2008－CVIM－162 (17)
2008／3／10

Comparative Study of Methods for Recognizing Human Actions from a Real Video Sequence

Weiqing WANG† Jun OHYA‡

† ‡ Graduate School of Global Information and Telecommunication Studies, Waseda University 1011 Okuboyama Nishi-Tomida, Honjo-shi, Saitama, 367-0035 Japan

E-mail: † wang.weiqing@ruri.waseda.jp, ‡ ohya@waseda.jp

Abstract This paper explores the effectiveness of using three image features instead of synthesized human motion data by using the real video sequence. We have compared three algorithms that recognize the observed action generated by an unknown person, who is not included in the database. We tested the 4 methods using 3 single image features with 4 human actors and 5 classes of action. In addition, we proved all the three method are useful for the human action recognition.

Keyword: Human Motion, Action Recognition Lt-s, N-mode SVD, Nearest Distance, Mesh, and Projection

1. Introduction

During the last two decades, detecting and recognizing human motions from video sequences is very important and became popular in the field of motion-based recognition. Although the field of action and activity representation and recognition is relatively old, it is still challenging. We can see that the pattern recognition and computer vision community have demonstrated a great and growing interest in human motion analysis from all the above activities. It means that human action recognition is still immature research field area. Thus, we are challenged to find a new method.

The aim of our research is to accurately classify the action being performed by an unknown human from real video sequence using a computer vision based approach, where the unknown human is not included in the database used for the classification process.

Our group has developed a tensor decomposition based method to modify Vasilescu's[1] tensor decomposition approach, But these methods have limitation that only works on motion capture data or synthesized human action sequences. Concerning the application needs, motion recognition technique should work with real images, because putting magnetic sensor to a human to be observed not only disturb the person but also seem unnatural. Therefore, in this paper we try to uses real human motion video sequences.

To verify the effectiveness of the features and motion recognition methods, the three kinds of image features (lt-s feature, projection feature and mesh feature) and four recognition algorithms are testified by using Matlab.

2. The overview of approach

To clarify both the effectiveness of different kind of image features, we extract image features from real video sequence, where the image features should represent different human body shapes quantitatively.

This thesis explores three kinds of image feature, i.e., Lt-s feature, projection feature and mesh feature. We test all single feature to find out which image feature can bring us the best results of the experiment.

We divide our approach into three steps. The first step is pre-processing, which is carried out before computing the features. The second step is extracting the image features (Lt-s, Projection, Mesh) based on the human silhouette and bounding box. The third step is the recognition process using different recognition algorithms, which we want to compare in this research.

Our approach is depicted in fig.1.

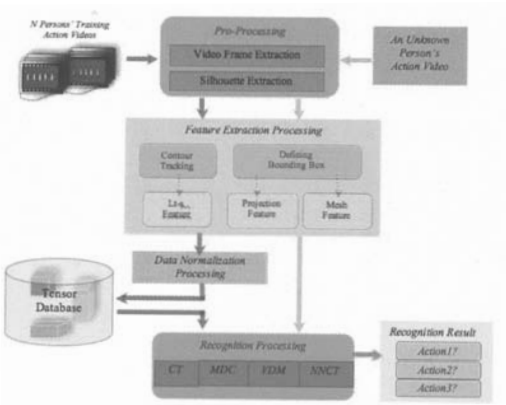

Figure 1. Conceptual Model of our Approach

3. Image Features

In this section, we describe the images features compared in our research.

3.1. Lts-feature

Lt-s feature is a set of the distance between the centroid of the human silhouette and each contour pixel of the silhouette. Basic concepts of Lt-s feature are as follows: video sequence is acquired, and then each frame extracted. Silhouette image is obtained by subtracting the original image from background image and thresholding the subtracted image. The center of mass C is obtained by computing the mean of white pixel in the human silhouette.

To obtain the distance $\mathbf{d}_i$ between centroid and each pixel along image contour is in Fig. 2, we start obtaining $\mathbf{P}_1$ as start point by scanning a pixel from the centroid vertically. Let **A** be a contour pixel; then, Lt-s = **CA** + $\mathbf{P}_1\mathbf{A}$. By computing the distance at each contour pixel, we obtain the Lt-s curve. Then, the Lt-s data in each frame are stored as feature vector. This process is continued until the end of all frames of video sequence.

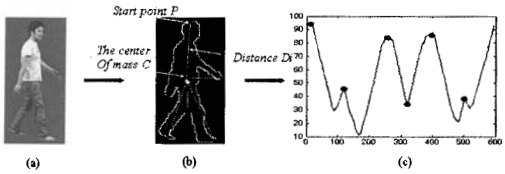

Figure 2. Method to Obtain Lt-s Feature Data.

3.2. Mesh feature

To get mesh data, as shown in Fig. 2, suppose we have $M \times N$ pixel in the bounding box **A**; then divide **A** into $m \times n$ sub- blocks. The size of each sub-block is M/m by N/n pixels. On each sub block, the ratio of human silhouette pixel's number over the number of pixels in the sub-blocks is computed. Let a_{0ij} (i=1,...m,j=1...n) be the ratio of the sub-block *ij*. Then, $f(a_{11},a_{12}...,a_{nm})$ is the mesh feature vector.

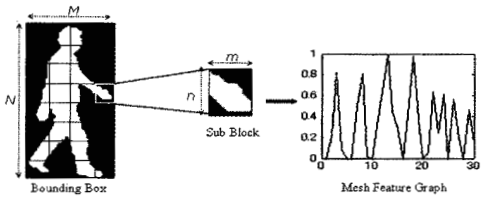

Figure. 3. Method to Obtain Mesh Feature Data.

3.3. Projection feature

To obtain projection feature data, suppose we have the size of bounding box is $M \times N$ pixels. Then, as shown in Fig. 3, in each horizontal line, the number of human silhouette pixels is counted. Similarly, in each vertical line, the number of silhouette pixels is counted. Suppose Ph*i* (i=1...N) and Pv*j* (j=1,...M) are the pixel number in i-th horizontal line and in the j-th vertical line, respectively. Then, $fp=(Ph_1,Ph_2....Ph_N,\ Pv_1,Pv_2...Pv_M)$ is the projection feature vector.

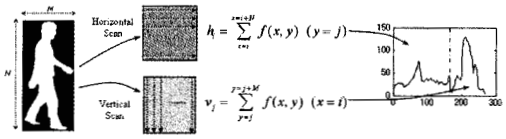

Figure 4. Pixels Calculation for Projection Feature

4. Tensor Decomposition

Basically, tensors are a generalization of the concept of a vector. A tensor can be considered to be a multi-dimensional or N-way array of data and as such is useful for the description of higher order quantities e.g. multivariate data [2]. In this thesis, we denote vector quantities by bold lower case letters (**a**, **b**), scalar quantities by lower case letters (a,b), matrices by bold uppercase letter (**A**,**B**), and tensor quantities in calligraphic letters ($\mathcal{A}, \mathcal{B}$). Generally unless explicitly stated throughout this thesis i, j refer to indices (counters) and I, J,K,L denote index upper bounds.

In multilinear algebra an Nth order tensor is written as $\mathcal{A} \in \Re^{I_1 x I_2 \cdots x I_N}$, and its elements are indexed as $a_{i1,i2,\ldots iN}$. An Nth order tensor has N mode spaces, for example in the case of a matrix, when N = 2, two mode spaces exist, a row space and a column space. In tensor terminology a matrix can be defined in terms of a set of mode-1 vectors (column vectors) or as a set of mode-2 vectors (row

vectors), e.g., Column-wise mode-1 representation $\mathbf{B} = [\mathbf{b}_{j1}...\mathbf{b}_{jN}]$, where an element of the matrix Bij, has a row index i and column index j. Considering the case of a third order tensor $\mathcal{A} \in \Re^{I_1 \times I_2 \times I_3}$, ($N = 3$), three mode spaces exist where mode- 1 corresponds to column space, mode-2 to row space, and mode-3 to depth space.

4.1 Tensor Unfolding

The main idea of a N-mode SVD derivation needs to consider an appropriate generalization of the link between the column (row) vectors and the left (right) singular vectors of a matrix. To be able to formalize this idea, we define "matrix unfolding" of a given tensor, i.e., matrix representations of that tensor in which all the column vectors are stacked one after the other [3].

A tensor $\mathcal{A} \in \Re^{I_1 \times I_2 \times \cdots \times I_N}$ can be represented in matrix form, $\mathbf{A}_{(n)}$, which is the result of unfolding (flattening) the tensor along dimension n where $n = I_1, I_2...., I_N$. Tensor unfolding can be considered as splitting a tensor into mode-n vectors and rearranging these vectors column-wise to form a matrix. In fig. 5, a visualization is presented which demonstrates how a 3rd order tensor is unfolded along mode-1 (I_1), mode-2 (I_2) and mode-3 (I_3) dimensions to form matrices $\mathbf{A}_{(1)}$ with size $I_1 \times I_2 I_3$, $\mathbf{A}_{(2)}$ with size $I_2 \times I_3 I_1$ and $\mathbf{A}_{(3)}$ with size $I_3 \times I_1 I_2$.

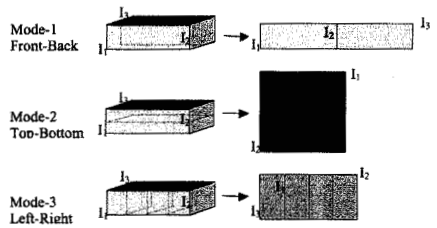

Figure 5. Tensor can be unfolded in three ways to obtain matrices comprising of its mode-1, mode-2 or mode-3 vectors.

4.1. Tensor Decomposition

4.2.1 Singular Value Decomposition (SVD)

Principal Component Analysis (PCA) is a version of Singular Value Decomposition (SVD), which is a 2–mode tool, commonly used in signal processing to reduce the dimensionality of the space and reduce noise. The singular value decomposition (SVD) of matrix A is represented by (4.1).

$$A = UWV^T \tag{4.1}$$

The matrix U is an orthogonal matrix, which spans the row space of A. The matrix V is an orthogonal matrix, which spans the column space of A. The column eigenvectors vectors of matrices U (likewise for V) are orthogonal to each other, describing a new orthogonal coordinate system for the space spanned by matrix. The columns ui and vi of the matrix U and V are called the left and right singular vectors. The diagonal element wi of matrix W are called the singular values, which are non-negative numbers in descending order, all off-diagonal elements are zeros.

The singular value decomposition has a variety of applications in scientific computing, signal processing, automatic control, and many other areas.

4.2.2 Higher Order Singular Value Decomposition (HOSVD)

In tensor notation, the N-mode tensor $\mathcal{B}$ that between tensor $\mathcal{A}$ and a matrix $\mathcal{M}$, is expressed as:

$$\mathcal{B} = \mathcal{A} \times_n M \tag{4.2}$$

In terms of tensor unfolding this can be solved as:

$$B_{(n)} = MA_{(n)} \tag{4.3}$$

Where $A_{(n)} \in \Re^{I_n \times (I_1 I_2 \cdots I_{n-1} I_{n+1} \cdots I_N)}$ is the resultant matrix of unfolding tensor $\mathcal{A}$ in direction n (mode-n), tensor $\mathcal{B}$ is founded by folding matrix $B_{(n)}$ back into tensor representation. As stated previously, a matrix has two associated modes, a vector row space and a vector column space.

Application of SVD to a matrix, B, results in the decomposition of the matrix into the product of an orthogonal column space $U1$, a diagonal singular value matrix and an orthogonal row space $U2$, which is written as:

$$B = U_1 \Sigma U_2^T \tag{4.4}$$

Using the mode-n product in Eq.(4.2) can be defined without the need of a generalized transpose as:

$$B = \Sigma \times_1 U_1 \times_2 U_2^T \tag{4.5}$$

For tensors, standard SVD cannot be utilized, therefore for $N>2$ tensor $\mathcal{D}$, by extension, higher order SVD (alternatively known as N-mode SVD) can be used. Like SVD which decomposes a matrix into 2 orthogonal spaces and a singular value matrix; HOSVD decomposes a tensor

into N orthogonal mode spaces $\mathbf{U_1}$, $\mathbf{U_2}$, ...$\mathbf{U_N}$ and a core tensor $\mathcal{Z}$. Using HOSVD [7] a tensor can be represented as the mode-n product between these N orthogonal subspaces and core tensor $\mathcal{Z}$, in Eq. (4.3).

$$\mathcal{A} = \mathcal{Z} \times_1 U_1^T \times_2 U_2^T \ldots \times_n U_n^T \ldots \times_N U_N^T \tag{4.6}$$

The core tensor, $\mathcal{Z}$, governs the interactions between the subspace (mode) matrices and it is analogous to the singular value matrix that results in standard SVD, but it does not have a diagonal structure and is a full tensor. As illustrated in Fig.6. HOSVD on a 3rd order tensor (N=3) might be result in decomposing the tensor into 3 orthogonal mode spaces (U_1, U_2 and U_3) and a 3rd order core tensor ($\mathcal{Z}$).

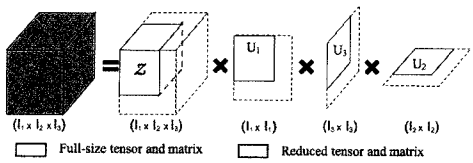

Figure 6. By N-mode SVD, N orthogonal vector spaces associated with an order-N tensor (the case N=3 is illustrated).

HOSVD algorithm for tensor decomposition as presented in [2] is given as:

For n=1 to N, 1) Unfold tensor, A, along dimension n to find matrix A(n).2) Apply SVD to matrix A(n).3) Set Un to the left-hand column space matrix of SVD. Solve the core tensor using the equation:

$$\mathcal{Z} = \mathcal{A} \times_1 U_1^T \times_2 U_2^T \ldots \times_n U_n^T \ldots \times_N U_N^T \tag{4.7}$$

4.2.3 Motion Tensor Analysis

Given motion sequences of several people, we define a data set tensor $\mathcal{D}$, ($\in \mathbb{R}^{N \times M \times T}$), where N (rows) is the number of people, M (columns) is the number of action classes, and T (depth) is the number of sequence samples. We apply the N-mode SVD algorithm to decompose this tensor as follows:

$$\mathcal{D} = \mathcal{Z} \times_1 U_1 \times_2 U_2 \times_3 U_3 \tag{4.8}$$

By denoting U_1, U_2, U_3 as P, A, F respectively, we get the product of a core tensor, and three orthogonal matrices as follows:

$$\mathcal{D} = \mathcal{Z} \times_1 P_1 \times_2 A_2 \times_3 F_3 \tag{4.9}$$

The *people* matrix in represented by

$$P = [p_1 p_2 \cdots p_n \cdots p_N]^T \tag{4.10}$$

Where person specific row vectors p_n^T span the space of person parameters, and encode the per-person invariance across actions. Thus *people* matrix P contains the human motion signatures. The *action* matrix in represented by

$$A = [a_1 a_2 \cdots a_m \cdots a_M]^T \tag{4.11}$$

Where action specific row vectors a_n^T span the space of action parameters, and encode the invariance for each action across different people. The row vectors of *frame* matrix in represented by

$$F = [f_1 f_2 \cdots f_t \cdots f_T]^T \tag{4.12}$$

Where specific row vectors f_n^T span the space of time series image features, and encode the invariance for all actions across different people.

The tensor

$$\mathcal{B} = \mathcal{Z} \times_2 A_2 \times_3 F_3 \tag{4.13}$$

contains a set of basis matrices for all the actions associated with particular actions.

The tensor

$$C = \mathcal{Z} \times_1 P_1 \times_3 F_3 \tag{4.14}$$

contains a set of basis matrices for all the people associated with particular people.

4.2. Algorithms of Human Action Recognition

Given motion sequences of several people, the database is represented by a tensor where M (rows) is the number of people, N (columns) is number of action classes, and T (depth) is the number of sequence samples, as shown in Fig.7.

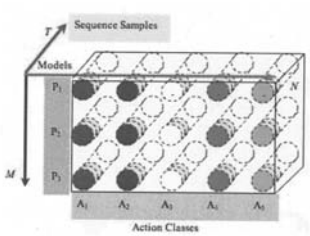

Figure 7. Motion Database Structure

The observed motion sequence of unknown person is represented by a tensor. Obviously, we do not know the action of D unknown.

Therefore, we assume unknown's action is ($j \in 1,...,N$) one of the N actions to be recognized. all of this new person's actions are synthesized. This process is repeated for all the actions j = 1 through N.

The four recognition algorithms based on the database tensor $\mathcal{D}$ are detailed in the following.

4.3.1 Algorithm by using Core Tensor

As described in Section 4.1.3, using HOSVD any tensor can be represented by a core tensor $\mathcal{Z}$, as indicated in Eq. (4.8). This algorithm uses the core tensor $\mathcal{Z}$ of the database tensor $\mathcal{D}$ for obtaining the recognition result by finding the best assumption for $\mathcal{D}_{\text{unknown}}$. However, if we simply append the synthesized actions to the database tensor $\mathcal{D}$, the core tensors of $\mathcal{D}$ and the appended tensor cannot be compared, because the sizes of the two tensors are different. Instead, one person's action in the database tensor $\mathcal{D}$ is replaced by the unknown action to get the synthesized action data $\mathcal{D}_{ij}$, so that the core tensor of $\mathcal{D}_{ij}$ and $\mathcal{D}$ have the same size. This replacement is repeated for all the persons (M in total) in $\mathcal{D}$, and the core tensor is computed for each time. The difference between the new and original core tensors is obtained by computing the summation of the absolute values of element-wise differences. The replacement that gives the minimal difference could correspond to the case in which the synthesized actions are very similar to the replaced actions. Thus, the assumed action for this particular replacement is determined as the recognition result.

4.3.2 Algorithm by Using Vector Distance Measures

In this algorithm we do not use the core tensor for action recognition, to avoid computing the core tensor $M \times N$ times. Instead, the motion vector $\mathbf{a}_{(m,n)}$ in database tensor $\mathcal{D}$ can be compared with the unknown action's vector $\mathbf{a}_u$ (we assume the unknown action is one of the N actions to be recognized.), because the two action vectors have the same size. The *Euclidean distance* between two vectors $\mathbf{x},\mathbf{y}$ is defined as:

$$\text{Dis}(\mathbf{a}_u, \mathbf{a}_{(m,n)}) = \left\| \mathbf{a}_u - \mathbf{a}_{(m,n)} \right\| \Rightarrow \min \tag{4.15}$$

$\text{Dis}(\mathbf{a}_u, \mathbf{a}_{(m,n)})$ that gives the shortest distance in Eq.(4.15) could correspond to the case in which the motion vector in the database is very similar to the unknown action. Thus, the assumed action is determined as the recognition result.

4.3.3 Algorithm by using Minimum-Distance Classifiers

In general, an action class, w_j,is characterized by its mean vector m_j . That is, we use the mean vector of each population of training vectors as being representative of that class of vectors:

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in w_j} \mathbf{x} \quad (j = 1,2,...,W) \tag{4.16}$$

Where N_j is the number of training action vectors from class w_j and the summation is taken over these vectors; W is the number of action classes. One-way to determine the class membership of an unknown action vector $\mathbf{x}$ is to assign it to the class of its closest prototype. We then assign $\mathbf{x}$ to class w_j if $d(\mathrm{m_j},\mathbf{x})$ is the smallest distance.

$$d(\mathbf{m}_j, \mathbf{x}) = \min_j \left\{ d(\mathbf{m}_j, \mathbf{x}) \right\} \tag{4.18}$$

That is, the smallest distance implies the best match.

Suppose that all the mean action vectors $\bar{\mathbf{a}}_u$ are organized as columns of a matrix space A by flatten the database tensor $\mathcal{D}$ with action-mode. Then computing the distances from unknown person's action vector $\mathbf{a}_u$ to all the mean action vectors $\bar{\mathbf{a}}_u$ is accomplished by using Eq. (4.16).

4.3.4 Algorithm by using Nearest-Neighbor of Core Tensor

This algorithm uses Nearest-Neighbor classification method, which is one of the most fundamental and simple classification methods, to work out classification of all the training actions. Then we use the core tensor concept to find out recognition result without using the Euclidean distance. For this aim, firstly we suppose that all the mean action vectors $\bar{\mathbf{a}}_u$ are organized as columns of a matrix space A by flatten the database tensor $\mathcal{D}$ with action-mode. Therefore, we get average motion tensor $\mathcal{A}$, and solve for the core tensor $\mathcal{Z}_n$

For each mean motion vector $\bar{\mathbf{a}}_u$ in tensor $\mathcal{A}$ replace with

unknown action signature to get synthesized action data $\mathcal{A}_n$, and solve for the core tensor $\mathcal{Z}_{nu}$. This replacement is repeated for all mean motion vectors $\overline{a_{(u,t)}}$ in tensor $\mathcal{A}$, and the core tensor is computed for each replacement. The difference between the new and original core tensors is obtained by computing the summation of the absolute values of element-wise differences. The replacement that gives the minimal difference could correspond to the case in which the synthesized action classes are very similar to the replaced actions. Thus, the assumed action class for this particular replacement is determined as the recognition result.

5. Experimental Results

5.1. Data Acquisition

For the evaluation, we recorded twenty video sequences containing five kinds of human actions: walking, jumping, crossing arms in front of body, sit down and get up, waving arms over head, performed by four actors. The human actors consist of three of young people and an aged people. We choose four different human models, because these human actors have different physical characters, they also perform actions with different styles both in form and speed. Therefore, there are more realistic data for our experiment can be provided. We used a static color CCD video camera with 30fps frame rate, which is positioned on one side of a 5 meters long walkway. The sequences were down sampled the spatial resolution of 240×320 pixels and have a length of two seconds in average. The programs used MATLAB R2006a to write and performed the applications for realizing the motion recognition methods, which we need to compare.

5.2. Recognition Results

After we create all the tensor databases, then we continue to test the recognition process step. In this test we use "Leave one-out" validation.

Table.1 Comparison of the Accuracy among 4 different Motion Recognition Methods

Method / Features	Core Tensor	Vector Distance Measure	Minimum-Distance Classifiers	Nearest Neighbor Core Tensor
Lt-s	85%	85%	85%	65%
Projection	60%	80%	80%	65%
Mesh	90%	85%	95%	70%

6. Conclusion

From all the recognition results above, we have some notes for conclusion.

◆ Each of the four algorithms by using the above database in our experiment convergent to a desired accuracy. It also demonstrated that all of these algorithms and the approach are effective and efficient to human motion recognition. However, it also shows that the method based on Core Tensor was not robust to all the image feature data.

◆ The image features (Lt-s, Projection, Mesh) we used in the experiments are useful for the motion feature data. The mesh feature has the most significant results for recognizing front-view motion image than the two others. However, those experimental results are limited in this set of data above.

◆ We also found that without Core Tensor, the other two methods, that we used can shorten processing time so that recognition process can work on real time.

References

1. M.A.O.Vasilescu, "Human Motion Signatures: Analysis, Synthesis, Recognition", International Conference on Pattern Recognition (ICPR'02).
2. Rovshan Kalanov, Jieun Cho and Jun Ohya, "A Study of Synthesizing of New Human Motions from Sampled Motions Using Tensor", ICME2005 (IEEE International Conference on Multimedia and Expo), CD-ROM Proceedings, 4 pages, (2005.7).
3. Acep Irawan, et al., Tensor Decomposition Framework for Recognizing an Unknown Person Action from Video sequence Using Image Features, FIT2007. on Information Technology (2007.9).