Ph.D Dissertation

View Invariant Action Recognition Using Generalized 4D Motion Features

일반화된 4차원 동작 특징을 이용한 시선각에 무관한 행동 인식

By

Sun Jung Kim

August 2014

SCHOOL OF ELECTRICAL ENGINEERING AND

COMPUTER SCIENCE

COLLEGE OF ENGINEERING

SEOUL NATIONAL UNIVERSITY

Abstract

In this thesis, we propose a method to recognize human action and their orientation independently of viewpoints using generalized 4D [x,y,z,t] motion features. The conventional action recognition methods assume that the camera view is fixed and people are standing towards the cameras. However, in real life scenarios, the cameras are installed at various positions for their purposes and the orientation of people are chosen arbitrarily. Therefore, the images can be taken with various views according to the position of camera and the orientation of people. To recognize human action and their orientation under this difficult scenario, we focus on the view invariant action recognition method which can recognize the test videos from any arbitrary view.

For this purpose, we propose a method to recognize human action and their orientation independently of viewpoints by developing 4D space-time interest points (4D-STIPs, [x,y,z,t]) using 3D space (3D-S, [x,y,z]) volumes reconstructed from images of a finite number of different views. Since the 3D-S volumes and the 4D-STIPs are constructed using volumetric information, the features for arbitrary 2D space (2D-S, [x,y]) viewpoint can be generated by projecting the 3D-S volumes and 4D-STIPs on corresponding test image planes. With these projected features, we construct motion history images (MHIs) and non-motion history images (NMHIs) which encode the moving and non-moving parts of an action respectively. Since MHIs cannot guarantee a good performance when moving parts of an object show similar patterns, we propose NMHIs and combine it with MHIs to add the information from stationary parts of an object in the description of the particular action class. To reduce the dimension of MHIs and NMHIs, we apply class-augmented principal component analysis (CA-PCA) which uses class

information for dimension reduction. Since we use the action label for reducing the dimension of features, we obtain the principal axis which can separate each action well. After reducing the feature dimension, the final features are trained by support vector data description method (SVDD) and tested by support vector domain density description (SVDDD). As for the recognition of action orientation, the features are reduced the dimension using orientation label. Similarly, the reduced features are trained by SVDD and tested by SVDDD.

The proposed 4D-STIPs can be applied to view invariant recognition of action and their orientation, and we verify that they represent the properties of each action compactly in experiments. To assume arbitrary test view as in real applications, we develop a new testing dataset which is totally different from the training dataset. We verify our algorithm by training action models using the multi-view IXMAS dataset and testing using SNU dataset. Experimental results show that the proposed method is more generalized and outperforms the state-of-the-art methods, especially when training the classifier with the information insufficient about the test views. As for the recognition of action orientation, we experiment with SNU dataset taken from 5 different orientations to verify recognition performance. The recognition of action orientation can be helpful in analyzing the video by providing the information about interactions of people.

Keywords: 4D space-time interest points, view invariant action recognition, recognition of action orientation, 3D reconstruction

Student ID Number: 2009-30180

Contents

List of Figures

List of Tables

Chapter 1

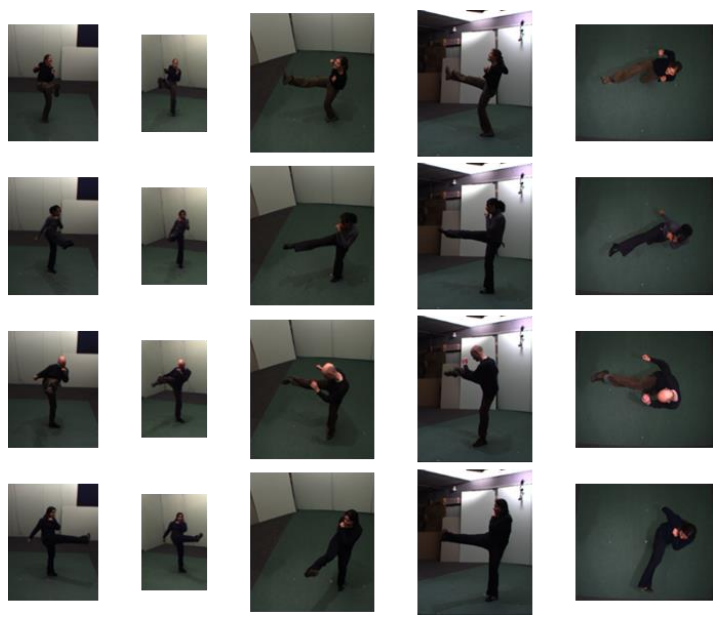
Introduction

1.1 Motivations

These days, surveillance cameras are everywhere and people do not aware of cameras. Therefore, cameras may capture people from enormously various views according to human orientation and camera positions as in Figure 1.1. However, in the conventional view dependent action recognition datasets such as KTH dataset (Figure 1.2 (a)) [1], the position of camera is fixed and people are standing towards the camera. The conventional view dependent action recognition methods [3, 4, 5, 6, 7, 8, 9, 10, 11, 12] have shown good recognition performance in the fixed camera view with the methods such as local features and bag of visual-words. However, in real human action recognition scenarios unlike dataset with artificially controlled views, not only the position of cameras but also the orientation of people can be changed as in Figure 1.2 (b) [2]. The conventional local features are highly dependent on the camera view because the information captured by local features is significantly different for different camera views. In order to achieve higher recognition rate, these methods should train classifier in

Figure 1.1: Various views. Cameras may capture people from enormously various views according to human orientation and camera positions.

(a)



(b)

Figure 1.2: Datasets for action recognition. (a) The KTH dataset for view dependent action recognition. The position of camera is fixed and people are standing towards the camera in the fixed position [1]. (b) The IXMAS dataset for view invariant action recognition. The dataset is taken from various camera views and the position and the orientation of actor are chosen arbitrarily [2].

the same view where the testing will be carried out. Therefore, the model trained on the dataset made in artificially controlled views is of no use in practical applications. Despite of new advancements in the action recognition, the gap between its current capabilities and the use in real life applications remains large. The substantial variations of camera positions and human orientation in the video data make the action recognition a challenging task and limit the applicability of the conventional methods. Hence we focus on the view invariant action recognition which is applicable to real life scenarios by handling the test videos from any arbitrary view.

The action recognition research can be categorized largely into two cases of gesture and general action recognition. The gesture recognition is usually for human-computer interaction in predefined settings. Most actual applications for gesture recognition assume human and computer face each other and the recognition in front view images is performed. However, recognition of actions in general settings such as surveillance environments requires view invariant recognition. This is because the orientation of human in the surveillance camera vary significantly. The substantial variations of human pose in surveillance scenario make the action recognition a challenging task and limit the applicability of the conventional methods. To solve view invariant action recognition problem in general condition, various methods are proposed and those methods can be separated as 3D space (3D-S, [x,y,z]) and 2D space (2D-S, [x,y]) approaches. In recognition step, 3D-S approach requires special settings such as multiple number of cameras or RGB-D sensors, whereas 2D-S approach uses just 2D-S images from a single camera.

In the 3D-S approach, several test images from multiple views or the RGB-D sensors to measure depth information are required. [2] proposed the motion history volumes which requires multiple view images in testing step. The methods

in [13, 14] also required images from multiple cameras for 3D-S pose estimation and [15] proposed multi-sensor fusion strategy. But for obtaining test images from multiple views, those many cameras have to be deployed to overlap a target region at all times, which is too expensive and so impractical in most of the situations such as wide area surveillance. Recently, many approaches for action recognition using RGB-D sensor were proposed in [16, 17, 18, 19, 20, 21]. The RGB-D sensor offers discerning information by providing 3D-S structural information. However, these methods require additional depth sensor which could not work well in outdoor or at long distance. In addition, they assume the action is captured from the front view. Even when using RGB-D sensors, view invariant recognition still requires multiple view information because the depth information from a single RGB-D sensor could not cover the information in the opposite side.

As for the 2D-S approach, the research has been conducted in three directions: 1) designing of view invariant features, 2) designing a classifier based on transfer learning, and 3) learning of features from multiple views. As for the first issue, [22] proposed a view invariant feature by using the properties that the significant changes of direction in 3D-S motion trajectory are preserved. [23] developed a quasi view invariant approach by representing each action as a unique curve using trajectories of body-joints. [24] introduced the concept of fundamental ratios having view invariant characteristics. [25, 26] proposed a view invariant feature using temporal self-similarities. However, the descriptors of this scheme are inherently limited to be strictly view invariant and could not work well when the view changes on a large scale. Furthermore, view invariant features can not recognize the orientation of action since most of existing view invariant features are still insufficient for making features used commonly for all the views.

To cope with the inherent limitation of view invariant features, transfer learning concept were reported. Recently many variants of the transfer learning ap-

proach have been developed in [27, 28, 29, 30, 31, 32, 33]. In these methods, source and target views were defined; source view is used for training purpose, whereas the target view is for testing phase. This transfer learning approach has an impractical assumption that source and target views are defined in advance. [28, 29, 30, 33] suggested a visual dictionary based on features obtained from both predefined source and target views. For association between source and target views, the visual dictionary has been constructed using features extracted from both predefined source and target views. In this approach, the classifier is trained with visual words from the source view, but the visual dictionary is made using videos from the target view as well as the source view. Hence, this approach could not guarantee the performance for the new view which does not participate in the process making visual dictionary. This implies the transfer learning approach is limited to a set of predefined views and so it does not provide arbitrary view invariant property. Furthermore, those methods can not recognize the orientation of action, since they do not consider the orientation information when they make visual dictionary.

To achieve the complete view invariant action recognition using single camera, all the features from all of views should be accumulated. Since it is not possible to obtain information from all of views in practice, most of the existing methods deployed a finite number of views to generalize the recognition ability [34, 35, 36, 37, 38, 39]. [34] represented an image sequence using bag of spatio-temporal features called video-words, and [35] made 3D space-time (3D-ST, [x,y,t]) volumes by concatenating the actor's 2D-S silhouettes. [36] proposed feature-tree to index large scale 3D-ST features and [37] computed the features with 3D-ST histogram of oriented gradients (HOG) obtained from the image sequence of multiple views. [38] proposed action recognition method based on the local motion signatures by tracking 2D-S HOG. [39] formulated the view label of action as a latent variable in

latent kernelized structural support vector machine (SVM) and implicitly infer it during both learning and inference. However, these methods still do not guarantee a satisfying performance on the new test view which is not contained in the training. To achieve a satisfactory result, acquiring data from different views using a large number of cameras is required. Furthermore, the methods accumulating features from finite number of views can not recognize the orientation of action since they consider the same action from different views as the same class in training step.

1.2 Contents of the research

We live in a three-dimensional [x,y,z] world, however, images taken are two-dimensional [x,y]. The information such as depth is not available in the 2D-S images (Figure 1.3 (a)). To recognize actions view invariantly with 2D-S images, the best solution is to use the videos from all possible views as in Figure 1.3 (b), however, it is impractical to deploy infinite number of cameras. Otherwise, 3D-S volumes have complete three-dimensional information (Figure 1.4 (a)). Therefore, if we have 3D-S volumes, we can generate all possible views by projecting the 3D-S volumes into 2D-S image planes (Figure 1.4 (b)). My thesis starts from the motivation that we can utilize 3D-S volumes for view invariant action recognition rather than trying to understand 3D-S world with 2D-S images. For this purpose, we first reconstruct 3D-S volumes by computing the visual hulls via back projecting the multi-view 2D-S silhouettes [40] and calculate generalized 4D space-time (4D-ST, [x,y,z,t]) motion features from 3D-S volumes. With 3D-S volumes and 4D space-time interest points (4D-STIPs, [x,y,z,t]) calculated from 3D-S volumes, we solve the view invariant action recognition problem.

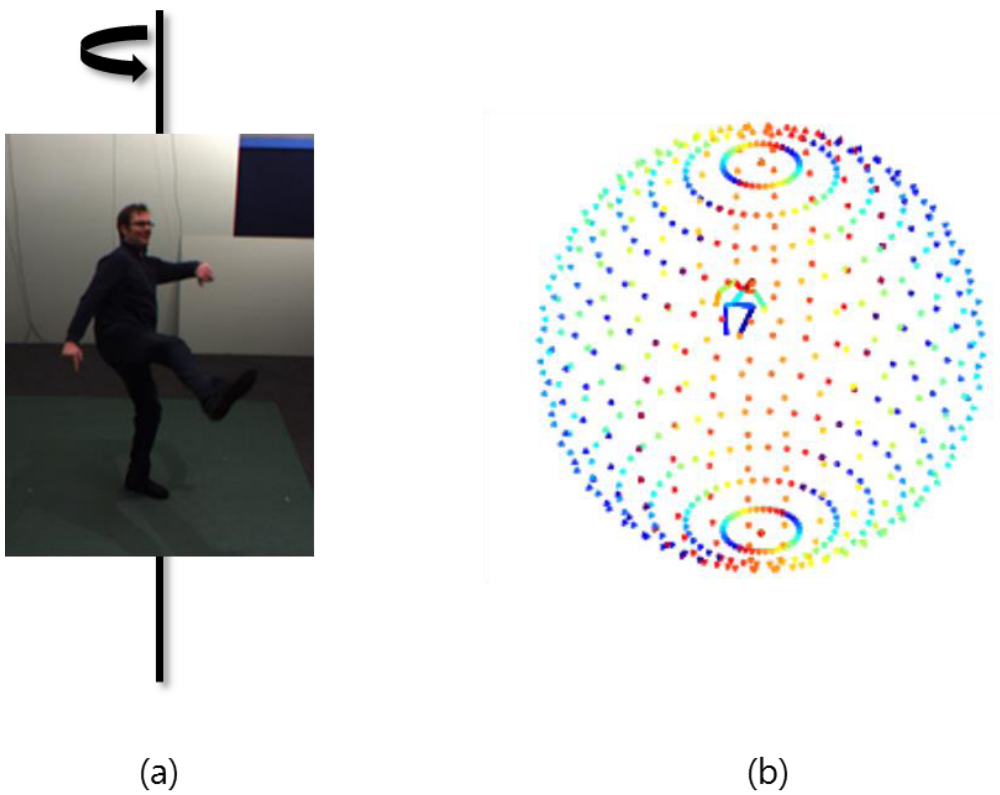


Figure 1.3: 2D-S images. (a) The information such as depth is not available in 2D-S images [2]. (b) All possible views around a person in three-dimensional space [26].

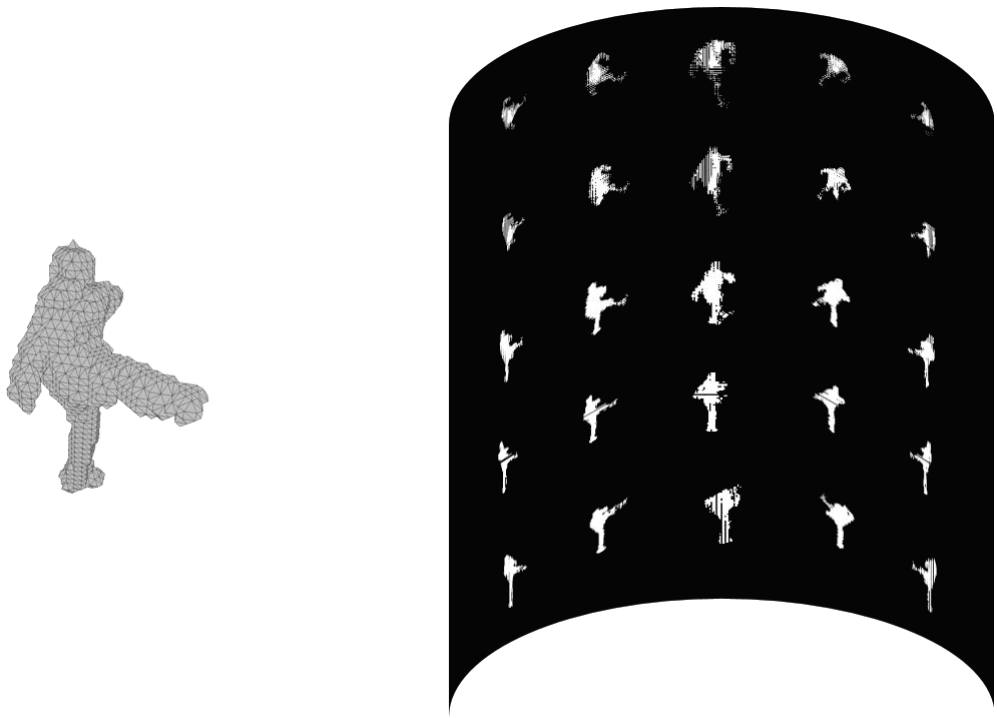


Figure 1.4: 3D-S volume. (a) 3D-S volume for view invariant action recognition [2]. (b) We can generate all possible views by projecting the 3D-S volumes into 2D-S image planes.

1.2.1 Generalized 4D motion features

The system for action recognition can be divided into two parts of vision and machine learning [41]. We represent posture or extract local features from videos [42, 43, 44, 45, 46, 47, 48] in the vision part. In the machine learning part, we develop action models by mapping posture or local features into meaningful motion classes [49, 50, 51, 52, 53, 5, 54, 55, 56, 57]. We propose a method of extracting local features from 3D-S volumes for view invariant action recognition.

In the 2D-S domain (2D-S image), interest points with a significant local variation of image intensities have been extensively investigated in [58, 59, 60, 61]. However, interest points extracted from 2D-S domain is inappropriate for reflecting characteristics of actions which are varied over time, since they do not consider the variations along the time axis. To complement this, interest points in 3D space-time [x,y,t] domain (2D-S image sequences) which show significant variations along both space [x,y] and time [t] axis have been proposed in [47, 45]. However, these 3D space-time interest points (3D-STIPs, [x,y,t]) are highly dependent on the camera view because the same actions in 2D-S image sequence captured from different camera view are significantly different. To solve this problem, we propose 4D-STIPs in 4D-ST domain (3D-S volume sequences) by extending the 3D-STIPs. Since the proposed 4D-STIPs are constructed using volumetric information, the features for arbitrary 2D-S viewpoint can be generated by projecting 4D-STIPs on corresponding image planes, which enables action recognition in any camera view point.

We first explain the basic concept of interest points by introducing the Harris corner detector [59]. Then, we introduce the method of 3D-STIPs proposed by [47] which extends the notion of Harris corner detector into the time [t] domain. We also briefly explain two approaches of 3D reconstruction (calibration-based

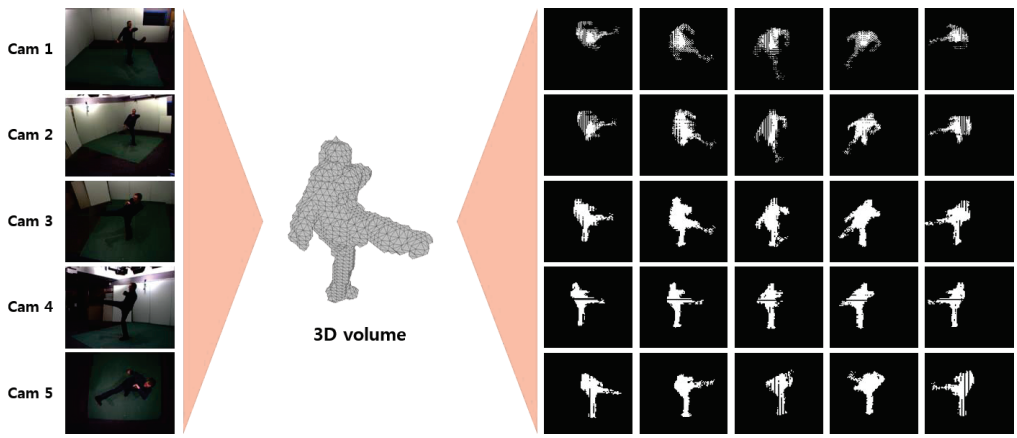


Figure 1.5: Generation of features. By projecting the features in 3D-S to the arbitrary 2D-S image planes, we can generate the features from any number of views even when just 5 cameras are used.

approach and image-based approach) for 3D-S volumes. With the 3D-S volumes reconstructed from several images of different views, we propose 4D-STIPs by extending the method based on 3D-STIPs. And we also propose the variant of 3D-STIPs which take into account the simultaneous gradient variation in all 3 dimensions [x,y,t] to focus on the motion of important spatial corner points.

1.2.2 View invariant action recognition

In this thesis, we propose a method to be able to extract 4D-ST motion features covering arbitrary views from image sequences of finite camera views (5 camera views in this thesis). Unlike the existing methods which extract features from the limited number of 2D-S images of different views, we can generate the features from any number of views even when just 5 cameras are used as in Figure 1.5. This can be done by projecting the features in 3D-S to the arbitrary 2D-S image planes. Consequently, the features in the proposed method contain characteristics at each view, while most of the existing view invariant features lose characteristics

at each view because they are obtained by extracting common properties from all views. Unlike the methods which learn features from multiple views, the proposed method shows good recognition results even when the images of test view are not available during the training step and does not need to predefine the test view as in transfer learning methods. And the proposed method assumes that just a single test video from an arbitrary view is given and, therefore, extra equipments such as multiple cameras or RGB-D sensor are not required.

We first introduce the features and machine learning algorithms used in our proposed method and then explain the proposed method. In experiments, we train and test in different datasets to evaluate the generalization performance for arbitrary view videos. The proposed method is verified to outperform the state-of-the-art methods for the images whose test view is not provided during the training step.

1.2.3 Recognition of action orientation

We propose a method to recognize the orientation of action, i.e. the orientation of human, which is not tried in view invariant action recognition field. The proposed method can recognize the orientation of actor in the test video since our training sets, which are projections of 4D-ST motion features to various image planes, contain the orientation information. On the other hand, the view invariant features can not recognize the orientation of action since most of the existing view invariant features can not cover all the information about each view for making features used commonly for all the views. Transfer learning related methods do not consider the orientation information when they make visual dictionary, and, therefore they can not recognize the orientation of action. The methods accumulating features from finite number of views can not recognize the orientation of action since they consider the same action from different views as the same class

in training step.

Since the process for making features is similar with the view invariant action recognition method, we introduce the training and testing step for recognition of action orientation. In experiments, we test the effectiveness of the proposed method using SNU dataset which was taken from 5 different orientations. The proposed method will be useful for recognizing the more complex human activities such as human-human interactions.

Chapter 2

Generalized 4D Motion Features

2.1 Introduction

Human action recognition is becoming one of core parts in computer vision technology with its wide applicability such as visual surveillance, human-computer interaction, and video retrieval. The system for action recognition can be divided into two parts of vision and machine learning [41]. In the vision part, we represent posture or extract local features for action recognition from videos [42, 43, 44, 45, 46, 47, 48]. In the machine learning part, we develop action models by mapping posture or local features into meaningful motion classes [49, 50, 51, 52, 53, 5, 54, 55, 56, 57].

Many attempts to find local feature points for action recognition were reported [45, 47]. Among them, 3D space-time interest points (3D-STIPs, [x,y,t]) [47] which is developed by extending Harris corner detector [59] were widely used in action recognition. Harris corner detector finds 2D space interest points (2D-

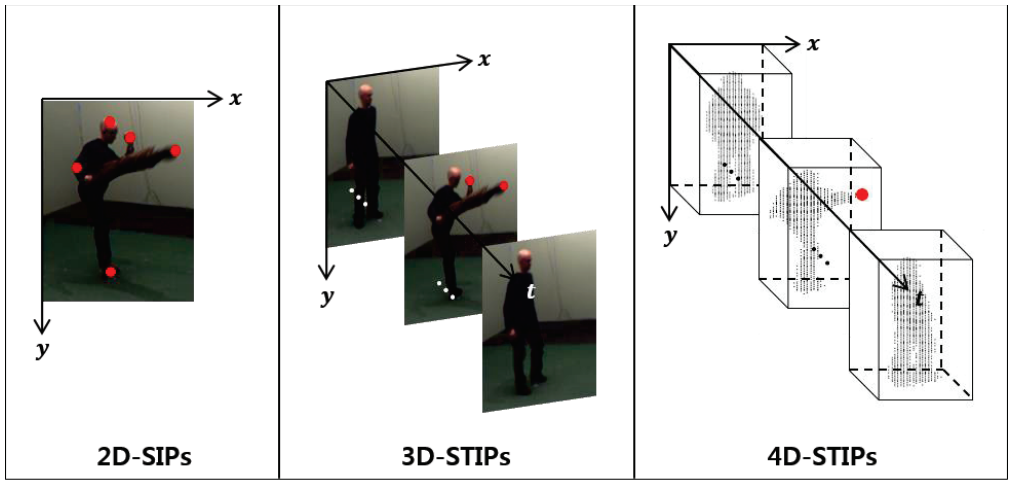


Figure 2.1: Interest points in various dimension. 2D-SIPs are extracted from one 2D-S image and therefore could not capture the important moment of varying actions. 3D-STIPs are extracted from 2D-S image sequence and developed to capture the significant variation along both space [x,y] and time [t] axis. However, 3D-STIPs are highly dependent on the camera view and the performance of classifiers using 3D-STIPs are highly degraded when the camera view varies significantly. 4D-STIPs are extracted from 3D-S volumes and, therefore, the features for arbitrary 2D-S viewpoint can be generated by projecting 4D-STIPs on corresponding image planes.

SIPs, [x,y]) such as spatial corner points in 2D space (2D-S, [x,y]) image. However, Harris corner detector is not appropriate to represent the properties of actions which vary according to the time, since it lacks of any information about time. To complement this, [47] developed 3D-STIPs having significant variation along both space [x,y] and time [t] axis in the 2D-S image sequence. However, these 3D-STIPs are highly dependent on the camera view points because the information captured from 2D-S image sequence is significantly different for different camera views. In real life scenarios, human orientation and camera positions are unknown and therefore it is necessary to research about view invariant action recognition.

We propose a method to recognize human actions independently of view-

Figure 2.2: 3D-S volume sequence [2]. 3D-S volume sequence of 'kick' action reconstructed from images of several number of different views.

points when just one 2D-S image sequence from a single camera is available. To solve this problem, we propose 4D space-time interest points (4D-STIPs, [x,y,z,t]) which extend the 3D-STIPs using 3D space (3D-S, [x,y,z]) volumes (Figure 2.2) reconstructed from images of several number of different views. Since the proposed features are constructed using volumetric information, the features for arbitrary 2D-S viewpoint can be generated by projecting 4D-STIPs on corresponding image planes and used for training step. We also propose the variant of 3D-STIPs, which take into account the simultaneous gradient variation in all 3 [x,y,t] dimension to focus on the motion of important spatial corner points. The proposed 3D-STIPs are used in recognition step since we assume that only one 2D-S image sequence from arbitrary view is available. Figure 2.1 shows differences of interest points according to the dimension. To calculate 4D-STIPs, we use 3D-S volumes from INRIA Xmas Motion Acquisition Sequences (IXMAS) [2] and experiments show that the proposed 4D-STIPs reflect the properties of each action well.

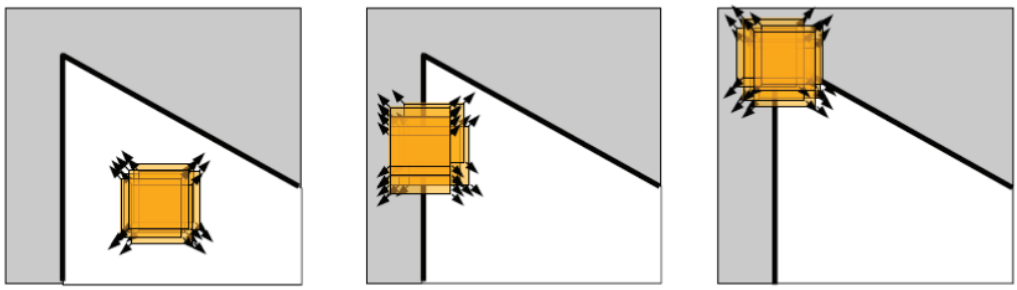


Figure 2.3: Basic idea of Harris corner detector [62]. This Figure shows the basic idea of Harris corner detector which finds corner points showing significant intensity change in all directions. At flat region, no intensity change in all direction. At edge, no intensity change along the edge direction. At corner point, significant intensity change in all directions.

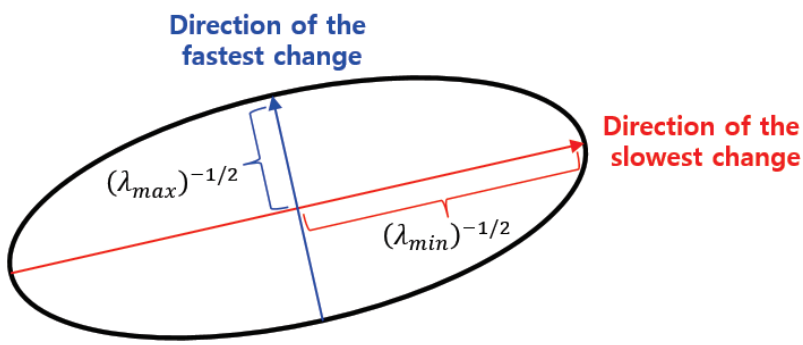


Figure 2.4: Eigenvalues of Harris corner detector [62]. We can find the corner points with eigenvalues of matrix M in equation (2.8). If the $\lambda^{-1/2}$ is small, that means the intensity shows the fastest change along that direction and if the $\lambda^{-1/2}$ is large, that means the intensity shows the slowest change along that direction.

2.2 Preliminaries

2.2.1 Harris corner detector

Our method extends the existing 3D-STIPs which build on the idea of spatial interest points, i.e. Harris corner detector [59], into the space [x,y] and time [t] domain. Therefore, in this section, we review the basic idea of Harris corner detector. The basic idea of Harris corner detector is to find corner points which show large change in intensity by shifting a window in any direction. In Figure 2.3, flat region shows no change in all directions and edge shows no change along the edge direction while corner point in an image shows significant change in all directions. The sum of squared difference between an image patch and a patch shifted by offset (u, v) can be represented by

$$E(u,v) = \sum_{x,y} w(x,y)(I(x+u, y+v) - I(x,y))^2, \tag{2.1}$$

where $w(x, y)$ is a Gaussian window function. The equation above considers only a set of shifts at every 45 degrees. To consider all small shifts, the equation is changed by Taylor's expansion as

$$E(u,v) = \sum_{x,y} w(x,y)((I_x u + I_y v + O(u^2, v^2))^2. \tag{2.2}$$

For small shifts of (u, v), we have a bilinear approximation as

$$E(u,v) \simeq Au^2 + 2Cuv + Bv^2, \tag{2.3}$$

$$A = \sum_{x,y} w(x,y) I_x^2(x,y), \tag{2.4}$$

$$B = \sum_{x,y} w(x,y) I_y^2(x,y), \tag{2.5}$$

$$C = \sum_{x,y} w(x,y) I_x(x,y) I_y(x,y). \tag{2.6}$$

Equivalently we can represent the equation as

$$E(u,v) \simeq \begin{bmatrix} u, v \end{bmatrix} M \begin{bmatrix} u \\ v, \end{bmatrix}, \tag{2.7}$$

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}. \tag{2.8}$$

To find the corner points, we modify the equation above to ellipse equation with eigenvalues of M as

$$E(u,v) \simeq \begin{bmatrix} u, v \end{bmatrix} M \begin{bmatrix} u \\ v, \end{bmatrix} = const, \tag{2.9}$$

$$E(u,v) \simeq \begin{bmatrix} u, v \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = const, \tag{2.10}$$

$$E(u,v) \simeq \frac{u^2}{(\frac{1}{\sqrt{\lambda_1}})^2} + \frac{v^2}{(\frac{1}{\sqrt{\lambda_2}})^2} = const. \tag{2.11}$$

Then we can find the corner points with eigenvalues λ_1 and λ_2. If the $\lambda^{-1/2}$ is small, that means the intensity shows the fastest change along that direction and if the $\lambda^{-1/2}$ is large, that means the intensity shows the slowest change along that direction (Figure 2.4). Therefore, we can classify image points using eigenvalues λ_1 and λ_2. If two eigenvalues are large, that point is corner point and if one of eigenvalues is large, that point is on the edge. And if both eigenvalues are small then the intensity of image around that point is almost constant in all directions.

Figure 2.5: Detected Harris corner points [62]. This Figure shows the detected Harris corner points in 2D-S image.

The measurement of corner response at each pixel is calculated as

$$R = detM - k(traceM)^2, \tag{2.12}$$

$$detM = \lambda_1 \lambda_2, \tag{2.13}$$

$$traceM = \lambda_1 + \lambda_2, \tag{2.14}$$

where k is an empirical constant ($k = 0.04 \sim 0.06$) and the threshold of R is adjusted to obtain the desired number of corner points within an image. The detected Harris corner points are shown in Figure 2.5 [62].

2.2.2 3D space-time interest points

In this section, we review 3D-STIPs [47] which is developed by extending 2D-S interest points (Harris corner detector) into the space-time [x,y,t] domain. This 3D-STIPs detect local structures in space-time where the image values have significant local variations in both space [x,y] and time [t]. The resulting interest points reflect interesting events in videos and, therefore, they are widely used in action recognition area. The basic idea is very similar with Harris corner detector. While the Harris corner detector finds space corner points in a 2D-S image, the method of 3D-STIPs finds space-time corner point in an 2D-S image sequence. To model a space-time image sequence, we use a function $f\colon R^2 \times R \longmapsto R$ and construct its linear scale-space representation $L\colon R^2 \times R \times R^2_+ \longmapsto R$ by convolution of f with an anisotropic Gaussian kernel with independent spatial variance σ_l^2 and temporal variance τ_l^2

$$L(x, y, t; \sigma_l^2, \tau_l^2) = g(x, y, t; \sigma_l^2, \tau_l^2) * f(x, y, t), \tag{2.15}$$

$$g(x, y, t; \sigma_l^2, \tau_l^2) = \frac{1}{\sqrt{(2\pi)^3 \sigma_l^4 \tau_l^2}} \exp(-\frac{(x^2 + y^2)}{2\sigma_l^2} - \frac{t^2}{2\tau_l^2}). \tag{2.16}$$

We construct linear scale-space representation to control the scale of interest points, i.e. if we convolute an image sequence with large value of variances, large scale points are detected as interest points. Using a separate scale parameter for the temporal domain is essential, since the spatial and the temporal extents of events are in general independent. Similar to Harris corner detector, we consider a space-time second-moment matrix, which is 3-by-3 matrix composed of first order

spatial and temporal derivatives averaged using a Gaussian weighting function as

$$\mu(x,y,t) = g(x,y,t;\sigma_i^2,\tau_i^2) * \begin{bmatrix} L_x^2 & L_xL_y & L_xL_t \\ L_xL_y & L_y^2 & L_yL_t \\ L_xL_t & L_yL_t & L_t^2 \end{bmatrix}, \tag{2.17}$$

$$g(x,y,t;\sigma_i^2,\tau_i^2) = \frac{1}{\sqrt{(2\pi)^3\sigma_i^4\tau_i^2}} \exp(-\frac{(x^2+y^2)}{2\sigma_i^2} - \frac{t^2}{2\tau_i^2}), \tag{2.18}$$

$$L_x = \partial_x L, L_y = \partial_y L, L_t = \partial_t L, \tag{2.19}$$

here we relate the integration scales σ_i^2 and τ_i^2 to the local scales σ_l^2 and τ_l^2 according to $\sigma_i^2 = s\sigma_l^2$ and $\tau_i^2 = s\tau_l^2$. To detect interest points, we search for regions in f having significant eigenvalues $\lambda_1, \lambda_2, \lambda_3$ of $\mu(x,y,t)$. Among different approaches to find such regions, we propose here to extend the Harris corner function defined for the spatial domain into the space-time domain by combining the determinant and the trace of μ as follows

$$R = det(\mu) - k \cdot trace(\mu)^3, \tag{2.20}$$

$$det(\mu) = \lambda_1\lambda_2\lambda_3, \tag{2.21}$$

$$trace(\mu) = \lambda_1 + \lambda_2 + \lambda_3. \tag{2.22}$$

To show how positive local maxima of R correspond to points with high values of $\lambda_1, \lambda_2, \lambda_3 (\lambda_1 \leq \lambda_2 \leq \lambda_3)$, we define the ratios $\alpha = \lambda_2/\lambda_1$ and $\beta = \lambda_3/\lambda_1$ and re-write R as

$$R = \lambda_1^3(\alpha\beta - k(1+\alpha+\beta)^3). \tag{2.23}$$

From the requirement that $R \geq 0$, we get $k \leq \alpha\beta/(1+\alpha+\beta)^3$ and it follows that k assumes its maximum possible value $k = 1/27$ when $\alpha = \beta = 1$. For sufficiently

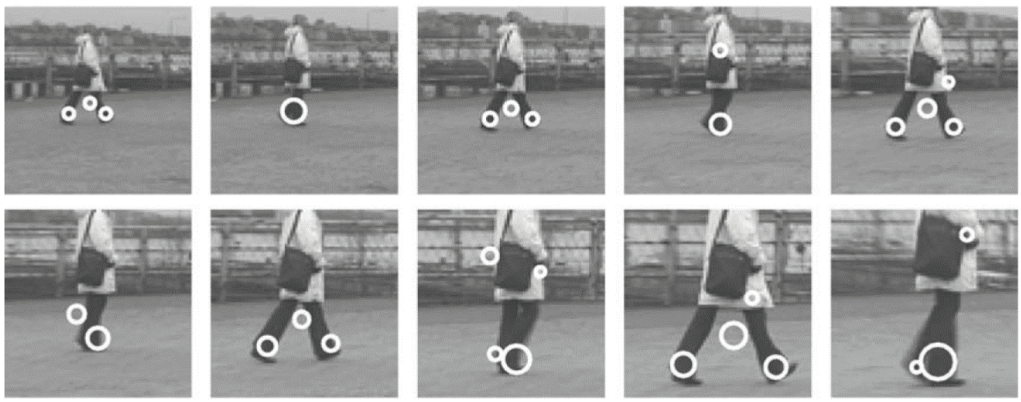


Figure 2.6: Detected 3D-STIPs [47]. This Figure shows the detected 3D-STIPs in 2D-S image sequence.

large values of k, positive local maxima of R correspond to points with high variation of the image values along both the spatial and the temporal directions. In particular, if we set the maximum value of α, β to 23 as in the spatial domain, the value of k to be used in R will be then be $k \approx 0.005$. Thus, space-time interest points of f can be found by detecting local space-time maximum in R. The detected 3D-STIPs are shown in Figure 2.6.

2.2.3 3D reconstruction

The goal of 3D reconstruction is to make a complete 3D-S object model with images taken from many camera viewpoints. The 3D reconstruction have had an enormous impact on a variety of applications including 3D-S modeling, object localization, object recognition and motion capture applications. Recently, the reconstructed 3D-S volumes are frequently used in action recognition area [2, 63, 35, 37].

Over the last years, a number of high quality algorithms have been developed [64, 65]. The algorithms can be divided into two categories whether it reconstructs

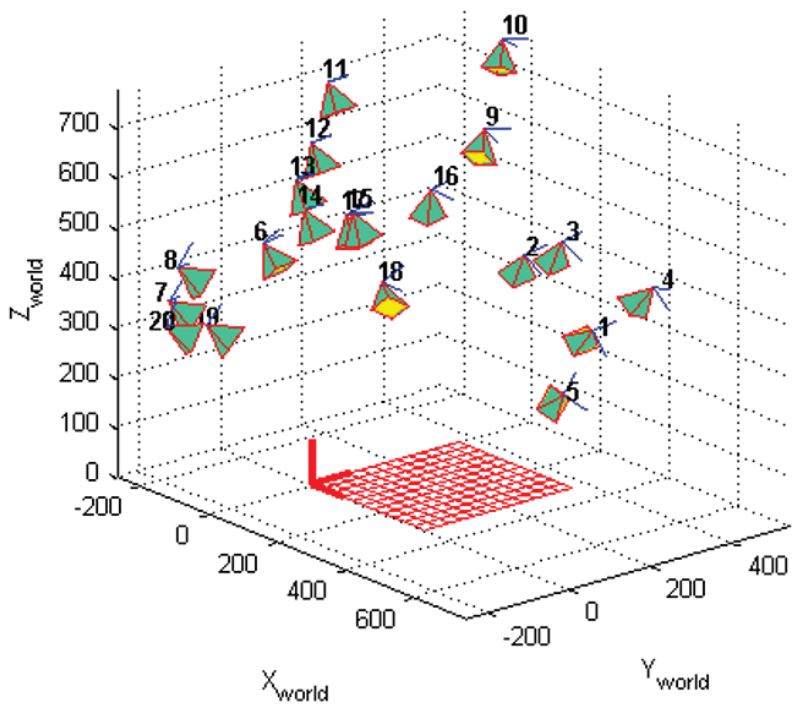


Figure 2.7: Calibration-based approach for 3D reconstruction [67]. In the calibration-based approach, we assume that several cameras are installed at fixed positions and images of a planar checkerboard are taken from those cameras.

object models from calibrated views or not. The approaches using calibrated views [66] are to start with an estimate of the silhouettes or boundaries of the object that are projected in 3D space for visual hull intersection (alternatively voxels are projected back to test if the silhouettes carve them out). The other image-based approaches [40] perform the visual hull intersection in the image plane without requiring to go in 3D space using planar homographies and foreground likelihood information from a set of arbitrary views. In this section, we will briefly introduce both categories (calibration-based and image-based approaches) for 3D reconstruction.

In the calibration-based approach, we assume that several cameras are in-

stalled at fixed positions and images of a planar checkerboard are taken from those cameras as in Figure 2.7. From [68], the point x in an image plane (corresponding to X_W in world coordinates) can be calculated by multiplying camera matrix P as

$$x = PX_W = KR[I|-\tilde{C}]X_W, \tag{2.24}$$

$$K = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.25}$$

The camera matrix P is consist of two parts of intrinsic parameter K and extrinsic parameter $R[I|-\tilde{C}]$. While the calibration matrix K contains the information about camera intrinsic parameters such as focal length f and image center of the camera (p_x, p_y), the extrinsic parameter contains rotation matrix of camera coordinates R and the inhomogeneous coordinates of the camera center $\tilde{C}$. Calibration is to find camera intrinsic and extrinsic parameters. With the grid positions of checkerboard in images of cameras, the intrinsic and the extrinsic parameters are calculated, and the camera calibration code is available on the website [67]. After calibration for all cameras are done, the object are taken from all cameras and then the silhouettes or boundaries of the object are calculated. The object is reconstructed by projecting the silhouettes or boundaries of the object in 3D space for visual hull intersection (alternatively voxels are projected back to test if the silhouettes carve them out).

In the image-based approach, instead of calibration we use planar homographies and foreground information from a set of arbitrary views. In Figure 2.8 (a), the scene is viewed from several angles with the cylinder object detected as foreground (white regions) in each view. One of the views, say I_1, is chosen as the reference view. Warping view I_i to the reference view using homography

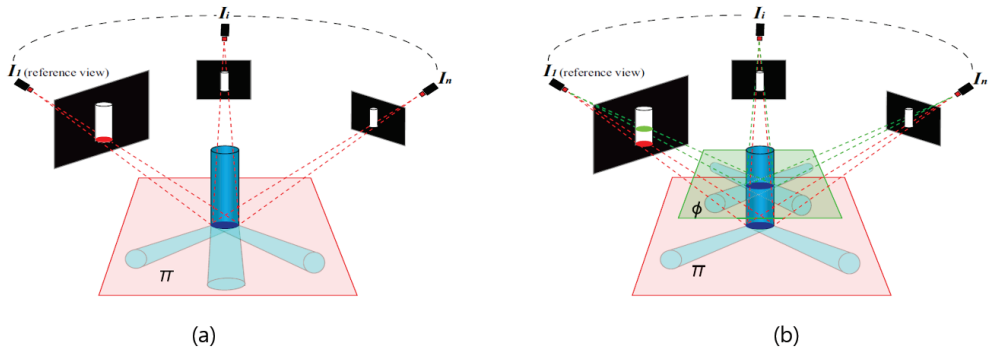


Figure 2.8: Image-based approach for 3D reconstruction [40]. (a) Warping the silhouettes of an object from the image plane to a plane in the scene using planar homography is equivalent to projecting the visual hull of the object onto the plane. The intersection of these shadows amounts to performing visual hull intersection on the plane. The result is the dark blue region that can be considered a slice of the cylinder cut out by π. (b) The same process can be performed on the second plane ϕ which delivers another slice of the cylinder.

$H_{i\pi 1}$ induced by scene plane π can be viewed as the foreground object casting a shadow on π. The shadow is then projected onto the reference view to complete the operation of the homographic warping. Clearly computing the shadow is equivalent to determining the region on π that falls inside the visual hull of the object image in I_i. The fusion of these shadows projected from various views therefore amounts to performing visual hull intersection on plane π, depicted by the dark blue region in Figure 2.8 (a). This process is performed implicitly, when we warp all the views onto the reference view and fuse them to obtain the red region in the reference view I_1. Without loss of generality, reference image plane I_1 after homographic fusion of foreground data can be viewed as a projectively transformed planar slice of the object. Starting with a reference plane in the scene (typically the ground plane), we perform visual hull intersection on successively parallel planes in the up direction along the body of object as in Figure 2.8 (b). If we have the homography $H_{i\pi j}$ induced by a reference scene plane π between

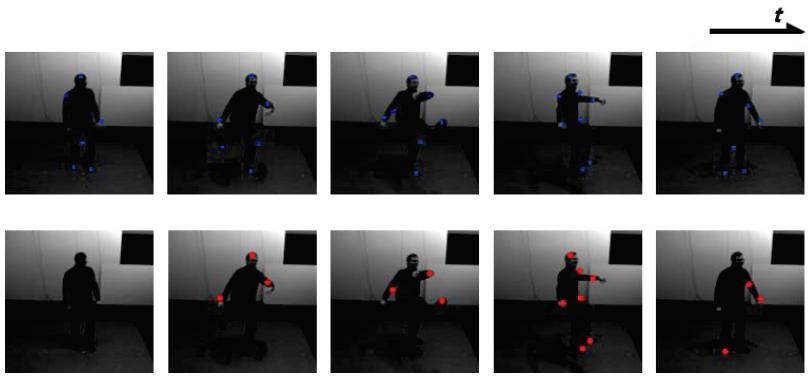


Figure 2.9: Two steps of proposed 3D-STIPs method. SIPs in each image (blue dots in the first row) and 3D-STIPs (red dots in the second row) from an image sequence.

views i and j, then the homography $H_{i\phi j}$ induced by a plane ϕ parallel to π in the reference normal direction is given by

$$H_{i\phi j} = (H_{i\pi j} + [0|\gamma v_{ref}])(I_{3\times 3} - \frac{1}{1+\gamma}[0|\gamma v_{ref}]), \tag{2.26}$$

where γ is scale factor and v_{ref} is the vanishing point which is computed by detecting vertical line segments in the scene and finding their intersection. By accumulating the occupancy grid, we can reconstruct 3D-S objects.

2.3 Proposed method

2.3.1 Modified 3D space-time interest points

We assume that actions can be described properly by focusing on the motion of spatial corner points such as head, hands and feet because the movement of body is meaningless for describing the details of actions. However, the conventional 3D-STIPs may be extracted from the body parts motions in addition to motions of head, hands and feet since existing 3D-STIPs do not take into account an

information about the simultaneous gradient variation in all 3 [x,y,t] dimensions. Therefore the existing method might extract interest points even when any one of the spatial or temporal derivatives becomes large. But, as the human action consists of orderly movements of body parts, the interest point should be extracted where the both spatial and temporal derivatives are simultaneously large. To extract temporal interest points only from spatially distinctive parts such as head, hands and feet, we modify the existing 3D-STIPs method by hierarchically extracting the STIPs in two steps. We first find the space interest points (SIPs) in each image and then decide SIPs having significant variation along time axis as STIPs. Figure 2.9 shows the results of two steps of our 3D-STIPs method. The first row shows the extracted SIPs (blue dots) in each 2D-S image and the second row shows the 3D-STIPs (red dots) from an image sequence. Note that the stationary SIPs are not selected as 3D-STIPs. To find SIPs, we define an image as a function $f\colon R^2 \longmapsto R$ and construct its linear scale-space representation $L\colon R^2 \times R_+ \longmapsto R$ by convolution of f with Gaussian kernel $g_l(x, y; \sigma_l^2)$ with zero mean and spatial variance σ_l^2 as

$$L(x, y; \sigma_l^2) = g_l(x, y; \sigma_l^2) * f(x, y), \tag{2.27}$$

$$g_l(x, y; \sigma_l^2) = \frac{1}{\sqrt{(2\pi)^2 \sigma_l^4}} \exp(-\frac{(x^2 + y^2)}{2\sigma_l^2}). \tag{2.28}$$

Similar to Harris corner detector, we construct a second-moment 2-by-2 matrix $\mu(x, y)$. It is obtained after averaging out the first order spatial derivatives using a Gaussian weighting function $g_i(x, y; \sigma_i^2)$, where the relationship between integration scale and local scale is $\sigma_i^2 = s\sigma_l^2$. The second-moment matrix is given as

$$\mu(x, y) = g_i(x, y; \sigma_i^2) * \begin{bmatrix} L_x^2 & L_x L_y \\ L_x L_y & L_y^2 \end{bmatrix}, \tag{2.29}$$

$$g_i(x, y; \sigma_i^2) = \frac{1}{\sqrt{(2\pi)^2 \sigma_i^4}} \exp(-\frac{(x^2 + y^2)}{2\sigma_i^2}), \tag{2.30}$$

$$L_x = \partial_x L, L_y = \partial_y L, \tag{2.31}$$

where s is a scaling constant. To detect SIPs, we search for points having significant eigenvalues λ_x, λ_y of $\mu(x, y)$ with

$$\lambda_x \geq T_s, \lambda_y \geq T_s, \tag{2.32}$$

where the threshold T_s can be adjusted to get desired number of SIPs. Now STIPs are found by choosing the points among extracted SIPs, which show large variation along the time axis. We use the function $\overline{f}\colon R^2 \times R \longmapsto R$ to represent an image sequence and construct its linear scale-space representation $\overline{L}\colon R^2 \times R \times R_+^2 \longmapsto R$ by convoluting it with the Gaussian kernel $\overline{g}_l(x, y, t; \sigma_l^2, \tau_l^2)$ with zero mean, and spatial and temporal variance σ_l^2, τ_l^2 as

$$\overline{L}(x, y, t; \sigma_l^2, \tau_l^2) = \overline{g}_l(x, y, t; \sigma_l^2, \tau_l^2) * \overline{f}(x, y, t), \tag{2.33}$$

$$\overline{g}_l(x, y, t; \sigma_l^2, \tau_l^2) = \frac{1}{\sqrt{(2\pi)^3 \sigma_l^4 \tau_l^2}} \exp(-\frac{(x^2 + y^2)}{2\sigma_l^2} - \frac{t^2}{2\tau_l^2}). \tag{2.34}$$

Then, the square of temporal derivative of $\overline{L}$ is convoluted with the Gaussian kernel $\overline{g}_i(t; \tau_i^2)$ with integration scale $\tau_i^2 = s\tau_l^2$ as

$$\overline{\mu}(x, y, t) = \overline{g}_i(t; \tau_i^2) * \overline{L}_t^2, \tag{2.35}$$

$$\overline{g}_i(t; \tau_i^2) = \frac{1}{\sqrt{2\pi\tau_i^2}} \exp(-\frac{t^2}{2\tau_i^2}), \tag{2.36}$$

$$\overline{L}_t = \partial_t \overline{L}. \tag{2.37}$$

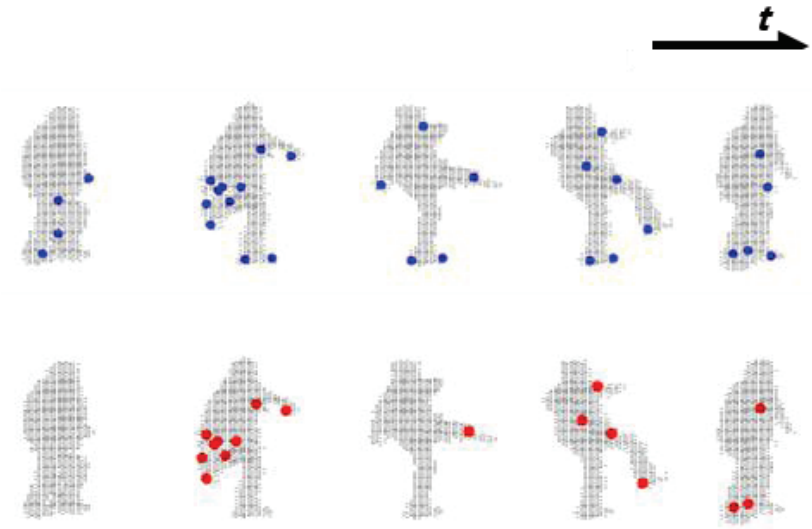


Figure 2.10: Two steps of proposed 4D-STIPs method. SIPs in each 3D-S volume (blue dots in the first row) and 4D-STIPs (red dots in the second row) from 3D-S volume sequence.

STIPs are selected by applying the threshold on $\overline{\mu}(x, y, t)$ as

$$\overline{\mu}(x, y, t) \geq T_{st}, \quad T_{st} = c \cdot \max_{(x,y,t) \in SIPs} \overline{\mu}(x, y, t), \tag{2.38}$$

where T_{st} is set by multiplying a constant c $(0 < c \leq 1)$ with the maximum value of $\overline{\mu}(x, y, t)$ evaluated at all positions of SIPs.

2.3.2 4D space-time interest points

Similar to 3D-STIPs, the 4D-STIPs are calculated by first extracting spatial corner points within each 3D-S volume and then by selecting the corner points having large variation along time axis (Figure 2.10). The first row shows the extracted SIPs (blue dots) in each 3D-S volume and the second row shows the 4D-STIPs (red dots) from 3D-S volume sequence. Note that the stationary SIPs

are not selected as 4D-STIPs. We model a 3D-S volume as $f : R^3 \longmapsto [0,1]$ and construct its linear scale-space representation $L : R^3 \times R_+ \longmapsto R$ by convoluting f with Gaussian kernel $g_l(x, y, z; \sigma_l^2)$ with zero mean and spatial variance σ_l^2 as

$$L(x, y, z; \sigma_l^2) = g_l(x, y, z; \sigma_l^2) * f(x, y, z), \tag{2.39}$$

$$g_l(x, y, z; \sigma_l^2) = \frac{1}{\sqrt{(2\pi)^3 \sigma_l^6}} \exp(-\frac{(x^2 + y^2 + z^2)}{2\sigma_l^2}). \tag{2.40}$$

To find spatial corners where f has significant variations in all directions, we consider a second moment matrix of $\mu(x, y, z)$ integrated over a Gaussian window $g_i(x, y, z; \sigma_i^2)$ with variance σ_i^2, where the relationship between integration scale and local scale is $\sigma_i^2 = s\sigma_l^2$ as

$$\mu(x, y, z) = g_i(x, y, z; \sigma_i^2) * \begin{bmatrix} L_x^2 & L_x L_y & L_x L_z \\ L_x L_y & L_y^2 & L_y L_z \\ L_x L_z & L_y L_z & L_z^2 \end{bmatrix}, \tag{2.41}$$

$$g_i(x, y, z; \sigma_i^2) = \frac{1}{\sqrt{(2\pi)^3 \sigma_i^6}} \exp(-\frac{(x^2 + y^2 + z^2)}{2\sigma_i^2}), \tag{2.42}$$

$$L_x = \partial_x L, L_y = \partial_y L, L_z = \partial_z L. \tag{2.43}$$

The points where all eigenvalues of $\mu(x, y, z)$ are greater than the threshold are selected as the spatial corner points such as

$$\lambda_x \geq T_s, \lambda_y \geq T_s, \lambda_z \geq T_s, \tag{2.44}$$

where T_s is adjusted to obtain the desired number of spatial corner points within each frames. A function $\overline{f} : R^3 \times R \longmapsto [0,1]$ describes a 3D-S volume sequence, and its linear scale-space representation $\overline{L} : R^3 \times R \times R_+^2 \longmapsto R$, is given by

convolution with the Gaussian kernel $\overline{g}_l(x, y, z, t; \sigma_l^2, \tau_l^2)$ with zero mean, and spatial and spatial and temporal variance σ_l^2, τ_l^2 as

$$\overline{L}(x, y, z, t; \sigma_l^2, \tau_l^2) = \overline{g}_l(x, y, z, t; \sigma_l^2, \tau_l^2) * \overline{f}(x, y, z, t), \tag{2.45}$$

$$\overline{g}_l(x, y, z, t; \sigma_l^2, \tau_l^2) = \frac{1}{\sqrt{(2\pi)^4 \sigma_l^6 \tau_l^2}} \exp(-\frac{(x^2 + y^2 + z^2)}{2\sigma_l^2} - \frac{t^2}{2\tau_l^2}). \tag{2.46}$$

The square of $\overline{L}_t$ is convoluted with the Gaussian kernel $\overline{g}_i(t; \tau_i^2)$ with integration scale $\tau_i^2 = s\tau_l^2$ as

$$\overline{\mu}(x, y, z, t) = \overline{g}_i(t; \tau_i^2) * \overline{L}_t^2, \tag{2.47}$$

$$\overline{g}_i(t; \tau_i^2) = \frac{1}{\sqrt{2\pi\tau_i^2}} \exp(-\frac{t^2}{2\tau_i^2}), \tag{2.48}$$

$$\overline{L}_t = \partial_t \overline{L}. \tag{2.49}$$

Then, the 4D-STIPs are computed from the previously obtained SIPs by applying threshold on $\overline{\mu}(x, y, z, t)$ as

$$\overline{\mu}(x, y, z, t) \geq T_{st}, \quad T_{st} = c \cdot \max_{(x,y,z,t) \in SIPs} \overline{\mu}(x, y, z, t), \tag{2.50}$$

where T_{st} is set as the fraction $(0 < c \leq 1)$ of the maximum value of $\overline{\mu}(x, y, z, t)$ evaluated at all positions of SIPs. The calculated 4D-STIPs are projected to the 2D-S views, which are defined with the camera matrices of the training views.

2.4 Experimental results

The IXMAS dataset (Figure 2.11) contains 1980 videos of total 11 action categories (check watch, cross arm, scratch head, sit down, get up, turn around, walk, wave, punch, kick, pick up) captured from 5 different views. The 3D-S

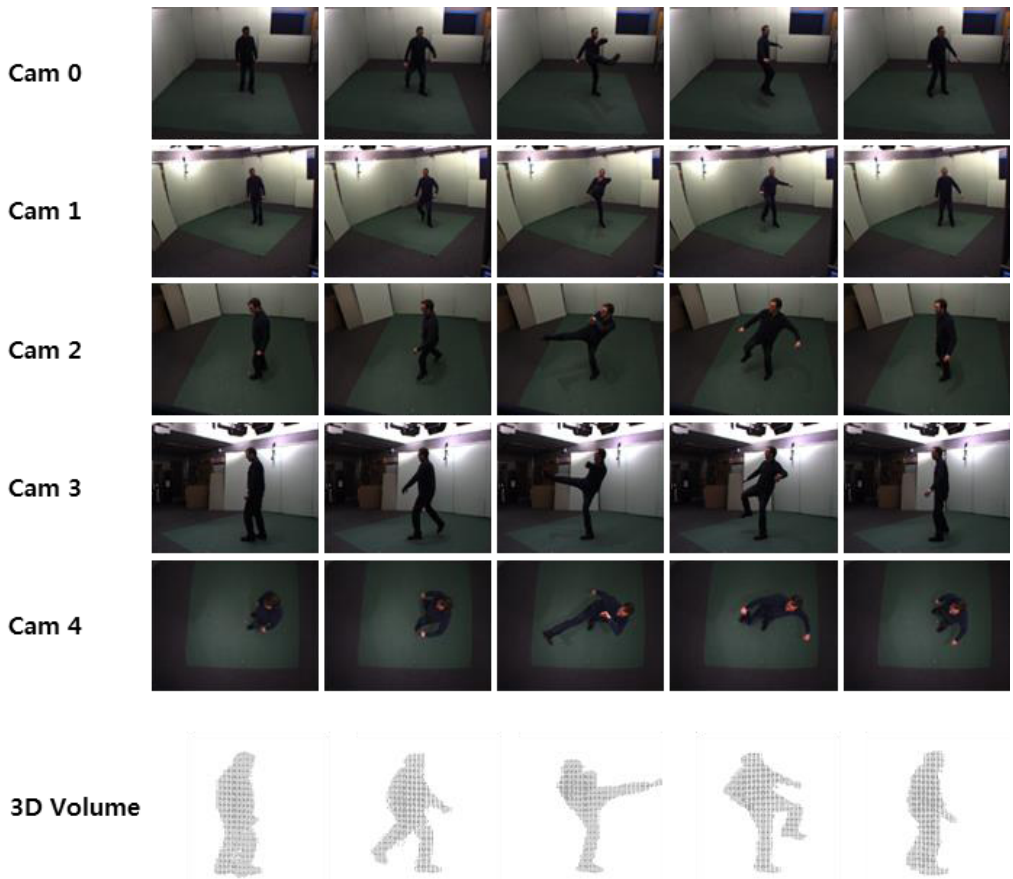


Figure 2.11: The IXMAS dataset [2]. Each row (1-5) shows 'kick' action sequence from different views and the last row shows 3D-S volumes reconstructed from the silhouettes of each column.

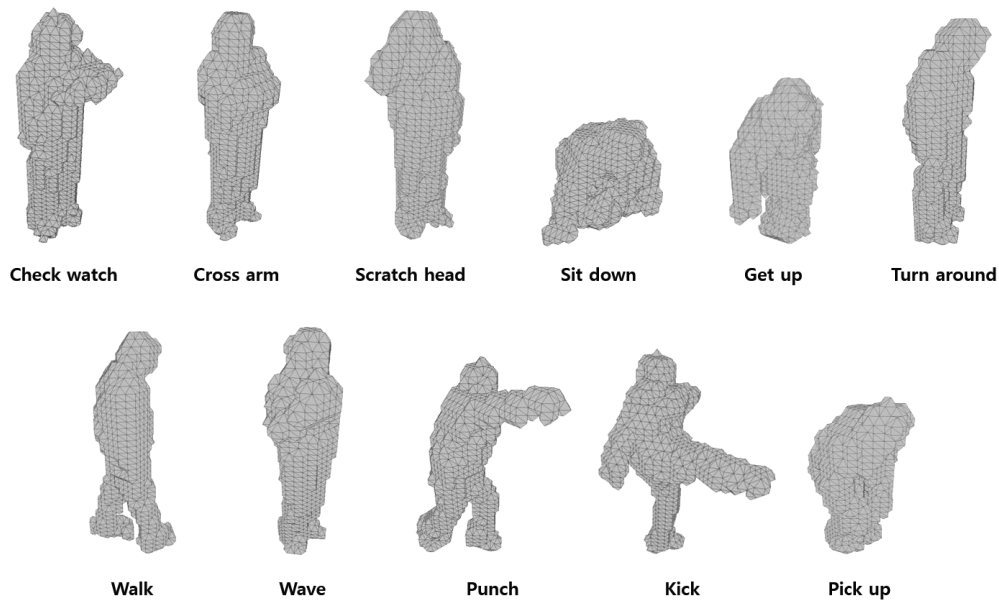


Figure 2.12: Various 3D-S volumes [2]. 3D-S volumes of 11 actions in IXMAS dataset.

volumes were reconstructed by computing the visual hulls via back projecting the multi-view 2D-S silhouettes [40]. In Figure 2.11, each row (1-5) shows 'kick' action sequence from 5 different views and the last row shows 3D-S volumes reconstructed from the silhouettes of each column. In this work, we used the reconstructed 3D-S volumes (Figure 2.12) provided along with the 5 different view videos. Each action in IXMAS was performed 3 times by 12 actors. The position and the orientation of actors were chosen arbitrarily.

We calculated variant 3D-STIPs using image sequences from 5 different views and 4D-STIPs using 3D-S volumes in IXMAS dataset. The 3D-STIPs were computed by using scale-normalized image sequences (161×161). The SIPs were calculated within convoluted images with $\sigma_l^2 = 6.25$ and integration scale $\sigma_i^2 = 3\sigma_l^2$. T_s was adjusted to have $7 \sim 10$ SIPs in each frame. For convoluting image sequences along time axis the parameters were set as $\tau_l^2 = 1$ and $\tau_i^2 = 4\tau_l^2$. The

constant c which determines T_{st} was set to 0.05.

The calculated 3D-STIPs are shown in Figure 2.13. The SIPs represented by blue dots are spatial corner points such as head, hands and feet and the red dots are 3D-STIPs which were obtained by selecting the corner points having large variation along time axis among SIPs. In the 'check watch' action, the 3D-STIPs are shown near the arms and hands parts, while they are shown near the legs and feet in the 'kick' action. Actions can be represented compactly using positions of 3D-STIPs.

For calculating 4D-STIPs, we used scale-normalized 3D-S volumes ($64 \times 64 \times 64$). The SIPs were calculated within convoluted volumes with $\sigma_l^2 = 0.5$ and integration scale $\sigma_i^2 = 2\sigma_l^2$. T_s was adjusted to get $3 \sim 6$ SIPs in each frame. For convoluting volume sequences along time axis the parameters were set as $\tau_l^2 = 1$ and $\tau_i^2 = 4\tau_l^2$. The constant c which determines T_{st} was set to 0.3.

The calculated 4D-STIPs are shown in Figure 2.14. The SIPs represented by blue dots are spatial corner points such as head, hands and feet and the red dots are 4D-STIPs which were obtained by selecting the corner points having large variation along time axis among SIPs. Actions can be represented compactly using positions of 4D-STIPs. Furthermore the features for arbitrary 2D-S viewpoint can be generated by projecting 4D-STIPs on corresponding image planes since 4D-STIPs are constructed using volumetric information, which enables view invariant action recognition. Figure 2.15 shows 4D-STIPs accumulated from 36 video sequences (3 times repeated by 12 actors) of each action. The positions of 4D-STIPs from different video sequences are similar and, therefore, 4D-STIPs show the properties of each action well.

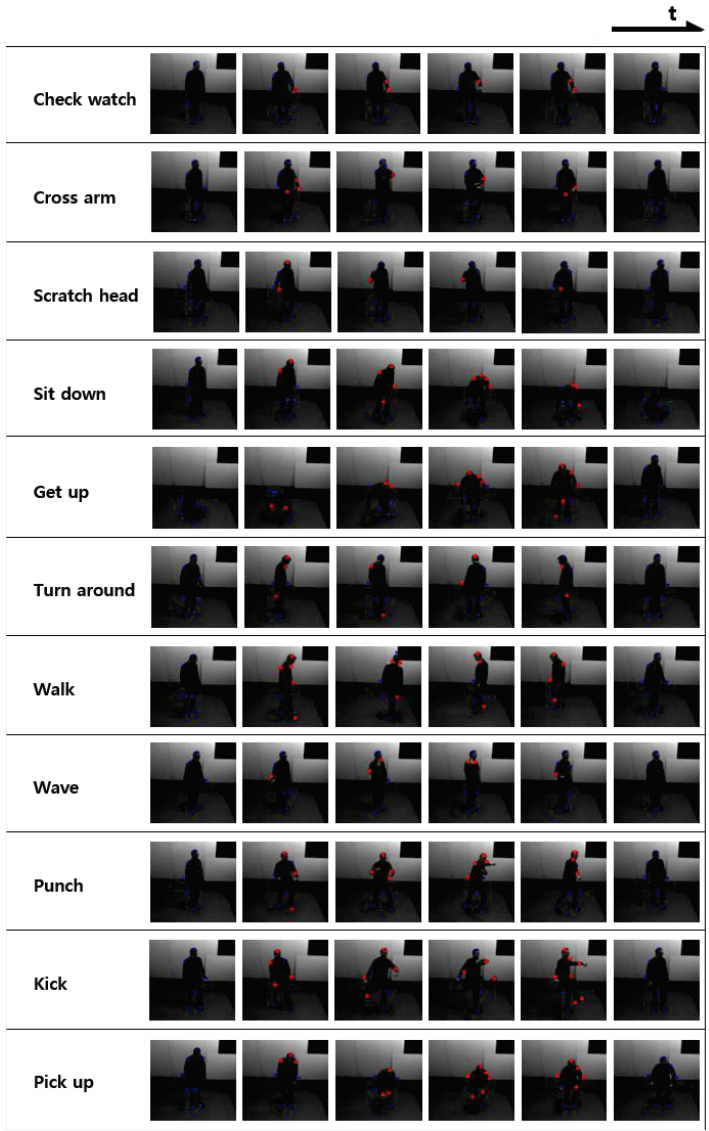


Figure 2.13: Detected 3D-STIPs. 3D-STIPs of 11 actions in IXMAS dataset using 2D-S image sequences. The SIPs represented by blue dots are spatial corner points such as head, hands and feet and the red dots are 3D-STIPs which are obtained by selecting the corner points having large variation along time axis among SIPs.

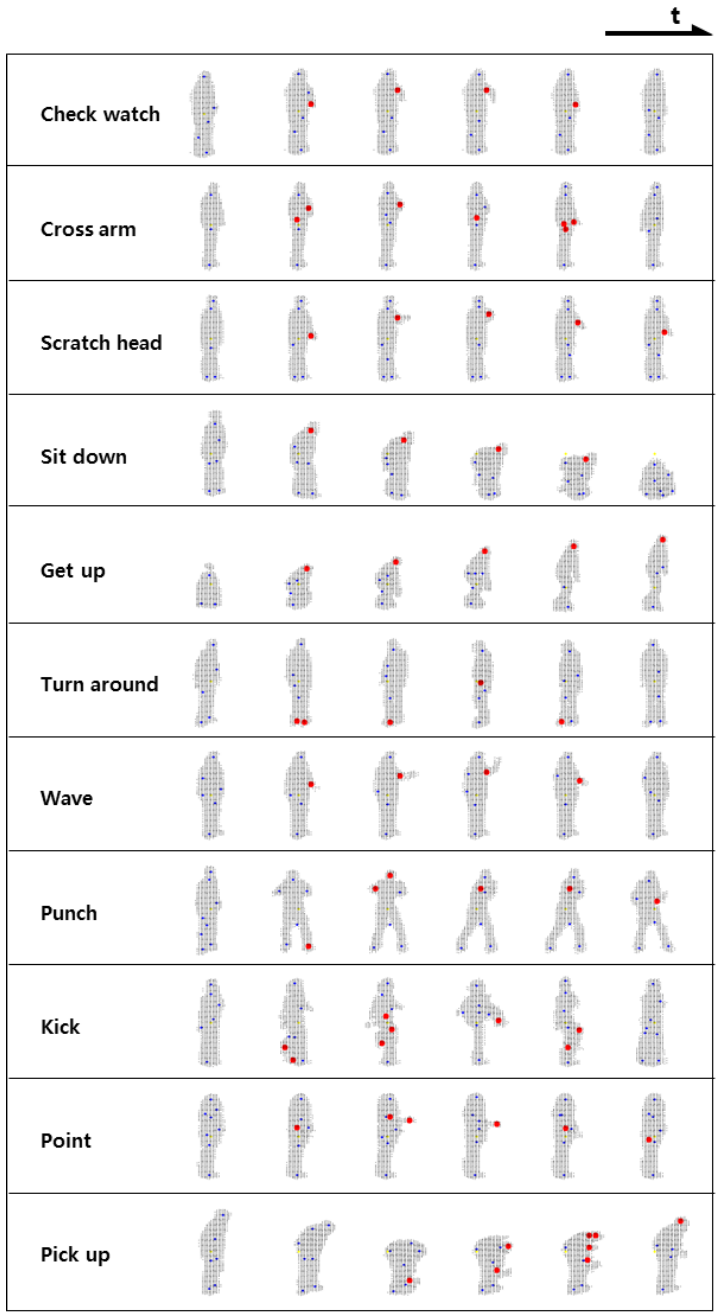


Figure 2.14: Detected 4D-STIPs. 4D-STIPs of 11 actions in IXMAS dataset using 3D-S volume sequences. The SIPs represented by blue dots are spatial corner points such as head, hands and feet and the red dots are 4D-STIPs which are obtained by selecting the corner points having large variation along time axis among SIPs.

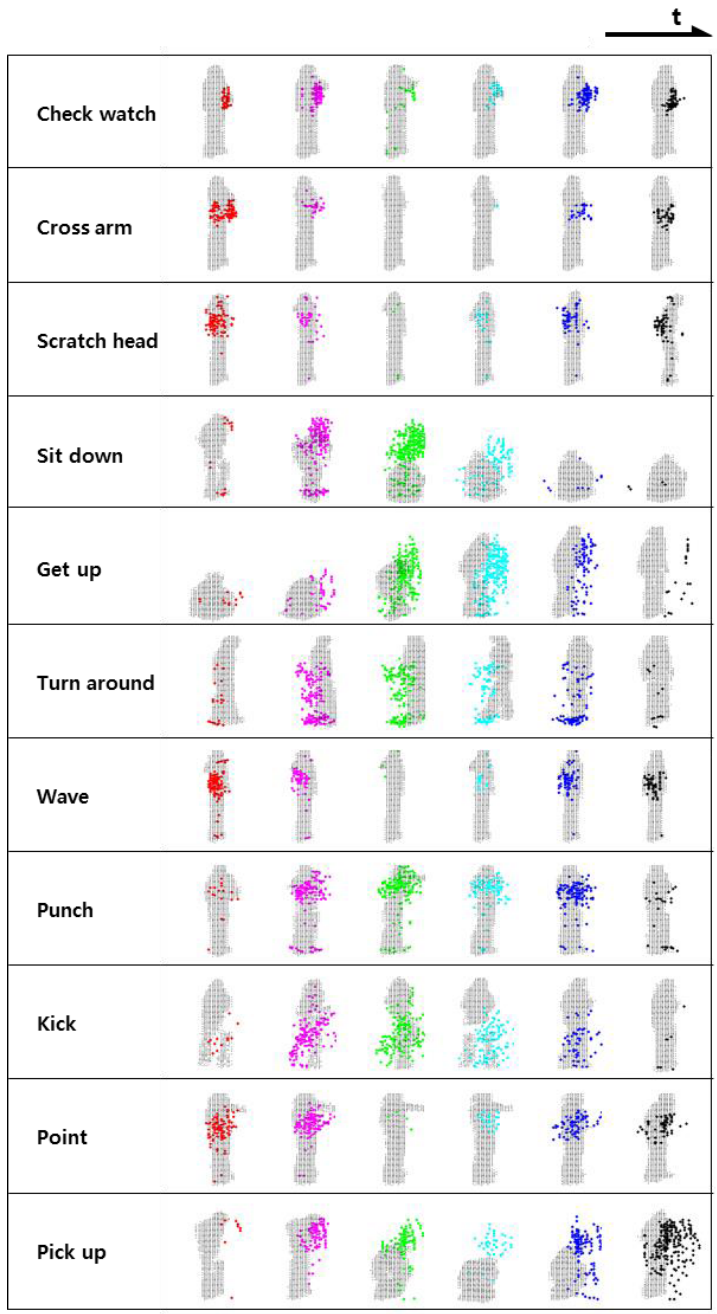


Figure 2.15: Collected 4D-STIPs. 4D-STIPs accumulated from 36 video sequences (3 times repeated by 12 actors) of each action. The positions of 4D-STIPs from different video sequences are similar and, therefore, 4D-STIPs show the properties of each action well.

2.5 Concluding remarks

In this chapter, we proposed a variant 3D-STIPs and new 4D-STIPs by extending the widely used 3D-STIPs using 3D-S volume sequences. With the proposed 4D-STIPs, we can generate features of all the views by projecting them to arbitrary 2D-S image planes. Therefore, the 4D-STIPs enable view invariant action recognition which is very useful for practical applications. The proposed 3D-STIPs and 4D-STIPs were calculated using multi-view IXMAS dataset and verified that they represent the properties of each action compactly.

Chapter 3

View Invariant Action Recognition

3.1 Introduction

In computer vision, human action recognition is a multifaceted research area, encompassing wide applications such as visual surveillance, human-computer interaction, and video retrieval. In chapter 1, we categorized action recognition research largely into two cases of gesture and general action recognition. Most actual applications for gesture recognition assume human and computer face each other and the recognition in front view images is performed. However, recognition of actions in general settings such as surveillance environments requires view invariant recognition. To solve view invariant action recognition problem in general condition, various methods are proposed and those methods can be separated as 3D space (3D-S, [x,y,z]) and 2D space (2D-S, [x,y]) approaches. In recognition step, 3D-S approach requires special settings such as multiple number of cameras or RGB-D sensors, whereas 2D-S approach uses just 2D-S images from a

single camera. Therefore the proposed method follows the 2D-S approach, the research has been conducted in three directions: 1) designing of view invariant features [22, 23, 25, 24, 26], 2) designing a classifier based on transfer learning [27, 28, 29, 30, 31, 32, 33], and 3) learning of features from multiple views [34, 35, 36, 37, 38, 39]. View invariant features are designed by extracting features which are observed or maintained for almost all views. However, the descriptors of this scheme are inherently limited to be strictly view invariant and could not work well when the view changes on a large scale. To cope with the inherent limitation of view invariant features, transfer learning concept were reported. However, this transfer learning approach has an impractical assumption that source and target views are defined in advance. In this approach, the classifier is trained with visual words from the source view, but the visual dictionary is made using videos from the target view as well as the source view. Hence this approach could not guarantee the performance for the new view which does not participate in the process making visual dictionary. This implies the transfer learning approach is limited to a set of predefined views and so it does not provide arbitrary view invariant property. To achieve the complete view invariant recognition using single camera, all the features from all of views should be accumulated. Since it is not possible to obtain information from all of views in practice, most of the existing methods deployed a finite number of views to generalize the recognition ability. However, these methods still do not guarantee a satisfying performance on the new test view which is not contained in the training. To achieve a satisfactory result, acquiring data from different views using a large number of cameras is required.

In this chapter, we propose a method for view invariant action recognition using the 4D space-time (4D-ST [x-y-z-t]) motion features proposed in chapter 2. To extract the 4D-ST features, 3D-S volumes are constructed by back-projecting

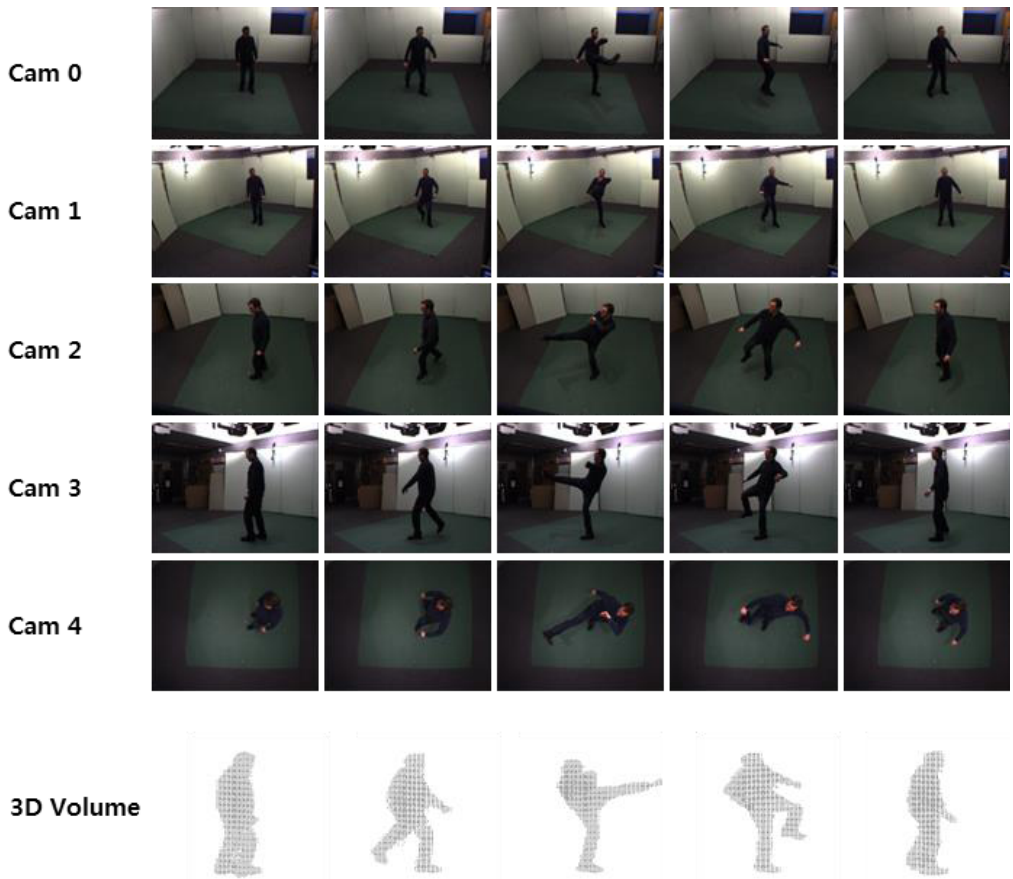


Figure 3.1: The IXMAS dataset [2]. Each row (1-5) shows 'kick' action sequence from different views and the last row shows 3D-S volumes reconstructed from the silhouettes of each column.

the silhouettes from 5 cameras of different views (Figure 3.1) with the method given by [40]. Then the 4D space-time interest points (4D-STIPs, [x,y,z,t]) are computed from the constructed 3D-S volumes. Unlike the existing methods which extract features from the limited number of 2D-S images of different views, depending on the number of used cameras, we can generate the features from any number of views even when just 5 cameras are used. This can be done by projecting the features in 3D-S to the arbitrary 2D-S image planes. Consequently, the proposed method shows good recognition results even when the images of test view are not available during the training step. A classifier is trained for each action class with the features projected from the 3D-S volumes and the 4D-STIPs. Silhouettes are formed for 2D-S planes after projecting the 3D-S volumes on the respective image planes. The silhouettes and the projected 4D-STIPs are used to build the motion history images (MHIs) [69] and the non-motion history images (NMHIs), which encode moving and non-moving aspects of an action respectively. While other methods rely on only moving aspects of an action, we use both aspects (moving and non-moving) to design features since the history of non-moving parts also reveals a discriminative information for action recognition. NMHIs give clues about the position of the body parts during the course of the action. To reduce the dimension of MHIs and NMHIs, we apply class-augmented principal component analysis (CA-PCA) proposed by [70]. CA-PCA is the dimension reduction algorithm which is appropriate for classification problem since it preserves separability of classes while reducing the feature dimension by using the class information. Since we use the action label for reducing the dimension of features, we obtain the principal axis which can separate each action well. After reducing the feature dimension, the final features are trained by support vector data description method (SVDD) [71]. In the recognition step, similarly we obtain MHIs and NMHIs from an image sequence by calculating silhouettes

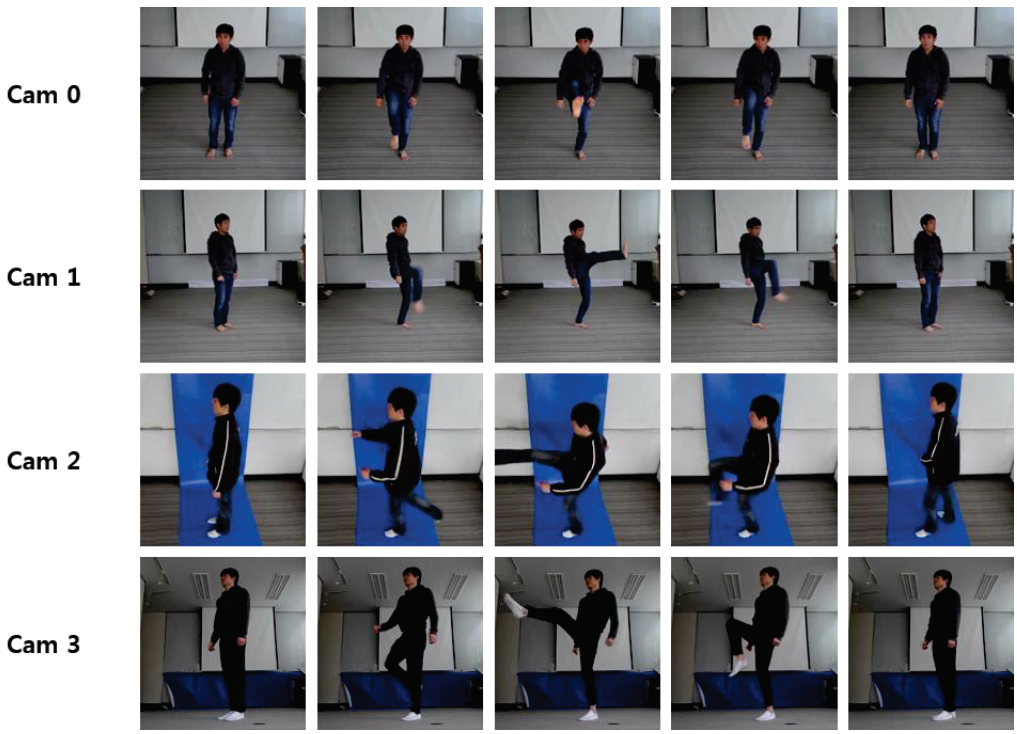


Figure 3.2: The SNU dataset. Each row (1-4) shows 'kick' action sequence from different views.

and 3D space-time interest points (3D-STIPs, [x,y,t]). The two features (MHIs, NMHIs) are reduced the dimension by the principal axis obtained using action label, and then recognized action with support vector domain density description (SVDDD) [72] classifier. In experiments, we train the models using INRIA Xmas Motion Acquisition Sequences (IXMAS) (Figure 3.1) [2] and test them a new SNU dataset (Figure 3.2) made for evaluating the generalization performance for arbitrary view videos.

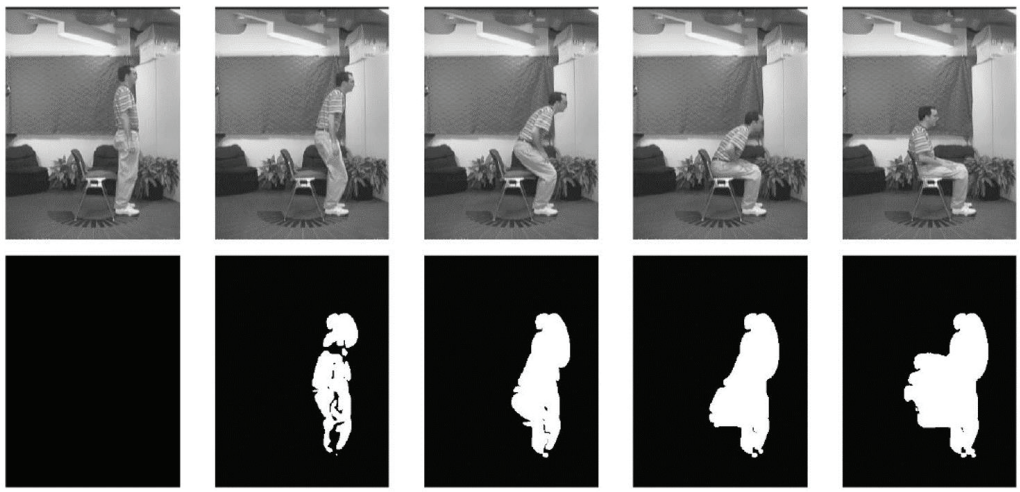


Figure 3.3: MEIs [69]. An example of someone sitting. The first row shows key frames and the second row shows the cumulative motion images.

3.2 Preliminaries

3.2.1 Motion history images

To reduce the data size and represent actions compactly, we adopt the notion of MHIs [69] which encodes the history of motion occurrences in an image. We first present the construction of a binary motion energy images (MEIs) which represent where motion has occurred in an image sequence. Next, we explain MHIs which is a scalar valued image where intensity is a function of recent motion. The MEIs and MHIs can be considered as a two component version of a temporal template.

When the action occurs, the action sequence sweeps out a particular region of the image, and the shape of region (where there is motion) can be used as a feature for action recognition. We refer to these binary cumulative motion images as MEIs. Let $I(x, y, t)$ be an image sequence and let $D(x, y, t)$ be a binary image sequence indicating regions of motion. $D(x, y, t)$ is equal to 1 if there exists motion at (x, y, t) in an image sequence and 0 otherwise. Then the binary MEIs $E_\tau(x, y, t)$

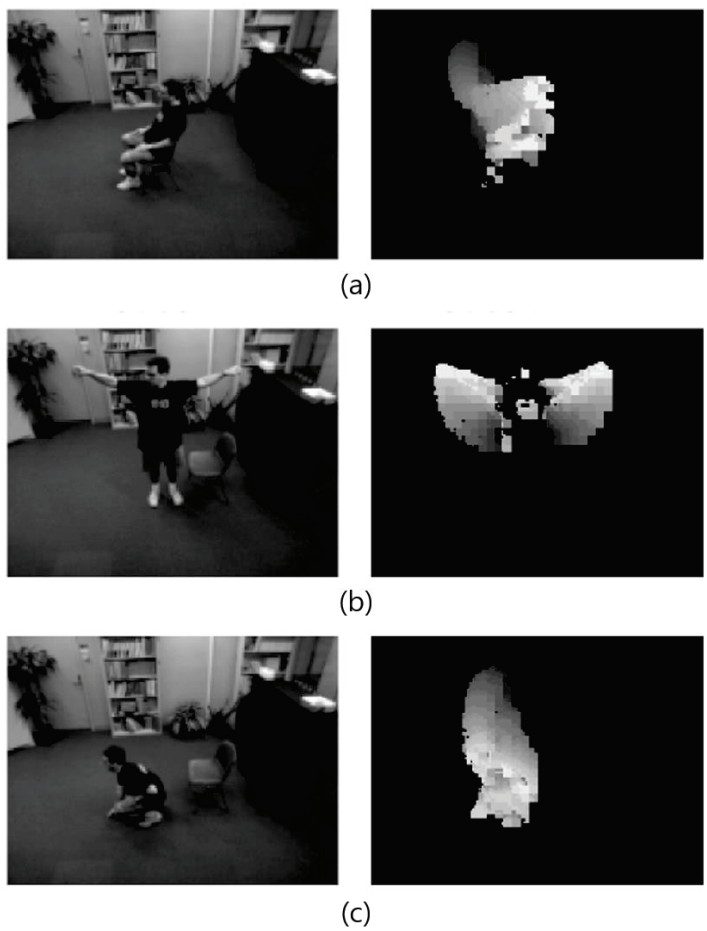


Figure 3.4: MHIs [69]. Examples of MHIs. (a) Sit down, (b) arms wave, and (c) crouch down.

is defined

$$E_\tau(x, y, t) = \bigcup_{i=0}^{\tau-1} D(x, y, t - i), \tag{3.1}$$

where the duration τ is the maximum duration for which a motion is stored. Figure 3.3 shows the example of MEIs.

To represent how (as opposed to where) motion the image is moving, we form MHIs. In the MHIs H_τ, pixel intensity is a function of the temporal history of motion at that point and represented as follows

$$H_\tau(x, y, t) = \begin{cases} \tau & \text{if } D(x, y, t) = 1 \\ max(0, H_\tau(x, y, t-1) - 1) & \text{otherwise} \end{cases}. \tag{3.2}$$

The result is a scalar-valued image where more recently moving pixels are brighter. The MEIs can be generated by thresholding the MHIs above zero. Figure 3.4 shows the example of MHIs.

3.2.2 Class-augmented principal component analysis

Feature dimension reduction is to generate a set of features that have smaller dimension than original data. While reducing the dimension of features the characteristics of data should be preserved sufficiently to classify the data. These dimension reduced features can improve not only the computational speed but also classification performance by removing non-relevant characteristics in a data set. The principal component analysis (PCA) [73] is a well known dimension reduction method which seeks the projection which best represents the data in a least square sense. This method is very effective to find the features for reducing the dimension, however, the reduced features may not appropriate for classification problem since the class information is not considered during the reducing the dimension. For example, if PCA is applied to the data set presented in Figure

3.5 (a), the calculated principal axis w_1 can not properly separate the data which belong to different classes. To solve this problem, [70] proposed the CA-PCA method which utilizes the class information while reducing the dimension of features. In order to use the class information, the new dimension which encodes the class information is augmented to the original data. A new data representation is defined by augmenting a new axis which is orthogonal to all original axis and assigning a value along this new axis according to the class information of each data.

In Figure 3.5 (b), the new represented data are plotted. By applying PCA to these new data representation, we can find the axis $w_1^{''}, w_2^{''}$. The projection onto the axis $w_1^{''}, w_2^{''}$ requires the value of data on class axis, and therefore these axis cannot be immediately applied to the test data whose class information is unknown. If the value of each data along class axis is carefully adjusted such that the variance along class axis is very small, $w_1^{''}, w_2^{''}$ can be approximated to $w_1^{'}, w_2^{'}$ which are composed of the original data axis and they can be used for reducing the dimension of data in classification problem.

Now we will explain how to 1) encode and 2) normalize the class information on class axis. Let the number of classes by n_{class}, then each class information for data X is represented by

$$C(X) = [c_1, c_2, ..., c_{n_{class}}]^T, \tag{3.3}$$

where class label c_i of X is a constant p if X belongs to class i, otherwise c_i is another constant n_i. Values of n_i are determined so that the mean of c_i becomes 0. p is determined so that the sum of the variances $\sum_{i=1}^{n_{class}} var(c_i)$ becomes σ^2 in which σ is selected as a scalar value much less than 1. Since the number of equation is $n_{class} + 1$ and the number of variables is $n_{class} + 1$, all n_i and p can

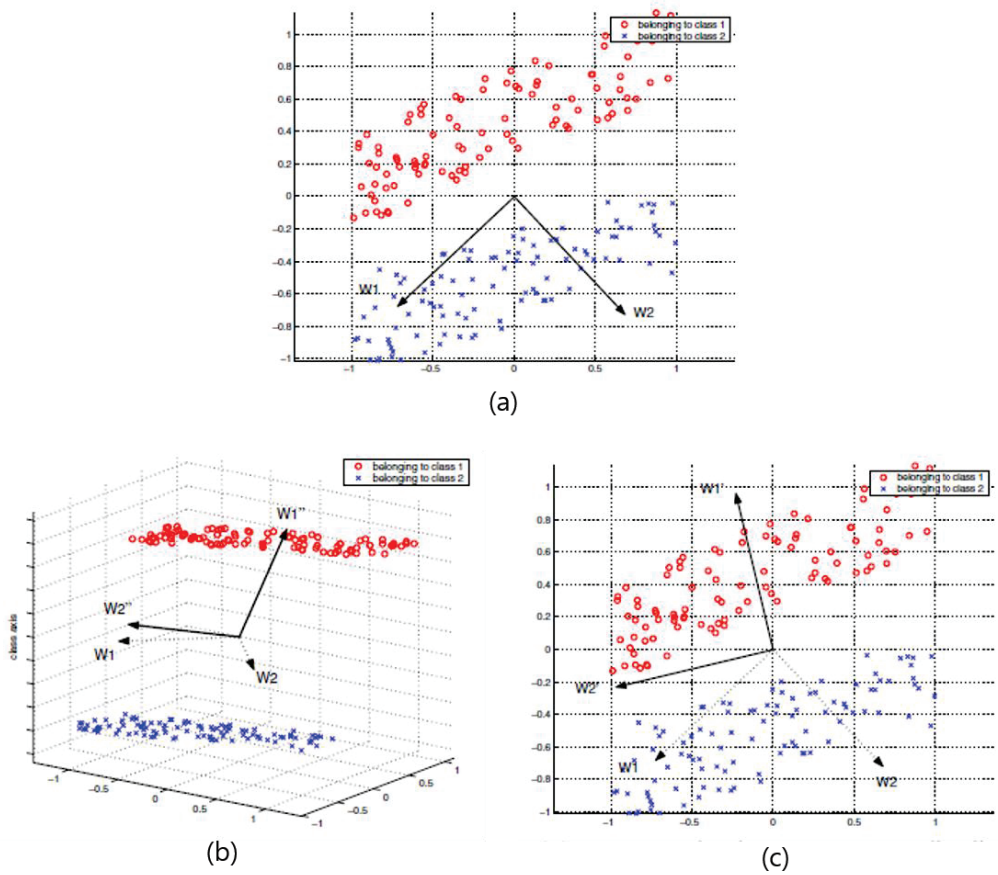


Figure 3.5: An example of CA-PCA [70]. (a) The axis w_1, w_2 extracted by PCA. (b) The axis $w_1^{''}, w_2^{''}$ extracted by PCA applied to the new data. (c) The axis $w_1^{'}, w_2^{'}$ selected from $w_1^{''}, w_2^{''}$.

be determined.

For example, assume that three data X_1, X_2, X_3 are given and the number of total classed n_{class} is 2. X_1 belongs to the first class, and the others belong to the second class. In this case the class information $C(X)$ for input data is expressed as follows

$$C(X_1) = [p, n_2]^T, \tag{3.4}$$

$$C(X_2) = [n_1, p]^T, \tag{3.5}$$

$$C(X_3) = [n_1, p]^T. \tag{3.6}$$

For this representation, if $p = \sqrt{10}\sigma/5$, $n_1 = \sqrt{10}\sigma/10$, and $n_2 = -2\sqrt{10}\sigma/5$, the average of c_1 and c_2 becomes 0 and the sum of the variances $\sum_{i=1}^{2} var(c_i)$ becomes σ^2.

The purpose of normalization is to maximize the effect of class information in the selection of principal components. The principal components in Figure 3.5 (a) are rotated to (c) by variances on the class axis. For this, the variance of data along the class axis needs to be set larger than the difference between the variances along axis on data plane. This condition can be described by the following equation. For all i and j,

$$\sigma^2 \geq \|\sigma_{w_i}^2 - \sigma_{w_j}^2\|, \tag{3.7}$$

where $\sigma_{w_i}^2$ is the variance along the axis w_i. This condition can be simultaneously satisfied for all pairs (i, j) by normalizing the variance along each axis to be 1. The difference between variances along each axis becomes 0 and the condition can be satisfied for any values of σ which is larger than 0. The normalization is

carried out for $j = 1, 2, ..., n_{input}$ by

$$\overline{x}_j = x_j / \sigma_j, \tag{3.8}$$

where $\overline{X}$ is the normalized data of X and x_j is the j^{th} element of X, and σ_j is the standard deviation of j^{th} element over the entire data set. After encoding the class information and normalization, each X is represented by

$$\overline{X}_i^a = \begin{bmatrix} \overline{X}_i \\ C(X_i) \end{bmatrix}. \tag{3.9}$$

Then, the normal PCA is applied to the set of $\overline{X}^a$ in order to obtain the principal components. The dimension of each principal component is $(n_{input} + n_{class})$. To reduce the input dimension, the reduced dimension should satisfy the condition $n_{reduced} < n_{input}$ and, therefore, $n_{reduced}$ principal components are selected along which the variance of the data is large as

$$\overline{W}^a = [\overline{x}_1^a, \overline{x}_2^a, ..., \overline{x}_{n_{reduced}}^a]. \tag{3.10}$$

Then the reduced data can be found by

$$X_{reduced} = \overline{W}^{aT} \overline{X}^a. \tag{3.11}$$

To avoid repeating the normalization for new data, the scaling factor of normalization can be introduced in W by following equation for $i = 1, 2, ..., n_{reduced}$ and $j = 1, 2, ..., n_{input}$ as

$$w_{ij}^a = \overline{w}_{ij}^2 / \sigma_j. \tag{3.12}$$

The other elements for $j = n_{input} + 1, n_{input} + 2, ..., n_{input} + n_{class}$ need not be

modified since the class information of test data will not be given. Then the equation (3.11) can be rewritten as

$$X_{reduced} = W^{aT} X^a, \tag{3.13}$$

where $W^a = [w_1^2, w_2^2, ..., w_{n_{reduced}}^2] = [W_{input}^T, W_{class}^T]^T$.

The dimension of augmented data is reduced by

$$\begin{aligned} X_{reduced} &= W^{aT} X^a, \\ &= [W_{input}^T, W_{class}^T] \begin{bmatrix} X \\ C(X) \end{bmatrix}, \\ &= W_{input}^T X + W_{class}^T C(X). \end{aligned} \tag{3.14}$$

However, the class information of test data is not given and, therefore, we can not adopt the equation (3.14) and should modify for the test data. If σ^2 is set to a much smaller value than 1, the elements from $W_{class}^T C(X)$ tend to be much smaller than those from $W_{input}^T X$ and the second term in equation (3.14) can be omitted. After omitting the second term, the final equation becomes as follows

$$\begin{aligned} X_{reduced} &= W_{input}^T X + W_{class}^T C(X), \\ &\simeq W_{input}^T X, \\ &= W^T X. \end{aligned} \tag{3.15}$$

The equation can be applied for any test data X without class information, in order to reduce the dimension of data.

3.2.3 Support vector data description

The objective of SVDD [71, 74] which is inspired by SVM [75, 76, 77] is to find a sphere or domain with minimum volume containing all or most of the data. Let $\{\vec{x_i}\} \subset X$ be the given training data set of n points with data space $X \subset R^d$. To minimize the volume of sphere, the objective function should be designed for minimizing the radius of sphere with constraints that the distance between the center of sphere and data is smaller than the radius as

$$F(R, \vec{a}) = R^2, \tag{3.16}$$

$$\|\vec{x_i} - \vec{a}\|^2 \leq R^2, \forall i, \tag{3.17}$$

where R is radius of sphere and $\vec{a}$ is the center position of the sphere. To allow the possibility of outliers in the training data set, the slack variable $\xi_i \geq 0$ are introduced with constraints that almost all objects are within the sphere as

$$F(R, \vec{a}) = R^2 + C\sum_i \xi_i, \tag{3.18}$$

$$\|\vec{x_i} - \vec{a}\|^2 \leq R^2 + \xi_i, \xi_i \geq 0, \forall i, \tag{3.19}$$

where C gives the trad-off between the volume of sphere and the number of error. To solve this problem, the Lagrangian is introduced as

$$L = R^2 + C\sum_i \xi_i - \sum_i \alpha_i (R^2 + \xi_i - \|\vec{x_i} - \vec{a}\|^2) - \sum_i \gamma_i \xi_i. \tag{3.20}$$

By KKT conditions we can get

$$\sum_i \alpha_i = 1, \tag{3.21}$$

$$\vec{a} = \frac{\sum\limits_i \alpha_i \vec{x_i}}{\sum\limits_i \alpha_i} = \sum\limits_i \alpha_i \vec{x_i}, \tag{3.22}$$

$$C - \alpha_i - \gamma_i = 0, \forall i, \tag{3.23}$$

$$\alpha_i (R^2 + \xi_i - \|\vec{x_i} - \vec{a_i}\|^2) = 0, \tag{3.24}$$

$$\gamma_i \xi_i = 0. \tag{3.25}$$

Since $\alpha_i \geq 0$ and $\gamma_i \geq 0$, we can get $0 \leq \alpha_i \leq C$ using equation (3.23). Therefore, the primal form of equation (3.16) can be transformed into the dual form with constraints as

$$L = \sum\limits_i \alpha_i (\vec{x_i} \cdot \vec{x_i}) - \sum\limits_{i,j} \alpha_i \alpha_j (\vec{x_i} \cdot \vec{x_j}), \tag{3.26}$$

$$\sum\limits_i \alpha_i = 1, \tag{3.27}$$

$$0 \leq \alpha_i \leq C, \forall i. \tag{3.28}$$

By maximizing equation (3.26) with respect to α_i, we obtain the solution α_i for $i = 1, ..., n$. When a sample x_i satisfies the inequality of (3.19), the corresponding Lagrange multiplier goes to zero. For samples satisfying the equality of (3.19), the Lagrange multiplier will become nonzero. The solution α_i is categorized into three types as follows

$$\|\vec{x_i} - \vec{a}\|^2 < R^2 \rightarrow \alpha_i = 0, \gamma_i = C, \tag{3.29}$$

$$\|\vec{x_i} - \vec{a}\|^2 = R^2 \rightarrow 0 < \alpha_i < C, \gamma_i > 0, \tag{3.30}$$

$$\|\vec{x_i} - \vec{a}\|^2 > R^2 \rightarrow \alpha_i = C, \gamma_i = 0. \tag{3.31}$$

From the above results, the domain description is represented in terms of only support vectors whose Lagrange multipliers satisfy $0 < \alpha_i < C$, which provides the sparse representation of the domain description. The domain center and the

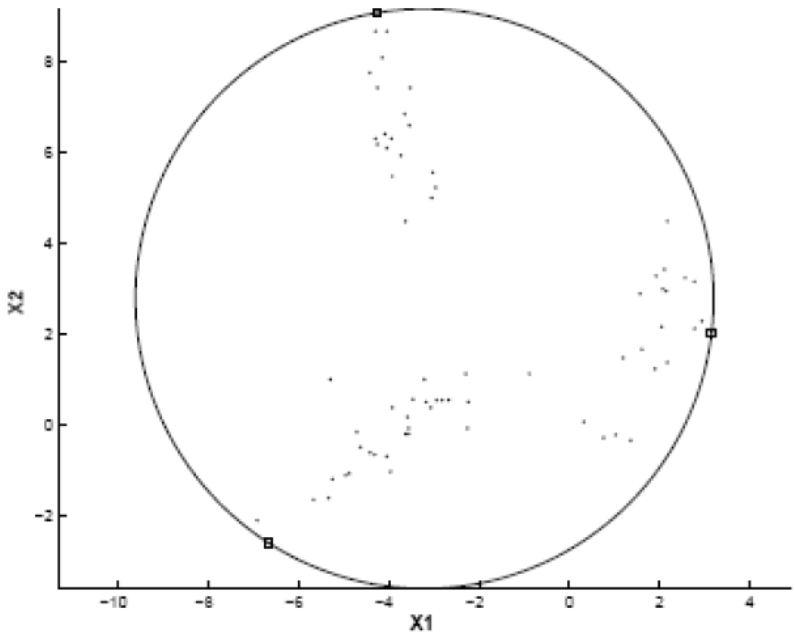


Figure 3.6: The linear SVDD [74]. The sphere by linear SVDD where the sample data are generated from the combination of 3 Gaussian distribution in two dimensional space.

boundary radius are determined by

$$\vec{a} = \sum_i \alpha_i \vec{x_i}, \tag{3.32}$$

$$R^2 = \|\vec{x_k} - \vec{a}\|^2 = (\vec{x_k} \cdot \vec{x_k}) - 2\sum_i \alpha_i (\vec{x_i} \cdot \vec{x_k}) - \sum_{i,j} \alpha_i \alpha_j (\vec{x_i} \cdot \vec{x_j}), \tag{3.33}$$

where x_k is a support vector. To determine whether a test point $\vec{z}$ is within the sphere, the distance between $\vec{z}$ and the center of the sphere can be calculated as follows

$$D^2(\vec{z}) = \|\vec{z} - \vec{a}\|^2 = \vec{z} \cdot \vec{z} - 2\sum_i \alpha_i (\vec{z} \cdot \vec{x_i}) + \sum_{i,j} \alpha_i \alpha_j (\vec{x_i} \cdot \vec{x_j}). \tag{3.34}$$

When $D^2(\vec{z}) \leq R^2$, a test data $\vec{z}$ is within the sphere.

The Figure 3.6 shows the sphere by linear SVDD where the sample data are

generated from the combination of 3 Gaussian distribution in two dimensional space. Normally, data are not spherically distributed and, therefore, in general we cannot expect to obtain a very tight description as shown in Figure 3.6. Since the problem is stated completely in terms of inner product between vectors, the method can be made more flexible, analogous to SVM. We replace all inner product $\vec{x_i} \cdot \vec{x_j}$ by a proper $K(\vec{x_i}, \vec{x_j})$ and the problem of finding a data domain description is given by

$$L = \sum_i \alpha_i K(\vec{x_i}, \vec{x_i}) - \sum_{i,j} \alpha_i \alpha_j K(\vec{x_i}, \vec{x_j}), \tag{3.35}$$

$$\sum_i \alpha_i = 1, \tag{3.36}$$

$$0 \leq \alpha_i \leq C, \forall i. \tag{3.37}$$

After the cost function (3.35) with constraints is solved, the distance between test data $\vec{z}$ and the center of domain in feature space can be calculated as follows

$$D^2(\vec{z}) = 1 - 2\sum_i \alpha_i K(\vec{z}, \vec{x_i}) + \sum_{i,j} \alpha_i \alpha_j K(\vec{x_i}, \vec{x_j}). \tag{3.38}$$

Figure 3.7 shows the domain by kernel SVDD using Gaussian kernel $K(\vec{x_i}, \vec{x_j}) = exp(-\|\vec{x_i} - \vec{x_j}\|^2/s^2)$, where the sample data are generated from the combination of 3 Gaussian distribution in two dimensional space. It shows that the domain in high dimensional feature space is more suitable than that in original input space for describing the data domain that characterizes the non-linearity of data structure.

3.2.4 Support vector domain density description

Since SVDD can determine whether a test point is within the domain by calculating the distance between test point and the center of the domain, it can

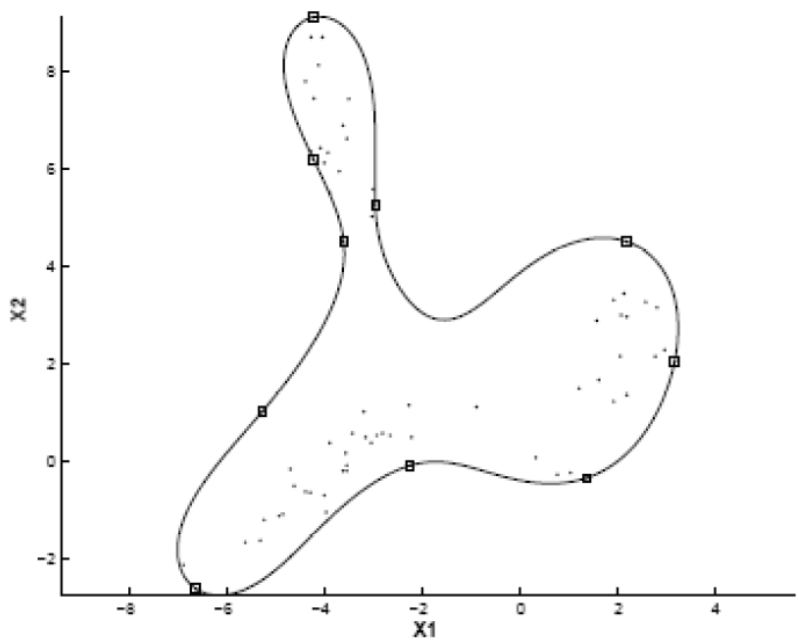


Figure 3.7: The kernel SVDD [74]. The domain by kernel SVDD where the sample data are generated from the combination of 3 Gaussian distribution in two dimensional space.

be applied to multi-class problems by obtaining each class boundary. When the training data between classes are separable, we can determine the class of data by choosing the class where the test data is included. However, since the classification problems often contain overlapping classes, the SVDD is not suitable for most multi-class classification task. To solve this problem [72, 74] proposed the SVDDD method which is based on the assumption that the probability of a sample to be in a class is proportional to its distance from the domain center (previously trained using SVDD in equation (3.38)) of that class. The following lemma shows the relation between distance function and probability density.

Lemma 3.2.1. *Assume that* $\vec{x_a}$, $\vec{x_b}$ *are drawn from unknown probability density function* $p(\vec{x})$ *in one class. Then, the distance function in* (3.38) *has the following property as*

$$\text{If } p(\vec{x_a}) \geq p(\vec{x_b}),\ D^2(\vec{x_a}) \leq D^2(\vec{x_b}). \tag{3.39}$$

Proof. For explaining the relation between SVDD and density, given n sample data $\vec{x_i} \subset X$ where X is drawn from probability density function $p(\vec{x})$ which is fixed but unknown, consider the following case

$$p(\vec{x_a}) \geq p(\vec{x_b}), \tag{3.40}$$

where $\vec{x_a}$ and $\vec{x_b}$ are elements of data set X. If equation (3.35) from data set X is maximized, because the quadratic optimization in SVDD assigns the large weight to the sample in low density region [71], we can obtain the inequality as follows

$$\alpha_a \leq \alpha_b. \tag{3.41}$$

By differentiating equation (3.35) for maximizing the cost function, we have

$$\frac{dL}{d\alpha_i} = 1 - \sum_j \alpha_j K(\vec{x_i}, \vec{x_j}) - \alpha_i = 0. \tag{3.42}$$

From equation (3.41) and (3.42), the following inequality can be given by

$$1 - \alpha_a \geq 1 - \alpha_b, \tag{3.43}$$

$$\sum_i \alpha_i K(\vec{x_a}, \vec{x_i}) \geq \sum_i \alpha_i K(\vec{x_b}, \vec{x_i}). \tag{3.44}$$

By multiplying both sides of equation (3.44) by -2 and adding them by constant, $1 + \sum_{i,j} \alpha_i \alpha_j K(\vec{x_i}, \vec{x_j})$, this leads to

$$\begin{aligned} 1 - 2\sum_i \alpha_i K(\vec{x_a}, \vec{x_i}) \quad &+ \quad \sum_{i,j} \alpha_i \alpha_j K(\vec{x_i}, \vec{x_j}) \\ &\leq \quad 1 - 2\sum_i \alpha_i K(\vec{x_b}, \vec{x_i}) + \sum_{i,j} \alpha_i \alpha_j K(\vec{x_i}, \vec{x_j}). \end{aligned} \tag{3.45}$$

The above inequality indicates that if $p(\vec{x_a}) \geq p(\vec{x_b})$, $D^2(\vec{x_a}) \leq D^2(\vec{x_b})$. □

Therefore, the decision function can be designed by employing the above relation between probability density and the data description as

$$\bar{p}(\vec{x}|\sigma) = \frac{1}{(2\pi\sigma^2)^{d/2}} exp(-\frac{D^2(\vec{x})}{2\sigma^2}). \tag{3.46}$$

Then, the parameter σ of equation (3.46) can be estimated as following lemma.

Lemma 3.2.2. *The maximum likelihood estimate of σ is given by*

$$\hat{\sigma}^2 = \frac{1}{n_{sv} d} \sum_{k=1}^{n_{sv}} \|\phi(\vec{x_{s_k}}) - \vec{a}\|^2, \tag{3.47}$$

where n_{sv}, d, and $\vec{x_{s_k}}$ denote the number of support vectors, sample dimension, and support vector respectively.

Proof. Suppose that X drawn independently from $p(\vec{x}|\sigma)$ contains n samples, $\vec{x_1}, \ldots, \vec{x_n}$. Then, because the samples were drawn independently, we have

$$\bar{p}(X|\sigma) = \prod_{i=1}^{n} \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{D^2(\vec{x_i})}{2\sigma^2}\right). \tag{3.48}$$

Instead of maximizing $\bar{p}(X|\sigma)$, we maximize the following log-likelihood

$$\begin{aligned} \mathcal{L}(\sigma) &\equiv \ln \bar{p}(X|\sigma), \\ &= \sum_{i=1}^{n} \ln \bar{p}(\vec{x_i}|\sigma), \\ &= \sum_{i=1}^{n} \left(\ln \frac{1}{(2\pi\sigma^2)^{d/2}} - \frac{D^2(\vec{x_i})}{2\sigma^2}\right). \end{aligned} \tag{3.49}$$

Then, the derivative of $\mathcal{L}(\sigma)$ with respect to σ is given by

$$\nabla_\sigma \mathcal{L} = \sum_{i=1}^{n} \left(-\frac{d}{\sigma} + \frac{D^2(\vec{x_i})}{\sigma^3}\right). \tag{3.50}$$

Therefore, $\hat{\sigma}^2$ can be obtained as follows by setting $\nabla_\sigma \mathcal{L}$ to zero

$$\hat{\sigma}^2 = \frac{1}{nd} \sum_{i=1}^{n} D^2(\vec{x_i}) = \frac{1}{nd} \sum_{i=1}^{n} \|\phi(\vec{x_i}) - \vec{a}\|^2, \tag{3.51}$$

where $\vec{a} = \sum_{i=1}^{n} \alpha_i \phi(\vec{x_i})$. The proposed method uses only support vectors for estimating $\hat{\sigma}$ since the center of domain is described in terms of support vectors. Then,

$$\hat{\sigma}^2 = \frac{1}{n_{sv} d} \sum_{k=1}^{n_{sv}} \|\phi(\vec{x_{s_k}}) - \vec{a}\|^2, \tag{3.52}$$

where k and n_{sv} denote the index and the number of support vectors, respectively, and $\vec{x_{s_k}}$ is kth support vector. □

Step 1. Get feature space distance function from SVDD training,
where N_C is the number of classes.

$$D_l^2(\vec{x})=1-2\sum_i \alpha_i K(\vec{x},\overrightarrow{x_i}) + \sum_{i,j} \alpha_i \alpha_j K\left(\overrightarrow{x_i},\overrightarrow{x_j}\right), l = 1, \dots, N_C$$

Step 2. Estimate parameter of proposed SVDDD function for each class C_l,
where d is the dimension space of data.

$$\overline{p_l}(\vec{x}|C_l) = \frac{1}{\left(2\pi\sigma_l^2\right)^{d/2}} exp\left(-\frac{D_l^2(\vec{x})}{2\sigma_l^2}\right), \sigma_l^2=\frac{R_l^2}{d}, l = 1, \dots, N_C$$

Step 3. Use SVDDD function as pseudo PDF,
where n_l is the number of data in class C_l and N is the total number of data.

$$class\ of\ \vec{z} = arg \max_{l=1,\dots,N_C} P(C_l)\overline{p_l}(\vec{z}|C_l), P(C_l) = \frac{n_l}{N}$$

Figure 3.8: The process of SVDDD. The process of SVDDD is compactly summarized as 3 steps.

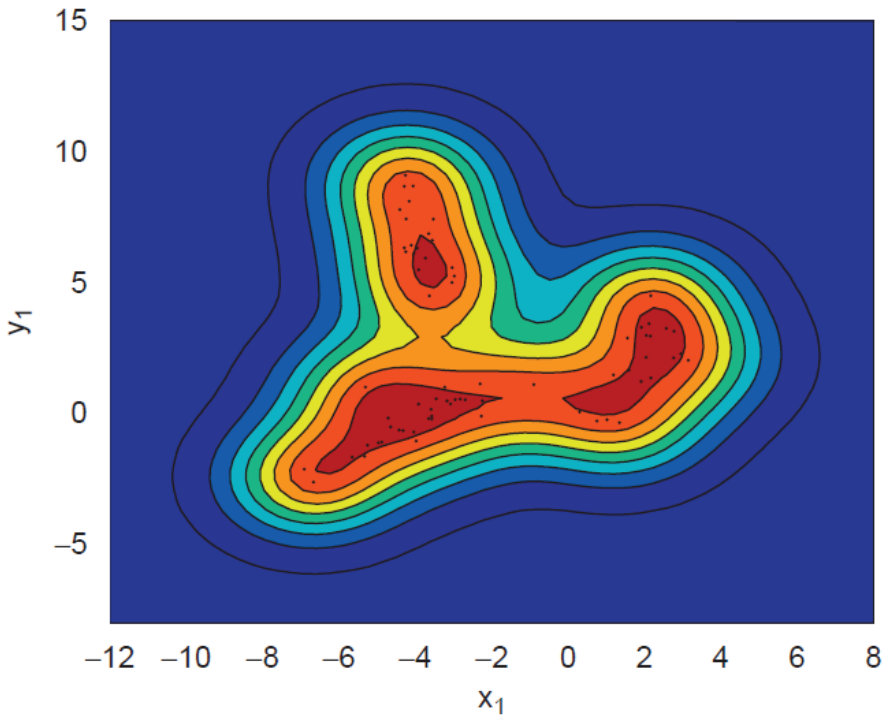


Figure 3.9: The density information of SVDDD [74]. The example of the density from the data in Figure 3.7. We can know that 3 high density region and the skewness of data can be described by the proposed method.

In Figure 3.8, the process of SVDDD is compactly summarized as 3 steps. Figure 3.9 shows the example of the density from the data in Figure 3.7. In this Figure, we can know that 3 high density region and the skewness of data can be described by the proposed method.

3.3 Proposed method

Figure 3.10 and Figure 3.11 illustrates the overall framework of our proposed approach. The training procedure is shown in Figure 3.10. First, 3D-S volumes are constructed by back-projecting the silhouettes from 5 cameras of different views (Figure 3.1) with the method given by [40]. To catch variation of body shape during actions, we obtain 2D-S silhouettes in N number of candidate views by projecting the 3D-S volume sequence with the camera matrix. Furthermore, to capture the interior movement of an object, we propose 4D-STIPs which are calculated from 3D-S volume sequences by extending the method based on 3D-STIPs [47] which finds variation points in the space [x,y] and time [t]. The 4D-STIPs are then projected to any desired candidate views. The projected silhouettes and 4D-STIPs are used to make the motion and non-motion features which correspond to the moving and stationary parts of an action. To describe moving parts of an action, MHIs [69] are constructed from both projected silhouettes and 4D-STIPs. Silhouettes reflect the changes in outlines of an object and the STIPs encode the variations occurring inside of an object. Similarly NMHIs are used, which encode the history of stationary parts of an action. While other methods rely on only moving aspects of an action, we use both aspects (moving and non-moving) to design features. For training, the dimension of MHIs and NMHIs is reduced by CA-PCA [70] which uses class information for dimension reduction. We use the action label of the training data for CA-PCA. The final features with reduced

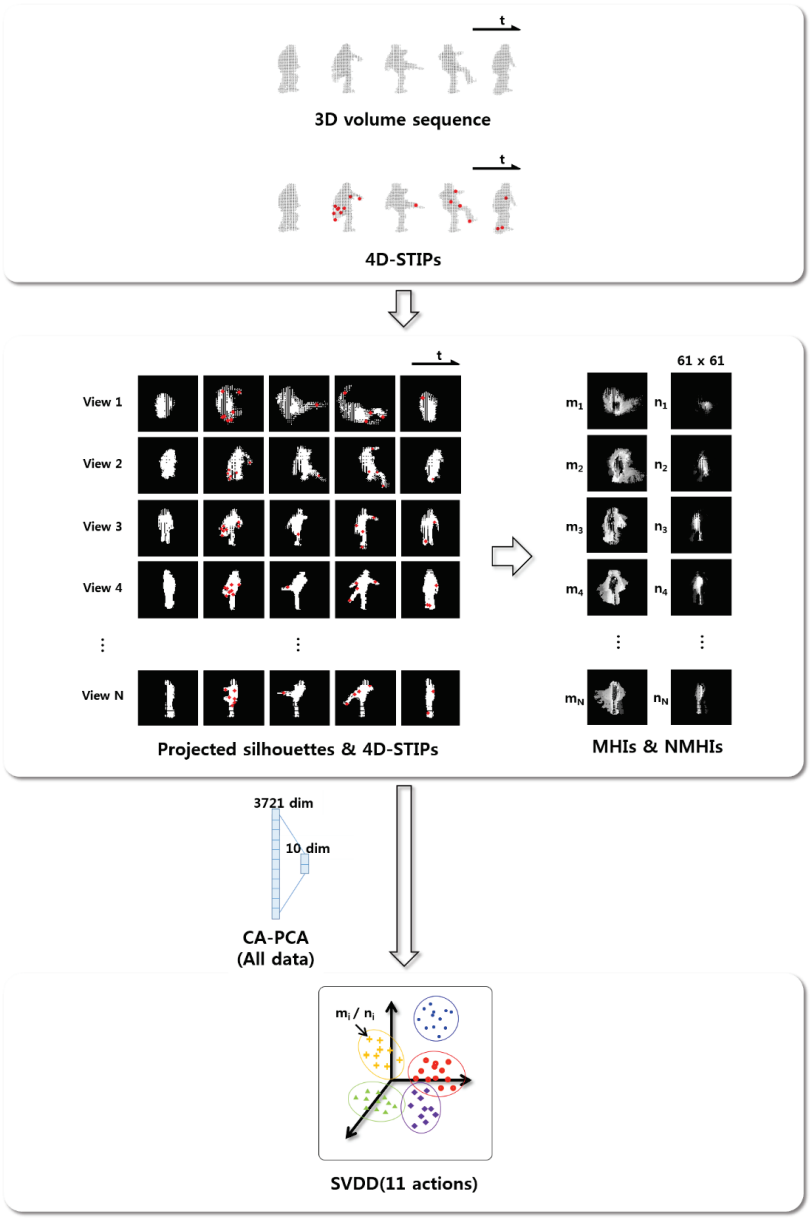


Figure 3.10: Overview for training step. In training step, reconstructed 3D-S volumes and 4D-STIPs are projected to N number of candidate views. The projected silhouettes and 4D-STIPs are then represented by MHIs and NMHIs. We reduce the dimension of MHIs and NMHIs using CA-PCA with action label and then train the resulting features with SVDD. The features from MHIs and NMHIs are trained separately by SVDD.

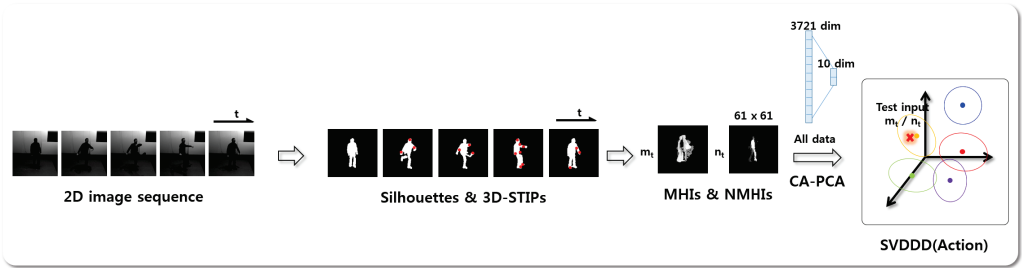


Figure 3.11: Overview for testing step. In testing step, silhouettes and 3D-STIPs are extracted from the test image sequence and reduced MHIs and NMHIs are tested by SVDDD. Final score for each class is calculated by summing the SVDDD scores of the features from MHIs and NMHIs.

dimension are trained using SVDD [71].

In the test stage (Figure 3.11) silhouettes and 3D-STIPs are extracted using an image sequence from the test camera view. For action recognition, both features (motion and non-motion) are tested separately by SVDDD [72]. The score for action recognition is calculated for each action class by summing the SVDDD scores of two features.

3.3.1 Silhouettes

Silhouettes are used as features in many action recognition approaches [27, 35]. By projecting the scale-normalized-3D-S volumes with camera matrices, the silhouettes in candidate 2D-S views can be obtained (Figure 3.12). From [68], the point x in an image plane (corresponding to X_W in world coordinates) can be calculated by multiplying camera matrix P as

$$x = PX_W = KR[I| - \tilde{C}]X_W, \tag{3.53}$$

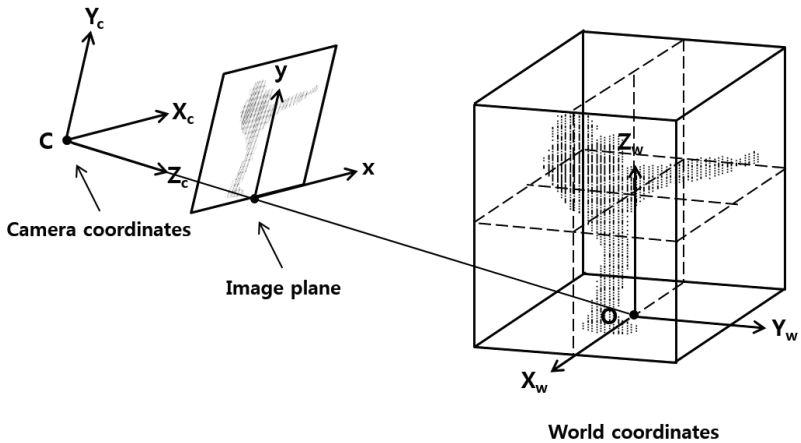


Figure 3.12: Projection. Scale-normalized-3D-S volumes are projected to desired image planes using camera matrix P. By changing P, the silhouettes in a large numbers of views can be easily produced.

$$\tilde{C} = r \begin{bmatrix} \sin\varphi\cos\theta \\ \sin\varphi\sin\theta \\ \cos\varphi \end{bmatrix}, K = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}, R = inv(\begin{bmatrix} X_{\tilde{C}} & Y_{\tilde{C}} & Z_{\tilde{C}} \end{bmatrix}). \quad (3.54)$$

In equation (3.54), the inhomogeneous coordinates of the camera center $\tilde{C}$ at desired view are defined in the spherical coordinate system as (r, θ, φ). K is the calibration matrix and R is the rotation matrix of camera coordinates. In the calibration matrix, f and (p_x, p_y) are focal length and principal point (image center) of the camera respectively. The rotation matrix can be calculated by inverse of the matrix consisting of camera axes $X_{\tilde{C}}, Y_{\tilde{C}}, Z_{\tilde{C}}$. Different silhouettes from different views are constructed by changing P and projecting the 3D-S volume sequence with this camera matrix. This can produce numerous silhouettes, corresponding to those many views, which exceeds the actual number of camera views.

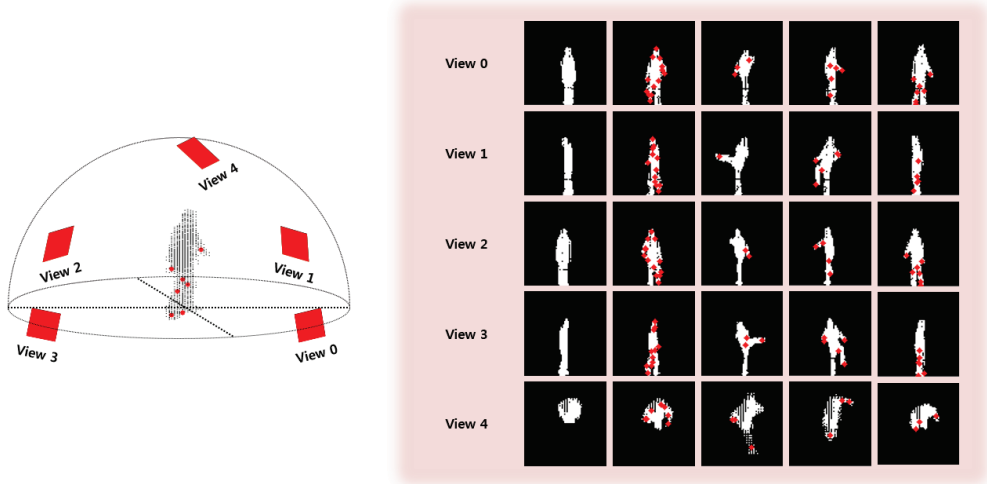


Figure 3.13: Projection example. 3D-S volumes and 4D-STIPs are projected to predefined image planes using camera matrices. The left side of the Figure shows the positions of predefined 5 image planes (i.e. N is 5 in Figure 3.10) and the right side shows the sequence of projected silhouettes and 4D-STIPs at each view.

3.3.2 Space-time interest points

Actions can be recognized using silhouettes to some extent, however, they do not properly describe the interior motion and shape. To solve this problem, we use 4D-STIPs to capture the interior movement of an object for better recognition performance. These features are constructed by extending the concept of widely used 3D-STIPs. We extract 4D-STIPs from 3D-S volume sequence and project them to the candidate views for training. Here, only visible points which are not screened by occupancy grids of 3D-S volume are projected to the image plane. For example, if certain 4D-STIPs are located at the backside of body and the image plane is in the front side, then those 4D-STIPs are regarded as being occluded by the body and they are not projected to the image plane. Figure 3.13 shows the projection examples of 3D-S volumes and 4D-STIPs using camera matrices. The left side of the Figure shows the positions of predefined 5

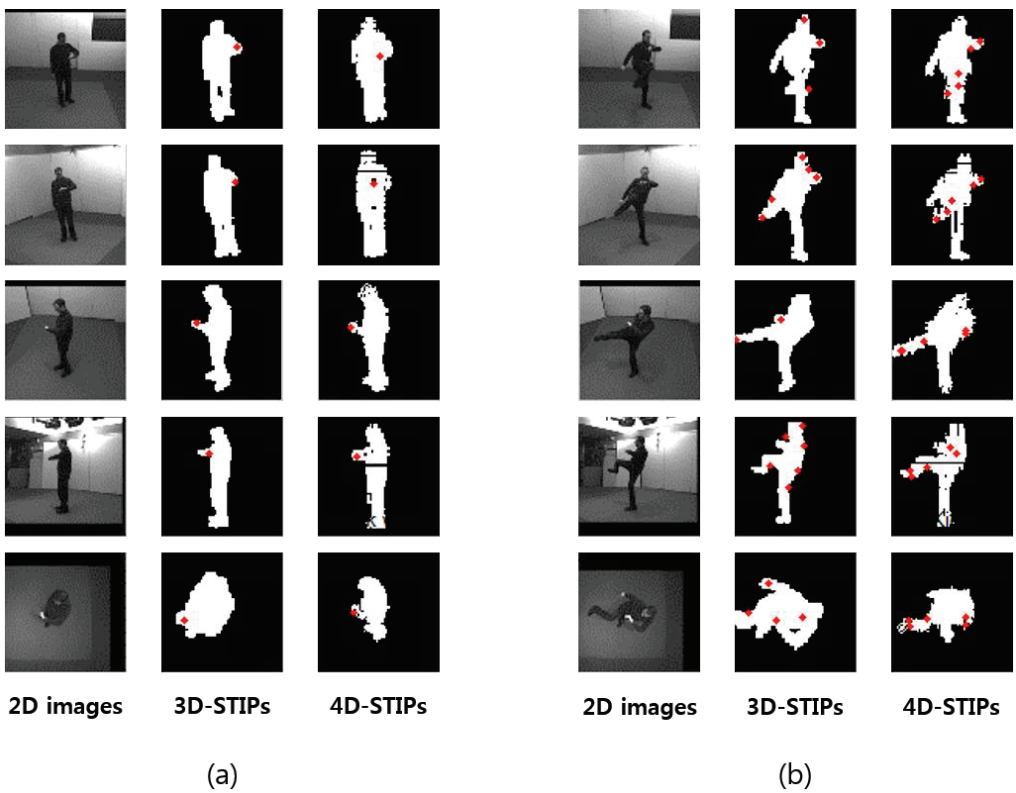


Figure 3.14: The comparison of 3D-STIPs and 4D-STIPs. We compare 3D-STIPs and 4D-STIPs projected to similar view image planes of action (a) 'check watch' and (b) 'kick'. The 3D-STIPs and projected 4D-STIPs are strictly different, however, both capture the important moments of an action.

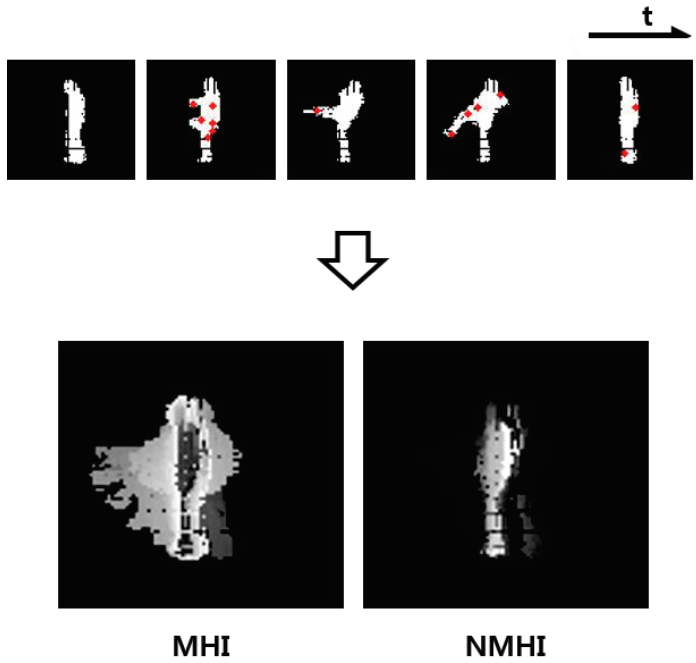


Figure 3.15: The process for MHI and NMHI. MHI encodes the history of motion occurrences in an image while NMHI provides an information about stationary parts of an object.

image planes (i.e. N is 5 in Figure 3.10) and the right side shows the sequence of projected silhouettes and 4D-STIPs at each view. On the contrary, in the test step, we extract 3D-STIPs in test video because only a single test video is available from an arbitrary view. In Figure 3.14, we compare 3D-STIPs and 4D-STIPs projected to image planes of similar view. The 3D-STIPs and projected 4D-STIPs are strictly different, however, both capture the important moments of an action. The detailed explanation about STIPs are described in chapter 2.

3.3.3 Motion history images and Non-motion history images

To reduce the data size and represent actions compactly, we adopt the notion of MHIs which encodes the history of motion occurrences in an image. However,

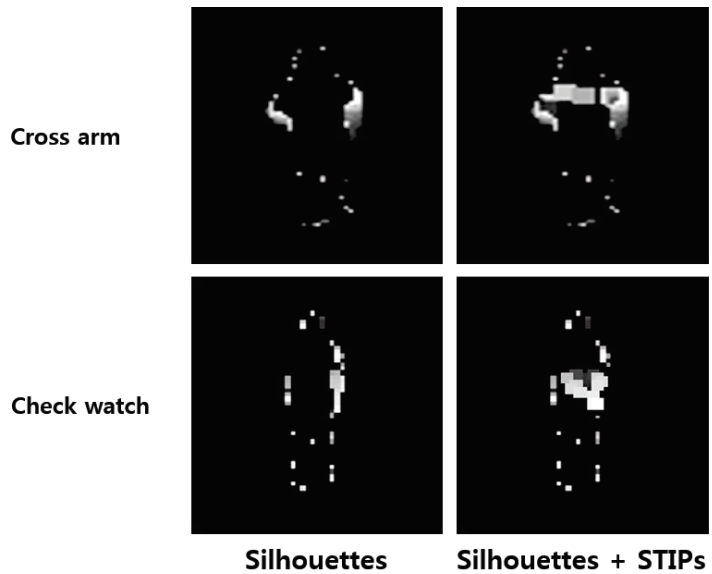


Figure 3.16: MHIs of silhouettes and STIPs. MHIs of 'cross arm' and 'check watch' using silhouettes (Left) and using both silhouettes and STIPs (Right).

only MHIs cannot guarantee a good performance when moving parts of an object show similar patterns. For this reason, we propose NMHIs and combine it with MHIs to add an information from stationary parts of an object in the description of the particular action class. By combining the moving as well as non-moving information, the recognition performance is increased. Figure 3.15 shows the process of MHI and NMHI.

To make MHIs for moving part of an object, the spatio-temporal positions of motion occurrence are captured, either by variations of silhouettes or by positions of STIPs at time t. Figure 3.16 shows two different MHIs of the action 'cross arm' using only silhouettes (left) and using both silhouettes and STIPs (right). We can notice that the MHIs using only silhouettes cannot reflect the movement occurring in front of the actor while the MHIs using both silhouettes and STIPs can also

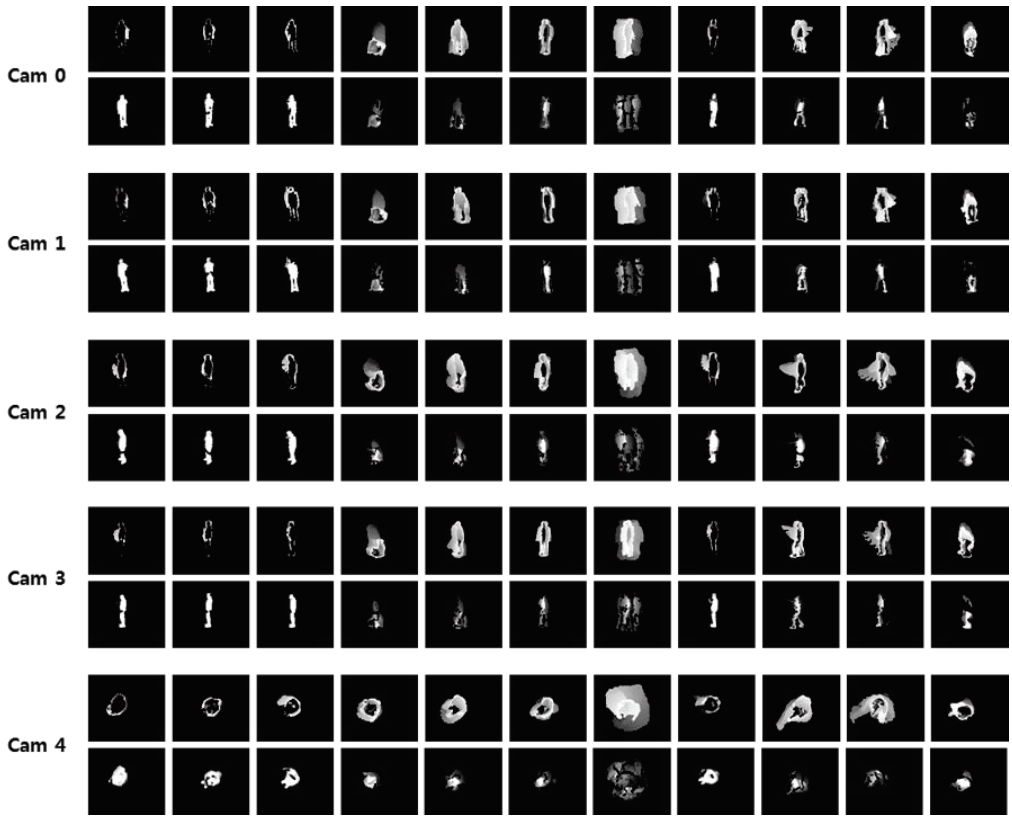


Figure 3.17: Examples of MHIs and NMHIs. Examples of MHIs and NMHIs for 11 actions and 5 camera positions in IXMAS dataset.

reflect the movement of the arm inside the silhouettes. The MHIs is defined as

$$h_M(x,y,t) = \begin{cases} \tau & \text{if } D(x,y,t) = 1 \\ max(0, h_M(x,y,t-1) - 1) & \text{otherwise} \end{cases}, \qquad (3.55)$$

where τ is the maximum duration for which a motion is stored and $D(x,y,t)$ is equal to 1 if there exists motion at (x,y,t) in an image sequence and 0 otherwise.

While the MHIs capture motion information within image sequences, we also encode non-motion information by introducing NMHIs, which have a large value

when certain part is stationary for long time. The NMHIs are defined as

$$h_{NM}(x,y,t) = \begin{cases} h_{NM}(x,y,t-1) & \text{if } D(x,y,t)=1 \\ h_{NM}(x,y,t-1)+1 & \text{otherwise} \end{cases} . \tag{3.56}$$

Using information of the stationary part can provide additional evidence in many cases. For example, MHIs of 'kick' and 'get up' are similar, however, they are different in NMHIs since an upper body of 'kick' action is stationary while that of 'get up' action is not. Figure 3.17 shows examples of MHIs and NMHIs for 11 actions and 5 camera positions in IXMAS dataset [2].

3.3.4 Training and Testing

Figure 3.18 shows the process of training step for view invariant action recognition. When we make MHIs and NMHIs, we divide the total sequence into L sections and calculate MHIs and NMHIs for each section as in the Figure 3.19. With this scheme, each action sequence is represented by L number of MHIs and L number of NMHIs. The reason why we divide each action sequence into small one is that any interesting motion history at a given location can be obliterated by recent movement. Then, each section of MHI (NMHI) is represented as a feature vector by concatenating its columns. The same sections of vectorized MHIs (NMHIs) for all action class are collected and the dimension is reduced with CA-PCA using action class labels. After reducing the feature (derived from MHIs and NMHIs) dimension, they are separately trained using SVDD. Finally, we obtain $2L$ number of CA-PCA principal axis and $2L$ number of SVDD results (the center and the radius of each action domain).

Figure 3.20 shows the process of testing step for view invariant action recognition. In testing step, similarly we compute L number of MHIs and L number of NMHIs using silhouettes and 3D-STIPs from 2D-S image sequence and they are

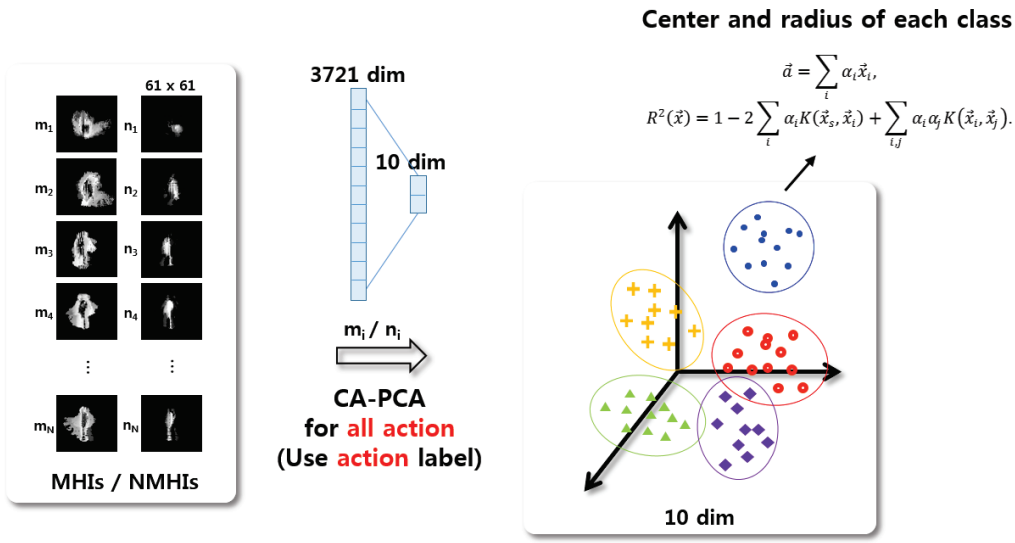


Figure 3.18: Training for view invariant action recognition. The dimension of MHIs (NMHIs) for all action class is reduced with CA-PCA using action class labels. After reducing the feature (derived from MHIs and NMHIs) dimension, they are separately trained using SVDD.

vectorized. Then, each section of MHI (NMHI) are reduced with corresponding CA-PCA principal axis and recognized the actions with SVDDD classifier. The SVDDD algorithm is based on the assumption that the probability of a sample to be in a class is proportional to its distance from the domain center (previously trained using SVDD) of that class. We use the sum of the $2L$ number of SVDDD scores (MHIs and NMHIs) as a final decision score and assign the samples of input test video to the class of the highest decision score.

3.4 Experimental results

The performance of the proposed algorithm was evaluated on the IXMAS (Figure 3.1) and our own SNU datasets (Figure 3.2). The IXMAS dataset contains videos of total 11 action categories captured from 5 different views. The 3D-S volumes

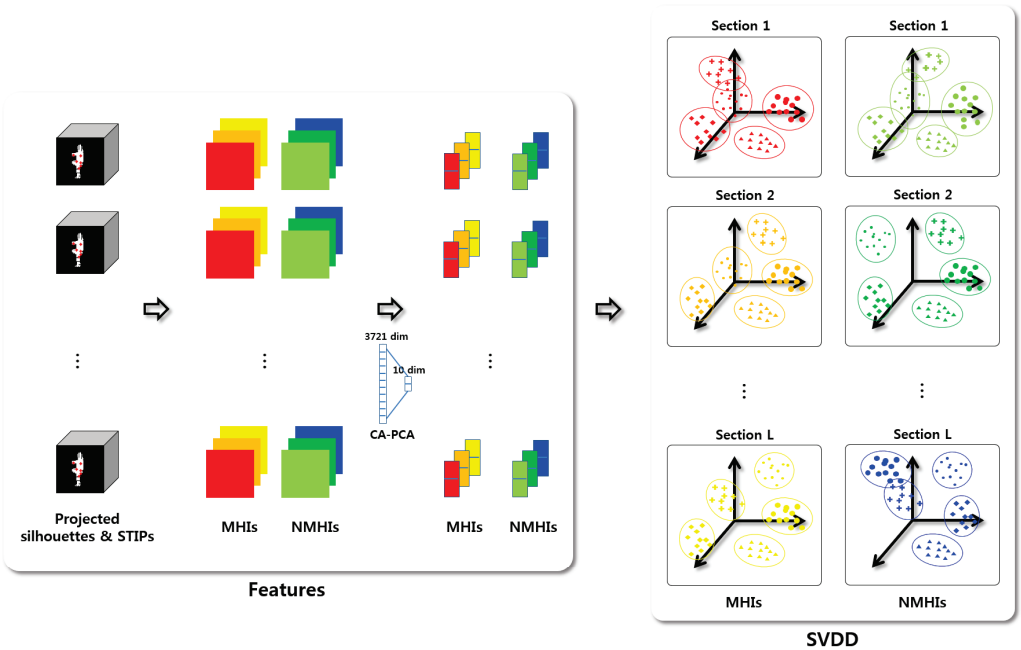


Figure 3.19: Sections of MHIs and NMHIs. We divide the total sequence into L sections and calculate MHIs and NMHIs for each section. The same sections of vectorized MHIs (NMHIs) for all action class are collected and the dimension is reduced with CA-PCA using action labels. After reducing the feature (derived from MHIs and NMHIs) dimension, they are separately trained using SVDD. Finally, we obtain $2L$ number of CA-PCA principal axis and $2L$ number of SVDD results.

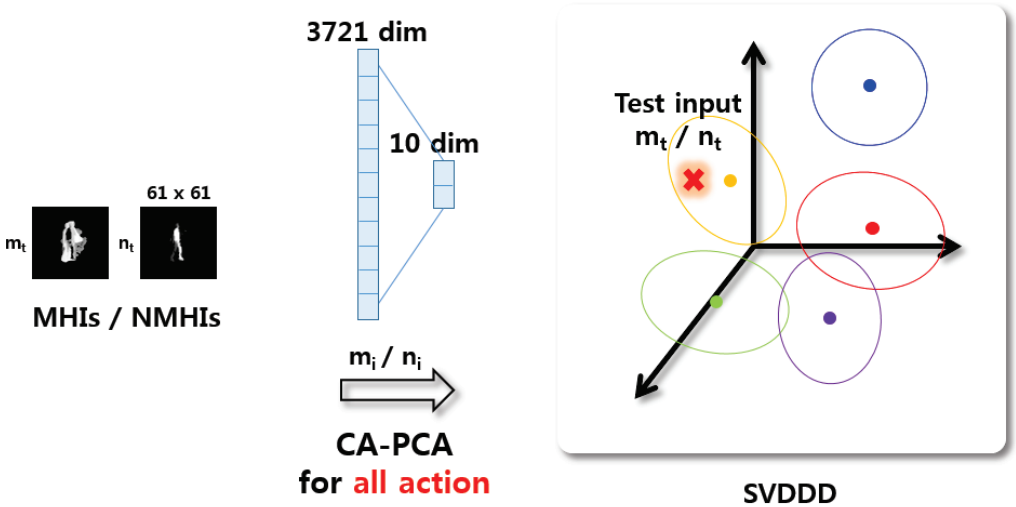


Figure 3.20: Testing for view invariant action recognition. The dimension of MHIs (NMHIs) is reduced with CA-PCA calculated in training step using action label. After reducing the feature (derived from MHIs and NMHIs) dimension, they are separately tested using SVDDD.

were reconstructed by computing the visual hulls via back projecting the multi-view 2D-S silhouettes. In this work, we used the reconstructed 3D-S volumes provided along with the multi-view videos. The each action was performed 3 times by 12 actors. The position and the orientation of actors were chosen arbitrarily. To show robustness of the proposed algorithm, we collected a new SNU dataset which contains 4 different views for 11 actions as in IXMAS, and each was performed 3 times by 15 actors. We developed the proposed algorithm using Matlab on PC with i5 core2duo, 3.3GHz, and 4GB RAM. The total training time for 15840 ($11 actions \times 12 actors \times 3 times \times 40 views$) features was around 3 days, and the most of the training time was spent for extracting 4D-STIPs. The testing time for one video input with 40 frames is around 13s ($3.08 fps$), and the most of the testing time are for calculating 3D-STIPs.

To compute training features, we used scale-normalized 3D-S volumes (64 $\times$

64×64) and calculated 4D-STIPs on them. The space interest points (SIPs) were calculated within convoluted volumes with $\sigma_l^2 = 0.5$ and integration scale $\sigma_i^2 = 2\sigma_l^2$. T_s was adjusted to get $3 \sim 6$ SIPs in each frame. For convoluting volume sequences along time axis the parameters were set as $\tau_l^2 = 1$ and $\tau_i^2 = 4\tau_l^2$. The constant c, which determines T_{st}, was set to 0.3. For all datasets, the projected image size was 61×61 and the number of frames was fixed to 40. When we made MHIs and NMHIs, we divided the total sequence into 3 (i.e. L is 3 in Figure 3.19) and calculated MHIs and NMHIs for each part. With this scheme, each action sequence was represented by 3 MHIs and 3 NMHIs. The dimension of each MHIs and NMHIs was reduced to 10 using CA-PCA. We trained 6 MHIs and NMHIs separately using SVDD and the parameter C, which controls the error rate and complexity of the class boundary, was set to 0.9 and variance of Gaussian kernel was set to 0.09.

During testing, the features were computed by using scale-normalized image sequences (161×161) and then 3D-STIPs were computed. The SIPs were calculated within convoluted images with $\sigma_l^2 = 6.25$ and integration scale $\sigma_i^2 = 3\sigma_l^2$. T_s was adjusted to have $7 \sim 10$ SIPs in each frame. For convoluting image sequences along time axis the parameters were set as $\tau_l^2 = 1$ and $\tau_i^2 = 4\tau_l^2$. The constant c, which determines T_{st}, was set to 0.05 and the image sequence was rescaled to $61 \times 61 \times 40$. Similar to training step, we made 3 MHIs and 3 NMHIs for a test image sequence. The dimension of each MHIs and NMHIs was reduced to 10 using CA-PCA. Finally, the 6 SVDDD scores (3 for MHIs and 3 for NMHIs) were calculated separately and summed together for the final decision score.

We proposed NMHIs to add the information from stationary parts of an object in the description of the particular action class. To verify the effects of the NMHIs, we measured the classification performance with and without NMHIs. In table 3.1, we show recognition rates per camera. Without NMHIs, large degradation

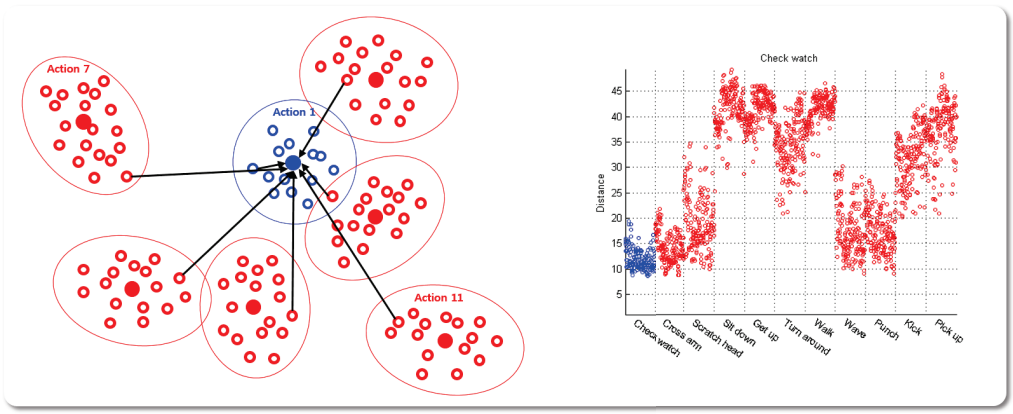
Table 3.1: Effect of NMHIs. To verify the effects of the NMHIs, we measured the classification performance with and without NMHIs. Without NMHIs, large degradation in performance (5.7%) is shown which implies NMHIs take a significant role in recognition performance.

Algorithm	Avg	Cam 0	Cam 1	Cam 2	Cam 3	Cam 4
MHIs	65.8	71.3	72.4	70.3	72.4	42.8
MHIs + NMHIs	71.5	72.1	73.8	77.1	77.4	57.0

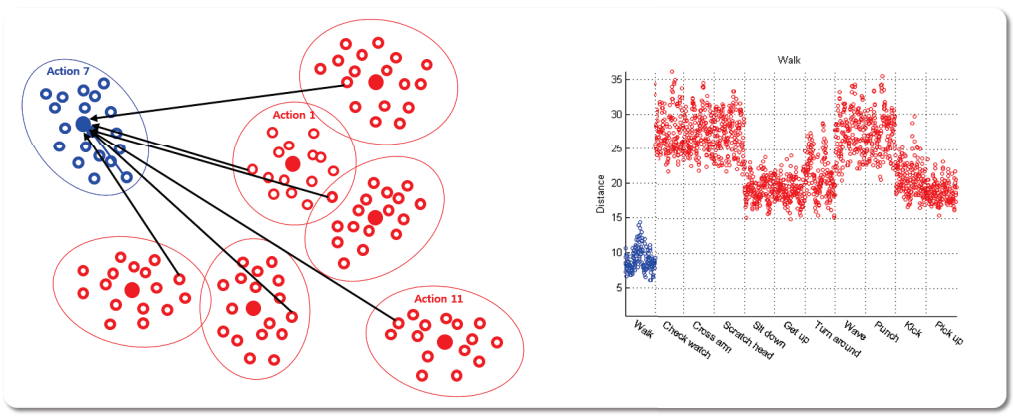
in performance (5.7%) is shown which implies NMHIs take a significant role in recognition performance.

To analysis the separability of the SVDD domains in high dimensional feature space, we measured the within-class and between-class distances for all actions. In Figure 3.21, the features included in the chosen action are depicted as blue dots while the features of all actions except the chosen action are depicted as the red dots. Here, we used the inverse of SVDDD score (equation (3.46)) as a distance since the inverse of SVDDD score is proportional to the distance from a feature to the center of SVDD domain. Figure 3.21 (a) shows the distances from all features to SVDD center of 'check watch' action and Figure 3.21 (b) shows the distances from all features to SVDD center of 'walk' action in feature space. In Figure 3.21 (a), since the action 'check watch' is similar with actions which use arms and hands such as 'cross arm', 'scratch head', 'wave', and 'punch', the distance from that features to 'check watch' SVDD domain center is similar with that of 'check watch' features. However, in Figure 3.21 (b), since the action 'walk' is much different from other actions, we can see that the SVDD domain of 'walk' is well separable. Figure 3.22, 3.23, 3.24 show the within-class and between-class distances of all 11 actions.

For a direct comparison to the results reported in [63, 25, 35, 36, 37, 38, 39], we first experimented with IXMAS dataset using leave-one-out strategy. With



(a)



(b)

Figure 3.21: Within-class and between-class distances. Within-class (red dots) and between-class (blue dots) distances of all features. (a) The distances from all features to SVDD center of 'check watch' action in feature space. Since the action 'check watch' is similar with actions which use arms and hands such as 'cross arm', 'scratch head', 'wave', and 'punch', the distance from that features to 'check watch' SVDD domain center is similar with that of 'check watch' features. (b) The distances from all features to SVDD center of 'walk' action in feature space. Since the action 'walk' is much different from other actions, we can see that the SVDD domain of 'walk' is well separable.

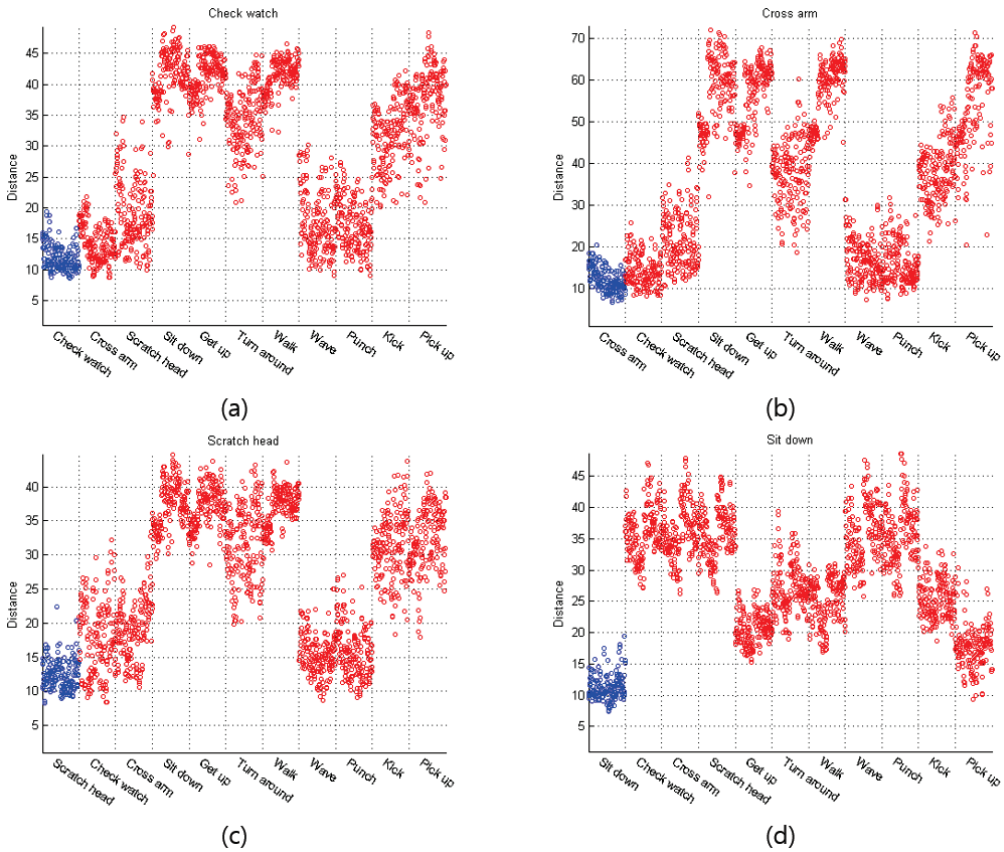


Figure 3.22: Within-class and between-class distances. Within-class (red dots) and between-class (blue dots) distances of action (a) check watch, (b) cross arm, (c) scratch head, and (d) sit down.

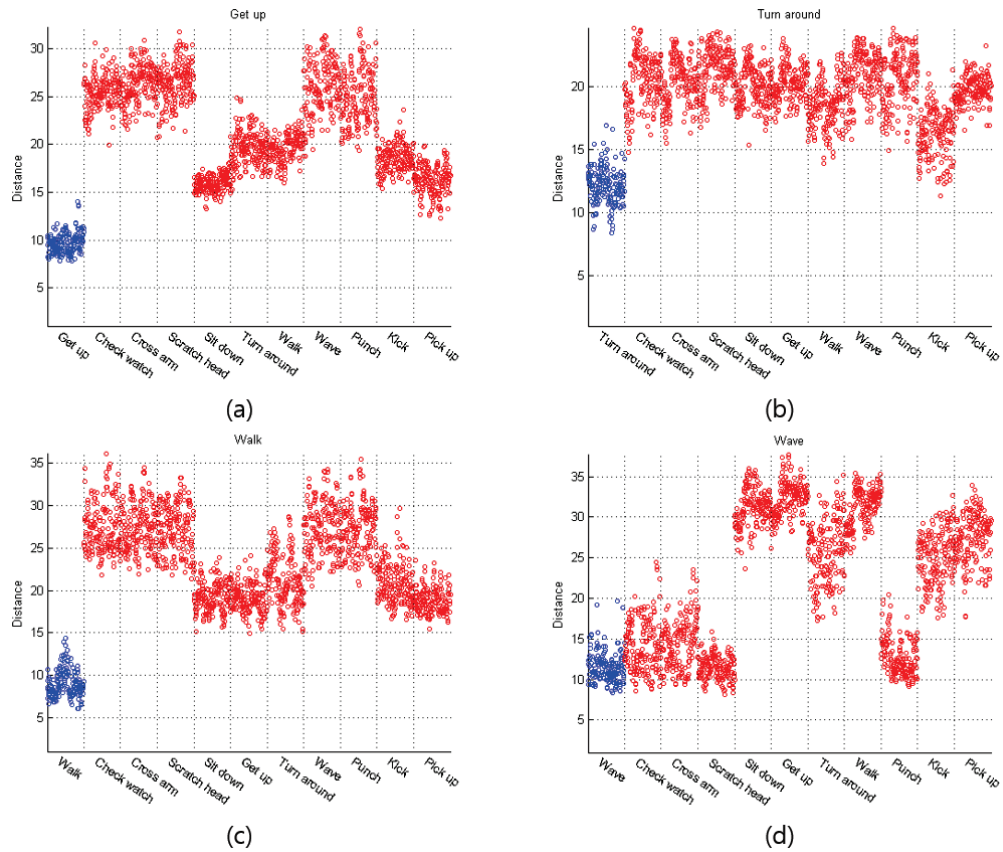


Figure 3.23: Within-class and between-class distances. Within-class (red dots) and between-class (blue dots) distances of action (a) get up, (b) Turn around, (c) walk, and (d) wave.

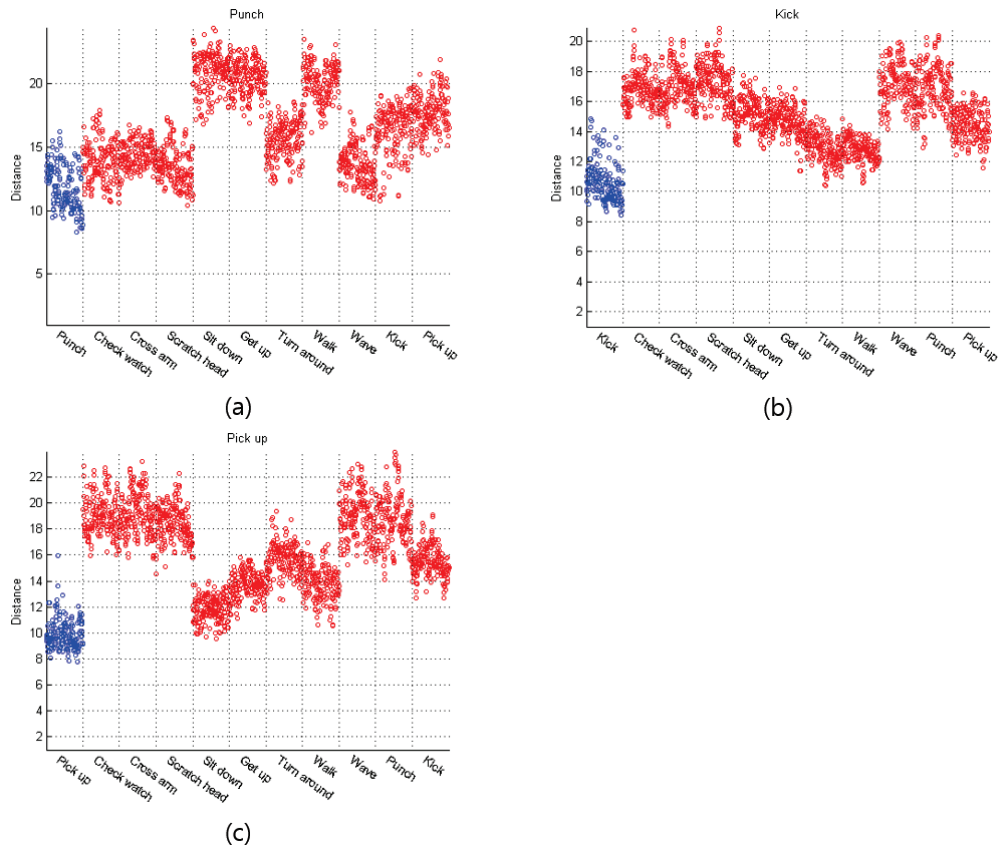


Figure 3.24: Within-class and between-class distances. Within-class (red dots) and between-class (blue dots) distances of action (a) punch, (b) kick, and (c) pick up.

Table 3.2: Experimental result of IXMAS dataset. The recognition rates (in %) per camera when we train and test using IXMAS dataset.

IXMAS	Avg	Cam 0	Cam 1	Cam 2	Cam 3	Cam 4
Ours (known view)	77.2	78.9	81.1	82.6	81.8	61.6
Ours (unknown view)	71.5	72.1	73.8	77.1	77.4	57.0
Wu (2012)	91.1	94.4	91.0	91.7	89.6	88.9
Kaaniche (2011)	71.7	75.3	67.1	69.5	75.0	—
Weinland (2010)	83.5	87.0	88.3	85.6	87.0	69.7
Reddy (2009)	66.5	69.6	69.2	62.0	65.1	—
Yan (2008)	64.0	72.0	53.0	68.0	63.0	—
Junejo (2008)	72.7	74.8	74.5	74.8	70.6	61.2
Weinland (2007)	57.9	65.4	70.0	54.3	66.0	33.6

IXMAS dataset, we experimented in two cases: In the first case (Ours (known view) in Table 3.2), training were performed under the same experimental conditions with the existing methods for fare comparison, where we used images from 5 different known camera views in IXMAS dataset. And we tested an image whose view is included in the training. In the second case (Ours (unknown view) in Table 3.2), instead of the known view images, we used the proposed generalized 4D-ST motion features to train the model in arbitrary views. When projecting the 4D-ST motion features into arbitrary view for training, we do not know the exact camera matrices to make the 5 different camera views in IXMAS dataset. Therefore features in exact testing views may not be included in training and only features in similar views were included. This is a reasonable testing for completely view invariant action recognition method, where test view is completely unknown. To cover the various orientation of people and the camera view, for each action the 3D-S volumes were projected on 40 different image planes (i.e. N is 40 in Figure 3.10).

In Table 3.2, we show recognition rates per camera and compare it against other methods. We have excluded the experimental results related to transfer

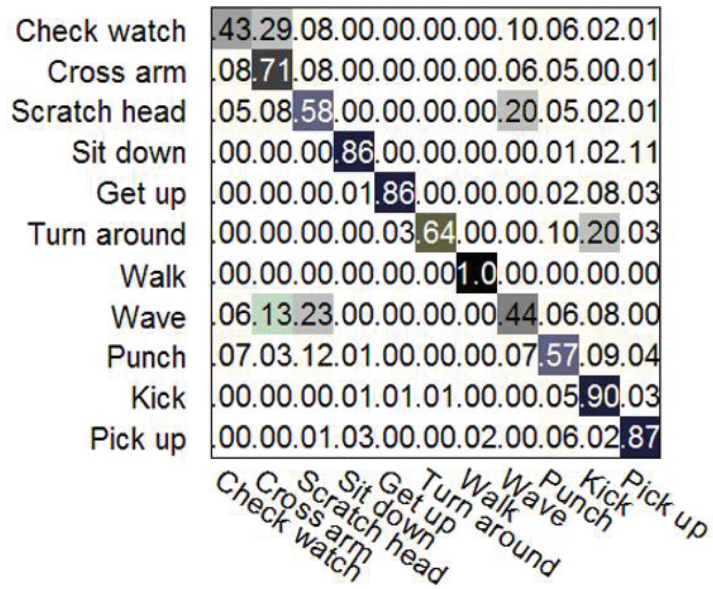


Figure 3.25: Confusion matrix of IXMAS dataset. The confusion matrix of IXMAS dataset when we use generalized 4D-ST motion features.

learning since they assume predefined source and target views, which have different experimental settings. When we use images from multiple known views for training (the first case), the accuracies of our method for camera $0 \sim 4$ are in Table 3.2 (Ours (known view)). The proposed method shows the best performance except the results in [37] and [39]. We used 3D-S volumes, which have 0 or 1 values, to generate features of all the views. However, [37] and [39] used features like HOG from gray images which can not be calculated in 3D-S volumes, which leads to improved performance.

When using our proposed generalized 4D-ST motion features for training (the second case), the accuracies of our method for camera $0 \sim 4$ are in Table 3.2 (Ours (unknown view)). Performance is slightly degraded than the first case 'Ours (known view)' because the test view is completely random and might be entirely different from the training views. [37] and [39] have some prior information about

Table 3.3: Experimental result of SNU dataset. The recognition rates (in %) per camera when we train using IXMAS dataset and test using SNU dataset.

SNU	Avg	Cam 0	Cam 1	Cam 2	Cam 3
Ours	**73.4**	**71.7**	**71.7**	**78.0**	**72.3**
Wu (2012)	45.9	49.4	42.2	51.8	40.3
Weinland (2010)	67.6	69.5	69.0	70.3	61.7

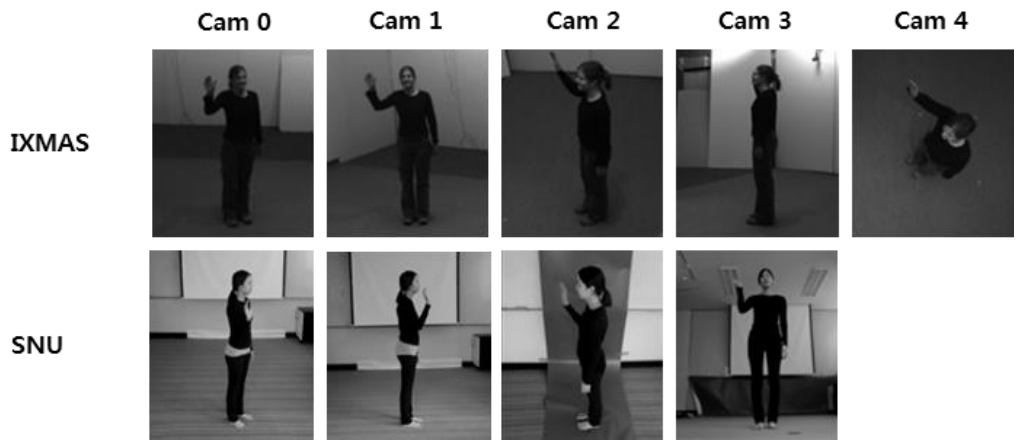


Figure 3.26: The comparision of IXMAS and SNU dataset. The 'wave' action in IXMAS (first row) dataset with 5 different views and the proposed SNU dataset (second row) with 4 different views.

test views (i.e. some test view information is included in the training set), which leads to improved performance but makes the system modeling ad-hoc in nature. On the other hand, we are proposing a more generalized model which does not use any information about test views i.e. they are completely arbitrary. Figure 3.25 shows the confusion matrix for testing 11 actions from IXMAS dataset when we use generalized 4D-ST motion features. The reason why the actions using arms and hands are misclassified is due to the high similarity among these actions.

In real-life scenarios, the information about orientation of people and the camera view in the test inputs might not be available during training. Therefore,

Figure 3.27: Confusion matrix of SNU dataset. The confusion matrix when we train using IXMAS dataset and test using SNU dataset.

in our second experimental setup, training and testing datasets were totally different. To evaluate the view invariant performance with testing images, which are different from the training images, we developed the SNU dataset containing 4 different camera views as shown in Figure 3.2. For evaluating generalization of all models (Ours and [37] and [39]), the training parameters and training dataset (IXMAS) were kept same as previous experimental setup. Table 3.3 shows our recognition rates per camera in SNU test dataset and compares it against the state-of-the-art methods ([37] and [39]), where the proposed method shows the best performance in all camera views. In this experiment, the recognition performance of methods by [37] and [39] is degraded than the IXMAS dataset because the testing view is not included in the training step. However, our algorithm outperforms this method even when the images of test view are not provided in the training step since we use generalized features. In Figure 3.26, the views

of camera $0, 1$ and 2 in SNU dataset are similar to the IXMAS dataset, while the camera 3 view (bottom view) is much different. The performance of [37] and [39] shows unstable recognition performance even with the video sequences on similar viewpoint and is degraded largely for testing on camera 3 view, which (or similar to this view) is not in the training set at all. Contrarily, the performance of the proposed algorithm is consistent regardless of views. Figure 3.27 shows the confusion matrix for testing 11 actions from SNU dataset.

3.5 Concluding remarks

In this chapter, we proposed a view invariant action recognition method useful for practical applications. We developed new 4D-STIPs by extending the widely used 3D-STIPs using 3D-S volume sequences. With the 3D-S volumes and the proposed 4D-STIPs, we can generate features of all the views by projecting them to arbitrary 2D-S image planes. We also proposed non-motion features to encode stationary part of an action, increasing the recognition performance. The proposed method was evaluated with multi-view IXMAS and our own SNU dataset and was verified to outperform the state-of-the-art methods, especially for the test view not provided during the training step. The model is said to be generalized in view point of that it robustly recognize actions from any arbitrary view and even from video captured at different views from training sets.

Chapter 4

Recognition of Action Orientation

4.1 Introduction

The ultimate purpose of computer vision is to interpret what happened in video clip. If we can recognize the orientation of action, it can be helpful in analyzing the video by providing the information about interactions of people. For example, if we recognize that someone shake hands and recognize the orientation of that actor, we can know to whom that actor shake hands with according to the orientation of that actor. In chapter 1, we divided 2D space (2D-S, [x,y]) approach for view invariant action recognition in three directions: 1) designing of view invariant features [22, 23, 25, 24, 26], 2) designing a classifier based on transfer learning [27, 28, 29, 30, 31, 32, 33], and 3) learning of features from multiple views [34, 35, 36, 37, 38, 39]. As for the first issue, view invariant features can not recognize the orientation of action since view invariant features lack of any information about each view for making features which are commonly used for

all the views. Transfer learning related methods do not consider the orientation information when they make visual dictionary, and, therefore they can not recognize the orientation of action. The methods accumulating features from finite number of views can not recognize the orientation of action since they consider the same action from different views as the same class in training step. [39] infer the view label of action in the training step using latent variable in latent kernelized structural support vector machine (SVM), however, there is no information about the exact orientation of learned view.

In this chapter, we propose a method to recognize the orientation of action, which is not tried in view invariant action recognition field. We can recognize the orientation of actor in the test video since our training sets, which are projections of 4D space-time (4D-ST [x-y-z-t]) motion features to various image planes, contain the orientation information. Differently from the action recognition, the features motion history images (MHIs) [69] and non-motion history images (NMHIs) are reduced the dimension using orientation label and trained by support vector data description method (SVDD) [71]. In the recognition step, MHIs and NMHIs are reduced the dimension by the principal axis obtained using orientation label, and then recognized orientation of action with support vector domain density description (SVDDD) [72]. The performance of the proposed algorithm is evaluated on the SNU dataset taken from 5 different orientations to show good recognition rates.

4.2 Proposed method

Figure 4.1 and Figure 4.2 illustrate the overall framework of our proposed approach. The training procedure is shown in Figure 4.1. The process to make MHIs and NMHIs is the same as the methods in the view invariant action recognition.

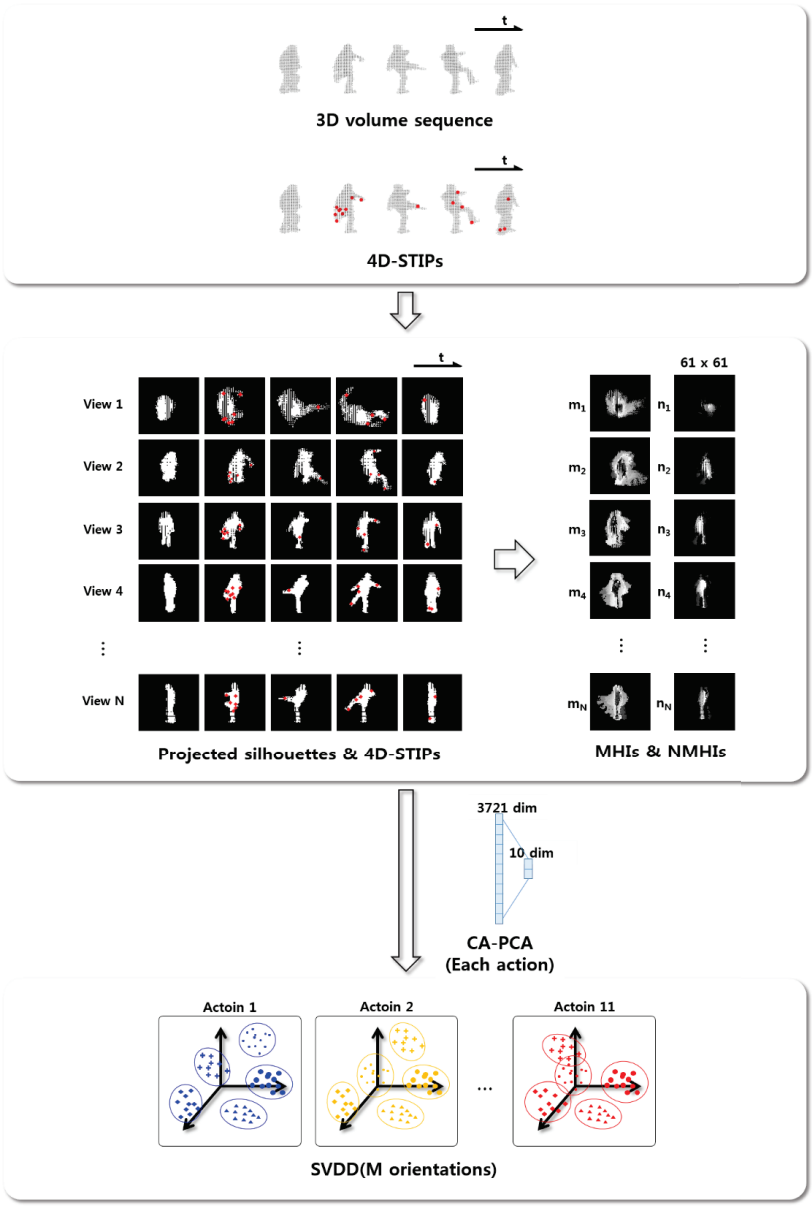


Figure 4.1: Overview for training step. In training step, reconstructed 3D-S volumes and 4D-STIPs are projected to N number of candidate views. The projected silhouettes and 4D-STIPs are then represented by MHIs and NMHIs. For each action class, we reduce the dimension of MHIs and NMHIs using CA-PCA with orientation label and then train the resulting features with SVDD. The features from MHIs and NMHIs are trained separately by SVDD.

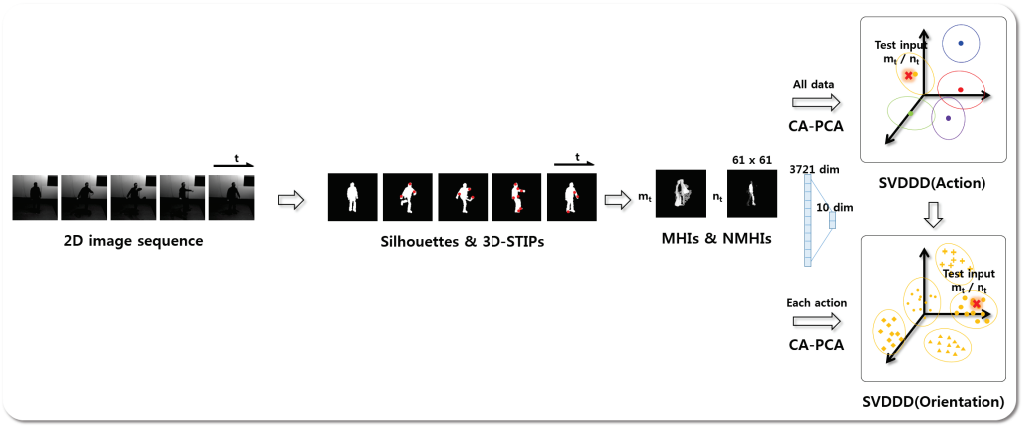


Figure 4.2: Overview for testing step. In testing step, silhouettes and 3D-STIPs are extracted from the test image sequence and MHIs and NMHIs are constructed. After recognizing action class, MHIs and NMHIs are reduced the dimension and tested by SVDDD using the SVDD results of recognized action class. Final score for each class is calculated by summing the SVDDD scores of the features from MHIs and NMHIs.

In the recognition of action orientation step, we collect train data for each action class and then apply class-augmented principal component analysis (CA-PCA) [70] using orientation label. Similarly the final features with reduced dimension for each action class are trained by SVDD.

In the test stage (Figure 4.2), similarly silhouettes and 3D-STIPs are extracted using an image sequence from the test camera view. The process for recognizing action orientation is performed after recognizing action class. After recognizing action, the test input is reduced using CA-PCA principal axis which is calculated previously for recognition of action orientation. Then the score for recognition of action orientation is calculated for each orientation class by summing the SVDDD scores of two features.

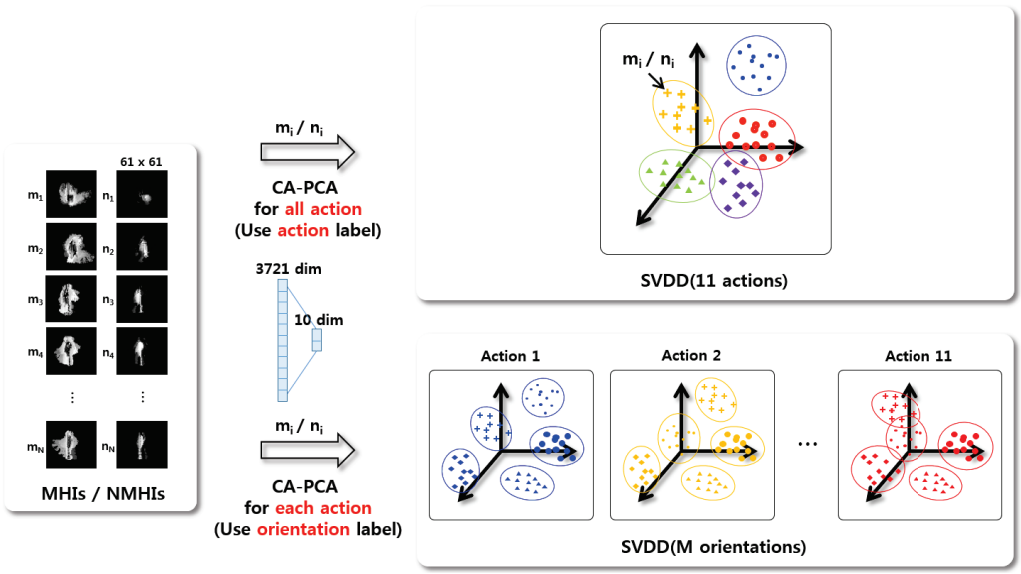


Figure 4.3: Training for recognition of action orientation. The dimension of MHIs (NMHIs) for each action class is reduced with CA-PCA using orientation class labels. After reducing the feature (derived from MHIs and NMHIs) dimension, they are separately trained using SVDD.

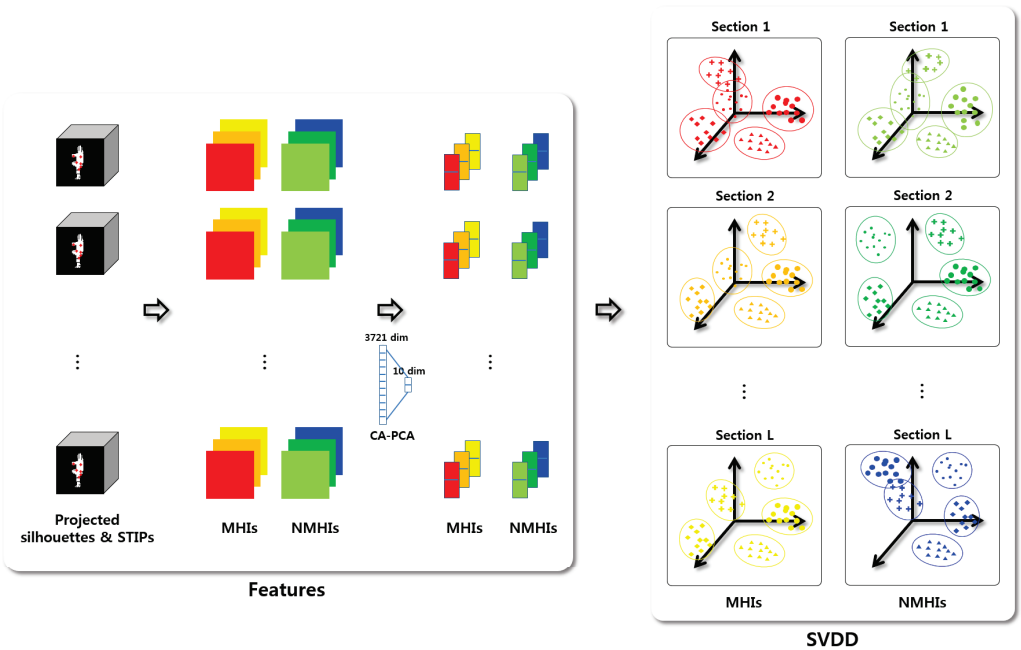


Figure 4.4: Sections of MHIs and NMHIs. We divide the total sequence into L sections and calculate MHIs and NMHIs for each section. The same sections of vectorized MHIs (NMHIs) for each action class are collected and the dimension is reduced with CA-PCA using orientation labels. After reducing the feature (derived from MHIs and NMHIs) dimension, they are separately trained using SVDD. Finally, we obtain $(2L \times N_C)$ number of CA-PCA principal axis and $(2L \times N_C)$ number of SVDD results, where N_C is the number of action class.

4.2.1 Training and Testing

The proposed method can recognize the orientation of action since the features, which are projections of 3D-S volumes and 4D-STIPs to various image planes, contain the orientation information. Figure 4.3 shows the process of training step for recognition of action orientation. Similar with the training step for view invariant action recognition, we use the vectorized L number of MHIs and L number of NMHIs for training for orientation of action recognition (Figure 4.4). When we classify actions, we collect features from all action classes and reduce the dimension using CA-PCA which uses action class information. However, when we classify the orientation of action, we collect the same sections of vectorized MHIs (NMHIs) for each action class and dimension is reduced with CA-PCA using orientation labels. The reason why we do not collect data for all action classes is that we want to find the more appropriate principal axis for recognition of action orientation. After reducing the feature (derived from MHIs and NMHIs) dimension, they are separately trained using SVDD. Finally, we obtain $(2L \times N_C)$ number of CA-PCA principal axis and $(2L \times N_C)$ number of SVDD results, where N_C is the number of action class.

Figure 4.5 shows the process of testing step for recognition of action orientation. The recognition of action orientation is performed after classifying action class. The each section of vectorized L number of MHIs and L number of NMHIs, which are calculated using silhouettes and 3D-STIPs from 2D-S image sequence in the previous action recognition step, are reduced with corresponding CA-PCA principal axis and recognized the actions with SVDDD classifier. We use the sum of the $2L$ number of SVDDD scores (MHIs and NMHIs) as a final decision score and assign the samples of input test video to the orientation of the highest decision score.

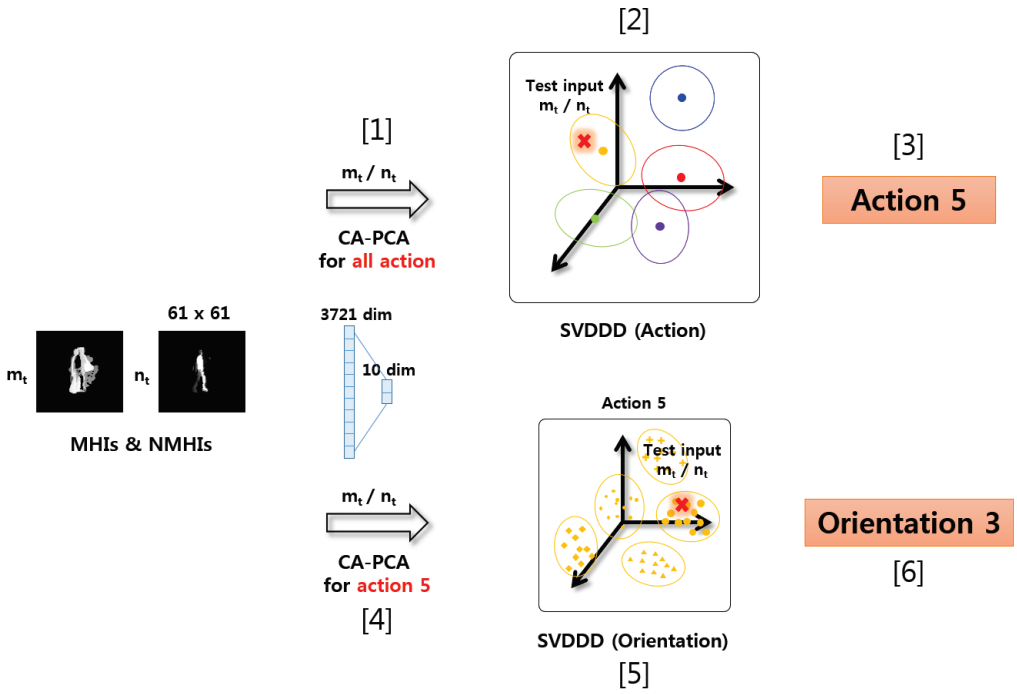


Figure 4.5: Testing for recognition of action orientation. The recognition of action orientation is performed after classifying action class. While the test input is reduced using CA-PCA with action label when we recognize action, the test input is reduced using CA-PCA with orientation label when we recognize the orientation of action.

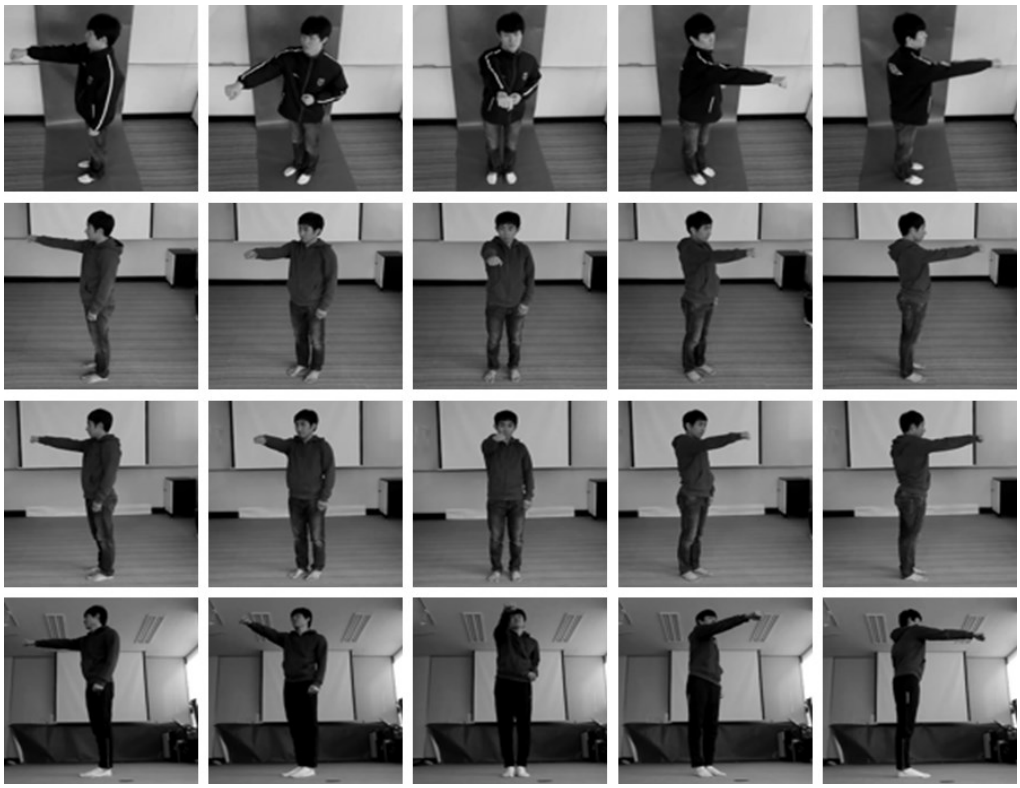


Figure 4.6: SNU dataset for recognition of action orientation. SNU dataset is taken from 5 different orientations for recognition of action orientation.

4.3 Experimental results

To recognize the orientation of action, we developed SNU dataset which was taken from 5 different orientations as in Figure 4.6. The SNU dataset contains 4 different views related to φ value and each view is consist of 5 orientations related to θ value of C in Figure 3.12. For recognition of action orientation experiment, we aligned 3D-S volumes as in Figure 4.7 and then projected 4D-ST motion features into 13 orientations with 15° interval to cover various camera views and orientation of people. Then the number of different image planes used for training becomes $13 \times 8 = 104$ (i.e. N is 104 in Figure 4.1), where 13 different orientations

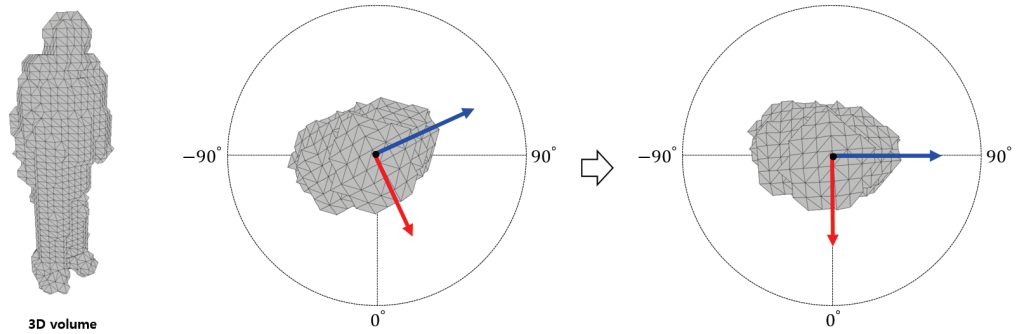


Figure 4.7: Alignment of 3D-S volumes. 3D-S volumes are aligned to look at the front for the recognition of action orientation.

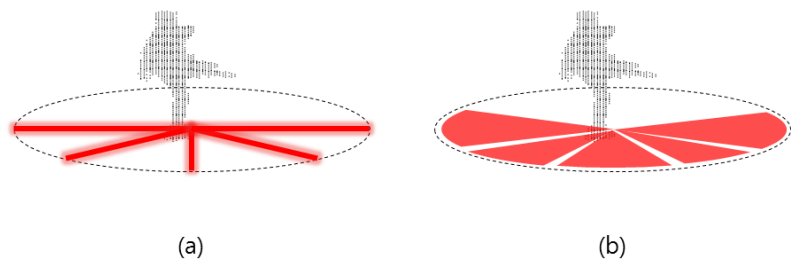


Figure 4.8: The accuracy of recognition of action orientation in two cases. This Figure shows (a) when we recognize correctly (0°) and (b) when we allow the adjacent orientation 45° as correct one.

Table 4.1: The recognition rates of action orientation in SNU dataset. The recognition rates of action orientation (in %) per camera when we recognize correctly (0°) and when we allow the adjacent orientation as correct one (15° ∼ 45°).

	Avg	Cam 0	Cam 1	Cam 2	Cam 3
0°	44.3	33.2	47.6	44.3	51.9
15°	63.3	67.0	57.5	66.3	62.4
30°	75.6	83.2	63.5	83.2	72.4
45°	92.5	91.5	92.3	93.0	93.1

Table 4.2: The recognition rates of action orientation per action and camera. The recognition rates of action orientation (in %) per action and camera (a) when we recognize correctly and (b) when we allow the adjacent orientation 15° as correct one.

	Avg	Cam 0	Cam 1	Cam 2	Cam 3
Check watch	29.9	17.6	20.8	46.4	34.6
Cross arm	53.6	50.0	40.9	55.3	68.4
Scratch head	68.8	30.0	81.8	70.6	92.9
Sit down	31.4	21.7	46.9	16.2	40.9
Get up	28.3	22.2	42.2	15.6	33.3
Turn around	45.2	37.8	44.4	48.4	50.0
Walk	25.9	22.2	25.0	29.3	27.3
Wave	47.8	32.4	55.6	50.0	53.3
Punch	56.9	37.5	55.0	72.2	63.0
Kick	52.5	48.7	59.1	48.8	53.3
Pick up	46.2	44.4	52.3	34.9	53.3
Avg	44.2	33.2	47.6	44.3	51.9

(a)

	Avg	Cam 0	Cam 1	Cam 2	Cam 3
Check watch	68.9	76.5	70.8	82.1	46.2
Cross arm	70.1	100.0	40.9	71.1	68.4
Scratch head	81.8	70.0	81.8	82.4	92.9
Sit down	53.7	47.8	53.1	59.5	54.5
Get up	44.6	44.4	48.9	42.2	42.9
Turn around	48.7	48.6	44.4	51.6	50.0
Walk	53.3	66.7	55.0	41.5	50.0
Wave	69.9	67.6	55.6	76.5	80.0
Punch	67.8	62.5	55.0	83.3	70.4
Kick	76.3	82.1	65.9	83.7	73.3
Pick up	61.5	71.1	61.4	55.8	57.8
Avg	63.3	67.0	57.5	66.3	62.4

(b)

Table 4.3: The recognition rates of action orientation per action and camera. The recognition rates of action orientation (in %) per action and camera when we allow the adjacent orientation (a) 30° (b) 45° as correct one.

	Avg	Cam 0	Cam 1	Cam 2	Cam 3
Check watch	88.9	100.0	70.8	100.0	84.6
Cross arm	78.3	100.0	50.0	92.1	71.1
Scratch head	95.9	100.0	90.9	100.0	92.9
Sit down	65.2	65.2	62.5	78.4	54.5
Get up	57.6	57.8	51.1	64.4	57.1
Turn around	55.0	67.6	44.4	58.1	50.0
Walk	68.5	91.1	62.5	63.4	56.8
Wave	87.4	82.4	70.4	97.1	100.0
Punch	74.6	75.0	55.0	94.4	74.1
Kick	87.8	89.7	72.7	95.3	93.3
Pick up	72.3	86.7	68.2	72.1	62.2
Avg	75.6	83.2	63.5	83.2	72.4

(a)

	Avg	Cam 0	Cam 1	Cam 2	Cam 3
Check watch	98.1	100.0	100.0	100.0	92.3
Cross arm	100.0	100.0	100.0	100.0	100.0
Scratch head	100.0	100.0	100.0	100.0	100.0
Sit down	83.3	78.3	81.3	91.9	81.8
Get up	82.6	80.0	84.4	77.8	88.1
Turn around	81.8	81.1	88.9	77.4	80.0
Walk	85.0	95.6	82.5	78.0	84.1
Wave	91.5	88.2	77.8	100.0	100.0
Punch	95.8	83.3	100.0	100.0	100.0
Kick	100.0	100.0	100.0	100.0	100.0
Pick up	98.9	100.0	100.0	97.7	97.8
Avg	92.5	91.5	92.3	93.0	93.1

(b)

are related to θ value and 8 different views are related to φ value of C in Figure 3.12. We recognized the orientation of action only when the action classification result was correct. Table 4.1 shows recognition rates of action orientation (in %) per camera when we recognize correctly (0°) and when we allow the adjacent orientation as correct one (15° $\sim$ 45°) (Figure 4.8). We allowed an error 45° as a maximum value since the SNU dataset was taken from 5 different orientations with 45° interval. In table 4.1, we can see that the recognition rates of action orientation show good performance when we allow the adjacent orientation 45° as correct one. Table 4.2, 4.3 show the recognition rates of action orientation (in %) per action and camera when we recognize correctly and allow the adjacent orientation (15° $\sim$ 45°) as correct one.

4.4 Concluding remarks

In this chapter, we proposed a method to recognize the orientation of action and the experimental results showed good recognition rates. The proposed method for recognition of action orientation can be useful for recognizing the complex human activities such as human-human interactions.

Chapter 5

Conclusions

In this thesis, we proposed a framework for the view invariant recognition of action and their orientation using generalized 4D [x,y,z,t] motion features with 3D space (3D-S, [x,y,z]) volumes to understand actions occurring in the three-dimensional [x,y,z] world. The contributions of this thesis are summarized as follows.

First, we proposed 4D space-time interest points (4D-STIPs, [x,y,z,t]) which extended the 3D space-time interest points (3D-STIPs, [x,y,t]) using 3D-S volumes reconstructed from images of several number of different views. Since the proposed features were constructed using volumetric information, the features for arbitrary 2D space (2D-S, [x,y]) viewpoint could be generated by projecting 4D-STIPs on corresponding image planes and used for training step. We also proposed a variant of 3D-STIPs by taking into account the simultaneous gradient variation in all 3 dimensions [x,y,t] to focus on the motion of important spatial corner points such as head, hands and feet. The conventional 3D-STIPs may be extracted from the body parts motions which are meaningless for describing the details of actions in many cases. This is because the existing method might extract interest points even when any one of the spatial or temporal derivatives

becomes large. To extract temporal interest points only from spatially distinctive parts such as head, hands, and feet, we modified the existing 3D-STIPs method by hierarchically extracting the STIPs in two steps. We first found the spatial interest points (SIPs) in each image and then decided SIPs having significant variation along time axis as STIPs. The same hierarchical process were applied in calculating 4D-STIPs. By doing this, we could focus on the discriminative movements of body parts for action recognition. We calculated the variant of 3D-STIPs using image sequences from 5 different views and 4D-STIPs using 3D-S volumes in IXMAS dataset. The proposed interest points were verified to be able to represent the properties of each action compactly.

Second, we proposed a method to recognize human action independently of viewpoints by using the proposed 4D space-time (4D-ST, [x,y,z,t]) motion features which can generalize the information from a finite number of views in training phase so as to show a satisfactory performance in arbitrary testing views. With the 3D-S volumes and the proposed 4D-STIPs, we could generate features of all the views by projecting them to arbitrary 2D-S image planes. In addition, we proposed non-motion features to encode stationary part of an action, which increased the recognition performance. The model is said to be generalized in that it robustly recognize actions from any arbitrary view and even from video captured at entirely different views from training sets. In experiments, we trained the models using IXMAS dataset constructed from 5 views and tested them with a new SNU dataset made for evaluating the generalization performance for arbitrary view videos and the proposed method showed the best performance in all camera views.

Third, we also proposed a method to recognize the orientation of action. We could recognize the orientation of actor in a reasonable performance since our training sets, which are projections of 3D-S volumes and 4D-STIPs to vari-

ous image planes, contain the orientation information. In the experiment for the recognition of action orientation, we subdivided the projection interval in training step and represented recognition rates depending on the required accuracies. The experimental results showed reasonable recognition performance. The recognition of action orientation can be very useful for recognizing the more complex human activities such as human-human interactions.

Bibliography

[1] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local svm approach," in *International Conference on Pattern Recognition (ICPR)*, 2004.

[2] D. Weinland, R. Ronfard, and E. Boyer, "Free viewpoint action recognition using motion history volumes," *Computer Vision and Image Understanding (CVIU)*, vol. 104, no. 2-3, 2006.

[3] I. Laptev and T. Lindeberg, "Velocity adaptation of space-time interest points," in *International Conference on Pattern Recognition (ICPR)*, 2004.

[4] I. Laptev, *Local Spatio-Temporal Image Features for Motion Interpretation*, Ph.D. thesis, Computational Vision and Active Perception Laboratory (CVAP), NADA, KTH, Stockholm, Sweden, April 2004.

[5] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *International Conference on Computer Vision (ICCV)*, 2005.

[6] I. Laptev and T. Lindeberg, "Local descriptors for spatio-temporal recognition," in *European Conference on Computer Vision Workshop (ECCV Workshop)*, 2006.

[7] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *Pattern Analysis and Machine Intelligence (PAMI)*, 2007.

[8] I. Laptev, M. marszalek, C. schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Computer Vision and Pattern Recognition (CVPR)*, 2008.

[9] J. Yuan, Z. Liu, and Y. Wu, "Discriminative subvolume search for efficient action detection," in *Computer Vision and Pattern Recognition (CVPR)*, 2009.

[10] A. Kovashka and K. Grauman, "Learning a hierarchy of discriminative space-time neighborhood features for human action recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 2010.

[11] H. Wang, A. Klaser, C. Schmid, and C. Liu, "Action recognition by dense trajectories," in *Computer Vision and Pattern Recognition (CVPR)*, 2011.

[12] A. Gilbert, J. Illingworth, and R. Bowden, "Action recognition using mined hierarchical compound features," *Pattern Analysis and Machine Intelligence (PAMI)*, vol. 33, no. 5, 2011.

[13] S. Amin, M. Andriluka, M. Rohrbach, and B. Schiele, "Multi-view pictorial structures for 3d human pose estimation," in *British Machine Vision Conference (BMVC)*, 2013.

[14] M. Burenius, J. Sullivan, and S. carlsson, "3d pictorial structures for multiple view articulated pose estimation," in *Computer Vision and Pattern Recognition (CVPR)*, 2013.

[15] F. Zhu, L. Shao, and M. Lin, "Multi-view action recognition using local similarity random forests and sensor fusion," *Pattern Recognition Letters (PRL)*, vol. 34, pp. 20–24, 2013.

[16] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3d points," in *Computer Vision and Pattern Recognition (CVPR)*, 2010.

[17] 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 (ECCV)*, 2012.

[18] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras," in *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[19] O. Oreifej and Z. Liu, "Hon4d: histogram of oriented 4d normals for activity recognition from depth sequences," in *Computer Vision and Pattern Recognition (CVPR)*, 2013.

[20] L. Xia and J. K. Aggarwal, "Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera," in *Computer Vision and Pattern Recognition (CVPR)*, 2013.

[21] J. Luo, W. Wang, and H. Qi, "Group sparsity and geometry constrained dictionary learning for action recognition from depth maps," in *International Conference on Computer Vision (ICCV)*, 2013.

[22] C. Rao, A. Yilmaz, and M. Shah, "View-invariant representation and recognition of actions," *International Journal of Computer Vision (IJCV)*, vol. 50, no. 2, pp. 203–226, 2002.

[23] V. Parameswaran and R. Chellappa, "View invariants for human action recognition," *International Journal of Computer Vision (IJCV)*, vol. 66, no. 1, pp. 83–101, 2006.

[24] Y. Shen and H. Foroosh, "View-invariant action recognition using fundamental ratios," in *Computer Vision and Pattern Recognition (CVPR)*, 2008.

[25] Junejo, E. Dexter, and I. Laptev, "Cross-view action recognition from temporal self-similarities," in *European Conference on Computer Vision (ECCV)*, 2008.

[26] I. Junejo, E. Dexter, I. Laptev, and P. Perez, "View-independent action recognition from temporal self-similarities," *Pattern Analysis and Machine Intelligence (PAMI)*, vol. 33, no. 1, pp. 172–185, 2011.

[27] A. Farhadi and M. K. Tabrizi, "Learning to recognize activities from the wrong view point," in *European Conference on Computer Vision (ECCV)*, 2008.

[28] J. Liu, M. Shah, B. Kuipers, and S. Savarese, "Cross-view action recognition via view knowledge transfer," in *Computer Vision and Pattern Recognition (CVPR)*, 2011.

[29] J. Zheng, Z. Jiang, and P. J. Phillips, "Cross-view action recognition via a transferable dictionary pair," in *British Machine Vision Conference (BMVC)*, 2012.

[30] B. Li, O. I. Camps, and M. Sznaier, "Cross-view activity recognition using hankelets," in *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[31] R. Li and T. Zickler, "Discriminative virtual views for cross-view action recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[32] Z. Zhang, C. Wang, B. Xiao, W. Zhou, S. Liu, and C. Shi, "Cross-view action recognition via a continuous virtual path," in *Computer Vision and Pattern Recognition (CVPR)*, 2013.

[33] J. Zheng and Z. Jiang, "Learning view-invariant sparse representations for cross-view action recognition," in *International Conference on Computer Vision (ICCV)*, 2013.

[34] J. Liu and M. Shah, "Learning human actions via information maximization," in *Computer Vision and Pattern Recognition (CVPR)*, 2008.

[35] P. Yan, S. Khan, and M. Shah, "Leaning 4d action feature models for arbitrary view action recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 2008.

[36] K. K. Reddy, J. Liu, and M. Shah, "Incremental action recognition using feature-tree," in *International Conference on Computer Vision (ICCV)*, 2009.

[37] D. Weinland, M. Ozuysal, and P. Fua, "Making action recognition robust to occlusions and viewpoint changes," in *European Conference on Computer Vision (ECCV)*, 2010.

[38] M. B. Kaaniche and F. Bremond, "Gesture recognition by learning local motion signature," in *Computer Vision and Pattern Recognition (CVPR)*, 2011.

[39] X. Wu and Y. Jia, "View-invariant action recognition using latent kernelized structural svm," in *European Conference on Computer Vision (ECCV)*, 2012.

[40] S. M. Khan, P. Yan, and M. Shah, "A homographic framework for the fusion of multi-view silhouettes," in *International Conference on Computer Vision (ICCV)*, 2007.

[41] D. Weinland, *Action representation and recognition*, Ph.D. thesis, LJK-INPG Grenoble, Rhone-Alpes, 655 avenue de l'Europe, 38330 Montbonnot, France, October 2008.

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

[43] J. Rittscher and A. Blake, "Classification of human body motion," in *International Conference on Computer Vision (ICCV)*, 1999.

[44] A. A. Efros, A. Berg, G. Mori, and J. Malik, "Recognizing action at a distance," in *International Conference on Computer Vision (ICCV)*, 2003.

[45] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *International Workshop on Performance Evaluation of Tracking and Surveillance*, 2005.

[46] A. Yilmaz and M. Shah, "Recognizing human actions in videos acquired by uncalibrated moving cameras," in *International Conference on Computer Vision (ICCV)*, 2005.

[47] I. Laptev, "On space-time interest points," *International Journal of Computer Vision (IJCV)*, vol. 64, no. 2/3, pp. 107–123, 2005.

[48] L. Wang and D. Suter, "Recognizing human activities from silhouettes: motion subspace and factorial discriminative graphical model," in *Computer Vision and Pattern Recognition (CVPR)*, 2007.

[49] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using hidden markov model," in *Computer Vision and Pattern Recognition (CVPR)*, 1992.

[50] M. Brand, N. Oliver, and A. Pentland, "Coupled hidden markov models for complex action recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 1997.

[51] Z. Ghahramani, "Learning dynamic bayesian networks," *Lecture Notes in Computer Science (LNCS)*, vol. 1387, pp. 168–197, 1998.

[52] N. Jojic, N. Petrovic, B. Frey, and T. Huang, "Transformed hidden markov models: Estimating mixture models of images and inferring spatial transformations in video sequences," in *Computer Vision and Pattern Recognition (CVPR)*, 2000.

[53] S. Park and J. K. Aggarwal, "Recognition of two-person interactions using a hierarchical bayesian network," in *International Workshop on Video Surveillance*, 2003.

[54] P. Peursum, G. West, and S. Venkatesh, "Combining image regions and human activity for indirect object recognition in indoor wide-angle views," in *International Conference on Computer Vision (ICCV)*, 2005.

[55] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas, "Conditional models for contextual human motion recognition," in *International Conference on Computer Vision (ICCV)*, 2005.

[56] S. B. Wang, A. Quattoni, L. P. Morency, D. Demirdjian, and T. Darrell, "Hidden conditional random fields for gesture recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 2006.

[57] L. P. Morency, A. Quattoni, and T. Darrell, "Latent-dynamic discriminative models for continuous gesture recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 2007.

[58] W. A. Forstner and E. Gulch, "A fast operator for detection and precise location of distinct points, corners and centers of circular features," in *Intercommission Workshop of the Int. Soc. for Photogrammetry and Remote Sensing*, 1987.

[59] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, 1988.

[60] T. Lindeberg, "Feature detection with automatic scale selection," *International Journal of Computer Vision (IJCV)*, vol. 30(2), pp. 77–116, 1998.

[61] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of Computer Vision (IJCV)*, vol. 37(2), pp. 151–172, 2000.

[62] R. Szeliski, "Notes on the harris detector," University Lecture, 2005.

[63] D. Weinland, E. Boyer, and R. Ronfard, "Action recognition from arbitrary views using 3d examplars," in *International Conference on Computer Vision (ICCV)*, 2007.

[64] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *Computer Vision and Pattern Recognition (CVPR)*, 2006.

[65] C. Wholer, *3D computer vision: efficient methods and applications*, Springer, New york, 2009.

[66] A. Laurentini, "The visual hull concept for silhouette-based image understanding," *Pattern Analysis and Machine Intelligence (PAMI)*, vol. 16, no. 2, pp. 150–162, 1994.

[67] "Camera calibration toolbox for matlab," `http://www.vision.caltech.edu/bouguetj/calib_doc/`.

[68] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, Cambridge University Press, Cambridge,UK, 2000.

[69] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *Pattern Analysis and Machine Intelligence (PAMI)*, vol. 23, no. 3, pp. 257–267, 2001.

[70] M. S. Park, J. H. Na, and J. Y. Choi, "Feature extraction using class-augmented principal component analysis (ca-pca)," *Lecture Notes in Computer Science (LNCS)*, vol. 4132, pp. 606–615, 2006.

[71] D. M. J Tax and R. P. W. Duin, "Support vector data description," *Machine Learning (ML)*, vol. 54, pp. 45–66, 2004.

[72] W. S. Kang and J. Y. Choi, "Domain density description for multiclass pattern classification with reduced computational load," *Pattern Recognition (PR)*, vol. 41, pp. 1997–2009, 2007.

[73] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience (JCN)*, vol. 3, pp. 71–86, 1991.

[74] W. S. Kang, *Domain density descriptions for pattern classification*, Ph.D. thesis, Seoul National University, Department of Electrical and Computer Engineering, ASRI, Seoul National University, 133-413 599 Gwanak-ro Gwanak-gu, Seoul 151-742, Korea, February 2009.

[75] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery 2*, pp. 121–167, 1998.

[76] V. N. Vapnik, *Statistical learning theory*, Wiley Interscience, 1998.

[77] J. A. K. Suykens, T. V. Gestel, J. D. Brabanter, B. D. Moor, and J. Vandewalle, *Least squares support vector machines*, World Scientific, 2002.

국문 초록

본 논문은 일반화된 4차원 [x,y,z,t] 동작 특징을 이용하여 시선각에 무관한 행동 및 행동 방향 인식 문제를 해결하는 것을 목적으로 수행되었다. 기존의 행동 인식 연구는 주로 카메라의 위치는 고정되어있고 사람들은 카메라를 바라보고 서있는 상황을 가정하였다. 그러나 실제 비디오나 감시카메라에 등장하는 사람들은 카메라를 의식하지 않고 자연스럽게 행동하기 때문에 제한된 조건, 환경에서 행동을 인식하는 것과 달리, 카메라의 위치와 사람의 방향에 따라서 다양한 시선각에서 영상이 촬영될 수 있다. 따라서 실제 어플리케이션에 적용하기 위해서는 무작위의 시선각에서 영상이 들어왔을 때 행동 인식을 하는 것이 필수적이며, 어떤 방향으로 행동하는 지 알 수 있다면 누구와 상호작용을 하는 지 아는데 도움을 줄 수 있다.

본 논문에서는 몇 개의 다른 시선각에서 찍힌 영상을 이용하여 3차원 [x,y,z] 입체를 복원하고, 연속된 3차원 입체에서 4차원 시공간 특징점을 구하는 방법을 제안하여 시선각에 무관한 행동 및 행동 방향 인식을 수행하였다. 3차원 입체 및 연속된 3차원 입체에서 구한 4차원 시공간 특징점은 모든 시선각에서의 정보를 갖고 있으므로, 원하는 시선각으로 사영을 하여 각 시선각에서의 특징을 얻을 수 있다. 사영된 실루엣과 4차원 시공간 특징점의 위치를 바탕으로 각각 움직이는 부분과 움직이지 않는 부분에 대한 정보를 포함하는 motion history images (MHIs)와 non motion history images (NMHIs)를 만들어 행동 인식을 위한 특징으로 사용을 하였다. MHIs만으로는 행동 시 움직이는 부분이 비슷한 패턴을 보일 때 좋은 성능을 보장할 수 없고 따라서 행동 시 움직이지 않는 부분에 대한 정보를 줄 수 있는 NMHIs를 제안하였다. 행동 인식을 위한 학습 단계에서 MHIs와 NMHIs는 클래스를 고려한 차원 축소 알고리즘인 class-augmented principal component analysis (CA-PCA)를 통해서 차원이 축소되며, 이 때 행동 라벨을 이용하여 차원을 축소하므로 각 행동이 잘 분리가 되도록하는 principal axis를 찾을 수 있다. 차원이 축소된 MHIs와 NMHIs는 support vector data description (SVDD) 방법으로 학습

되고, support vector domain density description (SVDDD)를 이용하여 인식된다. 행동 방향을 학습할때에는 각 행동에 대해 방향 라벨을 사용하여 principal axis를 구하며, 마찬가지로 SVDD로 학습을 하고 SVDDD를 이용하여 인식된다.

제안된 4차원 시공간 특징점은 시선각에 무관한 행동 및 행동 방향 인식에 사용될 수 있으며 실험을 통해 4차원 시공간 특징점이 각 행동의 특징을 압축적으로 잘 보여주고 있음을 보였다. 또한 실제 어플리케이션에서처럼 무작위의 시선각에서 영상이 들어왔을 경우를 가정하기 위하여 학습 데이터셋과 전혀 다른 새로운 인식 데이터셋을 구축하였다. 기존의 여러 시선각에서 촬영 된 IXMAS 행동 인식 데이터셋을 이용하여 학습을 하고, 학습 데이터셋과 다른 시선각에서 촬영한 SNU 데이터셋에서 인식 실험을 하여 제안한 알고리즘을 검증하였다. 실험 결과 제안한 방법은 학습을 위해 촬영한 영상에 포함되지 않는 시선각에서 테스트 영상이 들어왔을 경우에도 좋은 성능을 보이는 것을 확인하였다. 또한 5개의 방향으로 촬영된 SNU 데이터셋을 이용하여 행동 방향 인식 실험을 하였으며, 좋은 방향 인식률을 보이는 것을 확인하였다. 행동 방향 인식을 통해서 영상 내에서 여러 사람이 등장할 때 다른사람들과 어떻게 상호 작용을 하는지 정보를 알 수 있고, 이는 영상을 해석하는데 도움을 줄 수 있을 것으로 생각된다.

주요어: 4차원 시공간 특징점, 시선각에 무관한 행동 인식, 행동 방향 인식, 3차원 복원

학번: 2009-30180