博士論文

A STUDY ON HUMAN ACTIONS REPRESENTATION AND RECOGNITION

人の行動の表現と認識に関する研究

指導教員　タン ジュークイ 准教授

所　属　九州工業大学大学院工学府
機械知能工学専攻知能制御工学コース

学籍番号　13584201

氏　名　Sheikh Mohammad Masudul Ahsan

PhD Thesis

A STUDY ON HUMAN ACTIONS REPRESENTATION AND RECOGNITION

By

Sheikh Mohammad Masudul Ahsan

Student No.: 13584201

Supervised by

Professor Joo Kooi Tan

Department of Control Engineering

Kyushu Institute of Technology, JAPAN

March 2016

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Elaboration
ANN	Artificial Neural Network
DMHI	Directional Motion History Image
FPR	False Positive Rate
HFMEI	Hole filled MEI
HOG	Histogram of Oriented Gradient
HOR	Histogram-of-Oriented-Rectangles
k-NN	*k*-Nearest Neighbor
LBP	Local Binary Pattern
MEI	Motion Energy Image
MHI	Motion History Image
MHV	Motion History Volume
SVM	Support Vector Machine
SWT	Stationary Wavelet Transform
DMHI_H	Histogram created from DMHI
MEI_H	Histogram created from MEI
DMHI_MEI_H	Histogram created by concatenating DMHI_H and MEI_H
RNDSN_H	Histogram for the random selected snippets
SELSN_H	Histogram for the selective snippets
CLBPD_H	Histogram of constant bit arranged LBP extracted from DMHI
CLBPD_SELSN_H	Histogram created by concatenating CLBPD_H and SELSN_H
RLBPD_H	Histogram of rotated bit arranged LBP extracted from DMHI

Abbreviation	Elaboration
RLBPD_MEI_H	Histogram created by concatenating RLBPD_H and MEI_H
RLBPD_RNDSN_H	Histogram created by concatenating RLBPD_H and RNDSN_H
RLBPD_SELSN_H	Histogram created by concatenating RLBPD_H and SELSN_H
D_RLBPD_H	Histogram created by linear combination of DMHI_H and RLBPD_H
D_RLBPD_MEI_H	Histogram created by concatenating D_RLBPD_H and MEI_H
LBPM_H	Histogram of LBP image created from MHI
LBPM_RNDSN_H	Histogram created by concatenating LBPM_H and RNDSN_H
LBPM_SELSN_H	Histogram created by concatenating LBPM_H and SELSN_H

Abstract

In recent years, analyzing human motion and recognizing a performed action from a video sequence has become very important and has been a well-researched topic in the field of computer vision. The reason behind such attention is its diverse applications in different domains like robotics, human computer interaction, video surveillance, controller-free gaming, video indexing, mixed or virtual reality, intelligent environments, etc. There are a number of researches performed on motion recognition in the last few decades. The state of the art action recognition schemes generally use a holistic or a body part based approach to represent actions. Most of the methods provide reasonable recognition results, but they are sometimes not suitable for online or real time systems because of their complexity in action representation. In this thesis, we address this issue by proposing a novel action representation scheme.

The proposed action descriptor is based on a basic idea that rather than detecting the exact body parts or analyzing each action sequence, human action can be represented by a distribution of local texture patterns extracted from spatiotemporal templates. In this study, we use a novel way of generating those templates. Motion History Image (MHI) merges an action sequence into a single template. However, having the problem in overwriting old information by a new one in the MHI, we use a variant named Directional MHI (DMHI) to diffuse the action sequence into four directional templates. And then we use the Local Binary Pattern (LBP) operator, but with a unique way, a rotated bit arranged LBP, to extract the local texture patterns from those DMHI templates. These spatiotemporal patterns form the basis of our action descriptor which is formulated into a concatenated block histogram to serve as a feature vector for action recognition. However, the extracted patterns by LBP tends to lose the temporal information in a DMHI, therefore we take a linear combination of the motion history information and texture information to represent an action sequence. We also use some variants of the proposed action representation that include the shape or pose information of the action silhouettes as a form of histogram.

We show that, by effective classification of such histograms, i.e., action descriptor, robust human action recognition is possible. We demonstrate the effectiveness of the proposed method along with some variants of the method over two benchmark dataset; the Weizmann dataset and KTH dataset. Our results are directly comparable or superior to the results reported over these datasets. Higher recognition rates found in the experiment suggest that, compared to complex representation, the proposed simple and compact representation can achieve robust recognition of human activity for practical use. Besides the recognition rate, due to the simplicity of the proposed technique, it is also advantageous with respect to computational load.

Acknowledgements

Though only my name appears on the cover of this thesis, a great many people have contributed to its production. I owe my sincere gratitude to all those people who have made this dissertation possible.

First and foremost, I would like to express my profound gratitude and gratefulness to my academic supervisors **Prof. Joo Kooi Tan** and **Prof. Seiji Ishikawa** for their continuous guidance, motivation, patience, and providing me with an excellent atmosphere for doing research. From the guiding of the research to the revision of the thesis, their support, their encouragement, generosity and kindness were of paramount importance in the achievement of this work. For that, I will remain eternally grateful.

I would like to express my sincere acknowledgments to **Prof. Hyoungseop Kim**, **Prof. Shuichi Kurogi**, and **Prof. Takashi Morie** for their valuable suggestions as the members of my thesis evaluation committee.

I would like to thank the Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan for awarding me the Japanese government scholarship, which gave me the opportunity to come to Japan and achieve this work.

I am very much grateful to my wife **Mst. Nishrat Zahan,** my son **Sk. Mahzuz Ahsan**, and my daughter **Sk. Nuhaa Ahsan** for encouraging me to follow my dreams. My children's and especially my wife's contribution is invaluable to make my life comfortable and concentrating on research. They help me to get through the difficult times and give all the emotional support and entertainment. I am also equally grateful to my *mother, siblings* and relatives for supporting me physically and mentally.

I like to show equal thankfulness to **Dr. Md. Arifur Rahman Khan** and **Ms. Morsheda Begum**, who are also very close to me and support me in many respects. I want to express my heartiest thanks to **Ms. Keiko Matsuoka** who always helped me during the process of pregnancy and birth of Nuhaa. I would like to show my thankfulness to all my current and past lab members for their kind cooperation and helpfulness in accomplishing my experiments and make my university life smooth.

I would like to thank all my teachers and friends in Bangladesh, and all other people who helped me throughout the years.

I dedicate this work to my beloved ***Wife,***

Children, *and to my* ***Mother****.*

Chapter 1

Introduction

1.1 Background

From the prehistoric times, humans have thrived to invent various tools and instruments to emulate or even outdo human activities. In this process, the invention of computers revolutionized industrial works, being more precise, far more efficient and faster than humans. But this remains limited to repetitive and algorithmic tasks only. Recently, a greater effort is put by the researchers to extend the capabilities of computers in perceptual tasks, that humans do instinctively. In this era of computer and information technology, it is expected that in near future intelligent robots will live side by side with humans to serve the people.

Therefore, it is needed that such a robot should perform its actions with a human like degree of vision. We, humans, are able to recognize objects, scenes or environments most of the times independently and irrespective of the change of pose, variation of color or illumination. Researchers still battle to understand the underlying processes in the human brain or vision system that control these tasks.

Broadly, robots can be taught in two opposite ways [1]: To tell the robot in detail about what it has to do or to give the robots some learning mechanism and let the robot consider what the pertinent action is. Former strategy is more common for the robots to operate in highly controlled environment to do some pre-specified tasks. But such an approach is not suitable for the situation when the robot needs to adapt to a new or changing scenario. Moreover, it is difficult to pre-program in detail and specify all the new situations the robot might encounter [2]. An example of this is the Honda robot [3,4] which took nearly 10 years to program it for the capabilities like walking, climbing stairs, manipulating objects. On the other hand, researches are going on to give some learning capabilities to the robot using some learning strategy such as reinforcement

learning [5,6] or genetic algorithm [7], but in practice the learning power is still limited [1,8].

In the recent days, developing a vision based robotic system has got more attention in the field of computer vision research. The most of the vision based systems are developed to detect human motion in a video and analyze the motion for the classification or recognition purpose. The reason of such bias towards developing human motion analysis systems is its diverse applications in different domains. The applications that exploit the potential of the action recognition can be roughly grouped into three titles, surveillance, interaction, and analysis [9,10].

Surveillance applications cover some of the more classical types of problems related to automatic monitoring and understanding locations where a large number of people pass through. Vision based automatic recognition of abnormal or suspicious activity can replace or at least assist the operator in a security surveillance system.

Visual cues are the most important mode of nonverbal communication. *Interactive applications* try to use those visual cues like human pose or motion parameters to control something. This could be interfaces to games, e.g., as seen in EyeToy [11], Virtual Reality, or more generally, Human–Computer Interfaces. Interactive environments such as smart rooms [12] that can react to a user's gestures can benefit from vision based methods.

Videos have become a part of our daily life due to the availability of cheaper camera systems and free video sharing websites. It has become necessary to develop efficient indexing and storage schemes to improve user experience. *Analysis applications* annotate videos by analyzing the contents for easy retrieval. Some other human action analysis applications can be automatic diagnostics of orthopedic patients, analysis and optimization of an athlete's performances, recognizing humans based on behavioral cues such as human gait [13].

1.2 State of the Art

A generic action or activity recognition system includes the following major steps: (i) input video or sequence of images, (ii) extraction of concise low-to-mid level

features for action description, and (iii) high-level semantic interpretations [10]. Among them the most important step is the extraction of descriptors or image features that are discriminating enough with respect to human posture and motion for the quest of action classification. Existing action representation methods can broadly be categorized into two groups based on how they represent spatial or temporal structure of an action. Furthermore, spatial action representations can be of three types; body models, image models, and spatial statistics [14]. Temporal action representations can also be divided into three sub-types; grammars, templates and temporal statistics [14].

Body models - In each frame of the observed video, a 2D or 3D parametric estimate of the pose is recovered from a variety of available image features, and action recognition is performed based on such pose estimates. This is an intuitive and biologically-plausible approach to action recognition, which is supported by the work on visual interpretation of biological motion [14,15].

Image models do not require the detection and labeling of individual body parts. They only need to detect a region of interest (ROI) centered around a person and then features are computed densely on a regular grid bounded by the detected region. Since silhouettes are insensitive to color, texture, or contrast changes, many image models use silhouettes and contours [16,17] of the agent performing an action [14].

Spatial statistics representations decompose the image/video into smaller regions, not linked to body parts or image coordinates. Instead, actions are recognized based on the statistics of local features from all regions. The method first detects interest points [18,19] in an image, mostly at corners or blob like structures as local descriptors and then assign each region to a set of preselected vocabulary-features. Later, for classification, the method computes bag of features (BOF) that count the occurrence of the vocabulary-features within an image [14].

Action grammars- These models group the feature observations into states, and learn the temporal transition between these states using some probabilistic models like Hidden Markov Model (HMM). This type of representation has been used for instance in [20,21,22].

Rather than representing features and dynamics explicitly in a layered or sequential model, *Action templates* attempt to directly learn the appearance of complete temporal blocks of features. Most of the approaches that use templates are based on image models, such as [23,24] that build templates by stacking multiple silhouette images into a single volumetric representation [14].

Temporal statistic approaches attempt to build statistical models of the appearance of actions, without creating an explicit model of their dynamics. Typical examples are those methods that learn an appearance model of an action from a single characteristic keyframe [25,26] or from the histograms of (image, body or local) features over time [14]. Temporal bag of features has been used to represent sequences simply based on the frequency of feature occurrence over time [27,28,19,29].

1.3 Problem Statement

The state of the art action recognition schemes described in Section 1.2 mainly use two approaches for action representation; holistic and part-based representations. Holistic representation (image model, action template) focuses on the whole human body and then tries to search distinctive characteristics such as contours or pose. Part-based representations typically search for Space-Time Interest Points (STIPs) [18], and apply a robust description of the area around them and create a model [30]. Both approaches have their advantages and disadvantages. Holistic approaches are less amenable to action detection tasks because they do not have efficient methods for temporal segmentation and they do not generalize very well to incomplete or missing observations, e.g. occlusions. But otherwise, they are very simple and effective action representations, and in particularly attractive for action classification tasks because they straightforward integrate with powerful static classifiers such as support vector machines (SVM) or Adaboost [14]. Although part-based cases are better in case of partial occlusion, identifying individual body parts may cause overhead in total recognition time. Moreover, estimating parametric body models from images sometime uses special markers attached to the body, which makes them unsuitable for recognition tasks. On the other hand, marker-less systems are highly prone to false initialization

which limits their application. So, choosing the method will depend on the problem at hand needed to be solved.

In brief, most of the action recognition methods provide reasonable recognition result, but they are sometimes not suitable for online or real time systems because of their complexity in action representation. In this thesis, we are going to address this issue by proposing a novel action representation scheme and to evaluate the scheme. The proposed descriptor follows the holistic approach along with a spatial statistic model for the feature vector generation of an action.

1.4 Scope and Objective

The focus of this study is on the design, development, and evaluation of a vision based human motion analyzing system. The problem in hand is the recognition of human actions based on video-to-video matching concept. The goal of action recognition is to classify a given motion query into one of the pre-specified action categories defined in the action dataset.

The key objective of our research is to devise a simple and compact action representation scheme that can be applied to the recognition of different types of actions captured in different scenarios like indoor, outdoor, and variation in scale or worn cloths, etc. At the same time, our target is to keep the recognition time short enough for practical application while maintaining a higher accuracy. To realize the objective, our action descriptor is based on a basic idea that a human action descriptor can be represented as a distribution of local texture patterns extracted from a spatiotemporal template. To fulfil the purpose, rather than analyzing every frame or detecting the exact body parts, we are only interested in the distribution of those spatiotemporal patterns. In this research, we propose a novel way of constructing a spatiotemporal template that uses Directional Motion History Images (DMHI) and Local Binary Patterns (LBP).

1.5 Thesis Organization

The organization of the thesis is as follows:

In *Chapter 2*, we propose a novel method for the representation of human action which is based on the distribution of spatiotemporal texture pattern. Here, some related action representation methods are briefly described along with some variants of the proposed method.

The experimental environment, data set, and action recognition results of the proposed representation scheme are presented in *Chapter 3*. In this chapter, we also present the comparison of recognition rates with some state of the art action recognition methods reported employing the same dataset.

The possible future direction of work on the proposed method and the concluding words about the method are stated in *Chapter 4*.

Chapter 2

Action Representation and Recognition

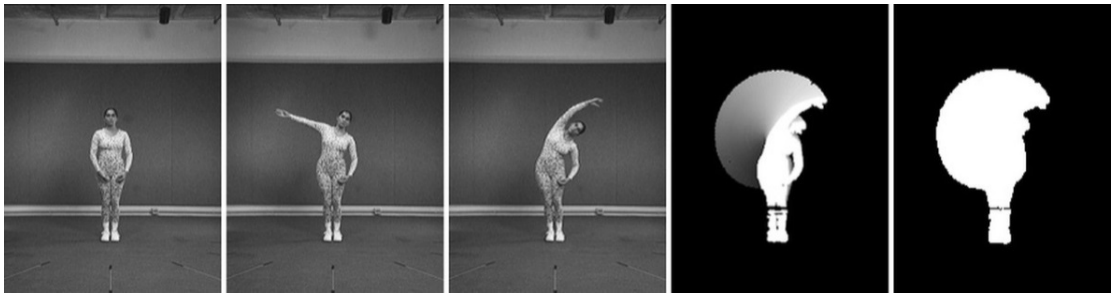
2.1 Background

Recognizing a performed action from a video sequence is a well-researched topic in the field of computer vision due to its various applications. In order to recognize various human actions, an action must be described compactly by some convenient representations. Though each application domains have their individual demands, in general, representation methods should be robust enough to cope with various actions performed by different people with several possible body configurations such as view angles, clothing, speed or posture variation. Moreover, for practical use, the designed methods must have the capability of adaptation to various types of environments like illumination change [31,32].

2.2 Related Works

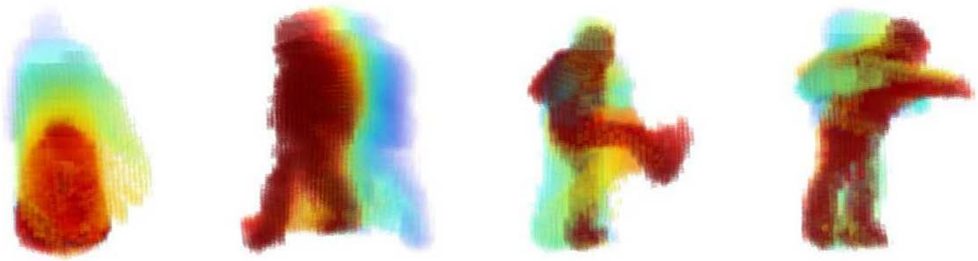
There are several methods [33] introduced in the literature for representing and recognizing a broad class of motion or action patterns. The most used low level features for action representation are points, box, silhouette, blob, contour, volume, and so on. Bobick and Davis [34,35] used motion energy images (MEI) and motion history images (MHI) for the first time as temporal templates to represent human actions. They extract silhouettes from a single view and combine the differences between subsequent frames of an action sequence. This results in a binary MEI which indicates where motion occurs. On the other hand, MHI is constructed where pixel intensities are a recency function of the silhouette motion. Recognition was done by using seven Hu moments. They have developed a virtual aerobics trainer that can watch and respond to the user as she/he performs the workout [32]. Fig. 2.1(a) shows MHI and MEI templates of an arm stretching exercise movement. Weinland et al. used multiple cameras to build motion

history volumes (MHV) which is an extension of the MHI [34] to 3D. Action classification was performed by aligning the volumes using Fourier transforms on the cylindrical coordinate system around the medial axis [36,37]. Fig. 2.1(b) shows MHVs of some actions such as - sit, walk, kick, punch. Related 3D approaches have been introduced by Blank et al. [23,38] and Yilmaz and Shah [24] who used time as the third dimension to form space-time shapes in the (x,y,t) space. These shapes were matched

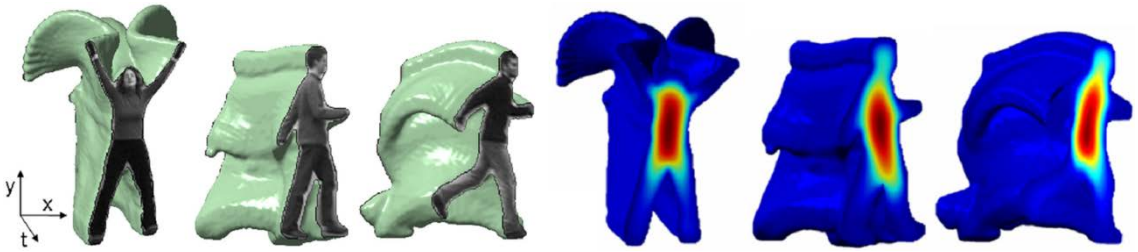


Frame 0 Frame 35 Frame 70 MHI MEI

(a)



(b)



(c)

Figure 2.1 State of the art action templates (a) motion history images (MHI) [39] (© IEEE, 2001), (b) motion history volumes (MHV) [36] (© Elsevier, 2006), and (c) space–time shapes [23] (© IEEE, 2005).

using features from Poisson equations and geometric surface properties, respectively [32]. Fig. 2.1(c) shows space-time shapes of jumping-jack, walking and running actions and solution to the Poisson equation on those shapes. Even though the shapes appear similar to MHVs, it is viewed from a single camera, whereas MHV needs multiple camera views.

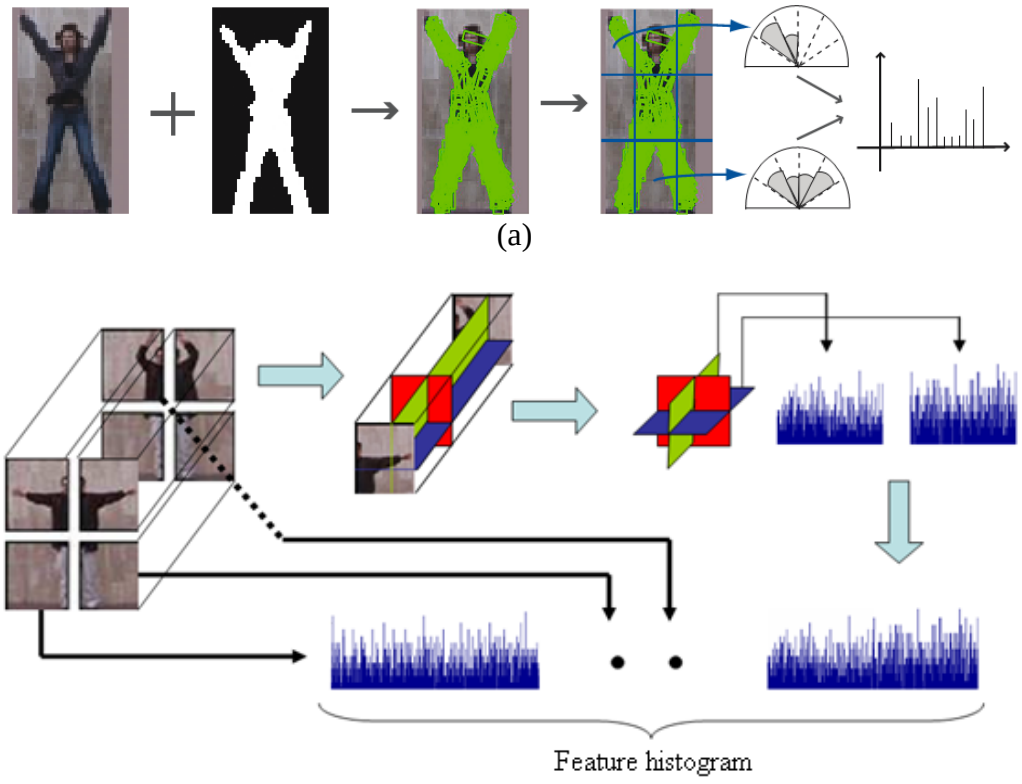
A histogram is a very popular statistic used in computer vision research. Many researchers [31,40,41] use histogram to represent their descriptors. Freeman and Roth [41] used orientation histograms for hand gesture recognition. Recently, Dalal and Triggs [42] used histograms of oriented gradients (HOGs) for human detection in images, which is found to be quite effective [40].

Ikizler and Duygulu use a pose descriptor named as Histogram-of-Oriented-Rectangles (HOR) for representing and recognizing human actions. They represent each human pose in an action sequence by oriented rectangular patches extracted over the human silhouette, and form a spatial oriented histograms to represent the distribution of these rectangular patches [40,43]. They used different classifiers like nearest neighbor, support vector machine, and dynamic time warping for the matching purpose. Fig. 2.2(a) shows an illustration of how the Histogram-of-Oriented-Rectangles are computed.

Kellokumpu et al. [31,44] extracted a histogram of local binary pattern (LBP) from MHI and MEI as static texture to represent action [32]. They also used another dynamic texture descriptor using LBP-TOP [45], which extracts LBP information from three orthogonal planes (*xy*, *xt*, and *yt*), i.e. they also include the time dimension for LBP extraction. They used HMMs to model the temporal behavior of action and hence to recognize them. Fig. 2.2(b) presents the dynamic texture descriptor generation process used by Kellokumpu et al. [31]. Yau et al. [46] used MHI to represent the temporal information of the mouth, which is generated using accumulative frame differencing of the video. The MHIs are decomposed into wavelet sub-images using discrete Stationary Wavelet Transform (SWT). Artificial neural network (ANN) with back propagation learning algorithm is used for classification. Kumar et al. [47] also used MHI and ANN for hand gesture classification [48].

Huang, et al. [49] represented a human action as a histogram of oriented gradient (HOG) [42] computed from MHI. First, they generated MHI by differential images from successive frames of a video, then HOG features were computed and supplied to a support vector machine (SVM) for action classification.

In the basic MHI method, old motion information can be wiped out by new motion information (from the later motion sequence). This overwriting surely causes poor recognition rate for natural motions that have complex nature and overlapping motion (e.g., sitting down and then standing up). To counter the problem, Ahad et al. [50] employs a variant of MHI called Directional MHI (DMHI) to represent a human action. They measure the optical flow from successive frames and partition the flow into four directions which are later used to compute the MHI. They also used Hu moments for the recognition purpose.



(a)

(b)

Figure 2.2 Existing feature extraction methods (a) histogram of oriented rectangles (HOR) [43] (© Springer, 2007), (b) dynamic feature histogram of Kellokumpu et al. [31] (©Springer, 2009).

Some methods such as [34,35] did not use any benchmark dataset to show the efficiency of their descriptors. Some other methods are not suitable for real-time recognition such as the HOR descriptor presented by Ikizler and Duygulu [40] which takes approximately one second per frame only for the rectangle extraction phase. Since Kellokumpu et al. used a volume based descriptor [31], it is also inexpedient for online recognition applications. In their descriptor, they have to wait for the next few frames to extract the LBP-TOP of a particular frame.

2.3 The Proposed Action Representation Method

Motivated by the previous works, we propose a simple and compact action representation method, which is presented in this section, for practical recognition problems. Fig. 2.3 shows the flowchart of the proposed method, which is self-descriptive. We first extract the foreground from an input video sequence. We assume that the video contains a predefined action performed by a human. We do not apply any human detection algorithm, because detecting a human in an image or in a video itself is an indepedentent research topic and fall outside of this research context. We then compute the directional motion template, specifically DMHI, and apply a LBP operator to extract the texture patterns in the DMHI. We also extract shape information from the silhouette of some selected frames. Later, the action region in a template is determined and partitioned. To generate the feature vector for each block, we compute histogram of those patterns. The histograms of all the blocks are then put together along with shape information to form an action descriptor. We use this descriptor to train a classifier if it is for a training video, otherwise we use the trained classifier to recognize the performed action.

2.3.1 Foreground Extraction

The sample videos in the action dataset we use are taken by a stationary camera. So, without constructing any complex background model, we simply take the absolute difference of the current frame with the static background frame and Otsu method of thresholding [51] is performed to get a binary foreground mask. Besides, before this

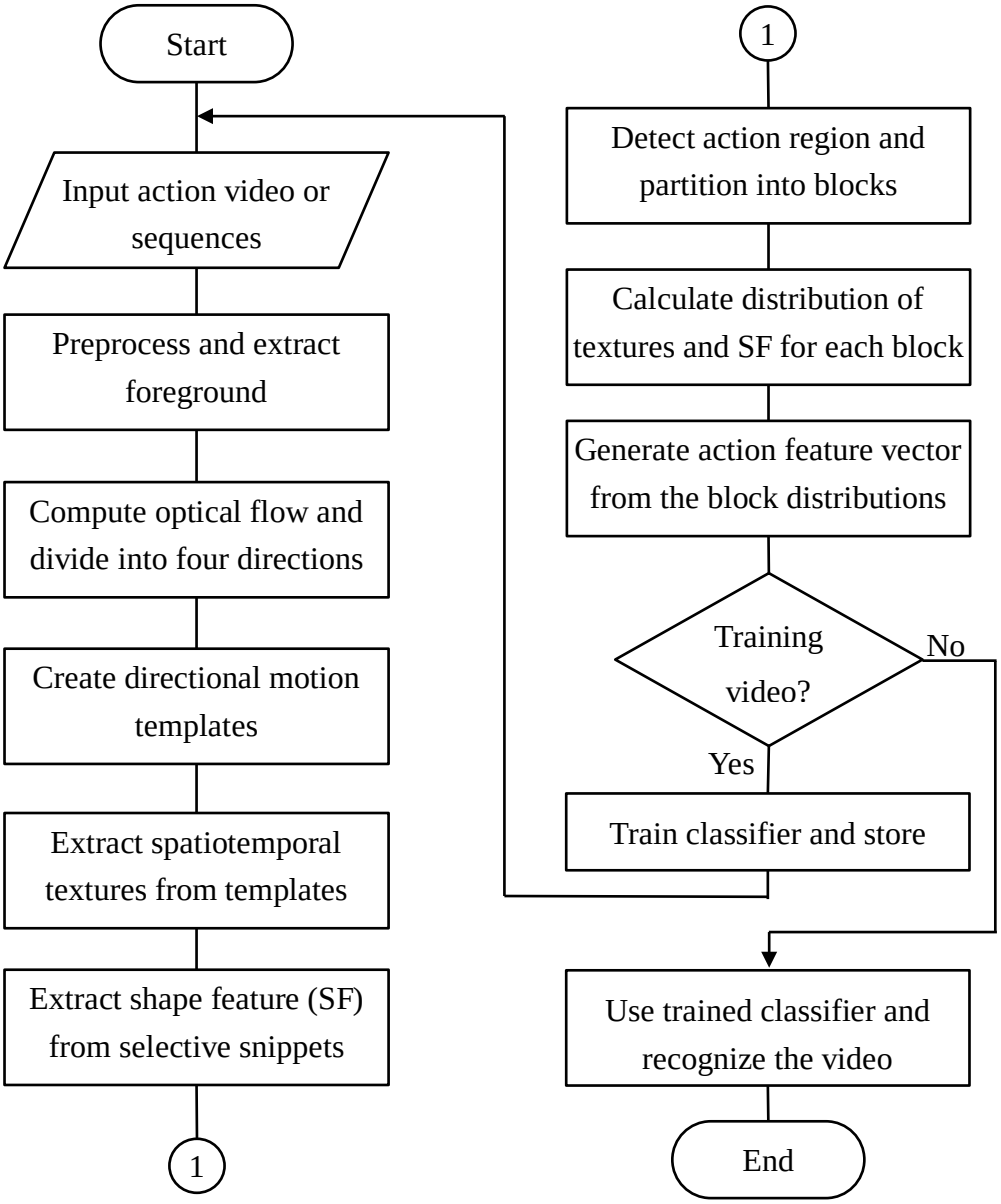


Figure 2.3 Outline of the proposed action recognition method.

process all the frames are passed through a Gaussian filter to reduce the effect of noise. Fig.2.4 shows a simple graphical illustration of the foreground extraction method. However, performing the subtraction of the background in a grayscale or RGB color space loses some foreground information for the action dataset in hand. Fig.2.4(c) shows some example of an extracted foreground in RGB color space using background subtraction and Otsu thresholding. Certainly it fails to extract leg part of the body which

is very important for this particular sidewalk action. This is because the color difference or contrast between the leg part and the background is very small and the Otsu method suppresses the value to zero. This may be overcome by using a manual threshold value lower than the Otsu threshold. But choosing a single manual threshold that works fine with all the video in the dataset is quite difficult, and sometimes produce more noisy

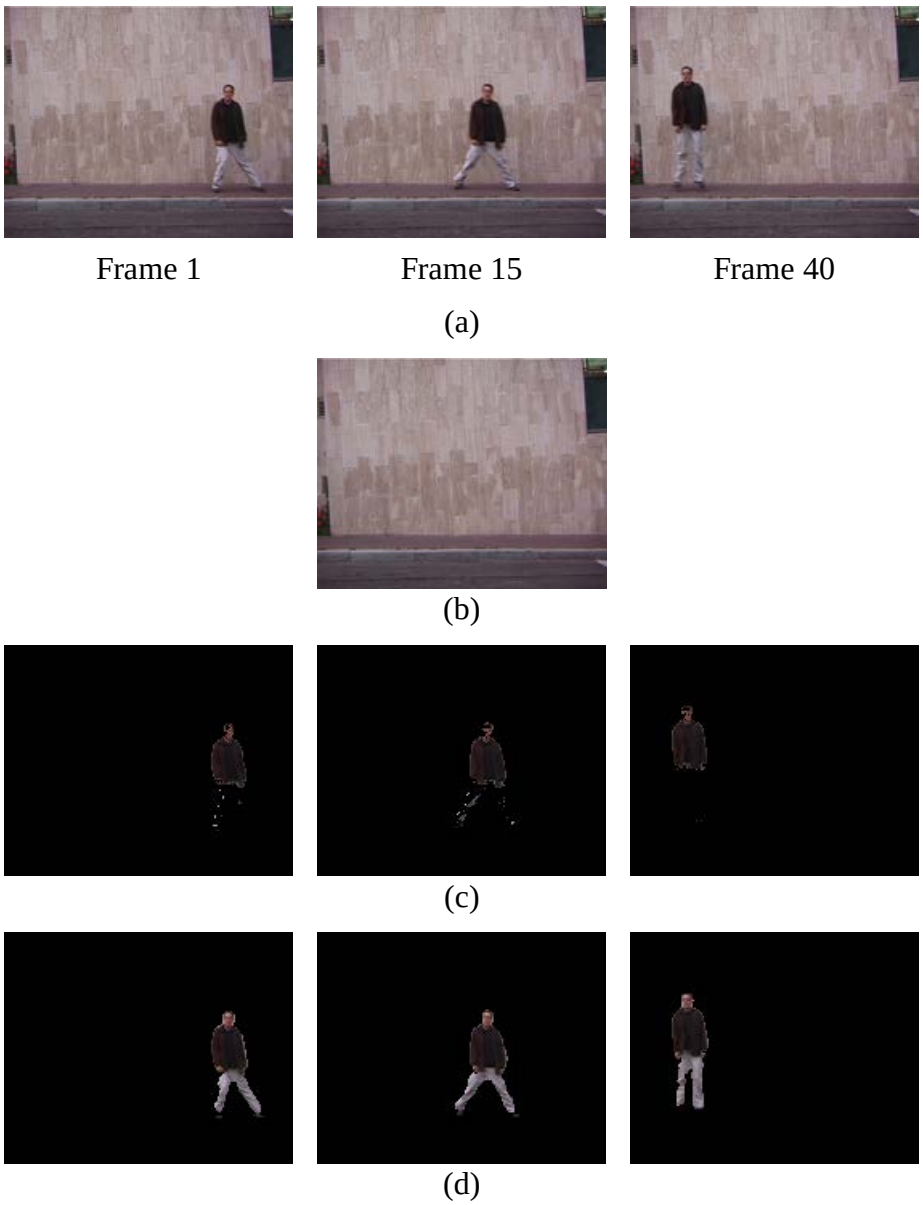


Figure 2.4 Simple illustration of foreground extraction (a) input frames, (b) background frame, (c) extracted foregrounds using the RGB color space, (d) extracted foregrounds using Lab color space and Eq. (2.1) – Eq. (2.6).

outcomes. Hence, rather than simple subtraction, we do all the processing in Lab color space and use Eqs. (2.1)-(2.6) in this process of foreground extraction, Fig. 2.4(d) shows the extracted foreground by this method.

The meanings of the used abbreviations in the equations are as follows: Df_t is the absolute difference of the current frame f_t, and the background frame f_{bg}, L, a, b are the pixel intensity of Df in the Lab color space, $\hat{m}_{fg_t}$ is the single channel intermediate frame where nonzero intensity represents the foreground, and after applying the Otsu threshold, we get m_{fgt} which is a binary frame containing the foreground mask of size $h_t \times w_t$. The suffix t means the frame at time t, and (x,y) is the spatial position of a pixel in a frame. We use this m_{fg_t} to get the actual foreground frame f_{fg_t}. The constants in Eq. (2.2) and Eq. (2.3) are experimentally determined. The method explained here is a little bit details of that presented in [52]. However, it is to be noted that any other method that extracts the silhouette of the subject will work just fine.

$$Df_t(x,y) = \left| f_t(x,y) - f_{bg}(x,y) \right| \tag{2.1}$$

$$a_t(x,y) = 0.25L_t(x,y) + 0.6a_t(x,y) + 0.15b_t(x,y) \tag{2.2}$$

$$b_t(x,y) = 0.2L_t(x,y) + 0.2a_t(x,y) + 0.6b_t(x,y) \tag{2.3}$$

$$\hat{m}_{fg_t}(x,y) = \sqrt{L^2{}_t(x,y) + a^2{}_t(x,y) + b^2{}_t(x,y)} \tag{2.4}$$

$$m_{fg_t} = OtsuThreshold\left(\hat{m}_{fg_t}\right) \tag{2.5}$$

$$f_{fg_t}(x,y) = \begin{cases} f_t(x,y) & \text{if } m_{fg_t}(x,y) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

2.3.2 The Motion Template

We have used Directional Motion History Image (DMHI) as a spatiotemporal template. As we mentioned earlier, the DMHI [50,53], is an extension of the basic Motion History Image (MHI) [34] which splits the motion into four different directions and then generates MHI for each direction. A MHI is a much effective representation of human motion, since it infuses several image sequences into only one template. In the

MHI, the temporal information is specified by the pixel intensity and thereby keeps track of motion information. The MHI uses a pixel intensity function $H_\tau(x,y,t)$ to compute the temporal history of motion at a particular time *t*. The function can be represented in a simple form, as in Eq. (2.7) with an update function $\Psi(x,y,t)$.

$$H_\tau(x,y,t) = \begin{cases} \tau & \text{if } \Psi(x,y,t) = 1 \\ \max(0, H_\tau(x,y,t-1) - \delta) & \text{otherwise} \end{cases} \tag{2.7}$$

$$\Psi(x,y,t) = \begin{cases} 1 & \text{if } D(x,y,t) \geq threshold \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

$$D(x,y,t) = \left| m_{fg_t}(x,y) - m_{fg_{t\pm\Delta}}(x,y) \right| \tag{2.9}$$

Here, *x*, *y*, and *t* show the position and time; the update function $\Psi(x,y,t)$ signals the presence of motion in the current frame $f(x,y)$; τ decides the temporal duration of a MHI, e.g., in terms of the number of frames or in terms of time (second / millisecond); and δ is the decay parameter whose value is 1 in the original MHI [34], and $D(x,y,t)$ gives the absolute difference between pixels of current silhouette or foreground mask with the

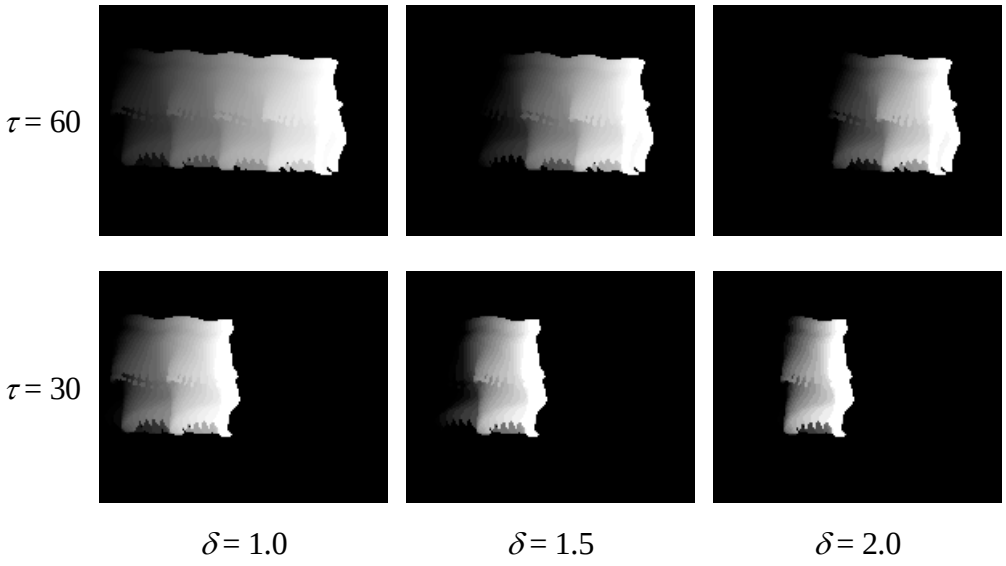


Figure 2.5 Effect of τ and δ in calculating the MHI template of a walking action.

silhouette of a step time difference Δ. This update function is called for every new video frame analyzed in the sequence. The result found by this computation is a grayscale image where brighter pixels represent more recent motion. Fig. 2.5 shows the MHI of a walking action with various values of τ and δ. Here, we can see that higher values of δ quickly remove the earlier trail of motion sequence, whereas τ controls the duration of an action, i.e., how long duration of a particular action will be taken into consideration for computing the MHI. The joint effect of τ and δ determines how many levels of quantization the MHI will have; a combination of a large τ and a small δ yields a smooth and continuous gradient, whereas a large τ and large δ provide a more discrete quantization of motion.

To create a DMHI, a moving region is initially tracked using a dense optical flow algorithm from successive frames that generally produces a vector denoting the horizontal (*x*-direction) and vertical (*y*-direction) motion of an object (Fig. 2.6(c)). Each of these horizontal and vertical motions are further rectified to positive and negative directions (Fig. 2.6(d)) and then a threshold is applied that results in four update functions denoting the directions right, left, up, and down. Fig. 2.6 shows the process of creating update functions. Fig. 2.6(a) shows two successive image frames of jumping jack action, Fig. 2.6(b) shows the direction and magnitude of the obtained dense optical flow in RGB color image (left-side). In the RGB image of dense flow, similar color means similar direction and higher saturation of the color denotes higher magnitude of the flow. We also present conventional sparse representation of the flow in Fig. 2.6(b) (right-side) just for better understanding of the flow.

The update functions found in the above mentioned process are simply binary images representing different directional motion regions. These update functions are then used in Eq. (2.7) to generate directional MHIs. So, for a DMHI, Eq. (2.7) becomes the one as given in Eq (2.10), where the four different directions are denoted by $\ell \in$ { *right*(+*x*), *left*(−*x*), *up*(+*y*), *down*(−*y*)}.

$$DMHI_{\tau}^{\ell}(x,y,t) = \begin{cases} \tau & \text{if } \Psi^{\ell}(x,y,t) = 1 \\ \max(0, DMHI_{\tau}^{\ell}(x,y,t-1) - \delta) & \text{otherwise} \end{cases} \tag{2.10}$$

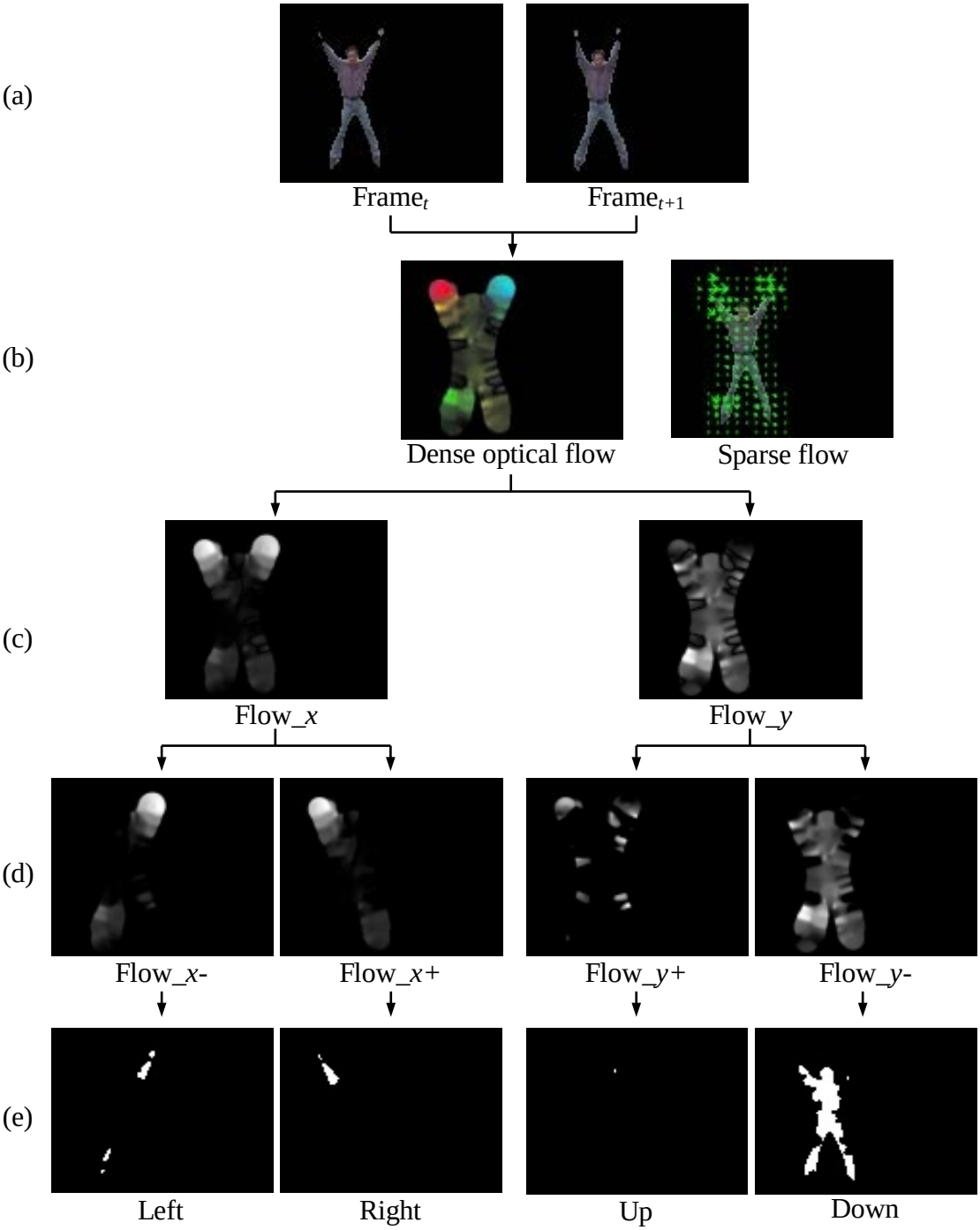


Figure 2.6 Creating directional update function (a) two successive frames, (b) dense optical flow (leftward), a sparse presentation of the dense flow (rightward), (c) optical flow divided into *x* and *y* direction, (d) flow is further split into positive and negative direction, (e) directional update functions found after applying a threshold to the flows.

Fig. 2.7 presents some example DMHIs of a jumping jack and a sidewalk action. Though it is not quite clear from Fig. 2.7(a), we can find that the *Left* and *Right* MHI only encapsulate the leftward and rightward motion of the hands. The encapsulation of directional motion is more evident in Fig. 2.7(b), since the motion in the performed action is mostly in a leftward direction. We can see that the *Left* MHI encapsulates almost all the motion information, and the *Right* MHI contains almost nothing. During side walk, bouncing upward and downward motions are captured by the *Up* MHI and Down MHI that can also be seen in Fig. 2.7(b).

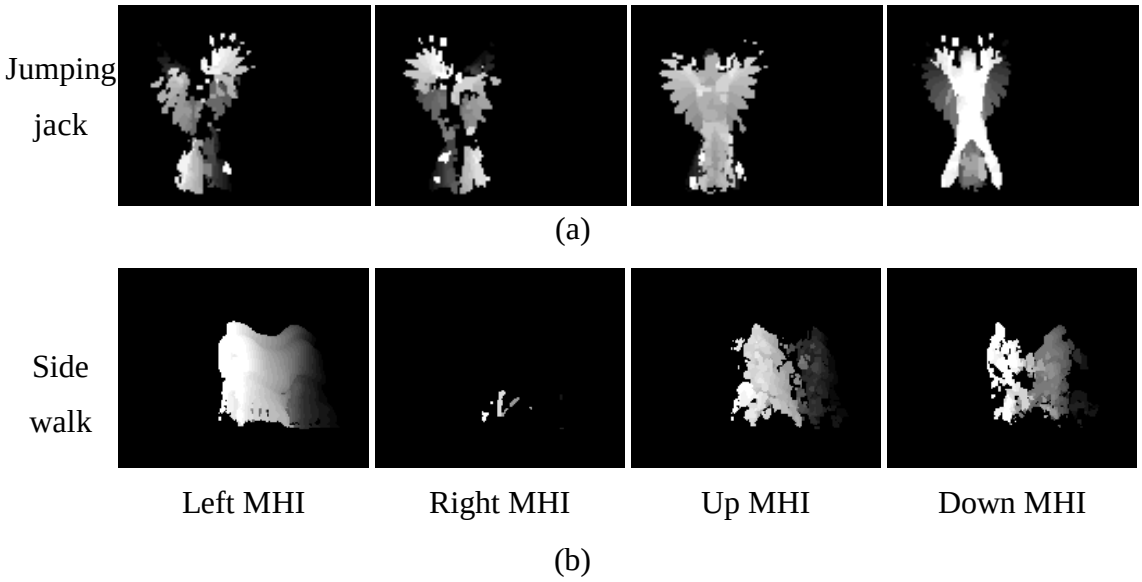


Figure 2.7 Example of DMHIs (a) jumping jack, (b) side walk action.

2.3.3 Extraction of Spatiotemporal Texture

Local Binary Pattern (LBP) is a self-similarity measure that was first introduced by Ojala et al. [54,55]. Due to its computational simplicity, efficiency and invariance to monotonic gray scale changes, it has become popular and has found horizons in various computer vision applications [56,57]. A detailed LBP related bibliography can be found online [58].

A texture T in a local 3×3 neighborhood of a monochrome texture image can be represented as the joint distribution of the gray levels of the nine image pixels (Eq. 2.11) having a spatial layout of the neighborhood as in Fig. 2.8 [54].

$$T = p(g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8) \tag{2.11}$$

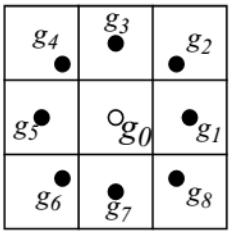


Figure 2.8 Original LBP layout of eight pixels in a 3x3 neighborhood.

As the first step towards gray scale invariance the gray value of the center pixel (g_0) is subtracted from the gray values of the surrounding pixels resulting Eq. (2.12). Assuming the $g_i - g_0$ (i = 1, 2, …, 8) are independent of g_0, which allows to factorize the Eq. (2.12) into Eq. (2.13). The distribution $p(g_0)$ describes the overall luminance, which unrelated to local image texture. By discarding it the Eq. (2.13) becomes Eq. (2.14). The invariance is achieved with respect to the scaling of the gray scale by considering just the signs of the differences instead of their exact values that yields Eq. (2.15) where $s(x)$ is defined by Eq. (2.16). Formulating the Eq. (2.15) slightly differently the expression for a LBP operator with 3×3 neighborhood can be written as Eq. (2.17).

$$T = p(g_0, g_1 - g_0, g_2 - g_0, ..., g_8 - g_0) \tag{2.12}$$

$$T \approx p(g_0)p(g_1 - g_0, g_2 - g_0, ..., g_8 - g_0) \tag{2.13}$$

$$T \approx p(g_1 - g_0, g_2 - g_0, ..., g_8 - g_0) \tag{2.14}$$

$$T \approx p\big(s(g_1 - g_0), s(g_2 - g_0), ..., s(g_8 - g_0)\big) \tag{2.15}$$

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{2.16}$$

$$LBP(g_0) = \sum_{i=1}^{8} B(g_i - g_0) \times 2^{i-1} \tag{2.17}$$

Simply stating, the LBP operator describes the local texture pattern of an image with a binary code, which can be obtained by taking a threshold of neighboring pixels

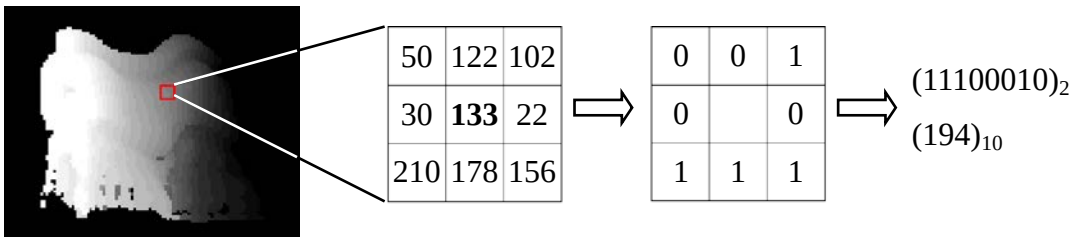


Figure 2.9 Example output of the original 3×3 LBP operator.

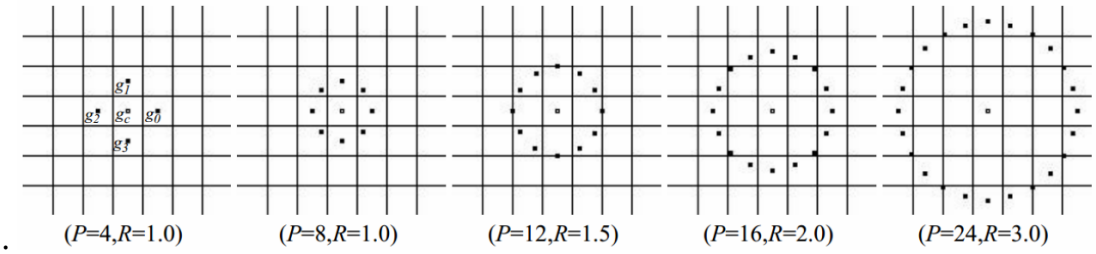


Figure 2.10 Circularly symmetric neighbor sets for LBP with different P and R [55] (© IEEE, 2002).

with the gray value of their center pixel. Fig. 2.9 shows an example output of the original LBP operator.

The original LBP operator used a 3×3 neighborhood, however, a generalized multiscale LBP operator can be defined as in Eqs. (2.18) - (2.19) with equally spaced sampling points P on a circular neighborhood with radius R. The layout of the neighborhood is shown in Fig. 2.10. In the following equations, g_c is the intensity of the center pixel and g_i (i = 0, 1, …, P-1) are the intensities of the sampling points derived from neighboring pixels by interpolation.

$$LBP(g_c) = \sum_{i=0}^{P-1} s(g_i - g_c) \times 2^i \tag{2.18}$$

$$s(x) = \begin{cases} 1 & \text{if } x \geq threshold \\ 0 & \text{otherwise} \end{cases} \tag{2.19}$$

In this research, we use a LBP operator with 3×3 neighborhood for the simplicity like the original one to extract texture information from the templates. But we use a

different arrangement of LBP bit position or layouts for different DMHIs, called as rotated bit arrangements. This is to give more strength to the pattern of a particular direction. Fig. 2.11(a) shows the binary values of the neighboring pixels for a basic LBP operator (extracted from Fig. 2.9), Fig. 2.11(b)-2.11(e) shows how we use the LBP bit arrangements for different DMHIs. Consider Fig. 2.11(b): Arrangement of bit positions is chosen in such a way that it will give more emphasis on the leftward motion. Other arrangements are chosen to have similar effects [48,59,32]. From Fig. 2.11, we can see that the same binary output (Fig. 2.11(a)) of the LBP operator can be assigned to a different decimal pattern value by rotating the bit arrangements, i.e., choosing a different starting and ending position for least and most significant bit, respectively. The generated LBPs corresponding to the DMHIs of a side walk (shown in Fig. 2.7(b)) are

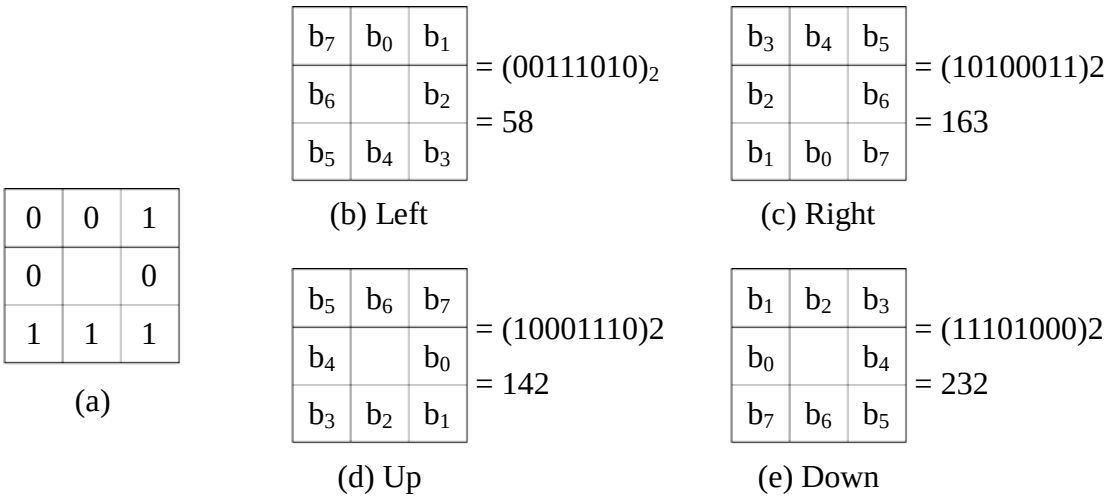
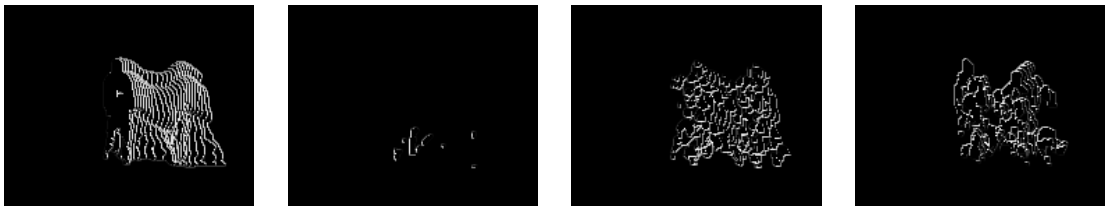


Figure 2.11 Illustration of different LBP bit arrangements.



Left LBP image Right LBP image Up LBP image Down LBP image

Figure 2.12 Example of LBP images corresponding to the DMHIs of a side walk action.

presented in Fig. 2.12. Since we use 3×3 neighborhood, the LBP value in decimal ranges from 0-255 that is shown as a grayscale image, i.e., the intensity of an LBP image denotes a pattern number.

2.3.4 Selective Snippets

We choose some frames containing the foreground mask from all the m_{fg_t} within the DMHI time duration, τ, of the performed action: We define those frames as selective snippets. We select only three frames as snippets and extract the pose information of the action in the form of a histogram (explained in Section 2.3.6) and call it as shape feature [52]. To select the snippets, we determine the minimum bounding rectangle of the foreground mask in every m_{fg_t} and choose the frames with the property in Eq. (2.20) – Eq. (2.23).

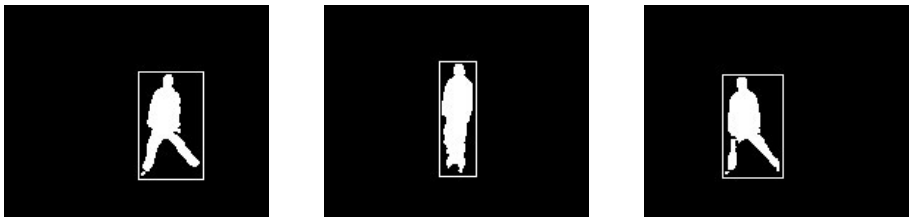
$$t_1 = \underset{t\in[0,1,\ldots,\tau]}{\arg\max}(h_t \times w_t) \tag{2.20}$$

$$t_2 = \underset{t\in[0,1,\ldots,\tau]}{\arg\max}\left(h_t \middle/ w_t\right) \tag{2.21}$$

$$t_3 = \begin{cases} \frac{t_1+t_2}{2} & \text{if } |t_1 - t_2| > \frac{\tau}{2} \\ \left[\max(t_1,t_2) + \frac{\tau - |t_1-t_2|}{2}\right] (\mathrm{mod}\,\tau) & \text{otherwise} \end{cases} \tag{2.22}$$

$$S_k = m_{fg_k} \;\; ; \;\; k = t_1, t_2, t_3 \tag{2.23}$$

Here, h_t, and w_t are the height and width of the bounding rectangle of the foreground mask in frame m_{fg_t} at time t. S_{t_1} is the snippet with the pose covering the maximum area in the frame, S_{t_2} is the snippet with the pose having the narrowest possible area, and S_{t_3} is simply an in-between snippet of S_{t_1} and S_{t_2}. We choose first two frames with a basic intuition that the pose with maximum covering area and pose with minimum covering area provide some distinctive information for classification. The third frame is chosen only to make the shape information more robust. Fig. 2.13 displays the snippets, containing the foreground mask, selected for shape feature extraction.



$S_{t_1} \leftarrow \max(h \times w)$ $S_{t_2} \leftarrow \max(h/w)$ $S_{t_3} \leftarrow$ middle

Figure 2.13 The snippets selected for the extraction of shape feature.

2.3.5 Finding Action Region

Sometimes extracted foreground contains some noise which is passed to the motion templates. Hence, for better recognition results, we determine the region in the template where the action is performed. Since we collect the texture statistics in a block basis, this also helps to alleviate the translational effect. To find a tight bounding area around the action region, we use MEI. The MEI is deduced by accumulating all the DMHIs to a single image and then taking the image with a threshold equal to zero [34]. Eq. (2.24) shows the formula to calculate MEI, where $DMHI_\tau^\ell$ is defined in Eq. (2.10). An example of MEI of a hand waving action is shown in Fig. 2.14(a). MEI sometimes contains holes within the action region. Then they are filled up and a hole filled MEI, called HFMEI, is created. In the hole filling process, a simple algorithm is used. For a pixel in a hole, we search for four non-hole pixels in four different directions (straight-left, straight-right, straight-up, and straight-down). If they develop, there will be a change in the hole pixel to non-hole pixel, otherwise it is better to leave it as it was, the details of the hole filling method can be found in [60]. Fig. 2.14(b) shows the HFMEI after hole filling process.

Then we try to find the minimum extent in the *x*-axis having the maximum ratio (which we call the information ratio) of non-zero pixels in that extent and the total number of non-zero pixels. For this purpose, we first determine the distribution of the information ratios for all possible extent combinations. Fig. 2.14(e) shows the information ratio distribution of the hole filled MEI in the *x*-axis. To express the information ratio mathematically, let us first consider that HFMEI is an image with *M* rows and *N* columns, and HFMEI(*x*,*y*) is either zero or one, since it is a binary image.

$$MEI_{\tau}(x,y,t) = \begin{cases} 1 & \bigcup_{\forall \ell} DMHI_{\tau}^{\ell}(x,y,t) > 0 \\ 0 & otherwise \end{cases} \tag{2.24}$$

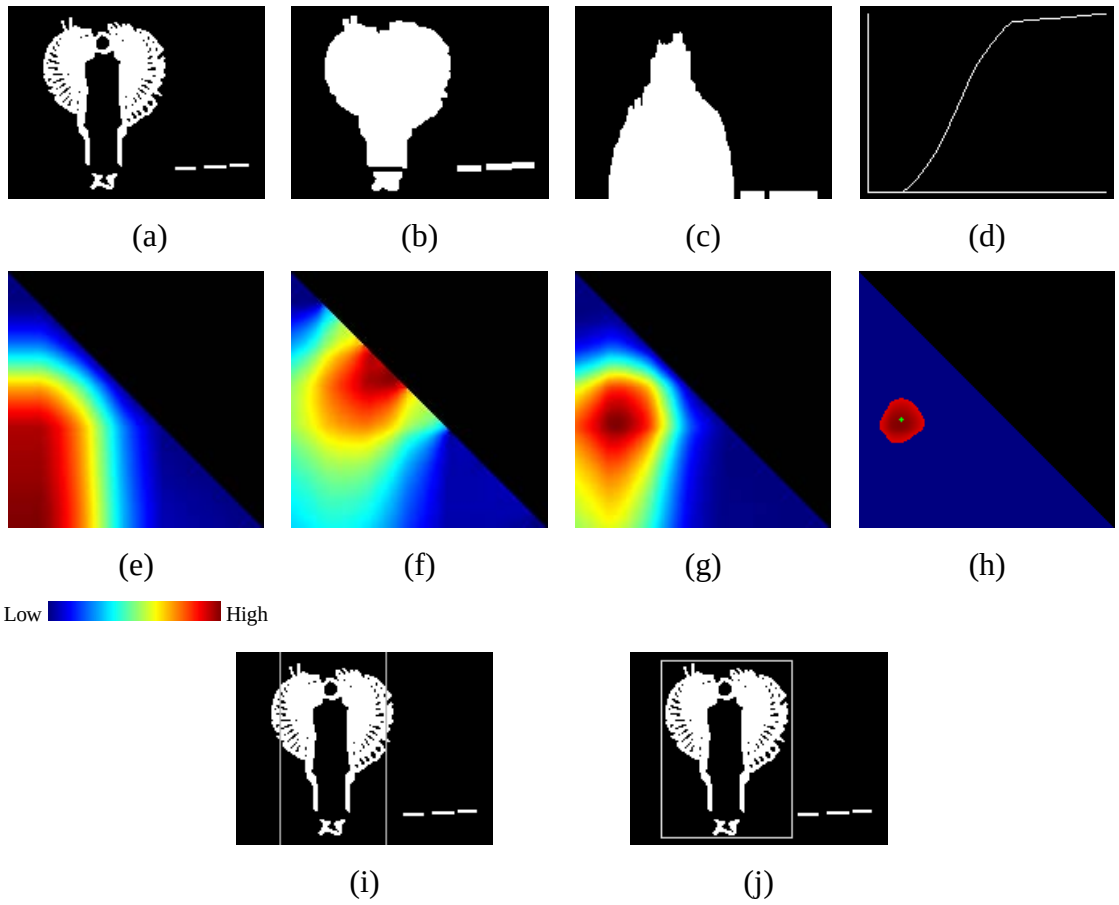


Figure 2.14 Images used to determine the action region (a) the MEI, (b) hole filled MEI (HFMEI), (c) vertical projection of HFMEI, (d) cumulative distribution of the vertical projected frequency, (e) distribution of information ratio, (f) density distribution, (g) multiplied density-information ratio distribution, (h) thresholded density-information highlighting the center, (i) initial *x*-extent, (j) final bounding box for the action region.

We compute the vertically projected frequency of non-zero pixels $nzf_v(i)$, in the HFMEI using Eq. (2.25), Fig.2.14(c) shows an example image. Then the cumulative frequency distribution $cf_v(i)$, is calculated using Eq. (2.26), a cumulative distribution of

the vertical projection histogram is presented in Fig. 2.14(d). Then we use Eq. (2.27) to compute the information ratio in *x*-axis $IR_X(x,y)$. The distribution of information ratio, shown in Fig. 2.14(e), is a square matrix where each column means the start position of the extent, i.e., the *x* value in *IR* matrix denotes left side (*lp*) of the extent. Similarly, row (the *y* value) stands for end position (right side, *rp*) of the extent. The cross point of (*x*,*y*) in the matrix represents the information ratio $IR_X(x,y)$ in that extent. As this matrix is a symmetric matrix, we only use the lower triangle of the matrix.

$$nzf_v(i) = \sum_{y=0}^{M-1} HFMEI(i,y) \quad ;\ i = 0,...,N-1 \tag{2.25}$$

$$cf_v(i) = \begin{cases} 0, & i = 0 \\ cf_v(i-1) + nzf_v(i-1), & i = 1,...,N \end{cases} \tag{2.26}$$

$$\begin{aligned} IR_X(x,y) &= IR_X(lp,rp) \\ &= \frac{cf_v(rp+1) - cf_v(lp)}{cf_v(N)} \quad ;\ lp, rp = 0,...,N-1 \text{ and } lp < rp \end{aligned} \tag{2.27}$$

$$\begin{aligned} DST_X(x,y) &= DST_X(lp,rp) \\ &= \frac{IR_X(lp,rp)}{rp - lp + 1} \quad ;\ lp, rp = 0,...,N-1 \text{ and } lp < rp \end{aligned} \tag{2.28}$$

The information ratio distribution is further divided by their corresponding extent length to get per unit information or information density, $DST_X(x,y)$, of the extent using Eq. (2.28). The density distribution is then normalized using min-max normalization (See Fig. 2.14(f)) and multiplied to the information ratio distribution (Fig. 2.1(g)). Then we take a threshold at the 90th percentile on the multiplied density-information distribution which results in an area with the maximum possible information with the minimum extent. We find the center of that area (Fig. 2.14(h)) and use the *x* value as left (*lp*) and *y* as a right position (*rp*) of the extent. Fig. 2.14(i) shows the initial extent found in this process. However, this extent sometimes loses some information. So we adjust the (*lp*, *rp*), which helps the utilized region to expand. But this adjustment is within the 10 percent of the initially determined extent length on each side. We then determine the

topmost and bottom most pixels within that extent and use their ordinate (y) values as top (tp) and bottom position (bp) for the bounding rectangle of the action region. Fig. 2.14(j) presents an example of bounding rectangle showing only the action region, excluding the noisy area in the MEI. The explained finding action region method was first described in [61].

2.3.6 Feature Vector Generation

An image texture can be described by its intensity distribution. We use a popular statistic, a histogram, to represent an action. We use two variant methods for feature vector generation. The first method is using only the LBP image to extract the texture distribution along with the selected snippets. The second method uses both the motion templates (DMHIs) and LBP images. In the second cases we take the linear combination of DMHI and LBP texture distribution along with shape information from MEI. As we mentioned earlier, we compute the feature vector only from the action region.

2.3.6.1 Feature Vector from LBP images and Snippets

The determined action regions of the LBP images are partitioned into $p \times q$ disjoint blocks. For each block, we compute a LBP histogram splitting the entire pattern ranges (256 patterns) into r equal sized bins. Fig. 2.15 shows an example of how the histogram is generated for a single block. After that, all these block histograms are multiplied by a weight (See Eq. (2.29)) and concatenated together in a raster scanning fashion to forman image histogram. Since an intensity histogram loses spatial information, rather than

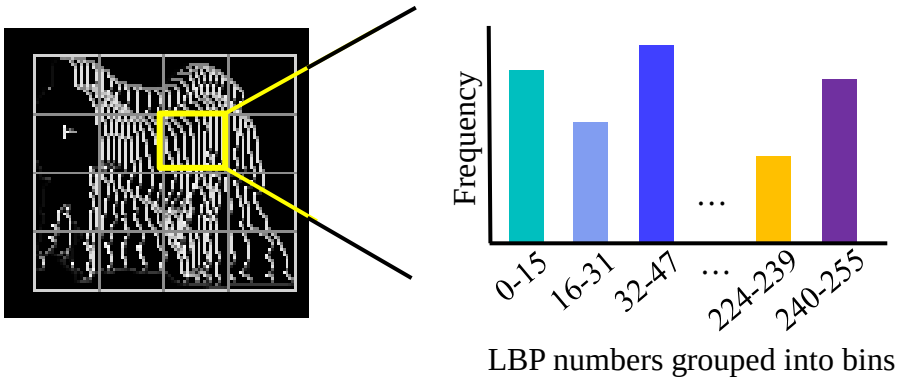


Figure 2.15 Computation of a block histogram.

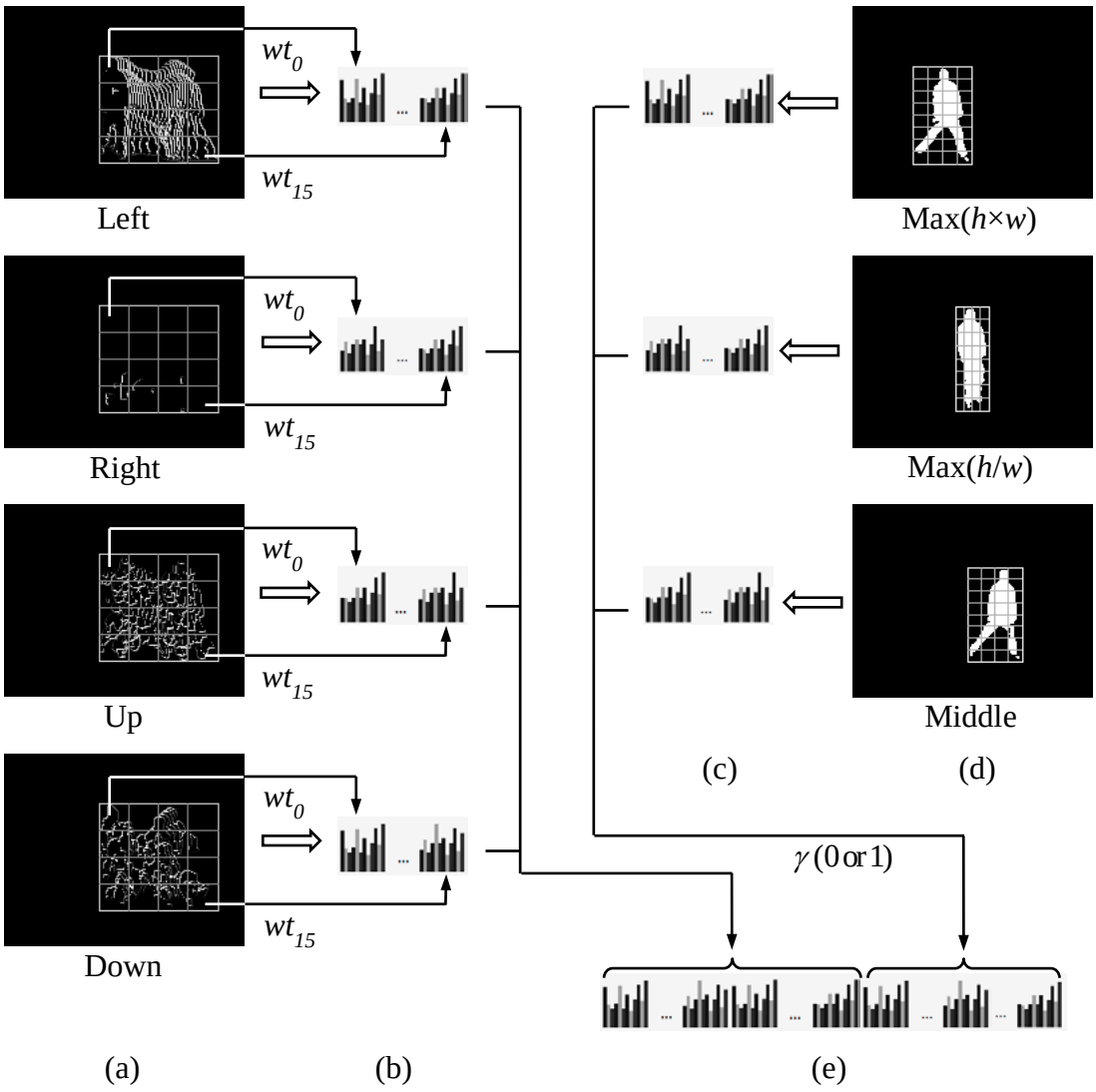


Figure 2.16 Generation of a feature vector from LBP images and snippets (a) LBP images where action regions are partitioned into 4×4 blocks, (b) concatenated block intensity (pattern) histograms of the LBP images, (c) block nonzero pixel frequency distribution of the selected snippets, (d) selective snippets partitioned into 8×4 blocks, (e) the feature vector composed of the concatenated histograms of all the LBP images and the snippets.

computing a single global histogram of an image, we calculate block histograms to encapsulate some spatial information into the feature vector. These histograms of all the

images are again concatenated together and normalized using L2 norm to constitute an action descriptor. This histogram created from LBP images can be individually treated as a feature vector during action recognition. We use the abbreviation RLBPD_H to denote this histogram, since we use rotated layout for LBP operator to extract the patterns from different DMHIs.

However, along with the LBP histogram, we also use shape feature of the action represented as a histogram of selective snippets. In this case, we partition the action region of the selected snippets into a constant 8×4 blocks. Since snippets are simply binary images, we then find the non-zero pixel distribution of the blocks which yields the histogram for the selected snippets (shortened as SELSN_H). The snippet histogram is then put together with LBP histogram to form a larger feature vector which is assigned an acronym RLBPD_SELSN_H, whose meaning is obvious. It is a histogram of LBP and selective snippets. Here, we use a control parameter γ (0 or 1) to make the snippet histogram optional in the action descriptor (RLBPD_SELSN_H) and thereby measure its importance in a recognition rate. Fig. 2.16 illustrates the construction of a feature vector for representing an action. The algorithm presented next explains the details of the steps necessary for computation of the feature vector.

Algorithm: *CreateFeatureVector*

Input: LBP images, L_i, i = 0, 1, 2, 3 and selective snippets S_j, j = 0, 1, 2 of an action

Initialization: Find the bounding box denoting the action region

LBP histogram $H_1 \coloneqq 0$

Snippets histogram $H_2 \coloneqq 0$

For each L_i (for1)

Partition the action region into $p \times q$ disjoint blocks

LBP Image histogram $LIH_i \coloneqq 0$

For each block b_k, k = 0, 1, …, $p \times q$-1 (for2)

Calculate weight wt_i for the block

$BH_k \coloneqq$ LBP histogram of b_k splitting the pattern's range (0 – 255) into r equal sized bins

$BH_k \coloneqq BH_k \times wt_i$

$LIH_i \coloneqq LIH_i \,||\, BH_k$, concatenate BH_k with LIH_i

End (for2)

$H_1 \coloneqq H_1 \,||\, LIH_i$, concatenate LIH_i with H_1

End (for1)

$H_1 \coloneqq$ L2_norm (H_1)

For each S_j (for3)

Partition the action region into 8×4 disjoint blocks

Snippet Image histogram $SIH_j \coloneqq 0$

For each block b_l, l = 0, 1, …, 31 (for4)

$f_l \coloneqq$ Count non-zero pixel frequency of b_l

$SIH_j \coloneqq SIH_j \,||\, f_l$, concatenate f_l with SIH_i

End (for4)

$H_2 \coloneqq H_2 \,||\, SIH_j$, concatenate SIH_j with H_2

End (for3)

$H_2 \coloneqq$ L2_norm (H_2)

Output: Feature vector, $FV \coloneqq H_1 \,||\, (H_2 \times \gamma)$, γ = 0 or 1, concatenate H_1 with H_2.

In the above algorithm, || is a concatenation operator, i.e., histograms are put side-by-side to form a larger histogram. The weight wt_i for each block is calculated using Eq. (2.29), where, NP_{B_i} is the number of non-zero pixels (each pixel is a pattern) in block i, and NP_A means the number of non-zero pixels in the action region, and ε is a constant (≈0) useful for no action scene. Here, wt_i is always greater than or equal to one and the sum of the inverse of the weights equals to one. Eq. (2.30) gives the maximum possible number of dimensions of the feature vector RLBPD_SELSN_H, where $p \times q$ is the number of blocks; r is the number of bins.

$$wt_i = \frac{1}{1 - {NP_{B_i}}/{NP_A + \varepsilon}}, \quad i = 0,1,..., p \times q - 1 \tag{2.29}$$

$$Dim_{RLBPD\_SELSN\_H} = 4 \times p \times q \times r + (8 \times 4) \times 3 \tag{2.30}$$

2.3.6.2 Feature Vector from DMHIs, LBP images and MEI

We use the histogram of both the DMHI and LBP images as well as the MEI to represent an action. In this case we create a histograms for LBP images (RLBPD_H) exactly as it is described in section 2.3.6.1. For DMHIs we go with similar fashion, i.e., action regions in the DMHIs are partitioned into $p \times q$ blocks, compute weighted block intensity histograms, and concatenate all the block histograms of the DMHIs into a single histogram (abbreviated as DMHI_H). These RLBPD_H and DMHI_H can be individually treated as a feature vector for recognition. However, we take the linear

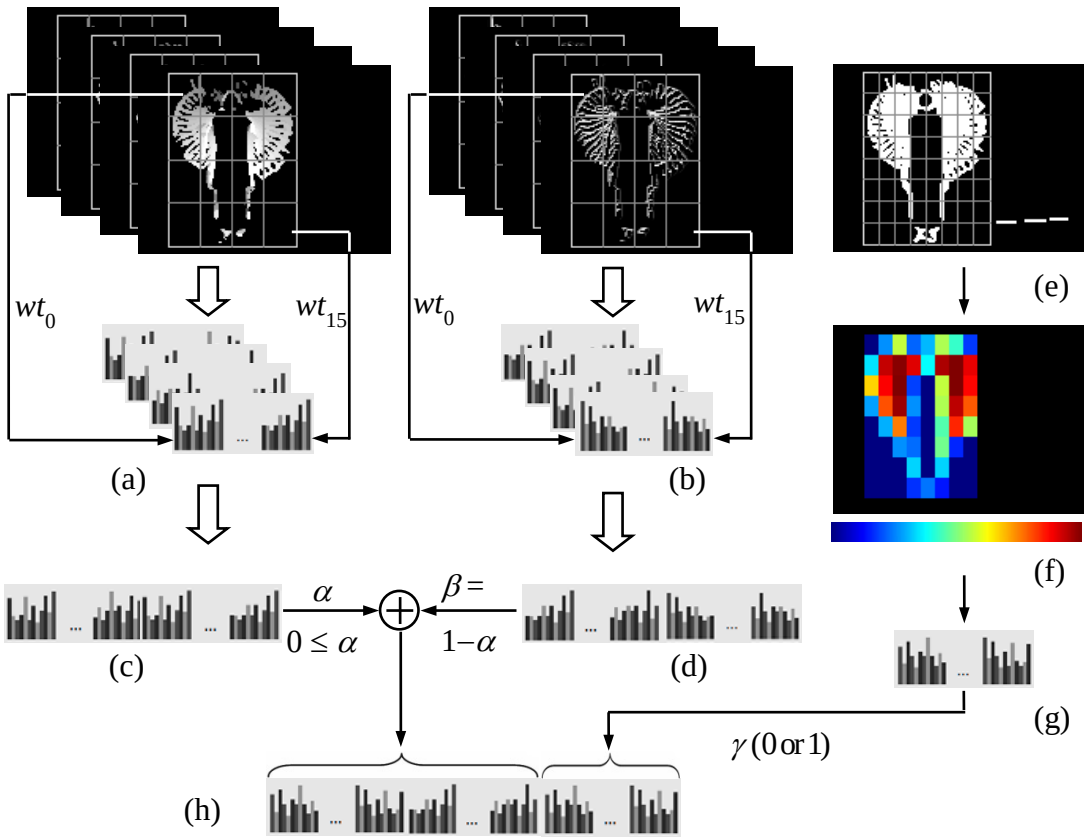


Figure 2.17 Generation of a feature vector from DMHIs, LBP images and MEI (a)-(b) block intensity histograms of DMHIs and LBP images, (c) DMHI histogram, (d) LBP histogram, (e)-(f) the MEI and the heat map of the blocks' non-zero pixel distribution, (g) linearized heat map – the MEI histogram, (h) the feature vector– component wise addition of (c) and (d) is concatenated with (g) (© IJICIC, 2015).

combination, i.e., a component wise addition of those histograms using Eq. (2.31) to generate the final feature vector for DMHI-LBP histogram representation (shortened as D_RLBPD_H).

Moreover, we use the MEI to have some shape information about the performed action. In this case, we partition the action region of MEI into $2p \times 2q$ blocks, and find the non-zero pixel distribution of the blocks which yields a MEI histogram (in short, MEI_H) as shown in Fig. 2.17 (g). The MEI_H is used to form a variety of D_RLBPD_H named as D_RLBPD_MEI_H by using Eq. (2.32).

Fig. 2.17 graphically illustrates the feature vector generation process explained above [61]. Weight $wt_i \geq 1$ for each block is calculated using the same Eq. (2.29) used before. Eq. (2.31) gives the feature vector of D_RLBPD_H, where, $0 \leq \alpha \leq 1$, and $\beta = 1 - \alpha$. Clearly, α = 0 or 1 means D_RLBPD_H becomes RLBPD_H or DMHI_H, respectively. Eq. (2.32) gives D_RLBPD_MEI_H, where || is a concatenation operator, again for α = 0 or 1, D_RLBPD_MEI_H becomes RLBPD_MEI_H or DMHI_MEI_H respectively. Eqs. (2.33) and (2.34) give the number of dimensions in the feature vector.

$$D\_RLBPD\_H = \alpha \times DMHI\_H + \beta \times RLBPD\_H \tag{2.31}$$

$$D\_RLBPD\_MEI\_H = D\_RLBPD\_H \;||\; MEI\_H \tag{2.32}$$

$$Dim_{D\_RLBPD\_H} = 4 \times p \times q \times r\,. \tag{2.33}$$

$$Dim_{D\_RLBPD\_MEI\_H} = 4 \times p \times q \times r + 2p \times 2q \tag{2.34}$$

2.4 Action Recognition

The task of recognition is considered as the final or long term goal of a motion or action analysis system. It is a kind of classification problem whose purpose is to classify the captured action as one of the several types of the learned actions. The past approaches of action recognition can be roughly classified into two groups: One group extracts a global feature descriptor from a video sequence [17,19,28] and assign a single label to the entire video. This method employs temporal characteristics for recognition. The methods based on this approach process either spatiotemporal data or temporal pose

estimated data. The other group extracts feature descriptor for each frame and assigns an action label to them [23,62,63]. This approach usually compares pre-stored information with the current image. The information may be templates [64], transformed templates [65], normalized silhouettes [66], or postures [67]. However, in both cases, a local label for the first approach can be obtained, if required, by extracting feature set until the desired frame is to be labeled. Similarly, a global label for the second approach is usually obtained by simple voting methods. In this research, we go with the first approach.

The recognition of an action can be performed at various levels of abstraction. Depending on the constraints and the requirements of specific applications, the recognition schemes are analyzed and the most suitable one is selected. Thus different recognition schemes and distance metrics are adopted in different works [68]. We focus on actions and do not explicitly consider context, such as the environment, interactions between persons or objects. These approaches fall outside the scope of this research. Moreover, we consider only full-body movements, this excludes the work on gesture recognition and other limb extraction based approaches.

Given an unseen action sequence, the recognition of human action becomes the process of motion classification. Recently, many different approaches have been used for action recognition. Some approaches directly match new sequences to training sequences or action prototypes. These methods do not explicitly model the variations in the temporal domain. A subcategory is that of discriminative classifiers that does not match, but rather classify the motion representation directly. There are some grammars and graphical models based classification methods that have a state-space character and model temporal variation implicitly. For action classification, in this research we use two classifiers, namely *k*-Nearest Neighbor algorithm, and Support Vector Machine (SVM). These classifiers actually fall in the first approach of direction recognition method.

2.4.1 *k*-Nearest Neighbor algorithm

In pattern recognition, the *k*-Nearest Neighbor algorithm (or *k*-NN for short) is a simple, non-parametric method used for classification and regression [69]. In the

training phase, the *k*-NN algorithm merely stores all the available training examples, mostly defined as a vector in a multidimensional feature space, and their class labels. In the classification phase, *k* is supplied as a user-defined constant – usually a small number, and an unlabeled vector (a query or test point) is classified by assigning the label which is the most frequent among the *k* training samples nearest to that query point. *k*-NN classifies new cases based on a similarity measure (e.g., distance functions). If $k = 1$, then the test point is simply assigned to the class of its nearest neighbor [70].

The ability of *k*-NN to cope with the variations in a performed action depends on the motion representation that is used, and the distance metric that is applied. In our research, we use five different distance metrics [71]. Considering the feature vector of an action as a point in a multidimensional space, we use popular Euclidean distance and Manhattan distance metric to measure the similarity between two points. Let, two vector $\boldsymbol{p} = (p_1, p_2, ..., p_n)$ and $\boldsymbol{q} = (q_1, q_2, ..., q_n)$ are two points in multidimensional space, then Eq. (2.35), and Eq. (2.36) defines the Euclidean and Manhattan distance respectively between these points. Besides these, since the feature vector of an action is actually a histogram, considering them as a discrete probability distribution, we use three other histogram similarity measuring methods, namely Chi-square distance, Bhattacharyya distance, and Pearson's correlation coefficient. Let two histograms be denoted by $\boldsymbol{H}_i$, and $\boldsymbol{H}_j$, each with n bins. Then Eq. (2.37) and Eq. (2.38) define the Chi-square and Bhattacharyya distance, respectively. Eq. (2.39) defines the correlation coefficient, where $\overline{H}_i, \overline{H}_j$ denotes the mean of their corresponding histogram. For all the distance measure except correlation coefficient, lower distance between two feature points or histograms means closer or more similar they are. On the other hand, for correlation coefficient, it is reverse, i.e., higher measures denote more similarity.

$$D_{Eucld}(\boldsymbol{p},\boldsymbol{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{2.35}$$

$$D_{Manht}(\boldsymbol{p},\boldsymbol{q}) = \sum_{i=1}^{n}|p_i - q_i| \tag{2.36}$$

$$\chi^2(\boldsymbol{H}_i, \boldsymbol{H}_j) = D_{chisq}(\boldsymbol{H}_i, \boldsymbol{H}_j) = \frac{1}{2}\sum_{k=1}^{n}\frac{\left(H_i(k) - H_j(k)\right)^2}{H_i(k) + H_j(k)} \tag{2.37}$$

$$D_{Bhatt}(\boldsymbol{H}_i, \boldsymbol{H}_j) = -\ln\sum_{k=1}^{n}\sqrt{H_i(k)H_j(k)} \tag{2.38}$$

$$D_{Corrl}(\boldsymbol{H}_i, \boldsymbol{H}_j) = \frac{\sum_{k=1}^{n}\left(H_i(k) - \overline{H}_i\right)\left(H_j(k) - \overline{H}_j\right)}{\sqrt{\sum_{k=1}^{n}\left(H_i(k) - \overline{H}_i\right)^2 \sum_{k=1}^{n}\left(H_j(k) - \overline{H}_j\right)^2}} \tag{2.39}$$

2.4.2 Support Vector Machine

Support Vector Machines are state-of-the-art large margin classifiers which have recently gained popularity within visual pattern recognition [72,73] and many other applications [28]. SVMs fall into the category of discriminative classifiers that distinguish between classes without explicitly modeling each. The action representation is simply regarded as a feature vector. Linear support vector machines learn a hyperplane in a feature space that is described by a weighted combination of support vectors. We present a brief review on a SVM here. For more details, refer to [74,75].

Consider the problem of separating a set of training data ($\boldsymbol{x}_1$, y_1), ($\boldsymbol{x}_2$, y_2), ... ($\boldsymbol{x}_m$,y_m) into two classes, where $\boldsymbol{x}_i \in \Re^N$ is a feature vector and $y_i \in \{-1,+1\}$ is a class label. If we assume that the two classes can be separated by a hyperplane $\boldsymbol{w}\cdot\boldsymbol{x} + b = 0$ in some high dimensional space, and that we have no prior knowledge about the data

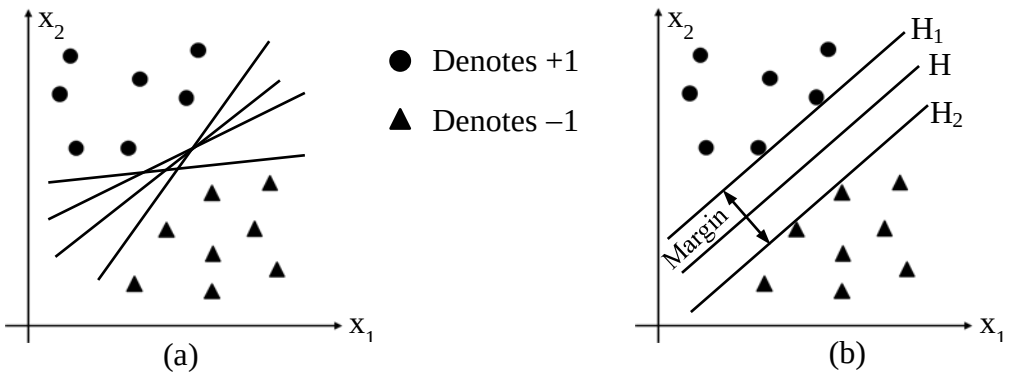


Figure 2.18 A support vector machine showing the separating hyperplane.

distribution, then the optimal hyperplane is the one which maximizes the margin [28,75]. Simply stating, SVM tries to find a hyperplane that maximizes the between class distance, and the marginal points or vectors of each class are called support vectors.

Fig. 2.18(a) shows that we can find as many hyperplanes as we wish to separate the two classes, but SVM determines the optimal one as shown in Fig. 2.18(b), where H is the optimal hyperplane, and support vectors are those which align with H_1, and H_2. SVMs have often been used in [28,40,76]. SVM needs the feature vector to be of a fixed length; for example, a histogram of code-words over a sequence of frames. Since our action descriptor also has a fixed size, SVMs are a suitable choice for action classification.

2.5 Summary

In this chapter, we concentrate on the different forms of action representations employed in recent years. At first, we present a detailed survey on human motion representation methods and its various challenges. We also describe, in detail, about two variants of the proposed action representation method, i.e., how we extract significant information for the task of human motion recognition. We also discuss about a number of issues concerning human motion recognition and explain about the classifiers that have been adopted in our work.

Chapter 3

Experiments and Results

3.1 Experiments

3.1.1 Dataset

We evaluate the performance of the proposed method by experimenting with two popular benchmark database: First one is the Weizmann action dataset [23], and the second one is KTH [28,77,78,62,79] action dataset. The reason of using benchmark datasets is that, we can easily and directly compare our results to other approaches reported in the literature.

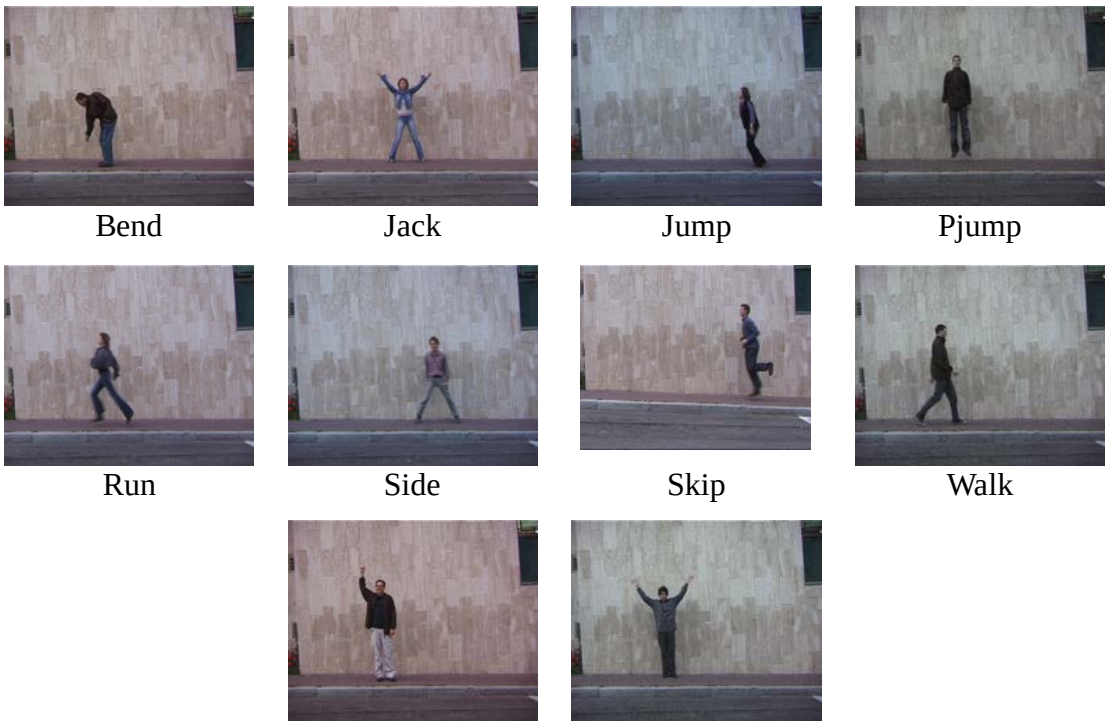


Figure 3.1 Sample frames of the different actions from Weizmann dataset.

The Weizmann dataset consists of 90 sample videos showing nine different people, each performing 10 natural actions such as “bend”, “jumping-jack” (“jack”), “jump-forward-on-two-legs” (“jump”), “jump-in-place-on-two-legs” (“pjump”), “run”, “gallop-side-ways” (“side”), “skip”, “walk,” “wave-one-hand” (“wave1”), and “wave-two-hands” (“wave2”). Fig.3.1 illustrates some sample frames of the Weizmann action dataset.

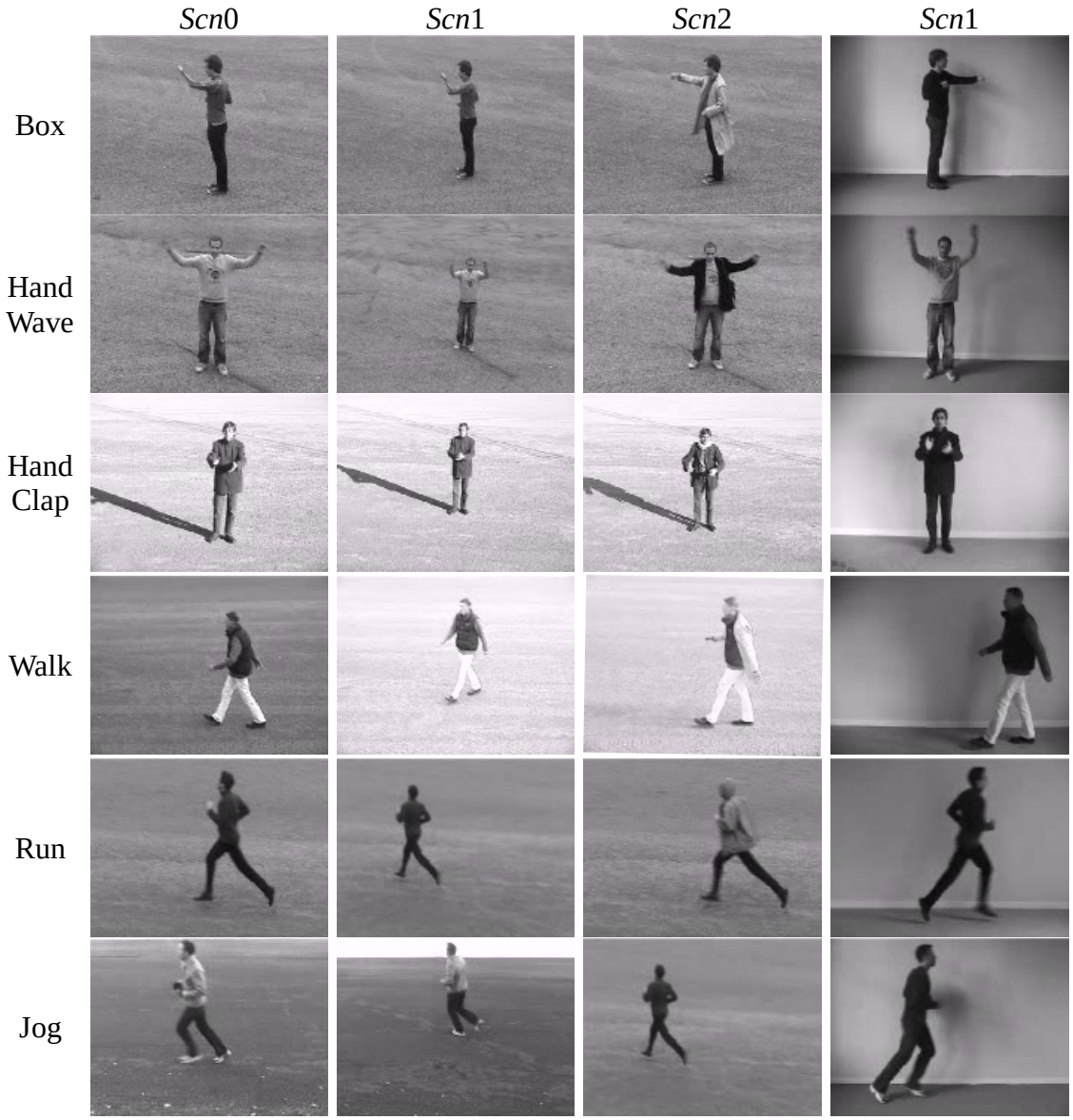


Figure 3.2 Sample frames of the KTH dataset along with different capturing conditions.

The KTH dataset consists of 600 sample videos of six different actions: “Boxing”, “Hand waving”, “Hand clapping”, “Walking”, “Running”, and “Jogging”. These actions are performed by 25 people in four different scenarios; outdoors (*Scn*0), outdoors with zooming, i.e., scale variations (*Scn*1), outdoors with variation in clothing (*Scn*2), and indoors (*Scn*3). For Walking, Running, and Jogging actions, we trim the video sequences so that the person always remains in view.

In the literature, KTH dataset has been treated either as one large set with strong intra-subject variations, or as four independent scenarios, which are trained and tested separately (i.e., four visually dissimilar databases, which share the same classes). We use both alternatives in our experiment.

3.1.2 Classifier Training and Testing Method

Cross validation is a common technique for estimating the performance of a classifier. One of the main reasons for using cross-validation instead of using the conventional validation (e.g., partitioning the data set into two sets of 70% for training and 30% for test) is that the error (e.g., Root Mean Square Error) on the training set in the conventional validation is not a useful estimate of model performance and thus the error on the test data set does not properly represent the assessment of model performance. This may be because there is not enough data available or there is not a good distribution and spread of data to partition it into separate training and test sets in the conventional validation method. In these cases, a fair way to properly estimate model prediction performance is to use cross-validation as a powerful general technique. In summary, cross-validation combines (averages) measures of fit (prediction error) to correct for the optimistic nature of training error and derive a more accurate estimate of model prediction performance [80].

In our experiment, we use stratified k-fold ($k = 10$) cross validation method to test the classifier. The original dataset is randomly partitioned into k equal sized subset in such a way that each partition resembles the global distribution of the dataset. Of the k subsets, a single subset is retained as the validation data for testing the model, and the remaining $k - 1$ subsets are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsets used exactly once as the

validation data. The *k* results from the folds are then averaged to produce a single estimation. Fig. 3.3 shows an example iteration of *k*-fold cross validation.

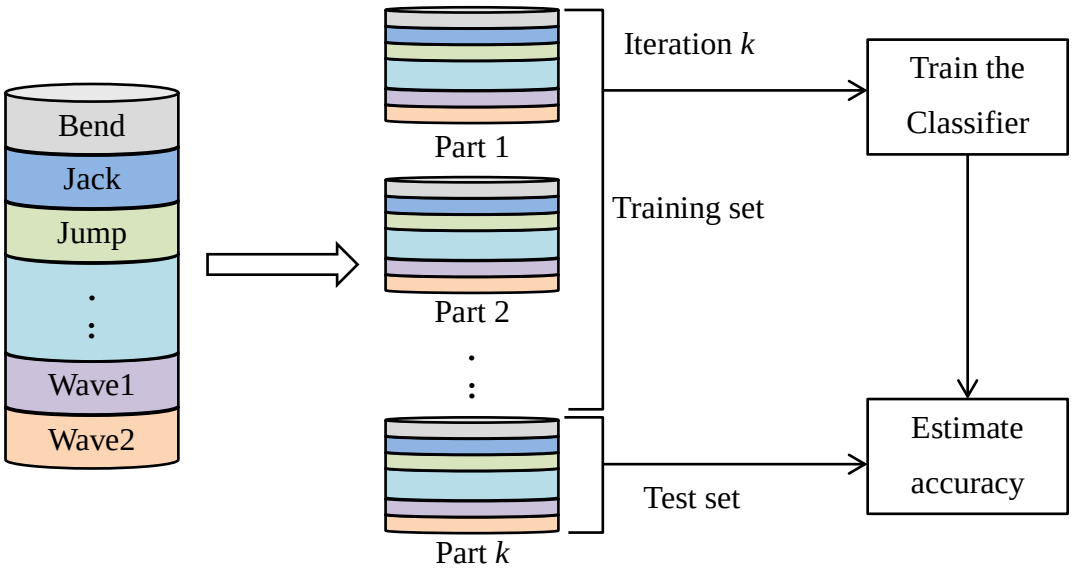


Figure 3.3 Example of stratified *k*-fold cross validation method.

During *k*-nearest neighbor classification there is no training phase, but for SVM classification, the multiclass action classification problem is reduced down to binary classification one. To recognize *k* action classes, we train a bank of *k* linear one-vs-rest and ${}^{k}C_{2}$ linear one-vs-one binary SVMs, each with identical weights. During the test, final decision is made on a max polling method, i.e., an unknown action is labeled with the class that gets the maximum votes by those SVM classifiers. We deliberately keep the classification part simple. One alternative would be to use multi-class SVM or to use a non-linear kernel for the SVM. However, using non-linear kernel yields very little improvement due to the high dimension of the feature vector.

3.1.3 Evaluation Terminology

The correctness of a classification can be evaluated by computing the number of correctly recognized class examples (true positives), the number of correctly recognized examples that do not belong to the class (true negatives), and examples that either were incorrectly assigned to the class (false positives) or that were not recognized as class

Table 3.1 Confusion matrix for binary classification.

		Classified as	
		Positive	Negative
Data Class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

examples (false negatives). These four counts constitute a confusion matrix shown in Table 3.1 for the case of the binary classification.

For multi-class classification, we use the following performance measures for each individual class C_i, $i = 0, 1, \ldots, l$, and the assessment is defined by TP_i, FN_i, FP_i, TN_i [81] and then compute the macro average performance measure i.e. take the average of the same measures calculated for C_i.

Precision

Precision is defined as the ratio of correctly recognized actions among the total number of actions that are recognized correctly or incorrectly. Eq. (3.1) defines the precision of class *i*, and Eq. (3.2) gives an average per-class precision of the data class labels with those of classifiers.

$$Precision_i = \frac{TP_i}{TP_i + FP_i} \times 100\% \tag{3.1}$$

$$Avg.\ Precision = \frac{1}{l}\sum_{i=1}^{l} Precision_i \tag{3.2}$$

Recall

Recall is defined as the percentage of successfully recognized actions among the total number of relevant actions n the dataset. Eq. (3.3) defines the recall of class *i*, and Eq. (3.4) gives an average per-class recall of a classifier to identify class labels.

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \times 100\% \tag{3.3}$$

$$Avg.\ Recall = \frac{1}{l}\sum_{i=1}^{l} Recall_i \tag{3.4}$$

False Positive Rate (FPR)

FPR is a measure of how often the classifier assigns a class label to an action when it does not belong to that class, i.e., it is the ratio of incorrectly recognized actions among all the test actions other than that action. Mathematically it is expressed by the Eqs. (3.5) and (3.6).

$$FPR_i = \frac{FP_i}{FP_i + TN_i} \times 100\% \tag{3.5}$$

$$Avg.\ FPR = \frac{1}{l}\sum_{i=1}^{l} FPR_i \tag{3.6}$$

Recognition Rate

The overall recognition rate or the balanced accuracy of the classifier is defined as the percentage of successfully recognized actions among the total number of actions. It can be defined by Eq. (3.7).

$$Recog.Rate = \frac{Number\ of\ correct\ recognition}{Total\ no.\ of\ test\ actions} \times 100\% \tag{3.7}$$

F Score

F score is the weighted harmonic mean of the precision and recall. We use a macro averaged *F* score, given by Eq. (3.8) to evaluate the classifier.

$$Fscore = \frac{2 \times Avg.Precision \times Avg.Recall}{Avg.Precision + Avg.Recall} \tag{3.8}$$

3.1.4 Experimental Parameters and Abbreviations

For tracking the optical flow, dense Gunnar Farneback [82] algorithm is used and a flow threshold of ±1 pixel is used to separate the x and y directional flow into right ($+x$), left ($-x$), up ($+y$), and down ($-y$) directions. We use temporal duration, $\tau = 0.9$ second and decay parameter, $\delta = 1$ (as it was in the original) in Eq. (2.10), i.e., only 0.9 second frames are used to create DMHI templates. Each DMHI is used to calculate the LBP image, where 3×3 neighborhood and *threshold* =1 are used for Eqs. (2.18) and (2.19). We use the same values for the parameters (p, q) used to partition the action region into blocks during feature vector generation. We use $p = q = 2, 4, 6$ when experimenting with Weizmann dataset and for KTH dataset we use $p = q = 2, 4, 6, 8$. The number of histogram bins, *are* = 8, 16, 32 is used for the both datasets. For the linear combination parameter α, we use values from 0 to 1, with a discrete step increase of 0.1. When using k - nearest neighbor classifier, we use the nearest neighbor parameter $k = 5$.

Although we have explained two methods for action representation in Section 2.3.6, we performed the experiment with some close variants of the method described in Section 2.3.6.1 and present the comparative results. For example, rather than using four DMHI, we can create a single MHI and then extract the textures from MHI with similar manner presented in Section 2.3.6.1. The representation methods used in the experiment are:

(i) Histogram of LBP image created from MHI (LBPM_H),

(ii) LBPM_H along with the randomly selected snippets histogram (LBPM_RNDSN_H),

(iii) LBPM_H along with the selective snippets (as described in Section 2.3.4) histogram (LBPM_SELSN_H),

(iv) Histogram of rotated bit arranged LBP image created from DMHI (RLBPD_H),

(v) RLBPD_H with random snippets histogram (RLBPD_RNDSN_H),

(vi) RLBPD_H with selective snippets histogram (RLBPD_SELSN_H),

(vii) Histogram of constant bit arranged LBP image created from DMHI (CLBPD_H), and its variants CLBPD_SELSN_H.

All these methods named above are used only when experimenting on Weizmann dataset and with a SVM classifier. In case of random snippets, rather than selecting the snippets based on any information (as described in Section 2.3.4), we simply choose three snippets within the DMHI time duration τ just in a random order.

In section 2.3.6.2 we propose a variant action representation method which is employed during the experiment using the KTH dataset. The abbreviations used for that representation are:

(i) Histogram of rotated bit arranged LBP image created from DMHI (RLBPD_H),

(ii) Histogram created from DMHI (DMHI_H) (details in Section 2.3.6.2),

(iii) Histogram extracted from MEI (MEI_H) as a shape feature (details in Section 2.3.6.2),

(iv) A linear combination of RLBPD_H and DMHI_H is shortened as D_RLBPD_H

(v) D_RLBPD_H along with MEI histogram (D_RLBPD_MEI_H),

3.2 Recognition Results

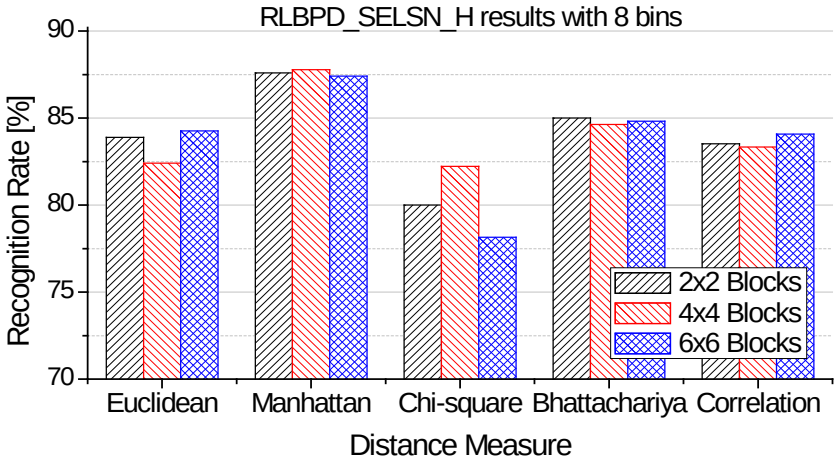
As we mentioned earlier, we use two benchmark action dataset for the experiment, the following section describes the results obtained on those datasets.

3.2.1 Results on Weizmann Dataset

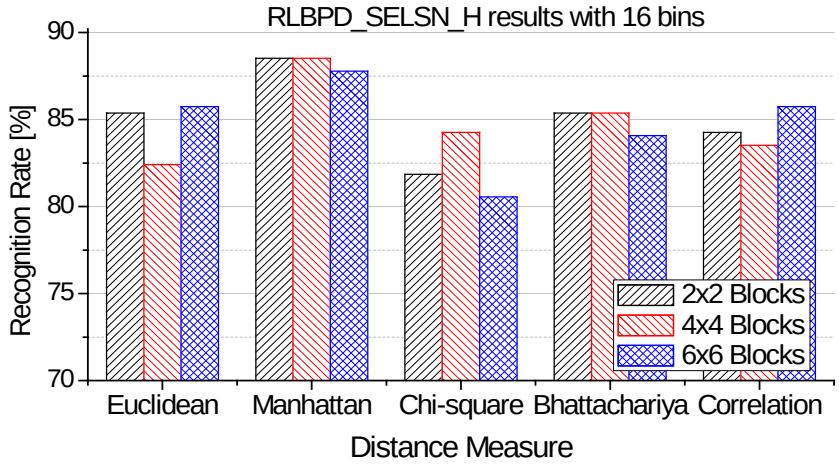
3.2.1.1 Results using *k*-NN Classifier

Fig. 3.4 shows the recognition rate obtained for the proposed action representation method RLBPD_SELSN_H by using *k*-NN classifier (k = 5) on the Weizmann dataset with different number of blocks, bins and distance or similarity measures. The results, presented here, are the average of three runs, i.e., the experiment is performed thrice, and each run consists of 10-fold cross validation. We notice that in all cases, using

(a)



(b)



(c)

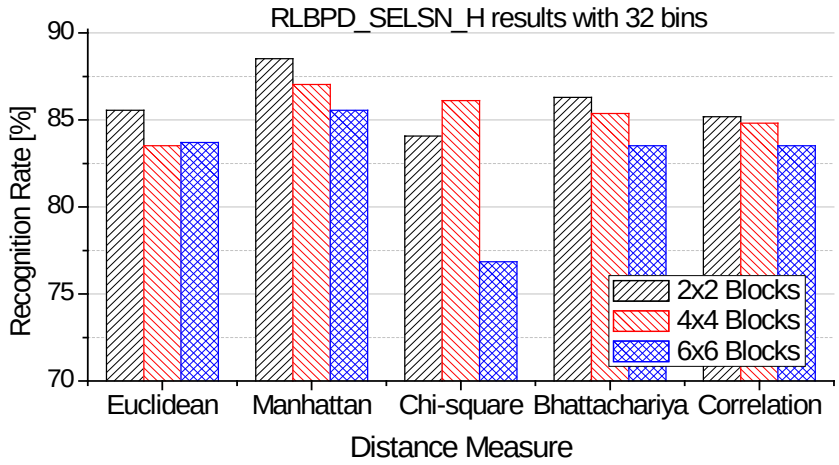


Figure 3.4 Recognition rates by using the *k*-NN classifier on Weizmann dataset for the representation RLBPD_SELSN_H with various numbers of blocks, distance metric, and (a) 8 bins, (b) 16 bins, (c) 32 bins.

Manhattan distance produces better results than other distance measures for same *p*, *q*, *r* values. The best found recognition rate is 88.52% for Manhattan distance, however, for all the cases, the overall recognition rate is below 90%.

Fig. 3.5 presents the *k*-fold iteration results found by *k*-NN classifier for the proposed method for specific $p = q = 4$, $r = 16$ values. In each iteration, the same training set and test set are used for *k*-NN classifier with different distance measurement

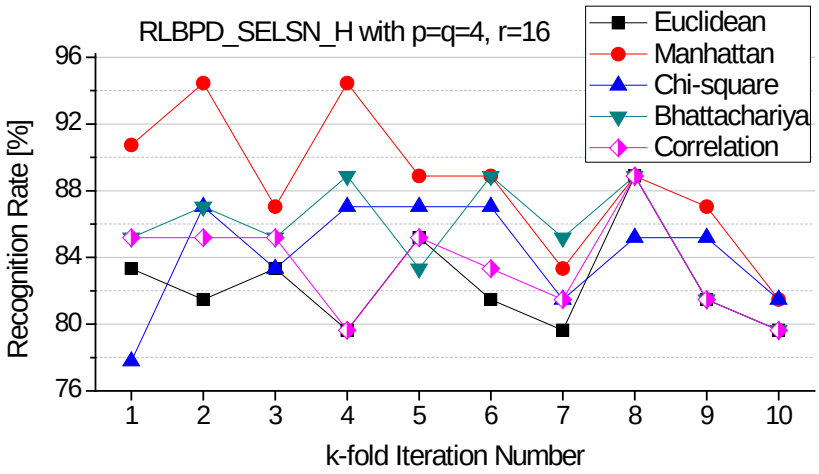


Figure 3.5 Recognition rates of RLBPD_SELSN_H representation found by the *k*-NN classifier for different *k*-fold iteration.

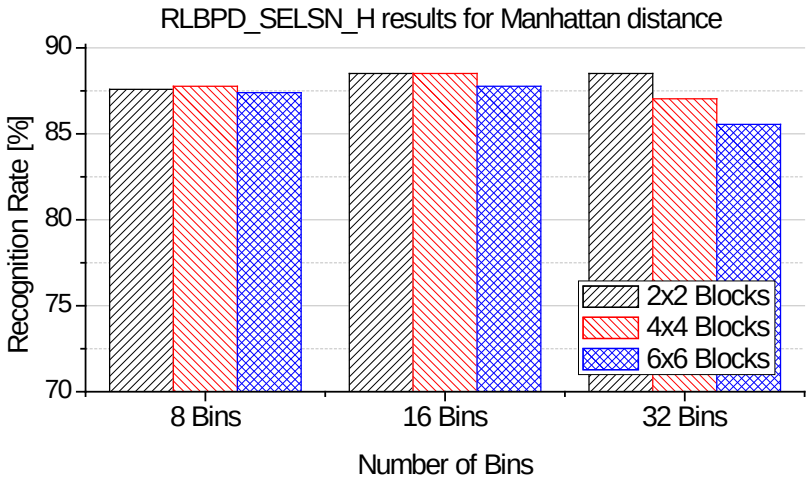


Figure 3.6 Relation between the number of blocks and histogram bins of RLBPD_SELSN_H with Manhattan distance for the *k*-NN classifier.

methods. We find that in almost every iteration Manhattan distance outperforms the other measuring methods.

Fig. 3.6 shows the effect of number of blocks and number of histogram bins on recognition rate. The relation is shown in Fig. 3.6 for a specific distance measure, Manhattan distance, in *k*-NN classification. We notice that the classifier generates almost identical result for 2×2 blocks with different bins. However, for a constant bin number, increasing the number of partition blocks does not always increase the recognition rate, rather it falls out, e.g., the recognition rate with 32 bins in Fig. 3.6.

Table 3.2 shows the confusion matrix of RLBPD_SELSN_H representation for the best recognition rate found by the *k*-NN classifier. The results are for Manhattan

Table 3.2 Confusion matrix for RLBPD_SELSN_H representation on Weizmann dataset using Manhattan distance and the *k*-NN classifier.

Actual \ Predicted	Bend	Jack	Jump	Pjump	Run	Side	Skip	Walk	Wave1	Wave2
Bend	1	0	0	0	0	0	0	0	0	0
Jack	0	1	0	0	0	0	0	0	0	0
Jump	0	0	0.83	0	0	0.09	0.07	0	0	0
Pjump	0	0	0	1	0	0	0	0	0	0
Run	0	0	0	0	1	0	0	0	0	0
Side	0	0	0.11	0	0	0.83	0	0.06	0	0
Skip	0	0	0.06	0	0.54	0	0.39	0.01	0	0
Walk	0	0	0	0	0.13	0.07	0	0.80	0	0
Wave1	0	0	0	0	0	0	0	0	1	0
Wave2	0	0	0	0	0	0	0	0	0	1

distance with $p = q = 4$, and $r = 16$. Here, the recognition rate is scaled down to one. All actions except for skipping the recognition rate are above 80%. Most of the skipping actions are labeled as running, because the skipping poses have very subtle difference from the running poses.

Table 3.3 shows the per-class and their macro averaged precision, recall, and FPR for *k*-NN classifier with Manhattan distance, and $p = q = 4$, and $r = 16$. Again we notice that the classifier provides maximum possible recall for running action, i.e., the classifier recognizes all the test running actions accurately, but the precision or consistency of recognition is low, which can also be seen by the FPR value. On the other hand, even if the classifier provides low recall for skipping action, but the precision is quite high. Since walking action bears similar poses with running action, we can see the recall of the walking action is in a little bit lower side (better seen in Table 3.2,

Table 3.3 Per-class precision, recall, and FPR of RLBPD_SELSN_H on Weizmann dataset using Manhattan distance and the *k*-NN classifier.

	Precision [%]	**Recall [%]**	**FPR [%]**
Bend	100	100	0
Jack	100	100	0
Jump	83.33	83.33	1.85
Pjump	100	100	0
Run	60	100	7.41
Side	83.33	83.33	1.8
Skip	84	38.89	0.82
Walk	91.49	79.63	0.82
Wave1	100	100	0
Wave2	100	100	0
Average	90.22	88.52	1.28

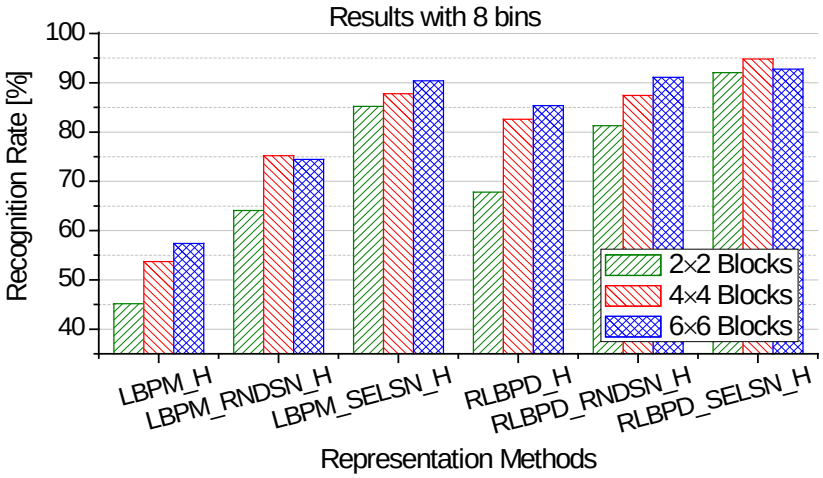
where some walking actions are recognized as running). The macro averaged *F* score for the recognition results presented in Table 3.3 is 89.4.

3.2.1.2 Results using SVM Classifier

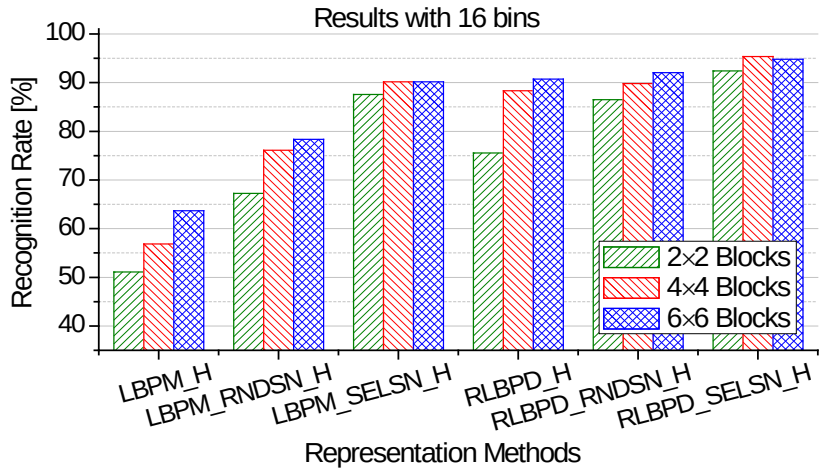
Fig. 3.7 shows the correct recognition rate of different representation methods, mentioned in section 3.1.4, for different number of blocks and bins using SVM classifier on Weizmann dataset. For all the cases, RLBPD_SELSN_H representation shows a better accuracy than other action representation methods having similar parameter values (same *p, q, r* values). We find that including the shape feature (i.e., a snippet histogram) in action representation greatly improves the recognition rate for LBPM_SELSN_H, but in case of RLBPD_SELSN_H, the impact is more for lower number of blocks. However, in all cases, including the shape feature increases overall performance. Also in Fig.3.7, increasing the number of bins and blocks does not linearly increase the accuracy, rather after some point it starts to fall down (e.g., 6×6 blocks and 8, 16 bins). The best found recognition rate is 95.37%, which is observed for RLBPD_SELSN_H with $p = q = 4$ and $r = 16$ bins. The result found by SVM classifier is higher than that of *k*-NN classifier that reports an accuracy of 88.52%.

Fig. 3.8 presents the comparison of using rotating bit arrangement (as it is described in Section 2.3.3) for creating different LBP images with that of a constant bit arrangement. Here CLBPD_H means histogram of constant bit arranged LBP image created from DMHI, i.e., histograms are created in similar fashion presented in Section 2.3.6.1. But we use a single layout of LBP neighborhood for all the DMHIs to extract the texture pattern. We find that in every combination of *p, q,* and *r* values, RLBPD_H (or its variant that includes shape information as a form of snippet histogram) performs better than CLBPD_H which justify our claim presented in section 2.3.3. This is because, if the same actions performed in leftward or rightward direction (e.g., walk, run, side-walk), the rotated arrangement of LBP tends to produce similar pattern values in the left or the right LBP image. These images in turn yield a histogram which is more consolidated and helps in better classification. The same thing also applies to upward or downward actions [52]. The average improvement of recognition rates by using rotated bit arranged LBP are 4.61% and 2.45% for without using and with using SELSN_H.

(a)



(b)



(c)

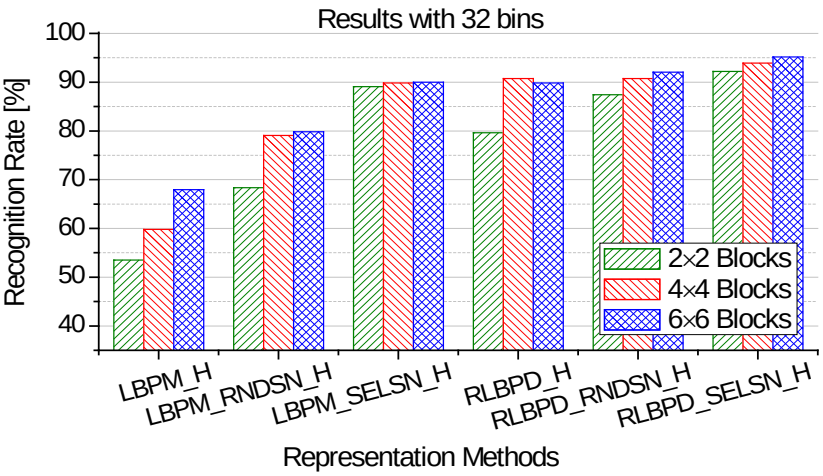


Figure 3.7 Correct recognition rates using the SVM classifier on Weizmann dataset for different representations with various numbers of blocks and (a) 8 bins, (b) 16 bins, (c) 32 bins.

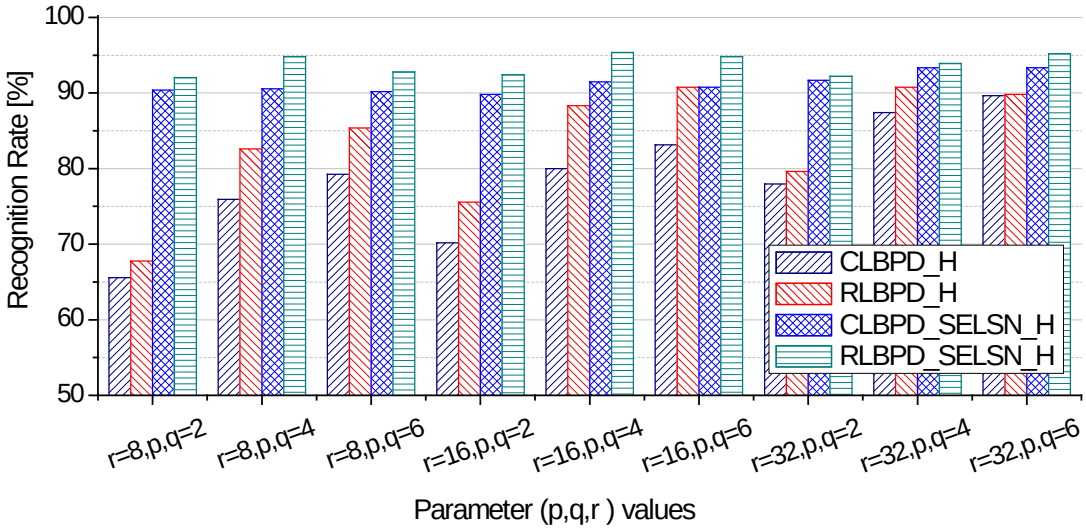


Figure 3.8 Performance comparison of using rotated and constant arranged bits for LBP image creation using SVM classifier on Weizmann dataset.

Table 3.4 shows the confusion matrix of RLBPD_SELSN_H representation for the best result using SVM classifier on Weizmann dataset. Here also recognition rates are scaled down to one for a better viewing. Like *k*-NN classifier, the classifier gives poor result for skipping actions. But SVM classifier provides much better recognition rate, 76%, compared to *k*-NN classifier (where it was 39% only). This time, the recognition rates of all the actions except for skipping are approximately 90% or above. As we mention earlier, in Weizmann dataset, some frames of the skipping poses have very subtle difference from the running poses, which is very difficult to distinguish even by a human. Therefore the classifier puts some skipping actions as running ones.

Table 3.5 shows the per-class and their macro averaged precision, recall, and FPR of RLBPD_SELSN_H representation for the SVM classifier with $p = q = 4$, and $r = 16$. Here we notice that the SVM classifier provides a recall for running action which is lower than the *k*-NN classifier, but better precision or true positive accuracy. We can also see from the table that, both skipping and walking actions generate improved recall as well as precision values than before. The overall per-class FPR is also reduced by using the SVM classifier. The macro averaged *F* score for the recognition results presented in Table 3.5 is 95.38.

Table 3.4 Confusion matrix for RLBPD_SELSN_H representation on Weizmann data set using the SVM classifier.

Actual \ Predicted	Bend	Jack	Jump	Pjump	Run	Side	Skip	Walk	Wave1	Wave2
Bend	1	0	0	0	0	0	0	0	0	0
Jack	0	1	0	0	0	0	0	0	0	0
Jump	0	0	0.87	0	0	0.04	0.09	0	0	0
Pjump	0	0	0	1	0	0	0	0	0	0
Run	0	0	0	0	0.91	0	0.04	0.05	0	0
Side	0	0	0	0	0	1	0	0	0	0
Skip	0	0	0.04	0	0.18	0	0.76	0.02	0	0
Walk	0	0	0	0	0	0	0	1	0	0
Wave1	0	0	0	0	0	0	0	0	1	0
Wave2	0	0	0	0	0	0	0	0	0	1

Table 3.6 summarizes the best recognition rates found by our experiment with different action representations as well as the best results reported by other methods in the literature on Weizmann dataset. The presented best result of the proposed RLBPD_SELSN_H representation is for the parameter p, q = 4, and r = 16. We include the results of both classifiers (k-NN, and SVM) we experiment with. All the methods use a SVM classifier except the one shown in parenthesis. These results are just indicative only, since different authors used a different number of frames for the feature vector generation or different types of classifiers and even different testing method like leave-one-out.

Some researchers perform their experiments with 9 actions from Weizmann dataset excluding the skipping action. We also do the same only for the proposed RLBPD_SELSN_H representation, since this representation method gives the best performance for all 10 actions. The best average recognition rate found by the proposed method for 9 actions is also presented in the Table 3.6 (parenthesized in some cases). Though the recognition rate found from the proposed method is not the best compared to other methods, the achieved accuracy is reasonable enough and quite fast (See Table 3.11 for computational time) for practical application.

Table 3.5 Per-class precision, recall, and FPR of RLBPD_SELSN_H representation on Weizmann dataset using a SVM classifier.

	Precision [%]	**Recall [%]**	**FPR [%]**
Bend	100	100	0
Jack	100	100	0
Jump	95.92	87.04	0.41
Pjump	100	100	0
Run	83.05	90.74	2.06
Side	96.43	100	0.41
Skip	85.42	75.93	1.44
Walk	93.10	100	0.82
Wave1	100	100	0
Wave2	100	100	0
Average	95.39	95.37	0.51

Table 3.6 Comparison of the recognition rate of the proposed method to other methods reported based on Weizmann Dataset.

	Reference	**No. of Actions used**	**Recognition Rate [%]**
Proposed method and its variants	LBPM_H	10	67.96
	LBPM_RNDSN_H	10	79.81
	LBPM_SELSN_H	10	90.37
	RLBPD_H	10	90.74
	RLBPD_ RNDSN_H	10	92.04
	RLBPD_ SELSN_H	10, (9)	95.37, (98.96)
	RLBPD_ SELSN_H (*k*-NN)	10	88.52
	CLBPD_ H	10	89.63
	CLBPD_ SELSN_H	10	93.33
Archived in literature	Kelllokumpu et al. [31]	10, (9)	98.9, (100)
	Scovanner et al. [83]	10	82.6
	Boiman and Irani [84]	9	97.5
	Neibles and Fei-Fei [62]	9	72.8
	Wang and Suter [17]	10	97.8
	Campos et al. [85]	10	96.7
	Ikizler and Duygulu [40]	9	100

3.2.2 Results on KTH Dataset

Each action video sequence is preprocessed to extract the foreground. In this phase, we extract the morphological gradient of the current frame which serves as a foreground mask. This mask generally contains lots of noise which is minimized by

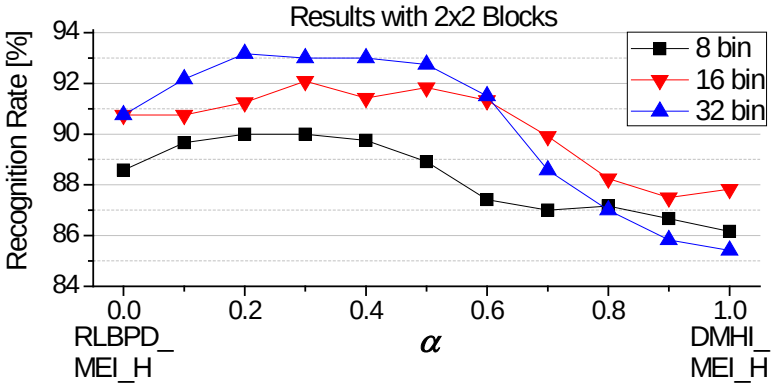
using the information of the previous frame. The method used here is a simplified version of the method presented in [86] which provides enough information for reasonable recognition. In case of KTH dataset, we only use a SVM classifier, so all the results presented in this section are obtained from a SVM classifier. Besides, all the presented results are average of at least two runs of the experiment.

Fig. 3.9 shows the correct classification rate (averaged on 6 actions of KTH dataset) of the proposed method D_RLBPD_MEI_H (described in Section 2.3.6.2) for different representation with various numbers of blocks and bins where one large set is considered as the data set. Abscissa of the graphs denotes different action representation methods with discrete values of α with a step difference 0.1 of Eq. (2.31), i.e., the leftmost results are for RLBPD_MEI_H, the rightmosts are for DMHI_MEI_H, and the in-between results are for D_RLBPD_MEI_H representation. In Fig. 3.9, for all cases RLBPD_MEI_H action representation method produces better accuracy than its corresponding (same p, q, r values) DMHI_MEI_H representation. But for D_RLBPD_MEI_H representation, the curves show that its accuracy increases up to a certain value of α (0.3-0.5) and then goes down again, i.e., the linear combination of the aforementioned representation method improves the accuracy. We obtain a maximum recognition rate of 95.6% for D_RLBPD_MEI_H representation. LBP operator highlights the patterns or texture lying in DMHI but loses the recency of motion. Hence, mixing a certain amount of DMHI_H information to the RLBPD_H gives better accuracy than RLBPD_H alone.

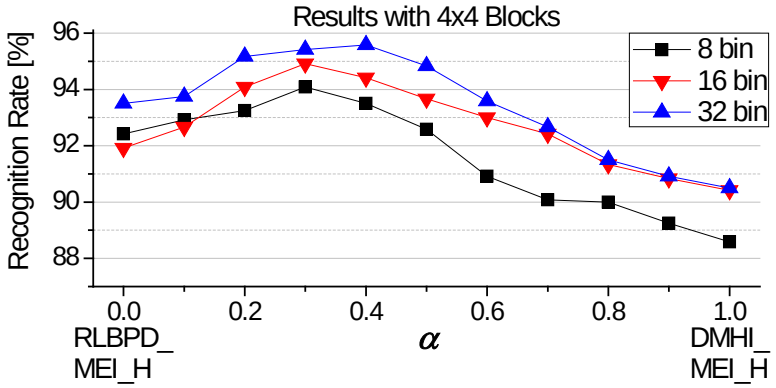
Also in Fig. 3.9, almost all cases, for a constant number of blocks, the higher number of bins produces better results. However, the reverse is not true. It can be better understood from Fig. 3.10 which shows the relation between the number of blocks and histogram bins for D_RLBPD_MEI_H with only for a specific $\alpha = 0.4$. It is clear from the Fig. 3.10 that partitioning the action region into more blocks does not improve the overall recognition rate for a specific number of bins.

Fig. 3.11 presents the average performance increase or gain in classification accuracy of D_RLBPD_MEI_H over D_RLBPD_H representation, i.e. the importance of using shape information as a form of MEI histogram. Keeping each p, q, r parameter

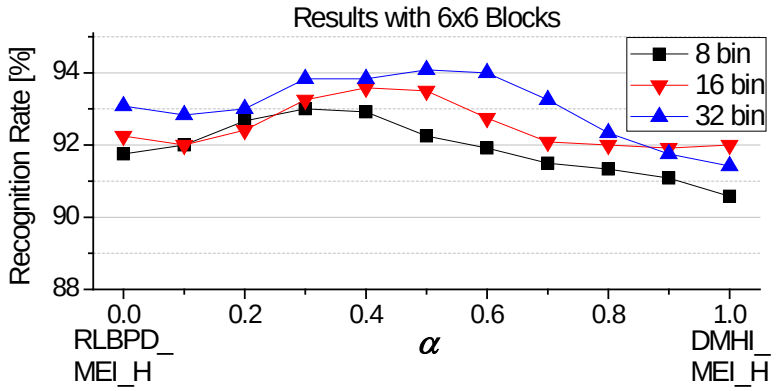
(a)



(b)



(c)



(d)

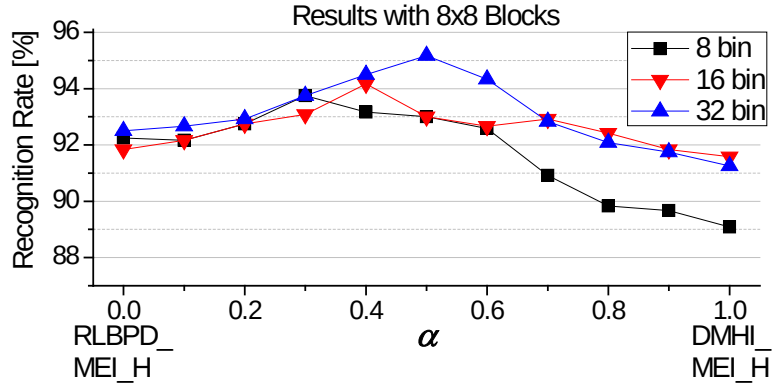


Figure 3.9 Recognition rates for different representation on KTH dataset with various numbers of bins and (a) 2×2, (b) 4×4, (c) 6×6, (d) 8×8 blocks.

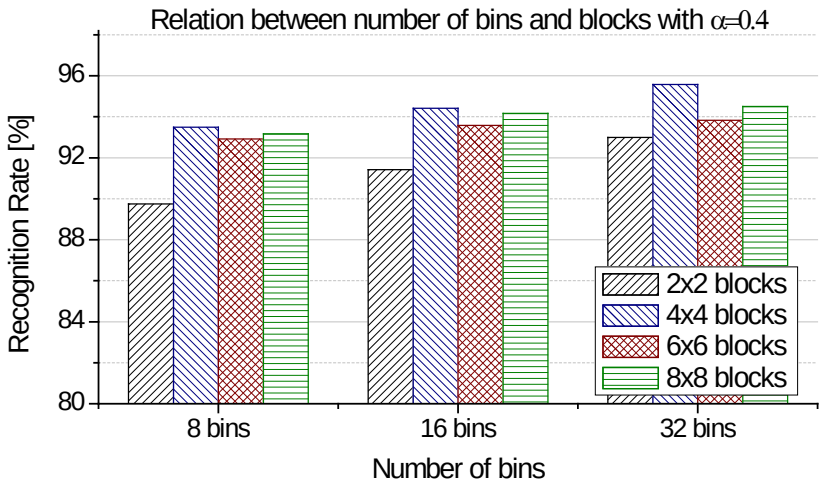


Figure 3.10 Relation between the number of blocks and histogram bins for D_RLBPD_MEI_H with $\alpha = 0.4$ on KTH dataset.

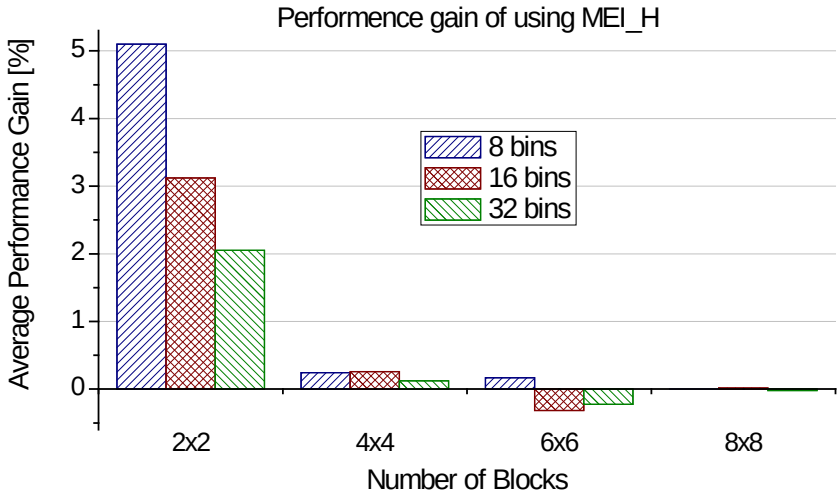
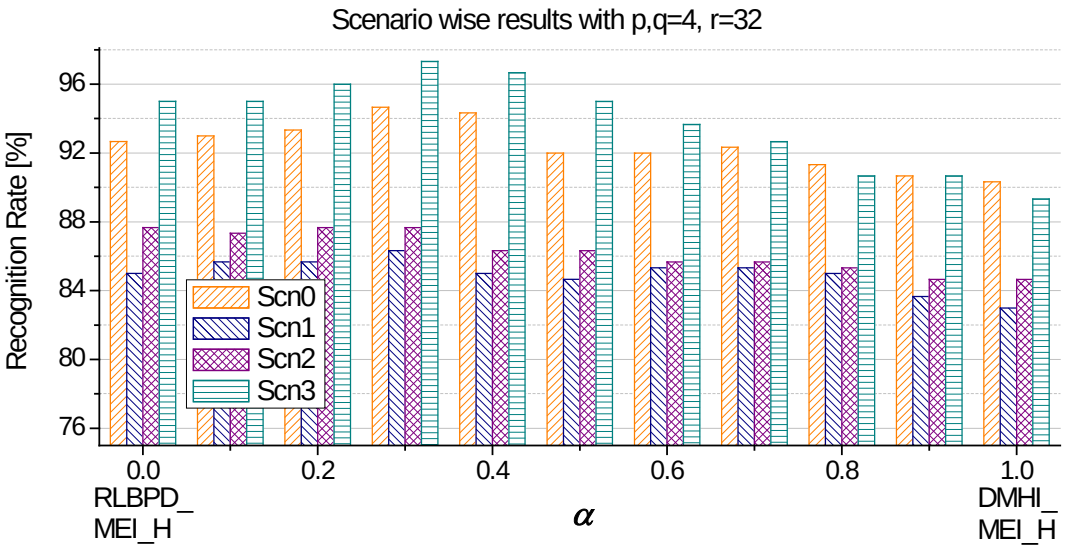


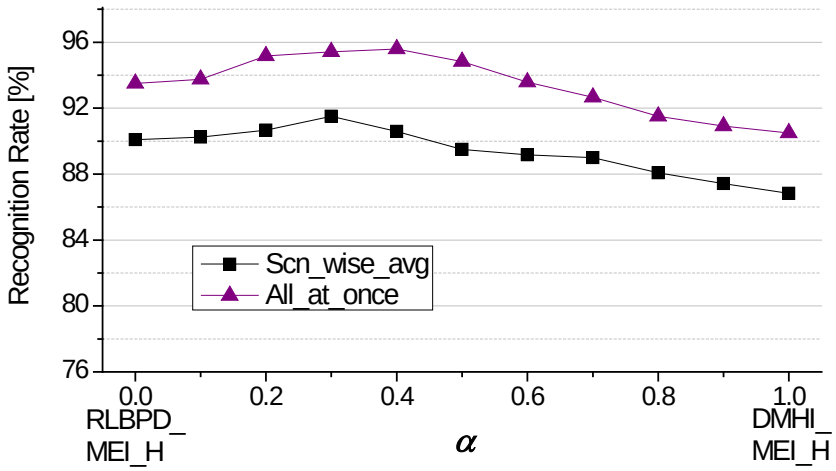
Figure 3.11 Average performance gain of D_RLBPD_MEI_H over D_RLBPD_H on KTH dataset.

values constant, the performance gain is measured using Eq. (3.9). The *R.R* along with the suffixes in Eq. (3.9) means the *Recognition Rate* (defined by Eq. (3.7)) for that specific parameter values given in the suffixes. We can see from Fig. 3.11 that, using the MEI_H along with D_RLBPD_H improves the performance significantly only in a lower number of blocks (e.g. 2×2 blocks). But, if we increase the number of blocks, there is no substantial gain of using MEI_H or sometimes the gain is even negative [61].

$$Gain_{D\_RLBPD\_MEI\_H_{p,q,r}} = \frac{1}{11}\sum_{\alpha=0,0.1,\ldots,1}\left(R.R_{D\_RLBPD\_MEI\_H_{p,q,r,\alpha}} - R.R_{DLBP\_H_{p,q,r,\alpha}}\right) \tag{3.9}$$



(a)



(b)

Figure 3.12 Scenario-wise recognition rate of D_RLBPD_MEI_H with *p*, *q* = 4, *r* = 32 (a) recognition rate for different scenarios, (b) comparison of the average of scenario-wise results with that of all scenario dataset taken as one large dataset.

As we mentioned earlier that actions in the KTH dataset are recorded in four different scenarios, we perform the experiment considering them as an individual dataset. Fig. 3.12 (a) presents the recognition results for different representation methods when the classifier is trained and tested on the action performed at different scenarios with p, q = 4, r =32. We find that for all α values, the indoor (*Scn*3) and the outdoor (*Scn*0) scenarios provide better results than the outdoor with zooming (*Scn*1) and clothing (*Scn*2) variations. The best found results are 94.67%, 86.35%, 87.67%, and 97.35% for *Scn0*, *Scn1*, *Scn2*, and *Scn3*, respectively. It is obvious that using different clothing may hide some part of the body and consequently the action pose becomes different (e.g., long overcoat may hide some leg portion). Similarly, zooming the camera while performing an action, sometimes overwrites some motion information. However, the results are still above 85% for *scn1* and *scn2* in the best case (α = 0.3). Fig. 3.12 (b) shows the comparison of the average scenario-wise results with the result found from the experiment taking all scenario-wise datasets as a large single dataset for same p, q, r values. Clearly, the classifier provides better performance for the large single dataset, and we can say that the classifier learns better when there are some variation within the class.

The confusion matrix of D_LBPD_MEI_H representation for the best found result on the KTH dataset with parameters p,q = 4, r = 32, α = 0.4 is presented in Table 3.7. The recognition rate is above 90% in all actions except jogging. Jogging action has very subtle difference between walking and running, and it varies with the person performing the action. Therefore the classifier puts some jogging actions as walking or running action. However, the accuracy of jogging is among the top compared to some other methods [28].

Table 3.8 shows the per-class and their macro averaged precision, recall, and FPR of D_RLBPD_MEI_H representation with p,q = 4, r = 32, α = 0.4. Here we notice that the recall and precision for running and jogging action are a little bit poor compared to other actions. This is because the inter-class variation between running and jogging actions in the dataset is sometimes quite indifferent. Considering a pretty large dataset, the classifier learns well that can be seen from the overall macro averaged per-class FPR

which is below 1%. The best case *F score* of the classifier for the recognition results presented in Table 3.8 is 95.59.

Table 3.7 Confusion matrix of D_RLBPD_MEI_H representation with p, q = 4, r = 32, α = 0.4 on KTH dataset.

Actual \ Predicted	Box	HW	Clap	Walk	Run	Jog
Box	99	0	0	1	0	0
HandW	0	100	0	0	0	0
Clap	0	0	100	0	0	0
Walk	1	0	0	97.5	0	1.5
Run	0	0	0	0	90.5	9.5
Jog	0	0	0	1	12.5	86.5

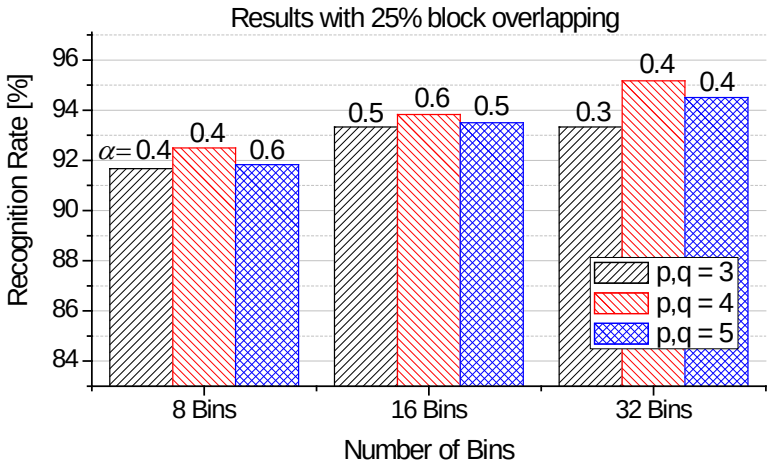
Table 3.8 Per-class precision, recall, and FPR of D_RLBPD_MEI_H representation on KTH dataset.

	Precision [%]	**Recall [%]**	**FPR [%]**
Box	99	99	0.20
HandWave	100	100	0.00
Clap	100	100	0.00
Walk	97.99	97.50	0.40
Run	87.86	90.50	2.50
Jog	88.72	86.50	2.20
Average	95.60	95.58	0.88

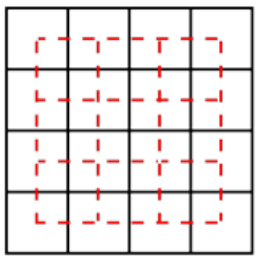
Though we mentioned in feature vector generation (Section 2.3.6) that we partition an action region into disjoint $p \times q$ blocks, but during an experiment with KTH dataset we use 25% block overlapping to see the result. Fig. 3.13(a) shows the recognition rate of D_RLBPD_MEI_H representation for different bins and p,q parameter values. We only present the best accuracies found for particular bins and p,q. The values that produce those results are shown on top of the histogram bars in Fig. 3.13(a). For this case, we observed the best possible recognition rate 95.17%, which does not surpass the result of the disjoint partitioning case (95.6%), rather block overlapping increases the feature vector dimension. Fig 3.13(b) shows an example of how the action region is partitioned into overlapping blocks. For the parameter, p,q, the number of blocks we can get for disjoint and overlapping partition are given by Eq. (3.10) and Eq. (3.10), respectively. Since block overlapping increases feature vector dimension, we use smaller p, q values during the experiment.

$$No.Blk_{DJ}(p,q) = p \times q \tag{3.10}$$

$$No.Blk_{OV}(p,q) = p \times q + (p-1) \times (q-1) \tag{3.11}$$



(a)



(b)

Figure 3.13 Results for overlapping blocks (a) best recognition rate for D_RLBPD_MEI_H with α value on top of the bar, (b) how the blocks are overlapped for a parameter value p,q = 4.

All the recognition rate mentioned above are per video result, however we apply our D_RLBPD_MEI_H representation to generate per frame recognition result. In this case we randomly choose 72 videos among the 600 videos in KTH dataset. Though videos are chosen randomly, we make a constraint that for each action class there are at least two videos for each of the four recording scenarios (outdoor, outdoor with cloth, outdoor with scale, indoor). These video frames were used to test with the already trained SVM classifiers (we have 10 classifiers, since we use 10-fold cross validation). These test frames are absolutely new to any of the 10 classifiers, since they are not used in the training phase at all. The best per frame average recognition rate, we found, is 86.42% with p,q = 8, r = 16, α = 0.2. However, using the ensemble decision of the classifiers, the recognition rate improves to 87.05%. Table 3.9 presents the per frame recognition rate for each action class and the number frames used in the testing process. Here, again, we can see that most of the confusion occurs with running and jogging action and thereby producing a low accuracy for each of them.

Table 3.10 summarizes the best experimental recognition rate of different action representation methods. Here, DMHI_MEI_H and RLBPD_MEI_H results are for p=q=6/4, r=16/32 and RLBPD_MEI_H results are for p,q = 4, r = 32, α = 0.4. The table also includes the best result reported on literature by other state of the art methods

Table 3.9 Frame level recognition rate of D_RLBPD_MEI_H on KTH dataset.

	No. of Frames Tested	**Recognition Rate [%]**
Box	300	100.0
HandWave	300	84.7
Clap	300	99.7
Walk	295	89.5
Run	150	62.7
Jog	215	68.4

on KTH dataset. It is worthy to mention that, these results are not directly comparable. Since, different authors used different classifiers even different testing method such as leave-one-out or different dataset splitting techniques. Overall, the result found by the proposed method is among the top-listed results that have been reported in the literature regarding KTH dataset.

Table 3.10 Comparison of accuracy of the proposed method to other state of the art methods reported based on KTH dataset.

	Reference	**Recognition Rate [%]**
Proposed	DMHI_MEI_H	92.0
	RLBPD_MEI_H	93.5
	D_RLBPD_MEI_H	95.6
	D_RLBPD_MEI_H (per frame)	87.1
Archived in literature	Schuldt et al. [28]	71.7
	Masumitsu et al. [87]	79.9
	Ke et al. [77]	80.9
	Dollar et al. [19]	81.2
	Niebles et al. [88]	81.5
	Ikizler et al. [40]	89.4
	Schindler [63]	90.1
	Wong et al. [79]	91.6
	Maninis et al. [89]	93.5
	Kellokumpu et al. [31]	93.8
	Kim et al. [78]	95.3

3.3 Computational Time

The run time of the method can be parted in two phases. First being the time to create the spatiotemporal templates, and the second is the feature vector generation and testing time, which depend on the values of p, q, r and α. Table 3.11 shows the average of per frame computational times (in milliseconds) of different phases for RLBPD_SELSN_H representation over Weizman dataset. The feature vector generation and testing times in Table 3.11 are for best recognition rate parameters p, q = 4, r = 16 and a frame size of 144×180 pixels. Table 3.12 presents the per frame execution time of the proposed D_RLBPD_MEI_H representation with parameter values p=q=4, r=32, α=0.4 and a frame size of 120×160 pixels. It should be noted that the experiment is done on a machine with a processor Intel® Core™ i7-3770, CPU speed 3.40 GHz, and memory 8GB. We implement the program in Microsoft Visual Studio 2010, and OpenCV 2.4.8 without applying any code optimization method. Moreover, it is worthy to mention that the total time reported in Table 3.11 and Table 3.12 excludes the foreground extraction time, since we are only interested in the action representation and recognition time. We are unable to compare the computational time of the proposed descriptor, since most of the state of the art methods does not report on their computa-

Table 3.11 Per frame computational time (in milliseconds) of the RLBPD_SELSN_H method on Weizmann dataset.

Template creation time	Feature vector generation time	Testing time	Total time
30.94	2.6	2.3	35.84

Table 3.12 Evaluation on the per frame execution time (in milliseconds) of the proposed D_RLBPD_MEI_H method on KTH dataset.

Template creation time			Feature vector generation time	Testing time	Total time
DMHI	LBP image	MEI & action Region			
19.49	6.67	3.83	10.0	5.95	45.94

tional times, except the HOR [40] that takes approximately one second per frame only for rectangle extraction phase which is far slower than the proposed one. We can see from the tables that in both cases the template creation time is almost identical, but D_RLBPD_MEI_H takes a little bit longer time than RLBPD_SELSN_H for feature vector generation and testing. This is obvious because, during the feature vector generation, D_RLBPD_MEI_H needs one extra step to combine the DMHI_H and RLBPD_H. Another reason is that the times are average of all possible p,q,r values: Actually, p,q = 2,4,6,8 in case of D_RLBPD_MEI_H, whereas RLBPD_SELSN_H uses p,q = 2,4,6.

3.4 Example of Recognition

In this section we present some action sequences in the form of image frames recognized by the SVM classifier for both Weizmann and KTH dataset for visual inspection. We label the actual action class and the recognized class on the image frame in blue and red color respectively.

Fig. 3.14 on the next page shows some samples of recognized action from Weizmann dataset. We can see from the figure that some sequences are misclassified; for example, running as walking, or skipping as running.

Fig. 3.15 shows some samples of recognized action from KTH dataset. Here also, we can notice that some jogging sequences are wrongly recognized as walking action.

Figure 3.14. Example of recognized action sequences from Weizmann dataset (continue to next page).

Frame 2 Frame 20 Frame 32 Frame 47

Frame 2 Frame 11 Frame 18 Frame 25

Frame 3 Frame 30 Frame 47 Frame 65

Frame 1 Frame 19 Frame 29 Frame 42

Frame 2 Frame 30 Frame 58 Frame 68

Figure 3.14 Example of recognized action sequences from Weizmann dataset (continued from previous page).

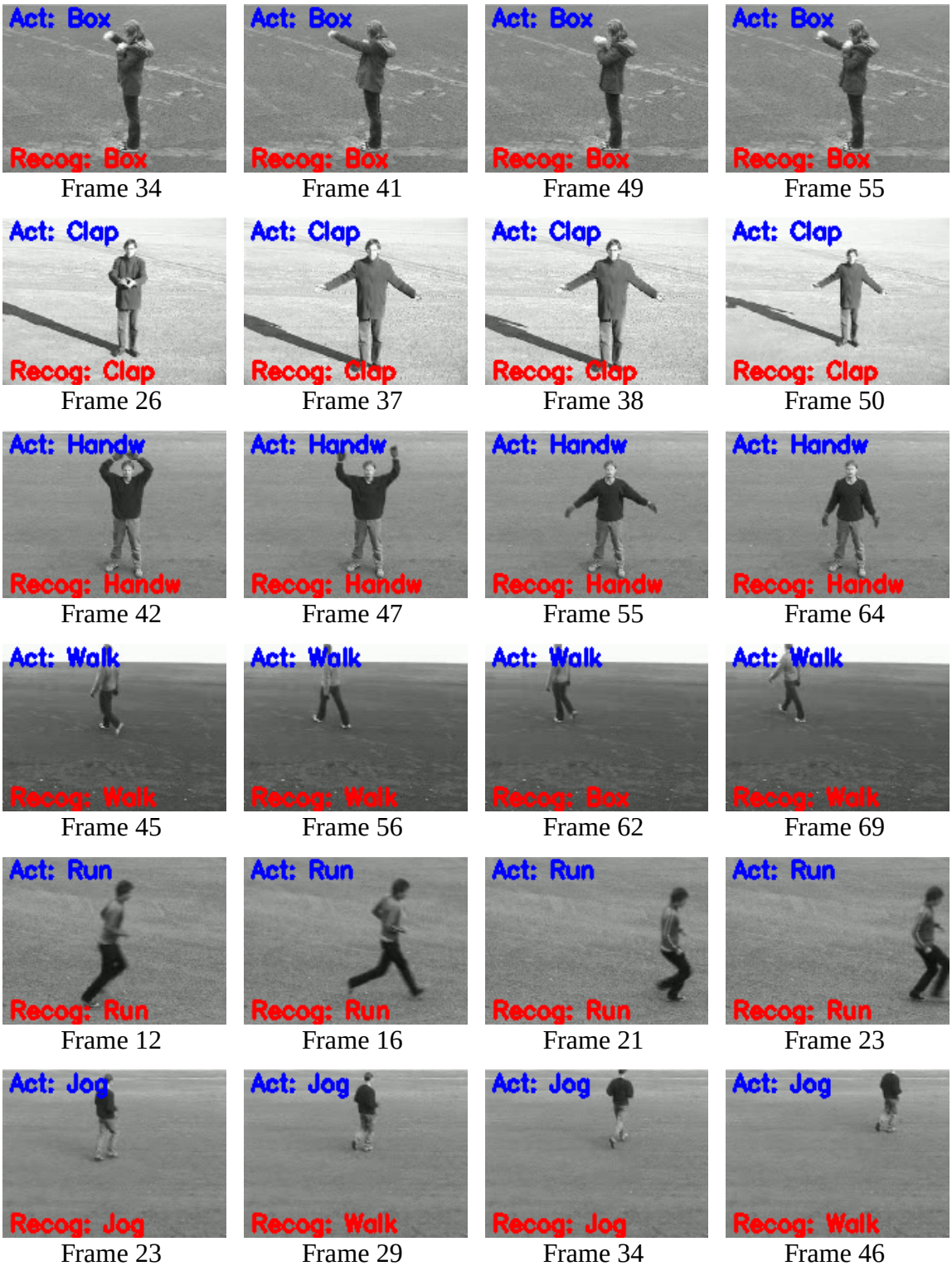


Figure 3.15 Example of recognized action sequences from KTH dataset.

3.5 Summary

In this chapter we have presented the details of how the experiment is performed, what evaluation methods are used, what were the dataset, etc. We have also presented the detail experimental results of the proposed action representation methods described in Chapter 2 along with some variants of it. The comparative results on action recognition rate with other state of the art methods were also presented. We found that the SVM classifier performs better than the *k*-NN classifier. Though the proposed action representation method fails in some cases, but in overall the average recognition rate is around the top list compared to existing methods.

Chapter 4

Conclusion

4.1 Thesis summary

In this thesis, we have approached to the problem of human action recognition from a texture descriptor perspective and proposed a novel descriptor that represents a human action as a histogram of LBP images created from DMHIs, i.e., a histogram of spatiotemporal texture. Our pose-descriptor is found to be simple and effective.

We create a temporal template from an action sequence as a form of DMHI. We then extract the spatiotemporal patterns that are present in the DMHI. We use a new idea of rotated bit arranged LBP for different DMHIs to extract the patterns. We formulate the patterns into histograms that serve as an action descriptor or a feature vector. Since the LBP tends to lose the temporal information in the DMHI, we combine the motion history information and texture information of an action sequence and found better results. We also use some variants that include the shape or pose information of the action. In this case, we choose some silhouettes of an action to extract the pose information. Rather than any temporal information we just use the size of the silhouette to select them. All the details of the proposed action representation method and the classifiers used for recognition are narrated in Chapter 2.

We show that by effective classification of such histograms, i.e., an action descriptor, robust human action recognition is possible. We demonstrate the effectiveness of our method over two benchmark dataset; the Weizmann dataset and KTH dataset. Our results are directly comparable/superior to the results presented over these datasets. We use a simple *k*-NN classifier and a SVM classifier for recognizing the actions. Though the SVM classifier performs better than the *k*-NN classifier, in both cases we have achieved reasonable recognition rates. Along with the proposed method, we experiment with some other similar methods: For example, rather than using DMHI,

we use MHI as a temporal template and extract the distribution of spatiotemporal texture from it. All the experimental results with various experimental parameters, with different evaluation methods, and the results of other existing methods are described in Chapter 3.

The novelty of this study is that we introduce an action descriptor that uses the distribution of texture patterns which exist in a temporal template like DMHI. To the best of our knowledge, this is a new method for representing an action. Moreover, we introduce a new way of creating a LBP image by using the rotated arrangement of LBP bits for different directional MHIs. The main objective is to describe an action as simple as possible with enough information for quick and reasonable classification. It has been shown that, without constructing any complex model, the proposed simple and compact descriptor performs well on different actions and the recognition rate is promising enough for practical use compared to the state of the art methods. Besides the recognition rate, the proposed technique is also advantageous with respect to computational load.

4.2 Discussion

The matching methods we present in this study suggest that we may not need a perfect modeling of the dynamics of human actions in order to reach satisfactory results. Our experiments show that the spatiotemporal textures of an action sequence encapsulate enough useful information for the action itself: Therefore, one can start with a good temporal template, before going into the details of dynamics.

Although we have achieved satisfactory performance for our proposed recognition system, there are, of course, some limitations in the current system. A more sophisticated foreground extraction method might increase the robustness of the recognition system. Especially in the KTH dataset, the contrast between foreground and background is very low; and hence, we observed that most of the misperception occurs due to the imperfect extraction of foregrounds. However, even with the noisy foreground information, our method reaches higher recognition rates, which means that the method is robust to noise. Since we only use the texture information of the action

region, we notice that, correct localization of the action region greatly affects the overall performance. Though we find that the descriptors that use the information from the DMHI template performs better than others, we observe that LBPM_SELSN_H (see Section 3.1.4 for elaboration) representation that extract texture distribution only from one MHI image performs quite well, too. We did not incorporate any direct mechanism for scale and rotation invariance or view point changes. However, our method successfully recognizes the actions with scale and view point changes which are present in KTH dataset. Actually, we always partition the action region in a constant number of blocks rather than fixed size blocks just to have some benefit for scale variation.

4.3 Future Scope of Works

This study is performed with a dataset having generic action classification problem. However, the method can easily be incorporated into some real life applications like gaming or human computer interaction without using any controller such as mouse, trackball, joystick, etc. The potential of the proposed method can also be applied to other related domains like a patient's activity monitoring system or automatic labeling of video sequences in a video dataset [61,52].

The application of the descriptor to more complex actions or scenarios could be other possible future work. The system is subjected to be comprehensively investigated in order to be practically implemented in crowded scenarios. The descriptor may be applied to achieve view invariant results by capturing the action from multiple camera views. The proposed method may be incorporated with a human detection system to recognize multiple persons' activities. The proposed scheme uses a fixed duration template to recognize video based recognition, therefore it performs poorly on per fame recognition. The scheme may be applied with a time window based templates to classify per frame actions for an improved results. However, this will probably increase the computational time, so, the decision is to be made on the nature of the recognition problem. Including the action performing speed could be a good choice to better separate the dynamic actions like walking, jogging and running.

References

[1] P. Bakker and Y. Kuniyoshi, "Robot see, robot do: An overview of robot imitation," *AISB96 Workshop on Learning in Robots and Animals*, pp. 3-11, 1996.

[2] S. Calinon and A. Billard, "Learning of gestures by imitation in a humanoid robot," *Imitation and Social Learning in Robots, Humans and Animals: Behavioural, Social and Communicative Dimensions*, C. L. Nehaniv and K. Dautenhahn, Eds. Cambridge Univ. Press, pp. 153-177, 2007.

[3] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka, "The development of Honda humanoid robot," *IEEE Int. Conf. on Robotics and Automation*, pp. vol2,1321-1326, 1998.

[4] K. Hirai, "Current and future perspective of Honda humamoid robot," *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. vol2,500-508, 1997.

[5] J. Peters and S. Schaal, "Reinforcement learning for parameterized motor primitives," *IEEE Int. Joint Conf. on Neural Networks*, pp. 73-80, 2006.

[6] A. Mcgovern and A. G. Barto, "Automatic discovery of subgoals in reinforcement learning using diverse density," *Int. Conf. on Machine Learning*, pp. 361-368, 2001.

[7] Y. Davidor, *Genetic Algorithms and Robotics: A heuristic strategy for optimization (Vol. 1)*. World Scientific, 1991.

[8] R. A. Brooks and M. J. Mataric, "Real robots, real learning problems," *Robot Learning*, J. Connell and S. Mahadevan, Eds. Springer, pp. 193-213, 1993.

[9] T. B. Moeslund, A. Hilton, and V. Kruger, "A survey of advances in vision-based human motion capture and analysis," *Computer Vision and Image Understanding*, vol. 104, no. 2-3, pp. 90-126, 2006.

[10] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine recognition of human activities: A survey," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1473-1488, 2008.

[11] R. Marks, “EyeToy, Innovation and Beyond,” 2010 [Online].

http://blog.us.playstation.com/2010/11/03/eyetoy-innovation-and-beyond/comment-page-2/#comment-478157

[12] A. Pentland, "Smart rooms, smart clothes," *Int. Conf. on Pattern Recognition*, pp. vol2,949-953, 1998.

[13] S. Sarkar, et al., "The humanID gait challenge problem: data sets, performance, and analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 162-177, 2005.

[14] D. Weinland, R. Ronfard, and E. Boyer, "A survey of vision-based methods for action representation,segmentation and recognition," *Computer Vision and Image Understanding*, vol. 115, no. 2, pp. 224-241, 2011.

[15] G. Johansson, "Visual perception of biological motion and a model for its analysis," *Perception & Psychophysics*, vol. 14, no. 2, pp. 201-211, 1973.

[16] A. S. Ogale, A. Karapurkar, G. Guerra-filho, and Y. Aloimonos, "View-invariant identification of pose sequences for action recognition," *Video Analysis and Content Extraction Workshop*, 2004.

[17] L. Wang and D. Suter, "Recognizing human activities from silhouettes: motion subspace and factorial discriminative graphical model," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1-8, 2007.

[18] I. Laptev and T. Lindeberg, "Space–time interest points," *Int. Conf. on Computer Vision*, pp. vol1 432-439, 2003.

[19] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," *IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 65-72, 2005.

[20] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using hidden Markov model," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 379-385, 1992.

[21] C. Bregler, "Learning and recognizing human dynamics in video sequences," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 568-574, 1997.

[22] T.-S. Wang, H.-Y. Shum, Y.-Q. Xu, and N.-N. Zheng, "Unsupervised analysis of human gestures," *IEEE Pacific Rim Conf. on Multimedia*, pp. 174-181, 2001.

[23] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *IEEE Int. Conf. on Computer Vision*, pp. 1395-1402, 2005.

[24] A. Yilmaz and M. Shah, "Action sketch: A novel action representation," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. vol1 984-989, 2005.

[25] Y. Wang, H. Jiang, M. S. Drew, Z.-N. Li, and G. Mori, "Unsupervised discovery of action classes," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1654-1661, 2006.

[26] L.-J. Li and L. Fei-Fei, "What, where and who? Classifying events by scene and object recognition," *IEEE Int. Conf. on Computer Vision*, pp. 1-8, 2007.

[27] P. Natarajan and R. Nevatia, "View and scale invariant action recognition using multiview shape-flow models," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1-8, 2008.

[28] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," *Int. Conf. on Pattern Recognition*, pp. 32-36, 2004.

[29] Y. Wang, P. Sabzmeydani, and G. Mori, "Semi-latent dirichlet allocation: A hierarchical model for human action recognition," *Human Motion – Understanding, Modeling, Capture and Animation*, A. Elgammal, B. Rosenhahn, and R. Klette, Eds. Springer, pp. 240-254, 2007.

[30] R. Mattivi and L. Shao, "Human action recognition using LBP-TOP as sparse spatio-temporal feature descriptor," *Int. Conf. on Computer Analysis of Images and Patterns*, pp. 740-747, 2009.

[31] V. Kellokumpu, G. Zhao, and M. Pietikäinen, "Recognition of human actions using texture descriptors," *Machine Vision and Applications*, vol. 22, no. 5, pp. 767-780, 2009.

[32] S. M. M. Ahsan, J. K. Tan, H. Kim, and S. Ishikawa, "Histogram of spatiotemporal Local binary patterns for human action recognition," *IEEE Joint Int. Conf. on Soft Computing and Intelligent Systems and Int. Symposium on Advanced Intelligent Systems*, pp. 1007-1011, 2014.

[33] R. Poppe, "A survey on vision-based human action recognition," *Image and Vision Computing*, vol. 28, no. 6, pp. 976-990, 2010.

[34] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, pp. 257-267, 2001.

[35] J. Davis and G. Bradski, "Real-time motion template gradients using Intel CVLib," *IEEE ICCV Workshop on Framerate Vision*, pp. 1-20, 1999.

[36] D. Weinland, R. Ronfard, and E. Boyer, "Free viewpoint action recognition using motion history volumes," *Computer Vision and Image Understanding*, vol. 104, no. 2, pp. 249-257, 2006.

[37] R. W. Poppe, "Discriminative vision-based recovery and recognition of human motion," PhD Dissertation, University of Twente, 2009. [Online]. http://doc.utwente.nl/60831/1/thesis_R_Poppe.pdf

[38] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247-2253, 2007.

[39] J. W. Davis, "Hierarchical motion history images for recognizing human motion," *IEEE Workshop on Detection and Recognition of Events in Video*, pp. 39-46, 2001.

[40] N. Ikizler and P. Duygulu, "Histogram of oriented rectangles: A new pose descriptor for human action recognition," *Image and Vision Computing*, vol. 27, no. 10, pp. 1515-1526, 2009.

[41] W. T. Freeman and M. Roth, "Orientation histograms for hand gesture recognition," *Int. Workshop on Automatic Face and Gesture Recognition*, pp. 296-301, 1995.

[42] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 886-893, 2005.

[43] N. Ikizler and P. Duygulu, "Human action recognition using distribution of oriented rectangular patches," *Human Motion - Understanding, Modeling, Capture and Animation*, pp. 271-284, 2007.

[44] V. Kellokumpu, G. Zhao, and M. Pietikäinen, "Human activity recognition using a dynamic texture based method," *British Machine Vision Conference*, pp. 885-894, 2008.

[45] G. Zhao and M. Pietikäinen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 915-928, 2007.

[46] W. C. Yau, D. K. Kumar, S. P. Arjunan, and S. Kumar, "Visual speech recognition using image moments and multiresolution wavelet images," *Int. Conf. on Computer Graphics, Imaging and Visualisation*, pp. 194-199, 2006.

[47] S. Kumar, D. K. Kumar, A. Sharma, and N. McLachlan, "Classification of hand movements using motion templates and geometrical based moments," *Int. Conf. on Intelligent Sensing and Information Processing*, pp. 299-304, 2004.

[48] S. M. M. Ahsan, J. K. Tan, H. Kim, and S. Ishikawa, "Recognizing Human Actions using Histogram of Local Binary Patterns," *IEEE/SICE Int. Symposium on System Integration*, pp. 54-59, 2013.

[49] C.-P. Huang, C.-H. Hsieh, K.-T. Lai, and W.-Y. Huang, "Human action recognition using histogram of oriented gradient of motion history image," *Int. Conf. on Instrumentation, Measurement, Computer, Communication and Control*, pp. 353-356, 2011.

[50] M. A. R. Ahad, T. Ogata, J. K. Tan, H. S. Kim, and S. Ishikawa, "View-based human motion recognition in the presence of outliers," *Int. Journal of Biomedical Soft Computing and Human Sciences*, vol. 13, no. 1, pp. 71-78, 2008.

[51] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62-66, 1979.

[52] S. M. M. Ahsan, J. K. Tan, H. Kim, and S. Ishikawa, "Spatiotemporal LBP and shape feature for human activity representation and recognition," *Int. Journal of Innovative Computing, Information and Control*, vol. 12, no. 1, pp. 1-13, 2016.

[53] M. A. R. Ahad, J. K. Tan, H. S. Kim, and S. Ishikawa, "Motion history image: its variants and applications," *Machine Vision and Applications*, vol. 23, no. 2, pp. 255-281, 2012.

[54] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Gray scale and rotation invariant texture classification with local binary patterns," *European Conf. on Computer Vision*, pp. 404-420, 2000.

[55] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971-987, 2002.

[56] M. Pietikäinen, A. Hadid, G. Zhao, and T. Ahonen, *Computer Vision Using Local Binary Patterns*. Springer, 2011.

[57] D. Huang, C. Shan, M. Ardabilian, W. Yunhong, and C. Liming, "Local binary patterns and its application to facial image analysis: a survey," *IEEE Trans. on Systems, Man, and Cybernetics, Part c: Applications and Reviews*, vol. 41, no. 6, pp. 765-781, 2011.

[58] LBP Bibliography. [Online]. http://www.cse.oulu.fi/CMV/LBP_Bibliography

[59] S. M. M. Ahsan, J. K. Tan, H. Kim, and S. Ishikawa, "Histogram of DMHI and LBP images to represent human actions," *IEEE Int. Conf. on Image Processing*, pp. 1440-1444, 2014.

[60] M. Baranwal, M. T. Khan, and C. W. D. Silva, "Abnormal motion detection in real time using video surveillance and body sensors," *Int. Journal of Information Acquisition*, vol. 8, no. 2, pp. 103-116, 2011.

[61] S. M. M. Ahsan, J. K. Tan, H. Kim, and S. Ishikawa, "Human action representation and recognition: An approach to a histogram of spatiotemporal templates," *Int. Journal of Innovative Computing, Information and Control*, vol. 11, no. 6, pp. 1855-1867, 2015.

[62] J. C. Niebles and L. Fei-Fei, "A hierarchical model of shape and appearance for human action classification," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1-8, 2007.

[63] K. Schindler and L. V. Gool, "Action snippets: how many frames does human action recognition require?," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1-8, 2008.

[64] C. W. Sul, K. C. Lee, and K. Wohn, "Virtual Stage: a location-based karaoke system," *IEEE Journal of MultiMedia*, vol. 5, no. 2, pp. 42-52, 1998.

[65] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio, "Pedestrian detection using wavelet templates," *IEEE Conf. on Computer Vision and Pattern*

Recognition, pp. 193-199, 1997.

[66] I. Haritaoglu, D. Harwood, and L. S. Davis, "Ghost: a human body part labeling system using silhouettes," *Int. Conf. on Pattern Recognition*, pp.77-82, 1998.

[67] L. W. Campbell and A. F. Bobick, "Recognition of human body motion using phase space constraints," *Int. Conf. on Computer Vision*, pp. 624-630, 1995.

[68] S. M. A. Eftakhar, "A Study on Human Motion Acquisition and Recognition Employing Structured Motion Database," PhD Dissertation, Kyushu Institute of Technology, 2012. [Online]. http://hdl.handle.net/10228/5291

[69] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175-185, 1992.

[70] Wikipedia contributors, "K-nearest neighbors algorithm," *Wikipedia, The Free Encyclopedia*, [Online]. https://en.wikipedia.org/w/index.php?title=K-nearest_neighbors_algorithm&oldid=708512163, accessed March 10, 2016.

[71] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *Int. Journal of Mathematical Models and Methods in Applied Sciences*, vol. 1, no. 4, pp. 300-307, 2007.

[72] C. Wallraven, B. Caputo, and A. Graf, "Recognition with local features: The kernel recipe," *IEEE Int. Conf. on Computer Vision*, pp. 257-264, 2003.

[73] L. Wof and A. Shashua, "Kernel principal angles for classification machines with applications to image sequence interpretation," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 635-640, 2003.

[74] N. Cristianini and J. S. Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. New York, USA: Cambridge University Press, 2000.

[75] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[76] D. Cao, O. T. Masoud, D. Boley, and N. Papanikolopoulos, "Human motion recognition using support vector machines," *Computer Vision and Image Understanding*, vol. 113, no. 10, pp. 1064-1075, 2009.

[77] Y. Ke, R. Sukthankar, and M. Hebert, "Spatio-temporal shape and flow correlation for action recognition," *IEEE Conf. on Computer Vision and Pattern Recognition,*

pp. 1-8, 2007.

[78] T. K. Kim, K. Y. K. Wong, and R. Cipolla, "Tensor canonical correlation analysis for action classification," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1-8, 2007.

[79] S. F. Wong, T. K. Kim, and R. Cipolla, "Learning motion categories using both semantic and structural information," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1-6, 2007.

[80] G. Seni and J. F. Elder, "Ensemble methods in data mining: improving accuracy through combining predictions," *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 1-126, 2010.

[81] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427-437, 2009.

[82] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," *Scandinavian Conference*, pp. 363-370, 2003.

[83] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," *ACM Int. Conf. on Multimedia*, pp. 357-360, 2007.

[84] O. Boiman and M. Irani, "Similarity by composition," *Neural Information Processing Systems*, pp. 177-184, 2006.

[85] T. d. Campos, et al., "An evaluation of bags-of-words and spatio-temporal shapes for action recognition," *IEEE Workshop on Applications of Computer Vision*, pp. 344-351, 2011.

[86] W.-C. Hu, "Real-time online video object segmentation based on motion detection without background construction," *Int. Journal of Innovative Computing, Information and Control*, vol. 7, no. 4, pp. 1845-1860, 2001.

[87] K. Masumitsu and T. Fuchida, "A proposition of human action recognition method considering co-occurrence of corner trajectories," *Int. Symposium on Artificial Life and Robotics*, 2014.

[88] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action

categories using spatial-temporal words," *Int. Journal of Computer Vision*, vol. 79, no. 3, pp. 299-318, 2008.

[89] K. Maninis, P. Koutras, and P. Maragos, "Advances on action recognition in videos using an interest point detector based on multiband spatio-temporal energies," *IEEE Int. Conf. on Image Processing*, pp. 1490-1494, 2014.