dan menganalisis hasil yang di dapatkan. Hasil yang diperoleh adalah dengan menggunakan ROI waktu pemrosesan citra menggunakan metode segmentasi MOG2 dan tracking dapat lebih cepat dibandingkan dengan waktu pemrosesan ketika tidak menggunakan ROI dengan selisih 0,026 s atau setara dengan 26 ms/frame.

Kata Kunci: *pengolahan citra, klasifikasi kendaraan, region of lnterest (ROl), waktu pemrosesan,*

IMPLEMENTATION OF REGION OF INTEREST (ROI) ALGORITHM TO IMPROVE CAR DETECTION AND CLASSIFICATION ALGORITHM

Abstract

Increasing resolution of an image is more detailed information will be obtained especially in the image used to detect vehicles. But, every singles areas are not allow to analize with higher speed process. If there are no restrictions for the area to be processed, the processing time in real time exceeds the maximum processing time that should be. The high processing time that occurs make less rapid data flow. The high processing time can affect to processing speed. Region Of Interest (ROI) is the right way to reduce the high processing time. ROI is able to mark certain areas so that it can be used to optimize system performance to detect, calculate and classify vehicles in realtime. Without ROI, processing is carried out on all pixels without exception. There are several steps taken in the research, namely analyzing existing problems, determining the ROI area, application of ROI before the image processing and analyzing the results obtained. The results obtained are by using ROI image processing time can be faster than the processing time when not using ROI.

Keywords : *image processing, car classification, region of lnterest (ROI), processing time*

1. PENDAHULUAN

Deteksi kendaraan menggunakan kamera merupakan topik yang menarik untuk dijadikan penelitian dan pengembangan (Adistya and Muslim, 2016; Alamsyah, 2017). Berbagai macam kamera digunakan untuk mendapatkan citra dengan kualitas tinggi. Semakin tinggi kualitas suatu citra semakin

detail informasi yang akan di peroleh. Informasi dari perolehan citra dapat digunakan untuk mendeteksi kendaraan.

Deteksi kendaraan dapat dilakukan dengan *contouring* yang dilakukan secara urut menggunakan fungsi-fungsi pengolahan citra. Fungsi pengolahan citra akan memproses semua piksel/wilayah pada *frame* pada proses deteksi Citra(Mu'arifah, 2018). Keseluruhan wilayah citra pada hasil deteksi pada umumnya digunakan untuk proses analisis. Semakin luas wilayah citra dapat mempengaruhi kecepatan proses deteksi. Pemilihan algoritma yang tepat berpengaruh terhadap kecepatan pemrosesan itu sendiri.

Waktu pemrosesan citra setiap *frame* digunakan untuk menghitung waktu yang dibutuhkan dalam menyelesaikan suatu proses setiap *frame*. Beberapa faktor yang mempengaruhi waktu pemrosesan salah satunya adalah kualitas citra atau resolusi citra. Semakin besar resolusi citra, semakin besar jumlah piksel pada citra tersebut dan semakin lama waktu yang dibutuhkan untuk mengolah citra tersebut (Ma et al. 2015). Faktor kedua adalah *processor* yang digunakan. Semakin bagus *processor* yang digunakan, proses dapat dipecah kedalam beberapa bagian untuk mempercepat waktu pemrosesan. Faktor ketiga adalah waktu datangnya *frame*. Semakin banyak jumlah *frame* yang ditangkap kamera setiap detik, semakin cepat waktu pemrosesan yang dibutuhkan untuk melakukan proses setiap *frame*nya. Jika terdapat *frame* baru yang sudah datang saat proses pada *frame* lama belum selesai, dapat akan menyebabkan antrian pada *frame* sehingga waktu pemrosesan keseluruhan akan menjadi lebih lama dan menyebabkan kegagalan dalam pemrosesan citra. Supaya waktu pemrosesan dapat dipercepat salah satunya dapat dilakukan dengan pembatasan wilayah yang akan di proses pada citra (Ma et al. 2015).

Cara yang dapat digunakan untuk pembatasan wilayah citra dapat dilakukan dengan menggunakan *Region Of Interest* (ROI). ROI merupakan cara yang tepat untuk mengurangi masalah tingginya waktu pemrosesan. ROI mampu memberikan tanda terhadap area tertentu sehingga dapat digunakan untuk mengoptimalisasikan kinerja system. Tanpa penggunaan ROI, pemrosesan pada citra dilakukan pada seluruh piksel citra tanpa terkecuali (Ma et al. 2015; Linda, 2010).

Pada penelitian yang telah dikembangkan oleh Ma (2012) ROI diimplementasikan untuk mendeteksi rute pelayaran. ROI digunakan untuk membatasi wilayah yang akan digunakan untuk menghemat proses komputasi, penyimpanan dan pengambilan keputusan secara cerdas. Implementasi ROI yang dikembangkan secara manual, menyebabkan berkurangnya tingkat akurasi. Pembuatan ROI dapat dilakukan secara otomatis dengan mendeteksi bagian terluar pada bagian atas dan bawah objek, sehingga dapat meningkatkan akurasi (Han and Vasconcelos, 2008). Implementasi ROI pada pengolahan ctra proses segmentasi dilakukan pada area yang telah ditentukan saja tanpa memotong gambar yang telah diperoleh sebelumnya(Fajrin, 2016). ROI juga dapat digunakan untuk kompresi data, metode ROI doginakan pada algoritma kompresi JPEG 2000(Linda, 2010).

Region of Interest (ROl) merupakan suatu bagian citra yang dipilih untuk di proses (Linda, 2010). Salah satu fungsi ROI yaitu untuk menandai suatu objek yang sedang bergerak (Kurniawan, 2015). Algoritma deteksi objek dilakukan dengan menempatkan daerah ROI pada area jalan yang dilewati oleh kendaraan. Kendaraan yang melintasi pada kawasan ROI tersebut yang akan dihitung atau di deteksi tanpa memisahkan kawasan ROI dari *frame* aslinya (Wicaksono, 2017; Sogen and Kusuma, 2014; Coifman et al., 1998).

Region of Interest (ROI) merupakan salah satu teknik segmentasi sebagai proses pengolahan citra dimana pengguna mampu mengolah citra yang mengandung informasi data citra yang dikehendaki. ROI bekerja dalam pengkodean secara berbeda pada area tertentu dari citra digital sehingga area citra yang lebih penting akan memiliki kualitas yang lebih baik dari area disekitarnya (Falah et al., 2016).

Pada penelitian ini bertujuan untuk mengatasi masalah waktu pemrosesan yang terjadi ketika proses deteksi kendaraan secara *realtime*. ROI ditempatkan pada proses sebelum pengolahan citra, hal ini berguna agar daerah yang dilakukan proses pengolahan citra berfokus pada *frame* ROI dan bukan pada seluruh *frame*. Sehingga, ROI dapat digunakan untuk membatasi kawasan pengamatan suatu kendaraan (Sogen and Kusuma, 2014) dan daerah ROI tidak dapat dibuat secara spesifik karena untuk menyesuaikan ke objek atau aplikasi yang lain merupakan suatu masalah yang tidak sederhana (Han and Vasconcelos, 2008).

2. METODOLOGI

Metodologi penelitian yang dilakukan dalam penelitian ini menggunakan *Region Of Interest* (ROI) untuk menyelesaikan permasalahan yang terjadi. Proses ROI ditempatkan sebelum proses pengolahan citra yang ditunjukkan pada Gambar 1.

Citra digital yang ditangkap oleh kamera kemudian akan dilakukan proses *resize* ukuran video sesuai dengan ukuran yang telah ditetapkan. kemudian dilanjutkan dengan proses ROI dan proses pengolahan citra.

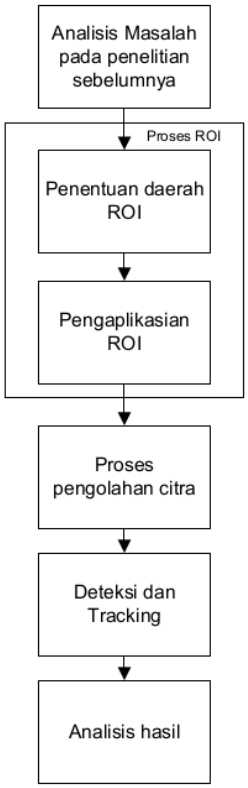
2.1. Analisis Masalah

Berdasarkan penelitian yang dilakukan oleh Mu'arifah (2018) untuk mendapatkan akurasi deteksi kendaraan membutuhkan waktu yang lama. Ketika diimplementasi secara *realtime*, deteksi kendaraan menjadi tidak akurat. Waktu datangnya *frame* lebih cepat dari waktu pemrosesan setiap *frame*. Daerah yang di proses adalah seluruh piksel yang terdapat didalam *frame*.

Rata-rata waktu pemrosesan untuk mengolah data citra, dimulai dari proses pemisahan antara *background* dan *foreground*, *tresholding*, hingga *tracking* kendaraan seperti pada pada Tabel 1. adalah 0,248 atau setara dengan 248 ms. Sedangkan waktu pemrosesan maksimal adalah 33 ms. Berdasarkan dari hasil rata-rata waktu pemrosesan masalah utamanya adalah bagaimana meningkatkan kecepatan waktu pemrosesan. Sehingga pada penelitian ini mengusulkan solusi dengan memberi area pemrosesan menggunakan ROI. Untuk menunjukkan keberhasilan beberapa macam *processor* antara lain laptop, Raspberry Pi3 dan Odroid Xu4.

2.2. Proses ROI

Proses ROI yang terjadi adalah dengan memilih area pada *frame* video. Proses ROI dalam penelitian ini digunakan untuk membatasi atau memperkecil area pemrosesan. Pembatasan area pemrosesan dilakukan dengan cara menentukan area jalan raya. Pembatasan area jalan raya dilakukan agar objek-objek yang berada diluar area tersebut tidak menjadi penambahan *noise* pada citra yang akan di proses. Tujuan daripada pengimplementasian ROI dalam penelitian ini adalah untuk meningkatkan waktu pemrosesan sehingga dapat diimplementasikan pada mini PC(Personal Computer) seperti Raspberry Pi dan Odroid UX4. Implementasi ROI di dalam penelitian ini terdapat dua tahapan, pertama menentukan titik penentuan untuk daerah ROI dan kedua mengaplikasikan ROI dalam program sistem, sehingga mampu mempercepat waktu pemrosesan.



Gambar 1. Metodologi Penelitian

Tabel. 1. Waktu pemrosesan sebelum menggunakan ROI

No	Pagi	Siang	Sore	Malam
	Tanpa ROI	Tanpa ROI	Tanpa ROI	Tanpa ROI
1	0,303	0,292	0,19	0,229
2	0,29	0,297	0,202	0,196
3	0,352	0,264	0,189	0,164
4	0,26	0,329	0,21	0,197
5	0,351	0,212	0,218	0,17
6	0,388	0,285	0,244	0,268
7	0,306	0,288	0,245	0,255
8	0,256	0,354	0,217	0,275
9	0,344	0,294	0,168	0,245
10	0,306	0,308	0,211	0,21
…	…			
…	0,278	0,259	0,223	0,195
Rata-Rata (second)	**0,279**	**0,286**	**0,211**	**0,218**
Total rata-rata (No ROI)		**0,248**		

ROI digunakan untuk membatasi area deteksi kendaraan,video yang digunakan, diambil di *underpass* Jombor Yogyakarta dengan menggunakan *webcam* Logitech C270 dengan resolusi 1280 x 960, dengan maksimal 30fps. Format video disimpan dalam ekteknsi *mp4*. Waktu pengambilan data dilakukan pada waktu pagi, siang, sore dan malam hari pada titik yang sama, dengan catatan pengambilan data dilakukan dalam kondisi cuaca cerah.

Pengolahan data citra dilakukan dengan bahasa pemrograman Python dan menggunakan OpenCV untuk *library* pengolahan data citra. Pengujian dilakukan dilakukan dengan menggunakan video

yang telah direkam sebelumnya pada waktu yang telah disebutkan pada bagian sebelum ini.

1) Menentukan daerah ROI

Penerapan algoritma ROI dilaksanakan dengan cara memproses *frame* secara utuh (Mu'arifah, 2018). ROI digunakan untuk mengoptimalkan algoritma dengan melakukan pembatasan pada daerah tertentu pada *frame* sehingga proses yang hanya dilakukan pada bagian ROI dan bukan pada keseluruhan *frame*. Implementasi ROI pada *frame* dapat ditunjukkan pada Gambar 2. Gambar 3. Menunjukkan frame asli sebelum penambahan proses ROI menunjukkan masih banyaknya *noise* atau gangguan saat proses *tracking* kendaraan. Pada Gambar 4. menunjukkan bahwa setelah ditambahkan proses ROI *noise* atau gangguan dapat berkurang sehingga waktu pemrosesan dapat menjadi lebih cepat.

Tahapan penentuan wilayah ROI pada deteksi kendaraan adalah sebagai berikut:

Langkah 1. Ambil titik terluar garis penghitung yang berada tepat pada tepi jalan

Langkah 2. Gunakan titik terluar kanan dan kiri pada tepi jalan

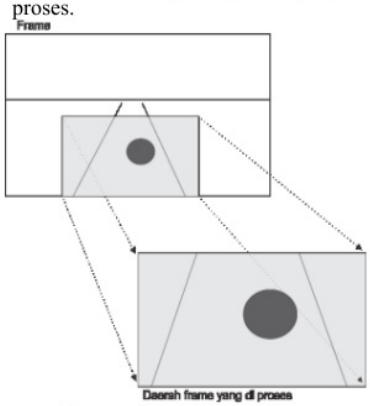
Langkah 3. Letak piksel titik terluar pada sebelah kiri, dikurangi dengan konstanta yang telah ditentukan. Kemudian pada bagian sebelah kanan, letakan piksel titik terluar ditambah dengan konstanta yang ditentukan.

Langkah 4. Tarik garis lurus sejajar pada ke empat titik sehingga terbentuk wilayah ROI.

Penentuan wilayah pada ROI dimulai dari titik kiri atas ke titik kanan bawah. Proses pengambaran ROI ditunjukkan pada Gambar 5.

Proses impementasi ROI dilakukan sebagai berikut:

1. Mendeklarasikan fungsi ROI
2. Memanggil fungsi ROI untuk implementasikan pada *frame* yang akan di proses.



Gambar 2. ROI pada *Frame*

Gambar 3. *Frame* asli

Gambar 4. *Frame* hasil ROI

Proses deklarasi fungsi pada ROI dapat ditunjukkan pada Program 1. berikut.

```
def get_roi(grabbed, frame):
     (LINE_BOTTOM, _) =
frame.shape[:2]
     roi =
frame[LINE_TOP:LINE_BOTTOM,
LINE_LEFT:LINE_RIGHT]

     return grabbed,roi
```

Program 1. Proses Pendefinisian Fungsi ROI

2) Mengaplikasikan ROI

Kemudian fungsi ROI implementasikan pada prosses *frame*. Implementasi ROI ditunjukkan pada Program 2. berikut.

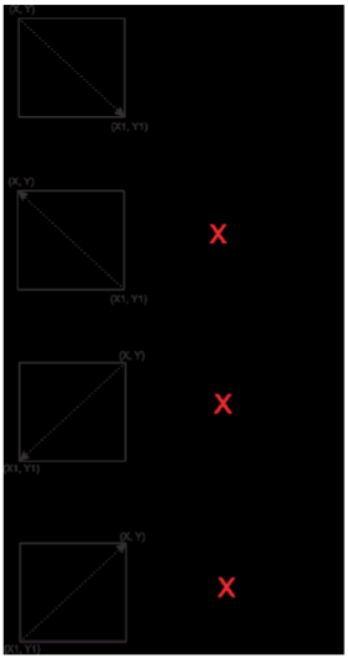
```
frame_cp = frame.copy()
grab_roi, frame_roi =
get_roi(grabbed, frame_cp)
```

Program 2. Proses Pemanggilan dan Pengimplementasian Fungsi ROI

3) Melakukan proses proyeksi *frame* ROI dengan *frame* sebelum dikenai proses ROI.

Proyeksi dilakukan dengan melihat titik tengah objek berada. Apabila objek berada di daerah tertentu pada ROI maka titik X objek akan ditambahkan dengan line left. Line left merupakan titik awal pada saat menentukan titik piksel untuk lebar jalan pada piksel.

Sedangkan titik y pada objek akan ditambahkan dengan line top. Line lop merupakan titik y pada garis deteksi. Sehingga posisi objek pada ROI akan sama dengan posisi objek pada *frame*. Hal ini ditunjukkan pada contoh program 3.



Gambar 5. Arah titik pembuatan ROI

```
x += LINE_LEFT
y += LINE_TOP
```

Program 3. Proyeksi *Frame* ROI

Proyeksi koordinat ROI ke *frame* diperlukan karena koordinat ROI dimulai kembali dari 0,0 dan bukan dari koordinat *frame*. Penggambaran proyeksi daerah pada ROI ditunjukkan dalam Gambar 6. berikut.

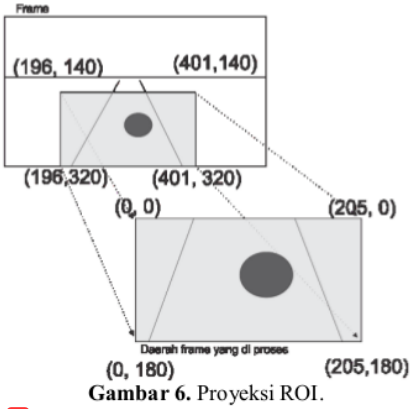
Proyeksi daerah ROI dimulai pada koordinat (196,140) pada *frame* utama sebagai proyeksi dari koordinat (0,0), sedangkan pada koordinat (205,0) diproyeksikan pada koordinat (401,140), kemudian pada koordinat (205,180) di proyeksikan pada koordinat (401,320) dan proyeksi untuk kordinat (0,180) adalah koordinat (196,320).

2.3. Proses Pengolahan Citra

ROI yang ditempatkan sebelum proses pengolahan citra berguna agar daerah yang dilakukan proses pengolahan citra berfokus pada *frame* ROI dan bukan pada seluruh *frame*. Pengolahan citra sendiri merupakan sebuah disiplin ilmu yang mempelajari tentang bagaimana memanipulasi sebuah citra untuk meningkatkan kualitas dari citra (Kurniawan, 2015). Dalam memanipulasi citra dilakukan beberapa cara yang disebut dengan teknik-teknik pengolahan citra.

Pengolahan citra pada penelitian terdapat lima(5) tahapan, yaitu :

1. Pemisahan objek dengan latar belakangnya menggunakan Background Substractor MOG2.
2. Penggunaan thresholding untuk mempertegas tepi objek
3. Smoothing citra menggunakan Gaussian Blur
4. Thresholding kembali untuk mempertegas objek
5. Penutupan lubang pada objek menggunakan morfologi closing.



Gambar 6. Proyeksi ROI.

2.4. Deteksi dan *Tracking*

Proses deteksi dilakukan dengan menggunakan fungsi *Contouring* yang terdapat pada OpenCv. Fungsi *contour* merupakan pengenalan kontur pada citra, fungsi *contour* dapat digunakan untuk mendeteksi bentuk suatu objek dengan cara menggabungkan titik-titik yang terbentuk dari setiap lekuk pada suatu kontur. Selain itu fungsi *Countour* juga dapat digunakan untuk menganalisa bentuk objek, mendeteksi dan mengenali suatu objek. Setelah proses deteksi selanjutnya dilakukan proses *tracking*.

Tracking atau pelacakan merupakan suatu teknik untuk mengidentifikasi suatu objek yang sama sehingga tidak kehilangan objek yang akan di dedeteksi, dalam hal ini *tracking* berguna untuk mengetahui arah pergerakan suatu objek.

Fungsi *tracking* dilakukan dengan menggunakan titik tengah objek agar dapat disimpan dalam suatu *array*. Fungsi *tracking* dilaksanakan dengan menggunakan fungsi yang ada pada OpenCV yaitu, fungsi *convexHull* untuk memperbaiki bentuk dengan mengabaikan cekungan pada tepian objek dan *boundingRect* untuk memberikan bingkai berbentuk persegi pada objek supaya dapat diketahui berapa ukuran suatu objek.

3. HASIL DAN PEMBAHASAN

Pengujian dilakukan dengan mengumpulkan data video yang telah ditentukan batasan-batasan pengambilannya, misal tinggi kamera, fps kamera, waktu pengambilan dan lain-lain. Pengujian

dilakukan dengan melihat waktu pemrosesan yang terjadi ketika digunakan algoritma sebelum menggunakan ROI dan kemudian dilakukan perbandingan dengan melihat waktu pemrosesan yang terjadi ketika ditambahkan proses ROI yang dijalankan pada beberapa mesin antara lain (laptop, raspberry pi3 dan odroid xu4). Hasil pengujian menggunakan Laptop ditunjukkan dalam Tabel 2, hasil pengujian menggunakan Raspberry Pi3ditunjukkan dalam Tabel 3, sedangkan hasil pengujian menggunakan Odroid UX4 ditunjukkan dalam Tabel 4.

3.1. Pengujian Waktu Pemrosesan Pada Laptop.

Rata-rata waktu pemrosesan yang terjadi pada laptop sebelum meggunakan ROI yaitu sebesar 35 ms dengan spesifikasi laptop yang digunakan adalah sebagai berikut :

a. lntel core i3-350M, 2.26 GHz
b. Cache 3MB
c. Memory: 8GB

Dari waktu pemrosesan sebelum menggunakan ROI yang diperoleh, dapat dilihat bahwa rata-rala waktu pemrosesan tidak melebihi dari waktu pemrosesan maksimalnya. Waktu pemrosesan maksimal dapat diketahui dengan persamaan apabila *frame* per second (fps) yang digunakan adalah 25 fps maka waktu pemrosesan maksimal adalah 1125 = 0,04 second atau setara dengan 40 ms. Rata-rata waktu pemrosessan menggunakan ROI sebesar 0,0094 s atau setara dengan 9,4 ms untuk setiap *frame*. Hal ini menunjukkan bahwa waktu pemrosesan sebelum dan setelah diimplementasikan penggunaan ROI memiliki perbedaan. Waktu pemrosessan setelah implementasi proses ROI pada pengujian yang dilakukan menggunakan laptop memiliki waktu pemrosessan yang jauh lebih cepat dibandingkan dengan waktu pemrosessan sebelum implementasi proses ROI. Baik menggunakan ROI atau tidak, waktu pemrosesssan pada laptop masih dapat digunakan karena tidak melebihi dari batas waktu proses maksimalnya.

Tabel 2. Pengujian waktu pemrosesan pada Laptop

No	Pagi			Siang			Sore			Malam		
	Tanpa ROI	ROI	Selisih	Tanpa ROI	ROI	Selisih	Tanpa ROI	ROI	Selisih	Tanpa ROI	ROI	Selisih
1	0,0350	0,0070	0,0280	0,0290	0,0120	0,0170	0,0310	0,0060	0,0250	0,0260	0,0070	0,0190
2	0,0180	0,0070	0,0110	0,0190	0,0070	0,0120	0,0170	0,0040	0,0130	0,0170	0,0030	0,0140
3	0,0180	0,0050	0,0130	0,0220	0,0070	0,0150	0,0170	0,0020	0,0150	0,0170	0,0030	0,0140
4	0,0180	0,0040	0,0140	0,0190	0,0070	0,0120	0,0180	0,0020	0,0160	0,0160	0,0030	0,0130
5	0,0340	0,0070	0,0270	0,0210	0,0080	0,0130	0,0190	0,0030	0,0160	0,0200	0,0040	0,0160
6	0,0230	0,0060	0,0170	0,0290	0,0080	0,0210	0,0200	0,0020	0,0180	0,0190	0,0040	0,0150
7	0,0200	0,0050	0,0150	0,0260	0,0090	0,0170	0,0160	0,0040	0,0120	0,0230	0,0040	0,0190
8	0,0210	0,0050	0,0160	0,0250	0,0090	0,0160	0,0160	0,0040	0,0120	0,0200	0,0030	0,0170
9	0,0200	0,0050	0,0150	0,0290	0,0090	0,0200	0,0310	0,0030	0,0280	0,0260	0,0030	0,0230
10	0,0210	0,0060	0,0150	0,0190	0,0080	0,0110	0,0170	0,0040	0,0130	0,0170	0,0040	0,0130
Rata-Rata	**0,0345**	**0,0121**	**0,0225**	**0,0324**	**0,0080**	**0,0243**	**0,0345**	**0,0087**	**0,0257**	**0,0397**	**0,0090**	**0,0307**

Tabel 3. Pengujian pada Raspberry Pi3

No	Video Waktu Pengujian											
	Pagi			Siang			Sore			Malam		
	Tanpa ROI	ROI	Selisih	Tanpa ROI	ROI	Selisih	Tanpa ROI	ROI	Selisih	Tanpa ROI	ROI	Selisih
1	0,3028	0,0301	0,2727	0,2921	0,0330	0,2591	0,1896	0,0413	0,1484	0,2295	0,0299	0,1996
2	0,2877	0,0306	0,2571	0,2969	0,0320	0,2649	0,2022	0,0318	0,1704	0,1961	0,0487	0,1475
3	0,3522	0,0483	0,3039	0,2640	0,0275	0,2366	0,1893	0,0642	0,1251	0,1638	0,0304	0,1334
4	0,2601	0,0317	0,2285	0,3293	0,0274	0,3019	0,2095	0,0277	0,1818	0,1971	0,0294	0,1677
5	0,3508	0,0408	0,3100	0,2123	0,0383	0,1740	0,2177	0,0288	0,1889	0,1704	0,0199	0,1505
6	0,3883	0,0484	0,3399	0,2849	0,0426	0,2423	0,2438	0,0383	0,2055	0,2681	0,0381	0,2300
7	0,3058	0,0223	0,2834	0,2879	0,0410	0,2470	0,2452	0,0300	0,2152	0,2552	0,0351	0,2201
8	0,2559	0,0465	0,2094	0,3535	0,0312	0,3223	0,2167	0,0282	0,1886	0,2749	0,0351	0,2398
9	0,3438	0,0492	0,2946	0,2944	0,0320	0,2624	0,1682	0,0279	0,1403	0,2455	0,0342	0,2113
10	0,3064	0,0583	0,2480	0,3084	0,0328	0,2757	0,2101	0,0290	0,1811	0,2096	0,0349	0,1747
Rata-Rata	**0,2791**	**0,0419**	**0,2372**	**0,2861**	**0,0382**	**0,2479**	**0,2112**	**0,0402**	**0,1709**	**0,2177**	**0,0433**	**0,1745**

Tabel 4. Pengujian Implementasi ROI menggunakan Odroid Xu4

No	Video Waktu Pengujian			
	Pagi	Siang	Sore	Malam
1	0,27214	0,28790	0,28925	0,28764
2	0,00552	0,00480	0,00508	0,00524
3	0,00661	0,00588	0,00609	0,00650
4	0,00700	0,00588	0,00624	0,00647
5	0,00701	0,00595	0,00594	0,00698
6	0,00864	0,00601	0,00646	0,00732
7	0,00897	0,00609	0,00618	0,00830
8	0,00802	0,00604	0,00640	0,00758
9	0,00791	0,00613	0,00614	0,00774
10	0,00796	0,00604	0,00645	0,00752
Total rata-rata	**0,00959**			

3.2. Pengujian Waktu Pemrosesan Pada Raspberry Pi3

Pengujian waktu pemrosessan setelah pengimplementasian ROI menggunakan Raspberry Pi3 ditunjukkan dalam Tabel 3.

Waktu pemrosesan yang didapat setelah pengimplementasian ROI menggunakan Raspberry Pi3 yaitu sebesar 41 ms. Hal ini masih diatas waktu pemrosesan maksimalnya yaitu 40 ms. Akan tetapi, dapat dilihat bahwa waktu pemrosesan menggunakan ROI pada pengujian yang dilakukan menggunakan Raspberry Pi3 memiliki waktu pemrosesan yang jauh lebih cepat daripada waktu pemrosesan yang dilakukan sebelum diimplementasikannya ROI ketika dilakukan pengujian menggunakan Raspberry Pi3 yang memiliki selisih waktu pemrosesan sebesar 207.53 ms.

3.3. Pengujian Waktu Pemrosesan Setelah Menggunakan ROI pada Odroid Xu4

Untuk memastikan apakah waktu pemrosesan juga berpengaruh terhadap *processor* yang digunakan, maka pengujian menggunakan salah satu mini pc yang lain selain menggunakan Raspberry Pi3 yaitu menggunakan Odroid Xu4 dengan spesifikasi yang dimiliki oleh Odroid Xu4 adalah sebagai berikut :

a. CPU - Samsung Exynos544 Cortex-A15 + Korteks A7 2 GHz (Octacore 64 bit)
b. GPU - ARM Mali T628-MP6 Octa Core 600MHz
c. RAM - 2GB LPDDR3 SDRAM

Berdasarkan hasil pengujian setelah pengimplementasian proses ROI menggunakan Odroid Xu4 (Tabel 4.) memiliki waktu pemrosesan sebesar 10 ms. Pengujian menggunakan Odroid Xu4 memiliki rata-rata sebesar 0,0095 s atau setara dengan 9,5 ms. Dibandingkan dengan beberapa pengujian yang dilakukan sebelumnya, waktu pemrosesan yang di dapatkan jauh lebih kecil apabila dibandingkan dengan pengujian yang dilakukan pada Raspberry Pi3. Akurasi deteksi menggunakan proses ROI memiliki tingkat akurasi maksimal. Tabel 5. menunjukkan bahwa hasil deteksi setelah penambahan proses ROI rata-rata lebih besar dari 70%. Hal ini menunjukkan bahwa proses ROI dapat meningkatkan akurasi karena informasi yang diperoleh dari citra tidak hilang. Selain itu dengan ditambahkannya proses ROI mampu mempercepat waktu pemrosesan dan mampu mengurangi banyak *noise* yang ada sehingga akurasi menjadi lebih optimal. Apabila tidak diimplementasikan ROI, hal yang dapat terjadi adalah waktu pemrosesan citra yang dibutuhkan melebihi dari waktu maksimal pemrosesan, sehingga mengakibatkan adanya beberapa *frame* yang tidak dikenai proses pengolahan citra, sehingga akan mempengaruhi perhitungan akrasi. Hasil perhitungan akurasi menggunakan proses ROI pada kondisi *realtime* ditunjukkan pada Tabel 5 berikut.

Tabel 5. Akurasi ROI

Waktu Pengujian	Raspberry Pi3	Odroid XU 4
pagi	82.83%	97.19%
siang	71.84%	94.68%
sore	78.19%	93.50%
malam	64.88%	63.97%
Rata-rata akurasi	74.44%	87.33%

4. KESIMPULAN

Berdasarkan hasil dari penelitian yang telah dilakukan dapat diambil kesimpulan sebagai berikut:

Penerapan proses ROI dapat digunakan sebagai solusi untuk mengatasi masalah waktu pemrosesan. Hal ini dibuktikan dengan pengujian yang dilakukan sebelum dan setelah pengimplementasian proses ROI. Hasil pengujian waktu pemrosessan menggunakan laptop sebelum pengimplementasian ROI memiliki waktu pemrosesan sebesar 35 ms sedangkan setelah diimplementasikannya proses ROI memiliki waktu pemrosesan sebesar 9,5 ms. Pada pengujian yang dilakukan menggunakan Raspberry Pi3 sebelum diimplementasikannya proses ROI memiliki waktu pemrosesan yaitu sebesar 248 ms sedangkan setelah diimplementasikannya proses ROI, pengujian menggunakan Raspberry Pi3 memiliki waktu pemrosesan sebesar 41 ms. Hasil

pengujian waktu pemrosesan penerapan ROI pada Odroid Xu4 sebesar 10 ms.

Dengan demikian dapat disimpulkan bahwa waktu pemrosesan dipengaruhi oleh luas bagian yang diproses serta jenis prosessor yang digunakan. Dengan demikian dibuktikan bahwa penerapan ROI efektif untuk mempercepat waktu pemrosesan pada system deteksi dan kalsifikasi kendaraan.

Implementasi ROI pada penelitian ini masih di lakukan dengan menggunakan proses cropping *frame* sehingga harus dilakukan proses proyeksi *frame* ROI pada *frame* asli. Proses proyeksi dilakukan karena koordinat *frame* asli dengan *frame* ROI berbeda. Selanjutnya pengembangan algoritma ROI dapat dilakukan tanpa harus melakukan proyeksi sehingga proses dapat dilakukan dalam satu *frame*. Selain itu impementasi dari proses ROI dapat dilakukan secara otomatis.

DAFTAR PUSTAKA

ADISTYA, R, dan MUSLIM, M A. 2016. "Deteksi Dan Klasifikasi Kendaraan Menggunakan Algoritma Backpropagation Dan Sobel." *Journal of Mechanical Engineering and Mechatronics* 1 (2): 65–73.

ALAMSYAH, D. 2017. "Pengenalan Mobil pada Citra Digital Menggunakan HOG-SVM," JURNAL TEKNIK INFORMATIKA DAN SISTEM INFORMASI.

COIFMAN, B. DAVID, B. PHILIP, M. dan JITENDRA, M. 1998."A Real-Time Computer Vision System for Vehicle Tracking and Traffic Surveillance." *Transportation Research Part C: Emerging Technologies* 6 (4): 271–88. https://doi.org/10.1016/S0968-090X(98)00019-9.

FAJRIN, H.R. 2016. "Perbandingan Metode Untuk Perbaikan Kualitas Citra Mammogram." *Jurnal SIMETRIS* 7 (2): 657–64.

FALAH, R.F., NURHAYATI, O.D. dan MARTONO, K.T. 2016."Aplikasi Pendeteksi Kualitas Daging Menggunakan Segmentasi *Region of Interest* Berbasis *Mobile*." *Jurnal Teknologi dan Sistem Komputer* 4 (2): 333-343

HAN, S. dan NUNO, V. 2008. "Object-Based Regions of Interest for Image Compression." *Data Compression Conference Proceedings*, 132–41. https://doi.org/10.1109/DCC.2008.94.

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

LINDA, A.S. 2010. "Penerapan *Region Of Interest* (ROI) Pada Metode Kompresi." *Penerapan Region Of Interest (ROI) Pda Metode Kompre*, 1–14. https://doi.org/10.14710/JTSISKOM.3.2.2015.320-334.

MA, L, BIN, X., FUKUN, B., HE, C, dan YING, Y. 2015. "*Region Of Interest*s Extraction Based on," 1–4.

MU'ARIFAH, S. 2018. "Internet of Things (Iot) Untuk Menghitung Dan Mengklasifikasi Jenis Kendaraan Bermotor Berbasis Pengolahan Citra Digital Tugas Akhir."

SOGEN, M., DWIYANTO, T., dan TUBAGUS, M.K. 2014. "Computer Vision." https://doi.org/10.1007/978-0-387-31439-6.

WICAKSONO, D.W. 2017. "Pengembangan Sistem Estimasi Kecepatan Pada Kendaraan Bergerak Berbasis Pengolahan Citra Digital," 127. http://repository.its.ac.id/2054/.

IMPLEMENTASI ALGORITMA REGION OF INTEREST (ROI) UNTUK MENINGKATKAN PERFORMA ALGORITMA DETEKSI DAN KLASIFIKASI KENDARAAN

ORIGINALITY REPORT

7%	6%	1%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.scribd.com Internet Source	3%
2	Submitted to Sriwijaya University Student Paper	2%
3	Submitted to Universiti Teknikal Malaysia Melaka Student Paper	2%
4	media.neliti.com Internet Source	1%

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		