Feature Representations for Human Activity Recognition in Color and Depth Sequences

Xiaodong Yang

Department of Electrical Engineering

The City College of New York
The City University of New York

A Dissertation Submitted for the Degree of

Doctor of Philosophy

2015

To my loving parents

Acknowledgements

I would like to first express my supreme gratitude to my adviser Prof. YingLi Tian for all the advice, patience, and support to my research and life. It is Prof. Tian who directs me into the challenging but rewarding fields of computer vision and machine learning. She always provides me with great freedom to explore my interests and ideas. Her insightful guidance and advising shape me towards a researcher for the future life.

I would also like to thank Dr. Zhu Liu and Dr. Raja Bala. They were my research mentors when I interned at AT&T Labs Research in 2013 and Xerox PARC in 2014, respectively. They spent quite a lot of time with me in research discussions and offered me a great deal of assistance in system development, paper revision, and patent application. I also cherish the research experiences of working with many wonderful people including Dr. Behzad Shahraray and David Gibbon at AT&T Labs Research as well as Dr. Jayant Kumar and Dr. Qun Li at Xerox PARC.

During my Ph.D. study, I have also obtained tremendous helps from many friends. I owe sincere gratitude to my colleagues including Dr. Shizhi Chen, Dr. Chucai Yi, Chenyang Zhang, Yang Xian, Xuejian Rong, Yuancheng Ye, and Shuai Yuan. Besides, my great appreciation also goes to Dr. Liangliang Cao for his generous suggestions and supports.

My special appreciation is dedicated to my parents with whom I share my success and frustration. The great love and encouragement from them motivate me to overcome difficulties and attain my goals. Finally, I would like to thank all the committee members (Prof. YingLi Tian, Dr. Rogerio Feris, Prof. Ioannis Stamos, Prof. Zhigang Zhu, and Prof. Michael Grossberg) for attending my dissertation defense and providing valuable comments to improve this dissertation.

Abstract

Nowadays tremendous amounts of visual data including images and videos are explosively increasing with the rapid development of digital cameras and the Internet. As one of the most active and promising research areas in computer vision, recognition of human activity has demonstrated great capacity and potential for automatic and intelligent analysis of visual information. Motivated by this trend and demand, this dissertation is dedicated to developing effective and efficient feature representation techniques to recognize human activities in both color and depth modalities. We first present a general surveillance event detection system which explores motion feature extraction, video clip representation, and detection model learning to handle the large-scale noisy and imbalanced data learning problem. We then propose a novel framework of super sparse coding vector to jointly model and aggregate low-level motion features and spatio-temporal cues into a discriminative and compact representation. Along with the advance of imaging techniques, the recent emergence of cost-effectiveness and easy-operation depth sensors greatly facilitates the task of human activity recognition. Our first approach using depth cameras is to employ the skeleton joints recovered from depth maps to capture the action information of static posture, motion property, and overall dynamics. Based on the projected depth maps, we also introduce the approach of depth motion maps to characterize the accumulated motion intensity and distribution in a global manner. In order to explicitly capture the 3-dimensional shapes and motion cues, we extend the surface normal to polynormal which is used to jointly encode the local geometric and temporal information. We then propose a novel scheme of super normal vector to aggregate the low-level polynormal into a discriminative representation. Extensive experiments demonstrate that the feature representation techniques developed in this dissertation achieve the state-of-the-art performances on various activity recognition tasks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the current era of data divulgence, we are now facing a flourishing of visual information including images and videos than ever before. As one of the most promising research areas in computer vision, recognizing human activity has caught the interest from both academia and industry in the past decades. Automatic human activity recognition has been widely applied to a number of real-world applications, e.g., surveillance event detection, human-computer interaction, content-based video search and summarization, etc. Most research work on activity recognition focuses on the videos captured by conventional visible light cameras. Recent emergence of cost-effective depth sensors facilitates a variety of visual recognition tasks including activity recognition as well. Then how to effectively represent and efficiently exploit human activities from large-scale video sequences is an open problem. This dissertation is dedicated to developing effective feature representations for human activity recognition in both color and depth sequences.

Recognition of human activity can be performed at various abstract levels. Here we adopt the taxonomies proposed in [49]. A *movement* is a primitive motion pattern that can be depicted at the limb level, e.g., right leg forward. An *action* contains a series of atomic movements, e.g., running. An *activity* consists of complex sequence of actions, e.g., a football team scoring a goal. These three levels roughly correspond to the low-level, mid-level, and high-level vision tasks. However, there is no hard boundary but a significant gray area among the three levels. In this dissertation, human activity is used to indicate the general categories if not specified. It can involve hand gestures, single person, multiple people, and human-object interactions.

Recent years have witnessed the growing popularity in applications of human activity recognition. With the relentless growth of online videos, it has become necessary to develop effective content-based video summarizing and searching methods [8] to improve video storage and indexing. Along with the deployment of

huge amounts of surveillance cameras, security agencies are also seeking intelligent solutions [15] to assist or replace human operators for traditional surveillance systems that are with heavy demand of human monitors. As one of the most important modes in nonverbal communication, understanding human activity is able to enable computers and environments to better interact with humans [17]. Similar to physical attributes (e.g., fingerprint), human activity (e.g., gait) can also function as important behavioral biometrics [57] to recognize human identity with the advantage that subject cooperation is not required in collection. Moreover, motion synthesis [16] is widely applied in gaming and movie industry to produce a large variety of realistic human actions.

It is of great challenge to recognize human activity from video sequences in the wild due to large variations caused by the factors such as viewpoint, occlusion, scaling, motion style, performance duration, cluttered background, etc. To represent human activity in a global way, the spatio-temporal volume based methods [4] [20] [78] are proposed by describing the region of interest of a person as a whole. These features usually originate from either edges, silhouettes, or optical flow. Similar to other global representations, they are sensitive to noise, occlusion, and viewpoint change. Most recent approaches of human activity recognition hinge on the bag-of-visual-words (BOV) representations [38] [43] [71] which consist in computing and aggregating statistics from local space-time features [72] [73]. In this framework, a video representation can be obtained by extracting low-level features, coding them over a visual dictionary, and pooling the codes in some well-chosen spatio-temporal cells which are used to globally capture the spatial layout and temporal order. A significant progress has been made in the development of local space-time features [12] [32] [37] [79]. After low-level feature extraction, the approaches similar to those used in image classification [39] [53] [82] are generally employed.

As the imaging technique advances, the advent of depth sensors (e.g., Microsoft Kinect and Asus Xtion Pro) brings great benefits to a variety of visual recognition tasks including object recognition [36], indoor place segmentation [61], environment reconstruction [54], as well as human activity recognition [51] [75] [87]. Compared to conventional color frames in activity recognition, depth sequences provide the following merits: (1) additional shape cues to provide more informative geometric description, which has been successfully applied to recover skeleton joints from a single depth map; (2) precluded color and texture, which significantly eases the problems of human detection and segmentation; (3) robustness to visible lighting, which greatly benefits to the systems monitoring in the dark environment. A number of representations of human activity in depth sequences have been explored, ranging from skeleton joints [75] [84], cloud points [74], projected depth maps [42] [89], local interest points [23] [80], to surface normals [51] [87].

The main contributions of this dissertation focus on developing effective feature representations for human activity recognition in both color and depth sequences. We present a set of low-level features (e.g., ActionHOG, depth motion map, polynormal) to jointly characterize the motion and appearance cues. We also introduce effective models to capture the spatial layout and temporal order (e.g., super location vector, adaptive spatio-temporal pyramid). Moreover, we propose novel schemes to aggregate low-level features into the discriminative representations (e.g., super sparse coding vector, super normal vector).

The remainder of this dissertation is organized as follows. We divide the whole dissertation into two main parts, i.e., feature representations in color videos (Chapters 2 and 3) and depth sequences (Chapters 4 to 6). Chapter 2 presents a general surveillance event detection system. We introduce an efficient method to extract spatio-temporal features and an effective algorithm to handle the highly imbalanced large-scale data learning. In Chapter 3, we propose a novel framework of super sparse coding vector (SSCV) to aggregate low-level motion features and model spatio-temporal information in a discriminative and compact representation. Chapter 4 describes an effective approach of EigenJoints based on skeleton joints to recognize human activities in depth sequences. A global representation of human activity using depth motion maps (DMM) is presented in Chapter 5. In Chapter 6, we propose a novel scheme of super normal vector (SNV) based on the extended surface normals. Finally, Chapter 7 concludes the remarks of this dissertation.

Chapter 2

Human Activity Recognition in Surveillance Event Detection

Automatic event detection of surveillance videos has many real-world security applications for public areas e.g., airports, banks, supermarkets, etc. [29] [83]. In comparison to some constrained scenarios with limited people and definite activities, real-world surveillance videos encounter various challenges including large variances of viewpoint, scaling, lighting, and highly cluttered background. We develop a general surveillance event detection system which is evaluated by the TRECVID Surveillance Event Detection (SED) [62]. This task provides a corpus of 144-hour surveillance videos under five camera views captured from the London Gatwick International Airport. In this dataset, 99-hour videos can be used as the training set with annotations of temporal extents and event labels. Our system is evaluated on all the seven events defined by this task, i.e., CellToEar, Embrace, ObjectPut, PeopleMeet, PeopleSplitUp, PersonRuns, and Pointing.

In this chapter, we present a general surveillance event detection system, one of the most important and challenging applications in human activity recognition. In the proposed system, a sliding temporal window is used as the detection unit, which is represented by a histogram of low-level spatio-temporal features including ActionHOG and STIP. We estimate the spatial priors of a variety of events through the spatial distributions of activities under different camera views in the training set. Compared to the linear kernel in SVM, non-linear kernels tend to have superior accuracy but with significantly increased computational cost in both training and testing. The explicit feature mapping is therefore employed to approximate non-linear kernels to facilitate the large-scale linear SVM learning. In order to deal with the highly imbalanced nature (i.e., negative samples are far more than positive samples) of surveillance data, we introduce the CascadeSVMs algorithm which contains a set of cascaded linear SVM corresponding to the specific events and camera views.

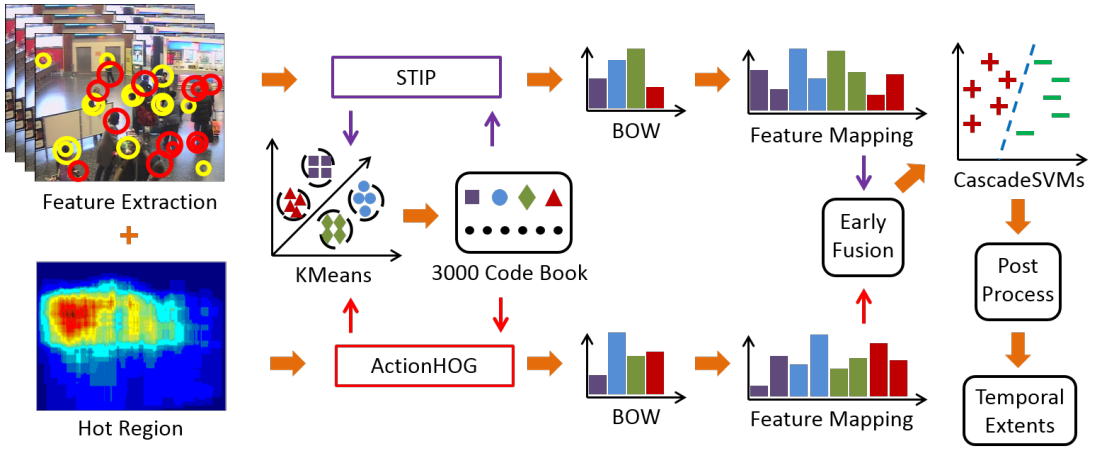


Figure 2.1: Overview of the proposed surveillance event detection system.

The remainder of this chapter is organized as follows. Section 2.1 introduces the overall system architecture. In Section 2.2 and 2.3, we provide detailed procedures of feature extraction and video representation. Section 2.4 describes the CascadeSVMs algorithm and post processing. A variety of experimental results and discussions are presented in Section 2.5. Finally, Section 2.6 summarizes the remarks of our system.

2.1 System Overview

Figure 2.1 demonstrates the surveillance event detection system which includes four main components: (1) low-level feature extraction, (2) video (temporal sliding window) representation, (3) event learning and prediction by CascadeSVMs, (4) post-processing to localize temporal extents of a detected event.

Most recent work on human activity recognition demonstrates that local spatio-temporal features are more robust to posture, occlusion, illumination, and cluttered background compared to global features. A spatio-temporal feature usually includes two phases: detection (i.e., a feature detector localizes interest points in a spatio-temporal space) and description (i.e., a feature descriptor computes representations of detected points). The space-time interest point (STIP) [37] employs extended Harris corner detector to localize interest points with large gradient magnitude in both spatial and temporal domains. Each interest point is described by the combination of histogram of gradients (HOG) and histogram of optical flow (HOF). However, it is quite restrictive to have large intensity changes in both spatial and temporal dimensions, which often results in insufficient detections. Instead of using spatio-temporal volumes, we propose to extract spatial

and temporal information separately as in [67]. An efficient detector, e.g., speed up robust features (SURF) [2], is first applied to extract salient points in the spatial domain. These points are then filtered by temporal or motion constraints, e.g., motion history image (MHI) [4], to remove the static background points. To characterize the appearance and motion information, we compute HOG features for each interest point from both image and motion channels. In addition, because of specific camera views and scenes in the surveillance scenario, the occurrence of specific events is usually biased to a certain spatial range [88]. We further exploit this spatial prior to eliminate a large amount of points from unrelated regions.

After local feature extraction, feature coding and pooling are used to aggregate the low-level spatio-temporal features to represent each video clip. We employ the bag-of-visual-words (BOV) to represent each temporal sliding window. In our system, a visual dictionary with the size of 3,000 is first learned through K-means. The local soft assignment [45] is used to encode low-level features. The local soft assignment coding is able to achieve comparable classification accuracy but with much less computational cost compared to other more complicated coding methods such as sparse coding [46] and locality-constrained linear coding [76]. After feature coding, we choose the max pooling to aggregate the coded features. Before learning event models, the explicit feature mapping [68] is applied to the BOV features. This is motivated by approximating large-scale non-linear SVM through the linear one which enjoys much more computational efficiency in both training and testing.

Having obtained the video clip representations, the event models can be learned by a linear SVM solver [14]. However, the surveillance data is highly imbalanced because positive events are far less frequent than negative ones, e.g., the sequences of the event CellToEar are only 0.31% of the entire video. We therefore introduce the CascadeSVMs algorithm to handle this difficulty. In each stage of this algorithm, positive and negative samples with the same amount are used to train a classifier that favors to the positive class. This leads each individual classifier to have a high detection rate but also a high false alarm rate. By cascading multiple classifiers (e.g., 5-7), we are able to filter out considerable false alarms but maintain a reasonable detection rate. The feature-level fusion is then employed to combine multiple features.

A simple post processing is performed over the detections to determine the temporal localization of each event and further remove false alarms. It is observed that most positive samples continuously last for a certain number of frames because temporal extents of most events could cover several sliding windows. We therefore merge neighboring positive predictions into a single positive detection. Based on our empirical observation, we also remove those isolated positive predictions or other positive ones mixed with too many negative predictions.

2.2 Low-Level Feature Extraction

We extract two types of low-level features including STIP and ActionHOG in our system. STIP [37] detects spatio-temporal interest points by searching significant variations in both space and time. HOG/HOF descriptors are then computed based on the space-time neighborhoods of detected interest points to capture the local appearance and motion information. STIP detector combined with HOG/HOF descriptors has been widely used in human activity recognition and detection tasks [85] [90]. However, it suffers insufficiency due to the rigorous assumptions of large gradients in space and time, as well as inefficiency because of computational cost in its detector and descriptor. In order to provide complementary motion features, we propose an efficient local spatio-temporal feature ActionHOG which detects interest points with spatially distinctive shapes and temporally sufficient motions.

An efficient interest point detector (e.g., Harris corner detector [25] or SURF detector [2]) is first employed to localize spatial interest points. Motion information is then used to remove the spatial interest points from static background, i.e., only those spatial interest points with sufficient motion are retained as the spatio-temporal interest points. We provide two motion channels including motion history image (MHI) and optical flow to eliminate the static spatial interest points. MHI [4] is a real-time motion template generated by stacking consecutive frame differences. As shown in Figure 2.2, the brighter pixels on MHI correspond to more recent motion. MHI gradients also reflect the action direction cues. We can use MHI as a motion mask to filter the static spatial interest points that are with lower intensities on the MHI channel. Moreover, the magnitude of optical flow associated with each spatial interest point can be utilized as a motion filter as well.

Figure 2.2: The spatio-temporal interest points detected by STIP and ActionHOG. Brighter pixels on MHI correspond to more recent motion. Gradients on MHI also provide motion direction information.

We compute HOG descriptor to characterize the local appearance information from the image channel and local motion cues from the MHI and dense optical flow channels. In our system, SURF detector is used to localize spatial interest points because it is able to generate scale information and maintain computational efficiency. The dominant orientations of interest points are removed as motion directions also provide important clues for activity recognition. MHI is employed as the motion channel to eliminate static spatial interest points and compute motion descriptors.

Figure 2.2 demonstrates the spatio-temporal interest points detected by SITP and ActionHOG. As shown in this figure, ActionHOG provides denser and complementary interest points to STIP. In addition, ActionHOG detects fewer points from background. Table 2.1 compares the computational costs and detected number of spatio-temporal interested points of STIP and ActionHOG. The statistics are based on a subset of the training data in TRECVID SED. We report the run time using C++ on a desktop with a single 2.13GHz CPU. As shown in this table, ActionHOG is over 10 times faster in terms of processing each frame and about 20 times faster in terms of computing each interest point than STIP. The number of detected points per frame of ActionHOG is about 2 times as that of STIP. Our source code for computing ActionHOG is available online.[1]

Table 2.1: Comparisons between STIP and ActionHOG. SURF detector is used as the spatial interest point detector and MHI is used as the motion channel in ActionHOG.

Feature	Speed (frm/sec)	Speed (ms/pnt)	Number (pnt/frm)
STIP	0.6	29.9	56
ActionHOG	6.4	1.5	107

2.3 Event Spatial Priors

Due to the highly cluttered background in surveillance videos, a significant amount of interest points are detected from the irrelevant activities. In order to get rid of these noisy points, we build hot region masks to model the spatial priors according to the specific events and camera views. Since the surveillance videos are captured by fixed cameras in a certain public area, we observe that the occurrence of certain events is biased or concentrates in some specific regions as shown in Figure 2.3. In order to capture the spatial priors, we manually annotate the bounding

[1] http://yangxd.org/code

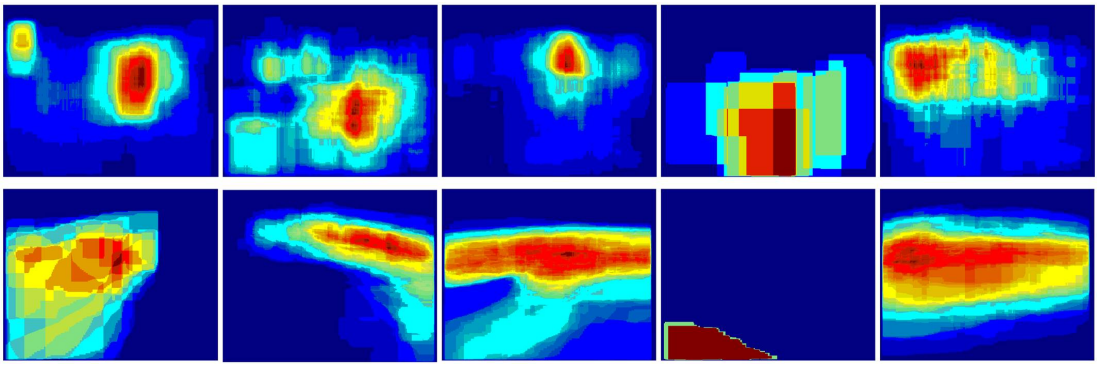


Figure 2.3: Examples of spatial priors, i.e., hot region maps, of event ObjectPut (top) and event PersonRuns (bottom) corresponding to the five camera views from left to right.

boxes of people performing activities to build the hot regions. A hot region map $H_{c,e}$ of camera view c and event e is obtained by $H_{c,e} = \left(\sum_i A^i_{c,e}\right) / N_{c,e}$, where $A^i_{c,e}$ is the ith annotated frame (a binary map) in camera view c and event e with foreground or activity region in a bounding box and $N_{c,e}$ is the total number of annotated frames for camera view c and event e. This spatial prior can be utilized to differentiate activity and non-activity regions by thresholding $H_{c,e} > \mu_{c,e}$. The interest points from the non-activity regions are removed in the following process as illustrated in Figure 2.1.

2.4 Video Clip Representation

We employ BOV combined with the spatial pyramid to represent each sliding window. The spatial pyramid [39] is applied to globally and roughly capture the spatial geometry of a video scene. It subdivides a video into a set of spatial cells in a coarse-to-fine manner. Each cell is represented independently and the cell-level histograms are finally concatenated as the video-level histogram. The temporal pyramid introduced in [38] is not used due to the explosion of feature dimension and memory cost.

A visual dictionary $\mathbf{D}_{m\times n} = (\mathbf{d}_1 \dots \mathbf{d}_n)$ for each low-level features is obtained by K-means. The dictionary size n is empirically set to 3,000 and $\mathbf{d}_k \in \mathbb{R}^m$ is a visual word. The local soft assignment [45] is used to code each feature $\mathbf{x}_i \in \mathbb{R}^m$ to $\boldsymbol{u}_i$:

$$\boldsymbol{u}_{i,k} = \frac{\exp\left(-\beta \hat{d}\left(\boldsymbol{x}_i, \mathbf{d}_k\right)\right)}{\sum_{j=1}^{n} \exp\left(-\beta \hat{d}\left(\boldsymbol{x}_i, \mathbf{d}_j\right)\right)}, \tag{2.1}$$

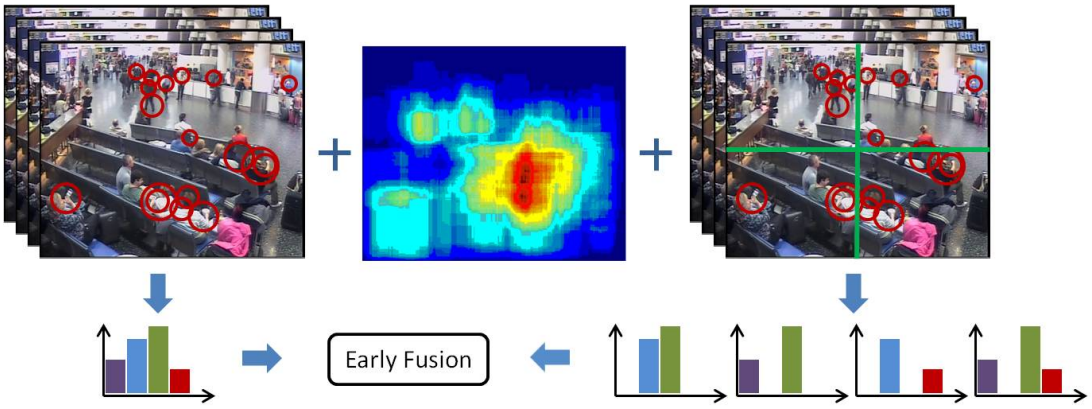


Figure 2.4: The spatial pyramid of 2 levels with 1×1 and 2×2 spatial grids in each level. A camera and event dependent spatial prior map is used to remove feature points from background.

$$\hat{d}\left(\boldsymbol{x}_i, \mathbf{d}_k\right) = \begin{cases} \|\boldsymbol{x}_i - \mathbf{d}_k\|^2 & \text{if } \mathbf{d}_k \in N_K\left(\boldsymbol{x}_i\right), \\ \infty & \text{otherwise,} \end{cases} \tag{2.2}$$

where $\boldsymbol{u}_{i,k}$ denotes the kth coefficient of code $\boldsymbol{u}_i$, $\hat{d}\left(\boldsymbol{x}_i, \mathbf{d}_k\right)$ is a local version of the original distance $\|\boldsymbol{x}_i - \mathbf{d}_k\|^2$, $N_K(\boldsymbol{x}_i)$ are the K nearest neighboring descriptors of $\boldsymbol{x}_i$, and β is a smoothing factor. In our system, we empirically set $K = 200$ and $\beta = -1$.

The max pooling is then used to aggregate $\boldsymbol{u}_i$ from a spatial grid w_j of a temporal sliding window by:

$$\boldsymbol{h}_{j,k} = \max_{i \in w_j} \boldsymbol{u}_{i,k}, \text{ for } k = 1, \ldots, n, \tag{2.3}$$

where $\boldsymbol{h}_{j,k}$ denotes the kth coefficient in $\boldsymbol{h}_j$ and w_j $(j = 1, \ldots, W)$ is the jth spatial grid. We use two levels of grids (1×1 and 2×2) so each sliding window generates $W = 5$ spatial pyramid grids as shown in Figure 2.4. We concatenate the histograms $\boldsymbol{h}_j$ into $\boldsymbol{h}$ as the BOV representation for each temporal sliding window. Note: in the coding and pooling process, we only use the features $\boldsymbol{x}_i$ within the activity regions in the corresponding spatial prior maps, as illustrated in Figure 2.4.

In our system, a sliding window with the size of 60 frames strides in every 15 frames. This generates a large amount of data, e.g., 600K samples from training set. It could be very inefficient for SVM with non-linear kernels to learn and predict on such scale data. On the other hand, SVM with the linear kernel is in general much more efficient in both training and testing. However, linear SVM tends to have inferior recognition accuracy compared to the non-linear ones. In order to deal with this difficulty, we approximate non-linear kernels by the

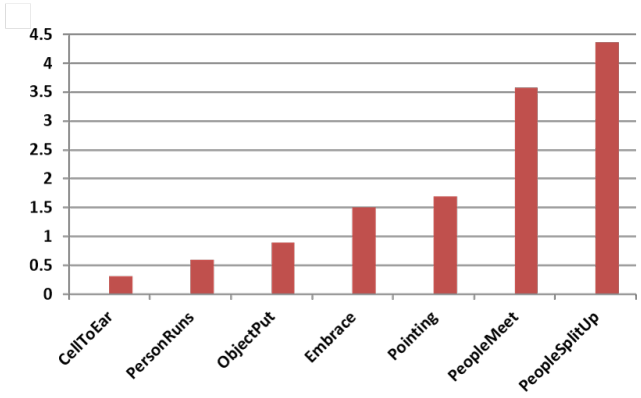


Figure 2.5: Proportions (%) of video sequences containing positive events in the training set.

explicit feature mapping [68] to enable more efficient linear SVM with little loss in accuracy. In feature mapping, each feature vector $\boldsymbol{h} \in \mathbb{R}^h$ is mapped to a feature space with moderately higher dimensionality through an explicit feature mapping $\psi : \mathbb{R}^h \rightarrow \mathbb{R}^{(2r+1)h}$ such that the inner product in this space can approximate the non-linear kernel distance $\mathcal{K}$, i.e., $\left\langle \psi(\boldsymbol{h}), \psi(\boldsymbol{h}^{'}) \right\rangle \approx \mathcal{K}(\boldsymbol{h}, \boldsymbol{h}^{'})$. We set $r = 3$ in our system to approximate the χ^2 kernel.

2.5 Event Model Learning

The temporal sliding window scheme (e.g., 60-frame window strides in every 15 frames) used in the system generates quite imbalanced data (i.e., negative samples are far more than positive samples) as shown in Figure 2.5. Most positive events are less than 2% of the entire video sequences. For example, CellToEar and PeopleSplitUp are the least and most frequent events which are only 0.3% and 4.4% of the whole training video sequences, respectively. In order to overcome this imbalance, we introduce the CascadeSVMs algorithm. In each stage of this algorithm, the same-amount positive and negative samples are employed to train a classifier that favors to the positive class. This leads the classifier in each stage to a high detection rate but a high false alarm rate as well. We then mine the hard negative samples for training in the next stage. By cascading multiple classifiers, it is able to remove a significant amount of false alarms but maintain a reasonable detection rate. In order to reduce the intra-class variance and memory requirement, the event models are learned according to each specific event and camera view. This system therefore contains 35 models of 7 events under 5 camera views.

Algorithm 1: CascadeSVMs algorithm for learning event models

Input: training set $S := \{S^+, S^-\}$
maximum iteration number c
initialization $C_0 := \{\}$ and $S_1^- :=$ random $|S^+|$ samples from S^-
Output: event and camera view dependent model C

for $i := 1$ **to** c **do**
  $w^+ := 1.0$ and $w^- := 1.0$
  for $j := 1$ **to** t **do**
    $M_i :=$ **LIBLINEAR**(S^+, S_i^-, w^+, w^-)
    positive accuracy $:= M_i(S^+)$
    if positive accuracy $> \theta$ **then**
      break
    end
    $w^+ := w^+ + \tau$
  end
  $C_i := C_{i-1} \cup M_i$
  $S^- := \{s \mid s \in S^-$and $C_i(s) =$ positive$\}$
  SORTSAMPLES(S^-)
  $S_{i+1}^- := S^-(1, \ldots,$ num$)$, where num $:= \min(|S^+|, |S^-|)$
  if $|S_{i+1}^-| < |S^+|$ **then**
    break
  end
end

Suppose we have a training set $S = \{S^+, S^-\}$ for each event under each camera view. The CascadeSVMs algorithm adaptively subdivides the negative set into a series of partitions S_i^- with the same size of $|S^+|$ according to the ranked prediction scores and iteratively learns a group of binary linear SVM classifiers M_i that favors to positive samples. These classifiers are cascaded as the event model $C = \{M_1, \ldots, M_{|C|}\}$. The outline of our proposed learning process is shown in Algorithm 1.

The classifier M_i in the ith stage is learned by an adaptive weighting method. This is used to ensure a classifier in each stage could correctly predict most positive samples. We initialize both positive and negative weights as 1.0. After training a classifier by LIBLINEAR [14], we evaluate this classifier only on the positive samples. If the classification accuracy is greater than a threshold $\theta = 0.9$, this classifier is assigned to M_i. Otherwise, the positive weight is increased by $\tau = 0.05$ and the classifier is retrained with the updated class weights. This process is repeated until the accuracy on positive samples is up to θ or the maximum

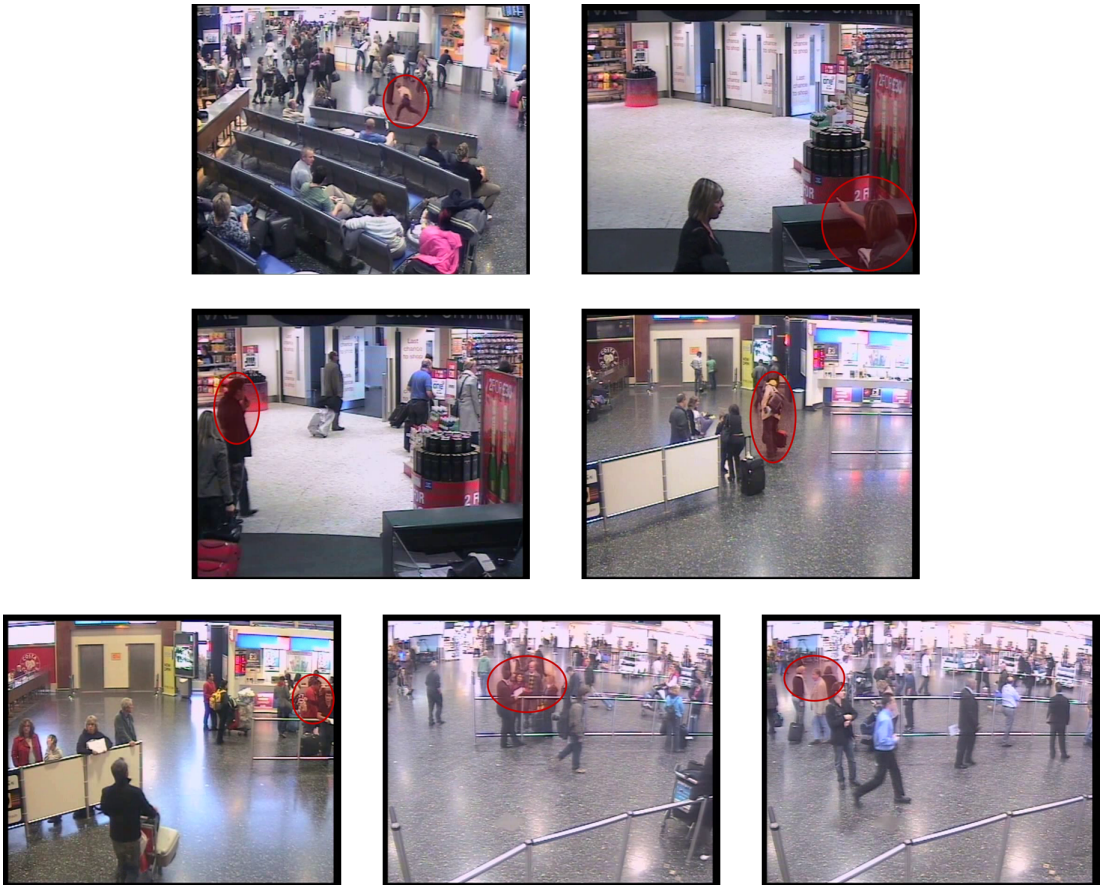


Figure 2.6: Red regions demonstrate the events defined in the TRECVID SED. From top to bottom and left to right: PersonRuns, Pointing, CellToEar, ObjectPut, Embrace, PeopleMeet, and PeopleSplitUp.

iteration number t is reached. As the positive samples are much less than the negative ones, we employ all the positive samples for training. In each stage, we keep the negative samples S_i^- as the same amount of the positive ones. To update S_i^-, we first filter S^- by only preserving those negative samples that cannot be correctly classified by using the current classifier pool C_i. We then sort the left negative samples from S^- in descending order based on their scores and choose the first $|S^+|$ samples (more confusing ones) as S_{i+1}^-. This cascading process terminates if the maximum iteration number $c = 10$ is reached or the left negative samples are fewer than the positive ones. In our system, the size $|C|$ of event models is between 5 and 10. Our source code of CascadeSVMs is available online.[1]

[1]http://yangxd.org/code

Event	Actual DCR		Minimum DCR	
	2011 Best	Ours	2011 Best	Ours
CellToEar	1.0365	1.0086	1.0003	1.0003
Embrace	0.8840	0.9552	0.8658	0.9351
ObjectPut	1.0006	1.0158	0.9983	1.0003
PeopleMeet	0.9820	1.0082	0.9724	0.9885
PeopleSplitUp	0.9099	0.9843	0.8809	0.9787
PersonRuns	0.8924	0.9702	0.8370	0.9623
Pointing	0.9783	1.0895	0.9730	0.9987

Figure 2.7: Comparisons between our results and the best results in 2011.

Event	Rank	Best 2012 System ADCR	MediaCCNY Primary Run				
			ADCR	MDCR	#CorDet	#FA	#Miss
CellToEar	3	1.0007	1.0086	1.0003	1	42	193
Embrace	4	0.8000	0.9552	0.9351	20	212	155
ObjectPut	3	0.9983	1.0158	1.0003	1	53	620
PeopleMeet	2	0.9799	1.0082	0.9885	14	120	435
PeopleSplitUp	3	0.8433	0.9843	0.9787	6	50	181
PersonRuns	2	0.8346	0.9702	0.9623	6	80	101
Pointing	5	0.9813	1.0895	0.9987	29	356	1034

Figure 2.8: Comparisons between our system (MediaCCNY) and best systems in TRECVID SED 2012.

2.6 Experimental Results

As illustrated in Figure 2.6, we experiment on all the 7 events defied by NIST. They correspond to the three-level categories, i.e., single-person event (PersonRuns, Pointing), person-object event (CellToEar, ObjectPut), multiple-people event (Embrace, PeopleMeet, PeopleSplitUp). In TRECVID SED 2012, 15-hour of videos with the frame resolution of 720×576 at 25 fps captured by 5 fixed cameras are provided as the evaluation set. Because of the temporal sliding window scheme used in the system, a continuous event could be chopped into several detected windows. Therefore after classifier predictions, we employ a post processing to group the continuous positive windows to determine the final temporal location of a detected event. In the merging process, we use a tolerance λ_t (3 used in our system) which means two positive predictions disconnected by less than λ_t negative predictions can still be merged together. The other benefit of

post processing is to further remove false alarms. After the merging process, a group will be removed from positive detections if the ratio of negative predictions (holes) in a merged group is greater than λ_h (0.3 used in our system).

We fist compare our results to the best results of 2011 in Figure 2.7 by the detection cost rate (DCR) which is a linear combination of miss detections and false alarms. A perfect system has 0 DCR meaning 0 miss detection and 0 false alarm. Therefore the lower DCR is, the more robust a system is. Actual DCR and Minimum DCR are the primary and secondary metrics, respectively. As shown in Figure 2.7, we achieve the best ADCR in the event of CellToEar. The performances of our system evaluated in all 7 events are among the top 3 compared to other systems. Figure 2.8 presents the comparison between our system and the best systems in 2012. The rank column indicates our rankings among all participants in terms of Actual DCR. Our system achieves top 3 performance in CellToEar, ObjectPut, PeopleMeet, PeopleSplitUp, and PersonRuns.

In general, the number of correct detections is still much lower than the number of missed ones in CellToEar and ObjectPut. From a high lever observation, the performance of single person events (e.g., PersonRuns) and multiple people events (e.g., PeopleMeet) have relatively better results than the person-object events (e.g., ObjectPut). The detection error tradeoff (DET) curves of our system and other participants in all events are demonstrated in Figure 2.9. These curves plot event-averaged miss detection probabilities vs. false alarm rates by varying a detection threshold.

2.7 Summary

In this chapter, we have presented the detailed implementations of a general surveillance event detection system. Our system starts from extracting low-level features of STIP-HOG/HOF and ActionHOG from each temporal sliding window. The camera and event dependent spatial priors are applied to remove detected features from the non-activity regions. We employ local soft assignment and max pooling to aggregate the filtered low-level features. The final representation is further augmented by feature mapping and spatial pyramid. We combine multiple features through feature-level fusion. The CascadeSVMs algorithm is introduced to handle the large-scale noisy and imbalanced data learning. In the evaluation of 7 event detection tasks of TRECVID SED 2012, our system achieves top 3 performances in 5 events.

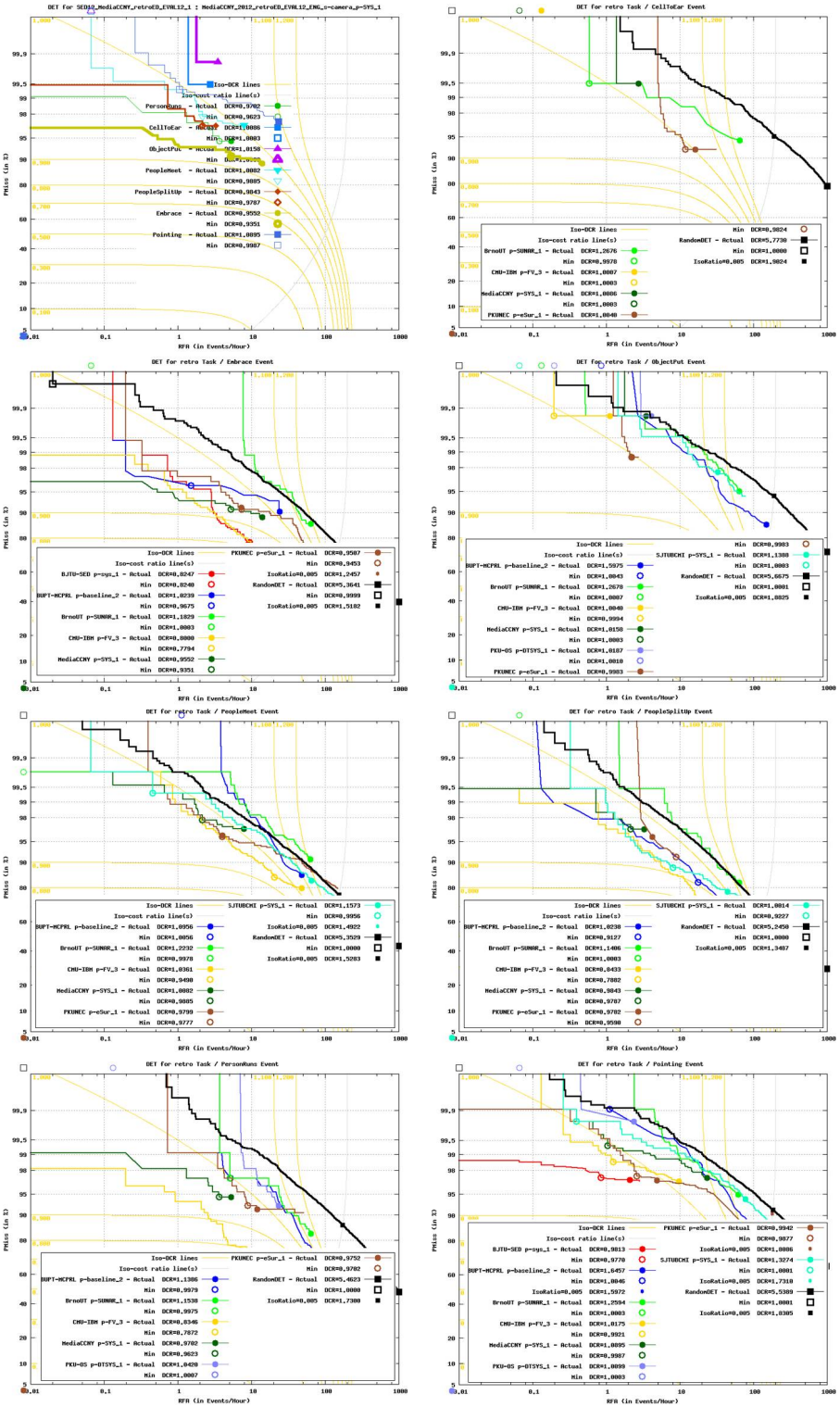


Figure 2.9: The detection error tradeoff (DET) curves of our system and other systems in each event.

Chapter 3

Super Sparse Coding Vector with Spatio-Temporal Awareness

In this chapter, a novel framework for human activity recognition is proposed based on sparse coding. We introduce an effective coding scheme to aggregate low-level descriptors into the super descriptor vector (SDV). In order to incorporate the spatio-temporal information, we present a novel approach of the super location vector (SLV) to model the space-time locations of local interest points in a much more compact way compared to the spatio-temporal pyramid representations. SDV and SLV are in the end combined as the super sparse coding vector (SSCV) which jointly models the motion, appearance, and location cues. This representation is computationally efficient and yields superior performance while using linear classifiers. In the extensive experiments, our approach significantly outperforms the state-of-the-art results on the public benchmark datasets.

As introduced in the previous chapter, most recent recognition approaches of human activity are based on the bag-of-visual-words (BOV) representations. In the basic BOV framework, a visual dictionary is learned by K-means and used to quantize low-level features through hard-assignment [38]. A number of coding variants have been proposed and reported to achieve the state-of-the-art results in image and video recognition, e.g., local soft assignment [45], sparse coding [82], and locality-constrained linear coding [76]. These approaches reduce information loss by relaxing the restrictive cardinality constraint in coding descriptors. Accordingly, average pooling can be replaced by max pooling [82]. Recently, several coding schemes have emerged to encode descriptors with respect to the visual words that they are assigned to, e.g., Fisher vector [53], super vector coding [93], and vector of locally aggregated descriptors [28]. These methods usually retain high order statistics and yield noticeably better results [56].

The basic BOV aggregates the assignments over an entire video sequence to generate the final representation. This obviously incurs a loss of information

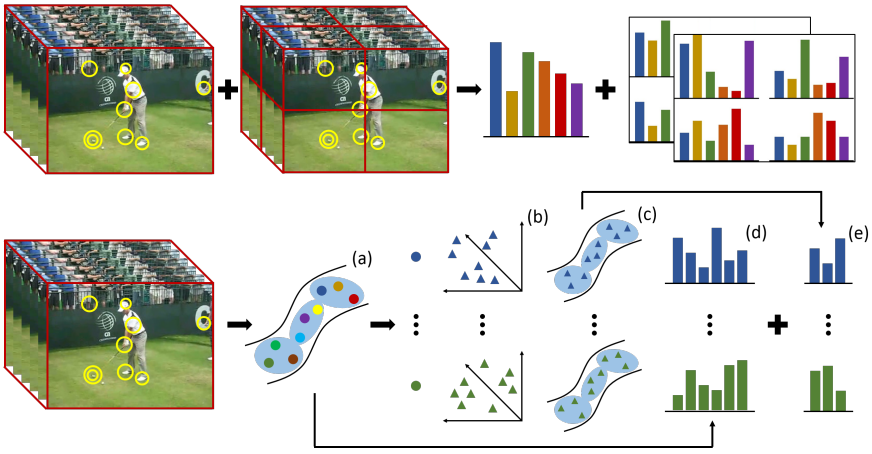


Figure 3.1: Frameworks of STP (up) and SSCV (bottom). STP represents a video by concatenating BOVs from the entire sequence and spatio-temporal cells. SSCV jointly models the motion, appearance, and location information. (a) A visual dictionary of descriptors is learned by sparse coding. (b) 3D space-time locations are associated to each visual word in (a) according to the assignments of descriptors. (c) A visual dictionary of locations is learned by sparse coding for each set in (b). SSCV is obtained by the combination of (d) SDV and (e) SLV.

by discarding all the spatio-temporal locations of local space-time features. An extension to the completely orderless BOV is the spatio-temporal pyramid (STP) [38], inspired by the spatial pyramid matching (SPM) [39] for image classification. In this approach, a video sequence is repeatedly and evenly subdivided into a set of spatial and temporal cells where descriptor-level statistics are pooled. It can be used to roughly capture the spatial layout and temporal order of an action sequence. However, the concatenation of BOV histograms over a number of subvolumes of a video dramatically increases feature dimensions, which further increase the learning and memory costs.

In this chapter, we propose a novel recognition framework on low-level feature coding and spatio-temporal information modeling, as illustrated in Figure 3.1. We first employ a sparse coding method [46] to compute the visual dictionary and coefficients of local descriptors. Each descriptor is coded by recording the difference of the local descriptor to all visual words. The coefficient-weighted difference vectors are then aggregated for each visual word through the whole video. These vectors of all visual words are in the end concatenated as the representation of super descriptor vector (SDV), which is used to characterize the motion and appearance cues. We further model the spatio-temporal information by computing the super location vector (SLV) of the space-time coordinates of local descriptors assigned to each visual word. We combine SDV and SLV as the

super sparse coding vector (SSCV) which jointly models the motion, appearance, and spatio-temporal information.

The main contributions of this chapter are summarized as follows. First, we provide an effective coding scheme to aggregate low-level features into a discriminative representation, which hinges on a much smaller visual dictionary. Second, we propose a novel approach to incorporate the spatio-temporal information in a much more compact representation, which correlates and models the motion, appearance, location cues in a unified way. Third, we perform a systematic evaluation of the state-of-the-art coding and pooling methods in the context of human activity recognition.

The remainder of this chapter is organized as follows. Section 3.1 introduces the related work of feature aggregation and spatio-temporal models. Section 3.2 describes the detailed procedures to compute SDV, SLV, and SSCV. A variety of experimental results and discussions are presented in Section 3.3. Finally, Section 3.4 summarizes the remarks of this chapter.

3.1 Related Work

In this section, we introduce the notations used throughout this chapter and summarize the related work on aggregating local descriptors and modeling spatial (temporal) information. We represent a video sequence $\mathcal{V}$ by a set of low-level descriptors $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ in $\mathbb{R}^{m \times n}$ and associated locations $\boldsymbol{\mathcal{L}} = \{\boldsymbol{l}_1, \ldots, \boldsymbol{l}_n\}$ in $\mathbb{R}^{3 \times n}$. C indicates the space-time cells defined in a spatio-temporal pyramid with C_j denoting the jth cell. $\mathbf{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_K\}$ is a visual dictionary with K visual words $\mathbf{d}_k \in \mathbb{R}^m$.

3.1.1 Feature Aggregation

Let $\mathcal{F}$ and $\mathcal{G}$ denote the coding and pooling operators, respectively. The final representation of $\mathcal{V}$ is the vector $\boldsymbol{z}$ obtained by sequentially coding, pooling, and concatenating over all space-time cells:

$$\boldsymbol{\alpha}_i = \mathcal{F}(\boldsymbol{x}_i), \quad i = 1, \ldots, n, \tag{3.1}$$

$$\boldsymbol{h}_j = \mathcal{G}\left(\{\boldsymbol{\alpha}_i\}_{i \in C_j}\right), \quad j = 1, \ldots, |C|, \tag{3.2}$$

$$\boldsymbol{z}^T = \left[\boldsymbol{h}_1^T \ldots \boldsymbol{h}_{|C|}^T\right]. \tag{3.3}$$

In the basic BOV framework, hard assignment $\mathcal{F}$ minimizes the distance of $\boldsymbol{x}_i$ to $\mathbf{D}$ which is usually learned by K-means. $\mathcal{G}$ performs the averaging over each pooling cell C_j:

$$\boldsymbol{\alpha}_i \in \{0,1\}^K, \quad \boldsymbol{\alpha}_{i,j} = 1 \text{ iff } j = \arg\min_k \|\boldsymbol{x}_i - \mathbf{d}_k\|_2^2, \tag{3.4}$$

$$\boldsymbol{h}_j = \frac{1}{|C_j|} \sum_{i \in C_j} \boldsymbol{\alpha}_i. \tag{3.5}$$

In order to enhance the probability density estimation, soft assignment was introduced in [18]. It codes a descriptor $\boldsymbol{x}_i$ by multiple visual words in $\mathbf{D}$ using a kernel function (e.g., the Gaussian function) of the distance between $\boldsymbol{x}_i$ and $\mathbf{d}_k$. Liu et al. proposed the local soft assignment in [45] to further improve the membership estimation to visual words. By taking account of the underlying manifold structure of local descriptors, $\mathcal{F}$ in local soft assignment only employs the $\mathcal{K}$ nearest visual words $N_{\mathcal{K}}(\boldsymbol{x}_i)$ to code a descriptor $\boldsymbol{x}_i$ and sets its distances of the remaining visual words to infinity:

$$\boldsymbol{\alpha}_{i,k} = \frac{\exp\left(-\beta \hat{d}\left(\boldsymbol{x}_i, \mathbf{d}_k\right)\right)}{\sum_{j=1}^{K} \exp\left(-\beta \hat{d}\left(\boldsymbol{x}_i, \mathbf{d}_j\right)\right)}, \tag{3.6}$$

$$\hat{d}\left(\boldsymbol{x}_i, \mathbf{d}_k\right) = \begin{cases} \|\boldsymbol{x}_i - \mathbf{d}_k\|^2 & \text{if } \mathbf{d}_k \in N_{\mathcal{K}}\left(\boldsymbol{x}_i\right), \\ \infty & \text{otherwise,} \end{cases} \tag{3.7}$$

where β is a smoothing factor to control the softness of assignment. As for $\mathcal{G}$ in the local soft assignment, it was observed that max pooling in the following equation outperformed average pooling:

$$\boldsymbol{h}_{j,k} = \max_{i \in C_j} \boldsymbol{\alpha}_{i,k}, \text{ for } k = 1, \ldots, K. \tag{3.8}$$

Parsimony has been widely employed as a guiding principle to compute sparse representation with respect to an overcomplete visual dictionary. Sparse coding [46] approximates $\boldsymbol{x}_i$ by using a linear combination of a limited number of visual words. It is well known that the ℓ_1 penalty yields a sparse solution. So the sparse coding problem can be solved by:

$$\min_{\mathbf{D}, \boldsymbol{\alpha}} \frac{1}{n} \sum_{i=1}^{n} \left(\frac{1}{2} \|\boldsymbol{x}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1 \right), \tag{3.9}$$

$$\text{subject to } \mathbf{d}_k^T \mathbf{d}_k \leq 1, \forall k = 1, \ldots, K,$$

where λ is the sparsity-inducing regularizer to control the number of non-zero coefficients in $\boldsymbol{\alpha}_i$. It is customary to combine sparse coding with max pooling as shown in Eq. (3.8).

Fisher vector [53] extends the BOV representation by recording the deviation of $\boldsymbol{x}_i$ with respect to the parameters of a generative model, e.g., the Gaussian mixture model (GMM) characterized by $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\sigma}_k,\ k = 1, \ldots, K\}$. π_k, $\boldsymbol{\mu}_k$, and $\boldsymbol{\sigma}_k$ are the prior mode probability, mean vector, and covariance matrix (diagonal), respectively. Let γ_i^k be the soft assignment of $\boldsymbol{x}_i$ to the kth Gaussian component. We obtain the Fisher vector of $\boldsymbol{X}$ by concatenating the gradient vectors from K Gaussian components:

$$\boldsymbol{\rho}_k = \frac{1}{n\sqrt{\pi_k}} \sum_{i=1}^{n} \gamma_i^k \left(\frac{\boldsymbol{x}_i - \boldsymbol{\mu}_k}{\boldsymbol{\sigma}_k} \right), \tag{3.10}$$

$$\boldsymbol{\tau}_k = \frac{1}{n\sqrt{2\pi_k}} \sum_{i=1}^{n} \gamma_i^k \left[\frac{(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^2}{\boldsymbol{\sigma}_k^2} - 1 \right], \tag{3.11}$$

where $\boldsymbol{\rho}_k$ and $\boldsymbol{\tau}_k$ are m-dimensional gradient vectors with respect to $\boldsymbol{\mu}_k$ and $\boldsymbol{\sigma}_k$ of the kth Gaussian component. The relative displacements of descriptors to the mean and variance in Eq. (3.10-3.11) retain more information lost in the traditional coding process. The superiority of Fisher vector was recently identified in both image classification [56] and activity recognition [77].

3.1.2 Spatial and Temporal Information

The orderless representation of a video completely ignores the spatial and temporal information, which could have conveyed discriminative cues for activity recognition. We briefly outline the relevant representative work that attempts to account for the spatial and temporal locations of low-level features.

The dominant approach to incorporate spatial and temporal information is the spatio-temporal pyramid (STP), as illustrated in Figure 3.1. Inspired by the spatial pyramid matching (SPM) [39], Laptev et al. [38] proposed to partition a video into a set of space-time cells in a coarse-to-fine manner. Each cell is represented independently and the cell-level histograms $\boldsymbol{h}_j$ are finally concatenated into the video-level histogram $\boldsymbol{z}$ as in Eq. (3.2-3.3). This representation has been proven to be effective when the action categories exhibit characteristic spatial layout and temporal order.

In image classification, the feature augmentation based methods were proposed in [48] [55] to append a weighted location $\boldsymbol{l}_i$ to the corresponding descriptor $\boldsymbol{x}_i$. As opposed to SPM, this approach does not increase the feature dimensionality thus makes the learning more efficient. Krapac et al. [33] introduced the

spatial Fisher vector to encode the spatial layout of local image features. The location model can be learned by computing per visual word the mean and variance of spatial coordinates of the assigned local image patches. While these representations are more compact, the evaluation results only showed marginal improvements over SPM in terms of classification accuracy.

3.2 Super Sparse Coding Vector

We describe the detailed procedures of computing SSCV in this section. We propose a novel feature coding scheme based on sparse coding to aggregate descriptors and locations into discriminative representations. The space-time locations are included as part of the coding step, instead of only coding motion and appearance cues and leaving the spatio-temporal coherence to be represented in the pooling stage. This enables SSCV to jointly characterize the motion, appearance, and location information.

3.2.1 Modeling Space-Time Features

We represent each local feature as the descriptor-location tuple $\boldsymbol{f}_i = (\boldsymbol{x}_i, \boldsymbol{l}_i)$. By employing a generative model (e.g., GMM) over descriptors and locations, we model $\boldsymbol{f}_i$ as:

$$p(\boldsymbol{f}_i) = \sum_{k=1}^{K} p(w=k)p(\boldsymbol{x}_i|w=k)p(\boldsymbol{l}_i|w=k), \tag{3.12}$$

where $p(w=k)$ indicates the prior mode probability of the kth Gaussian component in the descriptor mixture model, and w is the assignment index. We assume the prior mode probabilities are equal, i.e., $p(w=k) = 1/K, \forall k$. The kth Gaussian of descriptors is defined by:

$$p(\boldsymbol{x}_i|w=k) = \mathcal{N}\left(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\sigma}_k\right), \tag{3.13}$$

where $\boldsymbol{\mu}_k$ and $\boldsymbol{\sigma}_k$ are the mean and covariance (diagonal) of the kth Gaussian. As illustrated in Figure 3.1, we jointly model the spatio-temporal information by associating the locations of descriptors to the corresponding visual descriptor word, i.e., the Gaussian of descriptors in this context. We define the spatio-temporal model by using a GMM distribution over the locations associated with the kth visual word:

$$p(\boldsymbol{l}_i|w=k) = \sum_{g=1}^{G} \pi_{k_g} \mathcal{N}\left(\boldsymbol{l}_i; \boldsymbol{\mu}_{k_g}, \boldsymbol{\sigma}_{k_g}\right), \tag{3.14}$$

where π_{k_g}, $\boldsymbol{\mu}_{k_g}$, and $\boldsymbol{\sigma}_{k_g}$ are the prior mode probability, mean, and covariance (diagonal) of the gth Gaussian of locations in the kth visual descriptor word. We again assume the prior mode probabilities are equal, i.e., $\pi_{k_g} = 1/G, \forall k, g$.

3.2.2 Computing Super Descriptor Vector

We utilize sparse coding to learn a visual dictionary and code descriptors. We aggregate the coefficient-weighted differences between local descriptors and visual words into a vector, rather than directly pooling the coefficients.

The likelihood or generation process of $\boldsymbol{x}_i$ is modeled by the probability density function in Eq. (3.13). The gradient of the log-likelihood of this function with respect to its parameters describes the contribution of the parameters to the generation process [26]. Here we focus on the gradient with respect to the mean:

$$\frac{\partial \ln p(\boldsymbol{x}_i | w = k)}{\partial \boldsymbol{\mu}_k} = \rho_i^k \boldsymbol{\sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k) , \tag{3.15}$$

where ρ_i^k denotes the posterior $p(w = k|\boldsymbol{x}_i)$. If we make the three approximations:

1. the posterior is estimated by the sparse coding coefficient, i.e., $\rho_i^k = \alpha_i^k$,
2. the mean is represented by the visual word in sparse coding, i.e., $\boldsymbol{\mu}_k = \mathbf{d}_k$,
3. the covariance is isotropic, i.e., $\boldsymbol{\sigma}_k = \epsilon \mathbb{I}$ with $\epsilon > 0$,

Eq. (3.15) can be simplified to $\alpha_i^k (\boldsymbol{x}_i - \mathbf{d}_k)$, where α_i^k is the coefficient of the ith descriptor $\boldsymbol{x}_i$ to the kth visual word $\mathbf{d}_k$ in Eq. (3.9).

We choose sparse coding in the approximation because it is much cheaper to compute the means (dictionary) compared to the Expectation Maximization (EM) algorithm in training GMM. Especially, it was recently shown in [10] that a reasonably good dictionary can be created by some simple methods, e.g., random sampling in a training set. Moreover, our empirical evaluations show the approximations based on sparse coding improves the recognition accuracy. We then apply average pooling to aggregate the coefficient-weighted difference vectors for each visual word:

$$\boldsymbol{u}_k = \frac{1}{n} \sum_{i=1}^{n} \alpha_i^k (\boldsymbol{x}_i - \mathbf{d}_k) . \tag{3.16}$$

The final vector representation $\boldsymbol{U}$ of SDV is the concatenation of $\boldsymbol{u}_k$ from K visual words and is therefore with the dimensionality of mK:

$$\boldsymbol{U} = \left[\boldsymbol{u}_1^T \dots \boldsymbol{u}_K^T \right]^T . \tag{3.17}$$

SDV has several remarkable properties: (1) the relative displacements of descriptors to visual words retain more information lost in the traditional coding process; (2) we can compute SDV upon a much smaller dictionary which reduces the computational cost; (3) it performs quite well with simple linear classifiers which are efficient in terms of both training and testing.

3.2.3 Computing Super Location Vector

The descriptors quantized to the same visual word exhibit characteristic spatio-temporal layout. In order to capture this correlation between motion, appearance, and location, we associate space-time locations to the visual descriptor words that corresponding descriptors are assigned to. We also employ sparse coding to learn a visual location dictionary to code the location set associated with each visual descriptor word, as illustrated in Figure 3.1(c). The coefficient-weighted differences between locations and visual location words are aggregated as the spatio-temporal representation.

To describe the contribution of the parameters to the generation process of $\boldsymbol{l}_i$, we take the gradient of the log-likelihood of Eq. (3.14) with respect to the mean:

$$\frac{\partial \ln p\left(\boldsymbol{l}_i | w=k\right)}{\partial \boldsymbol{\mu}_{k_g}} = \rho_i^{k_g} \boldsymbol{\sigma}_{k_g}^{-1} \left(\boldsymbol{l}_i - \boldsymbol{\mu}_{k_g}\right), \tag{3.18}$$

where $\rho_i^{k_g}$ denotes the posterior $p\left(t=g|\boldsymbol{l}_i, w=k\right)$ and t is the assignment index. We can interpret $\rho_i^{k_g}$ as a spatio-temporal soft assignment of a descriptor location $\boldsymbol{l}_i$ associated with the kth visual descriptor word to the gth Gaussian component in the location mixture model.

If we enforce the three similar approximations as in Section 3.2.2, Eq. (3.18) can be simplified to $\alpha_i^{k_g}\left(\boldsymbol{l}_i - \mathbf{d}_{k_g}\right)$, where $\alpha_i^{k_g}$ is the sparse coding coefficient of the ith location $\boldsymbol{l}_i$ to the gth visual location word $\mathbf{d}_{k_g}$ associated with the kth visual descriptor word $\mathbf{d}_k$. As illustrated in Figure 3.1(b), let $\boldsymbol{\mathcal{L}}_k$ indicate the set of locations that are associated to the kth visual descriptor word according to the positive assignments of their descriptors, i.e., $\boldsymbol{\mathcal{L}}_k = \left\{\boldsymbol{l}_i | \alpha_i^k > 0\right\}$. We then employ the average pooling to aggregate the coefficient-weighted difference vectors for each visual location word:

$$\boldsymbol{v}_{k_g} = \frac{1}{|\boldsymbol{\mathcal{L}}_k|} \sum_{\boldsymbol{l}_i \in \boldsymbol{\mathcal{L}}_k} \alpha_i^{k_g} \left(\boldsymbol{l}_i - \mathbf{d}_{k_g}\right). \tag{3.19}$$

The concatenation of $\boldsymbol{v}_{k_g}$ from G visual location words associated with K visual descriptor words forms the final representation $\boldsymbol{V}$ of SLV:

$$\boldsymbol{V} = \left[\boldsymbol{v}_{1_1}^T \ldots \boldsymbol{v}_{1_G}^T \ldots \boldsymbol{v}_{K_1}^T \ldots \boldsymbol{v}_{K_G}^T\right]^T. \tag{3.20}$$

SLV shares the same remarkable properties as SDV. Moreover, SLV can be computed on the much smaller visual descriptor dictionary (e.g., $K = 100$) and visual location dictionary (e.g., $G = 5$). If we combine SDV and SLV, the resulting vector is of $(m + 3G)K$ dimensions, where the descriptor dimensionality m (e.g., 162 in STIP [38]) is normally much larger than $3G$. So another major benefit is that, as opposed to STP, SLV only slightly increases feature dimensions thus makes the learning and predicting more efficient.

We adopt the two normalization schemes introduced in [53] on SDV and SLV, i.e., signed square rooting and ℓ_2 normalization. As illustrated in Figure 3.1, each visual word in (a) is in the end characterized by two parts, i.e., $\boldsymbol{u}_k$ in (d) and $[\boldsymbol{v}_{k_1} \dots \boldsymbol{v}_{k_G}]$ in (e). They are used to model the motion (appearance) and location cues, respectively. We summarize the outline of computing SSCV of an action sequence in Algorithm 2.

Algorithm 2: Computation of SSCV

Input: a video sequence $\mathcal{V}$
a visual descriptor dictionary $\mathbf{D}^x = \{\mathbf{d}_k\}$
a visual location dictionary $\mathbf{D}^l = \{\mathbf{d}_{k_g}\}$
Output: SSCV $\boldsymbol{Z}$

compute spatio-temporal features $\boldsymbol{X} = \{\boldsymbol{x}_i\}$ and $\boldsymbol{\mathcal{L}} = \{\boldsymbol{l}_i\}$ from $\mathcal{V}$
compute coefficients $\{\alpha_i^k\}$ of $\boldsymbol{X}$ on $\{\mathbf{d}_k\}_{k=1}^K$ by sparse coding
for visual descriptor word $k = 1$ **to** K **do**
  $\boldsymbol{u}_k :=$ average pooling $\alpha_i^k (\boldsymbol{x}_i - \mathbf{d}_k)$, $\boldsymbol{x}_i \in \boldsymbol{X}$
  associate locations to the kth visual descriptor word: $\boldsymbol{\mathcal{L}}_k = \{\boldsymbol{l}_i | \alpha_i^k > 0\}$
  compute coefficients $\left\{\alpha_i^{k_g}\right\}$ of $\boldsymbol{\mathcal{L}}_k$ on $\left\{\mathbf{d}_{k_g}\right\}_{g=1}^G$ by sparse coding
  for visual location word $g = 1$ **to** G **do**
    $\boldsymbol{v}_{k_g} :=$ average pooling $\alpha_i^{k_g} \left(\boldsymbol{l}_i - \mathbf{d}_{k_g}\right)$, $\boldsymbol{l}_i \in \boldsymbol{\mathcal{L}}_k$
  end
  $\boldsymbol{Z}_k := \left[\boldsymbol{u}_k^T, \boldsymbol{v}_{k_1}^T \dots \boldsymbol{v}_{k_G}^T\right]^T$
end
$\boldsymbol{Z} := \left[\boldsymbol{Z}_1^T \dots \boldsymbol{Z}_K^T\right]^T$
signed square rooting and ℓ_2 normalization

3.3 Experiments and Discussions

In this section, we extensively evaluate the proposed method on the two public benchmark datasets: HMDB51 [34] and YouTube [43], as shown in Figure 3.2 and

Figure 3.2: Frames of the sampled categories from the HMDB51 dataset [34].

Figure 3.3, respectively. In all experiments, we employ LIBLINEAR [14] as the linear SVM solver. Experimental results show that our algorithm significantly outperforms the state-of-the-art methods. Our source code for computing SSCV is available online.[1]

3.3.1 Experimental Setup

Datasets The HMDB51 dataset (Figure 3.2) [34] is collected from a wide range of sources from digitized movies to online videos. It contains 51 action categories and 6,766 video sequences in total. This dataset includes the original videos and the stabilized version. Our evaluations are based on the original ones. We follow the same experimental setting as [34] using three training/testing splits. There are 70 videos for training and 30 videos for testing in each class. The average accuracy over the three splits is reported as the performance measurement. The YouTube

[1] http://yangxd.org/code

Figure 3.3: Frames of the sampled categories from the YouTube dataset [43].

dataset (Figure 3.3) [43] contains 11 action classes collected under large variations in scale, viewpoint, illumination, camera motion, and cluttered background. This dataset contains 1,168 video sequences in total. We follow the evaluation protocol as in [43] by using the leave-one-out cross validation for a pre-defined set of 25 groups. We report the average accuracy over all classes as the performance measurement.

Low-Level Feature Extraction We evaluate our approach on five low-level visual contents using appearance and motion features. STIP is used to detect sparse interest points and compute HOG/HOF as the descriptor [38]. Motivated by the success of dense sampling in image classification and action recognition, we also employ the dense trajectories [71] to densely sample and track interest points from several spatial scales. Each tracked interest point generates four descriptors: HOG, HOF, trajectory (TRA), and motion boundary histogram (MBH). HOG focuses on static appearance cues, whereas HOF captures local motion information.

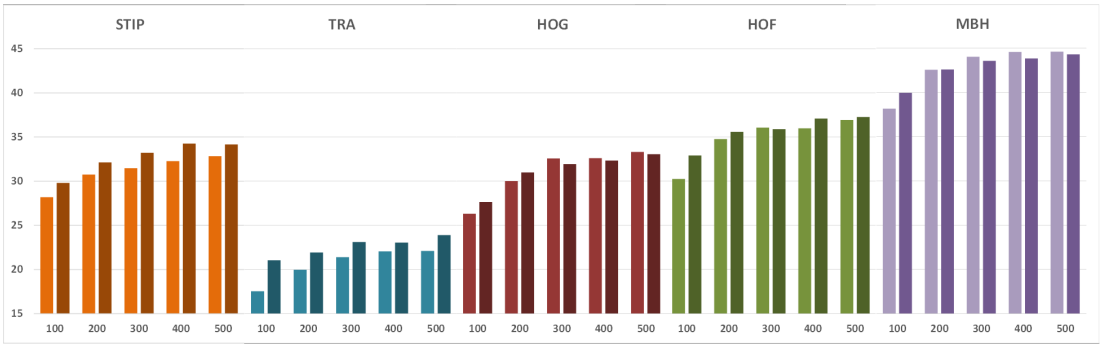


Figure 3.4: Recognition accuracy (%) of FV and SDV using different descriptors with a variety of visual dictionary size K on the HMDB51 dataset. The bars in light color and dark color denote the results of FV and SDV, respectively. This figure is better viewed on screen.

TRA characterizes the geometric shape of a trajectory. MBH computes gradient orientation histograms from horizontal and vertical spatial derivatives of optical flow. It has been proven effective to represent motion information and suppress camera motion. So for each action sequence, we compute five features: STIP (162), HOG (96), HOF (108), TRA (30), and MBH (192), where the number in parentheses denotes the descriptor dimensionality.

3.3.2 Evaluation of Feature Aggregation Schemes

In this section, we compare and analyze the performance of a variety of feature aggregation schemes. We focus on the HMDB51 dataset for a detailed evaluation of the coding and pooling parameters. Note: the spatio-temporal information is discarded in the experiments of this section.

The baseline aggregation method is the hard assignment (Hard) paired with average pooling in Eq. (3.4-3.5). The local soft assignment (LocalSoft) and max pooling in Eq. (3.6-3.8) are employed with $\mathcal{K} = 10$ nearest neighbors and $\beta = 1$. We also adopt the sparse coding (SC) with max pooling in Eq. (3.8-3.9) and set the regularizer $\lambda = 1.2/\sqrt{m}$ as suggested in [46]. As a successful feature aggregation scheme, Fisher vector (FV) in Eq. (3.10-3.11) is compared as well. Before computing FV, we follow the preprocess in [53] to apply PCA to project the descriptors to half dimensions. This step is mainly used to decorrelate the data and make it better fit the diagonal covariance matrix assumption in GMM, and meanwhile reduce computational complexity.

SDV is compared to other feature aggregation schemes in Table 3.1. We set the visual dictionary size $K = 4,000$ for Hard, LocalSoft, SC, and $K = 500$ for FV and SDV. As shown in this table, LocalSoft consistently outperforms Hard due

Table 3.1: Recognition accuracy (%) of different aggregation schemes using a variety of descriptors on the HMDB51 dataset.

	Hard	LocalSoft	SC	FV	SDV
STIP	19.2	24.5	28.6	32.8	**34.2**
TRA	17.3	18.7	21.9	22.1	**23.9**
HOG	21.0	25.3	31.5	**33.3**	**33.1**
HOF	22.0	25.8	34.5	36.9	**37.3**
MBH	31.1	32.6	36.1	**44.6**	**44.3**

to the enhanced membership estimation of descriptors to visual words. While still inferior to our method, SC largely improves the accuracy over Hard and LocalSoft by introducing the sparsity in coding descriptors. SDV outperforms FV in STIP, TRA, and HOF, and yields comparable results to FV in HOG and MBH. We further conduct a more detailed evaluation of FV and SDV as shown in Figure 3.4. SDV systematically outperforms FV in STIP and TRA, irrespective of the visual dictionary size. For HOG, HOF, and MBH, SDV achieves higher recognition accuracy than FV in a relatively small size. SDV and FV tend to have comparable results as the visual dictionary size increases. In addition to the superior recognition accuracy, SDV is computationally more efficient. This is because more information is stored per visual word, which enables SDV to perform quite well by using a much more compact visual dictionary. We use $K = 500$ to compute SDV in the following experiments if not specified.

3.3.3 Evaluation of Spatio-Temporal Models

Here we evaluate different approaches on modeling the spatio-temporal information and report results for the HMDB51 dataset.

STIP is first used to investigate the impact of the size of visual location dictionary on SLV. As shown in Figure 3.5, the results of SLV ranges from 22.4% to 25.0% as G increases from 5 to 40. The performance of SDV is plotted as a reference. When SDV and SLV are combined to SSCV, it is not very sensitive to the size and achieves the best result using only 5 visual location words. In the following experiments, we use $G = 5$ to compute SLV. Figure 3.6 demonstrates the results of SLV, SDV, and SSCV for a variety of features. SSCV consistently and significantly outperforms SDV for all features. This shows SLV is effective to model and provide the complementary spatio-temporal information to the motion and appearance cues in SDV. It is interesting to observe that SLV based on the pure space-time information even outperforms SDV for the feature TRA.

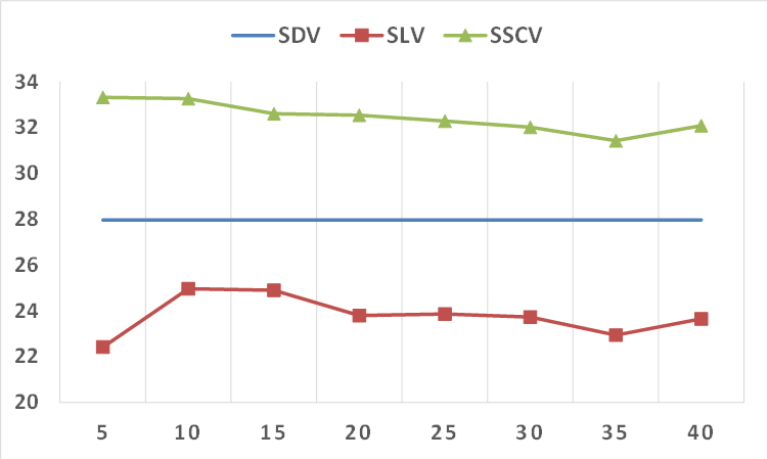


Figure 3.5: Recognition accuracy (%) of SLV and SSCV using STIP with a variety of visual location dictionary size G on the HMDB51 dataset

SSCV is then compared to the widely used spatio-temporal pyramid (STP) on modeling the space-time information. We use in our experiments four different spatio-temporal grids. For the spatial domain we employ a 1×1 whole spatial block and a 2×2 spatial grid. For the temporal domain we apply the entire sequence and two temporal segments. The combination of these subdivisions in both spatial and temporal domains generates 15 space-time cells in total. We compute a separate SDV from each cell and concatenate them as the final representation of STP. As shown in Table 3.2, both STP and SSCV improve the results because of the spatio-temporal cues complemented to SDV. However, for all features SSCV achieves more significant improvements than STP, while with much more compact representation. In our experimental setting, the dimensions of STP and SSCV are $15mK$ and $(15+m)K$, where m is the descriptor dimensionality. So in comparison to STP, our approach can also considerably reduce the computation and memory costs in both training and testing.

3.3.4 Comparison to State-of-the-Art Results

In this section, we compare our performance to the state-of-the-art results on the two benchmark datasets: HMDB51 and YouTube. SSCV is first compared to the results in [71] for each individual feature as demonstrated in Table 3.3, SSCV significantly outperforms the approach in [71], though both methods are based upon the same features. This is mainly because SDV is more representative than BOV to capture the motion and appearance information, and SLV is more effective than STP to model the spatio-temporal cues. Moreover, SSCV employs the linear SVM which is more efficient than the non-linear SVM with χ^2 kernel

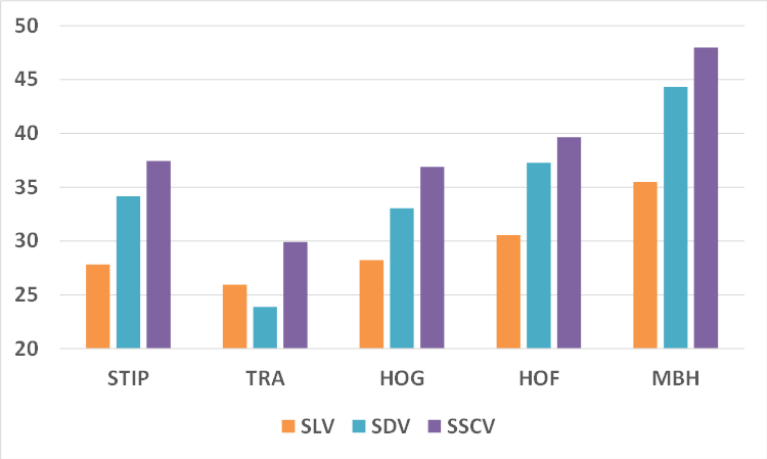


Figure 3.6: Recognition accuracy (%) of SLV, SDV, and SSCV using a variety of low-level features on the HMDB51 dataset.

used in [71]. We combine all the features and compare with the most recent results in the literature as displayed in Table 3.4. We can observe that SSCV outperforms the state-of-the-art results on the two datasets.

Table 3.2: Recognition accuracy (%) of STP and SSCV on modeling the spatio-temporal information for a variety of features on the HMDB51 dataset.

	STIP	TRA	HOG	HOF	MBH
SDV	34.2	23.9	33.1	37.3	44.3
STP	35.4 (+1.2)	28.8 (+4.9)	34.4 (+1.3)	38.1 (+0.8)	46.9 (+2.6)
SSCV	**37.4** (+3.2)	**29.9** (+6.0)	**36.9** (+3.8)	**39.7** (+2.4)	**48.0** (+3.7)

3.4 Summary

In this chapter, we have presented a novel framework for human activity recognition in conventional color videos. An effective coding scheme SDV is proposed to capture motion and appearance cues by sparse coding low-level descriptors and average pooling coefficient-weighted difference vectors between descriptors and visual words. A novel approach SLV is introduced to incorporate the spatio-temporal cues in a compact and discriminative manner. The combination of SDV and SLV constitutes the final representation of SSCV which jointly models the motion, appearance, and location information in a unified way. Our approach is

extensively evaluated on two public benchmark datasets and compared to a number of most recent results. Experimental results demonstrate that our approach significantly outperforms the state-of-the-art methods.

Table 3.3: Comparison of SSCV and the state-of-the-art method for each individual feature on the HMDB51 and YouTube datasets.

HMDB51	STIP	TRA	HOG	HOF	MBH
WKSL'13 [71]	-	28.0	27.9	31.5	43.2
SSCV	37.4	**29.9**	**36.9**	**39.7**	**48.0**

YouTube	STIP	TRA	HOG	HOF	MBH
WKSL'13 [71]	69.2	67.5	72.6	70.0	80.6
SSCV	**77.4**	**70.9**	**80.4**	**77.0**	**83.2**

Table 3.4: Comparison of SSCV to the state-of-the-art results as reported in the cited publications on the HMDB51 and YouTube datasets.

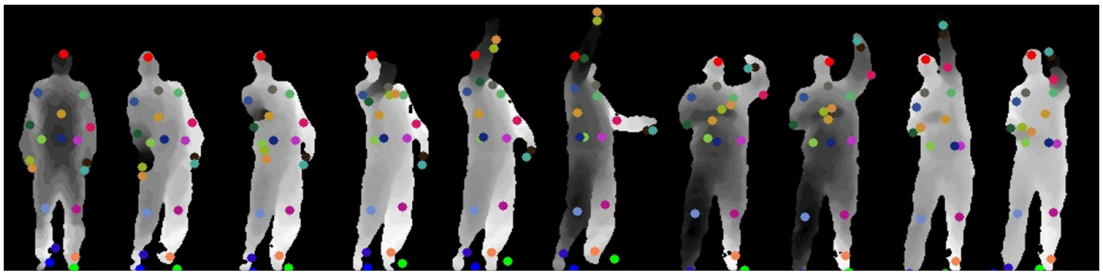
HMDB51		%	YouTube		%
GGHW'12	[21]	29.2	ICS'10	[50]	75.2
WWQ'12	[77]	31.8	LZYN'11	[40]	75.8
JDXLN'12	[30]	40.7	BSJS'11	[3]	76.5
WKSL'13	[71]	48.3	BT'10	[7]	77.8
PQPQ'13	[52]	49.2	WKSL'13	[71]	85.4
JJB'13	[27]	52.1	PQPQ'13	[52]	86.6
SSCV		**53.9**	SSCV		**88.0**

Chapter 4

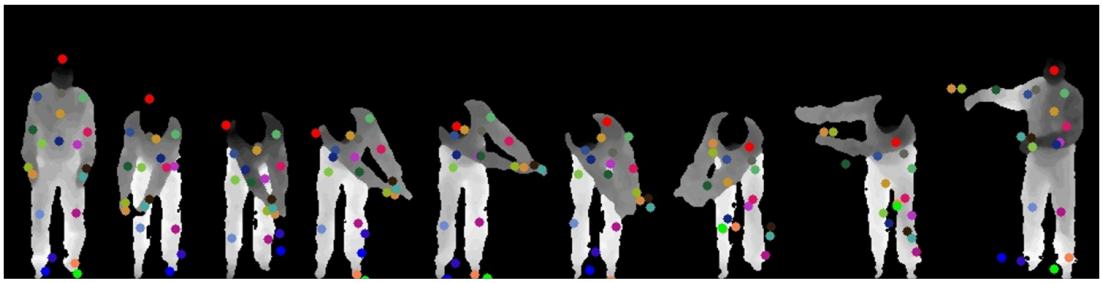
Effective 3D Activity Recognition Using EigenJoints

In this chapter, we propose an effective method to recognize human activities using skeleton joints recovered from depth sequence captured by RGB-D cameras. We design a novel feature representation for activity recognition based on differences of skeleton joints, i.e., EigenJoints, which combines the information of static posture, motion property, and overall dynamics. The accumulated motion energy (AME) is then proposed to perform informative frame selection. This is able to preclude noisy frames and reduce computational complexity. We employ the non-parametric Naive-Bayes-Nearest-Neighbor (NBNN) to classify multiple activity categories. The experimental results on several challenging datasets demonstrate that our approach outperforms the previous methods. In addition, we investigate how many frames are necessary for our method to perform classification in the scenario of online activity recognition. We observe that the first 30% frames are sufficient to achieve comparable results to that using the entire video sequence.

Most conventional research on human activity recognition mainly concentrates on the video sequences captured by traditional RGB cameras [37] [71] [85]. In this case, a video is a sequence of 2D frames of RGB images arranged in chronological order. There has been extensive research in the literature on activity recognition for such videos. The spatio-temporal volume-based methods have been widely used to measure the similarity between subsequence volumes of videos. In order to enable an accurate similarity measurement, a variety of spatio-temporal volume detection and representation methods have been proposed [12] [32] [37] [67] [71]. On the other hand, the trajectory-based approaches have been explored for recognizing human activities as well [9] [63]. In these approaches, human activities can be interpreted by a set of key joints or detected interest points. However, it is not trivial to quickly and reliably extract and track skeleton joints from traditional RGB videos. Along with the advance of the imaging techniques, such as



(a) Tennis Serve



(b) Golf Swing

Figure 4.1: Sampled sequences of depth maps and skeleton joints in actions of (a) tennis serve and (b) golf swing. Each depth map includes 20 joints. The joints of each body part are encoded in corresponding colors.

the emergence of RGB-D cameras, e.g., Microsoft Kinect and ASUS Xtion Pro Live, it has become practical to capture color frames as well as depth maps in real time. Depth maps are able to provide additional body shape information to differentiate the actions with similar 2D projections from a single view. It has therefore motivated recent research work to investigate activity recognition using the depth information.

The biological observation [31] suggests that human actions could be modeled by the motion of a set of skeleton joints. The MoCap system [24] is employed to extract 3D joint positions by using markers and high precision camera array. With the emergence of RGB-D cameras, we are able to recover 3D positions of skeleton joints in real time and with reasonable accuracy [60]. In this chapter, we focus on recognizing human activities using skeleton joints extracted from sequence of depth maps. Figure 4.1 demonstrates the depth sequences with 20 extracted skeleton joints from each depth map in the actions of *tennis serve* and *golf swing*. As illustrated in this figure, the perception of each action can be reflected by the motions of individual joints (i.e., motion property) and the configuration of different joints (i.e., static postures). Compared to the cloud points of human body in depth maps, the skeleton joint representation is more compact.

In this chapter, we propose a novel feature representation by adopting the differences of skeleton joints in both temporal and spatial domains to explicitly model the dynamics of each individual joint and the configuration of a group of joints. We then apply principal component analysis (PCA) to the joint differences to obtain the EigenJoints to reduce noise and redundancy. Similar to the affect recognition in [22], the temporal segments of an action can be intuitively approximated by the status of neutral, onset, apex, and offset. The discriminative information is however not evenly distributed in these status. We propose a measurement of accumulated motion energy (AME) to quantize the distinctiveness of each frame. The less distinctive frames are then pruned to remove noise and reduce computational complexity. We employ the non-parametric Naive-Bayes-Nearest-Neighbor (NBNN) [5] as the classifier to recognize multiple action categories. In accordance with the principles behind NBNN based image classification, we avoid quantization of descriptor and compute the video-to-class distance, instead of the video-to-video distance. In addition, most existing methods perform activity recognition by operating on the entire video sequence. However, this is not practical to the online systems which require as few frames as possible for recognition. We therefore investigate how many frames are sufficient to obtain reasonably accurate action recognition in our framework. Experimental results on the MSRAction3D dataset [42] demonstrate that a short sub-sequence (e.g., the first 30% frames) of the entire video is sufficient to perform activity recognition, with quite limited gains as more frames added in. This observation is important for making online decisions and reducing latency when humans interact with machines.

The remainder of this chapter is organized as follows. Section 2.1 reviews the existing methods for human activity recognition. In Section 2.2, we provide detailed procedures to compute EigenJoints from each depth map. Section 2.3 briefly introduces NBNN classifier. Section 2.4 describes the informative frame selection by using accumulated motion energy (AME). A variety of experimental results and discussions are presented in Section 2.5. Finally, Section 2.6 summarizes the remarks of this chapter.

4.1 Related Work

In traditional RGB videos, human activity recognition mainly focuses on analyzing spatio-temporal volumes. The core of these approaches is to detect and represent space-time volumes. Bobick and Davis [4] stacked foreground regions of a person to explicitly track shape changes. The stacked silhouettes formed motion history images (MHI) and motion energy images (MEI), which served as an action descriptor for template matching. In most recent work, local spatio-

temporal features are widely used. Similar to object recognition using sparse local features in 2D images, an activity recognition system first detects interest points (e.g., STIP [37]) and then computes descriptors (e.g., HOG/HOF [38]) based on the detected local spatio-temporal volumes. These local features are then aggregated (e.g., bag-of-visual-words) to represent activities. The trajectory based approaches are more similar to our method which models activities by the motion and configuration of a set of points extracted from human body. Sun et al. [63] extracted trajectories through pair-wise SIFT matching between neighboring frames. The stationary distribution of a Markov chain model was then used to compute a velocity description.

The availability of depth sensors has recently made it possible to capture depth maps in real time. This facilitates a variety of visual recognition tasks, such as human pose estimation and activity recognition. Shotton et al. [60] proposed an object recognition method to predict 3D positions of body joints from a single depth image. This scheme was further extended in [19] and [64] by aggregating votes from a regression forest and incorporating dependency relationships between body part locations, respectively. Li et al. [42] proposed a Bag-of-3D-Points model for activity recognition. They sampled 3D representative points from the contours of depth maps of a body surface projected onto three orthogonal Cartesian planes. An action graph was then used to model the sampled 3D points for recognition. Their experimental results validated the superiority of 3D silhouettes over 2D silhouettes from a single view. However, the sampling of 3D points incurred a great amount of data which resulted in expensive computations in clustering training samples from all classes. In [81] Xia et al. used histogram of 3D joint locations (HOJ3D) to represent posture. They transferred skeleton joints into a spherical coordinate to achieve view-invariance. The temporal information was then encoded by a discrete hidden Markov model (HMM). Sung et al. [65] employed both visual and depth channels to recognize human daily activities. The skeleton joints were used to model body pose, hand position, and motion information. They also extracted histogram of oriented gradients (HOG) features from regions of interest in gray images and depth maps to characterize the appearance cues. A hierarchical maximum entropy Markov model (MEMM) was used to decompose an activity to a set of sub-activities.

Most of the above systems hinge on the entire video sequence to perform action recognition. In the online scenario, a system is however supposed to require as few observations as possible. Schindler and Gool [58] first investigated how many frames were required to enable action classification in RGB videos. They found that short action snippets with a few frames (e.g., 1 to 7 frames) were almost as informative as the whole video. In order to reduce the observational latency, i.e., the time a system takes to observe sufficient information for a good classification, Ellis et al. [13] proposed to recognize actions based upon an individual canonical

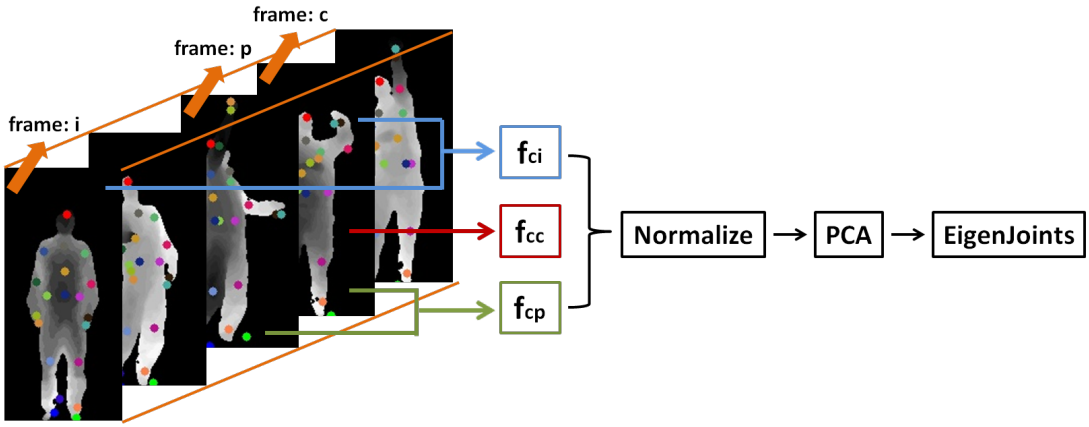


Figure 4.2: The framework of representing EigenJoints. In each frame, we compute three features of f_{ci}, f_{cc}, and f_{cp} to capture the information of offset, posture, and motion. The normalization and PCA are then applied to obtain the descriptor of EigenJoints for each frame.

pose from a sequence of postures. The canonical pose covered the information of posture, motion, and overall variance by using skeleton joints. They used a classifier based on logistic regression to minimize the observational latency.

Motivated by the robust extraction of skeleton joints using RGB-D cameras, we propose a novel feature representation, EigenJoints, for activity recognition. In contrast to the traditional trajectory-based methods, EigenJoints are able to model actions through more informative and more accurate body joints without background points. Compared to the previous features using skeleton joints or depth maps, EigenJoints are more discriminative, more compact, and easier to compute.

4.2 Representation of EigenJoints

The proposed framework to compute EigenJoints is demonstrated in Figure 4.2. We employ the differences of 3D positions of skeleton joints to capture the information of static posture feature f_{cc}, consecutive motion feature f_{cp}, and overall dynamics feature f_{ci} in each frame-c. We then concatenate the three feature channels as $f_c = [f_{cc}, f_{cp}, f_{ci}]$. According to different experimental settings (e.g., cross-subject test or non-cross-subject test), two normalization schemes are introduced to obtain f_{norm}. In the end, PCA is applied to f_{norm} to generate EigenJoints.

As shown in Figure 4.2, the 3D coordinates of n joints can be obtained from the human pose estimation [60] in each frame: $X = \{x_1, \ldots, x_n\}$, $X \in \mathbb{R}^{3\times n}$. To characterize the static posture information of the current frame-c, we compute

the pair-wise joint differences within the current frame:

$$f_{cc} = \{x_i - x_j \mid i, j = 1, \ldots, n; i \neq j\}. \tag{4.1}$$

To capture the motion property of current frame-c, the joint differences are computed between the current frame-c and its preceding frame-p:

$$f_{cp} = \left\{x_i^c - x_j^p \mid x_i^c \in X_c; x_j^p \in X_p\right\}. \tag{4.2}$$

To represent the offset feature or the overall dynamics of the current frame-c with respect to the initial frame-i, we calculate the joint differences between frame-c and frame-i:

$$f_{ci} = \left\{x_i^c - x_j^i \mid x_i^c \in X_c; x_j^i \in X_i\right\}. \tag{4.3}$$

The initial frame tends to approximate the neutral posture. Combination of the three feature channels forms the preliminary feature representation for each frame: $f_c = [f_{cc}, f_{cp}, f_{ci}]$.

In Eq. (4.1-4.3) the orders of joints are in accordance to the specified joint index. However, the three elements (u, v, d) of a joint x might be of inconsistent coordinates, e.g., (u, v) are in the screen coordinate and d is in the world coordinate. So normalization is applied to f_c to avoid those elements in a greater numeric range dominating the ones in a smaller numeric range. We use a linear normalization scheme to scale each dimension in f_c to the range $[-1, +1]$. The other benefit of normalization is to reduce the intra-class variations from diverse people. In our experiments, we normalize f_c based on a single video sequence for the cross-subject test and based on whole training video sequences for the non-cross-subject test.

As illustrated in Figure 4.1, in each frame we use n joints which could result in a big feature dimension. f_{cc}, f_{cp}, and f_{ci} contain $n(n-1)/2$, n^2, and n^2 pair-wise comparisons, respectively. Each comparison generates 3 elements $(\Delta u, \Delta v, \Delta d)$. In the end, f_{norm} is with the dimension of $3 \times (n(n-1)/2 + n^2 + n^2)$. For example, if 20 skeleton joints are extracted in each frame, f_{norm} is with the dimension of 2,970. As skeleton joints are already high level information recovered from depth maps, this large dimension could be redundant and include noise, as illustrated in Figure 4.5. We therefore apply PCA to reduce the redundancy and noise in the centralized f_{norm}. The final compact feature representation is the EigenJoints, which is the descriptor of each frame. In the experiments, we observe that most eigenvalues are covered by the first few leading eigenvectors, e.g., the leading 128 eigenvalues weight over 95%.

4.3 Naive-Bayes-Nearest-Neighbor Classifier

We employ the Naive-Bayes-Nearest-Neighbor (NBNN) [5] as the classifier in activity recognition. The nearest-neighbor (NN) is a non-parametric classifier which has several advantages over most learning based classifiers: 1) naturally deals with a large number of classes; 2) avoids the overfitting problem; and 3) requires no learning process. Boiman et al. [5] argued that the effectiveness of NN was largely undervalued by the quantization of local image descriptors and the computation of image-to-image distance. Their observations showed that the frequent descriptors tend to have lower quantization error but the rare descriptors tend to have higher quantization error. However, most discriminative cues are contained in the rare descriptors. So the quantization used in bag-of-visual-words scheme degrades the discriminative power of descriptors. Moreover, the kernel matrix used by SVM computes image-to-image distance. But they observed that the distance computation of image-to-class distance which makes use of descriptor distributions over the whole class provided better generalization than the image-to-image distance.

We follow these conclusions in image classification and apply them to video classification, i.e., activity recognition. We directly utilize the frame descriptor of EigenJoints without quantization, and compute the video-to-class distance rather than video-to-video distance. In the context of NBNN, the activity recognition is performed by:

$$C^* = \arg\min_C \sum_{i=1}^{m} \|d_i - NN_C(d_i)\|^2, \tag{4.4}$$

where $d_i, i = 1, \ldots, m$ is the EigenJoints descriptor of frame-i in a testing video; m is the number of frames in this testing video; $NN_C(d_i)$ is the nearest neighbor of d_i in class-C. Experimental results show that the recognition accuracy based on NBNN outperforms that based on SVM. The approximate NN algorithm, such as k-d tree [1] or local NBNN [47] can be employed to reduce the computational complexity in the NBNN classification.

4.4 Informative Frame Selection

As in the affect recognition [22], temporal segments of an activity sequence can be intuitively approximated by the status of neutral, onset, apex, and offset. The discriminative information is not evenly distributed in these status, but concentrates more on the frames from onset and apex status. Moreover, motions of neutral and offset status are usually similar across different actions. So the informative frame selection corresponds to extracting the frames from onset and

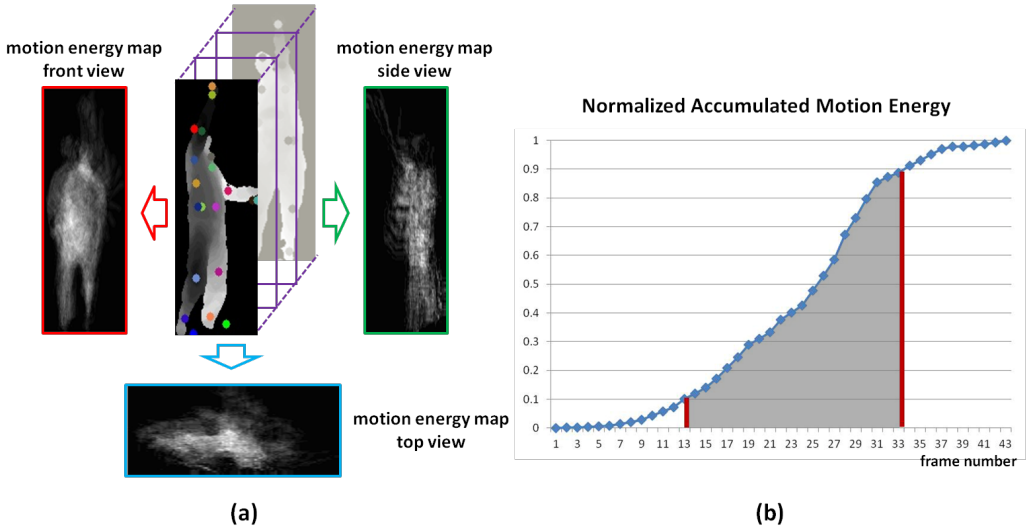


Figure 4.3: Computation of the accumulated motion energy (AME). (a) illustrates the motion energy maps associated with each projected view. (b) shows the normalized AME and selected informative frames.

apex but preclude the frames from neutral and offset. This process enables us to remove confusing frames and reduce computational cost in the nearest neighbor searching. We propose to use the accumulated motion energy (AME) to measure the distinctiveness of each frame:

$$\delta(i) = \sum_{v=1}^{3} \sum_{j=1}^{i} \left(|f_v^j - f_v^{j-1}| > \epsilon \right). \tag{4.5}$$

For frame-i, its depth map is first projected onto three orthogonal planes to generate three projected maps f_v, $v \in 1, 2, 3$. We compute $\delta(i)$ as the summation of motion energy maps. The motion energy maps of each frame are obtained by thresholding and accumulating the differences between consecutive projected maps, as shown in Figure 4.3(a). AME is then normalized by the ℓ_1-norm. Figure 4.3(b) illustrates the normalized AME of the action *tennis serve* from the MSRAction3D dataset. As shown in this figure, when the normalized AME is less than 0.1 or larger than 0.9, it increases slowly because motions in these frames are weak. It is also observed that most of these frames correspond to the status of neutral and offset. As for the frames whose normalized AME are between 0.1 and 0.9, they present significant motions and make the curve dramatically increase. Accordingly, these frames are from the status of onset and apex and cover more discriminative information. In our experiment, we therefore choose frames with the normalized AME between 0.1 and 0.9 as the informative frames.

4.5 Experiments and Discussions

We evaluate our proposed approach on three challenging datasets including MSRAction3D [42], Cornell Human Activity [65], and UCF Kinect [13]. We extensively compare with the existing methods to our approach under a variety of experimental settings.

4.5.1 Experiments on MSRAction3D Dataset

The MSRAction3D [42] is a benchmark dataset for 3D action recognition with sequences of depth maps and skeleton joints. It includes 20 action categories performed by 10 subjects facing the camera. Each subject performs each action 2 or 3 times. The depth maps are with the resolution of 320×240. For each skeleton joint, the horizontal and vertical locations are stored in the screen coordinate, and depth position is stored in the world coordinate. The 20 actions are selected in the context of gaming. As shown in Figure 4.4, the actions in this dataset capture a variety of motions related to arms, legs, torso, and their combinations.

In order to facilitate a fair comparison with the previous methods, we follow the same experimental settings as [42] [81] to split the 20 action categories into three subsets as listed in Table 4.1. In each subset, there are further three different tests: Test One (One), Test Two (Two), and Cross-Subject Test (CrSub). In Test One, one third of the subset is used as training and the rest as testing; in Test Two, two thirds of the subset is used as training and the rest as testing. Both of them are non-cross-subject tests. In cross-subject test, half of subjects are used for training and the rest ones used for testing.

Table 4.1: Three action subsets of MSRAction3D dataset used in our experiments.

Action Set 1 (AS1)	Action Set 2 (AS2)	Action Set 3 (AS3)
Horizontal Wave (HoW)	High Wave (HiW)	High Throw (HT)
Hammer (H)	Hand Catch (HC)	Forward Kick (FK)
Forward Punch (FP)	Draw X (DX)	Side Kick (SK)
High Throw (HT)	Draw Tick (DT)	Jogging (J)
Hand Clap (HC)	Draw Circle (DC)	Tennis Swing (TSw)
Bend (B)	Hands Wave (HW)	Tennis Serve (TSr)
Tennis Serve (TSr)	Forward Kick (FK)	Golf Swing (GS)
Pickup Throw (PT)	Side Boxing (SB)	Pickup Throw (PT)

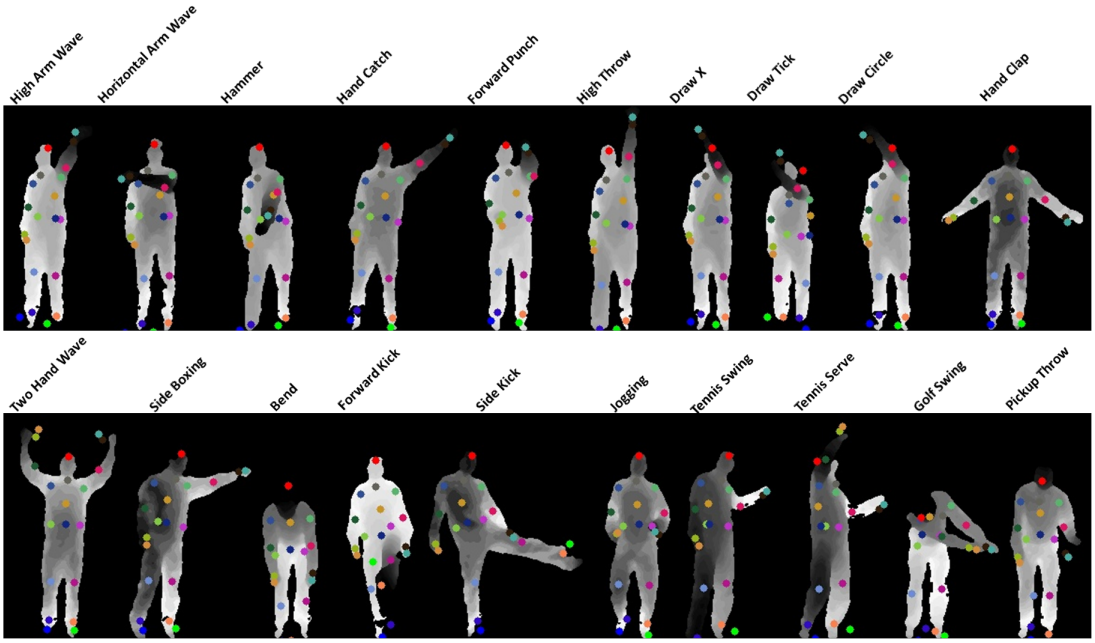


Figure 4.4: Examples of depth maps and skeleton joints associated with sampled frames in the MSRAction3D dataset.

4.5.1.1 Evaluation of EigenJoints and NBNN

We first evaluate energy distributions of joint differences to determine the dimensionality of EigenJoints. Figure 4.5(a) shows ratios between the sum of first few leading eigenvalues and the sum of all eigenvalues of f_{norm} under different test sets. As demonstrated in this figure, the first 128 eigenvalues (out of 2,970) occupy over 95% energy for all experimental settings. The distributions concentrate more in the first few leading eigenvalues for Test One and Test Two, where the first 32 eigenvalues have already weighted over 95%. The distribution scatters relatively more for cross-subject test, where the leading 32 eigenvalues cover about 85% of overall energy.

Figure 4.5(b) shows the recognition accuracies of EigenJoints based NBNN with different dimensions under various test sets. It is interesting to observe that the overall recognition rates under a variety of test sets are stable across different dimensions. For each dimensionality, our method performs well for Test One and Test Two which are non-cross-subject tests. While the performance in AS3CrSub is promising, the accuracies in AS1CrSub and AS2CrSub are relatively low. This is probably because actions in AS1 and AS2 are with similar motions, but AS3 groups complex but pretty distinct actions. For example, in AS1 *Hammer* tends to be confused with *Forward Punch*, and *Pickup Throw* consists of *Bend* and

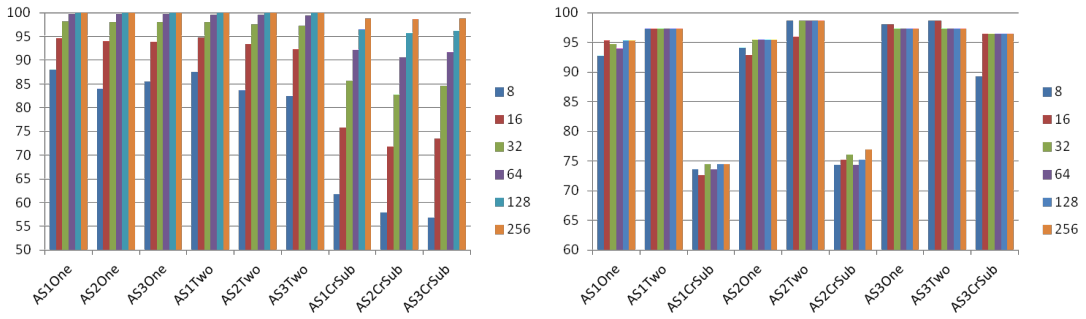


Figure 4.5: Left: ratios (%) between the sum of first few (8, 16, 32, 64, 128, 256) leading eigenvalues and the sum of all eigenvalues of f_{norm} under different test sets. Right: recognition accuracy (%) of NBNN based EigenJoints with different dimensions under various test sets.

High Throw. In cross-subject test, different subjects also perform actions with considerable variations but the number of subjects is limited. For example, some subjects perform action of *Pickup Throw* using only one hand whereas others using two hands. This results in great intra-class variations. The cross-subject performance can be improved by adding in more subjects.

Considering recognition accuracy and computational cost in NBNN classification, we choose 32 as the dimensionality for EigenJoints in all of our experiments. As high accuracies of Test One and Test Two (over 95%, see Figure 4.5), we only show the confusion matrix of our method under cross-subject test in Figure 4.6. Due to the considerable variations in the same actions performed by different subjects, cross-subject generates much larger intra-class variance than non-cross-subject. In AS1CrSub, most actions are confused with *Pickup Throw*, especially

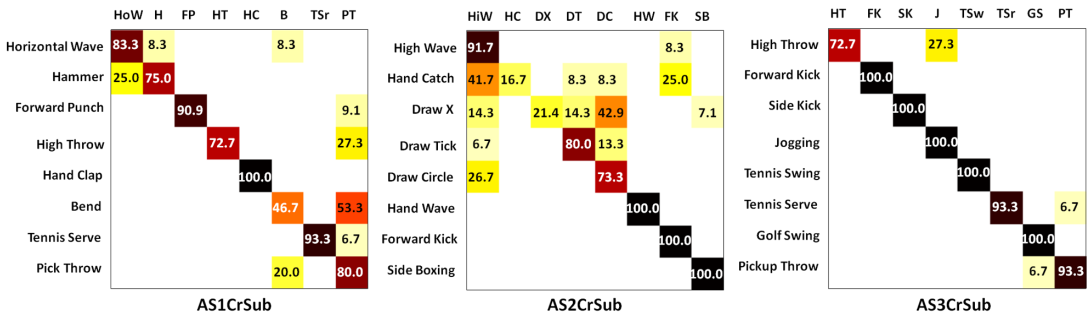


Figure 4.6: Confusion matrix of EigenJoints based NBNN in different action sets under cross-subject test. Each row corresponds to the ground truth label and each column indicates the recognition results.

Table 4.2: Comparisons of the overall recognition accuracies under three test sets.

Methods	Test One	Test Two	Cross Subject Test
Bag-of-3D-Points [42]	91.6	94.2	74.4
HOJ3D [81]	96.2	97.2	79.0
Ours	95.8	97.8	83.3

for *Bend* and *High Throw*. In AS2CrSub, *Draw X*, *Draw Tick*, and *Draw Circle* are mutually confused, as they contain highly similar motions. Although actions in AS3 are complex, they are with significant differences. So the recognition results are largely improved in AS3CrSub.

4.5.1.2 Comparisons to the State-of-the-Art Methods

SVM has been extensively used in computer vision to achieve the state-of-the-art performances in image and video classifications. We employ the bag-of-visual-words to represent a depth video by quantizing EigenJoints of each frame. K-means clustering is employed to compute the visual dictionary. We empirically choose $K = 500$ and RBF kernels to perform classification. The optimal parameters of RBF kernels are obtained by 5-fold cross-validation. Figure 4.7(a) compares the recognition accuracies based on NBNN and SVM. As shown in this figure, NBNN outperforms SVM in most testing sets. This observation also validates the superiority of the two schemes used in NBNN, i.e., non-quantization of EigenJoints and computation of video-to-class distance.

We further compare our approach with previous methods including Bag-of-3D-Points [42] and HOJ3D [81] under different testing sets in Figure 4.7(b). The overall accuracies are shown in Table 4.2. As shown in Figure 4.7(b), HOJ3D and our method significantly outperform Bag-of-3D-Points in most cases. The performance of our method is comparable to that of HOJ3D in non-cross-subject tests. However, under cross-subject tests, HOJ3D and our method behave quite differently. Our method performs much better than HOJ3D in AS3CrSub, but is inferior to HOJ3D in AS1CrSub and AS2CrSub. This is probably because AS1 and AS2 group similar actions which are more sensitive to the larger intra-class variations generated in cross-subject tests. So the leading factors computed by PCA might be biased by the large intra-class variations. But complex actions in AS3 present considerable inter-class variations which overweight intra-class variations. So the leading factors of PCA still correspond to variations of different action classes. As for the overall accuracies in Table 4.2, our method and HOJ3D achieve comparable results in Test One and Test Two. But our method signif-

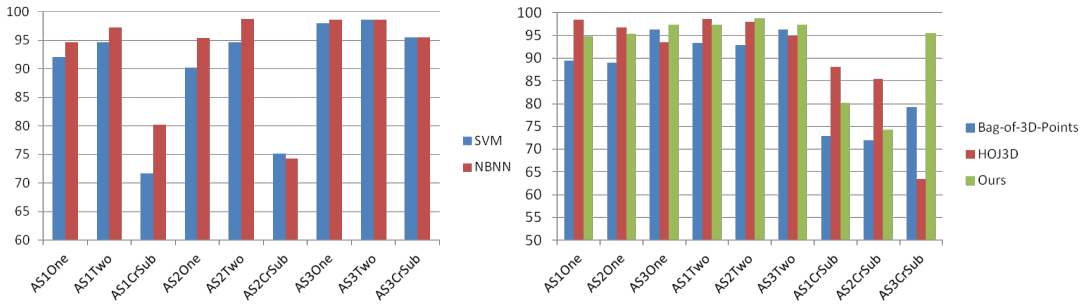


Figure 4.7: Left: comparison of the recognition accuracy (%) between SVM and NBNN based on EigenJoints. Right: recognition accuracy (%) of different methods under a variety of testing sets.

icantly outperforms HOJ3D under cross-subject test. This is more desirable in real-world applications. In addition to recognition accuracy, our method is more compact than Bag-of-3D-Points and HOJ3D.

4.5.1.3 How Many Frames Are Sufficient

Most existing methods recognize human activity using entire video sequences. We perform another experiment to investigate how many frames are sufficient to enable accurate action recognition in our framework. The recognition accuracies using different number of first few frames under a variety of test sets are illustrated in Figure 4.8. The sub-sequence is extracted from the first T frames of a given video. As shown in this figure, in most cases 15 to 20 frames, i.e., the first 30% to 40% frames are sufficient to achieve comparable recognition accuracies to the ones using the whole video sequence. There are rapid diminishing gains as more frames are added in. These results are highly relevant for activity recognition

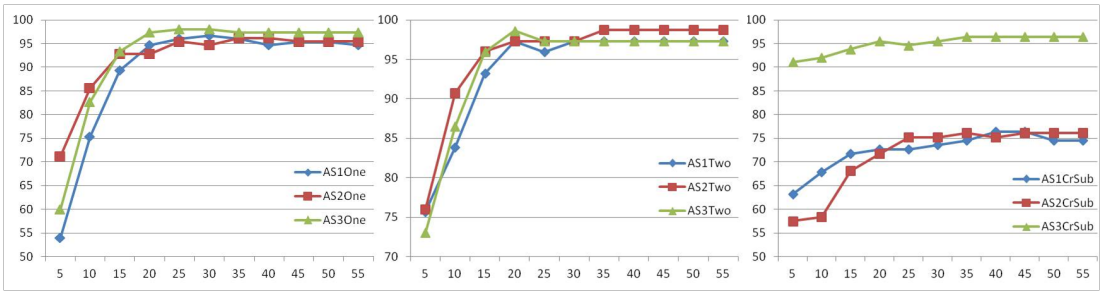


Figure 4.8: Recognition accuracy (%) with different number of first few frames in Test One (left), Test Two (middle), and Cross-Subject Test (right) on the MSRAction3D dataset.

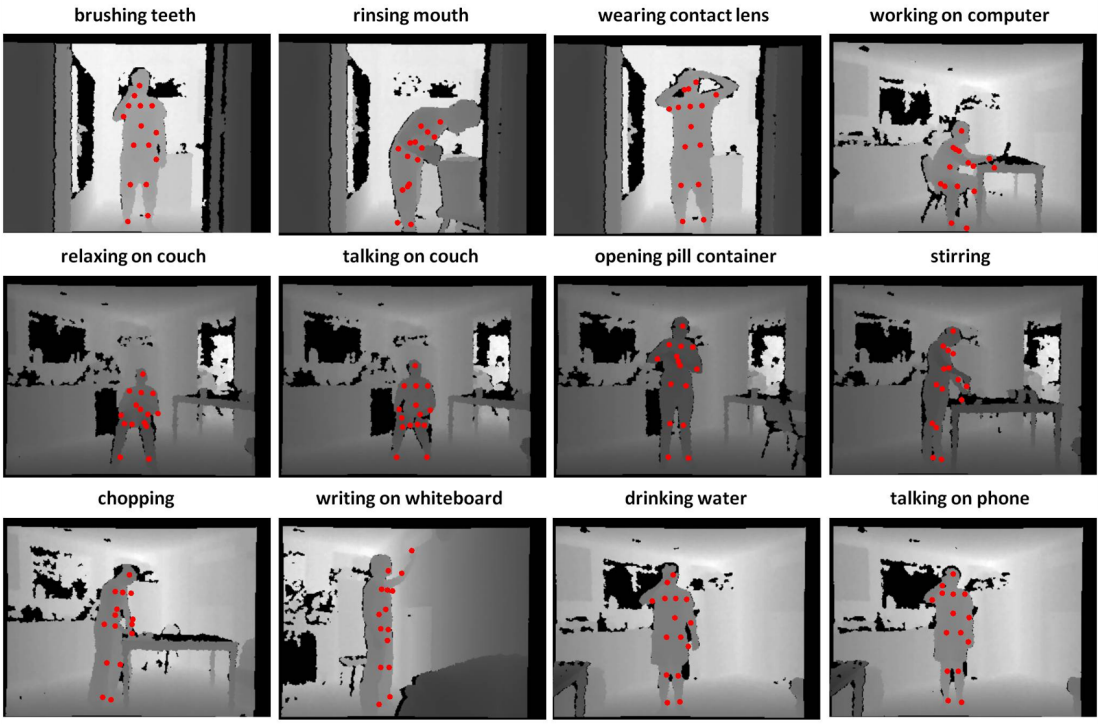


Figure 4.9: Examples of depth maps and skeleton joints associated with each frame of the 12 activities in the Cornell Human Activity dataset.

systems where decisions have to be made on line. An online system generally requires short latency that is mainly affected by two factors, i.e., 1) the time a system takes to observe sufficient frames for making a reasonable prediction and 2) the time a system takes to compute on the observations. Therefore cutting down the number of frames that a system reads in helps to reduce the costs in both of the two factors.

4.5.2 Experiments on Cornell Human Activity Dataset

The Cornell Human Activity [65] is a public dataset providing video sequences of RGB images with aligned depth maps captured by a Microsoft Kinect camera. In each frame, 15 skeleton joints in the world coordinate are available. These videos are with the resolution of 640×480 and at the frame rate of 30 Hz. This dataset includes 12 activities and 1 random action performed by 4 subjects in 5 different environments (i.e., office, kitchen, bedroom, bathroom, and living room). The 12 activities are selected in the context of human daily activities. As illustrated in Figure 4.9, activities in this dataset are captured in uncontrolled environments with quite cluttered households and involve extensive human-object interactions.

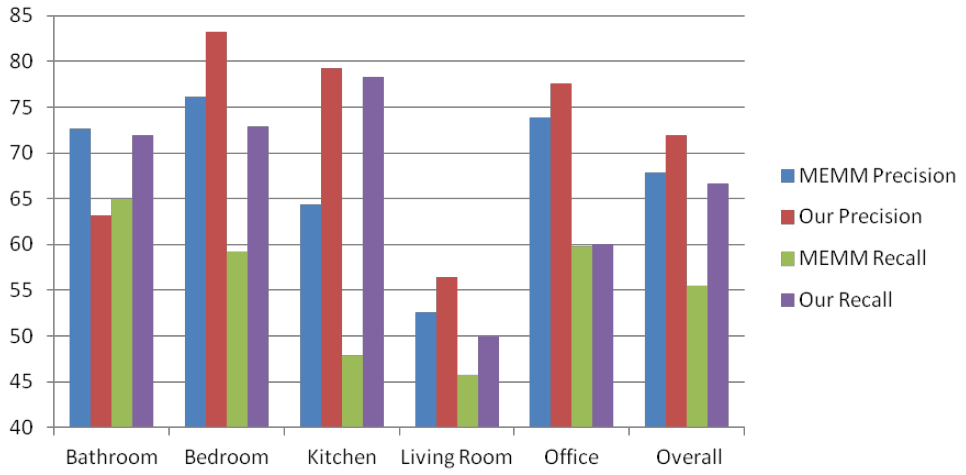


Figure 4.10: Comparison of the precision (%) and recall (%) of MEMM and our method under a variety of test sets.

Since neutral postures are removed in this dataset, we only employ f_{cc} and f_{cp} in Eq. (4.1-4.2) to compute EigenJoints. We follow the same experimental settings (subject-independent test) as [65] to split the 13 activities (with one extra background activity) into five different environments under cross-subject tests. Experimental results are reported as the average accuracy of the leave-one-out tests in Figure 4.10. The features used in hierarchical MEMM include visual frames, depth maps, and skeleton joints, which are much more complex than EigenJoints only using skeleton joints. However EigenJoints still significantly outperforms hierarchical MEMM, e.g., the overall precision and recall of our method are 71.9% and 66.6% which improves the results of hierarchical MEMM by 4.0% and 11.1%.

4.5.3 Experiments on UCF Kinect Dataset

We also evaluate our proposed method on the UCF Kinect dataset [13]. This dataset is collected by Microsoft Kinect and OpenNI. In each frame only 15 skeleton joints are available, RGB images and depth maps are not stored. It includes 16 actions performed by 16 subjects, as shown in Figure 4.11. Comparisons of recognition accuracy of our method and the latency aware learning (LAL) [13] are shown in Figure 4.11. Since depth maps are not available in this dataset, we do not perform frame selection but operate on the whole video sequences. In order to reduce observational latency, the LAL method searches a single canonical posture for recognition. But to facilitate a fair comparison, we only compare to their results based on the full video sequences. It can be seen from Figure

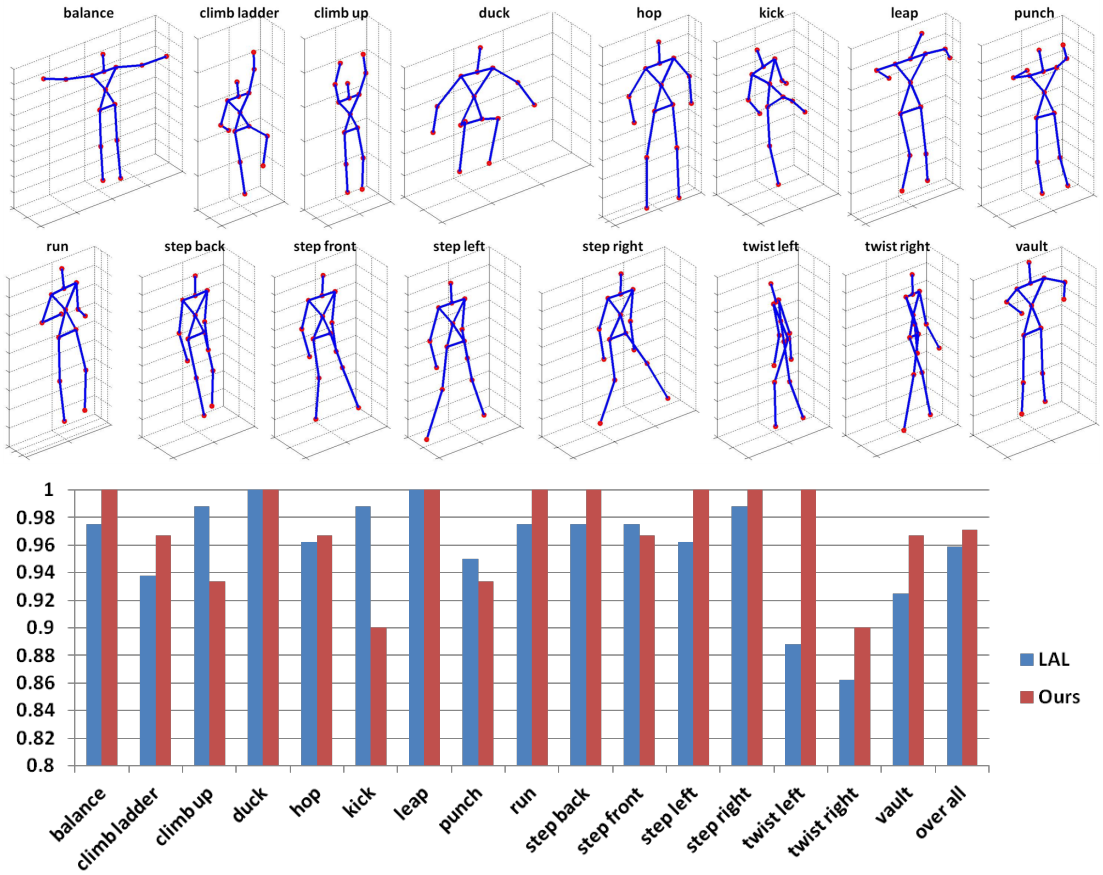


Figure 4.11: Top: the 16 actions and skeleton joints associated with each frame in the UCF Kinect dataset. Bottom: comparisons of recognition accuracy (%) of our method to LAL on this dataset.

4.11 that our method achieves better or equal accuracies in 12 out of 16 action categories. The average accuracy of all the 16 actions of our method is 97.1% which outperforms LAL by 1.2%.

4.6 Summary

In this chapter, we have presented an EigenJoints based activity recognition method using the NBNN classifier. The compact and discriminative frame representation of EigenJoints is effective to capture the properties of static posture, motion between consecutive frames, and overall dynamics with respect to the neutral status. The proposed measurement of AME quantizes the distinctiveness

of each frame. By using AME to prune less discriminative frames, we can remove noisy and ambiguous frames and reduce computational complexity. Comparisons between NBNN and SVM show that the non-quantization of descriptors and computation of video-to-class distance are more effective for activity recognition. In addition, we observe that the first 30% to 40% frames are sufficient to enable recognition with reasonably accurate results. This observation is relevant to the systems where recognition has to be made online. Experimental results on the three challenging datasets of MSRAction3D, Cornell Human Activity, and UCF Kinect demonstrate that our approach outperforms the previous methods.

Chapter 5

Depth Motion Maps based Histogram of Oriented Gradients

In this chapter, we propose an effective approach to recognize human activities in depth videos, where depth maps provide additional body shape and motion cues. In our approach, we project depth maps onto three orthogonal planes and accumulate global actions through the entire video sequence to generate the depth motion maps (DMM). We then compute histogram of oriented gradients (HOG) from DMM as the action representation of a depth video. Experimental results on the MSRAction3D dataset demonstrate that our approach outperforms the previous methods, although our representation is much more compact. In addition, we also investigate how many frames are required in this framework for activity recognition. We observe that a short sub-sequence of 30 to 35 frames on the MSRAction3D dataset is sufficient to achieve comparable recognition results to that operating on the whole video sequence.

Automatic human activity recognition has many real-world applications including content based video search, human-computer interaction, video surveillance, health care, and etc. As introduced in the preceding chapters, research of human activity recognition mainly concentrates on color video sequences captured by traditional RGB cameras. The representation methods based on spatio-temporal volumes have been extensively studied for activity recognition through measuring the similarities between local spatio-temporal volumes. In order to facilitate the accurate similarity measurements, a variety of detection and representation methods of spatio-temporal volumes have been proposed [12] [37] [38] [71]. Meanwhile, the trajectory based approaches have also been explored for recognizing human activities as well [24] [63]. In this case, human activities are interpreted by the combined motions of a set of key joints extracted from human body. However, in traditional videos it is nontrivial to quickly and reliably detect and track human body joints.

As the imaging technique advances, it has become feasible to capture color frames as well as depth maps in real time by RGB-D sensors. The depth maps are able to provide additional body shape and motion information to distinguish actions with similar projections from a single view. This greatly motivates recent research to explore activity recognition based on depth maps [42] [86]. In this chapter, we also focus on recognizing human activities using sequence of depth maps. We propose an effective and efficient approach to recognize human activities by extracting histograms of oriented gradients (HOG) from depth motion maps (DMM). DMM is generated by stacking motion energy of depth maps projected onto three orthogonal planes. The stacked motion energy of each action category produces a specific appearance and shape on DMM. This can be used to characterize the corresponding action categories. Motivated by the success of HOG in human detection [11], we adopt HOG descriptors to represent DMM. Compared to original depth data, the proposed DMM-HOG representation is more compact and more discriminative. Similar to the experiment in Chapter 4, we also investigate how many frames are sufficient to perform action recognition using DMM-HOG. Experimental results demonstrate that a short sub-sequence (e.g., 35 frames) is sufficient to obtain reasonably accurate recognition results. This result is also in consistence to the observation in Chapter 4. It provides important reference for the online systems to reduce the observational latency.

5.1 Related Work

The global representations of actions have been widely used in traditional videos captured by RGB cameras. For example, Bobick and Davis [4] accumulated the foreground motion region as the motion history image (MHI) to explicitly model the motion change. Tian et al. [67] employed Harris detector and HOG descriptor on MHI to detect and represent local motion information. Similar to MHI, our proposed DMM also stacks the foreground motion region to record where and how actions are evolved. However, there are the following main differences: 1) MHI only keeps most recent motions to capture the recency of action, while DMM accumulates global activities through the entire video sequence to represent the motion intensity; 2) DMM stacks motion regions from front, side, and top views, i.e., the three orthogonal projections of depth maps, while only a single view is available in MHI. In essence, the fundamental difference of previous methods to our approach is that they represent features based on 2D color frames, instead of 3D depth maps which capture additional shape and motion cues.

With the release of RGB-D sensors, research of activity recognition based on depth information has been actively explored. Li et al. [42] sampled a set of representative 3D points from depth maps to characterize the posture being per-

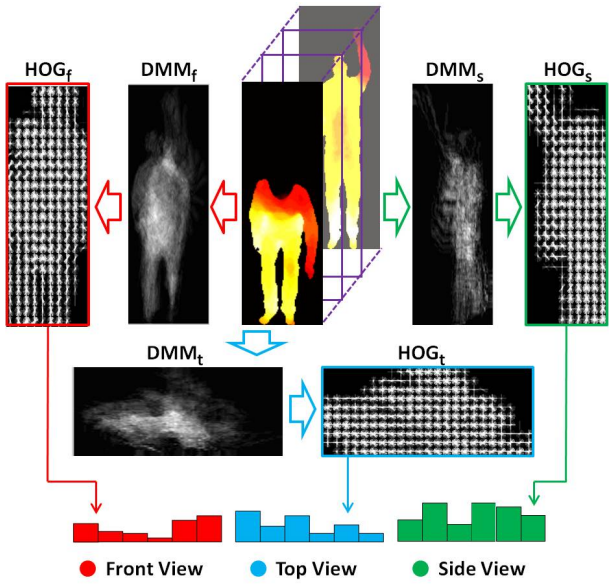


Figure 5.1: The framework of computing DMM-HOG. HOG descriptors extracted from depth motion map of each projection view are combined as DMM-HOG, which is used to represent the entire action video.

formed in each frame. They first projected depth maps onto three orthogonal planes and sampled 2D points at equal distance along the contours on the three projections. The sampled 3D points were then retrieved in depth maps according to the contour points. However, the sampled 3D points of each frame could generate a considerable amount of data which resulted in expensive computations in clustering training data of all classes. In Chapter 4, we have proposed an EigenJoints based activity recognition method by using a NBNN classifier. The compact representation of EigenJoints employed joints differences to capture action information of static postures, consecutive motions, and overall dynamics. However, the 3D positions of skeleton joints could be unstable or even complete wrong if there are sever occlusions [60].

5.2 Computation of DMM-HOG

Our framework to compute the representation of DMM-HOG is demonstrated in Figure 5.1. We project each depth map onto three planes and compute associated motion energy, which are then stacked to obtain DMM. Global HOG descriptors are extracted from the three depth motion maps and concatenated as the final representation of DMM-HOG.

5.2.1 Depth Motion Maps (DMM)

In order to make use of the additional shape and motion information provided by the 3D depth maps, we project each depth map onto three orthogonal planes. We set the region of interest of each projected map as the bounding box of foreground, i.e., non-zero region, which is further normalized to a fixed size. The normalization process is able to reduce intra-class variations, e.g., subject heights and motion extents of different subjects. So each 3D depth map generates three 2D maps according to the front, side, and top views, i.e., map_f, map_s, and map_t. As for each projected map, we obtain its motion energy by computing and thresholding the difference between two consecutive projected maps. The binary map of motion energy indicates the motion region or where motion happens in each temporal interval. It provides a strong clue of the action category being performed. We then stack the motion energy through the whole video sequence to generate the depth motion map DMM_v for each projection view:

$$DMM_v = \sum_{i=1}^{n-1} \left(|map_v^{i+1} - map_v^i| > \epsilon \right) \tag{5.1}$$

where $v \in \{f, s, t\}$ denotes the projection view; map_v^i is the projected map of the ith frame under projection view v; n is the number of frames; $|map_v^{i+1} - map_v^i| > \epsilon$ is the binary map of motion energy; and ϵ is the threshold. We empirically set $\epsilon = 50$ in our experiments. As shown in Figure 5.1, DMM generated from a video of *Pickup Throw* demonstrates a specific appearance and shape, which characterizes the accumulated motion distribution and intensity of this action. DMM representation encodes the 4D information of body shape and motion in the three projected planes, meanwhile significantly reduces the data of a depth sequence to only three 2D projection maps.

5.2.2 DMM-HOG Descriptor

HOG is able to characterize the appearance and shape on DMM by the distribution of local intensity gradients. The basic idea is to compute gradient orientation histograms on a dense grid of uniformly spaced cells. In each cell, 4 different normalizations, i.e., ℓ1-norm, ℓ2-norm, ℓ1-sqrt, and ℓ2-hys [11], are computed based on adjacent histograms. As for each depth motion map, we evenly sample 23×10 non-overlapping cells and 8 gradient orientation bins. So each DMM_v generates a descriptor HOG_v with a dimension of $4 \times 23 \times 10 \times 8 = 7,360$. As illustrated in Figure 5.1, we concatenate $[HOG_f, HOG_s, HOG_t]$ as the DMM-HOG descriptor which is the input to a linear SVM classifier to recognize human activities.

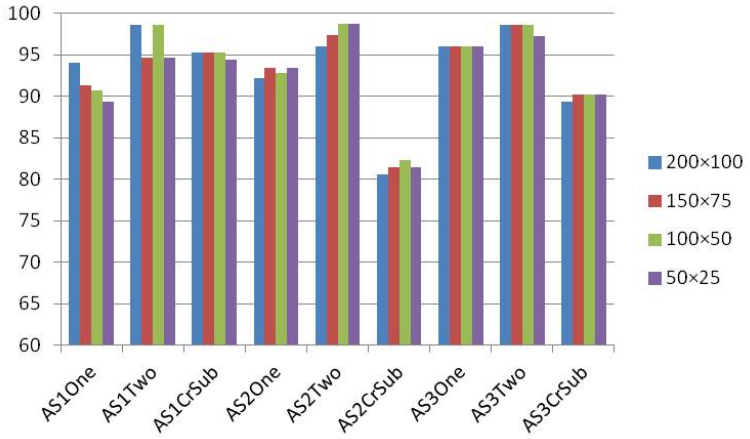


Figure 5.2: Recognition rates (%) of DMM with different normalization sizes under a variety of test sets.

5.3 Experiments and Discussions

The proposed method is evaluated on the MSRAction3D dataset [42]. We extensively compare our approach to the previous methods under a variety of experimental settings. We further investigate how many frames are sufficient to recognize actions using DMM-HOG.

5.3.1 Experimental Setup

As introduced in Section 4.5.1, MSRAction3D contains 20 action categories performed by 10 subjects. These action categories are chosen in the context of interactions with game consoles. We follow the same experimental settings as [42] to split the 20 categories into three subsets as listed in Table 4.1. For each subset, there are three different tests, i.e., Test One (One), Test Two (Two), and Cross-Subject Test (CrSub). In Test One, one third of the subset is used as training the rest as testing; in Test Two, two thirds of the subset is used as training and the rest as testing; in Cross-Subject Test, half subjects are used for training and the rest ones used for testing.

5.3.2 Evaluation of DMM-HOG

We first evaluate the effect of normalization size of DMM. As discussed in Section 5.2.1, we normalize the three depth motion maps to a fixed size. Figure 5.2 demonstrates recognition accuracy of DMM with different normalization sizes under a variety of test sets. The overall recognition rates of most test sets are

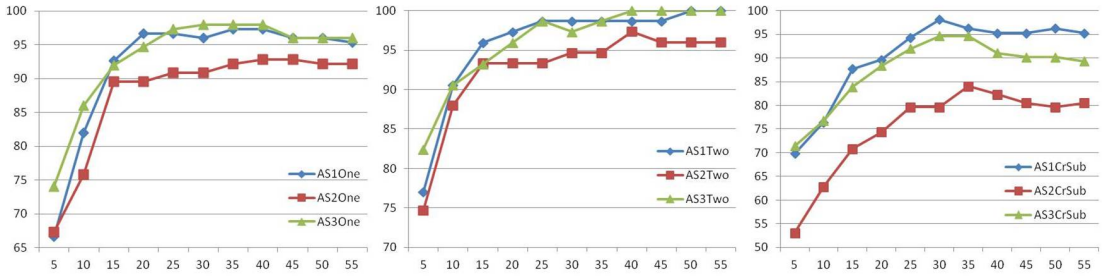


Figure 5.3: Recognition accuracy (%) with different numbers of first few frames in Test One (left), Test Two (middle), and Cross-Subject Test (right) on the MSRAction3D dataset. $30-35$ frames are sufficient to enable reasonably accurate recognition in most test cases.

similar across different DMM normalization sizes. For AS1One and AS1Two, the size of 200×100 achieves the best result, while for AS2CrSub the size of 50×25 outperforms others. Although lower resolutions are able to reduce computational cost in calculating HOG, we extract HOG descriptors only from the three depth motion maps, instead of each depth map. So for each video, the difference of computation time between different sizes is quite limited. The following experimental results are based on the size of 200×100. As shown in Figure 5.2, while the performance in AS1CrSub is promising, the recognition accuracy in AS2CrSub and AS3CrSub are relatively low. In Cross-Subject Test, different subjects perform actions with great variations but the amount of subjects is limited, which results in considerable intra-class variations. Furthermore, some actions in AS2 are quite similar, e.g., *Draw X*, *Draw Tick*, and *Draw Circle*, which also generates small inter-class variations. The performances on Cross-Subject Test might be improved by having more subjects.

5.3.3 How Many Frames Are Sufficient

Most existing systems recognize human activity by operating on the entire video sequence. Similar to EigenJoints in Chapter 4, we also conduct experiments to investigate how many frames are sufficient for action recognition with reasonably accurate results in the DMM-HOG framework. The recognition rates using different amount of frames under a variety of test sets are demonstrated in Figure 5.3. The sub-sequence is chosen from the first K frames in a given video. As shown in this figure, in most cases 30 to 35 frames are sufficient to achieve comparable results to the ones using the entire sequence, with quite limited gains or even some loss when more frames are added in. As affect recognition in [22], the temporal segments of an action can be intuitively approximated by the status of

Table 5.1: Comparisons of recognition accuracy (%) of different methods on the MSRAction3D dataset.

Test Set	3D Silhouettes [42]	EigenJoints [86]	DMM-HOG
AS1One	89.5	94.7	**97.3**
AS2One	89.0	**95.4**	92.2
AS3One	96.3	97.3	**98.0**
AS1Two	93.4	97.3	**98.7**
AS2Two	92.9	**98.7**	94.7
AS3Two	96.3	97.3	**98.7**
AS1CrSub	72.9	74.5	**96.2**
AS2CrSub	71.9	76.1	**84.1**
AS3CrSub	79.2	**96.4**	94.6

neutral, onset, apex, and offset. The most discriminative information is contained in the status of apex and onset, which are probably covered by the first 30 to 35 frames of the MSRAction3D dataset. The sequence after apex contributes little or even incurs noise. This observation is in consistence with the one in EigenJoints and provides important guides to reduce latency of an action recognition system where decisions have to be made on line. The following results are based on the sub-sequence of the first 35 frames.

5.3.4 Comparisons to the State-of-the-Art Methods

We compare our DMM-HOG approach with the previous methods including 3D Silhouettes [42] and EigenJoints [86] on the MSRAction3D dataset in Table 5.1. Recognition accuracy of 3D Silhouettes and EigenJoints are obtained from [42] and [86]. The best results under different test sets are highlighted in bold. As shown in this table, our method consistently and significantly outperforms 3D Silhouettes in all testing cases. The overall accuracy under non-cross-subject tests of our method is comparable to that of EigenJoints. But our method greatly outperforms EigenJoints for cross-subject tests. The significant improvement of our method in cross-subjects tests is probably because the normalization process in computing depth motion maps helps to reduce variations of different subjects, as well as the robust representation of DMM-HOG. In addition to recognition accuracy, out approach is much more compact than 3D Silhouettes. Moreover, our recognition result is achieved by using a short sub-sequence (35 frames), while 3D Silhouettes relied on the entire video sequences.

5.4 Summary

In this chapter, we have proposed an effective action recognition method by using DMM-HOG. The compact and discriminative representation is able to capture the global activities from front, side, and top views. Experimental results on the MSRAction3D dataset demonstrate that our approach significantly outperforms the previous methods. In addition, we observe that in our framework a short sub-sequence of 30 to 35 frames is sufficient to perform action recognition with reasonably accurate results.

Chapter 6

Super Normal Vector for Activity Recognition in Depth Videos

This chapter presents a new framework for human activity recognition from video sequences captured by a depth camera. We cluster hypersurface normals in a depth sequence to form the polynormal which is used to jointly characterize the local motion and shape information. In order to globally capture the spatial and temporal orders, an adaptive spatio-temporal pyramid is introduced to subdivide a depth video into a set of space-time cells. We then propose a novel scheme to aggregate the low-level polynormals into the super normal vector (SNV) which can be seen as a simplified version of the Fisher kernel representation. In the extensive experiments, we achieve classification results superior to all previous published results on the four public benchmark datasets, i.e., MSRAction3D, MSRDailyActivity3D, MSRGesture3D, and MSRActionPairs3D.

In the past decades, research on activity recognition mainly focused on recognizing actions from videos captured by conventional visible light cameras. As the imaging techniques advance, the recent emergence of low-cost and easy-operation depth sensors facilitates a variety of visual recognition tasks including activity recognition. Depth maps have several advantages with respect to traditional color frames in the context of activity recognition. First, they provide additional body shape and structure information, which has been successfully applied to recover skeleton joints from a single depth map. Second, color and texture are removed in depth maps, which eases the problems of human detection and segmentation. Third, depth sensors are insensitive to lighting change, which brings great benefits to the system monitoring in the dark environment.

It was recently shown in [51] [80] that conventional approaches based upon color sequences could not perform well on depth maps due to a large amount of false point detections fired on the spatio-temporally discontinuous regions. On the other hand, depth maps and color frames have quite different properties. The

3D Activity Dataset	Previous Best Results	Our Results
MSRAction3D	91.70% [91]	**93.45%**
MSRGesture3D	92.45% [51]	**94.72%**
MSRActionPairs3D	96.67% [51]	**98.89%**
MSRDailyActivity3D	85.75% [75]	**86.25%**

Table 6.1: Our results compared to the best published results so far on the four datasets (more detailed comparisons in Table 6.2-6.5).

descriptors based on brightness, gradient, and optical flow in traditional color sequences might be unsuited to represent depth maps. It is therefore intuitive to design action features according to the specific characteristics of depth sequences, e.g., cloud points [74] [75] and surface normals [51] [92].

In this chapter, we propose a novel activity recognition framework based upon the polynormal which is a group of hypersurface normals in depth sequences. A polynormal clusters the extended surface normals from a local space-time subvolume. It can be used to jointly capture the local motion and geometry cues. A sparse coding approach [46] is employed to compute the polynormal dictionary and coefficients. We record the differences between polynormals and visual words. The coefficient-weighted difference vectors are aggregated through spatial average pooling and temporal max pooling for each visual word. The vectors of all visual words are in the end concatenated as a feature vector, which can be viewed as a non-probabilistic simplification of the Fisher kernel representation [53]. We further subdivide a depth video into a set of space-time cells. An adaptive spatio-temporal pyramid is proposed to capture the spatial layout and temporal order in a global way. We concatenate the vectors extracted from all the space-time cells as the final representation of super normal vector (SNV).

We evaluate our method according to the standard experimental protocols on the four public benchmark datasets: MSRAction3D [42], MSRDailyActivity3D [75], MSRGesture3D [74], and MSRActionPairs3D [51]. Our results outperform the previous published ones as shown in Table 6.1.

The main contributions of this chapter is summarized as follows. First, we group hypersurface normals from a local space-time volume to polynormal which reserves correlations between local normals and is more robust against noise than individual normal [51]. Second, a novel approach is proposed to aggregate low-level polynormals into the discriminative representation of SNV. Third, our adaptive spatial-temporal pyramid is better adapted to retain the space-time orders than the widely used uniform cells [38] [51] [71] [75]. Moreover, our framework is flexible to combine with skeleton joints to compute SNV for each joint trajectory.

The remainder of this chapter is organized as follows. Section 6.1 introduces the related work on activity recognition using depth sequences. Section 6.2 describes the concept of polynormal. In Section 6.3, we provide detailed procedures to compute SNV. A variety of experimental results and discussions are presented in Section 6.4. Finally, Section 6.5 summarizes the remarks of this chapter.

6.1 Related Work

Research on activity recognition has explored a number of representations of depth sequences, ranging from skeleton joints [86], cloud points [74], projected depth maps [89], local interest points [23], to surface normals [51].

As introduced in Chapter 4, human actions can be modeled by the movements of skeleton joints. The moving pose descriptor was recently proposed in [91] by using the configuration, speed, and acceleration of joints. To reduce joint estimation errors, the pose set [70] selected the best-k joint configurations by segmentation and temporal constraints. The relative positions of pairwise joints were also used in [75] as a complementary feature to characterize the motion information. Compared to skeleton joints, cloud points are more robust to noise and occlusion. Wang et al. [74] [75] introduced local and random occupancy patterns to describe depth appearance. In local occupancy patterns, they subdivided the local 3D subvolumes associated with skeleton joints into a set of spatial grids and counted the number of cloud points falling into each grid. Similar representation based on cloud points was also applied to the 4D subvolumes sampled by a weighted sampling scheme in random occupancy patterns.

Approaches based on projected depth maps usually transform the problem in 3D to 2D. In Chapter 5, we stacked differences between projected depth maps as the depth motion maps where HOG was extracted as the global representation of a depth video. Several local interest point detectors specifically designed for depth data were recently proposed. DSTIP was introduced in [80] to localize activity-related interest points from depth videos by suppressing flip noise. Hadfield et al. [23] extended the detection algorithms of Harris corners, Hessian points, and separable filters to the 3.5D and 4D for depth sequences. As shown in [66], surface normal provides most geometric shape information of an object in 3D. HON4D [51] followed this observation to extend the surface normal to the 4D space and quantized them by the regular and discriminative learned polychorons.

Our method presented in this chapter proceeds along with this direction. It relies on the polynormal which is a local cluster of extended surface normals. We propose a novel approach to aggregate the low-level polynormals in each adaptive spatio-temporal cell. The concatenation of feature vectors extracted from all space-time cells forms the final depth video representation.

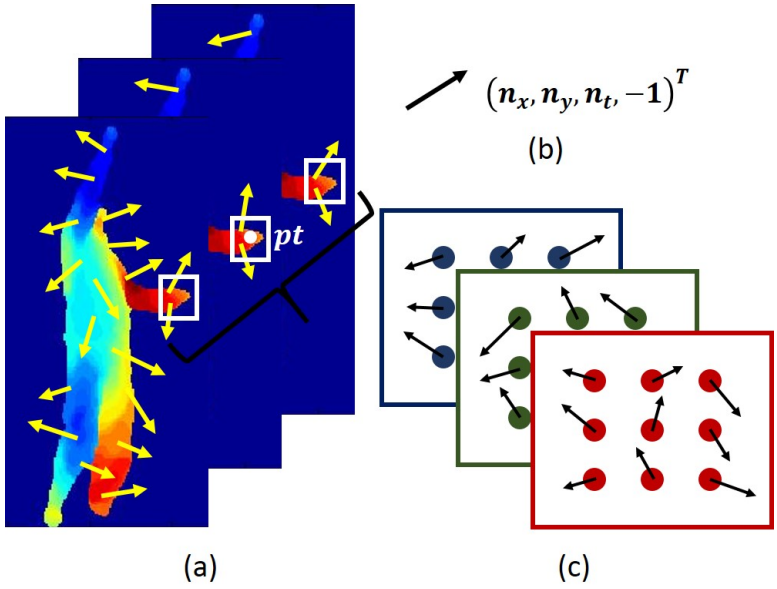


Figure 6.1: Illustration of generating polynormal of the cloud point *pt*. (a) shows a depth sequence of *tennis serve* and normal vectors associated with cloud points. For figure clarity, only a few normal vectors are visualized. The three white squared regions correspond to the neighborhood $\mathcal{L}$. (b) indicates the extended surface normal vector. (c) If $n_s = 9$ and $n_t = 3$, the polynormal of *pt* is consisted of the 27 neighboring normals.

6.2 Polynormal

The concept of a normal to a surface in 3-dimensional space can be extended to a hypersurface in m-dimensional space. The hypersurface can be viewed as a function $\mathbb{R}^{m-1} \rightarrow \mathbb{R}^1 : x_m = f(x_1, \ldots, x_{m-1})$, which is represented by a set of m-dimensional points that locally satisfy $F(x_1, \ldots, x_m) = f(x_1, \ldots, x_{m-1}) - x_m = 0$. The normal vectors to the hypersurface at these points can be computed by the gradient $\nabla F(x_1, \ldots, x_m) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_{m-1}}, -1\right)$. In the context of depth sequences, i.e., $m = 4$, each point satisfies $F(x, y, t, z) = f(x, y, t) - z = 0$. We therefore obtain the extended surface normal by

$$\boldsymbol{n} = \nabla F = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial t}, -1\right)^T. \tag{6.1}$$

The distribution of normal orientations is able to provide more informative geometric cues than the traditional gradient orientations [51]. Moreover, the motion cues are also embedded in the normal vector of Eq. (6.1). In order to

retain the correlation between neighboring normals and make them more robust to noise, we propose polynormal to cluster normals from a local spatio-temporal neighborhood. Similar schemes have been validated in other fields. For example, the spatial neighborhoods of low-level features are jointly encoded in deep learning [41] and macrofeatures [6].

A polynormal $\boldsymbol{p}$ associated with each cloud point in a depth video concatenates L normals in the local neighborhood $\mathcal{L}$ of this point:

$$\boldsymbol{p} = \left(\boldsymbol{n}_1^T, \ldots, \boldsymbol{n}_L^T\right)^T, \ \ \boldsymbol{n}_1, \ldots, \boldsymbol{n}_L \in \mathcal{L}. \tag{6.2}$$

The neighborhood $\mathcal{L}$ is a spatio-temporal depth subvolume determined by two parameters n_s and n_t, where n_s denotes the number of neighboring points in spatial and n_t indicates the number of neighboring maps in temporal. Figure 6.1 illustrates the concept of polynormal. A short sequence of the *tennis serve* action is shown in Figure 6.1(a). If we set $n_s = 9$ and $n_t = 3$, then the polynormal of the white point pt concatenates the 27 normals from the three adjacent depth maps as shown in Figure 6.1(c).

6.3 Computing Super Normal Vector

In this section, we describe the detailed procedures to compute SNV based on the low-level polynormals. We utilize the sparse coding to learn a dictionary and code polynormals. Instead of directly pooling the coefficients of coded polynormals, we aggregate the weighted differences between polynormals and visual words into a vector. A depth video is subdivided into a set of space-time cells by our proposed adaptive spatio-temporal pyramid. The feature vectors extracted from each cell are then concatenated as the final representation of SNV.

6.3.1 Aggregating Polynormals

In visual recognition, the global representation of an image or a video is usually obtained by extracting low-level features, coding them over a learned dictionary, and then pooling the distribution of codes in some well-chosen support regions. After the coding step, low-level features are discarded in the recognition pipeline. In our framework, we keep the low-level features by recording the differences between them and visual words. As shown in [28], [53], [93] the relative displacements can provide extra distribution information of low-level features.

We employ sparse coding to learn the dictionary and code polynormals. It is well known that the ℓ_1 penalty yields a sparse solution. Given a training set

of whitened polynormals $\boldsymbol{P} = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N)$ in $\mathbb{R}^{M \times N}$, the sparse coding problem can be solved by

$$\min_{\mathbf{D}, \boldsymbol{\alpha}} \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{2} \|\boldsymbol{p}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1 \right), \tag{6.3}$$
$$\text{subject to } \mathbf{d}_k^T \mathbf{d}_k \leq 1, \forall k = 1, \ldots, K,$$

where $\mathbf{D}$ in $\mathbb{R}^{M \times K}$ is the dictionary, each column $(\mathbf{d}_k)_{k=1}^K$ representing a visual word; $\boldsymbol{\alpha}$ in $\mathbb{R}^{K \times N}$ is the coefficients of sparse decomposition; λ is the sparsity inducing regularizer.

We ℓ_1-normalize each column $(\boldsymbol{\alpha}_i)_{i=1}^N$ to obtain the soft assignment $\alpha_{k,i}$ of polynormal $\boldsymbol{p}_i$ to the kth visual word. The size of the volume (depth sequences) where we perform the aggregation is $H \times W$ pixels and T frames. The volume corresponds to either the entire video sequence or a subsequence defined by a space-time cell. We denote by N_t the set of indices within the frame t. For each visual word, the spatial average pooling is first applied to aggregate the coefficient-weighted differences:

$$\boldsymbol{u}_k(t) = \frac{1}{|N_t|} \sum_{i \in N_t} \alpha_{k,i} \left(\boldsymbol{p}_i - \mathbf{d}_k \right), \tag{6.4}$$

where $\boldsymbol{u}_k(t)$ represents the pooled difference vector of the kth visual word in the tth frame. The temporal max pooling is then used to aggregate the vectors from T frames:

$$\boldsymbol{u}_{k,i} = \max_{t=1,\ldots,T} \boldsymbol{u}_{k,i}(t), \text{ for } i = 1, \ldots, M, \tag{6.5}$$

where $\boldsymbol{u}_k$ is the vector representation of the kth visual word in the whole volume; i indicates the ith component in corresponding vectors. The final vector representation $\boldsymbol{U}$ is the concatenation of the $\boldsymbol{u}_k$ vectors from the K visual words and is therefore of KM dimensions:

$$\boldsymbol{U} = \left(\boldsymbol{u}_1^T, \ldots, \boldsymbol{u}_K^T \right)^T. \tag{6.6}$$

In order to capture the global spatial layout and temporal order, a depth sequence is subdivided into a set of space-time cells. We extract a feature vector $\boldsymbol{U}$ from each cell and concatenate them as SNV. This representation has several remarkable properties. (1) The displacements to visual words retain some information lost in traditional feature quantization process. (2) We can compute SNV upon a much smaller dictionary (e.g., 100) which reduces computational cost. (3) SNV performs quite well with simple linear classifiers (e.g., SVM with linear kernel) which are efficient in terms of both training and testing.

6.3.2 Relationship with Fisher Kernel

We now demonstrate that our proposed SNV is a simplified non-probabilistic version of the Fisher kernel representation which has been successfully applied in the image classification tasks [56]. Fisher kernel assumes low-level features are distributed according to Gaussian mixture model (GMM).

In the framework of Fisher kernel, each feature descriptor is described by its deviations with respect to the GMM parameters $\beta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\sigma}_k, k = 1, \ldots, K\}$, where π_k, $\boldsymbol{\mu}_k$, and $\boldsymbol{\sigma}_k$ are the mixture weight, mean vector, and covariance matrix (diagonal) of the kth Gaussian component φ_k. The soft assignment of the descriptor $\boldsymbol{p}_i$ to the component φ_k is defined as:

$$\gamma_{k,i} = \frac{\pi_k \varphi_k\left(\boldsymbol{p}_i\right)}{\sum_{j=1}^{K} \pi_j \varphi_j\left(\boldsymbol{p}_i\right)}. \tag{6.7}$$

We denote by $\boldsymbol{p}_i$ a general descriptor and N_t a general pooling region in this context. We focus on the gradient $\boldsymbol{g}_k$ with respect to the mean vector $\boldsymbol{\mu}_k$ of the kth Gaussian:

$$\boldsymbol{g}_k = \frac{1}{|N_t|\sqrt{\pi_k}} \sum_{i \in N_t} \gamma_{k,i} \boldsymbol{\sigma}_k^{-1} \left(\boldsymbol{p}_i - \boldsymbol{\mu}_k\right). \tag{6.8}$$

If making the two hypotheses: (1) mixture weights are equal, i.e., $\pi_k = 1/K$ and (2) covariance matrices are isotropic, i.e., $\boldsymbol{\sigma}_k = \epsilon \mathbb{I}$ with $\epsilon > 0$, we can simplify Eq. (6.8) to

$$\boldsymbol{g}_k \propto \frac{1}{|N_t|} \sum_{i \in N_t} \gamma_{k,i} \left(\boldsymbol{p}_i - \boldsymbol{\mu}_k\right), \tag{6.9}$$

where $\gamma_{k,i}$ is simplified to $\varphi_k\left(\boldsymbol{p}_i\right) / \sum_{j=1}^{K} \varphi_j\left(\boldsymbol{p}_i\right)$. The two representations in Eq. (6.4) and Eq. (6.9) have the same form except the ways to obtain the weight ($\alpha_{k,i}$ and $\gamma_{k,i}$) and the center ($\mathbf{d}_k$ and $\boldsymbol{\mu}_k$). We utilize sparse coding to compute the weight and center, while GMM clustering is used in the Fisher kernel.

We choose sparse coding over GMM in our aggregation scheme because it is cheaper to compute the centers (dictionary), especially it was recently shown in [10] that a reasonably good dictionary can be created by some simple methods, e.g., random sampling a training set. In addition, our empirical evaluations show our method based on sparse coding improves the recognition accuracy.

6.3.3 Adaptive Spatio-Temporal Pyramid

In the spatial dimensions, we use a $n_H \times n_W$ grid to capture the geometry layout. As the depth information greatly facilitates human detection and segmentation,

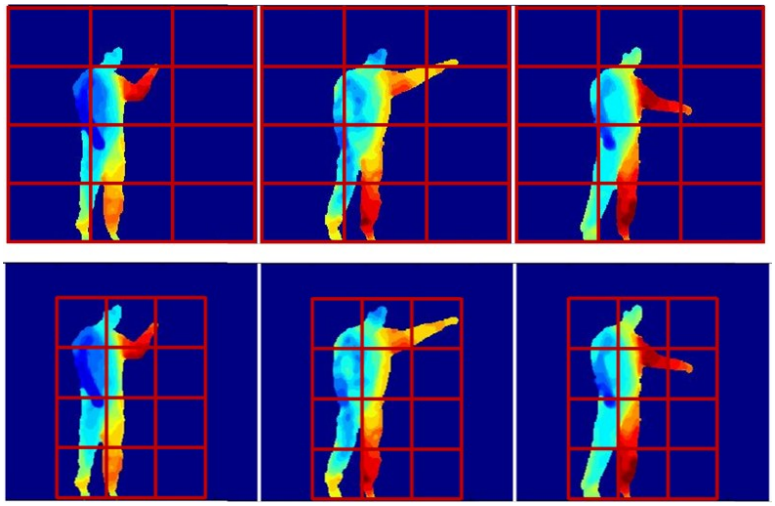


Figure 6.2: Comparison between the traditional (top) and our proposed (bottom) spatial grids. We put the 4×3 spatial grid on the largest bounding box of human body rather than on the entire frame.

we enforce the spatial grid on the largest bounding box of the human body, instead of on the entire frame as widely used in [51] [71]. Figure 6.2 illustrates the comparison between the two schemes of spatial grids.

The temporal pyramid was introduced by Laptev et al. [38] to take into account the rough temporal order of a video. It was also employed in depth sequences [51] [75] to incorporate cues from the temporal context. In these methods, a video sequence (either color or depth) is repeatedly and evenly subdivided into a set of temporal segments where descriptor-level statistics are pooled. However, different people could have varied motion speed or frequency when they are performing the same activity. It is therefore inflexible to handle this variance by evenly subdividing a video along the time axis. In addition, it is more desirable to pool low-level features within the similar activity status, e.g., neutral, onset, apex, and offset. In order to handle these difficulties, we propose an adaptive temporal pyramid based on the motion energy.

Given a depth sequence, we first project the ith frame $\boldsymbol{I}^i$ onto three orthogonal planes to obtain the projected maps $\boldsymbol{I}^i_v, v \in \{1, 2, 3\}$. The difference between two consecutive projected maps is then thresholded to generate a binary map. We compute the motion energy by accumulating summations of non-zero elements of binary maps as:

$$\varepsilon(i) = \sum_{v=1}^{3} \sum_{j=1}^{i-1} \text{sum}\left(\left|\boldsymbol{I}^{j+1}_v - \boldsymbol{I}^j_v\right| > \epsilon\right), \tag{6.10}$$

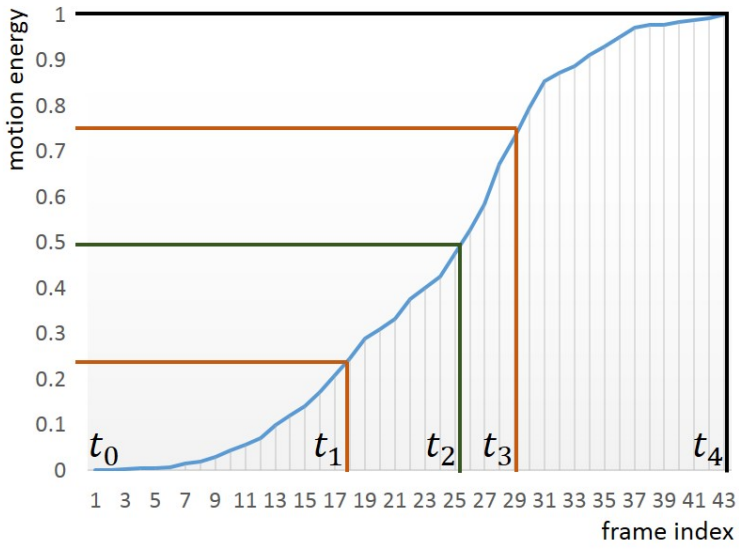


Figure 6.3: The frame index and associated motion energy used to build the adaptive temporal pyramid. The temporal segments are obtained by repeatedly and evenly subdividing the normalized motion energy vector instead of the time axis.

where $\varepsilon(i)$ is the motion energy of the ith frame; ϵ is the threshold; sum$(\cdot)$ returns the number of non-zero elements in a binary map. The motion energy of a frame reflects its relative motion status with respect to the entire activity.

Our proposed adaptive temporal pyramid is built upon this measurement as shown in Figure 6.3. We evenly subdivide the normalized motion energy vector into a set of segments, whose corresponding frame indices are used to partition a video. In the experiments, we use a 3-level temporal pyramid as illustrated in this figure: $\{t_0t_4\}$, $\{t_0t_2, t_2t_4\}$, and $\{t_0t_1, t_1t_2, t_2t_3, t_3t_4\}$. In together with the spatial grid, our adaptive spatio-temporal pyramid in total generates $n_H \times n_W \times 7$ space-time cells.

We summarize the outline of computing SNV of a depth video in Algorithm 3. The set of space-time cells V are chosen by the proposed adaptive spatio-temporal pyramid.

6.3.4 Joint Trajectory Aligned SNV

While the framework discussed above operates on the entire depth sequence, our method is flexible to combine with skeleton joints [60] to compute SNV based on each joint trajectory. This is useful in the scenarios where people significantly change their spatial locations in a depth video. The aggregation process is the

Algorithm 3: Computation of SNV

Input: a depth sequence
a dictionary $\mathbf{D} = (\mathbf{d}_k)_{k=1}^{K}$
a set of space-time cells $V = \{v_i\}$

Output: SNV

compute polynormals $\{\boldsymbol{p}_i\}$ from the depth sequence
compute coefficients $\{\boldsymbol{\alpha}_i\}$ of $\{\boldsymbol{p}_i\}$ by sparse coding
for cell $i = 1$ **to** $|V|$ **do**
  for visual word $k = 1$ **to** K **do**
    $\boldsymbol{u}_i^k :=$ spatial average pooling and temporal max pooling of $\alpha_{k,i}\,(\boldsymbol{p}_i - \mathbf{d}_k)$, where $\boldsymbol{p}_i \in v_i$
  end
  $\boldsymbol{U}_i := \left(\boldsymbol{u}_i^1, \ldots, \boldsymbol{u}_i^K\right)$
end
SNV $:= \left(\boldsymbol{U}_1, \ldots, \boldsymbol{U}_{|V|}\right)$

same as the earlier discussion, except the pooling region is based on the space-time volume aligned around each joint trajectory. It was also shown in dense trajectories [71] that descriptors aligned with trajectories were superior to those computed from straight cuboids.

As shown in Figure 6.4, the volume aligned with a joint trajectory can be viewed as a single video sequence with $H \times W$ pixels and T frames. We apply the adaptive spatio-temporal pyramid on this volume to obtain $n_H \times n_W \times 7$ space-time cells. In each cell, we use the same aggregation scheme, i.e., spatial average pooling and temporal max pooling of the coefficient-weighted difference vectors as in Eq. (6.4-6.5). The vectors from all the space-time cells are concatenated as the joint trajectory aligned SNV. We in the end combine the SNVs aligned with all the joint trajectories as the final representation of a depth sequence.

6.4 Experiments

In this section we extensively evaluate our proposed method on four public benchmark datasets: MSRAction3D [42], MSRGesture3D [74], MSRActionPairs3D [51], and MSRDailyActivity3D [75]. In all experiments, we set a 9×3 neighborhood for each cloud point to form the polynormal. We use 100 visual words in the sparse coding. The adaptive spatio-temporal pyramid is typically of $4 \times 3 \times 7$ space-time cells in height, width, and time, respectively. We employ LIBLINEAR [14] as the linear SVM solver. Our method is extensively compared to the existing depth-based approaches. The methods designed for color videos are not

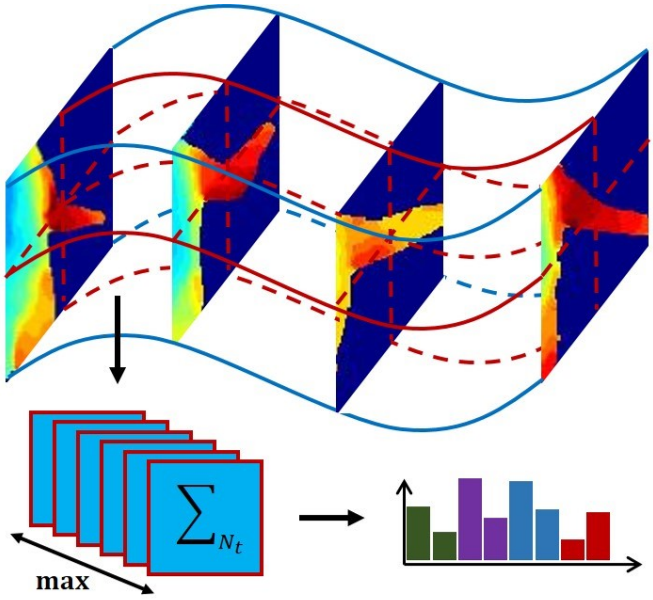


Figure 6.4: SNV based on the skeleton joint trajectory. A trajectory-aligned volume is subdivided into a set of space-time cells according to the adaptive spatio-temporal pyramid. Each cell generates a feature vector by the spatial average pooling and temporal max pooling.

included in our comparisons because they have been widely shown to be unsuited for depth maps. Experimental results show that our algorithm significantly outperforms the state-of-the-art methods on these datasets. Our source code for computing SNV is available online.[1]

6.4.1 Evaluation of SNV Parameters

Here we focus on the MSRAction3D dataset for the detailed evaluations of parameters and settings in SNV. Figure 6.5(a) compares the recognition accuracies of SNV with different number of visual words K. Though SNV achieves the best result using $K = 100$ visual words, the general accuracy is very stable with respect to K. We then evaluate the size of local neighborhood $\mathcal{L}$ to form a polynormal. As discussed in Section 6.2, the size of $\mathcal{L}$ is determined by $\mathcal{L}_x \times \mathcal{L}_y \times \mathcal{L}_t$. Figure 6.5(b) shows the recognition accuracies of SNV with different sizes of $\mathcal{L}$. If no local temporal cue is encoded, i.e., $\mathcal{L}_t = 1$, increasing the spatial size of $\mathcal{L}$ improves the recognition accuracy, e.g., from $1 \times 1 \times 1$, $3 \times 3 \times 1$, to $5 \times 5 \times 1$. When $\mathcal{L}_x$ and $\mathcal{L}_y$ are fixed, the accuracy based on $\mathcal{L}_t > 1$ is much higher than the

[1] http://yangxd.org/code

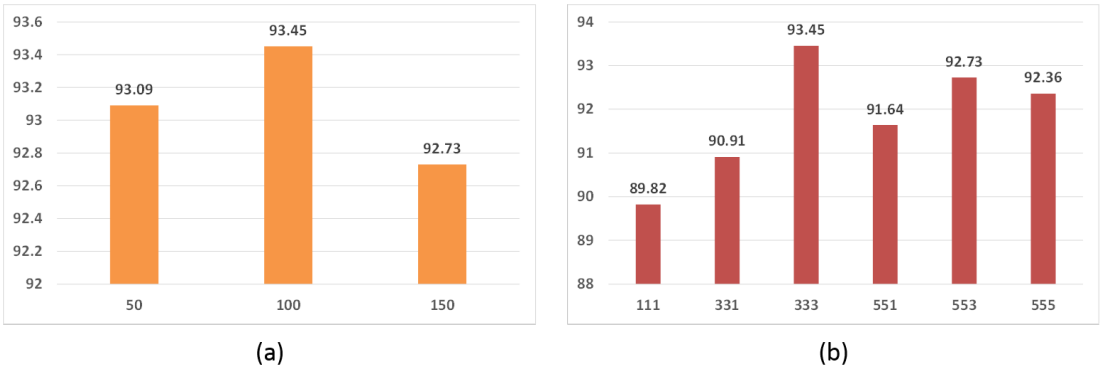
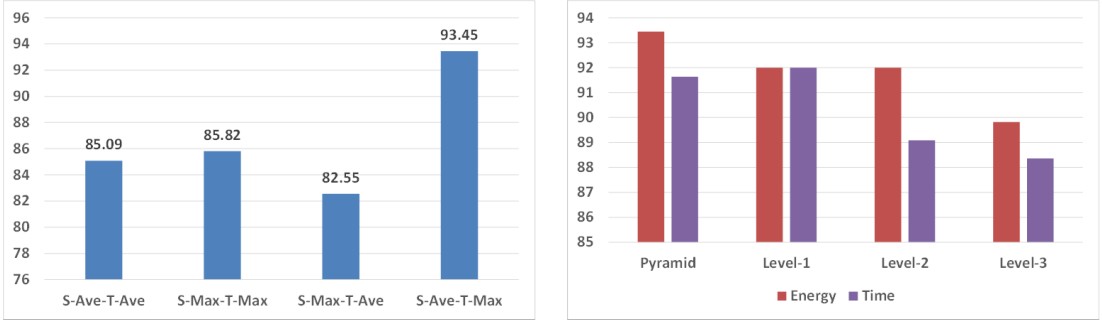


Figure 6.5: Recognition accuracies (%) of SNV with (a) different numbers of visual words and (b) various sizes $\mathcal{L}_x\mathcal{L}_y\mathcal{L}_t$ of $\mathcal{L}$ to form the polynormal.

ones with $\mathcal{L}_t = 1$, e.g., the result of $3 \times 3 \times 3$ significantly outperforms the one of $3 \times 3 \times 1$. In addition, the overall performance of a polynormal is superior to that of an individual normal. This shows the local temporal information embedded in polynormal helps to characterize the low-level motion cues. In the following experiments, we use the $3 \times 3 \times 3$ local neighborhood $\mathcal{L}$ to form the polynormal and 100 visual words.

As described in Eq. (6.4-6.5), we apply spatial average pooling and temporal max pooling to aggregate polynormals. This design can be validated from our empirical observation. Figure 6.6(a) compares recognition accuracies of different combinations of spatial/temporal and average/max pooling. We observe the appropriate choice of pooling in spatial and temporal is critical to the performance of final representation. Figure 6.6(b) demonstrates the comparisons between traditional temporal pyramid and our adaptive temporal pyramid. In level-1, both methods have the same temporal segment (i.e., the entire video sequence), so they have the same recognition accuracy. But in level-2 and level-3, our approach based on motion energy largely outperforms traditional method based on time. When we combine the 3 levels into a temporal pyramid, our adaptive pyramid achieves 1.81% improvement to the traditional pyramid.

To analyze the computational complexity of SNV, we compute SNV from 567 depth videos with the resolution of 320×240. We report the run time using MATLAB on a desktop with a single 2.13GHz CPU and 24G RAM. The average computational speed is 0.13 frame per second. Figure 6.7 shows the percentage of time spent on each step of computing SNV. The coding process takes most of the time with 53%. The pooling process is the second most time-consuming step with 37%. The computation of polynormal only takes 4%. The run time can be improved by parallel computing and reducing densely sampled cloud points.



(a) (b)

Figure 6.6: Recognition accuracies (%) of SNV with different combinations of spatial/temporal and average/max pooling in (a). Comparisons between our proposed adaptive temporal pyramid based on motion energy and the traditional pyramid based on time in (b).

6.4.2 MSRAction3D Dataset

The MSRAction3D [42] is a dataset of depth sequences captured by a RGB-D camera. It contains 20 action categories performed by 10 subjects facing the camera. Each action is performed 2 or 3 times by each subject. The 20 actions are selected in the context of gaming and cover a variety of movements related to arms, legs, torso, etc., as illustrated in Figure 4.4.

In order to facilitate a fair comparison, we follow the same experimental setting as [75]. SNV achieves the accuracy of 93.45% which significantly outperforms the previous methods. If we only keep the first level (i.e., $\{t_0 t_4\}$ in Figure 6.3) of the adaptive temporal pyramid, the accuracy goes down to 91.64%. This decrease shows action recognition benefits from the cues in the global temporal order. The confusion matrix of our method is demonstrated in Figure 6.11. As shown in the confusion matrix, our method works very well on most actions. The recognition errors concentrate on those quite similar actions, e.g., *hand catch* to *high throw* and *draw circle* to *draw tick*.

We compare the performance of SNV with other results in Table 6.2. The approaches based on joints are vulnerable to the errors of joint estimation due to severe self-occlusions. So the model in [70] selects the best-k joint configurations which largely remove inaccurate joints. The method in [91] utilizes pose, speed, and acceleration of joints. While still inferior to our method, the approaches in [69] [74] [75] improve the results in [81] [84] because cloud points are more resistant to occlusions and provide additional shape cues compared to skeleton joints. SNV outperforms HON4D [51] by 4.56%, though both methods are based upon hypersurface normals. This is probably because (1) polynormals obtain

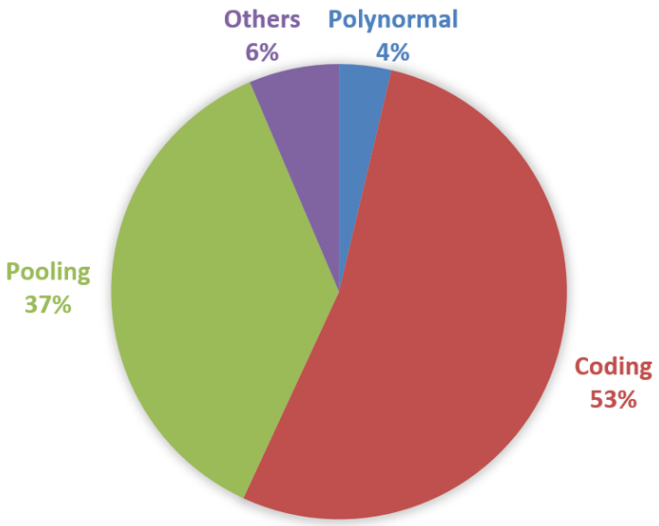


Figure 6.7: Percentage of time spent on each major step of computing SNV with default parameter setting.

more discriminative local motion and shape information than individual normals; (2) sparse coding is more robust than the polychoron and learned projectors; (3) our aggregation scheme, i.e., spatial average pooling and temporal max pooling of weighted difference vectors, is more representative than the sum pooling of inner production values; (4) the adaptive pyramid is more flexible than the uniform cells to capture the global spatio-temporal cues.

6.4.3 MSRGesture3D Dataset

The MSRGesture3D [74] is a dynamic hand gesture dataset of depth sequences captured by a depth camera. As illustrated in Figure 6.8, it contains 12 dynamic hand gestures defined by the American Sign Language (ASL). There are 10 subjects, each one performing each dynamic gesture 2 or 3 times. This dataset presents more self-occlusions than MSRAction3D.

The leave-one-out cross-validation scheme as [74] is used in our evaluation. SNV obtains the state-of-the-art accuracy of 94.74% which outperforms the previous methods as shown in Table 6.3. The confusion matrix of SNV is shown in Figure 6.12. Our method performs pretty well on most dynamic gestures. The most confusion occurs in recognizing the gestures *green* which shares similar motions to *j* but with different fingers. As joint estimation is not available for hands, the joint-based methods [70] [75] [81] [84] [91] cannot be used in this application.

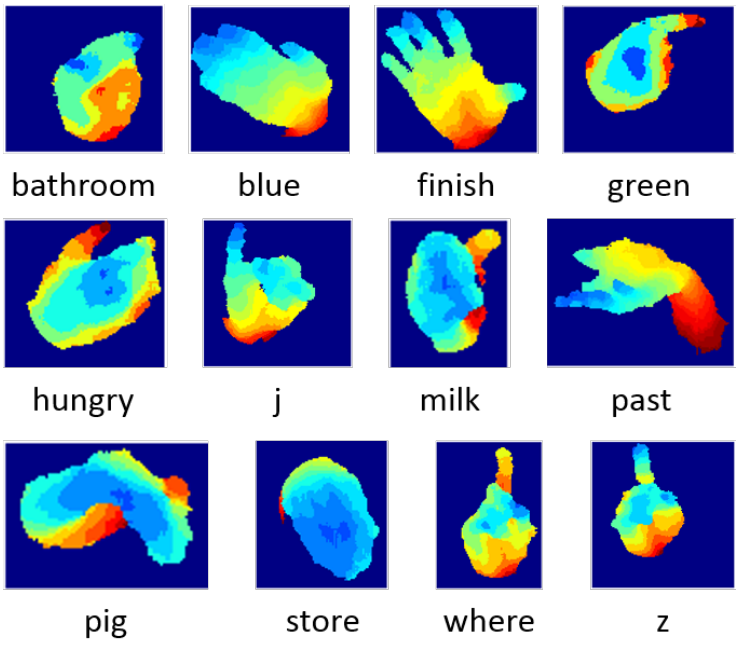


Figure 6.8: Examples of depth maps associated with sampled frames in the MSRGesture3D dataset.

6.4.4 MSRActionPairs3D Dataset

The MSRActionPairs3D [51] is a paired-activity dataset of depth sequences captured by a depth camera. It contains 12 activities (i.e., 6 pairs in Figure 6.9) of 10 subjects with each subject performing each activity 3 times. This dataset is collected to investigate how the temporal order affects activity recognitions.

The same evaluation setup as [51] is used in our experiment. SNV achieve the state-of-the-art accuracy of 98.89%. The detailed comparison to other approaches is demonstrated in Table 6.4. The skeleton feature [75] only involves pair-wise difference of joint positions within each frame. The LOP feature [75] is used to characterize the depth appearance. It counts the number of cloud points falling into each spatial grid of a depth subvolume. There is no temporal information encoded in the two features. In depth motion maps [89], depth sequences are collapsed onto three projected maps where temporal orders are eliminated. These methods therefore suffer the inner-paired confusion. The skeleton and LOP features equipped with a uniform temporal pyramid improves the recognition result as the global temporal order is incorporated. However, this result is still significantly inferior to ours.

It is therefore crucial to capture the spatio-temporal orders to distinguish the activities with similar motion and shape cues. In our method, the space-time

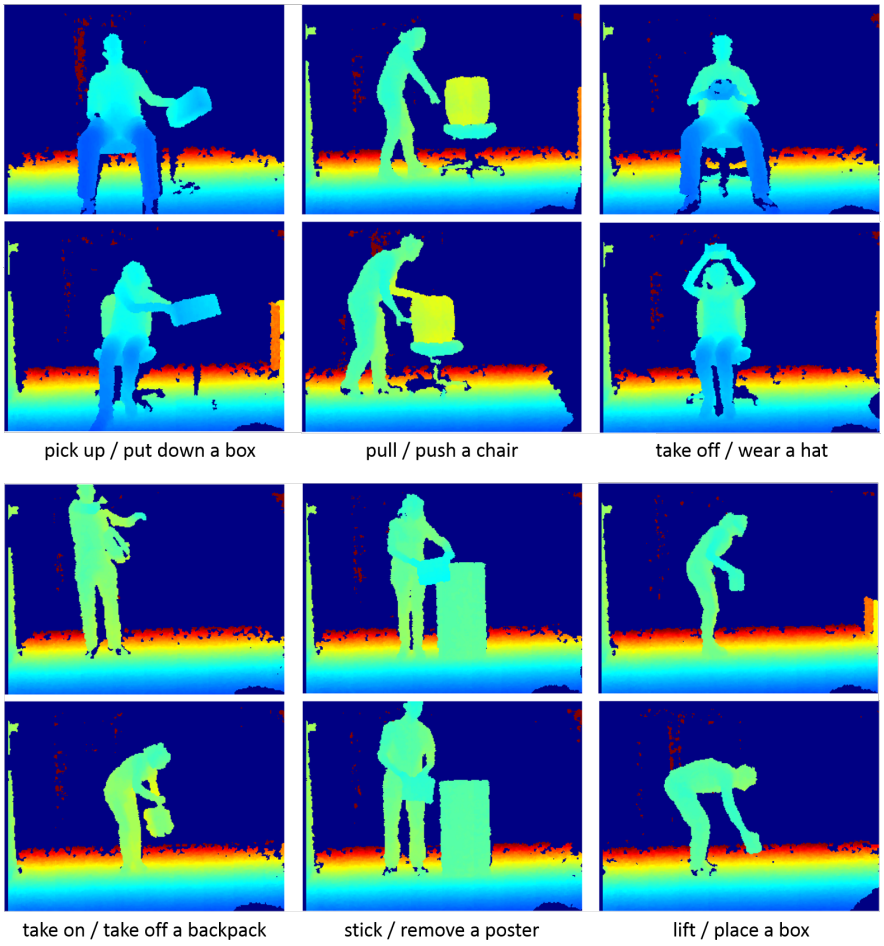


Figure 6.9: Examples of depth maps associated with sampled frames in the MSRActionPairs3D dataset.

orders are embedded in two levels, i.e., polynormals and the adaptive pyramid, which characterize the local and global spatio-temporal orders, respectively. It is interesting to observe that SNV achieves an accuracy of 97.78% if no temporal pyramid is used. This promising result demonstrates the local motion cues enclosed in the polynormals reflect the temporal orders pretty well. Because of the high recognition accuracy, the confusion matrix on this dataset is omitted.

6.4.5 MSRDailyActivity3D Dataset

The MSRDailyActivity3D [75] is a daily activity dataset of depth sequences captured by a depth camera. As shown in Figure 6.10, there are 16 daily activities which are performed by 10 subjects. Each subject performs each activity twice,

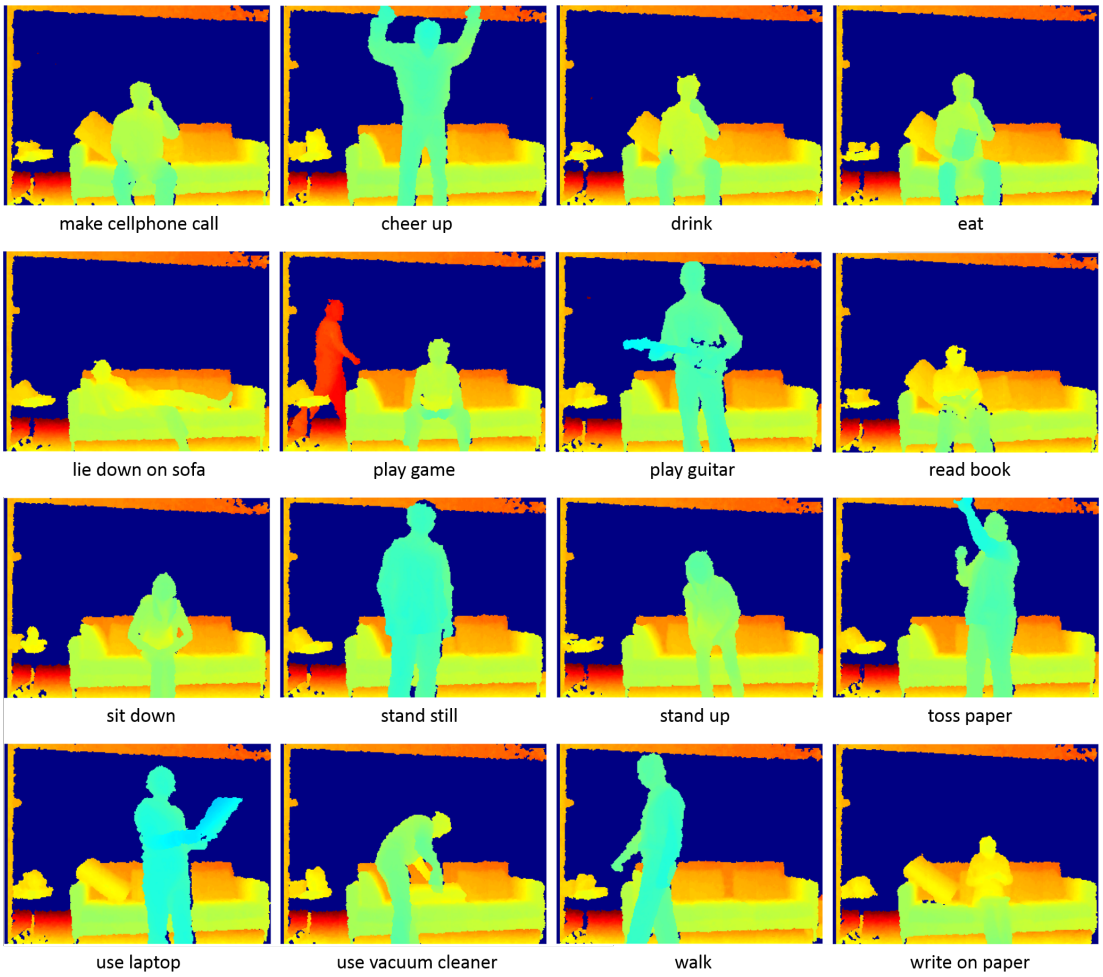


Figure 6.10: Examples of depth maps associated with sampled frames in the MSRDailyActivity3D dataset.

one in standing position and the other in sitting position. Compared to the other three datasets, people in this dataset present large spatial and scaling changes. Moreover, most activities involve human-object interactions.

In order to handle the significant spatial and scaling changes, we employ the joint trajectory aligned SNV on this dataset. Each joint is tracked through the entire depth sequence. A patch is associated with each joint in each frame. Because depth values inversely vary with an object size, we set an adaptive size s/z to each patch, where $s = 300\text{K}$ is a scale factor and z is the depth value of a joint in the current frame. Unlike the fixed patch size used in [51], the adaptive size is more robust to handle the scaling change. So the patch size in Figure 6.4 is not necessary to be consistent. We compute SNV and joint position difference

feature for each joint trajectory. The actionlet ensemble model [75] is then used to combine the features from multiple joints.

We follow the same experimental setting as [75] and obtain the accuracy of 86.25%. The confusion matrix is shown in Figure 6.13. Most recognition errors occur in the almost still activities, e.g., *read book*, *write*, and *use laptop*. Since most activities involve human-object interactions, this dataset can be used to evaluate how the motion and shape information are correlated. It could be insufficient to capture motion and shape independently because some activities share quite similar motion cues but present distinct shape properties. SNV jointly encodes local motion and shape information in polynormals which in the high level reflect the co-occurrence of hand motion and object shape.

Table 6.5 demonstrates the performance of our method compared to the previous approaches. Note: an accuracy of 88.20% was reported in [80], however, four activities with less motion (i.e., *sit or stand still*, *read books*, *write on paper*, and *use laptop*) were removed in their experiment. The holistic approach [89] suffers the non-aligned sequences. The methods [44] [59] [84] [91] based on either motion or shape information alone are significantly inferior to our method and the ones [51] [75] that jointly model the two cues.

6.5 Summary

We have presented a novel framework to recognize human activities from depth sequences. The polynormal based on extended surface normals jointly encodes local motion and shape cues. A new aggregation scheme is proposed by sparse coding polynormals, as well as spatial average pooling and temporal max pooling of the coefficient-weighted difference vectors between polynormals and visual words. We have introduced the adaptive spatial-temporal pyramid which is shown to be better adapted to retain the spatial and temporal orders. Our proposed framework is also flexible to be combined with the joint trajectory aligned depth sequence, which is well suited in the scenarios where significant spatial and scaling changes present. Our method is extensively evaluated on four public benchmark datasets and compared to a number of state-of-the-art approaches. Experimental results demonstrate that our method outperforms all previous approaches on these datasets.

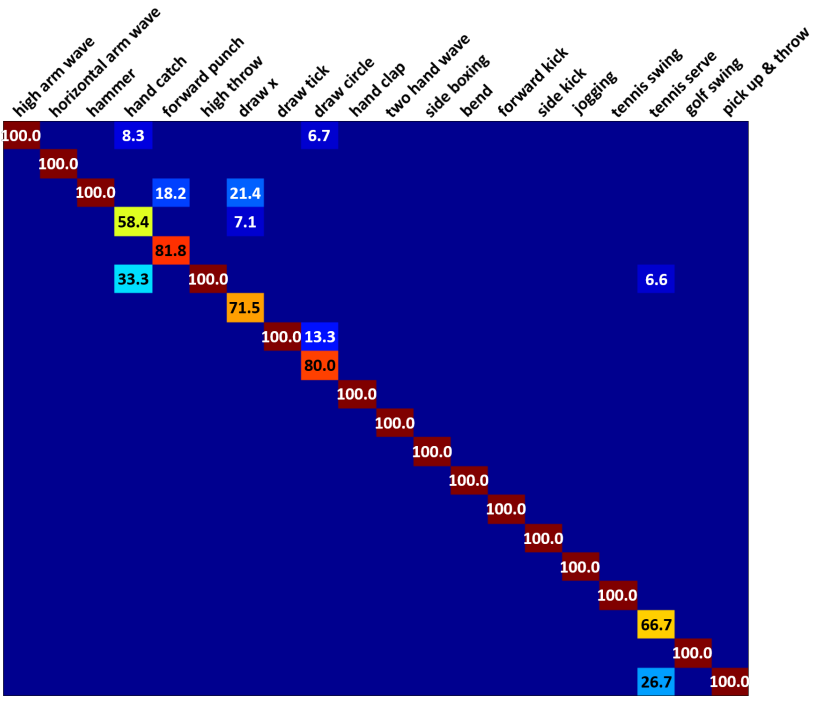


Figure 6.11: Confusion matrix of SNV on the MSRAction3D dataset. This figure is better viewed on screen.

Method	**Accuracy**
Bag of 3D Points [42]	74.70%
HOJ3D [81]	79.00%
EigenJoints [86]	82.30%
STOP [69]	84.80%
Random Occupancy Pattern [74]	86.50%
Actionlet Ensemble [75]	88.20%
Depth Motion Maps [89]	88.73%
HON4D [51]	88.89%
DSTIP [80]	89.30%
Pose Set [70]	90.00%
Moving Pose [91]	91.70%
Ours	**93.45%**

Table 6.2: Recognition accuracy comparison of our method and previous approaches on the MSRAction3D dataset.

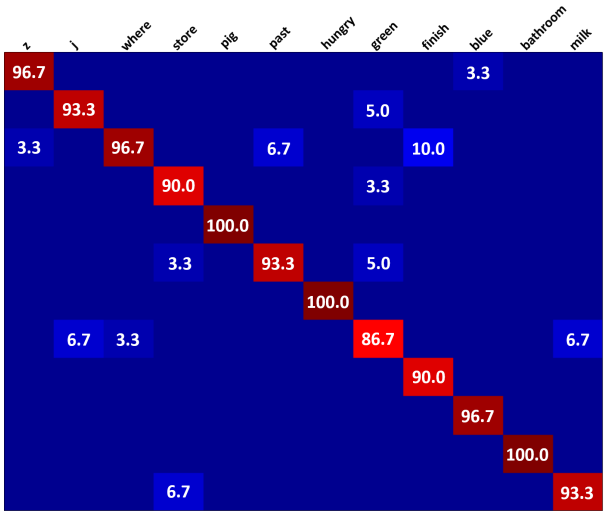


Figure 6.12: Confusion matrix of SNV on the MSRGesture3D dataset. This figure is better viewed on screen.

Method	**Accuracy**
Action Graph on Occupancy [35]	80.50%
Action Graph on Silhouette [35]	87.70%
Random Occupancy Pattern [74]	88.50%
Depth Motion Maps [89]	89.20%
HON4D [51]	92.45%
Ours	**94.74%**

Table 6.3: Recognition accuracy comparison of our method and previous approaches on the MSRGesture3D dataset.

Method	**Accuracy**
Skeleton + LOP [75]	63.33%
Depth Motion Maps [89]	66.11%
Skeleton + LOP + Pyramid [75]	82.22%
HON4D [51]	96.67%
Ours	**98.89%**

Table 6.4: Recognition accuracy comparison of our method and previous approaches on the MSRActionPairs3D dataset.

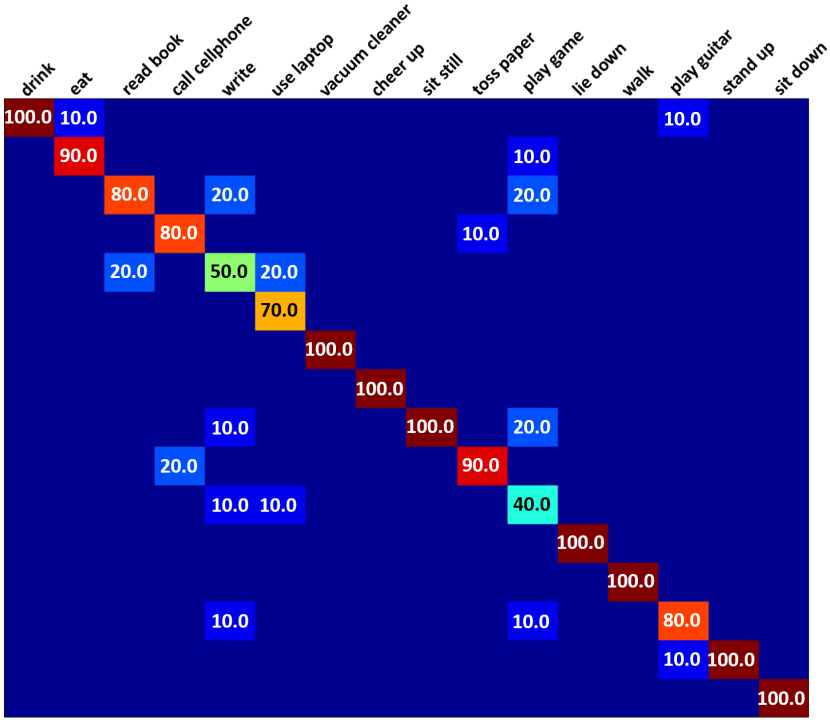


Figure 6.13: Confusion matrix of SNV on the MSRDailyActivity3D dataset. This figure is better viewed on screen.

Method	**Accuracy**
LOP [75]	42.50%
Depth Motion Maps [89]	43.13%
EigenJoints [84]	58.10%
Joint Position [75]	68.00%
NBNN + Parts + Time [59]	70.00%
RGGP [44]	72.10%
Moving Pose [91]	73.80%
Local HON4D [51]	80.00%
Actionlet Ensemble [75]	85.75%
Ours	**86.25%**

Table 6.5: Recognition accuracy comparison of our method and previous approaches on the MSRDailyActivity3D dataset.

Chapter 7

Conclusion

This dissertation is dedicated to developing effective feature representations for human activity recognition on the two visual sequences including color frames and depth maps. We have demonstrated both in theory and practice with clear performance gains in a variety of experiments. Specifically, we have developed the following methodologies for human activity recognition.

Surveillance Event Detection We present a general surveillance event detection system using a temporal sliding windows as the detection unit. ActionHOG is developed to compute low-level motion features in a very efficient way. We take advantage of the event spatial priors to remove a large amount of background features. The CascadeSVMs algorithm is introduced to deal with the highly imbalanced large-scale data learning problem in the context of video surveillance.

Super Sparse Coding Vector We propose an effective feature coding method to aggregate descriptors and locations of low-level features into the discriminative representation of super descriptor vector (SDV). We incorporate the spatio-temporal locations as part of the coding step to generate the compact representation of super location vector (SLV). The combination of SDV and SLV is the super sparse coding vector (SSCV) which jointly models the motion, appearance, and location information in a unified, discriminative, and compact way. Compared to the widely used spatio-temporal pyramid based methods, SSCV greatly improves recognition accuracy, and meanwhile significantly reduces memory cost and computational complexity.

EigenJoints We propose an effective approach of EigenJoints to recognize human activities based on skeleton joints. EigenJoints encodes the information of static posture, motion property, and overall dynamics. We also introduce the accumulated motion energy (AME) to select informative depth maps, which helps

to reduce ambiguous features and computational complexity. We adopt NBNN as the classifier to recognize multiple activity categories. Experimental results demonstrate EigenJoints outperforms the previous methods on three benchmark datasets.

Depth Motion Maps We propose the depth motion maps (DMM) for action recognition in depth sequences by accumulating projected depth maps from three orthogonal views. DMM characterizes the specific appearances and shapes, i.e., the accumulated motion intensity and distribution, of different action categorizes. We then compute HOG from DMM as a global representation of the entire depth sequence. We also investigate how many frames are necessary for our method to perform classification in the scenario of online recognition. We observe that a short sub-sequence is sufficient to achieve comparable recognition results to that using the whole video sequence, with quite limited gains as more frames added in.

Super Normal Vector We present a novel and effective framework of super normal vector (SNV) to recognize human activities in depth videos. We extend the concept of surface normal to polynormal to jointly characterize the local motion and shape cues. We then propose a novel scheme to aggregate low-level polynormals based on the coefficient weighted difference vectors between polynormals and visual words. By making use of the depth information, we introduce an adaptive spatio-temporal pyramid which is more adapted and precise to capture the geometric layout and temporal order. Experimental results on the four benchmark datasets demonstrate SNV achieves the performance superior to all previous published methods.

To look into the future, we will proceed to our research on several open exciting challenges for human activity recognition. I believe multiple feature fusion is crucial for more robust activity recognition. One promising direction is to exploit the complementary information from both color and depth channels in computing different levels of representations. In addition, it is an interesting issue to model and recognize group activity for analyzing complex social interactions. I also expect to explore the deep learning techniques to obtain more powerful and generic feature representations in large-scale video recognition. Moreover, it is of great potential to incorporate parallel processing and cloud computing in developing real-world vision systems.

Appendix A

Publications During Ph.D. Study

1. X. Yang and Y. Tian. Super Normal Vector for Activity Recognition Using Depth Sequences. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

2. X. Yang and Y. Tian. Action Recognition Using Super Sparse Coding Vector with Spatio-Temporal Awareness. European Conference on Computer Vision (ECCV), 2014.

3. S. Chen, X. Yang, and Y. Tian. Discriminative Hierarchical K-Means Tree for Large-Scale Image Classification. IEEE Transactions on Neural Networks and Learning Systems (TNNLS), 2014.

4. X. Rong, C. Yi, X. Yang, and Y. Tian. Scene Text Recognition in Multiple Frames based on Text Tracking. IEEE International Conference on Multimedia & Expo (ICME), 2014.

5. X. Yang, S. Yuan, and Y. Tian. Assistive Clothing Pattern Recognition for Visually Impaired People. IEEE Transactions on Human-Machine Systems (THMS), 44(2), 234-243, 2014.

6. X. Yang and Y. Tian. Polynormal Fisher Vector for Activity Recognition from Depth Sequence. SIGGRAPH ASIA Workshop on Autonomous Virtual Humans and Social Robots, 2014.

7. Y. Xian, X. Rong, X. Yang, and Y. Tian. CCNY at TRECVID 2014: Surveillance Event Detection. NIST TRECVID Workshop, 2014.

8. X. Yang and Y. Tian. Effective 3D Action Recognition Using EigenJoints. Journal of Visual Communication and Image Representation (JVCIR), 25(1), 2-11, 2014.

9. X. Yang, Z. Liu, E. Zavesky, D. Gibbon, B. Shahraray, and Y. Tian. AT&T Research at TRECVID 2013: Surveillance Event Detection. NIST TRECVID Workshop, 2013.

10. C. Zhang, X. Yang, and Y. Tian. Histogram of 3D Facets: A Characteristic Descriptor for Hand Gesture Recognition. IEEE International Conference on Automatic Face and Gesture Recognition (FG), 2013. (Oral)

11. C. Mazuera, X. Yang, and Y. Tian. Visual Speech Learning Using Dynamic Lip Movement based Video Segmentation and Comparison. IEEE International Conference on Bioinfomatics and Biomedicine (BIBM), 2013. (Oral)

12. X. Yang and Y. Tian. Texture Representations Using Subspace Embeddings. Pattern Recognition Letters (PRL), 34(10), 1130-1137, 2013.

13. X. Yi, X. Yang, and Y. Tian. Feature Representations for Scene Text Character Recognition: A Comparative Study. International Conference on Document Analysis and Recognition (ICDAR), 2013.

14. C. Zhang, X. Yang, C. Yi, Y. Tian, Q. Yu, A. Tamrakar, A. Divakaran. CCNY-SRI at TRECVID 2013 intED: A Human Interactive Event Detection System. NIST TRECVID Workshop, 2013.

15. Y. Tian, X. Yang, C. Yi, and A. Arditi. Toward A Computer Vision based Wayfinding Aid for Blind Persons to Access Unfamiliar Indoor Environments. Machine Vision and Applications (MVA), 24(3), 521-535, 2013.

16. F. Zaman, X. Yang, and Y. Tian. Monitoring Activity of Taking Medicine by Incorporating RFID and Video Analysis. Network Modeling Analysis in Health Informatics and Bioinformatics, 2(2), 61-70, 2013.

17. S. Wang, X. Yang, and Y. Tian. Detecting Sinage and Doors for Blind Navigation and Wayfinding. Network Modeling Analysis in Health Informatics and Bioinformatics, 2(2), 81-93, 2013.

18. X. Yang, C. Zhang, and Y. Tian. Recognizing Actions Using Depth Motion Maps based Histograms of Oriented Gradients. ACM Multimedia (MM), 2012.

19. X. Yang, C. Yi, L. Cao, and Y. Tian. MediaCCNY at TRECVID 2012: Surveillance Event Detection. NIST TRECVID Workshop, 2012.

20. X. Yang and Y. Tian. EigenJoints based Action Recognition Using Naive-Bayes-Nearest-Neighbor. IEEE CVPR Workshop on Human Activity Understanding from 3D Data, 2012.

21. F. Zaman, X. Yang, and Y. Tian. Robust and Effective Component based Banknote Recognition for the Blind. IEEE Transactions on System, Man, and Cybernetics (TSMC) Part C, 42(6), 1021-1030, 2012.

22. X. Yang, S. Yuan, and Y. Tian. Recognizing Clothes Patterns for Blind People by Confidence Margin based Feature Combination. ACM Multimedia (MM), 2011.

23. F. Zaman, X. Yang, Y. Tian. Robust and Effective Component based Banknote Recognition by SURF Features. IEEE Wireless and Optical Communications Conference (WOCC), 2011.

24. X. Yang, Y. Tian, C. Yi, and A. Arditi. Context based Indoor Object Detection as An Aid to Blind Persons Accessing Unfamiliar Environments. ACM Multimedia (MM), 2010.

25. X. Yang and Y. Tian. Robust Door Detection in Unfamiliar Environments by Combining Edge and Corner Features. IEEE CVPR Workshop on Computer Vision Applications for Visually Impaired, 2010.

26. Y. Tian, X. Yang, and A. Arditi. Computer Vision based Door Detection for Accessibility of Unfamiliar Environments to Blind Persons. International Conference on Computers Helping People with Special Needs (ICCHP), 2010.

References

[1] S. Arya and H. Fu. Expected Case Complexity of Approximate Nearest Neighbor Searching. In *Symposium of Discrete Algorithms*, 2000. 39

[2] H. Bay, A. Ess, and L. Gool. SURF: Speed Up Robust Features. *Computer Vision and Image Understanding*, 2008. 6, 7

[3] S. Bhattacharya, R. Sukthankar, R. Jin, and M. Shah. A Probabilistic Representation for Efficient Large-Scale Visual Recognition Tasks. In *Computer Vision and Pattern Recognition*, 2011. 32

[4] A. Bobick and J. Davis. The Recognition of Human Movement Using Temporal Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2001. 2, 6, 7, 35, 51

[5] O. Boiman, E. Shechtman, and M. Irani. In Defense of Nearest Neighbor based Image Classification. In *Computer Vision and Pattern Recognition*, 2008. 35, 39

[6] Y. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning Mid-Level Features for Recognition. In *Computer Vision and Pattern Recognition*, 2010. 62

[7] W. Brendel and S. Todorovic. Activities as Time Series of Human Postures. In *European Conference on Computer Vision*, 2010. 32

[8] S. Chang. The Holy Grail of Content based Media Analysis. *IEEE Multimedia*, 2002. 1

[9] H. Cheng and J. Hwang. Integrated Video Object Tracking with Applications in Trajectory based Event Detection. *Journal of Visual Communication and Image Representation*, 2011. 33

[10] A. Coates and A. Ng. The Importance of Encoding versus Training with Sparse Coding and Vector Quantization. In *International Conference on Machine Learning*, 2011. 23, 64

[11] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Computer Vision and Pattern Recognition*, 2005. 51, 53

[12] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior Recognition via Sparse Spatio-Temporal Features. In *VS-PETS*, 2005. 2, 33, 50

[13] C. Ellis, S. Masood, M. Tappen, J. Laviola, and R. Sukthankar. Exploring the Trade-Off between Accuracy and Observational Latency in Action Recognition. *International Journal on Computer Vision*, 2013. 36, 41, 47

[14] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 2008. 6, 12, 26, 67

[15] R. Feris, A. Hampapur, Y. Zhai, R. Bobbitt, L. Brown, D. Vaquero, Y. Tian, H. Liu, and M. Sun. Case Study: IBM Smart Surveillance System. *Intelligent Video Surveillance: Systems and Technology*, 2009. 2

[16] D. Forsyth, O. Arikan, L. Ikemoto, J. O'Brien, and D. Ramanan. Computational Studies of Human Motion: Part 1, Tracking and Motion Synthesis. *Found. Trends. Comput. Graph. Vis.*, 2005. 2

[17] W. Freeman, P. Beardsley, H. Kage, K. Tanaka, K. Kyuma, and C. Weissman. Computer Vision for Computer Interaction. In *SIGGRAPH*, 2000. 2

[18] J. Gemert, C. Veenman, A. Smeulders, and J. Geusebroek. Visual Word Ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009. 20

[19] R. Gishick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient Regresssion of General Activity Human Poses from Depth Images. In *International Conference on Computer Vision*, 2011. 36

[20] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as Space-Time Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007. 2

[21] O. Gross, Y. Gurovich, T. Hassner, and L. Wolf. Motion Interchange Patterns for Action Recognition in Unconstrained Videos. In *European Conference on Computer Vision*, 2012. 32

[22] H. Gunes and M. Piccardi. Automatric Temporal Segment Detection and Affect Recognition from Face and Body Display. *IEEE Transactions on Systems, Man, and Cybernetics (Part B)*, 2009. 35, 39, 55

[23] S. Hadfield and R. Bowden. Hollywood 3D: Recognizing Actions in 3D Natural Scenes. In *Computer Vision and Pattern Recognition*, 2013. 2, 60

[24] L. Han, X. Wu, W. Liang, G. Hou, and Y. Jia. Discriminative Human Action Recognition in the Learned Hierarchical Manifold Space. *Image and Vision Computing*, 2010. 34, 50

[25] C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Alvey Vision Conference*, 1988. 7

[26] T. Jaakkola and D. Haussler. Exploiting Generative Models in Discriminative Classifiers. In *Neural Information Processing Systems*, 1998. 23

[27] M. Jain, H. Jegou, and P. Bouthemy. Better Exploiting Motion for Better Action Recognition. In *Computer Vision and Pattern Recognition*, 2013. 32

[28] H. Jegou, M. Douze, C. Schmid, and P. Perez. Aggregating Local Descriptors into a Compact Image Representation. In *Computer Vision and Pattern Recognition*, 2010. 17, 62

[29] S. Ji, W. Xu, M. Yang, and K. Yu. 3D Convolutional Neural Netwwork for Human Action Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012. 4

[30] Y. Jiang, Q. Dai, X. Xue, W. Liu, and C. Ngo. Trajectory-based Modeling of Human Actions with Motion Reference Points. In *European Conference on Computer Vision*, 2012. 32

[31] G. Johansson. Visual Perception of Biological Motion and A Model for Its Analysis. *Attention, Perception, Psychophysics*, 1973. 34

[32] A. Klaser, M. Marszaek, and C. Schmid. A Spatio-Temporal Descriptor based on 3D Gradients. In *British Machine Vision Conference*, 2008. 2, 33

[33] J. Krapac, J. Verbeek, and F. Jurie. Modeling Spatial Layout with Fisher Vector for Image Categorization. In *International Conference on Computer Vision*, 2011. 21

[34] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: A Large Video Database for Human Motion Recognition. In *Computer Vision and Pattern Recognition*, 2011. vii, 25, 26

[35] A. Kurakin, Z. Zhang, and Z. Liu. A Real-Time System for Dynamic Hand Gesture Recognition with A Depth Sensor. In *European Signal Processing Conference*, 2012. 77

[36] K. Lai, L. Bo, X. Ren, and D. Fox. A Large-Scale Hierarchical Multi-View RGB-D Object Dataset. In *International Conference on Robotics and Automation*, 2011. 2

[37] I. Laptev. On Space-Time Interest Points. *International Journal of Computer Vision*, 2005. 2, 5, 7, 33, 36, 50

[38] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning Realistic Human Actions from Movies. In *Computer Vision and Pattern Recognition*, 2008. 2, 9, 17, 18, 21, 25, 27, 36, 50, 59, 65

[39] S. Lazebnik, C. Schmid, and J. Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *Computer Vision and Pattern Recognition*, 2006. 2, 9, 18, 21

[40] Q. Le, W. Zou, S. Yeung, and A. Ng. Learning Hierarchical Invariant SpatioTemporal Features for Action Recognition with Independent Subspace Analysis. In *Computer Vision and Pattern Recognition*, 2011. 32

[41] H. Lee, R. Grosse, R. Ranganath, and A. Ng. Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representation. In *International Conference on Machine Learning*, 2009. 62

[42] W. Li, Z. Zhang, and Z. Liu. Action Recognition based on a Bag of 3D Points. In *CVPR Workshop on Human Communicative Behavior Analysis*, 2010. 2, 35, 36, 41, 44, 51, 54, 56, 59, 67, 70, 76

[43] J. Liu, J. Luo, and M. Shah. Recognizing Realistic Actions from Videos in the Wild. In *Computer Vision and Pattern Recognition*, 2009. viii, 2, 25, 27

[44] L. Liu and L. Shao. Learning Discriminative Representations from RGB-D Video Data. In *International Joint Conference on Artificial Intelligence*, 2013. 75, 78

[45] L. Liu, L. Wang, and X. Liu. In Defense of Soft-Assignment Coding. In *International Conference on Computer Vision*, 2011. 6, 9, 17, 20

[46] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online Dictionary Learning for Sparse Coding. In *International Conference on Machine Learning*, 2009. 6, 18, 20, 28, 59

[47] S. McCann and D. Lowe. Local Naive Bayes Nearest Neighbor for Image Classification. In *Computer Vision and Pattern Recognition*, 2012. 39

[48] S. McCann and D. Lowe. Spatially Local Coding for Object Recognition. In *Assian Conference on Computer Vision*, 2012. 21

[49] T. Moeslund, A. Hilton, and V. Kruger. A Survey of Advances in Vision based Human Motion Capture and Analysis. *Compuer Vision and Image Understanding*, 2007. 1

[50] N., Ikizler-Cinbis, and S. Sclaroff. Object, Scene, and Actions: Combining Multiple Features for Human Action Recognition. In *European Conference on Computer Vision*, 2010. 32

[51] O. Oreifej and Z. Liu. HON4D: Histogram of Oriented 4D Normals for Activity Recognition from Depth Sequences. In *Computer Vision and Pattern Recognition*, 2013. 2, 58, 59, 60, 61, 65, 67, 70, 72, 74, 75, 76, 77, 78

[52] X. Peng, Y. Qiao, Q. Peng, and X. Qi. Exploring Motion Boundary based Sampling and Spatio-Temporal Context Descriptors for Action Recognition. In *British Machine Vision Conference*, 2013. 32

[53] F. Perronnin, J. Sanchez, and T. Mensink. Improving the Fisher Kernel for Large-Scale Image Classification. In *European Conference on Computer Vision*, 2010. 2, 17, 21, 25, 28, 59, 62

[54] X. Ren, D. Fox, and K. Konolige. Change Their Perception: RGB-D for 3-D Modeling and Recognition. *IEEE Robotics and Automation Magazine*, 2013. 2

[55] J. Sanchez, F. Perronnin, and T. Campos. Modeling the Spatial Layout of Images Beyond Spatial Pyramids. *Pattern Recognition Letters*, 2012. 21

[56] J. Sanchez, F. Perronnin, T. Mensink, and J. Verbeek. Image Classification with the Fisher Vector: Theory and Practice. *International Journal on Computer Vision*, 2013. 17, 21, 64

[57] S. Sarkar, P. Phillips, Z. Liu, I. Vega, P. Grother, and K. Bowyer. The Human ID Gait Challenge Problem: Datasets, Performance, and Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005. 2

[58] K. Schindler and L. Gool. Action Snippets: How Many Frames Does Human Action Require? In *Computer Vision and Pattern Recognition*, 2008. 36

[59] L. Seidenari, V. Varano, S. Berretti, A. Bimbo, and P. Pala. Recognizing Actions from Depth Cameras as Weakly Aligned Multi-Part Bag-of-Poses. In *Computer Vision and Pattern Recognition Workshop*, 2013. 75, 78

[60] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-Time Pose Recognition in Parts from Single Depth Images. In *Computer Vision and Pattern Recognition*, 2011. 34, 36, 37, 52, 66

[61] N. Silberman, D. Hoiem, P. Kolhi, and R. Fergus. Indoor Segmentation and Support Inference from RGBD Images. In *European Conference on Computer Vision*, 2012. 2

[62] A. Smeaton, P. Over, and W. Kraaij. Evaluation Campaigns and TRECVID. In *ACM Workshop on Multimedia Information Retrieval*, 2006. 4

[63] J. Sun, X. Wu, S. Yan, L. Cheong, T. Chua, and J. Li. Hierarchical Spatio-Temporal Context Modeling for Action Recognition. In *Computer Vision and Pattern Recognition*, 2009. 33, 36, 50

[64] M. Sun and J. Shotton. Conditional Regression Forests for Human Pose Estimation. In *Computer Vision and Pattern Recognition*, 2012. 36

[65] J. Sung, C. Ponce, B. Selman, and A. Saxena. Unstructured Human Activity Detection from RGBD Images. In *International Conference on Robotics and Automation*, 2012. 36, 41, 46, 47

[66] S. Tang, X. Wang, T. Han, J. Keller, M. Skubic, S. Lao, and Z. He. Histogram of Oriented Normal Vectors for Object Recognition with a Depth Sensor. In *Asian Conference on Computer Vision*, 2013. 60

[67] Y. Tian, L. Cao, Z. Liu, and Z. Zhang. Hierarchical Filtered Motion for Action Recognition in Crowded Videos. *IEEE Transactions on Systems, Man, and Cybernetics (Part C)*, 2011. 6, 33, 51

[68] A. Vedaldi and A. Zisserman. Efficient Additive Kernels via Explicit Feature Maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012. 6, 11

[69] A. Vieira, E. Nascimento, G. Oliveria, Z. Liu, and M. Campos. STOP: Space-Time Occupancy Patterns for 3D Action Recognition from Depth Map Sequences. In *Process in Pattern Recognition, Image Analysis, Computer Visin, and Applications*, 2012. 70, 76

[70] C. Wang, Y. Wang, and A. Yuille. An Approach to Pose based Action Recognition. In *Computer Vision and Pattern Recognition*, 2013. 60, 70, 71, 76

[71] H. Wang, A. Klaser, C. Schmid, and C. Liu. Dense Trajectories and Motion Boundary Descriptors for Action Recognition. *International Journal of Computer Vision*, 2013. 2, 27, 30, 31, 32, 33, 50, 59, 65, 67

[72] H. Wang and C. Schmid. Action Recognition with Improved Trajectories. In *International Conference on Computer Vision*, 2013. 2

[73] H. Wang, M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of Local SpatioTemporal Features for Action Recognition. In *British Machine Vision Conference*, 2009. 2

[74] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3D Action Recognition with Random Occupancy Patterns. In *European Conference on Computer Vision*, 2012. 2, 59, 60, 67, 70, 71, 76, 77

[75] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining Actionlet Ensemble for Action Recognition with Depth Cameras. In *Computer Vision and Pattern Recognition*, 2012. 2, 59, 60, 65, 67, 70, 71, 72, 73, 75, 76, 77, 78

[76] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-Constrained Linear Coding for Image Classification. In *Computer Vision and Pattern Recognition*, 2010. 6, 17

[77] X. Wang, L. Wang, and Y. Qiao. A Comparative Study of Encoding, Pooling, and Normalization Methods for Action Recognition. In *Assian Conference on Computer Vision*, 2012. 21, 32

[78] D. Weinland, R. Ronfard, and E. Boyer. Free Viewpoint Action Recognition Using Motion History Volumes. *Computer Vision and Image Understanding*, 2006. 2

[79] G. Willems, T. Tuytelaars, and L. Van Gool. An Efficient Dense and Scale-Invariant Spatio-Temporal Interest Point Detector. In *European Conference on Computer Vision*, 2008. 2

[80] L. Xia and J. Aggarwal. Spatio-Temporal Depth Cuboid Similarity Feature for Activity Recognition Using Depth Camera. In *Computer Vision and Pattern Recognition*, 2013. 2, 58, 60, 75, 76

[81] L. Xia, C. Chen, and J. Aggarwal. View Invariant Human Action Recognition Using Histograms of 3D Joints. In *CVPR Workshop on Human Activity Understanding fomr 3D Data*, 2012. 36, 41, 44, 70, 71, 76

[82] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear Spatial Pyramid Matching Using Sparse Coding for Image Classification. In *Computer Vision and Pattern Recognition*, 2009. 2, 17

[83] X. Yang, Z. Liu, E. Zavesky, D. Gibbon, B. Shahraray, and Y. Tian. AT&T Research at TRECVID 2013: Surveillance Event Detection. In *NIST TRECVID Workshop*, 2013. 4

[84] X. Yang and Y. Tian. EigenJoints based Action Recognition Using Naive Bayes Nearest Neighbor. In *CVPR Workshop on Human Activity Understanding from 3D Data*, 2012. 2, 70, 71, 75, 78

[85] X. Yang and Y. Tian. Action Recognition Using Super Sparse Coding Vector with Spatio-Temporal Awareness. In *European Conference on Computer Vision*, 2014. 7, 33

[86] X. Yang and Y. Tian. Effective 3D Action Recognition Using EigenJoints. *Journal of Visual Communication and Image Representation*, 2014. 51, 56, 60, 76

[87] X. Yang and Y. Tian. Super Normal Vector for Activity Recognition Using Depth Sequences. In *Computer Vision and Pattern Recognition*, 2014. 2

[88] X. Yang, C. Yi, L. Cao, and Y. Tian. MediaCCNY at TRECVID 2012: Surveillance Event Detection. In *NIST TRECVID Workshop*, 2012. 6

[89] X. Yang, C. Zhang, and Y. Tian. Recognizing Actions Using Depth Motion Maps based Histograms of Oriented Gradients. In *ACM Multimedia*, 2012. 2, 60, 72, 75, 76, 77, 78

[90] J. Yuan, Z. Liu, and Y. Wu. Discriminative Subvolume Search for Efficient Action Detection. In *Computer Vision and Pattern Recognition*, 2009. 7

[91] M. Zanfir, M. Leordeanu, and C. Sminchisescu. The Moving Pose: an Efficient 3D Kinematics Descriptor for Low-Latency Action Recognition and Detection. In *International Conference on Computer Vision*, 2013. 59, 60, 70, 71, 75, 76, 78

[92] C. Zhang, X. Yang, and Y. Tian. Histogram of 3D Facets: A Characteristic Descriptor for Hand Gesture Recognition. In *International Conference on Automatic Face and Gesture Recognition*, 2013. 59

[93] X. Zhou, K. Yu, T. Zhang, and T. Huang. Image Classification Using Super-Vector Coding of Local Image Descriptors. In *European Conference on Computer Vision*, 2010. 17, 62