DIPA PNBP FT SENIOR

RANCANG BANGUN PENGHITUNG JUMLAH KENDARAAN MENGGUNAKAN METODE *TRACKING FEATURE POINT* BERBASIS RASPBERRY

(LAPORAN PENELITIAN)

Oleh:

Dr. Sri Purwiyanti, S.T., M.T. **NIDN 0004107301**
Herlinawati, S.T., M.T. **NIDN 0014037102**
Umi Murdika, S.T. M.T. **NIDN 0006027205**

Dibiayai oleh PNBP Fakultas Teknik Tahun 2017

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG

TAHUN 2017

HALAMAN PENGESAHAN

1.	**Judul Penelitian**	:	Rancang Bangun Penghitung Kendaraan Menggunakan Metode Tracking Feature Point Berbasis Raspberry
2.	**Bidang Penelitian**	:	Rekayasa
3.	**Ketua Peneliti**		
	a. Nama Lengkap	:	Dr. Sri Purwiyanti, S.T., M.T., Ph. D.
	b. Jenis Kelamin	:	Perempuan
	c. NIP	:	197310041998032001
	d. Disiplin Ilmu	:	Teknik Elektronika
	e. Pangkat/Golongan	:	Penata Muda Tk. I/IIIb
	f. Jabatan Struktural	:	Asisten Ahli
	g. Fakultas/Jurusan	:	Teknik/Teknik Elektro
	h. Alamat	:	Ged. FT Unila, Jl. Sumantri Brojonegoro No. 1 Bandar Lampung 35145
	i. Telepon/Faks/E-mail	:	08111942111
	j. Alamat Rumah	:	-
4.	**Jumlah Anggota Peneliti**	:	2
5.	**Anggota Peneliti**	:	

No	Nama	Bidang Keahlian	Jurusan	Universitas
1	**Herlinawati, S.T., M.T.**	T. Elektronika	T. Elektro	Unila
2	**Umi Murdika, S.T., M.T**	T. Elektronika	T. Elektro	Unila

6.	**Lokasi Penelitian**	:	Lab. Teknik Elektronika
7.	**Jumlah Pendanaan**	:	Rp. 10.000.000,00 (Sepuluh Juta Rupiah)
8.	**Sumber Pendanaan**	:	PNBP FakultasTeknik Unila Tahun 2017

Mengetahui,
Ketua Jurusan Teknik Elektro

Dr. Ing. Ardian Ulvan, S.T., M.Sc.
NIP. 19731128 1999031005

Bandar Lampung, 5 November 2017
Ketua pelaksana

Dr. Sri Purwiyanti, S.T., M.T., Ph. D.
NIP. 197310041998032001

Menyetujui,

a.n. Dekan Fakultas Teknik
Wakil Dekan I

Dr.Eng. HelmyFitriawan
NIP. 197509282001121002

Ketua LPPM Unila

Warsono, Ph.D.
NIP: 196302161987031003

I. PENDAHULUAN

1.1 Latar Belakang

Seiring meningkatnya perekonomian dan pertambahan jumlah penduduk setiap tahunnya, maka jumlah kendaraan yang beroperasi di jalan raya semakin banyak. Hal ini dibuktikan pula dengan rekor tingginya penjualan kendaraan, yang mencapai 3,326,380 unit pada tahun 2010. Maka, wajar saja jika di kota-kota besar sangat sering sekali terjadi kemacetan karena kendaraan yang ada sudah kurang sebanding dengan fasilitas jalan yang ada. Pertumbuhan jumlah kendaraan seolah-olah memberikan keuntungan pada peningkatan kualitas kehidupan. Namun, hal tersebut akan berdampak pada penurunan terhadap kualitas udara di lingkungan sekitar yang disebabkan oleh kepadatan lalu lintas, dan akan menyebabkan kemacetan.

Untuk menentukan kepadatan rata-rata lalu-lintas diperlukan adanya survey penghitungan jumlah kendaraan yang melintas di jalan raya. Pelaksanaan survey tersebut biasanya dilakukan oleh seorang pengamat sehingga dimungkinkan terjadinya *human error* dalam proses penghitungan karena terlalu padatnya jumlah kendaraan yang lewat, pengaruh lingkungan atau kondisi internal peneliti itu sendiri sehingga mengakibatkan kurang akuratnya proses penghitungan yang dilakukan langsung oleh seorang peneliti. Selain rentan terjadinya *human error*, penghitungan yang dilakukan oleh manusia memerlukan biaya tersendiri untuk setiap pelaksanaannya sehingga kurang efisien. [1]

Penelitian ini berjudul Rancang Bangun Model Penghitung jumlah Kendaraan Pada Jalan dua Jalur dengan menggunakan Metode *Background Subtraction* berbasis *Web cam*, merupakan salah satu jenis penelitian yang telah banyak diaplikasikan salah satu contoh pada penelitian yang sebelumnya yaitu *Counting Car using OpenCV , Visual Studio 2010 (VC++) , and boost* yang menghitung jumlah kendaraan dan kecepatan yang hanya jalan satu arah. Berdasarkan pada kondisi tersebut, fokus penelitian ini adalah mengembangkan dari penelitan sebelumnya, yaitu merancang sebuah model penghitung jumlah kendaraan pada jalan dua jalur berbasis *webcam.* Data yang didapatkan ini dapat di gunakan oleh dinas lalu lintas jalan raya dan pemerintah setempat sebagai referensi untuk mengetahui jumlah kendaraan yang melintas pada jalan raya agar pemerintah dapat memperbaiki jalan yang sudah ada, pelebaran jalan, dan juga penambahan jalan baru atau penataan rute baru.

1.2 Tujuan Penelitian

Tujuan penelitian yang hendak diperoleh adalah:

1. Membuat sebuah program yang dapat mendeteksi jumlah kendaraan yang melintas di dua jalur.
2. Dapat menerapkan metode *background Subtraction* pada model pendeteksi jumlah kendaraan yang melintas di dua jalur.

1.3 Manfaat Penelitian

1. Membuat alat yang dapat memudahkan perhitungan jumlah kendaraan yang melintas yang ada di jalan raya.
2. Menerapkan teknik *background Subtraction* untuk deteksi kendaraan yang melintas di dua jalur.

II. TINJAUAN PUSTAKA

2.1 Background subtraction

Background subtraction adalah proses untuk menemukan objek pada gambar dengan cara membandingkan gambar yang ada dengan sebuah model latar belakang. Prosedur *background subtraction* terdiri dari 3 tahap yaitu *pre-processing*, *background modeling*, dan *foreground detection.*

Gambar 1. *Background subtraction* -1 gambar awal, 2- *background model*, -3 hasil *background subtraction*, -4 hasil *background subtraction* setelah *threshold*

Tahapan dalam *Background subtraction*:

a. Pre-procesing

Pada tahap ini data awal dari kamera (atau input lainnya) diproses menjadi bentuk yang dapat dimengerti oleh bagian program lain. Pada tahapan awal ini dilakukan *noise removal* dan eliminasi objek kecil pada gambar agar menjadi lebih

informative. Eleminasi objek kecil dilakukan dengan munggunakan *mathematical morphology* yaitu *transformasi opening.*

b. Background modeling

Tahap ini bertujuan untuk membentuk model *background* yang konsisten, namun tetap dapat beradaptasi dengan perubahan lingkungan yang ada. Model harus dapat mentoleransi tingkat perubahan lingkungan, namun tetap *sensitive* dalam mendeteksi pergerakan dari objek yang relevan. Algoritma *Background modeling* sendiri sangat banyak, namun pada tugas ahkir ini akan memakai *appoxing median filter*, karena proses kompotasinya cepat dan hasil cukup memuaskan.

c. Foreground detection

Pada tahap ini, dilakukan proses ektraksi *foreground* dari *background.* Secara sederhana hal ini dilakukan dengan persamaan:

$$Rr,c = Ir,c - B\,r,\,c \quad \ldots\ldots\ldots(1)$$

Dengan: R=hasil r =baris

I=gambar saat ini c = kolom

B=background model

nilai R lalu dibandingkan dengan nilai threshold yang telah ditentukan, jika lebih besar dari nilai threshold maka piksel di I(r,c) dapat dianggap berbeda dengan piksel di B(r,c). Nilai threshold sendiri dipakai untuk mengurangi error yang disebabkan noise pada gambar digital. [1]

2.2 *Motion* (Gerak)

Teknik dasar untuk mendeteksi perubahan antara dua buah citra $f(x, y, t_i)$ dan $f(x,\ y,\ t_{i-1})$ dari data video pada waktu ke$-t_i$ dan ke$-t_{i-1}$ adalah dengan membandingkan dua citra piksel demi piksel, dimana salah satu atau keduanya terdapat objek yang bergerak sehingga menghasilkan nilai tidak nol yang berkorespondensi dengan komponen citra yang tidak statis, sehingga menghilangkan komponen citra yang sifatnya statis.[2]

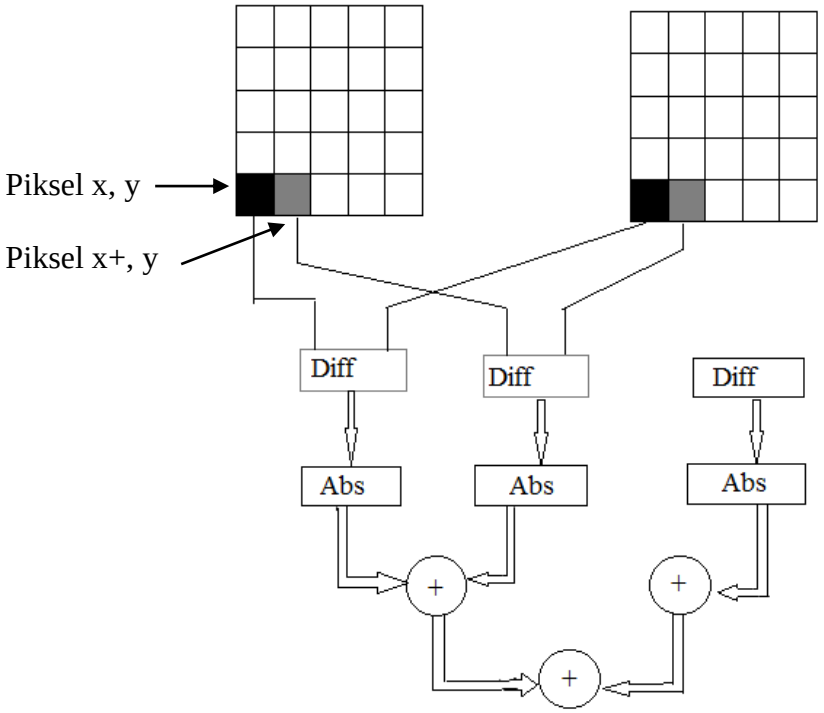
2.3 *Motion Detection*

Sebuah citra yang diambil dari kamera akan digunakan sebagai bahan utama untuk proses Analisa. Teknik dasar untuk deteksi gerak yaitu menggunakan perbedaan nilai antara frame pada waktu t dan t_{-1} dan membandingkannya. Secara matematis dapat digunakan rumus [5]:

$$SAD = \sum_i^n [I(t_i) - \mathrm{I}(t_{-1})] \quad \text{.....(2)}$$

Dimana n adalah jumlah frame dalam suatu video, sedangkan $I(t_i)$ adalah gambar I pada waktu i. Ilustrasi pencocokan dan perbandingan antar frame ditunjukan pada Gambar 2.

Gambar 2 berikut menunjukan konsep dasar pencocokan dan perbandingan antara dua buah frame pada waktu t_i dan t_{i-1}. Dimana terdapat 2 buah frame yang dibandingan antara piksel x, y dan piksel x+, y. [3]



Gambar 2. Pencocokan nilai piksel antar frame

2.4 Threshold

Thresholding merupakan salah satu teknik segmentasi yang digunakan untuk citra dengan perbedaan nilai intensitas yang signifikan antara latar belakang dengan perbedaan nilai intensitas yang signifikan antara latar belakang dan objek utama (katz,2000). Dalam pelaksanaanya *thresholding* membutuhkan suatu nilai yang digunakan sebagai nilai pembatas antar objek utama dengan latar belakang, dan nilai tersebut dinamakan dengan threshold.

Thresholding digunkan untuk mempartisi citra dengan mengatur nilai intensitas semua piksel yang lebih besar dari nilai *threshold* T sebagai latar depan dan yang lebih kecil dari nilai *threshold* T sebagai latar belakang. Biasanya pengaturan nilai

threshold dilakukan berdasarkan *histogram grayscele* (*Gonzales dan woods, 2002; fisher, dkk,2003; xiaoyi majon, 2003).*

Thresholding adalah metode yang paling sederhana dari segmentasi. Setiap individi piksel didalam grayscale ditandai sebagai *"objek"* piksel jika nilai mereka lebih besar dari nilai *thresholding* (dijadikan sebuah objek yang lebih terang dari backgroundnya) dan sebagai *"background"* piksel sebaliknya diberikan nilai *"I"* dan piksel background diberikan nilai *"0".*

Parameter kunci di dalam thresholding merupakan pilihan dalam melakukan *threshold.* Terdapat berbagai metode dalam memilih nilai mean atau medan pada dasarnya jika piksel objek lebih terang dibandingkan dengan *background* maka piksel objek tersebut juga lebih terang dari rata-ratanya. Pada gambar yang masih memiliki *noise* dengan *background* dan nilai objek, mean dan median akan bekerja maksimal dalam *threshold.* Dalam pendekatan yang lebih dalam, dapat pula dilakukan dengan cara membuat sebuah *histogram* dari intensitas citra piksel dan menggunakan *valley point* sebagai nilai *threshold*. Dengan melakukan pendekatan histogram memungkinkan adanya beberapa nilai rata-rata pada piksel *background* dan objek, tetapi nilai piksel tersebut mempunyai beberapa variasi nilai yang masih berada pada sekitar nilai rata-rata itu. Dari citra yang mempunyai *valley point* yang tidak jelas.

Pencarian metode *threshold* yang sederhana tidak memerlukan pengetahuan yang lebih tentang citra dari *thresholding* pun bias bekerja pada citra yang memiliki noise, metode *iterative* merupakan yang baik dilakukan seperti:

1. memilih *initial* dari *threshold* (T). dapat dilakuakn random atau menurut metode yang diinginkan
2. citra ini disegmentasikan kedalam piksel objek dan piksel background seperti dibawah ini

   - G1 = {f(m,n) :f(m,n) > T}
   - G2 = {f(m,n) :f(m,n) < T}

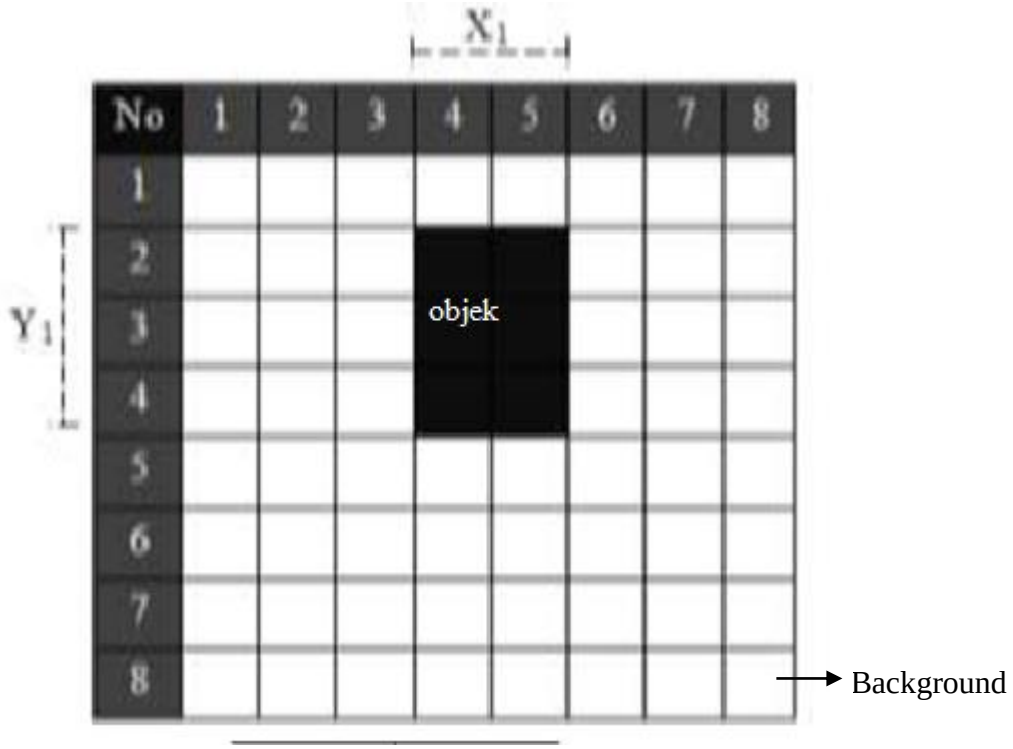
   Dimana :

   - G1 adalah nilai piksel objek.
   - G2 adalah nilai piksel background.
   - f(m,n) adalah nilai dari piksel yang terletak pada m^{th} kolom dan n^{th} baris
3. hitung nilai rata-rata *gray value* $\mu 1$ dan $\mu 2$ pada piksel dalam G1 dan G2
4. hitungan nilai *threshold* baru:

   $T = ½ (\mu 1 + \mu 2)$
5. ulangi langkah ke 2 sampai dengan langkah ke 4 dengan nilai T yang berbeda sampai nilai *threshold* yang baru sama dengan nilai yang sebelumnya.

2.5 Feature ekstraction

Proses untuk mencari vitur suatu kendaraan di butuhkan contoh gambar yang digunakan sebagai acuan. Gambar yang digunakan seagai acuan untuk mendiskripsikan vitur kendaraan seperti panjang dan lebar kendaraan. Variable-variabel yang dibutuhkan untuk mendapatkan suatu ciri kendaraan adalah menghitung banyak piksel berdasarkan pada sumbu X dan Y (gambar 4).



Gambar 3. Proses ekstrasi ciri kendaraan

Gambar 3 menunjukan bahwa pada sumbu X_1 mengindikasikan dari lebar suatu kendaraan dimana letak posisi titik berada pada titik 4 – 5 sehingga X_1 = 2 unyuk mendapatkan panjang kendaraan mengacu pada sumbu Y. untuk Y_1 mengindikasikan dari panjang suatu kendaraan dimana posisi titik berada pada titik 2 – 4 sehingga $Y_1 = 3$. Sehingga didapat suatu fitur kendaraan berupa panjang kendaraan (Y_1=3) dan lebar kendaraan $X_1 = 2$. [4]

2.7 Penentuan Objek Bergerak

Objek bergerak ditentukan dengan cara apabila terdapat piksel-piksel yang memiliki nilai intensitas lebih besar dari nilai ambang, sementara piksel-piksel yang memiliki intensitas lebih kecil dari nilai ambang maka piksel tersebut dianggap sebagai *background*. Proses penentuan apakah nilai intensitas tiap piksel

lebih kecil atau lebih besar dari nilai ambang dapat ditentukan dengan menggunakan persamaan :

$$\frac{|f_T - \mu_B|}{\sigma_B} \leq T$$

Dimana f_T adalah nilai intensitas piksel saat ini, μ_B adalah nilai intensitas piksel pada model latar belakang, $\sigma_B$adalah nilai varian dari distribusi normal latar belakang. [5]

2.7 Kamera *(web cam)*

Kamera adalah alat paling populer dalam aktivitas fotografi. Fotografi berasal dari kata dalam bahasa yunani yaitu "*Photos"* artinya cahaya dan "*Grafo"* artinya melukis, jadi fotografi adalah proses melukis dengan media cahaya.

Web cam merupakan salah satu jenis kamera yang merupakan sebuah perangkat keras berupa kamera yang digunakan untuk melihat gambar atau video bergerak yang berhubungan dengan PC atu Laptop, Hardwere ini sendiri hadir pertama kali pada tahun 1992 dan mulai terhubung ke internet pada tahun 1993. Hingga saat ini, wabcam sudah hadir dengan merek, bentuk dan fasilitas yang beragam. dan kegunaanya pun sangat beragam, mulai dari untuk merekam video atau media mengambil foto hingga media belajar dan pertemuan online. Gambar 5 merupakan salah satu jenis web cam yang sudah hadir dengan merk *LOGITECH HD.* [8]

Gambar 4. *Web cam*

III METODE PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian ini mulai dilaksanakan mulai bulan Juni – Oktober 2017 di Laboratorium Teknik Elektronika, Jurusan Teknik Elektro, Universitas Lampung.

3.2 Spesifikasi Alat

Spesifikasi alat adalah sebagai berikut :

1. Kamera digunakan sebagai pengindera, yang bertugas mengambil citra objek yang ada di depannya. Kamera ini menggunakan masukan *DC* 5V, menggunakan kabel usb 2.0, dan menghasilkan keluaran berupa video dengan format file *AVI*.
2. Hasil keluaran dari kamera akan secara otomatis disimpan pada SD Card, pada penelitian ini diggunakan SD Card dengan kapasitas 16 GB.
3. File video dengan format *AVI* pada SD *Card* akan dijadikan sebagai masukan pada pengolahan citra yang akan dilakukan.

4. Pengolahan citra pada penelitian kali ini menggunakan perangkat lunak OpenCV 2.4.10 dan *Visual Studio* Express 2010 pada Laptop sebagai media pemrograman.

3.3 Spesifikasi Sistem

Spesifiasi sistem adalah sebagai berikut :

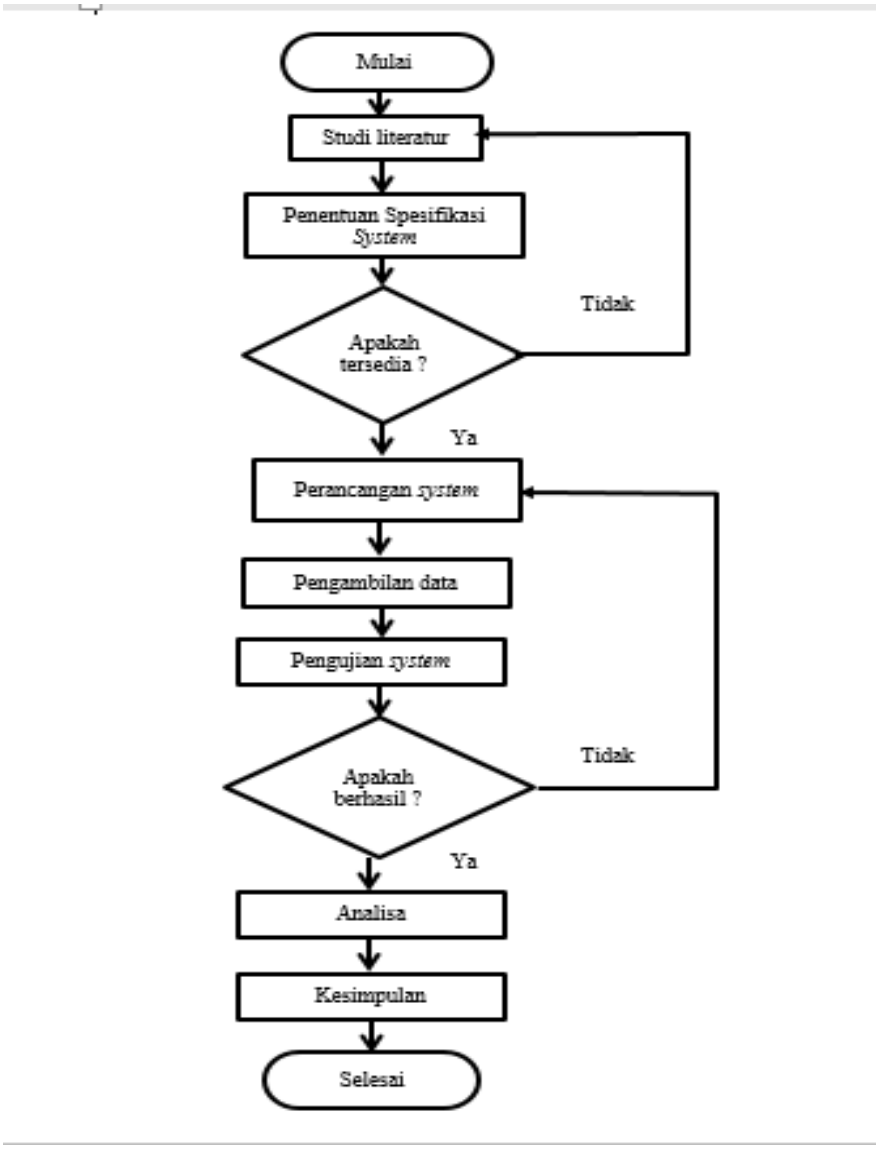
Mampu mengambil video dari objek di depan dengan devais pengindra kamera, selanjutnya keluarannya yang berupa file video akan menjadi masukan sistem. Sistem yang akan dibangun menggunakan perangkat lunak OpenCV 2.4.10 dan *Visual Studio* Express 2010 sebagai media pemrogramannya. Selanjutnya sistem diharapkan mampu melakukan pendeteksian pola pada dimensi objek yang diamati. Jenis yang akan diamati adalah sebuah kendaraan yang melintas.

3.4 Metode yang Digunakan

Di dalam penelitian ini video diambil dengan menggunakan kamera yang diletakkan di atas tiang tangga penyeberangan. Kamera ini berada di atas tiang tangga penyeberangan dalam keadaan diam (statik). Data yang telah diambil akan disimpan di dalam sebuah microSD yang kemudian akan diolah menggunakan *software OpenCV* 2.4.10 dan *Visual Studio* Express 2010.

3.5 Diagram Alir Penelitian

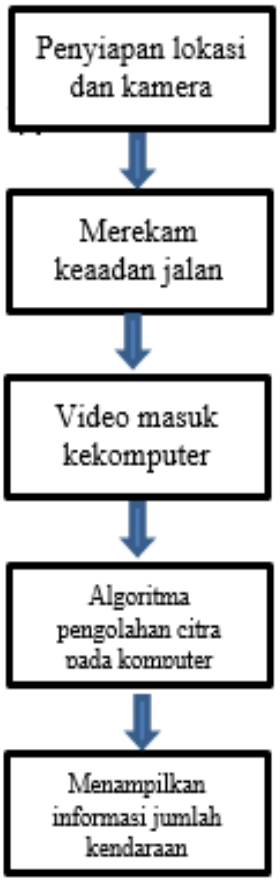
Diagram alir penelitian ini dibuat untuk memperjelas langkah-langkah kerja yang akan dilakukan dalam penelitian, diperlihatkan pada gambar dibawah ini :



Gambar 5. Diagram Alir Penelitian

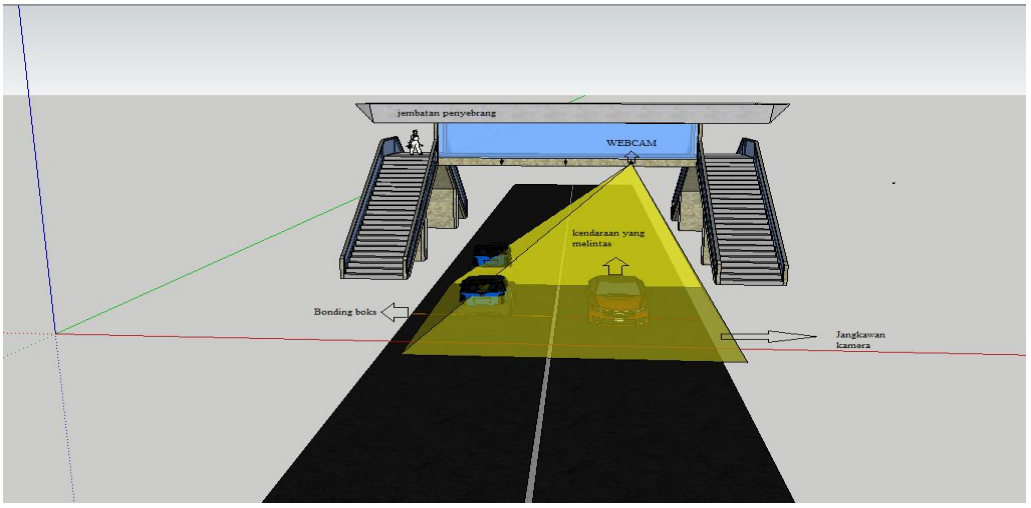
3.6 Perancangan Perangkat Keras

Rancangan perangkat keras ini secara keseluruhan dapat dilihat pada gambar dibawah ini. Setelah pengambilan citra, komputer mengolah video menggunakan algoritma yang telah dibuat secara *real time*

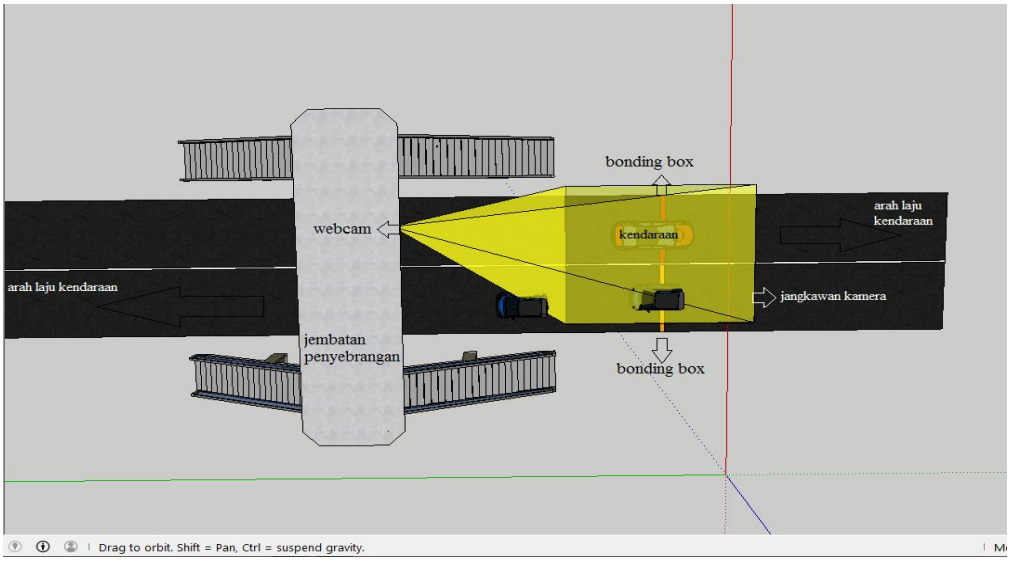


Gambar 6. Blok diagram perangkat keras

Secara umum perangkat keras yang digunakan telah dijadikan satu kesatuan yang meliputi, model jalan dua jalur, berikut merupakan gambaran dari rancangan pada prototype:



Gambar 7. Rancang Bangun Alat Tampak Depan



Gambar 8. Rancang Bangun Alat Tampak Atas

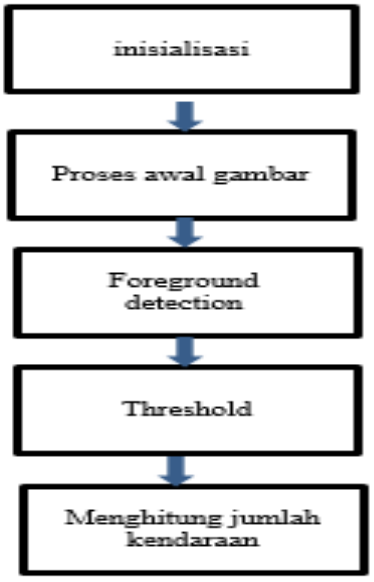
3.7 Perancangan Program

Perangkat yang digunakan pada penelitian ini yaitu berfungsi untuk OpenCV 2.4.10 dan *Visual Studio* Express 2010. Sistem kerja OpenCV 2.4.10 adalah

untuk mengoperasikan komponen atau *library* yang terdapat pada *Visual Studio Express 2010* agar sepenuhnya dapat diatur dan digunakan untuk menciptakan *system* yang diinginkan.

3.7.1 Konfiguras perangkat lunak *Background Subtraction*

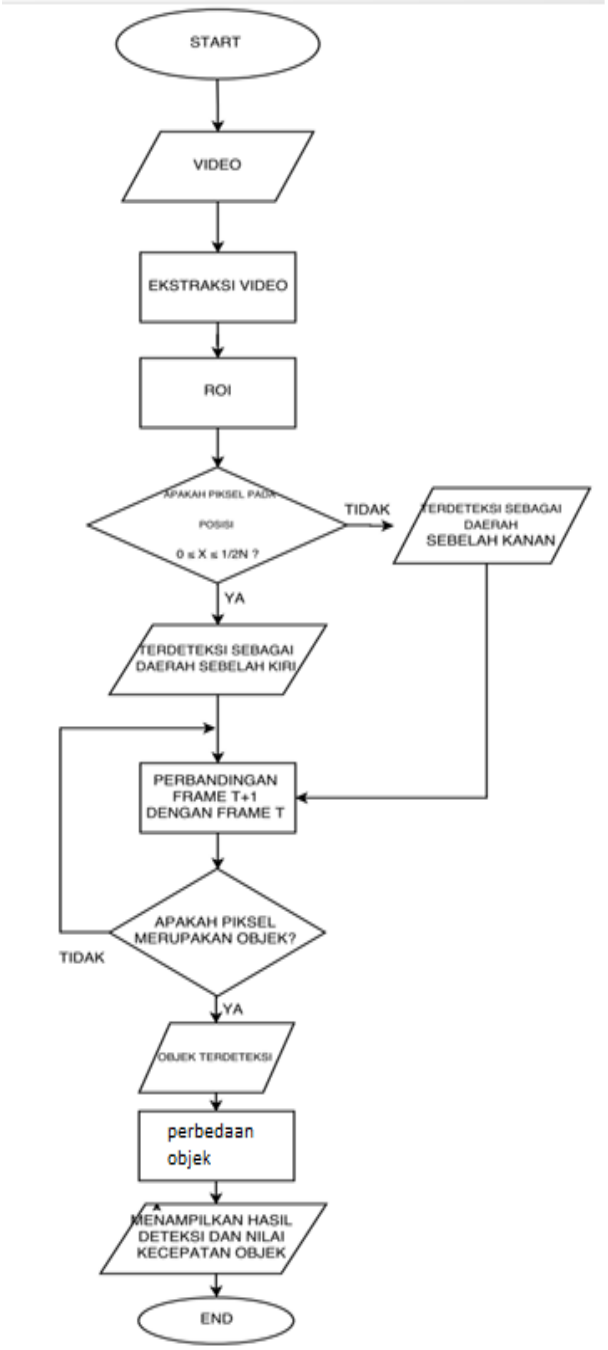
Untuk metode *Background Subtraction*, metode ini mengambil keuntungan dari *background* yang statis atau tidak bergerak, lalu menggunakan algoritmanya untuk memisahkan objek yang bergerak dari *background* yang statis. Algoritma *Gaussian Mixture* menggunakan *foreground detection* untuk memisahkan kendaraan yang bergerak dari objek-objek lain yang statis. Pada Gambar 9 dapat dilihat konfigurasi perangkat unak dari algoritma *Background Subtraction.*



Gambar 9. Konfigurasi perangkat lunak

3.7.2 Perancangan Model Sistem perangkat lunak

Secara keseluruhan sistem dapat dilihat pada Gambar 10.



Gambar 10 Diagram Blok Sistem Keseluruhan

Gambar 10 menunjukan diagram alir metode yang digunakan, dimulai dari input yang akan diolah, kemudian citra RGB pada video akan di ekstraksi yang akan diubah menjadi bentuk frame. Setelah itu akan dilakukan inisialisasi latar belakang awal yang akan digunakan sebagai *background* referensi untuk dibandingkan dengan frame selanjutnya, dalam proses perbandingan ini apabila piksel pada frame selanjutnya dideteksi sebagai objek maka output akan menunjukan bahwa objek terdeteksi setelah itu akan dibuat model *background* baru, sementara apabila piksel dianggap sebagai *background* maka akan langsung dibuat model background-nya.

Pembuatan model background dilakukan dengan cara meng-update parameter, setelah itu akan kembali lagi untuk proses pembandingan 2 buah frame kembali, proses ini berlangsung hingga frame terakhir. Setelah didapatkan hasil *background* dan objek maka selanjutnya yaitu perbanding jalur jalan sebelah kiri dan kanan menggunakan metode *ROI* (*region of interest)* setelah didapatkan perbandingan frame T+1 dengan frame T maka didapatkan objek yang terdeteksi, dan selanjutnya membedakan objek menggunakan metode *threshold* dan setelah didapatkan maka akan terdeteksi jumlah kendaraan di dua jalur.

3.8 Perolehan Citra

Citra yang digunakan dalam pemerosesan dalam penelitian ini didapatkan dari sebuah *webcam* yang diletakan secara statik diatas tiang penyebrangan jalan, Citra video yang didapatkan oleh kamera akan diolah menggunakan *softwere Open CV*

2.4.10 dan *Visual Studio Express 2010* yang sudah terinstal pada laptop, dan selanjutnya akan ditampilkan *secra real time* pada monitor.

IV. HASIL DAN PEMBAHASAN

4.1.Hasil

Pada penelitian ini hasil yang didapatkan dari pengambilan citra dengan menggunakan *webcam* berupa citra dinamis atau video. Setelah data hasil didapatkan maka citra video diubah menjadi citra gambar atau frame. Frame yang sebelumnya merupakan citra RGB diubah menjadi sebuah citra grayscale, hal ini dilakukan untuk mengurangi beban komputasi pada proses pengolahan citra.

Citra yang didapatkan adalah citra dengan ukuran 1920 x 1080 piksel pada setiap frame nya, format video yang didapatkan adalah mp4 yang selanjutnya dalam proses pengolahan citra akan diubah ke dalam bentuk frame dalam format *Joint Photographcs Expert Group* (JPEG). Format gambar tersebut merupakan sebuah format citra yang dirancang agar bisa memampatkan data dengan rasio 1:16.

Proses pengambilan citra pada penelitian ini dilakukan dengan pengambilan data langsung di jalan raya tepatnya di Jl. Zainal Abidin Pagar Alam Bandar Lampung Lampung di atas jembatan penyeberangan, dengan kamera diletakan secara statis dengan tinggi sekitar 3,5 meter dari permukaan jalan, dalam waktu 5 menit dengan frame 30 fps untuk pengambilan video, dan juga pembuatan model jalan dua jalur untuk pengambilan data secara langsung

(Realtime), untuk data *Real time* kamera *(webcam)* diletakan secara statis diatas model jalan dua jalur dengan tinggi 50 cm.

Gambar 11. Tempat penambilan data di jalan raya

Gambar 11 merupakan gambar dari tempat pengambilan data di jalan raya untuk diolah agar terlihat objek kendaraan yang melintas di dua jalur di jalan raya, dengan kamera diletakan secara statis di atas tiang jembatan penyeberangan dengan tinggi sekitar 3,5 meter dari jalan.

Gambar 12. Proses pengambilan data

Sementara Gambar 12 menunjukan bahwa dalam proses pengambilan data, *action - cam* diletakan pada tiang jembatan menggunakan tripod dalam keadaan statik. Pengambilan data video dilakukan dalam durasi waktu 5 menit. Setelah proses pengambilan gambar ini selesai maka tahap selanjutnya adalah pengolahan data citra yang didapatkan dengan memindahkan *SD card* pada *action-cam* ke *computer* yang sudah ter-*install* perangkat lunak *visual studio C++* yang dilengkapi dengan *library Open cv.*

Gambar 13 Model jalan dua jalur

Gambar 13 merupakan gambar dari model jalan dua jalur tempat penelitan secara *realtime* di lakukan. Dengan webcam diletakkan statis di atas tiang penyangga dengan tinggi 30 cm, model lebar jalan sebesar 9 cm, lebar model mobil 2 cm, dan model motor 1 cm, kendaraan inilah yang selanjutnya akan dijadikan objek pada penelitian ini.

4.1.1. Hasil prolehan citra

Proses pengambilan data pada penelitian ini dilakukan pada tempat yang berbeda, yaitu pada jalan raya, dengan pengambilan data video dalam durasi waktu 5 menit dan juga pada model yang di lakukan secara realtime. Dalam video citra

mengandung objek berupa kendaraan yang melintas seperti mobil dan motor, citra-citra inilah yang dijadikan sebagai data dalam penelitian ini.

Citra hasil perolehan awal adalah sebuah citra video berdurasi 3 detik dengan frame rate 30 fps, sehingga total frame yang didapatkan dalam penelitian ini berjumlah 90 frame setiap kondisinya. Contoh citra yang didapatkan adalah sebagai berikut:

Frame 43 *Frame 90*

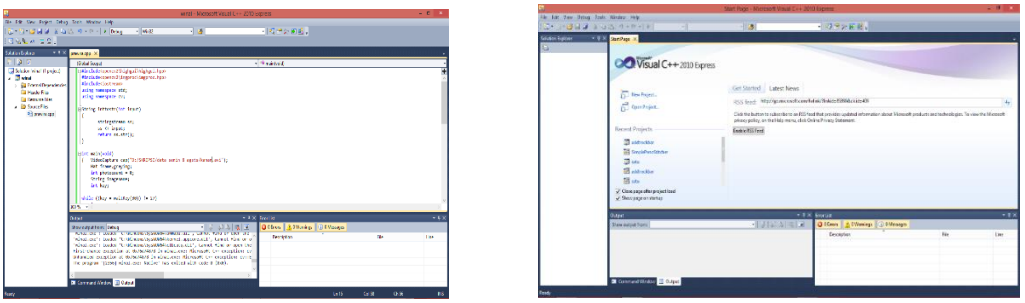
Frame 1 *Frame 2*

Gambar 14 Hasil Perolehan Citra Awal

Gambar 14 menunjukan contoh hasil dari perolehan citra awal yang didapatkan oleh *action-cam* pada kondisi pengambilan data di jalan raya dan pengambilan *frame* pada model jalan dua jalur.

4.1.2 Hasil Pengolahan Awal

Perancangan Sistem Pendeteksi jumlah kendaraan ini menggunakan Microsoft Visual Studio 2010 yang telah terintegrasi oleh Open CV 2.3.10 yang dijadikan *library* pemrogramannya, pada visual studio 2010 penulis menggunakan bahasa C++ dalam pemrogramannya,



Gambar 15 Dekstop Microsoft Visual C++ 2010

Bahasa pemrograman untuk *Windows* yang dipakai oleh *Microsoft* saat ini antara lain Visual C++, Visual C#, dan Visual Basic. Kesalahan pemilihan bahasa dapat berakibat pada kemampuan sistem yang rendah, sintaks yang panjang, dan yang jarang diperhatikan oleh kebanyakan orang adalah eksekusi aplikasi membutuhkan waktu yang lama.

Keunggulan *Visual C++* dalam hal kecanggihan dan kecepatan proses serta banyaknya fasilitas yang tersedia disebabkan karena *Visual C++* memiliki tipe data yang banyak, serta dapat mengatur pengalokasian memori dengan baik. Program C++ yang kita tulis untuk satu jenis *platform*, bisa kita *compile* dan jalankan di *platform* lain dengan tanpa ataupun hanya sedikit perubahan. Visual studio dapat dipasangkan dengan beberapa *library* dan *compiler* eksternal seperti

opencv, FLTK, openGL, dan lainnya tergantung fungsi dan kebutuhan yang diperlukan. Proses eksekusi program bahasa C++ lebih cepat, dengan demikian, sistem yang dibuat dengan bahasa C++ akan menjadi aplikasi yang efisien dan kompetitif.

Setelah mendapatkan hasil video dari Dash-Cam langkah selanjutnya adalah menyiapkan video tersebut agar dapat diolah lebih lanjut. Mengubah format warna RGB hasil dari masing masing *frame* yang terdapat pada video hasil menjadi format warna abu abu adalah langkah awal yang harus dilakukan.

Format warna abu abu dibutuhkan untuk pengolahan citra agar dapat mendeteksi titik sudut pada tahap selanjutnya dengan waktu yang relative lebih cepat. Format warna keabuan membuat format pixsel pada semua *frame* dalam sebuah video menjadi 8 bit dan akan lebih mudah untuk pengolahan selanjutnya.

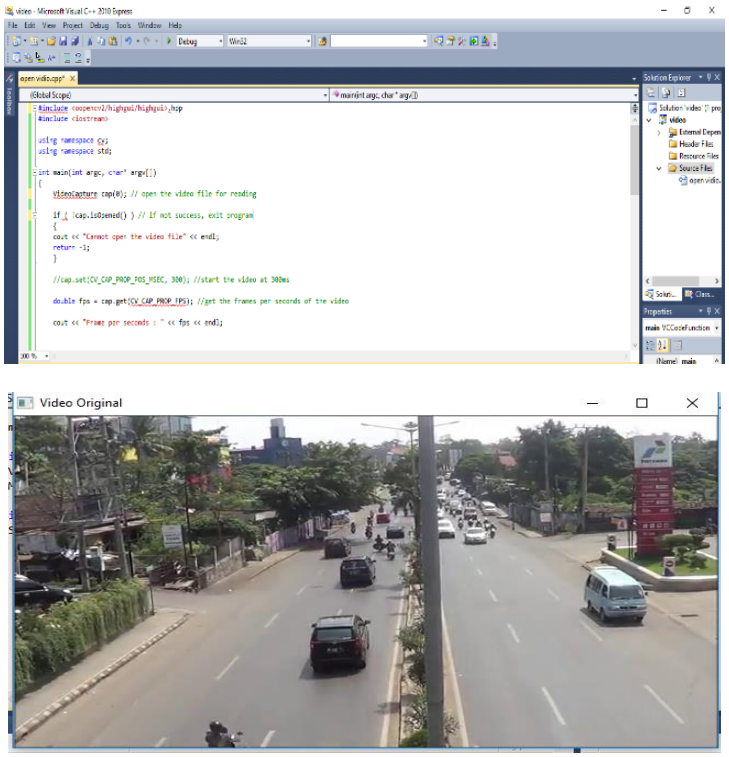
Proses yang dilakukan pertamakali pada penelitian ini adalah mengubah citra RGB menjadi citra *grayscale.* Hal ini dilakukan untuk mengurangi beban komputasi agar waktu yang diperlukan dalam peroses pengolahan citra lebih cepat. Konversi citra RGB menjadi citra dalam arus keabuan ini di lakukan dengan menggunakan persamaan

$$I = 0{,}2989 \times R + 0{,}5870 \times G + 0{,}1141 \times B \qquad \ldots..(1)$$

Dimana R merupakan nilai komponen merah, G menunjukan nilai komponen hijau, dan B menunjukan komponen biru.

4.1.3 Implementasi Input dan Penampilan Video

Implementasi ini diambil dari flowchart, yaitu pada langkah video yang digunakan sebagai input, dan pada langkah *get a next frame* (gambar ditampilkan pada tampilan aplikasi yang telah dibuat). Kode 4.2.1 merupakan implementasi input video dan penampilan video yang sudah dianalisis. Analisis video (*preprocessing* dan *processing*) dilakukan pada fungsi *grabFrame* dan menghasilkan gambar dengan tipe data *Mat*. Hasil tersebut akan dikonversi menjadi tipe data Image, lalu ditampilkan pada Image View yang ada di tampilan aplikasi.



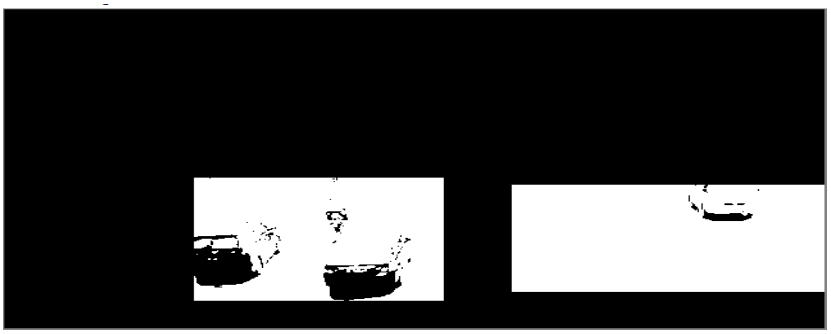
Gambar 16. Inisialisasi untuk input video dan output video

4.1.4. Implementasi Preprocessing Video

Setelah langkah Pemrosesan awal atau preprocessing image yang membuka file video, selanjutnya adalah proses pemisahan objek (*foreground*) dengan latar belakang objek (*background*). Proses pemisahan objek dengan latar belakangnya disebut *Background Subtractor*, *Background Subtractor* sendiri merupakan fitur yang telah disediakan oleh OpenCV.

```
cvtColor(diff_im, gray1, CV_BGR2GRAY);
threshold(gray1,gray2, 175, 255, CV_THRESH_BINARY);
imshow("Hasil Background Substraction",gray2);
Mat threshold_output;
```

Gambar 17 Program *Background Subtraction*



Gambar 18 Hasil eksekusi program *Background Subtraction*

4.1.5 Implementasi Processing Video

Tahap processing video terdiri dari pencarian bentuk mobil, inisialisasi objek mobil, pembaruan posisi mobil. Langkah-langkah untuk mendapatkan Region of Interest (ROI) pada objek yang diinginkan adalah sebagai berikut:

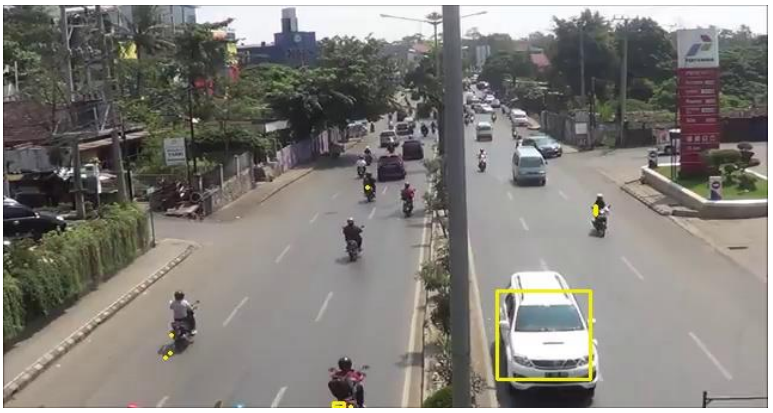
a. Deteksi tepi objek

Deteksi tepi ini menggunakan Background Subtractor sebagai input. Output dari Background Subtractor yang sebelumnya berupa gambar berwarna putih pada

objek akan dibuat berwarna hitam kecuali setiap daerah tepi pada objek tersebut. Deteksi tepi ini juga dapat dibilang menampilkan garis terluar yang membentuk objek.

```
for( int i = 0; i < contours.size(); i++ )
    {
           approxPolyDP( Mat(contours[i]), contours_poly[i], 5, true );
           boundRect[i] = boundingRect( Mat(contours_poly[i]));
           minEnclosingCircle( (Mat)contours_poly[i], center[i], radius[i] );
    }

//Mat drawing = Mat::zeros( threshold_output.size(), CV_8UC3 );
for( int i = 0; i< contours.size(); i++ )
    {
      //Scalar color = Scalar( rng.uniform(0, 255), rng.uniform(0,255), rng.uniform(0,255) );
     // rawContours( drawing, contours_poly, i, color, 1, 8, vector<Vec4i>(), 0, Point() );
      rectangle( cap, boundRect[i].tl(), boundRect[i].br(), Scalar(0,240,240), 2, 8, 0 );
      //circle( drawing, center[i], (int)radius[i], color, 2, 8, 0 );
    }
```



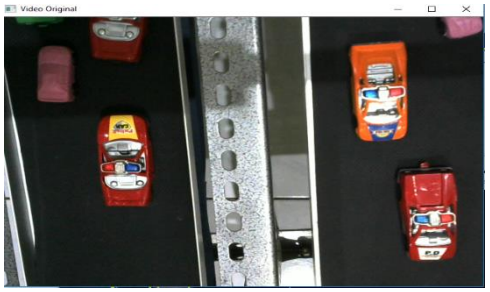
Gambar 19. Hasil program deteksi objek

b. Hasil Perbandingan Dua Buah *Frame*

Gambar 20 menujukkan hasil perbandingan antara dua buah frame. Setelah pengurang frame T dan model background T-1 didapatkan, apabila nilai sebuah piksel hasil pengurangan ini lebih besar dari nilai *threshold* yang dikalikan dengan nilai varian maka piksel tersebut akan bernilai 225 dengan demikian piksel

ini merupakan piksel objek. Agar struktur bentuk objek dapat lebih sempurna maka akan dilakukan sebuah operasi *morfologi closing* .

Frame 4



Frame 49

Gambar 20. Hasil Perbandingan Dua Buah *Frame*

c. Penentuan koordinat titik terluar, nilai tinggi dan lebar

Data-data yang harus diperoleh sebelum memperoleh Region of Interest (ROI) adalah data nilai koordinat titik terluar (x,y), nilai tinggi dan nilai lebar. Sama seperti pada langkah sebelumnya, output dari program sebelumnya akan menjadi input untuk program selanjutnya. Dalam hal ini, output dari program deteksi tepi akan diproses untuk mendapatkan ketiga nilai tersebut.

```
for( int i = 0; i < contours.size(); i++ )
    {
           approxPolyDP( Mat(contours[i]), contours_poly[i], 5, true );
           boundRect[i] = boundingRect( Mat(contours_poly[i]));
           minEnclosingCircle( (Mat)contours_poly[i], center[i], radius[i] );
    }
```

```
objek: 13
objek: 14
objek: 13
objek: 13
objek: 14
objek: 13
objek: 13
objek: 14
objek: 14
objek: 14
objek: 14
objek: 14
objek: 13
objek: 12
objek: 12
objek: 15
objek: 12
objek: 15
objek: 14
objek: 15
objek: 14
objek: 16
objek: 14
objek: 14
objek: 15
```

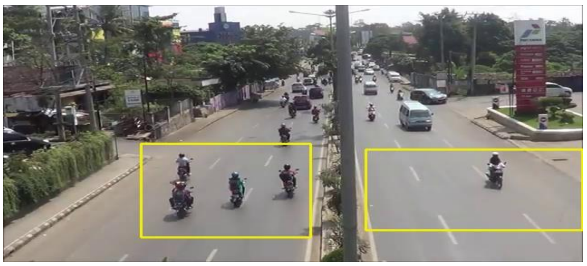
Gambar 21. Gambar penetuan kordinat ROI

d. Mendapatkan Region of Interest (ROI)

Setelah didapat data koordinat terluar (x,y), koordinat nilai tinggi (ymax) dan koordinat nilai lebar (xmax), maka Region of Interest (ROI) dapat digambar secara otomatis di setiap tepi objek.

```
//set ROI
Rect rectRoi = new Rect(0,0,297,176);
roi = new Mat(frameClone, rectRoi);

//background subtraction
private BackgroundSubtractorMOG2 mog;
private double learningRate = 0.001;
private double imageThreshold = 20;
private int history = 1500;
mog =
org.opencv.video.Video.createBackgroundSubtractorMOG2(history,
imageThreshold, false);
mog.apply(frameClone, foreground, learningRate);
```

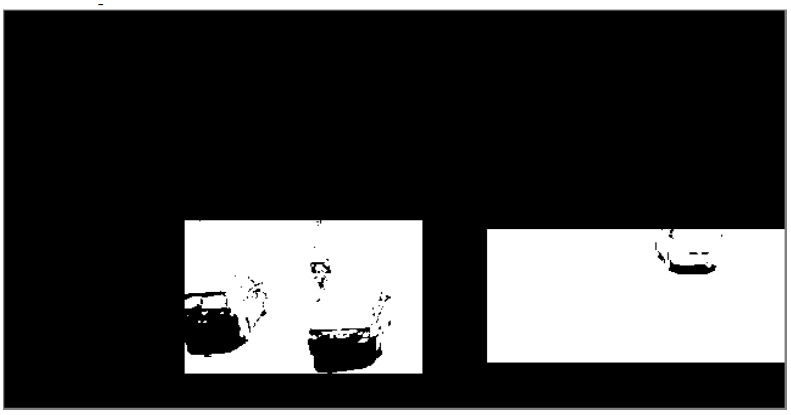


Gambar 22. Hasil eksekusi program menampilkan Region of Interest (ROI)

4.1.6 Hasil Penentuan Objek Bergerak

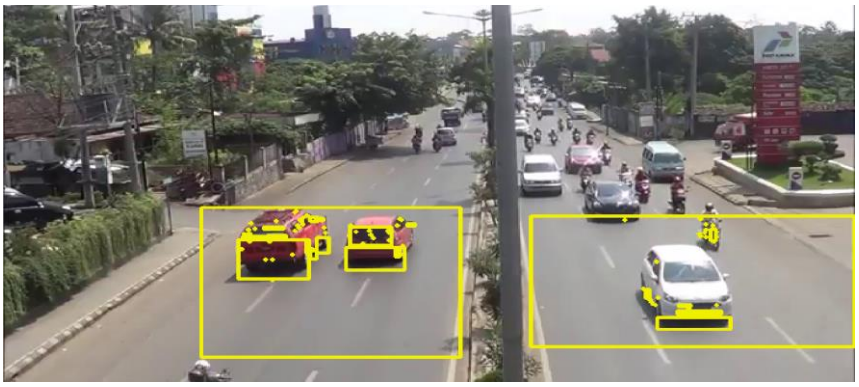
Setelah pengurang frame T dan model background T-1 didapatkan, apabila nilai sebuah piksel hasil pengurangan ini lebih besar dari nilai *threshold* yang dikalikan

dengan nilai varian maka piksel tersebut akan bernilai 225 dengan demikian piksel ini merupakan pikes objek. Agar dapat struktur bentuk objek dapat lebih sempurna maka akan dilakukan sebuah oprasi *morfologi closing* .



Gambar 23. Citra biner tanpa morfologi closing

Gambar 23 merupakan frame hasil pentuan objek tanpa menggunakan *morfologi closing*, karena struktur objek yang tidak sempurna maka sistem akan mengenali objek lebih dari satu.



Gambar 24. Hasil penentuan objek tanpa *morfologi closing*

Gambar 24. merupakan frame pada video, terlihat bahwa sistem mengenali lebih dari satu objek walaupun pada kenyataanya pada frame ini hanya terdapat dua buah objek. Untuk mengurangi hal–hal demikian maka diperlukan sebuah morfologi yang berfungsi untuk lebih menyempurnakan struktur bentuk objek.

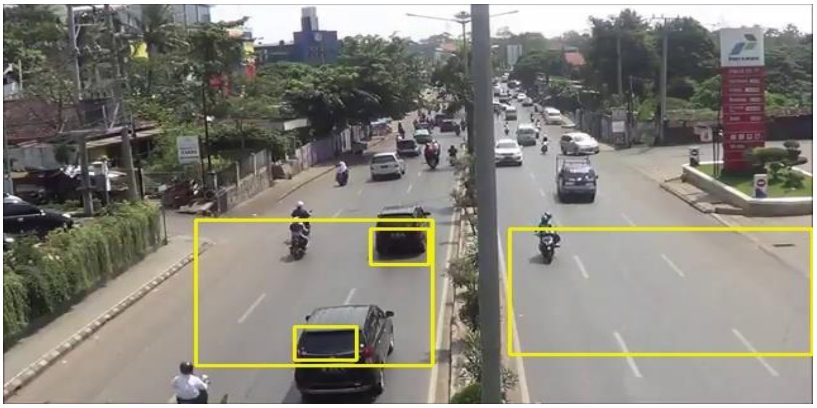
Morfologi closing merupakan penggabungan dua buah operasi, yaitu penggabungan antara operasi dilasi yang diikuti oleh operasi erosi. Dimana pengertian operasi dilasi merupakan sebuah operasi pada citra yang menjadi piksel – piksel yang seharusnya latar belakang menjadi bagian dari objek berdasarkan nilai kernel atau *structuring element* yang ditentukan. Sementara operasi erosi adalah operasi citra yang menjadikan piksel – piksel yang seharusnya objek menjadi bagian dari latar belakang berdasarkan nilai karnel yang ditentukan.

4.1.7 Implementasi Output

Setelah sistem mengumpulkan informasi mobil dan motor, sistem akan mengecek apakah masih terdapat mobil yang terdeteksi pada gambar. Langkah ini merupakan langkah dari *conditional statement* pada flowchart, yaitu Diagram Blok Keseluruhan Sistem. Jika kondisinya nol, maka sistem akan mengambil frame selanjutnya pada video.

Gambar 25. citra biner Hasil *morfologi closing*

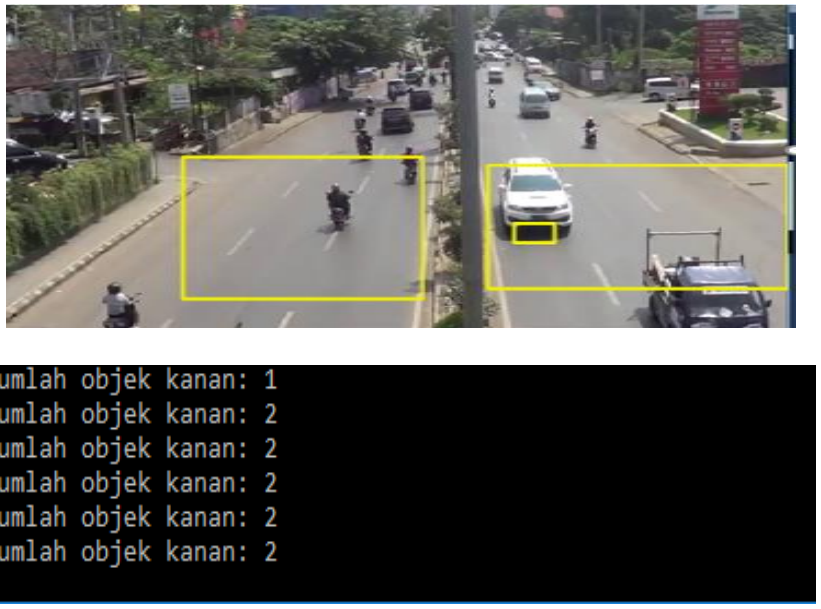
Pada Gambar 25 struktur bentuk citra lebih jelas dan lebih rapat dibandingkan Gambar 23 adapun hasil penetuan objek yang dihasilkan yaitu:



Gambar 26. Hasil penentuan objek

Gambar 26 merupakan hasil penentuan objek. Berbeda dengan Gambar 23, pada Gambar 26 jumlah objek sudah sesuai dengan kenyataanya, dimana jumlah objek yang dihasilkan pada sistem berjumlah dua buah objek.

Gambar 27. Jumlah objek sebelah kiri



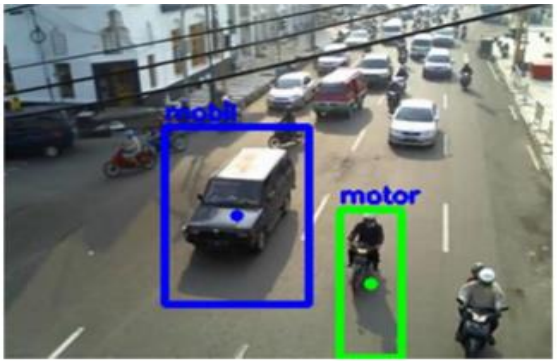
Gambar 28. Jumlah objek sebelah kanan

4.2 Pembahasan

Pada pembahasan ini, sistem akan diuji dengan beberapa pengujian, diantaranya system akan diuji seberapa akurat sistem mengklasifikasi kendaraan, sistem akan diuji seberapa akurat melakukan penghitung jumlah kendaraan di dua jalur, Pengujian tersebut disertai juga analisis yang dapat mendukung untuk pengembangan sistem selanjutnya untuk mewujudkan *Intelligent Transportation System.*

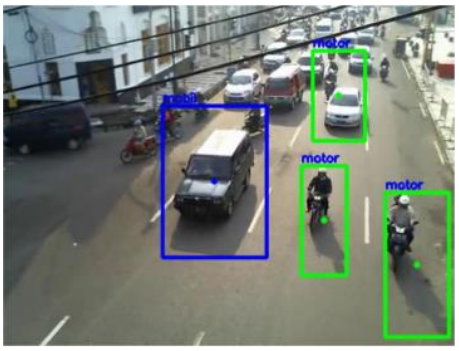
A. Analisis keakuratan sistem klasifikasi kendaraan

Analisis keakuratan sistem dalam hal mengklasifikasi kendaraan ini menggunakan video pertama. Algoritma yang dipakai untuk mengklasifikasikan kendaraan ini berdasarkan ukuran kendaraan itu sendiri. Hasil output yang diinginkan dari sistem ini ditunjukan oleh Gambar 29.

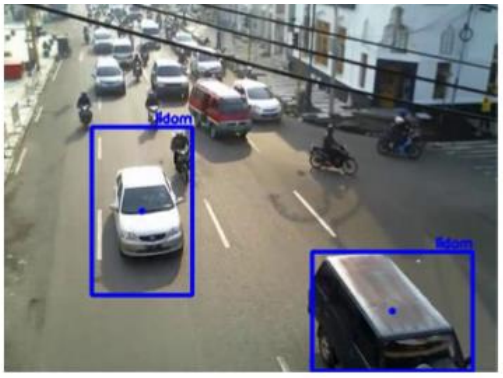


Gambar 29. Hasil yang diinginkan sistem

Pada awal mula sistem klasifikasi kendaraan bermotor ini akan terjadi kesalahan klasifikasi sesaat sistem dalam hal mengklasifikasikan mobil seperti pada Gambar 20. Hal ini terjadi karena kondisi video yang dipakai memiliki objek yang bergerak mendekati kamera. Saat objek jauh dari fokus kamera maka objek akan terlihat lebih kecil daripada ketika objek berada atau mendekati fokus kamera. Kesalahan klasifikasi sesaat sistem akan berakhir ketika objek berada atau hamper mendekati titik fokus kamera, seperti terlihat pada Gambar 21.



Gambar 30. Kesalahaan klasifikasi sesaat sistem



Gambar 31. Kesalahan klasifikasi sesaat sistem berakhir

Kesalahan lain dari sistem klasifikasi terjadi ketika dua atau lebih objek saling berdekatan. Saat kondisi tersebut, sistem akan mengartikan objek yang lebih dari satu tersebut menjadi satu objek. Kesalahan ini timbul karena algoritma background subtractor pada Open CV 2.4.9 belum dapat menghilangkan noise berupa bayangan dari objek. Gambar kesalahan dari kondisi ini ditunjukan oleh Gambar 22 kemudian gambar output algoritma background subtractor ditunjukan oleh Gambar 32.

Gambar 32. Kesalahan sistem mendeteksi objek yang saling Berdekatan

Gambar 33. Penyebab kesalahan deteksi objek yang saling Berdekatan

Kesalahan sistem pada Gambar 33 tersebut terjadi karena bayangan objek menyentuh objek lain sehingga sistem menganggapnya sebagai satu objek. Kesalahan tersebut juga akan berdampak pada kesalahan sistem klasifikasi karena selain sistem menganggapnya sebagai satu objek, objek tersebut juga memiliki ukuran yang lebih besar. Sistem klasifikasi yang dibuat telah memiliki rentang nilai ukuran objek secara tetap, jika terjadi penambahan ukuran tersebut hal yang mungkin terjadi adalah sistem menganggap dua atau lebih objek yang berdekatan tersebut menjadi sebuah mobil.

Kesalahan sistem tersebut dapat diatasi dengan meng-upgrade versi OpenCV yang memiliki fitur BackgroundSutractorGMG. Fitur tersebut mampu menghilangkan

noise sistem yang berupa bayangan. Fitur tersebut menurut referensi tersedia pada OpenCV versi 3.0.0. Sejauh penulis mengikuti perkembangan, OpenCV versi 3.0.0 masih berupa Release Candidate (RC) 1 pada bulan Mei 2015. Versi tersebut masih belum memiliki fitur BackgroundSubtractorGMG.

B. Analisis keakuratan sistem penghitung kendaraan

Analisis keakuratan sistem penghitung kendaraan bermotor ini menggunakan kondisi video 1, karena video 1 tidak memiliki noise berupa bayangan objek. Berbeda dengan video 1, video 2 cenderung memiliki noise yang komplek karena saat perekaman kondisi matahari membentuk sudut lancip sehingga bayangan yang terbentuk lebih lebar daripada video 1.Penghitungan tingkat akurasi sistem menggunakan rumus berikut:

$$\textit{presentasi akurasi sistem} = 100\% - \%\ \textit{error}$$

dengan rumus presentasi error sistem dihitung melalui rumus

$$\%\ \text{error} = \frac{selisih\ penghitungan\ manual\ dengan\ sistem}{penghitungan\ manual}\ x\ 100\%$$

Parameter yang diuji dalam analisis keakuratan sistem penghitung kendaraan ini menggunakan faktor nilai framerate video. Analisis pengaruh nilai framerate video diperlukan untuk menentukan dalam rentang berapa nilai framerate video memiliki tingkat akurasi yang optimal. Berikut hasil pengujian hubungan framerate dengan tingkat akurasi sistem untuk video 1:

Tabel 4.1 Hubungan nilai framerate video terhadap tingkat akurasi

Nilai framerate video (fps	Nilai penghitungan siste	Nilai penghitungan manual	Presentase error (%)	Presentase akurasi sistem (%
8	20	44	54,54	45,46
10	21	44	52,27	47,43
12	25	44	43,18	56,82
15	26	44	40,9	59,1
20	39	44	11,36	88,64
24	45	44	2,27	97,73
25	46	44	4,54	95,46
30	46	44	4,54	95,46

Tabel di atas menunjukkan hubungan nilai framerate video memiliki hasil maksimal pada nilai 24 fps. Nilai framerate maksimal yang diambil adalah 30 fps (*framerate per seconds*) karena kamera webcam yang dipakai mengeluarkan nilai *framerate* maksimal 30 fps. Nilai framerate video berbanding lurus terhadap tingkat akurasi yang dicapai karena semakin tinggi framerate sebuah video maka video tersebut akan lebih detail begitu pula sebaliknya.

Video adalah frame gambar yang bergerak, misalnya nilai framerate video adalah 10 fps (*frame per seconds*) maka dalam video tersebut tiap detiknya menampilkan 10 frame gambar. Jadi, semakin besar nilai *framerate* video akan semakin banyak gambar yang ditampilkan setiap detiknya atau dapat disebut lebih detail. Namun,

teori tersebut nampaknya tidak terjadi pada purwarupa ini. Hal tersebut dikarenakan faktor penghitungan objek yang berkelipatan. Artinya satu objek dapat terhitung lebih daripada satu saat nilai *framerate* tinggi, sedangkan saat nilai framerate rendah indikator *tracking* akan lebih cepat berpindah sehingga terjadi kemungkinan bahwa indikator tersebut tidak melewati nilai koordinat garis penghitung. Jadi secara teoritis semakin detail sebuah video memberikan frame gambarnya setiap maka semakin mudah image processing mengolah data koordinatnya. Namun, khusus untuk sistem ini terdapat kelemahan jika framerate terlalu besar. Solusi dari permasalahan ini adalah mengembangkan metode tracking pada purwarupa selanjutnya.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah melakukan perancangan sistem, dan membuat pengimplementasian purwarupa sistem klasifikasi dan penghitung Kendaraan dua jalur berbasis pengolah citra ini, maka dapat diambil kesimpulan sebagai berikut :

1. Berdasarkan hasil pengujian yang telah dilakukan, sistem klasifikasi telah mampu mengklasifikasi jenis kendaraan sepeda motor dan mobil.
2. Berdasarkan hasil pengujian yang telah dilakukan, sistem penghitung jumlah kendaraan mampu menghitung kendaraan di dua jalur
3. Nilai framerate video berbanding lurus dengan tingkat akurasi yang dihasilkan untuk sistem ini.

4. Nilai framerate video yang memberikan tingkat akurasi paling optimal terjadi pada rentang nilai 24 fps.
5. Kondisi jalan satu arah memiliki keakuratan hitung yang lebih baik dari kondisi jalan dua arah karena metode hitung yang menempatkan koordinat nilai absis (x) maupun ordinat (y) pada video.

5.2 Saran

Dari hasil pengujian penghitung otomatis ini diketahui bahwa masih terdapat beberapa kekurangan dan dimungkinkan untuk pengembangan lebih lanjut. Oleh karenanya, penulis merasa perlu untuk memberi saran demi peningkatan kualitas dan pengembangannya yaitu sebagai berikut:

1. Sistem tracking pada model jalan dua jalur ini masih sangat sederhana sehingga dapat ditingkatkan.
2. Untuk pengembangan lebih lanjut, bayangan dari objek asli yang bergerak dapat dieliminasi.
3. Algoritma penghitung ini masih memiliki kemungkinan menghitung satu objek menjadi lebih dari satu dalam kondisi tertentu, sehingga algoritma tersebut dapat dikembangkan.

DAFTAR PUSTAKA

[1]. Widiyardini S.T.,“Implementasi Deteksi Gerakan untuk sistem pemantauan ruangan menggunakan webcam”, Malang, 2010

[2]. Kadir, Abdul. Susanto, Adhi. 2012. *Pengolahan Citra Teori dan Aplikasi.* Yogyakarta

[3]. Amaluddin, F., dkk. 2015. *Klasifikasi Kendaraan Menggunakan Gaussian Mixture Model (GMM) dan Fuzzy Cluster Means (FCM), Jurnal EECCIS Volume 9 No.1, pp. 19-24.* Program Studi Magister Teknik Elektro. Universitas Brawijaya

[4]. Febriyanto, A. 2013. *Analisis Kinerja Background Substraction dan Haar-Like Feature Untuk Monitoring Pejalan Kaki Menggunakan Kamera Webcam, Laporan Tugas Akhir*. Fakultas Sains dan Teknologi. Universitas Islam Negeri Sunan Kalijaga

[5]. Solichin, Achmad. Harjoko, Agus. 2013. *Metode Background Substraction untuk Deteksi Objek Pejalan Kaki pada Lingkungan Statis.* Fakultas Teknologi Informasi. Universitas Budi Luhur.

[6]. Fadliasyah, S.si (2007). *Computer vision* dan *pengolahan citra* Yogyakarta: Andi

[7]. Stackoverflow,http://stackoverflow.com/review/suggested-edits/8575035 (diakses 22 April 2015, pukul 07.55)

[8]. Barandiaran, J., Cortez, A., Nieto, M., Otaegui, O., Sanchez, P., Unzueta, L., 2011. Adaptive Multi-Cue Background Subtraction for Robust Vehicle Counting and Classification. IEEE Transactions on Intelligent Transportation System Journal.