[W

;/--n

**TUGAS AKHIR – KI141502**

**PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGGUNAKAN FITUR MATRIKS KOVARIAN**

**RINA WIJAYA KUSUMA WARDHANI**

**NRP** **05111440000021**

**Dosen Pembimbing I**

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

**Dosen Pembimbing II**

**Dini Adni Navastara, S.Kom., M.Sc.**

**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 FITUR MATRIKS KOVARIAN**

**RINA WIJAYA KUSUMA WARDHANI**

**NRP 05111440000021**

**Dosen Pembimbing I**

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

**Dosen Pembimbing II**

**Dini Adni Navastara, S.Kom., M.Sc.**

**DEPARTEMEN INFORMATIKA**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2018**

****

**UNDERGRADUATE THESES – KI141502**

**HUMAN ACTIVITY RECOGNITION IN VIDEO USING FEATURE COVARIANCE MATRICES**

**RINA WIJAYA KUSUMA WARDHANI**

**NRP 05111440000021**

**Supervisor I**

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

**Supervisor II**

**Dini Adni Navastara, S.Kom., M.Sc.**

**DEPARTMENT OF INFORMATICS**

**Faculty of Information and Communication Technology**

**Sepuluh Nopember Institute of Technology**

**Surabaya 2018**

***[Halaman ini sengaja dikosongkan]***

# LEMBAR PENGESAHAN

**PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGGUNAKAN FITUR MATRIKS KOVARIAN**

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer

pada

Bidang Studi Komputasi Cerdas dan Visi

Program Studi S-1 Jurusan Teknik Informatika

Fakultas Teknologi Informasi dan Komunikasi

Institut Teknologi Sepuluh Nopember

Oleh

**RINA WIJAYA KUSUMA WARDHANI**

**NRP : 05111440000021**

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Dr. Eng. Nanik Suciati, S.Kom., M.Kom. .....................

NIP: 197906262005012002 (Pembimbing 1)

1. Dini Adni Navastara, S.Kom., M.Sc. .....................

NIP: 198701032014041001 (Pembimbing 2)

**SURABAYA**

**JANUARI, 2018**

***[Halaman ini sengaja dikosongkan]***

**PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGGUNAKAN FITUR MATRIKS KOVARIAN**

|  |  |  |
| --- | --- | --- |
| **Nama Mahasiswa** | **:** | **RINA WIJAYA KUSUMA WARDHANI** |
| **NRP** | **:** | **05111440000021** |
| **Jurusan** | **:** | **Informatika FTIK-ITS** |
| **Dosen Pembimbing 1** | **:** | **Dr. Eng. Nanik Suciati, S.Kom., M.Kom.** |
| **Dosen Pembimbing 2** | **:** | **Dini Adni Navastara, S.Kom., M.Sc.** |

# Abstrak

Saat ini, pengenalan aktivitas manusia pada data video merupakan persoalan yang cukup menantang di bidang visi komputer. Beberapa alat yang dapat digunakan untuk membantu pengenalan aktivitas manusia antara lain sensor *Accelerometer, Gyroscope, Camera,* dan GPS. Beberapa pengaplikasian pengenalan aktivitas manusia digunakan dalam sistem keamanan atau sarana hiburan.

Pada tugas akhir ini akan dilakukan pengembangan sistem pengenalan aktivitas manusia pada video yang berasal dari *Closed-Circuit Television* (CCTV) di Departemen Informatika ITS Surabaya. Sistem ini menggunakan fitur matriks kovarian dari fitur *optical flow* dimana fitur lokal ini dapat menangkap gerakan dinamis sebagai karakteristik dari aktivitas manusia pada video. Pengunaan fitur kovarian dilakukan untuk mengurangi jumlah *bag of feature vector* hasil dari ekstraksi fitur *optical flow* menjadi jauh lebih kecil namun tetap dapat merepresentasikan aktivitas manusia. Klasifikasi yang digunakan dalam sistem ini adalah *Nearest Neigbour Classifier*. Secara umum, pengembangan sistem ini dilakukan dalam enam tahapan proses yakni *preprocessing*, pengumpulan fitur *optical flow*, perhitungan matriks kovarian, perhitungan matriks log kovarian, pembentukan model klasifikasi, dan yang terakhir klasifikasi jenis aktivitas.

Uji coba menggunakan dataset CCTV menunjukkan bahwa metode yang digunakan pada tugas akhir ini menghasilkan akurasi terbaik sebesar % serta masing-masing hasil recall dan precision pada tiap kelas aktivitas sebesar % dan % untuk aktivitas berlari, % dan % untuk aktivitas berjalan, serta % dan % untuk aktivitas melambaikan kedua tangan.

***Kata kunci:*** ***Pengenalan Aktivitas Manusia, Video, CCTV, Optical Flow, Matriks Kovarian.***

**HUMAN ACTIVITY RECOGNITION IN VIDEO USING FEATURE COVARIANCE MATRICES**

|  |  |  |
| --- | --- | --- |
| **Student’s Name** | **:** | **RINA WIJAYA KUSUMA WARDHANI** |
| **Student’s ID** | **:** | **05111440000021** |
| **Department** | **:** | **Informatika FTIK-ITS** |
| **First Advisor** | **:** | **Dr. Eng. Nanik Suciati, S.Kom., M.Kom.** |
| **Second Advisor** | **:** | **Dini Adni Navastara, S.Kom., M.Sc.** |

# *Abstract*

*Currently, human recognition activities in video data are quite challenging in the field of computer vision. Some tools that can be used to help human recognition activities include Accelerometer, Gyroscope, Camera, and GPS. Many applications of human recognition activities in media systems or entertainment facilities.*

*In this final project, the development of human activity recognition system will be implemented on video from Closed-Circuit Television (CCTV) in Informatics Department ITS Surabaya. This system uses the covariance matrix feature of the optical flow feature which is the local features that can be used in the human video process. The use of covariance matrix features is done to reduce the number of bags of feature vector so it becomes much smaller but still can represent human activity. The classification used within this system is the Nearest Neigbour Classification. In general, the development of this system is done in the process of pre-processing, optical flow flow, matrix calculation, covarian log matrix calculation, model formation, and activity classification.*

*Trials using the CCTV dataset shows that the methods used in this final project has best accuration of % which has the result of recall and precision in each activity class as % and % for running, % and % for walking, and % and % for waving both hands.*

***Keywords: Human Activity Recognition, Video, CCTV, Optical Flow, Covarian Matrix.***

# KATA PENGANTAR



Alhamdulillahirabbil’alamin, segala puji bagi Allah SWT, yang atas izin dan karunianya penulis dapat menyelesaikan Tugas Akhir yang berjudul ***“Pengenalan Aktivitas Manusia pada Video Menggunakan Fitur Matriks Kovarian”*.**

Dalam pengerjaan tugas akhir ini, penulis mendapatkan banyak sekali ilmu baru dan memperdalam ilmu-ilmu yang sebelumnya telah diajarkan selama masa perkuliahan di Teknik Informatika ITS.

Terselesaikannya Tugas Akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada:

1. Allah SWT dan Nabi Muhammad SAW.
2. Orang tua penulis Bapak Hadi Subito dan Ibu Trining Hastuti yang telah memberikan dukungan moral, spiritual dan material serta senantiasa memberikan doa demi kelancaran dan kemudahan penulis dalam mengerjakan tugas akhir.
3. Dr. Eng. Nanik Suciati, S.Kom., M.Kom. dan Dini Adni Navastara, S.Kom., M.Sc. selaku dosen pembimbing penulis yang telah memberi ide, nasihat dan arahan sehingga penulis dapat menyelesaikan tugas akhir dengan tepat waktu.
4. Keempat saudara kandung penulis Mbak Rani, Mbak Nindy, Rini, dan Hito serta seluruh keluarga besar yang telah memberikan dukungan besar baik secara langsung maupun secara implisit.
5. Mas Ilham Gurat Adillion selaku pembimbing ketiga yang telah memberikan banyak pencerahan juga nasihat kepada penulis serta menjadikan penulis bahan uji coba pelatihan menjadi dosen pembimbing sesungguhnya.
6. Rafiar Rahmansyah dan Ali Davito Hady yang telah menyemangati dan menghibur selama pengerjaan tugas akhir.
7. Teman-teman admin Laboratorium Pemrograman yang telah menemani dan memberikan semangat kepada penulis dalam menyelesaikan tugas akhir ini di Laboratorium Pemrograman.
8. Teman-teman LayToLele seperti Maya, Vina, Vennia, Sukma, Fitri, dan Dewayu
9. Pihak-pihak lain yang tidak bisa penulis sebutkan satupersatu. Penulis menyadari masih ada kekurangan dalam penyusunan tugas akhir ini.

Penulis mohon maaf atas kesalahan, kelalaian maupun kekurangan dalam penyusunan tugas akhir ini. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan ke depan.

Surabaya, Juni 2018

Penulis

# DAFTAR ISI

[LEMBAR PENGESAHAN v](#_Toc516086480)

[Abstrak vii](#_Toc516086481)

[*Abstract* ix](#_Toc516086482)

[KATA PENGANTAR xi](#_Toc516086483)

[DAFTAR ISI xiii](#_Toc516086484)

[DAFTAR GAMBAR xvi](#_Toc516086485)

[DAFTAR TABEL xvii](#_Toc516086486)

[DAFTAR KODE SUMBER xviii](#_Toc516086487)

[BAB I PENDAHULUAN 1](#_Toc516086488)

[1.1 Latar Belakang 1](#_Toc516086489)

[1.2 Rumusan Masalah 2](#_Toc516086490)

[1.3 Batasan Masalah 3](#_Toc516086491)

[1.4 Tujuan 3](#_Toc516086492)

[1.5 Manfaat 3](#_Toc516086493)

[1.6 Metodologi 3](#_Toc516086494)

[1.6.1 Penyusunan Proposal Tugas Akhir 3](#_Toc516086495)

[1.6.2 Studi Literatur 4](#_Toc516086496)

[1.6.3 Analisis dan Desain Perangkat Lunak 4](#_Toc516086497)

[1.6.4 Implementasi Perangkat Lunak 4](#_Toc516086498)

[1.6.5 Pengujian dan Evaluasi 5](#_Toc516086499)

[1.7 Sistematika Penulisan Laporan Tugas Akhir 6](#_Toc516086500)

[BAB II DASAR TEORI 9](#_Toc516086501)

[2.1 Pengenalan Aktivitas Manusia 9](#_Toc516086503)

[2.2 *Optical Flow* 10](#_Toc516086504)

[2.3 Matriks Kovarian 13](#_Toc516086505)

[2.4 Normalisasi 14](#_Toc516086506)

[2.5 Matriks Log-Kovarian 14](#_Toc516086507)

[2.6 *Nearest-Neighbor* (NN) *Classification* 15](#_Toc516086508)

[2.7 *Confusion Matrix* 15](#_Toc516086509)

[BAB III PERANCANGAN PERANGKAT LUNAK 19](#_Toc516086510)

[3.1 Lingkungan Desain dan Implementasi 19](#_Toc516086513)

[3.2 Perancangan Sistem 19](#_Toc516086514)

[3.3 Perancangan Data 22](#_Toc516086515)

[3.4 Perancangan Proses 25](#_Toc516086516)

[3.4.1 *Preprocessing* 25](#_Toc516086517)

[3.4.2 Pengumpulan Fitur *Optical Flow* 26](#_Toc516086518)

[3.4.3 Perhitungan Matriks Kovarian 28](#_Toc516086519)

[3.4.4 Perhitungan Matriks Log Kovarian 29](#_Toc516086520)

[3.4.5 Pembuatan Model NN *Classifier* 31](#_Toc516086521)

[3.4.6 Klasifikasi Akivitas 31](#_Toc516086522)

[BAB IV IMPLEMENTASI 33](#_Toc516086523)

[4.1 Lingkungan Implementasi 33](#_Toc516086525)

[4.1.1 Perangkat Keras 33](#_Toc516086526)

[4.1.2 Perangkat Lunak 33](#_Toc516086527)

[4.2 Implementasi Preprocessing 33](#_Toc516086528)

[4.3 Implementasi Pengumpulan Fitur *Optical Flow* 34](#_Toc516086529)

[4.3.1 Pengambilan Fitur 34](#_Toc516086530)

[4.3.2 Inisialisasi 35](#_Toc516086531)

[4.3.3 Perhitungan Fitur *It , u, dan v* 36](#_Toc516086532)

[4.3.4 Perhitungan Fitur *ut* dan *vt* 36](#_Toc516086533)

[4.3.5 Perhitungan Fitur *Div, Vor, Gten,* dan *Sten* 37](#_Toc516086534)

[4.3.6 Penggabungan 12 Fitur Menjadi Matriks Fn 38](#_Toc516086535)

[4.3.7 Normalisasi 39](#_Toc516086536)

[4.4 Implementasi Perhitungan Matriks Kovarian 39](#_Toc516086537)

[4.5 Implementasi Perhitungan Matriks Log Kovarian 40](#_Toc516086538)

[4.6 Implementasi Pembuatan Model NN *Classifier* 41](#_Toc516086539)

[4.7 Implementasi Klasifikasi Aktivitas 41](#_Toc516086540)

[4.8 Implementasi Pengukuran Kinerja Sistem 41](#_Toc516086541)

[BAB V UJI COBA DAN EVALUASI 43](#_Toc516086542)

[5.1 Lingkungan Uji Coba 43](#_Toc516086543)

[5.2 Data Uji Coba 43](#_Toc516086544)

[5.3 Skenario Uji Coba 44](#_Toc516086545)

[5.4 Skenario Uji Coba pada Dataset Weizmann 44](#_Toc516086546)

[5.5 Skenario Uji Coba pada Dataset CCTV 46](#_Toc516086547)

[5.4.1 Skenario Uji Coba Panjang Frame L 46](#_Toc516086548)

[5.4.2 Skenario Uji Coba Waktu Pengambilan Data 47](#_Toc516086549)

[5.6 Evaluasi Uji Coba pada Dataset Weizmann 47](#_Toc516086550)

[5.7 Evaluasi Uji Coba pada Dataset CCTV 47](#_Toc516086551)

[BAB VI KESIMPULAN DAN SARAN 49](#_Toc516086552)

[6.1 Kesimpulan 49](#_Toc516086554)

[5.1 Saran 49](#_Toc516086555)

[DAFTAR PUSTAKA 51](#_Toc516086556)

[BIODATA PENULIS 53](#_Toc516086557)

***[Halaman ini sengaja dikosongkan]***

# DAFTAR GAMBAR

***[Halaman ini sengaja dikosongkan]***

# DAFTAR TABEL

***[Halaman ini sengaja dikosongkan]***

# DAFTAR KODE SUMBER

***[Halaman ini sengaja dikosongkan]***

# BAB I PENDAHULUAN

## Latar Belakang

Saat ini, pengenalan aktivitas manusia pada data video merupakan persoalan yang cukup menantang di bidang visi komputer. 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 [1]. Pengenalan aktivitas manusia pada data video telah diterapkan di beberapa bidang serta digunakan sebagai sistem keamanan atau sarana hiburan. Untuk itulah penulis ingin menerapkan pengenalan aktivitas manusia pada data video yang berasal dari *Closed-Circuit Television* (CCTV) di Departemen Informatika ITS Surabaya.

Ada dua komponen dasar pada setiap algoritma pengenalan aktivitas yang memengaruhi hasil akurasi dan efisiensi yaitu representasi model aktivitas dan metode klasifikasi yang digunakan. Model yang digunakan untuk merepresentasikan aktivitas dapat dibagi menjadi lima kategori yaitu *shape models, motion models, geometric human body models, interest-point models, dan dynamic models*. Metode klasifikasi standar yang paling sering digunakan antara lain *nearest-neighbor* (NN) classifier, *support vector machine* (SVM), dan *boosting* [1].

Pengenalan aktivitas dengan *shape-based models* menggunakan estimasi akurasi dari siluet objek yang bergerak di setiap *frame* video sehingga membentuk *silhouette tunnel*. Model ini bersifat invarian terhadap pencahayaan, warna, dan tekstur dari objek yang bergerak, namun tidak semua data video memiliki model ini [1-10]. Pengenalan aktivitas dengan *geometric human body models* menggunakan bagian tubuh manusia sebagai modelnya sehingga akan sulit didapat dari data video dengan objek yang kecil dan *background* yang luas [11-14]. Pengenalan aktivitas dengan *interestpoint models* dilakukan dengan mengekstrak *intereset-point* dari pertimbangan informasi struktural dan mendeteksi *cuboids* di daerah yang memiliki kemungkinan besar mengalami pergerakan [15-20]. *Dynamic models* merupakan model yang paling baru digunakan dalam pengenalan aktivitas [21], [22]. Model ini mendeskripsikan postur statis dari suatu aksi sebagai sebuah *state* dan mendeskripsikan *dynamics* (variasi temporal) dari suatu aksi menggunakan transisi model *state-space*. Model yang terakhir yaitu *motion models*. Model ini mengekstrak berbagai karakteristik pergerakan dan deformasi objek. Model ini menggunakan fitur kinematic yang berasal dari *optical flow* untuk merepresentasikan aktivitas [23-30]. *Motion-model* dikatakan sebagai atribut yang paling diskriminatif dalam merepresentasikan suatu aksi [1].

Tugas akhir ini mengusulkan penggunaan fitur matriks kovarian sebagai representasi ringkas dari kumpulan fitur lokal yang berasal dari *optical flow* sehingga memiliki dimensi yang lebih sedikit. Selain itu, tugas akhir ini menggunakan metode *nearest-neighbor* (NN) *classifier* untuk klasifikasi yang diketahui tahan terhadap keanekaragaman bentuk aktivitas, perubahan sudut pandang kamera, dan objek dengan resolusi rendah. Secara konseptual, *framework* ini cukup sederhana, membutuhkan ruang penyimpanan yang kecil, serta memiliki perhitungan yang memungkinkan untuk implementasi secara real time [1].

## Rumusan Masalah

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

Bagaimana cara mendapatkan fitur lokal berbasis *optical flow* dari data video?

Bagaimana cara mendapatkan representasi model aktivitas manusia menggunakan fitur matriks kovarian?

Bagaimana cara klasifikasi aktivitas manusia menggunakan *nearest-neighbor* (NN) *classifier*?

Bagaimana mengetahui kinerja dari sistem pengenalan aktivitas manusia pada data video?

## Batasan Masalah

Permasalan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, diantaranya sebagai berikut:

Data yang digunakan pada tugas akhir ini berupa data video yang diperoleh dari CCTV di Departemen Informatika ITS Surabaya.

Aktivitas hanya dilakukan oleh manusia dan berobjek tunggal.

Aktivitas yang dapat dikenali berjumlah tiga yaitu aktivitas berlari, berjalan, dan melambaikan kedua tangan,

## Tujuan

Tujuan dari tugas akhir ini adalah membuat sistem pengenalan aktivitas manusia pada video yang terekam oleh CCTV menggunakan fitur matriks kovarian.

## 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.

## Metodologi

### 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.

### Studi Literatur

Pada tahap ini dilakukan pencarian informasi dan studi literatur sejumlah referensi tentang metode-metode yang akan digunakan dalam pengenalan aktivitas manusia pada data video. Informasi dan studi literatur tersebut didapat dari buku, internet, dan materi-materi kuliah yang berhubungan dengan metode yang digunakan.

### Analisis dan Desain 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 dilakukan desain sistem dan desain proses-proses yang ada.

### Implementasi Perangkat Lunak

Pengembangan sistem dalam tugas akhir nantinya akan menggunakan matlab sebagai compiler dan library yang telah tersedia di matlab.

### Pengujian dan Evaluasi

Pada tahapan ini dilakukan uji coba terhadap sistem yang telah dibuat. Pengujian ini betujuan untuk mengetahui unjuk kerja dari sistem pengenalan aktivitas pada data video. Pada uji coba ini, langkah pertama adalah melatih sistem dengan data training untuk membentuk model aktivitas. Langkah selanjutnya akan diberikan data uji untuk menguji model aktivitas yang telah terbentuk serta menghasilkan prediksi jenis aktivitas dari data uji. Parameter yang akan digunakan sebagai bahan uji coba adalah ukuran dari overlapping frame yang digunakan pada proses preprocessing video.

Setelah uji coba, dilakukan juga evaluasi terhadap kinerja dari sistem ini dengan cara melakukan pengukuran accuracy dan f-measure (precision dan recall) dari hasil uji coba yang berupa prediksi aktivitas terhadap aktivitas sebenarnya. Perhitungan accurac dan f-measure dapat dilihat pada Tabel 1 dan persamaan (1.1), (1.2), dan (1.3). Pengujian dan evaluasi akan dilakukan dengan melihat kesesuaian dengan perencanaan. Tahap ini dimaksudkan untuk mengetahui kinerja sistem, mengevaluasi jalannya sistem, mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Nilai Sebenarnya** | |
| ***TRUE*** | ***FALSE*** |
| **Nilai Prediksi** | ***TRUE*** | (*True Positive*) *Correct Result* | (*False Positive*) *Unexpected Result* |
| ***FALSE*** | (*False Negative*) *Missing Result* | (*True Negative*) *Correct Absence of Result* |

(1.1)

(1.2)

(1.3)

## 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:

**Bab I Pendahuluan**

Bab yang berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan tugas akhir. Selain itu permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

**Bab II Dasar Teori**

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.

**Bab III Perancangan Perangkat Lunak**

Bab ini berisi tentang desain sistem yang disajikan dalam bentuk *pseudocode*.

**Bab IV Implementasi**

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa *code* yang digunakan untuk proses implementasi.

**Bab V Uji Coba dan Evaluasi**

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.

**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.

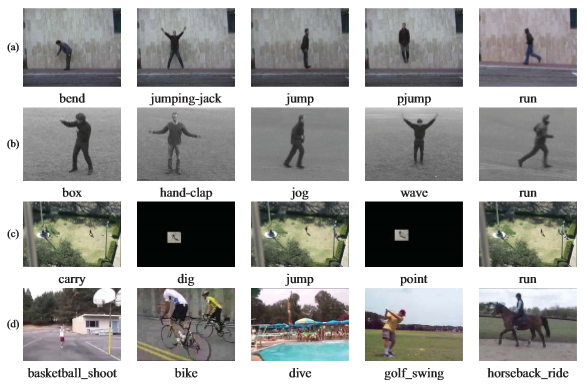
***[Halaman ini sengaja dikosongkan]***

# BAB II DASAR TEORI



## Pengenalan Aktivitas Manusia

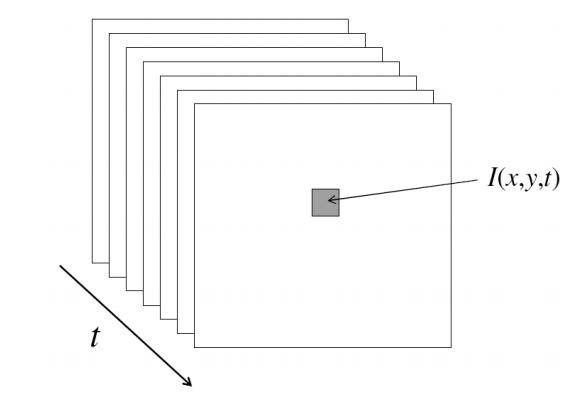
Pengenalan aktivitas manusia dapat dikelompokkan kedalam beberapa jenis. Pertama, pengenalan pada aktivitas sederhana yang dilakukan manusia seperti berjalan, naik tangga, turun tangga, jogging, dan lain-lain. Kedua, pada aktivitas kompleks yang biasanya mengkombinasikan aktivitas menjadi aktivitas dengan 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, Camera, dan GPS [2]. Contoh aktivitas manusia pada data video ditunjukkan pada Gambar 2.1.



Gambar 2.1 Contoh Data Citra Aktivitas Manusia pada Video

## *Optical Flow*

*Optical flow* merupakan fitur lokal yang dapat menangkap gerakan dinamis yang merupakan karakteristik aktivitas manusia dari data video dan terdiri dari 12 dimensi fitur. Pada beberapa tahun belakangan ini, sudah banyak penelitian yang membahas mengenai perhitungan *optical flow*. Pada kasus ini, yang akan digunakan adalah varian dari metode *Horn and Schunck*, yang mengoptimalkan fungsional berdasarkan residu dari batasan intensitas dan persyaratan regularisasi kehalusan [3].



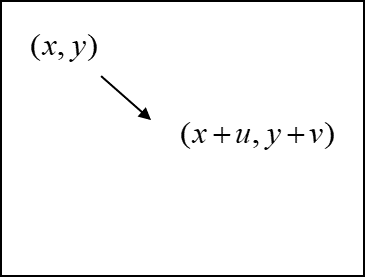
Gambar 2.2 Posisi Intensitas Terhadap Posisi Piksel dan Frame

menunjukkan intensitas pencahayaan pada rangkaian video di posisi piksel seperti ilustrasi pada Gambar 2. mewakili vector optical flow yang sesuai, dimana **.** Berdasarkan dan dapat didefinisikan vector fitur lokal pada persamaan (2.1).

(2.1)

dimana (himpunan semua koordinat piksel pada segmen video), adalah turunan parsial pertama dari yang berkenaan dengan waktu , sehingga dapat didefinisakan seperti pada persamaan (2.2).

(2.2)



Gambar 2.3 Ilustrasi Perpindahan Intensitas pada Frame

dan adalah komponen *optical flow* yang merupakan perpindahan posisi piksel dengan intensitas I berdasarkan seperti ilustrasi pada Gambar 3. Diasumsikan intensitas piksel antara *frame* satu dengan *frame* lainnya bersifat konstan, sehingga dapat didefinisikan seperti pada persamaan (2.3). sementara dan adalah turunan parsial pertamanya yang berkenaan dengan .

(2.3)

(*divergence*) adalah perbedaan spasial dari *flow field* yang ada pada setiap posisi piksel. *Divergence* menangkap jumlah ekspansi lokal yang dapat mengindikasikan perbedaan aktivitas. *Divergence* didefinisikan pada persamaan (2.4).

(2.4)

(*voricity*) digunakan untuk mengukur putaran lokal di sekitar sumbu yang tegak lurus terhadap bidang *flow field*. Pada konteks *optical flow*, *voricity* berpotensi menangkap gerakan melingkar lokal dari objek yang bergerak. *Voricity* didefinisikan pada persamaan (2.5).

(2.5)

Untuk menggambarkan dan ada dua matriks yang berperan, yaitu tensor gradien dari *optical flow* dan laju regangan tensor yang didefinisikan pada berurutan pada persamaan (2.6) dan (2.7).

(2.6)

(2.7)

dan adalah invarian tensor yang tetap konstan tidak peduli sistem koordinat apa yang mereka rujuk. dan juga merupakan sifat skalar yang menggabungkan komponen tensor gradien sehingga menghasilkan struktur lokal. dan didefinisikan berurutan pada persamaan (2.8) dan (2.9).

(2.8)

(2.9)

dimana menunjukkan *trace operation*.

## Matriks Kovarian

Sampel video biasanya berdimensi sangat tinggi. Tentu sangat tidak praktis mempelajari struktur global sampel video *learning* dan membangun klasifikasi secara langsung di ruang berdimensi tinggi. Oleh karena itu, dilakukanlah pendekatan pemodelan “*bag of dense local feature vector*” dimana serangkaian fitur lokal diekstrak dari data video untuk merepresentasikan aktivitas. Namun, dimensi sekumpulan vector fitur lokal tersebut bahkan lebih besar daripada sampel video yang diekstraksi karena jumlah piksel yang dikali dengan ukuran vector fitur tersebut. Sehingga diperlukannya pengurangan dimensi.

Fitur matriks kovarian dapat menyediakan representasi yang sangat baik untuk pengenalan aktivitas. Selain sederhana dan efektif, matriks kovarian dari fitur lokal memiliki ruang penyimpanan dan kebutuhan proses yang kecil. [4] [5]

menunjukkan “*bag of feature vector*” dan adalah ukuran dari . Matriks kovarian dari dapat didefinisikan seperti pada persamaan (2.10).

(2.10)

dimana adalah mean dari vector fitur yang didefinisikan pada persamaan (2.11).

(2.11)

Matriks kovarian menyediakan cara alami untuk menggabungkan beberapa vector fitur. Dimensi dari matriks kovarian hanya berhubungan dengan dimensi dari vector fitur. Jika berdimensi , maka merupakan matriks berukuran . Karena jauh lebih kecil dari , matriks akan memiliki dimensi yang jauh lebih kecil dibandingkan dengan dimensi dari “*bag of feature vector*” yang membutuhkan dimensi.

## Normalisasi

Normalisasi merupakan teknik untuk menstandarkan atau menyamakan rentang data sehingga tidak ada satu atribut yang terlalu dominan atas atribut yang lain. Salah satu teknik normalisasi adalah normalisasi skala.

Normalisasi skala merupakan teknik penstandaran data pada suatu rentang tertentu, umumnya 0-1. Diketahui nilai maksimal dari suatu piksel citra sebagai 𝑁𝑚𝑎𝑥 dan nilai minimal dari suatu piksel citra adalah 𝑁𝑚𝑖𝑛. Citra yang dinormalisasi disimbolkan dengan N. Nilai hasil normalisasi didapatkan dari rumus perhitungan normalisasi skala ditunjukkan pada persamaan (2.12)

(2.12)

## Matriks Log-Kovarian

Himpunan dari semua matriks kovarian dengan ukuran tertentu tidak membentuk sebuah ruang vector karena tidak diikuti oleh perkalian dengan scalar negatif, sehingga membentuk “*closed convex cone*” [6]. Kebanyakan dari algoritma *machine learning* bekerja dengan fitur yang berada pada ruang Euclidean, bukan *convex cone.* Oleh karena itu, diperlukannya pemetaan *convex cone* pada matriks kovarian ke dalam ruang vector dengan menggunakan logaritma dari matriks kovarian [7]. Perhitungan matriks kovarian dijelaskan pada persamaan (2.13).

(2.13)

dimana adalah *orthonormal eigenvectors* dan adalah matriks diagonal dari *eigenvalues*. Selanjutnya didefinisikan seperti pada persamaan (2.14).

(2.14)

Dimana adalah matriks diagonal dari matriks dengan mengganti entri diagonal matriks dengan nilai logaritmanya.

## *Nearest-Neighbor* (NN) *Classification*

Nearest-neighbor classification merupakan algoritma yang paling banyak digunakan untuk klasifikasi. Algoritma ini cukup sederhana dan mudah; diberikan sebuah sampel data testing. Temukan sampel yang paling mirip dengan sampel data testing tersebut pada himpunan data training, dimana kemiripan diukur dari suatu perhitungan jarak antara keduanya. Selanjutnya, beri label kelas pada sampel data testing tersebut.

Keberhasilan dari klasifikasi menggunakan NN classifier bergantung pada metrik jarak yang digunakan. Pada kasus ini, metric yang akan digunakan adalah pengukuran jarak Euclidean antara logaritma dari representasi aktivitas matriks kovarian.

(2.15)

dimana adalah logaritma matriks kovarian dan menunjukkan norma *Frobenius* pada matriks. Jarak yang didefinisikan pada persamaan (2.15) dapat ditunjukkan sebagai metrik *Riemannian* untuk jenis matriks kovarians. Metrik Ini disebut sebagai metrik *log-Euclidean*.

## *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.

Table 2.1 *Confusion Matrix*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | **Nilai Prediksi** | | |
| ***1*** | ***2*** | ***3*** |
| **Nilai Sebenarnya** | ***1*** |  |  |  |
| ***2*** |  |  |  |
| ***3*** |  |  |  |

Keterangan:

= Jumlah data uji yang kelas prediksinya sama

dengan kelas yang sebenarnya.

= 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*. *Accurac*y adalah perbandingan jumlah total data yang prediksi kelas hasil klasifikasinya sesuai dengan *ground truth* terhadap seluruh data. Rumus perhitungan dari akurasi ditunjukkan pada persamaan (2.16).

(2.16)

*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 (2.17), (2.18), (2.19).

(2.17)

(2.18)

(2.19)

*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 (2.20), (2.21), (2.22).

(2.20)

(2.21)

(2.22)

***[Halaman ini sengaja dikosongkan]***

# BAB III PERANCANGAN PERANGKAT LUNAK

Pada bab ini akan dibahas perancangan sistem pengenalan aktivitas manusia pada video. Perancagan yang dibuat meliputi perancangan sistem, perancangan data, dan perancangan proses. Hasil dari proses ini berupa diagram yang akan digunakan sebagai acuan untuk proses implementasi sistem.



## Lingkungan Desain dan Implementasi

Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam desain dan implementasi pembuatan sistem disebutkan dalam Tabel 3.1.

Table 3.1 Lingkungan Perancangan Sistem

|  |  |  |
| --- | --- | --- |
| **No.** | **Jenis** | **Spesifikasi** |
| 1 | Prosesor | Intel(R) Core(TM) i3-3240 CPU @ 3.40GHz 3.40 GHz |
| 2 | Memori | 4.00 GB |
| 3 | Sistem Operasi | Windows 10 Enterprise 64-bit |
| 4 | Perangkat Lunak | Matlab R2018a  Ms. Excel 2016  Ms.Word 2016 |

## Perancangan Sistem

Perancangan sistem dilakukan untuk menggambarkan proses sistem pengenalan aktivitas manusia pada video secara keseluruhan. Rancangan sistem ini ditunjukan dalam bentuk diagram alir pada Gambar 3.1 yang terdiri dari satu data masukan, lima proses, dan satu data keluaran. Data masukan yang digunakan adalah video yang berasal dar *CCTV* di Departemen Informatika ITS lantai 3, sebagai salah satu tempat yang paling sering terjadi aktivitas manusia seperti berjalan atau berlari. Contoh potongan video *CCTV* tersebut ditunjukkan pada Gambar 3.2.

Gambar 3.1 Diagram Alir Sistem Secara Keseluruhan

Citra Video Hasil *Preprocessing*

*Preprocessing*

Pengumpulan

Fitur*Optical Flow*

Perhitungan Matriks Kovarian

Perihitungan Matriks

Log Kovarian

Video

Jenis Aktivitas

Mulai

Selesai

Matriks Fn

Matriks Kovarian

Matriks Log Kovarian

Model Klasifikasi

Data *Training*

Data *Testing*

Pembuatan Model

NN *Classifier*

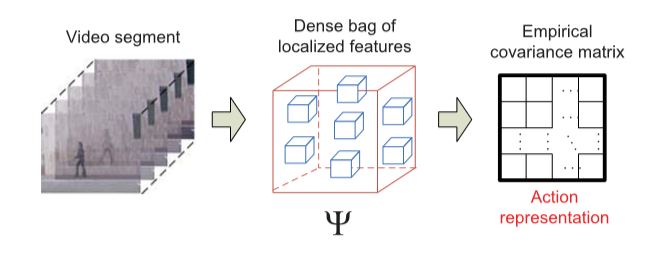
Klasifikasi



Gambar 3.2 View CCTV Departemen Informatika ITS

Proses *preprocessing* adalah proses untuk menyiapkan citra video sebelum masuk ke proses pengumpulan fitur. Persiapan tersebut berupa pengumpulan video berdasarkan aktivitas, *resize* video, dan mengubah video menjadi citra *greyscale*. Hasil *preprocessing* tersebut digunakan sebagai data masukan proses pengumpulan fitur.

Proses pengumpulan fitur terdiri dari proses perhitungan dan penggabungan fitur-fitur *optical flow*. Fitur *optical flow* terdiri dari 12 dimensi fitur yang diambil dari setiap piksel citra video. Sehinga hasil dari proses ini berupa *bag of localized features* berukuran N 12 dimana N adalah jumlah seluruh piksel dalam satu segmen video. *bag of localized features* inilah yang akan menjadi data masukan pada proses perhitungan matriks kovarian sebagai representasi aktivitas.

Seperti ilustrasi pada Gambar 3.2, setelah fitur-fitur *optical flow* dari suatu segmen video terkumpul, selanjutnya adalah merepresentasikannya menjadi sebuah aksi atau aktivitas dengan cara menghitung matriks kovarian dari fitur-fitur lokal tersebut. Hal ini bertujuan untuk mendapatkan dimensi yang jauh lebih kecil.

Gambar 3.3 Ilustrasi Perubahan Dimensi Video

Setelah diperoleh fitur dalam bentuk matriks kovarian, dilanjutkan dengan menghitung logaritma dari matriks kovarian tersebut. Hal ini bertujuan untuk memetakan bentuk matriks yang sebelumnya “*convex cone*” menjadi bentuk ruang *vector* sehingga dapat digunakan oleh *machine learning* untuk proses klasifikasi selanjutnya.

Proses klasifikasi dilakukan menggunakan *Nearest-Neighbor classifier* dengan jarak Euclidean sebagai *distance metric*-nya.

## Perancangan Data

Perancangan data dilakukan untuk memastikan pengoperasian sistem berjalan dengan benar. Data masukan (*input*) adalah data yang diperlukan dalam pengoperasian sistem dan data keluaran (*output*) adalah data yang dihasilkan oleh sistem serta akan digunakan oleh pengguna.

Data masukan pada sistem pengenalan aktivitas manusia adalah data video berukuran 636 360 piksel dengan citra warna RGBdan memiliki *framerate* sebesar 25 *fps*. Waktu yang digunakan untuk menggambarkan suatu gerakan adalah sekitar 0,4 — 0,8 detik yang berarti sepanjang 10 — 20 frame untuk setiap segmennya.

Gambar 3.4 Titik dan Arah Gerak Aktivitas



Gambar 3.5 Contoh Citra Aktivitas Manusia pada CCTV - Berlari



Gambar 3.6 Contoh Citra Aktivitas Manusia pada CCTV - Berjalan



Gambar 3.7 Contoh Citra Aktivitas Manusia pada CCTV - Melambaikan Kedua Tangan

Satu data video menampilkan satu objek yang melakukan satu jenis aktivitas. Aktivitas yang dilakukan yaitu aktivitas berlari, berjalan, dan melambaikan kedua tangan. Masing-masing aktivitas tersebut terdiri dari beberapa ragam pola arah dan titik gerakan pada latar video. Seperti yang ditunjukkan pada Gambar 3.4, ada enam arah aktivitas berlari, enam arah aktivitas berjalan, dan empat titik aktivitas melambaikan kedua tangan. Contoh data citra video CCTV yang akan menjadi data masukan ditunjukkan pada Gambar 3.5, 3.6, dan 3.7. Masing-masing gerakan aktivitas dilakukan oleh enam orang.

Data keluaran sistem pengenalan aktivitas manusia merupakan prediksi aktivitas yang dilakukan objek pada data masukan baik itu berlari, berjalan, atau melambaikan kedua tangan.

## Perancangan Proses

Perancangan proses dilakukan untuk memberikan gambaran mengenai setiap proses yang terdapat pada sistem pengenalan aktivitas manusia. Bagian dari setiap proses utama sistem secara keseluruhan dapat dilihat pada Gambar 3.1.

### *Preprocessing*

Proses pertama yang dilakukan adalah *preprocessing* yaitu mengamnil citra video pada frame t, mengubah ukuran video yang sebelumnya 1280 720 piksel diperkecil menjadi 636 360 piksel, serta mengubah citra video dari RGB menjadi *greyscale*. Hal tersebut dilakukan untuk mengoptimalkan kerja sistem. Diagram alir proses ini ditunjukkan pada Gambar 3.5.

### Pengumpulan Fitur *Optical Flow*

Gambar 3.5 Diagram Alir Proses *Preprocessing*

Ambil Citra Video pada Frame t

Ubah Ukuran Citra Video

Ubah Citra Video Menjadi *Greyscale*

Video

Citra Video Hasil *Preprocessing*

Mulai

Selesai

Proses pengumpulan fitur pada video hasil *preprocessing* terdiri dari proses perhitungan fitur *optical flow* pada citra video dan proses penggabungan fitur-fitur tersebut dalam satu segmen video menjadi sebuah matriks Fn. Fitur-fitur *optical flow* tersebut terdiri dari 12 dimensi fitur yang akan diambil dari setiap piksel pada citra video.

Fitur-fitur tersebut antara lain . Fitur-fitur tersebut akan dikumpulkan menjadi satu matriks Fn pada setiap segmen video dengan panjang frame, sehingga satu segmen video akan memiliki ukuran dimensi fitur sebesar Diagram alir proses ini ditunjukkan pada Gambar 3.6.

Gambar 3.6 Diagram Alir Proses Pengumpulan Fitur *Optical Flow*

Matriks Fn

Hitung Fitur Div, Vor, Gten, Sten

Selesai

Normalisasi

Inisialisasi

Penggabungan 12 Fitur Menjadi Matriks Fn

Hitung Fitur ut dan vt

Hitung Fitur It, u, v

Ambil Fitur x, y, t

Citra Video Hasil *Preprocessing*

Mulai

Citra Video Sebelumnya

t % L = 1

t % L = 0

yes

yes

no

no

*Preprocessing*

Proses pertama yaitu mengambil fitur yang menunjukkan posisi-posisi piksel dan *frame* pada citra video. Selanjutnya adalah pengambilan fitur dengan terlebih dahulu mengecek apakah citra video merupakan citra pertama dalam satu segmen atau bukan. Jika citra masukkan termasuk citra pertama, maka akan dilakukan inisialisasi fitur . Jika bukan, maka akan dilakukan perhitungan fitur tersebut menggunakan data fitur dari citra video dari frame sebelumnya.

Proses selanjutnya adalah perhitungan fitur yang merupakan turunan parsial masing-masing fitur terhadap posisi frame . Dilanjutkan dengan perhitungan fitur menggunakan data fitur .

Selanjutnya, ke-12 fitur masing-masing piksel pada citra video digabungkan menjadi sebuah matriks Fn. Setelah itu dilakukan pengecekan apakah citra video tersebut merupakan citra terakhir dalam satu segmen atau bukan. Jika citra masukan bukan termasuk citra terakhir, maka dilakukan proses pengulangan ke proses *preprocessing* yaitu pengambilan citra video pada frame selanjutnya.

Setelah terkumpul fitur-fitur *optical flow* pada citra-citra video dalam satu segmen sepanjang L frame dalam bentuk matriks Fn, maka dilanjutkan ke proses terakhir yaitu normalisasi matriks Fn menggunakan normalisasi skala.

### Perhitungan Matriks Kovarian

Perhitungan Matriks Kovarian adalah proses pengolahan matriks Fn menjadi suatu data yang dapat digunakan untuk merepresentasikan sebuah aktivitas menggunakan matriks kovariannya. Hal tersebut dilakukan untuk mengurangi ukuran dimensi matriks Fn menjadi jauh lebih kecil yaitu dari ukuran menjadi . Diagram alir proses ini ditunjukkan pada Gambar 3.7.

Proses pertama adalah dengan menghitung mean matriks Fn terhadap ke-12 fitur optical flow, dilanjutkan dengan menghitung mariks kovariannya.

Gambar 3.7 Diagram Alir Proses Perhitungan Matriks Kovarian

Hitung Mean Fn

Hitung Matriks Kovarian

Matriks Fn

Matriks Kovarian

Mulai

Selesai

### Perhitungan Matriks Log Kovarian

Proses selanjutnya adalah menghitung logaritma dari matriks kovarian. Hal ini dilakukan untuk membentuk matriks menjadi suatu ruang vector sehingga dapat digunakan oleh proses selanjutnya yakni klasifikasi. Diagram alir proses ini ditunjukkan pada Gambar 3.8.

Hitung Matriks Diagonal Eigenvalues dari Matriks Kovarian

Hitung Matriks Orthonormal Eigenvectors dari Matriks Kovarian

Matriks Log Kovarian

Matriks Kovarian

Mulai

Selesai

Hitung Matriks Log Kovarian

Gambar 3.8 Diagram Alir Proses Perhitungan Matriks Log Kovarian

Proses pertama adalah perhitungan matriks diagonal eigenvalues dari matriks kovarian. Dilanjutkan dengan perhitungan matriks orthonormal eigenvectors dari matriks kovarian. Lalu perhitungan matriks log kovarian menggunakan perkalian matriks antara matriks diagonal eigenvalues dan matriks orthonormal eigenvectors tersebut. Hasil dari proses inilah yang menjadi data untuk proses pembuatan model klasifikasi dan proses klasifikasi selanjutnya.

### Pembuatan Model NN *Classifier*

Pembuatan model *training Nearest-Neighbor classifier* dilakukan untuk melatih sistem dalam merepresentasikan aktivitas menggunakan data training hasil dari proses perhitungan matriks log kovarian. Diagram alir proses ini ditunjukkan pada Gambar 3.9.

Pembuatan Model

NN *Classifier*

Matriks Log Kovarian

Model Klasifikasi

Mulai

Selesai

### Klasifikasi Akivitas

Proses terakhir adalah proses klasifikasi untuk mengetahui jenis aktivitas yang dilakukan menggunakan *Nearest-Neighbor classifier*. Data masukan pada proses ini ada dua yaitu matriks log kovarian yang merupakan data keluaran proses perhitungan matriks log kovarian sebagai data testing dan model klasifikasi yang merupakan data keluaran proses permbuatan model NN *Classifier*. Diagram alir proses ini ditunjukkan pada Gambar 3.9.

Proses pertama adalah penghitungan jarak antara data testing terhadap model klasifikasi menggunakan jarak Euclidean sebagai *distance metric*-nya atau dalam hal ini disebut sebagai *log-Euclidean metric*. Setelah itu, nilai jarak setiap log kovarian diurutkan dan diambil yang paling kecil untuk mendapatkan jarak terpendek. Jarak terpendek tersebut yang akan menunjukkan jenis aktivitas apa yang paling mirip sebagai hasil klasifikasi.

Gambar 3.9 Diagram Alir Proses Klasifikasi

Hitung Jarak Euclidean

Matriks Log Kovarian

Jenis Aktivitas

Mulai

Selesai

Cari Jarak Terpendek

Model Klasifikasi

# BAB IV IMPLEMENTASI

Pembahasan implementasi pada bab ini meliputi deskripsi lingkungan tahap implementasi, proses-proses pada tahap implementasi yang dikerjakan, beserta penjelasan fungsi-fungsinya dalam bentuk kode sumber.



## Lingkungan Implementasi

Beberapa perangkat pendukung yang digunakan pada proses implementasi sistem pengenalan aktivitas ialah sebagai berikut.

### Perangkat Keras

Lingkungan implementasi pada tugas akhir ini adalah sebuah *personal computer* (PC). Spesifikasi dari PC yang digunakan pada tugas akhir ini adalah memiliki prosesor Intel Core i3-3240 dengan kecepatan 3,40 GHz dan Random Access Memory (RAM) untuk proses menjalankan program sebesar 4,00 GB.

### Perangkat Lunak

Lingkungan implementasi pada tugas akhir ini adalah sebuah personal computer (PC). Spesifikasi PC dari sisi perangkat lunak menggunakan software MATLAB R2018a, Microsoft Office Excel dan Microsoft Office Word 2016.

## Implementasi Preprocessing

Proses *preprocessing* pada tugas akhir ini terdiri dari tiga tahap yaitu mengambil citra video pada frame t, mengubah ukuran citra, dan mengubahnya menjadi citra *greyscale*. Implementasi proses *preprocessing* ditunjukkan pada Kode Sumber 4.1.

|  |  |
| --- | --- |
| 1 | vid = VideoReader(videoFileName); |
| 2 | im = read(vid,t); |
| 3 | imResize = imresize(im,scale); |
| 4 | imNew = single(rgb2gray(imResize)); |

Kode Sumber 4.1 Implementasi *Preprocessing*

Baris 1 menunjukkan pembacaan file video. Baris 2 menunjukkan pengambilan citra video pada frame ke-t dari file video. Dilanjutkan dengan perubahan ukuran citra video pada baris 3 dan perubahan citra video menjadi *greyscale* pada baris 4.

## Implementasi Pengumpulan Fitur *Optical Flow*

Proses pengumpulan fitur *optical flow* pada tugas akhir ini terdiri dari beberapa tahap di antaranya perhitungan fitur dan penggabungan fitur. Tahap perhitungan fitur adalah tahap pengumpulan 12 fitur *optical flow* pada data masukan, yang selanjutnya akan digabung dengan fitur-fitur pada citra video lainnya dalam segmen yang sama menjadi matriks Fn. Penjelasan lebih rinci dari masing-masing tahapan tersebut adalah sebagai berikut.

### Pengambilan Fitur

Pengambilan fitur  yang menunjukkan posisi-posisi piksel dan *frame* pada citra video ditunjukkan pada Kode Sumber 4.2

|  |  |
| --- | --- |
| 1 | [m,n] = size(imNew); |
| 2 | row = reshape(1:m,m,[]); |
| 3 | matX = row\*ones(1,n); |
| 4 | col = reshape(1:n,[],n); |
| 5 | matY = ones(m,1)\*col; |
| 6 | matT = ones(m,n)\*t; |

Kode Sumber 4.2 Implementasi Pengambilan Fitur

Oleh karena pengumpulan fitur *optical flow* dilakukan pada setiap piksel citra masukan, maka dibuatlah masing-masing sebuah matriks yang berisi posisi piksel , posisi piksel , serta posisi frame *t* seukuran dengan citra video. Baris 1 menunjukkan pengambilan ukuran citra video. Baris 2 dan 3 menunjukkan pembuatan matriks fitur , baris 4 dan 5 menunjukkan pembuatan matriks fitur , dan baris 6 menunjukkan pembuatan matriks fitur *t*.

### Inisialisasi

Sebelum melakukan pengambilan fitur , terlebih dahulu dicek apakah citra video merupakan citra pertama dalam suatu segmen atau bukan. Jika citra masukkan termasuk citra pertama, maka akan dilakukan inisialisasi fitur-fitur yang dibutuhkan dalam perhitungan fitur *optical flow* selanjutnya. Implementasi proses inisialisasi ditunjukkan pada Kode Sumber 4.3.

|  |  |
| --- | --- |
| 1 | It = zeros(size(imNew),'single'); |
| 2 | U = zeros(size(imNew)); |
| 3 | V = zeros(size(imNew)); |
| 4 | Ux = zeros(size(imNew),'single'); |
| 5 | Uy = zeros(size(imNew),'single'); |
| 6 | Ut = zeros(size(imNew),'single'); |
| 7 | Vx = zeros(size(imNew),'single'); |
| 8 | Vy = zeros(size(imNew),'single'); |
| 9 | Vt = zeros(size(imNew),'single'); |
| 10 | imPrev = imNew; |
| 11 | Uprev = U; |
| 12 | Vprev = V; |

Kode Sumber 4.3 Implementasi Inisialisasi

Pada baris 1 sampai baris 9 secara berurutan merupakan inisialisasi fitur It, U, V, Ux, Uy, Ut, Vx, Vy, dan Vt. Sedangkan baris 10, 11, dan 12 merupakan inisialisasi citra video serta fitur u dan v pada frame sebelumnya.

### Perhitungan Fitur *It , u, dan v*

Fitur *optical flow* selanjutnya adalah fitur It, u, dan v. It adalah turunan parsial pertama dari intensitas terhadap frame t. u dan v Implementai perhitungan fitur It, u, dan v ditunjukkan pada Kode Sumber 4.4.

|  |  |
| --- | --- |
| 1 | gg = [0.2163,0.5674,0.2163]; |
| 2 | tInt = 0.5; |
| 3 | It = tInt\*It + (1-tInt)\*2\*conv2(gg,gg,imNew - imPrev,'same'); |
| 4 | for i = 1:MaxIterations |
| 5 | uAvg=conv2(U,kern,'same'); |
| 6 | vAvg=conv2(V,kern,'same'); |
| 7 | U = uAvg - dx.\*(dx.\*uAvg + dy.\*vAvg + dt)./(eta.^2 + dx.^2 +dy.^2); |
| 8 | V = vAvg - dy.\*(dx.\*uAvg + dy.\*vAvg + dt)./(eta.^2 + dx.^2 +dy.^2); |
| 9 | End |

Kode Sumber 4.4 Perhitungan Fitur *It , u,* dan *v*

Perhitungan fitur It diunjukkan pada baris 3, perhitungan fitur u dan v ditunjukkan pada baris 4 sampai baris 9 mengacu pada metode *Horn and Schunk.*

### Perhitungan Fitur *ut* dan *vt*

Fitur optical flow selanjutnya adalah  dan yang merupakan turunan parsial pertama fitur u dan v terhadap Implementasi perhitungan fitur dan ditunjukkan pada Kode Sumber 4.5.

|  |  |
| --- | --- |
| 1 | gg = [0.2163,0.5674,0.2163]; |
| 2 | tInt = 0.5; |
| 3 | Ut = tInt\*ut + (1-tInt)\*2\*conv2(gg,gg,U - Uprev,'same'); |
| 4 | Vt = tInt\*vt + (1-tInt)\*2\*conv2(gg,gg,V - Vprev,'same'); |

Kode Sumber 4.5 Implementasi Perhitungan Fitur *ut* dan *vt*

Perhitungan dan masing-masing ditunjukkan pada baris 3 dan 4.

### Perhitungan Fitur *Div, Vor, Gten,* dan *Sten*

Fitur optical flow selanjutnya adalah tur Div, Vor, Gten, dan Sten. (*divergence*) adalah perbedaan spasial dari *flow field* yang ada pada setiap posisi piksel. (*voricity*) digunakan untuk mengukur putaran lokal di sekitar sumbu yang tegak lurus terhadap bidang *flow field*. dan adalah invarian tensor yang tetap konstan tidak peduli sistem koordinat apa yang mereka rujuk. Implementasi fitur-fitur ini ditunjukkan pada Kode Sumber 4.6.

|  |  |
| --- | --- |
| 1 | gg = [0.2163,0.5674,0.2163]; |
| 2 | tInt = 0.5; |
| 3 | fU = U + Uprev; |
| 4 | Ux = tInt\*ux + (1-tInt)\*conv2(fU(:,[2:end end ]) - fU(:,[1 1:(end-1)]),gg','same'); |
| 5 | Uy = tInt\*uy + (1-tInt)\*conv2(fU([2:end end],: ) - fU([1 1:(end-1)],:),gg,'same'); |
| 6 | fV = V + Vprev; |
| 7 | Vx = tInt\*vx + (1-tInt)\*conv2(fV(:,[2:end end ]) - fV(:,[1 1:(end-1)]),gg','same'); |
| 8 | Vy = tInt\*vy + (1-tInt)\*conv2(fV([2:end end],: ) - fV([1 1:(end-1)],:),gg,'same'); |
| 9 | Div = Ux + Vy; |
| 10 | Vor = Vx - Uy; |
| 11 | Gten = 0.5\*(((Ux+Vy).\*(Ux+Vy)) - ((Ux.\*Ux) + (Uy.\*Vx) + (Vx.\*Uy) + (Vy.\*Vy))); |
| 12 | Sten = 0.5\*(((Ux+Vy).\*(Ux+Vy)) - ((Ux.\*Ux) + ((Uy+Vx).\*(Uy+Vx)./4) + ((Uy+Vx).\*(Uy+Vx)./4) + (Vy.\*Vy))); |

Kode Sumber 4.6 Implementasi Perhitungan Fitur *Div, Vor, Gten,* dan *Sten*

Seperti pada persamaan (2.6), (2.7), (2.8), dan (2.9), perhitungan Div, Vor, Gten, dan Sten membutuhkan fitur ux, uy, vx, dan vy yang perhitungannya ditunjukkan pada baris 4, 5, 7, dan 8. Dilanjutkan dengan perhitungan fitur Div, Vor, Gten, dan Sten pada baris 9, 10, 11, dan 12.

### Penggabungan 12 Fitur Menjadi Matriks Fn

Setelah mendapat 12 fitur *optical flow* dilakukanlah penggabungan fitur-fitur tersebut ke dalah sebuah matriks Fn. Implementasi penggabungan 12 fitur ini ditunjukkan pada Kode Sumber 4.7.

|  |  |
| --- | --- |
| 1 | temp(1,:) = reshape(x,1,[]); |
| 2 | temp(2,:) = reshape(y,1,[]); |
| 3 | temp(3,:) = reshape(t,1,[]); |
| 4 | temp(4,:) = reshape(u,1,[]); |
| 5 | temp(5,:) = reshape(v,1,[]); |
| 6 | temp(6,:) = reshape(It,1,[]); |
| 7 | temp(7,:) = reshape(Ut,1,[]); |
| 8 | temp(8,:) = reshape(Vt,1,[]); |
| 9 | temp(9,:) = reshape(Div,1,[]); |
| 10 | temp(10,:) = reshape(Vor,1,[]); |
| 11 | temp(11,:) = reshape(Gten,1,[]); |
| 12 | temp(12,:) = reshape(Sten,1,[]); |
| 13 | Fn = [Fn temp]; |

Kode Sumber 4.7 Implementasi Penggabungan 12 Fitur Menjadi Matriks Fn

Baris 1 sampai baris 12 menunjukkan perubahan matriks masing-masing fitur menjadi vektor baris. Selanjutnya vektor-vektor baris tersebut disatukan menjadi matriks Fnpada baris 13. Proses ini dilakukan sebanyak L frame dan menggabungkan seluruh fitur pada L frame citra video berurutan sebelum masuk ke proses selanjutnya.

### Normalisasi

Setelah iterasi sebanyak L frame, terkumpulah *bag of feature vector* pada matriks Fn. Sebelum dilakukannya perhitungan matriks kovariannya, terlebih dahulu fitur-fitur optical flow tersebut dinormalisasi agar terdistribusi merata. Implementasi normalisasi ditunjukkan pada Kode Sumber 4.8.

|  |  |
| --- | --- |
| 1 | [m,n] = size(Fn); |
| 2 | for i = 1 : m |
| 3 | Fn(i,:) = normalize(Fn(i,:),'range'); |
| 4 | End |

Kode Sumber 4.8 Implementasi Normalisasi

Baris pertama menunjukan pengambilan ukuran baris matriks Fn. Selanjutnya pada baris 2 sampai baris 4 dilakukan normalisasi skala pada setiap fitur *optical flow* pada matriks Fn sebanyak jumlah row matriks Fn yang merupakan banyaknya fitur yang dikumpulkan yaitu 12.

## Implementasi Perhitungan Matriks Kovarian

Proses selanjutnya setelah *bag of feature vector* pada matriks Fn dinormalisasi, dilakukanlah perhitungan matriks kovariannya untuk mengurangi jumlah fitur yang sangat banyak. Implementasi perhitungan matriks kovarian ditunjukkan pada Kode Sumber 4.9.

|  |  |
| --- | --- |
| 1 | [m,n] = size(Fn); |
| 2 | Fnmean = mean(Fn,2); |
| 3 | Fnmin = Fn - Fnmean; |
| 4 | Cov = Fnmin \* Fnmin.'\*1/n; |

Kode Sumber 4.9 Implementasi Perhitungan Matriks Kovarian

Baris pertama menunjukan pengambilan ukuran baris matriks Fn. Baris 2 menunjukan perhitungan mean matriks Fn terhadap masing-masing fitur. Selanjutnya pada baris 3 dan 4 adalah perhitungan matriks kovarian dari matriks Fn sesuai dengan persamaan (2.10).

## Implementasi Perhitungan Matriks Log Kovarian

Matriks kovarian dengan ukuran tertentu tidaklah membentuk suatu ruang vector karena tidak diikuti oleh perkalian dengan scalar negatif, sehingga membentuk “*closed convex cone*” yang mana kebanyakan dari algoritma *machine learning* bekerja dengan fitur yang berada pada ruang Euclidean, bukan *convex cone.* Oleh sebab itu dibutuhkanlah perhitungan matriks log kovarian untuk memetakan bentuk *convex cone* pada matriks kovarian ke dalam ruang vector. Implementasi perhitungan matriks log kovarian ditunjukkan pada Kode Sumber 4.10.

|  |  |
| --- | --- |
| 1 | [V,D] = eig(Cov); |
| 2 | Ddiag = diag(D); |
| 3 | Dlog = log(Ddiag); |
| 4 | D = diag(Dlog); |
| 5 | LogCov = V\*D\*V.'; |

Kode Sumber 4.10 Implementasi Perhitungan Matriks Log Kovarian

Baris 1 menunjukkan perhitungan untuk mendapatkan *orthonormal eigenvectors* dan *eigenvalues* matriks kovarian. Mengacu pada persamaan (2.14), perhitungan matriks diagonal dari matriks *eigevalues* dengan mengganti entri diagonal matriks dengan nilai logaritmanya ditunjukkan pada baris 2 sampai baris 4, dilanjutkan dengan perhitungan matriks log kovarian pada baris 5. Matriks log kovarian inilah yang akan merepresentasikan suatu aktivitas.

## Implementasi Pembuatan Model NN *Classifier*

Untuk melakukan proses klasifikasi dengan data *testing*, dibutuhkanlah suatu model data *training* sebagai pembanding dalam menentukan suatu kelas atau label. Implementasi pembuatan model data training dengan metode NN *classifier* ditunjukkan pada Kode Sumber 4.11.

|  |  |
| --- | --- |
| 1 | mdl = fitcknn(train,train\_label,'Distance', 'euclidean'); |

Kode Sumber 4.11 Implementasi Pembuatan Model NN *Classifier*

## Implementasi Klasifikasi Aktivitas

Proses selanjutnya adalah klasifikasi data testing untuk menentukan jenis aktivitas yang dilakukan. Implementasi klasifikasi aktivitas dengan metode NN classifier ditunjukkan pada Kode Sumber 4.12.

|  |  |
| --- | --- |
| 1 | test\_label = predict(mdl,test); |

Kode Sumber 4.12 Implementasi Klasifikasi Aktivitas

## Implementasi Pengukuran Kinerja Sistem

Untuk mengukur kinerja sistem pengenalan aktivitas pada video, digunakanlah metode pengukuran 10*-fold cross validation* pada saat klasifikasi aktivitas dengan metode NN *classifier.* Implementasi pengukuran kinerja sistem ditunjukkan pada Kode Sumber 4.13.

|  |  |
| --- | --- |
| 1 | X = xlsread(excelFileName,data); |
| 2 | Y = xlsread(excelFileName,dataLabel); |
| 3 | mdl = fitcknn(X,Y,'Distance','euclidean'); |
| 4 | cvmdl = crossval(mdl); |
| 5 | cvmdl\_predict = kfoldPredict(cvmdl); |
| 6 | cvmdl\_conf = confusionmat(cvmdl.Y,cvmdl\_predict); |
| 7 | validationAccuracy = 1 - kfoldLoss(cvmdl, 'LossFun', 'ClassifError'); |

Kode Sumber 4.13 Implementasi Pengukuran Kinerja Sistem

Baris 1 dan baris 2 masing-masing menunjukan pembacaan data uji dan data label klasifikasinya yang telah disimpan sebelumnya dalam file excel. Selanjutnya dilakukan pembentukkan model dengan metode NN *classifier* pada baris 3*.* Implementasi uji coba menggunakan *10-fold cross validation* ditunjukkan pada baris 4. Selanjutnya, untuk mendapatkan data label prediksi ditunjukkan pada baris 5 untuk perhitungan *confusion matrix* pada baris 6, dan yang terakhir perhitungan akurasi ditunjukkan pada baris 7.

# BAB V UJI COBA DAN EVALUASI

Dalam bab ini dibahas mengenai hasil uji coba sistem yang telah dirancang dan dibuat. Uji coba dilakukan untuk mengetahui kinerja sistem dengan lingkungan uji coba yang telah ditentukan

## Lingkungan Uji Coba

Lingkungan uji coba pada tugas akhir ini adalah sebuah personal computer (PC). Spesifikasi PC dari sisi perangkat keras adalah memiliki prosesor Intel Core i3 3240 dengan kecepatan 3,40 GHz dan memori untuk proses sebesar 4,00 GB. PC yang digunakan memiliki sistem operasi Windows 10. Pada sisi perangkat lunak, uji coba pada tugas akhir ini dilakukan dengan menggunakan software MATLAB R2018a dan Microsoft Excel.

## Data Uji Coba

Data uji coba yang digunakan sebagai data masukan adalah dataset Weizmann dan dataset CCTV. Dataset Weizmann merupakan video berukuran 144 x 180 piksel dengan *framerate* 25 *fps* dan citra warna RGB. Dataset ini memiliki 10 jenis aktivitas yaitu *run, walk, wave2, jump, pjump, jack, side, skip, wave1, dan bend*. Masing-masing aktivitas dilakukan oleh 9-10 orang yang berbeda. Contoh citra video dataset weizmann ditunjuukan pada Gambar 5.1.

Dataset CCTV merupakan video berukuran 360 x 636 piksel dengan *framerate* 25 *fps* dan citra warna RGB. Dataset ini memiliki 3 jenis aktivitas yaitu berlari, berjalan, dan melambaikan kedua tangan. Masing-masing aktivitas dilakukan oleh 6 orang yang berbeda. Contoh citra video dataset CCTV ditunjukkan pada Gambar 3.6, 3.5, dan 3.7.

## Skenario Uji Coba

Uji coba dilakukan untuk mengetahui nilai-nilai parameter yang tepat untuk digunakan pada masing-masing proses. Nilai parameter yang tepat penting untuk diketahui karena penggunaan parameter yang tepat akan memberikan hasil yang terbaik pada keluaran tiap proses. Selain menentukan nilai parameter, uji coba berguna melihat performa sistem dengan metode lainnya. Skenario pengujian dilakukan pada dua jenis data yaitu pada dataset weizmann dan data CCTV.

Skenario uji coba pada data Weizmann dilakukan untuk mengetahui kesesuaian implementasi sistem dengan hasil riset yang tertera pada paper acuan. Skenario yang digunakan adalah uji coba panjang frame L.

Skenario uji coba pada data CCTV ialah sebagai berikut.

1. Uji Coba degan parameter panjang frame L
2. Uji Coba dengan parameter waktu pengambilan data CCTV

## Skenario Uji Coba pada Dataset Weizmann

Uji coba pada dataset weizmann dilakukan untuk mengetahui kesesuaian hasil akurasi dengan yang sudah diimplementasikan pada paper acuan. Uji coba dilakukan dengan melakukan variasi pada panjang frame L yang digunakan dalam satu segmen, yaitu L = 8 dan L – 20. Uji coba ini dilakukan pada sistem menggunakan metode 10-*fold cross validation*. Hasil uji coba tertera pada Tabel 5.1 yang menunjukkan hasil akurasi, *recall,* dan *precision* pada masing-masing kelas aktivitas*.*

Table 5.1 Hasil Uji Coba Sistem pada Dataset Weizmann

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | L = 8 | | L = 20 | |
| Aktivitas | Recall | Precision | Recall | Precision |
| Run | 90.75 | 88.89 | 98.60 | 99.07 |
| Walk | 92.98 | 94.15 | 99.42 | 99.42 |
| Wave2 | 91.80 | 90.51 | 99.78 | 98.91 |
| Jump | 88.61 | 88.83 | 98.61 | 99.30 |
| Pjump | 91.58 | 96.45 | 100.00 | 100.00 |
| Jack | 98.50 | 96.19 | 99.82 | 99.82 |
| Side | 87.66 | 87.89 | 98.17 | 98.53 |
| Skip | 86.06 | 85.65 | 98.99 | 97.67 |
| Wave1 | 92.71 | 93.66 | 99.38 | 99.58 |
| Bend | 98.61 | 97.43 | 99.15 | 99.78 |
| Akurasi | 92.59 % | | 99.31 % | |

## Skenario Uji Coba pada Dataset CCTV

Uji coba pada dataset CCTV dilakukan menggunakan dua parameter yaitu panjang frame L dan waktu pengambilan data sebagai berikut.

### Skenario Uji Coba Panjang Frame L

Waktu yang dibutuhkan untuk merepresentasikan aktivitas manusia adalah sekitar 0.4-0.8 detik. *Framerate* pada video CCTV adalah sebesar 25 *fps*. Dengan kata lain, waktu yang dibutuhkan untuk merepresentasikan suatu aktivitas dalam dataset CCTV adalah sebesar 10 – 20 frame. Oleh karena itu dilakukan pengujian pada panjang frame L sebesar 10, 15, dan 20 untuk mengetahui panjang frame yang paling bagus dalam merepresentasikan suatu aktivitas. Hasil uji coba sistem pada dataset CCTV dengan varian panjang frame L ditunjukkan pada Tabel 5.2 yang menunjukkan hasil akurasi, *recall,* dan *precision* pada masing-masing kelas aktivitas*.*

Table 5.2 Hasil Uji Coba Sistem pada Dataset CCTV dengan Panjang Frame L = 10, 15, 20

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Run | Walk | Wave | Akurasi |
| L = 10 | Recall |  |  |  |  |
| Precision |  |  |  |
| L = 15 | Recall |  |  |  |  |
| Precision |  |  |  |
| L = 20 | Recall | 86.00 % | 94.80 % | 98.05 % | 94.07 % |
| Precision | 98.74 % | 93.88 % | 96.85 % |

### Skenario Uji Coba Waktu Pengambilan Data

Uji coba sistem pada dataset CCTV juga dilakukan dengan varian waktu pengambilan data yaitu saat pagi, siang, dan malam hari. Hal ini dilakukan untuk mengetahui kinerja sistem dengan perbedaan intensitas cahaya yang dihasilkan tiap waktu. Hasil uji coba sistem pada dataset CCTV dengan varian waktu pengambilan data ditunjukkan pada Tabel 5.3 yang menunjukkan hasil akurasi, *recall,* dan *precision* pada masing-masing kelas aktivitas*.*

Table 5.3 Hasil Uji Coba Sistem pada Dataset CCTV dengan Waktu Pagi, Siang, dan Malam

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Run | Walk | Wave | Akurasi |
| Pagi | Recall |  |  |  |  |
| Precision |  |  |  |
| Siang | Recall |  |  |  |  |
| Precision |  |  |  |
| Malam | Recall |  |  |  |  |
| Precision |  |  |  |

## Evaluasi Uji Coba pada Dataset Weizmann

Berdasarkan hasil uji coba sistem pada dataset weizmann, diketahui bahwa hasil yang didapat oleh sistem pengenalan aktivitas pada video ini telah sesuai dengan hasil yang didapat dalam paper acuan, dimana penggunaan panjang frame L = 20 menghasilkan akurasi yang lebih baik dari penggunaan panjang frame L = 8. Hal sama terjadi pada hasil recall dan precision pada masing-masing kelas aktivitas.

## Evaluasi Uji Coba pada Dataset CCTV

Berdasarkan hasil uji coba sistem pada dataset CCTV dengan variasi panjang frame L yaitu L = 10, 15, dan 20, diketahui bahwa panjang frame yang paling baik dalam merepresentasikan suatu aktivitas pada dataset CCTV adalah panjang frame L = ? dengan akurasi sebesar ?.

Selanjutnya, dengan mengimplementasikan panjang frame terbaik yang didapat dari uji coba sebelumnya, dilakukan kembali uji coba sistem pada dataset CCTV dengan variasi waktu pengambilan video yaitu pagi, siang, dan malam hari. Hasil yang didapat dari uji coba ini menunjukkan bahwa akurasi terbaik didapat pada waktu ? hari dengan akurasi sebesar ?.

# BAB VI KESIMPULAN DAN SARAN

Bab ini membahas kesimpulan yang dapat diambil dari hasil uji coba dan perancangan sistem sebagai jawaban dari rumusan masalah yang telah dikemukakan dan saran yang berisi pengembangan yang dapat dilakukan lebih lanjut untuk menyempurnakan sistem pengenalan aktivitas manusia pada data video.



## Kesimpulan

Berikut merupakan kesimpulan yang dapat diambil dari proses pengembangan dan hasil uji coba sistem.

1. Fitur matriks kovarian dapat digunakan untuk mereduksi *bag of feature vector* dari fitur *optical flow* pada data video menjadi jauh lebih kecil namun tetap dapat merepresentasikan suatu aktivitas.
2. Berdasarkan uji coba yang dilakukan, implementasi sistem pengenalan aktivitas manusia pada video pada tugas akhir ini telah sesuai dengan yang telah dijabarkan oleh paper acuan.
3. Berdasarkan uji coba yang dilakukan, panjang frame yang paling baik dalam merepresentasikan aktivitas adalah sepanjang L frame untuk kasus pada dataset CCTV di Departemen Informatika ITS lantai 3.
4. Berdasarkan uji coba yang telah dilakukan, waktu terbaik pengambilan data untuk mencapai akurasi tertinggi adalah saat malam hari dimana intensitas cahaya pada saat itu censerung stabil oleh penerangan lampu.

## Saran

Saran yang dapat diberikan dalam pengembangan lebih lanjut untuk menyempurnakan sistem adalah sebagai berikut.

Sebaiknya sistem pengenalan aktivitas pada video CCTV dapat dilakukan secara *realtime* sehingga dapat dimanfaatkan dalam bidang keamanan di Departemen Informatika ITS ini. Selain itu, sebaiknya pengembangan sistem dilakukan dengan memperbanyak kelas aktivitas yang dapat diklasifikasi seperti aktivitas dinamis lainnya maupun aktivitas statis seperti duduk dan berdiri.

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | K. Guo, P. Ishwar and J. Konrad, "Action Recognition from Video Using Feature Covariance Matrices," in *Transaction on Image Processing, Vol. 22, No. 6*, IEEE, 2013, pp. 2479-2494. |
| [2] | S. Xian, "Activity Recognition with Smartphone Sensors Second Exam," 2014, pp. 5-6. |
| [3] | C. Zach, T. Pock and H. Bischof, "A duality based approach for realtime TV-L1 optical ﬂow," in *Proc. 29th DAGM Conf. Pattern Recognit.*, 2007, pp. 214-223. |
| [4] | O. Tuzel, F. Porikli and P. Meer, "Region covariance: A fast descriptor for detection and classiﬁcation,," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 589-600. |
| [5] | O. Tuzel, F. Porikli and P. Meer, "Pedestrian detection via classiﬁcation on Riemannian manifolds," in *Trans. Pattern Anal. Mach. Intell., vol. 30, no. 10*, IEEE, 2008, pp. 1713-1727. |
| [6] | R. Hill; S. Waters, "On the cone of positive semideﬁnite matrices," in *Linear Algebra Appl., vol. 90*, 1987, pp. 81-88. |
| [7] | V. Arsigny, P. Pennec and X. Ayache, "Log-Euclidean metrics for fast and simple calculus on diffusion tensors," in *Magn. Resonance Med., vol. 56, no. 2*, 2006, pp. 411-421. |

***[Halaman ini sengaja dikosongkan]***

# BIODATA PENULIS