Recognizing Human Actions Using Multiple Features

Jingen Liu
Computer Vision Lab
University of Central Florida
liujg@cs.ucf.edu

Saad Ali
Computer Vision Lab
University of Central Florida
sali@cs.ucf.edu

Mubarak Shah
Computer Vision Lab
University of Central Florida
shah@cs.ucf.edu

Abstract

In this paper, we propose a framework that fuses multiple features for improved action recognition in videos. The fusion of multiple features is important for recognizing actions as often a single feature based representation is not enough to capture the imaging variations (view-point, illumination etc.) and attributes of individuals (size, age, gender etc.). Hence, we use two types of features: i) a quantized vocabulary of local spatio-temporal (ST) volumes (or cuboids), and ii) a quantized vocabulary of spin-images, which aims to capture the shape deformation of the actor by considering actions as 3D objects (x, y, t)*. To optimally combine these features, we treat different features as nodes in a graph, where weighted edges between the nodes represent the strength of the relationship between entities. The graph is then embedded into a* k*-dimensional space subject to the criteria that similar nodes have Euclidian coordinates which are closer to each other. This is achieved by converting this constraint into a minimization problem whose solution is the eigenvectors of the graph Laplacian matrix. This procedure is known as* Fiedler Embedding*. The performance of the proposed framework is tested on publicly available data sets. The results demonstrate that fusion of multiple features helps in achieving improved performance, and allows retrieval of meaningful features and videos from the embedding space.*

1. Introduction

Action recognition in videos is an important area of research in the field of computer vision. The ever growing interest in characterizing human actions is fuelled, in part, by the increasing number of real-world applications such as action/event centric video retrieval, activity monitoring in surveillance scenarios, sports video analysis, smart rooms, human-computer interaction, etc. The classification of human actions has remained a challenging problem due to the sheer amount of variations in the imaging conditions (view-point, illumination etc.) and attributes of the individual (size, age, gender, etc.) performing the action.

In general, approaches for human action recognition can be categorized on the basis of the 'representation'. Some leading representations include learned geometrical models of the human body parts, space-time pattern templates, appearance or region features, shape or form features, interest point based representations, and motion/optical flow patterns. Specifically, [16, 17] utilized the geometrical models of human body parts where the action is recognized by searching for the static postures in the image that match the target action. The popular shape based representations include edges [15] and silhouettes of human body [18]. Recently, [10] used a series of 3D poses to represent the shape of the actor and [4] discovered the chaotic invariant features from the joint tracking for action recognition. The silhouette based representation is also extended to characterize actor's body outline through space and time ([8], [6]). The approaches based on volumetric analysis of video include [6, 12, 19] and [11]. Another important representation that has gained considerable interest recently is the use of space time interest points and their trajectories for action and activity analysis. Work by [22, 13, 23, 9], and [14] belong to this class of methods. The main strength of this representation is that it does not require any segmentation or tracking of the individual performing the action. In optical flow based approaches ([20, 21]), the idea is to directly use the optical flow as a basis for deriving a representation that can be used for recognition.

Most of the approaches described above advocate the use of single feature for action classification. However, we believe that though theoretically sound, the notion that a single feature will capture the range of complexity of human actions is pragmatically weak. Therefore, in this paper we address the specific issue of using multiple features for action recognition and propose a general framework for fusing information from complementary features. Specifically, we propose the use of two types of features: The first feature is a quantized vocabulary of spatio-temporal (ST) volumes (or cuboids) that are centered around 3D interest points in the video. The ST volumes are inherently local in nature,

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

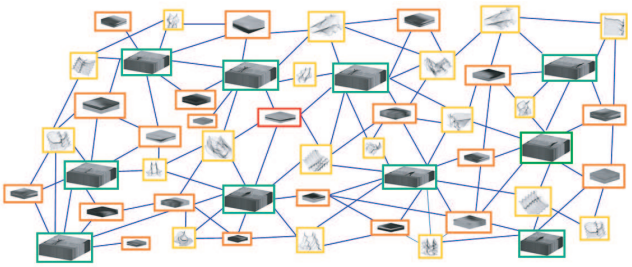

Figure 1. An illustration of the graph containing multiple entities as nodes. This includes ST features (red), Spin-Image features (yellow) and action videos (green). The goal of our algorithm is to embed this graph in a k-dimensional space so that similar nodes have geometric coordinates which are closer to each other.

and therefore capture the local appearance and motion information. The second feature is a quantized vocabulary of spin-images, which aims to capture the spatio-temporal information of the actor by considering actions as 3D objects (x, y, t) [8]. The 3D object itself is carved out by the contours of the individual performing the action. Note that the spin-image based features have not been used for action recognition before and our's is the first method to explore their utility for this task. Next, in order to optimally combine these features, we develop a framework that allows for learning of explicit and implicit relationships between different classes of features in a principled manner. The framework is based on the concept of Fiedler Embedding [7], which is an algebraic method that explicitly optimizes the closeness criteria and is similar to the Laplacian Eigenmap[25]. It embeds different entities into a common Euclidian space, and thus enables the use of simple Euclidian distances for discovering relationships between features.

It would be pertinent to mention that a number of approaches have recently appeared in the literature which propose *feature-level* fusion for improved object recognition and detection in images. The popular choice has been the fusion of patch based and contour based features. For instance, Opelt et. al. [1] used features derived from patches and color segments to learn a strong classifier in a boosting framework. Similarly, [5] used texture and shape context features in a multi-layer boosting framework. A combination of patches and contours were also used in [2]. However, in contrast to these recent advances made in the area of object recognition, there is hardly any method available that can perform feature fusion for improved action classification. In this paper, our aim is to fill this gap by developing a general framework that allows learning of relationships between different classes of action features.

2. Fiedler Embedding

In this section, we present the details of the procedure which is called Fielder Embedding. It was first proposed in [7] for information retrieval from a document corpus. In this paper we adapt this technique for action classification and show that how it can be used for discovering the relationships between different features. We start by describing the mathematical derivation of the embedding procedure and for this purpose we use the nomenclature used by [7].

Let $G = (V, E)$, where V is a set of vertices and E is a set of edges, represents a graph consisting of nodes which represent different classes of features as illustrated in Fig.1 . If two features i and j are related to each other, then we have an edge (i, j) with a non-negative weight, $w_{i,j}$, between them. The more similar the features are to each other, the higher the weight. Our goal is to embed this graph into a low-dimensional Euclidian space, so that vertices with a high weight between them become closer to each other in this space. As a result, the spatial proximity in this space can be used to identify vertices that are similar to each other even if they do not have a direct edge between them (the implicit relationship). In posing this geometric embedding problem as an algebraic minimization problem, we seek points in a k-dimensional space that minimize the weighted sum of the square of the edge lengths. If p_r and p_s are locations of vertices r and s, then the function can be written as

$$\text{Minimize} \sum_{(r,s)\in E} w_{r,s} \mid p_r - p_s \mid^2, \tag{1}$$

where $w_{r,s}$ represents the weight between the nodes r and s. If the graph has n vertices, and the target space has dimensionality k, then the positions of the vertices can be represented by an n x k matrix X. The Laplacian matrix L of this graph can be described as:

$$L(i,j) = \begin{cases} -w_{i,j} & \text{if } e_{ij} \in E \\ \sum_k w(i,k) & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where L is symmetric and positive semi-definite. Note that L is nothing but the negative of the matrix of weights with diagonal values chosen to make the row-sums zero. This will imply that p_r or p_s ($r = 1, 2, ..., n$ and $s = 1, 2, ..., n$) is a k- dimensional vector representing the coordinates of the vertex in the k-dimensional space. It can be shown that the solution of the above minimization problem in terms of matrices L and X is

$$\text{Minimize} \quad Trace(X^T L X), \tag{3}$$

$$\text{(i)} \quad \text{for} \quad i = 1, ..., k \quad X_i^T 1^n = 0, \quad \text{(ii)} \quad X^T X = \Delta. \tag{4}$$

The first constraint makes the median of point sets in the embedding space to be at the origin, while the second constraint avoids the trivial solution of placing all the vertices at the origin. In the above equation 1^n is a vector of n ones,

while Δ is a diagonal matrix of δ_i, which are some positive values. As shown in [7], the solution to the above minimization is $X = \Delta^{1/2}[Q_2, ..., Q_{k+1}]$, where Q is a matrix of normalized eigenvectors of L sorted in a non-decreasing order based on the eigenvalues λ_i of L. This implies that the coordinates of vertex i are simply the $i-$th entries of eigenvectors $2, ..., k+1$ of L. This solution is referred to as the *Fiedler embedding* of the graph. Note that Fiedler embedding is related to the more popular technique Latent Semeiotic Analysis (LSA) which is a linear approach. However, Fiedler Embedding is more powerful in that it allows one to capture a general set of relationships between many types of entities.

3. Action Recognition Framework

In this section, we describe our action recognition framework which utilizes the *Fiedler Embedding*. In our work, we choose two types of feature, *Spatial-Temporal* (ST) features and *Spin-Images*. These feature normally capture the strong variation of the data in spatial and temporal direction that are caused by the motion of the actor. However, ST features contain only the local appearance and motion information, and therefore ignore the shape of actors. To capture the holistic shape information, we consider actions as 3D objects in (x, y, t) and compute their spin-images. Once the features are computed for the given action corpus, we use the *Fiedler Embedding* to discover the relationships among features by projecting them into a common Euclidian space.

The main steps of the framework are: i) Learning of visual vocabularies from ST features and Spin-Images, ii) Construction of Laplacian matrix from the training videos, iii) Embedding and grouping of features. The algorithmic description of these steps is provided in Table 1.

Objective: Given N_v training action videos, embed all entities (ST features, Spin-Image features, Videos) into a common k-dimensional space.

1. Quantize visual information by learning visual vocabularies:
   - Learn vocabulary of ST features of size N_{st}.
   - Learn shape vocabulary based on spin-images of size N_{si}.
2. Construct a $(N_{st} + N_{si}) * N_v$ Feature-Action co-occurrence matrix, $\mathcal{S}$, by counting the frequency of features in each action video.
3. Weigh $\mathcal{S}$ by *tf-idf* to obtain a weighted co-occurrence matrix $\mathcal{S}'$.
4. Construct Laplacian matrix L as :
   - Video-Feature similarity blocks of L are computed directly from the corresponding value from $\mathcal{S}'$.
   - Video-Video and Feature-Feature similarity blocks are computed by using the Inner Product of rows of matrix $\mathcal{S}'$.
5. Perform eigen-decomposition of L such that $\mathcal{L} = \mathbb{V}^T\mathbb{D}\mathbb{V}$ where $\mathbb{V}$ is a set of eigenvectors and $\mathbb{D}$ contains the eigenvalues sorted in ascending order.
6. Construct a k-dimensional space. Select k eigenvectors corresponding to the k smallest eigenvalues excluding the zeros, say $\mathbb{U}=\{u_1, u_2, ..., u_k\}$ which is the basis of the k-dimensional space.
7. Map entities: A video $\mathbf{q}$ is mapped to the k space as: $\mathbb{D}^{1/2}\mathbb{U}^T\mathbf{q}/\|\mathbf{q}\|$.

Table 1. Main steps of the action recognition framework.

3.1. Feature Extraction and Representation

Spatiotemporal Features: We use the detector proposed in [14] for computing the ST features. This detector produces dense feature points and has performed reasonably well on action recognition tasks [13, 14]. Instead of using a 3D filter on spatio-temporal domain, they apply two separate linear filters to the spatial and temporal dimensions respectively. A response function can be represented as, $R = (I * g_\sigma * h_{ev})^2 + (I * g_\sigma * h_{od})^2$, where $g_\sigma(x, y)$ is the spatial Gaussian filter with kernel σ, $h_{ev}(t)$ and $h_{od}(t)$ are a quadrature pair of 1D Gabor filters applied along the time dimension. They are defined as $h_{ev}(t; \tau, \omega) = -cos(2\pi t\omega)e^{-t^2/\tau^2}$ and $h_{od}(t; \tau, \omega) = -sin(2\pi t\omega)e^{-t^2/\tau^2}$, where $\omega = 4/\tau$. They give a strong response for the intensity changes in temporal direction. The interest points are detected at locations where response is locally maximum. The ST volumes around the points are extracted and the gradient-based descriptors are learnt by PCA. All descriptors are quantized into *video-words* using k-means algorithm.

Spin-Image Features: Spin-Images have been successfully used for 3D object recognition [3]. However, it has not been used for action recognition before. For actions, the spin-images can provide a more richer representation of how the local shape of the actor is changing with-respect to different reference points. These reference points may correspond to different limbs of the human body. For extracting the spin-images, we consider an action video as a 3D object. There are two main steps of the procedure: 1) Generating Action Volume, and 2) Spin-Image Extraction.

Generating the Action Volume: We create a 3D action volume by stacking the sequence of contours of the actor. For this purpose, we first apply the background substraction algorithm to get the contour $\mathcal{C}^t$ at frame t. To generate the

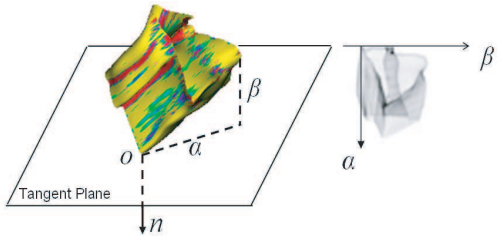

Figure 2. Left: the (α, β) coordinates of a surface point relative to the *orientated point* O. Right: the spin-image centered at O.

3D action volume, we find the correspondences between points of two consecutive contours $\mathcal{C}^t$ and $\mathcal{C}^{t+1}$ using the graph theoretic approach proposed in [8]. Suppose L and R are two point sets corresponding to $\mathcal{C}^t$ and $\mathcal{C}^{t+1}$ respectively, we create a bipartite graph with $|L| + |R|$ vertices. The weights of the edges connecting L and R are estimated from three items:proximity, orientation similarity and shape similarity. Assume c_i and c_j are two vertices from L and R respectively, the proximity d_{ij} between them is the $L2$ norm of their 3D coordinates (x, y, t). The orientation similarity α_{ij} is the angle between the spatial norms of the vertices, and the shape similarity ξ_{ij} is estimated from the neighbors. Then the weights are computed as,

$$w_{ij} = exp(-\frac{d_{ij}^2}{\sigma_d^2})exp(-\frac{\alpha_{ij}^2}{\sigma_\alpha^2})exp(-\frac{\xi_{ij}^2}{\sigma_\xi^2}). \quad (5)$$

Extracting Spin-Images: Johnson et al. [3] introduced the spin-images for recognizing objects in complex 3D scenes. A spin-image (SI) is generated by projecting the mesh vertices onto a tangent plane with respect to a reference surface point, called *orientated point*. The spin-image is an object centered feature, hence it is scale, rotation and pose invariant.

Fig. 2 illustrates the process of projecting a surface point onto a tangent plane with respect to the *orientated point* O. All the surface points are indexed by the radius (α) from o and the depth (β) from the tangent plane of O. The projection function is expressed as,

$$\alpha = \sqrt{\|x - o\|^2 - (n \cdot (x - o))^2}, \quad \beta = n \cdot (x - o),$$

where x and o are the 3D coordinates of the surface point and the *orientated point* O. Hence, all the 3D surface points are projected onto 2D plane. To produce a spin-image, we need to quantize (α,β) and build a 2D histogram, which is called the spin-image. There are several important parameters controlling the generation of the spin-image. The support length, which defines the size of the spin-image, determines the locality of the spin-image. With a larger support length, spin-image can capture the entire object shape and a smaller support length provides local shape information. Another important parameter is bin size, which determines the discrimination of the spin-image. Larger bin size will cause all points fall into the same bin, while small size will separate the neighboring points. In this paper, the bin size is set as the average length of the mesh resolution. Besides, uniform sampling is necessary for matching two shapes. Fig. 3 shows the action volumes and their selected corresponding spin-images. Instead of attempting pairwise matching of spin-images to match two actions, we use the bag of spin-images strategy. For this purpose, we first apply PCA to compress the dimensionality of the spin-image, and then use K-means to quantize them. We call the group of spin-images as a *video-word*. Finally, the action is represented by the bag of *video-words* model.

3.2. Construction of the Laplacian Matrix

The input for *Fiedler Embedding* is the Laplacian Matrix $\mathcal{L}$, which is a symmetric matrix constructed according to equation 2. In our case, we have three types of entities which are: ST features (ST), Spin-Image features (SI) and videos of actions. Therefore, $\mathcal{L}$ has the following block structure:

$$\begin{pmatrix} D_1 & E^T & F^T \\ E & D_1 & H^T \\ F & H & D_3 \end{pmatrix}, \quad (6)$$

where D_1,D_2,D_3,E, F, H, respectively, denote the block matrix of Video-Video, ST-ST, SI-SI, Video-ST, Video-SI and ST-SI. In principle, the relationship matrix can be expressed by any measurement (e.g. similarity, co-occurrence).

Theoretically, *Fiedler Embedding* maps the entities into a common k-dimensional space by evaluating the relation-

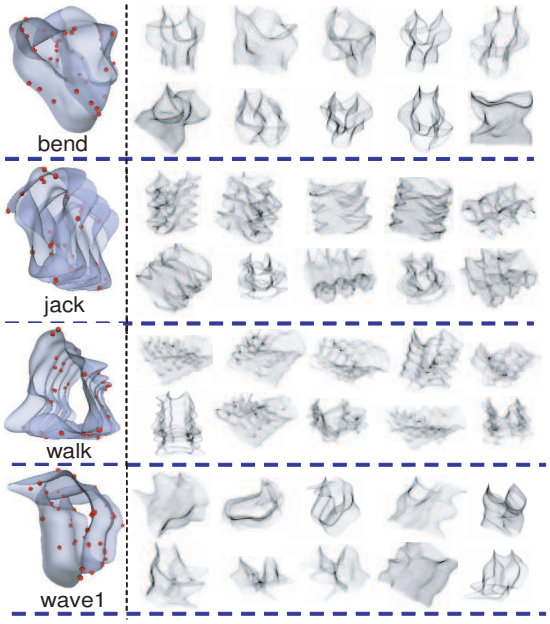

Figure 3. Some 3D (x,y,t) action volumes (the first column) with some of their sampled spin-images (red points are the *orientated points.*).

ship values between the entities. Therefore, the entities which have a stronger relationship might be placed at closer positions if the strength of relationship is evaluated using the common measurement. In other words, semantically similar entities stay closer to each other in the k-dimensional space. In practice, however, we must measure the similarity between different types of entities. For instance, the Video-Video similarity can be evaluated in term of the histogram intersection of features. Compared to histogram intersection, the frequency of occurrence is a much better measure for Video-ST or Video-SI similarity. Therefore, we can not directly compare how similar one action video V_1 is to another action video V_2 and Spin-Image SI_1 using the same type of measurement. This is because different measurements have different range. The solution lies in the proper normalization of different measurements.

In our implementation, we measure the similarities between actions and features (ST features or Spin-Image features) by *Term Frequency-Inverse Document Frequency*($tf-idf$). Assume the size of visual vocabulary (ST or Spin-Image vocabulary) is r, the action video V_d can be represented by a $r-$dimensional vector $(t_1, t_2, ..., t_i, ...t_r)$ and each entry is computed as, $t_i = \frac{n_{id}}{n_d} log \frac{N_v}{n_i}$, where n_{id} is the frequency of feature i in V_d, n_d is the total number of features in V_d, n_i is the number of feature n_i in the entire dataset and N_v is total number of action videos.

For video-video or feature-feature similarities (such as the blocks D_1, D_2 and D_3), we tried several different approaches e.g. Histogram Intersection, Cosine similarity and Inner Product. We observed that Histogram Intersection and Cosine can group similar actions together, but they were unable to group similar actions and features together. However, the Inner Product method can group the similar actions and features into the same cluster. We conjecture that this may be due to Histogram Intersection and Cosine similarity assigning disproportional weights to the video-video or feature-feature similarity, as compared to the video-feature similarity. In order to verify our conjecture, we checked the mean value of video-video block (D_1), feature-feature blocks (D_2 and D_3) and video-feature blocks (E,F,H) of the $\mathcal{L}$ matrix, Inner Product achieved close mean value for the three types of blocks, so it can treat the video-video, video-feature and feature-feature equally when assigning the weights. Thus, the action videos and features are more accurately mapped to the k-dimensional space using the Inner Product.

4. Experiments and Discussion

We applied our approach to two publicly available data sets: Weizmann action data set [6], and IXMAS [24]. On both data sets, our approach shows improvement over the baseline method which uses single type of feature. The default parameters for feature extraction are as follows. For ST features, we extracted 200 interest cuboids from each video with $\sigma = 1$ and $\tau = 2$. Then, we used k-means algorithm to generate the code books with sizes 200 and 1,000. For Spin-Image features, we uniformly sample 1,000 *orientated points* from each action volume. The k-means algorithm is applied again to quantize the feature vectors. We created two vocabularies with 600 and 1,000 *video-words* each. Finally, all the classification experiments were carried out by using K-Nearest Neighborhood (KNN) classifier with K=5.

4.1. Weizmann Action Dataset

This data set contains 10 actions performed by 9 different persons. There are total of 92 video sequences. We used leave-one-out cross-validation scheme, so we had 10 runs for each experiments, and the average accuracy is reported.

Qualitative Results. The qualitative results shows the strength of *Fiedler Embedding* which has the capability to embed semantically similar entities (e.g. action videos, spin-image features, ST features) to one close cluster in the k-dimensional space. Fig. 4 visually shows the results of the queries using different types of entities. In Fig. 4 (a)-(c), the queries consisted of features (*video-words* and shown by blue rectangles), and results of the query consisted of four nearest action videos (nearest in the Euclidian sense) from the embedding space. For each *video-word*, the percentages of the feature source (called the purity of the *video-word*) are also listed, which expresses the category specific property of the *video-word*. For instance, "wave2:99%, wave1:1%" means most of the features in this *video-word* are from action "wave2", therefore semantically it is related to "wave2". Fig. 4 (d)-(f) demonstrate three queries using features, where the returned results consist of the other nearest features from the embedding space. From Fig. 4 (f), we can note that the features of the action "run" are confused with the action "jump". In Fig. 4 (g) and (h), we performed queries using action videos. From the results, we infer that the action "pjump" is easier to recognize than the action "side". This is mainly because the *video-words* for action "pjump" have higher "purity" than those of "side". On the other hand, the action "side" might be confused with the action "run". The quantitative results in Fig. 6 further verifies this observation. In short, the qualitative results demonstrate that the constructed embedding space is meaningful, and provides us an insight into the relationships of the features.

Quantitative Results. In the following set of experiments, we show the quantitative results of our action recognition and compare them with the baselines. In order to show that the feature grouping is meaningful in the k-dimensional space, we compared the performance of classification using the original bag of *video-words* (term frequency) to that of weighted bag of *video-words* (weighted term fre-

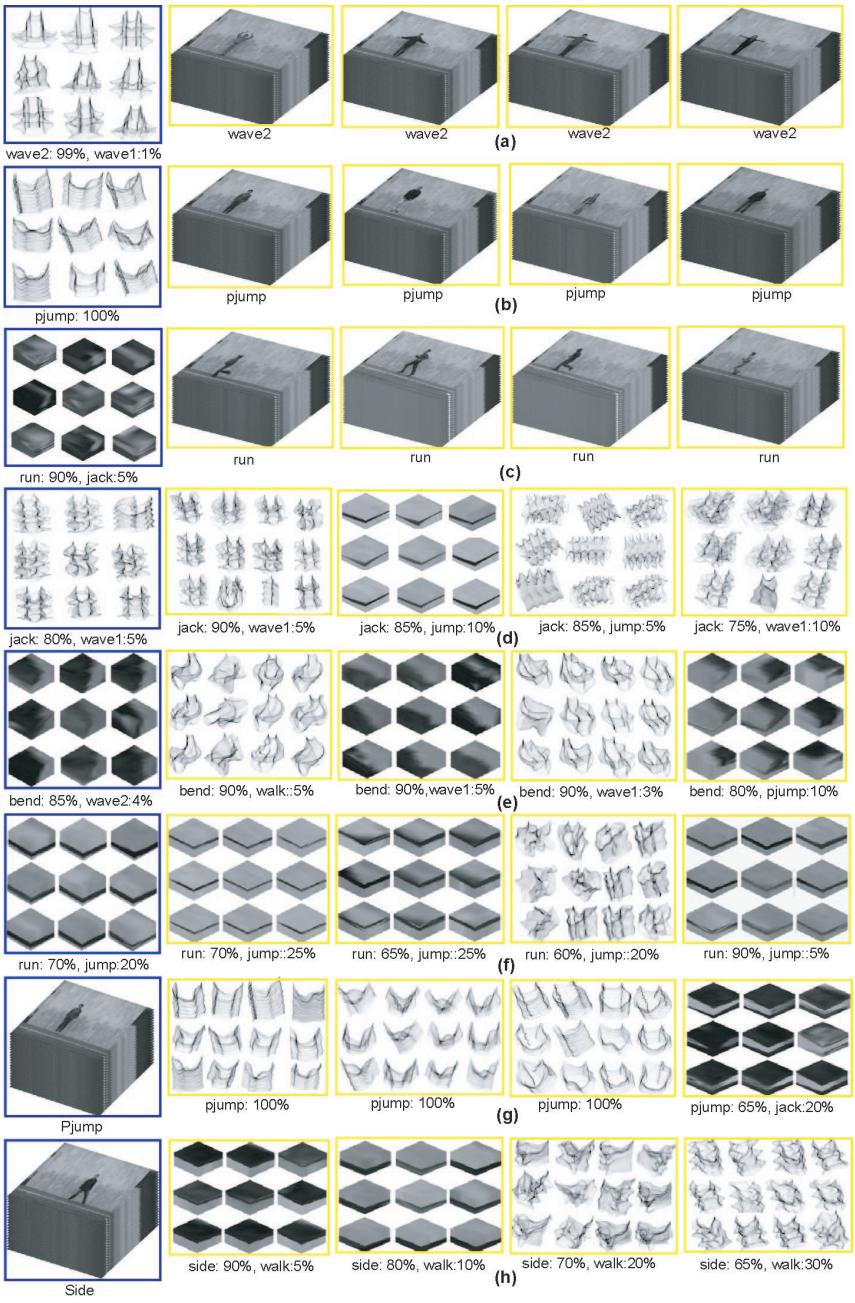

Figure 4. Figure shows different combinations of query-result that we used for qualitative verification of the constructed k-dimensional space. Each rectangle represents one entity (e.g. action video or a *video-word* (a group of features)). In (a)-(c), the features in blue which are from one *video-word* are used as query, and the 4 nearest videos in yellow from the k-dimensional space are returned. Under each *video-word*, the category component percentage is also shown (e.g. "wave2: 99%, wave1:1%" means 99% of features in this *video-word* are from "wave2" action). In (d) and (e), we respectively used ST features and Spin-Image features as query, and retrieved the nearest features in the k-dimensional space. In (f) and (g), two action videos are used as query, and the nearest features are returned.

quency) representation of the video. The weighted bag of *video-words* is constructed by using the meaningful feature groupings returned by our embedding as follows: suppose $\mathbf{t} = (t_1, t_2, ..., t_i, ...t_w)$ is the feature frequency representation of an action volume where w is vocabulary size, and $f(i, j)$ is a function which returns the j-th nearest neighbors of feature i, then we can estimate the new frequency of feature i as follows.

$$t'_i = \frac{1}{K} \sum_{j=1:K} t_{f(i,j)} \tag{7}$$

where we set K=5 (K is the number of nearest neighbors). Fig. 5 shows the performance comparison between the orig-

	Bend	Jack	Jump	Pjump	Run	Side	Walk	Wave1	Wave2	Average
Original Bag of Words	100.0	100.0	44.4	88.9	80.0	66.7	100.0	88.9	88.9	84.2
Weighted Bag of Words	100.0	100.0	55.6	100.0	80.0	100.0	100.0	88.9	88.9	90.4

(a)

	Bend	Jack	Jump	Pjump	Run	Side	Walk	Wave1	Wave2	Average
Original Bag of Words	100.0	100.0	11.1	88.9	80.0	11.1	66.7	88.9	100.0	71.9
Weighted Bag of Words	100.0	100.0	44.4	88.9	90.0	55.6	100.0	100.0	100.0	86.5

(b)

Figure 5. The comparison of the BOW approach with our weighted BOW method.

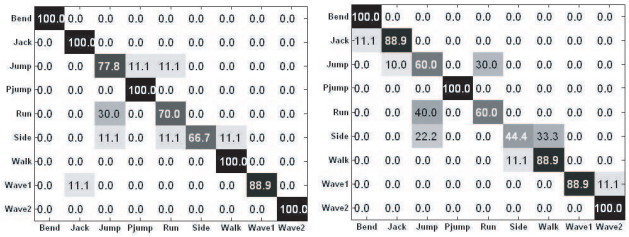

Figure 6. (a) Confusion table for *Fiedler embedding* with k=20. (b) Confusion table for LSA with k=25.

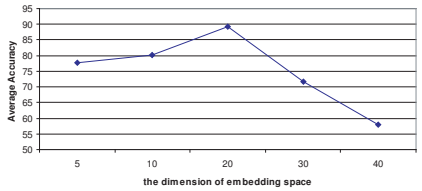

Figure 7. The variation of embedding dimension affects the performance. All experiments are carried out on $N_{si} = N_{ip} = 1,000$.

inal BOW model and our weighted BOW. In (a) and (c) we show the performance obtained using vocabulary of size $N_{si} = N_{ip} = 200$ and $N_{si} = N_{ip} = 1,000$ respectively. For the 1,000 dimensions case, our approach can improve 12% over the baseline. This is due to the fact that the constructed embedding space is semantically meaningful, and therefore it helps us discover the feature grouping that will eventually make the weighted BOW more discriminative.

In order to further verify the contribution of the relationships of different types of features, we embed "ST features" and "Spin-Image features" into two separated spaces, and also embed both into the common space. Fig.8 (a) and (b) show two sets of experiments carried on $N_{si} = N_{ip} = 200$ and $N_{si} = N_{ip} = 1,000$ respectively. It is obvious that multiple features can improve the performance by about 12%-24%. The reason is because the embedding procedure is discovering the explicit or implicit relationship between different type of features.

Next, we compared the performance of *Fiedler Embedding* based classification with the LSA. We used both features with vocabulary size of $N_{si} = N_{ip} = 1,000$. *Fiedler Embedding* achieves the best average accuracy of 89.26% when k=20, and LSA obtains 85.11% at $k = 25$. Fig.6 (a) and (b) show the confusion tables of *Fiedler embedding* and LSA respectively. Compared to the performance (71.9%)

	Bend	Jack	Jump	Pjump	Run	Side	Walk	Wave1	Wave2	Average
ST features	100	88.9	55.6	77.8	60	44.4	44.4	44.4	44.4	62.2
Spin-Image features	88.9	55.6	44.4	88.9	90	33.3	88.9	77.8	100	74.2
ST + Spin-Image features	100	100	88.9	100	70	44.4	88.9	88.9	100	86.8

(a)

	Bend	Jack	Jump	Pjump	Run	Side	Walk	Wave1	Wave2	Average
ST features	100	100	44.4	100	60	11.1	44.4	88.9	66.7	68.4
Spin-Image features	88.9	100	33.3	77.8	100	33.3	88.9	55.6	77.8	72.8
ST + Spin-Image features	100	100	77.8	100	70	66.7	100	88.9	100	89.3

(b)

Figure 8. The contributions of different features to the classification. (a)$N_{si} = N_{ip} = 200$, k=20,30 and 20 for ST features, Spin-Image features and the combination respectively. (b)$N_{si} = N_{ip} = 1,000$, k=20,70 and 20 for ST features, Spin-Image features and the combination respectively.

	cam0	cam1	cam2	cam3	average
ST+Spin-Image features	73.46	72.74	69.62	70.94	71.69
Spin-Image features	68.93	61.20	62.74	64.74	64.40
ST features	64.79	65.30	60.43	61.91	63.11

Figure 9. The performance comparison using different types of features.

of directly concatenating two types of features (the original BOW is shown in Fig5(b)) , the advantage of *Fiedler embedding* is obvious. In our experiments, we also observed that the variation of dimension k of the embedding space affects the performance. Fig.7 plots the performance variation against various choices of the dimension.

4.2. IXMAS Multiview dataset

We also applied our approach to the IXMAS multi-view data set [24]. It contains 14 daily-live actions performed three times by 12 actors. 13 action videos were selected for our experiments. Four views are included, except the top view. From each action, we extracted 200 cuboids and 400 spin-images, and separately quantized them into 1,000 *video-words*. We used 10 actors' videos for learning and the remaining two for testing. Two testing schemes were designed: recognition using single view, and recognition using multiple views. Fig. 9 shows the comparison of contribution by different types of features, which verifies that our approach can efficiently combine different types of feature. Fig.10 gives the recognition accuracy for each action using multiple types of feature. It is difficult to directly compare our results with [24] and [10] because of different experimental settings. But it is interesting to note that our approach works considerably better for the actions that involve full body motion, like in "walk", "pick up" and "turn around", while gets confused in action involving only a part of the body, like in "point", "cross arms" and "scratch head". One possible explanation is the fact that our features are orderless in nature. Average performance of each view outperforms that of [24], where the authors reported results for four views: {65.4, 70.0, 54.3, 66.0}(%). In the experiments of recognition using multiviews, we adopt the simple voting method. Fig.11 shows the confusion table of the recognition. The average accuracy is about 78.5%, which

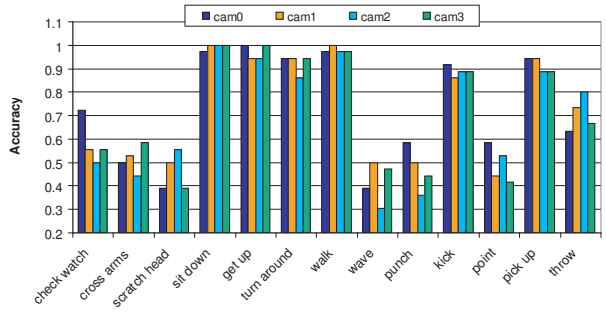

Figure 10. The recognition performance using all views for learning and only single view for testing.

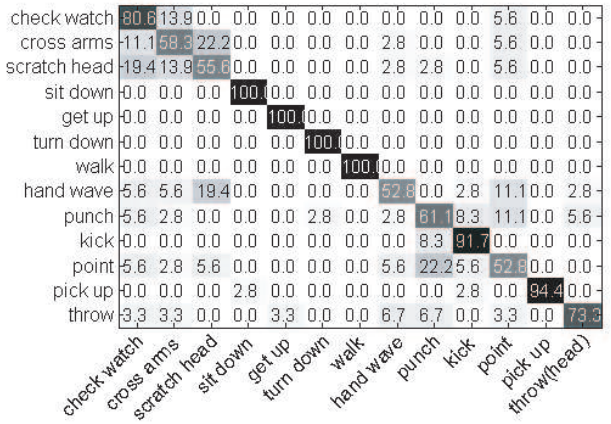

Figure 11. Confusion table for recognition using four views.

is comparable to that of [24]. Again, we can see the hand-related motions are confused with each other.

5. Conclusion

This paper employs multiple features for action recognition using Fiedler Embedding. The first feature is the quantized vocabulary of local spatio-temporal (ST) volumes (or cuboids) that are centered around 3D interest points in the video. The second feature is a quantized vocabulary of spin-images, which aims to capture the shape deformation of the actor by considering actions as 3D objects. We demonstrate that by embedding these features into a common Euclidian space, we can discover relationships among them and can use them to improve action recognition. The framework is general in nature and can be used with any type of features.

6. Acknowledgements

This research was funded by the US Government VACE program.

References

[1] A. Opelt, A. Pinz, M. Fussenegger and P. Auer, "Generic Object Recognition with Boosting", PAMI, 28(3), 2006. 2

[2] A. Opelt, A. Pinz, and A. Zisserman, "Fusing Shape and Appearance Information for Object Category Detection", BMVC, 2006. 2

[3] A. Johnson and M. Hebert. "Using Spin Images for Efficient Object Recognition in Cluttered 3D scenes", IEEE Tran. On PAMI, Vol. 21, No.5, May 1999. 3, 4

[4] S. Ali, A. Basharat and M. Shah, "Chaotic Invariants for Human Action Recognition", IEEE ICCV, 2007. 1

[5] W. Zhang, B. Yu, G.J. Zelinsky and D. Samaras, "Object Class Recognition Using Multiple Layer Boosting with Heterogenous Features", CVPR, 2005. 2

[6] M. Blank, M. Irani and R. Basri, "Actions as Space-Time Shapes", IEEE ICCV, 2005. 1, 5

[7] B. Hendrickson, "Latent Semantic Analysis and Fiedlder Retrieval", SIAM Linear Algebra and its Application, 421, pp. 345-355, 2007. 2, 3

[8] A. Yilmaz and M. Shah, "Actions Sketch: A Novel Action Representation", IEEE CVPR, 2005. 1, 2, 4

[9] J. Niebles and L. Fei-Fei, "A Hierarchical Model of Shape and Appearance for Human Action Classification", CVPR, 2007. 1

[10] F. Lv and R. Nevatia, "Single View Human Action Recognition using Key Pose Matching and Viterbi Path Searching", IEEE ICCV, 2007. 1, 7

[11] Y. Ke, R. Sukthankar and M. Hebert, "Efficient Visual Event Detection using Volumetric Features", IEEE ICCV, 2005. 1

[12] E. Shechtman and M. Irani, "Space-Time Behavior Based Correlation", IEEE CVPR, 2005. 1

[13] J. C. Niebles,H. Wang and L. Fei-Fei, "Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words", BMVC, 2006. 1, 3

[14] P. Dollar, V. Rabaud, G. Cottrell and S. Belongieet, "Behavior Recognition via Sparse Spatio-Temporal Features", IEEE International Workshop on VS-PETS, 2005. 1, 3

[15] C. Carlsson, S. Carlsson and J. Sullivan, "Action Recognition by Shape Matching to Key Frames", Workshop on Models Versus Exemplars in Compuetr Vision, 2001. 1

[16] G. Mori, X. Ren, A. A. Efros and J. Malik., "Recovering Human Body Configurations: Combining Segmentation and Recognition", CVPR, 2004. 1

[17] G. Mori and J. Malik, "Estimating Body Configurations using Shape Context Matching", IEEE ECCV, 2002. 1

[18] K.M. Cheung,S. Baker and T. Kanade, "Shape-From-Silhouette of Articulated Objects and its Use for Human Body Kinematics Estimation and Motion Capture", IEEE CVPR, 2003. 1

[19] T. S. Mahmood and A. Vasilescu, "Recognition Action Events from Multiple View Points", EventVideo01, 2001. 1

[20] J. Little and J. E. Boyd, "Recognizing People by Their Gait: The Shape of Motion", Journal of Computer Vision Research, 1998. 1

[21] A. A. Efros, A. C. Berg, G. Mori and J. Malik, "Recognizing Action at a Distance", IEEE International Conference on Computer Vision, 2003. 1

[22] I. Laptev and T. Lindeberg, "Space Time Interest Points", IEEE CVPR, 2003. 1

[23] C. Schuldt et. al., "Recognizing Human Actions: A Local SVM Approach", IEEE ICPR, 2004. 1

[24] D. Weinland, E. Boyer and R. Ronfard. Action recognition from arbitrary views using 3D examplars. In Proc. ICCV 2007. 5, 7, 8

[25] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation", Neural Computation, 15 (6):1373-1396, June 2003. Robert Pless. 2