A Hybrid Multi-Layer Wavelet-based Video Encoding Scheme For Computer Vision applications on Mobile Resource Constrained Devices

by

Naveen Kumar Aitha

(Under the direction of Suchendra. M. Bhandarkar)

Abstract

The use of multimedia-enabled mobile devices such as pocket PC's, smart cell phones and PDA's is increasing by the day and at a rapid pace. Networked environments comprising of these multimedia-enabled mobile devices are typically resource constrained in terms of their battery capacity and available bandwidth. Real-time computer vision applications typically entail the analysis, storage, transmission, and rendering of video data, and are hence resource-intensive. Consequently, it is very important to develop a content-aware video encoding scheme that adapts dynamically to and makes efficient use of the available resources. A Hybrid Multi-Layered Video (HMLV) encoding scheme is proposed which comprises of content-aware, multi-layer wavelet-based encoding of the image texture and motion, and a generative sketch-based representation of the object outlines. Each video layer in the proposed scheme is characterized by a distinct resource consumption profile. Experimental results on real video data show that the proposed scheme is effective for computer vision and multimedia applications such as face recognition and activity recognition in resource-constrained mobile network environments.

INDEX WORDS: layered media, Video streaming, Activity Recognition, Face Recognition

A Hybrid Multi-Layer Wavelet-Based Video Encoding Scheme For Computer Vision applications on Mobile Resource Constrained Devices

by

Naveen Kumar Aitha

B.Tech., IIIT- Hyderabad

Hyderabad, Andhra Pradesh, India.

2007.

A Thesis Submitted to the Graduate Faculty

of The University of Georgia in Partial Fulfillment

of the

Requirements for the Degree

Master of Science

Athens, Georgia

2010

A Hybrid Multi-Layer Wavelet-based Video Encoding Scheme For Computer Vision applications on Mobile Resource Constrained Devices

by

Naveen Kumar Aitha

Approved:

Major Professors: Suchendra M. Bhandarkar

Committee: Tianming Liu
Kang Li

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
December 2010

Dedication

This thesis is dedicated to my parents for their constant support and motivation. I would also like to dedicate this to my brother, sister and friends who were always with me in all of my times.

Acknowledgments

The research experience and the insights into life which I learnt during my two and half years stay at University of Georgia is unforgettable. My research at Virtual and Parallel Computing Lab(VPCL) has been an amazing roller-coaster experience. The support from the faculty at Computer Science department at University of Georgia is admirable. I would like to take this opportunity to thank my major professor-Dr. Bhandarkar for his patience and sharing his research ideas and life experiences, Dr. Kang Li for his invaluable suggestions for my research and his cool way of teaching, and Dr. Tianming Liu for participation in my committee. I would also like to thank my fellow VPCL lab members for their suggestions during weekly meetings. I would like to thank Anirban for his encouragement and having interesting discussions on improvement of my research and philosophical thoughts on life. Deepthi, thank you for listening to my problems and motivating me all the time. Also I would like to thank Soumya Shivakumar, Hari Devulapally, and Swetha Pandhiti for making my stay at UGA memorable.

And last but not the least, I would like to thank my family members for their support and encouragement during all phases of life.

Contents

Acknowledgments **v**

List of Figures **ix**

1 Hybrid Multi Layered Video Encoding **1**

1.1 Introduction 1

1.2 HMLV Encoding 4

1.2.1 Creating Video Component V_{SKETCH} 5

1.2.2 Encoding V_{MT} 8

1.2.3 Generation of Motion Layers 9

1.2.4 Motion-based Multi-Resolution (MMR) Encoding Scheme 9

1.2.5 Generating the Highest (Top-most) Layer V_{orig} 10

1.2.6 Generating the Base Layer V_{base} 10

1.2.7 Generating Intermediate Layers V_{mid} 10

1.2.8 Assessment of Visual Quality of V_{MT} 12

1.3 Combining V_{SKETCH} and V_{MT} 14

1.4 Resource Usage Profile 16

2 Evaluation of HMLV encoding **19**

2.1 HMLV for Mobile Internet-Based Multimedia Applications 19

2.2 Human Activity Recognition 20
2.2.1 Activity Manifold Learning 20
2.2.2 Locality Preserving Projections (LPP) 21
2.2.3 Representations of Visual Inputs 23
2.2.4 Subspace Learning 24
2.2.5 Activity Classification 25
2.2.6 Experimental Results 28
2.3 Face Recognition using Eigenfaces 34
2.4 Conclusions 36

Bibliography **43**

List of Figures

1.1 Hybrid Multi-Layered Video Encoding Scheme 6

1.2 The creation of *pixel-threads* for a video frame (a) The original video frame; (b) Edges detected in the video frame, and filtered to remove small, spurious edges; (c) Break-points detected in the edge contours generated in the previous step 7

1.3 Example of the Generation of a Hybrid frame from the Base Layer and Polyline Sketch (a) Original Frame, (b) Sketch Frame, (c) Motion-and-Texture Frame, (d) HMLV frame 11

1.4 The change in PSNR and File Size with respect to the GWT parameter . . . 12

1.5 Reconstruction of sample frame using progressive truncation of coefficients using GWT representation (top row) and DCT representation (bottom row) 13

1.6 State diagram depicting the state transition rules based on available residual battery time. The current state transitions to a higher state if the available residual battery time ($T_{battery}$) is greater than the remaining running time of the video (T_{video}). Similarly, the current state transitions to a lower state if $T_{battery} \leq T_{video}$. 15

1.7 Different Motion-and-Texture Levels (a) Base Layer V_{base0} with Background Motion Layer Removed (b) Base Layer with $\beta = 0.5$ and V_{SKETCH} overlay (c) Intermediate Layer (d) Original Layer with $\beta = 1.0$ 17

1.8 Power Consumption Profiles for Different Texture Levels 18

2.1 Examples of sample images of actions. From top to bottom: bend, jack, jump, pjump, run, walk, skip, side, wave1, wave2 26

2.2 Activity classification on different video layers using median Hausdorff distance-based similarity measure 30

2.3 Recognition rates with reduced dimensions 31

2.4 Change of classification rates with different values of K (nearest neighbors considered for learning lower dimensional embedding) 32

2.5 Sample faces images from ORL database 35

2.6 Comparison of Recognition rate vs number of Eigenfaces for different HMLV encoded video layers 36

Chapter 1

Hybrid Multi Layered Video Encoding[1]

1.1 Introduction

The modern era of mobile computing is characterized by the increasing deployment of broadband networks coupled with the simultaneous proliferation of low-cost video capturing and multimedia-enabled mobile devices, such as pocket PC's, smart cell phones and PDA's. Mobile computing has also triggered a new wave of mobile Internet-scale multimedia applications such as video surveillance, video conferencing, video chatting and community-based video sharing, many of which have found their way in practical commercial products. Mobile network environments, however, are typically resource constrained in terms of the available bandwidth and battery capacity on the mobile devices. These environments are also characterized by constantly fluctuating bandwidth and decreasing device battery life as a function of time. Consequently, it is desirable to have a multi-layered (or hierarchical) content-based

[1]N. K. Aitha, S. M. Bhandarkar: A Hybrid Multi-layered Video Encoding Scheme for Mobile Resource Constrained Devices, *International Workshop on Mobile Multimedia processing in conjunction with ICPR 2010*

video encoding scheme where distinct video layers have different resource consumption characteristics and provide information at varying levels of detail [2].

Traditional multi-layered video encoding scheme such as the MPEG-4 Fine Grained Scalability profile (MPEG-FGS), are based on the progressive truncation of DCT or wavelet coefficients [3]. There is an inherent trade-off between the bandwidth and power consumption requirements of each layer and the visual quality of the resulting video, i.e., the lower the resource requirements of a video layer, the lower the visual quality of the rendered video [3]. Note that the conventional MPEG-FGS multi-layered encoding is based primarily on the spectral characteristics of low-level pixel data. Consequently, in the face of resource constraints, the quality of the lower layer videos may not be adequate to enable a high-level computer vision or multimedia application. For a multi-layered video encoding technique to enable a high-level computer vision or multimedia application, it is imperative that the video streams corresponding to the lower encoding layers encode enough high-level information to enable the application at hand while simultaneously satisfying the resource constraints imposed by the mobile network environment.

In this paper, a *Hybrid Multi-Layered Video* (HMLV) encoding scheme is proposed which comprises of content-aware, multi-layer encoding of texture and motion and a generative sketch-based representation of the object outlines. Different combinations of the motion-, texture- and sketch-based representations are shown to result in distinct video states, each with a characteristic bandwidth and power consumption profile. The proposed encoding scheme is termed as *hybrid* because its constituent layers exploit texture-, motion- and contour-based information at both the object level and pixel level. The high-level content awareness embedded within the proposed HMLV encoding scheme is shown to enable high-level vision applications more naturally than the traditional multi-layered video encoding schemes based on low-level pixel data.

A common key feature of computer vision and multimedia applications on mobile devices

such as smart phones, PDAs and pocket PC's is video playback. Video playback typically results in fast depletion of the available battery power on the mobile device. Various hardware and software optimizations have been proposed to reduce the power consumption during video playback and rendering [2]. Most of the existing work in this area has concentrated on reducing the quality of the video, to compensate for battery power consumption.

More recently, Chattopadhyay and Bhandarkar [1] have proposed a content-based multi-layered video representation scheme for generating different video layers with different power consumption characteristics. The video representation is divided into two components (i) a *Sketch* component, and (ii) a *Texture* component. The Sketch component has two different representations i.e., *Polyline* and *Spline*. The Texture component comprises of three distinct levels (in decreasing order of perceptual quality) denoted by V_{org}, V_{mid} and V_{base}. A combination of any of the three Texture levels and the two Sketch levels are used to generate six distinct levels of video with different resource consumption characteristics.

In this paper, we extend the work in [1] by effectively increasing the number of perceptual layers in the underlying video representation. This allows for a much finer degree of control on the underlying resource consumption while ensuring optimal perceptual quality of the rendered video for the computer vision or multimedia application on hand. The overall goal is to optimize the performance of the relevant computer vision or multimedia application within the specified resource constraints. In the proposed HMLV scheme, we retain the sketch component of the multi-layered video representation described in [1]. We enhance the texture component described in [1] by using a Gabor Wavelet Transform (GWT)-based representation for the underlying image texture and by including motion layers. The various texture layers are generated uniformly via progressive truncation of the GWT coefficients. The decomposition of the underlying video into motion layers also allows one to order the objects within the field of view based on their approximate depth from the camera.

The GWT is a special case of the Short-time Fourier Transform and is used to determine

sinusoidal frequency and phase content of local sections of a signal as it changes over time. In recent years, the multichannel GWT has been used for texture analysis and texture representation at multiple scales and orientations. The Gabor filter is a linear filter obtained by computing the GWT coefficients at a specific scale and orientation. A Gabor filter bank is a collection of Gabor filters at multiple scales and orientations. A set of filtered images is obtained by convolving the input image with the bank of Gabor filters. Each of these filtered images represents the input image texture at a certain scale and orientation. The convolution of an input image with a Gabor filter bank bears close resemblance to the processing of images within the primary visual cortex [5]. Sahoolizadeh et al.[39] have addressed the problem of face recognition using Gabor Wavelets and Neural Networks. Hong and Bartels [7] have addressed the issue of segmentation of remotely sensed LIDAR (light detection and ranging) data using Gabor wavelets and statistical feature. We show that the GWT representation can be used for human activity recognition as well.

Shape and kinematics are two important things to be considered in human movement analysis [30]. It is difficult to accurately extract kinematics from real videos using current imperfect vision techniques. Alternatively, focusing on shape, human action can be regarded as a temporal process in which human silhouettes continuously change over time. GWT representation represents the input frame at different orientations and scales which helps is capturing the change in the dynamic human shape. The proposed HMLV scheme is discussed in detail in the following sections.

1.2 HMLV Encoding

The input video is decomposed into two components: (i) a Sketch component denoted by V_{SKETCH} and, (ii) the combined Motion-and-Texture component denoted by V_{MT}. The V_{SKETCH} component is a Generative Sketch-based Video (GSV) representation where the

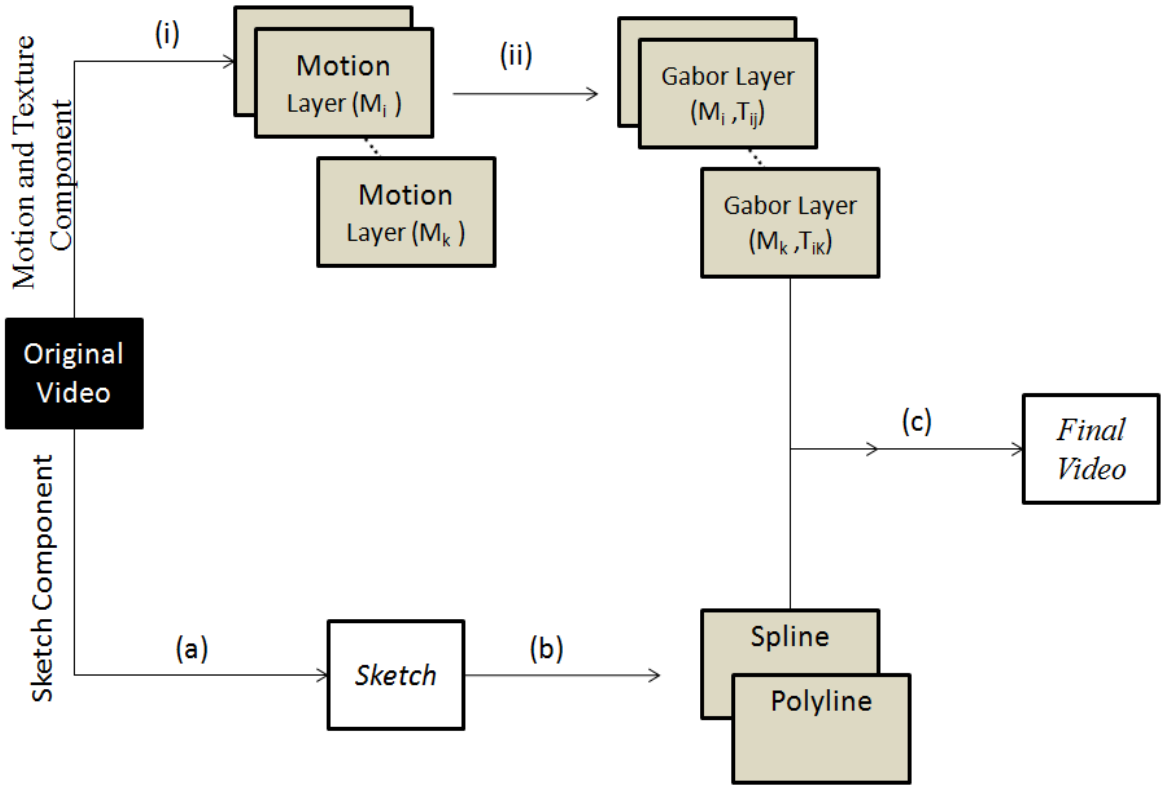
outlines of the objects are represented using sparse parametric curves [1]. The Texture component in [1] is replaced by a combined Motion-and-Texture component V_{MT} in the proposed HMLV scheme since the selection of motion layers is strongly coupled with the process of generating the texture layers via retention (or deletion) of the GWT coefficients for the chosen motion layers. The V_{MT} component in the proposed HMLV scheme is represented by four layers, i.e., a base layer video, two intermediate mid-layer videos and the original video. The combination of the Sketch component and different Motion-and-Texture layer videos (base, mid-level or the original video) yields distinct video states $\Gamma = (V_{MT}, V_{SKETCH})$ with unique resource utilization characteristics. Figure 1.2 outlines the proposed HMLV scheme.

1.2.1 Creating Video Component V_{SKETCH}

The sketch video component is generated as discussed in [1]. The sketch based component V_{SKETCH} essentially represents the outlines of the objects in the video. The video component V_{SKETCH} represents a video stream as a sequence of sketches, where each sketch in turn is represented by a sparse set of parametric curves. The V_{SKETCH} component is essentially a $Generative\ Sketch-based\ Video$ (GSV) representation which is generated by first dividing the video into a series of $Groups\ of\ Pictures$ (GOPS), in a manner similar to standard MPEG video encoding [21]. Each GOP consists of N frames (typically, $N = 15$ for standard MPEG/H.264 encoding) where each frame is encoded as follows:

1. The object outlines are extracted in each of the N frames. These outlines are represented as a sparse set of curves.

2. The curves in each of the N frames are converted to a suitable parametric representation.



HLV Encoding Scheme: (i) Motion Layers(k) are generated for the Motion and Texture component, (ii) The chosen k motion layers are encoded at different texture level, (a) Sketch component extracted from the video, (b) Polyline and Spline representations are generated, (c) One of the sketch representations is chosen for each of k motion layers superimposed together to form the final video

Figure 1.1: Hybrid Multi-Layered Video Encoding Scheme

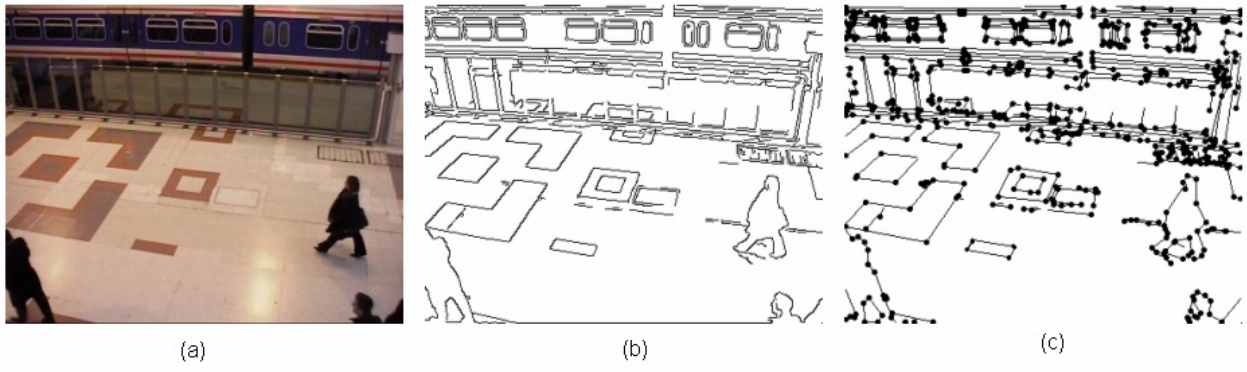


Figure 1.2: The creation of *pixel-threads* for a video frame (a) The original video frame; (b) Edges detected in the video frame, and filtered to remove small, spurious edges; (c) Break-points detected in the edge contours generated in the previous step

3. A temporal consistency criterion is used to remove spurious curves, which occur intermittently in consecutive frames, to remove an undesired flickering effect.

4. Finally, the parametric curves in the N frames of the GOP are encoded in a compact manner. The first frame of the GOP enumerates the curve parameters in a manner that is independent of their encoding, analogous to the I-frame in MPEG H.264 video encoding standard. The remaining $N-1$ frames in the GOP are encoded using motion information derived from previous frames, in a manner analogous to the P-frames in the MPEG H.264 video encoding standard [13].

The GSV encoding scheme is similar the MPEG video encoding standard. The GOP is a well established construct in the MPEG standard that enables operations such as fast forward, rewind and frame dropping to be performed on the encoded video stream. Motion vectors are used in the GSV encoding scheme to reduce temporal redundancy in a manner similar to MPEG video encoding, where motion vectors are used to describe the translation of frame blocks relative to their positions in previous frames. The error vector, in the case of the GSV encoding scheme, has the same form as the encoded representation of the moving object(s)

in the video. This is analogous to the MPEG video encoding standard, where the encoding error is represented in the form of macroblocks similar to the macroblock representation of the moving object(s) in the video.

The parametric curves used to represent the object outlines in each frame are termed as "$pixel-threads$". A pixel-thread is derived from a polyline P[0, N], which is a continuous and piecewise linear curve made of N connected segments. A polyline can be parameterized using a parameter $\alpha \in R$ (set of real numbers) such that P(α) refers to a specific position on the polyline, with P(0) referring to first vertex of the polyline and P(N) referring to its last vertex. The pixel-threads contain information only about the vertices (or break points) of the polyline. These break points can be joined by straight line segments (as in the case of polyline), or by more complex spline-based functions to create smooth curves. The dynamic nature of pixel-threads is modeled by the processes of birth and evolution of pixel-threads over time. A detailed description of these processes is presented in [1].

1.2.2 Encoding V_{MT}

Multi-scale representation of the image texture is achieved using GWT coefficients at different scales and orientations. The generation of different Motion-and-Texture layers is dependent on two factors, i.e., the number of Motion layers selected and the truncation parameter (β) for the GWT coefficients (i.e., Texture level) used to represent each Motion layer.

The Motion-and-Texture component of the HMLV representation comprises of four distinct layers denoted by: V_{base}, V_{I1}, V_{I2} and V_{orig}, where V_{base} is the lowest-level layer encoded using the fewest GWT coefficients for all the Motion layers; V_{orig} is the highest layer which is represented using the maximum number of GWT coefficients for all Motion layers resulting in a video of the highest visual quality; and V_{I1} and V_{I2} are the mid-level layers where the Motion layers that are deemed important are encoded using more GWT coefficients and the rest using fewer GWT coefficients.

1.2.3 Generation of Motion Layers

The original video is first processed to extract the different Motion layers, which form the basis for the generation of the Motion-and-Texture component. For any two successive video frames, the motion parameters for the Tomasi and Shi feature points [25] are estimated using an optical flow function [26]. After estimating the motion vectors, an adaptive K-means clustering technique [20] is used to cluster the motion vectors. Assuming that image pixels belonging to a single object share similar motion, spatial information is exploited in the clustering of the motion vectors to generate distinct Motion layers. The background motion layer is assumed to have constant or zero motion, and the remaining layers to have non-zero motion. A novel Motion-based Multi-Resolution (MMR) encoding scheme is proposed to encode distinct Motion layers at varying levels of visual quality.

1.2.4 Motion-based Multi-Resolution (MMR) Encoding Scheme

Each frame is represented by selecting the relevant Motion layers and the GWT coefficient truncation parameter (β) that is used to encode each Motion layer. The values of β for each Motion layer is chosen from the set$\{0.0, 0.5, 0.7, 0.8, 0.9, 1.0\}$. β signifies the % of energy in the coefficients to be retained from the encoding when the GWT coefficients are ordered in descending order of magnitude. A value of 1.0 for β signifies to encode the frame/layer with all the GWT coefficients where as a value of 0.5 represents to encode the layer/frame using coefficients which capture the 50% of the energy in the GWT coefficients. The Motion layer is then encoded with the corresponding number of GWT coefficients. Let F_i be the Motion-and-Texture frame of the video which is to be combined with the Sketch component to constitute the final video. Let $M_{i1}, M_{i2}....M_{ik}$ be the k Motion layers to be encoded in frame i. Let $T_{i1}, T_{i2}....T_{ik}$ be the k texture levels (based on the β) for the k motion layers found in frame i. Then (M_{ij}, T_{ij}) represents the motion layer j of frame i which is encoded

using the Texture level T_{ij}. The final Motion-and-Texture frame is generated via overlay of all the Motion layers. i.e.,

$$F_i = \sum_{j=0}^{j=k}(M_{ij}, T_{ij})$$

where $0 \leq T_{ij} \leq 1$ is the normalized value for the β.

1.2.5 Generating the Highest (Top-most) Layer V_{orig}

Using the above MMR encoding scheme, the original video can be generated using all the GWT coefficients, i.e., $T_{ij} = 1, \forall i, j$. This is tantamount to the encoding of each of the Motion layers with all the GWT coefficients resulting in full reconstruction of each frame in the video stream.

1.2.6 Generating the Base Layer V_{base}

The lowest-level Motion-and-Texture layer can be generated using very few GWT coefficients for all the motion layers, i.e., $T_{ij} = 0.5, \forall i, j$. Using only a few GWT coefficients results in a smoothed reconstruction of the Motion layer, which, when overlayed with the Sketch component, improves the visual appeal of the frame. Deleting entirely the background Motion layer in V_{base} generates the layer V_{base0} which is deemed to have a lower power consumption profile than the base layer V_{base}.

1.2.7 Generating Intermediate Layers V_{mid}

Most of the commonly video available encoding techniques are not content-aware, but in the proposed HMLV encoding scheme the intermediate layers are designed in such a manner that they represent perfectly the high-level contents of the video and encode the information accordingly. The two intermediate layers are generated as follows:

(i) Encode the motion layer which is farthest from the camera with zero GWT coefficients and

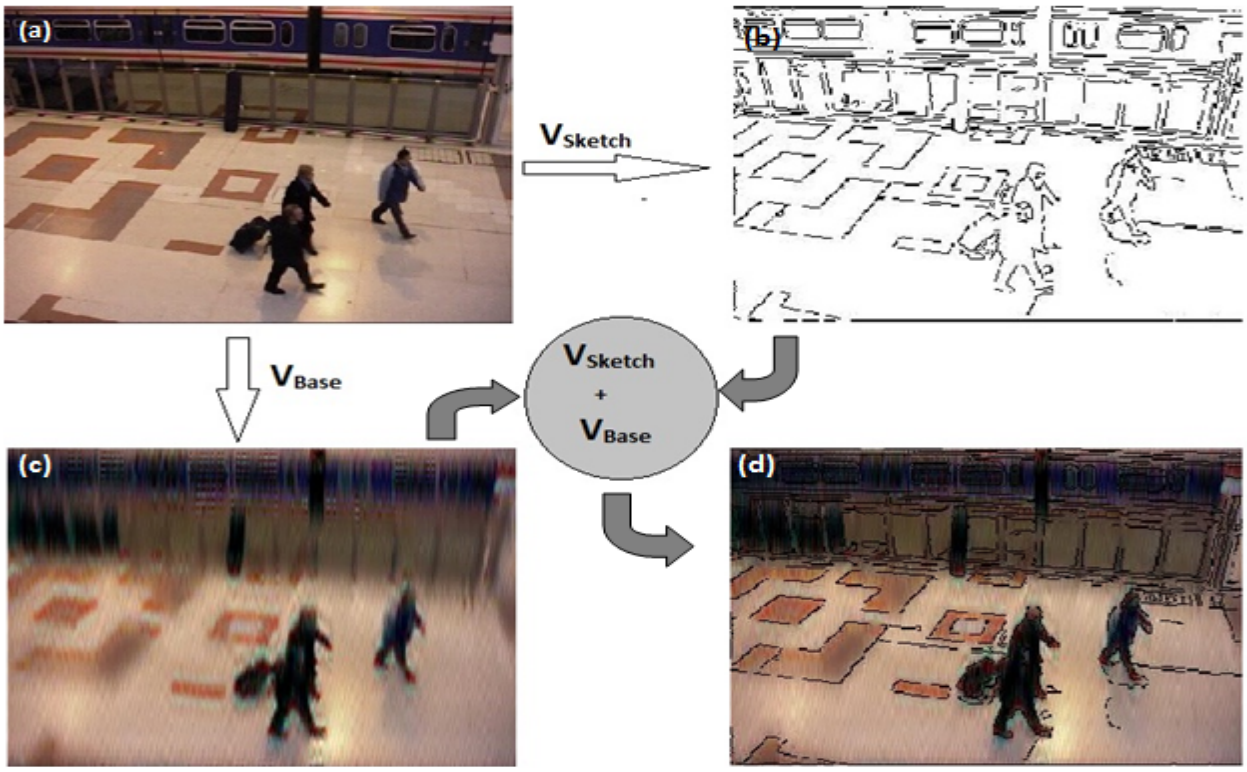


Figure 1.3: Example of the Generation of a Hybrid frame from the Base Layer and Polyline Sketch (a) Original Frame, (b) Sketch Frame, (c) Motion-and-Texture Frame, (d) HMLV frame

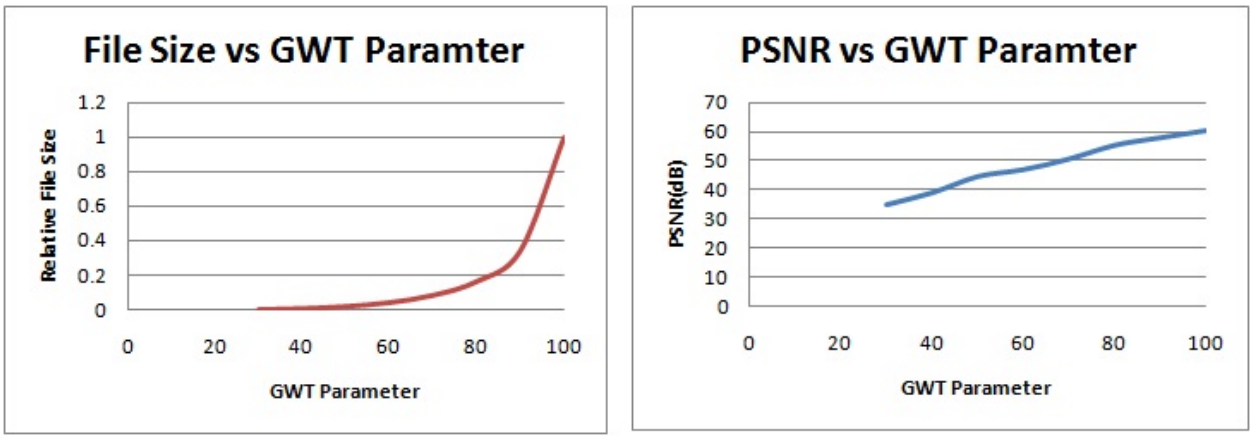


Figure 1.4: The change in PSNR and File Size with respect to the GWT parameter

all the other motion layers including background motion layer with the maximum number of GWT coefficients. (i.e., $T_i = 1$).

(ii) Encode the motion layer corresponding to the background at a low Texture level (i.e., with the fewest GWT coefficients) and all other layers at a high Texture level (i.e., $T_i = 1$).

1.2.8 Assessment of Visual Quality of V_{MT}

The visual quality of each of the aforementioned video layers comprising of V_{MT} can be assessed via subjective evaluation, as well as in terms of PSNR values. A quantitative evaluation of the average PSNR of a sample video with respect to the GWT parameter is depicted in Figure 1.4. Lower values of the GWT parameter results in corresponding lower values of the video file size (denoted by filesize). We have empirically observed that β values in the range of [0.5,0.6] can be used to generate the video layer V_{base} resulting in very small filesize value, albeit at the cost of low visual quality. However, few oriented and directional information is retained in the video layer V_{base}, to the point that the visual quality of the resulting video improves significantly when the object outlines from the GSV

Figure 1.5: Reconstruction of sample frame using progressive truncation of coefficients using GWT representation (top row) and DCT representation (bottom row)

representation are superimposed on the video layer V_{base}.Figure 1.5 shows the sample texture frame reconstructed using progressive truncation of GWT representation (top row) and DCT representation (bottom row). The GWT representation provides finer degree of control on generating more number of texture layers which have different power consumption profiles.

Figure 1.3 shows the hybrid frame with the Motion-and-Texture component V_{base} overlayed with the GSV sketch component. The specific HMLV video state is generated by overlaying the sketch component V_{SKETCH} over an appropriately chosen layer of V_{MT}. In the current implementation, the HMLV encoding is done off-line. Consequently, the run times of various procedures for generating GSV and each of the texture layers are not very critical. In the next section, it is shown how different combinations of V_{SKETCH} and V_{MT} result in distinct video states where each video state has a characteristic resource consumption profile.

1.3 Combining V_{SKETCH} and V_{MT}

In the proposed HMLV scheme, V_{MT} and V_{SKETCH} are obtained independently of each other. A suitable V_{MT} frame is generated and written to the frame buffer by the video controller. The Sketch component is extracted subsequently and superimposed on the Motion-and-Texture frame. The components are processed independently; only the order in which they are rendered is different. The V_{MT} frame is rendered first followed by the superimposition of the V_{SKETCH} frame.

Let us suppose that the texture component has L levels of resolution. From our earlier discussion we can see that we have 5 levels of texture representation, (i.e., $L = 5$) given by $V_{orig}, V_{I1}, V_{I2}, V_{base}, V_{base0}$ in the decreasing order of the visual quality and level 0 which represents the complete absence of the texture component.

Here V_{MT}^0 represents the texture video in the complete absence of the texture.

V_{MT}^{L-1} corresponds to the original texture layer with highest visual quality.

$V_{MT}^j (1 \leq j \leq L-2)$ are the texture video layers corresponding to the intermediate layers which have the visual quality between V_{MT}^0 and V_{MT}^{L-1}. The current state of the HMLV is represented as:

$$\Gamma(\textit{texture-level}, \textit{sketch-level}) = (V_{MT}^{\textit{texture-level}}, V_{SKETCH}^{\textit{sketch-level}}) \tag{1.1}$$

such that $0 \leq \textit{texture-level} \leq L-1$ and $\textit{sketch-level} \epsilon \{\textit{no-sketch}, \textit{polyline-sketch}, \textit{spline-sketch}\}$.

The above mentioned state representation allows for various resolution of texture with superimposition of sketch-based representations of varying degree of complexity. For example, $\Gamma(0, \textit{polyline-sketch})$ represents the video with only polyline sketch with no texture (supposed to be the lowest quality video). $\Gamma(L\text{-}1, \textit{no-sketch})$ represents the highest quality

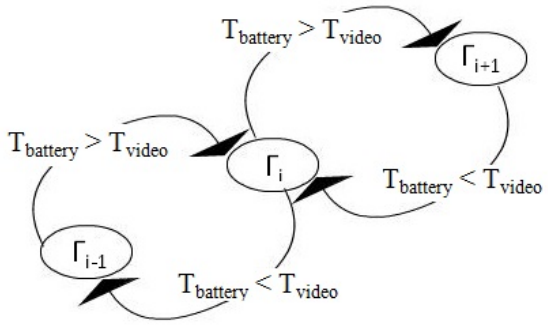


Figure 1.6: State diagram depicting the state transition rules based on available residual battery time. The current state transitions to a higher state if the available residual battery time ($T_{battery}$) is greater than the remaining running time of the video (T_{video}). Similarly, the current state transitions to a lower state if $T_{battery} \leq T_{video}$.

texture with no sketch superimposition. Having discussed the representation of the current state of the HMLV encoding scheme, we evaluate the battery consumption as function of the quality of the video.

The video states are arranged in a linear order starting with highest quality video and encoding with the lowest quality towards end. The first state of video is deemed to consume highest battery power and the lowest video layer is deemed to consume the lowest battery power. Thus, it is essential to order the different states of the videos in the order of battery consumption.

Let $Battery\text{-}time(\mathbf{X}, \mathbf{t})$ be the battery time estimate provided by the operating system on the playback device t seconds after the video playback has been initiated, where $\mathbf{X}$ denotes the state of the video during the playback. Let Γ=($\Gamma_1, \Gamma_2, ...\Gamma_s$) be the S distinct video states. We define a relation $\leq_p$ such that $\Gamma_i \leq_p \Gamma_j$ implies that

$$Battery\text{-}Time(\Gamma_i, t) \geq Battery\text{-}Time(\Gamma_j, t), t > 0 \tag{1.2}$$

In other words, the states are linearly ordered from left to right such that for any state (expect for the states $\Gamma(L, \textit{no-sketch})$ and $\Gamma(0, \textit{polyline-sketch})$) the sketch on its left consumes less power while decoding the entire video whereas the state on its right consumes more power. The $\textit{Battery-time}(\Gamma(\textit{current-state}, t)$ is estimated using a simple operating system call, which predicts the remaining battery time based on the current system load. The experimental results shown in the next section that the HMLV representation states indeed have different power consumption characteristics.

1.4 Resource Usage Profile

This paper discusses the efficient representation of the motion and texture within a video frame using motion layers and the β resulting in a simple HMLV encoding framework. In Figure 1.3, we can observe the generation of a hybrid frame from a Base Layer frame and Polyline Sketch frame. Figure 1.7 shows different Motion-and-Texture layers associated with a single frame that are generated using the proposed HMLV encoding scheme, i.e., the base layer, the intermediate layers and the original layer. We can see that the final visual appeal of the frame is increased by the overlay of the Sketch component over the base and intermediate Motion-and-Texture layers. The base layer and the intermediate Motion-and-Texture layers when overlaid with the Sketch component can be used effectively in video surveillance and object tracking applications where the finer details of the video are not very useful.

The power consumption profiles for different video states are shown in Figure 1.8. The overall power consumed during the video playback process is calculated in terms of the available battery time. It can be seen from the graph that the lower levels of texture take less battery power than the higher levels. The V_{Sketch} layer followed by V_{Base0} layer (i.e., the base level texture without the background motion layer) consume the minimum amount of battery power whereas the original video consumes the maximum amount of battery power

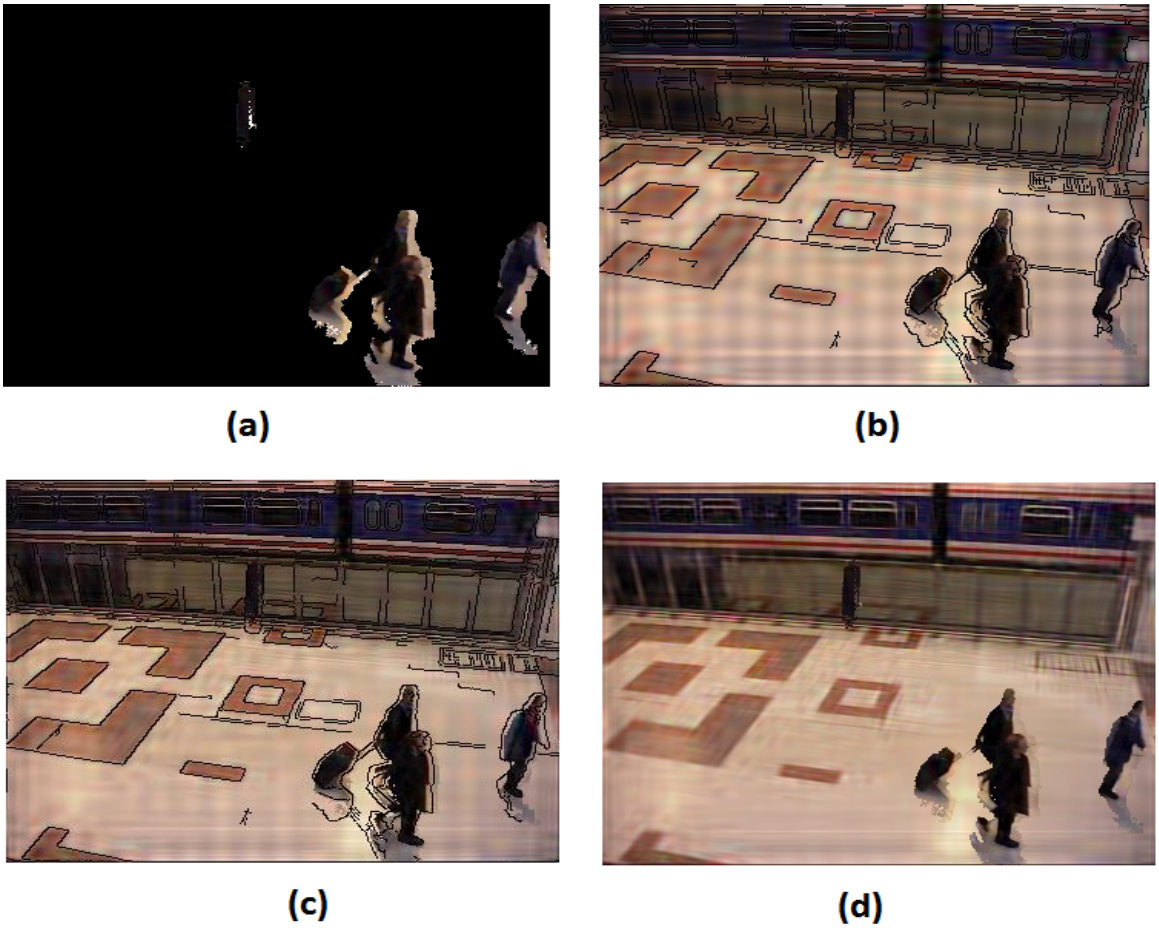


Figure 1.7: Different Motion-and-Texture Levels (a) Base Layer V_{base0} with Background Motion Layer Removed (b) Base Layer with $\beta = 0.5$ and V_{SKETCH} overlay (c) Intermediate Layer (d) Original Layer with $\beta = 1.0$

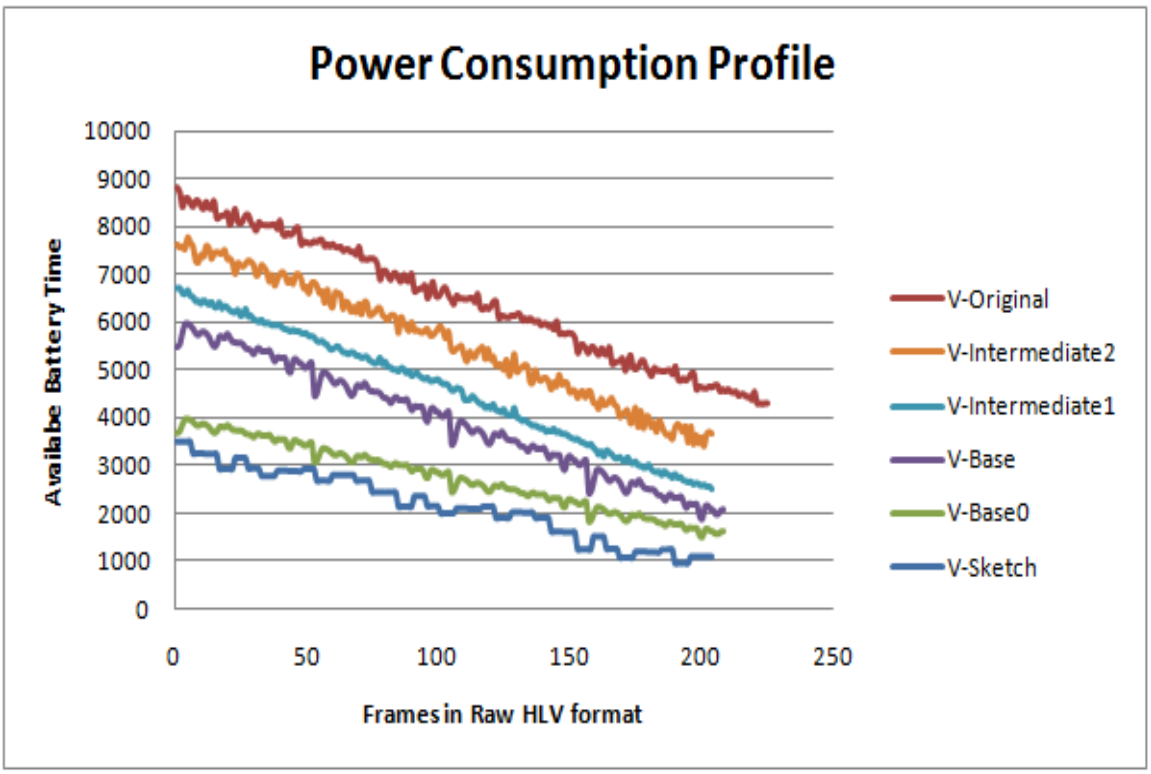


Figure 1.8: Power Consumption Profiles for Different Texture Levels

on the device. All experiments have been performed using a Dell Inspiron 1525 laptop PC with 2.0GHZ CPU, 2GB RAM, and a 250GB, 5400 rpm hard drive running in battery mode.

Chapter 2

Evaluation of HMLV encoding[1]

2.1 HMLV for Mobile Internet-Based Multimedia Applications

In the proposed HMLV scheme, the various video states (layers) are generated by altering the visual quality of the encoded video. The video states generated in the proposed scheme are an approximation to the original MPEG encoded video. This raises a logical question - Is this approximation good enough? The fact that all the objects discernible in MPEG-encoded video are also discernible in HMLV encoded video is revealed by the subjective evidence gathered from human subjects. However, in the interest of objectivity we have evaluated the proposed HMLV scheme in the context of some important computer vision and multimedia applications in a resource-constrained mobile Internet environment using quantitative performance metrics i.e, face recognition, human activity recognition. All experiments were performed using a Dell Inspiron 1525 laptop PC with 2.0GHZ CPU, 2GB RAM, and a 250GB, 5400 rpm hard drive running in battery mode. In all of the experiments, four

[1]N. K. Aitha, S. M. Bhandarkar: A Hybrid Multi-Layer Video Encoding Scheme For Computer Vision applications on Mobile Resource Constrained Devices, To be submitted to *IEEE Transactions Circuits and Systems for Video Technology 2011*

different HMLV layers (HMLV-1, HMLV-10, HMLV-9, HMLV-5) were used, i.e.,HLMV-1 = $(null, spline - sketch)$ represented as binary silhouettes, HMLV-10 = $(V_{org}, null)$, HMLV-9= $(V_{I1}, spline - sketch)$, HMLV-5 = $(V_{base}, spline - sketch)$.

2.2 Human Activity Recognition

For the task of activity recognition, we learn explicit representations for the dynamic shape manifolds of moving humans. Given a video sequence of moving humans, a lower dimensional embedding of the movements is captured by Locality Preserving Projection (LPP). The high dimensional data is projected to lower dimensional space to characterize the spatiotemporal property of the action, as well as to preserve the geometric structure[27]. The embedded action trajectories are matched using the median Hausdorff distance similarity measure. The nearest neighborhood classification is used for action classification. The Human activity recognition is performed on three levels of HMLV videos generated from the recent dataset with ten different actions performed by nine different subjects. The accuracy of the human activity recognition is experimentally verified. The following sections details the manifold learning, manifold projection and and experimental results for human activity recognition.

2.2.1 Activity Manifold Learning

To learn the complete structure of a activity in a high-dimensional manifold is a formidable task. Consequently, we use Locality Preserving Projection (LPP) for this goal based on the following considerations: a) LPP explicitly models the manifold structure by an adjacency graph, which yields an efficient subspace learning algorithm to discover the intrinsic structure of the action space; b) LPP shares some of the data representation properties of nonlinear techniques such as Locally Linear Embedding (LLE), e.g.,the locality preserving characteristic; c) the LPP is obtained by finding the optimal linear approximations to

eigenfunctions of the Laplace Beltrami operator [29]. This linearity naturally leads to low computation complexity and is, thus, more efficient for practical tasks; and d) although a few nonlinear methods (e.g., Isomap, LLE) do yield impressive results on some benchmark artificial datasets, the resulting mappings are defined only for the training data points. How to evaluate the mapping on a new test data yet remains unclear [33]. In contrast, LPP may be simply applied to any new data point.

2.2.2 Locality Preserving Projections (LPP)

The problem of linear dimensionality reduction problem can be formalized as follows: Given a set of high dimensional data points $x_1, x_2, x_3, ...x_n$ in R^d, find a transformation matrix A that maps these n points to a set of low dimensional points $y_1, y_2, y_3, ..., y_n$ in R^l ($l << d$) such that y_i represents x_i. i.e., $y_i = A^T x_i$

The following is algorithmic procedure for LPP as described in [29]

Constructing the adjacency Graph

Let G denote a graph with n nodes. An edge is inserted between nodes i and j if x_i and x_j are proximal. The proximity relation is defined based on ϵ-neighborhoods ($\epsilon \in R$) i.e., if $\|x_i - x_j\|^2 < \epsilon$ then x_i and x_j are neighbors, where the norm is the usual euclidean norm in R^d or K-nearest neighbors, ($K \in N$) i.e., x_i is among the K- nearest neighbors of x_j or x_j is among the K- nearest neighbors of x_i.

Choosing the edge weights of G

The edge weights evaluate the local structure of the data space. W is a sparse and symmetric n x n matrix where the element w_{ij} denotes the weight of the edge joining nodes i and j. Note that $w_{ij} = 0$ if there is no edge between nodes i and j. Also two variations for weighting

the edges can be considered- **a.** Based on heat Kernel: $w_{ij} = e^{-\|x_i - x_j\|^2/t}$ $(t \in R)$ **b.** Binary Weighting: $w_{ij} = 1$ iff nodes i and j are connected; $w_{ij} = 0$ otherwise.

Eigenmaps

The optimal locality-preserving projection (LPP) the locality can be determined by minimizing the following objective function based on the standard spectral graph theory [29].

$$min(\sum_{i,j}(y_i - y_j)^2 w_{ij}) \tag{2.1}$$

The above minimization problem is to ensures that if x_i and x_j are proximal, then y_i and y_j are proximal as well. Let e denote a transformation vector, the objective function can be modified[29]

$$\begin{aligned} 1/2\sum_{i,j}(y_i - y_j)^2 w_{ij} &= 1/2\sum_{i,j}(e^T x_i - e^T x_j)^2 w_{ij} \\ &= \sum_i e^T x_i d_{ii} x_i^T e - \sum_{i,j} e^T x_i w_{ij} x_i^T e \\ &= e^T X(D - W)X^T e \\ &= e^T X L X^T e \end{aligned}$$

where D is a diagonal matrix whose entries are column (or row) sums of the symmetric weight matrix W, i.e., $d_{ij} = \sum_i w_{ij}$, $L = D - W$ is the Laplacian matrix, and X is the data matrix $[x_1, x_2, ..., x_n]$. Matrix D provides a natural measure on the data points, i.e., the larger the value of d_{ii} (corresponding to y_i), the more "important" y_i is. Therefore, a constraint is imposed [29] as follows:

$$y^T Dy = 1 \Rightarrow e^T XDX^T e = 1. \tag{2.2}$$

Accordingly, the minimization problem reduces to determining the optimal value of e:

$$arg(\min_{e^T XDX^T e=1} e^T XLX^T e = 1) \tag{2.3}$$

The solution to the minimization problem in equation (2.3) is tantamount to the solution of the generalized eigenvalue and eigenvector problem

$$XLX^T e = \lambda XDX^T e \tag{2.4}$$

Let the column vectors $e_0, e_1, ...e_{l-1}$ be the solutions of equation (2.4), ordered according to their eigenvalues $\lambda_0 < \lambda_1 < \lambda_2 < ....\lambda_{l-1}$. Thus, the embedding of each data point is represented by

$$y_i = E^T x_i \tag{2.5}$$

Where E represents the embedding function $E = [e_0, e_1.....e_l - 1]$. The obtained projections are actually the optimal linear approximation to the eigenfunctions of the Laplace Beltrami operator on the manifold[29]. For more details on the justification of LPP we refer the interested reader to [29].

2.2.3 Representations of Visual Inputs

Thus foreground layer or the motion (activity) layer from the HMLV video state represents the effective feature space for the human activity recognition. The basic assumption here is that the motion (activity) layer in the any of the HMLV video states is known, i.e., the video sequence of Region of Interest (ROI) (i.e., human performing action) can be obtained

from the original video. Considering that establishing correspondences between landmarks on the foreground is not always feasible because of the temporal changes in topology and self-occlusions, the foreground sequences are directly used as visual inputs for subspace learning. The different feature space considered here are the Discrete Cosine Transform (DCT) coefficients, raw foreground images, and HMLV activity layer of video states HMLV-1 (represented as binary silhouettes), HMLV-5, HMLV-9, and HMLV-10. (i.e., HMLV-1 = $(null, spline - sketch)$ HMLV-10 = $(V_{org}, null)$, HMLV-9= $(V_{I1}, spline - sketch)$, HMLV-5 = $(V_{base}, spline - sketch)$)

2.2.4 Subspace Learning

The original image representations (foreground layer representation) are both noisy and expensive to analyze, so LPP is used to embed activities into a lower dimensional subspace for more compact representation. Consider c action classes (i.e., different actions) where each class represents a video sequence from a foreground activity layer denoting a specific human actions. Each frame with a resolution of $M \times N$, is converted into an $d-$ dimensional ($d = M \times N$) vector v in a raster-scan manner. For GWT representation, for each frame the d GWT coefficients(d=Number of GWT coefficients) of that Foreground layer are considered. Let $v_{i,j}$ be the jth input frame in the ith class and N_i the number of input frames in the ith class. The total number of training samples is given by $N_t = N_1 + N_2 + N_3....N_c$, and the whole training data can be represented by

$$\begin{aligned} X &= [v_{1,1}, v_{1,2}....., v_{1,N1}, v_{2,1}.....v_{c,Nc}] \\ &= [\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, ...\mathbf{x_{Nt}}] \end{aligned}$$

where each column of X is an $d-$dimensional data point. For each action class, multiple sequences may be freely added to the training data without altering the following training

process.

To construct an affinity matrix W, the neighborhood of each point is directly determined by its K- nearest neighbor points based on the distance measured in the input space. To measure the distance between two data points $\mathbf{x_i}$ and $\mathbf{x_j}$, we compute the cosine of the angle between the two vectors

$$S_{i,j} = cos(\mathbf{x_i}, \mathbf{x_j}) = \frac{\mathbf{x_i}.\mathbf{x_j}}{(|\mathbf{x_i}|.|\mathbf{x_j}|)}. \tag{2.6}$$

The ϵ neighborhood is not used to construct the adjacency graph because it is difficult to choose an appropriate value of ϵ for real world problems. To avoid selection of an extra parameter (i.e., for heat kernel-based distance measure) in the learning process, we used simple 0-1 weighting scheme to set the weight of the edge in the adjacency graph, i.e.,

$$w_{i,j} = \begin{cases} 1, & \textit{the nodes } i \textit{ and } j \textit{ are connected} \\ 0, & \textit{otherwise} \end{cases} \tag{2.7}$$

we solve for the "eigenmapping" problem in equation (2.4) to obtain the embedding function E. The embedding results of the original data is given by $Y = E^T X$. Each data point v is embedded into a point p in the lower-dimensional subspace. Accordingly, the sequential movement of a certain action is accordingly mapped into a trajectory in a lower-dimensional parameter space.

2.2.5 Activity Classification

Activity classification can be solved through measuring motion similarities between the reference motion patterns and test samples in the low-dimensional embedding space. Assume the two action sequences are respectively mapped to $l \times T$ matrices - $A_1(l \times T_1)$ and $A_2(l \times T_2)$, where l is the reduced dimensionality, and T_1 and T_2 are the durations of these complete activities respectively. Note that the same activities can have different temporal durations

Figure 2.1: Examples of sample images of actions. From top to bottom: bend, jack, jump, pjump, run, walk, skip, side, wave1, wave2

due to speed changes (but have same trajectory in 3D space), and different activities may have significantly different temporal durations. We use the median Hausdorff distance to measure the motion similarity between the two lower-dimensional curves. Before computing the similarity measure, each column vector p in the matrices A_1 and A_2 is normalized, i.e., $p = p/\|p\|$.

Median Hausdorff Distance

A distance measure that can handle the changes in duration and temporal shifts is ideal for measuring the motion similarity of two activities. The Hausdorff distance measure provides a means of determining the resemblance of one point set to another, by examining the fraction of points in one set that lie near the other set (and vice versa). A variant of the Hausdorff measure is called the median Hausdorff distance measure and is based on the computation of the median of the minimum distance between the data points in one set from those in the other set as follows:

$$S(A_1, A_2) = median_i(min_j(\|A_1(i) - A_2(j)\|)) \tag{2.8}$$

Since the Hausdorff distance measure is oriented, to ensure symmetry, final distance is modified to

$$MHD(A1, A2) = S(A_1, A_2) + S(A_2, A_1) \tag{2.9}$$

where $MHD(A1, A2)$ denotes the median Hausdorff distances between sets $A1$ and $A2$. The smaller the distance measure, the more similar are the two activities. Computing the median rather than the mean, makes the MHD measure robust to the presence of outliers in the input data.

Classifier

The activity classification is performed using a nearest-neighbor classifier. Let T represent a test action sequence and R_i represent the ith reference action sequence, test action as the class c that can minimize the similarity distance between the test sequence and all reference patterns, i.e.,

$$c = arg_i \min MHD(T, R_i) \tag{2.10}$$

where MHD is the median Hausdorff distance measure.

2.2.6 Experimental Results

In this paper we used the dataset reported in [35], is reasonably sized (in terms of the number of subjects, actions and videos), compare to other concurrent action databases available in public domain. It consists of 90 low-resolution videos (180 x 144, 25 fps) of nine different people, each performing 10 different natural actions. These actions include bending (*bend*), jumping jack ($jack$), running (run), Walking ($walk$), jumping-forward-on-two-legs ($jump$), jumping-in-place-on-two-legs ($pjump$), galloping sideways ($side$), waving-one-hand ($wave1$), waving-two-hands ($wave2$). These actions are either periodic (e.g. run and walk) or non-periodic actions (e.g., *bend*), and either stationary (e.g., $wave1$ and $wave2$) or non-stationary motions along both horizontal (e.g., $side$) and vertical (e.g., $jack$). Sample frames from different action videos can be seen in the Figure 2.1 In this dataset different people were asked to perform same action which provides more realistic data.

Representation of Visual Inputs

For the effective representation of the input sequences, the foreground masks from the activity layer are directly considered as the inputs for activity recognition. Discrete Cosine Transform (DCT) coefficients, Raw foreground images, and HMLV-1 (represented as binary sil-

houettes) HMLV-5, HMLV-9, and HMLV-10 layers (i.e., HMLV-10 = $(V_{org}, null)$, HMLV-9= $(V_{I1}, spline-sketch)$, HMLV-5 = $(V_{base}, spline-sketch)$, HMLV-1 = $(null, spline-sketch)$) of the videos are considered as the inputs for activity recognition.

Traditional multi-layered scheme such as MPEG-4 Fine Grained Scalability profile (MPEG-FGS), are based on progressive truncation of DCT or wavelet coefficents[3]. For the completion sake, we choose to evaluate the activity recognition by encoding the activity layer with the DCT coefficients. As we have discussed in the assesment of the visual quality section, GWT provides finer degree of control in generating more number of texture layers with different power consumption profiles. We used all of the DCT coefficients of the activity texture layer as one of the input vectors for evaluation.

Data Processing

There are 90 videos of 9 people performing 10 different actions. All the action videos except the one for the *bend* contains more than one action sequence. Two complete cycles of the action are considered starting from the mid portion of the video for each of the action sequences. There total of 171 sequences (9 x 2 x 9 + 9 x 1) i.e., each person has one sequence of *bend* and 2 sequences of a complete cycle of all other actions. Each of the action sequence frame is normalized to same dimension (64 $\times$ 48 pixels) properly centered, and converted into a 3072-dimensional vector. A considerable number of visual input vectors are used for the subspace learning phase.

Experimental Setup

The classifier determines which class a given measurement belongs to using the nearest-neighbor criterion. From the training samples considered, we compute an overall unbiased estimate of the true recognition accuracy using the leave-one-out cross validation method.In each iteration of the cross validation procedure we omit one action sequence performed by

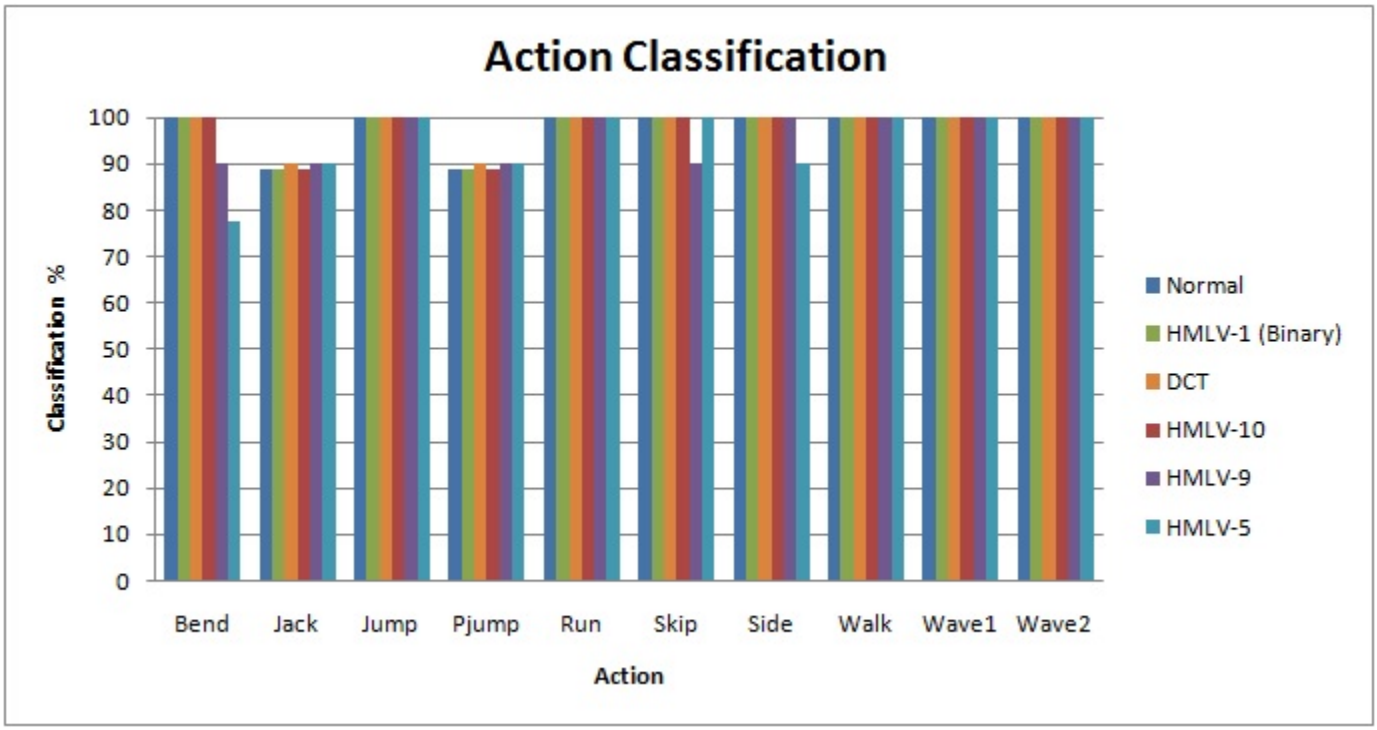


Figure 2.2: Activity classification on different video layers using median Hausdorff distance-based similarity measure

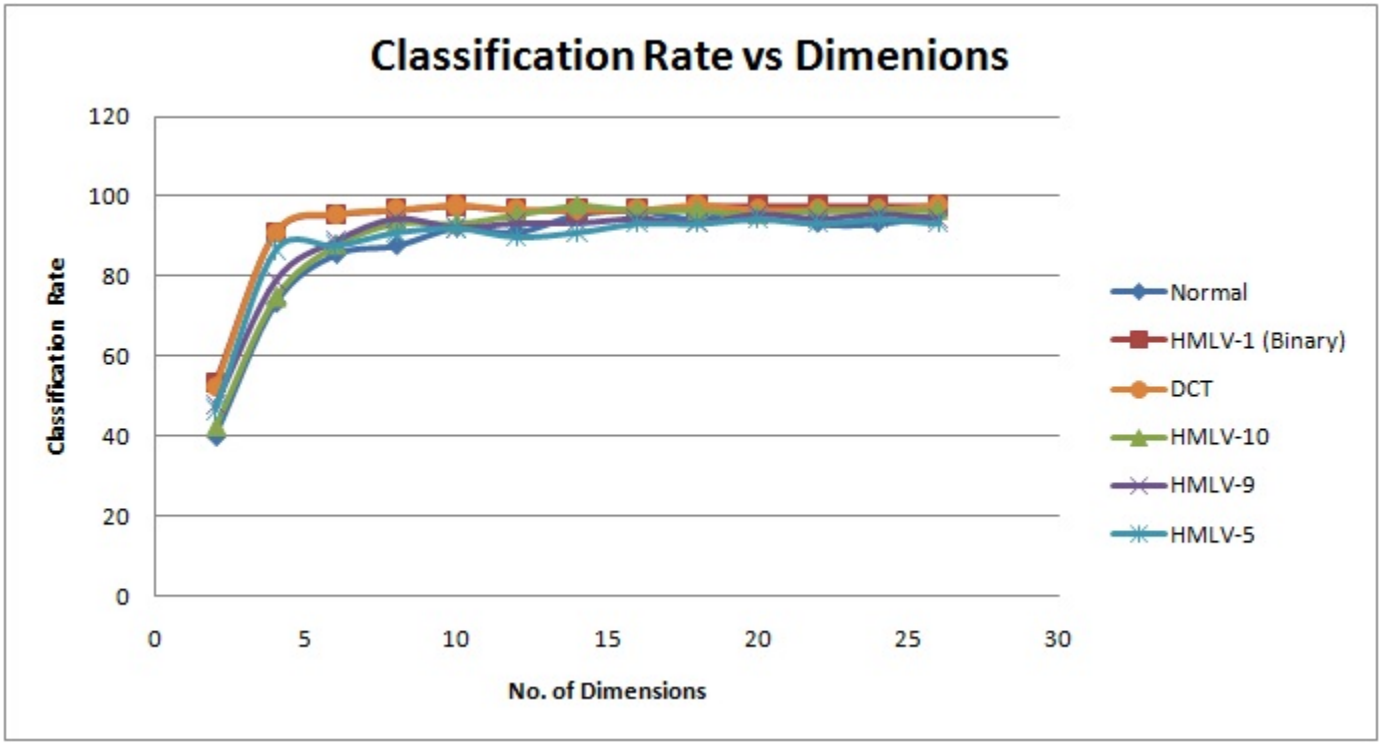


Figure 2.3: Recognition rates with reduced dimensions

a certain subject. Considering the fact that the repeated performances of the same action performed by the same subject vary only slightly, the repeat action sequence performed by the same subject is also removed, while all other action sequences are retained. The training is performed on all the retained action sequences, and the omitted sequence is classified. If the omitted sequence is classified correctly then we can conclude that multiple instances of the same action sequence performed by different subjects correlate significantly with each other.

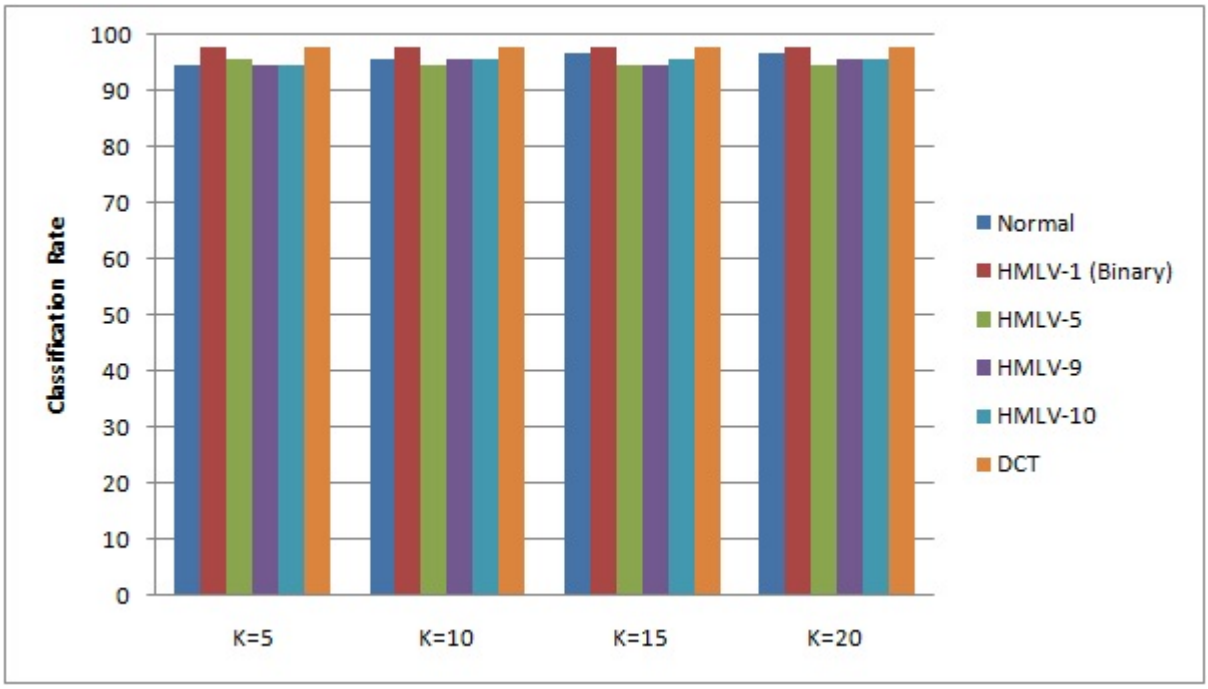


Figure 2.4: Change of classification rates with different values of K (nearest neighbors considered for learning lower dimensional embedding)

Results and Analysis

The video states HMLV-1, HMLV-5, HMLV-9, and HMLV-10 (i.e., HMLV-10 = $(V_{org}, null)$, HMLV-9= $(V_{I1}, spline-sketch)$, HMLV-5 = $(V_{base}, spline-sketch)$, HMLV-1 = $(null, spline-sketch)$) are considered for the experiments including the raw and HMLV-1 (binary). From Figure 2.2, we can conclude that all the HMLV video states under consideration i.e., HMLV-1, HMLV-5, HMLV-9, and HMLV-10 perform well in the context of activity recognition. However, HMLV-10 performs the best for activity recognition followed by HMLV-9 and HMLV-5. This is because that the highest layer encodes a greater number of GWT coefficients and thus can capture the change in the human shape during the activity being performed. The activity recognition performed using the HMLV-10 layer is equivalent to the activity recognition using the raw image and binary silhouettes since HMLV-10 layer encodes all of underlying dynamic shape information. Using HMLV-10 video state the recognition rates are higher than those obtained with the other HMLV video states i.e., HMLV-5 and HMLV-9. This behavior is expected as some of the Gabor coefficients are dropped in the lower lower layers. The β values for HMLV-5, HMLV-9 and HMLV-10 are 0.5, 0.8 and 1.0 respectively. The activity recognition depends on the extent of the human shape information retained in the video sequences. Although there is a huge difference between the quality of the videos corresponding to HMLV-5 and HMLV-10, the activity recognition rates are not significantly different. The raw and binary silhouettes video sequences are also seem to yield close to 100% classification. The DCT representation of the texture layer also yields the classification results close to that of the HMLV-10, but the GWT representation gives the finer degree of control in generating more number of texture layers.

From Figure 2.3, we can see that the reduced dimensionality of the manifold embedding does not make huge difference on the activity classification rate beyond 10 dimensions. Once again, the HMLV-10 video state yields better results than the other HMLV video states. The selection of the value K (nearest neighbors considered for learning lower dimensional

embedding) also does not impact the performance which is evident from Figure 2.4. Overall lowest correct classification rate for 20 dimensions and any value of K is 94.44% (for the HMLV-5 video state). This shows that the HMLV encoding is well suited for activity recognition on mobile resource-constrained devices. The activity recognition method is easy to implement and can be used for activity recognition using any of the HMLV video states on mobile resource-constrained devices.

2.3 Face Recognition using Eigenfaces

The HMLV scheme was also evaluated in the context of face recognition performed using the well known $Eigenfaces$ approach [36]. The eigenfaces are essentially eigenvectors that are derived from the covariance matrix of the probability distribution of the high-dimensional vector space of human faces (with known identity) that are stored in a database. The eigenfaces constitute a basis set of the "standardized faces", derived from the training set of human faces in the database via principal component analysis(PCA). The training set of human faces is used to estimate the mean vector and covariance matrix of the probability distribution of the high dimensional vector space of human faces stored in the database under varying conditions, i.e., at different times, under varying illumination, varying facial expressions and varying facial details. Each human face in the database can be expressed as a unique linear combination of these Eigenfaces. An unknown face is also expressed as a linear combination of these Eigenfaces. The Eigenface coefficients of the unknown face are compared with those of each of the faces in the database by computing Euclidean distance in the vector space spanned by the eigenfaces. The $Eigenfaces$ approach based is essentially an appearance-based face recognition method. In our implementation of the face recognition system, the ORL database [38] is used for training and testing purposes. The database contains gray scale images scaled to 32×32 pixels with frontal profiles of 40

Figure 2.5: Sample faces images from ORL database

subjects with 10 different images per subject. For some subjects, the images were taken at different times, and under varying conditions of ambient lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). Figure 2.5 shows the sample face images of 4 subjects used in the *Eigenfaces* algorithm. The graph in Figure 2.6 compares the face recognition rate for each of the HMLV layers under consideration, i.e., HMLV-5, HMLV-9, and HMLV-10. The recognition accuracy was tested for varying number of eigenfaces used in the recognition process. All the training images were generated from the same HMLV video layers for the respective testing purposes. As it can be seen from the graph, the recognition accuracy decreases slightly in the case of video layer HMLV-5 whereas there is no major difference in recognition accuracy between the layers HMLV-9 and HMLV-10. This is expected since HMLV-5 encodes fewer GWT coefficients and hence less directional and oriented texture information than HMLV-9 and HMLV-10.

The above experimental results of the above face recognition experiments based on the

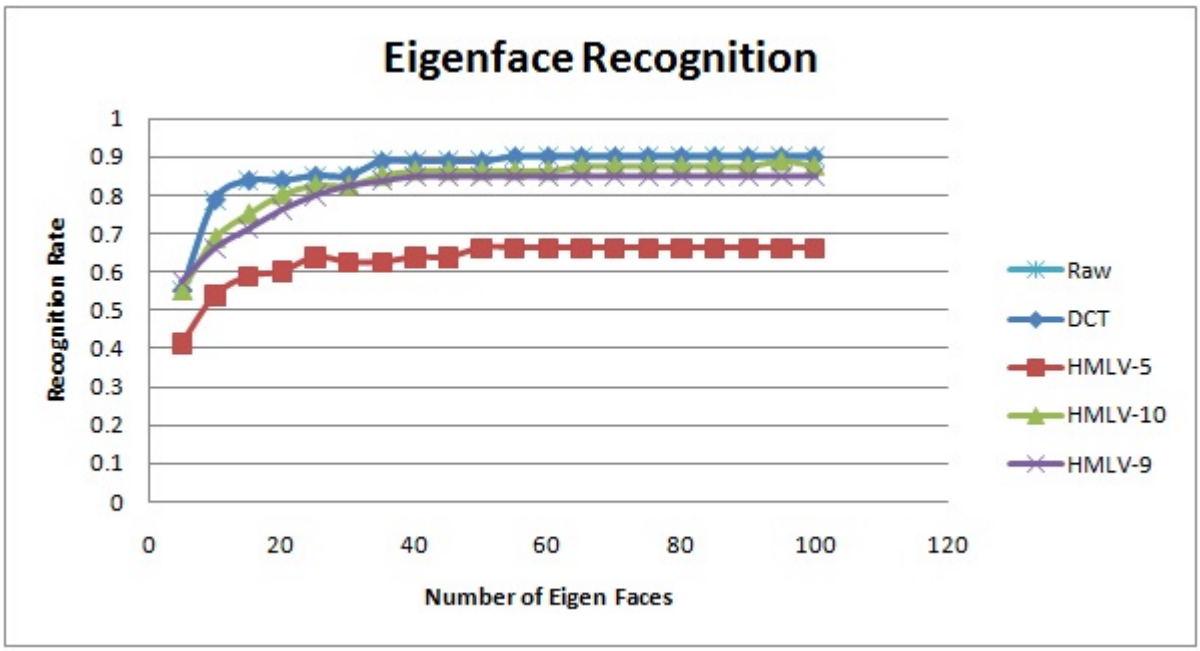


Figure 2.6: Comparison of Recognition rate vs number of Eigenfaces for different HMLV encoded video layers

Eigenfaces approach using HMLV-encoded videos shows that the proposed HMLV encoding scheme is well suited for face recognition and similar computer vision tasks in resource-constrained environment. These resource-constrained environments are characterized typically by limited available battery capacity and bandwidth on mobile client devices.

2.4 Conclusions

A new wave in the mobile Internet-based multimedia applications has been triggered by the increasing deployment of broadband networks and simultaneous proliferation of low-cost video capturing multimedia-enabled mobile devices. These mobile networked environments are typically resource constrained in terms of available bandwidth and battery capacity

on mobile devices. Multimedia applications that typically entails analysis, transmission, storage and rendering of video data are resource-intensive. Since the available bandwidth in the mobile Internet is constantly changing and the battery life of a mobile video capturing and rendering device decreases with time, it is desirable to have a video representation scheme that adapts dynamically to the available resources. To this end, this paper presents an integrated Hybrid Multi-Layered Video representation framework for contour-, motion- and texture-based encoding using object outline sketches, motion layers and GWT coefficient truncation parameters (β). The earlier HLV encoding scheme [1] generated only three texture levels V_{orig}, V_{mid} and V_{base} resulting in six distinct HLV encoding levels, whereas the proposed HMLV encoding technique generates five Motion-and-Texture levels - the original layer, two intermediate layers and a two base layers. This results in a more fine-grained HMLV representation which make more efficient use of the available resources. This is very much evident from the power consumption profiles of different HMLV layers.

The proposed HMLV encoding scheme is shown to be effective for mobile Internet based multimedia applications such as face recognition and human activity recognition on resource-constrained mobile devices. The texture representation using progressive truncation of GWT coefficients gives the finer degree of control in generating texture layers which have different power consumption profiles. Our current experiments are limited to a single, potentially mobile device, i.e., a Dell Inspiron 1525 laptop PC with 2.0GHZ CPU, 2GB RAM, and a 250GB, 5400 rpm hard drive running in battery mode. We intend to include other types of mobile devices such as PDAs, iPhones and pocket-PCs in our future experiments. Our current experiments are also limited to the measurement of resource (battery power and bandwidth) consumption on the mobile end-user device on which the video is decoded and rendered. In our future work, we intend to investigate end-to-end computer vision systems and multimedia systems implemented in a mobile environment. This would involve a detailed study of both, resource consumption issues and mobile networking issues in each stage of the

computer vision system or multimedia system dealing with acquisition, encoding, storage, indexing, retrieval, transmission, decoding and rendering of the video data.

We also plan to perform a thorough quality 461 of the proposed HMLV encoding scheme based on an extensive survey of a significant user population. In such a survey, the users would be required to provide qualitative feedback on their viewing experience for the different HMLV states and compare their viewing experience with HMLV-encoded video to that with conventional MPEG-encoded video. The aforementioned survey would serve to ascertain the superiority of the proposed HMLV encoding scheme to the conventional MPEG encoding standard. In the current implementation, the HMLV encoding is done off-line. Consequently, the run times of the various procedures for generating the GSV and each of the texture layers V_{org}, V_{I1}, V_{I2} and V_{base}, V_{base0} are not very critical. Future work will focus on enabling real-time HMLV encoding.

Bibliography

[1] Chattopadhyay S, Bhandarkar S. M. Hybrid layered video encoding and caching for resource constrained environments, *Jour. Visual Communication and Image Representation*, 19(8):573-588, 2008.

[2] Sikora, T. Trends and prespectives in image and video coding, *Proceedings of the IEEE*, 93(1):6-17, 2005.

[3] Dai, M., Loguinov, D. Analysis and Modeling of MPEG-4 and H.264 Multi-Layer Video Traffic, *Proceedings of IEEE Infocom*, 2005.

[4] Feichtinger H G and Strohmer T Gabor Analysis Algorithms *Journal of Functional Analysis*, 208(1):194-228 2004.

[5] Lee, T.S., Image Representation using 2D Gabor Wavelets, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10) 1996.

[6] Daugman, J.G., Uncertainty relations for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters, *Journal of the Optical Society of America A*, 2:1160-1169, 1985.

[7] Hong, W., Bartels, M., Unsupervised Segmentation Using Gabor Wavelets and Statistical Features in LIDAR Data Analysis, *International Conference on Pattern Recognition*, 667-670, 2006.

[8] He, Z., Liang, Y., Chen, L., Ahmad, I., Wu, D., Power-rate-distortion analysis for wireless video communication under energy constraints, *IEEE Transactions on Circuits and Systems for Video Technology*, 15(5):645-658, 2005.

[9] Khan S., Shah, M. Object based segmentation of video using color motion and spatial information *IEEE Conf. Computer Vision and Pattern Recognition*, 2:746-751, 2001.

[10] Besl P. J., Jain, R. Segmentation through Variable-Order Surface Fitting *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10(2):167-192, 1988.

[11] Hakeem, A., Shafique, K., Shah, M. An object-based video coding framework for video sequences obtained from static cameras, *Proceedings of the 13th Annual ACM International Conference on Multimedia*, 608-617, 2005.

[12] Chattopadhyay S, Bhandarkar S. M, Li K. FMOE-MR: content-driven multi-resolution MPEG-4 fine-grained scalable layered video encoding, *Proc. ACM Multimedia Computing and Networking Conference (ACM MMCN. 07)*, 650404-1- 11, 2007.

[13] Chattopadhyay S, Bhandarkar S. M, Li K. Ligne-Claire Video Encoding for Power Constrained Mobile Environments, *Proc. ACM Conf. Multimedia, Augsburg, Germany, Sept. 2007*, 1036-1045, 2007.

[14] Liang, C., Mohapatra, S., Zarki, M.E., Dutt, N., Venkatasubramanian, N. backlight optimization scheme for video playback on mobile devices. *Proc. Consumer Communications and Networking Conference (CCNC 2006)*, 3(2):833-837, 2006.

[15] Cucchiara, R., Grana, C., Prati, A., Vezzani, R. Computer vision techniques for PDA accessibility of in-house video surveillance, *Proc. ACM SIGMM International Workshop on Video Surveillance, Berkeley, CA*, 8797, 2003.

[16] Ku, C.-W., Chen, L.-G., Chiu, Y.-M. A Very Low Bit-Rate Video Coding System based on Optical Flow and Region Segmentation Algorithms, *Proceeding of the SPIE Conf. Visual Communication and Image Processing, Taipei*, 3:13181327, 1995.

[17] Mohapatra, S., Cornea, R., Dutt, N., Nicolau, A., Venkatasubramanian, N. Integrated power management for video streaming to mobile handheld devices., *Proc. ACM Multimedia, Berkeley, CA, November 2003*, 582-592, 2003.

[18] Silven, O., Jyrkk, K. Observations on power-efficiency trends in mobile communication devices, *EURASIP Journal on Embedded Systems*, 2007(1):17-17, 2007.

[19] Bourgeois, J., Mory, E., Spies, F. Video transmission adaptation on mobile devices, *Journal of Systems Architecture*, 49:475-484, 2007.

[20] Chen, H., Wu, X., Hu, J. Adaptive K-Means clustering Algorithm, *Proc. of SPIE*, 6788,67882A(2):167-192, 2007.

[21] Richardson, I.E.G. : H.264 and MPEG-4 Video Compression, *Video Coding for Next Generation Multimedia. Wiley, New York (2004)*, 2004.

[22] Ni, P., Isovic, D., Fohler, G. : User-friendly H.264/AVC for remote browsing, *Proc. ACM Multimedia, Santa Barbara, CA, October 2006*, 643:646, 2006.

[23] Rosin, P.L., West, G.A.W. Non-parametric segmentation of curves into various representations, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(12):1140-1153, 1995.

[24] Canny, J. A computational approach to edge detection *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679-698, 1986.

[25] Shi, J., Tomasi, C. Good Features to Track, *IEEE Conference on Computer Vision and Pattern Recognition*, 593-600, 1994.

[26] Lucas, B. D., Kanade, T. An Iterative Image Registration Technique with an Application to Stereo Vision, *International Joint Conference on Artificial Intelligence*, 674-679, 1981.

[27] Atallah, M.J Linear time algorithm for the Hausdorff distance between convex polygons., *Information Processing Letters*, 17(4):207-209, 1983.

[28] Cedras, C., Shah, M. Motion-based recognition: A survey, *Image Vis. Comput.*,13(2):129-155, 1995.

[29] He, X. and Niyogi, P. Locality preserving projections, *presented at the Int. Conf. Advances in Neural Information Processing Systems*, 2003.

[30] Veeraraghavan, A., Roy-Chowdhury, A., Chellappa, R. Role of shape and kinematics in human movement analysis, *Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2004*, 1:730-737, 2004.

[31] Liang, W., Hu, W., Tan, T. Recent developments in human motion analysis, *Pattern Recognit.*, 36(3):585:601, 2003.

[32] Liang, W., David, S. Learning and Matching of Dynamic Shpae Manifolds for Human Action Recognition, *IEEE Trans. on Image Processing*, 16(6):1646-1661, 2007.

[33] He, X., Yan, S., Hu, Y., Niyogi, P., and Zhang, H. Face recognition using laplacianfaces, *IEEE Trans. Pattern Anal. Mach. Intell*, 27(3):328-340, 2005.

[34] Belkin, M., Niyogi, P. Laplacian eigenmaps and spectral techniques for embedding and clustering, *Proc. Int. Conf. Advances in Neural Information Processing Systems*, 585:591, 2001.

[35] Gorelick, L., Blank, M., Shechtman, E., Irani, M., Basri, R. Actions as Space-Time Shapes, *Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247-2253, 2007.

[36] Turk, M., Pentland, A. Eigenfaces for Recognition, *Jour. Cognitive Neuroscience*, 3(1):71-86, 1991.

[37] Deng, C., Xiaofei, H., Yuxiao, H., Jiawei, H., Thomas, H. Learning a Spatially Smooth Subspace for Face Recognition, *Proc. IEEE Conf. Computer Vision and Pattern Recognition Machine Learning*, 2007.

[38] Ferdinando, S., Andy, H. Parameterisation of a Stochastic Model for Human Face Identification, *Proceedings of 2nd IEEE Workshop on Applications of Computer Vision*, 1994.

[39] Sahoolizadeh, H., Sarikhanimoghadam, D., and Dehghani, H. Face Detection using Gabor Wavelets and Neural Networks, *World Academy of Science, Engineering and Technology 45 2008*, 2008.