

TUGAS AKHIR - KI141502

Pengenalan Aktivitas Manusia pada Video Menggunakan Variation Energy Images Features

FINTANTO CENDIKIA
NRP 05111440000098

Dosen Pembimbing I
Dr.Eng. Nanik Suciati, S.Kom, M.Kom

Dosen Pembimbing II
Dr.Eng. Chastine Fatichah, S.Kom, M.Kom

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]



TUGAS AKHIR - KI141502

Pengenalan Aktivitas Manusia pada Video Menggunakan Variation Energy Images Features

**FINTANTO CENDIKIA
NRP 05111440000098**

**Dosen Pembimbing I
Dr.Eng. Nanik Suciati, S.Kom, M.Kom**

**Dosen Pembimbing II
Dr.Eng. Chastine Fatichah, S.Kom, M.Kom**

**DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018**

[Halaman ini sengaja dikosongkan]



UNDERGRADUATE THESES - KI141502

HUMAN ACTIVITY RECOGNITION ON VIDEO USING VARIATION ENERGY IMAGE FEATURES

**FINTANTO CENDIKIA
NRP 05111440000098**

**Supervisor I
Dr.Eng. Nanik Suciati, S.Kom, M.Kom.**

**Supervisor II
Dr.Eng. Chastine Fatichah, S.Kom, M.Kom.**

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2018**

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

Pengenalan Aktivitas Manusia pada Video Menggunakan Variation Energy Images Features

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada

Bidang Studi Komputasi Cerdas dan Visi
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :

FINTANTO CENDIKIA

NRP : 05111440000098

Disetujui oleh Dosen Pembimbing Tugas Akhir :

Dr.Eng. Nanik Suciati, S.Kom, M.Kom
NIP: 19710428 199412 2 002

(Pembimbing 1)

Dr. Chastine Fatichah, S.Kom, M.Kom
NIP: 19751220 200112 2 002

(Pembimbing 2)

**SURABAYA
JUNI 2018**

[Halaman ini sengaja dikosongkan]

PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGUNAKAN VARIATION ENERGY IMAGES FEATURES

Nama Mahasiswa : FINTANTO CENDIKIA
NRP : 05111440000098
Jurusan : Informatika FTIK-ITS
Dosen Pembimbing 1 : Dr.Eng. Nanik Suciati, S.Kom, M.Kom.
Dosen Pembimbing 2 : Dr. Chastine Fatichah, S.Kom, M.Kom.

ABSTRAK

Sekarang ini, pengenalan aktivitas manusia pada data video telah menjadi hal yang umum dibicarakan dalam penelitian visi komputer. Hal ini dikarenakan oleh kompleksitas pada data video berupa noise, interaksi objek ganda, intensitas cahaya, dan lain-lain. Penerapan konsep pengenealan aktivitas manusia berdasarkan citra telah diterapkan juga pada berbagai bidang. Dalam bidang hiburan, pengenalan aktivitas dapat digunakan untuk memperbaharui interaksi manusia dan komputer. Dalam bidang kesehatan, pengenalan aktivitas manusia dapat dimanfaatkan untuk membantu rehabilitas pasien. Dalam bidang keamanan, pengenalan aktivitas manusia dapat digunakan untuk deteksi aktivitas yang mencurigakan. Biasanya data video pada bidang keamanan didapat dari *Closed Circuit Television* atau biasa disebut CCTV. Oleh karena itu penulis ingin menerapkan konsep pengenalan aktivitas manusia pada data video CCTV yang berada di Departemen Informatika ITS Surabaya lantai 3 di depan ruang himpunan mahasiswa. Aktivitas yang digunakan dalam tugas akhir ini adalah berlari, berjalan dan melambai.

Ada 3 tahapan utama dalam pemrosesan aktivitas ini. Tahapan-tahapan tersebut adalah segmentasi objek, ekstraksi fitur, dan klasifikasi. Teknik segmentasi objek yang digunakan adalah *Background Subtraction*. Hal ini dikarenakan teknik ini sederhana

dan efektif digunakan untuk segmentasi dari kamera yang statis serta memiliki kemampuan untuk beradaptasi dengan perubahan latar belakang yang terjadi. Kemudian dari objek yang telah dideteksi tersebut akan dibentuk *variation energy image*-nya yang kemudian akan diekstraksi fiturnya yaitu fitur bentuk dan fitur perpindahan. Untuk klasifikasi aktivitasnya, akan digunakan teknik *multiclass-SVM* (*Support Vector Machine*).

Uji coba menggunakan dataset CCTV menunjukkan bahwa metode yang digunakan pada tugas akhir ini menghasilkan akurasi 92,7% pada pagi hari, 91,8% pada siang hari, 93,4% pada malam hari dan 95,8% pada data campuran pagi, siang dan malam hari dengan jumlah *frame* pembentuk VEI yang digunakan adalah 7 *frames*. Pengujian menggunakan metode *k-fold Cross Validation* dengan nilai *k* sebesar 10.

Kata kunci: Video Processing, Human Action Recognition, Background Subtraction, Multiclass SVM, Variation Energy Images.

HUMAN ACTION RECOGNITION ON VIDEO USING VARIATION ENERGY IMAGE FEATURES

Student's Name : FINTANTO CENDIKIA
Student's ID : 0511144000098
Department : Informatika FTIK-ITS
First Advisor : Dr.Eng. Nanik Suciati, S.Kom, M.Kom.
Second Advisor : Dr. Chastine Fatichah, S.Kom, M.Kom.

ABSTRACT

Today, the human activity recognition in video data has become a common feature of computer vision research. This is due to the complexity of the video data in the form of noise, interaction of multiple objects, the intensity of light, and others. The application of the concept of human activity based on image has been applied also to various fields. In the entertainment field, the introduction of activity can be used to update human and computer interactions. In the field of health, the introduction of human activities can be utilized to aid patient rehabilitation. In the field of security, the human activity recognition can be used for the detection of suspicious activity. Usually video data in the security field is obtained from Closed Circuit Television or commonly called CCTV. Therefore the authors want to apply the concept of the human activity recognition on CCTV video data residing in the Department of Informatics ITS Surabaya 3rd floor in front of the student set room. The activities used in this final project are running, walking and waving.

There are 3 main stages in processing this activity. The stages are object segmentation, feature extraction, and classification. Object segmentation technique used is Background Subtraction. This is because this technique is simple and effective to use for the segmentation of static cameras and has the ability to adapt to the background changes that occur. Then from the object

that has been detected will be formed variation energy image then will be extracted feature that is form feature and motion feature. For the classification of activities, will be used multiclass-SVM (Support Vector Machine).

Trials using the CCTV dataset showed that the current method yielded 92.7% in the morning, 91.8% during the day, 93.4% on day and 95.8% in morning, day and day data by amount frames VEI which is 7 frames. The test used k-fold Cross Validation method with k value of 10.

Keywords: Video Processing, Human Action Recognition, Background Subtraction, Multiclass SVM, Variation Energy Images.

KATA PENGANTAR

Segala puji dan syukur bagi Tuhan yang Maha Esa, yang telah melimpahkan rahmat, dan berkat-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul

PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGUNAKAN VARIATION ENERGY IMAGES FEATURES

Pengerjaan Tugas Akhir ini merupakan suatu kesempatan yang berharga bagi penulis. Dengan pengerjaan Tugas Akhir, penulis dapat memperdalam, meningkatkan, serta menerapkan apa yang telah didapatkan penulis selama menempuh perkuliahan di Teknik Informatika ITS.

Terselesaikannya Tugas Akhir ini tidak lepas dari bantuan dan dukungan dari berbagai pihak. Dan dalam kesempatan ini penulis mengucapkan rasa syukur dan terima kasih kepada:

1. Tuhan yang Maha Esa, karena atas izin-Nya lah penulis dapat menyelesaikan Tugas Akhir dengan baik.
2. Kedua orangtua penulis, dan anggota keluarga lainnya yang tiada hentinya memberikan dukungan doa, moral, dan material kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
3. Bapak Dr. Darlis Herumurti, S.Kom., M.Kom., selaku ketua jurusan Teknik Informatika ITS
4. Bapak Radityo Anggoro, S.Kom., M.Sc. selaku Koordinator Tugas Akhir di Teknik Informatika ITS.
5. Ibu Dr.Eng. Nanik Suciati, S.Kom, M.Kom selaku pembimbing I Tugas Akhir yang telah memberikan banyak yang telah memberikan bimbingan dan dukungan selama penulis menyelesaikan Tugas Akhir.
6. Ibu Dr.Eng. Chastine Fatichah, S.Kom, M.Kom selaku pembimbing II Tugas Akhir yang telah memberikan banyak waktu untuk berdiskusi dan memberi semangat dan motivasi kepada penulis untuk menyelesaikan Tugas Akhir.

7. Bapak dan Ibu Dosen di Jurusan Teknik Informatika yang telah memberikan ilmu selama penulis kuliah di Teknik Informatika.
8. Rekan se-topik penelitian, Rina Wijaya Kusuma Wardhani dan Dzaky Zakiyal Fawwaz yang bersedia memberikan waktu untuk berdiskusi dalam pengerjaan Tugas Akhir.
9. Bayu Sektiaji, Hari Setiawan dan Rafiar Rahmansyah yang bersedia meluangkan waktu untuk membantu dalam pembuatan dataset.
10. Seluruh Staf dan karyawan Teknik Informatika yang telah memberikan bantuan selama penulis kuliah di Teknik Informatika.
11. Seluruh teman sekontrakan di MES E/103 Dharmawan, Nobby, Widhi, Hanendyo, dan Fahmi.
12. Rekan-rekan di laboratorium Komputasi Cerdas dan Visi yang telah lama saling mendukung dan menyemangati selama pengerjaan Tugas Akhir.
13. Rekan-rekan di laboratorium Interaksi, Game dan Seni yang telah mendukung pengerjaan Tugas Akhir dan menyemangati selama pengerjaan Tugas Akhir.
14. Seluruh rekan-rekan TC 2014 yang saya banggakan.

Penulis memohon maaf apabila terdapat kekurangan dalam penulisan Tugas Akhir ini. Kritik dan saran penulis harapkan untuk perbaikan dan pembelajaran di kemudian hari. Semoga Tugas Akhir ini dapat memberikan manfaat yang sebesar besarnya.

Surabaya, Juni 2018

Fintanto Cendikia

DAFTAR ISI

LEMBAR PENGESAHAN	vii
ABSTRAK	ix
ABSTRACT	xi
KATA PENGANTAR	xiii
DAFTAR ISI	xv
DAFTAR GAMBAR	xix
DAFTAR TABEL	xxi
DAFTAR KODE SUMBER	xxiii
BAB I PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Rumusan Permasalahan	2
1.3. Batasan Masalah	2
1.4. Tujuan.....	3
1.5. Manfaat.....	3
1.6. Metodologi	3
1.7. Sistematika Penulisan Laporan Tugas Akhir.....	5
BAB II DASAR TEORI	7
2.1. Pengenalan Aktivitas Manusia	7
2.2. Background Subtraction	7
2.3. Gaussian Mixture Model (GMM).....	8
2.3.1. Tahap Pencocokan Input terhadap Distribusi	9
2.3.2. Tahap <i>Update</i> Parameter.....	9
2.3.3. Tahap Pemilihan Distribusi Latar Belakang	10
2.4. Binary Image atau Gambar Biner	11
2.5. Operasi Morfologi	12
2.5.1. <i>Opening</i>	12
2.5.2. <i>Closing</i>	12
2.6. Cropping	13
2.7. Centroid	14
2.8. Median Filter	14
2.9. Variation Energy Images (VEI).....	15
2.10. Ekstraksi Fitur	16
2.10.1. Ekstraksi Fitur Bentuk	16

2.10.2. Ekstraksi Fitur Pergerakan	17
2.11. Support Vector Machine	17
2.12. Multiclass SVM Classifier.....	20
2.13. Confusion Matrix.....	20
BAB III PERANCANGAN PERANGKAT LUNAK.....	23
3.1. Data.....	23
3.1.1. Data Masukan	23
3.1.2. Data Keluaran	24
3.2. Desain Sistem Secara Umum.....	25
3.3. Preprocessing.....	26
3.4. Background Subtraction	28
3.5. Cropping ROI	28
3.6. Pembuatan VEI.....	31
3.7. Ekstraksi Fitur.....	32
3.8. Pelatihan dan Pengujian Multiclass SVM	34
3.9. 10-fold Cross Validation.....	34
BAB IV IMPLEMENTASI.....	37
4.1. Lingkungan implementasi	37
4.2. Implementasi.....	37
4.2.1. <i>Pre-processing</i>	37
4.2.2. <i>Background Subtraction</i>	38
4.2.3. <i>Cropping</i> Objek.....	39
4.2.4. Pembuatan VEI	39
4.2.5. Ekstraksi Fitur	40
4.2.6. Pelatihan <i>Multiclass</i> SVM	40
4.2.7. Pengujian <i>Multiclass</i> SVM	41
4.2.8. Pengujian Pengukuran Akurasi Sistem	42
BAB V PENGUJIAN DAN EVALUASI	45
5.1. Lingkungan Pengujian.....	45
5.2. Data Uji Coba	45
5.3. Skenario Uji Coba.....	47
5.3.1. Perbandingan RGB dan YCbCr pada <i>Background Subtraction</i>	48
5.3.2. Uji Coba Kemampuan Adaptasi Proses <i>Background Subtraction</i>	48

5.3.3. Perbandingan Performa Saat Pagi Hari.....	52
5.3.4. Perbandingan Performa Saat Siang Hari.....	54
5.3.5. Perbandingan Performa Saat Malam Hari	56
5.3.6. Perbandingan Performa Saat Pagi, Siang dan Malam Hari 58	
5.3.7. Uji Coba Kecepatan Sistem untuk Mengenali Satu Gerakan pada Data Video	59
5.4. Analisis Hasil Uji Coba	61
BAB VI KESIMPULAN DAN SARAN	65
6.1. Kesimpulan.....	65
6.2. Saran.....	66
DAFTAR PUSTAKA	67
LAMPIRAN.....	69
BIODATA PENULIS	75

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

Gambar 2.1. Contoh Siluet [1]	8
Gambar 2.2. Gambar Biner	11
Gambar 2.3. (a) sebelum opening (b) setelah opening	12
Gambar 2.4. (a) Citra asli (b) Opening (c) dan (d) Closing.....	13
Gambar 2.5. (a) Citra masukan (b) Setelah <i>cropping</i>	13
Gambar 2.6. Median Filter [12].....	15
Gambar 2.7. Variation Energy Images	15
Gambar 2.8. Pembagian kuadran	16
Gambar 2.9. SVM berusaha menemukan <i>hyperplane</i> terbaik yang memisahkan kedua kelas -1 dan $+1$ [14]	17
Gambar 3.1. Data Masukan.....	24
Gambar 3.2. Arah dan Posisi Gerakan	24
Gambar 3.3. Diagram alir pembelajaran sistem	25
Gambar 3.4. Diagram alir pengujian sistem.....	26
Gambar 3.5. Diagram alir <i>preprocessing</i>	27
Gambar 3.6. RGB <i>frame</i>	27
Gambar 3.7. YCbCr <i>frame</i>	27
Gambar 3.8. Masukan dan keluaran proses <i>cropping</i> objek.....	28
Gambar 3.9. Diagram alir <i>Background Subtraction</i>	29
Gambar 3.10. Masukan dan keluaran <i>Background Subtraction</i> sebelum operasi morfologi	30
Gambar 3.11. Keluaran setelah operasi morfologi.....	30
Gambar 3.12. Diagram alir proses <i>cropping</i> objek	31
Gambar 3.13. Diagram alir pembuatan VEI.....	32
Gambar 3.14. VEI	32
Gambar 3.15. Diagram Alir Ekstraksi Fitur	33
Gambar 3.16. Skema 10-fold <i>Cross Validation</i>	35
Gambar 5.1. (a) Citra RGB dan YCbCr pada Pagi Hari (b) Citra RGB dan YCbCr pada Siang Hari (c) Citra RGB dan YCbCr pada Malam Hari	47
Gambar 5.2. (a) Citra RGB, Citra YCbCr, dan Hasil <i>Background Subtraction</i> pada Saat Pagi (b) Citra RGB, Citra YCbCr, dan Hasil <i>Background Subtraction</i> pada Saat Memasuki Siang (c) Citra	

RGB, Citra YCbCr, dan Hasil *Background Subtraction* setelah Adaptasi.....51

Gambar 5.3. (a) VEI berlari (b) VEI berjalan.....63

Gambar 5.4. (a) Citra masukan (b) Siluet (c) VEI.....63

DAFTAR TABEL

Tabel 2.1. <i>Confusion Matrix</i>	20
Tabel 3.1. Fitur yang Diekstraksi	33
Tabel 4.1. Perangkat yang digunakan	37
Tabel 5.1. Spesifikasi Dataset	46
Tabel 5.2. Perbandingan RGB dan YCbCr sebagai Masukan	49
Tabel 5.3. Hasil Uji Coba Kemampuan Adaptasi	50
Tabel 5.4. Jumlah VEI yang Terbentuk pada Pagi Hari	52
Tabel 5.5. <i>Confusion Matrix</i> Pagi Hari	52
Tabel 5.6. Rata-rata Precision dan Recall Pagi Hari	53
Tabel 5.7. Jumlah VEI yang Terbentuk pada Siang Hari	54
Tabel 5.8. <i>Confusion Matrix</i> pada Siang Hari	55
Tabel 5.9. Rata-rata Precision dan Recall Siang	55
Tabel 5.10. Jumlah VEI yang Terbentuk pada Malam Hari.....	56
Tabel 5.11. <i>Confusion Matrix</i> pada Malam Hari.....	57
Tabel 5.12. Rata-rata Precision dan Recall Malam	57
Tabel 5.13. Jumlah VEI yang Terbentuk.....	58
Tabel 5.14. <i>Confusion Matrix</i>	59
Tabel 5.15. Rata-rata Precision dan Recall Pagi, Siang dan Malam	59
Tabel 5.16. Hasil Uji Coba Kecepatan Sistem	60
Tabel 5.17. Hasil <i>Masking Foreground Binary Mask</i> Terhadap Citra Asli	62

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

Kode Sumber 4.1. <i>Pre-processing</i>	37
Kode Sumber 4.2. <i>Background Subtraction</i>	39
Kode Sumber 4.3. <i>Cropping</i> objek.....	39
Kode Sumber 4.4. Pembuatan VEI	40
Kode Sumber 4.5. Ekstraksi fitur	40
Kode Sumber 4.6. Pelatihan <i>multiclass</i> SVM	41
Kode Sumber 4.7. Pengujian <i>multiclass</i> SVM	42
Kode Sumber 4.8. Pengukuran Akurasi Sistem	43

[Halaman ini sengaja dikosongkan]

BAB I

PENDAHULUAN

Pada bab ini dibahas mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika laporan tugas akhir. Diharapkan dari penjelasan dalam bab ini gambaran tugas akhir secara umum dapat dipahami.

1.1. Latar Belakang

Human activity recognition (HAR) adalah suatu sistem yang bertujuan untuk mengenali aksi dan tujuan dari satu atau beberapa manusia. *Human activity recognition* adalah salah satu topik yang menarik dalam visi komputer, telah diteliti secara ekstensif akhir-akhir ini. Hal ini disebabkan oleh kompleksitas pada data video berupa halangan, kekacauan, interaksi objek ganda, perubahan cahaya, dan lain-lain. Selain itu, adanya persoalan pada proses pengambilan video seperti penyimpangan, pergerakan, dan sudut pandang kamera, serta kompleksitas pada aktivitas manusia itu sendiri sebagai objek non-rigid yang memiliki beragam kelas aktivitas .

Pengenalan aktivitas manusia pada data video telah diterapkan di beberapa bidang serta digunakan sebagai sistem keamanan atau sarana hiburan. Pada sistem keamanan, HAR dapat dimanfaatkan untuk menggantikan manusia dalam hal memonitor keadaan melalui *closed-circuit television*. Jika sistem mendeteksi aktivitas mencurigakan, sistem dapat mengirimkan peringatan kepada manusia. Tertarik akan hal ini, penulis ingin menerapkan pengenalan aktivitas manusia pada data video yang berasal dari *Closed-Circuit Television* (CCTV) di Departemen Informatika ITS Surabaya.

Pendekatan yang digunakan dalam *human activity recognition* biasanya terdiri dari dua tahapan, yaitu mengekstraksi fitur-fitur dari *frames* yang terdapat pada data video kemudian melatih sistem penggolong untuk menggolongkan fitur-fitur tersebut. Aktivitas manusia dapat dibagi menjadi empat aspek [1]: postur tubuh, aktivitas individu, aktivitas interaksi, dan aktivitas kelompok. Dalam tugas akhir ini, yang dibahas adalah aktivitas individu.

Dalam penelitian-penelitian mengenai *human activity recognition* ada beberapa metode yang umumnya digunakan, yaitu pengenalan aktivitas manusia berdasarkan pelacakan jalur aktivitas [2], *3D Convolutional Neural Networks* [3], dan seterusnya. Namun, hasil penelitian-penelitian ini dirasa kurang memuaskan karena memiliki kompleksitas yang tinggi. Hal ini menyebabkan waktu komputasi yang cukup lama dan membutuhkan perangkat keras yang canggih untuk menjalankan sistem tersebut. Oleh karena ini juga, penulis ingin menerapkan pengenalan aktivitas manusia menggunakan fitur *variation energy images* karena memiliki kompleksitas yang rendah [1].

Pada tugas akhir ini akan dilakukan implementasi pengenalan aktivitas manusia berdasarkan *variation energy images features* [1]. Dalam pembuatan sistem pengenalan aktivitas manusia ini diperlukan beberapa tahapan, yaitu *Background Subtraction* untuk mendapatkan *foreground*, *cropping* untuk mendapatkan *region of interest*-nya, ekstraksi fitur bentuk dan fitur perpindahan dari *variation energy images* yang terbentuk, dan pengklasifikasian menggunakan *multiclass-SVM*.

1.2. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana cara segmentasi objek dari data video.
2. Bagaimana cara mendapatkan fitur bentuk dan fitur pergerakan *Variation Energy Images*.
3. Bagaimana cara mengklasifikasikan aktivitas manusia menggunakan SVM (*Support Vector Machine*).
4. Bagaimana mengetahui kinerja dari sistem pengenalan aktivitas manusia pada data video.

1.3. Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Data yang digunakan pada tugas akhir ini berupa data *video* yang diperoleh dari *CCTV* di Departemen Informatika ITS Surabaya lantai 3 depan ruang himpunan.

2. Karena kompleksitas dan keberagaman aktivitas yang ada, pengenalan aktivitas hanya berupa aktivitas yang berasal dari objek tunggal.
3. Aktivitas yang dapat dikenali berjumlah tiga yaitu aktivitas berjalan, berlari, dan melambai.
4. *Deployment* sistem dilakukan dalam bentuk *console*.
5. Bahasa pemrograman yang dipakai adalah Matlab.

1.4. Tujuan

Tujuan yang ingin dicapai dalam pembuatan tugas akhir ini adalah untuk membuat sistem pengenalan aktivitas manusia pada data *video* CCTV berdasarkan *variation energy images features* dengan metode klasifikasi *Support Vector Machine(SVM)*.

1.5. Manfaat

Manfaat tugas akhir ini adalah sebagai langkah awal dalam membangun sistem pengawasan dan keamanan di Departemen Informatika ITS Surabaya melalui data rekaman video CCTV.

1.6. Metodologi

Tahapan-tahapan yang dilakukan dalam pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. Penyusunan proposal Tugas Akhir.

Proposal tugas akhir ini akan mendeskripsikan dan membahas mengenai rencana pembuatan sistem pengenalan aktivitas manusia pada data video CCTV. Secara detil, proposal tugas akhir ini berisi tentang beberapa bagian yaitu latar belakang diajukannya tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat dari hasil pembuatan tugas akhir. Selain itu, dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir dan ringkasan isi yang membahas metode yang akan digunakan dalam tugas akhir. Sub bab metodologi merupakan penjelasan mengenai tahapan penyusunan tugas akhir. Terdapat pula sub bab jadwal pengerjaan yang menjelaskan jadwal pengerjaan tugas akhir dan di akhir

bagian terdapat daftar pustaka untuk mencantumkan referensi yang digunakan dalam tugas akhir.

2. Studi literatur

Pada tahap ini dilakukan pencarian, pengumpulan, penyaringan, pemahaman, dan pembelajaran literatur yang berhubungan dengan *variation energy images*, *multi-class SVM classifier*, *Background Subtraction*, *Gaussian Mixture Model*, *morphological image processing*, *k-fold Cross Validation*. Literatur yang digunakan meliputi: buku referensi, jurnal, dan dokumentasi internet.

3. Perancangan perangkat lunak

Tahap ini meliputi perancangan sistem berdasarkan studi literatur dan pembelajaran konsep teknologi dari perangkat lunak yang ada. Tahap ini mendefinisikan alur dari implementasi. Langkah-langkah yang dikerjakan juga didefinisikan pada tahap ini. Pada tahapan ini dibuat *prototype* sistem, yang merupakan rancangan dasar dari sistem yang akan dibuat. Kemudian dilakukan desain suatu sistem dan desain proses-proses yang ada.

4. Implementasi perangkat lunak

Implementasi merupakan tahap membangun rancangan program yang telah dibuat. Pada tahapan ini merealisasikan rancangan yang terdapat pada tahapan sebelumnya, sehingga menjadi sebuah program yang sesuai dengan apa yang telah direncanakan.

5. Pengujian dan evaluasi

Pada tahap ini dilakukan uji coba terhadap perangkat lunak yang telah dibuat untuk mengetahui kemampuan algoritma yang dipakai, mengamati kinerja sistem, serta mengidentifikasi kendala yang mungkin timbul pada sistem yang dibuat.

6. Penyusunan buku Tugas Akhir.

Pada tahapan ini disusun buku yang memuat dokumentasi mengenai pembuatan serta hasil dari implementasi perangkat lunak yang telah dibuat.

1.7. Sistematika Penulisan Laporan Tugas Akhir

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini.

1. Bab I. Pendahuluan
Bab ini berisi penjelasan mengenai latar belakang masalah, tujuan, dan manfaat dari pembuatan Tugas Akhir. Selain itu rumusan permasalahan, batasan masalah, dan sistematika penulisan juga merupakan bagian dari bab ini.
2. Bab II Tinjauan Pustaka
Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.
3. Bab III Perancangan Perangkat Lunak
Bab ini berisi tentang desain sistem yang disajikan dalam bentuk *pseudocode*.
4. Bab IV. Implementasi
Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa *code* yang digunakan untuk proses implementasi.
5. Bab V. Hasil Uji Coba dan Evaluasi
Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.
6. Bab VI. Kesimpulan dan Saran
Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

[Halaman ini sengaja dikosongkan]

BAB II

DASAR TEORI

Bab ini berisi penjelasan teori-teori yang berkaitan dengan pembuatan sistem pengenalan aktivitas manusia berdasarkan *variation energy images (VEI) features* dengan metode klasifikasi SVM. Penjelasan ini bertujuan untuk memberikan dasar teori yang mendasari pengembangan perangkat lunak.

2.1. Pengenalan Aktivitas Manusia

Tujuan dari pengenalan aktivitas manusia adalah analisa aktivitas manusia secara otomatis atau interpretasi berdasarkan kejadian yang sedang berlangsung dari data video [4].

Pengenalan aktivitas manusia dapat dikelompokkan kedalam beberapa jenis. Pertama, pengenalan pada aktivitas sederhana yang dilakukan manusia seperti berjalan, naik tangga, turun tangga, jogging, dan lain-lain. Kedua, pada aktivitas kompleks yang biasanya mengkombinasikan aktivitas menjadi aktivitas dengan waktu yang lama, seperti; menunggu bus, mengemudi dan lain-lain. Sebuah aktivitas juga dapat dilakukan hanya dengan beberapa anggota tubuh seperti; mengetik, melambaikan tangan dan lain-lain. Aktivitas manusia juga dapat dikenali dengan menggunakan sensor seperti *Accelerometer*, *Gyroscope*, kamera, dan *GPS*. Pengenalan aktivitas manusia dalam tugas akhir ini akan dilakukan dengan sensor kamera.

Algoritma pengenalan aktivitas manusia terbagi dalam 3 tahapan utama [4], yakni *Background Subtraction*, ekstraksi fitur dan klasifikasi. *Background Subtraction* dilakukan untuk mendapatkan objek yang ingin diamat, kemudian akan dilakukan ekstraksi fitur dari objek tersebut. Setelah fitur-fitur didapatkan, akan dilakukan klasifikasi untuk menentukan aktivitas apa yang sedang berlangsung.

2.2. Background Subtraction

Background Subtraction yang dikenal juga sebagai *Foreground Detection*, merupakan sebuah teknik dalam bidang pemrosesan citra dan visi computer dimana foreground dari sebuah

citra diekstraksi untuk pemrosesan lebih lanjut [5]. Secara umum *region of interest* dari sebuah citra adalah objek (manusia, mobil, teks, dan lain-lain) yang berada pada *foreground*-nya. *Background Subtraction* memiliki beberapa teknik seperti *gaussians mixture models*, *median background model*, dan lain-lain.

Hasil dari *background detection* adalah gambar biner yang bisa disebut juga siluet. Siluet ini yang nantinya akan diproses lebih lanjut dalam pengenalan aktivitas manusia. Contoh siluet dapat dilihat pada Gambar 2.1.



Gambar 2.1. Contoh Siluet [1]

Pada tugas akhir ini, teknik *Background Subtraction* yang digunakan adalah *Gaussians Mixture Models*. *Background Subtraction* digunakan untuk mendapatkan objek manusia.

2.3. *Gaussian Mixture Model (GMM)*

Gaussian Mixture Model (GMM) adalah sebuah tipe *density model* yang terdiri dari komponen fungsi fungsi Gaussian. Komponen fungsi ini terdiri dari weight yang berbeda untuk menghasilkan multi-model density. GMM digunakan untuk memodelkan warna-warna latar belakang dari tiap piksel. Tiap piksel memiliki GMM-nya sendiri dan data yang diolah adalah warna piksel yang didapat dari input. Model-model GMM terbentuk dari data warna piksel berdasarkan waktu. Model yang terbentuk dibagi menjadi dua bagian, model latar belakang dan model *non-background* atau *foreground*. Jumlah model GMM yang digunakan mempengaruhi jumlah model latar belakang. Semakin besar jumlah model GMM yang dipakai semakin banyak model latar belakang yang dimiliki oleh suatu piksel [6]. Terdapat beberapa tahap proses untuk metode ini yaitu tahap pencocokan input terhadap distribusi dan tahap pemilihan distribusi yang mencerminkan

latar belakang. Di dalam tahap pencocokan terdapat tahap *update* parameter [7].

2.3.1. Tahap Pencocokan Input terhadap Distribusi

Pada tahap ini input dicocokkan dengan semua distribusi sampai ditemukan distribusi yang paling cocok. Suatu piksel dikatakan masuk dalam suatu distribusi jika nilai piksel tersebut masuk dalam jarak 2.5 standar deviasi dari sebuah distribusi.

$$\mu_k - 2.5 * \sigma_k < X_t < \mu_k + 2.5 * \sigma_k \quad (2.1)$$

dimana X_t adalah vector dari warna piksel (R, G, B) pada waktu t , μ_k adalah vector nilai mean (R, G, B) dari Gaussian ke k , dan σ_k sebagai standar deviasi dari Gaussian ke k . Apabila piksel tidak cocok dengan semua distribusi yang ada maka piksel tersebut dianggap sebagai *foreground* dan dibuat suatu distribusi baru dengan menggantikan distribusi yang paling tidak mencerminkan latar belakang. Distribusi baru memiliki nilai mean sesuai dengan nilai piksel, nilai varians yang tinggi, dan nilai weight yang kecil. Nilai awal yang diberikan pada variabel tersebut sangat mempengaruhi performa dari algoritma GMM.

2.3.2. Tahap Update Parameter

Pada tahap ini dilakukan pembaruan terhadap nilai dari parameter-parameter GMM yang nantinya digunakan untuk mengolah masukan selanjutnya. Nilai yang diperbarui terdiri dari *weight*, *mean*, dan variasi. Nilai *weight* di-update tiap waktu. Untuk mengupdate nilai *weight* digunakan rumus (2.2).

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t} + \alpha(M_{k,t}) \quad (2.2)$$

dimana $\omega_{k,t}$ adalah weight dari Gaussian ke k pada waktu t , α adalah learning rate dan nilai $M_{k,t}$ adalah 1 untuk model yang cocok dan 0 untuk model lainnya. Setelah nilai weight diperbarui dilakukan normalisasi sehingga total *weight* dari semua distribusi tidak lebih dari satu.

Nilai mean dari suatu distribusi diperbarui setiap ada nilai piksel yang cocok dengan distribusi tersebut. Untuk memperbarui nilai *mean* digunakan rumus (2.3).

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho X_t \quad (2.3)$$

dimana

$$\rho = \alpha \eta(X_t | \mu_k, \Sigma_k) \quad (2.4)$$

dimana η adalah fungsi *Gaussian Probability Density* (GPD), ρ adalah ukuran *temporal window* yang digunakan untuk mengisi *probability density function*.

$$\eta(X | \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{1}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)} \quad (2.5)$$

dimana Σ adalah matriks kovarian, $|\Sigma|$ adalah determinan dari kovarian, pangkat T adalah *transpose* matriks, pangkat -1 adalah invers matrix, e adalah eksponen, π adalah phi, dan n adalah ukuran vektor X (R, G, B) dimana *covariance* didapat dari :

$$\Sigma_{k,t} = \sigma_k^2 I \quad (2.6)$$

dimana I adalah matrix identitas dan σ_k^2 adalah varians dari Gaussian ke k .

Nilai standar deviasi dari suatu distribusi diperbarui setiap ada nilai piksel yang cocok dengan distribusi tersebut. Untuk memperbarui nilai standar deviasi digunakan rumus:

$$\sigma_t^2 = (1 - \rho) \sigma_{t-1}^2 + \rho (X_t - \mu_t)^t (X_t - \mu_t) \quad (2.7)$$

2.3.3. Tahap Pemilihan Distribusi Latar Belakang

Pada tahap ini dipilih model-model yang mencerminkan latar belakang. Pertama model-model diurutkan berdasarkan ω/σ^2 sehingga distribusi yang paling mencerminkan latar belakang tetap di atas dan yang tidak mencerminkan latar belakang ada di bawah yang nantinya digantikan oleh distribusi yang lain. Untuk memilih B distribusi

pertama yang dijadikan distribusi latar belakang digunakan rumus (2.8).

$$B = \arg \min(\sum_{k=1}^b \omega_k > T) \quad (2.8)$$

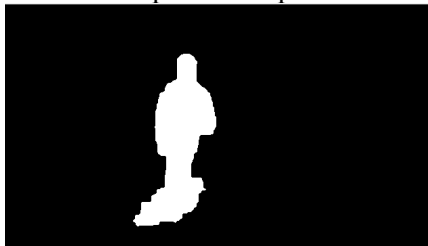
dimana T adalah proporsi terkecil dari data yang sebaiknya dihitung sebagai latar belakang.

2.4. *Binary Image* atau Gambar Biner

Gambar biner merupakan gambar digital yang setiap pikselnya hanya memiliki dua kemungkinan nilai warna yakni 0 merepresentasikan warna hitam dan 1 merepresentasikan warna putih [8].

Biasanya proses pembineran dilakukan dengan membulatkan keatas atau kebawah untuk setiap nilai keabuan dari piksel yang berada diatas atau dibawah nilai threshold. Metode untuk menentukan besarnya nilai *threshold* disebut *thresholding*. Citra biner diperoleh melalui proses pemisahan piksel-piksel berdasarkan derajat keabuan yang dimilikinya. Piksel yang memiliki derajat keabuan lebih kecil dari nilai threshold diberikan nilai 0, sementara piksel yang memiliki derajat keabuan yang lebih besar dari nilai *threshold* batas diberikan nilai 1 [8].

Pada pembuatan siluet hasil *Background Subtraction*, gambar biner didapatkan dari hasil menentukan apakah sebuah piksel merupakan latar belakang atau bukan. Jika merupakan latar belakang, piksel tersebut akan diisi oleh 0 dan akan diisi 1 jika bukan merupakan latar belakang. Gambar biner dapat dilihat pada Gambar 2.2.



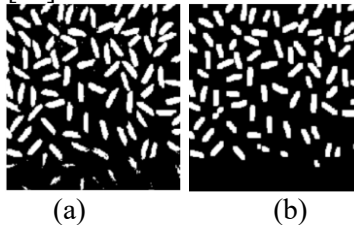
Gambar 2.2. Gambar Biner

2.5. Operasi Morfologi

Operasi morfologi memiliki peranan yang penting dalam pengolahan citra digital [9]. Prinsip dari operasi morfologi adalah mengekstrak komponen-komponen citra yang berguna dalam representasi dan deskripsi bentuk seperti ekstrak *boundary* dari suatu region. Operasi morfologi sering juga digunakan untuk menghilangkan *noise* pada citra [10].

2.5.1. Opening

Opening umumnya digunakan untuk menghaluskan kontur dari sebuah objek, memutuskan garis tipis yang menghubungkan dua region besar dan menghilangkan tonjolan tipis. Ilustrasi penggunaan operasi opening dengan structuring element ditunjukkan pada citra seperti pada Gambar 2.3. Gambar 2.3 (a) menunjukkan citra sebelum dilakukan opening. Kemudian pada Gambar 2.3 (b), menunjukkan hasil opening dari citra [10].

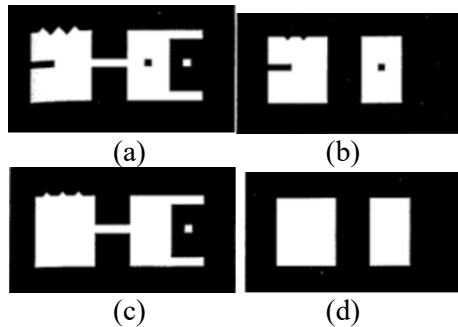


Gambar 2.3. (a) sebelum opening (b) setelah opening

2.5.2. Closing

Closing digunakan untuk menghaluskan bagian dari kontur tetapi berbeda dengan opening, *closing* biasanya menyatukan bagian kecil yang terputus dan menyatukan cekungan yang panjang dan tipis. *Closing* juga digunakan untuk menghilangkan lubang kecil dan mengisi celah yang terdapat dalam kontur [10]. Ilustrasi penggunaan operasi opening dan closing ditunjukkan seperti pada Gambar 2.4. Operasi opening seolah-olah akan membuka sebuah citra seperti pada Gambar 2.4 (b). Sebaliknya, operasi closing seolah-olah menutup

sebuah citra seperti yang terlihat pada Gambar 2.4 (c) dan Gambar 2.4(d).



Gambar 2.4. (a) Citra asli (b) Opening (c) dan (d) Closing

2.6. *Cropping*

Cropping adalah proses pemotongan citra pada koordinat tertentu dengan ukuran tertentu untuk mendapat kan *region of interest* (ROI). ROI adalah suatu daerah tertentu dari sebuah citra yang ingin diamati lebih lanjut. ROI dalam tugas akhir ini adalah siluet manusia.

Dalam menentukan koordinat untuk melakukan *cropping* ini, teknik yang digunakan adalah menemukan *centroid* dari objek yang akan di-*crop* [11]. Setelah ditemukan *centroid* dari objek, *cropping* akan dilakukan dengan ukuran sebesar objek itu sendiri. Tinggi *cropping* akan ditentukan dari tinggi dari objek tersebut, dan lebar *cropping* akan mengikuti lebar dari objek tersebut. Contoh proses *cropping* dapat dilihat pada Gambar 2.5.



Gambar 2.5. (a) Citra masukan (b) Setelah *cropping*

2.7. Centroid

Centroid adalah koordinat titik tengah dari suatu objek. Terkadang, *centroid* dapat berada diluar dari objek. Untuk menentukan *centroid* dari sebuah citra biner sama dengan cara menemukan *centroid* dari sebuah matriks. Untuk menentukan *centroid*, dapat digunakan persamaan (2.9).

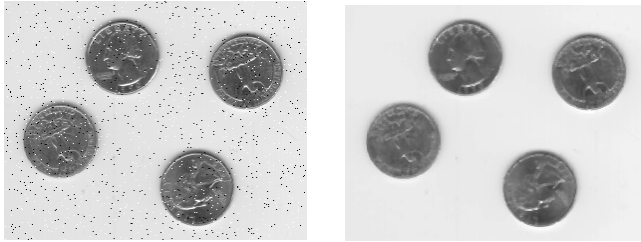
$$(cX, cY) = \frac{\sum_{x=1, y=1}^{x=M, y=N} (x, y) [P(x, y) = 1]}{(M, N)} \quad (2.9)$$

dimana cX, cY adalah koordinat *centroid*, M adalah kolom dari matriks, N adalah baris dari matriks, $P(x, y)$ adalah nilai matriks pada posisi x, y .

2.8. Median Filter

Metode *median filter* merupakan filter non linier yang dikembangkan oleh Tukey. Metode tersebut berfungsi untuk mengurangi noise dan menghaluskan citra. Dikatakan non linier karena cara kerja *filter* ini tidak termasuk kedalam kategori operasi konvolusi. Operasi nonlinier dihitung dengan cara mengurutkan nilai intensitas sekelompok piksel, kemudian mengganti nilai piksel yang diproses dengan nilai tertentu [12].

Pada *median filter* suatu *window* atau penapis yang memuat sejumlah piksel ganjil digeser titik per titik pada seluruh daerah citra. Lalu nilai – nilai tersebut diurutkan secara *ascending* untuk kemudian dihitung nilai mediannya. Nilai median tersebut akan menggantikan nilai yang berada pada pusat bidang *window*. Dari beberapa penelitian metode ini sangat cocok untuk reduksi *noise salt and pepper*. Contoh penerapan *median filter* dapat dilihat pada Gambar 2.6.



Gambar 2.6. Median Filter [12]

2.9. *Variation Energy Images (VEI)*

VEI adalah sebuah gambar yang dapat mendeskripsikan fitur-fitur dari sebuah aksi [1]. VEI terbentuk dengan mencari perbedaan antar siluet ditiap frame selama beberapa frame. Contoh VEI dapat dilihat pada Gambar 2.7. Perbedaan siluet dapat dicari dengan persamaan (2.10). $V(x,y)$ diinisialisasi sebagai gambar hitam atau seluruh pikselnya bernilai 0.

$$V(x, y) = \sum_{x=1, y=1}^{M, N} |V(x, y) - A(x, y)| \quad (2.10)$$

dimana $V(x, y)$ adalah nilai piksel VEI pada posisi (x, y) , M adalah panjang gambar, N adalah lebar gambar, $A(x, y)$ adalah nilai piksel gambar siluet pertama pada posisi (x, y) .



Gambar 2.7. Variation Energy Images

2.10. Ekstraksi Fitur

Setelah VEI terbentuk, akan dilakukan ekstraksi fitur. Fitur-fitur yang akan diekstraksi terbagi menjadi dua bagian, yaitu fitur bentuk dan fitur pergerakan [1].

2.10.1. Ekstraksi Fitur Bentuk

VEI yang terbentuk akan dibagi menjadi empat kuadran seperti gambar. Masing-masing kuadran akan dicari area gerak efektifnya dengan rumus sebagai berikut:

$$S = \frac{\sum_{x=1}^M \sum_{y=1}^N V(x,y)}{M \cdot N} \quad (2.11)$$

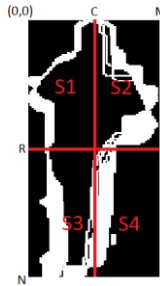
$$S_1 = \frac{\sum_{x=1}^C \sum_{y=1}^R V(x,y)}{C * R} \quad (2.12)$$

$$S_2 = \frac{\sum_{x=C}^M \sum_{y=1}^R V(x,y)}{(M-C) * R} \quad (2.13)$$

$$S_3 = \frac{\sum_{x=1}^C \sum_{y=R}^N V(x,y)}{C * (N-R)} \quad (2.14)$$

$$S_4 = \frac{\sum_{x=C}^M \sum_{y=R}^N V(x,y)}{(M-C) * (N-R)} \quad (2.15)$$

Dimana S adalah area gerak efektif keseluruhan, $S1$, $S2$, $S3$, dan $S4$ adalah area gerak efektif di masing-masing kuadran. Pembagian kuadran dapat dilihat pada Gambar 2.8



Gambar 2.8. Pembagian kuadran

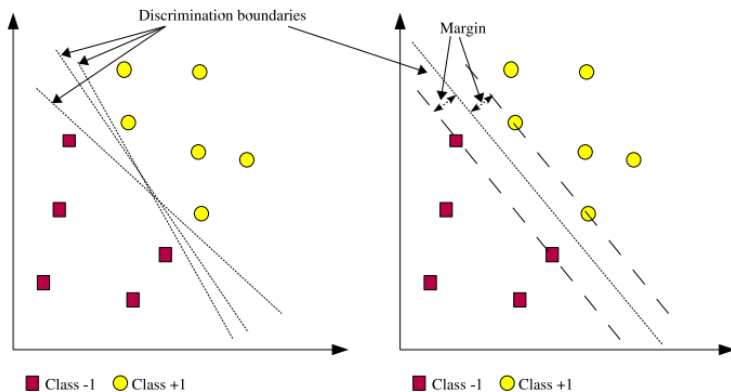
2.10.2. Ekstraksi Fitur Pergerakan

Fitur pergerakan dibentuk dengan menggunakan persamaan (2.15). $d(p, q)$ merupakan jarak antara dua titik koordinat p dan q , digunakan untuk mencari jarak perpindahan objek. p dan q adalah koordinat *centroid* objek pada *frame* pertama dan *frame* ke- N sebelum dilakukan *cropping*. N adalah jumlah *frame* yang digunakan untuk membentuk VEI. Misal, VEI dibentuk menggunakan 7 *frame*, maka p adalah titik *centroid* objek pada *frame* pertama dan q adalah titik *centroid* objek pada *frame* ke-7.

$$d(p, q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2} \quad (2.16)$$

2.11. Support Vector Machine

Konsep SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah kelas pada *input space* [13]. Gambar 2.9 memperlihatkan beberapa *pattern* yang merupakan anggota dari dua buah class : +1 dan -1 . *Pattern* yang tergabung pada class -1 disimbolkan dengan warna merah (kotak), sedangkan *pattern* pada class +1, disimbolkan dengan warna kuning (lingkaran) [14].



Gambar 2.9. SVM berusaha menemukan *hyperplane* terbaik yang memisahkan kedua kelas -1 dan +1 [14]

Problem klasifikasi dapat diterjemahkan dengan usaha menemukan garis (*hyperplane*) yang memisahkan antara kedua kelompok tersebut. Berbagai alternatif garis pemisah (*discrimination boundaries*) ditunjukkan pada Gambar 2.9. *Hyperplane* pemisah terbaik antara kedua class dapat ditemukan dengan mengukur margin *hyperplane* tersebut dan mencari titik maksimalnya. *Margin* adalah jarak antara *hyperplane* tersebut dengan *pattern* terdekat dari masing-masing kelas. *Pattern* yang paling dekat ini disebut sebagai *support vector*. Garis solid pada Gambar 2.9 sebelah kanan menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua kelas, sedangkan titik merah dan kuning yang berada dalam lingkaran hitam adalah *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM [13].

Tiap data yang tersedia dinotasikan sebagai $x_i = \{x_1, x_2, \dots, x_N\} \in R^n$ dimana n adalah dimensi data atau banyaknya fitur dan N adalah banyaknya data. Kemudian label dari masing-masing data dinotasikan dengan $y_N \in \{-1, +1\}$. Diasumsikan kedua class -1 dan $+1$ dapat terpisah secara sempurna oleh *hyperplane* berdimensi n , yang didefinisikan pada persamaan (2.16)

$$w \cdot x_i + b = 0 \quad (2.17)$$

Data x_i yang tergolong ke dalam kelas negatif adalah mereka yang memenuhi pertidaksamaan (2.17).

$$w \cdot x_i + b \leq -1 \quad (2.18)$$

Adapun data x_i yang tergolong ke dalam kelas positif, adalah mereka yang memenuhi pertidaksamaan (2.18).

$$w \cdot x_i + b \geq 1 \quad (2.19)$$

Optimal margin dihitung dengan memaksimalkan jarak antara *hyperplane* dan *pattern* terdekat. Jarak ini dirumuskan sebagai $1/||w||$ dimana $||w||$ adalah dari vektor normal *weight* w . Untuk memaksimalkan jarak tersebut, w harus diminimalkan dengan cara

diformulasikan ke dalam *Quadratic Programming (QP) problem*, meminimalkan persamaan (2.19) dengan konstrain (2.20).

$$||w||^2 \quad (2.20)$$

$$y_i(w \cdot x_i + b) - 1 \geq 0, \forall_i \quad (2.21)$$

dimana y_i adalah label ke- i (+1 atau -1). Kemudian masalah di atas dapat diselesaikan dengan persamaan *Lagrange Multipliers* pada (2.21).

$$L(w, b, \alpha) = \frac{1}{2} |w|^2 - \sum_{i=1}^N \alpha_i y_i (w \cdot x_i + b - 1) \quad (2.22)$$

α_i adalah *Lagrange multiplier* yang berkorespondensi dengan x_i . Nilai α_i adalah nol atau positif. Untuk menyelesaikan masalah tersebut, pertama-tama minimalkan L terhadap w , dan memaksimalkan L terhadap α_i . Dengan memodifikasi (2.21), *maximization problem* di atas dapat direpresentasikan dalam α_i seperti pada persamaan (2.22) dengan memperhatikan konstrain (2.23).

$$\sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j x_i x_j \quad (2.23)$$

$$\alpha_i \geq 0, \sum_{i=1}^N \alpha_i y_i = 0 \quad (2.24)$$

Solusi dari *problem* ini akan menghasilkan banyak α_i dengan nilai nol. Data yang berkorespondensi dengan α_i yang tidak nol, merupakan *support vectors*, yaitu data yang memiliki jarak terdekat dengan *hyperplane*.

Namun penjelasan di atas berdasarkan asumsi bahwa kedua belah kelas dapat terpisah secara sempurna oleh *hyperplane*. Akan tetapi, pada umumnya kedua buah kelas tersebut tidak dapat terpisah secara sempurna. Hal ini menyebabkan proses optimisasi tidak dapat diselesaikan, karena tidak ada w dan b yang memenuhi pertidaksamaan (2.20). Untuk itu, pertidaksamaan (2.20) dimodifikasi dengan memasukkan *slack variable* menjadi (2.24).

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, \forall_i \quad (2.25)$$

Demikian juga dengan (2.19), sehingga diperoleh

$$\frac{1}{2}|w|^2 + C \sum_{i=1}^N \xi_i \quad (2.26)$$

Parameter C bertugas mengontrol *tradeoff* antara *margin* and *classification error*. Semakin besar nilai C , semakin besar *penalty* yang dikenakan untuk tiap *classification error*.

2.12. Multiclass SVM Classifier

Pada dasarnya SVM merupakan pengklasifikasi biner, yang artinya hanya dapat mengklasifikasi dua kelas saja. Oleh karena itu, untuk melakukan klasifikasi lebih dari dua kelas, SVM harus dilatih untuk memiliki dua atau lebih pengklasifikasi biner [14].

Sebuah teknik yang ada untuk melakukan pengklasifikasian lebih dari dua kelas adalah *one versus one*. Pengklasifikasi akan membuat kombinasi pasangan antar kelas dan membuat pengklasifikasi biner untuk setiap pasang. Dalam fase pengklasifikasian, semua pengklasifikasi biner akan diuji. Setiap pengklasifikasi biner akan memberikan sebuah hasil. Hasil-hasil ini merupakan sebuah voting dan kelas yang memiliki voting terbanyak merupakan hasil klasifikasinya [14].

2.13. Confusion Matrix

Confusion matrix merupakan matriks yang mengandung informasi tentang kelas sebenarnya dan prediksi yang dihasilkan oleh sistem klasifikasi. *Confusion matrix* banyak digunakan untuk menguji performa dari suatu metode klasifikasi. Struktur *confusion matrix* untuk tiga kelas ditunjukkan pada Tabel 2.1 sebagai berikut.

Tabel 2.1. Confusion Matrix

		Nilai Prediksi		
		1	2	3
Nilai Sebenarnya	1	TP1	F1.2	F1.3
	2	F2.1	TP2	F2.3
	3	F3.1	F3.2	TP3

Keterangan:

TP = Jumlah data uji yang kelas prediksinya sama dengan kelas yang sebenarnya.

F = Jumlah kelas prediksi yang tidak sesuai dengan kelas sebenarnya.

Beberapa nilai evaluasi yang bisa dihitung berdasarkan confusion matrix untuk mengetahui performa *classifier* yaitu *accuracy*, *recall*, dan *precision*. *Accuracy* adalah perbandingan jumlah total data yang prediksi kelas hasil klasifikasinya sesuai dengan *ground truth* terhadap seluruh data. Rumus perhitungan dari akurasi ditunjukkan pada persamaan (2.26).

$$Accuracy = \frac{TP1+TP2+TP3}{TP1+TP2+TP3+F1.2+F1.3+F2.1+F2.3+F3.1+F3.2} \quad (2.27)$$

Recall adalah perbandingan dari jumlah data yang prediksi kelas hasil klasifikasinya sesuai dengan *ground truth* terhadap seluruh data berkelas benar pada ground truth. Rumus perhitungan Sensitivity ditunjukkan pada persamaan (2.27), (2.28) dan (2.29).

$$Recall1 = \frac{TP1}{TP1+F1.2+F1.3} \quad (2.28)$$

$$Recall2 = \frac{TP2}{TP2+F2.1+F2.3} \quad (2.29)$$

$$Recall3 = \frac{TP3}{TP3+F3.1+F3.2} \quad (2.30)$$

Precision adalah adalah perbandingan dari jumlah data yang prediksi kelas hasil klasifikasinya sesuai dengan *ground truth* terhadap keseluruhan data yang terklarifikasi benar. Rumus perhitungan precision ditunjukkan pada persamaan (2.30), (2.31) dan (2.32).

$$Precision1 = \frac{TP1}{TP1+F2.1+F3.1} \quad (2.31)$$

$$Precision2 = \frac{TP2}{TP2+F1.2+F3.2} \quad (2.32)$$

$$Precision3 = \frac{TP3}{TP3+F1.3+F2.3} \quad (2.33)$$

[Halaman ini sengaja dikosongkan]

BAB III

PERANCANGAN PERANGKAT LUNAK

Pada bab ini dijelaskan mengenai rancangan sistem perangkat lunak yang akan dibuat. Perancangan yang dijelaskan meliputi data dan proses. Data yang dimaksud adalah data yang akan diolah dalam perangkat lunak baik digunakan sebagai pembelajaran maupun pengujian sehingga tujuan Tugas Akhir ini bisa tercapai. Proses yaitu tahap-tahap yang ada dalam sistem sebagai pengolah data meliputi *pre-processing*, *Background Subtraction*, pembuatan VEI, ekstraksi fitur, dan klasifikasi SVM.

3.1. Data

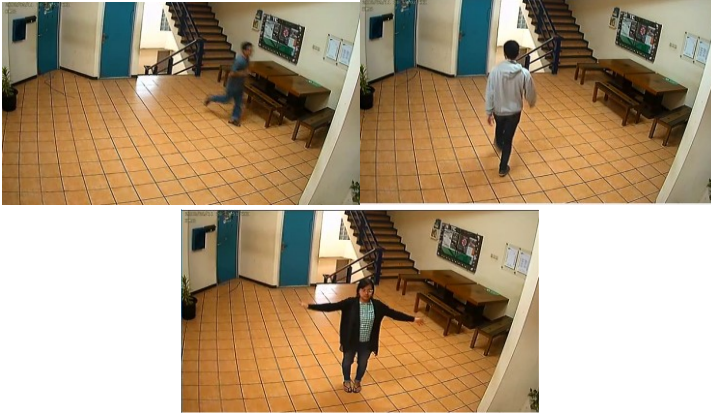
Pada sub bab ini akan dijelaskan mengenai data yang digunakan sebagai masukan perangkat lunak untuk selanjutnya diolah dan dilakukan pengujian sehingga menghasilkan data keluaran yang diharapkan. Kumpulan data atau dataset yang digunakan pada penelitian ini didapatkan dari hasil rekaman CCTV di departemen informatika lantai 3 depan ruang himpunan.

3.1.1. Data Masukan

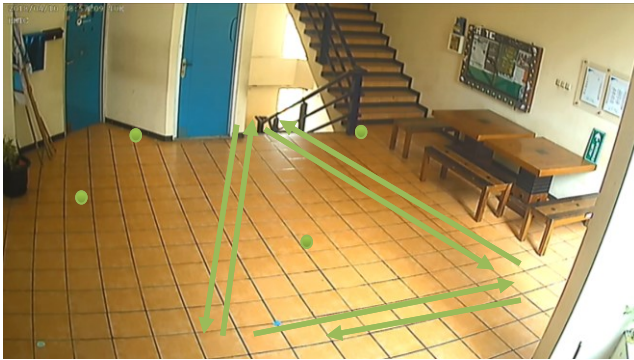
Data masukan adalah data yang digunakan sebagai masukan dari sistem. Data masukan pada sistem pengenalan aktivitas manusia adalah data video berukuran 636x360 piksel dengan citra warna RGB dan memiliki *framerate* sebesar 25 *fps*. Contoh data masukan dapat dilihat pada Gambar 3.1.

Satu data video menampilkan satu objek yang melakukan satu jenis aktivitas. Aktivitas yang dilakukan yaitu aktivitas berlari, berjalan, dan melambatkan kedua tangan. Masing-masing aktivitas tersebut terdiri dari beberapa ragam pola arah dan titik gerakan pada latar video. Seperti yang ditunjukkan pada Gambar 3.2, ada enam arah aktivitas lari, enam arah aktivitas jalan, dan empat titik aktivitas melambatkan kedua tangan. Contoh potongan video *CCTV* yang akan menjadi data masukan ditunjukkan pada Gambar 3.1. Masing-masing aktivitas diperagakan oleh enam orang secara bergantian. Dataset

direkam dalam tiga kategori waktu yang berbeda yaitu pagi pukul 08.00-10.00, siang pukul 13.00-14.00, dan malam pukul 20.00-22.00. Sehingga jumlah video rekaman aktivitas adalah 18 video untuk masing-masing kategori waktu atau total 54 video.



Gambar 3.1. Data Masukan



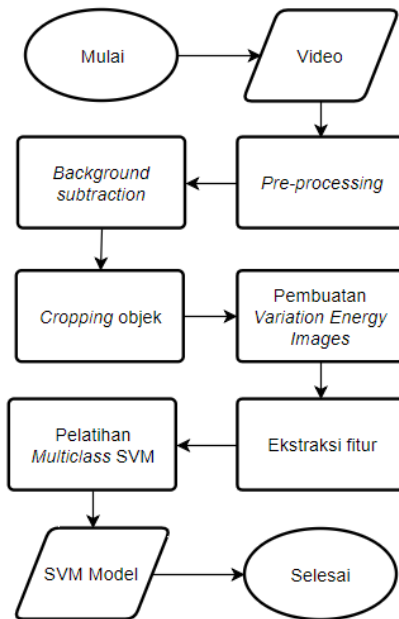
Gambar 3.2. Arah dan Posisi Gerakan

3.1.2. Data Keluaran

Data keluaran sistem pengenalan aktivitas manusia merupakan prediksi aktivitas yang dilakukan objek pada data masukan yaitu lari, jalan, atau melambatkan kedua tangan.

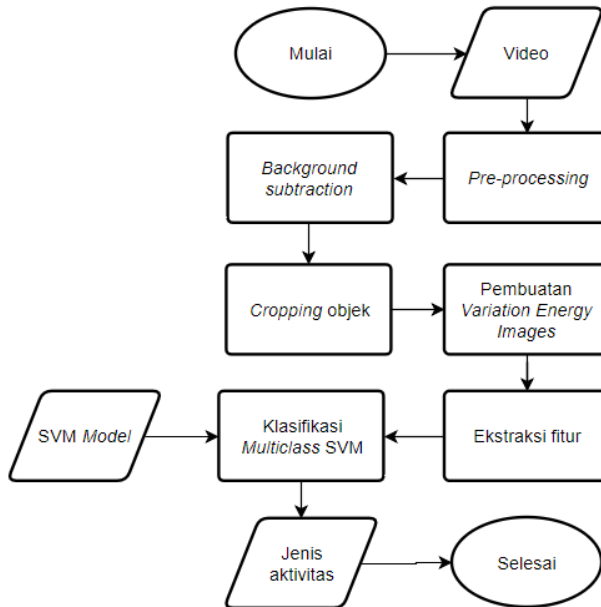
3.2. Desain Sistem Secara Umum

Rancangan perangkat lunak pengenalan aktivitas manusia terbagi dalam dua tahap, yakni tahap pembelajaran dan tahap pegetesan. Pada tahap pembelajaran proses dimulai dengan *Background Subtraction*, dilanjutkan dengan *cropping* objek, pembentukan VEI, ekstraksi fitur, dan terakhir dilakukan pembelajaran model SVM. Diagram alir proses pembelajaran ditunjukkan pada Gambar 3.3.



Gambar 3.3. Diagram alir pembelajaran sistem

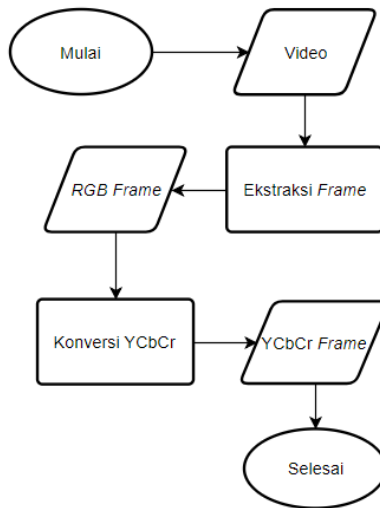
Pada tahap pengtesan proses dimulai dengan membaca masukan berupa video. Proses pengenalan aktivitas manusia terdiri dari beberapa proses besar, yaitu *preprocessing*, *Background Subtraction*, pembuatan VEI, ekstraksi fitur, pengklasifikasian dengan SVM. Diagram alir proses pengujian sistem ditunjukkan pada Gambar 3.4.



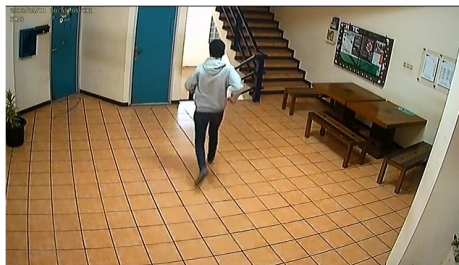
Gambar 3.4. Diagram alir pengujian sistem

3.3. Preprocessing

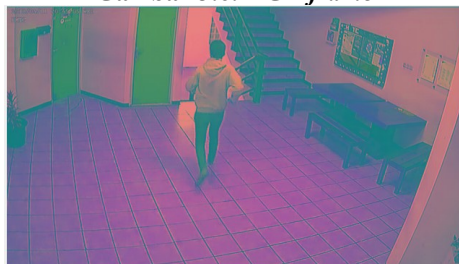
Setiap frame video yang diproses tidak langsung dilakukan *Background Subtraction*, ekstraksi fitur dan pengujian untuk menentukan aktivitas apa yang sedang dilakukan oleh objek. Tahap awal yang dilakukan adalah *preprocessing* yang bertujuan untuk mengurangi bayangan dari objek. Pada tahap ini, data masukan yang memiliki *color space* RGB dikonversikan menjadi *color space* YCbCr. *Color space* YCbCr dipilih karena dapat mengurangi bayangan objek lebih baik dibandingkan dengan *color space* lainnya. Diagram alir proses *preprocessing* dapat dilihat pada Gambar 3.5. Masukan citra RGB dapat dilihat pada Gambar 3.6 dan keluaran citra YCbCr dapat dilihat pada Gambar 3.7.



Gambar 3.5. Diagram alir *preprocessing*



Gambar 3.6. RGB *frame*



Gambar 3.7. YCbCr *frame*

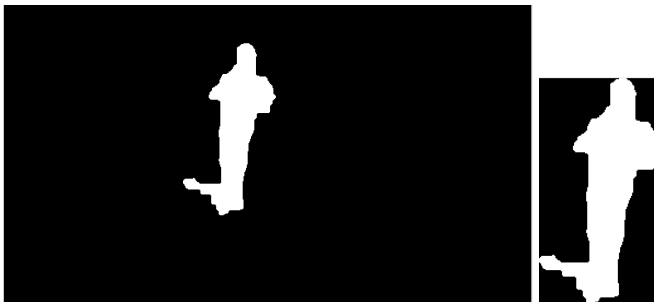
3.4. *Background Subtraction*

Pada tahap ini, sistem akan menentukan *pixel* mana yang merupakan *background*, dan *pixel* mana yang merupakan *foreground*. Hasil dari deteksi *foreground* akan perbaiki dulu dengan filter-filter yang ada, seperti operasi morfologi, *median filter* dan sebagainya. Hal ini bertujuan mengurangi *noise* yang ada. Diagram alir tahap *Background Subtraction* dapat dilihat pada Gambar 3.9.

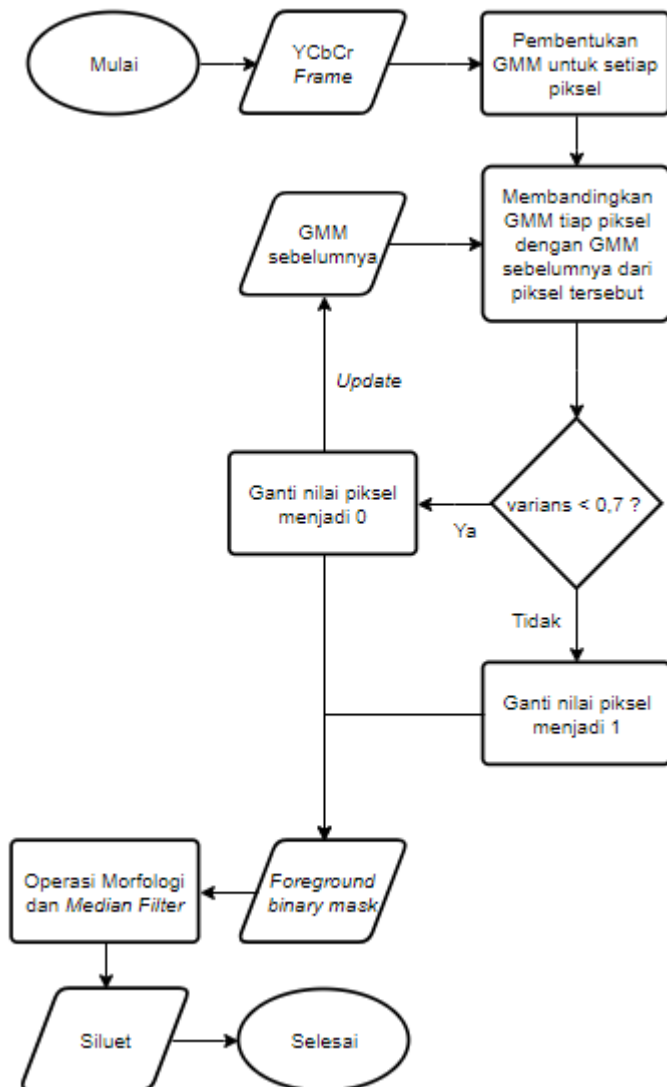
Masukan dari tahapan ini adalah *frame* yang telah dikonversi menjadi YCbCr, sedangkan keluaran dari tahapan ini adalah *binary mask* atau siluet dari objek. Masukan dan keluaran *Background Subtraction* sebelum dilakukan operasi morfologi dapat dilihat pada Gambar 3.10 dan keluaran setelah dilakukan operasi morfologi dapat dilihat pada Gambar 3.11. Operasi morfologi yang dilakukan adalah *opening* dan *closing*.

3.5. *Cropping ROI*

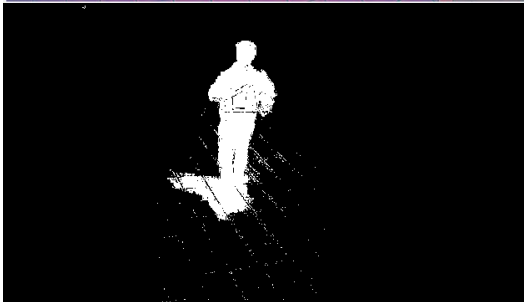
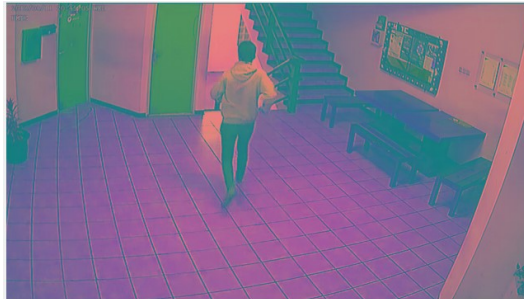
Objek akan di-*crop* sesuai dengan ukuran objeknya. Kemudian hasil *cropping* akan di-*resize* ke ukuran 400x200. Hal ini bertujuan untuk menyamakan ukuran objek agar VEI yang didapat lebih presisi. Alur dari *cropping* objek dapat dilihat pada Gambar 3.12. Masukan dan keluaran dari tahap ini dapat dilihat pada Gambar 3.8.



Gambar 3.8. Masukan dan keluaran proses *cropping* objek



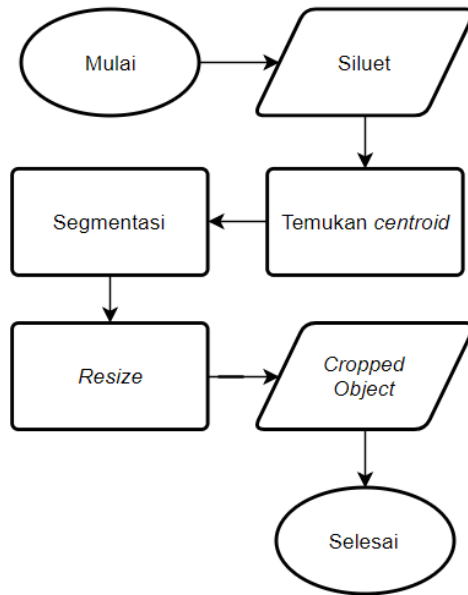
Gambar 3.9. Diagram alir *Background Subtraction*



Gambar 3.10. Masukan dan keluaran *Background Subtraction* sebelum operasi morfologi



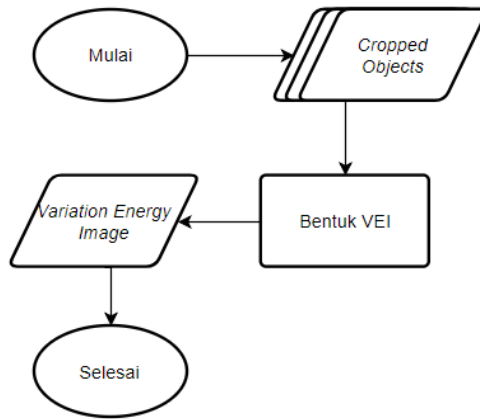
Gambar 3.11. Keluaran setelah operasi morfologi



Gambar 3.12. Diagram alir proses *cropping* objek

3.6. Pembuatan VEI

Pada tahap ini akan dibuat VEI dari setiap beberapa *frame*. VEI terbentuk dengan mencari perbedaan antar siluet setiap *frame* selama beberapa *frame*. Alur dari tahap ini dapat dilihat pada Gambar 3.13. Contoh VEI dapat dilihat pada Gambar 3.14. Perbedaan dapat dicari menggunakan persamaan (2.10).



Gambar 3.13. Diagram alir pembuatan VEI

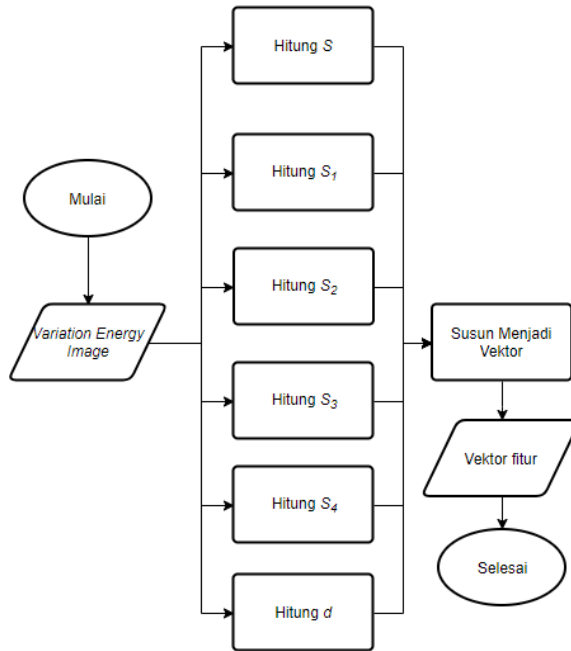


Gambar 3.14. VEI

3.7. Ekstraksi Fitur

VEI yang didapat akan diekstraksi fitur-fiturnya. Fitur-fitur tersebut adalah S , S_1, S_2, S_3, S_4, d seperti yang dapat dilihat pada Tabel 3.1. Rumus untuk mendapatkan fitur-fitur tersebut dapat dilihat pada formula (2.11) sampai dengan (2.16) Setelah kedelapan fitur terekstraksi, fitur-fitur ini akan disusun menjadi sebuah vektor berukuran 6×1 . Vektor ini yang akan digunakan pada tahapan

pelatihan dan klasifikasi menggunakan *multiclass* SVM. Diagram alir ekstraksi fitur dapat dilihat pada Gambar 3.15.



Gambar 3.15. Diagram Alir Ekstraksi Fitur

Tabel 3.1. Fitur yang Diekstraksi

Fitur	Deskripsi	Contoh Nilai	Persamaan
S	Area gerak keseluruhan	37.213800	(2.11)
S_1	Area gerak pada kuadran 1	14.701200	(2.12)

S_2	Area gerak pada kuadran 2	22.164550	(2.13)
S_3	Area gerak pada kuadran 3	70.014150	(2.14)
S_4	Area gerak pada kuadran 4	43.186050	(2.15)
d	Jarak perpindahan dari frame pertama sampai frame terakhir yang digunakan untuk membentuk VEI	47.367165	(2.16)

3.8. Pelatihan dan Pengujian *Multiclass SVM*

Setelah vektor yang berisi enam fitur tersebut ditemukan, akan dilakukan klasifikasi dengan metode *multiclass Support Vector Machine(SVM)*. Tahap ini terbagi jadi dua, yaitu pelatihan dan pengujian. Dataset untuk pelatihan adalah dalam bentuk vektor berukuran 6 yang didapat dari data video dengan susunan S, S_1, S_2, S_3, S_4, d dari VEI *features*. Data ujinya adalah VEI *features* dalam bentuk vektor berukuran 6 yang didapat dari data video dengan susunan S, S_1, S_2, S_3, S_4, d . Pengujian akan dilakukan dengan *10-fold cross validation*.

3.9. *10-fold Cross Validation*

Cross-validation (CV) adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritma dimana data dipisahkan menjadi dua subset yaitu data proses pembelajaran dan data validasi / evaluasi. Model atau algoritma dilatih oleh subset pembelajaran dan divalidasi oleh subset validasi. Selanjutnya

pemilihan jenis CV dapat didasarkan pada ukuran dataset. Biasanya CV K-fold digunakan karena dapat mengurangi waktu komputasi dengan tetap menjaga keakuratan estimasi.

10 fold CV adalah salah satu K fold CV yang direkomendasikan untuk pemilihan model terbaik karena cenderung memberikan estimasi akurasi yang kurang bias dibandingkan dengan CV biasa, leave-one-out CV dan bootstrap. Dalam 10 fold CV, data dibagi menjadi 10 fold berukuran kira-kira sama, sehingga kita memiliki 10 subset data untuk mengevaluasi kinerja model atau algoritma. Untuk masing-masing dari 10 subset data tersebut, CV akan menggunakan 9 fold untuk pelatihan dan 1 fold untuk pengujian seperti diilustrasikan pada Gambar 3.16.

[illegible]

Gambar 3.16. Skema 10-fold Cross Validation

[Halaman ini sengaja dikosongkan]

BAB IV IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi yang dilakukan berdasarkan rancangan yang telah dijabarkan pada bab sebelumnya. Implementasi kode program dilakukan sepenuhnya menggunakan bahasa Matlab.

4.1. Lingkungan implementasi

Spesifikasi perangkat keras dan perangkat lunak yang digunakan ditampilkan pada Tabel 4.1.

Tabel 4.1. Perangkat yang digunakan

Perangkat	Spesifikasi
Perangkat keras	Prosesor: Intel® Core™ i5-4460 CPU @ 3.20 GHz Memori: 16.00 GB
Perangkat lunak	Sistem Operasi: Windows 10 Pro Perangkat Pengembang: MATLAB R2017b Perangkat Pembantu: Microsoft Word, Microsoft Excel, VLC Player

4.2. Implementasi

Sub-bab implementasi ini menjelaskan tentang implementasi proses yang sudah dijelaskan pada bab desain perangkat lunak.

4.2.1. *Pre-processing*

Tahapan ini cukup sederhana karena hanya mengonversi frame dari *color space* RGB menjadi YCbCr. Implementasi ini dapat dilihat pada Kode Sumber 4.1.

```
1. function processed = preprocessing(image)
2.     processed = rgb2ycbcr(image);
3. end
```

Kode Sumber 4.1. *Pre-processing*

4.2.2. *Background Subtraction*

Implementasi *Background Subtraction* dapat dilihat pada Kode Sumber 4.2. Proses *Background Subtraction* menggunakan *library* dari MATLAB yaitu *vision.ForegroundDetector*. Inisialisasi parameter dari *vision.ForegroundDetector* dilakukan pada baris 1. Pemanggilan proses *vision.ForegroundDetector* dilakukan pada baris 10. Hasil dari proses *vision.ForegroundDetector* berupa *binary image* yang ditampung dalam variabel *mask*. *Binary image* tersebut masih memiliki banyak *noise*, oleh karena itu diperlukan operasi morfologi untuk mengurangi *noise*. Operasi morfologi yang digunakan adalah *open* dan *close*. Penggunaan *open* dengan *structuring element* berbentuk persegi dengan ukuran 3×3 dilakukan pada baris 12. Penggunaan *close* dengan *structuring element* berbentuk persegi dengan ukuran 20×20 dilakukan pada baris 13.

Untuk menghilangkan *noise* yang masih tersisa setelah penerapan operasi morfologi, dilakukan *fill* dan *median filter*. Penggunaan *fill* dilakukan pada baris 14, sedangkan penggunaan *median filter* dilakukan pada baris 15.

Setelah semua proses tersebut dilakukan, akan dilakukan deteksi *blob*. Deteksi *blob* dilakukan dengan *vision.Blobanalysis* yang merupakan *library* MATLAB. Inisialisasi *vision.Blobanalysis* dilakukan pada baris 4, penggunaannya berada pada baris 17. Hasil dari analisis *blob* adalah *centroid* dari objek dan *bounding boxes*-nya.

Proses dari fungsi ini memberikan tiga variabel hasil, yaitu *mask*, *centroids*, dan *bboxes*. *Mask* adalah gambar biner, *centroids* adalah *centroid* dari objek yang berada pada *mask*, dan *bboxes* adalah *bounding boxes* untuk objek yang berada pada *mask*.

```

1. obj.detector = vision.ForegroundDetector('NumGaussians
   ', 5, ...
2.           'NumTrainingFrames', 20, 'MinimumBackground
   dRatio', 0.2);
3.
4. obj.blobAnalyser = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...

```

```

5.         'AreaOutputPort', true, 'CentroidOutputPort', true, ...
6.         'MinimumBlobArea', 1200);
7. ...
8. function [centroids, bboxes, mask] = detectObjects(frame, i)
9.
10.     mask = obj.detector.step(frame);
11.
12.     mask = imopen(mask, strel('rectangle', [3,3]))
13.     ;
14.     mask = imclose(mask, strel('rectangle', [20, 20]));
15.     mask = imfill(mask, 'holes');
16.     mask = medfilt2(mask);
17.     [~, centroids, bboxes] = obj.blobAnalyser.step(mask);
18. end

```

Kode Sumber 4.2. Background Subtraction

4.2.3. Cropping Objek

Cropping objek dilakukan untuk mendapatkan objek dari gambar keseluruhan. Proses ini dilakukan dengan Kode Sumber 4.3. Baris 1 menunjukkan proses *cropping*, dan baris 2 menunjukkan proses *resize* gambar.

```

1. segmentedObject = imcrop(mask, bboxes(size(bboxes,1),:));
2. resizedImage = imresize(segmentedObject,[y_size x_size]);

```

Kode Sumber 4.3. Cropping objek

4.2.4. Pembuatan VEI

VEI diinisialisasi dengan gambar hitam, kemudian disetiap frame yang memiliki objek, akan dicari variasinya. Inisialisasi

dilakukan pada baris 1 dan pencarian variasi dilakukan pada baris 2. Implementasi ini dapat dilihat pada Kode Sumber 4.4.

```
1. vei = zeros(y_size,x_size);
2. ...
3. vei = imabsdiff(double(vei),double(resizedImage(:, :, 1)
));
```

Kode Sumber 4.4. Pembuatan VEI

4.2.5. Ekstraksi Fitur

Fitur yang akan diekstraksi adalah S , S_1, S_2, S_3, S_4, d . Kode Sumber untuk ekstraksi fitur dapat dilihat pada Kode Sumber 4.5.

```
1. d = sqrt(((finishCoor(1)-
startCoor(1)).^2+(finishCoor(2)-startCoor(2)).^2));
2.
3. S = sum(vei)/(y_size*x_size);
4. S = sum(S);
5. S1 = sum(sum(vei(1:(y_size/2),1:(x_size/2))))/((y_size
/2)*(x_size/2));
6. S2 = sum(sum(vei(1:(y_size/2),(x_size/2):x_size)))/((y
_size/2)*(x_size/2));
7. S3 = sum(sum(vei((y_size/2):y_size,1:(x_size/2))))/((y
_size/2)*(x_size/2));
8. S4 = sum(sum(vei((y_size/2):y_size,(x_size/2):x_size))
)/((y_size/2)*(x_size/2));
```

Kode Sumber 4.5. Ekstraksi fitur

4.2.6. Pelatihan *Multiclass* SVM

Proses pelatihan *multiclass* SVM dapat dilihat pada Kode Sumber abc. Baris 2 sampai 10 adalah untuk menentukan file mana yang akan dibaca untuk mendapatkan data latih dan kelasnya dan bagaimana menginterpretasikan datanya.

Baris 11 sampai 14 adalah pengaturan untuk melaksanakan pelatihan *multiclass* SVM. Setelah *multiclass* SVM dilatih, model

multiclass SVM akan disimpan ke dalam sebuah *file* bernama 'trained_model.mat'. Model ini nantinya akan digunakan untuk melakukan klasifikasi.

```

1. function trainMultiSVM()
2.     training_file = fopen('training_data.txt','r');
3.     classFile = fopen('class_data','r');
4.     formatSpec = '%f %f %f %f %f %f';
5.     formatSpec_class = '%f';
6.     semi = char(59);
7.     [data_training,count] = fscanf(training_file, [for
    matSpec semi]);
8.     [training_class,class_count] = fscanf(classFile, [
    formatSpec_class semi]);
9.
10.    data_training = reshape(data_training,[],6);
11.    t = templateSVM('Standardize',1,'KernelFunction','
    gaussian');
12.    trained_model = fitcecoc(data_training,training_cl
    ass,'Learners',t,'FitPosterior',1,...
13.        'ClassNames',{'1','2','3'},...
14.        'Verbose',2);
15.    save('trained_model.mat','trained_model');
16. end

```

Kode Sumber 4.6. Pelatihan *multiclass* SVM

4.2.7. Pengujian *Multiclass* SVM

Model *multiclass* SVM yang sudah terlatih akan dibuka pada baris 2. Fungsi ini membutuhkan enam parameter, yaitu S , S_1, S_2, S_3, S_4, d yang merupakan hasil ekstraksi fitur. Enam parameter ini akan disusun menjadi sebuah vektor dan *multiclass* SVM akan menentukan aktivitas yang sedang berlangsung. Penentuan dilakukan pada baris 4. Keseluruhan kode untuk pengujian *multiclass* SVM dapat dilihat pada Kode Sumber 4.7.

```

1. function activity = predictSVM(S,S1,S2,S3,S4,d)
2.     load('trained_model.mat');
3.     test = [S S1 S2 S3 S4 d];

```

```

4.     [activity,Score] = predict(trained_model,test);
5. end

```

Kode Sumber 4.7. Pengujian *multiclass* SVM

4.2.8. Pengujian Pengukuran Akurasi Sistem

Untuk mengukur kinerja sistem pengenalan aktivitas manusia pada video, digunakanlah metode pengukuran *10-fold cross validation* pada saat klasifikasi aktivitas dengan metode SVM *classifier*. Implementasinya dapat dilihat pada Kode Sumber 4.8.

Pada baris 2 dan 3 masing-masing menunjukkan pembacaan data dan kelasnya yang telah diekstraksi dan disimpan dalam file txt. Setelah itu akan dilakukan pembuatan dan pelatihan *multiclass* SVM. Hal ini dilakukan pada baris 7 sampai baris 11. Baris 12 akan melakukan pengukuran akurasi latihan pada SVM yang telah dibuat. Selanjutnya, kode untuk melakukan *10-fold cross validation* ditunjukkan pada baris 20. Untuk mengetahui tingkat kesalahan yang terjadi pada *10-fold cross validation*, akan dilakukan perintah *kfoldloss* seperti pada baris 18.

```

1. clear;
2. data_training = dlmread('./real_train/Combined/all_25d
   ata.txt');
3. training_class = dlmread('./real_train/Combined/all_25
   class.txt');
4. formatSpec = '%f %f %f %f %f %f';
5. formatSpec_class = '%f';
6.
7. t = templateSVM('Standardize',1,'KernelFunction','gaus
   sian');
8. trained_model = fitcecoc(data_training,training_class,
   'Learners',t,'FitPosterior',1,...
9.     'ClassNames',{ '1', '2', '3'},...
10.    'Verbose',2);
11. save('trained_model_all_25_frames.mat','trained_model'
   );
12. [label,~,score,Posterior] = resubPredict(trained_model
   , 'Verbose',1);
13. [m,n] = size(training_class);

```

```

14. idx = randsample(size(data_training,1),m,1);
15.
16. CVMdl = crossval(trained_model);
17.
18. oosLoss = kfoldLoss(CVMdl);
19.
20. [oofLabel,score] = kfoldPredict(CVMdl);
21. kfoldTable = table(training_class(idx),oofLabel(idx),score(idx,:),...
22.     'VariableNames',{'TrueLabel','PredLabel','Score'})
23. ;
24. writetable(kfoldTable,'all_25frames_5.xlsx');
25. oofLabel = cell2mat(oofLabel);
26. ConfMat = confusionmat(training_class,oofLabel);
27. fclose('all');

```

Kode Sumber 4.8. Pengukuran Akurasi Sistem

[Halaman ini sengaja dikosongkan]

BAB V

PENGUJIAN DAN EVALUASI

Pada bab ini akan dijelaskan hasil uji coba dan evaluasi program yang telah selesai diimplementasi.

5.1. Lingkungan Pengujian

Lingkungan uji coba yang akan digunakan adalah,

1. Perangkat Keras
Prosesor: Intel® Core™ i5-4460 CPU @ 3.20 GHz
Memori: 16.00 GB.
Sistem Operasi: 64-bit .
2. Perangkat Lunak
Sistem Operasi: Windows 10 Pro.
Perangkat Pengembang: MATLAB R2017a.
Perangkat pembantu: Notepad++, Microsoft Office,
Videopad video editor

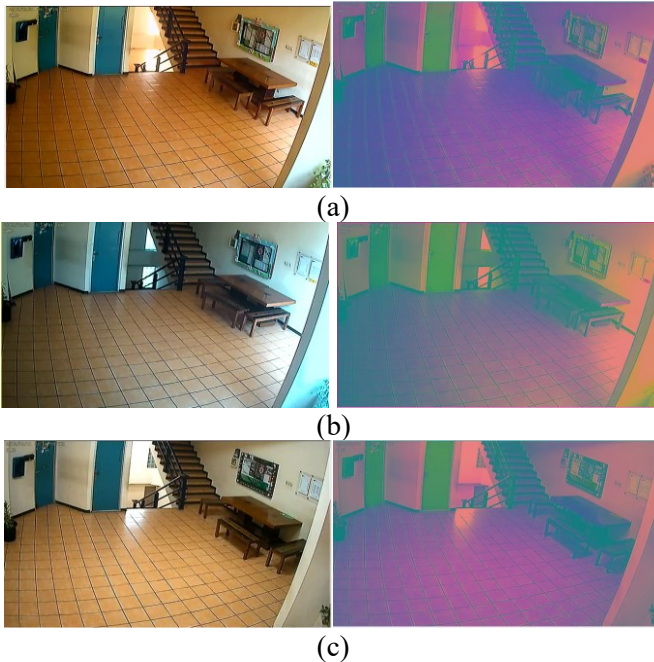
5.2. Data Uji Coba

Data yang digunakan untuk uji coba implementasi sistem pengenalan aktivitas manusia adalah hasil rekaman CCTV di Departemen Informatika lantai 3 depan ruang himpunan. Contoh data uji dapat dilihat pada Gambar 3.1. Data uji coba ini berupa video yang berjalan pada 25 *frames* per detik. Spesifikasi detail dataset dapat dilihat pada Tabel 5.1. Masing-masing peraga akan melakukan aksi berlari sebanyak enam kali dengan arah yang berbeda-beda, enam aksi berjalan sebanyak enam kali dengan arah yang berbeda-beda, dan empat kali melambai pada posisi yang berbeda-beda. Arah berlari dan berjalan dan posisi melambai dapat dilihat pada Gambar 3.2. Arah pergerakan berlari dan berjalan ditunjukkan dengan tanda panah dan posisi melambai ditunjukkan dengan titik. Pada uji coba dan pelatihan, video ini digabungkan berdasarkan peraga, aktivitas dan waktu pengambilan (pagi, siang, malam).

Tabel 5.1. Spesifikasi Dataset

Keterangan	Spesifikasi	
Ukuran resolusi	636x360 piksel	
Frame rate	25 FPS (<i>frame per second</i>)	
Ekstensi	.avi	
Jumlah video	54	
Jumlah orang yang melakukan aktivitas	6	
Kategori waktu	Pagi, siang, malam (3)	
Aktivitas yang dilakukan / orang / kategori waktu	6 walk, 6 run, 4 wave (16)	
Jumlah video / kategori waktu	18	
Jumlah video per kelas	Berlari	18
	Berjalan	18
	Melambai	18
Durasi video	Berlari	20-35 detik
	Berjalan	30-45 detik
	Melambai	10-30 detik
Ukuran file	Berlari	3000-5000 KB
	Berjalan	5000-6500 KB
	Melambai	1500-3000 KB

Data yang digunakan untuk menguji kemampuan adaptasi proses *Background Subtraction* adalah hasil rekaman CCTV di Departemen Informatika lantai 3 depan ruang himpunan juga. Namun, tidak ada peraga didalam video ini. Jadi, hanya rekaman latar belakang dengan waktu dan pencahayaan yang berbeda. Video ini diambil pada pagi hari pukul pagi pukul 08.00-10.00, siang pukul 13.00-14.00, dan malam pukul 20.00-22.00. Data ini selanjutnya akan disebut sebagai dataset *background change*. Contoh *frame* latar belakang untuk pagi hari dapat dilihat pada Gambar 5.1 (a), latar belakang untuk siang hari dapat dilihat pada Gambar 5.1 (b), dan latar belakang untuk malam hari dapat dilihat pada Gambar 5.1 (c).



Gambar 5.1. (a) Citra RGB dan YCbCr pada Pagi Hari (b) Citra RGB dan YCbCr pada Siang Hari (c) Citra RGB dan YCbCr pada Malam Hari

5.3. Skenario Uji Coba

Pada sub bab ini akan dijelaskan mengenai skenario uji coba yang telah dilakukan. Skenario uji coba yang dilakukan yaitu memvariasikan jumlah *frame* yang dikumpulkan untuk membuat VEI. Variasi yang digunakan adalah 5, 7, 15, dan 25 *frames*. Pemilihan skenario uji coba yang dilakukan pada dua faktor yaitu jumlah *frame* yang digunakan untuk membentuk VEI dan waktu pengambilan data uji. Pengukuran performa dilakukan dengan cara mencari tingkat kesalahan sistem yang diuji dengan metode *k-fold cross-validation*. Pada pengujian ini digunakan 10 *fold*.

Pada percobaan ini ada terdapat enam macam skenario uji coba, yaitu:

1. Perbandingan hasil keluaran proses *Background Subtraction* dengan masukan dengan *color space* RGB dan *color space* YCbCr.
2. Uji coba kemampuan adaptasi proses *Background Subtraction*.
3. Perbandingan performa pembentukan VEI menggunakan 5, 7, 15, dan 25 *frames* pada data video yang diambil pada saat pagi hari.
4. Perbandingan performa pembentukan VEI menggunakan 5, 7, 15, dan 25 *frames* pada data video yang diambil pada saat siang hari.
5. Perbandingan performa pembentukan VEI menggunakan 5, 7, 15, dan 25 *frames* pada data video yang diambil pada saat malam hari.
6. Perbandingan performa pembentukan VEI menggunakan 5, 7, 15, dan 25 *frames* pada data video campuran pagi, siang, dan malam hari.
7. Uji coba kecepatan sistem untuk mengenali satu gerakan pada data video secara langsung.

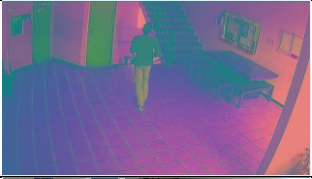
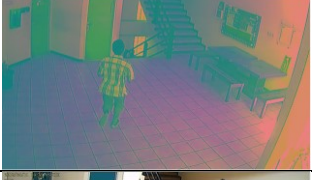
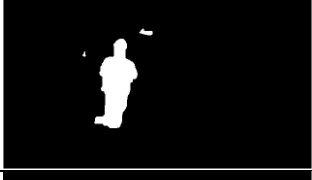
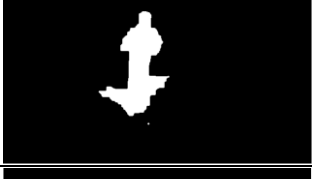
5.3.1. Perbandingan RGB dan YCbCr pada *Background Subtraction*

Uji coba ini dilakukan untuk mengetahui ukuran bayangan yang terdeteksi pada saat proses *Background Subtraction*. Hasil dapat dilihat pada Tabel 5.2. Hasil uji coba menunjukkan bahwa setelah dilakukan konversi menjadi YCbCr, bayangan yang terdeteksi menjadi berkurang.

5.3.2. Uji Coba Kemampuan Adaptasi Proses *Background Subtraction*

Uji coba dilakukan untuk mengetahui seberapa lama waktu atau berapa *frame* yang dibutuhkan untuk beradaptasi dengan perubahan yang terjadi pada latar belakang. Uji coba akan dilakukan dengan dataset *background_change*.

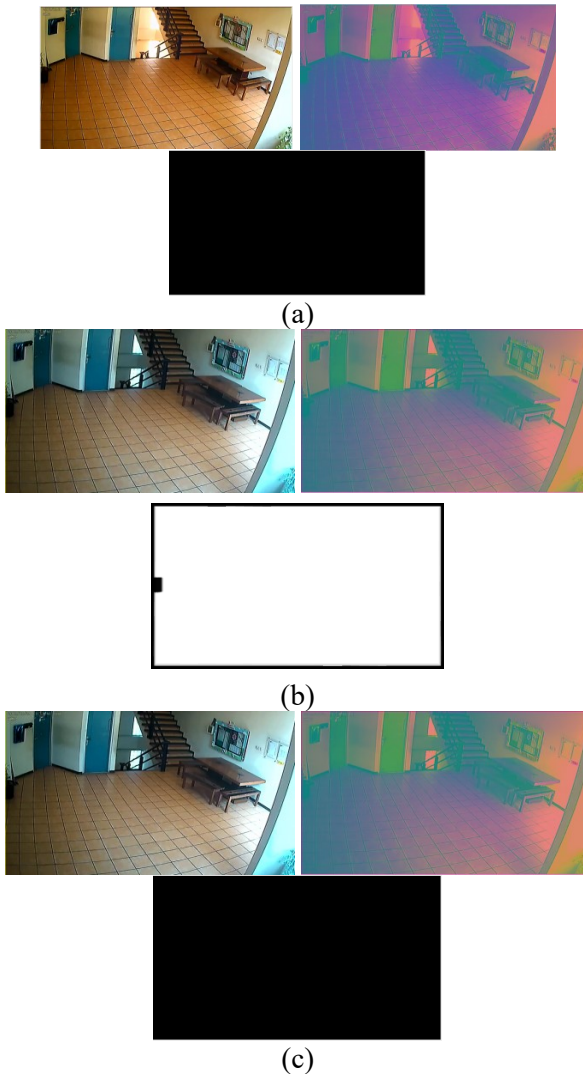
Tabel 5.2. Perbandingan RGB dan YCbCr sebagai Masukan

Waktu	Masukan	Keluaran
Pagi (RGB)		
Pagi (YCbCr)		
Siang (RGB)		
Siang (YCbCr)		
Malam (RGB)		
Malam (YCbCr)		

Tabel 5.3. Hasil Uji Coba Kemampuan Adaptasi

Sekarang (Panjang Frame)	Berikutnya	Frame untuk Beradaptasi (Frame)
Pagi	Siang	78
Siang	Malam	73
Malam	Pagi	75
Rata-rata		75,3 (3 detik)

Hasil uji coba dapat dilihat pada Tabel 5.3. Dari hasil tersebut, terlihat bahwa untuk beradaptasi dengan latar belakang dengan intensitas cahaya yang baru membutuhkan waktu sekitar 3 detik atau 75,3 *frames*. Pada uji coba ini, tidak ada pengaruh antara menggunakan citra masukan RGB dengan menggunakan citra masukan YCbCr. Hasil uji coba adaptasi dari pagi hari ke siang hari dapat dilihat pada Gambar 5.2. Pada Gambar 5.2.(a), kita dapat melihat pada saat pagi, *Background Subtraction* tidak mendeteksi *foreground* apapun. Namun, begitu memasuki *frame* siang hari, *Background Subtraction* mendeteksi perubahan drastis pada keseluruhan latar belakang akibat berubahnya pencahayaan secara keseluruhan, hal ini menyebabkan seluruh latar belakang siang hari terdeteksi menjadi *foreground* selama beberapa saat seperti yang terlihat pada Gambar 5.2.(b). Setelah beberapa 75,3 *frames*, *Background Subtraction* akan mengadaptasi dan memperbarui model latar belakang menjadi siang karena selama 75,3 *frames* tersebut tidak berubah lagi atau statis, sehingga bisa dengan aman diasumsikan sebagai latar belakang seperti yang terlihat pada Gambar 5.2.(c).



Gambar 5.2. (a) Citra RGB, Citra YCbCr, dan Hasil *Background Subtraction* pada Saat Pagi (b) Citra RGB, Citra YCbCr, dan Hasil *Background Subtraction* pada Saat Memasuki Siang (c) Citra RGB, Citra YCbCr, dan Hasil *Background Subtraction* setelah Adaptasi

5.3.3. Perbandingan Performa Saat Pagi Hari

Uji coba dilakukan untuk mengetahui performa sistem pada pagi hari. Menggunakan jumlah *frame* yang berbeda pasti akan mendapatkan jumlah VEI yang berbeda pula. Dari masing-masing enam video yang dimasukkan untuk setiap aktivitas saat pagi hari, hanya ada tiga video berlari, tiga video berjalan, dan empat video melambai yang berhasil dibentuk VEInya. Hal ini disebabkan terjadinya perubahan pencahayaan secara mendadak ditengah video dan proses pengambilan video yang berbeda waktu sehingga berbeda pencahayaan. Jumlah perolehan VEI menggunakan masing-masing *frame* dapat dilihat pada Tabel 5.4.

Tabel 5.4. Jumlah VEI yang Terbentuk pada Pagi Hari

<i>Jumlah Frames</i>	Jumlah VEI	Akurasi 10-fold (%)
5	582	70,9
7	410	92,7
15	169	89,9
25	97	87,6

Tabel 5.5. Confusion Matrix Pagi Hari

Jumlah frames pembentuk VEI	Kelas	Prediksi		
		Berlari	Berjalan	Melambai
5	Berlari	89	49	7
	Berjalan	42	120	25
	Melambai	10	37	203

7	Berlari	70	7	0
	Berjalan	7	135	5
	Melambai	2	9	175
15	Berlari	58	5	0
	Berjalan	6	32	3
	Melambai	1	2	62
25	Berlari	39	1	1
	Berjalan	4	14	2
	Melambai	1	3	32

Tabel 5.6. Rata-rata Precision dan Recall Pagi Hari

<i>Jumlah Frames</i>	<i>Aspek</i>	Aktivitas			Rata-rata(%)
		Berlari	Berjalan	Melambai	
5	Precision (%)	61,4	64,2	81,2	68,9
	Recall (%)	63,1	58,3	86,3	69,3
7	Precision (%)	90,9	91,8	94,1	92,3
	Recall (%)	88,6	89,4	97,2	91,7
15	Precision (%)	92,1	78	95,4	88,5
	Recall (%)	89,2	82,1	95,4	88,9
25	Precision (%)	95	70	88,9	84,7
	Recall (%)	88,6	77,8	91,4	85,9

Hasil uji coba pembentukan VEI dengan 7, 15, dan 25 *frames* pada pagi hari, dapat dilihat pada Tabel 5.4. Dari tabel tersebut, terlihat bahwa pembuatan VEI menggunakan 7 *frames* menghasilkan akurasi yang paling tinggi dengan akurasi 92,7%. *Confusion matrix* dapat dilihat pada Tabel 5.5. Hasil perhitungan *precision* dan *recall* dapat dilihat pada Tabel 5.6.

5.3.4. Perbandingan Performa Saat Siang Hari

Uji coba dilakukan untuk mengetahui performa sistem pada pagi hari. Menggunakan jumlah *frame* yang berbeda pasti akan mendapatkan jumlah VEI yang berbeda pula. Dari masing-masing enam video yang dimasukkan untuk setiap aktivitas saat siang hari, hanya ada tiga video berlari, empat video berjalan, dan empat video melambai yang berhasil dibentuk VEInya. Hal ini disebabkan terjadinya perubahan pencahayaan secara mendadak ditengah video dan proses pengambilan video yang berbeda waktu sehingga berbeda pencahayaan. Jumlah perolehan VEI menggunakan masing-masing *frame* dan akurasinya dapat dilihat pada Tabel 5.7. Dari tabel tersebut, terlihat bahwa pembuatan VEI menggunakan 7 *frames* menghasilkan akurasi yang paling tinggi dengan akurasi 91,8%. *Confusion matrix* pada siang hari dapat dilihat pada Tabel 5.8. Hasil perhitungan *precision* dan *recall* dapat dilihat pada Tabel 5.9.

Tabel 5.7. Jumlah VEI yang Terbentuk pada Siang Hari

<i>Jumlah Frames</i>	Jumlah VEI	Akurasi 10-fold (%)
5	567	79,4
7	437	91,8
15	143	90,2
25	93	89,2

Tabel 5.8. Confusion Matrix pada Siang Hari

Jumlah frames pembentuk VEI	Kelas	Prediksi		
		Berlari	Berjalan	Melambai
5	Berlari	60	57	2
	Berjalan	31	171	10
	Melambai	2	15	219
7	Berlari	65	14	0
	Berjalan	13	146	4
	Melambai	0	5	190
15	Berlari	58	4	0
	Berjalan	6	15	2
	Melambai	1	1	56
25	Berlari	49	4	0
	Berjalan	2	9	3
	Melambai	0	1	25

Tabel 5.9. Rata-rata Precision dan Recall Siang

Jumlah Frames	Aspek	Aktivitas			Rata- rata(%)
		Berlari	Berjalan	Melambai	
5	Precision (%)	50,4	80,7	92,8	74,6
	Recall (%)	86,5	80,7	81,2	83

7	Precision (%)	82,3	89,6	97,4	89,8
	Recall (%)	83,3	88,5	97,9	89,9
15	Precision (%)	93,5	65,2	96,5	85,1
	Recall (%)	89,2	75	96,5	86,9
25	Precision (%)	92,5	64,3	96,2	84,3
	Recall (%)	96,1	64,3	89,3	83,2

5.3.5. Perbandingan Performa Saat Malam Hari

Uji coba dilakukan untuk mengetahui performa sistem pada pagi hari. Menggunakan jumlah *frame* yang berbeda pasti akan mendapatkan jumlah VEI yang berbeda pula. Dari masing-masing enam video yang dimasukkan untuk setiap aktivitas saat malam hari, hanya ada tiga video berlari, empat video berjalan, dan empat video melambai yang berhasil dibentuk VEInya. Jumlah perolehan VEI menggunakan masing-masing *frame* dapat dilihat pada Tabel 5.10. Dari tabel tersebut, terlihat bahwa pembuatan VEI menggunakan 7 *frames* menghasilkan akurasi yang paling tinggi dengan akurasi 93,4%. *Confusion matix* pada malam hari dapat dilihat pada Tabel 5.11. Hasil perhitungan *precision* dan *recall* dapat dilihat pada Tabel 5.12.

Tabel 5.10. Jumlah VEI yang Terbentuk pada Malam Hari

<i>Jumlah Frames</i>	Jumlah VEI	Akurasi 10-fold (%)
5	1387	84,7
7	987	93,4
15	435	89,4
25	252	88,9

Tabel 5.11. *Confusion Matrix* pada Malam Hari

Jumlah frames pembentuk VEI	Kelas	Prediksi		
		Berlari	Berjalan	Melambai
5	Berlari	89	49	7
	Berjalan	42	120	25
	Melambai	10	37	203
7	Berlari	57	14	0
	Berjalan	7	464	20
	Melambai	0	24	401
15	Berlari	125	9	0
	Berjalan	10	152	14
	Melambai	1	12	112
25	Berlari	79	5	0
	Berjalan	8	84	5
	Melambai	0	10	61

Tabel 5.12. Rata-rata Precision dan Recall Malam Hari

Jumlah Frames	Aspek	Aktivitas			Rata- rata(%)
		Berlari	Berjalan	Melambai	
5	Precision (%)	17,6	90,8	91,1	66,5
	Recall (%)	86,5	80,7	81,2	83
7	Precision (%)	80,1	94,5	94,4	89,7

	Recall (%)	89,1	92,4	95,2	92,2
15	Precision (%)	93,3	86,4	89,6	89,8
	Recall (%)	91,9	87,8	88,9	89,6
25	Precision (%)	94	86,6	85,9	88,8
	Recall (%)	90,8	84,8	92,4	89,4

5.3.6. Perbandingan Performa Saat Pagi, Siang dan Malam Hari

Uji coba dilakukan untuk mengetahui performa sistem pada pagi hari. Menggunakan jumlah *frame* yang berbeda pasti akan mendapatkan jumlah VEI yang berbeda pula. Jumlah perolehan VEI menggunakan masing-masing *frame* dapat dilihat pada Tabel 5.13.

Tabel 5.13. Jumlah VEI yang Terbentuk

Jumlah Frames	Jumlah VEI	Akurasi 10-fold (%)
5	2535	78,7
7	1834	95,8
15	747	92,4
25	442	92,3

Dari tabel tersebut, terlihat bahwa pembuatan VEI menggunakan 7 *frames* menghasilkan akurasi yang paling tinggi dengan akurasi 95,8%. *Confusion matrix* untuk data campuran ini dapat dilihat pada Tabel 5.14. Hasil perhitungan *precision* dan *recall* dapat dilihat pada Tabel 5.15.

Tabel 5.14. *Confusion Matrix*

Jumlah <i>frames</i> pembentuk VEI	Kelas	Prediksi		
		Berlari	Berjalan	Melambai
5	Berlari	163	204	79
	Berjalan	61	841	104
	Melambai	8	84	991
7	Berlari	216	11	0
	Berjalan	13	761	27
	Melambai	1	25	780
15	Berlari	251	8	0
	Berjalan	11	208	21
	Melambai	1	16	231
25	Berlari	170	8	0
	Berjalan	9	116	6
	Melambai	0	11	122

5.3.7. Uji Coba Kecepatan Sistem untuk Mengenali Satu Gerakan pada Data Video

Uji coba ini dilakukan untuk mengetahui waktu yang diperlukan sistem untuk mengenali sebuah gerakan. Uji coba dilakukan dengan menggunakan fungsi MATLAB *cputime*. Fungsi ini akan menampilkan total waktu *cpu* yang digunakan oleh MATLAB. Sehingga, waktu awal akan dicatat saat akan melakukan ekstraksi fitur dari VEI yang telah terbentuk, kemudian waktu akhir akan dicatat saat sistem telah mengeluarkan hasil prediksi. Kecepatan sistem dalam

mengenali satu gerakan akan dihasilkan dari pengurangan waktu akhir dengan waktu awal. Hasilnya dapat dilihat pada Tabel 5.16. Percobaan dilakukan sebanyak lima kali dengan rata-rata waktu yang diperlukan untuk mengenali satu gerakan adalah 0,028125 detik.

Tabel 5.15. Rata-rata Precision dan Recall Pagi, Siang dan Malam

Jumlah Frames	Aspek	Aktivitas			Rata-rata(%)
		Berlari	Berjalan	Melambai	
5	Precision (%)	36,5	83,6	91,5	70,6
	Recall (%)	70,3	74,5	84,4	76,4
7	Precision (%)	95,2	95	96,8	95,7
	Recall (%)	93,9	95,5	96,7	95,4
15	Precision (%)	96,9	86,7	93,2	92,2
	Recall (%)	95,4	89,7	91,7	92,3
25	Precision (%)	95,5	88,6	91,7	91,9
	Recall (%)	94,9	85,9	93,3	92,1

Tabel 5.16. Hasil Uji Coba Kecepatan Sistem

Percobaan	CPU Time (detik)	Rata-rata (detik)
1	0,015625	0,028125
2	0,015625	
3	0,031250	
4	0,015625	
5	0,031250	

5.4. Analisis Hasil Uji Coba

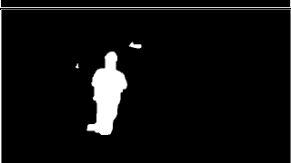
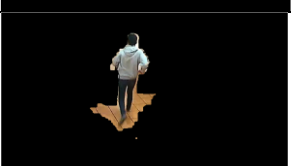
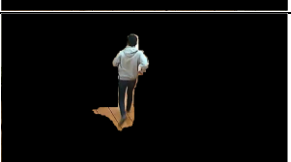
Dari uji coba *Background Subtraction*, terbukti bahwa penggunaan *color space* YCbCr memberikan hasil yang lebih baik dari pada *color space* RGB. *Color space* YCbCr mampu mengurangi bayangan yang terdeteksi tanpa mempengaruhi bentuk objek yang akan diteliti. Hal ini dapat dilihat pada Tabel 5.17 yang menunjukkan bentuk objek pada saat dilakukan *masking* antara *foreground binary mask* terhadap citra aslinya.

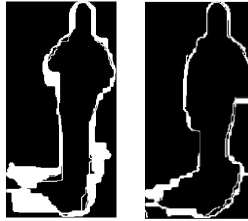
Pada saat terjadi perubahan pencahayaan secara tiba-tiba, *Background Subtraction* dapat beradaptasi hanya dengan rata-rata 75 *frames* atau 3 detik. *Background Subtraction* akan bekerja secara sempurna apabila perubahan pencahayaan terjadi secara berkala. Perubahan pencahayaan secara mendadak mempengaruhi hasil segmentasi objek, kemudian hasil segmentasi objek akan mempengaruhi hasil pembentukan VEI. Lalu VEI yang dihasilkan akan mempengaruhi kinerja sistem dalam mengenali aktivitas tersebut. Hal ini membuat sistem harus berhenti mengenali aktivitas selama 3 detik saat terjadi perubahan pencahayaan secara mendadak. Setelah 3 detik tersebut, baru sistem akan melanjutkan pengenalan aktivitas kembali.

Dari keempat uji coba yang telah dilakukan, diketahui bahwa pembuatan VEI dengan 7 *frames* adalah yang terbaik pada pagi, siang, malam dan gabungan pagi, siang, dan malam. Akurasinya mencapai 92,7% pada pagi hari, 91,7% pada siang hari, 93,4% pada malam hari, dan 95,8% pada keseluruhan data. Dari uji coba tersebut, masih terjadi misklasifikasi antara berjalan dengan berlari, berlari dengan berjalan, dan berjalan dengan melambai. Terjadinya misklasifikasi antara berjalan dan berlari dikarenakan pergerakan kuadran 1 dan kuadran 2 yang mirip, sedangkan kuadran 3 dan kuadran 4 yang seharusnya berbeda namun menjadi sama karena adanya pengaruh bayangan. VEI berlari dan berjalan dapat dilihat pada Gambar 5.3. Terjadinya misklasifikasi antara berjalan atau berlari dengan melambai adalah karena adanya pantulan objek pada latar belakang (lantai atau pintu) yang mengakibatkan terjadinya keanehan pada saat perhitungan fitur bentuk. Contoh kesalahan ini dapat dilihat pada Gambar 5.4. Pada

Gambar 5.4.(c) dapat dilihat pada kuadran 1 dan 2 memiliki banyak pergerakan padahal aktivitasnya adalah berjalan.

Tabel 5.17. Hasil *Masking Foreground Binary Mask Terhadap Citra Asli*

<i>Foreground Binary Mask</i>		Hasil <i>Masking</i>	
			
			
			
			
			
			



(a)

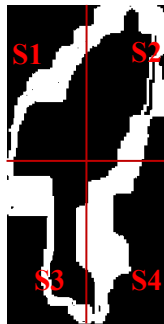
(b)

Gambar 5.3. (a) VEI berlari (b) VEI berjalan

(a)



(b)



(c)

Gambar 5.4. (a) Citra masukan (b) Siluet (c) VEI

Berdasarkan hasil uji coba diatas, waktu pengambilan pada malam hari memberikan hasil yang paling bagus karena memiliki pencahayaan yang stabil dibandingkan dengan pagi dan siang hari yang pencahayaannya tergantung cuaca.

Dari uji coba kecepatan sistem dalam mengenali suatu aktivitas memiliki hasil yang memuaskan, dengan rata-rata waktu 0,028125

detik, paling cepat 0,015625 detik dan paling lama 0,031250 detik. Waktu diukur mulai dari ekstraksi fitur sampai pada selesai klasifikasi. Hasil ini membuat sistem ini dapat diterapkan secara *real-time* pada skenario uji coba berikutnya.

BAB VI

KESIMPULAN DAN SARAN

Bab ini membahas tentang kesimpulan yang didasari oleh hasil uji coba yang telah dilakukan pada bab sebelumnya. Kesimpulan nantinya sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, juga terdapat saran yang ditujukan untuk pengembangan penelitian lebih lanjut di masa depan.

6.1. Kesimpulan

Dari hasil uji coba yang telah dilakukan, dapat diambil kesimpulan sebagai berikut :

1. *Background Subtraction* dapat digunakan sebagai metode untuk melakukan segmentasi objek karena teknik ini sangat cocok dengan kondisi video yang diambil dari kamera CCTV yang statis.
2. Berdasarkan uji coba, pengubahan *color space* citra inputan dari RGB menjadi YCbCr dapat mengurangi bayangan yang terdeteksi namun tidak bisa menghilangkan bayangan secara keseluruhan.
3. Berdasarkan uji coba yang telah dilakukan, sistem ini mampu beradaptasi secara langsung dengan perubahan cahaya yang terjadi secara berkala namun tidak dapat beradaptasi secara langsung perubahan cahaya yang terjadi secara mendadak. Pada perubahan pencahayaan yang mendadak dapat diadaptasi apabila ada minimal *75 frames* atau 3 detik tersedia setelah terjadi perubahan pencahayaan.
4. Berdasarkan uji coba yang telah dilakukan, jumlah *frame* yang digunakan untuk membentuk VEI yang paling baik dalam merepresentasikan aktivitas adalah sepanjang *7 frames* untuk dataset CCTV di Departemen Informatika ITS Lantai 3 depan ruang himpunan. Pembentukan VEI menggunakan *7 frames* menghasilkan akurasi 92,7% pada pagi hari, 91,8% pada siang hari, 93,4% pada malam hari dan 95,8% pada data campuran

pagi, siang dan malam hari. Uji coba ini dilakukan menggunakan *10-fold cross validation*.

5. Berdasarkan uji coba yang telah dilakukan, waktu terbaik pengambilan data untuk mencapai akurasi tertinggi adalah saat malam hari karena intensitas cahaya pada waktu itu cenderung stabil oleh penerangan lampu.
6. Waktu yang dibutuhkan sistem untuk mengekstraksi fitur, kemudian melakukan klasifikasi rata-rata 0,028125 detik, paling cepat 0,015625 detik dan paling lama 0,031250 detik.

6.2. Saran

Saran yang diberikan untuk pengembangan perangkat lunak ini adalah:

1. Mengimplementasikan sistem secara *realtime*.
2. Melakukan uji coba terhadap dataset yang beragam.
3. Pengembangan sistem untuk pengenalan aktivitas lebih dari satu manusia.
4. Sebaiknya pengembangan sistem dilakukan dengan memperbanyak jenis aktivitas yang dapat dikenali.

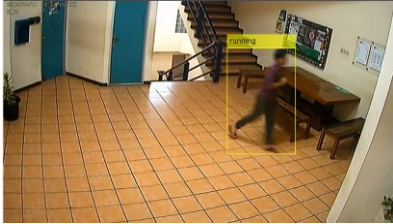
DAFTAR PUSTAKA

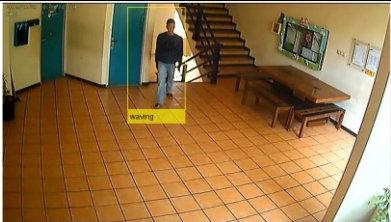
- [1] X. Haihui dan Q. Wu, "Human Action Recognition based on Variation Energy Images Features," *11th International Conference on Natural Computation (ICNC)*, 2015.
- [2] H. W. dan C. S. , "Action Recognition with Improved," *IEEE International Conference on Computer Vision*, 2013.
- [3] S. J. W. X. dan M. Y. , "3D Convolutional Neural Networks," *IEEE Transactions on Pattern Analysis*, vol. 35, no. 1, 2013.
- [4] M. Hassan, T. Ahmad dan . M. A. Javaid, "Human Activity Recognition using Motion History," *International Journal of Scientific & Engineering Research*, vol. 5, no. 8, 2014.
- [5] M. Piccardi, "Background subtraction techniques: a review," *IEEE International Conference on Systems, Man and Cybernetics*, pp. 3099-3104, 2004.
- [6] P. W. Power dan J. A. Schoonees, "Understanding Background Mixture Models for Foreground," dalam *Proceedings Image and Vision Computing*, New Zealand, Industrial Research Limited, 2002, p. 267.
- [7] S. Rostianingsih, R. Adipranata dan F. S. Wibisono, "Adaptive Background Dengan Metode Gaussian Mixture".
- [8] R. Munir, "Citra Biner," dalam *Pengolahan Citra Digital*, Bandung, Penerbit Informatika Bandung, pp. 167-182.
- [9] Z. Zivkovic, "Improved Adaptive Gaussian Mixture Model for Background," dalam *Pattern Recognition, ICPR Proceedings of the 17th International Conference* .
- [10] Eddy Nurraharjo, "Implementasi Morphology Concept and Technique dalam Pengolahan Citra ," *Jurnal Teknologi Informasi DINAMIK*, vol. 16, no. 2, pp. 134-138, 2011.

- [11] Anonymous, “imcrop,” MATLAB, [Online]. Available: <https://www.mathworks.com/help/images/ref/imcrop.html>. [Diakses 25 June 2018].
- [12] M. Stork, “Median Filters Theory and Applications,” University of West Bohemia.
- [13] Anonymous, “medfilt2,” MATLAB, [Online]. Available: <https://www.mathworks.com/help/images/ref/medfilt2.html>. [Diakses 25 June 2018].
- [14] R. T. Faihah, “support vector machine svm,” [Online]. Available: <https://iefha.wordpress.com/2010/11/27/support-vector-machine-svm/>.
- [15] A. M. S, “Multi Class SVM,” January 2018. [Online]. Available: <https://www.mathworks.com/matlabcentral/fileexchange/62061-multi-class-svm>.
- [16] A. S. Nugroho, “Pengantar Support Vector Machine,” 8 February 2007.
- [17] C. Campbell and Y. Ying, "Learning with Support Vector Machines," Morgan & Claypool, 2018.

LAMPIRAN

1. Sampel Hasil Pengenalan

Masukan	VEI	Keluaran	<i>Truth</i>	Hasil
		 (berlari)	Berlari	Benar
		 (berjalan)	Berlari	Salah

			Berjalan	Benar
			Berjalan	Salah

(berjalan)

(melambai)

		 (melambai)	Melambai	Benar
---	---	--	----------	-------

2. Panjang Data Video

Waktu	Peraga	Aktivitas	Durasi (detik)	Jumlah <i>Frames</i>
Pagi	1	Berlari	21	525
		Berjalan	42	1050
		Melambai	26	650
	2	Berlari	23	575
		Berjalan	36	900
		Melambai	34	850
	3	Berlari	25	625
		Berjalan	34	850
		Melambai	24	600
	4	Berlari	22	550
		Berjalan	40	1000
		Melambai	19	475
	5	Berlari	25	625
		Berjalan	37	925
		Melambai	22	550
	6	Berlari	31	775
		Berjalan	34	850
		Melambai	24	600
Siang	1	Berlari	18	450
		Berjalan	38	950
		Melambai	24	600
	2	Berlari	17	425
		Berjalan	37	925
		Melambai	24	600
	3	Berlari	23	575
		Berjalan	27	675

	4	Melambai	20	500
		Berlari	19	475
		Berjalan	28	700
		Melambai	24	600
	5	Berlari	26	650
		Berjalan	37	925
		Melambai	28	700
	6	Berlari	19	475
		Berjalan	30	750
		Melambai	24	600
Malam	1	Berlari	35	875
		Berjalan	46	1150
		Melambai	14	350
	2	Berlari	22	550
		Berjalan	37	925
		Melambai	15	375
	3	Berlari	27	675
		Berjalan	42	1050
		Melambai	13	325
	4	Berlari	31	775
		Berjalan	47	1175
		Melambai	15	375
	5	Berlari	33	825
		Berjalan	43	1075
		Melambai	14	350
	6	Berlari	29	725
		Berjalan	43	1075
		Melambai	22	550

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS



Fintanto Cendikia atau biasa dipanggil Tanto dilahirkan di Medan pada tanggal 12 Agustus 1997 dan dibesarkan di Medan. Penulis adalah anak kedua dari empat bersaudara. Penulis menempuh pendidikan di SD Brigjend Katamso Medan (2003-2009) SMP Santo Thomas 1 Medan (2009-2011), dan SMA Santo Thomas 1 Medan (2011-2014). Setelah lulus SMA penulis melanjutkan ke jenjang perkuliahan di Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya. Bidang Studi yang diambil oleh penulis pada saat kuliah di Teknik Informatika ITS adalah Komputasi Cerdas dan Visi. Selama menempuh kuliah penulis aktif sebagai anggota Himpunan Mahasiswa Teknik Computer (HMTTC) ITS pada departemen kaderisasi dan pemetaan . Penulis dapat dihubungi melalui alamat *email* fincendi@gmail.com.