TUGAS AKHIR - SM141501

DETEKSI KENDARAAN MENGGUNAKAN *HISTOGRAM OF ORIENTED GRADIENTS* DAN *REAL ADABOOST*

TUFFAHATUL UMMAH
NRP 1211 100 048

Dosen Pembimbing
Dr. Dwi Ratna Sulistyaningrum, S.Si., MT.

JURUSAN MATEMATIKA
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember
Surabaya 2017

FINAL PROJECT - SM141501

VEHICLE DETECTION USING HISTOGRAM OF ORIENTED GRADIENTS AND REAL ADABOOST

TUFFAHATUL UMMAH
NRP 1211 100 048

Supervisor
Dr. Dwi Ratna Sulistyaningrum, S.Si.,MT.

DEPARTMENT OF MATHEMATICS
Faculty of Mathematics and Natural Science
Sepuluh Nopember Institute of Technology
Surabaya 2017

LEMBAR PENGESAHAN

DETEKSI KENDARAAN MENGGUNAKAN *HISTOGRAM OF ORIENTED GRADIENTS* DAN *REAL ADABOOST*

VEHICLE DETECTION USING HISTOGRAM OF ORIENTED GRADIENTS AND REAL ADABOOST

TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat
Untuk memperoleh gelar Sarjana Sains
Pada bidang minat Ilmu Komputer
Program Studi S-1 Jurusan Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember Surabaya

Oleh :
TUFFAHATUL UMMAH
NRP. 1211 100 048

Menyetujui,

Dosen Pembimbing,

Dr. Dwi Ratna Sulistyaningrum, S.Si., M.T.
NIP. 19690405 199403 2 003

Mengetahui,
Ketua Jurusan Matematika
FMIPA ITS

Dr. Imam Mukhlash, S.Si., M.T.
NIP. 19700831 199403 1 003

Surabaya, 19 Januari 2017

DETEKSI KENDARAAN MENGGUNAKAN *HISTOGRAM OF ORIENTED GRADIENTS* DAN *REAL ADABOOST*

Nama : Tuffahatul Ummah
NRP : 1211 100 048
Jurusan : Matematika
Dosen Pembimbing : Dr. Dwi Ratna S., S.Si., MT.

Abstrak

Sistem deteksi kendaraan merupakan salah satu teknologi yang sangat penting karena memiliki banyak aplikasi dalam bidang lalu lintas seperti pemantauan lalu lintas, penghitungan jumlah kendaraan yang lewat, penghitungan kecepatan kendaraan yang melaju, dan lain sebagainya. *Histogram Of Oriented Gradients* (HOG) adalah *descriptor* fitur yang digunakan untuk deteksi objek. HOG mendeskripsikan fitur berdasarkan histogram lokal dari orientasi gradien yang diberi bobot dengan *magnitude* gradien. *Real AdaBoost* adalah algoritma *learning* yang mengkombinasikan *weak classifier* menjadi *strong classifier* yang merepresentasikan output akhir dari classifier yang didorong. Penelitian ini bertujuan untuk mendeteksi kendaraan pada citra statis dengan menggunakan metode *Histogram Of Oriented Gradients* (HOG) dan *Real Adaboost.* Tahapan dari deteksi kendaraan ini yaitu pra-pemrosessan, proses ekstraksi fitur dengan HOG dan proses klasifikasi dengan *Real Adaboost.* Dari 257 citra kendaraan dan 35 citra bukan kendaraan didapatkan hasil pengujian dengan tingkat akurasi sebesar 91.78 %.

Kata kunci : Deteksi Kendaraan, Ekstraksi Fitur, *Histogram Of Oriented Gradients* (HOG), *Real Adaboost.*

“Halaman ini sengaja dikosongkan”

VEHICLE DETECTION USING HISTOGRAM OF ORIENTED GRADIENTS* AND *REAL ADABOOST

Name	**: Tuffahatul Ummah**
NRP	**: 1211 100 048**
Department	**: *Mathematics***
Supervisor	**: Dr. Dwi Ratna S., S.Si., MT.**

Abstract

Vehicle detection system is an important technology because it has many applications in traffic field such as traffic monitoring, counting the number of vehicles passing, calculating the speed of an oncoming vehicle, and so on. Histogram Of Oriented Gradients (*HOG*) *is a feature descriptor used for the purpose of object detection. HOG describes features based on local histogram of gradient orientation weighted by gradient magnitude. Real AdaBoost is a learning algorithm which combined weak classifier into strong classifier that represents the final output of the boosted classifier. This research aims to detect vehicles in static image using Histogram Of Oriented Gradients* (*HOG*) *methods and Real Adaboost. The steps of vehicle detection are pra-processing, feature extraction process with HOG and classification process with Real AdaBoost. The testing result shows that the system can detect vehicles with an accuracy level of 91.78 % from 292 testing image, from 257 image of vehicle and 35 image of not vehicle.*

Key words **:** *Vehicle Detection, Feature Extaction, Histogram Of Oriented Gradients (HOG), Real Adaboost.*

“Halaman ini sengaja dikosongkan”

KATA PENGANTAR

Alhamdulillaah wa syukurillaah penulis panjatkan kepada Allah SWT. karena dengan limpahan ni'mat dan rahmat-Nya yang tiada tara juga izin-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul: **"Deteksi Kendaraan Menggunakan *Histogram of Oriented Gradients* dan *Real AdaBoost*"** yang merupakan salah satu persyaratan dalam menyelesaikan Program Studi S-1 di Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam (FMIPA) Institut Teknologi Sepuluh Nopember Surabaya.

Tugas akhir ini dapat penulis selesaikan dengan baik walaupun pada awalnya jatuh bangun karena banyak kendala yang menghadang, tapi penulis bisa mendapatkan semangat kembali berkat kerja sama, bantuan, dan dukungan dari banyak pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Bapak Dr. Imam Mukhlash, S.Si, M.T. selaku Ketua Jurusan Matematika ITS.
2. Ibu Dr. Dwi Ratna Sulistyaningrum, S.Si., M.T. selaku dosen pembimbing yang senantiasa meluangkan waktunya untuk memberikan bimbingan dan dukungan kepada penulis.
3. Bapak Dr. Budi Setiyono, S.Si., M.T., Bapak Drs. Soetrisno, MI.Komp., Ibu Soleha, S.Si., M.Si., dan Bapak Drs. Komar Baihaqi, M.Si. selaku dosen penguji yang telah bersedia memberikan masukan berupa kritik dan saran yang bersifat membangun guna kesempurnaan tugas akhir ini.
4. Bapak Dr. Didik Khusnul Arif, S.Si., M.Si. selaku Ketua Program Studi Jurusan Matematika ITS.
5. Bapak Dr. Darmaji, S.Si., M.T. selaku dosen wali yang senantiasa membimbing dan menasehati penulis.
6. Seluruh jajaran dosen Matematika ITS yang dengan sabar dan ikhlas mendidik mahasiswa-mahasiswinya dan juga para staf jurusan Matematika ITS yang memberikan pelayanan dengan baik.

Penulis menyadari bahwa tugas akhir ini masih jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan saran dan kritik dari pembaca. Akhir kata, semoga tugas akhir ini bermanfaat bagi semua pihak.

Surabaya, Desember 2016

Penulis

special thanks to

1. Bapak Samsul Hadi dan Ibu Mas'ulah tercinta yang selalu memberikan dukungan, nasehat, dan doa pada penulis selama ini.
2. Mbak Nia, Zuhri, Dhiya' yang selalu memotivasi penulis untuk menyelesaikan tugas akhir ini.
3. Joko, Habib, Farah, Desy, Mbak Nurul terima kasih atas bantuan, semangat, dan dukungannya, semoga Allah SWT. membalas kebaikan kalian dengan pahala.
4. Anggit, Daul, terima kasih atas semuanya, jazaakillaahu khoiron.
5. Ruhul Jadid'ers, everlasting ukhuwah, terima kasih atas tiga tahun yang sangat berharga di kontrakan tercinta.
6. Squad Baitul Izzah, terima kasih atas satu semester menemani perjalanan penulis dalam menyelesaikan tugas akhir ini.
7. Teman-teman Pencitraan 115 di bawah bimbingan Ibu Dospem tercinta, dukungan dan semangat kalian sungguh berarti.
8. Forkom Akhwat 2011, *sharing* ilmu dengan kalian sungguh mengasyikkan.
9. Teman-teman seperjuangan Wisuda 115, menghabiskan waktu dengan kalian adalah waktu yang sangat berharga.
10. Keluarga besar Bani Tohir-Nizar dan Abdul Bari, teman-teman GEMBELIS, dan teman-teman MA Maskumambang angkatan 2011, terima kasih telah mengisi hari-hari penulis.

11. Kelompok Neyman Pearson dan SC Mbak Galuh, terima kasih atas bimbingan dan kebersamaan selama menjadi mahasiswa baru.
12. Squad Ibnu Muqlah 1213 dan 1314 dan teman-teman angkatan 2011 yang telah memberikan pengalaman dan kenangan berharga bagi penulis.
13. Takahashi Minami, AKB48, dan JKT48, terima kasih telah memberikan suntikan semangat melalui perjuangan kalian, penulis juga tidak akan menyerah begitu saja karena "*Doryoku wa kanarazu mukuwareru*".

Tentu saja masih banyak pihak lain yang turut andil dalam penyelesaian tugas akhir ini yang tidak bisa penulis sebutkan satu persatu. Semoga Allah membalas dengan balasan yang lebih baik bagi semua pihak yang telah membantu penulis. *Aaamiin yaa rabbal'aalamiin.*

“*Halaman ini sengaja dikosongkan*”

DAFTAR ISI

DAFTAR GAMBAR

“Halaman ini sengaja dikosongkan”

DAFTAR TABEL

“Halaman ini sengaja dikosongkan”

DAFTAR LAMPIRAN

“*Halaman ini sengaja dikosongkan*”

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Deteksi objek adalah salah satu objek penelitian yang penting dalam ilmu *computer vision* sehingga mendapatkan banyak perhatian di dalamnya. Berkembang pesatnya pertumbuhan komputer bertenaga tinggi, semakin banyaknya kamera berkualitas tinggi dan ekonomis, dan meningkatnya penelitian dalam analisis citra menjadikan banyak peneliti tertarik dalam menemukan algoritma deteksi objek. Deteksi objek memiliki aplikasi yang sangat luas, seperti robotika, analisis citra medis, pengawasan, dan interaksi *human-computer*.

Tujuan dari deteksi objek adalah untuk mendeteksi semua contoh objek dari *class* yang sudah diketahui, seperti kendaraan. Deteksi kendaraan merupakan tugas yang berat karena latar belakang dari kendaraan terletak di jalan yang sangat rumit dan kendaraan cenderung muncul dengan intensitas dan warna yang beragam. Penampilan objek dalam *scene* yang diamati juga berubah cukup sering bergantung pada orientasi kamera.

Banyak penelitian yang dilakukan dalam deteksi objek, seperti penelitian yang dilakukan oleh Satish Madhogaria, Paul M. Baggenstoss, Marek Schikora Wolfgang Koch, dan Daniel Cremers dengan judul *Car Detection by Fusion of HOG and Causal MRF* (2015) yang memperkenalkan algoritma dua tahap untuk deteksi mobil pada *aerial image* menggunakan metode HOG dan SVM [1]. Dengan menggunakan *Google Earth Data Set*, penelitian tersebut berhasil mendeteksi mobil dengan tingkat deteksi sebesar 78%. Lalu penelitian lain dilakukan oleh Daisuke Aoki dan Junzo Watada dengan judul *Human Tracking Method Based on Improved HOG+Real AdaBoost* [2]. Daisuke Aoki dan Junzo Watada mengusulkan sebuah metode untuk menyelesaikan masalah dari beberapa faktor (pakaian, tipe badan, arah, perubahan perspektif, perubahan cahaya) yang membuat sulit untuk mendeteksi manusia. Mereka mengkombinasikan fitur

Histogram of Oriented Gradients (HOG) dengan *boosting* dua-tahap dan metode *Real AdaBoost*. Dengan pengujian pada tiga pola data uji yang berbeda, didapatkan 17% *error* untuk pola pertama dengan latar belakang sederhana, 49,5% dengan latar belakang lebih kompleks, dan 38,7% dengan perubahan ukuran manusia yang dideteksi. Berdasarkan penelitian yang telah dipaparkan, penulis dapat menyimpulkan bahwa metode HOG dan metode *AdaBoost* memiliki tingkat akurasi yang cukup tinggi dalam deteksi objek.

HOG (*Histogram of Oriented Gradients*) pertama kali diperkenalkan oleh Dalal dan Triggs (2005) untuk deteksi manusia [3]. Namun dalam perkembangannya, penelitian dilakukan untuk mendeteksi objek yang beragam, seperti kendaraan, wajah manusia, pengenalan karakter tulisan manusia, dan lain-lain. Metode HOG sebagai metode ekstraksi fitur lebih unggul daripada metode ekstraksi fitur lain karena HOG beroperasi pada *cell* lokal dan penampilan lokal objek dan bentuk bisa dikarakterisasi lebih baik oleh distribusi *gradient* intensitas lokal atau *edge direction* [3], selain itu HOG *invariant* (tidak berubah) untuk transformasi *geometric* dan *photometric* karena ketika *cell* dirotasi maupun ditranslasi tidak akan memberikan pengaruh pada nilai HOG. Selanjutnya, *AdaBoost* sebagai metode *machine learning* telah digunakan secara luas untuk klasifikasi data dan deteksi objek karena kekuatan dan efisiensinya. Hal itu dikarenakan *AdaBoost* mengkombinasikan *weak classifier* berdasarkan sampel yang diboboti ulang untuk membentuk *strong classifier*, artinya sampel yang diklasifikasikan salah akan ditambahi bobotnya dan sampel yang diklasifikasikan benar akan dikurangi bobotnya, sehingga kombinasi tersebut dapat meningkatkan performa klasifikasi. Salah satu varian *AdaBoost* adalah *Real AdaBoost*, yang menghitung *weak hypothesis* dengan mengoptimalkan batas atas dari *training error* sehingga berkonvergensi lebih cepat daripada *AdaBoost* dan waktu komputasi juga lebih cepat [4]. Berdasarkan penjelasan di atas,

maka Tugas Akhir ini membahas mengenai deteksi kendaraan menggunakan HOG dan *Real AdaBoost*.

1.2 Rumusan Masalah

Rumusan masalah dari Tugas Akhir ini adalah:

1. Bagaimana mengimplementasikan metode HOG dan *Real AdaBoost* untuk deteksi kendaraan yang terdapat pada suatu citra?
2. Bagaimana performansi dan tingkat keberhasilan metode HOG dan *Real AdaBoost* untuk deteksi kendaraan yang terdapat pada suatu citra?

1.3 Batasan Masalah

Batasan-batasan yang ada dalam pembuatan Tugas Akhir ini adalah:

1. Sampel *training* yang masuk ke dalam sistem di-*resize* ukuran dimensinya menjadi 128 piksel × 128 piksel.
2. Jenis kendaraan adalah sepeda motor, mobil, dan truk/bus.
3. Objek kendaraan pada citra *training* memiliki prosentase luas minimal 30 % dari luas frame citra.

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah:

1. Mengimplementasikan metode HOG dan *Real AdaBoost* untuk deteksi kendaraan.
2. Menganalisa performansi dan tingkat keberhasilan metode HOG dan *Real AdaBoost* untuk deteksi kendaraan.

1.5 Manfaat

Manfaat dari Tugas Akhir ini adalah:

1. Sebagai alternatif untuk pelacakan kendaraan sehingga nantinya dapat digunakan untuk analisis pengaturan lalu lintas, pengaturan sistem parkir, dan lain-lain.
2. Sebagai referensi dan informasi tentang penggunaan metode HOG dan *Real AdaBoost*.

1.6 Sistematika Penulisan Tugas Akhir

Sistematika penulisan Tugas Akhir ini disusun berdasarkan beberapa bab sebagai berikut:

1. BAB I Pendahuluan
   Bab ini menjelaskan tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan dan manfaat dari penelitian yang dilakukan, serta sistematika penulisan Tugas Akhir.
2. BAB II Kajian Teori
   Bab ini menjelaskan tentang studi penelitian sebelumnya dan beberapa teori pendukung yang berkaitan dengan Tugas Akhir ini.
3. BAB III Metodologi Penelitian
   Bab ini menjelaskan tentang langkah-langkah yang digunakan untuk menyelesaikan Tugas Akhir ini.
4. BAB IV Perancangan dan Implementasi
   Bab ini menjelaskan tentang perancangan dan implementasi sistem, serta proses pembuatan sistem secara keseluruhan.
5. BAB V Pengujian dan Pembahasan Hasil
   Bab ini menjelaskan tentang hasil uji coba sistem. Kemudian dicatat sebagai bahan untuk merumuskan beberapa kesimpulan dari Tugas Akhir ini.
6. BAB VI Kesimpulan dan Saran
   Bab ini berisi tentang kesimpulan dari hasil penelitian dan saran untuk pengembangan dari Tugas Akhir ini.

BAB II
KAJIAN TEORI

2.1 Penelitian Sebelumnya

Beberapa penelitian sebelumnya yang terkait dengan tugas akhir ini yaitu penelitian yang dilakukan oleh Daisuke Aoki dan Junzo Watada dengan judul *Human Tracking Method Based on Improved HOG+Real AdaBoost* [2]. Daisuke Aoki dan Junzo Watada mengusulkan sebuah metode untuk menyelesaikan masalah dari beberapa faktor (pakaian, tipe badan, arah, perubahan perspektif, perubahan cahaya) yang membuat sulit untuk mendeteksi manusia. Mereka mengkombinasikan fitur *Histogram of Oriented Gradients* (HOG) dengan *boosting* dua-tahap dan metode *Real AdaBoost.* Dengan pengujian pada tiga pola data uji yang berbeda, didapatkan 17% *error* untuk pola pertama dengan latar belakang sederhana, 49,5% dengan latar belakang lebih kompleks, dan 38,7% dengan perubahan ukuran manusia yang dideteksi.

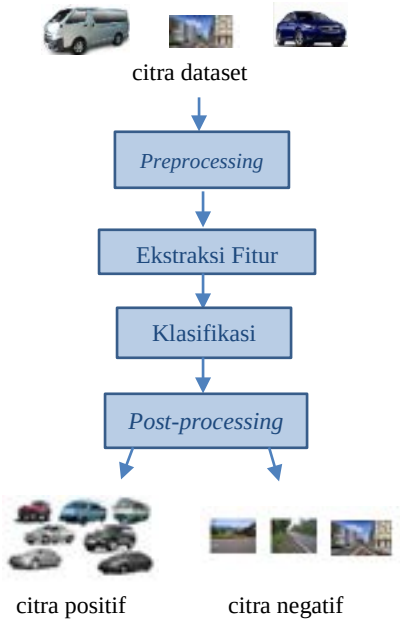
Lalu penelitian lain oleh Satish Madhogaria, Paul M. Baggenstoss, Marek Schikora Wolfgang Koch, dan Daniel Cremers dengan judul *Car Detection by Fusion of HOG and Causal MRF* (2015) yang memperkenalkan algoritma dua tahap untuk deteksi mobil pada *aerial image* menggunakan metode HOG dan SVM [1]. Dengan menggunakan *Google Earth Data Set*, penelitian tersebut berhasil mendeteksi mobil dengan tingkat deteksi sebesar 78%.

2.2 Deteksi Objek

Deteksi objek adalah teknologi komputer yang bertujuan untuk mendeteksi dan melokalisir objek dari kategori yang sudah dikenal (*class*) dalam citra digital statis atau frame video [5]. Deteksi objek memiliki banyak kegunaan, misalnya digunakan untuk penginderaan jauh pada citra geologi bumi, geografi bumi, atau lingkungan hidup. Penginderaan jauh menggunakan sistem deteksi objek untuk merekam foto udara maupun foto satelit, lalu

dilakukan identifikasi, pengenalan, analisis, dan klasifikasi. Selain itu bisa digunakan di bidang robotika untuk mendeteksi objek berdasarkan warna dengan sensor kamera. Deteksi objek juga bisa digunakan untuk kamera pengawas CCTV pada sebuah gedung atau pada jalan raya. Sistem deteksi objek digunakan untuk mendeteksi gerakan dengan menggunakan kamera pengawas sebagai input untuk menampilkan gambar.

Sistem deteksi objek biasanya terdiri dari dua prosedur utama, yaitu ekstraksi fitur dan klasifikasi. Langkah ekstraksi fitur yaitu menyandikan tampilan visual citra dalam fitur yang sudah dikenal, sedangkan langkah klasifikasi yaitu menentukan, dengan menggunakan fitur yang diekstraksi sebelumnya dan *classifier* khusus, apakah sebuah daerah/citra mengandung *object of interest* atau tidak.



Gambar 2.1 Struktur umum sistem deteksi objek [6]

Pada Gambar 2.1 ditunjukkan struktur umum dari sistem deteksi objek. Pertama citra sampel sebagai data *input* dimasukkan ke sistem untuk *preprocessing* sebelum dilakukan proses ekstraksi fitur. *Preprocessing* citra bertujuan untuk perbaikan gambar dari distorsi yang tidak diinginkan atau untuk meningkatkan beberapa fitur citra untuk diproses pada langkah selanjutnya. Contoh *preprocessing* misalnya *resizing* yaitu mengubah resolusi atau ukuran vertikal dan horizontal dari suatu citra. Lalu *grayscaling* yaitu proses konversi warna RGB ke *grayscale*. Citra berwarna (*Red*, *Green*, *Blue*) akan diubah ke dalam format *grayscale* sehingga warnanya akan menjadi lebih sederhana yaitu keabuan. Konversi RGB ke *grayscale* menggunakan persamaan *grayscale* = 0.299 × R + 0.587 × G + 0.11 × B. Setelah itu dilakukan ekstraksi fitur untuk mendapatkan fitur citra. Fitur citra tersebut nantinya yang akan digunakan untuk proses klasifikasi pada proses selanjutnya, dan akan terklasifikasi citra positif dan negatif berdasarkan data *training*.

2.3 Ekstraksi Fitur

Ekstraksi fitur yaitu proses transformasi dari kumpulan *input* data yang diberikan menjadi kumpulan fitur (*feature vector* yaitu fitur yang dikelompokkan dari data yang diberikan) [7]. Pada proses ini, komponen pada citra diekstrak dan dikarakterisasi. Fitur adalah atribut individual atau properti dari sebuah objek yang relevan dalam mendeskripsikan dan mengenali objek dan bisa membedakannya dari objek lain. Fitur suatu objek dapat dikenali bisa berdasarkan tekstur, warna, maupun bentuk, atau penggabungan ketiganya. Ada beberapa metode yang digunakan untuk ekstraksi fitur berbasis tekstur, misal *Wavelet Correlogram*, *Gabor Wavelet Correlogram*, *Gray Level Coocurrence Matrices*, *Local Binary Patterns* (LBP) dan lain sebagainya. Metode yang digunakan untuk ekstraksi fitur berbasis warna misalnya *Fuzzy Colour Histogram* (FCH). Kemudian metode yang digunakan untuk ekstraksi fitur berbasis

bentuk misalnya metode *Histogram of Oriented Gradients* (HOG).

2.3.1 *Histogram of Oriented Gradients*

Histogram of Oriented Gradients (HOG) adalah sebuah fitur yang mendeskripsikan distribusi arah spasial (ruang) pada setiap daerah citra. Gagasan dari HOG adalah bahwa penampilan objek lokal dapat digambarkan cukup baik dengan distribusi gradien intensitas lokal atau arah tepi. HOG diperkenalkan oleh Naveet Dalal dan Bill Trigs pada tahun 2005. Tujuan utamanya yaitu untuk mengembangkan *detector* yang *robust* (kuat) untuk mendeteksi manusia dengan penampilan secara penuh pada video [3]. *Robust* di konteks ini artinya manusia yang dideteksi bisa jadi berjalan, berlari, dari sisi samping atau membungkuk. HOG juga bisa digunakan untuk mendeteksi objek berbasis bentuk yang lain [3]. Pada HOG, histogram berisi *channel-channel* arah/orientasi gradien dari piksel-piksel pada citra, dimana penampilan serta bentuk objek dapat diketahui melalui hasil komputasi gradien dari citra. Ide dasar dari HOG adalah dengan membagi citra menjadi daerah spasial kecil (*cell*), untuk tiap daerah, dibuat histogram orientasi gradien yang didapatkan dari perhitungan *magnitude* gradien dan orientasi gradien dari semua piksel dalam daerah tersebut. Kemudian nilai orientasi tiap pikselnya dikuantisasi ke dalam 9 *bin* (kanal) menggunakan histogram. Kontribusi piksel terhadap tiap *bin* bergantung pada nilai *magnitude* gradiennya. Histogram-histogram tersebut kemudian dinormalisasi dan digabungkan untuk mendapatkan *descriptor* akhir.

Langkah-langkah ekstraksi fitur dengan HOG adalah sebagai berikut [2, 11]:

1) Normalisasi Gamma atau *Square-Root* dan Warna

Tahap pre-processing citra ialah normalisasi warna dan nilai gamma atau *square-root*. Tahap ini bisa diabaikan karena memiliki pengaruh yang kecil pada kinerja [3]. Pada tahap normalisasi gamma dihitung $\sqrt{p}$ (normalisasi *square-root*) atau $\log(p)$ (normalisasi gamma) dari setiap piksel p pada

citra input. Definisi normalisasi *square-root* yaitu meng-*compress* (memampatkan) intensitas piksel jauh lebih kecil daripada normalisasi gamma. Berdasarkan penelitian Dalal dan Triggs, normalisasi *square-root* dapat meningkatkan akurasi. Sedangkan dengan menggunakan normalisasi gamma/*power law*, pendekatan ini mungkin merupakan "*over-correction*" dan menurunkan kinerja. Selain dilakukan normalisasi gamma, juga dilakukan normalisasi warna. Citra RGB akan diproses menjadi citra *grayscale* untuk memudahkan penghitungan gradien citra. Citra *grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pikselnya, artinya nilai *Red = Green = Blue*. Citra yang ditampilkan dari citra jenis ini terdiri atas warna abu-abu, bervariasi pada warna hitam pada bagian intensitas terlemah dan warna putih pada intensitas terkuat.

2) Komputasi Gradien

Menggunakan informasi gradien untuk mengolah citra bisa membantu mendapatkan informasi *contour* citra dan mengurangi pengaruh perubahan cahaya. Pada tahap ini, akan dihitung gradien vertikal $f_x(x, y)$ dengan Persamaan (2.1) dan gradien horizontal $f_y(x, y)$ dengan Persamaan (2.2), sehingga untuk nilai piksel citra tepi tidak berubah. Setelah mendapatkan nilai informasi gradien, akan dihitung orientasi gradien (θ) dengan Persamaan (2.4) dan magnitude (m) dengan Persamaan (2.3) untuk setiap piksel pada citra [2].

$$f_x(x, y) = I(x + 1, y) - I(x - 1, y) \quad (2.1)$$

$$f_y(x, y) = I(x, y + 1) - I(x, y - 1) \quad (2.2)$$

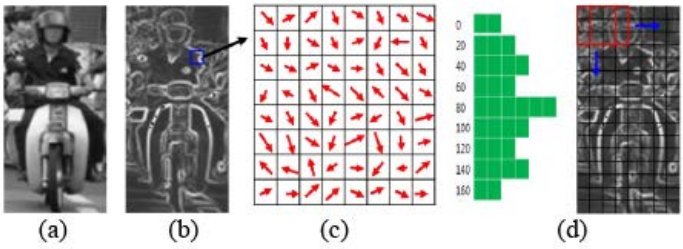
dengan $I(x, y)$ adalah nilai intensitas citra di (x, y)

$$m(x, y) = \sqrt{f_x(x, y)^2 + f_y(x, y)^2} \quad (2.3)$$

$$\theta(x, y) = tan^{-1}\left(\frac{f_y(x,y)}{f_x(x,y)}\right) \quad (2.4)$$

Untuk gambar berwarna, gradien dihitung secara terpisah untuk tiap *channel* warna, dan untuk tiap piksel, dipilih *channel* dengan berat gradien tertinggi.

Contoh hasil perhitungan *magnitude* gradien sebuah citra dapat dilihat pada Gambar 2.2 (b) dan Gambar 2.3 (b). Dapat dilihat bahwa tepi objek terlihat lebih jelas karena gradien horizontal arah *y* menghasilkan tepi objek berupa garis vertikal dan diagonal dari citra input, gradien arah vertikal *x* menghasilkan tepi objek berupa garis horizontal dan diagonal dari citra input sedangkan total gradien merupakan gabungan dari gradien arah *x* dan gradien arah *y*.



Gambar 2.2 Proses Ekstraksi Fitur HOG (a) Citra asli (b) Menghitung *magnitude* (c) *Cell* dan orientasi *bin* (d) Normalisasi *block* untuk 3 × 3 *cell* [8]

3) Menentukan Orientasi *Bin*

Ekstraksi HOG adalah pendekatan *window* tunggal, citra dibagi menjadi daerah yang disebut *block*, dan pada saat yang sama, tiap *block* dibagi menjadi daerah yang lebih kecil yang disebut *cell*. Satu histogram tiap *cell* diekstrak lalu didapatkan satu *descriptor* per *block* lalu digabungkan. *Overlapping* (tumpang tindih) antara *block* bertujuan untuk memastikan konsistensi di seluruh citra dan mengurangi pengaruh variasi lokal.

Setiap histogram memiliki sejumlah *bin* yang sama, yang menentukan ketelitiannya. *Bin* merepresentasikan orientasi gradien (*angle*) dan harus sama ditempatkan pada

$0° - 180°$ untuk gradien "*unsigned*" atau $0° - 360°$ untuk gradien "*signed*". Satu histogram tiap *cell* dihitung; tiap piksel pada *cell* berkontribusi pada penambahan histogram dimana histogram tersebut bergantung pada nilai *magnitude*-nya yang berhubungan dengan orientasi gradien, artinya sebuah *bin* dipilih berdasarkan arah (*angle*) dan *vote* (nilai yang akan diberikan pada *bin*) dipilih berdasarkan *magnitude*. *Vote* bisa jadi fungsi dari *magnitude*, tetapi terbukti bahwa menggunakan secara langsung nilai *magnitude* itu sendiri dapat memberikan hasil yang lebih baik. Jika diperlukan, *vote* bisa juga diinterpolasi (ditambahkan) secara bilinear antara *bin* bertetangga yang berbeda.

Gambar 2.2 (c) menunjukkan proses orientasi bin pada sebuah *cell*. Dapat dilihat bahwa tiap piksel citra terdapat panah-panah yang menunjukkan gradien, dimana panah menunjukkan arah gradien yang menggambarkan perubahan intensitas dan panjangnya menunjukkan *magnitude* yang menggambarkan seberapa besar perbedaan perubahan intensitas.

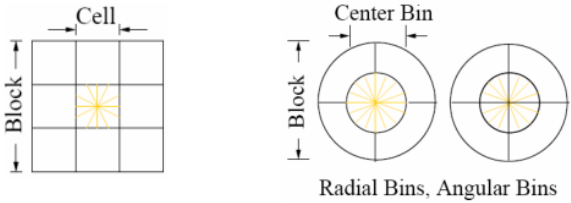
4) Normalisasi *Block*

Normalisasi kontras lokal yang efektif adalah hal yang penting untuk menyelesaikan permasalahan pencahayaan dan variasi *foreground-background*. *Cell-cell* digabung untuk membentuk sebuah *block*. *Block* bisa jadi berbentuk *rectangular* (R-HOG) atau *circular* (C-HOG) seperti ditunjukkan pada Gambar 2.3. Kemudian histogram yang besar dibentuk dengan mengkombinasikan semua histogram yang dibuat di dalam sebuah *block*. Setelah dikombinasikan, histogram dinormalisasi dengan Persamaan (2.5) berikut [9]:

$$v_n = \frac{v_i}{\sqrt{\|v\|_2^2 + \epsilon^2}} \qquad (2.5)$$

dengan v_n adalah vektor yang dinormalisasi, v_i adalah vektor fitur yang berhubungan dengan histogram yang digabungkan

untuk satu *block*, bentuk dari $\|v\|_2$ adalah $\sqrt{\sum_i x_i^2}$ sehingga $\|v\|_2^2 = v_1^2 + v_2^2 + \cdots + v_{36}^2$ jika satu *block* berukuran 2 × 2 *cell* dan ϵ suatu konstanta kecil bernilai 1 untuk menghindari pembagian oleh nol.



(a) (b)

Gambar 2.3 Bentuk *Cell* (a) R-HOG (b) C-HOG [3]

Setelah itu histogram untuk setiap *block* dikumpulkan untuk membuat vektor fitur berukuran 1-D. Selanjutnya vektor fitur tersebut ditempatkan dalam algoritma *learning* yang akan memutuskan apakah kendaraan terdeteksi atau tidak berdasarkan data *training*.

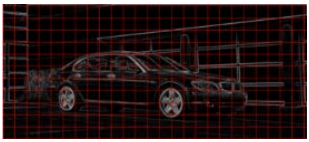
Untuk perhitungan vektor fitur yang dihasilkan, misalkan untuk sebuah citra berukuran 128 × 128, ukuran *cell* 8 × 8, dan ukuran *block* 16 × 16 dimana satu *block* terdiri dari dua *cell*, maka menghasilkan 225 *block* dalam sebuah citra karena satu *block* bisa *slid* 15 kali ketika orientasi horizontal dan vertical (*block* = 128 : 8 – 1 × 128 : 8 – 1). Sementara itu, terdapat 9 *channel bin* dalam satu *cell* dan 36 ((2 × 2) × 9) fitur dalam satu *block*. Maka, akan didapatkan jumlah fitur adalah 8100 (225 × 36) untuk citra berukuran 128 × 128. Secara ringkas, panjang fitur HOG bisa dihitung sebagai berikut:

Panjang fitur HOG
= *Block* × *Cell* Per *Block* × *Bin* Per *Cell*
= (128 : 8-1) × (128 : 8-1) × (2 × 2) × 9
= 15 × 15 × 4 × 9
= 8100

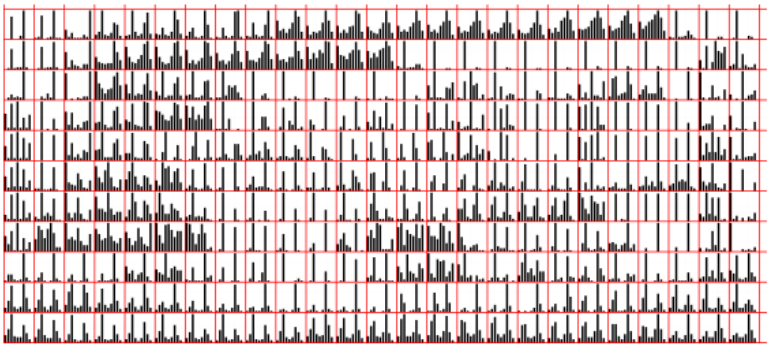
Gambar 2.4 (d) menunjukkan vektor fitur dari citra mobil yang diperoleh dari penghitungan gradien citra untuk mendapatkan *magnitude* dan orientasi gradiennya dan kemudian setiap *cell* dihitung orientasi *bin*nya berdasarkan arah (*angle*) yang berhubungan dengan *magnitude*.

(a)

(b)



(c)



(d)

Gambar 2.4 (a) Citra awal (b) *Magnitude* gradien (c) Contoh pembagian *cell* (d) *Descriptor Histogram of Oriented Gradients* (dinormalisasi dalam setiap *cell*) [3]

2.4 Klasifikasi menggunakan Algoritma *Real AdaBoost*

Klasifikasi adalah proses di mana unsur-unsur individu dikelompokkan menurut kesamaan antara elemen dan deskripsi grup. Dengan kata lain, klasifikasi yaitu menghubungkan kategori

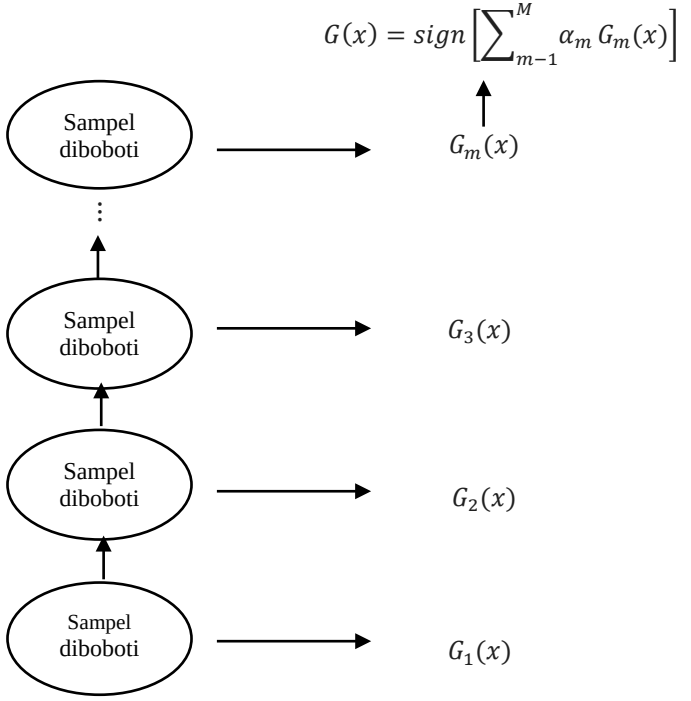
standar atau label kelas kepada contoh tertentu dari dataset input, kemudian memisahkan data input ke dalam kategori yang berbeda (kelas) dengan fitur-fitur umum.

AdaBoost adalah algoritma mesin pembelajaran yang diusulkan oleh Y. Freund dan R. Scaphire, yang memiliki tingkatan yang tinggi dari rasio deteksi dan mudah diimplementasikan. *AdaBoost* adalah metode *ensemble* (atau *meta-learning*) yang menyusun sebuah *classifier* dalam mode iteratif. Algoritma *boosting* dapat meningkatkan performa *weak learner L* dengan memanggil secara iterasi pada distribusi dari *training set* yang berbeda untuk menemukan beberapa *weak classifier h* dan kemudian mengkombinasikannya menjadi *strong classifier H.* Pada setiap iterasi, *AdaBoost* memanggil algoritma *learning* sederhana (disebut *base learner*/*weak learner*) yang kembali pada sebuah *classifier*, dan memberikan koefisien berat pada *classifier* tersebut [4]. Pada proses ini, *AdaBoost* meningkatkan berat dari *instance* yang mengalami kesalahan klasifikasi (*misclassified*) dan mengurangi berat dari *instance* yang terklasifikasi dengan benar oleh *weak learners* sebelumnya, sehingga pada iterasi selanjutnya *AdaBoost* akan fokus pada *instance* yang sulit diklasifikasikan, dengan kata lain, sampel yang tidak diklasifikasikan dengan benar oleh *classifier* di dalam rangkaian akan dipilih lebih sering dibandingkan dengan sampel yang diklasifikasikan dengan benar. Dengan demikian, *AdaBoost* mencoba menghasilkan *weak classifier* baru yang lebih baik untuk memprediksikan sampel yang pada *weak classifier* sebelumnya memiliki performansi yang buruk.

Klasifikasi terakhir akan diputuskan oleh "*vote*" berat dari *weak classifiers.* Lebih kecil *error* dari *weak classifiers*, maka lebih besar beratnya pada *vote* terakhir. Hasil *vote weak classifier* itu ditambahkan bersama dan diambil *signum* (fungsi *signum* adalah fungsi matematika untuk memperoleh nilai real) sebagai label yang diprediksi (-1 atau +1) untuk diberikan kepada *instance* x_i , dan *magnitude* $|h(x)|$ sebagai "*confidence*" (kepercayaan) pada prediksi ini. Sehingga, jika nilai $|h(x)|$ dekat

atau jauh dari nol, maka diartikan sebagai prediksi kepercayaan rendah atau tinggi. Skema *AdaBoost* diperlihatkan pada Gambar 2.5. $G_1, \dots, G_m$ adalah *weak classifier* yang didapatkan dari *training* oleh *weak learner* pada sampel yang diboboti. Kemudian akan didapatkan *strong classifier* $G(x)$ dari kombinasi *weak classifier* dan *alpha*-nya.

AdaBoost memiliki output binary +1 dan -1, sementara *Real AdaBoost* adalah versi peningkatan dari *AdaBoost* dimana *weak learner*nya diperbolehkan memiliki output *real number* $f_m(x) \in R$. *Signum* (tanda) dari output memberikan label prediksi [-1, +1] dan *magnitude*nya (besar) memberikan sebuah ukuran *confidence* pada prediksi ini. Dibandingkan dengan *AdaBoost*, *Real AdaBoost* memiliki tingkat akurasi lebih tinggi karena jumlah *weak classifier* sudah ditetapkan [13].



Gambar 2.5 Skema *AdaBoost* secara umum

Algoritma *Real AdaBoost* [2, 10] adalah sebagai berikut:

1. Diberikan: sampel *training* $S = \{(x_1, y_1), \dots \dots, (x_m, y_m)\}$, ruang *weak classifier h*. Dalam himpunan *S*, $x \in X$ adalah sampel vektor, *m* adalah banyak sampel, $y = \pm 1$ adalah label kelas yang berhubungan dengan sampel *training* x_i, dimana ketika x_i adalah citra dengan kendaraan maka $y = +1$, dan $y = -1$ jika yang lain.
2. Inisialisasi bobot sampel adalah $D_t(i) = \frac{1}{m}, i = 1, \dots, m$.
3. Ambil *weak learner*, untuk $t = 1, \dots, T$ (T adalah jumlah *weak learner* yang dipilih):
   a. Untuk tiap *weak classifier h* lakukan:
      i. Bagi sampel vektor *X* menjadi $x_1, \dots \dots, x_n$.
      ii. Dengan berat sampel training D_t, hitung probabilitas kepadatan distribusi:
      $$W_l^j = P(x_i \in x_j, y_i = l) = \sum_{i:x_i \in x_j \wedge y_i = l} D_t(i)$$
      dengan $l = \pm 1$.
      Probabilitas kepadatan distribusi W_l^j dihasilkan dengan menghitung fitur sampel *training* x_i dan menerapkan berat sampel *training* $D_t(i)$ ke bilangan BIN j dari histogram *one-dimensional* yang berhubungan dengan nilai fitur.
      iii. Atur *output* dari *weak classifier*
      $$h_t(x) = \frac{1}{2} \ln \left(\frac{W_{+1}^j + \varepsilon}{W_{-1}^j + \varepsilon} \right)$$
      dengan ε adalah konstanta positif kecil = 0.0000001.
      iv. Hitung faktor normalisasi
      $$z = 2 \sum_j \sqrt{W_{+1}^j W_{-1}^j}$$
      z merepresentasikan derajat pemisahan antara distribusi. Nilai yang kecil dari z menunjukkan pemisahan jumlah antara kelas distribusi positif dan negatif. Lebih kecil nilainya maka lebih besar

pemisahannya. Oleh karena itu, nilai terkecil dari *z* digunakan untuk memilih *classifier* terlemah pada tiap siklus.

b. Pilih h_t untuk memperkecil Z

$$Z_t = \min_{h \in H} Z$$

$$h_t = \arg\min_{h \in H} Z$$

c. Update bobot dari sampel *training*

$$D_{t+1}(i) = \frac{D_t(i)\exp[-y_i h_t(x_i)]}{Z_t}$$

Z_t adalah faktor inisialisasi untuk membuat D_{t+1} menjadi sebuah p.d.f.

4. *Strong classifier* yang terakhir adalah:

$$H(x) = sign\left[\sum_{t=1}^{T} h_t(x) - \theta\right]$$

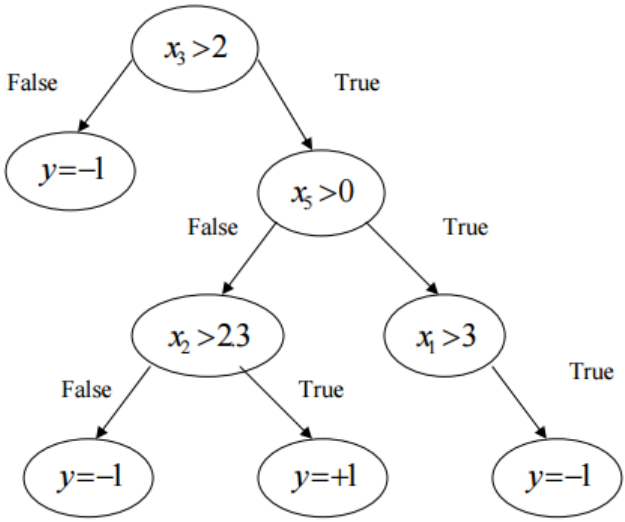
dimana θ adalah sebuah *threshold* yang bernilai nol.

Confidence dari *H* didefinisikan sebagai:

$$Conf_H(x) = |\textstyle\sum_t h_t(x) - \theta|$$

Weak classifier dapat berupa algoritma *learning* yang bisa membuat prediksi sedikit lebih baik daripada menebak secara acak [14]. *Classification and Regression Trees* (CART) (Breiman dkk, 1984) adalah salah satu metode klasifikasi yang digunakan dalam metode *boosting.* CART adalah sebuah pohon graph, sebagaimana ditunjukkan pada Gambar 2.6, dimana *leave* (daun) merepresentasikan hasil klasifikasi dan *node* (simpul) merepresentasikan hipotesis yang dibuat selama *training.* Cabangnya ditandai dengan *true* atau *false.* Untuk mengklasifikasikan sampel tunggal, nilai *input* yang sesuai akan diberikan pada *root* (x_3 pada Gambar 2.6), lalu turun, sampai mencapai daun (nilai yang berhubungan dengan daun adalah kelas dari sampel yang diberikan). Pada setiap langkah, dihitung

nilai prediksi yang berhubungan dengan *node* saat itu, lalu dipilih *node* selanjutnya (atau daun) yang terhubung dengan saat itu oleh cabang dengan nilai prediksi *node* saat ini.



Gambar 2.6 Contoh CART

Pada setiap *split* (pemisahan), CART memilih atribut yang memberikan *information gain* tertinggi atau *Gini impurity* terkecil.

1. *Gini Impurity*

Gini impurity menghitung seberapa sering elemen yang dipilih secara acak dari himpunan akan salah diberi label jika secara acak diberi label sesuai dengan distribusi label di *subset* tersebut. *Gini impurity* bisa dihitung dengan menjumlahkan probabilitas f_i item dengan label i yang dipilih, dikalikan dengan, probabilitas 1 - f_i kesalahan dalam mengkategorikan item tersebut. *Gini impurity* bernilai nol jika kategori item tunggal.

Jika ada beberapa kelas/kategori j, $i \in \{1, 2, \ldots, j\}$, f_i adalah *fraction* (pecahan) dari item yang dilabeli dengan *class* i pada himpunan.

$$I_G(f) = \sum_{i \neq k} fifk$$

2. *Information Gain*
   - Entropy: $H(T) = I_E(p_1, p_2, \dots, p_n) = -\sum_{i=1}^{j} p_i log_2 p_i$. dengan p_i adalah proporsi *T* pada kelas *i*
   - Gain :

$$\boldsymbol{Gain(F, A)} = \boldsymbol{Entropy(F)} - \sum_{\boldsymbol{v \in Values(A)}} \frac{|\boldsymbol{S_v}|}{|\boldsymbol{S}|} \boldsymbol{Entropy(S_v)}$$

“*Halaman ini sengaja dikosongkan*”

BAB III
METODOLOGI PENELITIAN

Dalam pengerjaan Tugas Akhir ini ada beberapa tahapan, antara lain:

3.1 Tahap Penelitian

1. Studi literatur.
   Pada tahap pertama ini akan dilakukan pengkajian tentang deteksi objek, proses ekstraksi fitur menggunakan *Histogram of Oriented Gradients*, dan proses klasifikasi menggunakan algoritma *Real AdaBoost.* Studi ini dilakukan dengan membaca buku, jurnal, artikel yang terkait, dan diskusi dengan dosen pembimbing maupun teman-teman yang paham dengan *image processing*.
2. Pengumpulan data.
   Pengumpulan data dilakukan untuk memperoleh informasi yang dibutuhkan untuk mencapai tujuan Tugas Akhir. Data yang digunakan dalam Tugas Akhir ini berupa citra kendaraan meliputi sepeda motor, mobil, dan bus/truk. Data terdiri dari data *training* positif (kendaraan) dan data *training* negatif (bukan kendaraan) dan data *testing* positif dengan citra kendaraan yang bervariatif di jalan raya dan data *testing* negatif. Proses pengumpulan data dilakukan secara manual dengan menggunakan kamera digital, pencarian di *Google* dan *Flickr*, dan diambil sebagian dari INRIA *Car Dataset*.
3. Perancangan dan implementasi aplikasi deteksi kendaraan.
   Aplikasi deteksi kendaraan akan dirancang dengan menggunakan metode *Histogram of Oriented Gradients* dan *Real AdaBoost*.
4. Uji coba dan evaluasi
   Uji coba dan evaluasi dilakukan dengan melakukan simulasi sistem deteksi terhadap citra kendaraan dan citra bukan kendaraan. Proses ini untuk membantu memahami cara penggunaan sistem dan menganalisis hasil uji coba

dan evaluasi sehingga dapat diberikan saran untuk pengembangan penelitan selanjutnya Kemudian, untuk mengetahui seberapa kuat sistem deteksi yang telah dibangun, nantinya juga akan ditambahkan *noise* (derau) pada sebagian citra *testing* kendaraan.

3.2 Blok Diagram Deteksi Kendaraan dengan HOG dan *Real AdaBoost*

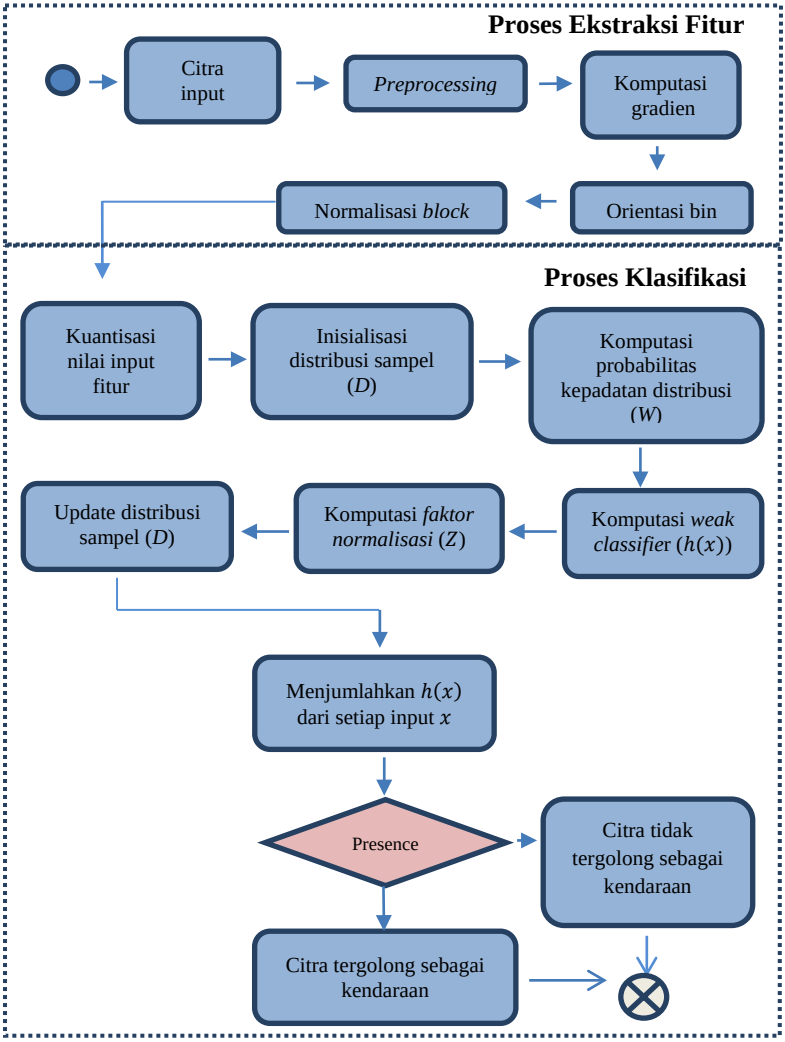
Gambar 3.1 menunjukkan tahapan deteksi kendaraan menggunakan *Histogram of Oriented Gradients* dan *Real AdaBoost* yang terdiri dari beberapa proses utama, yaitu *preprocessing*, ekstraksi fitur dari citra input menggunakan metode HOG, proses training menggunakan metode *Real AdaBoost*, dan proses testing.

Beberapa proses utama yang dilakukan adalah sebagai berikut:

a. *Preprocessing* citra: Data yang digunakan dalam sistem ini adalah citra kendaraan dan citra bukan kendaraan yang diperoleh dari INRIA *Car Dataset*, pencarian di *google* dan *flickr*, dan pengambilan manual dengan kamera digital. Tahap *pra-processing* citra yang dilakukan adalah *resizing* citra menjadi ukuran citra yang lebih kecil agar proses selanjutnya tidak membutuhkan waktu yang lama disebabkan dimensi citra yang besar. Tahap *preprocessing* selanjutnya adalah konversi citra RGB menjadi citra *grayscale*. Citra input berupa citra berwarna (RGB) akan diubah menjadi citra *grayscale.*
b. Ekstraksi fitur menggunakan *Histogram of Oriented Gradients* (HOG). Pada proses ini akan dilakukan ekstraksi dan karakterisasi komponen pada citra. Ekstraksi fitur dilakukan dengan menggunakan metode HOG yang akan menghasilkan vektor fitur dan disimpan di dalam database fitur.
c. Klasifikasi dengan *Real AdaBoost.* Pada tahap ini terdapat dua proses, yaitu proses *training* dan proses *testing*. Pada proses *training*, *weak learner* di-*training* pada distribusi sampel, dan dilakukan pembobotan ulang pada data *training*, jika

diklasifikasikan salah maka *Real AdaBoost* akan menambah bobotnya, dan sebaliknya. Pada tahap ini dihasilkan *weak classifier* lalu disimpan pada *database*. Kemudian proses *testing* yaitu proses pengambilan keputusan.

d. Pengambilan keputusan. Tahap ini merupakan tahap uji coba sistem. Pada tahap ini, dilakukan *signum* pada *weak classifier/weak hypothesis* yang didapatkan dari proses *training* yang menentukan apakah citra uji terdeteksi sebagai citra kendaraan atau tidak.



Gambar 3.1 Blok diagram tahapan penelitian

BAB IV
PERANCANGAN DAN IMPLEMENTASI

Pada bab ini menjelaskan mengenai perancangan sistem dan hasil implementasi semua proses yang telah dirancang sebelumnya. Pembahasan perancangan sistem diawali dengan penjelasan tentang lingkungan perancangan sistem, perancangan data, perancangan proses, serta perancangan antar muka.

4.1 Perancangan Sistem

Tampilan dari sistem deteksi kendaraan dibangun dengan tampilan yang sederhana. *Software* yang digunakan untuk membangun sistem deteksi kendaraan adalah Matlab 2013.

4.1.1 Perancangan Data

Data yang digunakan dalam sistem deteksi kendaraan ini dibagi menjadi 3 macam, yaitu data masukan yaitu data citra kendaraan dan bukan kendaraan, data proses yaitu data ketika tahap-tahap pemrosesan citra kendaraan sedang dilakukan, dan data keluaran berupa hasil deteksi sebagai kendaraan atau bukan kendaraan.

4.1.1.1 Data Masukan

Data masukan dalam sistem deteksi kendaraan ini adalah citra kendaraan meliputi sepeda motor, mobil, dan bus/truk dan citra bukan kendaraan seperti perkotaan, gedung, dan lain sebagainya dengan format .jpg dan berupa citra RGB.

Pada Gambar 4.1 ditampilkan 24 dari 532 sampel positif serta 24 dari 150 sampel negatif dari keseluruhan data *training* yang digunakan pada sistem deteksi kendaraan ini. Lalu pada Gambar 4.2 ditampilkan beberapa data citra *testing* yang digunakan dalam sistem pengujian. Data masukan nantinya akan dibedakan menjadi 2 yaitu data pengujian dan data pelatihan.

(a)

(b)

Gambar 4.1 Sebagian Sampel *Training* (a) Sampel Positif (b) Sampel Negatif

Gambar 4.2 Sebagian Citra *Testing*

4.1.1.2 Data Proses

Data proses adalah data yang digunakan selama proses berjalannya sistem yang merupakan hasil pengolahan dari data masukan untuk diproses kembali menjadi data keluaran di tahap selanjutnya. Data proses yang digunakan dalam proses ini ditunjukkan oleh Tabel 4.1.

Tabel 4.1 Data Proses

No.	Nama Data	Keterangan
1.	Citra *resize*	Citra yang di-*resize* menjadi 128 × 128.
2.	Citra *grayscale*	Citra keabuan berukuran 128 × 128.
3.	Fitur citra	Matriks fitur citra yang didapatkan dari proses ekstraksi fitur.
4.	*Weak classifier*	Output *weak classifier*.

4.1.1.3 Data Keluaran

Data keluaran yang dihasilkan dari sistem ini adalah hasil akhir dari proses deteksi yang menentukan apakah sebuah citra yang diuji terdeteksi sebagai kendaraan atau tidak. Data berupa informasi ini diperoleh dari proses ekstraksi fitur dengan metode HOG. Setelah mendapatkan fitur dengan HOG, dilakukan proses klasifikasi menggunakan metode *Real AdaBoost*, dimana proses ini terbagi menjadi dua proses yaitu proses *training* dataset fitur HOG yang diperoleh, dan proses *testing* data. Dari proses *training*

akan dihasilkan output *weak classifier*. Kemudian pada proses *testing* akan didapatkan hasil akhir berupa *strong classifier* yang didapat dari output *weak classifier*, lalu hasil output itu ditambahkan bersama dan diambil *signum* dari penjumlahan output tadi.

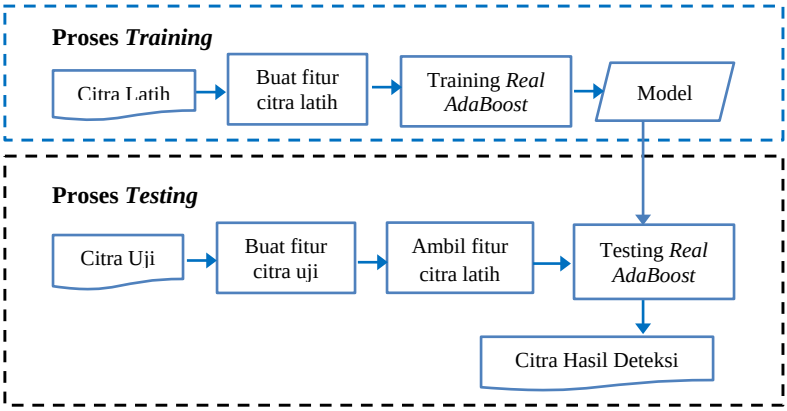
4.1.2 Gambaran Proses Secara Umum

Pada subbab ini akan diuraikan perancangan proses untuk membangun sebuah sistem deteksi kendaraan. Perancangan proses ini bertujuan untuk memperjelas hubungan antar proses beserta langkah-langkah yang dilakukan pada setiap proses. Secara garis besar proses yang dilakukan adalah:

1. Data input, berupa citra kendaraan dan citra bukan kendaraan yang disimpan dalam media penyimpanan. Selanjutnya data dimasukkan ke dalam sistem.
2. *Preprocessing* sebelum dilakukan proses ekstraksi fitur. Pada proses ini, citra input yang masuk ke dalam sistem akan diterapkan proses *resizing* ukuran dimensi citra menjadi 128 × 128 dan proses *grayscaling* dengan merubah citra input yang mulanya berupa citra RGB menjadi citra *grayscale.*
3. Proses ekstraksi fitur dengan metode HOG yaitu proses yang dilakukan terhadap citra input yang sebelumnya telah dilakukan *preprocessing* untuk memperoleh ciri citra kendaraan yang akan digunakan untuk proses berikutnya, yaitu proses klasifikasi.
4. Proses klasifikasi dengan *Real AdaBoost*, merupakan proses pengklasifikasian antara citra kendaraan dan citra bukan kendaraan. Ada 2 proses yang dilakukan pada proses ini, yaitu:
    a. Proses *training* (pelatihan), adalah proses untuk melatih sistem dengan data masukan yang telah diolah sebelumnya sehingga mampu mengenali apabila diberi masukan yang baru. Umumnya pada proses *training* sistem akan dihasilkan sebuah model, dalam sistem deteksi kendaraan ini diperoleh model berupa himpunan

output *weak classifier* yang akan digunakan untuk pengambilan keputusan pada proses *testing* nantinya.

b. Proses *testing* (pengujian), yaitu proses pengambilan keputusan terhadap citra uji yang dimasukkan. Dengan mendapatkan *strong classifier* yang didapat dari output *weak classifier*, lalu hasil output itu ditambahkan bersama dan diambil *signum* dari penjumlahan output tadi untuk mendapatkan sebuah keputusan apakah citra yang diuji terdeteksi sebagai kendaraan atau tidak.



Gambar 4.3 Diagram Alir Proses Secara Umum

Gambar 4.3 menunjukkan diagram alir proses deteksi kendaraan secara umum, dimana terdapat proses *training* dengan urutan proses citra latih masuk ke dalam sistem lalu dilakukan proses ekstraksi fitur untuk mendapatkan fitur citra latih kemudian dilakukan *training Real AdaBoost* untuk mendapatkan suatu model dan proses *testing* untuk menguji citra uji berdasarkan model yang didapatkan.

4.1.2.1 Data Input

Citra data input diambil sebagian dari INRIA *Car Dataset*, pencarian dari *Google* dan *Flickr*, dan pengambilan manual

dengan kamera digital. Ketentuan pengambilan gambar untuk data masukan adalah pencahayaan yang cukup.

Dari proses pengambilan citra ini diperoleh data *training* sebanyak 532 citra kendaraan dari bus, truk, mobil, dan sepeda motor dan 150 citra bukan kendaraan berupa jalan raya, gedung, tempat parkir dan *background* lainnya. Sedangkan untuk data *testing* diambil dari sisa *data training* yang tidak dipakai dan diambil citra kendaraan di jalan raya yang lebih bervariatif.

4.1.2.2 Proses *Preprocessing*

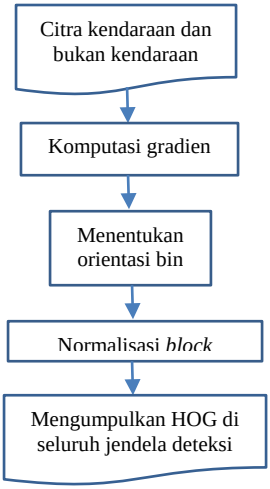
Data input yang masuk ke dalam sistem deteksi sebelum melangkah pada tahap ekstraksi fitur akan dilakukan proses *preprocessing.* Pada tahap *preprocessing* ini terdapat dua proses yaitu proses *resizing* ukuran citra supaya citra yang diproses nantinya memiliki dimensi yang lebih kecil sehingga proses ekstraksi fitur maupun proses klasifikasi berjalan lebih cepat, karena semakin kecil dimensinya maka semakin sedikit fitur yang dihasilkan. Setelah proses *resizing*, dilakukan proses *grayscaling* dengan merubah citra input RGB menjadi citra *grayscale*.

4.1.2.3 Proses Ekstraksi Fitur dengan *Histogram of Oriented Gradients* (HOG)

Ekstraksi fitur merupakan proses yang bertujuan untuk menentukan ciri dari suatu citra yang mampu membedakan antara citra kendaraan dan bukan kendaraan. Metode *Histogram of Oriented Gradients* (HOG) merupakan salah satu metode yang digunakan untuk ekstraksi fitur ke dalam kelas-kelas yang berbeda.

Pada proses ekstraksi fitur dengan HOG ini akan dihitung *magnitude* gradien dan orientasi gradien pada bidang lokal yang disebut *cell* dan dikonversi menjadi histogram. Gambar 4.4 menunjukkan diagram alir proses ekstraksi fitur dengan HOG. Untuk menghitung *descriptor* HOG, pertama dihitung gradien horizontal dan vertikal, selanjutnya didapatkan *magnitude* dan orientasi gradiennya. Citra dibagi ke dalam *cell-cell* berukuran 8

× 8. Kemudian nilai orientasi tiap pikselnya dikuantisasi ke dalam 9 *bin* (kanal) dalam sudut 180° menggunakan histogram. Dalam Tugas Akhir ini akan digunakan *unsigned gradient* dengan 9 bin setiap *cell* (1 bin = 20°). Kontribusi piksel terhadap tiap *bin* bergantung pada nilai *magnitude* gradiennya. Nilai-nilai dari seluruh *bin* dari tiap *cell* akan membentuk garis vektor dengan jumlah garis sama dengan jumlah *bin* yang dibentuk, yaitu 9 garis dengan sudut 20°antara garis satu dengan garis yang selanjutnya. Untuk memaksimalkan fitur HOG maka dibentuk sebuah *block* yang merupakan gabungan dari beberapa *cell* dengan ukuran 16 × 16. Teknik *overlapping block* dilakukan dengan membuat *block* yang bersebelahan saling *overlap* (tumpang tindih) dan menghitung kembali bagian yang *overlap*. Kemudian histogram yang terbentuk pada tiap *block* dinormalisasi, setelah itu digabungkan semua vektor pada semua *cell*.



Gambar 4.4 Diagram Alir Proses Ekstraksi Fitur dengan HOG

4.1.2.4 Proses Klasifikasi dengan *Real AdaBoost*

Tahapan setelah proses ekstraksi fitur dengan *Histogram of Oriented Gradients* (HOG) adalah proses klasifikasi dengan

metode *Real AdaBoost*. Proses klasifikasi ini dibagi menjadi dua proses. Proses pertama adalah proses *training* dari *dataset* berupa fitur yang didapatkan dari proses ekstraksi fitur HOG. Dan proses kedua adalah proses *testing* data dimana akan diklasifikasi sejumlah citra *testing* yang telah ditentukan apakah citra tersebut terdeteksi sebagai citra kendaraan atau bukan.

1. Proses *Training*

   Proses ini merupakan proses dilakukannya *training weak learner* dengan diberikan distribusi yang diboboti (*weighted distribution*), bobot sampel diinisialisasi sama. *Weak learner* yang digunakan dalam Tugas Akhir ini adalah *Classification and Regression Trees* (CART). CART adalah sebuah pohon keputusan dimana *leave* (daun) adalah *output* hasil klasifikasi dan *node* dalam membagi pohon untuk meminimumkan tingkat kesalahan. Proses *training* dengan CART ditunjukkan pada Gambar 4.5. Pada proses *training*, $\boldsymbol{X_i = x_{i,1}, x_{i,2}, \dots, x_{i,m}}$ adalah vektor fitur dari *instance* $\boldsymbol{x_i}$. Parameter yang dicari adalah $\boldsymbol{h}$ (output *weak classifier*). Sedangkan parameter yang diinputkan adalah *training* set $\boldsymbol{S = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)}$, $\boldsymbol{y_i}$ bernilai 1 jika $\boldsymbol{x_i}$ berupa citra kendaraan atau $\boldsymbol{y_i}$ bernilai -1 jika $\boldsymbol{x_i}$ berupa citra bukan kendaraan. Diagram alir proses *training* dapat dilihat pada Gambar 4.6.

   Untuk membangun CART, *information gain* pada setiap pohon akan dibandingkan. *Split* dengan *information gain* tertinggi akan diambil sebagai *split* pertama dan proses akan berlanjut sampai semua *node* anak menjadi *pure*, atau sampai *information gain* bernilai 0.

   Misal, diberikan tiga fitur hasil ekstraksi berukuran 1 × D.
   F_1 = [0.567 0.231 0.161 0.485 0.631] dengan label +1,
   F_2 = [0.938 0.157 0.848 0.665 0.734] dengan label -1,
   F_3 = [0.534 0.164 0.297 0.931 0.793] dengan label +1.
   Akan dicari *threshold* atau *information gain* untuk proses

klasifikasi. Pertama, dihitung *entropy* dengan rumus

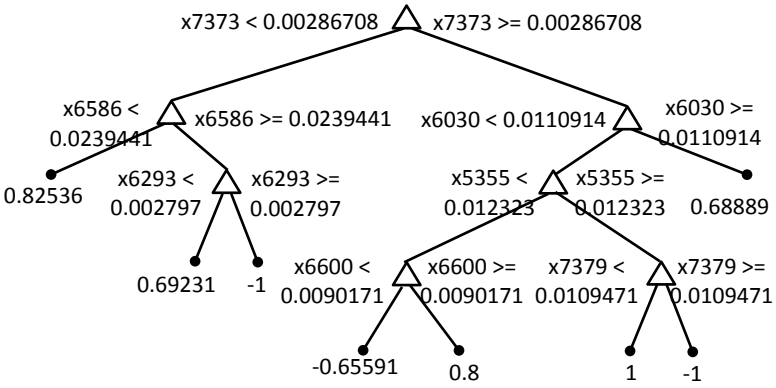
$$\begin{aligned}\textbf{Entropy}(\mathbf{F}) &= \sum_{i=1}^{j} -\boldsymbol{p_i log_2 p_i} \\ &= -\frac{\mathbf{2}}{\mathbf{3}}\boldsymbol{log_2}\frac{\mathbf{2}}{\mathbf{3}} - \frac{\mathbf{1}}{\mathbf{3}}\boldsymbol{log_2}\frac{\mathbf{1}}{\mathbf{3}} \\ &= \mathbf{0.389 + 0.528} \\ &= \mathbf{0.917}\end{aligned}$$

Kemudian dihitung *information gain* dengan rumus

$$\boldsymbol{Gain(F, A) = Entropy(F) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)}$$

dengan *values*(A) adalah himpunan semua nilai yang mungkin untuk atribut A, dan $\boldsymbol{S_v}$ adalah subset *S* untuk atribut A yang mempunyai nilai *v*. Sehingga, *information gain* $\boldsymbol{Gain(F, A) = 0.917 - \left(\frac{2}{3}\right)0.389 - \left(\frac{1}{3}\right)0.528 = 0.2117}$. Hasil *information gain* ini akan digunakan sebagai *threshold* untuk *split* pada *node* pertama, untuk *node* kedua dan seterusnya dihitung *information gain*nya lagi.

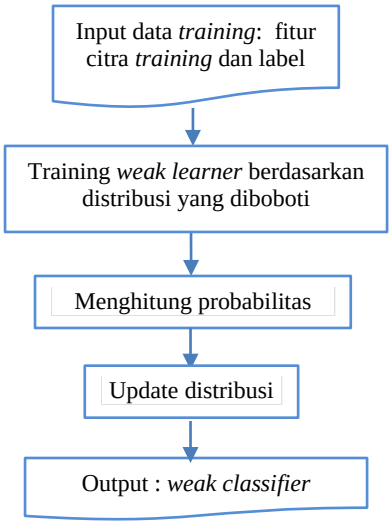
Setelah itu dihitung *output* dari *weak classifier, output* ini berupa bobot yang menunjukkan tingkat *error* ketika proses *training*, lebih kecil *error* dari *weak classifier*, maka lebih besar bobotnya pada *vote* terakhir. Selanjutnya bobot sampel di*update*, untuk sampel yang salah diklasifikasikan *AdaBoost* akan menambah bobotnya, dan sampel yang diklasifikasikan benar akan dikurangi bobotnya. Proses *training* ini diulang sampai *weak learner* telah dit*raining* semuanya.



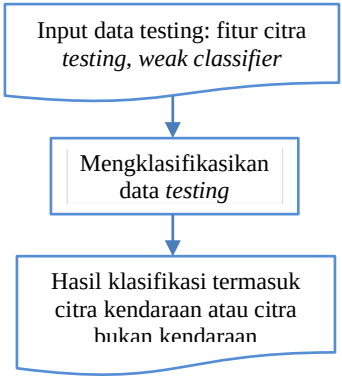
Gambar 4.5 Proses *Training* CART

2. Proses *Testing*

   Pada proses sebelumnya telah diperoleh nilai h. Proses selanjutnya adalah mendeteksi apakah suatu citra adalah citra kendaraan. Hasil output dari semua *weak classifier* ditambahkan bersama, dan diambil *signum* dari penjumlahan output untuk diberikan kepada *instance* x_i, dan *magnitude* $|h(x)|$ sebagai "*confidence*" (kepercayaan) pada prediksi ini dimana jika nilai akhir lebih besar dari 0 maka akan memberikan *output* +1 yang berarti citra terdeteksi sebagai kendaraan, dan sebaliknya jika nilai akhir lebih kecil dari 0 maka akan memberikan -1 yang berarti citra bukan merupakan kendaraan [15]. Diagram alir proses testing dapat dilihat pada Gambar 4.7.



Gambar 4.6 Diagram Alir Proses *Training Real AdaBoost*



Gambar 4.7 Diagram alir proses *testing Real AdaBoost*

4.2 Implementasi

Perancangan program yang telah dibangun selanjutnya diimplementasikan pada bahasa pemrograman dengan

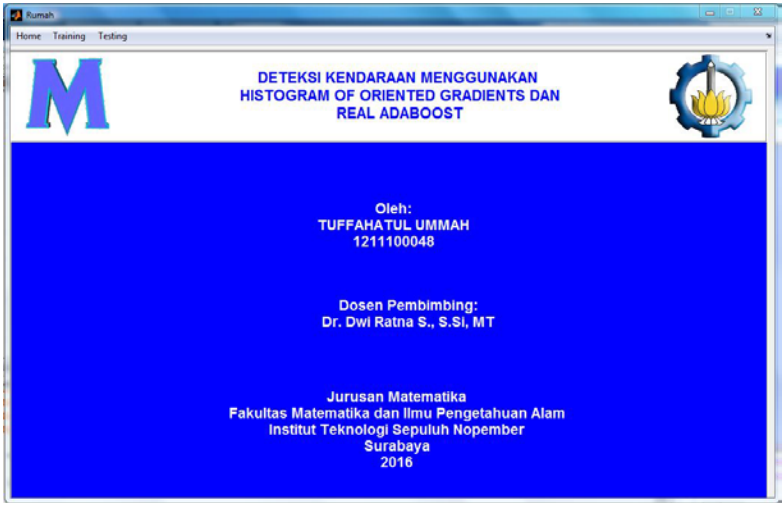
menggunakan Matlab. Pembahasan dalam implementasi sistem meliputi implementasi antarmuka (*interface*) sistem, implementasi tahap ekstraksi fitur dengan *Histogram of Oriented Gradients* (HOG), dan terakhir tahap klasifikasi dengan metode *Real AdaBoost*.

4.2.1 Implementasi Antarmuka

Pada Tugas Akhir ini, antarmuka sistem dibangun dengan menggunakan form dan kontrol yang terdapat pada Matlab. Adapun antarmuka-antarmuka yang diimplementasikan untuk menunjang penelitian Tugas Akhir ini adalah sebagai berikut.

4.2.1.1 Halaman Utama

Halaman utama merupakan antarmuka yang berisi menu-menu untuk menampilkan antarmuka-antarmuka lainnya dalam sistem. Hasil implementasi halaman utama dapat dilihat pada Gambar 4.7.



Gambar 4.8 Halaman Utama

Halaman utama dibuat dalam bentuk yang sederhana, terdiri dari 3 bagian utama, diantaranya :

1. Bagian *title bar*
2. Bagian *menu bar*
3. Bagian *main window*

Title bar merupakan bagian yang menunjukkan judul dari antarmuka yang sedang ditampilkan. Di bawah *title bar* terdapat *menu bar* yang berisi sederetan menu-menu yang digunakan oleh sistem. Adapun kegunaan menu-menu yang ditampilkan pada antarmuka utama sistem disajikan dalam Tabel 4.2.

Tabel 4.2 Kegunaan Menu Sistem

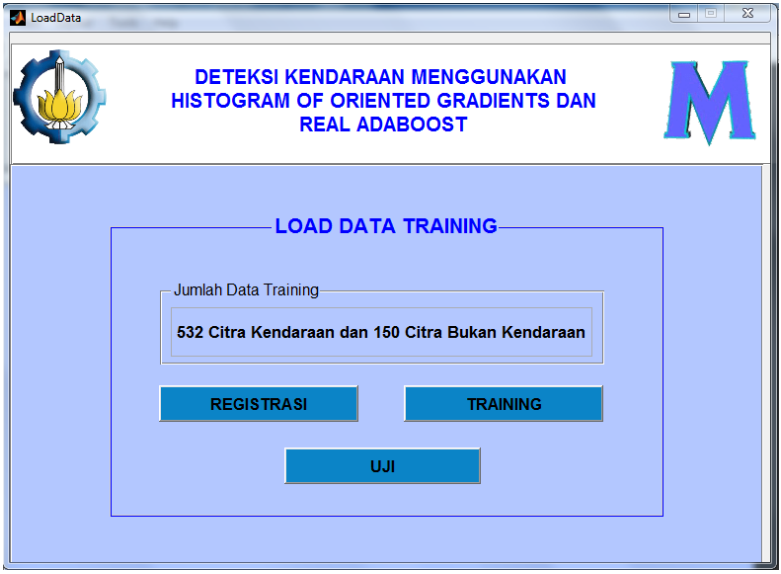
Menu	Kegunaan
Pelatihan Load Data	Melakukan pelatihan agar sistem dapat mengenali kendaraan dan bukan kendaraan dengan data masukan yang telah diproses.
Pengujian	Melakukan pengujian, pengamatan, & penilaian pada sistem, seberapa baik hasil yang didapatkan pada data masukan tertentu.

Main window merupakan bagian antarmuka yang digunakan untuk menampilkan berbagai antarmuka lain di dalam sistem.

4.2.1.2 Antarmuka Pelatihan

Antarmuka Pelatihan berguna untuk melakukan pelatihan terhadap citra kendaraan atau bukan kendaraan. Pada antarmuka pelatihan ini disediakan Load Data untuk melatih citra dalam satu folder sekaligus. Pada pelatihan ini dilakukan pengujian**,** pengamatan, dan penilaian pada sistem, seberapa baik hasil yang

didapatkan pada data masukan tertentu. Gambar 4.8 menunjukkan antarmuka pelatihan citra kendaraan.



Gambar 4.9 Antarmuka Pelatihan Citra Kendaraan

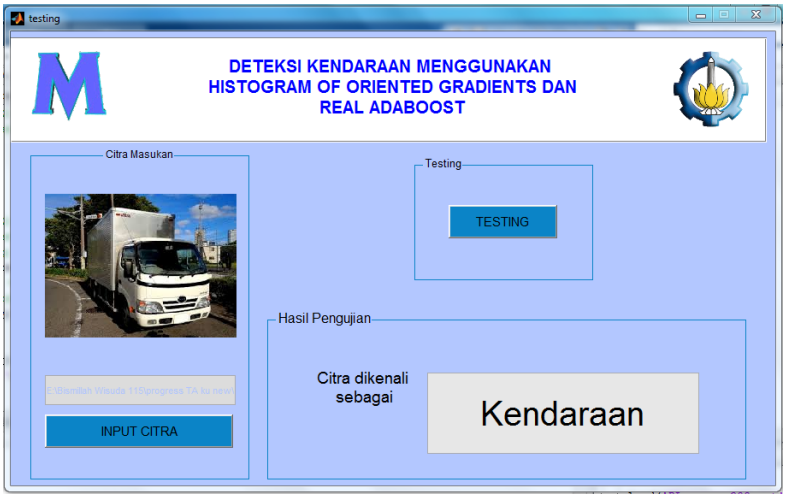
4.2.1.3 Antarmuka Pengujian

Antarmuka Pengujian digunakan untuk melakukan pengujian terhadap sistem berdasarkan algoritma *Real AdaBoost*. Hasil implementasi anatarmuka Pengujian dapat dilihat pada Gambar 4.9.

Gambar 4.10 Antarmuka Pengujian Citra Kendaraan

Pada antarmuka Pengujian terdapat tiga tombol. Fungsi dari tombol-tombol tersebut adalah sebagai berikut :

1. Tombol input citra digunakan untuk mengambil citra kendaraan atau bukan kendaraan yang tersimpan pada komputer.
2. Tombol ekstraksi fitur melakukan proses ekstraksi fitur dengan HOG terhadap citra yang akan diuji. Citra hasil pemrosesan tersebut akan ditampilkan pada *axes* yang telah dibuat pada halaman Pengujian.
3. Tombol Pengujian digunakan untuk melakukan proses ekstraksi fitur dan pengujian citra, sehingga didapatkan keputusan apakah citra yang diuji berupa citra kendaraan atau tidak. Hasil pengujian tersebut akan ditampilkan pada *textfield* Hasil Pengujian. Gambar 4.10 di bawah ini menunjukkan hasil pengujian sistem deteksi kendaraan.



Gambar 4.11 Hasil Pengujian Sistem Deteksi Kendaraan

4.2.2 Implementasi Tahap Input Data

Proses input data membutuhkan interaksi pengguna untuk mencari dan mengambil citra yang dibutuhkan di dalam media penyimpanan. Proses ini diimplementasikan menjadi sebuah program ke dalam fungsi yang telah tersedia di Matlab. Fungsi dijalankan adalah ketika tombol "Input Citra" dipilih.

4.2.3 Implemantasi Tahap *Preprocessing*

Pada tahap *preprocessing*, citra di*resize* menjadi ukuran 128 × 128, kemudian citra berwarna dirubah menjadi citra *grayscale*. Kode program implementasi tahap *preprocessing* ini diimplementasikan menjadi sebuah program berikut:

```
im = imresize(im,[128 128]);
im = rgb2gray(im);
```

4.2.4 Implementasi Tahap Ekstraksi Fitur

Proses selanjutnya adalah tahap ekstraksi fitur dengan *Histogram of Oriented Gradients*. Proses ini merupakan proses pembuatan fitur-fitur sebagai pengetahuan untuk sistem agar nantinya dapat mengklasifikasikan citra uji dengan baik. Proses ini diimplementasikan menjadi sebuah program ke dalam fungsi :

```
function [feature] = ekstraksi(im)
```

Pada proses ekstraksi fitur HOG ini, gradien horizontal dan vertikal citra dihitung untuk mencari *magnitude* dan orientasi gradiennya. Lalu citra dibagi menjadi *block* berukuran 16 × 16 piksel, dan pada saat yang sama, tiap *block* dibagi menjadi *cell* berukuran 8 × 8 piksel. Histogram dibentuk pada tiap piksel berdasarkan pada nilai *magnitude* yang berhubungan dengan orientasi *bin. Bin* berjarak merata dalam sudut $0° - 180°$, dan tiap *bin* berisi 9 garis sudut, dimana setiap garis sudut bernilai $20°$. Histogram yang terbentuk pada tiap *block* kemudian digabungkan dan dinormalisasi.

Output dari fungsi ini adalah vektor fitur citra. Kode program selengkapnya dapat dilihat pada lampiran A1.

Fitur citra berbentuk matriks 1 × D dengan D adalah dimensi ruang fitur. Vektor fitur kemudian di-*transpose* menjadi matriks D x 1 untuk data input di proses klasifikasi.

4.2.5 Implementasi Tahap Klasifikasi Citra

Setelah dilakukan proses pembuatan fitur, maka proses selanjutnya adalah proses klasifikasi. Klasifikasi dengan metode *Real AdaBoost* ini terdiri dari dua proses yaitu proses *training* dan proses *testing*. Pada tahap klasifikasi ini, penulis menggunakan GML Adaboost Matlab Toolbox [16].

4.2.5.1 Proses *Training*

Proses *training* merupakan proses untuk melatih *weak learner* berdasarkan distribusi sampel yang diboboti, dan akan didapatkan *weak classifier* dan beratnya.

Kode program implementasi tahap klasifikasi ini dapat dilihat pada Kode Sumber 4.1.

```
function training_Callback(hObject, eventdata,
handles)
load databasetraining128
TrainData = C128;
TrainLabels = D128;
weak_learner = tree_node_w(3);
[RLearners16 RWeights16] =
RealAdaBoost(weak_learner, TrainData,
TrainLabels, 200);
```

Kode Sumber 4.2 Fungsi Training

4.2.5.2 Proses *Testing*

Proses *testing* bertujuan untuk mengklasifikasi apakah suatu citra adalah citra kendaraan atau tidak. Jika suatu citra yang di *testing* menghasilkan *output* +1 maka citra tersebut terdeteksi sebagai citra kendaraan. Sedangkan jika citra tersebut menghasilkan *output* -1 maka citra tersebut terdeteksi sebagai citra bukan kendaraan.

Proses pengujian diimplementasikan menjadi sebuah program berikut :

```
ResultR = sign(Classify(RLearners16,
RWeights16, ControlData));
if ResultR == 1
    set(handles.hasiluji, 'String',
'Kendaraan');
    fprintf('Kendaraan');
    fprintf('\n');
else
    set(handles.hasiluji, 'String', 'Bukan
Kendaraan');
    fprintf('Bukan Kendaraan');
    fprintf('\n');
end
```

Kode program selengkapnya dapat dilihat pada lampiran A1.

“*Halaman ini sengaja dikosongkan*”

BAB V
PENGUJIAN DAN PEMBAHASAN HASIL

Bab ini menjelaskan mengenai proses pengujian yang dilakukan terhadap sistem deteksi kendaraan dengan menggunakan metode *Histogram of Oriented Gradients* dan *Real AdaBoost*. Hasil pengujian kemudian dibahas untuk mengetahui unjuk kerja sistem secara keseluruhan dalam menjalankan fungsi yang diinginkan. Selanjutnya dijelaskan mengenai hasil pengujian terhadap sistem yang telah diimplementasikan pada bab sebelumnya, yaitu pengujian tahap *preprocessing*, pengujian tahap ekstraksi fitur dengan *Histogram of Oriented Gradients* (HOG) dan pengujian tahap klasifikasi dengan metode *Real AdaBoost* dengan beberapa variasi pengujian untuk mengetahui keakuratan sistem. Bab ini diakhiri dengan pembahasan hasil pengujian yang telah dilakukan.

5.1 Lingkungan Pengujian Sistem

Lingkungan pengujian dari sistem deteksi kendaraan meliputi perangkat keras dan lunak komputer. Detail dari perangkat keras dan lunak yang digunakan dapat dilihat pada Tabel 5.1.

Tabel 5.1 Lingkungan Pengujian Sistem

Perangkat Keras	Prosesor	: Intel® Core(TM) i3-2310M CPU @ 2.10GHz 2.10GHz
	Memory	: 2.00 GB
Perangkat Lunak	Sistem Operasi	: Windows 7 Ultimate 32-bit
	Tools	: MATLAB R2013

5.2 Pengujian Tahap *Preprocessing*

Pengujian ini dilakukan terhadap proses-proses pada tahap pengolahan citra. Pengujian bertujuan untuk mengetahui bahwa proses-proses pada tahap pengolahan citra sudah benar. Pada tahap ini terdapat 2 proses yaitu *resizing* dan *grayscaling*.

1. Pengujian proses *resizing*
   Pengujian proses ini bertujuan untuk memperkecil dimensi citra menjadi 128 × 128. Gambar 5.2 adalah citra hasil *resizing.*
2. Pengujian proses *grayscaling*
   Pengujian proses ini bertujuan untuk memudahkan penghitungan gradien citra. Gambar 5.3 adalah citra hasil *grayscaling.*

(a) (b)

Gambar 5.1 Citra input (a) citra kendaraan, (b) citra bukan kendaraan

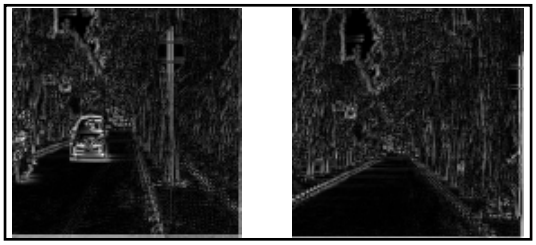
Gambar 5.2 Pengujian proses *resizing* untuk citra kendaraan dan citra bukan kendaraan

Gambar 5.3 Pengujian proses *grayscaling* untuk citra kendaraan dan citra bukan kendaraan

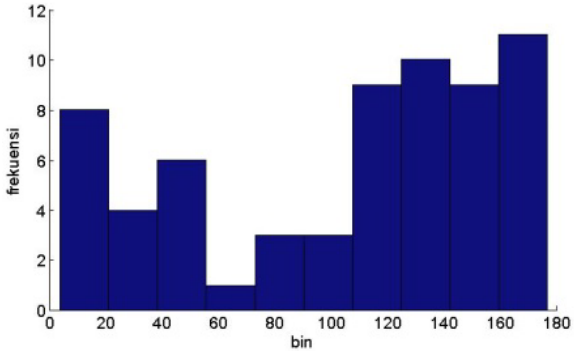
5.3 Pengujian Tahap Ekstraksi Fitur

Pengujian proses ini menggunakan data citra yang telah diproses pada pra-pemrosesan. Tahap ekstraksi fitur ini menggunakan metode *Histogram of Oriented Gradients*. Pada tahap ini terdapat 3 proses yaitu:

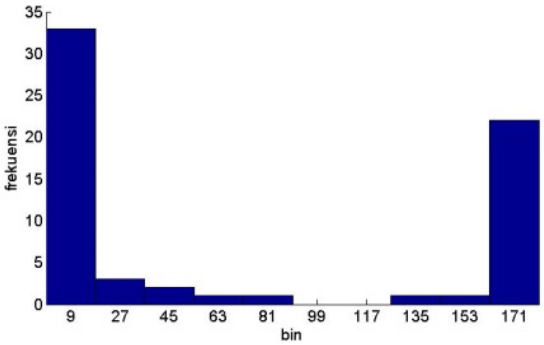
1. Pengujian proses komputasi gradien
   Pengujian proses ini bertujuan untuk mendapatkan informasi *contour* citra dan mengurangi perubahan cahaya.
2. Pengujian proses orientasi *bin.*
   Pengujian proses ini bertujuan untuk mendapatkan orientasi *bin* berdasarkan komputasi gradien tiap piksel citra.
3. Pengujian proses visualisasi fitur HOG.
   Pengujian proses ini bertujuan untuk menampilkan fitur HOG secara keseluruhan.



Gambar 5.4 Pengujian proses komputasi gradien untuk citra kendaraan dan citra bukan kendaraan

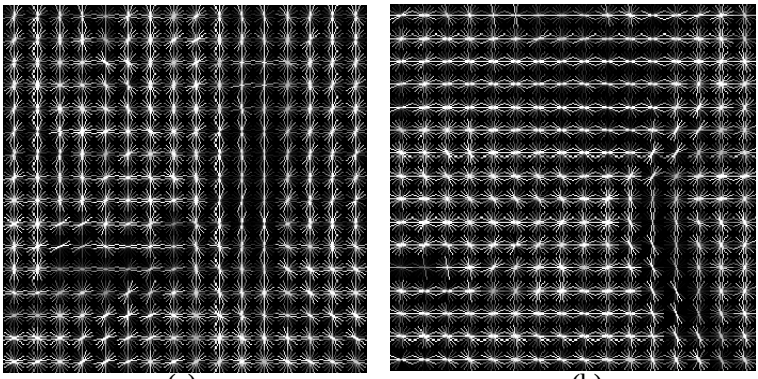


(a)



(b)

Gambar 5.5 Pengujian proses orientasi bin (a) citra kendaraan, (b) citra bukan kendaraan



(a) (b)

Gambar 5.6 Pengujian proses visualisasi fitur HOG (a) citra kendaraan, (b) citra bukan kendaraan

Gambar 5.5 (a) dan (b) menunjukkan hasil orientasi *bin* pada citra kendaraan dan citra bukan kendaraan di dalam sebuah *cell* yang berukuran 8 × 8. Orientasi *bin* didapat dari nilai orientasi gradien dari tiap piksel. Pada gambar ditunjukkan bahwa orientasi *bin* berjarak merata dari $0° - 180°$, dan pada setiap rentang *bin*, histogram akan diberikan sesuai dengan gradien mereka. Dapat dilihat bahwa citra kendaraan memiliki orientasi bin yang lebih merata daripada citra bukan kendaraan dikarenakan adanya objek kendaraan pada citra yang menyebabkan hasil orientasi *bin* memiliki perbedaan yang jauh.

Gambar 5.6 menunjukkan visualisasi fitur HOG pada citra kendaraan dan citra bukan kendaraan. Setelah didapatkan orientasi *bin* pada setiap piksel di dalam *cell*, *cell-cell* tersebut lalu digabung untuk membentuk sebuah *block* dan semua histogram dikombinasikan dan dinormalisasi untuk mendapatkan visualisasi fitur HOG.

5.4 Pengujian Tahap Klasifikasi dengan *Real AdaBoost*

5.4.1 Proses *Training*

Data pada sistem deteksi kendaraan ini terdiri dari 974 citra. Data dibagi menjadi data pelatihan dan data uji. Data

pelatihan terdiri dari 532 citra kendaraan dan 150 citra bukan kendaraan. Sedangkan data uji merupakan sisa dari jumlah keseluruhan data yang tidak digunakan untuk data *training*.

Pengujian proses ini menggunakan fitur yang telah diperoleh dari proses ekstraksi fitur dengan *Histogram of Oriented Gradients* dan bertujuan untuk mengetahui data yang diperlukan dalam proses berikutnya.

Data keluaran yang dihasilkan akan disimpan dalam file matlab dengan format .mat. Data-data yang disimpan akan berbentuk informasi seperti yang terlihat pada Tabel 5.2.

Tabel 5.2 Hasil Proses *Training Real AdaBoost*

Citra Input	***Weak Classifier***	**Output**
Kendaraan	*CART tree*	16.5323
Bukan Kendaraan	*CART tree*	-2.2593

Tabel 5.2 merupakan hasil proses *training Real AdaBoost* sebagai pengetahuan sistem untuk proses berikutnya. *Weak classifier* keseluruhan yang didapatkan dari proses *training* ini adalah *cell array* dari *weak classifiers* yang berjumlah 800 dan output mereka. Nilai output pada Tabel 5.2 menunjukkan salah satu nilai output dari salah satu *weak classifier*. Dapat dilihat bahwa nilai output untuk citra kendaraan memiliki nilai lebih besar dari 0, dan sebaliknya nilai output untuk citra bukan kendaraan memiliki nilai lebih kecil dari 0. Informasi inilah yang nantinya digunakan dalam proses *testing*.

5.4.2 Proses *Testing*

Pengujian sistem menggunakan 257 citra kendaraan dan 35 citra bukan kendaraan. Pengujian proses ini untuk menentukan apakah suatu citra terdeteksi sebagai citra kendaraan atau bukan. Hasil dari pengujian sistem dapat dilihat pada Tabel 5.3.

Tabel 5.3 Hasil Proses *Testing*

Citra Input / Hasil Deteksi	Citra Kendaraan		Citra Bukan Kendaraan
	Ramai	Sepi	
Kendaraan	135	100	2
Bukan Kendaraan	10	12	33

Pada penelitian ini dilakukan pengujian pada kasus ketika jalan ramai dan sepi. Jalan ramai artinya citra dengan kendaraan lebih dari satu, dan jalan sepi artinya citra dengan kendaraan hanya ada satu. Berdasarkan Tabel 5.3, citra kendaraan pada saat jalan ramai yang terdeteksi benar sebagai kendaraan sebanyak 135 citra, sedangkan citra kendaraan pada saat jalan sepi yang terdeteksi benar sebagai kendaraan sebanyak 100 citra.

Sebaliknya, citra bukan kendaraan yang terdeteksi benar sebagai bukan kendaraan sebanyak 33 citra, dan terdeteksi salah sebagai kendaraan sebanyak 2 citra.

Tabel 5.4 Hasil Proses *Testing* dengan Ditambahi *Noise*

Citra Input / Hasil Deteksi	Citra Kendaraan	Citra Bukan Kendaraan
Kendaraan	10	1
Bukan Kendaraan	2	5

Untuk mengetahui seberapa kuat sistem deteksi kendaraan yang telah dibangun, maka dicoba menguji citra uji yang ditambahi *noise*. Dalam tugas akhir ini, *noise* yang digunakan adalah *Gaussian noise* dengan nilai mean = 0,02 dan nilai variance = 0,03. Contoh citra *testing* yang ditambahi *noise* ditunjukkan pada Gambar 5.7. Proses pengujian ini hanya

mengambil beberapa sampel dari citra uji, yaitu 18 citra. Pada Tabel 5.4 ditunjukkan hasil proses *testing* dengan ditambahi *noise*, citra bukan kendaraan yang terdeteksi salah sebagai citra kendaraan sebanyak satu citra, dan citra kendaraan kendaraan yang terdeteksi salah sebagai citra bukan kendaraan sebanyak dua citra.

Tabel 5.5 Hasil Proses *Testing* dengan Objek Kotak dan Orang yang Ditambahkan pada Citra Bukan Kendaraan

Citra Input / **Hasil Deteksi**	**Citra Bukan Kendaraan dengan Objek Kotak dan Orang**
Kendaraan	4
Bukan Kendaraan	6

Selain ditambahkan *noise*, data *testing* citra bukan kendaraan ditambahkan dengan gambar objek kotak yang mirip dengan kendaraan dan gambar orang, yang ditunjukkan pada Gambar 5.8. Hasil proses *testing* dengan objek kotak dan orang dapat dilihat pada Tabel 5.5. Data *testing* yang digunakan terdapat 10 citra bukan kendaraan, dan ditambahkan gambar objek kotak seperti meja kantor atau *box* dan gambar orang. Dari 10 citra yang diuji, empat citra terdeteksi salah sebagai kendaraan, dan enam citra terdeteksi benar sebagai bukan kendaraan.

Gambar 5.7 Contoh Citra *Testing* yang Ditambahi *Noise*

Gambar 5.8 Contoh Citra *Testing* yang Ditambahi Kotak dan Orang

5.5 Pembahasan Hasil Pengujian

Pembahasan hasil pengujian difokuskan pada hasil pengujian dan proses pendeteksian citra kendaraan. Pada tugas akhir ini pembahasan hasil pengujian ini digunakan untuk mengetahui kinerja sistem deteksi citra kendaraan menggunakan metode *Histogram of Oriented Gradients* dan *Real AdaBoost*. Dari hasil yang didapatkan pada subbab sebelumnya, sistem mampu mendeteksi kendaraan dengan cukup baik. Pada subbab ini akan diberikan cara menghitung tingkat akurasi sistem dengan menggunakan Persamaan 5.1.

$$\text{Akurasi} = \frac{\text{KB+BKB}}{\text{jumlah citra uji}} \times 100\% \qquad (5.1)$$

dengan KB adalah jumlah citra kendaraan benar dan BKB adalah jumlah citra bukan kendaraan benar.

Penghitungan akurasi dilakukan pada data *training* dan data *testing*. Data *training* yang diuji terdiri dari 532 citra kendaraan dan 150 citra bukan kendaraan. Dari data *training* yang telah diuji, hasil deteksi menunjukkan bahwa semua citra baik citra kendaraan maupun citra bukan kendaraan dideteksi benar sebagai kendaraan maupun bukan kendaraan, atau dengan kata lain, data *training* yang diuji memiliki akurasi sebesar 100 %. Selanjutnya, data *testing* merupakan sisa dari jumlah keseluruhan data yang tidak digunakan untuk data *training*, yaitu 257 citra kendaraan dan 35 citra bukan kendaraan. Berdasarkan hasil pengujian pada Tabel 5.3 didapatkan jumlah citra kendaraan benar yaitu 235 citra dan jumlah citra bukan kendaraan benar yaitu 33 citra. Sehingga penghitungan akurasinya adalah,

$$\text{Akurasi} = \frac{235 + 33}{292} \times 100\% = 91.78\ \%$$

Tabel 5.7 Hasil Perhitungan Akurasi

Citra Input	**Akurasi (%)**
Data *Training*	100 %
Data *Testing*	91,78 %

5.6 Pembahasan Penyebab Besar Kecilnya Akurasi

Penyebab utama rendahnya akurasi sistem deteksi citra kendaraan dengan *Histogram of Oriented Gradients* dan *Real AdaBoost* adalah kurang banyaknya data *training* citra bukan kendaraan yang dilatih, perbandingan jumlah data *training* antara citra kendaraan dan citra bukan kendaraan yang terlampau jauh sehingga mengurangi tingkat akurasi dari sistem deteksi.

“*Halaman ini sengaja dikosongkan*”

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan eksperimen dan pembahasan terhadap hasil pengujian yang telah dilakukan terhadap sistem deteksi kendaraan menggunakan metode *Histogram of Oriented Gradients* dan *Real AdaBoost*, maka dapat diambil beberapa kesimpulan sebagai berikut:

1. Tugas Akhir ini telah berhasil melakukan deteksi kendaraan dengan menggunakan metode *Histogram of Oriented Gradients* dan *Real AdaBoost*, dengan tahapan prosesnya yaitu pra-pemrosesan, ekstraksi fitur dengan HOG, dan proses klasifikasi dengan *Real AdaBoost*.
2. Metode *Histogram of Oriented Gradients* dan *Real AdaBoost* pada Tugas Akhir ini dapat digunakan untuk mendeteksi kendaraan dengan tingkat akurasi sebesar 91.78 %. Pengujian dilakukan pada 257 citra kendaraan dan 35 citra bukan kendaraan.
3. Hasil pengujian untuk data *testing* yang ditambahi *noise* memiliki tingkat akurasi yang cukup baik, yaitu 83.33 %.
4. Citra dengan objek kotak mirip kendaraan memiliki tingkat akurasi yang rendah karena kurang banyaknya jumlah dan keberagaman data *training* citra bukan kendaraan yang dilatih.

6.2 Saran

Dengan melihat hasil yang dicapai pada penelitian ini, penulis menyarankan untuk menambah keberagaman data *training* untuk citra bukan kendaraan. Dan untuk pengembangan selanjutnya tidak hanya sebatas deteksi kendaraan pada citra statis tapi dilanjutkan untuk deteksi kendaraan pada citra bergerak (video).

"*Halaman ini sengaja dikosongkan*"

DAFTAR PUSTAKA

[1] Madhogaria, Satish dkk. "Car Detection by Fusion of HOG and Causal MRF". **IEEE Transactions on Aerospace and Electronic Systems**, 51(1): 575-590, 2015.

[2] Aoki, Daisuke dan Junzo Watada. (2015). "Human Tracking Method Based on Improved HOG+Real AdaBoost". **2015 10th AsianControl Conference (ASCC)**.

[3] Dalal, Navneet dan Bill Triggs. (2005). "Histograms of Oriented Gradients for Human Detections". **Proc. IEEE Conf.on Computer Vision & Pattern Recognition**. 886-893.

[4] Wu, Shuqiong dan Hiroshi Nagahashi. (2015). "Analysis of Generalization Ability for Different AdaBoost Variants Based on Classification and Regression Trees". **Hindawi Publishing Corporation, Journal of Electrical and Computer Engineering**.

[5] Szeliski, Richard. "Object Detection". **Computer Vision: Algorithms and Application**. Springer. 658-666, 2010.

[6] Sancho, C. R. (2014). "Pedestrian Detection Using A Boosted Cascade of Histogram of Oriented Gradients". **Information Coding Group Department of Electrical Engineering**.

[7] Nixon, Mark S. dan Alberto S. Aguado, "Feature Extraction and Image Processing". Elsevier Ltd., 2008.

[8] Lee, Yeunghak dkk. "Spatial Regions Periodicity based Detection of Two-wheelers Using Histogram of Oriented Gradients". **International Journal of Multimedia and Ubiquitous Engineering**. 10(4): 325-336, 2015.

[9] Chen, Pei-Yin dkk. (2014). "An Efficient Hardware Implementation of HOG Feature Extraction for Human Detection". **IEEE Transactions On Intelligent Transportation Systems**. 15(2), 656-662.

[10] Wu, Bo dkk. (2004). "Fast Rotation Invariant Multi-View Face Detection Based on Real Adaboost". **Proceedings of**

the Sixth IEEE International Conference on Automatic Face and Gesture Recognition (FGR'04).

[11] Balázs, Kégl. (2009). "Introduction to AdaBoost". Tersedia: http://users.lal.in2p3.fr/kegl/teaching/stages/notes/tutorial. [8 April 2016]

[12] Sirkunan, Jeevan. (2012). "Histogram Of Oriented Gradients Architecture : Gradient Computation". **Thesis Faculty of Electrical Engineering Universiti Teknologi Malaysia**.

[13] Scahapire, R.E. dan Yoram Singer. (1999). "Improved Boosting Algorithms Using Confidence-rated Predictions".

[14] Lih-Heng, Chan dan Salleh Sh-Hussain. (2008). "Solving Two-Class Classification Problem Using Adaboost". **Center Biomedical Engineering Universiti Teknologi Malaysia**.

[15] Harrington, Peter. (2012). "Machine Learning in Action". Manning Publications.

[16]http://graphics.cs.msu.ru/en/science/research/machinelearning/adaboosttoolbox

LAMPIRAN A

A1. Kode Fungsi Untuk Ekstraksi Fitur dengan HOG

```
function [feature] = ekstraksi(im)

citra=im;
im = rgb2gray(im);

im=double(im);
rows=size(im,1);
cols=size(im,2);
Ix=im; %Basic Matrix assignment
Iy=im; %Basic Matrix assignment

%proses komputasi gradien
for i=1:rows-2
    Iy(i,:)=(im(i,:)-im(i+2,:));
end
for i=1:cols-2
    Ix(:,i)=(im(:,i)-im(:,i+2));
end

angle=atand(Ix./Iy);
angle=imadd(angle,90);
magnitude=sqrt(Ix.^2 + Iy.^2);

% Remove redundant pixels in an image.
angle(isnan(angle))=0;
magnitude(isnan(magnitude))=0;

feature=[]; %initialized the feature vector
%Iterations for Blocks
for i = 0: rows/8 - 2
    for j= 0: cols/8 - 2

        mag_patch = magnitude(8*i+1 : 8*i+16 ,
8*j+1 : 8*j+16);
        ang_patch = angle(8*i+1 : 8*i+16 , 8*j+1
: 8*j+16);
```

LAMPIRAN A (LANJUTAN)

```
block_feature=[];

        %Iterations for cells in a block
        for x= 0:1
            for y= 0:1
                angleA =ang_patch(8*x+1:8*x+8,
8*y+1:8*y+8);
                magA   =mag_patch(8*x+1:8*x+8,
8*y+1:8*y+8);
                histr  =zeros(1,9);

                %Iterations for pixels in one
cell
                for p=1:8
                    for q=1:8

                        alpha= angleA(p,q);

                        % Binning Process (Bi-
Linear Interpolation)
                         if alpha>10 &&
alpha<=30
                            histr(1)=histr(1)+
magA(p,q)*(30-alpha)/20;
                            histr(2)=histr(2)+
magA(p,q)*(alpha-10)/20;
                        elseif alpha>30 &&
alpha<=50
                            histr(2)=histr(2)+
magA(p,q)*(50-alpha)/20;
                            histr(3)=histr(3)+
magA(p,q)*(alpha-30)/20;
                        elseif alpha>50 &&
alpha<=70
                            histr(3)=histr(3)+
magA(p,q)*(70-alpha)/20;
```

LAMPIRAN A (LANJUTAN)

```
                    histr(4)=histr(4)+
magA(p,q)*(alpha-50)/20;
                elseif alpha>70 &&
alpha<=90
                    histr(4)=histr(4)+
magA(p,q)*(90-alpha)/20;
                    histr(5)=histr(5)+
magA(p,q)*(alpha-70)/20;
                elseif alpha>90 &&
alpha<=110
                    histr(5)=histr(5)+
magA(p,q)*(110-alpha)/20;
                    histr(6)=histr(6)+
magA(p,q)*(alpha-90)/20;
                elseif alpha>110 &&
alpha<=130
                    histr(6)=histr(6)+
magA(p,q)*(130-alpha)/20;
                    histr(7)=histr(7)+
magA(p,q)*(alpha-110)/20;
                elseif alpha>130 &&
alpha<=150
                    histr(7)=histr(7)+
magA(p,q)*(150-alpha)/20;
                    histr(8)=histr(8)+
magA(p,q)*(alpha-130)/20;
                elseif alpha>150 &&
alpha<=170
                    histr(8)=histr(8)+
magA(p,q)*(170-alpha)/20;
                    histr(9)=histr(9)+
magA(p,q)*(alpha-150)/20;
                elseif alpha>=0 &&
alpha<=10
                    histr(1)=histr(1)+
magA(p,q)*(alpha+10)/20;
                    histr(9)=histr(9)+
magA(p,q)*(10-alpha)/20;
```

LAMPIRAN A (LANJUTAN)

```
                        elseif alpha>170 &&
alpha<=180
                            histr(9)=histr(9)+
magA(p,q)*(190-alpha)/20;
                            histr(1)=histr(1)+
magA(p,q)*(alpha-170)/20;
                        end
                    end
                end

                block_feature=[block_feature
histr]; % Concatenation of 9 histograms to form
one block feature
            end
        end
        % Normalize the values in the block
using L2-Norm

block_feature=block_feature/sqrt(norm(block_feat
ure)^2+.01);

        feature=[feature
block_feature];%Features concatenation
    end
end
feature(isnan(feature))=0; %Removing Infinitiy
values

% Normalization of the feature vector using L2-
Norm
feature=feature/sqrt(norm(feature)^2+1);

for z=1:length(feature)
    if feature(z)>0.2
         feature(z)=0.2;
    end
end
```

LAMPIRAN A (LANJUTAN)

A2. Kode Program untuk Proses *Testing* dengan *Real AdaBoost*

```
function testing_Callback(hObject, eventdata,
handles)
% PROSES GRAYSCALE CITRA
tic
im = handles.data1;
im=imresize(im,[128 128]);
im = rgb2gray(im);
im=double(im);
rows=size(im,1);
cols=size(im,2);
Ix=im; %Basic Matrix assignment
Iy=im; %Basic Matrix assignment

%proses komputasi gradien
for i=1:rows-2
    Iy(i,:)=(im(i,:)-im(i+2,:));
end
for i=1:cols-2
    Ix(:,i)=(im(:,i)-im(:,i+2));
end

angle=atand(Ix./Iy);
angle=imadd(angle,90);
magnitude=sqrt(Ix.^2 + Iy.^2);

% Remove redundant pixels in an image.
angle(isnan(angle))=0;
magnitude(isnan(magnitude))=0;

feature=[]; %initialized the feature vector
%Iterations for Blocks
for i = 0: rows/8 - 2
    for j= 0: cols/8 - 2
```

LAMPIRAN A (LANJUTAN)

```
        mag_patch = magnitude(8*i+1 : 8*i+16 ,
8*j+1 : 8*j+16);
        %mag_patch = imfilter(mag_patch,gauss);
        ang_patch = angle(8*i+1 : 8*i+16 , 8*j+1
: 8*j+16);

block_feature=[];

        %Iterations for cells in a block
        for x= 0:1
            for y= 0:1
                angleA =ang_patch(8*x+1:8*x+8,
8*y+1:8*y+8);
                magA   =mag_patch(8*x+1:8*x+8,
8*y+1:8*y+8);
                histr  =zeros(1,9);

                %Iterations for pixels in one
cell
                for p=1:8
                    for q=1:8

                        alpha= angleA(p,q);

                        % Binning Process (Bi-
Linear Interpolation)
                         if alpha>10 &&
alpha<=30
                            histr(1)=histr(1)+
magA(p,q)*(30-alpha)/20;
                            histr(2)=histr(2)+
magA(p,q)*(alpha-10)/20;
                        elseif alpha>30 &&
alpha<=50
                            histr(2)=histr(2)+
magA(p,q)*(50-alpha)/20;
```

LAMPIRAN A (LANJUTAN)

```
                    histr(3)=histr(3)+
magA(p,q)*(alpha-30)/20;
                elseif alpha>50 &&
alpha<=70
                    histr(3)=histr(3)+
magA(p,q)*(70-alpha)/20;
                    histr(4)=histr(4)+
magA(p,q)*(alpha-50)/20;
                elseif alpha>70 &&
alpha<=90
                    histr(4)=histr(4)+
magA(p,q)*(90-alpha)/20;
                    histr(5)=histr(5)+
magA(p,q)*(alpha-70)/20;
                elseif alpha>90 &&
alpha<=110
                    histr(5)=histr(5)+
magA(p,q)*(110-alpha)/20;
                    histr(6)=histr(6)+
magA(p,q)*(alpha-90)/20;
                elseif alpha>110 &&
alpha<=130
                    histr(6)=histr(6)+
magA(p,q)*(130-alpha)/20;
                    histr(7)=histr(7)+
magA(p,q)*(alpha-110)/20;
                elseif alpha>130 &&
alpha<=150
                    histr(7)=histr(7)+
magA(p,q)*(150-alpha)/20;
                    histr(8)=histr(8)+
magA(p,q)*(alpha-130)/20;
                elseif alpha>150 &&
alpha<=170
                    histr(8)=histr(8)+
magA(p,q)*(170-alpha)/20;
                    histr(9)=histr(9)+
magA(p,q)*(alpha-150)/20;
```

LAMPIRAN A (LANJUTAN)

```
                        elseif alpha>=0 &&
alpha<=10
                            histr(1)=histr(1)+
magA(p,q)*(alpha+10)/20;
                            histr(9)=histr(9)+
magA(p,q)*(10-alpha)/20;
                        elseif alpha>170 &&
alpha<=180
                            histr(9)=histr(9)+
magA(p,q)*(190-alpha)/20;
                            histr(1)=histr(1)+
magA(p,q)*(alpha-170)/20;
                        end
                    end
                end

                block_feature=[block_feature
histr]; % Concatenation of 9 histograms to form
one block feature

            end
        end
        % Normalize the values in the block
using L2-Norm

block_feature=block_feature/sqrt(norm(block_feat
ure)^2+.01);

        feature=[feature
block_feature];%Features concatenation

    end
end
feature(isnan(feature))=0; %Removing Infinitiy
values
```

LAMPIRAN A (LANJUTAN)

```
% Normalization of the feature vector using L2-
Norm
feature=feature/sqrt(norm(feature)^2+1);

for z=1:length(feature)
    if feature(z)>0.2
         feature(z)=0.2;
    end
end
t1=toc
f=feature;
F=f';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% PROSES TESTING REAL ADABOOST %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

MaxIter = 200; % boosting iterations
tic
ControlData   = F;

load RLearners16
load RWeights16

% constructing weak learner
weak_learner = tree_node_w(3); % pass the number
of tree splitsy to the constructor

% evaluating on control set
ResultR = sign(Classify(RLearners16, RWeights16,
ControlData));

% obtaining real valued results
Confidence = Classify(RLearners8, RWeights8,
ControlData);
Tetta = 0.2; % TP/FP regulating threshold;
Y = sign(Confidence - Tetta);
```

LAMPIRAN A (LANJUTAN)

```
t2=toc
if ResultR == 1
    set(handles.hasiluji, 'String',
'Kendaraan');
    fprintf('Kendaraan');
    fprintf('\n');
else
    set(handles.hasiluji, 'String', 'Bukan
Kendaraan');
    fprintf('Bukan Kendaraan');
    fprintf('\n');
end
```

BIODATA PENULIS

Penulis bernama lengkap Tuffahatul Ummah, yang biasa dipanggil Tutut. Penulis dilahirkan di Gresik pada tanggal 16 Januari 1993. Penulis lulus dari MI Maskumambang, MTs Maskumambang, dan MA Maskumambang dan melanjutkan pendidikan di Matematika ITS pada tahun 2011. Semenjak SMA penulis suka dengan yang berbau komputer sehingga mengambil bidang minat Ilmu Komputer. Selama kuliah penulis mempelajari beberapa bahasa pemrograman. Bahasa Pemrograman yang pernah penulis pelajari adalah C, C++, dan Java. Semasa menempuh jenjang pendidikan S-1, penulis juga aktif dalam kegiatan non-akademis diantaranya aktif di LDK maupun LDJ. Untuk mendapatkan informasi yang berhubungan dengan Tugas Akhir ini dapat ditujukan ke alamat email: toechfah@gmail.com.