UNIVERSITÄT
KOBLENZ · LANDAU
Fachbereich 4: Informatik

Human Action Recognition in Video Data using Color and Distance

by

Rawya Al-Akam

Approved Dissertation thesis for the partial fulfillment of the requirements
for a Doctor of Natural Sciences (Dr. rer. nat.)

Fachbereich 4: Informatik
Universität Koblenz-Landau

Chair of PhD Board:	Prof. Dr. Ralf Lämmel
Chair of PhD Commission:	Prof. Dr. Patrick Delfmann
Examiner and Supervisor:	Prof. Dr.-Ing. Dietrich Paulus
Further Examiner:	J.-Prof. Dr.-Ing. habil. Kai Lawonn

Date of the doctoral viva: June 09, 2021

Acknowledgments

This thesis summarizes five years of intense research, development of algorithms, and experimental implementations that would not have been realized without the great support of several people.

First of all, I would like to express my gratitude to my supervisor Prof. Dr. Dietrich Paulus for his valuable ideas, suggestions, and comments. I will always be grateful to him for his guidance and patience in transforming me into a scientist. His excellent professional ethic motivated me to rigorously engage with research in the Computer Vision field.

I wish to thank JProf. Dr.-Ing. Kai Lawonn for immediately agreeing to be a second referee. My sincere thanks also go to the PhD committee.

I would also like to thank the AGAS group for their valuable advice and help during my graduate studies. Special thanks to Maria Lebeda, Christian Fuchs, Raphael Memmesheimer, and Nick Theisen for helping and supporting me during the period of my PhD study.

Great thanks also to collaborator and co-author, Dr. Salah Al-Daaragi, PhD in Robotic-Human-Interaction from the University of Kaiserslautern, Germany, for his feedback on this thesis.

Thank you very much to American master student Eric Babcock at the department of Mathematics/ University of Koblenz-Landau, Germany, and Susan Bahlol from the USA - Ms.c. in Computer Science/Image Processing for doing English proofread to this thesis.

Thanks to the Iraqi Ministry of Higher Education and Scientific Research (MHESR) for the financial support of my PhD study at the University of Koblenz-Landau.

Special thanks and most important words of love and gratitude are kept for my husband Mohammed and my children Fatima and Hussein for being so supportive to me and for sharing all of the happiest and stressful days of our lives.

My last thankful words are kept for my parents, my sisters, and my brothers. It is thanks to them I was able to come to the finish line of my PhD study.

Thank You!

Abstract

Human action recognition from a video has received growing attention in computer vision and has made significant progress in recent years. Action recognition is described as a requirement to decide which human actions appear in videos. The difficulties involved in distinguishing human actions are due to the high complexity of human behaviors as well as appearance variation, motion pattern variation, occlusions, etc. Many applications use human action recognition on captured video from cameras, resulting in video surveillance systems, health monitoring, human-computer interaction, and robotics.

Action recognition based on RGB-D data has increasingly drawn more attention to it in recent years. RGB-D data contain color (Red, Green, and Blue (RGB)) and depth data that represent the distance from the sensor to every pixel in the object (object point).

The main problem that this thesis deals with is how to automate the classification of specific human activities/actions through RGB-D data. The classification process of these activities utilizes a spatial and temporal structure of actions. Therefore, the goal of this work is to develop algorithms that can distinguish these activities by recognizing low-level and high-level activities of interest from one another. These algorithms are developed by introducing new features and methods using RGB-D data to enhance the detection and recognition of human activities.

In this thesis, the most popular state-of-the-art techniques are reviewed, presented, and evaluated. From the literature review, these techniques are categorized into hand-crafted features and deep learning-based approaches. The proposed new action recognition framework is based on these two categories that are approved in this work by embedding novel methods for human action recognition. These methods are based on features extracted from RGB-D data that are evaluated using machine learning techniques.

The presented work of this thesis improves human action recognition in two distinct parts. The first part focuses on improving current successful hand-crafted approaches. It contributes into two significant areas of state-of-the-art: Execute the existing feature detectors, and classify the human action in the 3D spatio-temporal domains by testing a new combination of different feature representations. The contributions of this part are tested based on machine learning techniques that include unsupervised and supervised learning to evaluate this suitability for the task of human action recognition. A k-means clustering represents the unsupervised learning technique, while the supervised learning technique is represented by: Support Vector Machine, Random Forest, K-Nearest Neighbor, Naive Bayes, and Artificial Neural Networks classifiers. The second part focuses on studying the current deep-learning-based approach and how to use it with RGB-D data for the human action recognition task.

As the first step of each contribution, an input video is analyzed as a sequence of frames. Then, pre-processing steps are applied to the video frames, like filtering and smoothing methods to remove the noisy data from each frame. Afterward, different motion detection and feature representation methods are used to extract features presented in each frame. The extracted features are represented by local features, global features, and feature combination besides deep learning methods, e.g., Convolutional Neural Networks.

The feature combination achieves an excellent accuracy performance that outperforms other methods on the same RGB-D datasets. All the results from the proposed methods in this thesis are evaluated based on publicly available datasets, which illustrate that using spatio-temporal features can improve the recognition accuracy. The competitive experimental results are achieved overall. In particular, the proposed methods can be better applied to the test set compared to the state-of-the-art methods using the RGB-D datasets.

Kurzfassung

Die Erkennen menschlicher Handlungen anhand eines Videos hat in den letzten Jahren im Rechnersehen zunehmende Aufmerksamkeit erhalten und erhebliche Fortschritte gemacht. Aktionserkennung wird als eine Voraussetzung definiert, um festzustellen, welche menschlichen Handlungen in Videos vorkommen. Die Schwierigkeiten, die bei der Unterscheidung menschlicher Handlungen auftreten, sind auf die hohe Komplexität menschlicher Verhaltensweisen sowie auf Variationen des Aussehens, der Bewegungsmuster, Verdeckungen usw. zurückzuführen. Viele Anwendungen nutzen die Erkennung menschlicher Handlungen auf von Kameras aufgezeichneten Videos, wie z.B. Videoüberwachungssysteme, Gesundheitsüberwachung, Mensch-Computer-Interaktion und Robotik.

Die auf RGB-D-Daten basierende Aktionserkennung hat in den letzten Jahren zunehmend mehr Aufmerksamkeit gewonnen. RGB-D-Daten enthalten Farb- (Rot, Grün und Blau (RGB)) und Tiefendaten, die den Abstand vom Sensor zu jedem Pixel im Objekt (Objektpunkt) darstellen.

Das Hauptproblem, mit dem sich diese Arbeit beschäftigt, ist die Frage, wie die Klassifizierung bestimmter menschlicher Aktivitäten/Handlungen durch RGB-D-Daten automatisiert werden kann. Der Klassifizierungsprozess dieser Aktivitäten macht sich die räumliche und zeitliche Struktur von Handlungen zunutze. Ziel dieser Arbeit ist es daher, Algorithmen zu entwickeln, die diese Aktivitäten unterscheiden können, indem sie niedrig- und hochrangige Aktivitäten von Interesse voneinander unterscheiden. Diese Algorithmen werden durch die Einführung neuer Merkmale und Methoden unter Verwendung von RGB-D-Daten entwickelt, um die Erfassung und Erkennung menschlicher Aktivitäten zu verbessern.

In dieser Dissertation wird der Stand der Technik überprüft, vorgestellt und bewertet. In der Literaturübersicht werden die populärsten Methoden aus dem Stand der Technik in handgefertigte Merkmale und tiefe lernbasierte Ansätze kategorisiert. Der in dieser Arbeit vorgeschlagene neue Framework zur Handlungserkennung basiert auf diese beiden Kategorien, welche durch neue eingebettete Methoden zur menschlichen Handelungserkennung angewendet werden. Diese Methoden basieren auf Merkmalen, die aus RGB-D-Daten extrahiert und ausgewertet werden unter Verwendung von Techniken des maschinellen Lernens.

Die vorgestellte Arbeit dieser Dissertation verbessert die Erkennung menschlicher Handlungen in zwei verschiedenen Teilen. Der erste Teil konzentriert sich auf die Verbesserung derzeit erfolgreicher handwerklicher Ansätze. Er leistet einen Beitrag zu zwei wichtigen Bereichen des Standes der Technik: Ausführung der vorhandenen Merkmalsdetektoren und Klassifizierung der menschlichen Handlung in den räumlich-zeitlichen 3D-Domänen durch Testen einer neuen Kombination verschiedener Merkmalsdarstellungen. Die Beiträge dieses Teils werden auf

der Grundlage von Techniken des maschinellen Lernens getestet, welche unüberwachtes und überwachtes Lernen umfassen, um dessen Eignung für die Aufgabe der Erkennung menschlicher Handlungen zu bewerten. Das *k-means Clustering* stellt die Technik des unüberwachten Lernens dar, während das überwachte Lernen durch folgende Techniken dargestellt wird: *Support Vector Machine*, *Random Forest*, *K-Nearest Neighbor*, *Artificial Neural Networks* und *Naive Bayes Classifier*, und Klassifikatoren für künstliche neuronale Netze. Der zweite Teil konzentriert sich auf die Untersuchung des aktuellen, auf tiefem Lernen basierenden Ansatzes und wie man ihn mit RGB-D-Daten für die menschliche Aktionserkennungsaufgabe anwenden kann.

Als erster Schritt jedes Beitrags wird ein Eingabevideo als eine Folge von Einzelbildern analysiert. Dann werden Vorverarbeitungsschritte auf die Videobilder angewendet, wie Filter- und Glättungsverfahren, um die Rauschdaten aus jedem Bild zu entfernen. Anschließend werden verschiedene Methoden der Bewegungserkennung und Merkmalsdarstellung verwendet, um die in jedem Einzelbild dargestellten Merkmale zu extrahieren. Die extrahierten Merkmale werden durch lokale Merkmale, globale Merkmale und Merkmalskombinationen sowie durch Methoden des tiefen Lernens wie z.B. *Convolutional Neural Networks* dargestellt.

Durch die Kombination der Funktionen wird eine ausgezeichnete Genauigkeit erzielt, die andere Methoden bei denselben RGB-D-Datensätzen übertrifft. Alle Ergebnisse der in dieser Arbeit vorgeschlagenen Methoden werden auf der Grundlage öffentlich zugänglicher Datensätze bewertet, die zeigen, dass die Verwendung raum-zeitlicher Merkmale die Erkennungsgenauigkeit verbessern können. Die konkurrierenden experimentellen Ergebnisse werden insgesamt erreicht. Insbesondere lassen sich die vorgeschlagenen Methoden besser auf den Test Datensatz anwenden als die modernen Methoden mit den RGB-D-Datensätzen.

Contents

Acknowledgements **3**

Abstract **4**

Contents **9**

1 Introduction **13**

1.1 Human Action Recognition in Videos 13
1.2 Problem Statement 16
1.3 Motivation 17
1.4 Contributions 18
1.5 Thesis Outline 23

2 Related Work in Human Action Recognition **25**

2.1 Introduction 25
2.2 RGB-D Kinect Data 26
2.3 Human Action Representation 28
2.4 Categorization of Feature Representation 30
    2.4.1 Feature Representation: Local Features 32
    2.4.2 Feature Representation: Global Features 40
    2.4.3 Feature Representation: Features Combination 44
2.5 Machine Learning 47
    2.5.1 Supervised Learning 47
    2.5.2 Unsupervised Learning 51
2.6 Deep Neural Network-Based Approaches 53
    2.6.1 Deep Learning 53
2.7 Human Activity Datasets 62
    2.7.1 MSR Daily Activity 3D Dataset 62
    2.7.2 Online RGBD Dataset 62
    2.7.3 Gaming 3D Dataset 63
    2.7.4 Cornell Activity Dataset 63

2.7.5 NTU RGB+D Dataset 65

3 Background Theory 69

3.1 Introduction 69
3.2 Motion Detection 70
3.2.1 Background Subtraction 70
3.2.2 Optical Flow 71
3.2.3 Retina Model 73
3.3 Feature Extraction Methods 76
3.3.1 Local Spatio-Temporal Features 76
3.3.2 Global Spatio-Temporal Feature 82
3.4 Video-Action Representation 89
3.4.1 Bag-of-Features 90
3.5 Video-Action Recognition 91
3.5.1 Support Vector Machine 91
3.5.2 K-Nearest Neighbors 93
3.5.3 Random Forest 94
3.5.4 Naive Bayes Classifier 96
3.5.5 Artificial Neural Networks 97
3.6 Convolutional Neural Networks 99
3.6.1 2D Convolutional Neural Networks 103
3.6.2 3D Convolutional Neural Networks 104

4 Multi-Feature Extraction for Human Action Recognition 109

4.1 Introduction 109
4.2 The Proposed System Architecture 111
4.2.1 Pre-Processing Input Data 112
4.2.2 Bag-of-Features Extraction 113
4.2.3 Action Classification 117
4.3 Experimentations and Results 120
4.3.1 Dataset Split 120
4.3.2 Parameters Setup for Local and Global Features 120
4.3.3 Results and Discussion 121

5 Feature Extraction from 3D Trajectory and Global Descriptor 127

5.1 Introduction 127
5.2 Action Recognition System Architecture 129
5.2.1 3D Trajectory 129
5.2.2 Local Feature Extraction 130
5.2.3 Global Feature Extraction 132
5.2.4 Features Combination for Robust Action Recognition 134

5.3 Action Recognition 135
5.4 Experimental Results 135
    5.4.1 Experimental Setup 136
    5.4.2 Results and Discussions 137

6 Motion Saliency Detection for Effective Human Action Recognition 141
6.1 Introduction 141
6.2 Overview of the Proposed System Architecture 143
    6.2.1 Pre-Processing Input Data 143
    6.2.2 Retina Model 144
    6.2.3 Feature Extraction 146
    6.2.4 Action Classification 148
6.3 Experiments and Results 149
    6.3.1 Experimental Setup 149
    6.3.2 Results and Discussions 149

7 Activity Recognition Based on Dense 3D Optical Flow Co-occurrence Matrices 155
7.1 Introduction 155
7.2 Proposed System Overview 157
    7.2.1 Pre-processing Input Data 158
    7.2.2 3D Dense Optical Flow 158
    7.2.3 Gray Level Co-occurrence Matrices 160
    7.2.4 Haralick Features Extraction 160
    7.2.5 Bag-of-Words 161
7.3 Activity Recognition 161
7.4 Experimental and Results 162
    7.4.1 Parameters Setting 163
    7.4.2 Experimental Results 163

8 3D Convolution Neural Networks for Human Action Recognition 167
8.1 Introduction 167
8.2 Proposed Action Recognition System 169
    8.2.1 Pre-processing Input Data 169
    8.2.2 Dense Optical Flow 173
    8.2.3 3D Convolutional Neural Networks 173
    8.2.4 Feature Extraction from 3D-CNN 175
8.3 Experiments and Results 175
    8.3.1 Implementation Parameters 176
    8.3.2 Implementation details using RGB and Depth 176
    8.3.3 Train-Test using SVM Classifier 179

8.3.4 Illustration and Comparison Results of the 3D-CNN System . . 182

9 Conclusion and Future Work 185
9.1 Key Contributions 185
9.2 Summary 191
9.3 Future Works 191

A RGB-D Action/Activity Datasets 195
A.1 Examples of Datasets 195
A.1.1 MSR Daily Activity 3D Dataset 195
A.1.2 Online RGBD Dataset 197
A.1.3 Gaming 3D Dataset 198
A.1.4 Cornell Activity Dataset 198
A.1.5 NTU RGB+D Dataset 201

B Tiny-Dnn Framework 203
B.1 General 203
B.2 CNN Construction 203
B.3 CNN Training 204
B.4 Using a Pre-Trained CNN 205

Abbreviations 207

Symbols 210

List of Tables 213

List of Figures 215

Own-Publications 219

Bibliography 221

Internet-Resources 245

Curriculum-Vitae 246

Chapter 1

Introduction

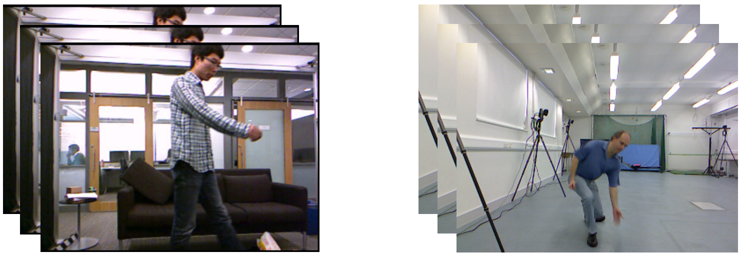
This chapter introduces and motivates the topic of this thesis. In Section 1.1, action recognition in videos is introduced. In Section 1.2, the problem statement is presented. In Section 1.3, the motivation for enhancing action recognition and its applications are presented. In Section 1.4, the objective and significant contributions to the proposed work are presented. Finally, the structure of this thesis is explained in Section 1.5.

1.1 Human Action Recognition in Videos

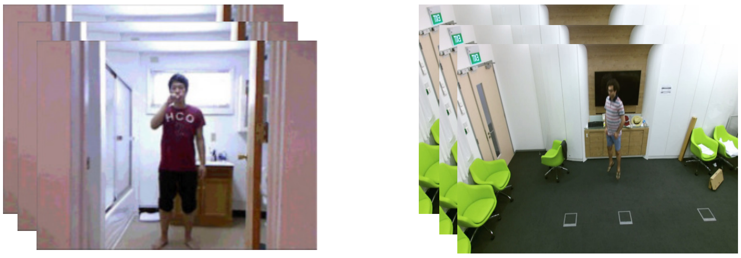
Human action and activity recognition from videos is a challenging field in real-world applications and has drawn an increasing amount of attention in recent years. Due to significant variations within a high dimension of video data, clutter background, partial occlusion, and varying motion speed, explicit action detection and recognition remain a big challenge. Efficient solutions to this challenging and difficult problem can enable several useful applications such as monitoring, visual surveillance, human-robot cooperation, and medical monitoring systems [JBCS13].

There are several types of human activities relevant to the topic of action recognition. These activities are categorized into four different levels depending on their complexity [AR11]: Gestures, actions, interactions, and group activities. Gestures are basic movements of a person's body part and are the atomic components that characterize a person's meaningful movement. An example of such a gesture is represented by *'raising hands'* and *'extending an arm'*. Actions are single-person activities or individual activities that can consist of several gestures organized in time, e.g., *'walking'* and *'bowling'*, *'brushing teeth'* and *'cleaning a sofa'*. Interactions are human activities in which two or more persons or objects are involved. For example, 'two-persons shaking-hand' is an interaction between two objects. Finally, group activities are defined as activities produced by conceptual groups composed of multiple persons such as *'two groups fighting'*and *'a group having a meeting'*.

In this thesis research, the main focus is to improve the performance and recognition accuracy of single person action and activities from real-time video sequences such as in Figure 1.1. These images in the figure are extracted from the original video sequence of the public datasets that will be used later in this thesis's experimentations.



Walking Bowling



Brushing teeth Jumping up

Figure 1.1: Action and activities video samples, such as: Walking [WLWY12], Bowling [BMA12], Brushing teeth [SPSS11], and Jumping up [SNGW16].

A video is a continuous sequence of images, with targeted semantics and a source of rich information. A video has two fundamental advantages over still images, such as the ability to maintain a consistently different set of perspectives on a scene and the ability to capture the temporal (i.e., dynamic) evolution of the event. The latest developments in distance sensors have had an undeniable influence on research and applications of machine and computer vision fields. Sensor devices provide depth information about the scene view and objects that help solve problems looking for RGB images or videos [HSXS13].

RGB-D images [LS13] are digital images that built up by pixels, i.e., a small picture element arranged in two-dimensional space. Pixels are represented by numerical values. The RGB pixels include three numerical values representing the intensity of the Red (R), Green (G), and Blue (B) colors, respectively. These values are often 8-bit numbers ranging from $0 - -255$, where higher numbers correspond to higher intensities. These RGB pixels, comprised of 8-bit values can represent $256^3 \approx 16.7 \cdot 10^6$ different colors. Figure 1.2 shows an example of RGB images.

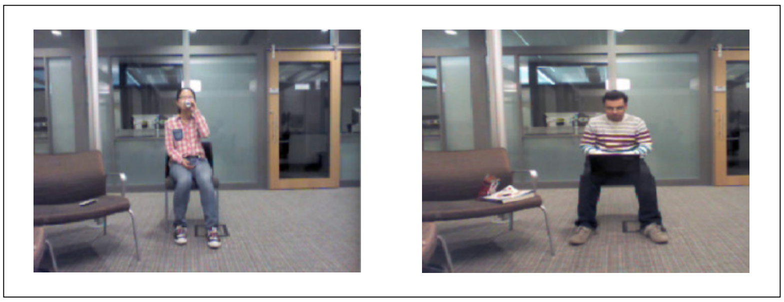


Figure 1.2: RGB images example, compare to [YLY15].

Depth (D) information is necessary for humans to recognize objects from an image, since the objects may consist of many color blocks and various texture that must occupy a continuous region in space. The depth image gives an alternative to find humans in the scene. The depth images can be recorded by using a camera, such as Microsoft Kinect [HSXS13] (see Chapter 2, Section 2.2), which is a sensory motion input device that allows users to control it and interact with it via a natural user interface that uses gestures and voice (spoken) commands. This camera uses an infrared laser projector coupled with a monochrome camera to generate the depth-map that functions under any ambient light conditions. The depth-map is in standard Video Graphics Array (VGA) resolution, i.e., which allows for resolutions higher than 640×480 pixels, such as 800×600 or 1024×768. Each pixel contains a value that represents the distance between the camera and the object in millimeters. Some depth samples are shown in Figure 1.3.

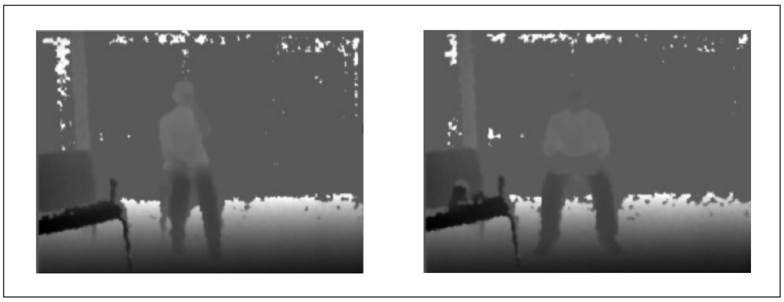


Figure 1.3: Depth images example, compare to [YLY15].

Classical action and activity recognition tasks mainly depend on hand-crafted features, which can be categorized into local, global, and motion representation approaches. The local feature extraction methods consist of two steps: Detection and description, such as the Spatio-Temporal Interest Points (STIP) [Lap05] detector, and Histogram of Oriented Gradients (HOG), Histogram of Optical Flow (HOF) [LMSR08], and Improved Dense Trajectories (IDT) [WS13] descriptors. These methods are widely used as a local feature for the human action recognition task. Local feature extraction approaches are much more efficient and robust in real scene applications. While the global feature extraction [BD00, LMB$^+$05] approaches represent the video sequence as a whole by capturing the general appearance from each frame and motion information from the differentiation between two concussive frames in video sequences, i.e., the global descriptor is used to represent the state of motion in the whole frame at a single moment in time. Even though the global approaches are very sensitive to occlusion, cluttering, and shift, they are still used commonly for human action recognition tasks [SLY16].

Regrettably, the hand-crafted features-based encoding methods, such as Bag-of-Words (BoW) [VVV16, CDF$^+$04] and Fisher vector [PD07], are represented as universal visual that does not consider much about temporal information for video-based action recognition [YCXL17]. Some methods combine different types of features and action representations to improve action recognition performance.

Additionally, in recent years, the use of neural networks and deep learning methods are showing significant progress in several fundamental problems in computer vision, including action and activity recognition. One of the deep learning methods is Convolutional Neural Networks (CNN). The CNN is utilized to solve different image processing tasks such as object and action recognition, and also, the classification is done from not only static images but also from a dynamic sequence of images. Moreover, the recent development of depth sensors has enabled the capture of useful 3D structures of scenes and objects [HSXS13]. This capture helps the vision move from 2D to 3D, such as 3D object recognition, 3D scene understanding, and 3D action recognition.

1.2 Problem Statement

This thesis focuses on the human action recognition problem in RGB and depth images/videos, recorded under various environmental conditions varying from a constant, clean background to complex, cluttered, and moving backgrounds. In the work of this thesis, a variety of different human actions have been considered from single-person activities such as *‘walking’*, *‘running’*, etc., to human activities comprising complex interaction such as *‘fighting’*, *‘hand shaking’*, etc.

This thesis work aims to address the action recognition issues by introducing new methods for feature extraction, representation, and classification to improve human action recognition performance and accuracy.

The work in this thesis motivates to address the action recognition problem from the fact that the foreground region carries more robust information about the action. This helps to eliminate background interference and makes the method more robust to background fluctuation. The remaining chapters have explained the methods that will be used in detail.

1.3 Motivation

The development of computer vision has encouraged the appearance of different novel recognition methods in both 2D images and 3D videos. Although it is still difficult and challenging to recognize a specific object from a dataset of images due to changes in viewpoint, illumination, partial occlusions, and intra-class difference. Many successful methods have been proposed, including those that are successfully extended from the image domain into video analysis and action recognition. However, current methods still need improvement, especially for real-world videos and movies, which have wide variations in people's posture and clothes, dynamic background, and partial occlusions. To conquer these deficiencies, many researchers focus on part-based approaches for which only the 'interesting' parts of the video are analyzed, rather than the whole video. These 'parts' can be trajectories or flow vectors of corners and spatio-temporal interest points. Although part-based approaches are promising, they still suffer from inaccurate detection and tracking of interesting parts due to background clutter and motion which prevents a clear and informative representation. These approaches will be discussed in Chapter 2, Section 2.4.

The ability to detect, track, recognize, and analyze human motion is helpful for a wide range of high-level applications that rely on representations extracted from visual input. During the past few years, many approaches have been proposed to address these problems [Pop10, AR11, ZS16]. Some examples of applications that could benefit from reliable and efficient recognition of human action include but are not limited to the following:

Intelligent Video Surveillance (IVS): Video surveillance is a monitoring process of persons and objects of interest with the help of video cameras. In recent years, video surveillance has attracted much attention due to the increasing demand for a security system. Security and surveillance systems are usually dependent on a network of video cameras that are controlled or monitored by a human operator who must be aware of the activity in the camera's field of vision [TZEH15]. Additionally, the surveillance system used to monitor the overall public places, such as airport terminals, government buildings, and banks.

Health Monitoring (HM): Health monitoring and preventative care system for patients have applications to accurately identify and track people in their environments and understand and analyze the activities of the patients. For example, implement a supervision system capable of monitoring a person's activity in their own home without violating intimacy. The main idea is to collect information from various sensors placed in the house and on mobile devices and infer the most probable sequence of activities performed by the supervised person [SSH16]. Usually, health monitoring systems detect the continuous movement of elderly people, automatically detect their activities, and recognize any abnormality as it happens. Therefore, recognition of human action is becoming increasingly critical and is widely used in medicine, particularly in the areas of health surveillance of the elderly.

Human-Object Interactions (HOI): Human-object interaction poses various challenges to the recognition of human activities. Dealing with the interactions between humans and objects is a challenging task, which consists of various reasons such as the size, position, shape, and color of an object [SC16].

Human-Computer Interaction (HCI): Is a way to study how people interact with computers. Today, video cameras are not only used for video surveillance, laptops and mobile phones, but they are also utilized for human-computer interaction between humans and computers and allow a natural way of human communication with a device. Thus, one of the critical requirements for the sides of this sensor is the detection of gestures and short actions recognition [LK99].

Human-Robot Interaction (HRI): Human-robot interaction is also an important application of vision-based activity recognition. HRI gives a robot the ability to detect human activities that is very important for HRI. One of the HRI applications is presented by Xia et al. [XGAR15], recognizes interaction activities from a robot-centered viewpoint area. Their procedure helps the robot realize the following environment tasks in order to detect the intention of the people around them.

This thesis has used several challenges that are implemented and applied on publicly available datasets designed for human action recognition, which are still very challenging in this field and highlight the large ongoing scope for improvement.

1.4 Contributions

One of the biggest problems with action recognition (AR) is performing a similar action in several different ways by another person or even the same person. So, the key concern is *how to find a discriminative representation of actions between different RGB-D video sequences?* In other words, finding a way to separate different actions and classify together videos with similar actions performed by different people in pose and motion.

Action recognition remains a challenging problem due to differences in visual and moving images of people and actions, camera perspective, noise, occlusions, and significant video data.

This thesis research improves feature extraction methods related to scalable action recognition by using different algorithms and provides an extensive experimental evaluation of comprehensive empirical evaluation, often outperforming state-of-the-art techniques. An overview of the main objective of this thesis are listed below:

- The first objective is to display and compare the existing local and global feature-based methods to determine which local or global feature extraction method and what is the video representation technique give good results. Also, obtain a feature representation for the CNN model from RGB-D input data.
- The second objective is to review and combine the existing local and global feature extraction methods to obtain the best feature values that work with the algorithms of the video representations and provide the best results in addition to the deep neural network.
- The third objective is to study and understand the limits of existing methods and to suggest new methods for human action recognition in videos to go further than the state-of-the-art limitations.

This thesis aims to tackle human action recognition challenges by using hand-crafted features (as in Chapters 4 to 7) and deep learning-based approaches (as in Chapter 8). Figure 1.4 shows the general structure of the proposed action recognition approaches. In the hand-crafted features, complementary pattern recognition pipeline levels are implemented, mainly input data, feature extraction, and classification. The deep-learning approach can evaluate the input data by seamlessly integrating these three pipelines parts, or it can be used as a feature extractor that is finally evaluated using classification methods. The main contributions of this thesis can be summarized as follows:

- It introduces a great deal of challenging research by representing the spatial and temporal structure of actions from color and depth data (RGB-D) based on different spatio-temporal feature extraction methods.
- It provides insights into new features for the detection rates of actions in RGB-D video sequences. These features are computed using the hand-crafted feature extraction methods.
- It classifies video sequences on which people perform actions into predefined action types. The classification is done based on machine learning techniques.
- It provides a contribution to the question of how RGB-D videos can be processed with deep neural networks, especially CNN.

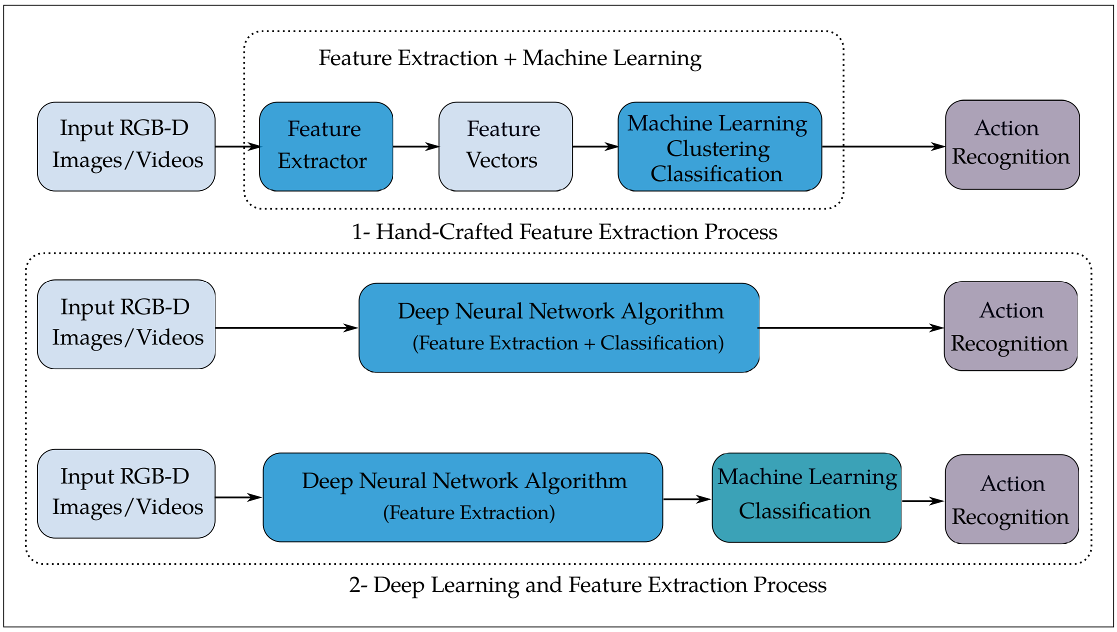

Feature Extraction + Machine Learning
Input RGB-D Images/Videos
Feature Extractor
Feature Vectors
Machine Learning Clustering Classification
Action Recognition
1- Hand-Crafted Feature Extraction Process
Input RGB-D Images/Videos
Deep Neural Network Algorithm (Feature Extraction + Classification)
Action Recognition
Input RGB-D Images/Videos
Deep Neural Network Algorithm (Feature Extraction)
Machine Learning Classification
Action Recognition
2- Deep Learning and Feature Extraction Process

Figure 1.4: The general structure of the proposed action recognition approach. 1- Feature extraction based on hand-crafted features and machine learning process (in Chapters 4, 5, 6, and 7). 2- Deep learning and feature extraction process (in Chapter 8).

To achieve these contributions, this thesis proposes new approaches of different feature extraction and combination methods to improve robust and efficient representations for human action recognition from RGB and depth videos. The following is a summary of the significant contributions:

1. **Multi-Feature Extraction for Human Action Recognition** (Chapter 4):

   This chapter proposes a novel system to analyze human body motions for recognizing human actions by using 3D videos (RGB and depth data). As a first, given a set of input RGB-D videos. The Bag-of-Features (BoFs) technique is used for recognizing human actions by extracting spatio-temporal features from all video sequences. In this work, the multi-features are computed using different feature extraction techniques. The local features are computed in two steps: The first step consists of detecting all interest keypoints from RGB video frames by using Speed-Up Robust Features (SURF) [BETV08] detector; then the motion points are filtered by using Motion History Image (MHI) [RTKI11] and Optical Flow (OF) [TCL15]. Furthermore, these important motion points are aligned to the depth frame sequences. In the second step, the feature vectors are computed by using a Histogram of Orientated Gradient descriptor [DTS06], this descriptor is applied around these motion points from both RGB and depth channels, then the feature vector values are combined in one RGB-D feature vector.

In addition to the local features computed first, another type of feature is computed and combined with local features. Seven Hu-moment shape features are extracted from the MHI. The Hu-moments shape invariant introduced by [Hu62] are used for global features and illustrated in two steps: Firstly, representing the spatial and temporal information about action using MHI, where the pixel density is a function of the redundancy of action. Secondly, the Hu-moments are used as descriptors of the MHIs.

This system performance is evaluated by using the BoW pipeline and classification methods. This system is invariant to scale, rotation, and illumination. All tested results are computed from datasets available to the public and often used in the community. This new feature combination method helps to reach recognition rates superior to other publications on the same dataset.

This work was presented at the International Conference on Machine Vision (ICMV) and published as a journal article in the International Journal of Machine Learning and Computing (IJMLC) because it is a novel work [AAP18c]. *The second part of this work was published at the International Journal of Advanced Computer Science and Applications (IJACSA)* [AaP17].

2. **Feature Extraction from 3D Trajectory and Global Descriptor** (Chapter 5): This chapter contribution aims to address the problem of recognizing human behavior by using a novel method based on combining 3D Trajectory shape features from RGB-D video frames [XZYT14], Motion Boundary Histogram (MBH) [WKSL13] descriptor and global GIST feature descriptor [WLJ14] from depth data.

   In this system, the combination of the local and global descriptors is proposed. The local descriptors are represented by 3D Trajectory and MBH. These descriptors represent a video as features extracted from a collection of patches. The global descriptor is represented by global GIST [WS07, WM10]. The main reasons for using the global GIST descriptor is illustrated in three steps [WLJ15]: First, the GIST feature descriptor caught global structural information through filtering an image with different scales and orientations. In realistic scenarios, it can be extracted more accurately than the silhouettes feature. Second, the computational time of the global GIST feature descriptor is less than the optical flow features. Third, the GIST feature can be represented as the concatenation of several local grids with implicit location information.

   The BoW model and classification pipeline are used for the evaluation of the method in this system. This new method represented by combining local and global features from several video actions improves performance on actions even with low movement rate. Furthermore, it outperforms the competing state-of-the-art feature-based human action recognition methods.

This work was presented at the International Conference on Pattern Recognition Applications and Methods (ICPRAM) and published by SCITEPRESS – Science and Technology Publications [AaP18b].

3. **Motion Saliency Detection for Effective Human Action Recognition** (Chapter 6): A novel approach applies in this chapter to show the advantages of using a Retina model [BCDH10] for saliency motion detection. With the help of this model, an efficient and fast bio-inspired module is developed for low-level image and video processing. On the Retina level, spatio-temporal filtering ensures correct structuring of the video data, i.e., removal of noise and illumination variation, static and dynamic contour enhancement, and motion event detection. This approach adopts the ideas of spatio-temporal analysis and global features extraction for effective action recognition from RGB-D video actions. Global features have been used to characterize the texture information from body motions on both RGB videos and depth data using Local Binary Pattern (LBP) descriptor [OPH96]. The LBP features are computed from the salient motion area of the Retina model, and feature vectors are finally collected as a histogram of LBP.

   For improving the performance of this implementation, the k-means clustering and Random Forest (RF) for classification are used to recognize the different actions from videos. This chapter approach is demonstrated that the proposed system achieves superior performance compared to the state-of-the-art methods, and all experimental results are tested on three public RGB-D datasets.

   This work was presented and published at the Conference on Computer Graphics, Visualization, and Computer Vision (WSCG) [AaAdP18].

4. **Activity Recognition based on Dense 3D Optical Flow Co-occurrence Matrices** (Chapter 7): In the contribution of this chapter, two important goals are presented. The first one aims to extract the motion and texture features from 3D sequences, by extending the method in [CCS16] to extract feature vector values from RGB-D video actions instead of RGB video. This method based on the Gray-Level Co-occurrence Matrix (GLCM) of the dense optical flow [BFB94] pattern and the well-known Haralick features [HSD73]. Haralick features are extracted from these matrices by measuring meaningful properties such as Energy, Contrast, Homogeneity, Entropy, Sum Average, and Correlation to capture local spatial and temporal characteristics of the motion through the neighboring optical flow field. The second goal is represented by a performance comparison of five different classifiers such as Artificial Neural Network (ANN), Naive Bayes classifier (NB), Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). These classifiers results are computed from BoW vectors.

   This work was presented and published in two events, the first part of this work was presented and published in the International Workshop on Sensor-based Ac-

tivity Recognition and Interaction (iWOAR) [AAP18a], and the second part was presented and published in the International Conference on Machine Vision (ICMV) [AAP19].

5. **3D Convolution Neural Networks for Human Action Recognition** (Chapter 8): This chapter proposes a novel 3D Convolutional Neural Network (3D-CNN) system that implicitly captures motion information between adjacent frames, and it is represented in two main steps. First, the optical flow is utilized to extract motion information from temporal domains of the various RGB-D video actions. This information is utilized to calculate the features vector values from a deep 3D-CNN model. Second, use the feature vectors to train and evaluate a 3D-CNN from three channels of the input video actions, i.e., RGB, depth, and combine information from both channels (RGB-D), to obtain a feature representation for a 3D-CNN model.

   This 3D-CNN model generates multiple information channels from adjacent video images and performs convolution in each channel separately. The final results of feature representation are achieved by the combination of information from all channels. For evaluating the accuracy results, a 3D-CNN based on different data channels is trained. Additionally, the possibilities of feature extraction from 3D-CNN and classification by using an SVM classifier are proposed to improve and recognize human actions. This system is demonstrated that the evaluation of results from RGB-D channels is better than each channel results that are trained separately by 3D-CNN network and also outperformed the state-of-the-art on the same public datasets.

   This work was presented and published at the Conference on Computer Graphics, Visualization, and Computer Vision (WSCG) [AaPG18].

1.5 Thesis Outline

This thesis consists of 9 chapters, including the current introductory chapter of the thesis. Chapters 2 and 3 introduce the overview and theoretical background of the field of articulated human feature representation and recognition. Moreover, the evolution of human action recognition is presented to provide an introduction to different approaches, different features extraction, representation, and classification techniques used by researchers over the last few years. In addition, it includes a comprehensive review of popular, challenging datasets. From the presented material in these chapters, a basis for the methods used in this thesis is developed.

Chapters 4 to 8 describe proposed methods for improving human action and activity recognition based on motion detection, different feature extraction and representation methods, and how to improve the performance of these methods based on different classification algorithms, in addition to the use of deep neural networks. All thesis results are presented and already published in international journals and conferences. Each chapter will start with a brief introduction explaining the context of the chapter and its contribution and the experimental results.

Chapter 9 summarizes and concludes the thesis and presents the realization of each objective. Moreover, it also presents directions for future work.

After the main part of the thesis, there are the appendices. The back matter consists of a definition of all abbreviations and symbols, the bibliography, and references to internet-resources.

Chapter 2

Related Work in Human Action Recognition

This chapter presents a review of the state-of-the-art techniques for human action recognition based on hand-crafted features, machine learning, and deep neural network-based approaches. Furthermore, well-known public RGB-D datasets available for experiments will be presented to provide further insight into this field.

This chapter starts with an introduction in Section 2.1. Section 2.2 illustrates the categorization of RGB-D Kinect data. Action representation is illustrated in Section 2.3. Categorization of feature representation schemes is covered in Section 2.4. Sections 2.5, 2.6, and 2.7 define machine learning techniques, deep neural network-based approaches, and the human activity datasets.

2.1 Introduction

Human action and activity recognition is a relevant field of research for pattern recognition and computer vision applications such as robotics, human-computer interaction, monitoring, etc. The goal of the human action recognition system is to automatically analyze the actions of a person or a group of persons from a video, i.e., a sequence of images. In a simple case, where the video is split to contain only one execution of human action, the system's objective is to classify the video into its action category correctly.

Human action recognition-based systems are widely used in many real-world applications due to the progress in sensor and visual technology. Precisely, the increase of small size sensors has enabled smart devices to recognize daily human activities, actions, and interactions [YLM14]. Sensors translate a variety of parameters, such as displacement, location, movement, camera images, and color, from the real-world into data for the digital domain.

Action recognition is still a challenging task with videos even for the same type of action, due to complicated significant variations which make robust information extraction difficult. These difficulties include: First, the subject under observation distinguishes in appearance, position, and size. Second, the moving background, unstable camera, occlusion, and complex environment obstruct the observation. A wide-ranging study of different aspects and problems in the field of human action recognition from RGB videos has been included in several recent research papers [NJ14, WLJ15, CCS16, AEV17, UL19]. A variety of existing action recognition methods, such as low-level feature extraction and high-level feature representations, use algorithms which were initially designed for text and image domains and expand them over a video domain.

Effective human action recognition systems have balanced between the recognition reliability and feature extraction efficiency from the computational cost viewpoint. To achieve this objective, most researchers try to find a robust and reliable method to extract features and efficient classification algorithms.

In this thesis, the human action detection and recognition will be improved in 3D sensor data. These data are recorded using an RGB-D sensor.

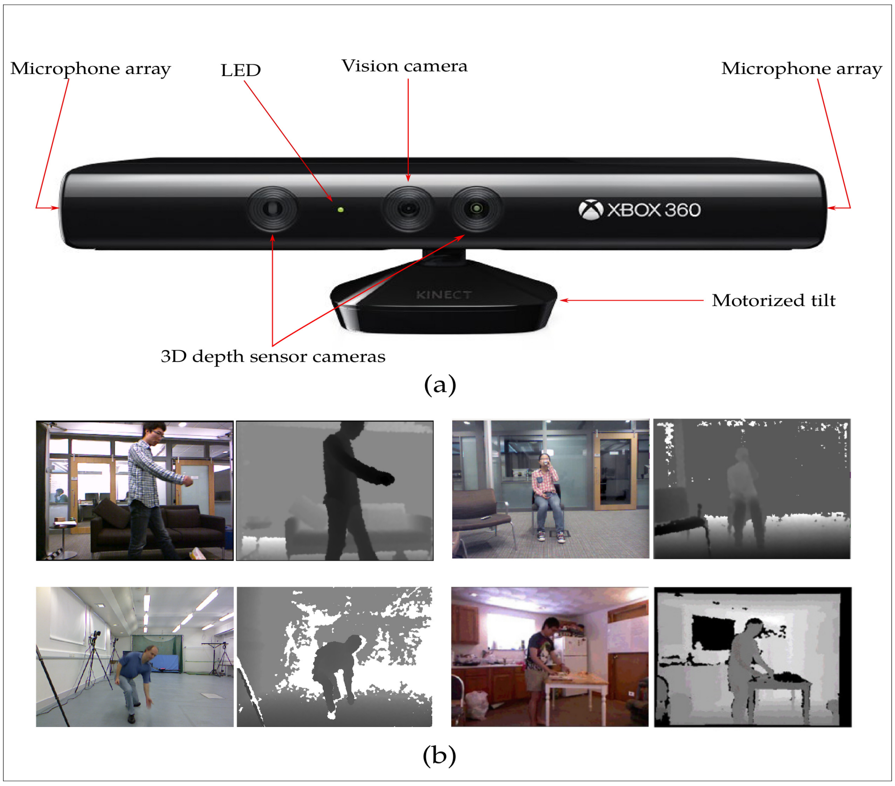
2.2 RGB-D Kinect Data

In recent decades, there were many types of computer vision researches focused on RGB images [CCFC13, MA16]. However, RGB images typically provide the objects in the scene with only appearance information. With this restriction in the information provided by RGB images, some problems such as splitting the background and foreground with similar colors or textures are difficult to overcome. Furthermore, the object represented by RGB images does not appear robust against common differences such as illumination changes, which makes the use of RGB image processing algorithms ineffective in practical situations. However, most researchers struggle to design more sophisticated algorithms; the other parts of the researches emphasize a new type of representation that can perceive the scene [CHLS17].

Yet, RGB-D images/videos are emerging data representations that can help solve fundamental problems due to their complementary nature of depth information and visual RGB information. Concurrently the combination of RGB and depth information can dramatically improve the classification accuracy for high-level tasks, i.e., image/video classification [TJYL13, TCLY15]. The core of the RGB-D images/videos is the depth image, which is usually generated by a distance sensor. Compared to a 2D intensity image, a distance image is robust to the variations in color, lighting, rotation angle, and scale. When this technology was established, sensors were expensive and hard to be obtained by researchers, which led to limited researches covering this topic at that time. However, with the release of the affordable Microsoft Kinect 3D sensor in November 2010, the purchase of RGB-D data became cheaper and more accessible.

Not surprisingly, research into computer vision algorithms based on RGB-D data has attracted much attention in recent years [CLV12, CZG+13, NVSH16, LAM17, SHJ18].

Kinect, as a 3D digital capturing device, can quickly collect the RGB and depth data of human activities. The Kinect device has a color camera, an Infrared Emitter (IR), and a depth sensor. This camera is capable of capturing colored images and depth information of each pixel in the scene. These capturing data contain visual and geometric information of the scene, which can improve activity recognition in accuracy and robustness. Examples of Kinect device and RGB-D images are shown in Figure 2.1.



(a)

(b)

Figure 2.1: Illustration of 3D camera and RGB-D images: a) Microsoft Kinect Device, compare to [TDL15]; b) Some examples of RGB-D images is captured by Kinect, compare to: (1) [WLWY12], (2) [BMA12], (3) [SPSS11], (4) [YLY15].

The color images that are captured by the Kinect are available in various resolutions and formats. The resolution of the images affects the data amount per frame. The format of the images determines the data type of the image's color, whether it is encoded as RGB or gray.

The depth information of the images calculates the distance to the closest object in millimeters at that precise spatial coordinates in the depth sensor's field of view. The depth image is captured in three different spatial resolutions: 640×480, 320×240, and

800×600 depending on the specified image format. This data can be used to track the movement of a person and background segmentation.

The infrared emitter (IR) beams an invisible infrared light, which is generated by a depth sensor determining objects' depth distance from the sensor. The key objectives of this IR stream are to improve external camera calibration [SJP13, Kar15] using a test pattern from both the RGB and IR camera. It is also used for calculating precise coordinates from one camera space to others and also capturing the IR image in darkness.

With the availability of the required tools to easily produce RGB-D images/videos, a wide range of applications can be served, e.g., computer vision, robotics, and medical imaging [HSXS13]. As several algorithms are proposed to solve the technical challenges in these fields, the other components generated to verify the algorithms have been compiled by an increasing number of RGB-D datasets. The use of publicly available RGB-D datasets can save researchers time and resources and allow a fair comparison of various algorithms. Therefore, the chosen of RGB-D datasets becomes important for the evaluation of different algorithms. RGB and depth data combine the characteristics of the color images that provide visual appearance information of an object in the image and 3D shape information. The depth image is insusceptible to the variations in color, rotation angle, scale, and illumination. With the low costs of 3D visual capturing systems, more and more researchers have been concentrating on a possible solution to recognizing human behavior by using 3D data [BMM12, NVSH16, MH17, SHJ18].

Although the Kinect sensor does neither provide accurate distance values nor has a high image quality, it is used in many research teams as it is common and quite affordable. More expensive sensors exist, such as Photonic Mixing Device (PMD) [LKH07], and Time-of-Flight (ToF) camera [FAT11], which provide better quality. The general principle is similar, though.

2.3 Human Action Representation

The human action recognition system becomes an essential part of many computer vision applications. It can be classified based on human action representation methods, see Figure 2.2, such as feature representation methods [Pop10, VNK15, ZWN^{+}17], action classification using the traditional machine learning techniques [ROD01, Kot07], and deep neural networks-based method [YCBL14, Lat17].

Concerning feature representation, the human action recognition system categorized into three classes depending on the method of representing or extracting the features from video data: local features [Low04, DTS06, KMS08, NY10, ZS16], global features [MAQM05, PSL09, SCMP14], and combination of the features, i.e., a combination of local and global features [QMXW10, ZLYC12, WLJ15]. These representations will be defined below in Section 2.4.

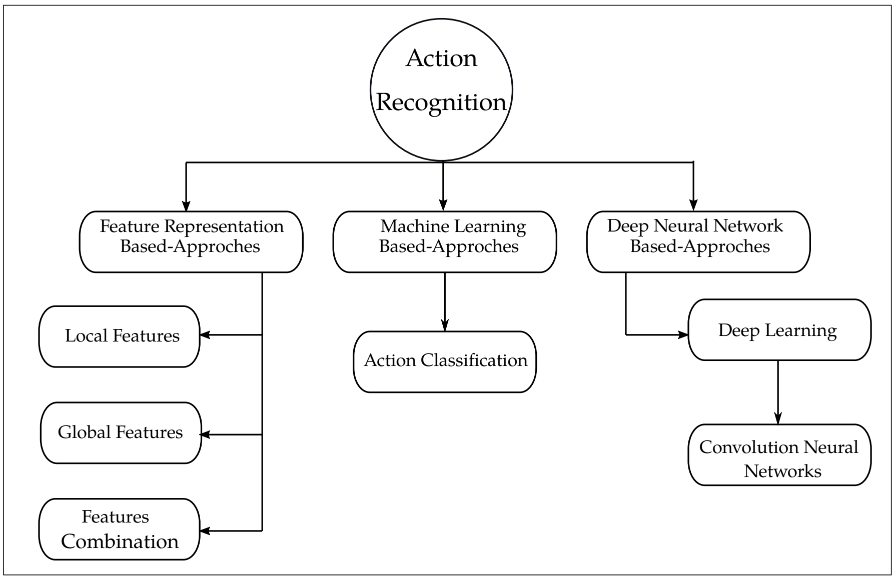

Action Recognition
Feature Representation Based-Approches
Machine Learning Based-Approches
Deep Neural Network Based-Approches
Local Features
Global Features
Features Combination
Action Classification
Deep Learning
Convolution Neural Networks

Figure 2.2: Human action representation.

The action classification is important in recognizing human behavior because the extracted features must be classified accordingly. The increasing number of video actions will lead to more challenges due to the higher overlap between classes. The classification based on traditional machine learning algorithms has two stages:

- Training stage: In this stage, a machine learning algorithm trains using a dataset comprised of the images/videos. The classification problem in this stage has two main steps: Firstly, feature extraction from input images/videos to get new feature vectors. Secondly, training the feature vectors by using the machine learning algorithm.

- Testing (predicting) stage: In this stage, the trained model utilizes to predict labels of unseen images/videos.

In addition to traditional machine learning, the recent deep learning-based approaches use large amounts of available training data. They are the recently developed approaches that can automatically learn features from the raw data in order to achieve human action recognition.

The difference between machine learning and deep neural network algorithms is in feature extraction. In machine learning algorithms, the hand-crafted features are needed to compute expert-designed feature detectors and descriptors [SAH17, UL19]. By contrast, in deep neural network algorithms, feature extraction is done automatically by the algorithm.

In the following sections, the categorization of feature representation techniques, machine learning, and deep neural networks will be explained briefly.

2.4 Categorization of Feature Representation

Different types of features can be represented from video data based on the characteristics of the feature representation. Each video data could be represented as:

$$f(x, y, t) \in V, \quad V : \mathbb{R}^3 \rightarrow \mathbb{R}^{n_v} \tag{2.1}$$

where, n_v represents the type of the video data, and it can be:
$n_v = 1$ refers to a gray-level image.
$n_v = 3$ refers to (R, G, B), i.e., color image.
$n_v = 4$ represents (R, G, B, D), where D refers to depth or distance.

In this work, video data can be regarded as a function f:

$$(R, G, B, D) = f(x, y, z) \tag{2.2}$$

Figure 2.3 shows an example of video sequence representation from RGB and depth sequence.

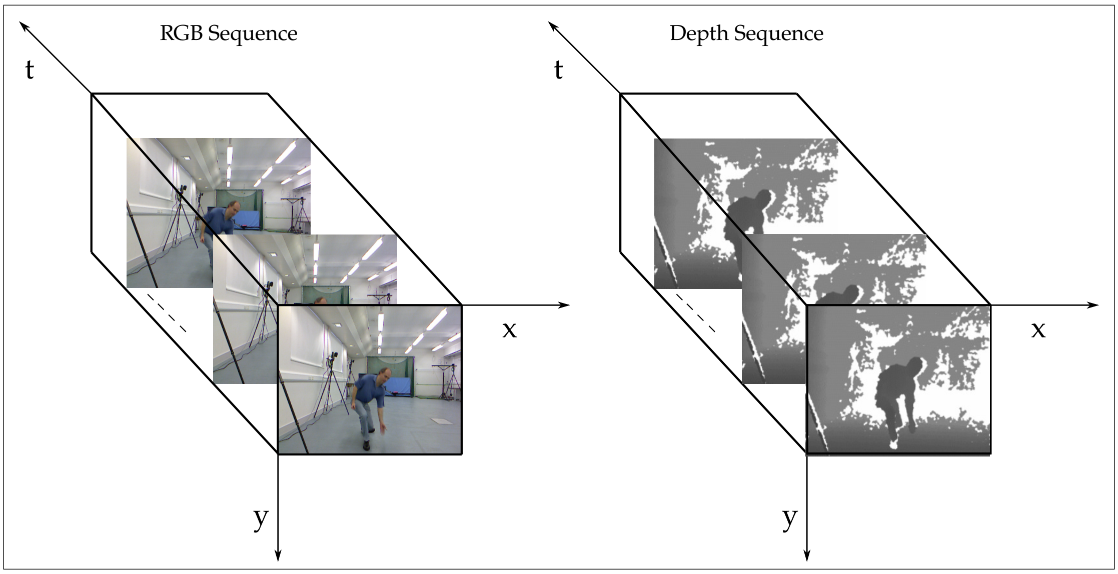


Figure 2.3: RGB and depth video sequence representation. x indicates the horizontal direction, y indicates the vertical direction, t indicates the time direction.

As mentioned previously in Section 2.3, the feature representation methods are classified into local and global features in addition to feature combinations (see Figure 2.4).

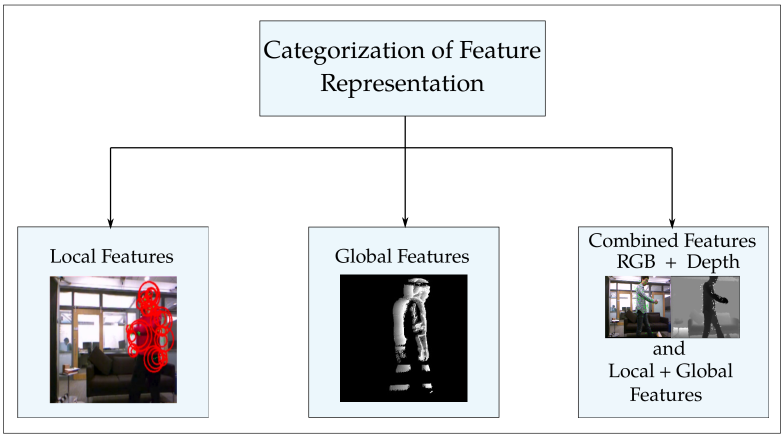

Categorization of Feature Representation
Local Features
Global Features
Combined Features RGB + Depth
and
Local + Global Features

Figure 2.4: Categorization of feature representation.

Global feature approaches generally extract the whole human body information. Then a global body structure and dynamics are utilized to represent human actions. In many cases, global approaches employ silhouette information or shape masks that are extracted from background subtraction or difference between images to represent actions, for instance, silhouette-based feature [DTGÇ06] and Motion History Images (MHIs) [BD01]. Formally, global features could be represented as G_F:

$$G_F : f \rightarrow \boldsymbol{v} \tag{2.3}$$

where $\boldsymbol{v}$ is a vector of real numbers, i.e., a feature vector.

Local feature approaches are represented by extracting local sub-regions or interest points from image/video sequences. Local features can also be edges or small image patches. In many cases, some measurements are usually taken from an area centered on a local feature and converted into descriptors. These descriptors are used to describe the local property, such as Scale-Invariant Feature Transform (SIFT) [Low99], Speeded up Robust Features (SURF) [BETV08], Trajectory [RS99], Histogram of Oriented Gradients (HOG), Histogram of Optical Flow (HOF), and Motion Boundary Histogram (MBH) [UDSS15].

In other words, the local features (L_F) select a region of the range $\mathbb{R}$ of f. Formally, a set of s local features extracted from an image or video could be represented as L_F:

$$L_F = \{\boldsymbol{v}_i\}_{i=1}^{s}, \quad \boldsymbol{v}_i \in \mathbb{R}^{n_d} \tag{2.4}$$

where, n_d is the number of dimensions. Typically, the local features can be represented as:

- A spatial region [Low04].
- A temporal region (time span) [JDX$^+$12].

- A projection of the vector space V to a selection of the dimension [LPB11].
- A combination of all these regions [CZG$^+$13].

In addition to the combination of the local regions, the other combination may be happened between local and global features for improving the new feature representation method [LMB$^+$05, NJ14, WLJ15]. The representation of features will be reviewed in the next following sections.

2.4.1 Feature Representation: Local Features

Local representation, also defined as local methods, encodes a video sequence as a set of local spatio-temporal features, i.e., local descriptors. These descriptors are extracted from spatio-temporal interest points (STIPs) that can be sparsely detected from video sequences by using detectors [DRB05, Lap05, BETV08].

Local features extracted from local areas in an image/video sequence are used to represent the local structure of a sample. Local areas have usually described the neighborhoods of points, which are determined by using an interest point detector or by a dense sampling of the image plane or video volume. Next, a feature vector is computed for each local area by describing its properties.

To encode video data as a local feature, interest points are first detected in a video frame and then describe them adequately to capture video information without needing to separate the objects from the background. Recent work has concentrated on making these new local features invariant to image transformation to be used in different applications such as a representation of objects, motion tracking, image alignment, 3D reconstruction, object recognition, and robotics. There are four characteristics when local features are used in these applications [JC13]:

- The areas around the point of interest must be localized in position and scale. Generally, points of interest in a scale-space search will be placed at local peaks and then filtered. The kept points are likely to remain stable against transformation.
- A description of these areas should be built. In an ideal condition, the description should be distinctive (reasonably differentiating an area around a point of interest) and robust to transformations caused by camera pose and lighting changes.
- To eliminate irrelevant and redundant local features, the feature space should be optimized to simplify the architecture of the classification system and enhance predictive performance as well as computational efficiency.
- A representation model should be built in a way that can describe the objects based on appropriate components and explicitly differentiate the objects from others.

Furthermore, the representation of local features characterizes the observation as a combination of local descriptors or patches. Spatial and temporal interest point detectors are used to detect the local patches first. Then, the local patches are combined to construct the sequence using the Bag-of-Features approach, which will be covered in more detail in Chapter 3, Section 3.4, for representing video actions. The advantages of local features representation are [Low99]:

- **Locality**: The features are represented as local, so they are robust to clutter and occlusion (no prior segmentation).
- **Distinctiveness**: Individual features can be compared and differentiated with an extensive database of objects.
- **Quantity**: Many features can also be created for small objects, i.e., there are hundreds or thousands of features in a single image.
- **Efficiency**: The computed features are close to real-time performance.
- **Extensibility**: Can easily be extended to exploit a wide range of different types of features in different situations, with each adding robustness.

Over the last years, feature detectors and descriptors computed from static and dynamic scenes of images are an active area of research, making them a popular topic in computer vision applications. The feature detectors and descriptors methods are part of the covered research area in this thesis.

Feature Detectors

Features of images typically refer to the exciting part with meaningful information for the task of computer vision. They can generally be classified as low-level features and high-level features that come into being with the appearance of more computational problems and time constraints. In low-level feature representation methods, the essential step to the feature extraction from a video is to detect interest points. These points are regarded as more informative than others and are described using feature descriptors. To achieve this, various algorithms for identifying points of interest regions in the video have been developed by researchers, for instance:

Noguchi et al. [NY10] extracted visual and motion features by selecting candidate points using the SURF detector (see Chapter 3, Section 3.3.1). Afterward, the motion features at each point are subdivided with local temporal units to take into account the sequence of the movements and propose a spatio-temporal feature. When the local spatio-temporal features from training video data are extracted, k-means clustering (see Section 2.5.2) is applied to generate a codebook from the extracted features. Then for each training video, a BoW vector is generated based on the codebook. Finally, they classified human action by train an SVM with the generated BoW vectors. The experiment results were evaluated using RGB videos.

Yang et al. [YTYC12] proposed the SURF-MHI method to extract spatial and temporal information separately from RGB videos. They followed the method proposed by Tian et al. [TCLZ12]. The SURF detector is first applied to extract visually distinctive points in the spatial domain. Then, The extracted SURF points are filtered by temporal (motion) constraints from the MHI (see Chapter 3, Section 3.3.2) that is generated by differencing adjacent frames. To characterize shape and motion information, HOG features (see Section 3.3.1) were computed for each interest point from images and MHI channels. HOG descriptor can be well used to characterize local shape information from the image channel and local motion information from the MHI channel by calculating local gradients' distributions. Then, BoW vectors were generated based on the codebook computed by k-means clustering. Finally, a linear SVM was applied for evaluation.

Benoit et al. [BCDH10] showed efficient modeling of the processing of the Retina model along with illustrating the advantages of using such modeling for a low-level image. The Retina model will be discussed in detail in Chapter 3, Section 3.2.3. At the Retina model, double spatio-temporal filtering occurs and ensures the proper structure of video data, i.e., static and dynamic contour enhancement, noise, and illumination variation removal. The Retina model is designed for use in all contexts where contours and contrasts represent important information. As it can be used on either a standard camera or any other type of image, e.g., infrared, Xray images, given that luminance information is considered the input data. As an illustration from this research, the Retina model can be used as a generic pre-processing step to improve the input data, and it can be a very suitable solution for any application that requires the extraction of low-level features to enhance the detection potential.

Sabin et al. [SBL14] used a Retina model for extraction BoW to introduce the video pre-processing strategies, see Figure 2.5. This Retina model is also used to detect the salient areas from video frames and to construct spatio-temporal descriptors. This pre-processing strategy helps to increase the robustness of local features such as noise and lighting variations. The results of the Retina model in this work were experimented using an RGB dataset. These with state-of-the-art local features, such as SIFT, SURF, and FREAK. Finally, the KNN classifier was applied due to it is appropriate for comparing different video descriptors.

Feature Descriptors

So-called 'descriptors' are used to describe the physical aspects of image details, such as shape, size, coloration, a direction of change in color or intensity, and amount of motion. The role of the descriptor is to characterize the local image details around the location, i.e., in a local neighborhood surrounding interest points, identified by a feature

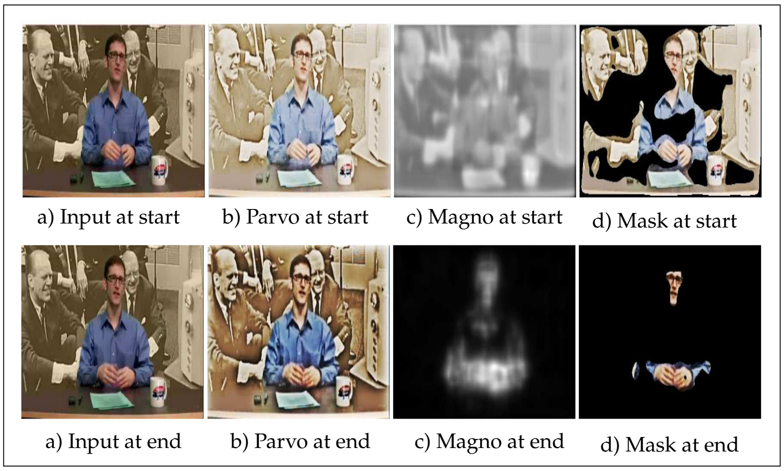


Figure 2.5: Effects of Parvo and Magno preprocessing and segmented "blobs" of low-level spatio-temporal saliency at the beginning and end of a 20-frame temporal window around the keyframe [SBL14].

detector. The feature detector locates spatio-temporal interest points in the image or video sequence; such interest points contribute the maximum to the context of image or video. Spatio-temporal interest points detector is operated on a stack of images denoted by $I(x, y, t)$, where I is an input image that mapped either RGB or depth [JS10]. To capture more information and features from detected interest points, a description around these interest points is needed to represent and encode the video sequence. Therefore, these local descriptors proposed to describe local interest points, each of which is calculated at the location (x, y, t). Afterward, a local patch, which is described to represent the actions, is considered around each detected interest point. Different descriptors used in human action recognition applications will be introduced in this section.

Many proposed approaches are implemented to extract and describe robust information from video by using feature descriptors that are adapted successfully from the image domain into the video domain toward enhancing the accuracy of human action recognition.

Dalal et al. [DTS06] proposed a detector that combines gradient-based appearance descriptors, which is represented by HOG with the differential optical flow (OF) (details of OF will be explained in Chapter 3, Section 3.2.2) based motion descriptors in a framework of linear SVM classifier. Both motion and appearance channels use histogram orientated voting to obtain a robust descriptor. The HOG descriptor is used to capture edge distributions in images or video frames that allow dense sampling of different scales and locations in the spatio-temporal representation. The calculation of the interest points descriptor based on Lowe [Low04]. The interest points are generated by calculating the image gradient magnitudes and orientations in an area (region) around the interest point locations at each image sample points. Image magnitudes are sampled around the interest point locations by using the scale of the interest points to select the level of Gaussian blur for the image, i.e., removes high-frequency components from the image (low-pass filter) using Gaussian filter. To achieve an orientation invariance, the descriptor coordinates and the gradient orientations rotate relative to the interest points orientation. The Gaussian window, which is indicated by an overlapping circle, weighs these points. Then the output samples are combined into orientation histograms that summarize the content over subareas.

Uijlings et al. [UDSS15] addressed the problem of computational efficiency by improving video classification in three steps. Firstly, they proposed several speed-ups for densely sampled based on local feature descriptors such as a Histogram of Oriented Gradients (HOG), a Histogram of Optical Flow (HOF) and Motion Boundary Histograms (MBH) descriptors, these descriptors will be discussed later in Chapter 3, Section 3.3.1. Secondly, they explore the trade-off between efficiency and computational performance of descriptors in terms of frame sampling rate and the nature of the optical flow (OF) method. Finally, they explore the trade-off between efficiency and computational performance for the video representation based on comparison of the vector quantization methods such as k-means clustering, hierarchical k-means based visual vocabulary, i.e., partition the dataset recursively into a tree of the cluster with k branches at each node and RF-based vocabulary.

Kellokumpu et al. [KZP08] described the human action recognition by utilizing the dynamic texture feature descriptors. These features are used for human detection to extract Local Binary Pattern from Three Orthogonal Planes (LBP-TOP) features from the spatio-temporal domains of the input images. Moreover, these features are also used to detect volumes of human boundaries and to characterize human movements. The detection method can find human regions in spatio-temporal data. The activity classification experiments are performed on the RGB images using Hidden Markov Models (HMMs).

Xia et al. [XA13] presented a filtering method to extract STIPs from depth videos (DSTIP) as shown in Figure 2.6, which effectively reduces the noisy measurements. Additionally, the researcher built a local depth cuboid similarity feature (DCSF) to represent the 3D depth cuboid throughout the DSTIPs with an adaptable supporting size. A cuboid codebook is built by clustering the DCSF using the k-means algorithm with Euclidean distance. The center of the clusters defines the spatio-temporal codewords, and each feature vector is assigned to a codeword using Euclidean distance. Thus, each depth sequence can be represented as a bag-of-codewords from the codebook. Finally, the SVM classifier is used for computing the performance of the action recognition system from depth data.

Local descriptors can also be obtained from Trajectories. Feature Trajectories are one of the effective representation methods for video data. Therefore, Trajectory approaches indicate that it is appropriate to identify human actions by tracking joint positions and explicate an activity as a set of space-time Trajectories [MS02, JDX$^+$12].

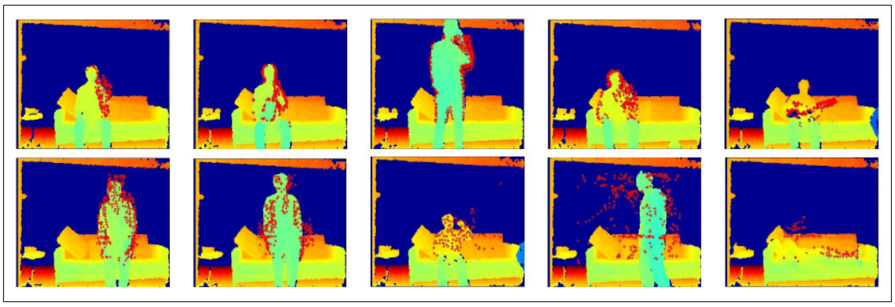


Figure 2.6: Example of extracted DSTIPs that are projected on to (x, y) coordinate with one depth frame from the video. Action type arranged from left to right, up to down: *Drinking-sitting, eating, drinking-standing, call cellphone, playing guitar, sitting down, standing up, tossing, walking and laying-down*, compare to [XA13].

One of the main advantages of using Trajectory is being discriminative [Joh75]. Many approaches were proposed for action recognition based on the Trajectories, as in the following:

Wang et al. [WKSL11] proposed a shape descriptor that was used to encode a local motion pattern and shape characteristics of the extracted Dense Trajectories (DT). A sequence of displacement vectors describes the shape of the Trajectory that is normalized by the sum of displacement vector size. To describe video sequence, The DT is extracted by sample dense points and track these points depend on the displacement information of a dense optical flow field. This descriptor extended the motion coding scheme based on MBH [WKSL13] that developed in the field of human detection. For detecting the motion and structure data, HOG, HOF, and MBH are obtained from a space-time volume which aligned with the Trajectory. HOG and HOF feature descriptors are standard methods that have led to excellent results in many datasets [LMSR08]. The MBH represents the gradient of the optical flow that was proposed initially for human detection [DTS06]. For embedding more structure information, the volume subdivided a spatio-temporal grid of size $(b_\sigma \times b_\sigma \times b_\tau)$, as shown in Figure 2.7. Suppose b bins are used for HOG and MBH, and $b+1$ (1 for static) bins for HOF, then a $(b_\sigma \times b_\sigma \times b_\tau \times b)$ vector was computed for HOG, $(b_\sigma \times b_\sigma \times b_\tau \times (b+1))$ for HOF and $(b_\sigma \times b_\sigma \times b_\tau \times b \times 2)$ for MBH (2 is corresponding to the horizontal and vertical flow components). For evaluating the performance of Dense Trajectories, a standard BoFs approach was used by constructing a codebook firstly for each descriptor, Trajectory, HOG, HOF, MBH, separately. To obtain the codebook, k-means clustering was applied, and the resulting histograms of visual word occurrences were used as video descriptors. For classification, a multi-class classification SVM was used.

Koperski et al. [KBB14] proposed two video descriptors for action recognition. First descriptors investigated the DT to extract 3D Trajectory from RGB-D videos by combining motion and depth information. The researchers were also used relative Tra-

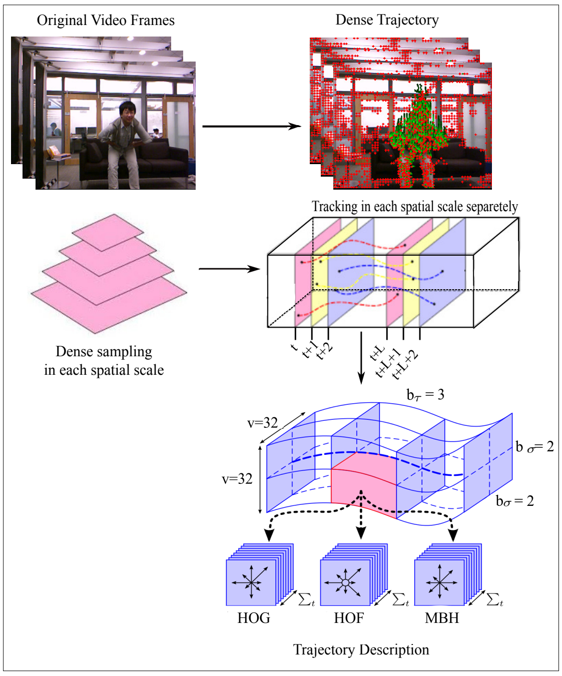


Figure 2.7: Illustration of Dense Trajectory description, compare to [WKSL11].

jectories to encode spatial information of features. The second descriptor improved the performance of actions with low movement rate, i.e., reading, writing. For this implementation, the SURF detector was used to catch appearance features even when there are no Trajectories points detected. Afterward, the SURF descriptor for each detected point was computed based on RGB appearance. BoW was obtained by using the k-means algorithm, then a codebook for each descriptor was constructed. A non-linear SVM was used as a classifier.

Xiao et al. [XZYT14] developed action recognition of human in RGB-D sequences by extending the DT feature of RGB sequences [WS13] to the 3D Trajectory in RGB-D sequences. The 2D positions of the DT of RGB sequences were mapped to the equivalent positions in the depth image. To characterize the 3D Trajectories, they applied MBH on the depth channel to proposed 3D Trajectory shape descriptors. For obtaining the feature representation of RGB-D videos. Finally, a linear SVM was used for classification.

A summary of some approaches that were presented before and used the local features for human action recognition tasks will be illustrated in Table 2.1.

Authors	Summary	Modality
Noguchi et al. [NY10]	Used SURF detector. k-means clustering was utilized for a codebook generation. A BoW vector is generated based on the codebook. Trained an SVM with the generated BoW vectors.	RGB
Yang et al. [YTYC12]	Proposed the SURF-MHI-HOG method to extract spatial and temporal information separately from RGB videos. Then, BoW vectors are generated and finally evaluated using a linear SVM classifier.	RGB
Benoit et al. [BCDH10]	Used Retina model for a low-level image processing and to enhance the detection potential. Double spatio-temporal filtering occurs and ensures a proper structuring of video data.	RGB
Sabin et al. [SBL14]	Used a Retina model to detect the salient areas from video frames and to construct spatio-temporal descriptors for extracting BoW from local features such as: SIFT, SURF, and FREAK. The KNN was used for classification.	RGB
Dalal et al. [DTS06]	Proposed HOG detectors from combined appearance and motion in a linear SVM framework.	RGB
Uijlings et al. [UDSS15]	Used a BoW pipeline with HOG, HOF, and MBH descriptors to compute trade-off for video classification. Hierarchical k-means based visual vocabulary and RF-based vocabulary were used to explore the trade-off between efficiency.	RGB
Kellokumpu et al. [KZP08]	Extracted features from LBP-TOP descriptors. The HMM was performed to improve the classification activity.	RGB
Xia et al. [XA13]	Extracted DSTIPs and built a local DCSF to describe the 3D depth cuboid around the DSTIPs. PCA, k-means clustering, and SVM classifier were applied on the computed descriptors.	Depth
Wang et al. [WKSL11]	Proposed the Trajectory shape descriptor. HOG, HOF, and MBH are obtained for detecting the motion and structure data. BoFs, k-means clustering and multi-class SVM classifier were applied to evaluate the performance of the descriptor.	RGB
Koperski et al. [KBB14]	Combined motion and depth information to extract 3D Trajectory. Improved performance on actions with low movement rate based on SURF. BoW is obtained by using the k-means algorithm. A non-linear SVM is used as classifier.	RGB-D
Xiao et al. [XZYT14]	Proposed 3D Trajectory shape descriptors. Applied MBH on the depth channel. A linear SVM classifier was obtained for video feature representation.	RGB-D

Table 2.1: Summary of the previous approaches used local features for action recognition.

From the previous approaches, when facing the decision, *which local feature descriptor should be used?* The works of [YTYC12, BCDH10, WKSL11, XZYT14] have been selected to be used and improved by using the RGB-D videos. Therefore, in this thesis, the local feature detectors and descriptors will be also adopted, i.e, SURF detector, Retina model, HOG, HOF, LBP, DT, and MBH descriptors. These detection and description methods will be explained later in detail in Chapter 3, Section 3.3.1. These types of feature representation methods will be applied on RGB-D videos for computing the feature vectors. By means of classification methods, these vectors will be trained either directly or after combining them with global features representation methods (Section 2.4.2).

2.4.2 Feature Representation: Global Features

Methods based on global feature representations, also known as holistic methods, treat a video sequence as a whole instead of applying sparse sampling by utilizing STIP detectors or extracting trajectories. The global feature representation allows learning directly from raw images in video sequences and describes the overall properties of objects, such as objects silhouettes or contours. Global features representations have recently drawn increasing attention in several kinds of research [CRHV09, SCH09, MH17] because they can describe and encode more visual information from the whole images or video by protecting spatial and temporal structures of actions that occur in a video sequence. However, global feature descriptors are critical to partial occlusions and background changes, and they often require pre-processing steps such as segmentation, background subtraction, and tracking. In this thesis, the global feature extraction will be obtained by using different global descriptors (see Chapter 3, Section 3.3.2) to improve human action recognition using RGB-D videos.

Most of action recognition methods used global feature representations because of their excellent ability to protect the structural information of actions. In the following, researches regarding global features' calculation and implementation strategy will be reviewed.

Carletti et al. [CFP$^+$13] proposed a method for recognizing human actions in depth images, which are recorded by a Kinect sensor. In this work, the depth images were represented by the combination of three sets of well-known features, including Hu-moments, depth variations, and the $\Re$ transform (an enhanced version of the Radon transform). Finally, a GMM classifier was applied to the extracted features.

AlAzzo et al. [AATM17] extracted human action features from RGB videos by utilizing seven Hu-moment invariants method (see Chapter 3, Section 3.3.2). Hu-moment is applied to solve surveillance camera recording issues under various conditions such as side, position, illumination variations, and direction. For the classification process, a convenient Euclidean distance classifier (EDC) was performed to improve their action recognition method.

Wang et al. [WLJ14] proposed a compact representation using a global GIST feature descriptor and Two-Dimensional Principal Component Analysis-Two-Dimensional Linear Discriminant Analysis (2DPCA-2DLDA) algorithms. In this work, global grid-based GIST features are firstly obtained from video sequences and then described as GIST matrices. These matrices are reflected in the global structure to distribute the local grids and simultaneously represent the local grids. Finally, the extracted features were incorporated into a low-dimensional compact of the global descriptor, and the actions were recognized with an SVM classifier.

Douze et al. [DJS$^+$09] evaluated the accuracy and complexity of the global GIST feature descriptor from RGB images. They proposed several perceptual dimensions, e.g., naturalness, ruggedness, roughness, expansion, and openness, to present the dominant spatial structure of a scene. Then, these dimensions are approximated by using spectral and coarse localized information. In this work, the global GIST feature is used to develop a low dimensional representation of the scene that does not require any form of segmentation. The results obtained with GIST had high efficiency and small memory usage, which allows the researchers to extend the datasets' size. For evaluation, k-means clustering was used to map the computed feature vectors into BoW for forming a visual vocabulary.

In addition to the previous researches that computed features by using the global features descriptors, several approaches using shape masks and silhouette information to represent the human body and its dynamics play an active role in human action recognition. Shape analysis approaches aim to describe and localize the changes in the human body shape by converting video frames into static shape patterns and comparing the patterns with already stored patterns in the recognition stage. One of the shape analysis approaches that will be used in this thesis is Motion History Images (MHI), and it will be explained later in Chapter 3, Section 3.3.2.

Bobick et al. [BD01] used shape masks from different images to present temporal templates by projecting frames onto a single image using MHI and Motion Energy Image (MEI) as action representation. In their work, MEI represented a binary mask that indicated motion regions. In contrast, MHI indicated how motion happened by weighting motion regions according to the point occurrence. To construct a recognition performance, the researchers performed a matching algorithm for the temporal template; matching was done using seven Hu-moments.

Blank et al. [BGS$^+$05] presented human actions in RGB video sequences. In this work, the actions are viewed as silhouettes of moving body parts, which are based on the observation that human motion creates a space-time shape in the space-time volume. The space-time shapes contain both spatial information about the pose of the human and dynamic information. The extracted features' performance was shown by the relatively simple classification scheme using K-nearest neighbor classification and euclidean distance.

Tsai et al. [TCL15] proposed a global spatio-temporal representation from combined an OF and an MHI to represent the space-time action changes in the video sequence. This combination was represented local movements of body parts in the global temporal model, as shown in Figure 2.8. In their work, the OF-MHI provided a better distinctiveness power to describe local movements in a global time-space. Finally, an SVM classifier was used to train and test video actions.

The global feature is also represented as a texture feature descriptor. The texture has an important characteristic that is used to identify objects or areas/regions of interest

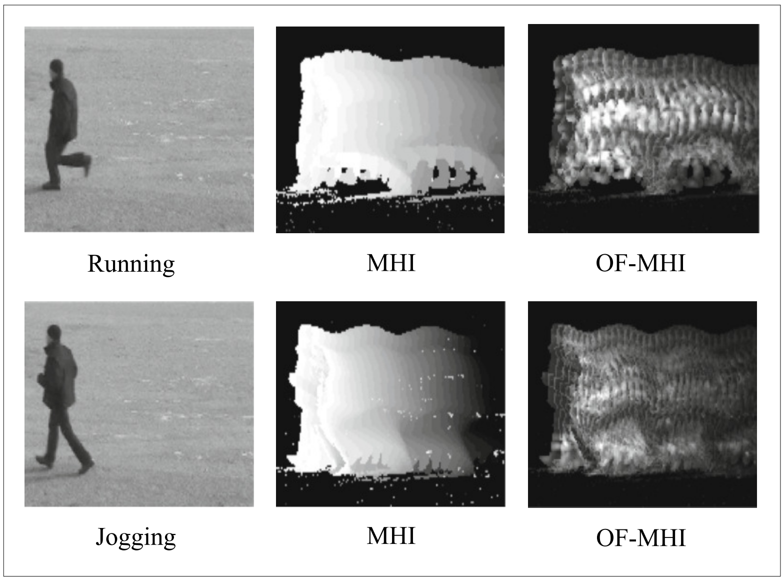


Figure 2.8: Global spatio-temporal representations from the MHI and the OF-MHI, compare to [TCL15].

in an image, as it includes valuable information about the structural arrangement of surfaces and uses for image classification tasks [HSD73], for instance:

Caetano et al. [CCS16] developed a spatio-temporal feature descriptor called Optical Flow Co-occurrence Matrices (OFCM). In their method, they extracted a robust set of texture features known as Haralick features (see Chapter 3, Section 3.3.2), these features are used to describe the flow patterns through calculating meaningful properties of co-occurrence matrices in order to capture local space-time characteristics of the motion over the neighboring OF orientation and magnitude. The OFCM was applied to RGB video for performing the action recognition task.

Muchtar et al. [MYJ$^+$17] improved the moving object detection system from RGB videos. The researchers proposed this system by combining bit-planes representation with the Gray-Level Co-occurrence Matrix (GLCM). This combination allowed the system to achieve the motion history and remove the shadow. The results of this approach were efficient and robust for detecting moving objects in real-time.

Lloyd et al. [LMMR16] proposed a method to detect events and to minimize violence automatically from RGB videos. Abnormal crowd detection was applied by utilizing computer vision techniques, as Haralick features were selected with GLCM texture feature analysis, which was implemented for crowd density estimation. A temporal coding was used to describe the crowd dynamics, and the parameter values were selected with an in depth evaluation. The method was highly effective at discriminating normal and abnormal behavior and could operate in real-time. A Random Forest (RF) classifier was used to evaluate the computed features.

A summary of the presented global feature methods will be listed in Table 2.2.

Authors	**Summary**	**Modality**
Carletti et al. [CFP$^+$13]	Extracted features from combined global features, including of Hu-moments, depth variations, and the $\Re$ transform. A GMM classifier is finally adopted.	Depth
AlAzzo et al. [AATM17]	Used seven Hu-moment invariants to extract global features on videos. A convenient Euclidean distance classifier (EDC) is performed as a classifier.	RGB
Wang et al. [WLJ14]	Proposed a compact representation using a global GIST feature and 2DPCA-2DLDA algorithm. The actions are recognized with SVM classifier.	RGB
Douze et al. [DJS$^+$09]	Used global GIST to develop a low dimensional representation of the scene without required a segmentation. k-means clustering is used for learning a feature vector and to map the descriptors into BoW for forming a visual vocabulary.	RGB
Bobick et al. [BD01]	Used MHI and MEI as the action representation. Seven Hu-moments was used as a matching algorithm for the temporal template to construct a recognition system.	RGB
Blank et al. [BGS$^+$05]	Viewed action as silhouettes of a moving body. The classification scheme was evaluated by using K-nearest neighbor and euclidean distance.	RGB
Tsai et al. [TCL15]	Combining an OF and a MHI to represent local movements of body parts in the global temporal model. An SVM classifier is used to train and test action models.	RGB
Caetano et al. [CCS16]	Developed a spatio-temporal feature descriptor called OFCM. Also, they extracts a set of features known as Haralick features to describe and capture local space-time features of the motion over the neighboring of OF. An SVM classifier was applied to evaluate the OFCM method.	RGB
Muchtar et al. [MYJ$^+$17]	Improved moving object detection by combining bit-planes representation with the GLCM.	RGB
Lloyd et al. [LMMR16]	Proposed a method to detect the events based on GLCM texture feature analysis and Haralick features. A RF classifier is used for training.	RGB

Table 2.2: Summary of the previous approaches used global features for action recognition.

From the previous approaches, when facing the decision, *which global feature descriptor should be used?* The work of [CCS16] is recommended to be used and improved by extending their 2D method to use with RGB-D videos (see Chapter 7).

Also, in this thesis, the global feature extraction will be obtained by using different global descriptors, such as seven Hu-moment invariants [AATM17], a global GIST feature, and Haralick texture features from the GLCM. For more details, these feature extraction methods will be explained later in Chapter 3, Section 3.3.2. These types of features will be applied to RGB-D videos to compute the feature vectors. Finally, extracted feature vectors will be trained either directly or combined with other local feature representation methods (Chapter 3, Section 3.3.1).

2.4.3 Feature Representation: Features Combination

Developing new methods by combining different feature representations has become an important issue in the action recognition field. Recently, several successful approaches have been proposed to address the issues of action recognition. A combination of features can be categorized into two strategies: Combination of local and global features [IcS10, NJ14, WLJ15] which are computed from RGB-D images, or combination of feature vector values which are computed from different image channels, i.e., RGB and depth information [ZLYC12, SL15, MAL18].

Combining features can take advantage of individual features and can represent a trade-off between performance and effectiveness. Feature combination is still a challenging task to improve human action recognition. Several approaches are based on combining the local and global features to improve human action recognition, for instance:

Qian et al. [QMXW10] combined global and local features in order to classify and recognize human actions. The global feature was depended on a binary MEI. The global feature was represented by contour coding of the motion energy image. As local features, an object's bounding box of human blobs in each frame was used. The computed features were classified using multi-class SVM and binary tree architecture.

Solmaz et al. [SAS13] proposed a GIST3D feature descriptor combined with HOG and HOF descriptors for proving the classification of RGB video actions. The GIST descriptor is based on a 3D filter bank to calculate the 3D spatio-temporal features. The recognition performance is achieved only in combination with local STIPs features. This work is comprising the low/medium level approaches to human action recognition. For classification, the researchers have trained their method by using a multi-class SVM with the linear kernel and histogram intersection kernel for STIPs.

Wang et al. [WLJ15] proposed a human action representation method by combining a global GIST feature with a local patch coding by used the BoW framework. The high-dimensional global features have been translated through patch segmentation and local coding into ordered form and compact concatenated visual words. Then, the computed BoW features were finally tested with an SVM classifier.

On the other approaches, the researchers proposed several feature combination methods from RGB-D information to recover low-level features and develop image descriptors, for instance:

Letouzey et al. [LPB11] projected the scene flow in the image domain to connect 3D motion with image flow through a projective matrix. They estimated a 3D motion field by combining geometric information from depth maps with intensity variations in RGB images to handle both arbitrary large motions and sub-pixel displacements.

Zhao et al. [ZLYC12] studied the representation of scenes through the computation of interest points in appearance and depth information, individually, i.e., they extracted interest points separately from RGB channels and then combined with depth map, as shown in Figure 2.9. In their research, the HOG and HOF were used to extract the features from RGB video, while a Local Depth Pattern (LDP) was used to represent depth features. The k-means clustering was applied to the computed set of both descriptors for obtaining the codebooks. Finally, a multi-class SVM classifier is used to classify human activities.

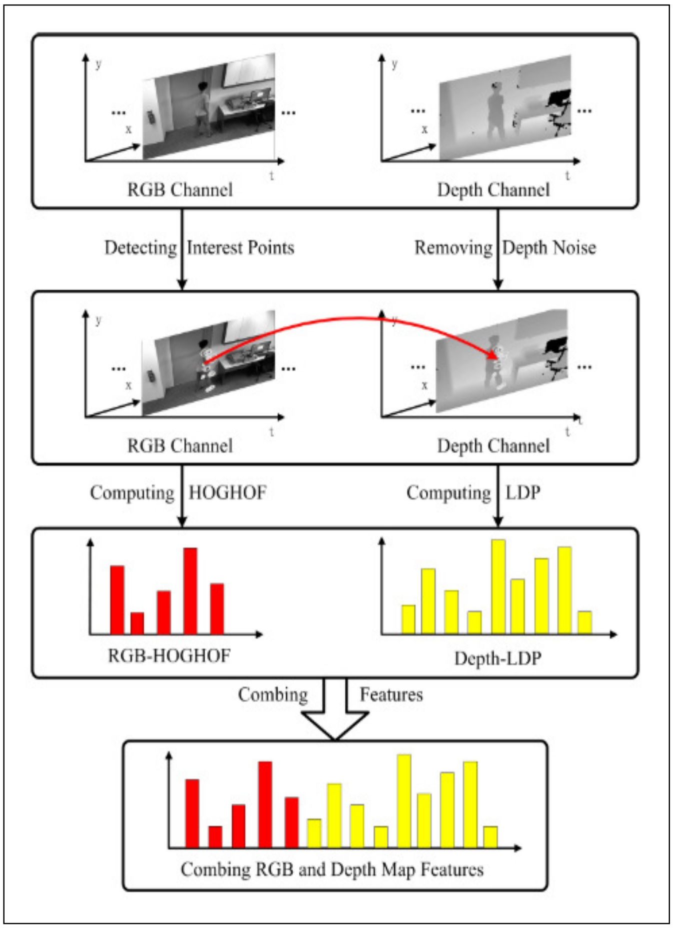


Figure 2.9: Combing RGB and Depth map features framework, compare to [ZLYC12].

Fanello et al. [FGMO13] combined motion and appearance features (see Figure 2.10) to develop one-shot real-time learning, i.e., 3D Histograms of Scene Flow (3DHOFs) and Global Histograms of Oriented Gradient (GHOGs). The combination method was used to improve action recognition from RGB-D video. The researchers have proposed a simultaneous online video segmentation and a linear SVM for recognition.

A summary of the presented features combination methods are illustrated in Table 2.3.

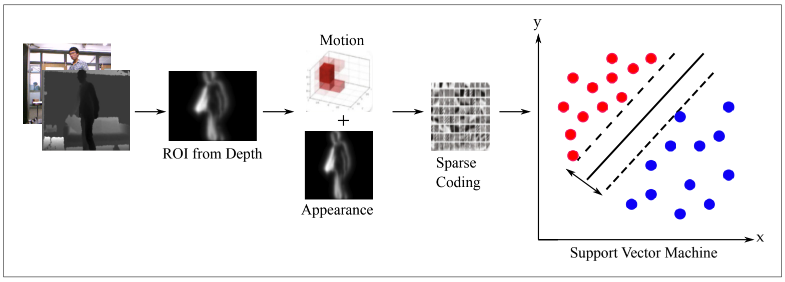


Figure 2.10: Overview of the recognition system, compare to [FGMO13].

Authors	Summary	Modality
Qian et al. [QMXW10]	Combined global feature represented by binary MEI and local features represented by the object's bounding box of human blobs. The combined features was classified using multi-class SVM and binary tree architecture.	RGB
Solmaz et al. [SAS13]	Proposed a spatio-temporal GIST3D feature descriptor that combined with HOG and HOF descriptors for proving the classification task. A multi-class SVM was trained.	RGB
Wang et al. [WLJ15]	Combined global GIST feature and local patch coding. BoW technique was used to represent the human actions. The recognition performances were tested with SVM classifier.	RGB
Zhao et al. [ZLYC12]	Extracted HOG and HOF from RGB, then combined with LDP from Depth. k-means clustering and SVM classifier were applied for computing the performance of the method.	RGB-D
Fanello et al. [FGMO13]	Combined motion and appearance features that are represented by 3DHOFs and GHOGs. Then proposed a simultaneous online video segmentation and action recognition by using linear SVMs.	RGB-D

Table 2.3: Summary of the previous approaches used features combination methods for action recognition.

The previous approaches illustrated different ways of the feature combination, such as combining a global with local features computed separately from input images/videos [WLJ15], or computing features from input RGB and depth data individually and then combine them to form the BoFs [ZLYC12]. Therefore, this thesis work is depended on the idea of the previous feature combination way. The combination of features will be applied to enhance the accuracy and benefit of feature representations. The calculation of feature vectors will obtain by two strategies: Firstly, features will be extracted from RGB-D channels based on the combination of different local and global descriptors. Secondly, features will be extracted from RGB and depth channels independently and then combine to represent the video action, or the computed depth features are dependent on the RGB features. Finally, the computed BoFs vectors will be trained by using the classification methods.

2.5 Machine Learning

Machine learning (ML) is a study of algorithms and statistical models, and it is a subset of artificial intelligence. Computer systems used ML to perform tasks without specific instructions instead of depending on patterns and inference. Thus, ML applies to computer vision, pattern recognition, and software engineering. ML is performed with minimal support from software programmers, which uses data to make decisions and allows it to be used interestingly in a variety of industries. ML algorithms are grouped into a taxonomy based on the wanted output of the algorithm. Standard algorithm types can be classified as supervised learning [Kot07] and unsupervised learning [JMF99], (see Figure 2.11). The ML algorithms are applied to the training datasets. When a new data instance comes in the play for prediction of the class label, the ML algorithm acts on the new instance and predicts its class based on previous experiences and records [Ayo10]. Many ML algorithms have been developed, designed, and adapted to study action and activity learning. The types of ML models that have been utilized for action and activity learning differ almost as widely as the types of sensors from which data is obtained.

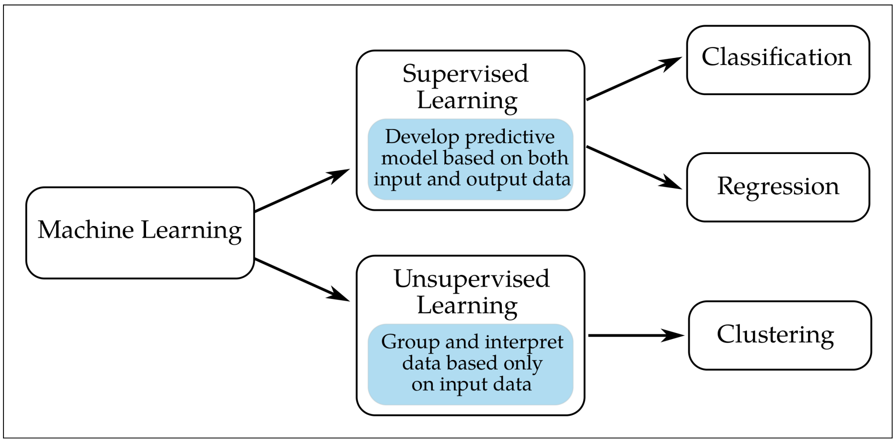

Machine Learning
Supervised Learning
Develop predictive model based on both input and output data
Unsupervised Learning
Group and interpret data based only on input data
Classification
Regression
Clustering

Figure 2.11: Machine learning techniques include both supervised and unsupervised learning.

2.5.1 Supervised Learning

Supervised learning algorithms are machine learning algorithms that are learned to map input data into a target attribute of the data (a class label). Typically, the algorithm learns a relation between the input data's attributes (features) and the target attribute by reducing a loss function defined on the pairs of input data and the corresponding destination attribute. The discovered relation is represented in a structure identified as a model. Models describe the hidden relations between the input data and target attributes and can be used to predict the objective attribute (label), provided the values of the input data [CK15]. In a supervised learning algorithm, it is necessary to follow specific steps to solve a given problem [Vap95]:

- Define the kind of training examples.
- Collect and organize a training set.
- Determine the representation of input feature of a learned function.
- Determine the learning function structure and the related learning algorithm.
- Complete the design and execute the learning algorithm on the collecting set of data.
- Evaluate the performance of the learned function.

There are two main kinds of supervised learning algorithms [CK15]: Classification and regression. The distinction within these kinds depends on the nature of the destination attribute. In classification, the destination attribute takes on categorical values such as class labels. Regression models, on the other hand, map the input data into a real value of destination attribute.

Many challenges in human behavioral modeling are formulated as supervised learning problems, for instance mapping sensors to action labels, detecting the devices that are used at home based on the current energy consumption trend and modeling the health of an older adult from sensor data.

Supervised learning aims to build a model of class labels' distribution in terms of input features. The obtained classifier then assigns class labels to the test instances, where the input feature is known, but the class label value is unknown. Due to the nature of this thesis tasks, the primary focus will be based on classification algorithms to know which actions could be recognized.

Classification

Classification is a way of predicting the class of given data. Sometimes, classes are called targets or labels. Classification is categorized as supervised learning, where the targets are also provided with the input datasets. The datasets have various actions, and the goal is to detect and recognize these actions in videos.

The process to classify data includes two stages: Firstly, the learning phase, the objective of learning is building a classifier by analyzing the labeled data; secondly, predicting phase that is based on the well-established model for predicting. This model has enough generalization ability, which means that the model has excellent classification accuracy on the training data and has a high classification performance for the future data, which supposedly have the same statistical allocation as the training data [YLY$^+$13].

Classification algorithms are divided into two types: Binary and multi-class. Binary classification classifies instances into one of two classes. In comparison, multi-class classifies instances into one of three or more distinct classes, i.e., a multi-class classifier is a classifier that ables to distinguish more than two classes, and it will be used in this thesis.

Multi-Class Classification In multi-class classification, it is assumed that there are a number of classes N_{cl}, and each sample is assigned to only one label. The aim of multi-class pattern recognition is to accurately map an input feature space Ω that consisting of N_{cl} classes with $N_{\text{cl}} > 2$, where $\cup_{i=1}^{N_{\text{cl}}} \Omega_i = \Omega$ and $\Omega_i \cap \Omega_j = \phi$ for $i \neq j$.

However, many classifiers work better for two-class classification problems ($N_{\text{cl}} = 2$). Thus, the N_{cl}-class pattern classification problem is broken down into N_{cl} two-class problems [OMI12]. Several popular binarization approaches for classes have been introduced to address the problems of multi-class pattern classification problems. One of the multi-class classification methods is the One-vs-All (OvA) [Ana95]. The OvA strategy is involved in training a single classifier per class and based on a reduction of the multi-class problem into N_{cl} binary problems. N_{cl} binary classifiers are designed for a problem with λ classes. For each class Ω_λ, a classifier f_λ is defined to distinguish features $\boldsymbol{v}$ belonging to Ω_λ from all other classes $\Omega - \Omega_\lambda$. This yields N_{cl} classification problems. A pattern is classified in the class whose relevant classifier has the highest activation power. The mathematical definition of the decision function, δ_{f}, is computed as follows [OMI12]:

$$\lambda^* = \delta_{\text{f}}(\boldsymbol{v}) = \arg \max_{\lambda \in \{1,...,N_{\text{cl}}\}} f_\lambda(\boldsymbol{v}) \tag{2.5}$$

where δ_{f} is the activation output of a list of classifiers f_λ for $\lambda \in \{1, ..., N_{\text{cl}}\}$, $\boldsymbol{v} \in \mathbb{R}^n$ is a vector of input features and λ^* is the resulting class label. The other method is One-vs-One (OvO) [HT98]. OvO discriminates each class from every other class by building a classifier for each pair of classes. This thesis explores supervised learning based on OvA multi-class classification approach to learn from training data and apply it to unseen data for predictions. In addition to unsupervised algorithms that can draw inferences from datasets with no labels to perform clustering (see Section 2.5.2).

This thesis presents several popular classification algorithms that will be used later due to a large number of machine learning algorithms. The classification algorithms are well defined by [Bis06, KZP06, MRS08]. The main classification algorithms will be represented in the following:

Support Vector Machine (SVM) [Vap95, BB98, CL11]: In these papers, the researchers maximize the distance between a hyperplane that distinguishes two types of data from instances on either side of it. They can perform linear and non-linear separation with a kernel function. Furthermore, the global minimum is reached, and a local minimum is prevented, which can occur in other search algorithms, including neural networks. Finally, they usually delivered decent results.

K-Nearest-Neighbors (KNN) [CH06, MRS08]: It identifies the closest K-nearest instances to the query instance and decides its label by choosing the single most common label of nearest instances. The critical drawback of this classifier is that it requires the storage of all the instances, and it is sensitive comparing instances to the choice of the similarity function. Besides, it is widely accepted that it is subject to irrelevant features.

Random Forest classifier (RF) [Bre01, GPT08]: It uses several decision trees and gives the class label on the basis of votes from each decision tree. In addition, a random set of features is used to separate each node. The RF can overcome the restriction of decision trees because the training data are always over-fit.

Naive Bayes Classifier (NB) [CC08]: It is the simplest Bayesian classifier based on the principle of Bayes' theory that all variables contribute to the classification and are mutually associated.

Artificial Neural Networks (ANNs) [KL90, Zha00]: It is a computational model where its functions and methods are based on the structure of the brain [KZP06]. ANNs consist of several connected units (neurons) grouped in a connection pattern. Units in a network are usually made up of three-layer types: An input layer with input units that obtain processed data, an output layer with output units that provide the result of the algorithm, and hidden layers with hidden units that process data. ANN learns the weights of the neurons' connections to determine how to map an input to an output. There are many types of ANNs: Single-layer perceptron, Radial Basis Function (RBF) network, Deep Neural Network (DNN), Convolutional Neural Network (CNN). A single layer perceptron is the most straightforward neural network that depends on a linear combination of weights and the feature vector. A DNN is a neural network that has at least one hidden layer of units between the input and output layers. An RBF is a three-layer feedback network. Each hidden unit performs a radial activation function, and a weighted sum of hidden unit outputs applies from each output unit. A CNN is another kind of neural network which can be added directly to the raw data, which automates the process of feature construction.

Several approaches applied the classification methods to improve human action recognition from the video [UDFS14, MSM15, MAaMV15]. In this thesis, the previous classification algorithms will be applied as a multi-class classifier for the recognition approach of human action, and they will be explained in detail in Chapter 3, Section 3.5.

2.5.2 Unsupervised Learning

Unsupervised learning represents a class of problems involving a model to extract or describe connections between data. Unlike supervised learning, unsupervised learning only operates on input data without outputs or target variables. In some pattern recognition difficulties, the training data consists of sets of input vectors $\boldsymbol{v}$ without corresponding target variables; these are named unsupervised learning problems.

The goal of using unsupervised learning is to discover the inherent groupings within the data, called clustering. Furthermore, it is used to estimate the data distribution within the input space, called density estimation. Moreover, it is used for visualization purposes by projecting data from a high-dimensional space into two or three dimensions [Bis06].

In unsupervised learning, labels or constraints between data are not known before, and it is up to the algorithm to cluster the videos according to their features. This clustering can have a set target of a number of clusters, or the number can be random, which makes it a challenge to find out how many clusters to choose. An example of a clustering algorithm that will be used in this thesis is k-means clustering.

k-Means Clustering

Clustering is a technique designed to separate the data into groups (clusters) where each group is built from similar data. The clustering aims to separate groups with similar types and allocate them into clusters [JMF99]. The k-means clustering is one of the unsupervised learning methods [YLY+13], that is used with unlabeled data, i.e., data without defined categories or groups because it has high efficiency on the data partition, especially in the large dataset. The goal of the k-means clustering is to define groups in the given data, where the number of groups indicated by the (k) variable. This clustering method is working iteratively to assign each data point to one of the (k) groups based on the provided features. The data points are clustered based on feature similarity. The k-means clustering algorithm's output results are: Centroids of the (k) clusters that can be used to label a new data, each centroid defines one of the clusters, which means each data point is allocated to its nearest centroid. Also, labels for the training data mean that each data point is allocated to a single cluster. k-means clustering divides $\boldsymbol{v}$ points into k clusters in which each cluster refers to a mean value of the cluster. Formally, this is described as reducing the sum of squares within each cluster.

$$\underset{Z}{\operatorname{argmin}} \sum_{j=1}^{k} \sum_{\boldsymbol{v}_i \in Z_j} \|\boldsymbol{v}_i - \boldsymbol{\mu}_j\|^2 \quad , \tag{2.6}$$

where, $\boldsymbol{v}_i$ is data points and $\boldsymbol{\mu}_j$ is the mean of feature vectors belonging to the cluster Z_j. The standard k-means clustering shown in Algorithm 1:

Algorithm 1 k-Means Clustering Algorithm

```
input  : Data points $\boldsymbol{v}_i \in V$, cluster count $k$
output: Set of partitions $Z_j \in Z$
$Q \leftarrow \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_k | \boldsymbol{\mu}_i$ randomly chosen points from $V\}$
$Z \leftarrow \{Z_1, Z_2, ..., Z_k | Z_j = \phi\}$
repeat{
    $Z_j \leftarrow \{\boldsymbol{v}_i \in V| \quad \|\boldsymbol{v}_i - \boldsymbol{\mu}_j\|^2 \leq \|\boldsymbol{v}_i - \boldsymbol{\mu}_m\|^2 \quad j \in \mathbb{N}, 1 \leq j \leq k\}$
    $\boldsymbol{\mu}_j \leftarrow \frac{1}{\|Z_j\|} \sum_{\boldsymbol{v}_i \in Z_j} \boldsymbol{v}_i$
until $Z_j = Z_{j-1}$;
```

The steps of the k-means algorithm are described as follows:

1. **Initialization**: The algorithm is typically initialized by allocating all means to a random value, i.e., the initial cluster center is k random dots, but some heuristics can select starting means depending on the data.

2. **Assignment step**: In this step, each point is allocated to the nearest mean by depending on the distance between the point and the means:
   $Z_j = \{\boldsymbol{v}_i \in V| \quad \|\boldsymbol{v}_i - \boldsymbol{\mu}_j\|^2 \leq \|\boldsymbol{v}_i - \boldsymbol{\mu}_m\|^2 \quad j \in \mathbb{N}, 1 \leq j \leq k\}$

   This distance is commonly the Euclidean distance but can be used the other distance measure. If each point has been allocated to one mean, the result is temporary clusters of k.

3. **Update step**: In each cluster, a new mean is calculated as the center of all cluster points, i.e., the average of the cluster point coordinates.

   $$\boldsymbol{\mu}_j \leftarrow \frac{1}{\|Z_j\|} \sum_{\boldsymbol{v}_i \in Z_j} \boldsymbol{v}_i$$

4. Repeat steps 2 and 3 until convergence.

The algorithm is efficient and straightforward, but it also has a number of drawbacks, including its high reliance on the initial conditions and the fact that large datasets are not likely to scale well. Almost all the solution found is a local minimum. Choosing the best from several rounds with varying initializations is a way to improve the solution, but there is no assured global minimum.

In the k-means algorithm, the initial center's selection is the key to getting an accurate result. If choosing the correct initial centroids will get a good result, but if it is not, the outcome will be worse. It can be formed as a large and low-density cluster divided into pieces or combine two similar or close clusters into a single group. Therefore, the initial centroids are usually randomly selected or used prior knowledge to label some of them to get a good result.

In this thesis, the k-means clustering algorithm will be performed for generating the dictionary with varied sizes to represent a video sequence using the Bag-of-Features (BoFs) approach (see Chapter 3, Section 3.4). In the training stage, after the extraction of features from training videos, the k-means algorithm is used to learn the visual word. Then, each feature is assigned to a specific visual codebook through the clustering process, and the histogram of visual codebooks can represent the video. The histograms represent training videos are used as input vectors for one classification method to construct a classifier. In the testing stage, the features are computed from a new input video; then, these features are mapped into a histogram vector by utilizing the descriptor coding method that uses the pre-trained visual words (dictionary/codebook). Finally, the histogram vector values are fed into the classifier to obtain the recognition accuracy results.

2.6 Deep Neural Network-Based Approaches

In recent years, there has been significant growth in the amount of research in the computer vision field to understand human actions in images and video sequences. There has been a focus on actions, where the action can be considered a stochastically predictable series of states. Stochastic methods build statistical models to represent a human action as a model contains a set of states. The models are trained statistically on feature vectors in order to create a general statistical model for the classification of action. In other words, the statistical model is designed to generate a sequence with a specified probability. Researchers have developed and utilized a wide range of stochastic techniques, such as a hidden Markov model (HMMs) [Bis06], but recently, the research direction goes toward deep learning methods [YCBL14, Lat17, DBPC19, ZA19] to demonstrate human action recognition. In this thesis, a deep learning technique will be used to recognize human actions in videos.

2.6.1 Deep Learning

Deep learning is a subfield of machine learning that aims to learn multiple representations and abstraction levels to make sense of data like speech, text, and images.

Deep learning approaches can treat the images or videos in their raw forms and automatically extract, represent, and classify features from them. These approaches utilize trainable feature extractors and computational models with multiple processing layers to represent and recognize actions.

As with other machine learning methods, deep learning approaches can be classified into two major categories, supervised and unsupervised methods [GLCL19]. Supervised methods identify an error function that depends on a task, which must be solved, and model parameters changed according to that error function. The error function compares to training data with a known outcome. These methods could provide an end-to-end learning structure, which ensures that a model learns to execute a task on the raw data. Unsupervised methods ordinarily describe an error function that can be minimized based on the model's reconstruction capacity. Both categories with the reconstruction error depended on the deep learning technique. An auxiliary error function may be specified, which requires some characteristics to the learned representation. For instance, sparse auto-encoders aim to make the learned representation sparse, improving the overall learning procedure, and giving a more discriminative representation.

Deep learning architectures have recently been proposed for solving human action recognition tasks. Differently from the standard human action recognition pipeline, deep architectures incorporate the feature extraction and classification in a single approach. Their features are rapidly learned from data being more distinctive. Furthermore, they solve some problems concerning the computation and adaptability of handcrafted features. Researchers have also used deep learning architecture for detecting and recognizing complex events in video sequences. The main two examples of deep learning techniques used for action recognition are convolutional neural networks (CNNs) [CS17, RGF$^+$18, SHJ18], and Recurrent Neural Network (RNN) [MP17, ZZZ$^+$19]. This thesis focuses on using CNN to represent human action recognition from RGB and depth video data (see Chapter 8).

Although research today focuses on deep learning in various computer vision applications, conventional pattern recognition pipelines, such as the BoFs approach, can still be used in many applications. Mainly for applications with small training data or forced computational resources. The BoFs approach is still an area of interest, especially with action recognition tasks. Videos add a more temporal dimension, whereas CNNs was originally designed to model static images and the encoding of motion dynamics in CNN is not clear. Most authors thus use CNNs learned on still images to model static images in action recognition. So, this thesis proposes different ways – *how to model motion in RGB-D videos*?

However, CNN also has two drawbacks in many practical applications: Firstly, because of the large number of examples, it also needs a large number of labeled samples, or at least a large number of synthetic testing samples are needed. Secondly, they are computer-consuming and require graphics processing units (GPUs) to be effectively trained and evaluated.

Convolution Neural Network

Convolution Neural Network (CNN) [LBBH98] is a deep model that preserves complicated hierarchical features through convolution operations alternating with sub-sampling operations on raw data. It is confirmed that CNN can achieve a visual target recognition performance by making appropriate adjustments during training. Also, CNN has invariance for a particular pose, lighting, and disordered environmental changes. Deep learning using a Convolutional Neural Network (CNN) was introduced by LeCun et al. [LKF10] in computer vision applications. The researchers have defined the convolutional networks as trainable multistage architectures composed of multiple phases. Each phase input and the output are sets of arrays named feature maps. For instance, if the input is a color image, each feature map will be represented a 2D array with a color channel of the input image, and a 3D array for an input video. At the output level, each feature map describes a specific feature extracted at all locations on the input (see Figure 2.12).

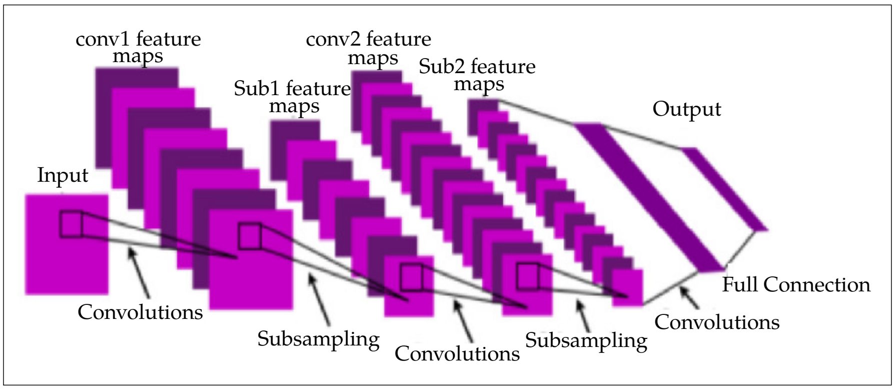


Figure 2.12: A typical CNN architecture with two feature stages [LKF10].

CNN has several advantages over deep neural networks [ZA19] includes: It like the human visual processing system, its structure is highly optimized for processing 2D and 3D images, and it is useful in learning and extracting abstractions of 2D features. Figure 2.13 shows the general structure of CNN consists of two main parts: Feature extraction and classification. In feature extractor layers, each layer of the network receives the output as input from the previous layer and transfers it to the next layer as input.

The architecture of CNN is composed of a combination of three types of layers: Convolution layer, max-pooling layer, and a classification layer. In the low and middle-level of the network, There are two kinds of layers: Convolutional layers and max-pooling layers. The max-pooling layer is effective in occupying shape changes. Furthermore, consisting of sparse connections with attached weights, CNN has significantly fewer parameters than a fully connected network of comparable size. Above all, CNN is trained with the gradient-based learning algorithm and suffers less from the decreasing

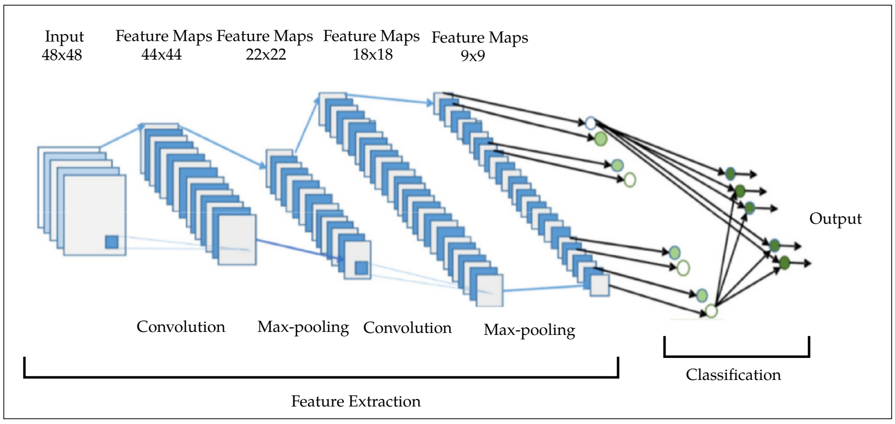


Figure 2.13: General structure of the CNN, includes an input layer, multiple alternating convolution and max-pooling layers, one fully-connected layer and one classification layer [ZA19].

gradient problem. Since the gradient-based algorithm trains the entire network in a way that directly minimizes an error criterion, CNN can provide highly optimized weights. The layers with even-numbered are for convolutions, and with odd-numbered are for max-pooling operations. The output nodes of convolution and max-pooling layers are gathered into a 2D plane described feature mapping. Typically, each plane of a layer is obtained from the concatenation of one or more levels of planes of previous layers. The nodes of plans are connected to a small area of each correlated plane of the previous layer. An individual node of the convolution layer extracts features by convolution operations on the input nodes from the input images [ZA19]. CNN uses a kernel, also known as a filter, to extract features from input images. A kernel is a matrix of values, called weights, which are trained to extract certain features. Depending on the kernel size for convolution and max-pooling operations, the dimensions of the features are reduced. Nevertheless, the number of feature maps typically increased to provide a more precise classification of the extracted feature from input images. The output of the last CNN layer acts as an input to a fully connected network, known as the classification layer.

In the classification layer, feed-forward neural networks have been used as the classification layer. In the classification layer, feed-forward neural networks have been used as the classification layer. The extracted feature in this layer is taken as inputs regarding the weight matrix dimension of the final neural network. Nevertheless, the network or learning parameters of fully connected layers are costly. The evaluation of the respective classes is computed in the top classification layer. The classifier gives output for the corresponding classes based on the highest evaluation. Mathematical details on different CNN layers will be discussed in Chapter 3, Section 3.6.

CNN is an efficient model class for understanding image content, providing state-of-the-art results on image segmentation, detection, recognition [CGGS12, GDDM13, FCNL13]. For instance, Razavian et al. [RASC14] improved the possibilities of feature extraction from CNN using the OverFeat network. They extracted feature vectors from the OverFeat network as generic image representation, and then they were used an SVM for the classification task. Their method proved that pre-trained deep CNN was suitable for generic feature extraction from RGB images. Athiwaratkun et al. [AK15] also improved the possibilities of utilizing a pre-trained network for extracting features from RGB images. Based on the extracted features, they evaluated the quality and performance of the feature values gained from different network layers. These feature values were used to be trained by SVM and RF classifiers.

Motivated by this CNN's success in image processing applications, researchers are working deeply to improve CNN for video processing. The first effort to use CNN in human action recognition was presented by Taylor et al. [TFLB10]. They proposed a model that learns feature map representations of RGB image sequences from pairs of consecutive images. Other researchers have found that motion-based features, such as optical flow, have a rich indication that could be directly fed as a network input. Accurate and efficient methods are available to calculate these kinds of features, some of which utilize GPU capabilities [FBK15]. The use of optical flow was demonstrated to increase the efficiency of CNNs on action recognition tasks.

Simonyan et al. [SZ14] Simonyan et al. [SZ14] proposed a two-stream CNN architecture from RGB video sequences. They used multiple optical flow images calculated from the sequences as an input to their CNN model. They introduced a structure dependent on spatial and temporal streams. The spatial stream performed action recognition from video sequences. Simultaneously, the temporal stream of CNN trained on dense optical flow volumes and saving the horizontal and vertical displacement vectors from successive action frames, i.e., recognizing action from motion using dense optical flow. Finally, a temporal CNN and a spatial CNN were combined to include individual scene and object features, as shown in Figure 2.14.

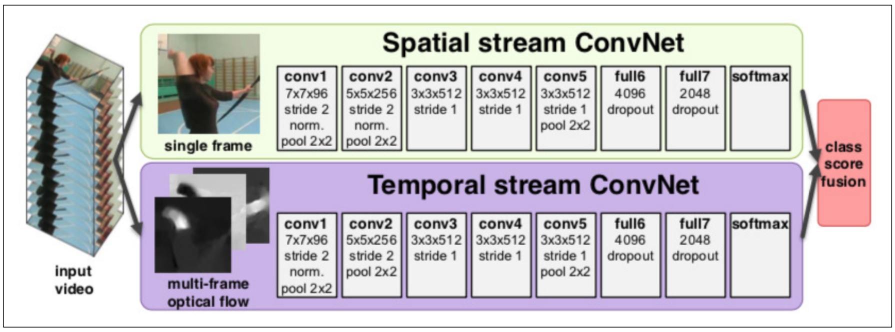


Figure 2.14: Overview of the two-stream CNN system [SZ14].

Karpathy et al. [KTS^{+}14] studied the performance of CNNs in large-scale video classification. They proposed to use of a multi-resolution CNN architecture and time information fusion. The multi-resolution CNN was improved by extending a CNN connectivity in the time domain to gain local spatio-temporal information. This method is used for human action recognition on raw RGB videos.

The early work of Ji et al. [JXYY13] introduced the innovation of inducing temporal information from raw RGB data directly. They were performed 3D convolutions (3D-CNN) on stacks of multiple adjacent video frames. Since then, many authors attempted to improve this kind of models, for instance:

Carreira et al. [CZ17] introduced a Two-Stream Inflated 3D ConvNet from the spatial and temporal domains. A temporal domain is treated by using an optical flow method. While a spatial domain is tested on the RGB image sequences. In their work, spatio-temporal feature extractors and pre-trained weights are provided to improve the human action recognition system.

Baccouche et al. [BMW^{+}11] proposed a neural-based deep model in order to classify sequences of human actions from RGB videos. They aimed to capture the characteristics of video data based on 3D-CNN. The network was trained to allocate a vector of spatio-temporal features to a small number of successive frames, as in Figure 2.15.

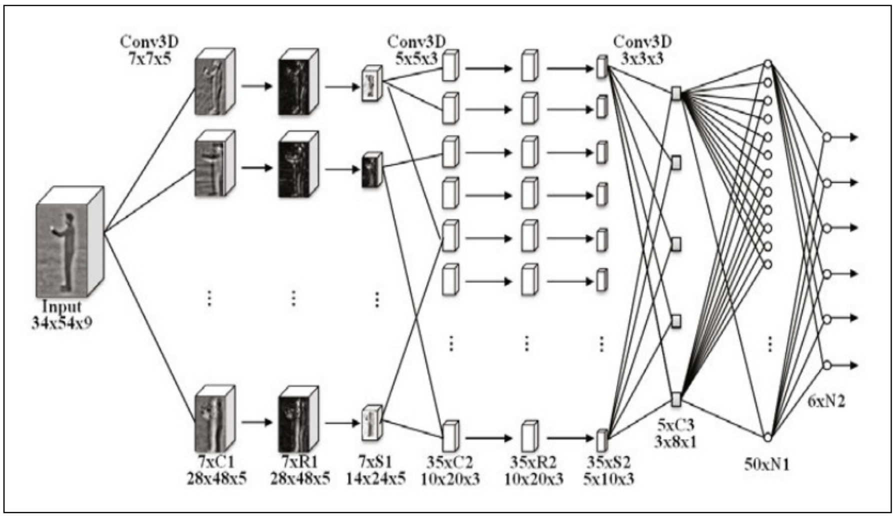


Figure 2.15: Architecture of 3D-CNN for spatio-temporal features extraction [BMW^{+}11].

Latah et al. [Lat17] used deep CNNs and SVM approaches to employ a human action recognition task from RGB video actions. A 3D-CNN approach was utilized to extract spatio-temporal features from adjacent frames. Then, an SVM classifier was used for classifying each instance based on previously extracted features. Additionally, they reduced the number of CNN layers and input frame resolution to satisfy memory limitations.

Recently due to the development of the depth sensor, different works use RGB-D images/videos as input to CNN. Some of these work are presented in the following:

Song et al. [SHJ18] improved scene recognition using RGB-D dataset. They used pre-trained CNN models and a fine-tuning method that was implemented for RGB to be used with the RGB-D dataset. For scene recognition, they integrated RGB and depth features by projecting them in a shared space and learning more from a multilayer classifier configured in an end-to-end network.

Wang et al. [WLG$^+$16] applied a 3D-CNN on depth map sequences to propose human action recognition. They extracted the body shape and motion information at several temporal scales by generated weighted depth motion maps (DMM), called Hierarchical Depth Motion Maps (HDMM). The HDMM was used to store temporal motion information from various views: Top, side, and front views. The three views have been configured to be used as an input to the CNN.

Wang et al. [WZC14] demonstrated a 3D activity recognition model using RGB-D sequences. They developed a network that consists of 3D convolutions and max-pooling operators using the video segments. They also implemented the latent variables in each convolutional layer and manipulated neuron activation. This model structure could be modified dynamically, taking into account the temporal variations of human activities. Figure 2.16 shows an example of the 3D convolution over spatial and temporal domains, where the 3D kernel value is 3 in the temporal dimension. Finally, a feature map was obtained across three adjacent frames by performing 3D-CNNs.

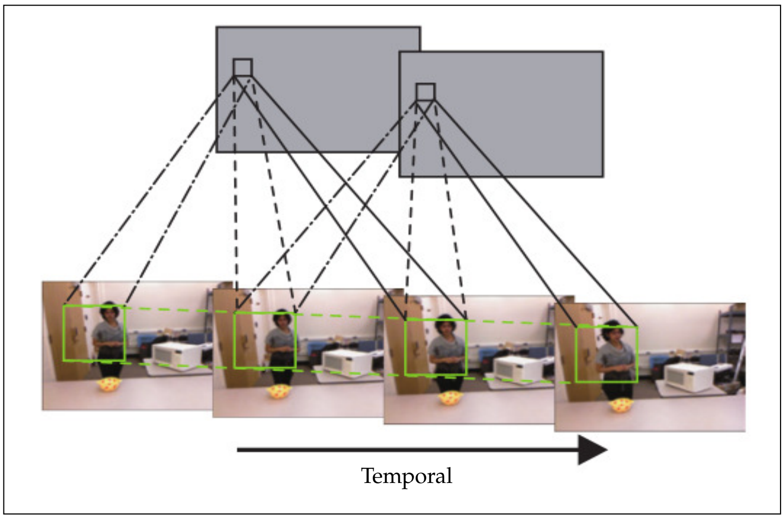


Figure 2.16: Illustration of the 3D-CNN over spatial and temporal domains. [WZC14].

Asadi et al. [AABR^{+}17] proposed a Multi-Modal Dense Trajectory (MMDT) by using scene flow to describe RGB-D videos. A framework of this work is presented to explore the results of hand-crafted and learning-based features alongside deep learning. The researchers extended 2D-CNN to MM2DCNN by including more scene flow as input and merging the output of all models to improve human action recognition.

Liu et al. [LAM17] proposed viewpoint invariant approach using RGB-D data to improve human action recognition, as shown in Figure 2.17. The view-invariant features are obtained from the DT of the RGB videos using a nonlinear learning transfer model and simultaneously extracted view-invariant human pose features based on a CNN model from the depth data. Then, Fourier Temporal Pyramid (FTP) was calculated over them. This method processed RGB and depth information separately to take advantage of using the RGB and depth modalities individually and computing spatio-temporal features from these modalities.

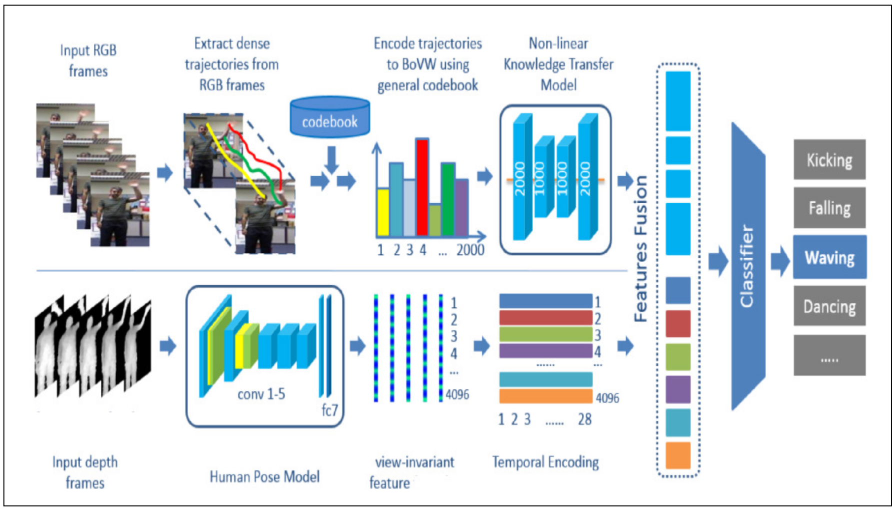


Figure 2.17: Viewpoint invariant RGB-D human action recognition [LAM17].

Table 2.4 summarized the presented state-of-the-art methods that used deep learning in their implementations, especially CNN, in image and video processing.

Authors	Summary	Modality
Razavian et al. [RASC14]	Proved the possibilities of feature extraction from CNN using the OverFeat network from images. SVM was used for classify the extracted features.	RGB
Athiwaratkun et al. [AK15]	Improved the possibilities of using features extracted from a pre-trained network. SVM and RF classifiers were used to train feature vectors.	RGB
Taylor et al. [TFLB10]	Used a CNN model to learn feature map representations of image sequences.	RGB
Simonyan et al. [SZ14]	Proposed two-stream CNN architecture form multiple optical flow images.	RGB
Karpathy et al. [KTS$^+$14]	Improved CNN from large-scale video by extending the connectivity of a CNN in the time domain to gain local spatio-temporal information.	RGB
Carreira et al. [CZ17]	introduced a Two-Stream Inflated 3D ConvNet from the spatial and temporal domains.	RGB
Baccouche et al. [BMW$^+$11]	Proposed the 3D-CNN to capture the nature of video data. The network is trained to allocate a vector of spatio-temporal features to a small number of successive frames and used the feature vectors to classify the entire sequences.	RGB
Latah et al. [Lat17]	Used a 3D-CNN approach to extract spatio-temporal features from adjacent video frames. SVM classifier was used for classifying each instance based on the extracted features.	RGB
Song et al. [SHJ18]	Improved RGB-D scene recognition by extended the existing RGB model of scene recognition, i.e., pre-trained RGB-CNN models and fine-tuning, to the target of the RGB-D data.	RGB-D
Wang et al. [WLG$^+$16]	Applied a 3D-CNN on depth map sequences to propose human action recognition. They extracted the body shape and motion information at several temporal scales by generated weighted DMM. These motion information were configured to be input to CNN.	Depth
Wang et al. [WZC14]	Built the network over the video segments, which was consisted of 3D convolutions and max-pooling operators. The feature map can obtain via performing 3D-CNN to demonstrate the model of 3D activity recognition.	Depth
Asadi et al. [AABR$^+$17]	Proposed a MMDT based on scene flow to describe video sequences for improving human action recognition. This work combined hand-crafted and learning-based features based on 2D-CNN.	RGB-D
Liu et al. [LAM17]	Improved action recognition based on viewpoint invariant features obtained from the DT of the RGB videos and simultaneously extracted view-invariant human pose features based on a CNN model from the depth data.	RGB-D

Table 2.4: Summary of the previous approaches based on Convolution Neural Networks (CNNs) for human action recognition.

In this thesis, a deep structured 3D model from RGB-D video can be viewed as an extension of these existing approaches, especially the method of [SZ14]. This method utilizes the optical flow representation of RGB video as an input to CNN. In Chapter 8, this CNN method is extended to be used with RGB and depth sequences.

Compared to images, the video size is often much large, making it challenging to feed a full video into deep learning architectures that are often very demanding in memory. For several iterations, CNN training requires considerable computing resources.

Researchers are also attempting to learn CNNs on sampled frames or very short video clips. However, video label information at the frame/clip level may be incomplete or even missing. This missing information contributes to the issue of the assigning of the false label. The CNN methods will be explained in detail in Chapter 3, Section 3.6.

2.7 Human Activity Datasets

The essential requirement to develop a human action recognition system using machine learning is to use appropriate human action datasets. These datasets should be sufficiently rich in a variety of human actions. Furthermore, the creation of such a dataset should be corresponding to real-world scenarios. This section presents five popular state-of-the-art action recognition datasets, such as MSR Daily Activity 3D (MSR3D) [WLWY12], Online RGBD (ORGBD) [YLY15], Gaming 3D (G3D) [BMA12], Cornell Activity (CAD-60) [SPSS12], and NTU RGB+D [SNGW16] datasets. The presented datasets are used throughout this thesis work to evaluate the proposed approaches. These datasets have been released for public use and address for these issues. More description of each dataset will be presented in the following sections.

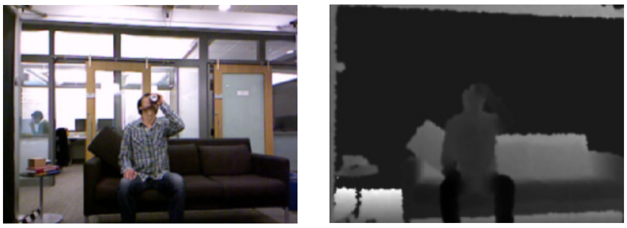
2.7.1 MSR Daily Activity 3D Dataset

The MSR Daily Activity 3D (MSR3D) dataset [19] [WLWY12] was collected by Microsoft and the Northwestern University in 2012. MSR3D dataset was recorded by a Kinect device and focused on daily activities in the living room. In this dataset, there is a sofa in the scene, as well as the camera has been fixed in front of it. This dataset includes 16 actions and ten subjects. Each subject does each activity in two different poses. These activities including: *"Drinking, eating, read a book, call cell phone, writing on a paper, using a laptop, using a vacuum cleaner, cheer up, sitting, still, tossing paper, playing a game, laying down on the sofa, walking, playing guitar, stand up, and sit down"*, can be used for supervised learning. The total number of activity videos is 320 samples. Some examples of activities are shown in Figure 2.18.

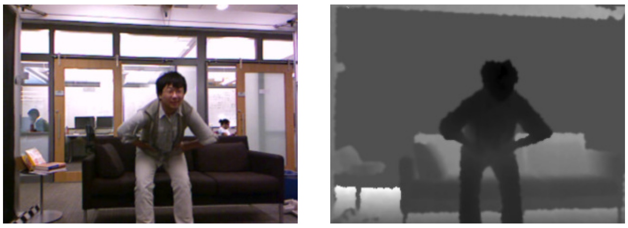
These datasets have ($320 \times 3 = 960$) files in total, where ($16 \times 10 \times 2 = 320$) files for each channel. The RGB channel and depth channel are recorded separately, so they are not accurately synchronized.

2.7.2 Online RGBD Dataset

The Online RGBD (ORGBD) action dataset [16] [YLY15] targets for human action recognition based on RGB-D video data. ORGBD dataset was recorded by the Kinect device and focused on human-object interaction. Each action was performed two times by 16 subjects.



Drinking



Sitting Down

Figure 2.18: Sample frames of MSR Daily Activity 3D dataset, such as Drinking, Sitting down, compare to [19] [WLWY12].

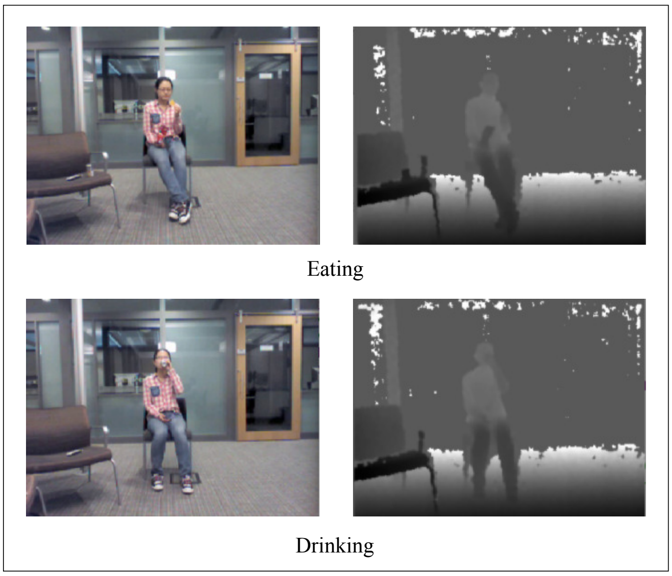
ORGBD dataset consists of 'seven' types of actions captured in the living room, such as: *"Drinking, eating, using a laptop, picking up a phone, reading phone (sending SMS), reading a book, and using a remote"*. The sample frames of the ORGBD dataset are shown in Figure 2.19.

2.7.3 Gaming 3D Dataset

Gaming 3D (G3D) dataset [1] [BMA12] was collected by Kingston University in 2012. G3D dataset was captured by Microsoft Kinect and focused on real-time action recognition in a gaming scenario. This dataset contains ten subjects performing 20 gaming actions, such as: *"Punch right, punch left, kick right, kick left, golf swing, tennis swing forehand, tennis swing backhand, tennis serves, defend, throw a bowling ball, aim and fire a gun, jump, run, walk, crouch, climb, wave, flap and clap, and steer a car"*. Figure 2.20 shows some image examples of G3D actions.

2.7.4 Cornell Activity Dataset

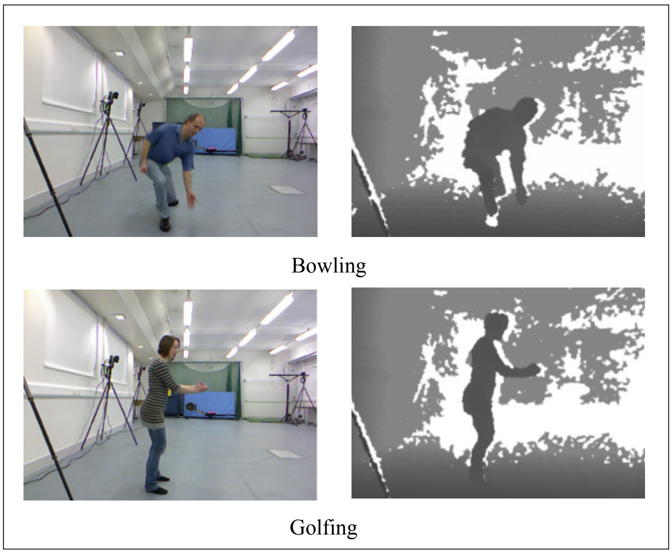
The Cornell Activity (CAD-60) dataset [17] [SPSS11] was captured by Cornell University in 2011 and included the RGB-D video sequences. It is motivated by the fact that true daily activities rarely happen in structured environments. Therefor, the actions were carried out in the uncontrolled background. CAD-60 dataset was captured



Eating

Drinking

Figure 2.19: Sample frames of Online RGBD action dataset, such as: Eating and Drinking, compare to [16] [YLY15].



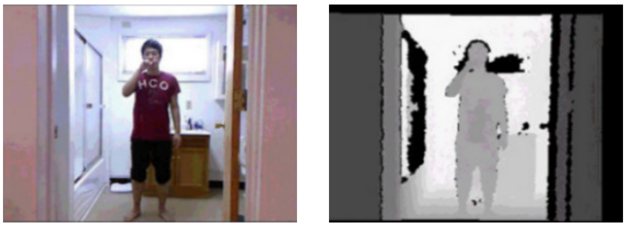
Bowling

Golfing

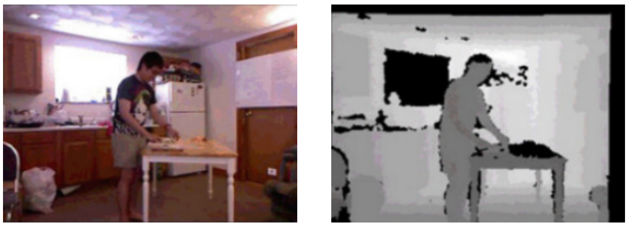
Figure 2.20: Color and depth samples from different gaming actions, such as: Bowling, Golfing, compare to [1] [BMA12].

by Microsoft Kinect, which included distinctive activities that were performed within five indoor environments: Office, kitchen, bedroom, bathroom, and living room.

In CAD-60 dataset, four subjects performed 12 different activities, such as: *"Rinsing a mouth, brushing teeth, wearing contact lens, talking on the phone, drinking water,* *opening pill container, chopping, stirring, talking on the couch, relaxing on the couch, writing on white-board, and working on a computer"*. Some samples from this dataset are shown in Figure 2.21.



Brushing Teeth

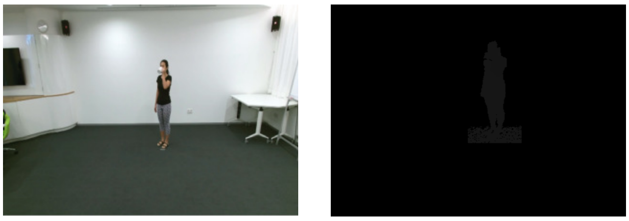


Cooking

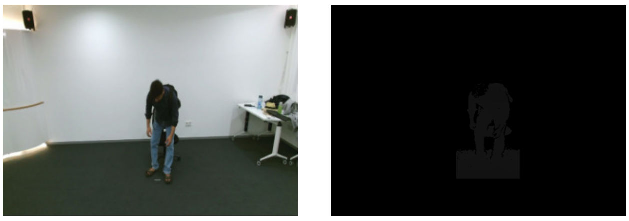
Figure 2.21: Sample frames selected from CAD-60 dataset from different actions. Such as: Brushing teeth, Cooking (Chopping), compare to [17] [SPSS11]

2.7.5 NTU RGB+D Dataset

The NTU RGB+D dataset [18] [SNGW16] was collected by Nanyang Technological University is in 2016. It is one of the largest scale benchmark dataset used for 3D action recognition tasks. It produced 56,880 RGB-D video samples of 60 separate actions. The 60 action classes in NTU RGB+D dataset were listed as: *"Drinking, eating, brushing teeth, brushing hair, dropping, picking up, throwing, sitting down, standing up, clapping, reading, writing, tearing up paper, wearing a jacket, taking off a jacket, wearing a shoe, taking off a shoe, wearing on glasses, taking off glasses, putting on a hat/cap, taking off a hat/cap, cheering up, hand waving, kicking something, reaching into a self pocket, hopping, jumping up, making/answering a phone call, playing with a phone, typing, pointing to something, taking a selfie, checking time on watch, rubbing two hands together, bowing, shaking head, wiping face, saluting, putting palms together, crossing hands in front, sneezing/coughing, staggering, falling down, headache, touching chest, touching back, touching neck, vomiting, fanning self, punching/slapping other person, kicking other person, pushing other person, patting other's back, pointing to the other person, hugging, giving something to other person, touching other person's* *pocket, handshaking, walking towards each other, and walking apart from each other"*. Figure 2.22 shows two sampled from this dataset.



Drinking



Sitting

Figure 2.22: RGB-D dataset images example from NTU RGB+D. On the right side, the depth image contrasts are adjusted to show the captured depth in a better format in the printed version, compare to [18] [SNGW16].

In this thesis, the NTU RGB+D dataset will be used due to the list of suitable datasets implemented for a CNN framework. CNN requires large datasets for training and testing purposes of providing reliable results. On the NTU RGB+D, 60 actions were collected from 40 human subjects in 80 different camera views. Each class contains 948 samples; single class features contain more samples than other action datasets. The dataset can be divided into three parts: *"Daily actions, health-related actions, and mutual actions"*. Different camera views subjects and setups provide better variance between samples of the same class and less artificial scenes.

Table 2.5 illustrates a summary of the presented public RGB-D datasets that will be used to evaluate the proposed contributions of this thesis:

Datasets	Samples	Classes	Subjects	View	Sensor	Modality
MSR Daily Activity 3D (MSR3D) [WLWY12]	320	16	10	1	Kinect 360	RGB-Depth + Joints
Online RGBD Action (ORGBD) [YLY15]	336	7	16	1	Kinect 360	RGB-Depth + Joints
Gaming 3D (G3D) [BMA12]	1467	20	10	-	Kinect 360	RGB-Depth + Joints
Cornell Activity (CAD-60) [SPSS11]	60	12	4	-	Kinect 360	RGB-Depth + Joints
NTU RGB+D [SNGW16]	56880	60	40	80	Kinect v.2	RGB+Depth + Joints +IR

Table 2.5: Summary of human activity RGB-D datasets.

For more details and sample images of different video action from five types of RGB and depth datasets could be found in Appendix A.

Chapter 3

Background Theory

After explaining related work used for action recognition in the previous chapter, this chapter introduces a theoretical background that is followed throughout this thesis work to improve human action recognition algorithms. The most popular state-of-the-art techniques will be reviewed for each step of the typical action recognition. This chapter is organized as follows: An introduction is presented in Section 3.1. Section 3.2 presents the motion detection methods. The features extraction methods are declared in Section 3.3. The video action representation is explained in Section 3.4. The video action recognition methods are presented in Section 3.5. Finally, Section 3.6 explains convolutional neural networks.

3.1 Introduction

Vision-based human action recognition is typically represented in two steps: Feature extraction and action classification. Given a series of video sequences of actions that the prototype will possibly be able to recognize. From each video, the significant features are extracted by using video processing techniques. In recent years, various action recognition techniques [WL11, NVBR12, BCK13] have been proposed. Most of them refer to the group of features extraction-based methods, as explained in Chapter 2, Section 2.4.

After the feature extraction step, the next step of human action recognition is the classification of action that has been described by appropriate feature sets [H́D11, SBL14, Tub17] extracted from images or videos. In this step, classification algorithms give a final result of the activity label. Classification algorithms keep evolving with machine learning methods. In recent years, the deep neural networks approaches, e.g., convolutional neural networks, are used for the classification tasks from images and videos [RGF$^+$18, LCX$^+$19].

3.2 Motion Detection

Motion detection from videos is a method to identify activity in a scene by observing the variations in the image sequence. Usually, this is achieved by matching pixels or frame references. Any changes between frames are considered as a detection. The resultant information forms the basis for high-level operations that involve well-segmented results, such as object classification and video action/activity recognition.

However, motion detection suffers from difficulties due to complex backgrounds, illumination variations within the scene, and moving objects. Various methods were suggested to solve these issues by utilizing only the moving object of interest. These methods are classified into three categories: Background subtraction, optical flow, and Retina model.

3.2.1 Background Subtraction

To detect a moving object from video sequences, a background subtraction method (object detection) is used. It is one of the popular methods for motion detection in video sequences [MR14]. The background subtraction concentrates on two steps: Firstly, build a statistical representation of the background, which is more representative, robust to noise, and sensible to new objects. Secondly, construct another statistical model called foreground representing the changes that occur on scene [TK12, MR14]. By applying the Background subtraction approach to each frame on video, the moving object can be tracked effectively.

Background subtraction creates a foreground mask for every frame in the video by subtracting the background image from the current frame. Furthermore, if the background view excluding the foreground objects is available, the foreground objects can be acquired by differentiating the background image from the current frame.

Background modeling consists of two significant steps: Background initialization and background updating. In the first step, an initial background model is determined, and in the second step, the model is updated to adapt potential changes in the scene. The purpose of using background subtraction is to initialize the background image and update it. The efficacy of the two will impact the accuracy of the test results. Regarding the initialization step, the background is estimated to be the previous frame. By using frame differencing, the background subtraction equation then becomes:

$$|I(x, y, t) - I(x, y, t-1)| > Th, \tag{3.1}$$

where, the previous frame, also known as the background frame, is represented as $I(x, y, t-1)$ and deducted from the current frame, $I(x, y, t)$. The threshold (Th) value is then utilized to the complete difference in order to receive the foreground mask. The main parameter in the thresholding method is the preferred threshold value. This value can be selected to be either automatic or manual. The reason for using the background subtraction method is to detect the foreground or objects in motion from the compression between the background frame and input video frame.

In real-time, the background needs to be updated so that the moving object is accurately extracted and the background model adapted better to light changes. Thus, the Mixture of Gaussian (MoG) algorithm is chosen to be updated every frame. The MoG algorithm description could be found in [Ziv04].

The background subtraction method [8] will be used later in Chapter 5 for extracting the moving object from the input video sequences. The moving object could be extracted from the background as in Equation (3.2), in this form, if the pixel difference is greater than the set threshold, it then locates that the pixels appear in the moving object, otherwise, as the background pixels. Then, the moving object can be observed after threshold process.

$$B(x,y,t) = \begin{cases} 1, |I(x,y,t) - I(x,y,t-1)| > Th \\ 0, \quad \text{otherwise} \end{cases}, \tag{3.2}$$

where, $B(x,y,t)$ represent the binary images. If $B(x,y,t) = 1$, this means that the pixel belongs to the foreground, and otherwise, the pixel belongs to the background.

3.2.2 Optical Flow

The optical flow (OF) method is used for detecting motion in a visual scene caused by the action of objects within a scene. The OF is estimated by calculating each pixel's motion velocity and direction between two successive video frames or images caused by an object or camera movement. OF is widely used in video detection due to its robust ability to describe motions in the video [YB98, WLG$^+$16].

Optical flow defines both the orientation as well as the velocity of motion. Using color intensity values, approximate the displacement of each image pixel over time in a video volume. Several methods measure the field of optical flow (u,v) based on optimization techniques such as differential-based, region-based, phase-based, and energy-based techniques [BFB94]. The most common and widely used methods in the literature are represented by Lucas et al. [LK81] and Farnebäck [Far03].

As an example, given a video sequence of images I, the optical flow's goal is to measure the pixel displacement $(\delta x, \delta y)$ of spatial coordinates (x,y) in the interval $[t, t+\delta t]$, so the following constraint of brightness is approached.

$$I^{t}_{x,y} = I^{\Delta t+t}_{\Delta x+x,\Delta y+y} \tag{3.3}$$

In the case of subtle motion, the Equation (3.3) can be reformulated as:

$$I^{\Delta t+t}_{\Delta x+x,\Delta y+y} \approx I^{t}_{x,y} + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t. \tag{3.4}$$

By using Equation (3.3) and Equation (3.4), it follows that:

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0, \tag{3.5}$$

or, by dividing with Δt, obtaining the following:

$$\frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t}\frac{\Delta t}{\Delta t} = 0, \tag{3.6}$$

which is expressed as:

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0, \tag{3.7}$$

where (u, v) are components of optical flow or velocity of $I^{t}_{x,y}$ and $(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, \frac{\partial I}{\partial t})$ are the partial image derivatives in the (x, y, t) position. Since only one equation is given with two unknown variables, u and v, additional restrictions are necessary, and neighboring pixels are usually provided.

In this thesis, the optical flow represented by Lucas-Kanade differential technique [LK81] and Farnebäck method [Far03] will be applied as a motion feature detector. These two methods represent the most popular methods used for optical flow computation efficiency.

Lucas-Kanade Method The optical flow representation defines the velocity of each pixel point individually in the image. However, it typically only defines the motion of two successive image frames and cannot adequately represent the entire duration of the action. The above equations are called the optical flow equation. In it, ∂x and ∂y can be computed as image gradients. Similarly, ∂t is the gradient along time. However, (u, v) is unknown, so, Lucas-Kanade is used to solve the problem of unknown variables. The mathematical computation of the Lucas-Kanade optical flow method could be found in [LK81].

The Lucas-Kanade differential method is one of the most commonly utilized methods due to its calculative effectiveness. It is called a sparse optical flow because it tracks only a specific number of points in the image. The Lucas-Kanade differential method is a spatial local least square approximation based on a constant-brightness premise in a local pixel neighborhood. This method will be used later in Chapter 4 as a motion detector to extract local features from RGB-D videos. The sparse optical flow is calculated using the Lucas-Kanade optical flow algorithm [11, 12] in the OpenCV library.

Farnebäck Method The Farnebäck method computes a dense optical flow based on the Gunnar Farnebäck algorithm. Farnebäck algorithm creates flow information for each pixel in the video source frame. The dense optical flow measured in a video between two consecutive frames. However, such computation may be slower than the Lucas-Kanade method, but it gives accurate results. These denser results are suitable for applications such as learning structure from motion and video segmentation [RF16]. The mathematical computation of the Farnebäck optical flow method could be found in [LK81]. This dense optical flow method will be used later in Chapters 5, 7, and 8 as a motion detector from RGB-D videos. The optical flow is calculated using the Farnebäck optical flow algorithm [11, 12] in the OpenCV library.

3.2.3 Retina Model

The Retina model is a bio-inspired booster system used to address the difficulties of the unsupervised segmentation of video moving objects. A large number of image processing modules influenced by human visual systems' biological models have been investigated [BCDH10]. The ultimate purpose is to copy the human visual system's potential for recognition. The human Retina in image processing allows for the reduction of noise and illumination differences as well as the enhancement of static and dynamic contours. This technique may also be used for the normalization of illumination and motion detection. The global architecture of the implemented Retina model [MCCRAC16] is shown in Figure 3.1 as a fusion of low-level processing modules. In essence, it is a layered model with:

- Photoreceptors can adjust the sensitivity of their neighborhood luminance, where the local contrast is improved.
- Outer plexiform layer (OPL) eliminates spatio-temporal noise and increases the high-frequency ratio of spatial contours while reducing and removing mean luminance.
- Inner plexiform layer (IPL) in which different channels of information can be identified. This thesis will be focused on two well-known channels: The Parvocellular (Parvo) channel, dedicated to spatial analysis that increases the contrast between static contours, and the Magnocellular (Magno) channel that strengthens moving contours and eliminates static ones.

In the human Retina, the fovea level (central vision) is described by the Parvo channel. In contrast, the Magno channel was significant outside of the fovea level (peripheral vision) due to specialized cells' relative variations. Taking into consideration both channels of information in the same area of the image can be exciting for computer vision because detail and motion data are accessible as parallel information in the same area. The OPL and IPL [BCDH10] are illustrated in the following.

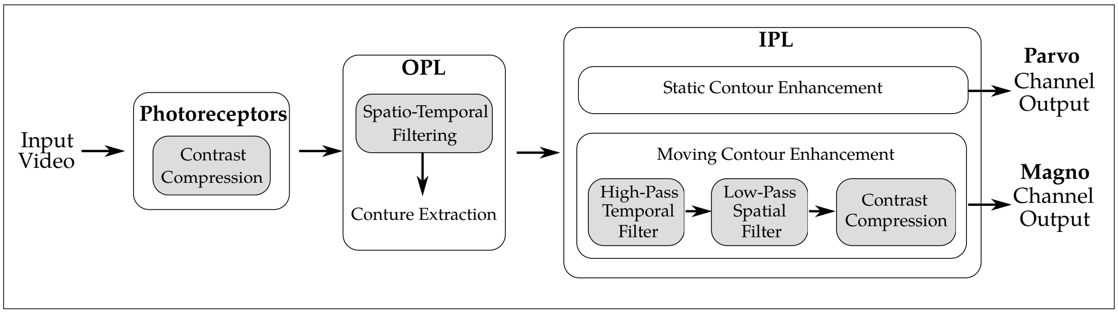


Figure 3.1: The Retina model [MCCRAC16].

OPL: Spatio-Temporal Filtering and Contour Enhancement

The OPL layer cellular cooperations can be modeled using a non-separable spatio-temporal filter whose transmission function for an electrical signal. The Retina-inspired is developed to be used in images and video processing, which is applied to each frame of a video stream, $I(x, y, t)$. So, the OPL filter $(R_{OPL}(f_x, f_y, f_t))$ [Her96] has a transfer function of the form:

$$\begin{aligned}
&R_{OPL}(f_x, f_y, f_t) = R_{ph}(f_x, f_y, f_t)[1 - R_h(f_x, f_y, f_t)]\\
&\text{with}\\
&R_{ph}(f_x, f_y, f_t) = \frac{1}{1+\beta_{ph}+2\alpha_{ph}(3-\cos 2\pi(f_x+f_y\sqrt{3})-\cos 2\pi(f_x-f_y\sqrt{3})-\cos 2\pi f_x)+j2\pi \mathrm{T}_{ph} f_t}\\
&R_h(f_x, f_y, f_t) = \frac{1}{1+\beta_h+2\alpha_h(3-\cos 2\pi(3f_x+f_y\sqrt{3})-cos 2\pi(3f_x-f_y\sqrt{3})-cos 2\pi f_y\sqrt{3})+j2\pi \mathrm{T}_h f_t},
\end{aligned} \tag{3.8}$$

where f_x, f_y and f_t represent spatial and temporal frequencies, these frequencies values are computed using the Fourier transform function [Lev46]. The R_{OPL} filter can be used as a difference between two low-pass spatial and temporal filters that model the network of photoreceptor (R_{ph}) and the horizontal cell network (R_h) of the Retina. The output of the horizontal cell network (R_h) is a very low spatial frequency. It can also be translated as the local luminance required in the local adaptation stage of the photoreceptors [BAHLC09]. In addition, the R_h filter's temporal low pass influence enables local luminance estimation to be permanently smoothed. Finally, the global R_{OPL} filter typically has a spatio-temporal high-pass effect on low frequencies, which contributes to a spectral whitening of the input. Its high-frequency low-pass effect tends to eliminate structural noise.

The R_{OPL} spatio-temporal filter involves several parameters: β_{ph} is the gain of R_{ph} filter. Setting β_{ph} to 0 helps in canceling luminance information, and a higher value requires partial processing of the luminance; β_{ph} is usually set to 0.7, which allows for a strong overall impact. β_h is the gain of the R_h filter. By setting this parameter to 0, only the contour information is extracted. On the other hand, increasing it allows the low-frequency information to be less reduced, i.e., the luminance would not be canceled.

T_{ph} and T_{h} are temporal filtering constants created by the cellular membranes' effects, allowing the temporal noise to be minimized. The spatial filtering constants (α_{ph} and α_{h}) are setting the spatial filtering capacities: Where α_{ph} and α_{h} sets the high and the low cut frequency, respectively. The OPL filter can eliminate spatio-temporal noise and increase contours. These two properties are often opposite since noise creates distracting contours, which often relate the enhancement of contours with the enhancement of noise. More details about the Retina model could be found in [Her96].

IPL: Contours Enhancement and Motion Dedicated Filtering

Contours enhancement: Parvo channel IPL Parvo is the next processing step of the OPL and applies to part of the Retina IPL, and it provides high contour sensitivity in foveal vision. The Parvo channel ganglion cells receive contour information from the OPL outputs. On this basis, they serve as a local enhancer that boosts the contour details. Parvo channel is also modeled similar to the photoreceptor by logarithmic law [BCDH10].

The Parvo channel processes colors and spatial information. It has a high resolution in the middle of the visual field and forms the foveal vision. It normalizes colors, increases local contrast, responds well to temporally sustainable signals, and fluidized rapid temporal variations. The parvo channel enhances the image in many ways. It improves the contours of medium spatial frequencies, which reflect spatial information and leave high-frequency components correlated with noise and compression. It also smooths movement, removes high temporal frequencies that can be easily affected by noise and video compression. An additional advantage is the local luminance change, which also increases details in very dark areas of the scene without amplifying the image noise [SBLC12].

Motion dedicated filtering: Magno channel On the Magno channel of the IPL, let's an input image is represented by $I = (x, y, t)$, on its output image, a temporal effect is introduced. This effect is modeled by a first-order high-pass temporal filter [Cha11]. This filter improves areas in which spatio-temporal changes occur. The high pass temporal filter of the IPL Magno channel provides the output signal with a temporal effect, clearly noticeable as a trace left by the moving objects.

The magnitude of the IPL Magno output signal depends on the velocity of the moving regions, i.e., high reactions in fast-moving regions and no reaction in static regions. The filter reaction is also more robust for moving contours perpendicular to the direction of motion. The tuning of the time constant allows the reaction to temporal changes in the scene to be modified. A low value only allows for accelerated changes, while a higher value allows slower changes to be improved. It affects not only the response to contours of moving objects but also parasite background movement. The response decays over time that contributes to fuzzy contours. The IPL Magno facilitates robust extraction of motion information with the benefit of using locally adapted contour extraction, even in dark or noisy environments.

The Parvo and Magno channels will be used in this thesis (Chapter 6) as spatio-temporal filtering to segment the motion from input videos. It is possible to apply the Retina filter to single pictures, video files, or live video captures [BCDH10]. The software of the Retina model is based on OpenCV [13], which offers a simple and compact library for image processing. In the thesis work, the Retina filtering is performed with C++ programming, and the display allows the processing result to be displayed in real-time.

3.3 Feature Extraction Methods

Low-level features play an essential role in feature representations of human actions. In the last decades, many spatio-temporal feature detection and description methods have been proposed and shown to be useful for action recognition. CNNs, in contrast, ca learn feature extractors automatically. This will also discussed later. Hand-crafted features, however, still play an important role in action recognition for state-of-the-art systems. This section focuses on the hand-crafted feature extraction methods that will be used later in this thesis.

3.3.1 Local Spatio-Temporal Features

This section describes local spatio-temporal feature detectors and descriptors. A detector is an algorithm that selects points from an image depending on some criterion. These points are called interest points (keypoints or salient points). A descriptor describes an image patch around the interesting points and represents it as a vector of values, these values could be as simple as the raw pixel, or it could be further complicated, such as a histogram of gradient orientations. Together interesting points and its descriptor are commonly called a local feature. A local feature is used for many computer vision tasks, such as object detection, object recognition, and image registration [KG13, GGDH17, LK81].

Spatio-Temporal Feature Detector

Interest points in videos are spatio-temporal interest points (STIP) within a video that can be used to build complex video processing systems such as action recognition and video retrieval systems [NVY$^+$14]. For computing the STIP points, the detector is usually used to select spatio-temporal locations and scale them in a video by maximizing specific saliency functions, such as Speeded-Up Robust Features (SURF) [BETV08].

SURF is a very fast detector and extracts features by tracking interest points. Feature detectors usually differ in the type and the sparsity of selected points [HAD14]. In this thesis, a SURF feature detector will be used to extract interest points from videos.

Speeded-Up Robust Features: The SURF detector was firstly proposed by Bay et al. [BETV08]. SURF provides a robust local feature detector and descriptor, which can be used in computer vision tasks such as object recognition, camera calibration, 3D reconstruction, and image registration [RTKI11, SBLC12, HAD14, JKAM18]. The SURF detector is an algorithm that extracts some unique interest points, i.e., keypoints, from an image or video sequence. These interest points are chosen as corners and blobs at distinctive locations in an image. It uses an intermediate image representation called Integral image [VJ01] to reduce the computation time. The Integral image is calculated from the input image. It is used in any rectangular area to speed up the calculations. The value of the Integral image is calculated by summing up the pixel values of the (x, y) coordinates from the source to the end of the image, which helps to make the computation time-invariant to adjust the size and is especially helpful when encountering large images. An Integral image $(I_{\sum}(\mathbf{x}))$ at a position $((\mathbf{x}) = (x, y)^{\top})$ refers to the sum of all pixels in an input image (I) within a rectangular area of the origin and $(\mathbf{x})$ as shown in Equation (3.9) [BETV08].

$$I_{\sum}(\mathbf{x}) = \sum_{i=0}^{i\leq x} \sum_{j=0}^{j\leq y} I(i, j) \tag{3.9}$$

The SURF feature detector is based on a determinant of the Hessian matrix [Lin91] for both scale and location due to its excellent performance of accuracy, it is used for deciding whether a point in an image can be chosen as an interesting point or not. The SURF detector uses to detect the interest points later in the next chapter.

Spatio-Temporal Feature Descriptors

A spatio-temporal feature descriptor represents an image or an image patch that simplifies the image by extracting essential and important information and throwing out extraneous information. A feature descriptor typically transforms an image of size $(w \times h \times N_{Ch})$ to a feature vector of size $(\mathbf{s})$. Where $(w$ and $h)$ represent the width and height of the image. N_{Ch} refers to the number of image channels. In thesis work, these channels are represented by RGB and depth channels, which means that there are four channels or gray and depth channels, i.e., two channels.

In this thesis, different feature descriptors are used to extract feature vector values from RGB-D videos, as explained in details below.

Histogram of Oriented Gradients (HOG) The HOG is a feature descriptor that is used in image processing and computer vision for detecting objects. The HOG descriptor technique counts orientation and gradients occurrences in localized parts, i.e., the complete image is broken into smaller regions. The gradients and orientation are then calculated for each region of an image detection area or Region of Interest (ROI) to extract the dense features from the images. Dense means that it extracts features for an ROI in the image or all locations in the image.

The HOG descriptor is introduced by Dalal et al. [DT05] for RGB images. It is currently regarded as one of the most important and widely used methods for visual human detection and recognition [SEE18, KGDE19]. HOG descriptor is experimentally proved to outperform other features for encoding human figures in human recognition. Simultaneously, the edges and interesting regions can be effectively used to encode object shapes and areas that are widely used for action representation.

To establish structure information in a descriptor [LMSR08], gradient magnitude responses have to be computed horizontally and vertically. It results in a 2D vector field per frame. In practice, the image is divided into small connected regions, called cells, the local neighborhood around a local feature is divided into a spatio-temporal grid. For each cell of the grid, compiled a histogram of edge orientations or gradient directions for the pixels within the cell. The histograms are then normalized using L2-normalization and concatenated the b-bins histograms over B-blocks to represent the final descriptor vector. Afterward, these responses need to be aggregated over blocks of pixels within both spatial and temporal directions. Then, the following step is to concatenate the responses of many adjacent pixel blocks. Finally, descriptors need to be normalized to reduce dimensionality, which leads to computational benefits or improved accuracy. This descriptor is used for the extraction of the features in video frames.

Dependent on the idea of HOG, Histograms of Oriented using depth data have been computed by [SA11], by following the same procedure of HOG descriptor on RGB images. It analyses a subdivision of a fixed region into cells, estimates descriptors per cell, and collects the oriented depth gradients into histograms. The histograms normalize with respect to depth noise to achieve a high level of robustness. The assumption is that a number of local depth variations will robustly describe local 3D shape and appearance.

Histogram of Optical Flow (HOF) The HOF is the most common local feature descriptors used for video processing introduced by Laptev et al. [LMSR08]. The optical flow displacement vectors are initially calculated horizontally and vertically. OF has been commonly used in video detection, as explained previously, due to its robust ability to describe motions in the video [YB98, WLG^{+}16]. OF is estimated by calculating each pixel's motion velocity and direction of two successive video frames induced by object or camera movement. To embed structured information in the HOF descriptor, follow the same idea as in the HOG descriptor, the local neighborhood surrounding a

local feature is divided into a spatial and temporal grid. A histogram descriptor is computed from each pixel in the interest region for each cell of the grid. The histograms are then normalized and concatenated into the final descriptor.
For representing local motion features, HOF descriptor is frequently used to describe the local neighborhoods of the detected interest points. The HOF features are calculated from each pixel in the region of interest. The optical flow magnitude of each pixel in the region of interest is determined into b-bins by their optical flow directions to achieve bins histogram represented as a HOF features. The HOF and HOG descriptors will be used later in the next chapters.

Motion Boundary Histogram (MBH) The MBH is a motion descriptor used for video classification task [UDSS15]. MBH descriptor is proposed by Dalal et al. [DTS06], who demonstrated its effectiveness to background and camera motion. The main idea of MBH is to describe the oriented gradients determined across the vertical and the horizontal optical flow components, i.e., separates the optical flow (OF) field into its (x and y) components. The benefit is that locally continuous motion movements of the camera tend to be avoided, and the description relies on optical flow variations between frames (motions boundaries). Spatial derivatives for the horizontal and vertical components of the optical flow are determined separately, and orientation information is quantized into histograms similar to the HOG descriptor. In addition, the MBH descriptor is based on the spatio-temporal grid concept by creating histograms of Bag-of-Features (BoFs). The MBH in thesis work is not only used for its aptitude of decreasing the camera movement but also as a motion descriptor for its action recognition, i.e., MBH employs as motion descriptor for Dense Trajectory. The MBH descriptor will be used in Chapter 5 to extract motion information from depth data.

Dense Trajectory (DT) DT was proposed by Wang et al. [WKSL11] for RGB data, i.e., 2D Dense Trajectory (2DTr). In this thesis, DT will be expanded to be used on RGB-D data by considering the depth information and motion features from RGB-D videos.

Wang et al. [WKSL11] proposed the efficient dense trajectories to describe videos motivated by the current success of dense sampling in image classification. Feature points for each frame in videos are sampled on a grid spaced by w pixels and tracked separately in each scale. According to video resolution, each frame is set to eight spatial scales by a factor $1\sqrt{2}$ videos. The dense field of optical flow between frame t and the next frame $t+1$ is $\mathcal{O}_t = (u_t, v_t)$, where u_t and v_t represent the vertical and horizontal components. For the feature point $\boldsymbol{X}_t = (x_t, y_t)$ at frame t, it can be tracked to $\boldsymbol{X}_{t+1} = (x_{t+1}, y_{t+1})$ at the next frame $t+1$ by by means of median filtering in the dense optical flow field $\mathcal{O}_t$, and the position of $\boldsymbol{X}_{t+1}$ is represented as in Equation (3.10):

$$\boldsymbol{X}_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (M * \mathcal{O}_t)|_{(x_t, y_t)}, \tag{3.10}$$

where M is the 3×3 kernel of median filtering. Points of subsequent frames are concatenated to form a trajectory: $(\boldsymbol{X}_t, \boldsymbol{X}_{t+1}, \boldsymbol{X}_{t+2}, ...)$. However, in tracking, there is a very common problem like drifting. To prevent this circumstance, the trajectory length is limited to Ls. If the tracking point is not located in a $w \times w$ neighborhood of each frame, a feature needs to re-sample and adds it to the tracking process.

DT provides a video representation depend on the densely sampled trajectories using dense optical flow and a set of feature descriptors such as HOG to represent the spatial appearance, HOF to extract the information of first-order motion, and MBH to extract the information of second-order motion. The dense sampling method is used for capturing local information from both the foreground and its surrounding objects, i.e., this method takes into consideration both motion appearance features.

DT algorithm computes DT over multi-scale images. A first step calculates dense sampling of feature points across the first frame to assure that feature points include all spatial positions and scales. A second step tracks points depend on the displacement information from a dense optical flow field to describe video frames. The DT descriptor extended the scheme of motion coding based on MBH which is developed in the human detection field and overcame camera motion problem and overlay the motion information in videos.

The Trajectory shape descriptor [WKSL11] is used to encode a local motion pattern and shape characteristics of the extracted DT. So, it describes the shape of a Trajectory by a series of displacement vectors normalized by the sum of vector magnitudes displacement. Mathematically, the Trajectory shape $(\boldsymbol{Tr})$ can be described by length L_s and a sequence of displacement vectors $(\Delta\mathbf{X}_t, \cdot\cdot\cdot, \Delta\mathbf{X}_{t+L_s-1})$ where $\Delta\mathbf{X}_t = (\mathbf{X}_{t+1} - \mathbf{X}_t) = (x_{t+1} - x_t, y_{t+1} - y_t)$. Then, normalizing the resulting vector by the sum of displacement vector magnitudes:

$$\boldsymbol{Tr} = \frac{(\Delta\mathbf{X}_t, \cdot\cdot\cdot, \Delta\mathbf{X}_{t+L_s-1})}{\sum_{j=t}^{t+L_s-1} \parallel \Delta\mathbf{X}_j \parallel} \tag{3.11}$$

The output vector $\boldsymbol{Tr}$ represents the trajectory descriptor. This descriptor is evaluated to represent Trajectories at multiple temporal scales for recognizing actions. Later in Chapter 5, the extraction of the feature Trajectory from the RGB-D image is calculated by extending the 2D to 3D Trajectory by adding depth information to each Trajectory point.

Local Binary Pattern (LBP) The LBP is a type of visual descriptor for texture classification introduced by Ojala et al. [OPH96, OPM02]. The LBP is an image operator that converts an image into an array or image of integer labels that describe the image's small-scale appearance. These labels or their statistics, usually the histogram, are then utilized for additional image analysis.

The LBP has been widely utilized for segmentation and texture classification because it is simple to use and efficient in defining the local spatial structures in an image. Additionally, it has been used in various computer vision applications, including image analysis, environment modeling, image retrieval, motion analysis, and outdoor scene analysis [WZC14, VHS15]. In recent years, LBP is used for human action recognition [ATKI14, CZL17] that also will be applied later in this thesis for computing the feature vectors. LBP operator identifies the image pixels with decimal numbers that encode the local structure around each pixel. LBP can be computed by taking a threshold of neighboring pixels with the gray value of their center pixel and the neighborhood in a (3 × 3) grid, Figure 3.2 shown an example of this computation, wherein a binary case, the original (3 × 3) neighborhood is thresholded by a value of the center pixel. The pixel values in the threshold neighborhood are multiplied by the weights given to the corresponding pixels. Finally, the 8-pixel values are then summed to get the number (149) of this texture unit.

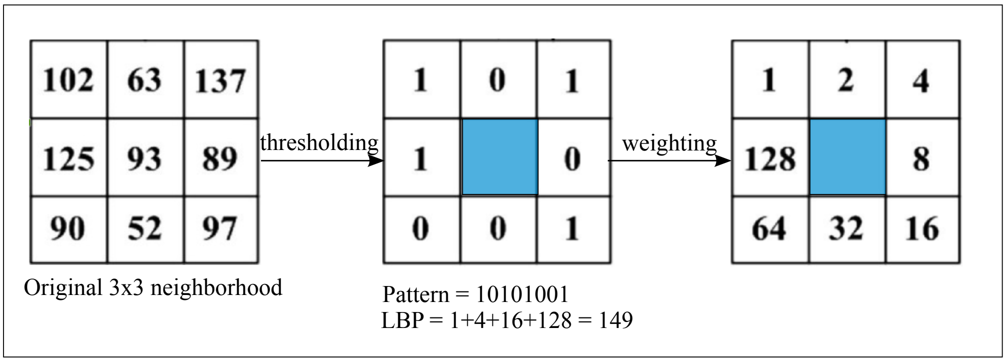


Figure 3.2: Illustration of the original LBP

The LBP method is invariant against a grayscale transformation. It can be conveniently combined with a simple contrast calculation by calculating the difference between the average gray level in each neighborhood of those pixels with the value (1) and those who have the value (0). Each pixel in the frame is compared with its eight neighbors. The resulting 8 values are then treated as an 8-bit binary number. A binary number is obtained by combining all these binary codes from the top-left pixel in a clockwise direction and using their corresponding decimal value for labeling. The obtained binary numbers are referred to a LBP codes. The original LBP at the position (x_c, y_c) can be defined in Equation (3.12) [KZP08]:

$$LBP_{x_c,y_c} = \sum_{s=0}^{P-1} B(g_s - g_c)2^s, \quad B(d) = \begin{cases} 1, d \geqslant 0 \\ 0, d < 0 \end{cases} \tag{3.12}$$

where g_c is the gray value of the center pixel (x_c, y_c) and g_s $(s = 1, 2, ...P)$ are the gray values at the P sampling points. And *B* is the binary value assigned to the neighboring pixel.

The LBP descriptor will be used in Chapter 6 to compute feature vectors form a salient area of the Retina model.

3.3.2 Global Spatio-Temporal Feature

The global features analysis approach has been applied in many document image analysis and recognition researches. Global representations encode extracted features as a whole and are obtained in a top-down manner [SG13]. Therefore, global descriptors are usually less time consuming to calculate and easy to implement compared to local methods; they give robust results in less challenging scenarios such as those with a static background.

Moreover, shape information-based feature is one of the first characteristics utilized to demonstrate human body structure and its dynamics for action recognition in videos, such as Motion History Images (MHI). MHI and different feature descriptors are used in this thesis to extract global feature vector values from RGB-D videos, as will be explained in the following:

Motion History Images (MHIs) MHIs proposed by Bobick et al. [BD01], are a real-time motion template which temporally layers successive image differences into a static image template. The motion history function is represented by pixel intensity at that location, where lighter values correspond to a more recent movement. The directional motion information can be directly calculated from the intensity gradients in the MHI. The gradients in the MHI are more practical to calculate relative to the optical flow. It is also reliable because the motion information in MHI is mostly along the contours of moving objects. Unwanted motion is thereby ignored in the inner regions of object contours. So, the reason for using the MHI is simplicity and low computation compared to the optical flow method. MHI has utilized motion shape information for recognizing actions from a video.

In a video sequence, the MHI translates 3D space-time information to a single 2D intensity image. In this process, the background in each image of the video sequence is subtracted to segment the foreground region. Each foreground pixel is then allocated a significant fixed intensity value that specifies the duration of action. Over time, it is minimized by a small constant value when the pixel becomes a background point. At each pixel location, the intensity value in the MHI thus records the history of temporal changes [HHLH11]. The pixel intensity in MHI M_{H_τ} is a function of the temporal history of the movement at that point. The MHI is formally defined as in Equation (3.13) [BD01]:

$$M_{H\tau}(x,y,t) = \begin{cases} \tau, & \text{if} \quad B(x,y,t) = 1 \\ \max(0, M_{H\tau}(x,y,t-1) - 1) & \text{otherwise} \end{cases} , \tag{3.13}$$

where $B(x, y, t)$ is a binary image of differences between frames that indicates the presence of moving objects at time frame t. The duration τ determines the temporal extent of the movement (e.g., in terms of frames). All detected foreground points, i.e., $B(x, y, t) = 1$, observed in the MHI representation have the same intensity value τ, irrespective of movement durations and moving speeds at individual pixels. Therefore, it is extremely susceptible to background noise and cannot characterize the local movements of a target object. The MHI will be used in the next chapter as a local filtered motion from the detected interest points (using SURF) and as input image to compute the moment-based features (using seven Hu-moments).

Hu-Moments Invariant Features Moments invariant are a shape descriptor that can be used over an image to characterize and describe the shape of an object. Hu Ming-Kuei [Hu62] has suggested the utilize of invariant moments for binary shape representation. The moments are invariant terms of rotation, scales, and translations, known as Hu-moments invariants. Hus' method had seven invariant moments, and they are computed to describe the shape feature dependent on the normalized central moments of the edge image.

Seven Hu-moment shape features are computed as follows: For two-dimensional $(M \times N)$ images of a density distribution function $I(x, y)$; where, $x = 0, 1, ..., M - 1$ and $y = 0, 1, ..., N - 1$, the geometric moment m_{pq} of $I(x, y)$ is computed as follows [RYS+06]:

$$m_{pq} = \sum_{x=0}^{x=M-1} \sum_{y=0}^{y=N-1} (x)^p.(y)^q I(x,y), \tag{3.14}$$

for $p, q = 0, 1, 2, 3, 4, \ldots..$, where p, q are positive integers and $(p + q)th$ is called the order of the moment of a density distribution function $I(x, y)$.

The moments of $I(x, y)$ translated by a quantity (a, b), are computed as:

$$\mu_{pq} = \sum_x \sum_y (x+a)^p.(y+b)^q I(x,y), \tag{3.15}$$

Then, to make these moments invariant to translation, the central moment μ_{pq} can be defined from Equation (3.16) as follows, by substituting the values $a = -\bar{x}$, and $b = -\bar{y}$:

$$\mu_{pq} = \sum_x \sum_y (x+\bar{x})^p.(y+\bar{y})^q I(x,y), \tag{3.16}$$

where,

$$\bar{x} = \frac{m_{10}}{m_{00}}, \bar{y} = \frac{m_{01}}{m_{00}}$$

Moreover, the scaling invariance of the central moment can be computed from normalizing moments of the scaled image by scaled energy of the original image to become invariant to scale change. Mathematically, it can be defined as follows:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu^{\gamma}{}_{00}}, \gamma = \frac{p+q}{2} + 1 \tag{3.17}$$

where γ is the value of the normalization factor.

The value of η_{pq} represents a set of nonlinear functions that are calculated by normalizing central moments, which are invariant to object rotation, translation, position, and scale change. The seven Hu-moments are derived as in Equation (3.18) [RYS$^+$06, KL90]:

$$\begin{aligned}
HM_1 &= \eta_{20} + \eta_{02}, \\
HM_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2, \\
HM_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2, \\
HM_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2, \\
HM_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
&\quad +(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[(\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2], \\
HM_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + \\
&\qquad 4\mu_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}), \\
HM_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})] + \\
&\quad (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} - \eta_{03}^2]
\end{aligned} \tag{3.18}$$

where the numerical values of HM_1 to HM_6 are very small. To prevent precision problems, the logarithms of the absolute values of these six functions, i.e., $\log |HM_i|$; where, $i = 1, .., 6$, are chosen as the action features among video frames. In this calculation, the first six moments have been proved to be invariant to translation, scale, and rotation, while the seventh-moment sign changes for image reflection. The seven Hu-moments feature will be applied in Chapter 4 for computing feature vectors from the MHIs, i.e., a vector of seven Hu-moments is calculated for each frame (MHI) in the video.

GIST Descriptor The GIST is a global image feature descriptor that helps to characterize various important statistics of a scene. The GIST of a scene is influenced by the ability of human to recognize different image information in a few milliseconds or in a single glance [OT01]. For instance, even when an image is blurred, image characteristics such as openness, expansion, naturalness and robustness can be easily recognized without knowing objects or scene details. A so-called GIST global image descriptor has been proposed to model those abilities of the human brain.

The first GIST descriptor was proposed by Olive et al. [OT01] and used it in scene classification. The GIST has been shown that general image properties can be computed based on spectral information that is coarsely located within the image. Discriminant spectral models were introduced, which represent various types of structures in the image. These templates have been shown to describe the properties of a scene. The scene description based on these scene properties is called the spatial envelope. The concept of spatial envelope mentioned in the study is a low-dimensional representation of a scene showing the correlation between the framework of the surface and the properties of the objects in it [OT01]. Moreover, the GIST has also been proven to be effective in object recognition [TMFR03]. Inspired by these works, the GIST feature is adopted into the video domain for human action recognition [WLJ13, WLJL15].

GIST descriptor extracts structural information through filtering an image with different scales and orientations. According to Equations (3.19) and (3.20) [WLJL15], Gabor filter transfer functions are adopted for filtering an image with various orientations and spatial resolution. The image is broken down into grids. The average filter effect is taken from each grid. All the averages are finally connected into a vector to represent the feature description of the whole image [WLJ15].

$$Gabor^{\hat{l}}_{\theta_i}(x,y) = \exp\left(\frac{-(x^2_{\theta_i}+y^2_{\theta_i})}{2\sigma^{2(\hat{l}-1)}}\right)\cos(2\pi(x_0x_{\theta_i}+y_0y_{\theta_i})), \tag{3.19}$$

and

$$\begin{cases} x_{\theta_i} = x\cos\theta_i + y\sin\theta_i \\ y_{\theta_i} = -x\sin\theta_i + y\cos\theta_i, \end{cases} \tag{3.20}$$

where x_0 and y_0 are the frequency of the sinusoidal component plane wave at an angle θ_i with the x–axis and y–axis. θ_i represents the number of orientations, and $\hat{l}$ is the number of scales. σ is the standard deviation of Gaussian function, θ_i is orientations of the scale $\hat{l}$, $\theta_i = \pi(i-1)/\theta_i$, $i = 1, 2, ..., \theta_{\hat{l}}$.

The GIST feature extraction steps are illustrated as follows:

1. Divide an image $I(x,y)$ with $h \times w$ size into $g \times g$ grids of the same size, so that $\acute{h} \times \acute{w}$ is the size of each grid, in which $\acute{h} = \frac{h}{g}$ and $\acute{w} = \frac{w}{g}$.
2. Each grid is convolved by g_c-channel filters using Gabor filter, and the results are cascaded to form grid features.

3. The GIST features of the grids are obtained by averaging the eigenvalue computed from each grid.
4. Cascade the g_c average eigenvalues computed from each grid to get the whole GIST features of the image.

In this thesis implementation, the GIST descriptor is computed from depth videos, and namely 3DGIST because it is applied to the spatio-temporal domain of the video sequence (see Chapter 5, Section 5.2.3). 3DGIST features are computed to perform human action recognition from input depth images. This descriptor captures the structural action information and also describes the location relationship of local grids.

Gray Level Co-occurrence Matrix (GLCM) GLCM was presented by Haralick et al. [HSD73] to be used to describe textural features analysis. It measures the second-order statistics associated with image attributes by taking into account the spatial relationship between pixels. GLCM describes how different gray levels combinations occur in the images. GLCM is one of the most popular techniques using a statistical approach in order to obtain a global feature. It consists of a square matrix, which assigns values to describe the image's distribution of occurrence. The number of rows and columns is equal to the number of grayscales in the image that can detect specific properties through the spatial distribution of grayscales in the image structure. The GLCM values represent the number of occurrences with grayscale value and connect by specific relationships.

The GLCM is evaluated by observing the frequency of the gray level pixel intensity value (p) that appear in the specified spatial pixel relationship with the (q) value. The spatial relationship can be measured in various ways. The default setting is between a pixel and its relative right neighbor. The relationship can be established with different offsets and angles. This means that GLCM values have relative frequencies with two adjacent pixels separated by a distance (d), and gray-level values, i.e., image intensity values, (p) and (q). The mathematical calculation of a GLCM $(\boldsymbol{G})$ can be specified over an $(M \times N)$ of an image (I), which parametrized by an offset $d = (\Delta x, \Delta y)$, as shown in Equation (3.21).

$$\boldsymbol{G}_{\Delta x, \Delta y}(\mathsf{p}, \mathsf{q}) = \sum_{i=1}^{M} \sum_{j=1}^{N} \begin{cases} 1, \text{if } I(i,j) = \mathsf{p} \text{ and} \\ \qquad I = (i + \Delta x, j + \Delta y) = \mathsf{q} \\ 0, \text{ otherwise} \end{cases} \tag{3.21}$$

where (p) and (q) are the pixel with gray level intensity values separated by a d distance, i and j are the spatial locations in the image (I) and the offset $(\Delta x, \Delta y)$, (M) and (N) are the dimensions of the image, and (d) is the co-occurrence matrix's directional offset. The distance value (d) ranges between (1) and image application of large d, i.e., $(d > 2)$, values to a smooth texture does not provide details of textural information.

In the proposed method of this thesis, the GLCM with $d = 1$ is used to avoid loss of information. Therefore, more likely pixel be correlated with a closely located pixel than with one located far away. So, the result will be presented by using $d = 1$ only.

Figure 3.3 shows the GLCM calculation by displaying an image and the associated co-occurrence matrix by using the given pixel spatial relationship (offset = $+1$ in the x-direction). For pairs $(2, 1)$ (pixel 2 followed by pixel 1), it is found twice in the image; then the GLCM image will have (2) as a value in the location corresponding to ($I_\mathsf{p} = 1$) and ($I_\mathsf{q} = 2$). The GLCM is a (256×256) matrix; I_p and I_q are the intensity values for an 8-bit image.

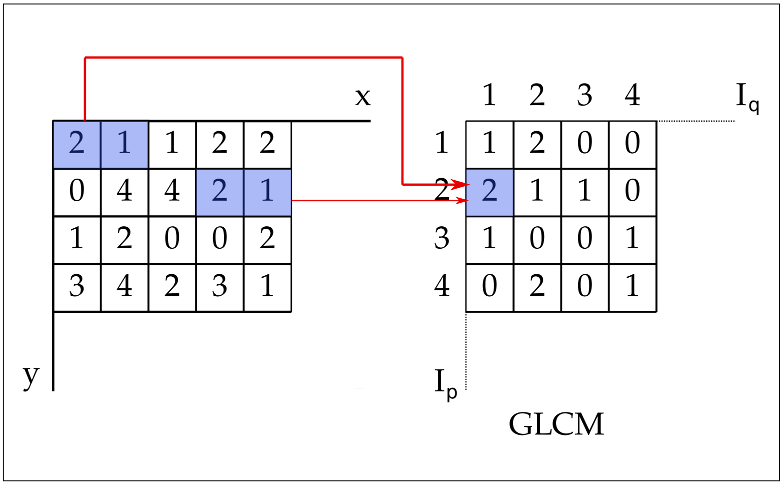


Figure 3.3: Description of the Gray Level Co-occurrence Matrix.

In Chapter 7, the GLCM will be computed from the optical flow fields of input 3D data. Note that in the proposed method, the GLCM matrices will be computed using the binarized magnitudes (M^N) and orientations (Θ) fields extracted from the RGB and depth channels, which will be discussed later in Chapter 7, Section 7.2.2, i.e., the matrices will not be computed from the image intensity values p and q. Then, the output of co-occurrence matrices will be used as the input to the Haralick texture descriptor for extracting feature vectors from videos.

Haralick Texture Features Haralick's texture features [HSD73] is one of the texture descriptor methods used to represent the correlation between the intensity of pixels in space.

The texture is defined as a repeated information pattern or a structure arrangement with regular intervals. Texture features generally refer to the surface properties and appearance of an object provided by the shape, size, arrangement, density, and proportion of its elemental components. An essential stage in collecting such features through texture analysis is the extraction of texture features. The texture is one of the most important characteristics that define an image.

A Haralick introduced fourteen measurements of texture features extracted from the co-occurrence matrix, which is known as a statistical technique for texture features. In this thesis implementation, only “six” Haralick features will be used in order to reduce the computational complexity of the proposed system. The basis for computing Haralick features is the Gray-Level Co-occurrence Matrix, i.e., $\boldsymbol{G}$ in Equation (3.21). These feature vector values are computed from the following Equations (see Equations (3.22) to (3.31)), where $\boldsymbol{G}(i,j)$ is the (i,j)-th entry of the normalized GLCM.

- Energy: Also known as Angular Second Moment (ASM), which calculates the homogeneity of an image. When pixels are very close, energy is high.

$$HF_1 = \sqrt{\sum_i \sum_j \boldsymbol{G}_r(i,j)^2} \tag{3.22}$$

- Contrast: Is a measure of the intensity or gray-level variations between the reference pixel and its neighboring. The visual perception describes an appearance difference between two or more parts of a field that are seen simultaneously or consecutively.

$$HF_2 = \sum_i \sum_j (i-j)^2 \boldsymbol{G}_r(i,j) \tag{3.23}$$

- Homogeneity: Calculates how similar the GLCM element distribution is to the GLCM diagonal.

$$HF_3 = \frac{\sum_i \sum_j \boldsymbol{G}_r(i-j)}{1+(i-j)^2} \tag{3.24}$$

- Entropy: Shows an amount of image information required for image compression. The high entropy image has a great contrast of one pixel to its neighbor and is not compressed with low contrast as a low-entropy image (many pixels have the same or similar value).

$$HF_4 = -\sum_i \sum_j \boldsymbol{G}_(i,j)[\ln \boldsymbol{G}_r(i,j)] \tag{3.25}$$

- Sum Average [2]:

$$HF_5 = \sum_{i=0}^{2(N_g-1)} i\boldsymbol{G}_{x+y}(i), \tag{3.26}$$

where N_g equal to 256 level, x and y represent the row and column coordinates of an entry in the co-occurrence matrix and $\boldsymbol{G}_{x+y}(i)$ is the probability of the co-occurrence matrix coordinates summing to $x + y$.

- Correlation: Computes a linear dependency of the gray-level values in the co-occurrence matrix. It shows how a reference pixel is connected to its neighbor.

$$HF_6 = \frac{\sum_i \sum_i (i,j)\boldsymbol{G}i,j - \mu_x\mu_y}{\sigma_x\sigma_y} \tag{3.27}$$

where $\mu_x; \mu_y$ are the means, and $\sigma_x; \sigma_y$ are the standard deviations, which are expressed as:

$$\mu_x = \sum_i \sum_j i\boldsymbol{G}(i,j) \tag{3.28}$$

$$\mu_y = \sum_i \sum_j j\boldsymbol{G}(i,j) \tag{3.29}$$

$$\sigma_x = \sqrt{\sum_i \sum_j (i - \mu_x)^2 \boldsymbol{G}(i,j)} \tag{3.30}$$

$$\sigma_y = \sqrt{\sum_i \sum_j (j - \mu_y)^2 \boldsymbol{G}(i,j)} \tag{3.31}$$

This feature extraction method will be applied in Chapter 7 for extracting the feature vectors for improving human action recognition from RGB-D videos.

3.4 Video-Action Representation

This section describes a video-action representation model that will be selected to use in this thesis for representing the extracted features based on their use in the literature and the results produced by several researchers [H́D11, WRLD13, ATK$^+$15, GPF17]. The selected technique is Bag-of-Features (BoFs). The BoFs approach is one of the most popular techniques for encoding local features. In recent years, it has shown impressive performance levels with local spatio-temporal features of videos. In a video sequence, the BoFs construct a histogram of feature occurrences.

3.4.1 Bag-of-Features

The Bag-of-features (BoFs) were widely used for video action recognition, representing images or videos as an orderless collection of features. The BoFs originate from the Bag-of-Words (BoW) representation used in document classification and textual information retrieval [H́D11].

The Bag-of-Words (BoW) is a prevalent representation utilized in natural language processing, information retrieval, and computer vision [Bil14]. It works on extracted local features, so a video is described as a bag of its local features, i.e., BoFs, which ignore their order but keep multiplicity. In a video sequence, the BoW representation encodes global statistics of local features, estimates a histogram of feature occurrences. The BoW model is widely used in action recognition, where the frequency of the local feature is measured and used for training a classifier. The first step of the BoW model is to construct a visual word, known as a dictionary or codebook. The dictionary is generated using local features derived from the training videos. Local features extracted from the testing videos are not used during dictionary construction.

Usually, the k-means algorithm is used for the dictionary generation (Chapter 2, Section 2.5.2). The k-means algorithm is an unsupervised learning algorithm that partitions specific set of N_f feature vectors $(\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_{N_f})$ into k clusters $(k \leq N_f)$, where each vector feature of the cluster is fixed in the cluster with the closest mean. The k-means clustering algorithm is designed to reduce the sum of the squares of the distances between the features and their nearest cluster centers.

A significant element of the k-means clustering algorithm is to choose the proper distance between a feature vector and the center of a cluster. Euclidean distance measurement is chosen to be used in this thesis due to the best results achieved by state-of-the-art. The Euclidean distance formula is represented in Equation (2.6), Chapter 2.

The BoW model can describe every video after generating the visual vocabulary (dictionary). The BoW model is a video sequence that assigns its features to the closest elements of the created visual vocabulary, i.e., the nearest cluster center. Consequently, given a set of local features extracted from a video sequence $(\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_{N_f})$, each feature vector will be assigned to the nearest cluster center using the nearest neighbor algorithm. The same distance will be used between a feature vector and the center of a cluster used to generate the dictionary. Local features of the j-th cluster are shown as:

$$Z_j = \{\boldsymbol{v}_k | j = \underset{i}{\operatorname{argmin}} \|\boldsymbol{v}_k - \boldsymbol{\mu}_i\|^2\}, \tag{3.32}$$

where $\boldsymbol{\mu}_i$ is the center of the i-th cluster Z_i.

The BoW model describes a video sequence as a histogram (H) of occurrences of local feature, and more specifically, as a k-elements feature vector $H = \{H_1, H_2, ..., H_k\}$ of quantized local features, where k is the number of dictionary elements. Each H

element of the feature vector defines the number of features allocated to the respective cluster $(H_i = |Z_j|)$. Typically, the BoW representation is used as an input to the classification methods, later in thesis work, for improving the accuracy performance of human action recognition.

The motivation for applying the BoW to the action recognition approach is to deal with the local descriptors for different videos. These kinds of descriptors contain the variable number of produced interest points. Certainly, the global feature does not have an interesting point related problem, but the problem is that it is sensitive to varying duration time of action. The BoW adds the statistical temporal details of a video occurrence and can therefore deal with long-term or multiple-cycle action videos. Given a video sequence $\mathbf{V} = F_i \ (1 \leqslant i \leqslant N_F)$, where F is a single frame (or its MHI), global features are determined on each frame or its variant frame, for instance, MHI (see Section 3.3.2), giving a sequence of feature vectors $F_v = G_{Fi} \ (1 \leqslant i \leqslant N_F)$, where G_{Fi} is the global feature, e.g., GIST, Hu-moment, or Haralick features, which forms the input to the BoW.

3.5 Video-Action Recognition

After representing features by computing BoW histograms, the next step in human action recognition is building a classification model to classify video actions and improve the performance of an action recognition system. Various classification approaches, according to classical machine learning aspects, are used. This section focuses on the existing classification algorithms that are used in this thesis experiments for improving the system approach in human action recognition tasks:

3.5.1 Support Vector Machine

A Support Vector Machine (SVM) is a supervised classifier [Vap95, BB98]. It is a good classifier for analyzing and classifying data because it is easy to learn and has a fast learning speed, even in large datasets. In binary classification, SVM is typically used to classify data with a similar feature value. It separates feature vectors using a hyperplane. It can minimize the overfitting problem that occurs in the training data. So, SVM uses hyperplanes in high dimensional space to split the training data with the most significant points' margin. It is a powerful and frequently used in the classification of extracted features [UDSS15]. The reason for finding the maximum margin hyperplanes is to offer the best generalization ability that allows the best classification accuracy. In other words, it has a good performance on the training data leaving many areas for the correct classification of feature data [WKRQ$^+$08].

An SVM takes a collection of training data and marks it as part of a group, then predicts whether the test data is a part of an existing class. The model of SVM represents the data as a point in space separated by a hyperplane [PS15]. The optimal hyperplane search function is computed in Equations (3.33) and (3.34) [MRS08]:

$$\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_i \xi_i \tag{3.33}$$

$$\{(\boldsymbol{v_t}_i, \lambda_i)\}, \lambda_i(\boldsymbol{w}^T\boldsymbol{v_t}_i + \mathsf{b}) \geq 1 - \xi_i \tag{3.34}$$

where, $\boldsymbol{w}$ is the weight vector. The C parameter represents a regularization term that provides a way to control over-fitting. When the value of C is significant, it is unattractive that data should not be observed at the expense of decreasing geometric margins, but it is simple to consider certain data points using slack variables and use a bold margin to model most of the data. $C\sum i\xi_i$ is a loss function, where ξ_i is a slack variable/misclassification vector $\boldsymbol{v_t}$ and $\xi \geq 0$. $\boldsymbol{v_t}_i$ is the train vector of i, and λ_i is a class train vector (class label) of i. b defines the bias value (represent an intercept term). The scoring function that used to find the test class in SVM is defined as in Equation (3.35):

$$f(\boldsymbol{v_s}) = \text{sign}(\boldsymbol{w}^T\boldsymbol{v_s} + \mathsf{b}) \tag{3.35}$$

where $f(\boldsymbol{v_s})$ presents the score function, and $\boldsymbol{v_s}$ is a test vector.

In this thesis, the SVM with Radial Basis Function (RBF) kernel is used for classification [CL11]. The kernel function is a measure of the similarity between two groups (or sets) of features. RBF is a Gaussian distribution and maps data into an infinite-dimensional Hilbert space. The RBF is calculated as in Equation (3.36) [MSM15]:

$$RBF(\boldsymbol{v_t}, \boldsymbol{v_s}) = \exp(\frac{\|\boldsymbol{v_t} - \boldsymbol{v_s}\|_2^2}{2\gamma^2}) \tag{3.36}$$

where, $RBF(\boldsymbol{v_t}, \boldsymbol{v_s})$ is kernel function. $\| \quad \|_2^2$ is the squared Euclidean distance between the two feature vectors $\boldsymbol{v_t}$ and $\boldsymbol{v_s}$.

Multi-class SVM SVM is a binary classifier. The binary classifier aims to determine whether the input data, expressed by real vectors, belongs to one of two classes. The multi-class SVM is an extension of the two-class SVM [CS11], which can allocate labels to data where the number of labels (classes or, in this thesis, daily activities) is more than two. The most popular way to transact with this situation is to split the multi-class problem into several binary class problems.

Two common techniques for multi-class classification are used: One-vs-One (OvO) and One-vs-All (OvA), as explained previously in Chapter 2, Section 2.5.1. Let the number of classes to recognize be denoted as N_{cl}. Formally, the OvA trains a classifier for each possible pair of classes, i.e., $\frac{N_{\text{cl}}(N_{\text{cl}}-1)}{2}$ classifiers. All binary classifiers are evaluated for each new test sample, and the test sample is allocated to the class selected by the majority of classifiers. The last technique, i.e., OvA, trains N_{cl} binary classifiers, each to separate the samples from samples of the remaining classes within in single class. All binary classifiers are tested for each new test sample. The test sample will be allocated to the class where classifier output is the largest value, i.e., most positive.

In this thesis, the OvA multi-class SVM will be applied later in the next chapters for computing the action recognition accuracy performance of the proposed action recognition system rather than just a class label. The OvO strategy is not realistic for a large-scale linear classification because of the large storage space needed for classifier models. While OvA is as reliable as any other method, it is believed that the underlying binary classifier is frequently customized classifications such as SVM.

3.5.2 K-Nearest Neighbors

K-Nearest Neighbor (KNN) classifier is the simplest method used for classification [CH06, MRS08]. KNN obtains a class membership by training feature descriptors for certain testing feature descriptors based on its nearest neighbor. KNN is used to classify unlabeled observations by assigning them to the class number of the most similar label, i.e., the testing feature is classified by a majority vote of its K nearest neighbors. Features of the observations are collected for both the training and testing data sets. K is an integer value and typically small and varied by the amount of the test class. If $K = 1$, the object is directly assigned to the class of its nearest neighbor [SJ10]. While if $K > 1$, the labels of the K closest classes will be checked, and the most common label is assigned.

In the KNN, three parameters should be represented. The first parameter, K, is assigned to the number of voting members. The second parameter is distance metric form, which is measured using Euclidean Distance (will be used in thesis experimentation), cosine metrics, absolute difference, and so on. The third parameter represents the rule of choosing an estimated class for the testing sample, and it is set to the nearest neighbor. The KNN classifier is computed from the distances between the testing video and each training sample, as defined by Equation (3.37) [MAaMV15].

$$dist(\boldsymbol{v_s}, \boldsymbol{v_{tj}}) = \arg\min{}_j\{dist(\boldsymbol{v_s}, \boldsymbol{v_{tj}})\} \tag{3.37}$$

where $dist$ represents the distance metric, and $\boldsymbol{v_s}$ is a testing vector sample, $\boldsymbol{v_t}$ is the training vector sample, where $j = [1, 2, ..., Tr]$, and Tr is the number of training class. The distance $dist$ represents the minimum distance (nearest neighbor) between $\boldsymbol{v_s}$ and each training class. The action class with minimum distance will be identified as the class to which the testing class membership.

Multi-class KNN In a typical KNN setting, each test object is allocated to a certain class dependant on the majority of its K nearest neighbors, as explained previously. Though, this method can be computationally costly for datasets that contain millions of test classes. Some previous researches [MAaMV15, PS15] have shown that KNN classifiers can greatly improve the quality of classification trained with the use of pre-labeled classes. Since a typical KNN strategy is introduced by [KS15], it can be highly challenging in performance. The multi-class concept is to compute a center for classes of feature vectors of the same class and apply the similarity metric individually on these centroids instead of each feature vector. In this thesis, the multi-class KNN classifier is used to obtain the class membership of the extracted testing feature based on its nearest (closest) neighbor from the extracted training feature.

3.5.3 Random Forest

Random Forest (RF) is an ensemble learning method introduced by Breiman [Bre01]. It is a supervised machine learning technique used for classification and regression tasks. In this thesis work, it will be used for classification. As the name suggests, the RF algorithm creates a forest with numerous decision trees and averages predictions over many individual trees. It was built to work rapidly over large datasets and, more importantly, be varied using random samples to construct each tree in the forest. Variety is gained by randomly picking attributes at each tree node and then using the attribute that gives the highest learning level. RF classifier uses bagging [Bre96] and random subspace method in building each tree to create an uncorrelated forest of trees. The final decision or prediction is based on the majority of votes from each of the decision tree nodes. RF uses bootstrap aggregating or bagging to reduce the risk of over-fitting and the required training time.

A decision tree is a tree consists of a collection of nodes and edges that are arranged in a tree-like structure. Each node in a tree is a feature (attribute), each connection is a decision (rule), and every leaf is an outcome (categorical or continuous value). In internal nodes, splits occur when class labels are stored in the terminal nodes identified as leaves.

Multi-class RF RF is a multi-way classifier, and it performs as follows: A ”forest” is composed of a T randomized decision tree collection. Bootstrapped samples of training data are used to create each tree. In comparison to the traditional classification trees, where the best division for a tree is picked from all predictors, RF trees are grown by selecting the best split predictor from a random set of predictors P_d. Each classification tree’s leaf nodes contain the posterior distribution of the classes [RM05].

An RF classification model consists of several trees. Expanding the number of trees is helped in increasing the classification accuracy up to a certain number of trees.

In RF algorithm, given a training set Tr of a feature vector (training samples) and their corresponding labels, trees are created to optimize a particular function by selecting parameters that divide the data at an internal node. Splits occur on inner nodes, while class labels are saved in the terminal nodes known as leaves. The measures of the degree of a split are called information gain. The information gain (G) is usually used for multi-class classification problems, and it is defined as the difference between the uncertainty of the start node and the weighted impurity of the two sub-nodes (child nodes). Information gain [CSK11] determines which feature is to be used to split the data and can be defined as:

$$G_{\hat{n}} = S(Tr_{\hat{n}}) - \sum_{i \in \hat{L},\hat{R}} \frac{|Tr^{i}_{\hat{n}}|}{|Tr|} S(Tr^{i}_{\hat{n}}) \tag{3.38}$$

where, $G_{\hat{n}}$ is the set of training points at node $\hat{n}$, $Tr^{\hat{L}}_{\hat{n}}$ and $Tr^{\hat{R}}_{\hat{n}}$ are the sets of points at the right and left child respectively of the parent node $\hat{n}$ after the split, and $S(Tr_{\hat{n}})$ represents the Shannon Entropy at node $\hat{n}$ before the split, where the Shannon Entropy is defined mathematically as:

$$S(Tr) = -\sum_{\lambda \in N_{\mathrm{cl}}} \hat{p}(\lambda) \log(\hat{p}(\lambda)), \hat{p}(\lambda) \tag{3.39}$$

where $\hat{p}(\lambda)$ denotes the probability of a sample being class λ.

For classification, the feature vector of the image is extracted and passed through each tree. Trained a new action from an input feature vector is classified by placing it down each of the trees in the forest. Each tree makes a classification decision by voting for that class. The forest selects the classification with the most votes (over all trees in the forest) [MDDB15].

RF is widely used in action recognition tasks due to some preferable characteristics [Bre01]:

- Robust to noise and outliers.
- Efficient and give higher prediction accuracy.
- Efficient with large datasets.
- Ability to accommodate multiple input features without deleting the feature.
- Reduction in over-fitting and RF classifier is more accurate than decision trees in most cases.
- Prediction based on input features important for classification.

The RF classifier will be used in Chapter 5 and Chapter 7 for improving the accuracy performance of the proposed action recognition systems.

3.5.4 Naive Bayes Classifier

Naive Bayes (NB) is a classification method based on the application of Bayes theorem [CC08]. It describes the probability (likelihood) of an event and depends on the priority knowledge of the conditions related to the event. Decisions on the Bayes theorem are related to the probabilities of inference that collect previous events' knowledge by predicting events using the rules base. The NB classifier has independent input variables, which assume that the existence of a particular feature of the class is unrelated to the presence of other features. NB classifier is an accurate, fast, and reliable algorithm with high accuracy and speed on large datasets. The NB algorithm relies on the conditional probabilities of activity perform for each given set of features, which is supposed that all features are statistically distinct from each other. The algorithm starts with counting and labeling all available features and the activity to be recognized, i.e., activity belongs to which classes.

Multi-class NB To demonstrate the concept of activity recognition using the NB classifier, let $\boldsymbol{v}_i$ denotes the feature vector observed at a time. The size of $\boldsymbol{v}_i$ depends on the number of the feature components selected from the extracted features. Given a set of class activities Ω_λ, the posterior probability of a certain activity can be computed using Bayes rule [Mar61] as shown in the Equation (3.40):

$$P(\Omega_\lambda \mid \boldsymbol{v}_i) = \frac{P(\boldsymbol{v}_i \mid \Omega_\lambda) P(\Omega_\lambda)}{P(\boldsymbol{v}_i)}, \tag{3.40}$$

where, $P(\Omega_\lambda \mid \boldsymbol{v}_i)$ is the likelihood of observing a set of feature values for a given activity label. Ω_λ represents the notation values of class λ, $\lambda \in \{1, 2, 3..., N_{\text{cl}}\}$, where N_{cl} specifies the number of classes, and $\boldsymbol{v}_i$ is the feature vector of sample i, $i \in \{1, 2, 3, ..., N_f\}$, where N_f denotes the number of features. $P(\Omega_\lambda)$ is the class value of the prior probability distribution, and $P(\boldsymbol{v}_i)$ is the evidence specific feature vector in the dataset or the probability of observation. Computing the probability term requires estimating the parameter combination. The number of combinations increases exponentially with the feature vector's increasing dimension, rendering this estimate impractical. The NB classifier solves these issues by putting the conditional independence assumption on the features while modeling $P(\Omega_\lambda \mid \boldsymbol{v}_i)$ irrespective of the underlying activity. Significantly, each of the features is considered conditionally independent of the other, provided the activity label. The likelihood can be estimated as a consequence of the probability estimates for each particular feature value in the vector, as formulated in Equation (3.42).

$$P(\boldsymbol{v}_i \mid \Omega_\lambda) = \prod_{i=1}^{N_f} P(\boldsymbol{v}_i \mid \Omega_\lambda) \tag{3.41}$$

The primary goal for an NB classifier is to estimate this conditional probability distribution, $P(\boldsymbol{v}_i \mid \Omega_\lambda)$, for each of the class activities. In the training stage, the NB classifier evaluates the probabilities $P(\Omega)$ from the training set, i.e., for all classes and all features values.

$$P(\Omega_\lambda \mid \boldsymbol{v}_i) \propto P(\Omega_\lambda) \prod_{i=1}^{N_f} P(\boldsymbol{v}_i \mid \Omega_\lambda) \tag{3.42}$$

To make a prediction of activity in the test stage, a test example will be predicted with label Ω if Ω contributes to the greatest value of all the class labels, as defined in the following:

$$\Omega_{nb} = \underset{\Omega_\lambda}{\operatorname{argmax}} \, P(\Omega_\lambda) \prod_{i=1}^{N_f} P(\boldsymbol{v}_i \mid \Omega_\lambda) \tag{3.43}$$

The NB classifier offers a simple approach, with accurate semantics, to represent and learn probabilistic knowledge, as known from machine learning. The approach is intended for use in supervised induction tasks. The performance objective is to predict the class of test example correctly and in which the class knowledge is used in the training samples. The NB will be applied in Chapter 7 for computing the recognition performance from the computed feature vectors.

3.5.5 Artificial Neural Networks

An artificial neural network (ANN) [KL90, Zha00] is a mathematical model or computational model dependent on biological neural networks. It consists of an interconnected artificial neurons group and processes information using a connection computation approach [SW17]. An ANN comprises several simple parallel processing elements whose function is determined by the network structure, connectivity strengths, and processing of elements or nodes. ANN consists of many neurons, which are organized in various layers and interconnected with each other. Each neuron is measured by mathematical computations to collect data, process them, and send information. The mathematical model of an ANN is depicted in Figure 3.4, given an input vectors represented as $\hat{x}_1, ..., \hat{x}_j$ each value x_i is multiplied by the respective weight ω_i and then summed up. Moreover, an additional bias ω_0 is added. The outputs of the ANN $\hat{y}$ is obtained by applying an activation function. The ANN's objective is to calculate a set of weights ω between the input, hidden, and output nodes, which will reduce the total sum of squared errors. During training, these weights ω_i will be modified by a learning parameter until the outputs are compatible with production.

The challenge of using artificial neural networks is to find parameters taught to learn from training without overfitting. If there are too many hidden nodes, the system can overfit the current data, and if there are too few, the system can keep the input values from being correctly fitting. The stop criterion must also be selected. This may involve

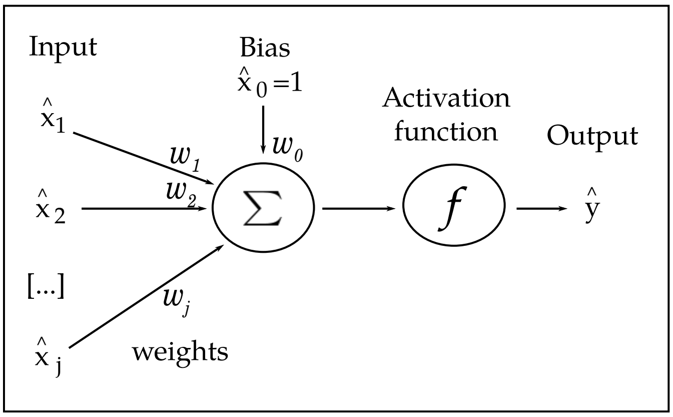


Figure 3.4: A mathematical model of an artificial neural network neuron, compare to [HKM06].

stopping when the cumulative error of the network falls below a predetermined error level or when there are a certain number of epochs (iterations).

In a neural network model, simple nodes (called neurons) are linked to form a network of nodes hence the term neural network. Neural networks consist of nodes or units connected by directed links. Each link has a numerical weight for which the strength and sign of the connection are calculated. Typically, ANN is organized in the form of layers, such as the input layer which includes a set of input nodes, one or more hidden layers, and an output layer. The number of hidden neurons affects the effectiveness of classification.

Neural networks with one or more hidden layers are called Multi-Layer Neural Networks (MLNN) or Multi-Layer Perceptron (MLP). This neural network type is referred to as a supervised learning network since the optimal output is needed. MLP is widely applied in recognition processes [TM17, MWA19], including human activity recognition. MLP will be used later in this thesis and trained with the back-propagation algorithm [RB93, LBOM12] to maximize the relative entropy criterion.

Multi-Layer Perceptron (MLP) is a supervised learning algorithm; it is a feed-forward net that consists of multiple layers with neurons (nodes) that interact using weighted connections. Each layer is fully connected to the next one with a certain weight $\hat{w}_{ij}$ (i.e., each node in a layer is connected to all nodes in the next layer). There is one hidden layer or more between the input and output layers [RRK$^+$90, KL90] [10]. Training is equal to find a proper weight for all connections to produce the desired output for a corresponding input. As the model is trained, the weight of individual neurons is locally modified based on their effect on an arbitrary error function.

All neurons at MLP are similar. Each of them has multiple input connections (it takes the output values from multiple neurons in the previous layer as input) and multiple outputs connections (it passes the response to multiple neurons in the next layer).

The values obtained from the previous layer are summed up by those individual weights for each neuron, plus the bias term. The sum is converted using the activation function $f(\hat{x})$, which can also be different for each neuron.

So, when the input features vector is transferred through the multiple layers downstream, a non-linear transformation is rendered and becomes linearly separable [RB93]. For an arbitrary output $\hat{x}$ of a layer $\hat{n}$, the output for $\hat{n}+1$ layer is the sum of the individual weights of each neuron and a bias function [LBOM12], Mathematically, given the outputs $\hat{x}_j$ of the layer $\hat{n}$, and the outputs $\hat{y}_i$ of the layer $\hat{n}+1$ are determined, as in Equation (3.44):

$$\hat{y}_i = \sum_{j=1}^{N_d} (\omega_{ij}^{\hat{n}+1} \hat{x}_j) + \omega_{ij}^{\hat{n}+1} + f(\hat{u}_j), \quad i = 1, ..., M_l \tag{3.44}$$

where, N_d is a dimensional vector for linear combination of M_l layers, $\omega_{ij}^{\hat{n}+1}$ represents the weight of the individual neuron and $f(\hat{u}_j)$ defines the bias function.

In the training phase of this thesis work, the extracted feature vectors are passed through MLP. This network has four layers, input layer, two hidden layers, and output layer, as in Figure 3.5, where the classifier is based on its characteristics to map the input feature vector into each of the possible action class. The activation function used in this neural network is asymmetrical sigmoid activation function [RB93], as defined in Equation (3.45).

$$f(\hat{x}) = \beta(1 - e^{-\alpha\hat{x}})/(1 + e^{-\alpha\hat{x}}) \tag{3.45}$$

where, $\beta = 1$ and $\alpha = 1$ represents the standard sigmoid and the default choice for MLP.

The back-propagation algorithm is used to train the MLP, where training is equivalent to find appropriate weights for all the connections such that the desired output is generated for a corresponding input.

3.6 Convolutional Neural Networks

A neural network is a system of connected artificial 'neurons' that transfer messages between each other. The connections have numerical weights tuned during the training process, ensuring that a properly trained network respond appropriately when an image or pattern is presented to recognize. The neural network consists of several layers of feature detecting 'neurons'. Every layer has several neurons that respond to various input combinations from the previous layers. A convolutional neural network (CNN) is a special case of the neural network.

The convolutional neural network (CNN) is essentially equivalent to a fully connected multi-layer perception (MLP). Both are networks of neurons with weights and

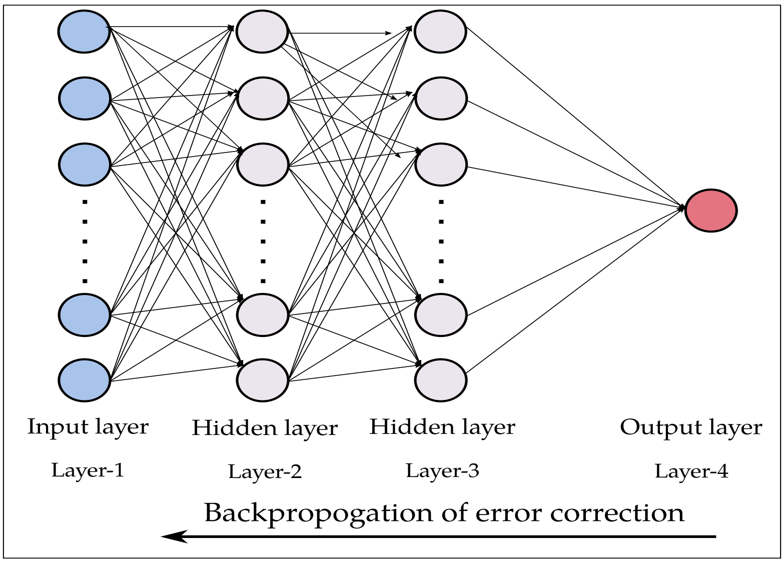


Figure 3.5: Multi-Layer perceptron neural network.

biases, and the core functionality is a dot product between input and neuron weights. Nevertheless, the key distinction from the MLP is that the CNN layer has a 3D network structure. In comparison to a completely connected MLP structure, neurons are connected locally to a small input data region in the grid.

CNN consists of one or more convolutional layers, activation function, and pooling/-subsampling layers. Each layer will usually have a different number of convolutional kernels as a non-linear activation function and may be a pooling mechanism to reduce the dimension of the output data. An example of such a layer is shown in Figure 3.6.

Typically, CNN architecture comprises several types of different layers, including the input layer, hidden layers, and the output layer. Each hidden layer can be either a convolution layer, a pooling/subsampling layer and followed by one or more fully connected layers, as represented in Chapter 2, Section 2.6.

The CNN networks have a wide application area, including robotics, video surveillance, and widely used today in computer vision for image classification [FCNL13, ZZG$^+$15, LZWG18], object tracking [CAS$^+$17], segmentation [CGGS12], object detection [GDDM13], and visual saliency detection [WMC16].

This recent performance of the CNNs highly depends on the improved computing capacity of modern GPUs and the availability of large-scale and complex datasets that make training models possible with millions of trainable parameters. Another important feature of CNNs is the reduction and facilitation of the associations and parameters used in the artificial neural model.

One of the significant disadvantages of deep convolutional neural networks is that they tend to overfit the data. They also suffer from disappearing and crashing gradients.

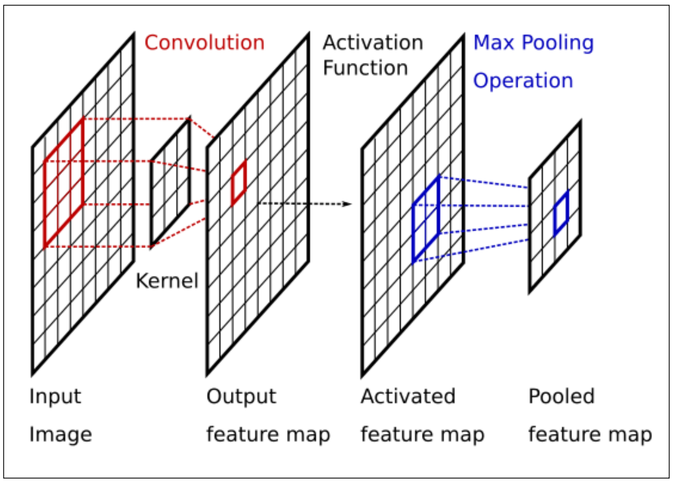


Figure 3.6: Basic of the CNN block. A single layer applies a kernel on an input filter followed by an activation function and a max pooling operation [GLCL19].

The solution to these problems has inspired many studies in different directions. In fact, various CNN components are studied and recommended, for example, activation or normalization layers, training methods, and network design. Most analysis focuses on recognizing images as a proven benchmark since large annotated datasets [RDS$^+$15] are accessible. Various other methods such as video and RGB-D images [SNGW16] have been extended and adapted to appeal to 3D data. In CNN, the convolutional layers are utilized for feature extraction, and the fully connected layer is utilized for classification, as explained in Chapter 2, see Figure 2.13.

In the classical model of pattern recognition, hand-crafted feature extraction from the input data collects specific information and removes unnecessary variabilities. The feature extractor is followed by a trainable classifier, while a standard neural network classified the feature vectors into classes. Convolution layers perform the task of the feature extractor in CNN. Convolution filter kernel weights are updated during the training process. Convolutional layers can extract the local features because they limit the vision fields of the hidden layers to be local. A typical CNN is composed of operations of different layers that are illustrated below:

Convolution Layers The convolution layer applies a set of adaptive filters to the input images or videos. The use of more than one convolution layer allows features to be extracted at different levels. Each convolution layer extracts a higher level of features than the previous layer. As an example, the first convolution layer extracts low-level features such as edges while the next levels extract high-level features such as circles, lines, and other shapes.

In convolution layers, feature maps from previous layers are convolved with learnable kernels. In order to generate the output features maps, the kernels' output is separated into a linear or non-linear activation function, like sigmoid, softmax function, identity functions, and rectified linear [ZA19].

Pooling Layers The pooling layer, also called the subsampling layer, could reduce the resolution of the features. This layer enhances the features against noise and distortion. Pooling can be achieved in two ways: Max-pooling and average pooling. In both states, the input is split into non-overlapping two-dimensional spaces [ZA19].

In this thesis, the max-pooling will be used. Max-pooling [AD16] takes the range $k_r \times k_r$ and outputs a single value, which is the maximum. For an input layer of $N_l \times N_l$, the output of the max-pooling layer is $\frac{N_l}{k_r} \times \frac{N_l}{k_r}$, as shown in Figure 3.7.

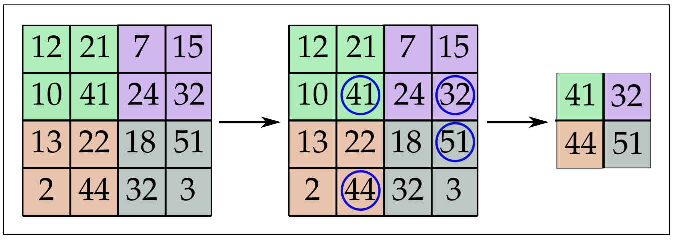


Figure 3.7: Max pooling process. The kernel size is 2×2.

Non-linear Layers In particular, CNN relies on a non-linear 'trigger' function to signal separate recognition of the probable features on each hidden layer. CNN can use a set of specific functions, like rectified linear units (ReLUs) and continuous trigger functions to efficiently enforce this non-linear trigger [ZA19].

A ReLU increases the non-linear characteristics of the decision function and the overall network without impacting the convolution layer's receptive fields. The benefit of using a ReLU is making the network trains faster in comparison to other non-linear functions used in CNN, e.g., absolute of hyperbolic tangent, hyperbolic tangent, and sigmoid.

Fully Connected Layer Fully connected layers on CNN are usually used as the final layers. Mathematically, fully connected layers sum a weighting of the previous layer and display the exact mix of 'ingredients' in order to calculate a particular performance result. In a fully connected layer case, all feature elements from the previous layer are used for calculating each element of the output feature [ZA19].

In this section, the relevant concepts of convolutional neural networks are introduced and followed in the thesis:

3.6.1 2D Convolutional Neural Networks

In 2D-CNNs, the convolutions are applied at the convolution layers to compute features from the local neighborhood (spatial domains) on feature maps in the previous layers. The input data in 2D-CNNs are convolved with 2D kernels. The convolution is done by calculating the sum of the dot product between input data and kernel. The kernel is striding over the input data to cover the full spatial dimension. Then, the additive bias is applied, and the convolved features are passed through the activation function to introduce the nonlinearity in the model [ORK$^+$15].

Formally, the output of the 2D convolution at position (x, y) on the F_m-th feature map in the ℓ-th layer is calculated in the following Equation (3.46) [JXYY13]:

$$\hat{x}^{xy}_{\ell F_m} = \mathsf{b}_{\ell F_m} + \sum_{J} \sum_{e=0}^{K_h-1} \sum_{n=0}^{K_w-1} \omega^{en}_{\ell F_m J} \hat{x}^{(x+e)(y+n)}_{(\ell-1)J} \tag{3.46}$$

where $\mathsf{b}_{\ell F_m}$ is the bias parameter for the F_m-th feature map of the ℓ-th layer. J indexes over the set of feature maps in the previous layer $(\ell - 1)$-th, which is connected to the current F_m-th feature map. K_h and K_w are the height and width of the kernel, respectively. $\omega^{en}_{\ell F_m J}$ is the value at the position (e, n) of the kernel connected to the J-th feature map. Figure 3.8 shows the 2D convolutional neural network representation.

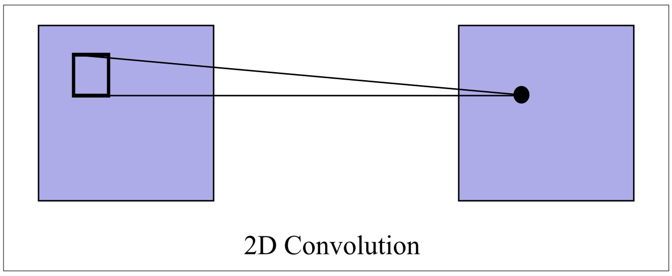


Figure 3.8: 2D convolutional neural network, compare to [JXYY13].

After calculating the convolution, nonlinearity is applied to calculate $\hat{y}_{\ell F_m}$, a Rectified Linear Unit (ReLU) activation function is used as the activation function in all layers for transforming the output codes to the probability values of class labels, ReLU is defined as:

$$\hat{y}_{\ell F_m}(\hat{x}^{xy}_{\ell F_m}) = \max(0, \hat{x}^{xy}_{\ell F_m}), \quad \text{i.e.,} \begin{cases} \hat{x}^{xy}_{\ell F_m} & \text{if} \quad \hat{x}^{xy}_{\ell F_m} \geqslant 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.47}$$

In general, each convolution layer will be followed by a sub-sampling layer (pooling). In the sub-sampling layers, the resolution of the feature maps is reduced by pooling over local neighborhoods on the feature maps of the previous layer, increasing the invariance to distortions on the inputs. CNN architecture can be built by stacking multiple layers of convolution and sub-sampling alternately. The CNN parameters, such as the bias $\mathsf{b}_{\ell F_m}$ and the kernel weight $\omega^{en}_{\ell F_m J}$, are typically trained with either supervised or unsupervised approaches [LBBH98, RHBL07]. Pooling is an important concept in CNNs that is used for down-sampling the input image nonlinearly. A pooling layer can apply max-pooling.

After a set of pairs of convolution and pooling layers, high-level reasoning is performed by one or more fully connected layers. Fully connected layers are an integral part of convolutional Neural Networks (CNNs), which have proven very successful in the recognition and classification of images.

3.6.2 3D Convolutional Neural Networks

In 2D-CNNs, 2D convolutions are used at the convolutional layers for extracting features from a local region on feature maps of the previous layer, i.e., extract features from the spatial dimensions only. When using CNN for video analysis problems, it is advisable to capture the motion information encoded in multiple adjacent frames. In order to do this, 3D-CNN was suggested to be performed in the convolution stages of CNNs for computing features from spatial and temporal domains of an input video, i.e., capturing the appearance and motion information encoded in multiple consecutive frames [JXYY13]. This model creates multiple channels of information from the input video sequences; the final representation of the feature is then gained by integrating information from all input channels. The 3D convolution is accomplished by transforming a 3D kernel into a cube created by stacking several consecutive frames together. This structure links the feature maps in the convolution layer to many consecutive frames in the previous layer and thus collect motion information [ORK$^+$15].

In 3D networks, given an input of form $(w \times h \times N_{Ch} \times N_F)$, in which w, h, N_{Ch}, N_F are width, height, number of channels, and number of frames (video length), when using a 3D convolution filter with depth of l_d where $l_d < N_F$, the outcome will be a 3D volume with 3D temporal features. The implementation of 3D max-pooling and 3D convolutional filters thus preserves temporal information at subsequent layers. Figure 3.9 illustrates the process of 3D convolution used in CNN.

Formally, the values of feature maps in 3D convolutional layer $\hat{x}_{\ell F_m}$ at the position (x, y, z) on the ℓ-th layer and F_m-th feature map are obtained by Equation (3.48) [JXYY13].

$$\hat{x}^{xyz}_{\ell F_m} = \mathsf{b}_{\ell F_m} + \sum_{J} \sum_{e=0}^{K_h-1} \sum_{n=0}^{K_w-1} \sum_{r=0}^{K_s-1} \omega^{enr}_{\ell F_m J} \hat{x}^{(x+e)(y+n)(z+r)}_{(\ell-1)J} \tag{3.48}$$

where, K_s is a temporal dimension 3D kernel size. $\omega^{enr}_{\ell F_m J}$ is the (e, n, r)-th kernel connected value to the F_m-th feature map on the previous layer.

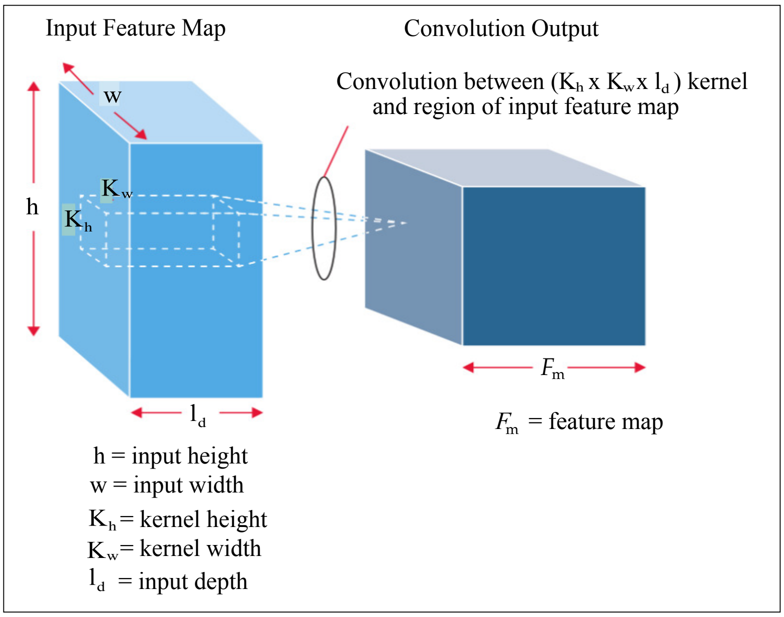


Figure 3.9: Representation of 3D convolution layer process, compare to [ORK+15].

Figure 3.10 shows the 3D convolutions where the convolution kernel size is 3 in the temporal dimension, and the connections sets are color-coded, which means that the same color refers to shared weights. In 3D-CNN, the same 3D kernel is used to overlap 3D cubes to extract motion features of the input video.

A 3D convolution kernel can only extract one form of feature from the frame cube and then reproduce the kernel weights in the whole cube. The general construction principle of CNN is to increase the number of feature maps in late layers by creating multiple types of features from the same set of lower-level maps. As in 2D convolution, several 3D convolutions with different kernels can be accomplished at the same position in the previous layer. A comparison between 2D and 3D convolutions are shown in Figure 3.11.

The 3D CNN architecture is similar to the 2D version, but as mentioned, the convolution kernels will be extended. The input of the model should also be updated to a successive frame stack.

Such as other types of neural networks, CNN training involves two phases: Feed-forward and updating. The feed-forward refers to the process of applying the training samples and receives the results. This process involves applying convolutions to the input image, applying an activation function to these convolutions, sub-sampling the results, and passing them on to subsequent layers. The update process refers to the changing of the weights to reduce error in the output. In this phase, the loss error on the output layer is calculated using a loss function, such as softmax function [WZC14]. Based on the errors, the weights of the network are changed according to a learning

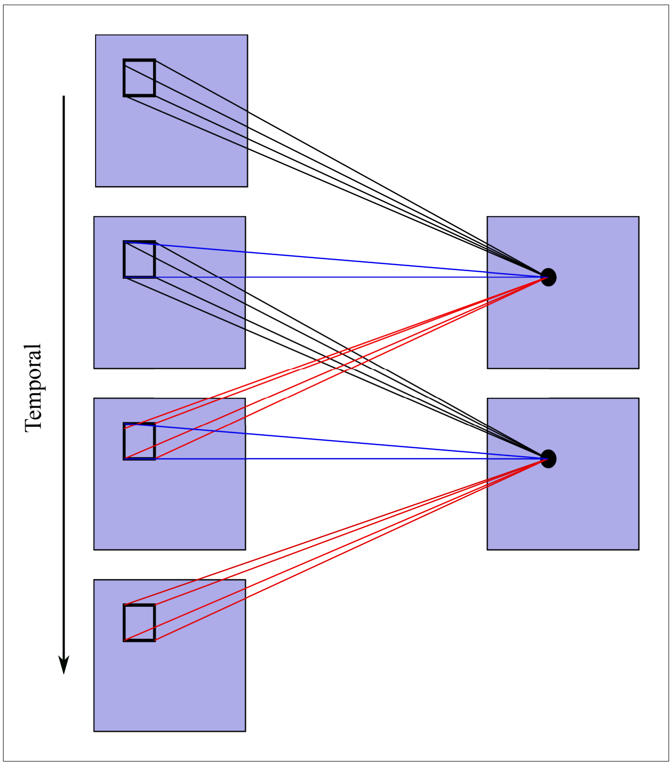


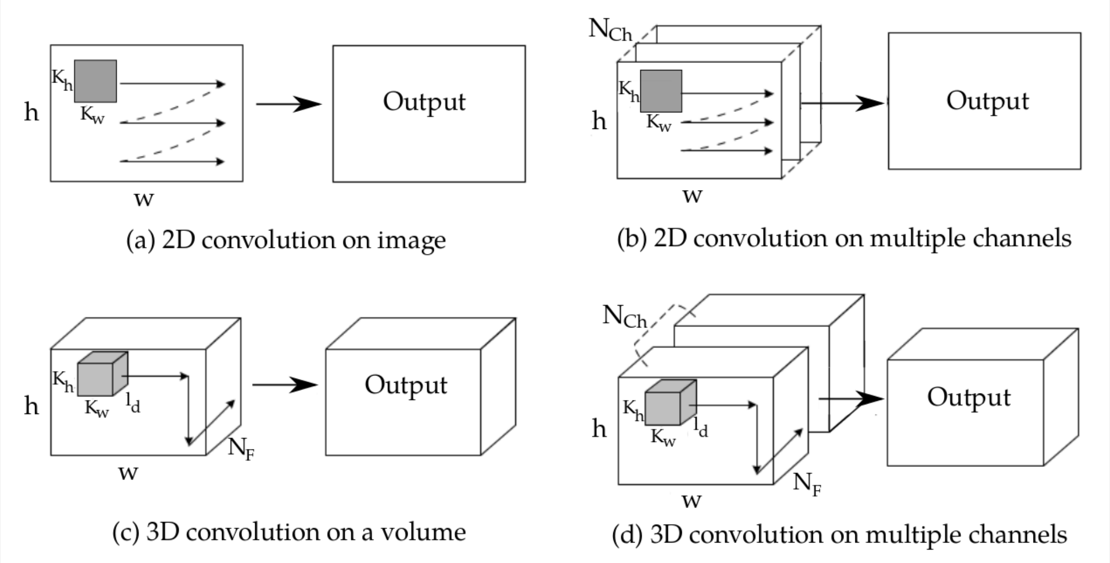
Figure 3.10: 3D convolutional neural network, compare to [JXYY13].

algorithm. The softmax function (ς) takes an N_d-dimensional of real numbers vector and transforms it into a vector of real number within $(0,1)$ range, and the sum of all output node values equals to 1, softmax function $\varsigma : \mathbb{R}^{N_d} \rightarrow \mathbb{R}^{N_d}$ is calculated by Equation (3.49) [KVJ11]:

$$\varsigma(\boldsymbol{v}_i) = \frac{\exp\left(\boldsymbol{v}_i\right)}{\sum_{j=1}^{N_d} \exp\left(\boldsymbol{v}_j\right)}, \quad \text{for} \quad i = 1, ..., N_d \quad \text{and} \\ \boldsymbol{v} = (\boldsymbol{v}_1, ..., \boldsymbol{v}_{\mathrm{N_d}}) \in \mathbb{R}^{N_d} \tag{3.49}$$

where the standard exponential function is implemented to each element $\boldsymbol{v}_i$ of the input vector $\boldsymbol{v}$, and these values are normalized by dividing by the sum of all these exponential to ensures that the summation of the components of the output vector $\varsigma(\boldsymbol{v}_i)$ is equal to 1.

The softmax loss functions can be obtained in order to train a CNN with the desired representation; this is one of the contributions of Chapter 8 in this thesis. Traditionally, CNN has been heavily used in the domain of multi-class classification. The task here is to predict one out of λ classes for a given action. Usually, this is achieved by computing the softmax.



h
K_h
K_w
w
Output
(a) 2D convolution on image
N_{Ch}
h
K_h
K_w
w
Output
(b) 2D convolution on multiple channels
h
K_h
K_w
l_d
N_F
w
Output
(c) 3D convolution on a volume
N_{Ch}
h
K_h
K_w
l_d
N_F
w
Output
(d) 3D convolution on multiple channels

Figure 3.11: Operations of 2D and 3D convolutions. (a) Applying 2D convolution on an image output an image, (b) Applying 2D convolution on multiple images as multiple channels also outputs an image. (c) Applying 3D convolution on a video volume, or (d) 3D convolution on multiple channels of video volume output another volume, compare to [LCX$^+$19].

For checking the recognition rate of the 3D-CNN, the CallBack function [15] will be used on testing data, training time, and progress for each epoch (one pass over the full training set) or mini-batch (only take a subset of all data during one iteration) while training. In this thesis work, the CallBack is used to validate the accuracy of the training data (for more details, see Appendix B).

The other contribution of using 3D-CNN in Chapter 8 is to extract features from the input video. The convolution and pooling layers perform feature extraction that are finally classified by using the classical machine learning techniques, e.g., SVM in this thesis will be used. This 3D-CNN will be implemented in this thesis for improving human action recognition from RGB-D video sequences.

Chapter 4

Multi-Feature Extraction for Human Action Recognition

This chapter presents a novel system to analyze human body motions from RGB-D videos to improve the action recognition task. This system is based on using two feature representation methods, including: Local spatial-temporal features and global features.

The work of this chapter includes two new methods to improve human action recognition. The first method was presented at the *10th International Conference on Machine Vision (ICMV 2017), Vienna, Austria*. Moreover, because it is a novel work, the reviewer recommended this work to be published in the *International Journal of Machine Learning and Computing (IJMLC)* [AAP18c]. The second method of this work was published at the *International Journal of Advanced Computer Science and Applications (IJACSA)* [AaP17].

The work flow of this chapter starts with an introduction in Section 4.1. Section 4.2 presents a proposed method and explains in detail the system analysis of action recognition. The results of this evaluation is presented in Section 4.3.

4.1 Introduction

Human action recognition using cameras is very active research, and it has been widely used in pattern recognition and computer vision studies to characterize people's behavior. The ability to build a system that can intelligently communicate with a human environment is essential for recognizing human actions from various video frames with different actions. In the last decade, the research on human activity recognition concentrated on recognizing human activities from videos captured by conventional visible light cameras (RGB camera) [YT14]. Recently, the action recognition studies have entered a new phase by technological advances and the emergence of low-cost depth sensor like Microsoft Kinect (see Chapter 2, Section 2.2).

This chapter categorizes the body motions on RGB-D videos instead of using only RGB video. Furthermore, it could answer the question:
How to represent the spatial and temporal structure of actions from color and depth data together?
The system of this work is invariant to scale, rotation, translation, and illumination. All experiments are conducted on datasets that are available to the public and often used in the community.

To address the action recognition problem in the proposed work, Bag-of-Features (BoFs) pipeline will be used, focusing on the feature extraction step for performance improvement. The proposed method includes a new combination of RGB and depth data and the combination of local and global features. This new combination provides sufficient complementary information. Using this new feature combination method improves performance on actions with low and high movements and reaches recognition rates superior to other publications that used similar datasets.

Experiments on these two combination methods will be performed on standard action datasets such as MSR3D daily activity dataset and Online RGBD action dataset, which showed that the proposed method outperforms the state-of-the-art features for action recognition. More specifically, the main contributions of this chapter are illustrated below:

- The input data are represented by RGB-D video actions.

- Each video frame consists of many interest points making their descriptions expensive to compute. However, not all the interest points are equally important.

- The important interest points will be filtered to estimate the importance of motion interest points and only keep those interest points in the motion area for action recognition.

- Proposing a novel combination of the local spatio-temporal from both RGB and depth channels into one feature vector of each video action. This combination demonstrates its usefulness for human action recognition.

- Computing global features for extracting human action features from RGB-D videos.

- Developing a novel combination of local and global features descriptors that outperform existing descriptors in action recognition with challenging real-world videos.

4.2 The Proposed System Architecture

This section presents the proposed system methods for improving human action recognition from multi-features combinations. Multi-features methods will be extracted based on the feature representation methods explained previously in Chapters 2 and 3. These methods are: Firstly, local spatio-temporal features are represented by applying the detector and descriptor methods. Secondly, global features are represented by shape features.

As demonstrated in Figure 4.1, system architecture includes three main components: (1) pre-processing to the input video data, (2) Bag-of-Feature extraction, (3) action classification.

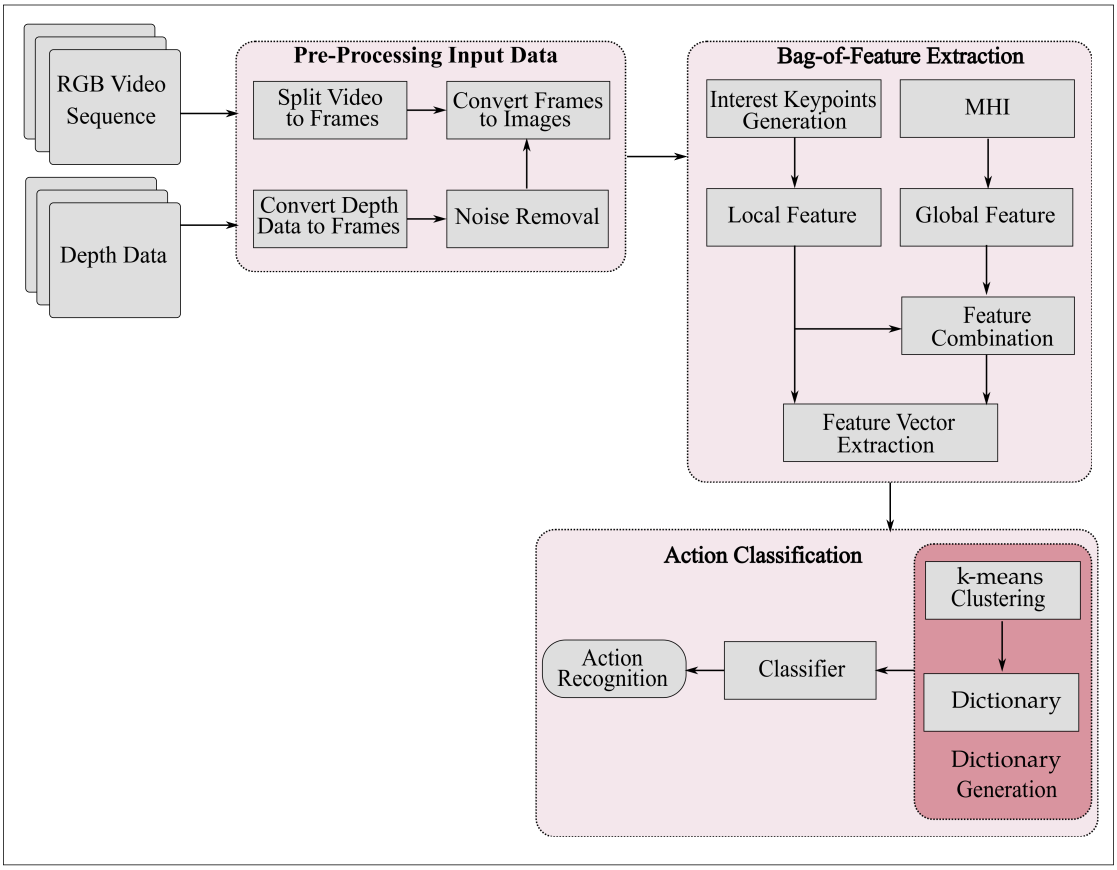


Figure 4.1: System analysis schematics of action recognition using RGB and depth data. Pre-processing to the input data; feature extraction; and classification.

Regarding a Bag-of-Feature extraction, local spatio-temporal features represented by local appearance and motion features will be computed. For local appearance and motion features, the method by [YTYC12] is improved to categorize the body motions on RGB-D videos instead of using only RGB video. So, this work is represented the spatial and temporal structure of actions from color and depth data, as well as combines motion features extracted from both channels in one feature vector for each video action as local features.

On the other hand, the Hu-moments shape invariant (see Chapter 3, Section 3.3.2) is used for global spatial-temporal features that are finally combined with local features to form the multi-features from RGB-D videos.

Then, the Bag-of-Words (BoW) pipeline (see Chapter 3, Section 3.4.1) is used to represent the computed features from input videos. At last, the BoW will be fed to the classifier for computing the system performance. Each step in Figure 4.1 will be illustrated in the following:

4.2.1 Pre-Processing Input Data

The input videos (color and depth) are analyzed as a frame sequence to extract features presented in each frame. In this work, a low resolution of (320×240) is used to reduce the system's computational complexity. The depth information captured by the Kinect camera is often noisy due to imperfections related to the Kinect infrared light reflections. To reduce depth noise and to eliminate the unmatched edges from the depth images, the joint-bilateral filter [CS12] is used to smooth depth images. Moreover, it is used to obtain consistent depth values in neighbor pixels and to reduce the errors at the object boundaries of the depth map. The joint-bilateral filter implementation of this work is based on the OpenCV library [4]. Formally, the joint-bilateral filter could be represented in Equation (4.1):

$$D_{(x)} = \frac{1}{\nu^{(x)}} \sum_{y \in \zeta_x} D_{(y)} f(x,y) G_g(\| D_m(x) - D_m(y) \|) G_h(\| I_D(x) - I_D(y) \|), \quad (4.1)$$

where $D_{(x)}$ is the depth value of pixel at the position x, $\nu_{(x)}$ is a normalization factor, ζ_x represents the set of the spatial neighborhood of position x, $f(x,y)$ is a 2D smoothing kernel also known as domain term for measuring the closeness of the pixels, x and y. While $G_g(\| D_m(x) - D_m(y) \|)$ denotes a depth range term that computes the pixel similarity of modeled depth map (D_m). $G_h(\| I_D(x) - I_D(y) \|)$ represents an intensity term to measure the intensity similarity in the intensity domain. $I_D(x)$ and $I_D(y)$ are the depth images intensity of x^{th} and y^{th} pixels, respectively. The functions f, G_g, and G_h are modeled as Gaussian. In this work, filter size $= 15 * 15$ is used. Additionally, for computing Gaussian function, σ for $f(x,y) = 3.5$, $G_g() = 15$ (pixel values between [0 - 255]), and $G_h() = 15$ are chosen.

4.2.2 Bag-of-Features Extraction

For feature extraction, the Bag-of-Features (BoFs) method is utilized because it is the most popular feature representation technique for videos to learn and recognize the different human actions. In this work, the Bag-of-Features consist of two feature representations: The local features have been computed from the spatial-temporal domain by implementing the feature detector and descriptor methods on 3D data. The procedure for extracting feature vectors includes four steps: Interest keypoints detection, filter motion keypoints, align interest keypoints to depth, and finally, feature vector extraction. The global features have been computed from 3D motion data using a moment invariant feature descriptor method.

Local Feature Extraction

A local spatio-temporal feature usually includes two stages: Detection and description. In the detection stage, a feature detector localizes interest points in a spatio-temporal space. While in the description stage, a feature descriptor computes representations of detected points. The local feature extraction steps shown in Figure 4.2 will be explained in the following:

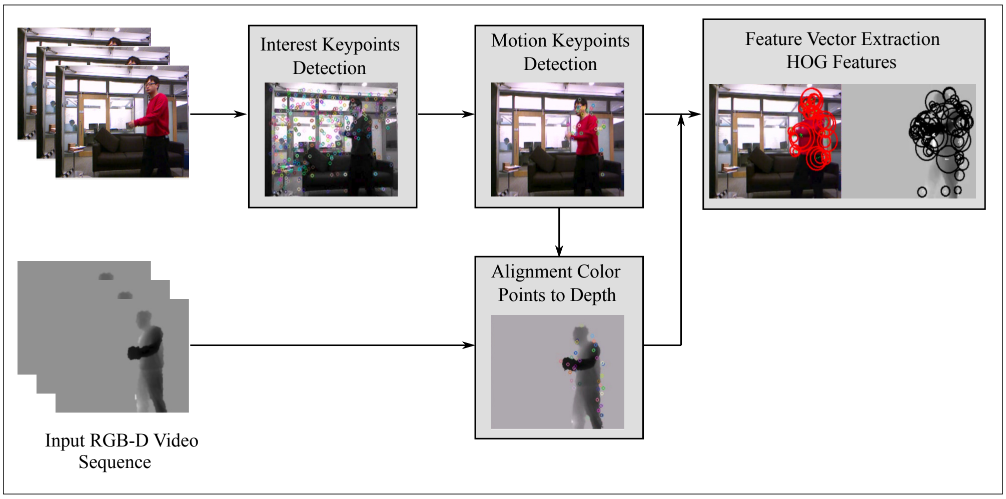

Interest Keypoints Detection
Motion Keypoints Detection
Feature Vector Extraction HOG Features
Alignment Color Points to Depth
Input RGB-D Video Sequence

Figure 4.2: Local feature extraction.

1. **Interest keypoints detection using SURF** To compute motion interest keypoints from RGB frame sequence, Speed-Up Robust Features (SURF) detector (see Chapter 3, Section 3.3.1) is applied as a first step to extract visually distinctive keypoints from the spatial domain. SURF detector is employed to localize interest points spatially, and it can generate additional scale information and maintain

computational efficiency. In this work, to provide essential evidence for action recognition, the dominant orientations of interest points are discarded as motion directions. Then, these SURF points are filtered to compute motion interest keypoints as will be declared in the next step.

2. **Filter motion keypoints using MHI-OF** After computing the SURF points, those points are filtered by the temporal (motion) of the Motion History Image (MHI) generated by the differences between adjacent frames (see Chapter 3, Section 3.3.2). The points with greater intensities in MHI reflect moving objects with more recent motion. The brighter pixels on MHI lead to the most recent motion. The gradients of MHI also represent the directional details of human action. Then, MHI has been utilized as a motion mask to eliminate interest points from the static background, i.e., as points of interest are picked only SURF points with the most recent motions or high MHI. The motion interest points from the MHI are then exploited to compute gradient-based optical flow. The motion vectors at each interest motion point from the MHI are calculated using the Lucas-Kanade optical flow detector (see Chapter 3, Section 3.2.2). The motion vector points are finally used to compute spatio-temporal features from moving objects.

   SURF/MHI-OF detects points of interest with spatially distinctive shapes and temporally appropriate motions. These detectors, therefore, provide complementary interest points.

   In this thesis work, to compute the MHI, the number of consecutive frames (N_F) are considered at a time (t) to obtain the MHI. In MHI H_τ (see Equation (3.13) in Chapter 3), the pixel intensity represents a function of the temporal motion history at that interest points, where brighter values correspond to more recent motion. In this work, τ is set to 20. For pixels with more recent motions, the MHI image is scaled to a grayscale image with a maximum intensity value, i.e., 255. Furthermore, the directional motion information can be explicitly calculated from the intensity gradients in the MHI.

3. **Align interest keypoints to depth images** After detecting the motion points $P(x, y, t)$ from RGB, these motion points are then aligned to the related depth images points $P_d(x, y, z, t)$, where (x, y, t) denote the coordinates and time of interest point on RGB images and (x, y, z, t) refer to the 3D coordinate and time of interest point on depth images.

   To do the alignment between the depth points with the corresponding color image points: Firstly, the motion feature points from RGB are matched with the corresponding depth interest points from SURF using the matched feature, then calculate a Homography matrix transforming [3]. The Homography is a transformation (3×3) matrix that maps the points in one image to the other image's corresponding points.

In the proposed method, to compute the motion point in depth dependently on the corresponding color, the alignment is done because the color and the depth camera in the experimented datasets recorded by the Kinect camera are not synchronized, and capture the scenes at slightly different time instances. Then, feature vectors will be computed from RGB and depth descriptors by depending on this detected motion points. Finally, the feature vectors from each channel are combined to form the feature vector from the RGB-D sequence.

4. **Feature vectors extraction using HOG descriptor** Feature extraction based on HOG descriptor (see Chapter 3, Section 3.3.1) is applied to describe the detected interest points in both spatial and temporal domains of RGB and depth channels. The HOG features are computed from the video images (appearance), MHI-OF channels. Then, these features are represented in one feature vector for each video action. Figure 4.3 shows the representation of the local features extraction steps.

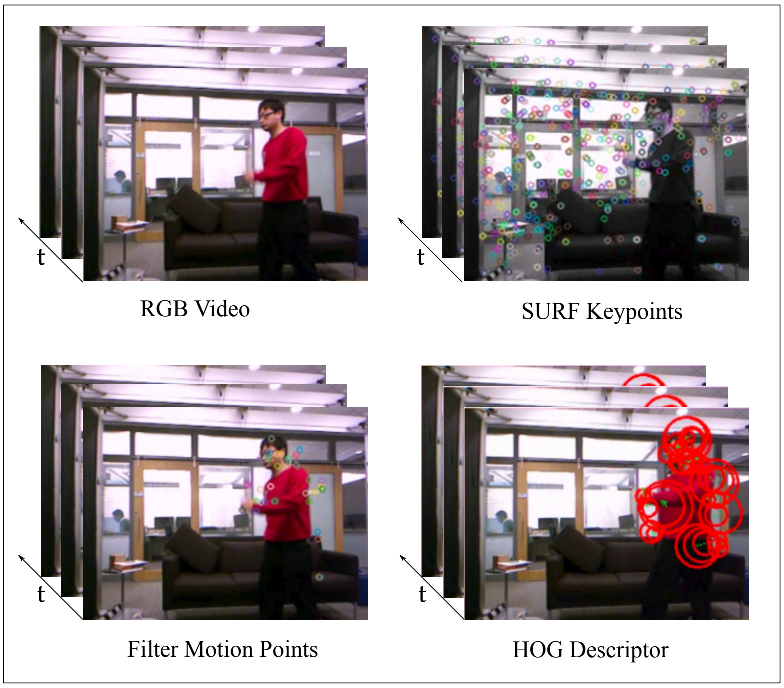


Figure 4.3: Local motion and appearance features representation. The first row shows the original RGB video and the all interested keypoints in the image. The second row shows the filtered motion points from the SURF points to be described by HOG feature descriptor.

In the proposed method, the local appearance and motion features from image channels are characterized by computing distributions of local gradients HOG in the neighborhood with (3×3) grid of patches at each interest keypoint in the intensity image and MHI, respectively. Normalized histograms of all patches are combined to represent feature vectors extracted using descriptors, i.e., HOG (in terms of the appearance features in the intensity image) and HOG-MHI-OF (in terms of the motion features in the MHI-OF). These vectors are then used as the input to the classifier for performing action recognition.

Global Feature Extraction

A global feature is usually represented by shape invariant features, such as the Hu-moments feature descriptor used in this work.

Hu-moments invariant feature descriptor: The global feature based on moment invariant will be computed into two steps:

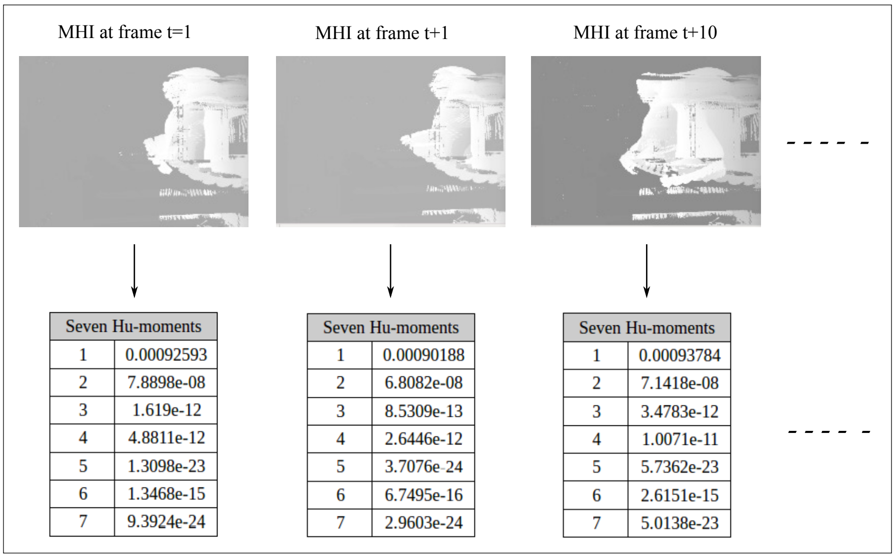
1. Represent the spatial and temporal information about an action in RGB and depth video sequence. In order to do that, MHI is used to express the motion flow or sequence using the temporal density of each pixel, where the pixel intensity is a function of the recency of action.

2. Hu-moments are used as descriptors of the motion history image. Hu-moments introduce seven nonlinear functions that are invariants under the object's translation, scale, and rotation. The set of seven moments are given by Equation (3.14) to Equation (3.18) to get each seven Hu-moments values from the motion history images (for more details about Hu-moments invariant features, see Chapter 3, Section 3.3.2). Figure 4.4 shows an example of the Hu-moments output. The set of seven moments (HM) are treated as seven feature vectors from each channel of RGB and depth. These features are extracted from the MHI of all images in the video. After that, the extracted features are combined into a vector to represent action in the RGB-D video as one feature vector.

Feature Vector Extraction

In order to represent the local appearance and motion information as well as global shape information, two different descriptors are used: HOG features descriptor and Seven Hu-moments shape features.

HOG features descriptor is applied on both RGB and depth video frames and combined feature vector values to generate the bag of local features. For vector generation, the HOG descriptor is implemented around each keypoints in video frames of RGB-D images and motion points computed by MHI-OF. The HOG can also be well adapted to characterize local shape information from the image channel and local motion information from the MHI channel by computing distributions of local gradients.



Seven Hu-moments	
1	0.00092593
2	7.8898e-08
3	1.619e-12
4	4.8811e-12
5	1.3098e-23
6	1.3468e-15
7	9.3924e-24

Seven Hu-moments	
1	0.00090188
2	6.8082e-08
3	8.5309e-13
4	2.6446e-12
5	3.7076e-24
6	6.7495e-16
7	2.9603e-24

Seven Hu-moments	
1	0.00093784
2	7.1418e-08
3	3.4783e-12
4	1.0071e-11
5	5.7362e-23
6	2.6151e-15
7	5.0138e-23

Figure 4.4: Seven Hu-moments of the MHI from video sequence.

Seven Hu-moments shape features are extracted from the MHI of the RGB-D sequence as in Figure 4.4. Then, the geometric moments feature vectors are computed from the combination of the Hu-moments seven values from the motion area in each RGB and depth frames.

Features combination After computing the separated two kinds of feature vectors representing by local and global features. These two feature vectors are finally combined to represent the action information from each RGB-D video. The two feature vectors are combined most simply. A first vector represents the local features. The vectors extracted from it are built by concatenating RGB and depth feature vectors into a higher dimensional vector, whereas each vector has two feature vectors. In other words, the spatial and temporal information computed from HOG and MHI-HOG is integrated into one feature vector. A second vector represents a combination of local and global feature vectors. It is also concatenated easily, i.e., the high-dimensional vector from local features combine with the global feature vectors from different RGB-D video actions to form the high-dimensional vector.

4.2.3 Action Classification

Classification is an essential step in any recognition task. Many classifiers are presented and used according to the type of action recognition task (see Chapter 3, Section 3.5). Firstly, the Bag-of-Words (BoW) vectors for all video sequences are computed in the training stage, and labels are appended according to the class. Then, these BoW vectors are fed into the multi-class classifier in order to train the model that is further used in the testing stage for human action recognition.

Figure 4.5 shows the action classification pipeline of the proposed approach, which starts from the training and testing feature vector sets until the classification step improves the performance of the action recognition task. A dictionary is generated using k-means clustering (see Chapter 2, Section 2.5.2), then the BoW vectors are generated using a cluster model that is finally fed into classifiers for classification. The action classification pipeline explains in the following:

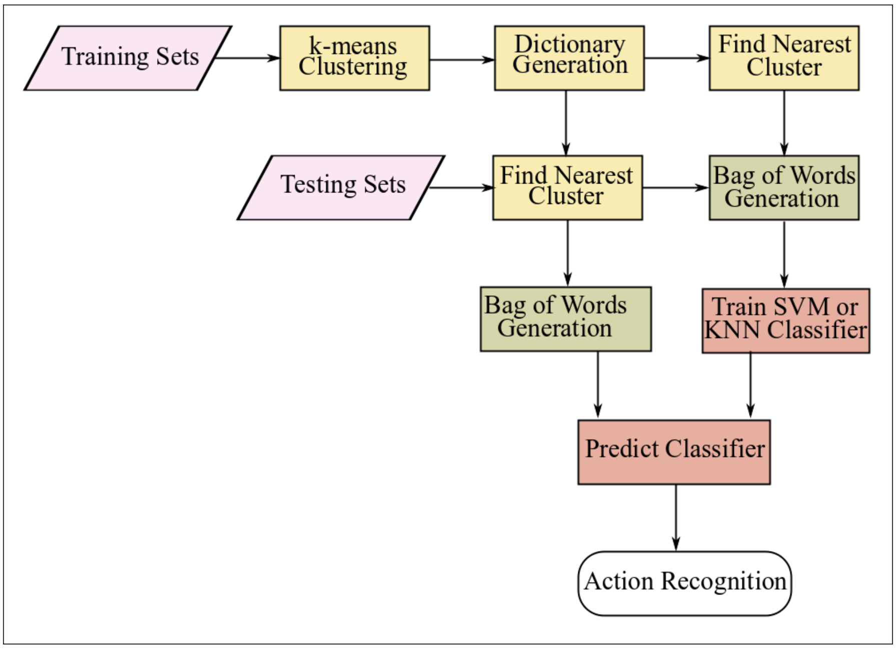


Figure 4.5: Pipeline of the action classification of the proposed approach including: Dictionary generation; Bag-of-Words vectors generation; and finally, classifiers used for classified action.

Dictionary Generation

After extracting features information from all RGB-D videos depending on the detector and descriptor strategy. A dictionary is generated from these feature vectors. The dictionary generation is an important step in the BoFs method, and the size of the dictionary is crucial for the recognition process. If the dictionary size is set too small, then the BoFs model cannot express all the keypoints. While if it is set too high, then it might lead to over-fitting and increasing the system's complexity. The dictionary is created by clustering using the k-means algorithm. The k-means clustering is applied on all BoFs from training videos; the k represents the dictionary size. The centroids of each cluster are combined to make a dictionary. Dictionary is generated to be used for making BoW vector.

Bag-of-Words Vector

In order to generate the Bag-of-Words (BoW) vector, each feature description of the video frame is compared with each centroid of the cluster in the dictionary using the Euclidean Distance Measure (EDM) (see Equation (2.6), Chapter 2). In this generation, each video is treated as a collection of features. Then, these features are quantized into the nearest visual words in terms of the EDM.

Then, check the difference of EDM; if the difference is small or the feature values are close to a particular cluster, the count of that index is increased. Similarly, all feature descriptors of video sequences are compared, and the number of indexes is increased for those whose feature description values come closest to the cluster values. Finally, the BoW vector is generated from a video sequence with a size $(F_v \times k)$, where F_v is the feature vector dimension of each video frame, and k is the dictionary size. The BoW vectors are computed for all videos of the actions.

Action Recognition

In this work, the BoW vectors are computed from all videos in the training and testing stages. These BoW vectors are fed into the multi-class classifier for predicting the computed features. In the proposed method, two different feature vectors are computed, i.e., the spatio-temporal local features and global features.

For the local features, a multi-class SVM classifier (see Chapter 3, Section 3.5.1) is used for predicting the features with RBF kernel. An RBF is maps data into an infinite-dimensional Hilbert space. The RBF is a Gaussian distribution, and it is calculated using Equation (3.36).

For the combined local and global features, a multi-class KNN (see Chapter 3, Section 3.5.2) for the combined features is used.

These two classification methods are applied to train the BoW vectors extracted from the training videos. It is further used in the testing stage for computing the accuracy of the human action recognition system.

Accuracy: Is a fraction of predictions that a classification model got right. In multi-class classification, accuracy is defined as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of classes}} \times 100\% \tag{4.2}$$

4.3 Experimentations and Results

This section discusses the experimental setup, training process and experimental results for the proposed technique. The proposed technique is tested on two well-known RGB-D action datasets: MSR Daily Activity 3D (MSR3D) and Online RGB-D (ORGBD) datasets. The description of these datasets presented in Chapter 2, Section 2.7, and comparative analysis are presented in the subsequent sections.

4.3.1 Dataset Split

In this thesis, the datasets are split based on the hold-out method [YS16]. The hold-out method is the most straightforward form of cross-validation. In this method, the dataset is divided into two independent sets:

- Training set contains examples used for learning and trains the system using the training dataset.
- Test/Validation set (hold-out set) contains examples used to assess the performance of a trained classifier.

The advantage of the hold-out method is usually preferable to a residual method and takes no longer to compute. However, its evaluation can have a high variance. The evaluation can be primarily based on which data points end in the training set and end in the test set. The evaluation can also vary significantly depending on how the division is performed.

In this experiment, the two datasets used for experimentation are divided into a 70% training dataset and 30% test dataset for MSR3D. Furthermore, for the ORGBD dataset, the same environment test setting is used, where half of the subjects are used as training data, and the rest of the subjects are used as testing data.

4.3.2 Parameters Setup for Local and Global Features

The local appearance from image and motion features are characterized by grids of the histogram of orientated gradient (HOG) (with 3×3 patches) around the motion interest points. Normalized histograms of all the patches are associated with the appearance features in the intensity image using HOG, and motion features using HOG-MHI-OF as descriptor vectors, which are finally fed in classifier to represent a human action recognition.

In this implementation, there is an (N_F) in each video action, the motion area or interesting points in each video images are dividing into (3×3) cells. One of the important reason to use a HOG feature descriptor is to describe a patch of an image which provides a compact representation. A (3×3) image patch for (4) image channels contains $(3 \times 3 \times 4 = 36)$ pixel values, and the gradient of this patch contains (2) values for magnitude and orientation per pixel which adds up to $(3 \times 3 \times 2 = 18)$ numbers. These gradient numbers are represented using a 6-bin for HOG and 8-bin histogram for HOG-MHI-OF. These selected values have been used on RGB and depth channels.

Each region with an interesting point on MHI channels is divided into (3×3) grids. MHI gradients are sampled into (8) orientation bins. So each SURF/MHI-HOG point generates a feature vector of $(2 \times 3 \times 3 \times 8 = 144)$ dimensions. For each interest point, HOG (54 dimensions) and HOG-MHI-OF (72 dimensions) descriptors can represent local appearance and motion properties that are finally concatenated into one feature vector for action classification.

In this thesis, dictionaries are constructed with k-means clustering. Two different k values are used as a dictionary size. For computing the local features, the best result with a value of $k = 100$, while combining the local and global features, the best result with a value of $k = 400$ as a dictionary size.

The classification test parameters are set at the following: For SVM classifier with RBF kernel, C parameter is 10, γ is set to 20, and the degree is 3. For the KNN classifier, the K value is set to 16. The One-vs-All approach for multi-class classification is followed in this thesis experimentations.

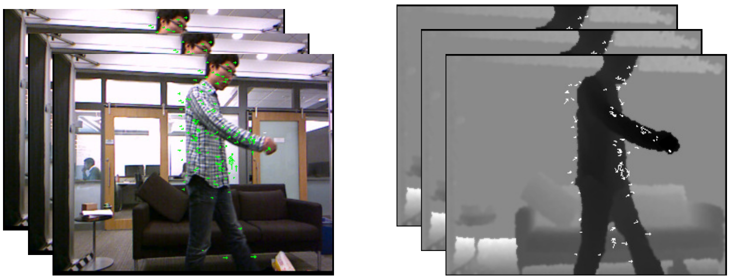
4.3.3 Results and Discussion

The action recognition framework depends on the extraction of two different feature representations, i.e., local and global features, and combining these features.

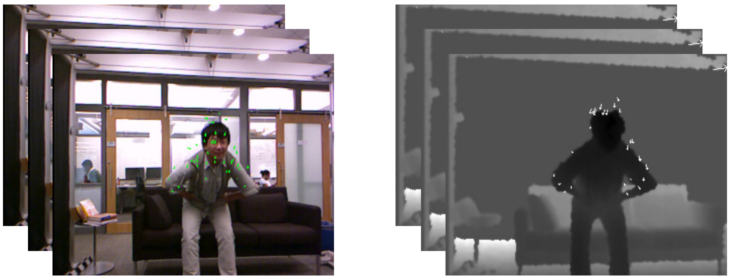
The local features extracted as follows: The SURF detector is used on spatial domains of RGB videos and then filtering these points by MHI and OF on temporal domains to extract the motion points from all video frames. Then these points are aligned to the depth sequence to get the RGB-D interest motion points as in Figure 4.6, which shows the position of interesting motion points in the RGB-D video frames.

The global features extracted as follows: The Hu-moment features are computed from the MHI channel on both RGB and depth video frames to compute the seven invariant features from each video frame. When using MHI, many conditions should be considered, such as the various side, positions, illumination, including person shadow, to guarantee the performance.

The last step in computing the feature vectors is combining the local and global Hu-moment features to represent the feature vector.



Motion Direction of Walking Action



Motion Direction of Sitting Action

Figure 4.6: Motion points in RGB and depth frames of different action represented by green points on RGB frame and white points on depth frames.

Evaluation on Datasets The action recognition system of this chapter is evaluated in two different public datasets, as mention previously. All testing results of the experiment are described using classifiers such as SVM and KNN. The SVM classifier uses local features represented by HOG, and HOG-MHI-OF. For the combined local and global Hu-moments features, the KNN classifier is used to test accuracy results. In both methods, the performance accuracy results outperform other state-of-the-art recognition rates that used different techniques with the same datasets.

Two different combinations are done in this evaluation, the first combination represented by the concatenation of RGB with depth modalities. Table 4.1 shows the local features comparison results between using the RGB only and the combination between RGB and depth data.

The results in the table were illustrated that the combination of two modalities gave a better classification performance than using only RGB. This means that the features obtained from the depth information help to increase the recognition performance in addition to RGB features. Whereas the depth sensor has many advantages over the RGB camera, it can work in total darkness, which makes it possible to explore the fundamental solution for traditional problems in human action classification. Furthermore, it provides 3D structural information as well as color image sequences in real-time.

Datasets	Modalities	Accuracy
MSR3D	RGB	57.65%
	RGB+Depth	**91.11%**
ORGBD	RGB	49.04%
	RGB+Depth	**92.86%**

Table 4.1: Experimental results of the extracted local features from RGB and RGB-D of the MSR3D and ORGBD datasets.

However, the depth camera also has a limitation, which can be partially enhanced by incorporating of RGB and depth. However, all the advantages make it interesting to use the RGB-D cameras in more challenging environments.

In this work, there are two reasons to combine depth information to the corresponding RGB frame: Firstly, the RGB data can be strictly impacted by the variation of lighting conditions, alongside the variety of subject's clothing and viewing angle. Both perspectives can influence the recognition results. In comparison to depth data, this data contains essential information for action representation. Secondly, because of both physical bodies and movements are presented as four dimensions, (x, y, z, t), in the real world. Human activities include not only spatio-temporal axes but also the depth axis, which represents the limitations of 3D scenes and activities that can be directly translated into image/video content. Therefore, this work has to rely on depth maps and color images for human action recognition.

The second feature combination is represented by the concatenation of local features and global Hu-moments descriptor. Table 4.2 shows the results of the combination of two different feature descriptor methods. The local features encode information regarding the RGB and depth modalities in addition to the invariant moment of the global features. The combination of different features shows the best result because the Hu-moments descriptor is invariant to rotation, translation, and scaling. In this work, Hu-moments are computed to represent the video as a global feature from the motion area of the sequence. This feature combination method gave a better accuracy performance than using only local features.

Datasets	Method	Classifier	Accuracy
MSR3D	Local Features	SVM	91.11%
	Local+Hu-moments	KNN	**100%**
ORGBD	Local Features	SVM	92.86%
	Local+Hu-moments	KNN	**96.42%**

Table 4.2: Comparison results on RGB-D datasets of two feature representation methods, local features and combination with global moments features.

Comparison of MSR3D dataset against state-of-the-art The proposed approach results in this chapter are compared with existing approaches on MSR3D dataset, as shown in Table 4.3.

Methods	**Accuracy**
CHAR [ZZSS16]	54.7%
Discriminative Orderlet [YLY15]	60.1%
Feature Covariance [PTDZ16]	65.00%
Moving Pose [ZLS13]	73.80%
S+MDMMs [AFCYN19]	89.14%
Proposed Method	
Local Features-SVM	**91.11%**
Local+Hu-moment-KNN	**100%**

Table 4.3: Comparison of recognition accuracy with other methods using MSR3D dataset.

The state-of-the-art methods in the table that are compared with are contained different feature calculation and classification methods, in addition to the different modalities of input videos. For instance, Zhu et al. [ZZSS16] proposed the continuous human action recognition (CHAR) algorithm based on skeletal data extracted from RGB-D images. Yu et al. [YLY15] used the skeleton and depth modalities as input data. They proposed a discriminative orderlet. An orderlet captures a specific spatial relationship among skeleton joints and defines a comparative relation of the shape information between a subregions group of depth data. Zanfir et al. [ZLS13] proposed a moving pose descriptor framework that used skeleton pose and kinematic information encoded as differential 3D quantities. The S+MDMMs method proposed by [AFCYN19] computed multiple motion information from the depth and transfer learning used to extract spatial information from RGB and depth data.

In this thesis, the proposed system is invariant to scale, rotation, and illumination. Furthermore, the new RGB and depth feature combination method could reach the recognition rates superior to other publications on this dataset. The combined local and global features can also improve performance and reach an excellent recognition rate besides the ability to improve performance on actions with low movement such as drinking, using a laptop.

Comparison of ORGBD dataset against state-of-the-art As the previous dataset, the same feature extraction and classification methods are applied to this dataset. Table 4.4 shows the comparative accuracy performance with the state-of-the-art methods that are used the same dataset in their experimentation and different feature extraction methods.

Methods	**Accuracy**
HOSM [DLCZ16]	49.5%
Orderlet+SVM [YLY15]	68.7%
Orderlet+ boosting [YLY15]	71.4%
Human-Object Interaction [MDDB15]	75.8%
Proposed Method	
Local Features-SVM	**92.86%**
Local+Hu-moment-KNN	**96.42%**

Table 4.4: Comparison of recognition accuracy with other methods using ORGBD dataset.

The state-of-the-art methods in the table are contained different feature extraction methods and the different modalities of input videos. A Hierarchical Self-Organizing Map (HSOM) was proposed by [DLCZ16] for the segmentation, classification, and estimation of ongoing human behaviors that take 3D skeletal joint positions as input obtained from depth data. Yu et al. [YLY15] used depth and skeleton data to present a visual representation, known as orderlet, for improving real-time human action recognition. An orderlet is an intermediate level feature, which captures the ordinary pattern between a set of low-level features. For a skeleton, a spatial relation between a set of joints is captured by an orderlet. For depth data, an orderlet describes a comparative relation of the shape information between a subregions group. Meng et al. [MDDB15] proposed specific features that defined human-object interaction in humans' sequences recorded from the depth sensor. Moreover, a human is defined by the collection of joints located in the skeleton, distances between inter-joints and distances between objects-joints is often used to define relations between human and object, i.e., the human-object interaction.

From the comparison with the other action recognition methods, the proposed methods could improve the new feature combination methods. It gives excellent performance on actions even with low movement and could reach recognition rates superior to other publications of the same RGB-D datasets. The final results show that the proposed method on RGB-D datasets could demonstrate the action recognition approach and significantly outperform the existing state-of-the-art methods. The best performance is achieved in computing local features because interest points are extracted solely from the RGB channel and aligned to the depth. Then combined the RGB and depth based descriptors values depending on these detected motion points. Moreover, the global feature represented by the Hu-moment invariants descriptor is used. This invariant feature descriptor is utilized to solve surveillance camera recording problems, such as different side, position, direction, and illumination, which are commonly utilized in object recognition due to their discriminations strength and robustness.

Chapter 5

Feature Extraction from 3D Trajectory and Global Descriptor

This chapter proposes a novel combination of local and global features approach by using three different feature descriptors to demonstrate the human action recognition task.

The method in this chapter was presented at *the 7th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2018), Funchal, Madeira-Portugal. This work was published by SCITEPRESS – Science and Technology Publications* [AaP18b].

The outline of this chapter starts from an introduction in Section 5.1. Section 5.2 presents a proposed system architecture of action recognition. The action classification method is explained in Section 5.3. The results of the proposed system are presented in Section 5.4.

5.1 Introduction

The approaches presented in Chapter 4 that recognize human action from RGB-D videos can be classified into two main categories: A combination of local features from RGB and depth sequence and a combination of local and global features. The essential steps of the local feature approach represented by extracting interest points from video sequences. Next, the local descriptor was utilized to describe the local properties from RGB and depth sequence. These local feature vector descriptors were then translated into the form of words using the BoW pipeline, and the combined words described human actions from the input RGB-D sequences. Finally, the last features were fed to the classifier to perform action recognition. The global descriptors usually extract the complete human body information, such as MHI. Then, the global feature descriptor was applied to compute the features from the motion regions. The performance may influence partial occlusion and background clutter. Despite encouraging results of RGB-D activity recognition in the previous chapter, activity recognition using RGB-D videos is still a challenging problem due to the camera motion and viewpoint changes, cluttered and dynamic background, illumination changes, etc. Moreover, the system has to cope with a high intraclass variability: Such as the actions performed by people wearing different clothes also having different postures and sizes.

In this chapter, human action recognition will be presented by combining local and global descriptors, i.e., 3D Dense Trajectory (3DTr) and MBH descriptors as local features and 3DGIST descriptor as a global feature. The Dense Trajectories descriptor is one of the most successful action recognition approaches, and it is suitable for situations, including a significant amount of motion. However, many generated trajectories are unrelated to the actual human behavior and can reduce efficiency due to noise and unstable background.

In the 2DTr framework, feature points are densely sampled in each frame and tracked using optical flow. Then, multiple descriptors, including the MBH, HOF, HOG, and Trajectory shape, can be obtained along the trajectories to describe motion, appearance, and shape. In this chapter, the 2DTr is extended to 3DTr by extracting motion information from depth data using the MBH descriptor. On the other hand, to effectively recognize human activities in RGB-D videos, the structural information is extracted from depth sequence using a global GIST feature descriptor. Finally, the new combined approach of local and global descriptors forms a new descriptor, namely 3DTrMBGG, i.e., combined 3DTr, MBH, and global 3DGIST feature descriptors.

Furthermore, to compute feature vectors and accuracy performance, the method of Chapter 4 is employed. This method includes the computation of BoW vectors that will be finally fed to the classifier for computing the performance of the action recognition task. The new descriptor outperforms the state-of-the-art descriptors that are used on RGB-D videos. More specifically, the major contributions of the proposed method are:

- Extract local 3D Trajectories (3DTr) by developing a 2DTrs for activity recognition from RGB-D videos.
- Propose a new approach to combine motion and depth information, i.e., MBH and Trajectory shape descriptors along the depth direction where the newly proposed descriptors characterize activity motion information in RGB-D videos.
- Propose 3DGIST by the extraction of global GIST from depth sequence.
- Combine local and global feature descriptors to form a new descriptor called 3DTrMBGG.
- Outperform state-of-the-art methods performance used the same public RGB-D action datasets that are used for testing in this work.

5.2 Action Recognition System Architecture

This section describes the proposed system architecture of the action recognition approach. Figure 5.1 illustrates the general steps of the action recognition progress of this chapter.

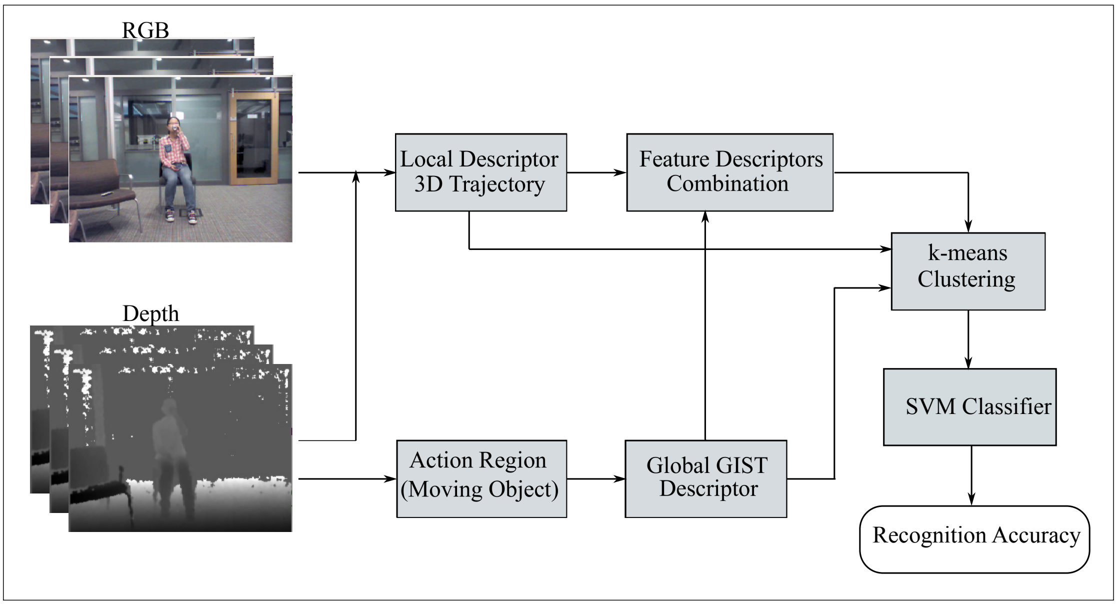

RGB
Depth
Local Descriptor 3D Trajectory
Feature Descriptors Combination
k-means Clustering
SVM Classifier
Action Region (Moving Object)
Global GIST Descriptor
Recognition Accuracy

Figure 5.1: General structure of the proposed system.

At the beginning of this system, the input data are represented by the RGB-D video. Each video is pre-processed by resizing the images and removing noise based on the same strategy used in Chapter 4, Section 4.2.1. These system architecture steps are illustrated in the following:

5.2.1 3D Trajectory

Trajectory is a model for a motion path of a moving object. In real-world, object acts essentially in 3D space. Intuitively, gaining complete 3D motion information leads to a more distinctive and robust activity presentation. Dense Trajectories produce encouraging results on the 2D image plane. However, motion information along the depth direction is missing in the 2D plane. The 2DTr was proposed by [WKSL11] [14] and it was explained in detail in Chapter 3, Section 3.3.1.

In this chapter, the 2DTr descriptor is extended to 3DTr by adding the corresponding depth value to each detected 2D Trajectory point, i.e., mapping the 2D locations of the Dense Trajectories from RGB video frames to the corresponding locations in the depth video frames. That can restore the 3DTr of the tracked interest points that capture important motion information along the depth direction. In 3DTr, the essential step is the estimation of the motion in the depth directions.

Depth Motion Estimation

At the beginning of depth motion estimation, the dense optical flow is computed from each RGB frame using the Farnebäck method (see Chapter 3, Section 3.2.2). This method makes a good compromise between accuracy and speed. The 3D motion estimation is a fundamental and challenging problem, but it could be solved using scene flow [LPB11, XZYT14].

To estimate the depth motion, suppose x and y are the coordinates of a point at a frame of the collected RGB video, z is the depth value of that point in the depth video, then the z point could be located with (x_t, y_t, z_t) coordinate in the RGB-D space. In this way, each point in the RGB-D data is treated as a 3D point.

The depth motion estimation is computed as follows: Suppose for point ($\mathbf{X_t}$) in the 2D RGB image plane, and $\boldsymbol{x}_t = (x_t, y_t)$ be its location at frame t is tracked to frame $(t + 1)$ at the point ($\mathbf{X}_{t+1}$) by using the Farnebäck optical flow method. The depth motion of point ($\mathbf{X_t}$) between two consecutive frames can be estimated by mapping $\boldsymbol{x}_t = (x_t, y_t)$ to the depth frame I_t^z, as in Equation (5.1):

$$\Delta\mathbf{X}_t^z = I_t^z(x_{t+1}, y_{t+1}) - I_t^z(x_t, y_t) \tag{5.1}$$

where $\Delta\mathrm{X}_t^z$ can be approximately considered as the depth flow $\mathbf{v}_z$. The depth flow field is computed by applying the extraction approach of ($\mathbf{v}_z$) to all points in the depth frame. $\mathbf{v}_z$ values are combined with the corresponding optical flow $(\mathbf{v}_x, \mathbf{v}_y)$, then the scene flow of (**X**) can be obtained as:

$$\mathbf{f}_t = (\mathbf{v}_x, \mathbf{v}_y, \mathbf{v}_z) \tag{5.2}$$

The 3DTr included 3D motion information obtained after computing the scene flow of all existing RGB Trajectory points. An example of 3DTr is shown in Figure 5.2.

5.2.2 Local Feature Extraction

Such 2DTrs, various appearance and motion features can be extracted along the 3Trs [XZYT14]. However, the MBH descriptor in the directions of MBH_x and MBH_y gave a good performance in RGB channels, as proposed by [WKSL13]. While in this work, i.e., by using RGB-D data, the MBH is extracted from the depth direction (MBH_z) besides these two MBH descriptors extracted from x and y directions, i.e., MBH_x and MBH_y. One of the significant advantages of MBH_z is robustness to camera motion along the depth direction. The model of MBH_x and MBH_y implemented by

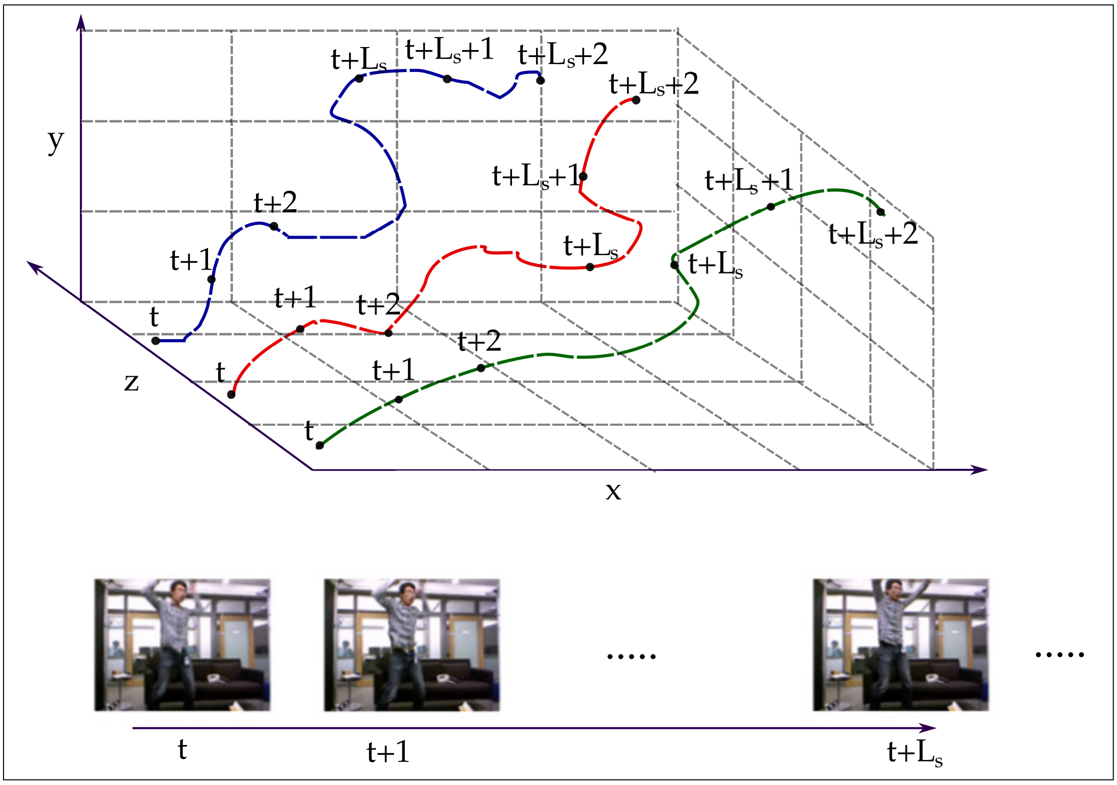


Figure 5.2: Example of the 3D Trajectories: The blue, red, and green lines represent 3D Trajectories. Trajectories sampled are tracked along with L frames.

[WKSL13, XZYT14] is also followed to extract MBH_z, where MBH descriptor splits the optical flow into horizontal and vertical components, and quantizes the derivatives of each component (see Chapter 3, Section 3.3.1). Finally, the concatenation of MBH_x, MBH_y, and MBH_z is done in an early fusion way to form the 3D activity description robust to camera motion as in Equation (5.3):

$$MBH_{xyz} = \left[MBH_x MBH_y MBH_z\right] \tag{5.3}$$

That is, for the depth flow $\mathbf{v}_z$, the orientation information is quantized into 8-bin histograms and the magnitude is employed for weighting. $2 \times 2 \times 3$ spatio-temporal grids are employed, which results in a $2 \times 2 \times 3 \times 8 = 96$ dimension for MBH_z.

Additionally, the Trajectory shape descriptor, along with RGB-D frames, is computed using Equation (3.11) in Chapter 3. By means, the Trajectory shape descriptor is computed in horizontal, vertical, and depth directions ($\boldsymbol{Tr}_x, \boldsymbol{Tr}_y$, and $\boldsymbol{Tr}_z$) to form the 3D Trajectory shape descriptor as in Equation (5.4):

$$\boldsymbol{Tr}_{xyz} = \left[\boldsymbol{Tr}_x \boldsymbol{Tr}_y \boldsymbol{Tr}_z\right] \tag{5.4}$$

The 3D Trajectories and local feature extraction process [XZYT14] is illustrated by Algorithm 2.

Algorithm 2 3D Trajectories and Local Feature Extraction

```
input : 2D Dense Trajectories
Extend 2D Dense Trajectories to 3D Trajectories

foreach 2D Trajectory $\boldsymbol{Tr}^{i}_{2D}$ do
    foreach 2D Trajectory point $\mathbf{X}^{j}_{2D}$ do
        • Map $\mathbf{X}^{j}_{2D}$ to the depth frame $I^{z}_{t}$

        • Evaluate the scene flow $\mathbf{f}_t$ of $\mathbf{X}^{j}_{2D}$
    end
    Extend $\boldsymbol{Tr}^{i}_{2D}$ to 3D Trajectory $\boldsymbol{Tr}^{i}_{3D}$ through availability check
    If $\boldsymbol{Tr}^{i}_{3D}$ is available, then extract activity features along it
end
output: 3D Trajectories and RGB-D action features
```

Additionally, appearance descriptors, such as HOG, and first-order motion, such as HOF, are also extracted from RGB. The appearance variations and motion are significant in training and testing samples and help to improve the performance.

Then, the 3D activity descriptors values are finally tested using the clustering and classification strategy to get the performance accuracy of the human action recognition system. Figure 5.3 shows the pipeline steps of human action recognition of 3D Trajectory.

5.2.3 Global Feature Extraction

The global-based descriptors encode more spatial and temporal information within video sequences in addition to the local feature descriptors. This descriptor method usually needs the human body's localization throughout background subtraction, alignment, or tracking, and it directly extracts and describes the entire characteristics of human silhouettes or contours. In this work, an action region is computed using background subtraction. Then, the 3DGIST feature descriptor is used to extract the global features from the action region of depth sequences.

Action Region

The background subtraction method (see Chapter 3, Section 3.2.1) is used to extract the shape of the human silhouette, which plays a very important role in recognizing human actions. Background subtraction is a popular algorithm for isolating a scene's moving

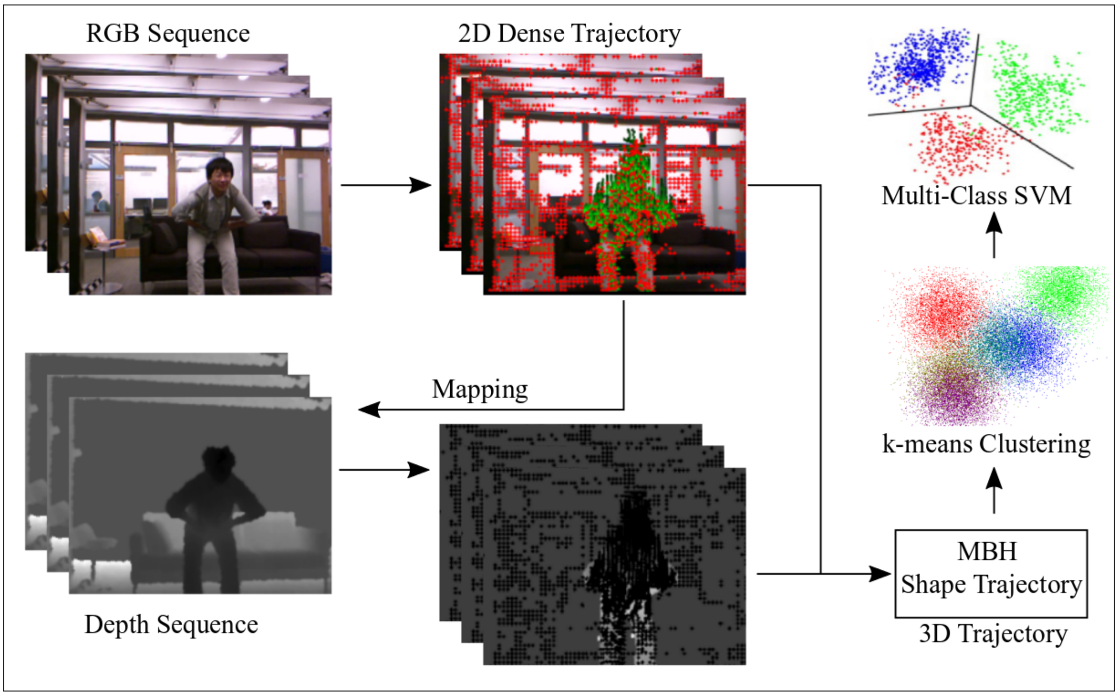


Figure 5.3: Pipeline of RGB-D action recognition with the 3D Trajectory.

parts by segmenting it into the background and foreground. Only the area where humans are located is extracted from each frame of a video.

Let's given a video with N_F frames; the human region in each frame is separated from the background, which is called Action-Region (A_R). i.e., extraction to the foreground from all frame sequences (moving object).

GIST Feature Computation from Depth

GIST feature is a kind of filter bank feature, as well it has global properties. In this work, the GIST descriptor (see Chapter 3, Section 3.3.2) is accumulated by the filter responses of the grids of A_R. The A_R is divided into $g \times g$ grids. The average filter response is computed from each grid. A computed vector is called a 3DGIST vector. The dimension of a 3DGIST vector is $\hat{l} \times \theta$. Finally, the whole region can be described by $(g \times g)$ 3DGIST vectors. These vectors capture the action structure. In this work, the 3DGIST feature is adopted into the video domain for the action recognition task. The 3DGIST feature is computed in the following steps:

- Compute 3DGIST descriptor by using a cluster of Gabor filters, as in Equations (3.19) and (3.20) that are discussed in Chapter 3, which the A_R Convolve with (32) Gabor filters at (4) scales and (8) orientations, resulting in (32) feature maps.

- Divide each feature map into (16) regions by 4×4 grids and then average the feature values within each region.
- Concatenate the (16) averaged values of all (32) feature maps, resulting in $(16 \times 32 = 512)$ 3DGIST descriptor dimensions. Therefore, the extracted features from each A_R are concatenated and resulting in 512-dimension 3DGIST feature vectors for each frame.

The overall features are then computed by the concatenation of all 3DGIST descriptors of A_R from the input video sequence that finally forms a global feature vector of action.

After computing the global 3DGIST feature vectors from all video sequences, the clustering and classification algorithms are applied to compute the accuracy of this feature extraction method when using only depth data. Figure 5.4 shows the structure pipeline of human action recognition from depth data based on the 3DGIST descriptor.

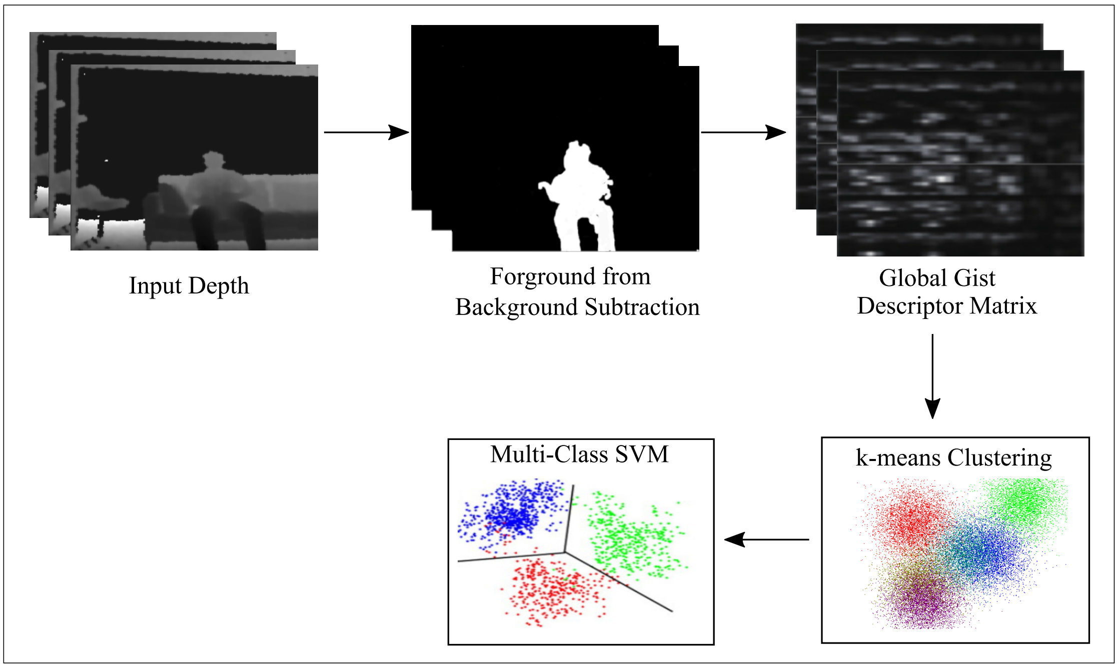

Input Depth
Forground from Background Subtraction
Global Gist Descriptor Matrix
k-means Clustering
Multi-Class SVM

Figure 5.4: Pipeline of depth activity recognition with the global 3DGIST descriptor.

5.2.4 Features Combination for Robust Action Recognition

This section has described the combination of local (i.e., 3DTrMB) and global feature (3DGIST) descriptors of video sequences. First, the 3D Trajectory features descriptor is used for feature extraction based on the same parameters used in 2DTr from RGB.

These parameters will be discussed in Section 5.4. Then, the motion information from depth images (MBH_z) is added to get finally 628 features for each frame. Second, the 3DGIST descriptor is used for feature extraction based on the scale and orientation by convolving it with Gabor filters from only depth images resulting in 512 features for each frame. Features of 3DTrMB and 3DGIST are normalized before concatenating them to form the overall feature vector of size 1140 feature's dimension. The normalization is required to ensure that individual feature vector values do not overshadow other values. Then, 3DTrMB and 3DGIST features of each frame are combined together to form an overall feature vector named 3DTrMBGG, as in Equation (5.5).

$$3DTrMBGG = \begin{bmatrix} 3DTrMB & 3DGIST \end{bmatrix} \tag{5.5}$$

Combining feature values has resulted in a better performance of the classifier since it utilizes the combined information from the local 3D Trajectory, MBH, and Global 3DGIST features analysis.

5.3 Action Recognition

After computing the features from video sequences, a multi-class SVM classifier (see Chapter 3, Section 3.5.1) has been used to classify the actions. This classifier is computed from the local feature, global features, and the combined features, i.e., local and global features.

The multi-class SVM classifier based on the One-vs-All approach (see Chapter 2, Section 2.5.1) with RBF kernel (Equation (3.36)) is used to compute the performance of the proposed method. Moreover, to estimate the best parameters for the SVM classifier, a grid search was conducted to know the best value for parameters C and γ. Here, γ represents the width of the RBF kernel, and C represents the weight of the error penalty. The proper set of (C, γ) helps in improving the overall accuracy of the SVM classifier. In this work, the C parameter equals 10, and gamma is set to 20. In clustering, The k-means clustering algorithm (see Chapter 2, Section 2.5.2) is used to generate the dictionary of visual words. Different k values are used in the experimentation, 400, 100, and 600 for 3DTrMB, 3DGIST, and 3DTrMBGG, respectively.

5.4 Experimental Results

In experiments, the proposed RGB-D activity recognition method is tested on two types of the public dataset, MSR Daily Activity 3D (MSR3D) and Online RGB-D (ORGBD) datasets (see Chapter 2, Section 2.7). In the experimentation, these datasets are split to train and test sets by following the method in Chapter 4, Section 4.3.1. The MSR3D dataset is divided into 70% for training and 30% for testing. While for the ORGBD dataset, the same environment test setting is used, where half of the subjects are used as training data, and the rest of the subjects are used as testing data.

In this work, extensive experiments are performed to study the effectiveness of the proposed method and improve the accuracy performance of human action recognition from RGB-D datasets.

5.4.1 Experimental Setup

The Trajectory shape descriptor parameters are the same as used by [WKSL13], where Trajectories sampled are tracked along with $L_s = 15$ frames (see Chapter 2, Figure 2.7), which is shown a graphical representation of the Trajectory procedure). In this procedure, firstly, feature points are densely sampled over the multi-scale pyramid. Then, Trajectories are constructed using dense optical flow. Finally, descriptors are computed around the Trajectory.

For the standard HOG descriptor, orientation is quantized into 8 bins. The magnitude is used for weighting. Therefore, the concatenation of the descriptor for each cell yields a final descriptor size of 96 for HOG. For HOF, 9 bins are chosen for those pixels whose optical flow value is within a threshold; therefore, in this case, the final descriptor size is 108.

The concatenation of two descriptors, i.e., MBH_x and MBH_y for RGB, has recently represented the MBH descriptor; both descriptors are constructed in the same way as the HOG descriptor, both of them has a total of 8 orientation bins. Hence, the $MBH_{x,y}$ descriptor's final size is $96 \times 2 = 192$. The HOG descriptor encodes spatial appearance while HOF captures first-order motion information. On the other hand, MBH_x and MBH_y refer to the x and y components of the optical flow. Therefore, the second-order motion information it encodes is the relative pixel movement, highlighting the image areas where the optical flow varies, i.e., the motion boundaries. In addition to the calculated MBH_x and MBH_y from the RGB frame, the MBH_z from the depth frame is also computed in this work with the size 96, then the final size of the $MBH_{x,y,z}$ descriptor from the RGB-D frame is 288.

In this chapter, three different models are proposed and experimented with, as in the following:

- The first model is implemented by using the local descriptor, Section 5.2.1. This descriptor is represented by combining the Trajectory shape descriptor with MBH from depth images to form the 3DTrMB. The feature descriptor dimensions from 3DTrMB are (628) for each frame in the video.

- The second model is implemented by using a global descriptor, Section 5.2.3. In this model, the feature vectors are extracted using the GIST descriptor from depth information (3DGIST). The feature descriptor dimensions from 3DGIST are (512) for each frame.

- The third model is represented by the combination of local and global descriptors, Section 5.2.4. The feature vectors are extracted by combining 3DTrMB and 3DGIST feature values in one vector to form 3DTrMBGG for each video action on RGB-D videos. The feature vector dimensions from 3DTrMBGG are (1140) for each frame in the video.

5.4.2 Results and Discussions

This section is demonstrated the evaluation performance of the proposed method using two different action datasets:

Evaluation using MSR3D Activity Dataset

The accuracy results of MSR3D are presented in Table 5.1. k-means clustering performed to the feature descriptors values at the training stage to label each action class, which yields a dictionary with size k. In these experiments, the k value of k-means is set to 400, 100, and 600 for 3DTrMB, 3DGIST, and 3DTrMBGG, respectively. Finally, the multi-class SVM classifier is applied to compute the proposed method's accuracy values. The computation of accuracy is computed using the same way in Chapter 4, Equation (4.2).

Dataset	Methods	Modality	Accuracy
MSR3D	Re-implemented 2DTr [WKSL11]	RGB	63.13%
	3DTrMB	RGB-D	92.0%
	3DGIST	Depth	93.33%
	3DTrMBGG	RGB-D	**95.62%**

Table 5.1: Experimental results on MSR3D dataset using different modalities.

These results summarize with the following conclusions:

- 3DTrMB from RGB-D outperforms 2DTr descriptor using only RGB. The 2DTr is re-implemented from the literature [WKSL11].
- 3DGIST descriptor from input depth only outperforms 2DTr and 3DTrMB descriptors, but the improvement against 3DTrMBGG is much smaller than the 2DTr descriptor.
- The new descriptor 3DTrMBGG outperforms the other three previous descriptors. The reason is due to the computation of motion, and structural information helps in increasing accuracy performance.

Comparison with state-of-the-art methods using the MSR3D dataset Table 5.2 summarizes the results, and it provides a comparison against the state-of-the-art that used the same datasets and different feature extraction methods.

Methods	**Accuracy**
Skeletal Shape Trajectories [ASS16]	70.0%
Relative Trajectories 3D [KBB14]	72.0%
MMDT [AABR$^+$17]	78.13%
3D Localized Trajectories [Pap20]	76.3%
Local Features-SVM (Chapter 4)	91.11%
Proposed Method	
3DTrMB	**92.0%**
3DGIST	**93.33%**
3DTrMBGG	**95.62%**

Table 5.2: Comparison of recognition accuracy with the state-of-the-art methods using MSR3D dataset.

The proposed method achieved a recognition rate of 92.0% from 3DTrMB, 93.33% from 3DGIST, and 95.62% from 3DTrMBGG methods. The results confirm that the proposed method outperforms the state-of-the-art methods recorded in Table 5.2. It is important to mention here that the depth information helped in increasing the performance of the proposed methods. Due to the proposed method, the computation of motion information, i.e., MBH, and structural information, i.e., GIST descriptor, could add more action information that helped in giving the correct class label. Thus a combination of descriptors leads to the performance gain. It is worth pointing out that combination works. The reason why it works is that HOG encodes appearance information, while other descriptors encode motion information; thus, the combination leads to the performance gain. In comparison with the other methods, some researchers depended on the skeleton data from depth. Depth information improves skeleton detection. Therefore many authors, as mentioned in Table 5.2 [ASS16] and [Pap20], focused on analyzing pose for action recognition. Nevertheless, skeleton detection still not robust and fail in more challenging scenarios, where the sensor is placed outside of the optimal working range, and serious occlusions occur. Another reason for the low recognition accuracies when there is no enough motion information computed from the input video, due to either characteristic of action or occlusions [KBB14, AABR$^+$17].

Evaluation using ORGBD Dataset

The accuracy results of ORGBD are presented in Table 5.3. Also, the k-means clustering is performed to the feature descriptors values at the training stage to label each action class. The same values of k in the previous datasets are used. Finally, the multi-class SVM classifier is applied for the computation of the proposed method's accuracy performance.

Dataset	Methods	Modality	Accuracy
ORGBD	Re-implemented 2DTr [WKSL11]	RGB	61.18%
	3DTrMB	RGB-D	81.52%
	3DGIST	Depth	83.81%
	3DTrMBGG	RGB-D	**97.62%**

Table 5.3: Experimental results on ORGBD dataset using different modalities.

These results summarize with the following conclusions:

- 3DTrMB from RGB-D outperforms 2DTr descriptor using only RGB.
- 3DGIST descriptor from input depth only outperforms 2DTr and 3DTr descriptors, but the improvement against 3DTrMB is much smaller than in the 2DTr descriptor.
- The new descriptor 3DTrMBGG outperforms the other three previous descriptors, and the improvement against 3DTrMB and 3DGIST is much higher than the 2DTr. The computation of motion and structural information could help in increasing accuracy performance. Additionally, the ORGBD dataset does not incorporate actions involving a significant amount of radial motion.

Comparison with state-of-the-art methods using the ORGBD dataset Table 5.4 compares the proposed method results with the existing state-of-the-art using the same dataset with different action recognition methods.

The proposed method achieved a recognition rate of 81.52% from 3DTrMB, 83.81% from 3DGIST, and 97.62% from 3DTrMBGG methods. The results confirm that the proposed method outperforms the state-of-the-art methods recorded in Table 5.4. About the comparison with other methods used ORGBD dataset, there are no more researches used it. Compared with the others' accuracy, one thesis used this dataset on the Trajectory method but depended on the skeleton joints [Pap20].

Some other researchers used this dataset on other different methods to improve human action recognition, as mentioned in Table 5.4. Ding et al. [DLCZ16] proposed the HSOM from 3D skeletal joint locations as input from depth maps. Yu et al. [YLY15] used the depth and skeleton joint as input to their system. Meng et al. [MDDB15] proposed the human-object interaction system using a set of joints located in the skeleton.

Methods	Accuracy
HOSM [DLCZ16]	49.5%
Orderlet+SVM [YLY15]	68.7%
Orderlet+ boosting [YLY15]	71.4%
Human-Object Interaction [MDDB15]	75.8%
3D Localized Trajectories [Pap20]	64.5%
2D Localized Trajectories [Pap20]	67.4%
Local Features-SVM (Chapter 4)	92.86%
Proposed Method	
3DTrMB	**81.52%**
3DGIST	**83.81%**
3DTrMBGG	**97.62%**

Table 5.4: Comparison of recognition accuracy with the state-of-the-art methods using ORGBD dataset.

As a summary, from the previous comparison with the other action recognition methods, the proposed method results outperform other previous methods on the same public datasets. These new feature extraction methods show the ability to recognize human action in high and low object movements and finally improved the human action recognition task. Moreover, the proposed method could improve the benefit of combining a set of local feature vectors with a global feature vector in a suitable manner. The advantage is that not only the main global attribute of human action is kept, but also the impact of occlusion and noise is reduced.

Chapter 6

Motion Saliency Detection for Effective Human Action Recognition

This chapter adopts the ideas of spatio-temporal analysis and global feature extraction. The global feature has been used to characterize texture information from body motions on RGB-D videos. The main contribution relies on pre-processing input videos to improve the feature extraction step and to have accurate results for recognizing human action.

The method in this chapter was presented and published at *the 26th International Conferences in Central Europe on Computer Graphics, Visualization, and Computer Vision (WSCG 2018), Pilsen, Czech Republic* [AaAdP18].

The outline of this chapter structures as follows. Section 6.1 presents an introduction. Section 6.2 describes an overview of the proposed system architecture method. Finally, the experimental and results are presented in Section 6.3.

6.1 Introduction

The proposed approaches presented in the previous two chapters, which recognize human action from RGB-D videos, use the new combinations of different local and global feature descriptors.

In this chapter, the improvement of action recognition will be achieved by using a different strategy. The method of spatio-temporal interest points has been successfully applied in human action recognition tasks. However, it often contains many interest points that are not relevant to human actions and affect the final recognition accuracy, such as the interest points lying in the complex background. Therefore, it is very important to filter the interest points before feature extraction, and the motion saliency analysis is an effective and standard method.

Several methods are used to detect the moving object and extract important information from videos, such as optical flow and background subtraction. In the previous chapters, these methods are used to segment a moving object from the video frame's background and track it, i.e., detect the area of interest (motion area) through the video frames. For human action recognition, the focus is to recognize the action of the subject in the video. Existing feature descriptor-based methods tend to be affected by the background of the video frames.

This chapter presents a novel system for human action recognition based on saliency object detection and extracts features using a global descriptor. The proposed system firstly detects salient objects in RGB-D video frames and then extracts features on such objects, i.e., feature extraction from the salient area. This system proposes a simple technique to detect and process only those video frames containing salient objects. Processing salient objects instead of all the frames makes the algorithm more efficient and suppresses the background pixels' interference on the detection process.

In this chapter, the human Retina model is applied for saliency detection from the motion area in videos. The resulting saliency object detection is used to compute the features from the spatio-temporal domains of the video. The extracted features are finally used along with the Random Forest classifier for improving human action recognition. Experiments are conducted on the three public action datasets that are shown the newly proposed method outperforms the state-of-the-art competing spatio-temporal feature-based human action recognition methods. More specifically, the significant contributions of the proposed method are illustrated below:

- Noise filtering: A high frequency spatial and temporal noise is filtered out.
- Detect saliency motion by using the spatial and temporal output from the Retina model.
- Extract features using the Local Binary Pattern descriptor as a global descriptor from the salient area, i.e., motion area.
- Take a powerful of a commonly used Bag-of-Words (BoW) pipeline and focus on the feature extraction step to address the action recognition problem.
- Show that the proposed method of using the LBP as a global descriptor could be accessed to a good action recognition performance.

The essential steps for the action recognition system utilized in this chapter will be explained in the next sections.

6.2 Overview of the Proposed System Architecture

This section describes the steps of the proposed action recognition system. The general steps of the proposed system are illustrated in Figure 6.1. This system contains the following three main steps: In the first step, the Retina model (see Chapter 3, Section 3.2.3) is used for pre-processing the input video. After that, saliency motion detection is computed from the video sequences. The second step represents the feature extraction step using the LBP descriptor (see Chapter 3, Section 3.3.1). Then, to address the action recognition problem, Bag-of-Words (BoW) pipeline is used, and finally, the feature vectors from all video sequences are generated. In the training step, including feature clustering and dictionary generation, features extracted from the training set are clustered to generate visual words. Histograms based on occurrences of visual words in the training set are used as features to train a classifier. Finally, a multi-class RF classifier is used to achieve action recognition. The following sections explain each step in detail.

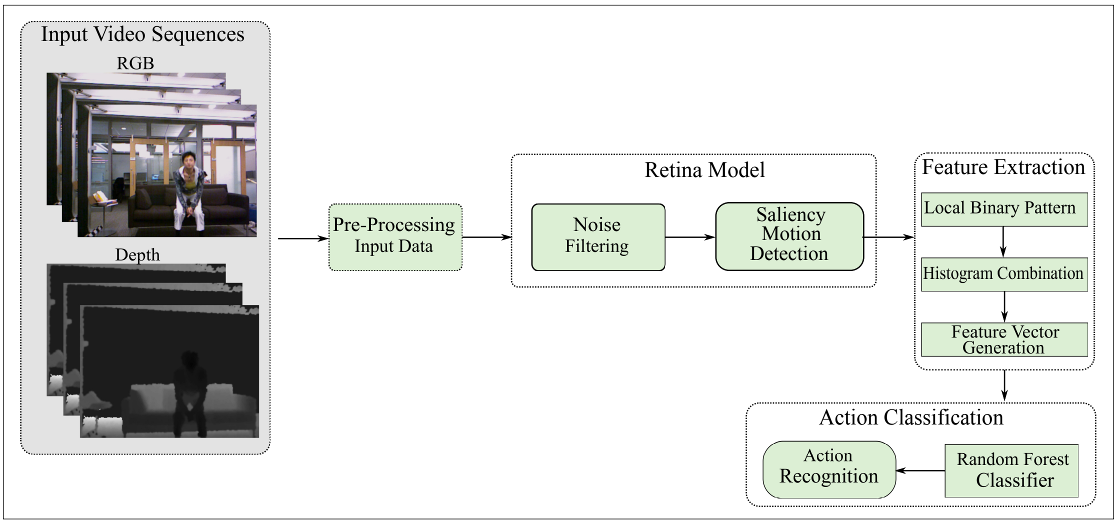


Figure 6.1: General structure steps of the system approach for human action recognition using RGB and depth video. Pre-Processing to the input video data, Motion detection, Feature Extraction, and Classification.

6.2.1 Pre-Processing Input Data

The first step in this system is pre-processed to the input data. When the input data is used as videos, it should be converted into frames (sequence of images) of size (100 × 100) for reducing the computational complexity of the system. In this work, the input data is represented by RGB and depth sequences. These data contain object appearance, shape, and motion characteristics. Because the depth images have noise, the spatial-temporal bilateral filtering is used to eliminate the unmatched edges from the depth images as in Equation (4.1), Chapter 4.

Then, before input RGB and depth sequence to the Retina model, these RGB and depth sequences are resized with `INTER_AREA` interpolation [7]. The interpolation method is used to shrink an image based on the re-sampling image using pixel area relation. This method is used for image decimation. A scaling factor value with value (2) is used along the horizontal and vertical axis for decimating the images.

6.2.2 Retina Model

The Retina model is a non-separable spatio-temporal filter model that can be used to preprocess the input video stream before applying the feature extraction. It also generates data channels for spatial and motion details analysis. The Retina model is explained in detail in Chapter 3, Section 3.2.3.

This model allows spatio-temporal processing of video sequences. It can whiten the image spectrum and remove the high-frequency spatial and temporal noise from the images, thus providing enhanced signals for the following processing stages. Additionally, this model can be primarily used for spatio-temporal video effects for a texture analysis by enhancing signal-to-noise ratio and enhancing information robust versus input images luminance ranges and motion analysis from both Retina channels, i.e., Parvocellular (Parvo) and Magnocellular (Magno).

Noise Filtering

At the Retina level, spatio-temporal filtering happens and guarantees an efficient structuring of video data [BCDH10], i.e., static and dynamic contour, noise removal, and illumination variation enhancement. Regarding the noise filtering:

- Noise reduction results from the non-separable spatio-temporal filtering. So, high frequency spatial and temporal noise is filtered out, and both outputs, Parvo and Magno channels, gain from this.

- At Parvo output, static textures are optimized, and noise is filtered, while on videos, temporal noise is eliminated. However, as human behaviors, moving textures are smoothed out. Then, moving object details can only be optimized if the Retina tracks and keeps them unchanged from its point of view.

- At Magno output, it allows for a cleaner motion detection with optimized noise errors even under difficult illumination conditions. As a compromise, the Magno output is a low spatial frequency signal that enables the blobs of events to be efficiently extracted.

This pre-processing increases the robustness of the feature extractor to disturbances such as noise and lighting variations. In addition, the Retina model can probably detect salient areas and compose spatio-temporal descriptors.

Motion Saliency Detection

The human Retina is used to create saliency maps to detect interest areas, i.e., moving objects in each video sequence. The human Retina model is available in OpenCV [13], and it is applied to the input video data. The Retina model de-correlation of details information of spatial and temporal by providing two output video channels, as illustrated below:

- The Parvo channel is the first channel of the Retina model. It is mainly active in the foveal Retina area and provides a perfect color vision for visual details with reduced spatio-temporal high-frequency noise. The Parvo channel processes the spatial details and colors. It enhances colors concerning the color temperature. It adapts to local luminance, thus enhancing local details in light and dark areas. From a temporal filtering point of view, it responds well to sustained signals while smoothing out fast temporal variations such as noise and fast motion. Furthermore, objects moving on the Retina projection are blurred. The Parvo Retina output is represented in Figure 6.2.

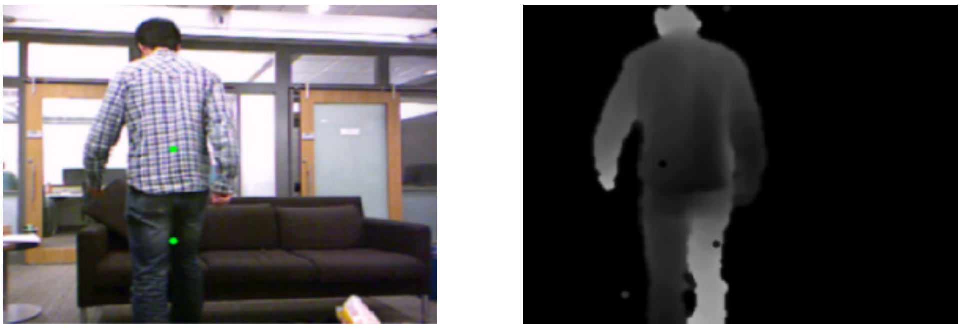
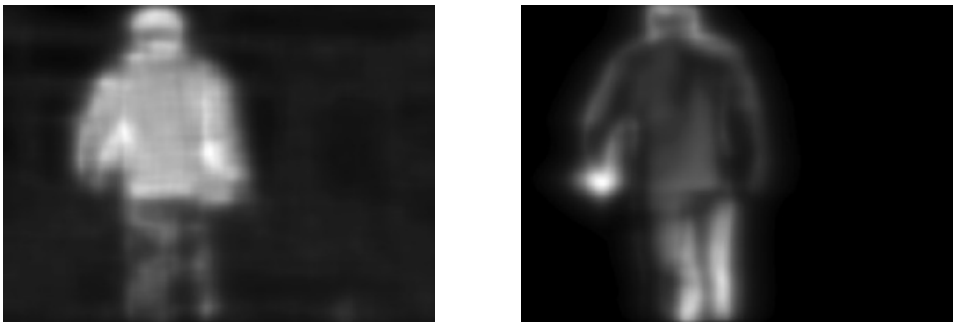

Parvocellular Output

Figure 6.2: The Retina Parvocellular channel (Parvo), left: RGB image and right: Depth image.

- The Magnocellular channel (Magno) is fundamentally active in the Retina environmental vision and sends signals related to change events (motion, moving events). It also improves the visual scene context and object classification from local contrast and noise removal benefits. The Magno channel does not distinguish between colors. However, it is sensitive to spatio-temporal events, giving strong responses to transient signals (quick spatio-temporal changes of light intensity, motion) and weak responses to slow-varying signals. This channel responds well to spatial boundaries during the first few tens of images after the Retina's initialization. Afterward, it only responds to motion and other transient events. The Magno Retina output is represented in Figure 6.3. A low-level spatio-temporal saliency detector can illuminate areas of possibly more interesting information dependent on the Magno channel.



Magnocellular Output

Figure 6.3: The Retina Magnocellular channel (Magno), left: RGB image and right: Depth image.

In this work, the Magno channel is implemented as a sequence of the gray-level image. This channel is used as a low-level spatio-temporal region of interest detector during a visual scene observation.

The output from the spatio-temporal Retina model, i.e., Parvo and Magno channels, represents the saliency motion area from spatial and temporal domains. From this area, the feature vectors are computed using the LBP descriptor.

6.2.3 Feature Extraction

The Bag-of-Features (BoFs) is the most popular feature representation technique in video action for recognizing the different human actions. The global feature vectors are computed from the spatio-temporal domain by depending on saliency motion detection and texture descriptor methods. The feature vectors are computed in two steps: Detect motion saliency from spatial and temporal images using the Retina model, as explained in Section 6.2.2, as well as extract features from the motion saliency area using the LBP descriptor.

Local Binary Pattern

After detecting the salient motion area from the input video, the texture LBP descriptor is used as a global descriptor to summarize the local structures from the motion area of the video sequences. In this work, the idea of spatio-temporal analysis and global feature extraction is adopted to characterize textures' information from body motions on RGB-D videos. Mathematical representation of the LBP operator is illustrated in Equation (3.12), Chapter 3.

Histogram Combination and Feature Vector Generation

Image texture can be described with its intensity histogram. So, the histogram of LBP images is used for representing an action. The feature vectors are only calculated for the active region in the images to make it invariant to the translational effect. Retina channels are used in this work to determine the action regions (motion area) in video sequences.

The histogram of the encoded motion area is obtained by applying the LBP operator and then used as a texture descriptor for that area in the images. These LBP histograms are combined to represent the spatio-temporal feature vectors from all images in a video, as in Figure 6.4.

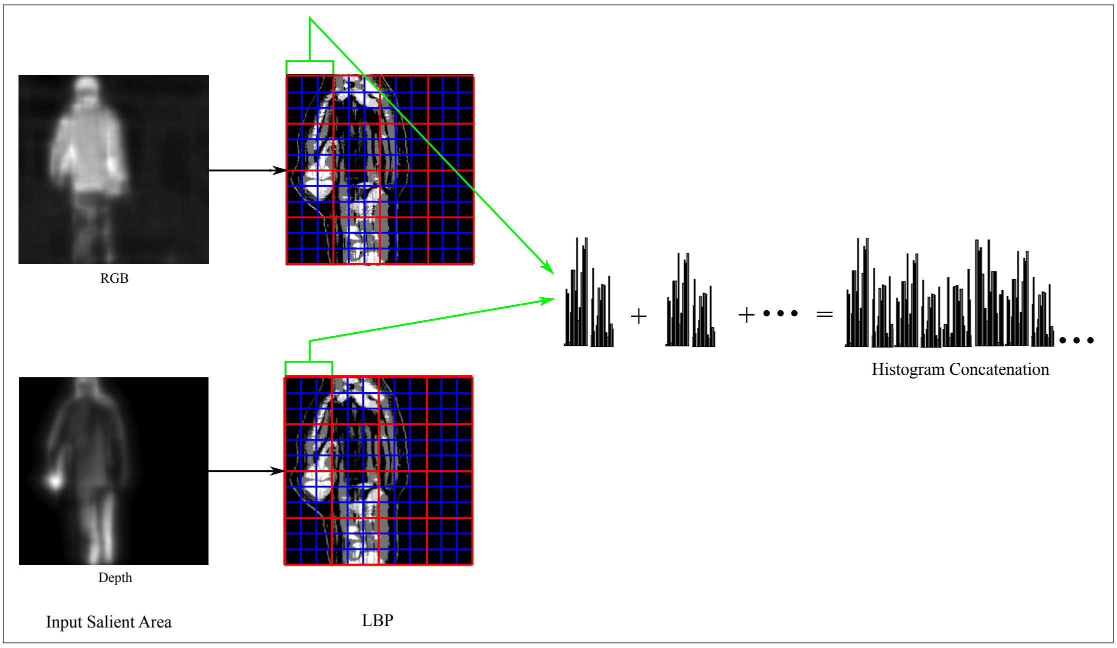


Figure 6.4: Feature vector generation.

The LBP histogram values are calculated in the following steps:

- Divide each input image into (4×4) blocks. After that, the examined area is divided into cells, i.e., 3×3 pixels for each cell, centered at the given pixel.

- Compare each pixel in a cell to each of its (8) neighbors, i.e., left-top, left-middle, left-bottom, right-top. Then, follow the pixels along a circle, i.e., clockwise or counter-clockwise. The considered neighbors can be adjusted by changing a circular radius around a pixel and by quantizing the angular space.

- If the value of the center pixel is greater than the value of the neighbor, compose "0". Otherwise, compose "1", which outputs an 8-digit binary number. Usually, this number converted to a decimal value for simplicity.

- Calculate the histogram of the frequency of each number that occurs over the cell, i.e., which pixels are extremely small, and which are larger than the center of each integration. This histogram can be described as a 256-dimensional feature vector.

- Normalize the histograms of all cells then concatenated them. This concatenation gives a feature vector for the entire area.

After computing the histogram from the motion area of video images, these values are combined to form the feature vector values representing the action in videos, such as walking, cleaning, etc. Then, the feature vector can be processed using the classifier method to recognize human action.

The LBP feature vector captures the global description of a saliency detected area due to local histograms' concatenation. The histogram's concatenation is up to the user, but it should be the same across all images.

6.2.4 Action Classification

For recognizing human actions, the classifier is needed after computing the feature vectors from the LBP descriptor. As a first, to represent the computed feature vectors, a BoW is used to encode the videos. The k-means clustering algorithm is used to generate the visual dictionary. The feature vectors are mapped to the nearest (closest) visual words, and then a video is represented as the frequency histogram through the visual words. After that, these BoW feature values are fed to the classifier for improving the action recognition performance.

In this work, Random Forest (RF) classifier (see Chapter 3, section 3.5.3) is used because it can handle thousands of input variables and large datasets. Moreover, the RF gives a good performance when it is used for action recognition tasks. In this work, the One-vs-All multi-class approach (Chapter 2, Section 2.5.1) is used. The best results are achieved with the RF classifier when the number of random trees is 10 and the maximum depth is 3.

6.3 Experiments and Results

Extensive experiments are performed using three different types of public action datasets to study the proposed method's effectiveness and the human action recognition system. These datasets included MSR Daily Activity 3D (MSR3D), Online RGBD (ORGBD), and Gaming 3D (G3D) datasets.

These input datasets are split into training and testing sets, as in Chapter 4, Section 4.3.1. These three datasets are split as follows: The MSR3D dataset is divided into 'seven' subjects for training and the rest of 'three' subjects for testing level, i.e.,70% for training and 30% for testing. For the ORGBD dataset, the same environment test setting is used, and it is split as half of the subjects are used as training data, and the rest of the subjects are used as testing data. Furthermore, the G3D dataset is divided by subjects, where the first 'seven' subjects (person) will be used for training (70% for training), and the remaining 'three' subjects are used for testing (30%).

6.3.1 Experimental Setup

In this experimentation, three necessary steps are done to compute the feature vector values: Motion detection using a spatio-temporal Retina model. The texture feature descriptor LBP is applied to the Retina output from both RGB and depth channels. Finally, the histogram from LBP is computed and combined with all histogram values to form the Bag-of-Words (BoW). The feature vector size is computed as $2^8 \times 2 \times (N_F - 1)$ from the gray-level images of RGB and depth channels, and N_F represents the number of video frames.

The recognition accuracy from the proposed system is computed from all feature vector values of different actions. The k-means clustering with different values of k is used as a dictionary size, where $k = 400$ for MSR3D and ORGBD datasets and $k = 800$ for the G3D dataset. Finally, the RF classifier is used for this experimentation to test the accuracy rates on three different types of datasets. The computation accuracy of the proposed method is based on Equation (4.2), Chapter 4.

6.3.2 Results and Discussions

The idea of the proposed method is computing a texture feature using the LBP descriptor from the spatio-temporal domains of the Retina motion detection model. Therefore, the Retinal strategy collects texture features in opposing color space from the Parvo and Magno pre-processed image frame instead of the original image. So, this method aims to take advantage of the "cleaner" Retina output channels to collect higher quality features than the baseline. The Retina channels are potentially used to detect salient areas and to compose features descriptor from spatio-temporal domains.

In the experimentation of this chapter, firstly extracted features from a single frame, i.e., keyframe, of the video scene. After that, three strategies of Retina used for texture features collection form the video sequences, as are described in the following:

- Retina Parvocellular (Parvo) extends the Retina strategy by obtaining texture features from a temporal domain of video frames centered on the keyframe. Additionally, the low-level saliency detector is applied for selecting more important features. Through the transient phase of the detector, large spatial features are selected, giving information about the scene's general composition. In contrast, the contribution of moving features is applied during the stationary state.
- Retina Magnocellular (Magno) extracts texture features from the same positions and time as Retina masking Parvo but uses grayscale attributes from the Magno channel instead of color features from the Parvo channel. These Magno features represent low-resolution data about the image's general appearance in the transient phase of the Retina. While these features give a rough description of contours perpendicular to the course of motion in the stable state.
- The last strategy is multichannel masking, which also extracts features at the same positions and time as the previous two approaches. However, it employs composite local features by concatenating the LBP feature of the image patch from the Parvo channel with the LBP (grayscale) feature at the same location but from the Magno channel. Through the stable state of the Retina, the feature from the Parvo channel embeds spatial appearance, while the feature from the Magno channel embeds the motion direction; this makes the combined spatio-temporal texture features.

Figure 6.5 shows an example of different Retina channels images and moving areas, where the original input is RGB and depth images are captured from the video sequence.

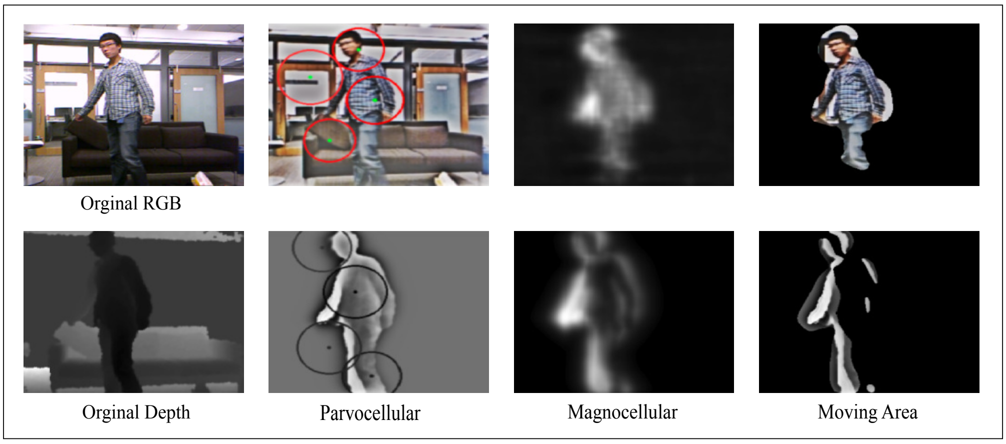


Figure 6.5: Effects of Parvocellular and Mgnocellular preprocessing and moving area from RGB and depth.

The experimental accuracy results are done on different Retina channels, as shown in Table 6.1, using three different datasets, i.e., MSR3D, ORGBD, and G3D datasets.

Retina Channels	**MSR3D**	**ORGBD**	**G3D**
Parvo+LBP	83.37%	93.13%	89.48%
Magno+LBP	**90.21%**	**96.86%**	**100%**
Parvo+Magno+LBP	81.11%	92.86%	71.43%

Table 6.1: Comparison of recognition accuracy using different Retina channels on three activity datasets.

The experiment results in the table illustrated that the best recognition accuracy is from the LBP features on the Magno Retina filter (motion detection) channel compared with other LBP on Parvo and Combination of Parvo-Magno Retinal channels. The reason behind that is the Magno channel gives intense energy in the detected area, while the Parvo channel is blurred there because there is a transient event.

Comparison with the state-of-the-art approaches The proposed approach has shown competitive performance compared to the state-of-the-art approaches.

In the MSR3D dataset, the Retina Magno and LBP accuracy results achieved the best performance that is reached 90.21%, and outperformed other approaches shown in Table 6.2. While the accuracy result from Parvo and LBP is reached 83.37%. The extracted LBP features from the combination of Parvo and Magno is reached 81.11%.

These two approaches are better than [ZZSS16, YLY15, PTDZ16, Pap20]. Another approach by [AFCYN19] is a little better because they used only skeleton data in their method, i.e., the author used a little information than in the proposed method that is dependent on the RGB and depth videos.

Methods	**Accuracy**
CHAR [ZZSS16]	54.7%
Discriminative Orderlet [YLY15]	60.1%
Feature covariance [PTDZ16]	65.00%
3D Localized Trajectories [Pap20]	76.3%
S+MDMMs [AFCYN19]	89.14%
Proposed Method	
Parvo+LBP	83.37%
Magno+LBP	**90.21%**
Parvo+Magno+LBP	81.11%

Table 6.2: Comparison results with other methods on MSR3D dataset.

In the ORGBD dataset, the accuracy results from all tested models achieved the best performance in the same-environment setting, as shown in Table 6.3. The achieved accuracy result from Magno-LBP is 90.21%, Parvo-LBP reached 93.13%, and the combination of Parvo-Magno and the extracted features from LBP are reached 92.86%.

Methods	**Accuracy**
HOSM [DLCZ16]	49.5%
3D Localized Trajectories [Pap20]	64.5%
Orderlet+SVM [YLY15]	68.7%
Orderlet+boosting [YLY15]	71.4%
Human-Object Interaction[MDDB15]	75.8%
Proposed Method	
Parvo+LBP	93.13%
Magno+LBP	**96.86%**
Parvo+Magno+LBP	92.86%

Table 6.3: Comparison results with other methods on ORGBD dataset.

In the G3D dataset, the Retina Magno and LBP accuracy results achieved the best performance that is reached to 100%, as shown in Table 6.2. While the accuracy result from Parvo and LBP is reached 89.48%. The combination of Parvo with Magno and the extracted features from LBP is reached 71.43%. The Parvo-LBP approach is better than other approaches that used the same datasets with different feature extraction methods

[BMA12, NWJ17, NJ14, Pap20], and the combination is not achieved better than the other researches. The result from Zhao et al. [ZXSJ19] approach looks better than the Parvo-LBP and Parvo-Magno-LBP approach because the authors in their research are used little information than in the proposed approach. Additionally, the testing results of this chapter depended on the same data processing for all datasets.

Methods	**Accuracy**
Bloom et al. [BMA12]	72.44 %
RBM [NWJ17]	84.0 %
RBM + HMM [NJ14]	86.40 %
2D Localized Trajectories [Pap20]	87.8%
HDM [ZXSJ19]	92.0%
Proposed Method	
Parvo+LBP	89.48%
Magno+LBP	**100%**
Parvo+Magno+LBP	71.43%

Table 6.4: Comparison results with other methods on G3D dataset.

As a summary, the proposed method represented by the feature extraction from saliency detection could be improved the human action recognition from RGB-D video. The overall system outperforms the state-of-the-art methods in terms of recognition accuracy for challenging action datasets. The proposed method can be useful for analyzing videos, especially for those with rich texture information.

Chapter 7

Activity Recognition Based on Dense 3D Optical Flow Co-occurrence Matrices

This chapter introduces a new human activity recognition system from RGB and depth video sequences. This system is investigated two different goals, including feature representation method and evaluation using different classification methods. These classifiers are applied to study the ability to learn and assess the system performance from 3D sensor data.

The method in this chapter was presented and published in two different events. The feature representation method was presented in *the 5th international Workshop on Sensor-based Activity Recognition and Interaction (iWOAR 2018), Berlin, Germany, and it was published by ACM* [AAP18a]. The rest of the experimentation results are included comparison results using different classifiers, which were presented at the *11th International Conference on Machine Vision (ICMV 2018), Munich, Germany, and it was published by SPIE* [AAP19].

The outline of this chapter structures as follows. An introduction is presented in Section 7.1. Section 7.2 describes in detail the proposed system overview of human activity recognition. Section 7.3 explains how to reach the activity recognition rate based on classification methods. Finally, the experimental and results are discussed in Section 7.4.

7.1 Introduction

In the previous chapters, many kinds of action recognition researches have been designed using different models to recognize the human activities of daily living. These models have significantly contributed to prove the efficient machine learning algorithm for action recognition tasks. Due to the significant challenges in action recognition include the reliability of prediction of each classifier as they differ according to the type of feature extraction methods.

In this chapter, three different challenges are implemented. Firstly, feature vectors are extracted from the motion area of RGB-D sequences. Secondly, despite the tremendous and varied work in the existing classification methods, the essential problem is that a single classifier cannot always lead to good recognition results. Sometimes, a classifier can outperform other classifiers on a particular problem, and that is the essential aim of the proposed work, i.e., find the proper classifier for the proposed method. Thirdly, the accuracy performance of the proposed method is computed to check the improvement of this method for the human activity recognition task.

The essential steps to improve human action recognition are motion estimation and feature extraction from videos. Typically, the most popular types of motion estimation from videos in the computer vision are dense optical flow and scene flow. Dense optical flow is apparent motion (translation) in the plane of an image, and scene flow is a 3D translation. These two types are usually evaluated as vector fields on the pixel grid.

In this chapter, the dense optical flow algorithm is applied for motion estimation, which calculates the displacement of brightness patterns between two successive frames. The intensity values of the neighboring pixels are used for this calculation. The algorithm that calculates the displacement for all image pixels is called the dense optical flow algorithm. Regarding the feature extraction, the Haralick feature is used for computing feature vectors from the estimated motion. These vectors finally fed to the classifier to assign an action label to the correct class. The main contributions of this chapter are illustrated below:

- Explore the local relations contained in the optical flow fields of RGB and depth sequences.
- Propose a new spatio-temporal feature descriptor, called 3D Optical Flow Gray Level Co-occurrence Matrices (3DOFGLCM). This descriptor is based on the GLCM that is computed over the optical flow field.
- GLCM expresses the orientation and magnitude components' distribution at a given offset over the optical flow from RGB and depth sequences.
- A set of measures well-known Haralick textural features is used to describe the flow patterns.
- Explore the different classifiers that show the effectiveness of the computed feature in the activity recognition performance.

7.2 Proposed System Overview

The human activity recognition overview of the proposed system is described in Figure 7.1. This proposed system has two main processes: Firstly, explore the local relations contained in the 3D optical flow field by proposing a novel spatio-temporal feature descriptor based on the gray level co-occurrence matrices (GLCM). The GLCM is computed over the optical flow field of the input RGB-D video to form a new descriptor, called 3DOFGLCM. Secondly, the Haralick texture features are extracted from the 3DOFGLCM matrices to compute the feature vectors from all input video sequences.

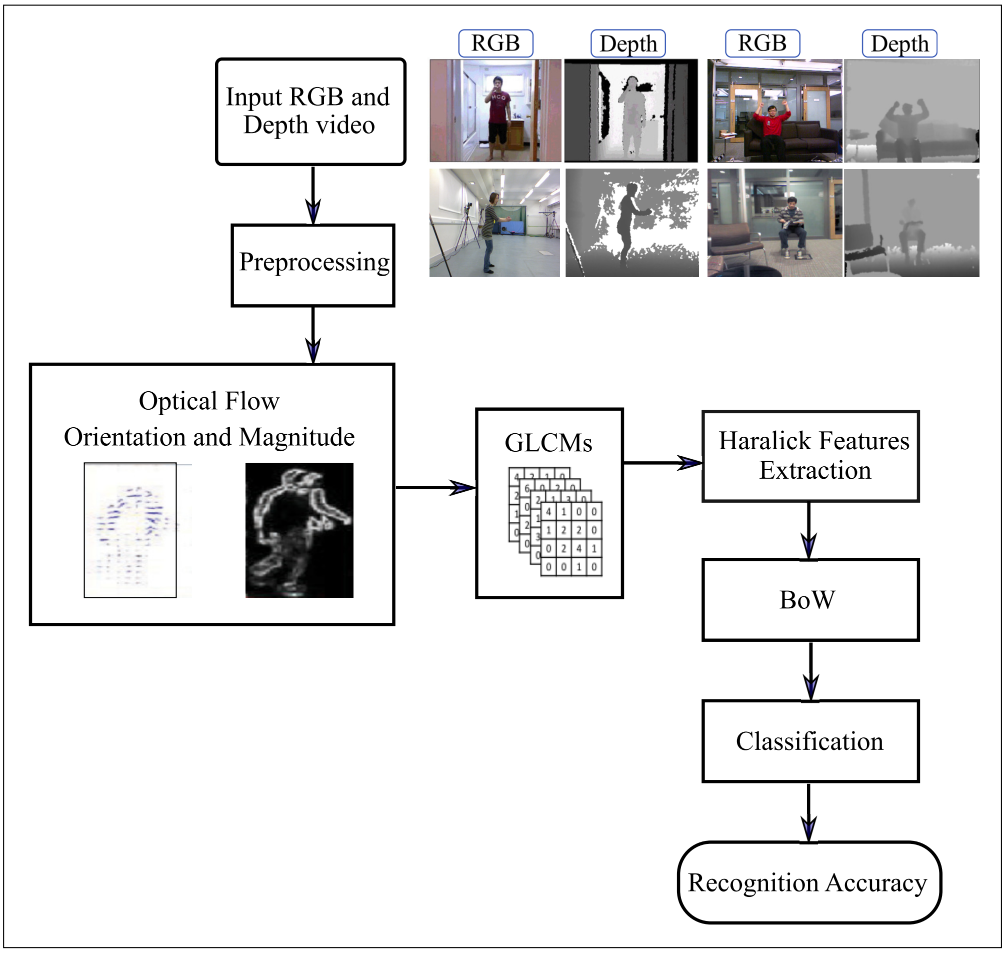

RGB
Depth
RGB
Depth
Input RGB and Depth video
Preprocessing
Optical Flow Orientation and Magnitude
GLCMs
Haralick Features Extraction
BoW
Classification
Recognition Accuracy

Figure 7.1: Overview of the proposed activity recognition system representing by preprocessing to the input RGB-D images, dense optical flow, feature extraction, classification and action recognition.

In addition to the feature computation, this work has also proposed a comparison of different classifiers to improve how they affect the ability to learn and estimate system accuracy performance from the computed features. Each step of the proposed system is illustrated in the following sections:

7.2.1 Pre-processing Input Data

Data pre-processing is an important step towards feature extraction and proper training of machine learning techniques. In this work, the input dataset consists of RGB and depth sequences. As the first step in this work, the input 3D sensor data should be processed. A low resolution of image size 320×240 is used to reduce the computational complexity of the proposed system. In this work, the RGB sequences are converted from color to grayscale levels. The depth map information captured by the Kinect camera is often noisy. Thus, spatial-temporal bilateral filtering is used (Equation (4.1), Chapter 4) to smooth and remove noise from depth information. The depth inpainting approach [5] is also applied to paint the missing areas of particular kinds, such as occlusions, holes caused by object-removal, or missing caused by sensor defects.

7.2.2 3D Dense Optical Flow

Optical flow is a displacement data generated to capture the temporal 3D motion of points from RGB-D sequences and to extract the important feature vector from these motion information.

Each video sequence, i.e., RGB and depth, is divided into pairs of consecutive grayscale frames. The dense optical flow is then used to calculate the motion between each pair of consecutive frames, taken at times t and $(t+\Delta t)$. This means that the optical flow evaluates a translation in the image plane, for each pixel location $\boldsymbol{x} = (x, y)$ in the domain of the RGB image I and pixel location $\overline{\boldsymbol{x}} = (\overline{x}, \overline{y})$ in the domain of the depth image D are assigned a motion vector $\boldsymbol{u} = (u, v)$ and $\overline{\boldsymbol{u}} = (\overline{u}, \overline{v})$ for RGB and depth image in the image plane, respectively, where u, v, $\overline{u}$, and $\overline{v}$ are measured in pixels. In this work, the dense optical flow will be computed in the orientation and magnitude fields of 3D data. Then, the co-occurrence of the optical flow is considered from RGB and depth data. Finally, the Haralick features will be computed from the co-occurrence matrices.

Optical flow orientation-magnitude fields The orientation field of optical flow captures the displacement information. In contrast, the magnitude field aims to extract motion information from the optical flow, which is the magnitude of the optical flow indicating the movement's velocity from a video sequence. The optical flow is computed using the Farnebäck method (see Chapter 3, Section 3.2.2). So, the spatial relationship of magnitude and orientation in local neighborhoods captures not only displacement by using orientation but also magnitude, which provides information about the movement's velocity.

To build co-occurrence matrices (GLCM) for each frame in RGB and depth sequences, two maps are built for each flow field computed on the frame based on the magnitudes and orientations of the flow field, i.e., these maps will be used as an input to the GLCM. In this way, for each video sequence that has (N_F) frames, there are $(N_F - 1)$ magnitude maps, i.e., $\mathbf{M}^{RGB}$ maps are computed from RGB frames, and $\mathbf{M}^{D}$ maps are computed from depth frames, and $(N_F - 1)$ orientation maps, i.e., Θ^{RGB} maps are computed from RGB frames, and Θ^{D} maps are computed from depth frames as in Equation (7.1):

$$\begin{aligned}
\mathbf{M}_{i,j}^{RGB} &= \sqrt{u_{i,j}^2 + v_{i,j}^2} \\
\mathbf{M}_{i,j}^{D} &= \sqrt{\overline{u}_{i,j}^2 + \overline{v}_{i,j}^2} \\
\Theta_{i,j}^{RGB} &= \tan^{-1}\left(\frac{v_{i,j}}{u_{i,j}}\right) \\
\Theta_{i,j}^{D} &= \tan^{-1}\left(\frac{\overline{v}_{i,j}}{\overline{u}_{i,j}}\right)
\end{aligned} \tag{7.1}$$

where u, v, $\overline{u}$, and $\overline{v}$ represent the horizontal and vertical motion components of each flow vector contained in the flow field of row and column, respectively.

Since the pixels values in $\mathrm{M}_{i,j}^{RGB}$ and $\mathrm{M}_{i,j}^{D}$ do not occupy all the range of the grayscale images [0, 255] for 8-bit grayscale images. Therefore, for a better result, the magnitude maps are normalized, as in the following Equation (7.2):

$$\begin{aligned}
(\mathbf{M}_{\mathbf{i,j}}^{\mathbf{N}})^{RGB} &= \frac{\mathbf{M}_{i,j}^{RGB} - min_{rgb}}{max_{rgb} - min_{rgb}} \times Max_{rgb} \\
(\mathbf{M}_{\mathbf{i,j}}^{\mathbf{N}})^{D} &= \frac{\mathbf{M}_{i,j}^{D} - min_{D}}{max_{D} - min_{D}} \times Max_{D},
\end{aligned} \tag{7.2}$$

where, $(\mathrm{M}_{\mathbf{i,j}}^{\mathbf{N}})^{RGB}$ and $(\mathrm{M}_{\mathbf{i,j}}^{\mathbf{N}})^{D}$ are the optical flow magnitude value at pixel position (i, j) from the grayscale of RGB images and depth images, respectively. min_{rgb}, max_{rgb}, min_D, and max_D are the minimum and maximum values in the magnitude maps computed from RGB and depth. Finally, $(\mathrm{M}_{\mathbf{i,j}}^{\mathbf{N}})^{RGB}$ and $(\mathrm{M}_{\mathbf{i,j}}^{\mathbf{N}})^{D}$ are the normalized optical flow at pixel position (i, j), and Max_{rgb} and Max_D are the maximum possible pixel value (255 for 8-bit grayscale images).

Moving object area detection The normalized optical flow fields contain noise, i.e., detect false movement. This problem can be eliminated by using a threshold-based segmentation method, such as *Otsu's* method [Ots79]. In this method, the grayscale image histogram is computed, and the optimal threshold value is determined based on the histogram peaks. The value of the threshold is chosen as an amount between two peaks of the image histogram. Thus, the values of pixels above the threshold are set to one, and the values below the threshold are set to zero [MPC19]. *Otsu's* implementation of this work is based on the OpenCV library [6]. *Otsu's* threshold method is applied to create a binary image representing the areas of more substantial optical flow (moving object area). The areas of higher optical flow are preserved, and the areas of lower optical flow are eliminated by applying *Otsu's* threshold method. The binarized images of the normalized optical flow fields are used to compute gray level co-occurrence matrices.

7.2.3 Gray Level Co-occurrence Matrices

GLCM is a matrix that describes the relative frequencies of a pair of gray-levels present at a certain distance $(\Delta x, \Delta y)$ apart and a particular angle α. Distance $(\Delta x, \Delta y)$ ranges from 1 to the input image size while α ranges in four directions, i.e., $0°$, $45°$, $90°$, and $135°$. GLCM created from various pairs of angles and distances gives quite various feature values. The extraction of textural information from images containing highly directional characteristics is essentially dependent on selecting the correct angle α.

Usually, all four directions are taken, and the mean of features calculated from all the four GLCMs. However, in such a process of taking the mean, the directional information of textured images is lost, and thus classifications do not achieve reasonable accuracy. The texture characteristics calculated along the relevant direction are quite different from those calculated along with the other three directions. Therefore, in this work, each co-occurrence matrices is considered with a distance of $(\Delta x, \Delta y) = 1$ and $\alpha = 0°$. The distance value (equal 1) is chosen to avoid loss of information. The values $45°$, $90°$, and $135°$ are not added through the feature's computation due to increased computational load, and the classifications will not improved an accuracy due to the loss of directional information. In this work, the GLCM is computed from the magnitude and orientation fields using Equation (3.21), in Chapter 3. Note that in the proposed method, the input values to the Equation (3.21) are used the binarized images of the magnitude and orientation maps using *Otsu's* method instead of using the image intensity values to compute the matrices in the standard formula.

7.2.4 Haralick Features Extraction

Feature extraction is an important task to capture information from the motion area to represent all motions and textures values in a video sequence. The six textural features are chosen to be extracted from the GLCM. These features are included: Energy, Contrast, Homogeneity, Entropy, Sum Average, and Correlation, and they are defined in Chapter 3, Equations (3.22) to (3.31).

In this work, optical flow, GLCM matrices, and Haralick features are computed from both RGB and the corresponding depth. For each co-occurrence matrix, a feature vector is generated with L^l dimensions per matrix, where L^l is the number of extracted Haralick features. In this computation, six Haralick features are extracted from RGB, and the same type of six features are computed from depth.

After generating feature vectors from RGB and depth sequences, all feature vectors are concatenated for providing final feature vectors with a length of $4 \times (\alpha \times L^l) \times N_F - 1$, where 4 represented the number of feature maps that were used to compute GLCM from RGB-D sequences. Then, the computed feature vectors are represented by using the BoW pipeline.

7.2.5 Bag-of-Words

The Bag-of-Words (BoW) is applied (see Chapter 3, Section 3.4.1) to represent the features extracted from the RGB-D sequences. For each video sequence, the BoW feature vector is computed. First, spatio-temporal features are quantized into visual words, and then a video is represented as the frequency histogram over the visual words. k-means clustering with Euclidean distance (see Equation (2.6), Chapter 2) is applied as the distance metric between the features and the closest visual word.

In this work, the BoW vectors are computed from all videos in the training and testing stages, as mentioned in Chapter 4, Section 4.2.3. These BoW vectors are fed into the one-against-all classifier to predict the computed features and evaluate the proposed method.

7.3 Activity Recognition

After computing the BoW from action samples, these BoW vectors are fed into a classification pipeline. In this work, to test the feature representation approach more thoroughly, five classification methods employed, such as Artificial Neural Networks (ANNs), Naive Bayes classifier (NB), Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM), these classifiers are explained in detail in Chapter 3, Section 3.5.

In this work, the datasets are split into training and testing sets. The BoW feature vectors are computed from all training set in the training stage, and labels are appended according to the class. These vectors are fed into the multi-class classifier (see Chapter 2, Section 2.5.1) to train the model that is further used in the testing stage for computing the performance of the proposed activity recognition system. The datasets are split using the hold-out method (see Chapter 4, Section 4.3.1). The accuracy of the proposed method is computed by using Equation (4.2).

For aiming at a fair comparison in this work, the same evaluation pipeline is applied for every classifier. The evaluation pipeline is included in two phases, i.e., training and testing phases. In the training phase, spatio-temporal feature vectors are densely extracted from training videos. Dense sampling extracts video motion information in space and time. The local features are encoded to be used for the classification task. In this work, the dictionary from these feature vectors is generated using the k-means clustering algorithm. In the testing phase, test video sequences are classified by applying the trained classifier obtained during the training phase. Therefore, for the testing video sequences, spatio-temporal feature vectors are extracted from the detected motion area. Then, the BoW feature vectors are generated using the dictionary previously created. Finally, the generated feature vectors are given as input to the trained classifier to predict the class label of the test video sequences.

7.4 Experimental and Results

This section describes the experimental results obtained from the 3D dense optical flow, GLCM, and Haralick features (3DOFGLCM) from different video action of the 3D sensor datasets. The main focus of this experimentation is to find an accurate classifier that tests on the computed feature descriptor. Five different classifiers are used to compute the accuracy of the human activity recognition task. Additionally, the best classifier results will be compared with the accuracy results to other feature descriptors from the literature. The classifiers used in this work are included Artificial Neural Networks (ANNs), Naive Bayes (NB) classifier, Support Vector Machine (SVM), K-Nearest Neighbor (KNN), and Random Forest (RF).

The feature vectors obtained from the proposed descriptor 3DOFGLCM are extracted from sequences of RGB and depth video actions. The video actions used in this experimentation are included four types of public datasets: G3D, CAD-60, MSR3D, and ORGBD datasets. These datasets are explained in detail in Chapter 2, Section 2.7.

As mentioned previously, the datasets are divided into two sets by using a hold-out method. The datasets are split into training and testing sets according to the number of persons (subjects) doing the same action. So, these four datasets are split as follows: The G3D dataset is divided by subjects where the first 'seven' subjects (person) will be used for training, and the remaining 'three' subjects are used for testing. The CAD-60 dataset has 4 subjects, and it is split as the first 'two' subjects for training and the remaining 'two' subjects for testing. The MSR3D dataset is divided into 'seven' subjects for training and the rest 'three' subjects for testing level, i.e.,70% for training and 30% for testing. For the ORGBD dataset, the same environment test setting is used, and it is split as half of the subjects are used as training data, and the rest of the subjects are used as testing data.

7.4.1 Parameters Setting

This section has presented the experimentation parameters that are used for feature vector extraction and the classification methods. Regarding the parameter setting for the 3DOFGLCM method, the offset distance $(\Delta x, \Delta y) = 1$ is used to create the co-occurrence matrix, also the extracted Haralick textural features $L^l = 6$ for each input map, i.e., RGB and depth. These parameter values will be followed in all experimented datasets.

Regarding the classifiers (see Chapter 3, Section 3.5), the parameters set for each classifier are different. For Artificial Neural Networks (ANN): It has four layers, i.e., the input layer size is equal to the number of features (column), two hidden layers, and one output layer. For Support Vector Machines (SVM), the multi-class SVM with RBF (radial basis function) kernel is used. The C parameter is 10, γ is set to 20, and the degree is 3. Naive Bayes classifier is a probabilistic machine learning model used for a classification task (see Chapter 3, section 3.5.4). For Random Forest, a multi-class RF implementation and a few parameters need to be chosen. The number of random trees is set to 100, and the maximum depth is 25. For K-Nearest Neighbor (KNN), a K parameter is defined as the default value, and it is calculated using the OpenCV library [9].

For all classifiers, the One-vs-All approach for multi-class classification is followed in this thesis experimentation.

7.4.2 Experimental Results

Table 7.1 and Figure 7.2 are showed different activity classification rates using the four activity recognition datasets and five machine learning classifier methods. The split data and accuracy values are computed based on the methods in Chapter 4, Section 4.3.1, and Equation (4.2) for the accuracy computations.

Datasets	**ANN**	**NB**	**SVM**	**KNN**	**RF**
G3D	68.30%	65.42%	77.08%	82.41%	**88.40%**
CAD-60	69.23%	76.92%	81.82%	90.19%	**95.45%**
MSR3D	56.16%	63.17%	79.51%	70.74%	**80.67%**
ORGBD	59.51%	67.88%	80.96%	71.56%	**83.01%**

Table 7.1: Comparison of recognition rates from different classifier on four different RGB-D datasets.

The comparison of accuracy results in Table 7.1 shows that the best accuracy recognition value when using the RF classifier compared with other classifiers. RF classifier contained a bunch of combined decision trees that can handle categorical features very well. Also, it can work with high dimensional spaces and a large number of training

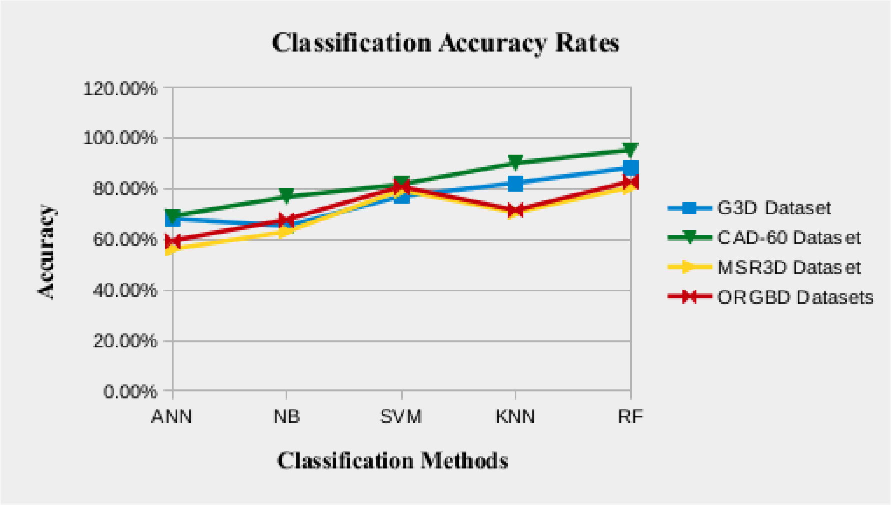


Figure 7.2: Accuracy rates comparison using four different RGB-D datasets.

examples. Regarding the 3DOFGLCM feature extracted method, the CAD-60 dataset gave the best results than other datasets in all classifier method. In this dataset, in most cases, people are standing in front of a camera, and there are no occlusions. In other tested datasets, there is an object in the scene, such as a sofa, or the same action sometimes performs twice, e.g., in the MSR3D dataset, a subject walking in front of a sofa and walking behind it.

Comparison against the state-of-the-art approaches The 3DOFGLCM proposed approach compares with several classic local spatio-temporal features from the literature. These comparisons are made with the same datasets and different feature extraction methods.

According to Table 7.2, a considerable improvement was obtained with 3DOFGLCM-RF reached to 88.40% of accuracy on the G3D datasets. The proposed approach has shown better performance compared to the state-of-the-art approaches. Nevertheless, the proposed method in Chapter 6 is still better. The saliency detection using Retina motion detection is better than motion detection using the OF.

For the experiments using the CAD-60 dataset, the 3DOFGLCM-RF achieved the best results, reaching 95.45 % accuracy. More motion information could be computed in this dataset because people are standing in front of a camera, and only four subjects perform 12 actions. Additionally, the proposed method is based on the feature extracted from RGB and depth images, while the other researchers have depended on the skeleton information in their feature computation.

For the experiments on the MSR3D daily activity dataset, Table 7.4 shows the experimental comparison with other methods. For this dataset, the 3DOFGLCM-RF also

Methods	**Accuracy**
Bloom et al. [BMA12]	72.44 %
RBM + HMM [NJ14]	86.40 %
LRBM [NWJ17]	87.94 %
Magno+LBP (Chapter 6)	100%
3DOFGLCM-RF	**88.40 %**
Proposed Method	

Table 7.2: Comparison of recognition performance with the different method used G3D dataset.

Methods	**Accuracy**
Object Offordences [KGS12]	71.4 %
Actionlet [WLWY14]	74.7 %
Pose Kinetic Energy [SA14]	91.9 %
CHAR [ZZSS16]	92.0 %
Joint Orientations [Mag20]	95.0%
Proposed Method	
3DOFGLCM-RF	**95.45 %**

Table 7.3: Comparison of recognition performance with the different method used CAD-60 dataset.

achieved the best result and reached 80.76 % accuracy than the other literature methods. The reason for achieving the best results is that the extracted features from RGB and corresponding depth cause this good result. At the same time, the other methods from literature depend on the RGB and skeleton information. This information is affected by occlusion and camera motion. The proposed method based on the saliency detection in the previous chapter is still better than the motion detection by OF and reached 90.21%

Methods	**Accuracy**
CHAR [ZZSS16]	54.7%
Discriminative Orderlet [YLY15]	60.1%
Feature covariance [PTDZ16]	65.00%
3D Localized Trajectories [Pap20]	76.3%
Moving Pose [ZLS13]	73.80%
Magno+LBP (Chapter 6)	90.21%
Proposed Method	
3DOFGLCM-RF	**80.76 %**

Table 7.4: Comparison of recognition performance with the different method used MSR3D dataset.

For the ORGBD dataset, Table 7.5 shows the proposed approach comparison results with several local spatial and temporal features from the literature. According to the table results, considerable improvement is obtained with the proposed method (3DOFGLCM-RF) that is reached 83.01% of accuracy. The saliency motion detection implemented in Chapter 6 is still better than the OF motion detection method and reached 96.86% in this dataset. Compared with other literature methods, the proposed method represented by 3DOFGLCM-RF could achieve the best results due to the computation of motion information and extracted features from RGB and depth together, i.e., it does not depend on only depth or only skeleton information.

Methods	**Accuracy**
HOSM [DLCZ16]	49.5%
3D Localized Trajectories [Pap20]	64.5%
2D Localized Trajectories [Pap20]	67.4%
Orderlet+SVM [YLY15]	68.7%
Orderlet+ boosting [YLY15]	71.4%
Human-Object Interaction[MDDB15]	75.8%
Magno+LBP (Chapter 6)	96.86%
Proposed Method	
3DOFGLCM-RF	**83.01%**

Table 7.5: Comparison of recognition performance with the different method used ORGBD dataset.

As a summary, all tested results were done by using RGB and depth datasets for human activity recognition and could achieve a good recognition performance compared to the literature methods. The same computation to a proposed feature descriptor obtained from the GLCM of 3D dense optical flow and Haralick features is compared using different machine learning classifiers. This comparison was made to improve the activity recognition performance and find the best classifier method that could be used with the proposed feature vectors extraction method computed from different activities.

A good performance was achieved from this features computation method because the feature captures important temporal motion information from 3D data, i.e., RGB and depth channels since orientations and magnitudes from RGB-D images give more information than one channel of the input images.

Chapter 8

3D Convolution Neural Networks for Human Action Recognition

In recent years, the use of neural networks in computer science has gained more and more interest. Significant hardware improvements, such as multi-core processors, efficient and affordable computation on GPUs, have boosted its popularity. Convolutional Neural Networks (CNNs) are utilized to solve image processing tasks, e.g., object recognition and classification. However, their application spectrum is not limited to static images. In this chapter, 3D-CNN will be used for human action recognition from video data. The task of action recognition involves tracking the temporal movement information and extracting features using the 3D-CNN.

The method in this chapter was presented and published at *the 26th International Conferences in Central Europe on Computer Graphics, Visualization, and Computer Vision (WSCG 2018), Pilsen, Czech Republic* [AaPG18].

The rest of this chapter is structured as follows. An introduction is presented in Section 8.1. Section 8.2 introduces the proposed approach to the system model. Finally, the experimental and results are provided in Section 8.3.

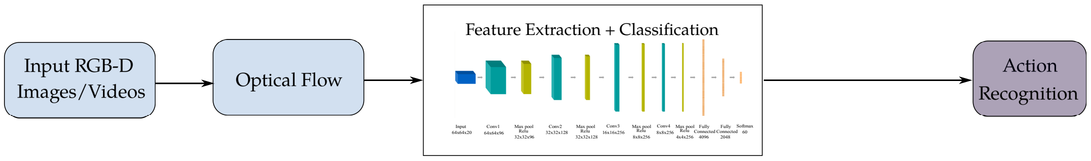
8.1 Introduction

In the previous chapters, human action recognition systems are represented using handcrafted features and machine learning methods. The experimentation results of the developed feature extraction methods gave good results and outperformed the state-of-the-art methods. In this chapter, a different way of feature extraction and training method will be implemented.

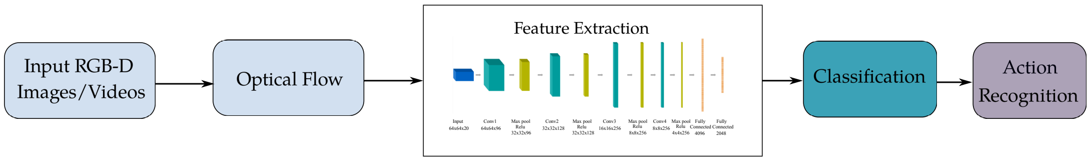
In this work, one of the deep learning methods, i.e., Convolution Neural Network (CNN), is used to improve human action recognition from 3D data. The CNN is a deep model that has complex hierarchical features through a convolutional operation that alternates with a sub-sampling process on the raw input images. Thus, CNN can automatically learn features from training videos. Moreover, CNN has invariance for a particular pose, illumination, and disorderly environmental change. However, CNN models can handle 2D input and reach good results in image classification. In this chapter, a new 3D-CNN models for action recognition are developed by proposed two new 3D-CNN models from RGB and depth video. These models are implemented and trained using the Tiny-Dnn framework [15] (for more details, see Appendix B). These models extract features from the temporal domains by implementing 3D convolutions that can capture motion information into multiple adjacent frames using optical flow (OF). The developed model produces multiple channels of information from the input frames, and the final feature representation integrates information from all channels.

Thus, in this chapter, human action recognition is improved from the temporal domain of RGB and depth videos of the big public RGB-D datasets. More specifically, the main contributions of this chapter are represented as follows:

- Propose a 3D Convolutional Neural Network (3D-CNN) model based on optical flow information for recognizing the action from video sequences.
- Use the 3D-CNN model for learning high-level descriptors from low-level motion features (optical flow) by using two input video channels, i.e., RGB and depth. These channels are represented as OF-RGB-CNN and OF-Depth-CNN models.
- Train temporal information, see Figure 8.1 (a), i.e., uses the optical flow from RGB and depth sequence as the input to 3D-CNN.
- Improve the possibility of extracted features from 3D-CNN of temporal information from RGB-D, see Figure 8.1 (b), which leads to two feature vectors because the RGB and depth are processed independently. These feature vectors are classified with a multi-class Support Vector Machine (SVM).
- Develop the 3D-CNN to be used with depth data instead of using only RGB sequences. This development could improve better performance with depth data than the RGB sequence.



a) Training 3D-CNN



b) Feature Extraction From 3D-CNN

Figure 8.1: Structure of the proposed two CNN models for improving human action recognition form RGB and depth sequences. Tow models are processed: a) Training 3D-CNN directly from the optical flow. b) 3D-CNN as a features extractor from the temporal domain.

8.2 Proposed Action Recognition System

In order to train the task of action recognition and because it is difficult to find a pre-trained CNN that is publicly available, two models of 3D-CNN are implemented, as shown in Figure 8.1. The first 3D-CNN model is trained and evaluated directly inside CNN. In the second model, the 3D-CNN is used as a feature extractor. The extracted features are trained and evaluated using a traditional machine learning classifier. The technical details of the proposed methodology system are shown in Figure 8.2. The process steps of this system will be explained in details in the following sections:

8.2.1 Pre-processing Input Data

The input dataset of this method is RGB and depth data channels. Each channel is pre-processed separately as explain below:

RGB Video Pre-processing

The original RGB dataset comes in sets of (.avi) videos format and has a resolution of (1920×1080) pixels. As a first step before the OF computation, the input video is converted to a sequence of frames. Then, each frame is cut to quadratic size since the subjects appear mostly around the image center. Later, each frame is resized to (360×360) pixels, and for lower computation complexity, each frame is converted to a grayscale image as required for the OF computation. Each video sequence is divided into (11) consecutive grayscale frames (F) with equal distance. The dense optical flow

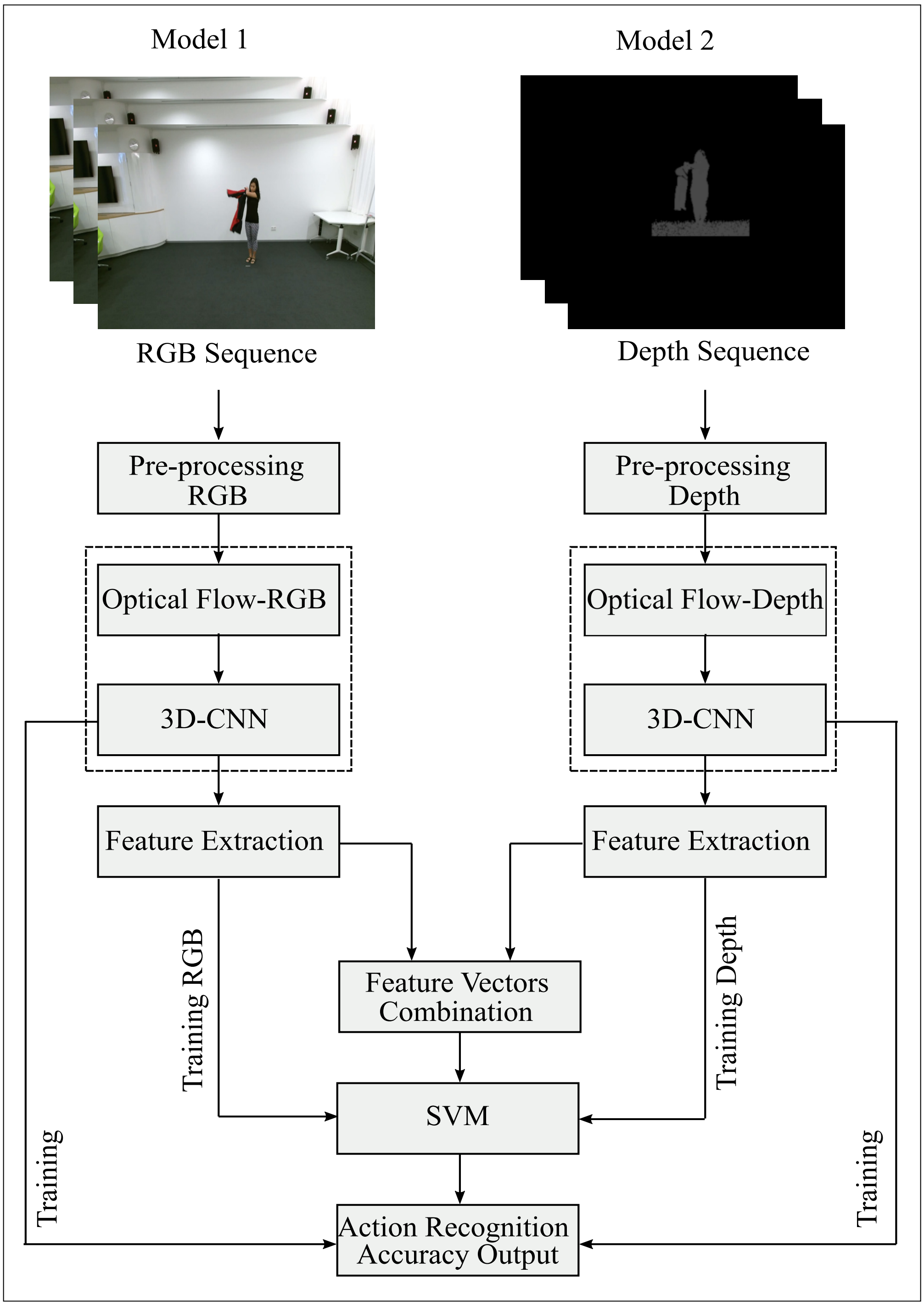


Figure 8.2: General convolutional neural networks system for human action recognition form RGB and depth sequences.

is then computed between each pair of consecutive frames $(F, F+1)$, resulting in (10) dense optical flow images. The output is a singed float, two channels image storing the horizontal d_x and vertical d_y displacement of each pixel location. Maximum and Minimum values over all frames are used for image normalization. Figure 8.3 shows a sample of the OF volume and it is corresponding RGB frames.

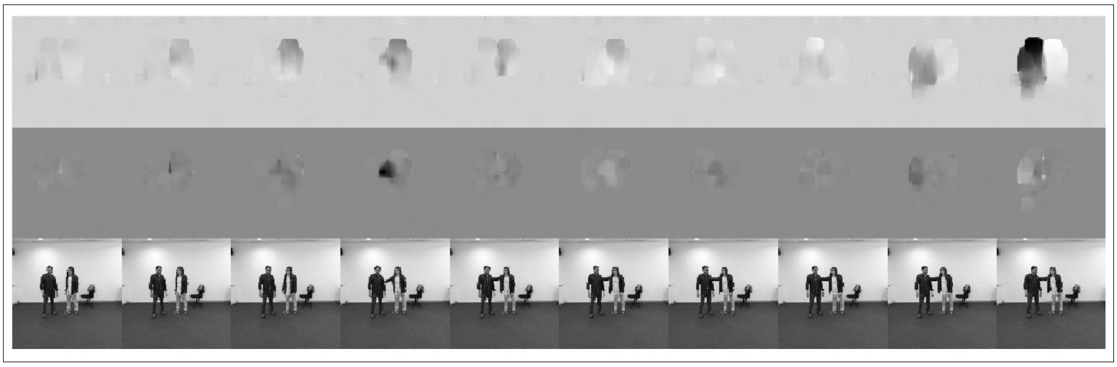


Figure 8.3: Sample OF volume and corresponding RGB frames. From top: Optical flow displacement d_x, d_y, original grayscale frame.

The two-channel images are further resized to (64×64) pixels and then split into vertical and horizontal components. These components are then stored in a sequence of image vectors and saved in two separate files. Each line corresponds to the (10) optical flow frames extracted from a single action video is resulting in a vector of length $(40, 960)$, respectively, i.e., two vectors for horizontal and vertical components. The final input to the 3D-CNN is a (20) frames OF-volume, consisting of horizontal and vertical OF images computed from an RGB video.

Depth Data Pre-processing

The original masked depth dataset comes as the sets of individual frames in a (.png) format with the resolution of (512×424) pixels. The individual image values are given in millimeters. The masked depth data is already pre-processed and extract foreground data from it. However, masked depth data still involves challenges. Robust noise can be found in a ground area in all samples, and it can not be easily removed because of occlusion with feet and legs. Lighting conditions or camera parameters could cause this noise. Some actions, such as falling, have significant movement in this area, so the lower image part cannot be simply cropped. Another issue, especially in the context of 3D-CNN, is the straight cut lines in the noise area since the network weights could adapt to these features. Other samples show depth values from objects and surroundings are not necessary but missed in the masking process, as shown in Figure 8.4. Due to time constraints, these issues are not concerned.

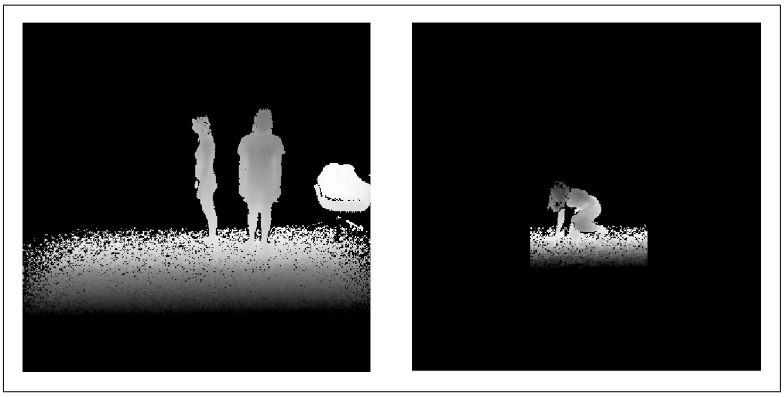


Figure 8.4: Sample frame of action from depth data. Left: Sample frame of action walking apart with noise and unnecessary object, right: Sample of action falling with overlapping of noise and body parts as well as straight cut lines in the noise area.

Each sample comes in a folder associated with sample number, action ID, camera setup, etc. To keep track of the sample order, a shell script is used to sort the samples by their action ID. The number of frames per sample is reduced to (10) frames with equal distance over the whole image sequence. The frames are first cut to resolution (400×400) to reduce unnecessary image space. Also, the quadratic shape can be beneficial for matrix and convolution operations in 3D-CNN training. The image values are then converted from millimeters to the range $[0, 255]$. Additionally, histogram spreading is applied for better visualization. The images are finally resized to (64×64) pixels in order to reduce memory consumption and training time. See Figure 8.5 for the processing stages of a sample.

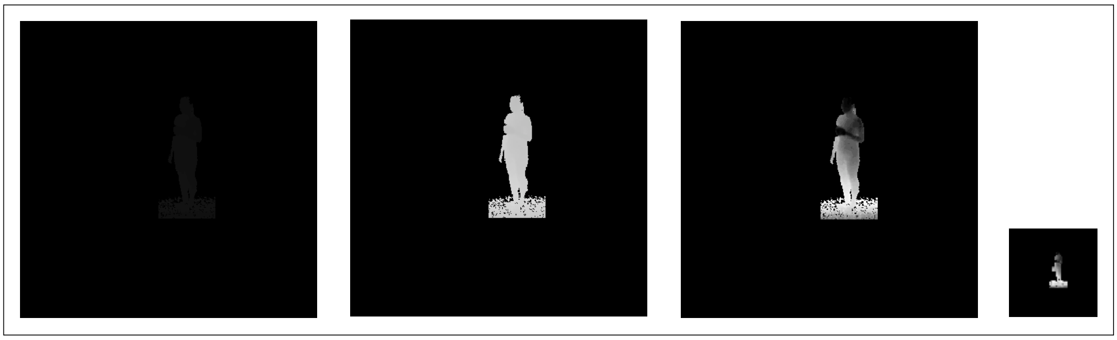


Figure 8.5: Sample frames of action drinking, from left to right: Original depth map, depth map rescaled to range $[0, 255]$, rescaled depth map after histogram spreading, final depth map resized to 64×64 pixels.

All (10) frames of a single action sample are then stored in row-wise order in a vector of size $(40, 960)$.

For storage, a text file and comma-separated value format are used. Each line corresponds to one action sample. Finally, the cross-subject split for training and test data, as well as further splitting into (4) training batches, are applied. The cross-view split is not tested with depth data.

8.2.2 Dense Optical Flow

As mentioned in Chapter 5, 7, the optical flow (OF) is used to capture the temporal motion information from RGB and depth videos. However, in these chapters, the OF was used to extract the hand-crafted features. While in this work, the OF displacement data is also generated to capture temporal motion information from RGB and depth data and then use it to train a 3D-CNN.

Each input video sequence is divided into pairs of consecutive grayscale frames. Then, the dense OF is computed between each pair of consecutive frames (F) and ($F+1$). For the OF computation, the Farnebäck optical flow method is applied (see Chapter 3, Section 3.2.2). The output is two channels image storing the horizontal d_x and vertical d_y displacement of each pixel location (x, y). Figure 8.6 shows a sample OF volume and corresponding RGB frames.

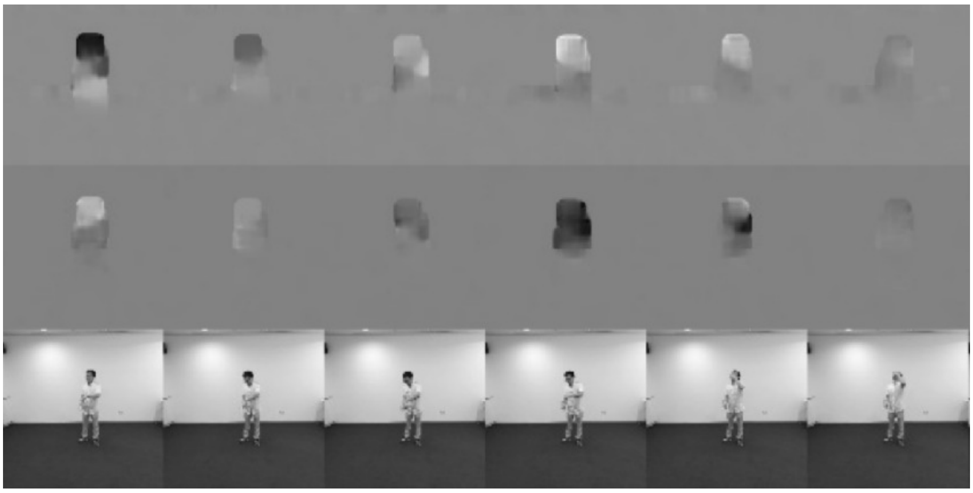
The OF from depth is similar to the OF from RGB. Only the input channel size is reduced to 10 (one channel per frame), and the number of output feature maps for each layer is reduced to one-half.

In this work, RGB and depth maps are processed independently. The two-channel images are further resized to (64×64) pixels and then divided into vertical and horizontal components leading to two feature vectors of 40960 dimensions, indicated as OF-RGB and OF-Depth, respectively. These components are then stored in a sequence of image vectors and finally used as input to the 3D-CNN.

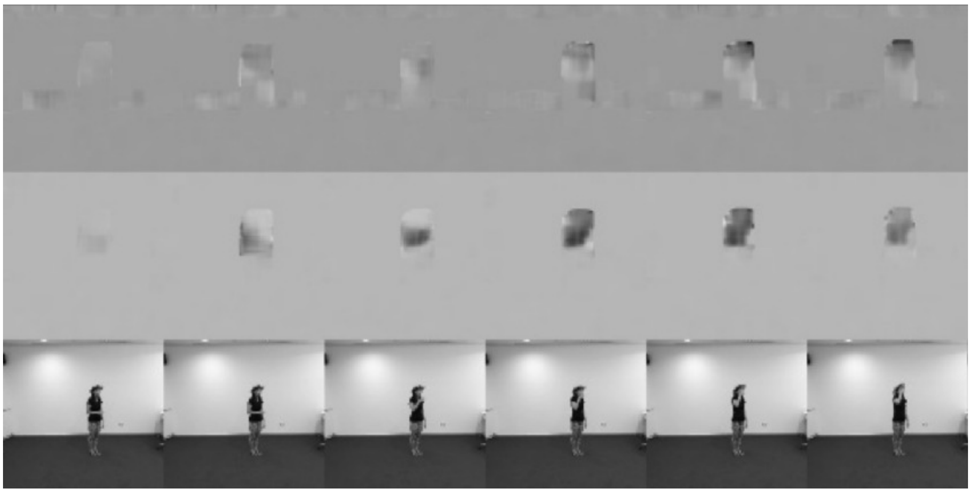
8.2.3 3D Convolutional Neural Networks

A standard CNN consists of two essential components: A feature extractor and a classifier. The feature extractor is used to filter input images into feature maps representing a set of features from the images. These features are represented as a low-dimensional vector and include corners, lines, edges, etc., which are relatively invariant to position shifting or distortions. The output features are then fed into the classifier, which is usually based on traditional artificial neural networks.

In this work, the 3D-CNN task involves tracking the temporal movement information and extracting features from the movement area. Feature extraction from 3D-CNN describes the process of utilizing the network weights and architecture to fit a new problem. For this purpose, data similarity and data size have to be taken into account. This 3D-CNN system can be trained directly from motion prediction in terms of OF for the



Eating



Drinking

Figure 8.6: Dense optical flow example. From top: Optical flow displacement dx, dy, original grayscale frame, these frames are extracted in the middle of optical flow volume and corresponding RGB frames.

human action recognition task. The OF-CNN model from the RGB sequence is represented in Figure 8.7, which consists of (4) convolution layers, each followed by a max-pooling layer and Rectified Linear Units (ReLU) activation. Two fully-connected layers and softmax activation generated the prediction output. These layers and 3D-CNN are explained in detail in Chapter 3, Section 3.6.

In this work, 3D-CNNs' inputs take the OF vectors of a single OF image volume. It treats the individual frames as image channels, resulting in 20 channels as input from

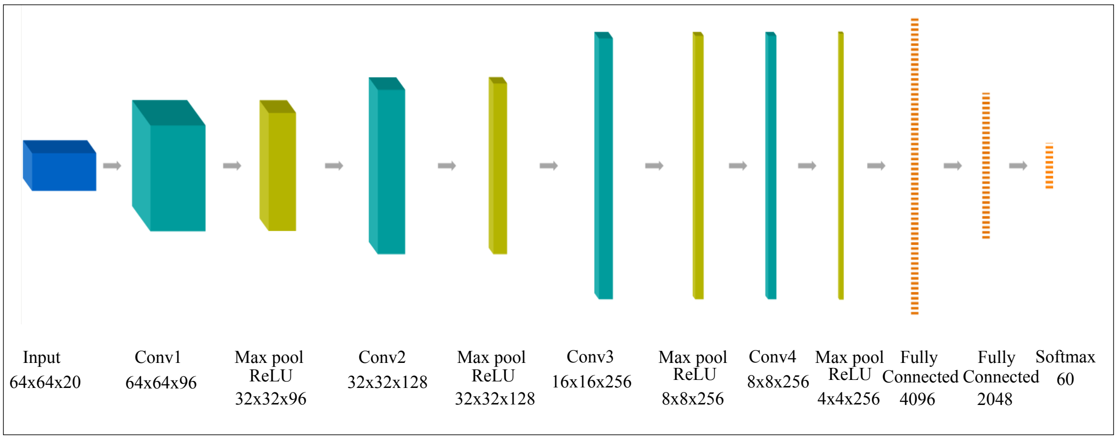


Figure 8.7: Proposed 3D-CNN architecture, transformation of input volume, convolutional, pooling and ReLU layer, and softmax output.

each video sequence; the image width and height is reduced to 64 pixels. The size of convolution filters is adjusted from 5 to 3.

This 3D-CNN system can be trained in two different folds, either evaluating OF-CNN directly from RGB and depth channels or by depending on the feature extracted from OF-CNN of both channels that are finally evaluated by using a multi-class SVM classifier (see Chapter 3, Section 3.5.1).

8.2.4 Feature Extraction from 3D-CNN

In this work, feature extraction using 3D-CNN is achieved from the output of layer 9 (convolutional), 10 (max pooling), and 12 (fully connected) with a feature vector length of 16384, 4096, and 2048 for each sample. Based on the findings of the extracted feature vectors of the OF-RGB-CNN model, the output from the first convolutional layer of the OF-CNN-Depth model is extracted. Then, the multi-class SVM classifier is used for training and evaluation of each model separately.

To explore another common method, which can yield further accuracy improvement, the multi-class SVM is trained by combining feature vectors extracted from the OF-RGB-CNN and OF-Depth-CNN models. The larger feature vectors are expected to deliver improved performance. Thus, the feature vectors are joined by concatenation to form a single feature vector of dimension 3072.

8.3 Experiments and Results

To test the proposed method and because 3D-CNN requires large datasets for training and testing purposes, the NTU RGB+D dataset (see Chapter 2, Section 2.7.5) is used in order to provide reliable results and the accuracy of the system. In this work, only RGB video data (136 GB) and masked depth maps (83 GB) are considered.

Two different train-test splits for the NTU RGB+D dataset are proposed. A cross-subject split divides the dataset into two groups, each containing 20 distinct subjects with $40,320$ training and $16,560$ test samples, respectively 71% and 29%. The cross-view split utilizes the different camera views for each action. The training set contains $37,920$ samples with front and side views of the action, and the test set hold $18,960$ samples with a (45) degree view. Due to time constraints, the OF-RGB-CNN is trained and evaluated for both dataset splits, i.e., cross-view and cross-subject split, while the OF-Depth-CNN and SVM classification is only considered for the cross-subject split.

8.3.1 Implementation Parameters

To determine the network parameters, a random train-test dataset split was trained on multiple parameter configurations. The cross-entropy-multi-class loss function and optimization function [BZK18] showed the best accuracy and speed; therefore, they are used for training. The convolutional layers are added gradually while the accuracy is evaluated. The number of feature maps for the convolutional layers was also increased in an iterative way until the test network reached reasonable accuracy.

The 3D-CNN model, as shown in Figure 8.7, consists of 4 convolution layers, each followed by a max-pooling layer and ReLU activation. Two fully-connected layers and softmax activation generate the prediction output. The image width and height are reduced to 64 pixels, and the size of convolution filters is adjusted from 5 to 3.

8.3.2 Implementation details using RGB and Depth

This section shows different details of experimentation using RGB and depth data separately.

Experimentation with RGB

The networks are trained to classify all (60) classes of the NTU RGB+D dataset. Since a single OF image volume consists of 81920 float values and a CPU based CNN architecture is used, an additional split into (8) training batches and (4) test batches are applied to lower memory consumption. For each batch, a mini-batch of (10) is used. Then, all (8) batches sum up to a single epoch. On an Intel Pentium G4600 at 3.6GHz and 8GB RAM training, a single batch of the NTU-60 class model takes the 1650s on average, respectively 44 hours (without data loading). For each epoch training and testing, accuracy is computed to observe the learning process and identify potential over-fitting. However, the network weights start to explode (drop to zero) between epochs (8 and

9), and the training is stopped. This error is caused by a division by zero when the cross-entropy-loss is computed in the Tiny-Dnn library.

Cross-View Evaluation NTU-60 The held-out test data is also split into (4) batches to lower memory consumption. Since the evaluation process per batch takes as much time as the training process, a full evaluation is only done for the epoch (1, 5, and 9). The overall accuracy is computed by averaging per batch accuracy. Figure 8.10 shows that the test accuracy starts to stagnate around the epoch (5), while train accuracy rises further. The best average train accuracy of 40.07% is found at the epoch (9). However, individual classes performed differently. The best class accuracy is reached for class 59 and class 60 (walking towards and apart from each other) with 97% and 95%. Another pair of different actions with dependable accuracy (68%, 66%) is class 15 and 16 (wear and take off the jacket). The lowest accuracy of 8% is reached for class 12 (writing). The top 10 classes can be seen in Table 8.1.

97%	walking towards	70%	hugging other person
95%	walking apart	69%	shake head
76%	hopping (one foot jumping)	68%	wear jacket
75%	pushing other person	67%	drop
73%	jump up	67%	cheer up
75.7% top 10 average			

Table 8.1: Top 10 accuracy classes NTU-60 cross-view evaluation.

Cross-Subject Evaluation NTU-10 The NTU-10 with the RGB-CNN model was initially used to determine parameters and the model to train the NTU-60. The NTU-10 uses only its first ten classes. The training for the NTU-10 model was stopped after (16) epochs since the evaluated accuracy showed no further improvement, see Figure 8.8.

The best accuracy of 62.97% is found at epoch (13) and about 22% higher than the NTU-60 model. The confusion matrix (see Figure 8.9) shows the misclassification for similar classes (1 to 4), i.e., drinking water, eating meals/snacks, brushing teeth, and brushing hair. However, it also shows its capability to distinguish different classes (8 and 9), i.e., sitting down, standing up, with mutual misclassification of 3% and 1%, respectively, and the ability to model temporal information.

Cross-Subject Evaluation NTU-60 The cross-subject RGB-CNN was trained over (12) epochs, i.e., (3) epochs more compared to the cross-evaluation. However, the best average accuracy of 40.67% is only slightly better, and it was also found at epoch (9) (see Figure 8.10).

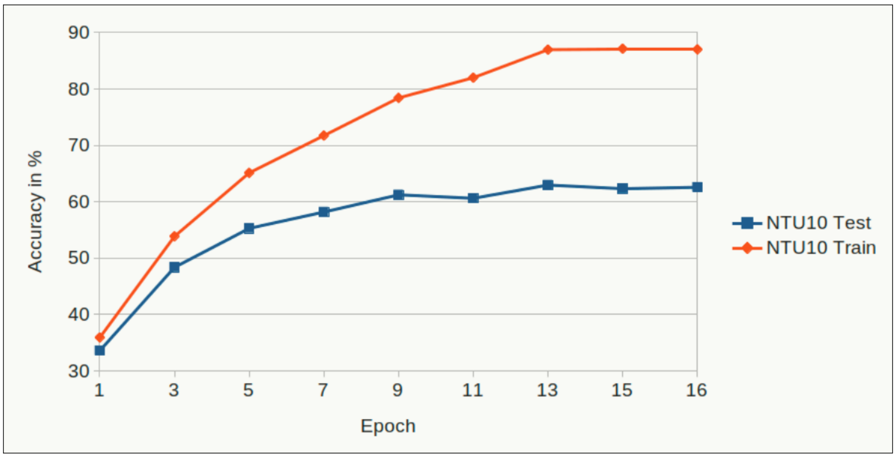
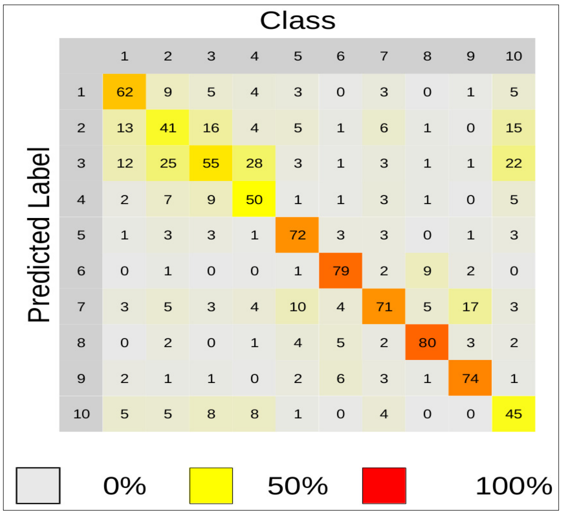


Figure 8.8: Test and train accuracy of NTU-10 model over 16 epochs.


Class
Predicted Label
☐ 0%
🟨 50%
🟥 100%

	1	2	3	4	5	6	7	8	9	10
1	62	9	5	4	3	0	3	0	1	5
2	13	41	16	4	5	1	6	1	0	15
3	12	25	55	28	3	1	3	1	1	22
4	2	7	9	50	1	1	3	1	0	5
5	1	3	3	1	72	3	3	0	1	3
6	0	1	0	0	1	79	2	9	2	0
7	3	5	3	4	10	4	71	5	17	3
8	0	2	0	1	4	5	2	80	3	2
9	2	1	1	0	2	6	3	1	74	1
10	5	5	8	8	1	0	4	0	0	45

Figure 8.9: Confusion matrix of test data evaluation for NTU-10 cross-subject, Accuracy: Light gray = 0%, bright yellow = 50% and bright red = 100%.

Similarly, class 59 and class 60 showed the best class accuracy of 91%. The lowest accuracy of 7% was reached for class 10 (clapping). The top 10 classes have only changed a bit, as shown in Table 8.2.

Experimentation with Depth

For the training process, the same parameters for the OF from RGB training are used. Data is loaded and converted to float values in the range $[-1, 1]$ as input for a 3D-CNN using the Tiny-Dnn framework. The training process is done over (12) epochs and takes about 6.7 hours per epoch, i.e., 80 hours for all epochs in total. Surprisingly, the Depth-

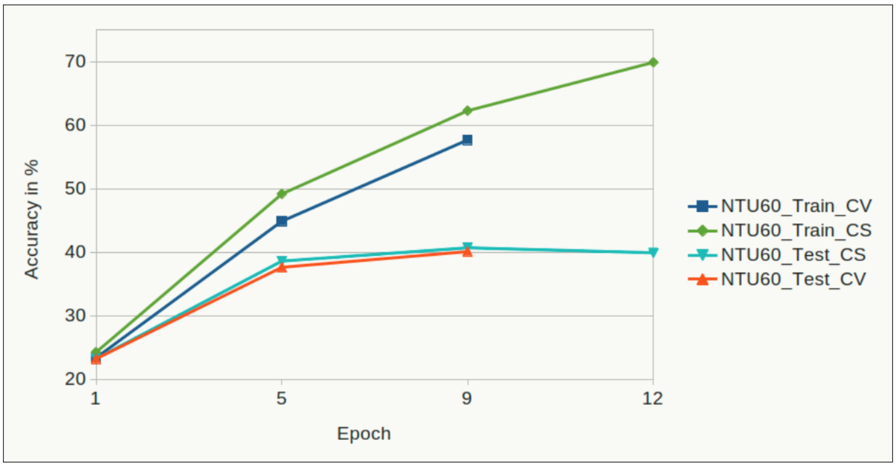


Figure 8.10: Test and train accuracy of NTU-60 CNN for Epoch 1, 5 and 9.

91%	walking towards	77%	pushing other person
91%	walking apart	69%	wear jacket
81%	hopping (one foot jumping)	68%	falling
80%	hugging other person	66%	drop
77%	jump up	63%	giving something to other person
76.3% top 10 average			

Table 8.2: Top 10 accuracy classes NTU-60 cross-subject evaluation.

CNN training takes nearly twice as much time as the RGB-CNN despite input size and parameter reduction. The training was stopped after (12) epochs since the test accuracy remained static, while the distance to training accuracy further increased.

Cross-Subject Evaluation NTU-60 The evaluation of the NTU-60 cross-subject split with the Depth-CNN model was shown a slightly improved average accuracy of 46.9% indifference to 40% of the RGB-CNN model. Especially classes 50 to 60, which involve two subjects, could profit from the depth data representation. Their average accuracy increased by 13% to 72%. In addition, the average top 10 accuracy improves by about 7%, as shown in Table 8.3. Results for other classes remained similar, while classes that achieved decent accuracy in the RGB-CNN model could further improve.

8.3.3 Train-Test using SVM Classifier

This section shows the experimental results when the 3D-CNN is used as a feature extractor. The extracted features are tested with NTU10 from the RGB dataset, i.e.,

98%	walking towards	80%	sitting down
92%	stand up	79%	take off jacket
92%	walking apart	74%	wear jacket
87%	hugging other person	74%	nod head/bow
84%	jump up	74%	shake head
83.4% top 10 average			

Table 8.3: Top 10 accuracy classes NTU-60 cross-subject evaluation.

ten classes from RGB dataset, and also are tested with NTU60 from RGB and depth datasets, as shown below:

SVM Classification of 3D-CNN Feature from NTU10

A multi-class SVM with RBF Kernel (see Chapter 3, Section 3.5.1) is used as a classifier to the extracted features. The NTU10 from the RGB-CNN model is used to explore its possibilities as a fixed feature extractor to review classification improvements. For this purpose, feature vectors are extracted from the output of convolutional, max pooling, and fully connected layers with a feature vector length of 16384, 4096, and 2048 for each sample.

The parameters are found using grid search and cross-evaluation. The SVM cross-subject test and train split are used for training and evaluation. Figure 8.11 shows that the SVM classification accuracy rises with layer depth and slightly outperforms the NTU-10 RGB-CNN with 66.27% accuracy, while earlier layers perform even worse.

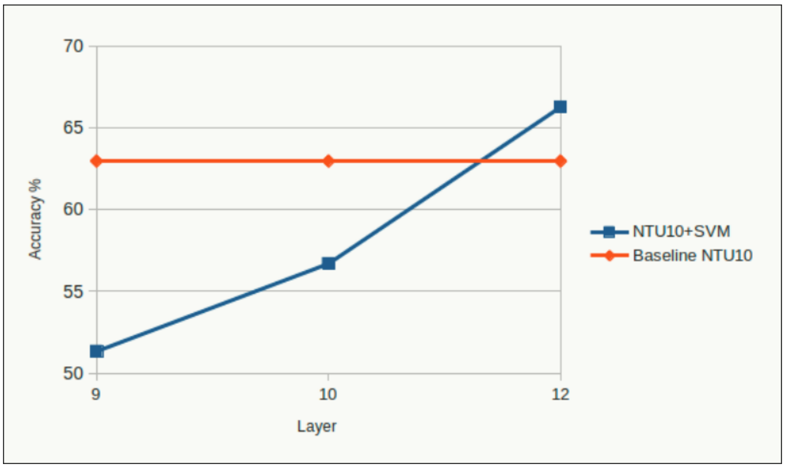


Figure 8.11: SVM classification accuracy for feature vectors extracted from different layers of the NTU10 CNN, convolutional (9), max pooling (10) and fully connected (12) layer.

SVM classification of 3D-CNN Features from OF-RGB and OF-Depth

Since SVM classification with feature vectors extracted from the NTU-10 model showed a small accuracy improvement, and the same method is applied to the NTU-60 of both models, i.e., OF-RGB-CNN and OF-Depth-CNN. Based on the extracted feature vectors of the NTU-10 model, the output from the first convolutional layer of the NTU-60 is extracted. An SVM is constructed by using a grid search to find the best parameters. This construction is done using the OpenCV library. For OF-RGB-CNN-SVM, feature vectors of length 2048 of the training set are used for training, while the testing set is used for evaluation. Again, accuracy could be slightly improved by 4% to 44% average accuracy.

The same process is applied to the OF-Depth-CNN-SVM model. Due to the input vector size reduction, the output feature vector size of the first fully connected layer is reduced to length 1024 compared to the OF-RGB-CNN model. Similar to previous observations on the OF-RGB-CNN, the accuracy could be increased from 46.9% to 50.2% compared to the OF-Depth-CNN only.

2-Model-SVM with Feature Fusion

In order to explore another common method that can yield further accuracy improvement, a 2-Model-SVM with feature fusion is trained to combine feature vectors extracted from the RGB and depth of the 3D-CNN model. The more significant feature vectors are expected to deliver improved performance. For this purpose, the feature vectors are concatenated to form a single feature vector of dimension 3072. Feature vectors are not further processed, i.e., normalized, rescaled, etc. SVM setup for training and testing procedures, as well as parameters search, are kept the same.

The results of the 2-Model-SVM show a significant accuracy increase, with a 65% average accuracy over the OF-Depth-CNN-SVM and OF-RGB-CNN-SVM models (as shown later in Table 8.5). Classes that already reached decent accuracy in previous attempts could slightly profit from the feature combination, while other low accuracy classes benefited even more. Class 1 reached 31% and 32% accuracy in the CNN approaches and doubled to 60% accuracy for the 2-Model-SVM as an example. The top 10 accuracy classes remained mostly the same, and their average accuracy increased to 94.9%, see Table 8.4.

The lowest accuracy of 34% was reached for classes writing and touch the neck. The highest misclassification of 29% is found for the action clapping, which is confused with the action rub hands. Higher misclassification can also be found for different actions such as wear and take off a shoe.

99%	standing up	96%	wear jacket
99%	walking towards	93%	jump up
98%	falling	92%	sitting down
98%	hugging other person	98%	hoping (one foot jumping)
97%	walking apart	88%	take off jacket
94.9% top 10 average			

Table 8.4: Top 10 accuracy classes NTU-60 cross-subject evaluation.

8.3.4 Illustration and Comparison Results of the 3D-CNN System

The experimental of the 3D-CNN system is illustrated in the following three case study steps:

- Optical flow is computed from RGB video data by reducing a single video sequence to (10) OF images (OF-volume), this resulting in (20) channel as input from each video sequence from vertical and horizontal components (d_x, d_y), the image width and height is reduced to 64 pixels. These OF volumes hold temporal motion information fed to the CNN model for feature extraction and training.

- Optical flow is computed by using depth data. In contrast to the OF-RGB-CNN approach, a depth data volume utilizes (10) frames with an equal distance extracted from the full sequence of depth images. Further, the OF-Depth-CNN is trained and evaluated similarly to the previous RGB-CNN.

- Both 3D-CNN models (OF-RGB-Depth-CNN) are used to explore the possibilities of feature extraction combined with the SVM classifier. For this purpose, each CNN serves as a fixed feature extractor. The evaluation of this classification method is then done separately for each CNN as well as for a combined model combining feature vectors of both CNN. The multi-class SVM with RBF Kernel is used, and the setup parameters search for training and test procedure is C to 10 and γ to $1e^{-05}$.

The comparison results are presented in Table 8.5, which is shown the comparison results of the previous experimental steps based on three different models from different input channels (RGB, depth, RGB-D). These results demonstrated that a 3D-CNN with low prediction accuracy could give feature values that yield better classification results with SVM.

Comparison against state-of-the-are Table 8.6 shows the comparison results with the other state-of-the-art methods using the same datasets. This comparison showed

Model	**Modality**	**Accuracy**
OF-RGB-CNN	RGB	40.6%
OF-RGB-CNN-SVM	RGB	**44**%
OF-Depth-CNN	Depth	46.9%
OF-Depth-CNN-SVM	Depth	**50.2**%
OF-RGB-Depth-CNN-SVM	RGB+Depth	**65**%

Table 8.5: Accuracy comparison results of 3D-CNN model and SVM classifier of the extracted 3D-CNN features using RGB, depth, and RGB-D sequences.

Methods	**Modality**	**Accuracy**
HOG^2 [SNGW16]	Depth	32.24%
Super Normal Vector [SNGW16]	Depth	31.82%
HON4D [SNGW16]	Depth	30.56%
DM3DCNN (feature-based) [DKBK20]	Depth	44.47%
DM3DCNN (fine tuning) [DKBK20]	Depth	54.37%
LSTM Encoder-Decoder [LPH$^+$17]	RGB	56%
Proposed Method		
OF-RGB-Depth-CNN- SVM	**RGB+Depth**	**65**%

Table 8.6: Comparison with the state-of-the-art methods on NTU-RGBD cross-subject split dataset.

that RGB and depth data outperform the accuracy results compared to other methods used RGB and depth data separately.

As a summary, the 3D-CNN from OF has proven that it can model temporal information for opposite class pairs, e.g., walk towards and apart from each other, with over 90% classification accuracy for both data splits. The same result can be seen for the depth based CNN. Also, similar actions such as jump up and hopping can be found in the top 10 performing classes. The NTU10 CNN was trained only in 10 classes, confirms this capability.

The reason for these misclassifications is probably caused by the small image size (64x64) and also due to dense optical flow. The blurred optical representation for two samples of class drinking and eating can be seen in Figure 8.6. The horizontal displacement (first image row) is also affected by the transition from the ground to a wall and causes noise in this area. Disturbing noise can also be found in the bottom area of the depth images, as well as objects, e.g., chairs, which are not associated with the action.

Chapter 9

Conclusion and Future Work

This thesis has presented and evaluated several novel methods for human action recognition in real video data. It also has demonstrated that the proposed methods outperform the state-of-the-art on various and challenging datasets. The thesis aims to develop methods for detecting persons in the scene and recognizing the actions/activities of the persons. These methods have been studied from four perspectives: RGB, grayscale, depth, and RGB-D images/videos.

To conclude the thesis work, the key contributions and results are summarized from all novel research work experiments in Section 9.1. The summary of the evaluated results is presented in 9.2. The interesting directions for future research in this field are indicated in Section 9.3.

9.1 Key Contributions

The major components of an automatic action/activity recognition system include data sources, feature extraction, feature representation, and classification in different RGB-D videos for action recognition. In this thesis, two different main approaches were proposed to improve human action recognition: Hand-crafted and deep neural network approaches.

- Hand-crated approaches were represented by the local and global feature representation methods, in addition to a combination of these feature representation methods. These features were aggregated using a Bag-of-Word (BoW) pipeline. Then, machine learning algorithms represented by classification methods were used to demonstrate BoW robustness and reliability for the action recognition task.

- Deep neural network approaches have become state-of-the-art for action recognition. In this thesis, the 3D-CNN approach was used to show how 3D-CNN can model the temporal information in videos and perform better when using the RGB-D sequences. However, the 3D-CNN could be challenged with longer and semantically richer complex videos, requiring more and more data and computational time to perform better than the hand-crafted feature methods.

The proposed methods of this thesis are concluded from two different parts of approaches that are used to improve human action recognition. The first part of contributions depends on the hand-crafted features and is concluded as follows:

In Chapter 4, a new system of human action recognition on 3D sensor data was proposed. This system started from processing, removing the noise from the input depth data, and aligning the RGB with the depth frames. Extraction of local and global feature vectors was proposed. The local feature vectors were represented by extracting these features from 3D sensor data using Speeded-Up Robust Features (SURF), Motion History Image (MHI), and optical flow (OF) for detecting motion interest points. The HOG descriptor was applied to images and MHI-OF from RGB and depth video channels to represent the appearance and motion features of all actions. The global features were extracted using a global Hu-moments shape descriptor from MHI. Finally, local and global vectors were combined into one vector for each RGB-D video action. These feature vector values were tested depending on the BoW pipeline using k-means clustering and One-vs-All multi-class classifiers. The presented approach is highly efficient and invariant to cluttered backgrounds, illumination changes, rotation, translation, and scale. The experiment results showed that the proposed system could productively recognize different actions even when they look similar, such as sitting down and standing up.
To test the proposed features extracted from these different actions and to compute the performance accuracy of the system, two machine learning classifiers were used, SVM and KNN.

- The SVM classifier was applied to local features extracted from the HOG descriptor on different spatio-temporal image channels (RGB and RGB-D). The recognition accuracy reached 57.65% on the MSR3D dataset and 49.04% on the ORGBD dataset when testing with the RGB channel only. While the accuracy reached 91.11% on the MSR3D dataset and 92.86% on the ORGBD dataset when using RGB and depth channels, this means that the depth channel added more information to the RGB channel that helped to increase the recognition performance.

- The KNN classifier was calculated from the combined vectors of local and global features that are computed from both RGB and depth video channels. The testing results from this combination were 100% on the MSR3D dataset and 85.71% on the ORGBD dataset.

The experimental results on the RGB-D dataset demonstrated that the proposed approach significantly outperforms the existing state-of-the-art methods that used the same datasets and different feature extraction methods.

In Chapter 5, a novel action recognition method was proposed. This method was represented by combining the 3D Trajectory (3DTr), Motion Boundary Histogram (MBH), and global GIST feature (3DTrMBGG) using the BoW pipeline. This work also improved the benefit of combining a set of local feature vectors with a single global feature vector in a suitable manner. The benefits of combining the features are to keep the main global attributes of human action and to reduce the impact of occlusion and noise.
The proposed approach included three steps listed as follows:

1. Starting from the extraction of local features by extending the 2D dense Trajectory method on RGB videos to 3D Trajectory on RGB-D videos.

2. Computed global features from depth frames into two steps: First, by applying background subtraction to find the important motion information, called Action-Region (A_R) in this work, and then computed global GIST feature vectors from each A_R in the sequence.

3. Combined the extracted local and global features to encode video information.

In the training step, features extracted from the training set were clustered using k-means to generate BoW, and the histograms based on occurrences of bag words in the training set were used as features to train classifiers. After the feature vectors were created, a multi-class SVM classifier was applied to achieve action recognition.
The evaluations using two challenging realistic scenario action datasets, such as MSR3D, and ORGBD datasets, demonstrated that the proposed method could recognize various actions in a large variety of RGB-D videos. The experiments and results were computed from these datasets in three different proposed models: The first model, local descriptor (3DTrMB), was extracted from RGB-D. The results achieved in this model are 92.0% and 81.52% from both datasets. The second model was a global descriptor (3DGIST). In this model, the feature vectors were extracted from depth data, and the results reached 93.33% and 83.81%, respectively. The third model is the local and global feature descriptors combination. The feature vectors were extracted by combining 3DTrMB with the 3DGIST feature values in one vector to form (3DTrMBGG) for each video action of RGB-D data. The accuracy results were 95.62% and 97.62% on the tested datasets, respectively.
The experimental results illustrated that the comparison between these three model results showed that local and global combinations gave the best accuracy value than the results from using local or global descriptors separately. These results also showed that the proposed system could effectively recognize the different activities with high movement rates like walking, cleaning, etc., and improves performance on actions with low

movement rates like reading, using a laptop, etc.

In Chapter 6, a new method for human action recognition was represented based on the Retina model and Local Binary Pattern (LBP) descriptor. The main idea of this work is to capture spatio-temporal relation of moving objects by depending on the spatio-temporal filtering Retina model for detecting saliency map (region of interest). Then, Local Binary Pattern (LBP) was applied as the texture feature extractor on the moving object from different Retinal channels to compute a bag of important feature information. The Retina channels were represented by Parvocellular (Parvo), Magnocellular (Magno), and the combination of both channels, Parvo-Magno, from both RGB and depth images. The histogram from LBP was computed from the saliency area of the videos sequence, and then all computed histogram values were combined into BoW vectors. The BoW algorithm encodes all the descriptors derived from each video into a single code. These feature values were tested using the Random Forest (RF) classification method. The proposed system was tested on three different public RGB-D datasets and achieved superior performance compared with state-of-the-art approaches. These datasets are MSR3D, and ORGBD, and G3D datasets.
Different accuracy values were achieved when tested the various channels of Retina models and LBP features. The classification accuracy results from the Parvo channel on these three datasets reached to 83.37%, 93.13%, and 89.48%, respectively. While from the Magno channel, the recognition accuracy reached to 90.21%, 96.86%, and 96.86%. However, when these two channels were combined, the results seem to be less performance than the only Magno channel, and it was achieved to 81.11%, 92.86%, and 71.43% by using the same datasets. The reason behind that, because the Magno channel gives intense energy in the detected area (motion area), and the Parvo channel is certainly blurred there since there is a transient event. Then, the temporal domain results were better than the spatio-temporal domains.

In Chapter 7, human activity recognition was proposed on 3D video by formulated a body activity recognition problem as a classification problem. In the proposed work, two goals were presented to improve human activity recognition:

1. A novel spatial-temporal feature descriptor technique was proposed from the RGB-D image sequence, called 3D Optical Flow Gray Level Co-occurrence Matrices (3DOFGLCM). Furthermore, the Haralick features were extracted from the co-occurrence matrices of optical flow fields. These features are responsible for measuring statistical properties such as energy, contrast, homogeneity, entropy, sum average, and correlation.

2. This technique presented a detailed comparison of the five popular classifiers algorithm to prove the performance of the proposed system in case of activity recog-

nition from the texture feature vector extracted from RGB-D images. These classifiers were represented by Artificial Neural Network (ANN), Naive Bayes (NB), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), and Random Forest (RF).

The experimental results of the proposed technique demonstrated that the RF classifier gave good accuracy results and outperformed other classifier models used on the same feature values. This system was tested on four different public activity datasets, G3D, CAD-60, MSR3D, and ORGBD datasets. The accuracy performance from RF reached 88.40%, 95.45%, 80.67%, and 83.01%, respectively, because this classifier contained a bunch of combined decision trees that can handle certain features very well and can also work with the high dimensional spaces and a large number of training examples. The accuracy results from other classifiers were represented as ANN classifier gave 68.30%, 69.23%, 56.16%, NB results were 65.42%, 76.92%, 63.17%, and 67.88%, SVM was achieved 77.08%, 81.82%, 79.51%, and 80.96%, and from KNN classifier got 82.41%, 90.19%, 70.74%, and 71.56% on the same previous four datasets, respectively.

It is important to emphasize in the previous contributions of four chapters that the proposed feature descriptor is only changed in the pipeline. Since the proposed action recognition systems' main goal is to compare the real contribution of the proposed feature descriptor, i.e., for each experiment, the recognition pipeline is the same, and the feature detector and descriptor are switched. Also, the classifier type is changed, sometimes based on one classifier, and in another time depended on two or more classifiers for the accuracy improvements.

The second part of the contributions is represented by using one of the deep neural network methods, i.e., Convolution Neural Network (CNN), in order to improve human action recognition from 3D sensor data.

In Chapter 8, a deep 3D Convolution Neural Network model was presented to classify and recognize human actions based on RGB-D data. This model extract features from temporal dimensions by performing 3D-CNN. A 3D-CNN utilizing optical flow volumes was trained for the task of action recognition on the NTU RGB+D dataset. Further, it was evaluated for two different tests and trains data splits. For comparison, a 3D-CNN based on depth data was trained and evaluated. The 3D-CNN additionally served as a fixed feature extractor. The extracted features from different layers were used for classification using an SVM. Finally, feature vectors of both 3D-CNN models were combined for a 2-Model-SVM classification.

The experimental results on NTU RGB+D datasets demonstrated that the combination of different modalities could give better performance than using each modality individually. The incorporation of RGB and depth modalities to compute 3D-CNN feature vectors and supervised learning for the evaluation yields better prediction accuracy than the original 3D-CNN. In this work, an SVM classifier was used, and the accuracy results values outperform the results from baseline CNN in the individual modalities. The depth based 3D-CNN model showed slightly improved accuracy than used only RGB. The 2-Model-SVM based on feature combination benefits from both feature representations and achieves decent performance.

The 3D-CNN approaches from RGB and depth provided good classification results for specific classes, e.g., 76.3% and 83.4% average accuracy for the top 10 classes of the cross-subject split dataset.

The OF-RGB-CNN had also proven that it is capable of modeling temporal information for opposite class pairs, e.g., walk towards and apart from each other, with over 90% classification accuracy for both data splits, the same result can be seen for the depth based 3D-CNN. Similar actions such as jumping up and hopping can also be found in the top (10) performing classes. The NTU10-CNN, which was trained only on (10) classes, confirms this capability.

Using the NTU10 model as a fixed feature extractor for SVM classification has also shown a small accuracy improvement of 3.3% for the output extracted from the first fully connected layer, compared to the baseline 3D-CNN. However, the feature vectors obtained from earlier layers performed worse, considering the small feature vector size of 2048 and fast SVM implementation, using a grid search to find optimal parameters. This method proved to be an option to improve 3D-CNN accuracy. The accuracy of both 3D-CNN models could be increased by about 4% using an SVM for feature vector classification, without additional processing. The combined 2-model SVM, i.e., OF-RGB and OF-Depth, showed a 15% improved accuracy, requiring little additional work.

The Tiny-Dnn is a good starting point to neural networks and suitable for small CNN. Switching the deep learning framework to a GPU based library could bring significant improvements. The factor time is a considerable limitation not only for final training but also to evaluate the effects of different parameters, parameter numbers, and additional layers. It is also common to train multiple models for the same dataset to choose the best performing model.

By using another CNN model to extract special image features, e.g., objects such as a toothbrush or food, and combining the OF-CNN features from depth and RGB could be eliminated the weakness of distinguishing similar classes, such as brushing teeth and eating. With better hardware and GPU computation, the image size could also be increased to deliver more robust results.

Adding dropout layers can also help in increasing accuracy and limit the chance of overfitting, as described in [HSK$^+$12]. Finally, data augmentation techniques, such as zoom, rotation, and translation, can help to reduce overfitting while the dataset is synthetically enlarged [PDKS15].

9.2 Summary

In summary, the deep neural network using videos is still an open area of research. The deep neural network methods outperformed the hand-crafted feature extraction methods in the case of images and text recognition systems but not in the cases of video processing tasks. Many computer vision problems are not studied using deep neural networks, such as robotic, 3D modeling, motion estimation, motion capture, and video processing, which cannot be easily implemented in a differentiable manner with the deep neural networks or deep learning. Therefore, these problems need to be solved using other traditional computer vision techniques, i.e., hand-crafted features and traditional machine learning techniques.

To solve these problems and to find efficient learning of spatio-temporal features in video action recognition yet, hybrid approaches combine traditional computer vision and deep learning that could offer the advantages traits of both methodologies [UL19, VMS$^+$19, BP20].

This thesis concludes that deep learning has not replaced traditional computer vision techniques and hence why deep learning should still be studied and taught. The works of this thesis illustrated that action recognition tasks still need hand-crafted features despite the development of the deep learning domain. The problem with deep learning methods, such as CNN, required many labeled videos for training, while most available datasets are relatively small or contain unlabeled videos. Meanwhile, most of the current action recognition approaches focused on deep learning mostly ignore the intrinsic difference between the temporal and spatial domains and consider the temporal components as feature channels when applying CNN architectures to model the video. Recently, CNN has been proposed to be used with RGB and depth images or videos, but it is not yet as successful as in the other area. Furthermore, the research using spatial and temporal 2D or 3D data is not advanced, and it remains an area of further research to use CNN with temporal and RGB-D sensor data.

On the other hand, the hand-crafted feature descriptors do not need extensive training datasets. It is also more straightforward and less ambiguous to understand the actual model with hand-crafted features.

Moreover, the thesis proposed methods show further improvement when used RGB and depth videos, as well as the combination of local and global feature representation methods. Better results than the current state-of-the-art were achieved on the public datasets for action recognition.

9.3 Future Works

This section introduces some of the future research directions which will follow in the future.

Saliency Detection for Local and Global Features In the previous work of Chapter 6, the Retina filter model was used to detect the salient motion area. As a future direction work, the Boolean Map-based Saliency (BMS) model [ZS13, KAD17] will be applied on RGB and depth image channel for extracting the orientation and motion from these images. BMS calculates saliency maps to detect a motion area by analyzing the topological structure of Boolean maps. After that, the salient motion area will be fed to the deep neural network, such as Convolution Neural Network (CNN) or Recurrent Neural Network (RNN), to learn the spatial and temporal behavior of different daily actions for improving human action recognition task.

3D Skeleton Joints for Human Action Recognition based on Convolution Neural Networks In the previous work of Chapter 8, human action recognition was proved based on RGB-D video actions. In future work, invariant characteristics of humans will be extracted from the 3D skeleton joints that are recorded by RGB-D sensors. Since skeleton joints usually have similar skeleton points, these configurations are suitable for different types of actions. This work will analyze a highly informative number of skeleton joints for action recognition using Shannon's entropy mechanism [Sha48] because some joint points are irrelevant and not moving, which appear as noise; this problem could reduce the performance of the action recognition system.

Appendix A

RGB-D Action/Activity Datasets

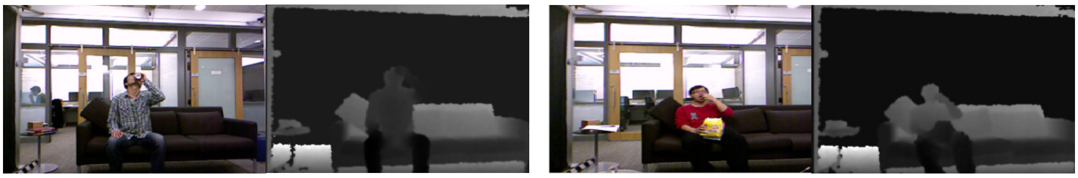
A.1 Examples of Datasets

In this thesis, five types of public datasets were used to test the proposed methods implemented in each chapter. Image examples of all activities of each dataset are reviewing in the following.

A.1.1 MSR Daily Activity 3D Dataset

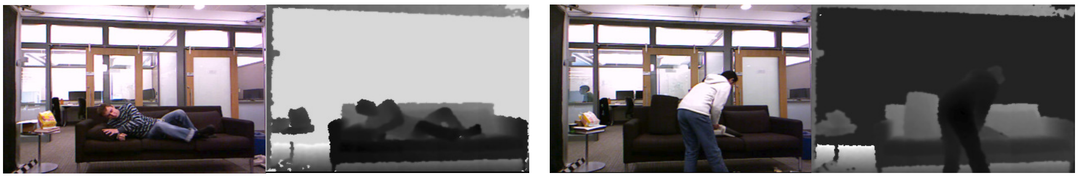
The MSR3D dataset [19] [WLWY12] was collected by Microsoft and Northwestern University in 2012 and focused on daily activities. The motivation was to capture the activities of human-daily in the living room.

MSR3D dataset was collected in an indoor environment with sixteen actions and (10) subjects (persons) as shown in Figure A.1, where each person performs each action twice, one is in sitting and the other in standing up position. The camera was fixed in front of the sofa. In addition to RGB and depth data, skeleton data are also recorded. The skeleton data's joint positions extracted by a tracker are very noisy due to the subjects being either sitting on or standing close to the sofa.

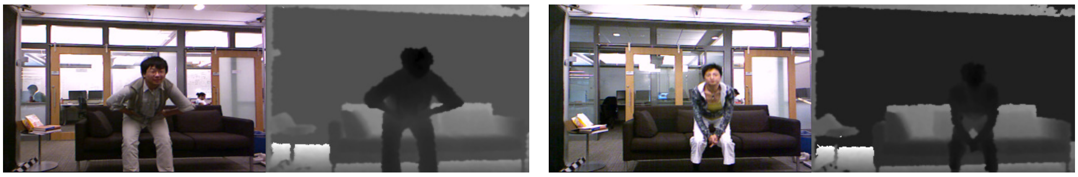


Drinking Eating

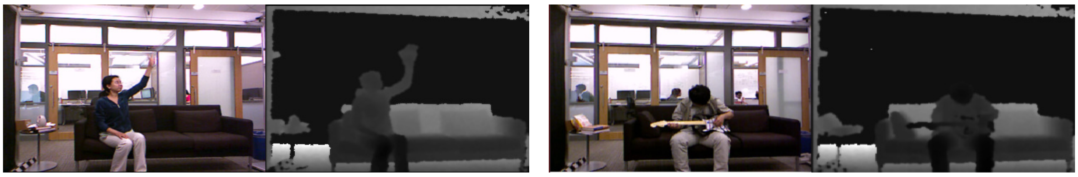
Walking Cheer Up



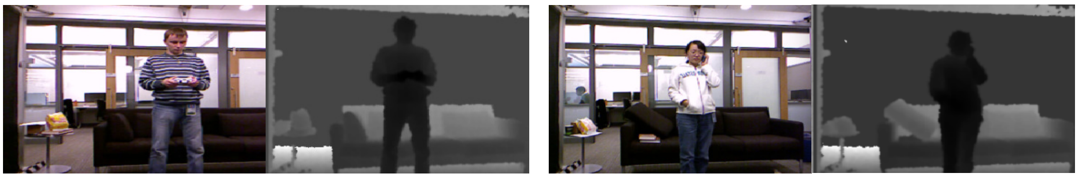
Lie Down Cleaning



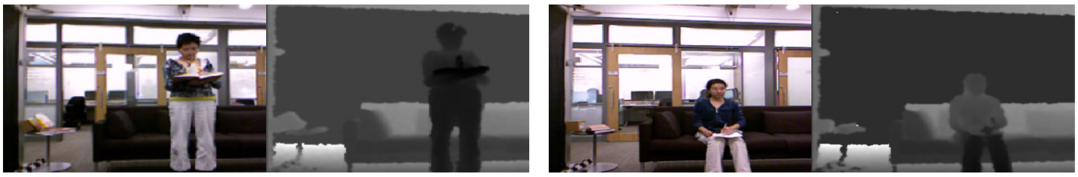
Sit Down Stand Up



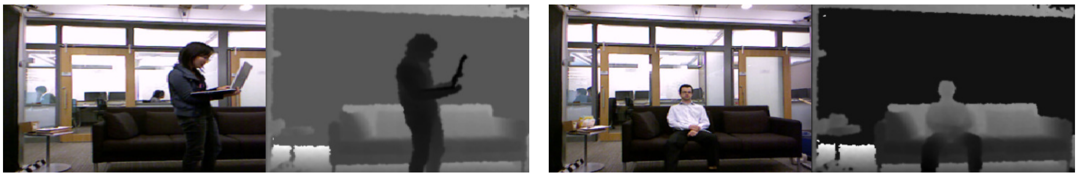
Toss Paper Play Gitar



Play Game Phone Call



Reading Writing



Using Laptop Still

Figure A.1: Sample frames of MSR Daily Activity 3D dataset, compare to [19].

A.1.2 Online RGBD Dataset

The targets of the Online RGBD (ORGBD) action dataset [16] [YLY15] is for human action recognition (human-object interaction) based on RGB-D video data. ORGBD dataset was captured by the Kinect device (RGBD sensor) and designed for three environments tasks: Same-environments, cross-environments, and continuous action recognition. There are seven categories of human actions performed by (16) subjects. Figure A.2 shows different sample frames of these seven actions [AaP18b]. In this thesis, the comparison results are done with the state-of-the-art methods on the same environment test setting, where half of the subjects are used for training data, and the rest of the subjects are used for testing data.

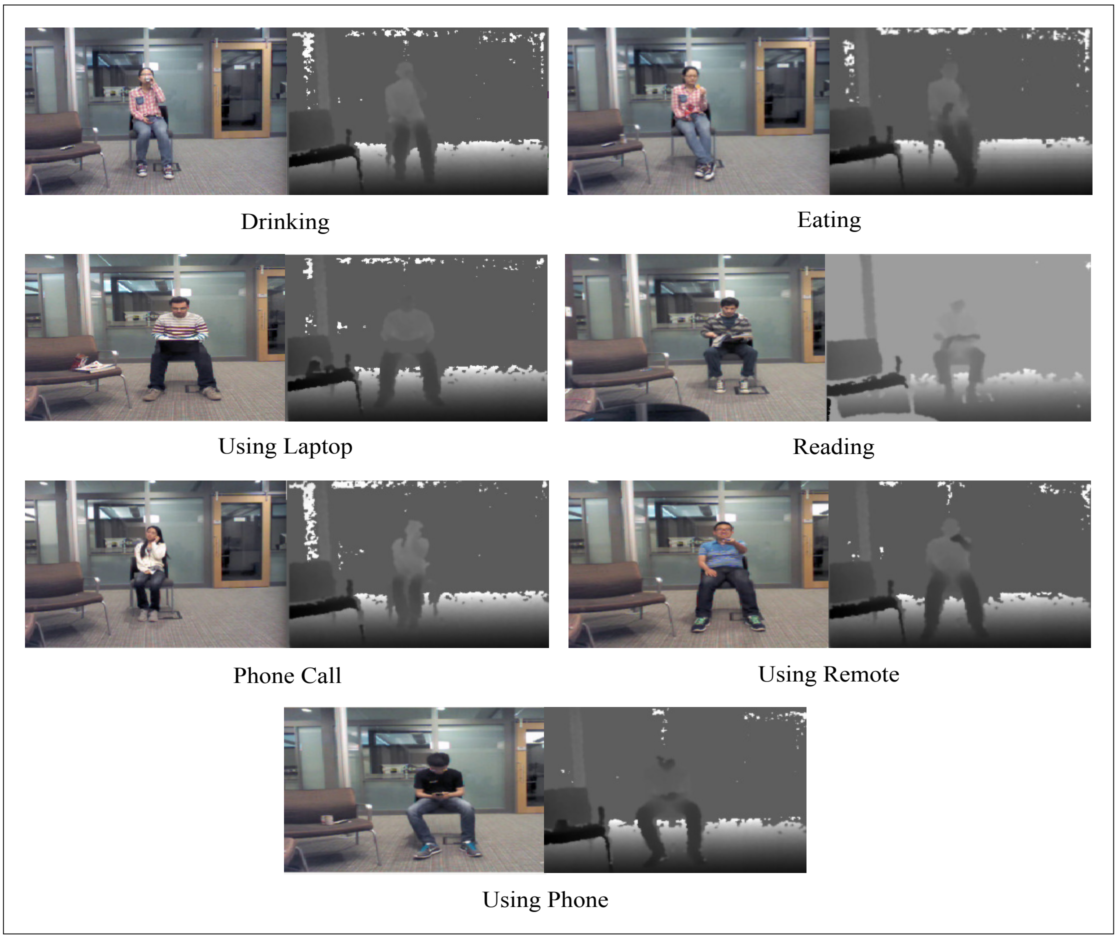

Drinking
Eating
Using Laptop
Reading
Phone Call
Using Remote
Using Phone

Figure A.2: Sample frames of Online RGBD Action Dataset, compare to [16].

A.1.3 Gaming 3D Dataset

Gaming 3D (G3D) dataset [1] [BAM16, BMA12] was captured by Kingston University in 2012, and it is focusing on real-time action recognition in the gaming scenario. G3D is publicly available for researchers to develop new action recognition algorithms for video games and benchmark their performance. G3D is used for real-time action recognition in gaming, containing synchronized video, depth, and skeleton data that has been captured. The three streams were recorded at 30fps in a mirrored view. The (.png) image format was selected for storing both the depth and color images as it is a lossless format. The resolution used to store both the depth and color images were (640×480). The raw depth information contains the depth of each pixel in millimeters and was stored in 16-bit grayscale and the raw color in 24-bit RGB (see Figure A.3). The 16-bits of depth data contains 13 bits for depth data and 3 bits to identify the player. The depth information was also mapped to the color coordinate space and stored in a 16-bit grayscale. Combining the color image with the mapped depth data allows the user to be segmented in the color image.

A.1.4 Cornell Activity Dataset

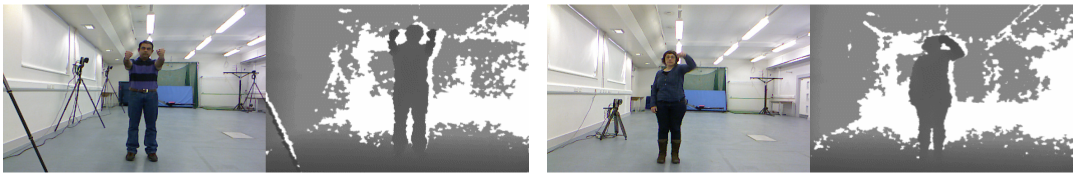
The CAD-60 dataset [17] [SPSS11] was captured by Cornell University in 2011, motivated by the fact that actual daily activities rarely occur in structured environments. Hence, the actions were performed within the uncontrolled background and comprised of RGB and depth video sequences of humans performing activities that are recorded using the Microsoft Kinect sensor. CAD-60 datasets are containing:

- Four subjects (two male, two female) that perform different actions.
- Five different environments (office, bedroom, bathroom, kitchen, and living room).
- Sixty RGB-D videos and twelve different activities.

CAD-60 dataset also contained the skeleton data, but only RGB-D data was used in this thesis research. RGB-D data has a resolution of (240×320). RGB is saved as a three-channel 8-bit (.png) file. Also, depth is saved as a single-channel 16-bit (.png) file. Due to the alignment of depth and RGB data, some pixels on the edges will have a value of 0.

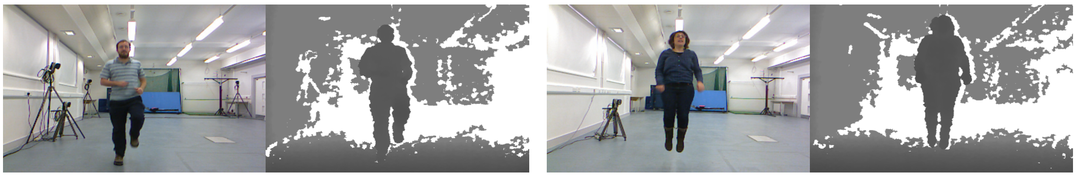
Bowling

Golfing



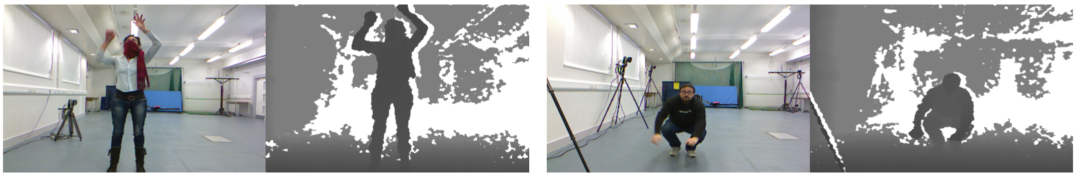
Driving

Fighting



Running

Jumbing



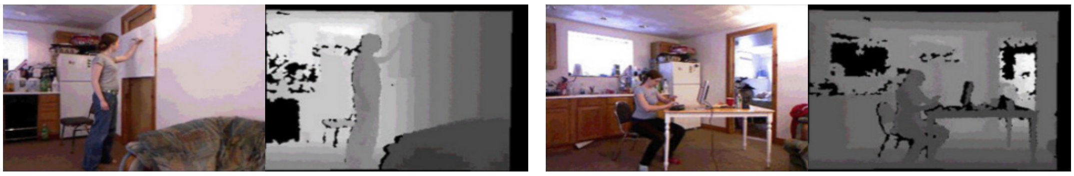
Climbing

Crouching

Figure A.3: Color and depth samples from different gaming actions (G3D), compare to [1].

Brushing Teeth

Cooking (Chopping)



Writing on White-Board

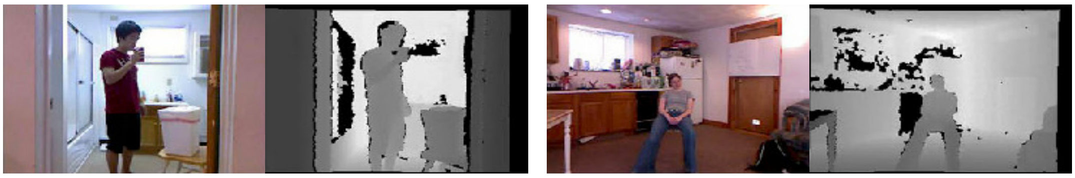
Working on Computer

Wearing Contact Lens

Talking on the Phone

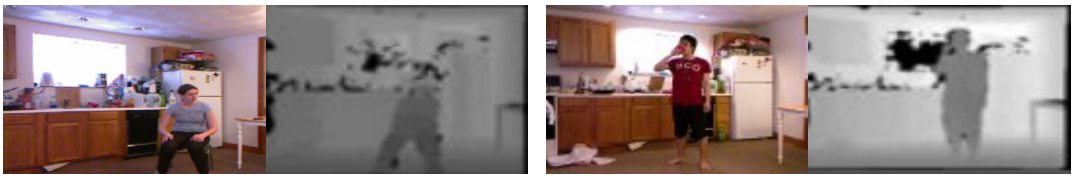
Opening Pill Container

Cooking (Stirring)



Rinsing Mouth with Water

Relaxing on Couch



Talking on Couch

Drinking Water

Figure A.4: Sample frames selected from CAD-60 dataset of different actions, compare to [17].

A.1.5 NTU RGB+D Dataset

NTU RGB+D dataset is used for human action and activity recognition, especially with the deep learning methods because it is a very large-scale dataset. The author of this dataset suggested two different evaluation criteria, including cross-subject and cross-view. Four major data modalities were recorded by Microsoft Kinect v2 sensors: RGB frames, depth maps, 3D joint information, and IR sequences. This dataset was captured by three cameras and contains (56880) sequences (with 4 million frames) of (60) classes performed by (40) subjects.

RGB videos are recorded in the resolution of (1920×1080), while the depth map is sequences of two-dimensional depth values in millimeters. For maintaining all the information, the authors Shahroudy et al. [SNGW16] were applied lossless compression for each frame. The resolution of the depth frames is (512×424) for each image. Joint information (skeleton) consists of 3-dimensional positions of (25) major body joints for detected and tracked human bodies in the scene. Additionally, the Infrared sequences are collected and stored frame by frame in resolution (512×424).

In this thesis, RGB and masked depth map data were used, the main purpose of providing masked depth map, as mention by [18] [SNGW16], was to have a smaller sized version of the original depth maps, and the position of the body skeletons was used to find regions of interest in the depth maps. The depth values were copied for the regions of interest (from the original depth maps) and set the other regions' depth to zero; this helped to achieve a much more efficient frame-wise compression rate.

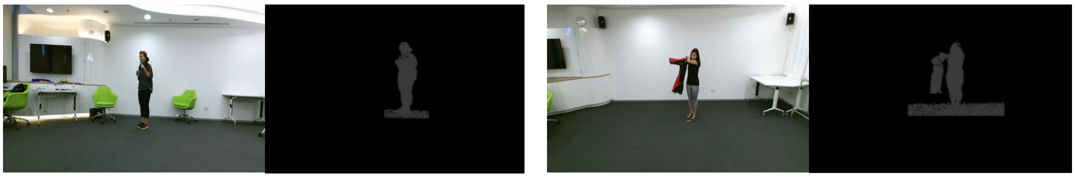
Figure A.5 shows only eight samples from the RGB-D dataset because it is huge data that has (60) classes. The original masked depth data map looks like a black image. Therefore, they were inverted to be clear, as in Equation A.1:

$$I_1 = 255 - I \tag{A.1}$$

where, I is the original black image, and I_1 is the inverted image.

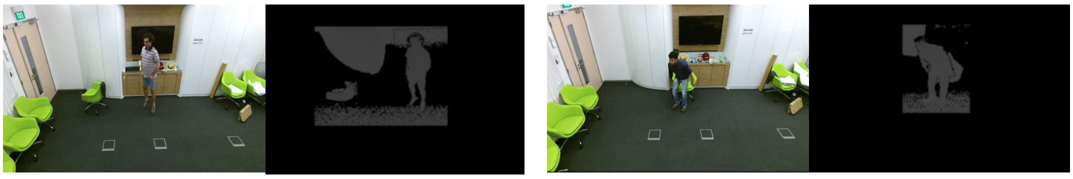
Drinking

Brushing Teeth



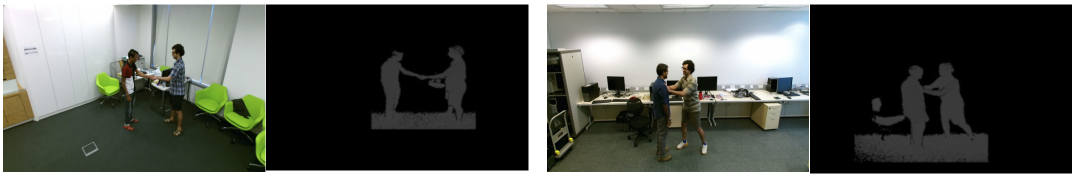
Eating

Wearing Jacket



Jumping Up

Sitting Down



Hand Shaking

Pushing

Figure A.5: NTU RGB-D dataset images example, compare to [18].

Appendix B

Tiny-Dnn Framework

B.1 General

Tiny-Dnn is a deep learning framework available at Github [15]. It is dependency-free and uses the `C++14` standard. The network computations are mainly done on the CPU, making Tiny-Dnn suitable for embedded devices. For running Tiny-Dnn, a `C++14` Compiler is needed, and the header file `tiny_dnn.h` has to be included. The framework supports sequential and graph models and features various layer types (fully connected, convolution, average and max pooling, dropout, etc.) and activation functions such as tanh, softmax, sigmoid, and ReLU. Also, standard loss functions and optimization algorithms are implemented.

B.2 CNN Construction

A network is constructed by defining its type and name, in this case `<sequential>` (see sample below).

```
network<sequential> net;

net << convolutional_layer(64,64,3,20,96,padding::same)
    << max_pooling_layer(64,64,96,2)
    << relu_layer()
    << convolutional_layer(32,32,3,96,128,padding::same)
    ...
    << fully_connected_layer(2048,10)
    << softmax_layer(10);
```

The type `<graph>` allows networks with multiple input layers (branches) and the possibility to merge layers at the determined position. The << operator is then used to add individual layers to the network.

The top `convolutional_layer(64,64,3,20,96,padding::same)` simultaneously serves as the input layer. First, parameters define the input image size `(64x64)`, followed by the size of the convolutional filter `(3x3)` and the number of image channels `(20)`, and output feature maps `(96)`. The padding::`same` is padding the input concerning a stride and filter size of the convolution operation to preserve the same size for the output volume. The output is then fed to the next layer. Pooling layers require the input volume size and the down-sampling factor for the output as a parameter. While the activation function has no parameter, fully connected and softmax layers only require the input respectively output size.

B.3 CNN Training

For training the constructed network, one of the optimization algorithms has to be chosen: Such as stochastic gradient descent, momentum and Nesterov momentum, adagrad, rmsprop, adam, adamax.

```
optimzer adamax;
```

Training and testing data is stored in a `vector<vec_t>` of type double. The image values are converted to the range $[-1.0, 1.0]$ and stored in row-wise order. A single vector holds all image channels consecutively. It is important to shuffle the order of training data; otherwise, the network performs poorly. This network can be done using the `C++` function `random_shuffle()` using the same seed value to shuffle images and labels.

```
vector<vec_t> train_images;
// corresponding labels, integers [0-9] for 10 class IDs
vector<label_t> train_labels;
```

The `net<loss function>.train(..)` statement defines the loss function and initiates the training process. Parameter `minibatch` is used to split the training data further into batches used for calculation and parameter update. The small batch size can be beneficial to lower memory consumption.

```
net.train<cross_entropy_multiclass>(optimizer,
        train_images, train_labels, minibatch,
        epochs, on_enumerate_minibatch,
        on_enumerate_epoch);
```

Callbacks `on_enumerate_epoch` can be used to execute code for each epoch or mini-batch, e.g., to validate accuracy, save the model or update the learning rate, etc.

```
// create callback for each epoch
auto on_enumerate_epoch = [&]() {
        // check accuracy of training data
        result res = net.test(train_images, train_labels);
        // print result
    res.print_summary(cout);
    // save model
    net.save(filepath);
}
```

The framework also offers methods to visualize the activations of a specific sample and layer. Further, the network weights can be visualized using its image class.

```
// visualize weights of layer 0
image<> img
= net.at<convolutional_layer>(0).weight_to_image();
img.write(filepath);

// visualize activations of last processed sample
img = net[0]->output_to_image();
img.write(filepath);
```

B.4 Using a Pre-Trained CNN

A saved model stores the trained weights as well as the model architecture, respectively, its layers. The `load()` function loads the model into a new network and can then be used for further training, evaluation, or weight extraction. Individual layers can be accessed using the `[ ]` operator, e.g., to get layer information or weight vectors.

```
net.load(filepath);
//extract weights
vector<vec_t*> extractedWeights = net[0]->weights();
// copy weights to new_net
network<sequential> new_net;
for (int w = 0; w < extractedWeights.at(0)->size(); w++) {
        new_net[0]->weights().at(0)->at(w) =
        extractedWeights.at(0)->at(w);
}
// freeze weights from layer 0
new_net[0]->set_trainable(false);
```

List of Abbreviations

RGB Red, Green, and Blue

RGB-D Red, Green, Blue, and Depth

D Depth

2D Two Dimensions

3D Three Dimensions

VGA Video Graphics Array

STIP Spatio-Temporal Interest Points

DT Dense Trajectory

IDT Improved Dense Trajectories

HOF Histogram of Optical Flow

BoW Bag-of-Words

ML Machine learning

GPUs Graphics Processing Units

CNN Convolutional Neural Networks

2D-CNN 2D Convolution Neural Network

3D-CNN 3D Convolutional Neural Network

ReLU Rectified Linear Units

IVS Intelligent Video Surveillance

HM Health Monitoring

HOI Human-Object Interactions

HCI Human-Computer Interaction

HRI Human-Robot Interaction

SIFT Scale-Invariant Feature Transform

SURF Speed-Up Robust Features

MHI Motion History Image

MEI Motion Energy Image

OF Optical Flow

HOG Histogram of Oriented Gradients

EDM Euclidean Distance Measure

ROI Region of Interest

BoFs Bag-of-Features

MBH Motion Boundary Histogram

LBP Local Binary Pattern

GLCM Gray-Level Co-occurrence Matrix

SVM Support Vector Machine

ANN Artificial Neural Network

RF Random Forest

NB Naive Bayes classifier

KNN K-Nearest Neighbors

IR Infrared Emitter

MMDT Multi-Modal Dense Trajectory

MM2DCNN multi-modal 2D Convolutional Neural Network

LBP-TOP Local Binary Pattern from Three Orthogonal Planes

2DPCA Two-Dimensional Principal Component Analysis

2DLDA Two-Dimensional Linear Discriminant Analysis

3DHOFs 3D Histograms of Scene Flow

GHOGs Global Histograms of Oriented Gradient

OvO One-vs-One

OvA One-vs-All

MSR3D MSR Daily Activity 3D

ORGBD Online RGBD

G3D Gaming 3D

CAD-60 Cornell Activity

NTU RGB+D Nanyang Technological University

MLNN Multi-Layer Neural Networks

MLP Multi-Layer Perceptron

RBF Radial Basis Function

BMS Boolean Map-based Saliency

RNN Recurrent Neural Network

DNN Deep Neural Network

2DTr 2D Dense Trajectory

3DTr 3D Dense Trajectory

List of Symbols

Symbol	**Description**
x	a horizontal co-ordinates of the image
y	a vertical co-ordinates of the image
I	image
i or (i, j)	counter index either in one-dimensional or two-dimensional indexation
N_{cl}	number of classes
λ	class label
G_F	global features
L_F	local features
$\boldsymbol{v}$	is a vector of real numbers
k	number of clusters in k-means clustering
$\mathbf{s}$	feature vector size
π	pi
w	image width
h	image height
$\boldsymbol{B}$	binary value
Th	threshold
$M_{H\tau}$	the temporal history of motion in images
τ	tau, the maximum duration of motion
$B(x, y, t)$	binary image
M or $M \times N$	which represent the size of one-dimensional or two-dimensional image
m	geometric moment
$HM1$ to $HM7$	seven Hu-moments
A_R	Action-Region
z	represents a depth direction
$\mathbf{f}$	scene flow
L_s	length of a trajectory
$\boldsymbol{Tr}$	Trajectory vector

F_m	feature map
d	distance value
$\boldsymbol{G}$	co-occurrence matrix
μ	mean
σ	standard deviation
$\boldsymbol{v_t}$	training vector
$\boldsymbol{v_s}$	testing vector
H	histogram output from the BoW
C	regularization term, which provides a way to control over-fitting
ξ	slack variables
δ_{f}	decision function
K	the number of neighbors (voting) on the test example's class in KNN classifier
Tr	number of training samples
Ts	number of testing samples
T	randomized decision trees
S	entropy
ω	weight
ς	softmax function
F	frame
N_F	number of frames
N_f	number of features
N_{Ch}	number of image channel
F_v	feature vector
AR	action recognition

List of Tables

2.1 Summary of the previous approaches used local features for action recognition. 39
2.2 Summary of the previous approaches used global features for action recognition. 43
2.3 Summary of the previous approaches used features combination methods for action recognition. 46
2.4 Summary of the previous approaches based on Convolution Neural Networks (CNNs) for human action recognition. 61
2.5 Summary of human activity RGB-D datasets. 66

4.1 Experimental results of the extracted local features from RGB and RGB-D of the MSR3D and ORGBD datasets. 123
4.2 Comparison results on RGB-D datasets of two feature representation methods 123
4.3 Comparison of recognition accuracy with other methods using MSR3D dataset 124
4.4 Comparison of recognition accuracy with other methods using ORGBD dataset 125

5.1 Experimental results on MSR3D dataset using different modalities. . . 137
5.2 Comparison of recognition accuracy with the state-of-the-art methods using MSR3D dataset 138
5.3 Experimental results on ORGBD dataset using different modalities. . . 139
5.4 Comparison of recognition accuracy with the state-of-the-art methods using ORGBD dataset 140

6.1 Comparison of recognition accuracy using different Retina channels . . 151
6.2 Comparison results with other methods on MSR3D dataset 152
6.3 Comparison results with other methods on ORGBD dataset 152
6.4 Comparison results with other methods on G3D dataset 153

7.1 Comparison of recognition rates from different classifier on four different RGB-D datasets. 163

7.2 Comparison of recognition performance with the different method used G3D dataset 165
7.3 Comparison of recognition performance with the different method used CAD-60 dataset 165
7.4 Comparison of recognition performance with the different method used MSR3D dataset 165
7.5 Comparison of recognition performance with the different method used ORGBD dataset 166

8.1 Top 10 accuracy classes NTU-60 cross-view evaluation. 177
8.2 Top 10 accuracy classes NTU-60 cross-subject evaluation. 179
8.3 Top 10 accuracy classes NTU-60 cross-subject evaluation. 180
8.4 Top 10 accuracy classes NTU-60 cross-subject evaluation. 182
8.5 Accuracy comparison results of 3D-CNN model and SVM classifier of the extracted 3D-CNN features using RGB, depth, and RGB-D sequences. 183
8.6 Comparison with the state-of-the-art methods on NTU-RGBD cross-subject split dataset. 183

List of Figures

1.1 Action and activities video samples 14
1.2 RGB images example 15
1.3 Depth images example 15
1.4 General structure of the proposed action recognition approach 20

2.1 Illustration of 3D camera and RGB-D data 27
2.2 Human action representation 29
2.3 RGB and depth video sequence representation 30
2.4 Categorization of feature representation 31
2.5 Effects of Parvo and Magno pre-processing 35
2.6 Example of extracted DSTIPs 37
2.7 Illustration of Dense Trajectory description 38
2.8 Global spatio-temporal representations from the MHI and the OF-MHI . 42
2.9 Combing RGB and Depth map features framework 45
2.10 Overview of the recognition system 46
2.11 Machine learning techniques include both supervised and unsupervised learning 47
2.12 A typical CNN architecture with two feature stages 55
2.13 General structure of the CNN 56
2.14 Overview of the two-stream CNN system 57
2.15 Architecture of 3D-CNN for spatio-temporal features extraction 58
2.16 Illustration of the 3D-CNN across both spatial and temporal domains. . 59
2.17 Viewpoint invariant RGB-D human action recognition 60
2.18 Sample frames of MSR Daily Activity 3D dataset 63
2.19 Sample frames of Online RGBD action dataset 64
2.20 Color and depth samples from different gaming actions 64
2.21 Sample RGB-D frames selected from CAD-60 dataset from different actions 65
2.22 RGB-D dataset images example from NTU RGB+D 66

3.1 The Retina model 74
3.2 Illustration of the original LBP 81

3.3 Description of the Gray Level Co-occurrence Matrix 87
3.4 A mathematical model of an artificial neural network neuron 98
3.5 Multi-Layer perception neural network 100
3.6 Basic of the CNN block 101
3.7 Max pooling process 102
3.8 2D convolutional neural network 103
3.9 Representation of 3D convolution layer process 105
3.10 3D convolutional neural network 106
3.11 Operations of 2D and 3D convolutions 107

4.1 System analysis schematics of action recognition 111
4.2 Local feature extraction 113
4.3 Local motion and appearance features representation 115
4.4 Seven Hu-moments of the MHI from video sequence 117
4.5 Pipeline of the action classification of the proposed approach 118
4.6 Motion points in RGB and depth frames 122

5.1 General structure of the proposed system 129
5.2 Example of the 3D Trajectories 131
5.3 Pipeline of RGB-D action recognition with the 3D Trajectory 133
5.4 Pipeline of depth activity recognition with the global 3DGIST descriptor 134

6.1 General structure steps of the system approach for human action recognition 143
6.2 The Retina Parvocellular channel (Parvo) 145
6.3 The Retina Magnocellular channel (Magno) 146
6.4 Feature vector generation. 147
6.5 Effects of Parvocellular and Mgnocellular preprocessing and moving area from RGB and depth. 151

7.1 Overview of the proposed activity recognition system 157
7.2 Accuracy rates comparison using four different RGB-D datasets 164

8.1 Structure of the proposed two 3D-CNN models for improving human action recognition form RGB and depth sequences 169
8.2 General Convolutional Neural Networks system for human action recognition form RGB and depth sequences 170
8.3 Sample OF volume and corresponding RGB frames 171
8.4 Sample frame of action from depth data 172
8.5 Sample frames of action drinking 172
8.6 Dense optical flow example 174
8.7 Proposed 3D-CNN architecture 175

8.8 Test and train accuracy of NTU-10 178
8.9 Confusion matrix of test data evaluation for NTU-10 cross-subject . . . 178
8.10 Test and train accuracy of NTU-60 179
8.11 SVM classification accuracy for feature vectors extracted from different layers of the NTU10 CNN 180

A.1 Sample frames of MSR Daily Activity 3D dataset 196
A.2 Sample frames of Online RGBD Action Dataset 197
A.3 Color and depth samples from different gaming actions 199
A.4 Sample frames selected from CAD-60 dataset from different actions . . 200
A.5 NTU RGB-D dataset images example 202

Own-Publications

[AaAdP18] AL-AKAM, Rawya ; AL-DARRAJI, Salah ; PAULUS, Dietrich: Human Action Recognition from RGBD Videos based on Retina Model and Local Binary Pattern Features. In: *26.Conference on Computer Graphics, Visualization and Computer Vision (WSCG), Plzen, Czech Republic*, 2018. – ISBN 9788086943428, S. 1–7

[AaP17] AL-AKAM, Rawya ; PAULUS, Dietrich: RGBD Human Action Recognition using Multi-Features Combination and K-Nearest Neighbors Classification. In: *International Journal of Advanced Computer Science and Applications (IJACSA)* 8 (2017), Nr. 10, 383–389. `http://dx.doi.org/10.14569/IJACSA.2017.081050`. – DOI 10.14569/IJACSA.2017.081050

[AAP18a] AL-AKAM, Rawya ; PAULUS, Dietrich: Dense 3D Optical Flow Co-occurrence Matrices for Human Activity Recognition. In: *5th International Workshop on Sensor-based Activity Recognition and Interaction*. New York, NY, USA : ACM, 2018 (iWOAR '18). – ISBN 978–1–4503–6487–4, 16:1–16:8

[AaP18b] AL-AKAM, Rawya ; PAULUS, Dietrich: Local and Global feature Descriptors Combination from RGB-Depth Videos for Human Action Recognition. In: *7th International Conference on Pattern Recognition Applications and Methods (ICPRAM), Funchal, Madeira-Portugal*, 2018. – ISBN 978–989–758–276–9, S. 265–272

[AAP18c] AL-AKAM, Rawya ; PAULUS, Dietrich: Local Feature Extraction from RGB and Depth Videos for Human Action Recognition. In: *International Journal of Machine Learning and Computing (IJMLC),* 8 (2018), Nr. 3, S. 274–279

[AAP19] AL-AKAM, Rawya ; PAULUS, Dietrich: Comparison of different machine learning models on feature extraction for human activity recognition from RGB-depth datasets. In: *Proc. SPIE 11041, Eleventh International Conference on Machine Vision (ICMV), Munich, Germany* Bd. 11041, 2019

[AaPG18] AL-AKAM, Rawya ; PAULUS, Dietrich ; GHARABAGHI, Darius: Human Action Recognition based on 3D Convolution Neural Networks from RGBD Videos. In: *26.Conference on Computer Graphics, Visualization and Computer Vision (WSCG), Plzen, Czech Republic*, 2018. – ISBN 9788086943428, S. 18–26

Bibliography

[AABR$^+$17] ASADI-AGHBOLAGHI, Maryam ; BERTICHE, Hugo ; ROIG, Vicent ; KASAEI, Shohreh ; ESCALERA, Sergio: Action Recognition from RGB-D Data: Comparison and Fusion of Spatio-Temporal Handcrafted Features and Deep Strategies. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)* (2017), S. 3179–3188. `http://dx.doi.org/10.1109/ICCVW.2017.376`. – DOI 10.1109/ICCVW.2017.376

[AATM17] AL-AZZO, Fadwa ; TAQI, Arwa M. ; MILANOVA, Mariofanna: 3D Human Action Recognition using Hu Moment Invariants and Euclidean Distance Classifier. In: *International Journal of Advanced Computer Science and Applications* 8 (2017), Nr. 4. `http://dx.doi.org/10.14569/IJACSA.2017.080403`. – DOI 10.14569/IJACSA.2017.080403

[AD16] AL-DARRAJI, Salah: *Perception of Nonverbal Cues for Human-Robot Interaction*. Department of Computer Science, University of Kaiserslautern, Ph.D. Theses, 2016. `https://www.dr.hut-verlag.de/9783843928526.html`

[AEV17] ALMEIDA, Ana Paula G S D. ; ESPINOZA, Bruno Luiggi M. ; VIDAL, Flavio De B.: Human Action Recognition in Videos: A comparative evaluation of the classical and velocity adaptation space-time interest points techniques. In: *Conference on Computer Graphics, Visualization and Computer Vision WSCG*, 2017. – ISBN 978–80–86943–50–3

[AFCYN19] AL-FARIS, Mahmoud ; CHIVERTON, John ; YANG, Yanyan ; NDZI, David: Deep learning of fuzzy weighted multi-resolution depth motion maps with spatial feature fusion for action recognition. In: *Journal of Imaging* 5 (2019), Nr. 10. `http://dx.doi.org/10.3390/jimaging5100082`. – DOI 10.3390/jimaging5100082. – ISSN 2313–433X

[AK15] ATHIWARATKUN, Ben ; KANG, Keegan: Feature Representation in Convolutional Neural Networks. In: *CoRR* abs/1507.02313 (2015). `http://arxiv.org/abs/1507.02313`

[AML08] AKHLOUFI, Moulay A. ; MALDAGUE, Xavier ; LARBI, Wael B.: A New Color-Texture Approach for Industrial Products Inspection. In: *JOURNAL OF MULTIMEDIA* 3 (2008), Nr. 3, S. 44–50. `http://dx.doi.org/10.4304/jmm.3.3.44-50`. – DOI 10.4304/jmm.3.3.44–50

[Ana95] ANAND, Mehrotra K. Mohan C. K. & Ranka-S. R.: Efficient classification for multiclass problems using modular neural networks. In: *IEEE Transactions on Neural Networks* 6 (1995), Nr. 1, S. 117–124. `http://dx.doi.org/10.1109/72.363444`. – DOI 10.1109/72.363444

[AR11] AGGARWAL, J. K. ; RYOO, M. S.: Human activity analysis: A review. In: *ACM Computing Surveys* 43 (2011), Nr. 3, 16:1–16:43. `http://dx.doi.org/10.1145/1922649.1922653`. – DOI 10.1145/1922649.1922653

[ASS16] AMOR, B. B. ; SU, J. ; SRIVASTAVA, A.: Action Recognition Using Rate-Invariant Analysis of Skeletal Shape Trajectories. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38 (2016), Nr. 1, S. 1–13. `http://dx.doi.org/10.1109/TPAMI.2015.2439257`. – DOI 10.1109/TPAMI.2015.2439257

[ATK^{+}15] AZHAR, Ryfial ; TUWOHINGIDE, Desmin ; KAMUDI, Dasrit ; SARIMUDDIN ; SUCIATI, Nanik: Batik Image Classification Using SIFT Feature Extraction, Bag of Features and Support Vector Machine. In: *Procedia Computer Science* 72 (2015), 24–30. `http://dx.doi.org/10.1016/j.procs.2015.12.101`. – DOI 10.1016/j.procs.2015.12.101. – ISSN 18770509

[ATKI14] AHSAN, S. M. M. ; TAN, J. K. ; KIM, H. ; ISHIKAWA, S.: Histogram of spatio temporal local binary patterns for human action recognition. In: *Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS)*, 2014, S. 1007–1011

[Ayo10] AYODELE, Taiwo O.: Types of Machine Learning Algorithms. Version: 2010. `http://dx.doi.org/10.5772/9385`. In: ZHANG, Yagang (Hrsg.): *New Advances in Machine Learning*. Rijeka : IntechOpen, 2010. – DOI 10.5772/9385, Kapitel 3

[BAHLC09] BENOIT, Alexandre ; ALLEYSSON, David ; HERAULT, Jeanny ; LE CALLET, Patrick: Spatio-temporal Tone Mapping Operator Based on a Retina Model. In: *Computational Color Imaging*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2009. – ISBN 978–3–642–03265–3, S. 12–22

[BAM16] BLOOM, Victoria ; ARGYRIOU, Vasileios ; MAKRIS, Dimitrios: Hierarchical transfer learning for online recognition of compound actions. In: *Computer Vision and Image Understanding* 144 (2016), S. 62–72. `http://dx.doi.org/10.1016/j.cviu.2015.12.001`. – DOI 10.1016/j.cviu.2015.12.001. – ISBN 1077–3142

[BB98] BURGES, C.J.C. ; BURGES, Chris J.: A Tutorial on Support Vector Machines for Pattern Recognition. In: *Data Mining and Knowledge Discovery* 2 (1998), January, S. 121–167. `http://dx.doi.org/10.1023/A:1009715923555`. – DOI 10.1023/A:1009715923555

[BCDH10] BENOIT, A. ; CAPLIER, A. ; DURETTE, B. ; HÉRAULT, J.: Using Human Visual System Modeling for Bio-Inspired Low Level Image Processing. In: *Computer Vision and Image Understanding* 114 (2010), Nr. 7, S. 758 – 773. `http://dx.doi.org/10.1016/j.cviu.2010.01.011`. – DOI 10.1016/j.cviu.2010.01.011

[BCK13] B, Girisha A. ; CHANDRASHEKHAR, M C. ; KURIAN, M Z.: Texture Feature Extraction of Video Frames Using GLCM. In: *International Journal of Engineering Trends and Technology* 4 (2013), Nr. 6, S. 2718–2721

[BD00] BRADSKI, G. R. ; DAVIS, J.: Motion segmentation and pose recognition with motion history gradients. In: *Proceedings of IEEE Workshop on Applications of Computer Vision* (2000), S. 238–244. `http://dx.doi.org/10.1109/WACV.2000.895428`. – DOI 10.1109/WACV.2000.895428. – ISBN 0769508138

[BD01] BOBICK, Aaron F. ; DAVIS, James W.: The Recognition of Human Movement Using Temporal Templates. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23 (2001), Nr. 3, S. 257–267. `http://dx.doi.org/10.1109/34.910878`. – DOI 10.1109/34.910878. – ISBN 0162–8828

[BETV08] BAY, Herbert ; ESS, Andreas ; TUYTELAARS, Tinne ; VAN GOOL, Luc: Speeded-Up Robust Features (SURF). In: *Computer Vision and Image Understanding* 110 (2008), Nr. 3, S. 346–359. `http://dx.doi.org/10.1016/j.cviu.2007.09.014`. – DOI 10.1016/j.cviu.2007.09.014. – ISBN 9783540338321

[BFB94] BARRON, J. L. ; FLEET, D. J. ; BEAUCHEMIN, S. S.: Performance of Optical Flow Techniques. In: *International Journal of Computer Vision* 12 (1994), Nr. 1, S. 43–77. `http://dx.doi.org/10.1007/BF01420984`. – DOI 10.1007/BF01420984. – ISBN 0818628553

[BGS$^+$05] BLANK, Moshe ; GORELICK, Lena ; SHECHTMAN, Eli ; IRANI, Michal ; BASRI, Ronen: Actions as Space-Time Shapes. In: *The Tenth IEEE International Conference on Computer Vision (ICCV'05)*, 2005, S. 1395–1402

[Bil14] BILIŃSKI, Piotr T.: *Human action recognition in videos*, Université Nice Sophia Antipolis, Ph.D. Thesis, 2014. `https://tel.archives-ouvertes.fr/tel-01134481`

[Bis06] BISHOP, Christopher M.: *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg : Springer-Verlag, 2006. `http://dx.doi.org/10.5555/1162264`. `http://dx.doi.org/10.5555/1162264`. – ISBN 0387310738

[BMA12] BLOOM, Victoria ; MAKRIS, Dimitrios ; ARGYRIOU, Vasileios: G3D: A gaming action dataset and real time action recognition evaluation framework. In: *Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)* (2012), 7–12. `http://dx.doi.org/10.1109/CVPRW.2012.6239175`. – DOI 10.1109/CVPRW.2012.6239175. – ISBN 2160–7508

[BMM12] BALLIN, Gioia ; MUNARO, Matteo ; MENEGATTI, Emanuele: Human Action Recognition from RGB-D Frames Based on Real-Time 3D Optical Flow Estimation. In: *Biologically Inspired Cognitive Architectures (BICA)* Bd. 196, 2012. – ISBN 978–3–642–34273–8, S. 65–74

[BMW$^+$11] BACCOUCHE, Moez ; MAMALET, Franck ; WOLF, Christian ; GARCIA, Christophe ; BASKURT, Atilla: Sequential Deep Learning for Human Action Recognition. In: *Human Behavior Understanding*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2011. – ISBN 978–3–642–25446–8, S. 29–39

[BP20] BASAVAIAH, Jagadeesh ; PATIL, Chandrashekar M.: Human Activity detection and Action Recognition in Videos using Convolutional Neural Networks. In: *Journal of Information and Communication Technology* 19 (2020), Nr. 2, 157–183. `http://e-journal.uum.edu.my/index.php/jict/article/view/5595`. – ISSN 2180–3862

[Bre96] BREIMAN, Leo: Bagging predictors. In: *Machine learning* 24 (1996), Nr. 2, S. 123–140. `http://dx.doi.org/doi.org/10.1007/BF00058655`. – DOI doi.org/10.1007/BF00058655

[Bre01] BREIMAN, Leo: Random forests. In: *Machine Learning* 45 (2001), Nr. 1, S. 5–32. `http://dx.doi.org/10.1023/A:1010933404324`. – DOI 10.1023/A:1010933404324. – ISBN 0885–6125

[BZK18] BERRADA, Leonard ; ZISSERMAN, Andrew ; KUMAR, M. P.: Smooth Loss Functions for Deep Top-k Classification. In: *CoRR* abs/1802.07595 (2018). `http://arxiv.org/abs/1802.07595`

[CAS^{+}17] Chen, L. ; Ai, H. ; Shang, C. ; Zhuang, Z. ; Bai, B.: Online multi-object tracking with convolutional neural networks. In: *2017 IEEE International Conference on Image Processing (ICIP)*, 2017. – ISSN 2381–8549, S. 645–649

[CC08] Cord, Matthieu ; Cunningham, Pdraig: *Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval (Cognitive Technologies)*. 1. Santa Clara, CA, USA : Springer-Verlag TELOS, 2008. http://dx.doi.org/10.5555/1370948. http://dx.doi.org/10.5555/1370948. – ISBN 354075170X

[CCFC13] Chaquet, Jose M. ; Carmona, Enrique J. ; Fernández-Caballero, Antonio: A Survey of Video Datasets for Human Action and Activity Recognition. In: *Comput. Vis. Image Underst.* 117 (2013), jun, Nr. 6, 633–659. http://dx.doi.org/10.1016/j.cviu.2013.01.013. – DOI 10.1016/j.cviu.2013.01.013. – ISSN 1077–3142

[CCS16] Carlos Caetano, Jefersson A. dos S. ; Schwartz, William R.: Optical Flow Co-occurrence Matrices: A novel spatiotemporal feature descriptor. In: *International Conference on Pattern Recognition (ICPR)*, 2016, 1947–1952

[CDF^{+}04] Csurka, Gabriella ; Dance, Christopher R. ; Fan, Lixin ; Willamowski, Jutta ; Bray, Cedric: Visual Categorization with Bags of Keypoints. In: *In Workshop on Statistical Learning in Computer Vision, ECCV*, 2004

[CFP^{+}13] Carletti, Vincenzo ; Foggia, Pasquale ; Percannella, Gennaro ; Saggese, Alessia ; Vento, Mario: Recognition of Human Actions from RGB-D Videos Using a Reject Option. In: *New Trends in Image Analysis and Processing – ICIAP*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2013. – ISBN 978–3–642–41190–8, S. 436–445

[CGGS12] Ciresan, Dan ; Giusti, Alessandro ; Gambardella, Luca M. ; Schmidhuber, Jürgen: Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images. Version: 2012. http://dl.acm.org/citation.cfm?id=2999325.2999452. In: *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*. Curran Associates, Inc., 2012, 2843–2851

[CH06] Cover, T. ; Hart, P.: Nearest Neighbor Pattern Classification. In: *IEEE Transactions on Information Theory* 13 (2006), September, Nr. 1, S. 21–27. http://dx.doi.org/10.1109/TIT.1967.1053964. – DOI 10.1109/TIT.1967.1053964. – ISSN 0018–9448

[Cha11] Chaparro, Luis F.: Chapter 11 - Introduction to the Design of Discrete Filters. Version: 2011. http://dx.doi.org/10.1016/B978-0-12-374716-7.00015-6. In: *Signals and Systems Using MATLAB*. Boston : Academic Press, 2011. – DOI 10.1016/B978–0–12–374716–7.00015–6. – ISBN 978–0–12–374716–7, S. 639 – 707

[CHLS17] Cai, Ziyun ; Han, Jungong ; Liu, Li ; Shao, Ling: RGB-D datasets using microsoft kinect or similar sensors: a survey. In: *Multim. Tools Appl.* 76 (2017), Nr. 3, 4313–4355. http://dx.doi.org/10.1007/s11042-016-3374-6. – DOI 10.1007/s11042–016–3374–6

[CK15] Cook, Diane J. ; Krishnan, Narayanan C.: Activity Learning. In: *Activity Learning: Discovering, Recognizing, and Predicting Human Behavior from Sensor Data*, Wiley Series on Parallel and Distributed Computing, 2015. – ISBN 9781119010258

[CL11] CHANG, Chih-Chung ; LIN, Chih-Jen: LIBSVM: A Library for Support Vector Machines. In: *ACM Transactions on Intelligent Systems and Technology* 2 (2011), Nr. 3. `http://dx.doi.org/10.1145/1961189.1961199`. – DOI 10.1145/1961189.1961199

[CLV12] CRUZ, L. ; LUCIO, D. ; VELHO, L.: Kinect and RGBD Images: Challenges and Applications. In: *25th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials*, 2012, S. 36–49

[CRHV09] CHAUDHRY, Rizwan ; RAVICHANDRAN, Avinash ; HAGER, Gregory ; VIDAL, René: Histograms of oriented optical flow and Binet-Cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*, 2009. – ISBN 9781424439935, S. 1932–1939

[CS11] CHAMASEMANI, F. F. ; SINGH, Y. P.: Multi-class Support Vector Machine (SVM) Classifiers – An Application in Hypothyroid Detection and Classification. In: *Sixth International Conference on Bio-Inspired Computing: Theories and Applications*, 2011, S. 351–356

[CS12] CAMPLANI, Massimo ; SALGADO, Luis: Efficient spatio-temporal hole filling strategy for Kinect depth maps. In: BASKURT, Atilla M. (Hrsg.) ; SITNIK, Robert (Hrsg.) ; International Society for Optics and Photonics (Veranst.): *Three-Dimensional Image Processing (3DIP) and Applications II* Bd. 8290 International Society for Optics and Photonics, SPIE, 2012, 127 – 136

[CS17] CHANG, Jing ; SHA, Jin: An efficient implementation of 2D convolution in CNN. In: *IEICE Electronics Express* 14 (2017), 01, Nr. 1, S. 1–8. `http://dx.doi.org/10.1587/elex.13.20161134`. – DOI 10.1587/elex.13.20161134

[CSK11] CRIMINISI, Antonio ; SHOTTON, Jamie ; KONUKOGLU, Ender: Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning. In: *Foundations and Trendstextregistered in Computer Graphics and Vision* 7 (2011), 01, 81-227. `http://dx.doi.org/10.1561/0600000035`. – DOI 10.1561/0600000035

[CZ17] CARREIRA, João ; ZISSERMAN, Andrew: Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In: *CoRR* abs/1705.07750 (2017). `http://arxiv.org/abs/1705.07750`

[CZG$^+$13] CHEN, Guang ; ZHANG, Feihu ; GIULIANI, Manuel ; BUCKL, Christian ; KNOLL, Alois: Unsupervised Learning Spatio-temporal Features for Human Activity Recognition from RGB-D Video Data. In: *Social Robotics*, Springer International Publishing, 2013. – ISBN 978–3–319–02675–6, S. 341–350

[CZL17] CHEN, Enqing ; ZHANG, Shichao ; LIANG, Chengwu: Action Recognition Using Motion History Image and Static History Image-based Local Binary Patterns. In: *International Journal of Multimedia and Ubiquitous Engineering* 12 (2017), Nr. 1, 203–214. `http://dx.doi.org/10.14257/ijmue.2017.12.1.17`. – DOI 10.14257/ijmue.2017.12.1.17. – ISSN 1975–0080

[DBPC19] DE BIE, Gwendoline ; PEYRÉ, Gabriel ; CUTURI, Marco: Stochastic Deep Networks. In: CHAUDHURI, Kamalika (Hrsg.) ; SALAKHUTDINOV, Ruslan (Hrsg.): *Proceedings of the 36th International Conference on Machine Learning* Bd. 97. Long Beach, California, USA : PMLR, 09–15 Jun 2019 (Proceedings of Machine Learning Research), S. 1556–1565

[DJS$^+$09] DOUZE, Matthijs ; JÉGOU, Hervé ; SANDHAWALIA, Harsimrat ; AMSALEG, Laurent ; SCHMID, Cordelia: Evaluation of GIST Descriptors for Web-scale Image Search. In: *Proceedings of the ACM International Conference on Image and Video Retrieval*. New York, NY, USA : ACM, 2009 (CIVR '09). – ISBN 978–1–60558–480–5, 19:1–19:8

[DKBK20] DU, Kanghui ; KACZMAREK, Thomas ; BRŠČIĆ, Dražen ; KANDA, Takayuki: Recognition of Rare Low-Moral Actions Using Depth Data. In: *Sensors* 20 (2020), Nr. 10, S. 2758

[DLCZ16] DING, Wenwen ; LIU, Kai ; CHENG, Fei ; ZHANG, Jin: Learning Hierarchical Spatio-temporal Pattern for Human Activity Prediction. In: *Journal of Visual Communication and Image Representation* 35 (2016), 103–111. `http://dx.doi.org/10.1016/j.jvcir.2015.12.006`. – DOI 10.1016/j.jvcir.2015.12.006. – ISBN 10473203

[DRB05] DOLLAR, Piotr ; RABAUD, Cottrell G. Vincent ; BELONGIE, Serge: Behavior Recognition via Sparse Spatio-Temporal Features. In: *Proceedings 2nd Joint IEEE International Workshop on VS-PETS*, 2005. – ISBN 0780394240, S. 2–5

[DT05] DALAL, Navneet ; TRIGGS, Bill: Histograms of oriented gradients for human detection. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* Bd. 1, 2005. – ISSN 1063–6919, S. 886–893 vol. 1

[DTGÇ06] DEDEOĞLU, Yiğithan ; TÖREYIN, B. U. ; GÜDÜKBAY, Uğur ; ÇETIN, A. E.: Silhouette-Based Method for Object Classification and Human Action Recognition in Video. In: *Computer Vision in Human-Computer Interaction*, Springer Berlin Heidelberg, 2006, 64–77

[DTS06] DALAL, Navneet ; TRIGGS, Bill ; SCHMID, Cordelia: Human detection using oriented histograms of flow and appearance, 2006. – ISBN 3–540–33834–9 978–3–540–33834–5, S. 428–441

[Far03] FARNEBÄCK, Gunnar: Two-frame Motion Estimation Based on Polynomial Expansion. In: *Proceedings of the 13th Scandinavian Conference on Image Analysis*. Berlin, Heidelberg : Springer-Verlag, 2003 (SCIA'03). – ISBN 3–540–40601–8, 363–370

[FAT11] FOIX, S. ; ALENYA, G. ; TORRAS, C.: Lock-in Time-of-Flight (ToF) Cameras: A Survey. In: *IEEE Sensors Journal* 11 (2011), Nr. 9, S. 1917–1926. `http://dx.doi.org/10.1109/JSEN.2010.2101060`. – DOI 10.1109/JSEN.2010.2101060

[FBK15] FORTUN, Denis ; BOUTHEMY, Patrick ; KERVRANN, Charles: Optical Flow Modeling and Computation: A Survey. In: *Computer Vision and Image Understanding* 134 (2015), 02. `http://dx.doi.org/10.1016/j.cviu.2015.02.008`. – DOI 10.1016/j.cviu.2015.02.008

[FCNL13] FARABET, C. ; COUPRIE, C. ; NAJMAN, L. ; LECUN, Y.: Learning Hierarchical Features for Scene Labeling. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (2013), Aug, Nr. 8, S. 1915–1929. `http://dx.doi.org/10.1109/TPAMI.2012.231`. – DOI 10.1109/TPAMI.2012.231. – ISSN 1939–3539

[FGMO13] FANELLO, Sean R. ; GORI, Ilaria ; METTA, Giorgio ; ODONE, Francesca: One-Shot Learning for Real-Time Action Recognition. In: *Pattern Recognition and Image Analysis*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2013. – ISBN 978–3–642–38628–2, S. 31–40

[GDDM13] GIRSHICK, Ross B. ; DONAHUE, Jeff ; DARRELL, Trevor ; MALIK, Jitendra: Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition* (2013), S. 580–587

[GGDH17] GKIOXARI, Georgia ; GIRSHICK, Ross B. ; DOLLÁR, Piotr ; HE, Kaiming: Detecting and Recognizing Human-Object Interactions. In: *CoRR* abs/1704.07333 (2017). `http://arxiv.org/abs/1704.07333`

[GLCL19] GEORGIOU, Theodoros ; LIU, Youfang ; CHEN, Wei ; LEW, Michael S.: A survey of traditional and deep learning-based feature descriptors for high dimensional data in computer vision. In: *International Journal of Multimedia Information Retrieval* (2019), S. 1 – 36. `http://dx.doi.org/10.1007/s13735-019-00183-w`. – DOI 10.1007/s13735–019–00183–w

[GPF17] GRZESZICK, Rene ; PLINGE, Axel ; FINK, Gernot A.: Bag-of-Features Methods for Acoustic Event Detection and Classification. In: *IEEE/ACM Trans. Audio, Speech and Language Processing* 25 (2017), Nr. 6, 1242–1252. `http://ieeexplore.ieee.org/document/7933055/`

[GPT08] GENUER, Robin ; POGGI, Jean-Michel ; TULEAU, Christine: Random Forests: some methodological insights. In: *arXiv preprint arXiv:0811.3619* (2008)

[HAD14] HENDAOUI, R. ; ABDELLAOUI, M. ; DOUIK, A.: Synthesis of spatio-temporal interest point detectors: Harris 3D, MoSIFT and SURF-MHI. In: *1st International Conference on Advanced Technologies for Signal and Image Processing, ATSIP* (2014), S. 89–94. `http://dx.doi.org/10.1109/ATSIP.2014.6834583`. – DOI 10.1109/ATSIP.2014.6834583. ISBN 9781479948888

[ḦD11] ḦARA, Stephen O. ; DRAPER, Bruce A.: Introduction to the Bag of Features Paradigm for Image Classification and Retrieval. In: *CoRR* abs/1101.3354 (2011). `https://arxiv.org/abs/1101.3354`

[Her96] HERAULT, J.: A model of colour processing in the retina of vertebrates: From photoreceptors to colour opposition and colour constancy phenomena. In: *Neurocomputing* 12 (1996), Nr. 2, S. 113 – 129. `http://dx.doi.org/10.1016/0925-2312(95)00114-X`. – DOI 10.1016/0925–2312(95)00114–X. – ISSN 0925–2312

[HHLH11] HUANG, Chin P. ; HSIEH, Chaur H. ; LAI, Kuan T. ; HUANG, Wei Y.: Human action recognition using histogram of oriented gradient of motion history image. In: *First International Conference on Instrumentation, Measurement, Computer, Communication and Control, IMCCC 2011*, 2011. – ISBN 9780769545196, S. 353–356

[HKM06] HODZIC, Nermin ; KONJIC, Tatjana ; MIRANDA, Vladimiro: Artificial Neural Networks Applied To Short Term Load Diagram Prediction. In: *8th Seminar on Neural Network Applications in Electrical Engineering*, 2006, S. 219–223

[HSD73] HARALICK, R. M. ; SHANMUGAM, K. ; DINSTEIN, I.: Textural Features for Image Classification. In: *IEEE Transactions on Systems, Man, and Cybernetics* SMC-3 (1973), Nov, Nr. 6, S. 610–621. `http://dx.doi.org/10.1109/TSMC.1973.4309314`. – DOI 10.1109/TSMC.1973.4309314. – ISSN 0018–9472

[HSK$^+$12] HINTON, Geoffrey E. ; SRIVASTAVA, Nitish ; KRIZHEVSKY, Alex ; SUTSKEVER, Ilya ; SALAKHUTDINOV, Ruslan: Improving neural networks by preventing co-adaptation of feature detectors. In: *CoRR* abs/1207.0580 (2012). http://arxiv.org/abs/1207.0580

[HSXS13] HAN, Jungong ; SHAO, Ling ; XU, Dong ; SHOTTON, Jamie: Enhanced computer vision with Microsoft Kinect sensor: A review. In: *IEEE Transactions on Cybernetics* 43 (2013), Nr. 5, S. 1318–1334. http://dx.doi.org/10.1109/TCYB.2013.2265378. – DOI 10.1109/TCYB.2013.2265378. – ISBN 2168–2267

[HT98] HASTIE, T. ; TIBSHIRANI, R.: Classification by pairwise coupling. In: *The Annals of Statistics* 26 (1998), 04, Nr. 2, S. 451–471. http://dx.doi.org/10.1214/aos/1028144844. – DOI 10.1214/aos/1028144844

[Hu62] HU, Ming-Kuei: Visual pattern recognition by moment invariants. In: *IRE Transactions on Information Theory* 8 (1962), Nr. 2, S. 179–187. http://dx.doi.org/10.1109/TIT.1962.1057692. – DOI 10.1109/TIT.1962.1057692. – ISBN 0096–1000

[HZRS15] HE, Kaiming ; ZHANG, Xiangyu ; REN, Shaoqing ; SUN, Jian: Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In: *CoRR* abs/1502.01852 (2015). http://arxiv.org/abs/1502.01852

[IcS10] IKIZLER-CINBIS, Nazli ; SCLAROFF, Stan: Object, Scene and Actions: Combining Multiple Features for Human Action Recognition. In: *11th European Conference on Computer Vision (ECCV)*, 2010. – ISBN 9783642155482, S. 494–507

[JBCS13] JIANG, Yu G. ; BHATTACHARYA, Subhabrata ; CHANG, Shih F. ; SHAH, Mubarak: High-level event recognition in unconstrained videos. In: *International Journal of Multimedia Information Retrieval* 2 (2013), Nr. 2, S. 73–101. http://dx.doi.org/10.1007/s13735-012-0024-2. – DOI 10.1007/s13735–012–0024–2. – ISBN 1373501200

[JC13] JIAN CAO, Qiang Cai Hai-sheng Li Jun-ping D. Dian-hui Mao M. Dian-hui Mao: A review of object representation based on local features. In: *Journal of Zhejiang University SCIENCE C* 14 (2013), Nr. 7, 495–504. http://dx.doi.org/10.1631/jzus.CIDE1303. – DOI 10.1631/jzus.CIDE1303. – ISSN 1869–1951

[JDX$^+$12] JIANG, Yu-Gang ; DAI, Qi ; XUE, Xiangyang ; LIU, Wei ; NGO, Chong-Wah: Trajectory-Based Modeling of Human Actions with Motion Reference Points. In: *Computer Vision – ECCV 2012*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012. – ISBN 978–3–642–33715–4, S. 425–438

[JKAM18] JEGHAM, I. ; KHALIFA, A. B. ; ALOUANI, I. ; MAHJOUB, M. A.: Safe Driving : Driver Action Recognition using SURF Keypoints. In: *30th International Conference on Microelectronics (ICM)*, 2018, S. 60–63

[JMF99] JAIN, A. K. ; MURTY, M. N. ; FLYNN, P. J.: Data Clustering: A Review. In: *ACM Computing Surveys (CSUR)* 31 (1999), Nr. 3, 264–323. http://dx.doi.org/10.1145/331499.331504. – DOI 10.1145/331499.331504. – ISSN 0360–0300

[Joh75] JOHANSSON, Gunnar: Visual motion perception. In: *Scientific American* 232 (1975), Nr. 6, 76–89. https://www.jstor.org/stable/24949822

[JS10] In: JIN, Rui ; SHAO, Ling: *Retrieving Human Actions Using Spatio-Temporal Features and Relevance Feedback*. London : Springer London, 2010. – ISBN 978–1–84996–507–1, S. 1–23

[JXYY13] JI, Shuiwang ; XU, Wei ; YANG, Ming ; YU, Kai: 3D Convolutional Neural Networks for Human Action Recognition. In: *IEEE Transaction on Pattern Analysis and Machine Intelligence* 35 (2013), Nr. 1, S. 221–231. `http://dx.doi.org/10.1109/TPAMI.2012.59`. – DOI 10.1109/TPAMI.2012.59. – ISBN 0162–8828 VO – 35

[KAD17] KALBOUSSI, Rahma ; ABDELLAOUI, Mehrez ; DOUIK, Ali: Video Saliency Detection Based on Boolean Map Theory. In: *Image Analysis and Processing(ICIAP)*. Cham : Springer International Publishing, 2017. – ISBN 978–3–319–68560–1, S. 119–128

[Kar15] KARAN, Branko: Calibration of kinect-type RGB-D sensors for robotic applications. In: *Fme Transactions* 43 (2015), S. 47–54

[KBB14] KOPERSKI, Michal ; BILINSKI, Piotr ; BREMOND, François: 3D Trajectories for Action Recognition. In: *The 21st IEEE International Conference on Image Processing (ICIP)*. Paris, France, 2014. – ISBN 9781479957514, 4176–4180

[KG13] KOMORKIEWICZ, Mateusz ; GORGON, Marek: Foreground object features extraction with GLCM texture descriptor in FPGA. In: *Conference on Design and Architectures for Signal and Image Processing*, IEEE, 2013. – ISBN 979–10–92279–01–6, 157-164

[KGDE19] KHEMMAR, Redouane ; GOUVEIA, Matthias ; DECOUX, Benoit ; ERTAUD, Jean-Yves: Real Time Pedestrian and Object Detection and Tracking-based Deep Learning. Application to Drone Visual Tracking. In: *27. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, 2019

[KGS12] KOPPULA, Hema S. ; GUPTA, Rudhir ; SAXENA, Ashutosh: Learning Human Activities and Object Affordances from RGB-D Videos. In: *CoRR* abs/1210.1207 (2012). `http://arxiv.org/abs/1210.1207`

[KL90] KHOTANZAD, Alireza ; LU, Jiin-Her: Classification of invariant image representations using a neural network. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 38 (1990), Nr. 6, S. 1028–1038. `http://dx.doi.org/10.1109/29.56063`. – DOI 10.1109/29.56063. – ISSN 0096–3518

[KMS08] KLASER, Alexander ; MARSZALEK, Marcin ; SCHMID, Cordelia: A Spatio-Temporal Descriptor Based on 3D-Gradients. In: *19th British Machine Vision Conference (BMVC)*. Leeds, United Kingdom : British Machine Vision Association, 2008

[Kot07] KOTSIANTIS, S. B.: Supervised Machine Learning: A Review of Classification Techniques. In: *Proceedings of the Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, IOS Press, 2007. – ISBN 978–1–58603–780–2, S. 3–24

[KS15] KĘPA, Marcin ; SZYMAŃSKI, Julian: Two Stage SVM and kNN Text Documents Classifier. In: *Pattern Recognition and Machine Intelligence*. Cham : Springer International Publishing, 2015. – ISBN 978–3–319–19941–2, S. 279–289

[KTS^{+}14] KARPATHY, A. ; TODERICI, G. ; SHETTY, S. ; LEUNG, T. ; SUKTHANKAR, R. ; FEI-FEI, L.: Large-Scale Video Classification with Convolutional Neural Networks. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. – ISSN 1063–6919, S. 1725–1732

[KVJ11] KRAPAC, Josip ; VERBEEK, Jakob ; JURIE, Frédéric: Modeling Spatial Layout with Fisher Vectors for Image Categorization. In: *International Conference on Computer Vision (ICCV)*. Barcelona, Spain : IEEE, November 2011, 1487-1494

[KZP06] KOTSIANTIS, S. B. ; ZAHARAKIS, I. D. ; PINTELAS, P. E.: Machine learning: a review of classification and combining techniques. In: *Artificial Intelligence Review* 26 (2006), Nov, Nr. 3, 159–190. `http://dx.doi.org/10.1007/s10462-007-9052-3`. – DOI 10.1007/s10462–007–9052–3. – ISSN 1573–7462

[KZP08] KELLOKUMPU, Vili ; ZHAO, Guoying ; PIETIKÄINEN, Matti: Human Activity Recognition Using a Dynamic Texture Based Method. In: *Procedings of the British Machine Vision Conference*, BMVA Press, 2008. – ISBN 1–901725–36–7, S. 88.1–88.10

[LAM17] LIU, Jain ; AKHTAR, Naveed ; MIAN, Ajmal: Viewpoint Invariant RGB-D Human Action Recognition. In: *International Conference on Digital Image Computing: Techniques and Applications (DICTA)* (2017), S. 1–8. `http://dx.doi.org/10.1109/DICTA.2017.8227505`. – DOI 10.1109/DICTA.2017.8227505

[Lap05] LAPTEV, Ivan: On space-time interest points. In: *International Journal of Computer Vision* 64 (2005), S. 107–123. `http://dx.doi.org/10.1007/s11263-005-1838-7`. – DOI 10.1007/s11263–005–1838–7. – ISBN 0–7695–1950–4

[Lat17] LATAH, Majd: Human action recognition using support vector machines and 3D convolutional neural networks. In: *International Journal of Advances in Intelligent Informatics* 3 (2017), Nr. 1, 47–55. `http://dx.doi.org/10.26555/ijain.v3i1.89`. – DOI 10.26555/ijain.v3i1.89. – ISSN 2548–3161

[LBBH98] LECUN, Y. ; BOTTOU, L. ; BENGIO, Y. ; HAFFNER, P.: Gradient-based learning applied to document recognition. In: *Proceedings of the IEEE* 86 (1998), Nov, Nr. 11, S. 2278–2324. `http://dx.doi.org/10.1109/5.726791`. – DOI 10.1109/5.726791. – ISSN 0018–9219

[LBOM12] LECUN, Yann A. ; BOTTOU, Léon ; ORR, Genevieve B. ; MÜLLER, Klaus-Robert: Efficient BackProp. In: *Neural Networks: Tricks of the Trade: Second Edition*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012. – ISBN 978–3–642–35289–8, 9–48

[LCX^{+}19] LIU, Zhiqiang ; CHOW, Paul ; XU, Jinwei ; JIANG, Jingfei ; DOU, Yong ; ZHOU, Jie: A Uniform Architecture Design for Accelerating 2D and 3D CNNs on FPGAs. In: *electronics* 8 (2019), Nr. 65. `http://dx.doi.org/10.3390/electronics8010065`. – DOI 10.3390/electronics8010065

[Lev46] LEVY, M. M.: Fourier transform analysis. In: *Journal of the British Institution of Radio Engineers* 6 (1946), Nr. 6, S. 228–246. `http://dx.doi.org/10.1049/jbire.1946.0033`. – DOI 10.1049/jbire.1946.0033

[Lin91] LINDEBERG, Tony: *Discrete scale-space theory and the scale-space primal sketch*. Stockholm, Ph.D. Dissertation, 1991. `http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-58570`

[LK81] LUCAS, Bruce D. ; KANADE, Takeo: An iterative image registration technique with an application to stereo vision. In: *International Joint Conference on Artificial Intelligence (IJCAI)*, 1981. – ISBN 0001–0782, S. 674–679

[LK99] LEE, Hyeon-Kyu ; KIM, J. H.: An HMM-based threshold model approach for gesture recognition. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21 (1999), Nr. 10, S. 961–973. `http://dx.doi.org/10.1109/34.799904`. – DOI 10.1109/34.799904. – ISSN 0162–8828

[LKF10] LECUN, Y. ; KAVUKCUOGLU, K. ; FARABET, C.: Convolutional networks and applications in vision. In: *Proceedings of IEEE International Symposium on Circuits and Systems*, 2010, S. 253–256

[LKH07] LINDNER, M. ; KOLB, A. ; HARTMANN, K.: Data-Fusion of PMD-Based Distance-Information and High-Resolution RGB-Images. In: *International Symposium on Signals, Circuits and Systems* Bd. 1, 2007, S. 1–4

[LMB$^+$05] LISIN, Dimitri A. ; MATTAR, Marwan A. ; BLASCHKO, Matthew B. ; LEARNED-MILLER, Erik G. ; BENFIELD, Mark C.: Combining Local and Global Image Features for Object Class Recognition. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* Bd. 3. Washington, DC, USA : IEEE Computer Society, 2005. – ISBN 0–7695–2372–2

[LMMR16] LLOYD, Kaelon ; MARSHALL, A. D. ; MOORE, Simon C. ; ROSIN, Paul L.: Detecting Violent Crowds using Temporal Analysis of GLCM Texture. In: *CoRR* abs/1605.05106 (2016). `http://arxiv.org/abs/1605.05106`

[LMSR08] LAPTEV, Ivan ; MARSZAŁEK, Marcin ; SCHMID, Cordelia ; ROZENFELD, Benjamin: Learning realistic human actions from movies. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. – ISBN 9781424422432, S. 1–8

[Low99] LOWE, David G.: Object recognition from local scale-invariant features. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision* 2 (1999), Nr. [8, 1150–1157. `http://dx.doi.org/10.1109/ICCV.1999.790410`. – DOI 10.1109/ICCV.1999.790410. – ISBN 0–7695–0164–8

[Low04] LOWE, David G.: Distinctive Image Features from Scale-Invariant Keypoints. In: *International Journal of Computer Vision* 60 (2004), Nr. 2, 91–110. `http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94`. – DOI 10.1023/B:VISI.0000029664.99615.94

[LPB11] LETOUZEY, Antoine ; PETIT, Benjamin ; BOYER, Edmond: Scene Flow from Depth and Color Images. In: *Procedings of the British Machine Vision Conference (Bmvc)*, 2011. – ISBN 190172543X, 46.1–46.11

[LPH$^+$17] LUO, Z. ; PENG, B. ; HUANG, D. ; ALAHI, A. ; FEI-FEI, L.: Unsupervised Learning of Long-Term Motion Dynamics for Videos. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. – ISSN 1063–6919, S. 7101–7110

[LS13] LIU, Li ; SHAO, Ling: Learning Discriminative Representations from RGB-D Video Data. In: *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, AAAI Press, 2013 (IJCAI '13). – ISBN 978–1–57735–633–2, 1493–1500

[LZWG18] LI, Z. ; ZHU, X. ; WANG, L. ; GUO, P.: Image Classification Using Convolutional Neural Networks and Kernel Extreme Learning Machines. In: *25th IEEE International Conference on Image Processing (ICIP)*, 2018, S. 3009–3013

[MA16] MAHJOUB, A. B. ; ATRI, M.: Human action recognition using RGB data. In: *2016 11th International Design Test Symposium (IDT)*, 2016, S. 83–87

[MAaMV15] MILANOVA, Mariofanna ; AL-ALI, Salim ; MANOLOVA, Agata ; VICTORIA, Fox: Human action recognition using combined contour-based and silhouette-based features and employing KNN or SVM classifier. In: *INTERNATIONAL JOURNAL OF COMPUTERS* 9 (2015), S. 37–47

[Mag20] MAGNANI, Antonio: *Human Action Recognition and Monitoring in Ambient Assisted Living Environments*, alma, Diss., Aprile 2020. `http://dx.doi.org/10.6092/unibo/amsdottorato/9373`. – DOI 10.6092/unibo/amsdottorato/9373

[MAL18] MUKHERJEE, Snehasis ; ANVITHA, Leburu ; LAHARI, T. M.: Human Activity Recognition in RGB-D Videos by Dynamic Images. In: *CoRR* abs/1807.02947 (2018). `http://arxiv.org/abs/1807.02947`

[MAQM05] MOKHBER, Arash ; ACHARD, Catherine ; QU, Xingtai ; MILGRAM, Maurice: Action Recognition with Global Features. In: *Computer Vision in Human-Computer Interaction*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2005. – ISBN 978–3–540–32129–3, S. 110–119

[Mar61] MARON, M. E.: Automatic Indexing: An Experimental Inquiry. In: *Journal of the ACM (JACM)* 8 (1961), Juli, Nr. 3, 404–417. `http://dx.doi.org/10.1145/321075.321084`. – DOI 10.1145/321075.321084. – ISSN 0004–5411

[MCCRAC16] MARTINS, Isabel ; CARVALHO, Pedro ; CORTE-REAL, Luís ; ALBA-CASTRO, José Luis: Bio-inspired Boosting for Moving Objects Segmentation. In: *Image Analysis and Recognition*. Cham : Springer International Publishing, 2016. – ISBN 978–3–319–41501–7, S. 397–406

[MDDB15] MENG MENG ; DRIRA, Hassen ; DAOUDI, Mohamed ; BOONAERT, Jacques: Human-object interaction recognition by learning the distances between the object and the skeleton joints. In: *11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)* (2015), 1–6. `http://dx.doi.org/10.1109/FG.2015.7284883`. – DOI 10.1109/FG.2015.7284883. ISBN 978–1–4799–6026–2

[MH17] MIN HUANG, Guo-Rong Cai-Hong-Bo Zhang-Donglin Cao Shao-Zi L. Song-Zhi Su S. Song-Zhi Su: Meta-action descriptor for action recognition in RGBD video. In: *IET Computer Vision* 11 (2017), Nr. 4, 301–308. `http://dx.doi.org/10.1049/iet-cvi.2016.0252`. – DOI 10.1049/iet–cvi.2016.0252. – ISSN 1751–9632

[MP17] MURAD, Abdulmajid ; PYUN, Jae-Young: Deep recurrent neural networks for human activity recognition. In: *Sensors* 17 (2017), Nr. 11, S. 2556

[MPC19] MENDES, Paulo A. S. ; PAULO COIMBRA, A.: Movement Detection and Moving Object Distinction Based on Optical Flow for a Surveillance System. In: *Transactions on Engineering Technologies*. Singapore : Springer Singapore, 2019. – ISBN 978–981–15–8273–8, S. 143–158

[MR14] MOHAN, A. S. ; RESMI, R.: Video image processing for moving object detection and segmentation using background subtraction. In: *First International Conference on Computational Systems and Communications (ICCSC)*, 2014, S. 288–292

[MRS08] MANNING, C. D. ; RAGHAVAN, P. ; SCHÜTZE, H.: Introduction to Information Retrieval. In: *Computational Linguistics* Bd. 35, 2008. – ISBN 978–0–521–86571–5, S. 482

[MS02] MIKOLAJCZYK, Krystian ; SCHMID, Cordelia: An Affine Invariant Interest Point Detector. In: *Computer Vision — ECCV 2002*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2002. – ISBN 978–3–540–47969–7, S. 128–142

[MSM15] MAHESH, Yashaswini ; SHIVAKUMAR, Dr. M. ; MOHANA, Dr. H. S.: Classification of human actions using non-linear svm by extracting spatio temporal hog features with custom dataset. In: *International Journal of Research In Science & Engineering (IJRISE)* (2015), S. 1–6

[MWA19] MYO, W. W. ; WETTAYAPRASIT, W. ; AIYARAK, P.: Designing Classifier For Human Activity Recognition Using Artificial Neural Network. In: *IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*, 2019, S. 81–85

[MYJ+17] MUCHTAR, K. ; YEH, C. ; JIAN, Z. ; LIN, C. ; LIN, W. ; HUANG, W.: Moving object detection based on image bit-planes and co-occurrence matrix in video surveillance. In: *IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW)*, 2017, S. 1–2

[NJ14] NIE, S. ; JI, Q.: Capturing Global and Local Dynamics for Human Action Recognition. In: *22nd International Conference on Pattern Recognition (ICPR)*, 2014. – ISSN 1051–4651, S. 1946–1951

[NVBR12] NOURANI-VATANI, Navid ; BORGES, Paulo V K. ; ROBERTS, Jonathan M.: A study of feature extraction algorithms for optical flow tracking. In: *Proceedings of Australasian Conference on Robotics and Automation, Victoria University of Wellington, New Zealand*, 2012. – ISBN 9780980740431, S. 1–7

[NVSH16] NGOC, Ly Q. ; VIET, Vo H. ; SON, Tran T. ; HOANG, Pham M.: A Robust Approach for Action Recognition Based on Spatio-Temporal Features in RGB-D Sequences. In: *International Journal of Advanced Computer Science and Applications* 7 (2016), Nr. 5. `http://dx.doi.org/10.14569/IJACSA.2016.070526`. – DOI 10.14569/IJACSA.2016.070526

[NVY+14] NALLAMOTHU, Rahul ; VINEETH, T. ; YADAV, Gaurav K. ; SETHI, Amit ; JACOB, Tony: Interest Point Detection in Videos Using Long Point Trajectories. In: *ACM International Conference Proceeding Series* Bd. 14, Association for Computing Machinery, 2014. – ISBN 9781450330619, S. 0–4

[NWJ17] NIE, Siqi ; WANG, Ziheng ; JI, Qiang: A Generative Restricted Boltzmann Machine Based Method for High-Dimensional Motion Data Modeling. In: *CoRR* (2017), 1–23. `http://dx.doi.org/10.1016/j.cviu.2014.12.005`. – DOI 10.1016/j.cviu.2014.12.005

[NY10] NOGUCHI, Akitsugu ; YANAI, Keiji: Extracting Spatio-temporal Local Features Considering Consecutiveness of Motions. In: *Computer Vision – ACCV 2009*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2010. – ISBN 978–3–642–12304–7, S. 458–467

[OMI12] OONG, Tatt H. ; MAT ISA, Nor A.: One-against-all Ensemble for Multiclass Pattern Classification. In: *Applied Soft Computing* 12 (2012), Nr. 4, S. 1303–1308. `http://dx.doi.org/10.1016/j.asoc.2011.12.004`. – DOI 10.1016/j.asoc.2011.12.004. – ISSN 1568–4946

[OPH96] OJALA, Timo ; PIETIKÄINEN, Matti ; HARWOOD, David: A comparative study of texture measures with classification based on feature distributions. In: *Pattern Recognition* 29 (1996), 51–59. `https://doi.org/10.1016/0031-3203(95)00067-4`

[OPM02] OJALA, Timo ; PIETIKÄINEN, Matti ; MÄENPÄÄ, Topi: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), Nr. 7, S. 971–987. `http://dx.doi.org/10.1109/TPAMI.2002.1017623`. – DOI 10.1109/TPAMI.2002.1017623. – ISBN 0162–8828

[ORK$^+$15] OVTCHAROV, Kalin ; RUWASE, Olatunji ; KIM, Joo-Young ; FOWERS, Jeremy ; STRAUSS, Karin ; CHUNG, Eric S.: Accelerating deep convolutional neural networks using specialized hardware. In: *Microsoft Research Whitepaper* 2 (2015), Nr. 11, S. 1–4

[OT01] OLIVA, Aude ; TORRALBA, Antonio: Modeling the shape of the scene: A holistic representation of the spatial envelope. In: *International Journal of Computer Vision* 42 (2001), Nr. 3, S. 145–175. `http://dx.doi.org/10.1023/A:1011139631724`. – DOI 10.1023/A:1011139631724. – ISBN 0920–5691

[Ots79] OTSU, N.: A Threshold Selection Method from Gray-Level Histograms. In: *IEEE Transactions on Systems, Man, and Cybernetics* 9 (1979), Nr. 1, S. 62–66. `http://dx.doi.org/10.1109/TSMC.1979.4310076`. – DOI 10.1109/TSMC.1979.4310076

[Pap20] PAPADOPOULOS, Konstantinos m.: *From Dense 2D to Sparse 3D Trajectories for Human Action Detection and Recognition*, Doctoral thesis, Apr-2020

[PD07] PERRONNIN, F ; DANCE, C: Fisher Kenrels on Visual Vocabularies for Image Categorizaton. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. – ISBN 1424411807

[PDKS15] PIGOU, Lionel ; DIELEMAN, Sander ; KINDERMANS, Pieter-Jan ; SCHRAUWEN, Benjamin: Sign Language Recognition Using Convolutional Neural Networks. In: *Computer Vision - ECCV 2014 Workshops*. Cham : Springer International Publishing, 2015. – ISBN 978–3–319–16178–5, S. 572–578

[Pop10] POPPE, Ronald: A Survey on Vision-Based Human Action Recognition. In: *Image Vision Computing* 28 (2010), jun, Nr. 6, 976–990. `http://dx.doi.org/10.1016/j.imavis.2009.11.014`. – DOI 10.1016/j.imavis.2009.11.014. – ISSN 0262–8856

[PS15] PRATAMA, Bayu Y. ; SARNO, Riyanarto: Personality classification based on Twitter text using Naive Bayes, KNN and SVM. In: *International Conference on Data and Software Engineering (ICoDSE)*, 2015. – ISBN 9781467384285, S. 170–174

[PSL09] PEREIRA, Rui ; SEABRA LOPES, Luis: Learning Visual Object Categories with Global Descriptors and Local Features. In: *Progress in Artificial Intelligence*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2009. – ISBN 978–3–642–04686–5, S. 225–236

[PTDZ16] PEREZ, Alexandre ; TABIA, Hedi ; DECLERCQ, David ; ZANOTTI, Alain: Feature covariance for human action recognition. In: *Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2016. – ISSN 2154–512X, S. 1–5

[QMXW10] QIAN, Huimin ; MAO, Yaobin ; XIANG, Wenbo ; WANG, Zhiquan: Recognition of human activities using SVM multi-class classifier. In: *Pattern Recognit. Lett.* 31 (2010), S. 100–111. `http://dx.doi.org/10.1016/j.patrec.2009.09.019`. – DOI 10.1016/j.patrec.2009.09.019

[RASC14] Razavian, A. S. ; Azizpour, H. ; Sullivan, J. ; Carlsson, S.: CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, S. 512–519

[RB93] Riedmiller, M. ; Braun, H.: A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: *IEEE International Conference on Neural Networks*, 1993, S. 586–591 vol.1

[RDS$^+$15] Russakovsky, Olga ; Deng, Jia ; Su, Hao ; Krause, Jonathan ; Satheesh, Sanjeev ; Ma, Sean ; Huang, Zhiheng ; Karpathy, Andrej ; Khosla, Aditya ; Bernstein, Michael ; Berg, Alexander C. ; Fei-Fei, Li: ImageNet Large Scale Visual Recognition Challenge. In: *International Journal of Computer Vision (IJCV)* 115 (2015), Nr. 3, S. 211–252. `http://dx.doi.org/10.1007/s11263-015-0816-y`. – DOI 10.1007/s11263–015–0816–y

[RF16] Radolko, M. ; Farhadifard, F.: Using trajectories derived by dense optical flows as a spatial component in background subtraction. In: *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, WSCG*, 2016. – ISBN 978–80–86943–57–2

[RGF$^+$18] Rueda, Fernando M. ; Grzeszick, Rene ; Fink, Gernot A. ; Feldhorst, Sascha ; Hompel, Michael ten: Convolutional Neural Networks for Human Activity Recognition Using Body-Worn Sensors. In: *informatics* 26 (2018), Nr. 5. `https://doi.org/10.3390/informatics5020026`

[RHBL07] Ranzato, M. ; Huang, F. J. ; Boureau, Y. ; LeCun, Y.: Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition. In: *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007. – ISSN 1063–6919, S. 1–8

[RM05] Rokach, L. ; Maimon, O.: Top-down Induction of Decision Trees Classifiers-a Survey. In: *Trans. Sys. Man Cyber Part C* 35 (2005), November, Nr. 4, 476–487. `http://dx.doi.org/10.1109/TSMCC.2004.843247`. – DOI 10.1109/TSMCC.2004.843247. – ISSN 1094–6977

[ROD01] Richard O. Duda, Peter E. Hart-David G. S. Peter Elliot Hart H. Peter Elliot Hart: *Pattern Classification*. Wiley, 2001 (A Wiley-interscience publication pt. 1). `https://books.google.de/books?id=YoxQAAAAMAAJ`. – ISBN 9780471056690

[RRK$^+$90] Ruck, D. W. ; Rogers, S. K. ; Kabrisky, M. ; Oxley, M. E. ; Suter, B. W.: The multilayer perceptron as an approximation to a Bayes optimal discriminant function. In: *IEEE Transactions on Neural Networks* 1 (1990), Dec, Nr. 4, S. 296–298. `http://dx.doi.org/10.1109/72.80266`. – DOI 10.1109/72.80266. – ISSN 1045–9227

[RS99] Rosales, Rómer ; Sclaroff, Stan: 3D Trajectory Recovery for Tracking Multiple Objects and Trajectory Guided Recognition of Actions. In: *In Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999, S. 117–123

[RSv05] Ribeiro, Pedro C. ; Santos-victor, Jose: Human activity recognition from video: modeling, feature selection and classification architecture. In: *International Workshop on Human Activity Recognition and Modeling (HAREM)* Citeseer, 2005, S. 61–78

[RSZ13] RAMPUN, Andrik ; STRANGE, Harry ; ZWIGGELAAR, Reyer: Texture Segmentation Using Different Orientations of GLCM Features. In: *Proceedings of the 6th International Conference on Computer Vision / Computer Graphics Collaboration Techniques and Applications*. New York, NY, USA : ACM, 2013. – ISBN 9781450320238, 1

[RTKI11] RAHMAN AHAD, Md A. ; TAN, J. K. ; KIM, H. ; ISHIKAWA, S.: SURF-based spatio-temporal history image method for action representation. In: *Proceedings of the IEEE International Conference on Industrial Technology* (2011), S. 411–416. `http://dx.doi.org/10.1109/ICIT.2011.5754412`. – DOI 10.1109/ICIT.2011.5754412. ISBN 9781424490660

[RYS+06] RIZON, Mohamed ; YAZID, Haniza ; SAAD, Puteh ; SHAKAFF, Ali Y. ; SAAD, Abdul R. ; MAMAT, Mohd R. ; YAACOB, Sazali ; DESA, Hazri ; KARTHIGAYAN, M: Object Detection using Geometric Invariant Moment. In: *American Journal of Applied Sciences* 2 (2006), Nr. 6, S. 1876–1878. `http://dx.doi.org/10.3844/ajassp.2006.1876.1878`. – DOI 10.3844/ajassp.2006.1876.1878. – ISBN 1546–9239

[SA11] SPINELLO, L. ; ARRAS, K. O.: People detection in RGB-D data. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, S. 3838–3843

[SA14] SHAN, Junjie ; AKELLA, Srinivas: 3D human action segmentation and recognition using pose kinetic energy. In: *IEEE International Workshop on Advanced Robotics and its Social Impacts*, 2014, 69–75

[SAH17] SARGANO, Allah ; ANGELOV, Plamen ; HABIB, Zulfiqar: A Comprehensive Review on Handcrafted and Learning-Based Action Representation Approaches for Human Activity Recognition. In: *Applied Sciences* 7 (2017), Nr. 1, S. 110. `http://dx.doi.org/10.3390/app7010110`. – DOI 10.3390/app7010110. – ISSN 2076–3417

[SAS13] SOLMAZ, Berkan ; ASSARI, Shayan M. ; SHAH, Mubarak: Classifying web videos using a global video descriptor. In: *Machine Vision and Applications* 24 (2013), Nr. 7, S. 1473–1485. `http://dx.doi.org/10.1007/s00138-012-0449-x`. – DOI 10.1007/s00138–012–0449–x

[SBL14] STRAT, Sabin ; BENOIT, AlexLi2010andre ; LAMBERT, Patrick: Retina enhanced bag of words descriptors for video classification. In: *22nd European Signal Processing Conference (EUSIPCO)*, 2014. – ISBN 978–0–9928–6261–9, S. 1307–1311

[SBLC12] STRAT, S. T. ; BENOIT, A. ; LAMBERT, P. ; CAPLIER, A.: Retina-enhanced SURF descriptors for semantic concept detection in videos. In: *3rd International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2012. – ISSN 2154–512X, S. 319–324

[SBS13] SIKIRIC, Ivan ; BRKIC, Karla ; SEGVIC, Sinisa: Classifying Traffic Scenes Using The GIST Image Descriptor. In: *CoRR* abs/1310.0316 (2013), 19–24. `http://arxiv.org/abs/1310.0316`

[SC16] SUBETHA, T. ; CHITRAKALA, S.: A survey on human activity recognition from videos. In: *International Conference on Information Communication and Embedded Systems (ICICES)*, 2016, S. 1–7

[SCH09] SUN, Xinghua ; CHEN, Ming yu ; HAUPTMANN, Alexander G.: Action recognition via local descriptors and holistic features. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* (2009), S. 58–65. http://dx.doi.org/10.1109/CVPRW.2009.5204255. – DOI 10.1109/CVPRW.2009.5204255

[SCMP14] SOMASUNDARAM, Guruprasad ; CHERIAN, Anoop ; MORELLAS, Vassilios ; PAPANIKOLOPOULOS, Nikolaos: Action recognition using global spatio-temporal features derived from sparse representations. In: *Computer Vision and Image Understanding* 123 (2014), 1–13. http://dx.doi.org/10.1016/j.cviu.2014.01.002. – DOI 10.1016/j.cviu.2014.01.002. – ISSN 1090235X

[SEE18] SEEMANTHINI, Manjunath S. S. K.: Human Detection and Tracking using HOG for Action Recognition. In: *In Procedia Computer Science, Elsevier B.V.* 132 (2018), Nr. 2018, S. 1317–1326. http://dx.doi.org/10.1016/j.procs.2018.05.048. – DOI 10.1016/j.procs.2018.05.048

[SG13] SATHYA, R. ; GEETHA, M.: Vision based Traffic Police Hand Signal Recognition in Surveillance Video - A Survey. In: *International Journal of Computer Applications* 81 (2013), 11, S. 1–10. http://dx.doi.org/10.5120/14037-2192. – DOI 10.5120/14037–2192

[Sha48] SHANNON, C. E.: A Mathematical Theory of Communication. In: *Bell System Technical Journal* 27 (1948), Nr. 3, 379-423. http://dx.doi.org/10.1002/j.1538-7305.1948.tb01338.x. – DOI 10.1002/j.1538–7305.1948.tb01338.x

[SHJ18] SONG, Xinhang ; HERRANZ, Luis ; JIANG, Shuqiang: Depth CNNs for RGB-D scene recognition: learning from scratch better than transferring from RGB-CNNs. In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, CoRR* abs/1801.06797 (2018)

[SJ10] SHAO, Ling ; JI, Ling: A Descriptor Combining MHI and PCOG for Human Motion Classification. In: *Proceedings of the ACM International Conference on Image and Video Retrieval.* New York, NY, USA : ACM, 2010, 236–242

[SJP13] In: SMISEK, Jan ; JANCOSEK, Michal ; PAJDLA, Tomas: *3D with Kinect.* 2013. – ISBN 978–1–4471–4639–1, S. 3–25

[SL15] SONG, Yan ; LIN, Yang: Combining RGB and Depth Features for Action Recognition based on Sparse Representation. In: *Proceedings of the 7th International Conference on Internet Multimedia Computing and Service*, 2015 (ICIMCS '15). – ISBN 9781450335287

[SLY16] SHAO, Ling ; LIU, Li ; YU, Mengyang: Kernelized Multiview Projection for Robust Action Recognition. In: *International Journal of Computer Vision* 118 (2016), Nr. 2, S. 115–129. http://dx.doi.org/10.1007/s11263-015-0861-6. – DOI 10.1007/s11263–015–0861–6. – ISSN 15731405

[SNGW16] SHAHROUDY, Amir ; NG, Tian-Tsong ; GONG, Yihong ; WANG, Gang: NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, S. 1010–1019

[SPSS11] SUNG, Jaeyong ; PONCE, Colin ; SELMAN, Bart ; SAXENA, Ashutosh: Human Activity Detection from RGBD Images. In: *Proceedings of the 16th AAAI Conference on Plan, Activity, and Intent Recognition*, AAAI Press, 2011 (AAAIWS'11-16), 47–55

[SPSS12] SUNG, Jaeyong ; PONCE, Colin ; SELMAN, Bart ; SAXENA, Ashutosh: Unstructured human activity detection from RGBD images. In: *IEEE International Conference on Robotics and Automation*, 2012. – ISBN 9781467314039, S. 842–849

[SSH16] SEBESTYEN, G. ; STOICA, I. ; HANGAN, A.: Human activity recognition and monitoring for elderly people. In: *IEEE 12th International Conference on Intelligent Computer Communication and Processing (ICCP)*, 2016, S. 341–347

[SW17] In: SAMMUT, Claude ; WEBB, Geoffrey I.: *Artificial Neural Networks*. Boston, MA : Springer US, 2017. – ISBN 978–1–4899–7687–1, S. 65–66

[SZ14] SIMONYAN, Karen ; ZISSERMAN, Andrew: Two-stream Convolutional Networks for Action Recognition in Videos. In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*. Cambridge, MA, USA : MIT Press, 2014 (NIPS'14), 568–576

[TCL15] TSAI, Du-ming ; CHIU, Wei-yao ; LEE, Men-han: Optical flow-motion history image (OF-MHI) for action recognition. In: *Signal, Image Video Processing* 9 (2015), Nr. 8, 1897–1906. `http://dx.doi.org/10.1007/s11760-014-0677-9`. – DOI 10.1007/s11760–014–0677–9. – ISSN 1863–1703

[TCLY15] TAO, Dapeng ; CHENG, Jun ; LIN, Xu ; YU, Jiang: Local Structure Preserving Discriminative Projections for RGB-D Sensor-Based Scene Classification. In: *Inf. Sci.* 320 (2015), November, Nr. C, 383–394. `http://dx.doi.org/10.1016/j.ins.2015.03.031`. – DOI 10.1016/j.ins.2015.03.031. – ISSN 0020–0255

[TCLZ12] TIAN, Yingli ; CAO, Liangliang ; LIU, Zicheng ; ZHANG, Zhengyou: Hierarchical filtered motion for action recognition in crowded videos. In: *IEEE Transactions on Systems, Man and Cybernetics Part C (Applications and Reviews)* 42 (2012), Nr. 3, S. 313–323. `http://dx.doi.org/10.1109/TSMCC.2011.2149519`. – DOI 10.1109/TSMCC.2011.2149519. – ISBN 1094–6977

[TDL15] TUVSHINJARGAL, Doopalam ; DORJ, Byambaa ; LEE, Deok-Jin: Hybrid Motion Planning Method for Autonomous Robots Using Kinect Based Sensor Fusion and Virtual Plane Approach in Dynamic Environments. In: *J. Sensors* 2015 (2015), S. 471052:1–471052:13. `http://dx.doi.org/10.1155/2015/471052`. – DOI 10.1155/2015/471052

[TFLB10] TAYLOR, Graham W. ; FERGUS, Rob ; LECUN, Yann ; BREGLER, Christoph: Convolutional Learning of Spatio-temporal Features. In: *Computer Vision – ECCV*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2010. – ISBN 978–3–642–15567–3, S. 140–153

[TJYL13] TAO, D. ; JIN, L. ; YANG, Z. ; LI, X.: Rank Preserving Sparse Learning for Kinect Based Scene Classification. In: *IEEE Transactions on Cybernetics* 43 (2013), Nr. 5, S. 1406–1417. `http://dx.doi.org/10.1109/TCYB.2013.2264285`. – DOI 10.1109/TCYB.2013.2264285

[TK12] TEUTSCH, M. ; KRÜGER, W.: Detection, Segmentation, and Tracking of Moving Objects in UAV Videos. In: *IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*, 2012, S. 313–318

[TM17] TALUKDAR, J. ; MEHTA, B.: Human action recognition system using good features and multilayer perceptron network. In: *International Conference on Communication and Signal Processing (ICCSP)*, 2017, S. 0317–0323

[TMFR03] TORRALBA, Antonio ; MURPHY, Kevin P. ; FREEMAN, William T. ; RUBIN, Mark A.: Context-based vision system for place and object recognition. In: *Proceedings Ninth IEEE International Conference on Computer Vision* Bd. 1, 2003, S. 273–280

[Tub17] TUBA, Tuba Milan Simian D. Eva: Support Vector Machine Optimized by Firefly Algorithm for Emphysema Classification in Lung Tissue CT Images. In: *25. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision WSCG*, 2017. – ISBN ISBN 978–80–86943–50–3

[TZEH15] TAHA, Ahmed ; ZAYED, M. E. K. Hala ; EL-HORBATY, El-Sayed M.: Human Activity Recognition for Surveillance Applications. In: *7th International Conference on Information Technology*, 2015, S. 577—-586

[UDFS14] UMAKANTHAN, S. ; DENMAN, S. ; FOOKES, C. ; SRIDHARAN, S.: Activity recognition using binary tree SVM. In: *IEEE Workshop on Statistical Signal Processing (SSP)*, 2014. – ISSN 2373–0803, S. 248–251

[UDSS15] UIJLINGS, J ; DUTA, I C. ; SANGINETO, E ; SEBE, Nicu: Video classification with Densely extracted HOG/HOF/MBH features: an evaluation of the accuracy/computational efficiency trade-off. In: *International Journal of Multimedia Information Retrieval, Springer-Verlag* 4 (2015), Nr. 1, 33–44. `http://dx.doi.org/10.1007/s13735-014-0069-5`. – DOI 10.1007/s13735–014–0069–5. – ISSN 2192–6611

[UL19] UDDIN, Md A. ; LEE, Young-Koo: Feature fusion of deep spatial features and handcrafted spatiotemporal features for human action recognition. In: *Sensors* 19 (2019), Nr. 7, S. 1599. `http://dx.doi.org/10.3390/s19071599`. – DOI 10.3390/s19071599

[Vap95] VAPNIK, Vladimir N.: Berlin, Heidelberg : Springer-Verlag, 1995. `http://dx.doi.org/10.1007/978-1-4757-2440-0`. `http://dx.doi.org/10.1007/978-1-4757-2440-0`. – ISBN 0387945598

[VHS15] VARGA, Domonkos ; HAVASI, László ; SZIRÁNYI, Tamás: Pedestrian detection in surveillance videos based on CS-LBP feature. In: *International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, 2015. – ISBN 9789633131428, S. 413–417

[VJ01] VIOLA, P. ; JONES, M.: Rapid object detection using a boosted cascade of simple features. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. (CVPR)* Bd. 1, 2001. – ISSN 1063–6919

[VMS$^+$19] VERNIKOS, I. ; MATHE, E. ; SPYROU, E. ; MITSOU, A. ; GIANNAKOPOULOS, T. ; MYLONAS, P.: Fusing Handcrafted and Contextual Features for Human Activity Recognition. In: *14th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP)*, 2019, S. 1–6

[VNK15] VRIGKAS, Michalis ; NIKOU, Christophoros ; KAKADIARIS, Ioannis A.: A Review of Human Activity Recognition Methods. In: *Front. Robotics and AI* 2 (2015), Nr. 28. `http://dx.doi.org/10.3389/frobt.2015.00028`. – DOI 10.3389/frobt.2015.00028

[vO03] ŠÍMA, Jiří ; ORPONEN, Pekka: General-Purpose Computation with Neural Networks: A Survey of Complexity Theoretic Results. In: *Neural Comput.* 15 (2003), Dezember, Nr. 12, 2727–2778. `http://dx.doi.org/10.1162/089976603322518731`. – DOI 10.1162/089976603322518731. – ISSN 0899–7667

[VVV16] VYAS, Kushal ; VORA, Yash ; VASTANI, Raj: Using Bag of Visual Words and Spatial Pyramid Matching for Object Classification Along with Applications for RIS. In: *Procedia Computer Science* 89 (2016), 457–464. `http://dx.doi.org/10.1016/j.procs.2016.06.102`. – DOI 10.1016/j.procs.2016.06.102. – ISBN 9619189124

[WKRQ+08] WU, Xindong ; KUMAR, Vipin ; ROSS QUINLAN, J. ; GHOSH, Joydeep ; YANG, Qiang ; MOTODA, Hiroshi ; MCLACHLAN, Geoffrey J. ; NG, Angus ; LIU, Bing ; YU, Philip S. ; ZHOU, Zhi-Hua ; STEINBACH, Michael ; HAND, David J. ; STEINBERG, Dan: Top 10 algorithms in data mining. In: *Knowledge and Information Systems* 14 (2008), Jan, Nr. 1, 1–37. `http://dx.doi.org/10.1007/s10115-007-0114-2`. – DOI 10.1007/s10115–007–0114–2. – ISSN 0219–3116

[WKSL11] WANG, H. ; KLÄSER, A. ; SCHMID, C. ; LIU, C.: Action Recognition by Dense Trajectories. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, S. 3169 – 3176

[WKSL13] WANG, Heng ; KLÄSER, Alexander ; SCHMID, Cordelia ; LIU, Cheng-Lin: Dense Trajectories and Motion Boundary Descriptors for Action Recognition. In: *Int J Comput Vis (2013) 103:60–79, Springer Science+Business Media New York 2013.* 103 (2013), Nr. 1, S. 60–79. `http://dx.doi.org/10.1007/s11263-012-0594-8`. – DOI 10.1007/s11263–012–0594–8. ISBN 1126301205

[WL11] WILL, What ; LEARN, W E.: Feature Extraction and Representation. Version: 2011. `http://dx.doi.org/10.1002/9781118093467.ch18`. In: *Practical Image and Video Processing Using MATLAB®*. 2011. – DOI 10.1002/9781118093467.ch18. – ISBN 9781118093467, 447–474

[WLG+16] WANG, P. ; LI, W. ; GAO, Z. ; ZHANG, J. ; TANG, C. ; OGUNBONA, P. O.: Action Recognition From Depth Maps Using Deep Convolutional Neural Networks. In: *IEEE Transactions on Human-Machine Systems* 46 (2016), Aug, Nr. 4, S. 498–509. `http://dx.doi.org/10.1109/THMS.2015.2504550`. – DOI 10.1109/THMS.2015.2504550. – ISSN 2168–2291

[WLJ13] WANG, Yangyang ; LI, Yibo ; JI, Xiaofei: Recognizing Human Actions Based on Gist Descriptor and Word Phrase. In: *Proceedings of International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC)* (2013), 1104–1107. `http://dx.doi.org/10.1109/MEC.2013.6885227`. – DOI 10.1109/MEC.2013.6885227. ISBN 9781479925650

[WLJ14] WANG, Yangyang ; LI, Yibo ; JI, Xiaofei: Human Action Recognition Using Compact Global Descriptors Derived from 2DPCA-2DLDA. In: *Proceedings of IEEE International Conference on Computer and Information Technology (CIT)*, 2014. – ISBN 9781479962389, S. 182–186

[WLJ15] WANG, Yangyang ; LI, Yibo ; JI, Xiaofei: Human Action Recognition Based on Global Gist Feature and Local Patch Coding. In: *International Journal of Signal Processing, Image Processing and Pattern Recognition* 8 (2015), Nr. 2, S. 235–246. `http://dx.doi.org/10.14257/ijsip.2015.8.2.23`. – DOI 10.14257/ijsip.2015.8.2.23

[WLJL15] WANG, Yangyang ; LI, Yibo ; JI, Xiaofei ; LIU, Yang: Comparison of Grid-Based Dense Representations for Action Recognition. In: *Intelligent Robotics and Applications, Springer International Publishing*. Cham : Springer International Publishing, 2015. – ISBN 978–3–319–22879–2, S. 435–444

[WLWY12] WANG, Jiang ; LIU, Zicheng ; WU, Ying ; YUAN, Junsong: Mining Actionlet Ensemble for Action Recognition with Depth Cameras. In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. – ISBN 9781467312288, 1290–1297

[WLWY14] WANG, Jiang ; LIU, Zicheng ; WU, Ying ; YUAN, Junsong: Learning actionlet ensemble for 3D human action recognition. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36 (2014), Nr. 5, S. 914–927. `http://dx.doi.org/10.1109/TPAMI.2013.198`. – DOI 10.1109/TPAMI.2013.198. – ISBN 9781467312264

[WM10] WANG, Yang ; MORI, Greg: Hidden Part Models for Human Action Recognition: Probabilistic vs. Max-Margin. In: *IEEE transactions on pattern analysis and machine intelligence* 33 (2010), Nr. 7, 1310–1323. `http://dx.doi.org/10.1109/TPAMI.2010.214`. – DOI 10.1109/TPAMI.2010.214. – ISBN 2009070453

[WMC16] WANG, X. ; MA, H. ; CHEN, X.: Salient object detection via fast R-CNN and low-level cues. In: *2016 IEEE International Conference on Image Processing (ICIP)*, 2016. – ISSN 2381–8549, S. 1042–1046

[WRLD13] WAN, Jun ; RUAN, Q ; LI, W ; DENG, S: One-shot learning gesture recognition from RGB-D data using bag of features. In: *Journal of Machine Learning Research* 14 (2013), 2549–2582. `http://jmlr.org/papers/v14/wan13a.html`. – ISSN 15324435

[WS07] WANG, Liang ; SUTER, David: Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model. In: *EEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (2007). `http://dx.doi.org/10.1109/CVPR.2007.383298`. – DOI 10.1109/CVPR.2007.383298. – ISBN 1424411807

[WS13] WANG, Heng ; SCHMID, Cordelia: Action Recognition with Improved Trajectories. In: *IEEE International Conference on Computer Vision (ICCV)*, 2013. – ISBN 978–1–4799–2840–8, S. 3551–3558

[WZC14] WANG, Yu ; ZHAO, Yongsheng ; CHEN, Yi: Texture classification using rotation invariant models on integrated local binary pattern and Zernike moments. In: *EURASIP Journal on Advances in Signal Processing* 2014 (2014), Nr. 1, 182. `https://doi.org/10.1186/1687-6180-2014-182`. – ISSN 1687–6180

[XA13] XIA, L. ; AGGARWAL, J. K.: Spatio-temporal Depth Cuboid Similarity Feature for Activity Recognition Using Depth Camera. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, S. 2834–2841

[XGAR15] XIA, L. ; GORI, I. ; AGGARWAL, J. K. ; RYOO, M. S.: Robot-centric Activity Recognition from First-Person RGB-D Videos. In: *IEEE Winter Conference on Applications of Computer Vision*, 2015. – ISSN 1550–5790, S. 357–364

[XZYT14] XIAO, Yang ; ZHAO, Gangqiang ; YUAN, Junsong ; THALMANN, Daniel: Activity Recognition in Unconstrained RGB-D Video Using 3D Trajectories. In:

SIGGRAPH Asia Autonomous Virtual Humans and Social Robot for Telepresence. New York, NY, USA : Association for Computing Machinery, 2014 (SA '14). – ISBN 9781450332439

[YB98] YACOOB, Y. ; BLACK, M. J.: Parameterized modeling and recognition of activities. In: *Sixth International Conference on Computer Vision (ICCV'98)*. Mumbai, India, january 1998, S. 120–127

[YCBL14] YOSINSKI, Jason ; CLUNE, Jeff ; BENGIO, Yoshua ; LIPSON, Hod: How Transferable Are Features in Deep Neural Networks? In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. Cambridge, MA, USA : MIT Press, 2014 (NIPS'14), 3320–3328

[YCXL17] YU, Sheng ; CHENG, Yun ; XIE, Li ; LI, Shao-Zi: Fully convolutional networks for action recognition. In: *IET Computer Vision* 11 (2017), Nr. 8, 744–749. `http://dx.doi.org/10.1049/iet-cvi.2017.0005`. – DOI 10.1049/iet–cvi.2017.0005. – ISSN 1751–9632

[YLM14] YURUR, O. ; LIU, C. H. ; MORENO, W.: A survey of context-aware middleware designs for human activity recognition. In: *IEEE Communications Magazine* 52 (2014), Nr. 6, S. 24–31. `http://dx.doi.org/10.1109/MCOM.2014.6829941`. – DOI 10.1109/MCOM.2014.6829941

[YLY$^+$13] YAO, Yukai ; LIU, Yang ; YU, Yongqing ; XU, Hong ; LV, Weiming ; LI, Zhao ; CHEN, Xiaoyun: K-SVM: An effective SVM algorithm based on K-means clustering. In: *Journal of Computers* 8 (2013), Nr. 10, S. 2632–2639. `http://dx.doi.org/10.4304/jcp.8.10.2632-2639`. – DOI 10.4304/jcp.8.10.2632–2639. – ISSN 1796203X

[YLY15] YU, Gang ; LIU, Zicheng ; YUAN, Junsong: Discriminative Orderlet Mining for Real-Time Recognition of Human-Object Interaction. In: *Computer Vision – ACCV 2014*, Springer International Publishing, 2015. – ISBN 978–3–319–16814–2, S. 50–65

[YS16] YADAV, S. ; SHUKLA, S.: Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification. In: *IEEE 6th International Conference on Advanced Computing (IACC)*, 2016, S. 78–83

[YT14] YANG, Xiaodong ; TIAN, Yingli: Super Normal Vector for Action Recognition Using Depth Sequences. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. – ISSN 1063–6919, S. 804–811

[YTYC12] YANG, Xiaodong ; TIAN, Yingli ; YI, Chucai ; CAO, Liangliang: MediaCCNY at TRECVID 2012: Surveillance event detection. In: *NIST TRECVID, Workshop*, 2012

[ZA19] ZAHANGIR ALOM, Chris Yakopcic Stefan Westberg-Vasit Sagan Mst Shamima Nasrin-Mahmudul Hasan Brian C. Van Essen Abdul A. S. Awwal Vijayan K. A. Tarek M. Taha T. Tarek M. Taha: A State-of-the-Art Survey on Deep Learning Theory and Architectures. In: *Electronics* 8 (2019), Nr. 3, S. 292. `http://dx.doi.org/10.3390/electronics8030292`. – DOI 10.3390/electronics8030292. – ISSN 2079–9292

[Zha00] ZHANG, G. P.: Neural Networks for Classification: A Survey. In: *Trans. Sys. Man Cyber Part C* 30 (2000), November, Nr. 4, S. 451–462. `http://dx.doi.org/10.1109/5326.897072`. – DOI 10.1109/5326.897072. – ISSN 1094–6977

[Ziv04] ZIVKOVIC, Z.: Improved adaptive Gaussian mixture model for background subtraction. In: *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)* Bd. 2, 2004, S. 28–31 Vol.2

[ZLS13] ZANFIR, Mihai ; LEORDEANU, Marius ; SMINCHISESCU, Cristian: The Moving Pose: An Efficient 3D Kinematics Descriptor for Low-Latency Action Recognition and Detection. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (2013), S. 2752–2759. `http://dx.doi.org/10.1109/ICCV.2013.342`. – DOI 10.1109/ICCV.2013.342. – ISBN 978–1–4799–2840–8

[ZLYC12] ZHAO, Yang ; LIU, Zicheng ; YANG, Lu ; CHENG, Hong: Combing RGB and Depth Map Features for Human Activity Recognition. In: *Proceedings of The Asia Pacific Signal and Information Processing Association Annual Summit and Conference*, 2012, S. 1–4

[ZS13] ZHANG, J. ; SCLAROFF, S.: Saliency Detection: A Boolean Map Approach. In: *2013 IEEE International Conference on Computer Vision*, 2013. – ISSN 1550–5499, S. 153–160

[ZS16] ZHEN, Xiantong ; SHAO, Ling: Action recognition via spatio-temporal local features: A comprehensive study. In: *Image and Vision Computing* 50 (2016), S. 1–13. `http://dx.doi.org/10.1016/j.imavis.2016.02.006`. – DOI 10.1016/j.imavis.2016.02.006

[ZWN+17] ZHANG, Shugang ; WEI, Zhiqiang ; NIE, Jie C. ; HUANG, Lei ; WANG, Shuang ; LI, Zhen: A Review on Human Activity Recognition Using Vision-Based Method. In: *Journal of Healthcare Engineering* 2017 (2017), Nr. 3090343. `http://dx.doi.org/10.1155/2017/3090343`. – DOI 10.1155/2017/3090343

[ZXSJ19] ZHAO, R. ; XU, W. ; SU, H. ; JI, Q.: Bayesian Hierarchical Dynamic Model for Human Action Recognition. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, S. 7725–7734

[ZZG+15] ZHANG, W. ; ZHAO, D. ; GONG, W. ; LI, Z. ; LU, Q. ; YANG, S.: Food Image Recognition with Convolutional Neural Networks. In: *IEEE 12th Intl Conf. on Ubiquitous Intelligence and Computing (UIC-ATC-ScalCom)*, 2015, S. 690–693

[ZZSS16] ZHU, Guangming ; ZHANG, Liang ; SHEN, Peiyi ; SONG, Juan: An Online Continuous Human Action Recognition Algorithm Based on the Kinect Sensor. In: *MDPI, Sensors (Basel)* 16 (2016), Nr. 2, 1–18. `http://dx.doi.org/10.3390/s16020161`. – DOI 10.3390/s16020161. – ISSN 1424–8220

[ZZZ+19] ZHANG, Hong-Bo ; ZHANG, Yi-Xiang ; ZHONG, Bineng ; LEI, Qing ; YANG, Lijie ; DU, Ji-Xiang ; CHEN, Duan-Sheng: A Comprehensive Survey of Vision-Based Human Action Recognition Methods. In: *Sensors (Basel, Switzerland)* 19 (2019), S. 5. `http://dx.doi.org/doi.org/10.3390/s19051005`. – DOI doi.org/10.3390/s19051005

Internet-Resources

[1] Victoria Bloom. http://dipersec.king.ac.uk/G3D/G3D.html, Last accessed on March 2019.

[2] Michael V. Boland. http://murphylab.web.cmu.edu/publications/boland/boland_node26.html, Last accessed on January 2019.

[3] OpenCV 3.0.0 dev documentation. https://docs.opencv.org/3.0.0/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html?highlight=findhomography, Last accessed on August 2018.

[4] OpenCV 3.0.0 dev documentation. https://docs.opencv.org/master/da/d17/group__ximgproc__filters.html, Last accessed on June 2020.

[5] OpenCV 3.0.0 dev documentation. https://docs.opencv.org/3.0.0/modules/photo/doc/inpainting.html, Last accessed on January 2019.

[6] OpenCV 3.0.0 dev documentation. https://docs.opencv.org/3.0.0/d7/d4d/tutorial_py_thresholding.html, Last accessed on December 2018.

[7] OpenCV 3.0.0 dev documentation (Geometric Image Transformations). https://docs.opencv.org/3.0.0/da/d54/group__imgproc__transform.html, Last accessed on September 2018.

[8] OpenCV 3.0.0 dev documentation (How to Use Background Subtraction Methods). https://docs.opencv.org/3.0.0/d1/dc5/tutorial_background_subtraction.html, Last accessed on May 2019.

[9] OpenCV 3.0.0 dev documentation (Machine Learning). https://docs.opencv.org/3.0.0/dd/de1/classcv_1_1ml_1_1KNearest.html, Last accessed on December 2018.

[10] OpenCV 3.0.0 dev documentation (Machine Learning). https://docs.opencv.org/3.0-beta/modules/ml/doc/neural_networks.html, Last accessed on February 2019.

[11] OpenCV 3.0.0 dev documentation (Motion Analysis and Object Tracking). https://docs.opencv.org/3.0-beta/modules/video/doc/motion_analysis_and_object_tracking.html, Last accessed on March 2019.

[12] OpenCV 3.0.0 dev documentation (Optical Flow). https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html, Last accessed on August 2018.

[13] OpenCV 3.0.0 dev documentation (Retina: a Bio mimetic human retina model). https://docs.opencv.org/3.0-beta/modules/bioinspired/doc/retina/index.html, Last accessed on April 2019.

[14] Wang et al. `http://lear.inrialpes.fr/people/wang/improved\_trajectories`, last access on September 2017.

[15] evilmucedin and beru. `https://github.com/tiny-dnn/tiny-dnn`, Last accessed on December 2017.

[16] Junsong Yuan Gang Yu, Zicheng Liu. `https://sites.google.com/site/skicyyu/orgbd`, Last accessed on March 2019.

[17] Robot Learning Lab. `http://pr.cs.cornell.edu/humanactivities/data.php`, Last accessed on March 2019.

[18] ROSE Lab (Rapid-Rich Object Search Lab. `http://rose1.ntu.edu.sg/datasets/actionrecognition.asp`, Last accessed on March 2019.

[19] Zicheng Liu. `http://www.uow.edu.au/~wanqing/#Datasets`, Last accessed on March 2019.

Curriculum Vitae

Personal Data

Name: Rawya Al-Akam

Date of Birth: January 09, 1984

Place of Birth: Babil, Iraq

Nationality: Iraqi

Address: Steinstr. 24b, 56073 Koblenz

Contact: rawya@uni-koblenz.de

Education

2004 - 2007: Ms.c. in computer science, Al-Nahrain University, Baghdad, Iraq

2001 - 2004: B.Sc. in computer science, Al-Nahrain University, Baghdad, Iraq

1998 - 2001: Baccalaureate, Al-Hilla Secondary School, Babil, Iraq

Working Experience

2015 - 2020: Researcher at Computer Vision Research Group, (Prof. Dr. Ing. Dietrich Paulus), University of Koblenz-Landau, Germany

2010 - 2012: Teacher assistant at Computer Engineering, Islamic University College, Iraq

2007 - 2008: Teacher assistant at Computer Science Department, Babylon University, Iraq

Grant

2015 - 2019: Research grants by the Iraqi Ministry of Higher Education and Scientific Research (MHESR) for studying a PhD at the University of Koblenz-Landau