PERANCANGAN SISTEM PENGHITUNG JUMLAH KENDARAAN PADA AREA PARKIR DENGAN METODE *BACKGROUND SUBTRACTION* BERBASIS *INTERNET OF THINGS*

Monica Sari Hariyanto[*)], Aghus Sofwan, dan Achmad Hidayatno

Departemen Teknik Elektro, Universitas Diponegoro
Jl. Prof. Sudharto, SH, Kampus UNDIP Tembalang, Semarang 50275, Indonesia

[*)]*E-mail: monicasari57@@gmail.com*

Abstrak

Kemajuan teknologi yang berkembang pesat dan semakin meluas saat ini tidak dapat dicegah. Hal ini sejalan dengan perkembangan penggunaan internet untuk mencakup kebuutuhan sehari hari. Salah satu implementasi yang berkembang di masyarakat ialah aplikasi *smart city*. Dalam konsep pengembangan yang lebih kecil, *smart city* dapat di terapkan pada sebuah kampus guna memudahkan sistem manajemen kampus yang disebut *smart campus* atau kampus pintar. *Smart campus* memiliki berbagai macam fitur yang diterapkan, salah satunya *smart open parkir*. Fitur tersebut digunakan untuk menganalisa jumlah kendaraan yang keluar masuk pada area parkir terbuka di sekitar kampus. Masalah yang sedang dihadapi pada area parkir tebuka adalah sistem penghitungan jumlah kendaraan yang terparkir yang masih dilakukan secara manual, sehingga masyarakat kampus kesulitan untuk menentukan tempat parkir kendaraannya. Dalam penelitian ini dirancang suatu sistem yang dapat menghitung jumlah kendaraan yang melintasi area parkir tebuka. Masukan sistem berupa video yang diperoleh dari kamera. Metode deteksi gerakan yang digunakan adalah background subtraction dengan algoritma gaussian mixture model. Sistem ini menggunakan Raspberry Pi sebagai komputer mini yang diaktifkan dengan pemberian catu daya yang bersumber dari panel surya. Adanya raspberry pi beserta komponen-komponen pelengkapnya seperti kamera sebagai media pengambilan video yang dimonitoring secara otomatis merupakan ciri dari IoT.

***Kata kunci**: background subtraction, Raspberry Pi, Internet of Things*

Abstract

Technological advances are growing rapidly and are increasingly widespread at this time can not be prevented. This is in accordance with the development of internet usage for daily needs. One implementation that has developed in the community is the smart city application. In a smaller development concept, smart city can be applied to a campus to facilitate the campus management system called smart campus or smart campus. Smart campus has a variety of features that can be applied, one of them is smart open parking. This feature is used to analyze the number of vehicles in and out of the open parking area around the campus. The problem that is often encountered in the open parking area is the system of calculating the number of parked vehicles that are still done manually, so that the campus community has difficulty determining the parking space of the vehicle. In this study a system that can calculate the number of vehicles that cross the open parking area is designed. Input system is a videos obtained from the camera. The motion detection method used is background subtraction with a gaussian mixture model algorithm. This system uses Raspberry Pi as a mini computer that is activated by providing a power supply sourced from solar panels. The presence of raspberry pi and its complementary components such as a camera as a video monitoring media that is automatically monitored is a feature of IoT.

Keywords : background subtraction, Raspberry Pi, Internet of Things

1. Pendahuluan

Kemajuan teknologi yang terus berkembang dengan pesat saat ini membuat banyak macam program untuk membantu mengembangkan produk berbasis *Internet of Things (IoT)*. Salah satu contoh bentuk pengembangan produk berbasis *Internet of Things (IoT)* ialah *smart city*. *Smart city* didefinisikan sebagai sebuah konsep pengembangan dan pengelolaan kota dengan pemanfaatan teknologi informasi dan komunikasi (TIK) untuk menghubungkan, memonitor, dan mengendalikan berbagai sumber daya yang ada di dalam kota dengan lebih efektif dan efisien untuk memaksimalkan pelayanan kepada warganya serta mendukung pembangunan yang berkelanjutan [1].

Dalam konsep pengembangan teknologi yang lebih kecil, *smart city* dapat diterapkan di dalam sebuah kampus yang disebut *smart campus* atau kampus pintar. *Smart campus*

merupakan sebuah konsep kampus yang menerapkan dan memadukan sistem pembelajaran, manajemen kampus, perpustakaan dan lain sebagainya dengan penggunaan teknologi informasi. Salah satu fitur yang ditawarkan dari *smart campus* adalah *smart open parking*. Fitur tersebut digunakan untuk menganalisa jumlah kendaraan yang keluar masuk pada area parkir terbuka di sekitar kampus. Konsep dari *smart campus* adalah intergrasi data ke internet dan akses data dari internet, berbasis *reporting* maupun *update feed*. *Updating data* tersebut bisa berasal dari informasi yang diberikan oleh masyarakat maupun perangkat mesin otonom melalui komunikasi internet dalam arsitektur *Internet of Things* (IoT). Arsitektur ini memungkinkan *updating* informasi secara *real time* dari berbagai lokasi dengan menggunakan koneksi internet.

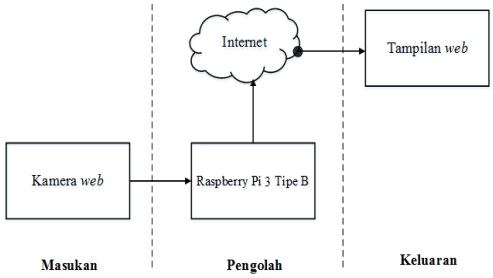
Terdapat beberapa Penelitian yang telah dilakukan sebelumnnya mengenai sistem pendeteksi objek. F. A. Surur [2] merancang sistem pengolahan citra yang diterapkan pada proses pendeteksian kendaraan dengan menggunakan salah satu metode pengolahan citra yaitu metode *gaussian mixture model* yang digunakan untuk menghitung *traffic* jumlah kendaraan yang melintasi jalan yang terekam pada kamera. Da Li, dkk[3] dan Wen-Jun Wang, dkk[4] merancang sistem yang dapat mendeteksi dan menghitung kendaraan menggunakan metode *background subtraction* dengan OpenCV. Metode tersebut cukup baik bila diterapkan untuk mendeteksi objek bergerak dengan latar diam, namun kurang akurat untuk mendeteksi objek bergerak bila pada latar terdapat gerakan atau derau. M. Harry Bintang [5] merancang aplikasi deteksi gerakan manusia yang terekam pada kamera keamanan menggunakan metode abackground subtraction dengan algoritma *gaussian mixture model*.

Pada Penelitian ini merancang sebuah sistem dalam fitur *smart campus* yaitu *smart open parking* yang diterapkan di Universitas Diponegoro. Masukan yang digunakan adalah sebuah video yang diambil menggunakan kamera digital yang ditempatkan di pintu masuk area parkir terbuka. Penghitungan jumlah kendaraan yang lewat diklasifikasikan menjadi dua jenis kendaraan yaitu mobil dan motor yang dilakukan dengan menggunakan metode *background subtraction*. Proses deteksi objek bergerak dengan metode *background subtraction* didasarkan pada perbedaan antara *background* referensi dengan *frame*. Hasil yang diperoleh dari sistem ini adalah pemberitahuan mengenai jumlah kendaraan yang terparkir pada area parkir, guna memudahkan pengguna untuk mencari area parkir yang tersedia di wilayah Universitas Diponegoro.

2. Metode

2.1. Blok Diagram Sistem

Sistem yang dirancang adalah sebuah alat untuk mendeteksi jumlah kendaraan yang keluar masuk area parkir Departemen Teknik Elektro UNDIP dengan mengklasifikasikan dua jenis kendaraan yaitu mobil dan motor. Deteksi trafik yang diterapkan memakai metode *background subtraction* dengan algoritma *gaussian mixture model* (GMM), fokus penelitian pada perbandingan laju pembelajaran (*learning rate*) dan perbandingan *region of interest* (ROI). Kamera mendeteksi adanya gerakan pada *region of interest* (ROI). Setiap kendaraan yang melewati ROI kemudian dihitung jumlahnya. Blok diagram dapat dilihat pada Gambar 1. Hasil dari proses perhitungan masuk ke *log* dalam ekstensi .txt. Selain masuk ke *log*, hasil dari proses perhitungan diunggah ke basis data di internet.



Gambar 1. Blok diagram sistem

2.2. Bagian Masukan

Kamera akan merekam keadaan suatu lokasi lalu mengirimkan rekaman tersebut ke Raspberry Pi. Kamera web ini memiliki *port* keluaran berupa USB (*Universal Serial Bus*) yang disambungkan ke Raspberry Pi.

2.3. Bagian Pengolah

Raspberry Pi adalah komponen yang bertanggung jawab sebagai pengolah [6]. Blok pengolah ini melakukan lima proses, yakni proses deteksi kendaraan, klasifikasi jenis kendaraan, penghitung jumlah kendaraan, *log* kendaraan, dan sinkronisasi data. Video yang direkam kamera kemudian akan diproses pertama kali oleh pendeteksi gerak kendaraan. Jika gerak kendaraan yang dideteksi masuk ke dalam *Region of Interest* (ROI) maka *counter* akan mencatat hitungan jumlah kendaraan. Terdapat 2 wilayah perhitungan yang digunakan dalam sistem, yaitu wilayah untuk mendeteksi kendaraan masuk (+) dan keluar(-).

2.3.1. Proses Deteksi Gerak

Proses deteksi gerak berfungsi untuk mendeteksi objek bergerak dari video yang direkam oleh kamera. Deteksi gerak menggunakan fungsi yang terdapat pada library OpenCV, melalui API (Application Programming Interface) C++ bernama cv2.BackgroundSubtractorMOG [7]. Fungsi tersebut memiliki parameter diantaranya history, varThreshold, dan bShadowDetection. Parameter history berfungsi untuk menetapkan jumlah frame terakhir yang memengaruhi model latar belakang. Parameter varThreshold berfungsi untuk mengatur nilai ambang pada

background subtraction. Parameter bShadowDetection berfungsi untuk mendeteksi bayangan pada video. Parameter lain yang harus diatur, yakni learningRate. Parameter learningRate mengatur nilai laju pembelajaran yang akan digunakan. Pada perancangan ini, parameter learningRate akan diatur sesuai dengan kondisi lingkungan. Nilai parameter learningRate pada kondisi lingkungan sistem akan ditentukan dari hasil pengujian variasi parameter learningRate.

2.3.2. Proses Klasifikasi Jenis Kendaraan

Proses klasifikasi jenis kendaraan digunakan untuk membedakaan jenis kendaraan yang terdeteksi mobil atau motor. Klasifikasi jenis kendaraan menggunakan data koordinat nilai tinggi dan data koordinat nilai lebar untuk menentukan ukuran kendaraan yang terdeteksi oleh sistem.

2.3.3. Proses Penghitung Jumlah Kendaraan

Proses penghitung jumlah kendaraan menggunakan sistem ROI (*Region of Interest*). Ada dua ROI yang digunakan dalam sistem ini, yaitu ROI untuk mendeteksi kendaraan yang masuk dan ROI untuk mendeteksi kendaraan yang keluar area parkir tebuka. Jika *centroid* pada *bounding box* masuk ke dalam ROI, maka kendaraan dihitung 1 sebagai kendaraan masuk dan -1 untuk kendaran yang keluar. *Centroid* yang sama akan dideteksi tiap *frame*nya untuk menghindari kendaraan dihitung berkali-kali.

2.3.4. Proses Log Kendaraan

Proses *log* kendaraan menggunakan *library* datetimeuntuk dapat beroprasi. Proses pertama *log* kendaraan yaitu dengan mengatur waktu data masuk *log*. Waktu diinisialisasi dalam variabel waktu_lalu dan waktu_skrg. Jika nilai pada variabel waktu_skrg habis dibagi 30 (30 detik) maka data akan dikirim ke bagian pengiriman basis data dan dicatat datanya dalam *log*. Penghitung juga akan diatur ulang menjadi 0 (counter = 0) setelah 30 detik.

2.3.5. Proses Sinkronisasi Data

Proses pengirirman data dalam sistem dijalankan oleh file PHP di internet [8]. Proses yang berlangsung pada Raspberry Pi adalah untuk sinkronisasi antara file PHP dengan program python. Data yang disimpan dalam variabel counter akan dilanjutkan ke sinkronisasi data pada file php setelah 30 detik.

2.4. Bagian Keluaran

Bagian keluaran berupa tampilan web yang dirancang dengan HTML dan PHP. File HTML sebagai tampilan web, sedangkan file PHP sebagai koneksi antara basis data dengan tampilan web. Basis data dibuat dengan menggunakan PHPMyAdmin pada dstp.puskom.undip.ac.id.

2.5. Bagian Penghubung Basis Data dengan Python

Bagian penghubung antara basis data dengan Python dirancang agar komputasi pada Raspberry Pi tidak berat. Bagian ini merupakan berkas PHP dengan nama cek2.php. Berkas cek2.php berisi *query*untuk memasukkan data ke basis data.

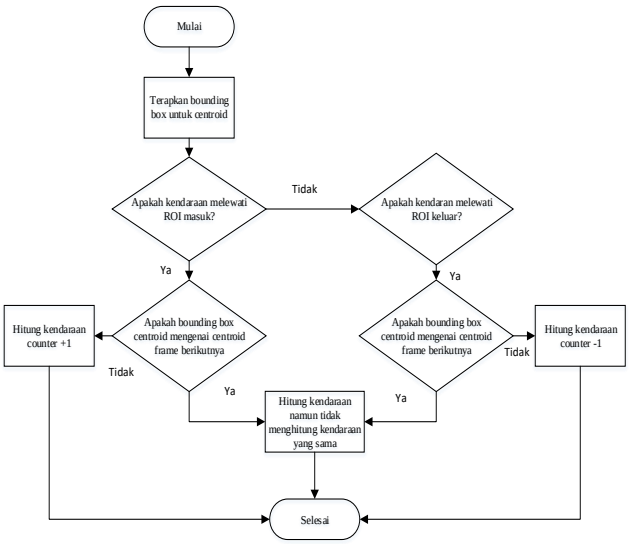
2.6. Bagian Penghubung Basis Data dengan Python

Bagian penghubung antara basis data dengan Python dirancang agar komputasi pada Raspberry Pi tidak berat. Bagian ini merupakan berkas PHP dengan nama cek.php. Berkas cek.php berisi *query* untuk memasukkan data ke basis data.

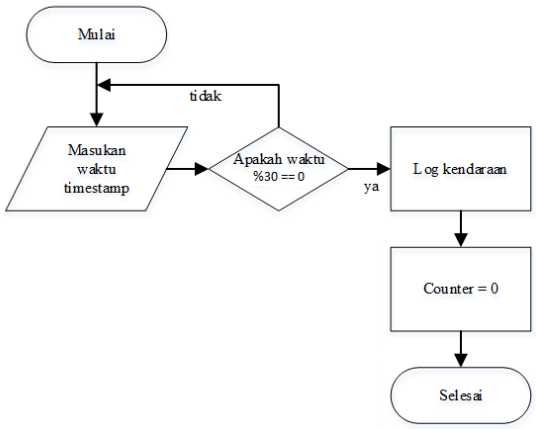
2.7. Perancangan Perangkat Lunak

Diagram alir pengujian deteksi ditunjukkan pada Gambar 2. Tahap pertama adalah ada masukan berupa video. Masukan video kemudian melewati proses *background subtraction*. Masukan video yang telah melalui proses *background subtraction* kemudian menentukan laju pembelajaran yang digunakan. Setelahnya, maka kendaraan telah terdeteksi. Proses selanjutnya adalah perhitungan jumlah kendaraan. Dalam perhitungan jumlah kendaraan *Region of Interest* (ROI) diterapkan dalam *frame*. Hasil dari kendaraan yang terdeteksi tiap *frame* ditambah dengan bounding box dan centroid. Setiap centroid menyentuh ROI maka kendaraan dihitung satu (counter + 1).

Proses selanjutnya adalah masuk pada *decision*, jika sisa bagi waktu dengan 30 (menandakan 30 detik) adalah nol, maka alur sistem berlanjut ke proses sinkronisasi basis data. Jika sisa bagi bukan nol, maka sistem kembali ke penerapan *background subtraction* ditunjukkan pada Gambar 3. Proses sinkronisasi basis data dilanjutkan ke pencatatan *log* jumlah kendaraan tiap 5 detik. Setelah *log* dicatat maka perhitungan jumlah kendaraan diatur kembali menjadi nol (counter = 0). Terdapat decision setelah atur ulang jumlah kendaraan, yaitu *break*. Tombol untuk melakukan *break* pada kibor adalah ‘Q’ yang akan menghentikan program sehingga alur selesai. Jika tombol tidak ditekan maka alur akan diulang dari penerapan *background subtraction*.



Gambar 2. Diagram alir fungsi pengolah data



Gambar 3. Diagram alir fungsi *log* kendaraan

2.8. Perancangan Perangkat Keras

Pada rangkaian alat, Raspberry Pi diberi catu daya yang bersumber dari panel surya yang dihubungkan dengan baterai dan disambungkan pada kamera . Perangkat keras yang telah dirangkai tersebut perlu dipasang pada sebuah tiang tempat pengambilan data, agar titik pengambilan video dapat lebih tinggi dan area jalan yang terekam dapat terlihat dengan jelas. . Perangkat keras yang sudah dirangkai pada tiang ditunjukkan pada Gambar 4.

Gambar 4. Rangkaian keseluruhan alat

3. Hasil dan Analisis

Masukan yang digunakan adalah video dengan durasi berbeda yang direkam dengan laju *frame* sebanyak 24 *frame* per detik. Posisi kamera tidak berubah pada berbagai pengujian. Seluruh pengujian dilakukan pada pagi pukul 08.00 sampai pukul 21.00. Hasil dari pengujian digunakan untuk menghitung tingkat keberhasilan dengan rumus $100\% - \left(\frac{Selisih}{Jumlah\ hitung\ manual}\right)$. Pengujian yang dilakukan dengan variasi beberapa parameter, yaitu:

1. Laju Pembelajaran
2. *Region of Interest* (ROI).

Program yang dirancang menggunakan metode .bashrcuntuk membuat sistem autostart. Bashrc adalah berkasyang mengatur tampilan serta beberapa perintah dari *console terminal*. Pada berkas bashrc perintah untuk menjalankan program dimasukkan. Apabila perintah sudah dimasukkan ke dalam berkas bashrc, maka perintah tersebut akan langsung dieksekusi oleh sistem sesaat setelah perangkat Raspberry Pi dinyalakan.

3.1. Pengujian Deteksi Gerak

Pengujian deteksi gerak bertujuan untuk mencari parameter laju pembelajaran yang sesuai untuk diterapkan pada kondisi lingkunngan sistem. Nilai laju pembelajaran ditentukan berdasarkan riwayat *frame*. Terdapat 4 variasi riwayat *frame* yang akan digunakan, yakni 100 *frame*, 200 *frame*, 300 *frame*, dan 400 *frame*. Sehingga variasi nilai laju pembelajaran yang akan digunakan adalah:

1. Riwayat *frame* = 100
   Laju pembelajaran = $\frac{\mathbf{1}}{\mathbf{riwayat}\ \boldsymbol{frame}} = \frac{\mathbf{1}}{\mathbf{100}} = \mathbf{0,01}$
2. Riwayat *frame* = 200
   Laju pembelajaran = $\frac{\mathbf{1}}{\mathbf{riwayat}\ \boldsymbol{frame}} = \frac{\mathbf{1}}{\mathbf{200}} = \mathbf{0,005}$
3. Riwayat *frame* = 300
   Laju pembelajaran= $\frac{\mathbf{1}}{\mathbf{riwayat}\ \boldsymbol{frame}} = \frac{\mathbf{1}}{\mathbf{300}} = \mathbf{0,0033}$
4. Riwayat *frame* = 400

$$\text{Laju pembelajaran}=\frac{1}{\text{riwayat } frame}=\frac{1}{400}=0{,}0025$$

3.1.1. Pengujian Variasi Laju Pembelajaran

Tabel 1. Hasil pengujian deteksi gerak dengan parameter laju pembelajaran

No. *frme*	*Frame* asli	0.01	0.005	0.0033	0.0025
20					
80					
150					
250					
350					

Berdasarkan Tabel 1. dapat dilihat terdapat perbedaan yang cukup jelas dari hasil deteksi gerak dengan parameter laju pembelajaran yang berbeda. Pada awal waktu saat tidak ada objek bergerak, semua laju pembelajaran akan menganggap ada objek bergerak. Laju pembelajaran akan memperbarui latar belakang setelah beberapa waktu. Latar belakang tersebut seharusnya ditandai dengan warna hitam karena tidak ada objek bergerak. Tiap laju pembelajaran memiliki waktu yang berbeda dalam memperbarui latar belakang. Selain itu, pada kondisi luar ruangan terdapat perubahan pencahayaan serta perubahan sesaat dari latar belakang, seperti dedaunan dan rumput akibat tertiup angin. Kedua hal tersebut mempengaruhi kinerja deteksi gerak. Saat ada objek bergerak, *foreground mask* yang dihasilkan masing-masing laju pembelajaran memiliki kinerja yang berbeda. Tiap variasi laju pembelajaran menghasilkan *foreground mask* yang menandai objek bergerak. Pada keadaan tidak ada objek bergerak, tidak ada *foreground mask* yang muncul pada masing-masing laju pembelajaran..

Tabel 2. Hasil pengujian pengaruh variasi laju pembelajaran dengan jumlah *frame* yang diproses.

Waktu (detik)	Masukan (jumlah *frame*)	Keluaran (jumlah *frame* tiap variasi laju pembelajaran)			
		0.01	0.005	0.0033	0.0025
4	96	39	73	90	91
8	192	53	89	118	155
12	288	61	91	130	168
16	384	70	105	159	191
20	480	97	118	176	215
24	576	106	152	201	225

Berdasarkan Tabel 2., Selama 24 detik, *frame* masukan yang akan diproses sebanyak 576 *frame*. Laju pembelajaran yang berbeda menghasilkan *frame* keluaran dengan jumlah yang berbeda. Pada laju pembelajaran 0,01, menghasilkan 106 *frame*. Laju pembelajaran 0,005, menghasilkan 152 *frame*. Laju pembelajaran 0,0033, menghasilkan 201 *frame*. Laju pembelajaran 0,0025, menghasilkan 225 *frame*. Pada awal waktu, sudah terdapat perbedaan jumlah *frame* keluaran yang jelas dari masing-masing parameter laju pembelajaran. Pada detik ke-4, laju pembelajaran 0,01 menghasilkan 28 *frame*. Jumlah tersebut merupakan yang paling sedikit bila dibandingkan dengan laju pembelajaran lain pada waktu yang sama.

3.2. Pengujian dengan Variasi ROI

Pada Penelitian ini dilakukan pengujian hasil perancangan pada sistem. Parameter pertama yang diuji adalah variasi ROI. Terdapat tiga variasi ROI yang akan digunakan, yaitu variasi pertama ($180 \leq cx < 280$) untuk area keluar dan ($280 \leq cx \leq 380$) untuk area masuk, variasi ke-2 ($220 \leq cx < 280$) untuk area keluar dan ($280 \leq cx \leq 340$) untuk area masuk, dan variasi ke-3 ($250 \leq cx < 280$) untuk area keluar ($280 \leq cx \leq 310$) untuk area masuk yang ditunjukan pada Gambar 5.

(a)

(b)

(c)

Gambar 5. (a) variasi ROI (180 ≤ cx <280) dan (280 ≤ cx ≤ 380)
(b) variasi ROI (220 ≤ cx < 280) dan (280 ≤ cx ≤ 340)
(c) variasi ROI (250 ≤ cx < 280) dan (280 ≤ cx ≤ 310)

Tabel 3. Pengujian data dengan variasi ROI

Variasi ROI		Perhitungan			Keses uaian
Keluar	Masuk	+	-	=	
Manual	Manual	1	-1	0	Sesuai
180 ≤ cx < 280	280 ≤ cx ≤ 380	1	-2	-1	Tidak Sesuai
220 ≤ cx < 280	280 ≤ cx ≤ 340	1	-1	0	Sesuai
250 ≤cx < 280	280 ≤ cx ≤ 310	0	-1	-1	Tidak Sesuai

Dari data Tabel 3. diatas dapat dilihat bahwa perhitungan jumlah kendaraan dengan menggunakan variasi ROI 180 ≤ cx ≤ 280 ≤ cx ≤ 380 menghasilkan pendeteksian dengan jumlah -1. Variasi ROI 220 ≤ cx ≤ 280 ≤ cx ≤ 340 menghasilkan pendeteksian dengan jumlah 0. Variasi ROI *250* ≤ cx ≤ 280 ≤ cx ≤ 310 menghasilkan pendeteksian dengan jumlah -1. Hasil data dari variasi ROI 220 ≤ cx ≤ 280 ≤ cx ≤ 340 tersebut sudah sesuai dengan penjumlahan manual. Luasan ROI yang tidak terlalu lebar dan terlalu kecil akan memudahkan program dalam mendeteksi dan akan mengurangi kesalahan dalam pendeteksian program.

3.3. Perbandingan Hasil Data

Pengambilan data dilakukan selama 4 hari, pengujian perbandingan data dilakukan menggunakan parameter yang telah ditetapkan seperti parameter laju pembelajaran dan ROI guna membandingkan data yang diperoleh, dan juga perbandingan data yang dilakukakan pada setiap jam.

3.3.1. Perhitungan Data Per Hari

Hasil pengujian perbandingan data dilakukan di area parkir Departemen Teknik Elektro UNDIP. Data yang dibandingkan merupakan data perhitungan sistem pada jam 14.00 WIB. Hasil dari pengujian digunakan untuk menghitung tingkat keberhasilan dengan rumus $100\% - \left(\frac{Selisih}{Jumlah\ hitung\ manual}\right)$. Berikut hasil pengujian perbandingan sistem :

Tabel 4. Tabel hasil perbandingan Perhitungan Kendaraan

Hari ke-	Jenis	Sistem	Manual	Total Kendaraan +	Total Kendaraan -	Selisih	Akurasi
1	Mobil	28	26	29	-3	2	92,30%
	Motor	63	56	67	-11	7	87,5%
2	Mobil	15	16	23	-8	1	93,75%
	Motor	54	48	61	-13	6	87,5%
3	Mobil	19	18	28	-10	1	94,11%
	Motor	44	39	54	-15	5	87,17%
4	Mobil	26	23	27	-7	3	86,95%
	Motor	45	43	63	-18	2	95,34%

Dari data Tabel 4. diatas terlihat bahwa selisih paling sedikit adalah 1 unit untuk kendaran mobil dan 7 unit untuk kendaran motor. Selisih tersebut didapatkan dengan menggunakan parameter ROI 220 ≤ cx ≤ 280 ≤ cx ≤ 340 dengan laju pembelajaran 0.01 dan frame rate 24. Rata-rata akurasi yang di dapat dari keseluruhan data adalah 90,57%. Selisih kendaraan disebabkan oleh adanya tumpang tindih Bounding box kendaran dan tingginya kecepatan kendaran sehingga mempengaruhi output jumlah kendaraan. Adanya Bounding box yang tindih menyebabkan deteksi dua kendaraan atau lebih sebagai satu kendaraan atau sebaliknya satu kendaraan terdeteksi oleh dua atau lebih Bounding box. Kecepatan kendaraan juga berpengaruh karena semakin cepat kendaraan, ada kemungkinan kendaraan tidak melewati ROI. Frame video yang dihasilkan setelah proses di Raspberry Pi menyebabkan video lag sehingga ROI tidak dapat menangkap citra latar depan kendaraan.

3.3.2. Perhitungan Data Per Jam

Parameter yang telah didapatkan dengan membandingkan data digunakan untuk mencari data perjamnya. Data per jam didapatkan dnegan mengambil data pada satu hari dengan durasi 08.00 sampai jam 21.00 WIB. Data yang dihitung adalah data kendaraan yang berada pada area parkir tiap dua jam. Setiap dua jam data di bandingkan antara perhitungan manual dengan sistem. Data di ambil pada tanggal 20 Juli 2018. Data yang didapat terdapat pada Tabel 5. berikut :

Tabel 5. Perbandingan Perhitungan Data Per Jam

Waktu	Jenis	Sistem	Manual	Selisih	Akurasi
08.00 – 10.00	Mobil	6	6	0	100%
	Motor	24	25	1	96%
10.00 – 12.00	Mobil	22	19	3	84,21%
	Motor	43	38	5	86,84%
12.00 – 14.00	Mobil	26	23	3	86,95%
	Motor	45	43	2	95,34%
14.00 – 16.00	Mobil	17	15	2	86,67%
	Motor	39	35	4	88.57%
16.00 – 18.00	Mobil	6	5	1	80%
	Motor	30	28	2	92,85%
18.00 – 20.00	Mobil	4	3	1	66%
	Motor	16	11	5	54,3%
20.00 – 21.00	Mobil	4	2	2	-
	Motor	16	9	7	22%

Data yang didapatkan memiliki selisih yang berbeda-beda. Nilai akurasi tertinggi terdapat pada jam 08.00 – 10.00 yaitu 100% untuk mobil dan 96% untuk motor, sedangkan akurasi terendah pada jam 20.00 – 21.00 yaitu 0% untuk mobil dan 22% untuk motor. Selisih kendaraan disebabkan kurangnya penerangan pada malam hari yang mengakibatkan program tidak dapat mendeteksi adanya kendaraan yang melintas adanya tumpang tindih *bounding box* kendaraan dan tingginya kecepatan kendaraan sehingga memengaruhi output jumlah kendaraan. Keakuratan data juga dipengaruhi oleh komputasi citra Raspberry Pi. Komputasi pengolahan citra pada Raspberry Pi tidak memiliki kecepatan yang tinggi karena spesifikasinya, sehingga terdapat delay dan lag.

4. Kesimpulan

Berdasarkan pengujian yang telah dilakukan, sistem penghitung trafik kendaraan dapat berjalan dengan optimal menggunakan laju pembelajaran 0.01 dan ROI $220 \leq cx \leq 280 \leq cx \leq 340$ pada Raspbery Pi 3. Penerapan deteksi gerak dengan algoritma *gaussian mixture model* pada sistem trafik kendaraan mulai dari proses deteksi gerak, klasifikasi kendaraan penghitung kendaraan, *log*, dan sinkronisasi data telah berhasil berjalan dengan baik. Penerimaan data berjalan dengan lancar di basis data dstp.puskom.undip.ac.id. Keakuratan data dipengaruhi oleh komputasi citra Raspberry Pi, pengaruh lain adalah adanya tumpang tindih *bounding box* kendaraan dan kecepatan kendaraan juga disebabkan kondisi cahaya yang kurang memadai saat malam hari menyebabkan tidak adanya kendaraan yang terdeteksi saat melintas. Rata-rata akurasi tingkat keberhasilan pada pengujian data per hari adalah 90,57%, sedangkan pada pengujian per jam akurasi tertinggi terdapat pada jam 08.00 – 10.00 yaitu 100% untuk mobil dan 96% untuk motor.

Referensi

[1] S. Harso, *Pengenalan dan Pengembangan Smart City*, no. April. LPIK ITB dan Smart City Initiatives Forum, 2008.

[2] F. A. Surur, "Analisis Trafik Kendaraan Dengan Metode Background Subraction di Kampus UNDIP Berbasis Internet Of Things (IoT),"pp. 4-9, 2017.

[3] D. Li, B. Liang, and W. Zhang, "Real-time moving vehicle detection, tracking, and counting system implemented with OpenCV," *ICIST 2014 - Proc. 2014 4th IEEE Int. Conf. Inf. Sci. Technol.*, pp. 631–634, 2014.

[4] W. J. Wang and M. Gao, "Vehicle Detection and Counting in Traffic Video Based on OpenCV," *Appl. Mech. Mater.*, vol. 361–363, pp. 2232–2235, 2013.

[5] M. Harry Bintang, "Aplikasi Deteksi Gerak Pada Kamera Keamanan Menggunakan Metode Background Subtraction dengan Algoritma Gaussian Mixture Model," pp. 117–125, 2017.

[6] S. Shah, *Learning Raspberry Pi*. Packt Publishing Ltd, 2015.

[7] OpenCV, "OpenCV library." [Online]. Available: http://opencv.org/. [Accessed: 09-Apr-2017].

[8] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision , architectural elements , and future directions," *Futur. Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.