ANALISA ALGORITMA PENGHITUNG KENDARAAN RODA EMPAT DALAM KONDISI SIANG dan MALAM HARI DENGAN METODE *FRAME* INTERSECTION

FOUR WHEELED VEHICLE COUNTER ALGORHTHYM at DAY and NIGHT ANALYSIS USING FRAME INTERSECTION METHOD

Brillian Bagus Pakerti Utama[1], Ratri Dwi Atmaja S.T,M.T[2] , Azizah S.T,M.T[3]
[1,2,3]Prodi S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Universitas Telkom, Bandung
Email : [1]brillian@telkomuniversity.com, [2]ratridwiatmaja@telkomuniversity.com, [3]Azizah@telkomuniversity.com

Abstrak

Sistem penghitung kendaraan adalah sebuah sistem yang berguna untuk menghitung kendaraan yang lewat pada suatu jalan. Di berbagai tempat, penghitungan kendaraan masih bersifat *manual* dan terkadang masih mengalami kesalahan dalam penghitungan ketika penghitungan dilakukan pada kondisi malam hari. Sehingga dibutuhkan sebuah sistem yang dapat menghitung kendaraan secara otomatis dan mampu menghitung kendaraan tidak hanya dalam keadaan siang hari tetapi juga dalam kondisi malam hari.

Pada pengerjaan tugas akhir, sistem penghitung kendaraan dalam kondisi siang dan malam hari menggunakan metode *frame intersection* dengan mencari irisan antara *background frame* dan *next frame* dengan menggunakan *logic 'and'*, kemudian sistem menghitung jumlah piksel hitam dari hasil irisan dan membandingkannya dengan nilai dari parameter *Intersection Threshold*. Tetapi, sebelumnya sistem akan mengubah video menjadi beberapa *frame*.

Dari penelitian ini didapatkan tingkat akurasi untuk kondisi siang hari sebesar 78% dan untuk kondisi malam hari sebesar 70% dengan parameter *Intersection Threshold* = 7000 dan *Loop Threshold* = 0. Dari penelitian ini didapatkan metode untuk menghitung kendaraan malam hari dengan menggunakan cahaya lampu yang dilakukan proses dilasi setelahnya.

Kemudian, pada penelitian ini terdapat proses pengubahan dari format *Red*, *Green*,dan *Blue* menjadi *Hue*, *Saturation*,dan *Value*. Dengan tujuan sistem menghitung nilai rata-rata dari *layer Value Brightness* pada setiap *frame*, karena berdasarkan beberapa percobaan didapatkan nilai rata-rata *Value Brightness* pada kondisi siang hari dan malam hari memiliki selisih nilai yang sangat jauh.

Kata kunci : perhitungan kendaraan, *frame intersection,* backgroung *frame*, and next *frame*.dan *basic research*

Abstract

Vehicle computing system is a useful for counting vehicles which passing on a road. In many places, vehicle counting is still manual and sometimes still errors in calculations especially at night conditions. So, need a system that can calculate the vehicle automatically and the system able to calculate the vehicle not only at daylight conditions but also at night conditions.

In the final project, the system using the frame intersection with finding a wedge between the background frame and the next frame by using logic 'and', then the system calculates the number of black pixels from the wedge and compared with the value of the parameter Intersection threshold. Before that the system will convert the video into multiple frames.

From this study, the level of accuracy for daylight conditions at 78% and for the night conditions of 70% with Intersection parameter Threshold = 7000 and Loop Threshold = 0. From this study, the method to calculate the vehicle night using lights that do process dilation thereafter.

Then, in this study there is a process of changing the format from Red, Green, and Blue became Hue, Saturation, and Value. So, system calculates the average value of layer Value Brightness on every frame. Because, based on some experiments obtained average value of layer Value Brightness on conditions during the day and night have much difference.

Keywords : vehicle counting, frame intersection, background frame, next frame, and basic research

1. Pendahuluan

Sistem yang dirancang tidak hanya akan mengembangkan algoritma yang sudah dikerjakan pada penelitan sebelumnya mengenai sistem monitoring parkir tetapi sistem ini dapat digunakan sebagai solusi dari perhitungan kendaraan yang secara manual dengan menggunakan alat penghitung mekanik yang dioperasikan secara langsung oleh manusia. Sehingga, diharapkan sistem ini dapat menggantikan tugas manusia sebagai penghitung kendaraan dijalan yang terdakang mengalami permasalahan ketidakakuratan perhitungan karena keterbatasan penglihatan manusia. Berdasarkan beberapa permasalahan tersebut, pada tugas akhir ini dibuat

sebuah program pada sebuah *software* komputasi berbasis matriks yang akan mengitung banyaknya kendaraan pada sebuah gambar bergerak atau *video*, dengan pengambilan gambar atau *video* yang diambil dari atas suatu jalan dengan kondisi siang hari dan malam hari. Adapun rumusan masalah pada Tugas Akhir ini adalah membuat sistem perhitungan jumlah kendaraan berbasis video *processing*, membuat sistem nilai akurasi yang optimal, dan mengukur tingkat akurasi dari sistem yang akan dibuat sesuai harapan.

Adapun tujuan dari pembuatan tugas akhir ini adalah sebagi membuat suatu sistem yang dapat menghitung suatu kendaraan pada suatu *video* yang berguna sebagai input dari sistem tersebut, membuat suatu sistem perhitungan jumlah kendaraan pada malam hari dengan tingkat akurasi yang optimal sebesar 60%, dan menganalisis metode *Intersection Frame* pada sistem penghitung kendaraan. Beberapa batasan masalah pada tugas akhir ini agar tidak menyimpang dari permasalahan adalah *video* yang digunakan *Non Real Time*, kendaraan yang dihitung hanya pada satu lajur saja, Tugas akhir ini bersifat *Basic Research*, *video* yang digunakan menggambarkan jalan dengan jarak antar kendaraan tidak kurang dari 50 meter atau dalam kondisi lancar, dan pada kondisi malam hari, kondisi jalan hanya mendapatkan penerangan dari lampu mobil atau tanpa lampu jalan. Tahap-tahap dari metode ini adalah studi literatur, pengumpulan data, proses melakukan simulasi, proses pengujian dan analisis, dan pembuatan laporan.

2. Dasar Teori / Metode Penelitian / Perancangan Sistem

2.1 Citra Digital[7]

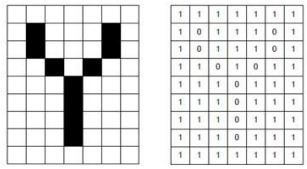
Citra *digital* merupakan suatu matriks dimana indeks baris dan kolomnya menyatakan suatu titik pada citra tersebut dan elemen matriksnya (yang disebut sebagai elemen gambar/ *pixel*/ piksel) yang menyatakan tingkat keabuan pada titik tersebut. Matrik yang dinyatakan citra *digital* yaitu dengan matriks berukuran N yang menyatakan banyaknya baris x M yang menyatakan banyaknya kolom.

N = jumlah baris 0 = y = N – 1
M = jumlah kolom 0 = x = M – 1
L = maksimal warna intensitas 0 = f(x,y) = L – 1

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \dots & f(1,M-1) \\ \vdots & \vdots & & \vdots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix} \quad (1.1)$$

2.2 Citra Biner

Dalam sebuah citra biner, setiap piksel hanya mempunyai dua kemungkinan dua nilai, seperti *on* dan *off*. Sebuah citra biner disimpan dalam matriks dengan nilai 0 dan 1. Sebuah citra biner disebut juga dengan citra yang hanya berisi hitam dan putih (BW).Sebuah citra biner dapat disimpan dengan tipe *double* atau *uint8*. Sebuah *array* bertipe *uint8* lebih banyak digunakan daripada *double*, karena tipe *uint8* menggunakan lebih sedikit memori[8].



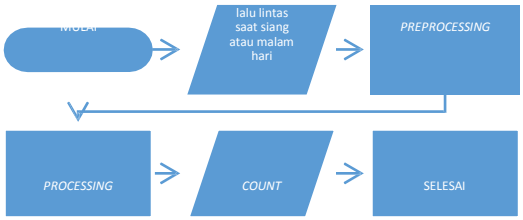
Gambar 2.1 Citra Biner

2.3 Citra HSV[9][10]

Model warna *Hue, Saturation, and Value* (HSV) memiliki definisi yang hampir sama dengan model warna *Hue, Saturation, and Lightness* (HSL). *Hue* adalah sebuah kombinasi dari warna-warna *Red, Green, and Blue*. *Saturation* menunjukan besarnya intensitas dari *Hue*, semakin cerah warna maka semakin besar pula nilai *saturation*. Sedangkan untuk warna hitam dan putih tidak memiliki nilai saturation. *Value* adalah sebuah nilia *brightness* dari suatu citra, semakin gelap suatu gambar maka semakin kecil nilai *value*nya.

Gambar 2.2 *HSV Cone*

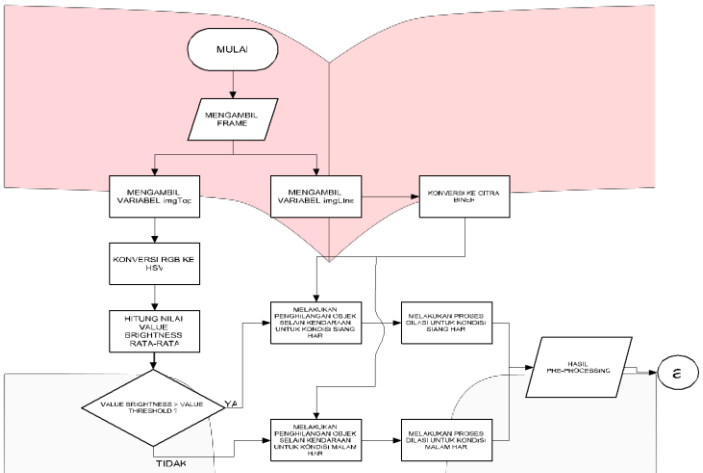
2.4 Model Sistem



Gambar 2.3 Diagram Alir Sistem

Dari diagram alir diatas, sistem secara garis besar memiliki dua proses utama, antara lain :

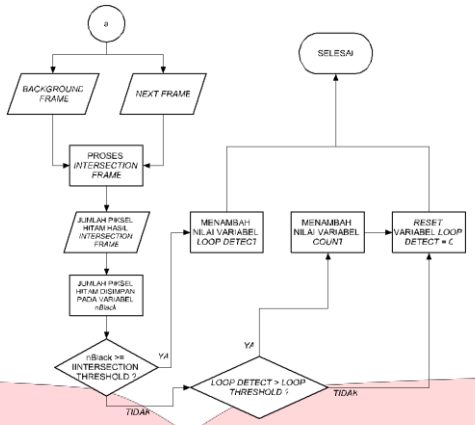
- *Pre-processing*



Gambar 2.4 Diagram Alir *Pre-Processing*

1. Proses inisiasi video
   Dalam proses ini, video yang berisi kondisi lalu lintas dengan format *colourspace Red, Green,and Blue*(RGB) diambil tiap layer warna merah, hijau,dan biru. Namun, jika video yang digunakan sebagai input adalah video dengan format YCbCr maka sebaiknya dilakukan konversi format *colourspace* ke dalam bentuk RGB lalu diubah lagi kebentuk HSV. Setelah itu, akan diambil *layer Value* saja yang memuat nilai *brightness* pada setiap piksel. Kemudian dilakukan perhitungan nilai rata-rata *layer Value*.
2. Penentuan *Pre-processing*
   Pada proses sebelumnya, nilai rata-rata *layer Value* telah didapatkan. Kemudian, hasil tersebut akan di tentukan apakah *frame* berada dalam kondisi siang hari ataukah malam hari. Jika, nilai rata-rata *Value* berada di bawah nilai *Value Threshold* maka *frame* tersebut akan terdeteksi dalam kondisi malam hari. Begitupun jika nilai rata-rata *Value* berada di atas nilai *Value threshold* maka *frame* tersebut terdeteksi dalam kondisi siang hari.
3. Citra Biner
   Frame-frame yang sudah di inisiasi dan ditaruh pada variabel *imgLine*, selanjutnya akan diubah menjadi citra biner agar mempermudah proses pengirisan antar *frame*. Objek yang terdeteksi sebagai kendaraan akan diberikan direpresentasikan sebagai piksel hitam, sedangkan untuk objek yang diidentifikasi selain kendaraan akan direpresentasikan sebagai piksel putih.

- *Processing*



Gambar 2.5 Diagram Alir *Processing*

Berikut adalah penjelasan dari proses perhitungan kendaraan :

1. Penentuan *Frame* Background
   Pada tahap ini sistem akan mengambil *frame* sebagai background apabila *frame* tersebut telah diproses pada loop proses sebelumnya, atau *frame* awal pemrosesan
2. Pengambilan *Frame* Baru
   Setelah *frame* background sudah di dapatkan, selanjutnya sistem akan mengambil *frame* yang baru dan menamainya sebagai *Next frame* dan akan menaruh *frame* pada *loop* sebelumnya sebagai *Background frame.*
3. Proses Pemisahan *Frame*
   Proses ini bermaksud untuk membagi suatu *frame* menjadi dua. Bagian pertama adalah hasil *crop* pada area tertentu dari sebuah *frame* yang berisikan kendaraan atau salah satu jalur jalan dan hasil tersebut akan dinamakan sebagai *imgLine*. Pada bagian kedua akan diambil area tertentu lainnya dari *frame* berupa bagian tengah hingga atas dari sebuah *frame* yang mengandung informasi apakah langit dalam kondisi siang atau malam, hasil dari proses tersebut akan disimpan pada variabel *imgTop*.
4. Konversi RGB ke HSV
   Kemudian setiap *frame* yang sudah diambil bagian dari tengah hingga atas akan dilakukan konversi dari format RGB ke HSV, setelah itu akan dilakukan proses perhitungan nilai rata-rata dari *Value Brightness*. Jika nilai rata-rata *Value Brightness* lebih besar dari *Value Threshold*, maka sistem akan menganggap *frame* tersebut berada pada kondisi siang hari, begitupun jika nilai rata-rata *Value Brightness* lebih kecil atau samadengan dari *Value Threshold*, maka sistem akan menganggap *frame* tersebut berada pada kondisi malam hari.
5. Intersection *Frame*
   Pada tahap ini *Next frame* dan *Background frame* akan dicari irisan nilainya pada setiap *piksel*. Dengan demikian akan didapatkan persamaan:

   |[illegible]| (2.1)

6. Perhitungan piksel hitam

   Setelah melalui proses *Frame Intersection* dari kedua *frame*, maka akan didapat hasil berupa *frame* irisan. Dalam *frame* ini berisi piksel-piksel yang berwana hitam. Selanjutnya, dari hasil tersebut akan dilakukan pada tahapan selanjutnya.
7. Penghitungan kendaraan
   Pada tahap ini, hasil dari tahap sebelumnya akan dibandingkan dengan *Intersection Threshold*, jika jumlah piksel hitam pada *frame* irisan kurang dari nilai *Intersection Threshold* selanjutnya akan dibandingkan lagi nilai dari variable *Loop Detect* dengan *Loop Threshold.* Perbandingan ini bertujuan untuk menunjukkan kekonsistenan dari syarat sebelumnya yang terpenuhi. *Loop Threshold* merupakan nilai minimum jumlah *frame* irisan yang telah memenuhi syarat pertama untuk kemudian disimpulkan bahwa memang terdapat objek pada *frame* selisih tersebut, sedangkan variabel *Loop Detect* merupakan nilai jumlah *frame* irisan yang telah memenuhi syarat pertama tersebut. Nilai *Loop Detect* akan terus bertambah jika *frame* irisan yang masuk selalu memenuhi syarat secara berturut-turut. Jika dalam proses penambahan nilai variabel *Loop Detect* yang

belum mencapai nilai *Loop Threshold* terdapat 1 *frame* irisan yang tidak memenuhi syarat, maka nilai variabel *Loop Detect* akan langsung berubah menjadi 0. Hal ini menunjukkan bahwa nilai *frame* irisan tidak konsisten.

3. Pembahasan

Berikut adalah spesifikasi dari perangkat keras maupun lunak yang digunakan dalam pengerjaan Tugas Akhir ini:

Perangkat Keras :

- Laptop : ASUS K46CB
- Prosesor : Intel® Core™ i7-3537U CPU @2.00 GHz (4CPUs), ~2.5Ghz
- Memori : 4098MB RAM
- VGA : NVIDIA GEFORCE 740M
- Kamera : Xiaomi Mi4i 1080p@30fps, 480p@120fps (5MP, f1/8)

Perangkat Lunak :

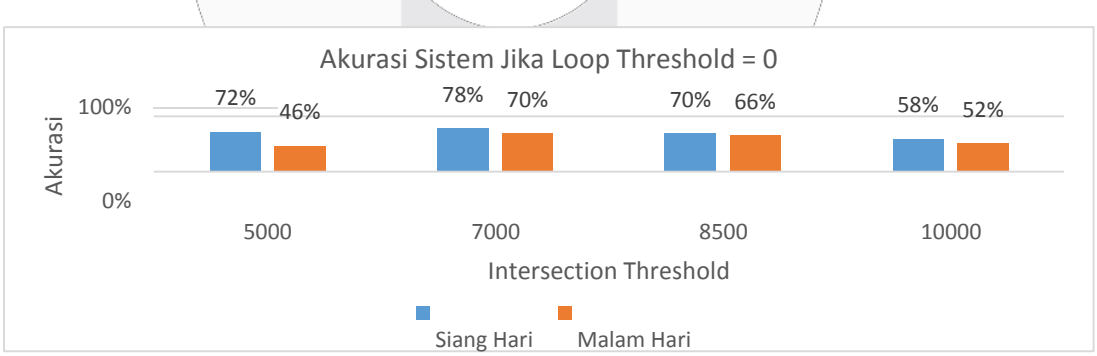
- Sistem Operasi : Windows 7 Ultimate 64bit
- MATLAB R2015a

Skenario pengujian dan analisis unjuk kerja sistem ini akan dilakukan sebanyak 3 skenario dengan video kendaraan pada siang hari dengan 24 *frames per second*, dan video kendaaraan pada malam hari dengan 14 *frames per second*, dimana kedua video kondisi tersebut sudah diambil tiap *frame*nya dan disimpan dalam beberapa folder sebanyak masing-masing 10 buah folder dan setiap folder terdapat 5 kendaraan yang lewat. Pada tahap *pre-processing* akan diambil bagian tengah hingga bagian atas dari *frame* sebesar 263x355 piksel. Kemudian akan diambil area tertentu dari sebuah *frame* yang berisikan kendaraan atau salah satu jalur jalan dan hasil tersebut ukuran sebesar 127 x 41 piksel.

Tabel 3.1 Parameter yang Digunakan pada Pengujian Sistem

Kondisi	*Value Threshold*	*Level Luminance*	Ukuran Strel Small Object	Ukuran Dilasi [Baris Kolom]
Siang Hari	0.02	0.4	10	[55 1]
Malam Hari		0.8	1	[1 45]

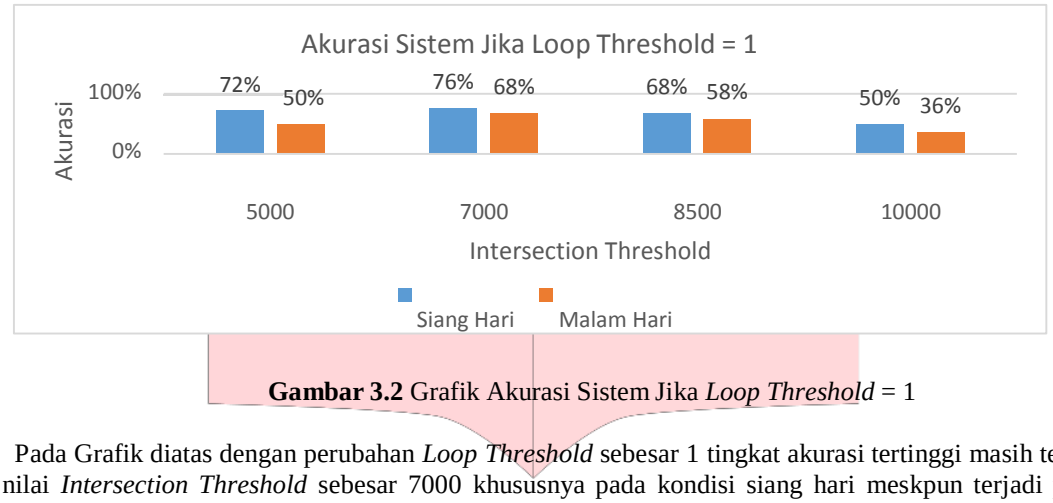
Setelah ditetapkan parameter pengujian seperti pada Tabel 3.1, kemudian akan dicari nilai akurasi sistem terhadap *Intersection Threshold* sebesar 5000, 7000, 8500, dan 10000 dengan merubah-rubah nilai parameter *Loop Threshold* sebesar 0, 1, dan 2.



Gambar 3.1 Grafik Akurasi Sistem Jika *Loop Threshold* = 0

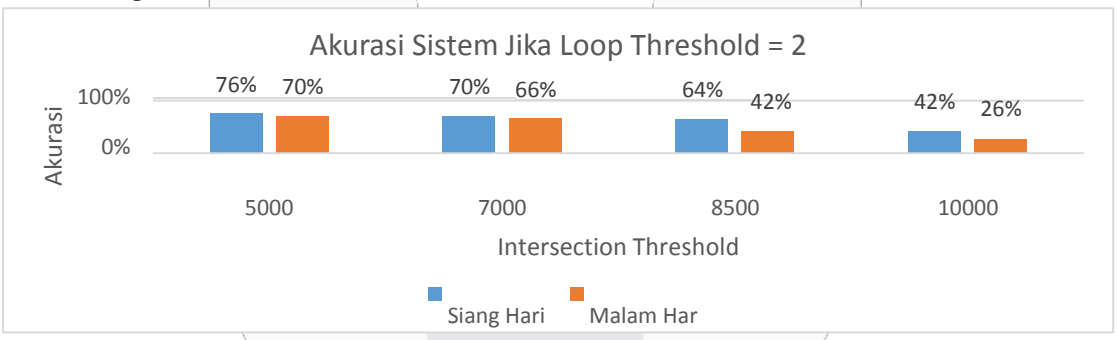
Pada grafik diatas didapatkan tingkat akurasi tertinggi pada kisaran nilai *Intersection Threshold* 7000 dan 8500 karena nilai dari jumlah piksel hitam pada data uji siang hari maupun malam hari berada pada nilai 5000 dan 9000, sehingga jumlah piksel hitam hasil proses intersection *frame* tidak berada jauh pada nilai tersebut. Dan oleh sebab itu juga, nilai akurasi cenderung lebih tinggi untuk kedua kondisi jika diberikan nilai *Intersection Threshold* dibawah 10000. Jika dilihat kembali grafik diatas, ketika nilai *Intersection Threshold* 5000, tingkat akurasi menurun menjadi sebesar 46% .Hal tersebut terjadi karena sistem mengalami kenaikan sensitivitas yang

tinggi khususnya pada kondisi malam hari dengan ditandai banyak objek kendaraan yang dideteksi lebih dari satu kali khususnya pada video ke-2 (malam hari). Kemudian, ketika nilai *Intersection Threshold* sebesar 10000 untuk siang dan malam hari penurunan tingkat akurasi terjadi karena terdapat objek/kendaraan pada masing-masing video yang tidak terdeteksi dan terhitung, bahkan pada kondisi malam hari pada video ke-1 dan ke-2 sama sekali tidak ada objek yang terdeteksi.



Gambar 3.2 Grafik Akurasi Sistem Jika *Loop Threshold* = 1

Pada Grafik diatas dengan perubahan *Loop Threshold* sebesar 1 tingkat akurasi tertinggi masih terjadi pada nilai *Intersection Threshold* sebesar 7000 khususnya pada kondisi siang hari meskpun terjadi penurunan tingkat akurasi jika dibandingkan dengan tingkat akurasi sistem pada skenario pengujian ke-2 (Metode Penyempurnaan Bentuk). Kemudian, hal tersebut menunjukkan bahwa dengan penambahan nilai *Loop Threshold* sebesar 1 mampu mengurangi kesalahan sistem dalam membaca satu objek/kendaraan terdeteksi dan terhitung lebih dari sekali. Kemudian, pada grafik diatas didapatkan tingkat akurasi tertinggi untuk kondisi siang hari sebesar 76% dan untuk kondisi malam hari sebesar 68%.



Gambar 3.3 Grafik Akurasi Sistem Jika *Loop Threshold* = 2

Dari Gambar 3.3, dapat dilihat bahwa akurasi tertinggi untuk kondisi siang dan malam hari terjadi ketika nilai *Intersection Threshold* sebesar 5000. Kemudian nilai akurasi terendah terjadi ketika nilai *Intersection Threshold* sebesar 10000. Hal tersebut menunjukkan semakin besar nilai *Loop Threshold* akan semakin menurunkan nilai akurasi jika nilai *Intersection Threshold* diatas rata-rata jumlah piksel hitam pada hasil proses *Intersection Frame*. Namun, akan meningkatkan nilai akurasi untuk nilai-nilai *Intersection Threshold* yang menyebabkan sensitivitas sistem menjadi tinggi atau pada nilai *Intersection Threshold* yang rendah.

Sehingga, jika sistem ingin menggunakan *Loop Threshold* sebesar 2 maka sebaiknya juga diikuti dengan perubahan nilai *Intersection Threshold* sebesar 5000.

4. Kesimpulan

Setelah sistem sudah dilakukan beberapa skenario percobaan dan dianalisis unjuk kerjanya, dapat diambil beberapa kesimpulan dari penelitian analisa algoritma penghitung kendaraan roda empat dalam kondisi siang dan malam hari dengan metode *frame intersection* antara lain :

1. Sistem penghitung kendaraan roda empat dalam kondisi siang dan malam hari dengan metode *frame intersection* masih memiliki keterbatasan terhadap parameter tertentu khususnya pada parameter *Intersection Threshold*, dimana hal tersebut dapat mempengaruhi tingkat akurasi.
2. Tingkat akurasi terbaik berdasarkan pengujian yang dilakukan adalah ketika parameter *Intersection Threshold* = 7000 dengan *Loop Threshold* = 0.
3. Sistem memiliki kendala dalam menghitung ketika dua kendaraan berada dalam jarak yang dekat, sehingga kedua kendaraan tersebut masih terdeteksi sebagai satu kendaraan.
4. Sistem yang direalisasikan sudah mampu menghilangkan objek-objek selain kendaraan khususnya garis jalan meskipun pada beberapa *frame* masih terdapat sedikit garis jalan.
5. Kemudian, dari percobaan yang dilakukan didapatkan bahwa untuk kondisi malam hari, lampu dari kendaraan dapat digunakan sebagai objek yang dapat dideteksi dan diproses (dihitung).
6. Tingkat akurasi sistem tidak mencapai 100% karena dipengaruhi oleh spesifikasi perangkat keras yaitu kamera yang berasal dari *smartphone* atau memiliki kualitas video yang belum baik jika dibandingkan kamera *webcam* ataupun kamera konvensional, alur dari *pre-processing* masih belum baik dalam mendeteksi kendaraan karena tidak melakukan pengolahan gambar pada variable *imgLine* pada domain RGB, dan sifat acak pada kendaraan di setiap video.

Daftar Pustaka

[1] Anonim, *Operation Manual : SONY VEGAS PRO 11.0*, SONY Corporation, Japan, 2011
[2] Anonim, *Edit Colors : Paint 6.2(Build 9200)*, Microsoft Corporation, United States, 2012
[3] Anonim, *Product Help :MATLAB 7.8.0 (R2009a)*, The MathWorks, United States, 2009
[4] Kuo, Sen, M dkk. 2006. *Real Time Signal Processing.* Edisi ke 2. London.
[5]Prasetyo, Eko. 2012. *Pengolahan Citra Digital dan Aplikasinya Menggunakan Matlab*. Yogyakarta. Penerbit Andi.
[6]Ryanto. *"Praktikum 2 Dasar Pengolahan Citra (1)"*. diunduh pada 30, November, 2013. http://lecturer.eepis-its.edu/~riyanto/**citra**-bab2.pdf .
[7]Sistama,Ranggi.2016. Simulasi Monitoring Objek yang Masuk dan Keluar untuk Mengontrol Ketersediaan Lahan Menggunakan *Video Processing*.Bandung : Telkom University
[8]Wijaya, Marvin Ch dan Agus Priyono. 2007. *Pengolahan Citra Digital Menggunakan MatLab Image Processing Toolbox*. Bandung. Informatika.
[9]Wikipedia. "HSL and HSV". 16 Desember 2016. http://www.lps.usp.br/hae/apostila/basico/HSI- wikipedia.pdf
[10]Wade,I."Hue-Saturation-Value Representation and Correlation of Multispectral/Multi-Modal Datasets".16 Desember 2016. http://thesis.library.caltech.edu/6027/10/10_Appendix_F_LWade.pdf
[11]Fisher, R, dkk."Dilation".16 Desember 2016. http://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm
[12]Dewantoro, Adrian Kurnia.2015. *Simulasi dan Analisis Sistem Penghitung Kepadatan Lalu Lintas dan Klasifikasi Kendaraan Berbasis Webcam dengan Metode Background Substraction*. Bandung : Telkom University .
[13]Syukur, Muhammad.2009. *Menentukan Kepadatan Lalu Lintas dengan Penghitungan JumlaH Kendaraan Berbasis Video Processing.* Bandung : Telkom University.
[14]Putri, Riski Dahlia I.2010. *Desain dan Simulasi Penghitung dan Pengidentifikasian Panjang Antrian pada Lampu Lintas dengan Metode Pengolahan Digital.* Bandung : Telkom University.
[15]Berlian W.,Marnala.2010. *Simulasi dan Analisis Sistem Penghitung Kepadatan Lalu Lintas Berbasis Kamera Digital dan Pengolahan Citra Digital.* Bandung : Telkom University.
[16]Wildan.2011. *Sistem Penghitung Kecepatan Sesaat Kendaraan Berbasis Camcorder dengan MEtode Background Substraction.*Bandung : Telkom University.

Lampiran

1. Tabel Perhitungan Akurasi Jika *Loop Threshold* = 0

VIDEO	KENDARAAN YANG LEWAT	*INTERSECTION THRESHOLD*							
		5000		7000		8500		10000	
		SIANG	MALAM	SIANG	MALAM	SIANG	MALAM	SIANG	MALAM
1	5	3	4	3	2	3	0	2	0
2	5	9	5	6	5	4	1	3	0
3	5	4	12	5	8	5	6	3	4

4	5	4	7	3	6	3	4	2	2
5	5	6	9	5	6	6	4	3	2
6	5	4	6	4	4	3	4	2	4
7	5	3	10	3	8	3	7	4	3
8	5	5	9	5	6	6	4	4	3
9	5	5	9	4	6	4	5	4	6
10	5	3	6	3	6	2	6	2	6
JUMLAH KENDARAAN	50	46	77	41	57	39	41	29	30
JUMLAH ERROR		14	27	11	15	15	17	21	24
AKURASI		0.72	0.46	0.78	0.7	0.7	0.66	0.58	0.52

2. Tabel Perhitungan Akurasi Jika *Loop Threshold* = 1

VIDEO	KENDARAAN YANG LEWAT	*INTERSECTION THRESHOLD*							
		5000		7000		8500		10000	
		SIANG	MALAM	SIANG	MALAM	SIANG	MALAM	SIANG	MALAM
1	5	3	4	2	2	2	0	2	0
2	5	9	3	6	2	4	0	2	0
3	5	4	12	5	8	4	6	3	3
4	5	4	7	3	5	3	3	2	2
5	5	6	8	5	4	5	2	3	1
6	5	4	5	4	4	2	4	2	2
7	5	3	9	3	7	3	6	3	2
8	5	5	8	5	5	5	4	3	2
9	5	5	8	4	6	4	3	4	3
10	5	3	6	3	6	2	5	1	3
JUMLAH KENDARAAN	50	46	70	40	49	34	33	25	18
JUMLAH ERROR		14	25	12	16	16	21	25	32
AKURASI		0.72	0.5	0.76	0.68	0.68	0.58	0.5	0.36

3. Tabel Perhitungan Akurasi Jika *Loop Threshold* = 2

VIDEO	KENDARAAN YANG LEWAT	*INTERSECTION THRESHOLD*							
		5000		7000		8500		10000	
		SIANG	MALAM	SIANG	MALAM	SIANG	MALAM	SIANG	MALAM
1	5	3	3	2	1	2	0	2	0
2	5	5	2	4	1	3	0	2	0
3	5	4	8	4	5	4	4	2	2
4	5	4	5	3	3	3	2	2	1
5	5	4	7	4	3	4	1	1	0
6	5	4	4	4	3	2	3	1	2
7	5	3	7	3	5	3	3	3	2
8	5	5	7	5	3	5	3	3	1
9	5	4	5	4	4	4	2	4	2
10	5	3	5	2	5	2	3	1	3
JUMLAH KENDARAAN	50	39	53	35	33	32	21	21	13
JUMLAH ERROR		12	15	15	17	18	29	29	37
AKURASI		0.76	0.7	0.7	0.66	0.64	0.42	0.42	0.26