3D Action Recognition by Learning Sequences of Poses

A. Method

0. Ideas

This paper is inspired by the intuition that under a good model, the same class action's distance is rather small while different class action's distance should be as big as possible. This thought is the same as SVM's. But how to define this kind of distance? It is important to know the fact that even the same action will act with different rate vary in time or space leads to different sequence length. An action is a sequence of poses (frames). Therefore, when two action are compared to calculate their distance, a time normalized measure should be taken. DTW(dynamic time warping [1]) is such a good technique.

[Problem to Discuss]

1. How to represent a pose properly?
2. How to calculate the distance between two sequences with different length?

1. Skeletal Representation (body part-based, frame level descriptor)

The proposed feature vector contains two part: a set of trajectory vectors and a set of edge vectors. N is the number joints, pose is the similar meaning with frame. An action is a sequence of poses.

1) trajectory vector (3 dimension each one)

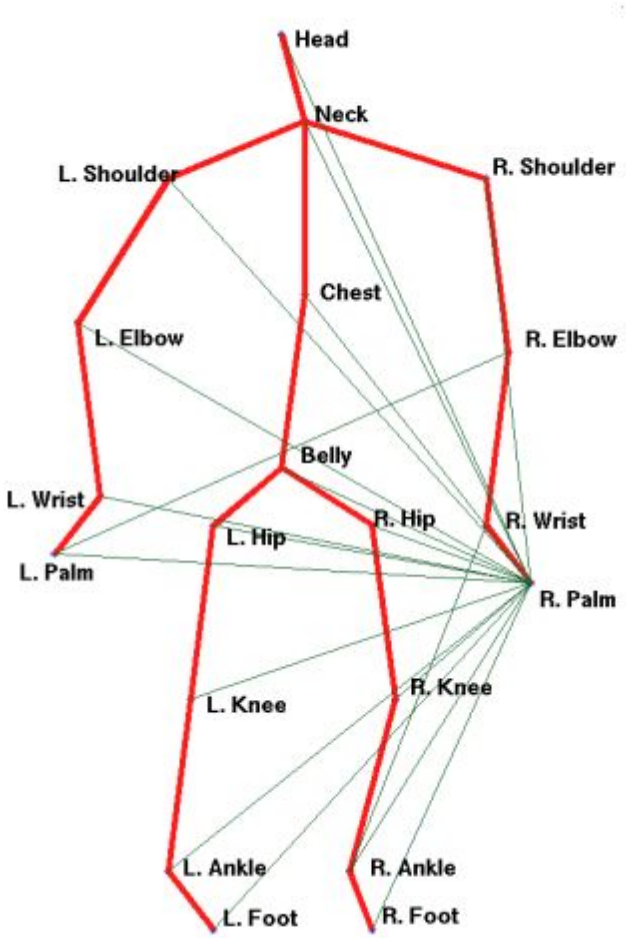
Thinking of two continous poses, the vector that connects each joint' position with their postion at the previous time instance is called a trajectory vector.

2) edge vector (3 dimension)

Pairwise connect each two different joints, we get a complete graph of joints, denoted as G. Each edge in G, say ei, has a score, which is a statistical magnitude. In each G in an action, an edge's(a vector decided by the two ends' coordinate) score is calculated as the mean of its temporal variance across the whole pose sequence. Select the top-largest-score N edges as the set of edge vector. Each edge vector here is not meant to an exact value, but a part which will be used to calculate in every frame through the sequence. For example, as the picture 1, the green edge is selected for calculating the edge vectors.

The reason for this process is not presented in the paper. But in my opionion, maybe it's trying to choose those joints who move a lot. For example, the action of picture 1 is waving his right hand, the

right part changes a lot across the pose sequence. As a result, the above process focus almost all edges on the right part.



Picture 1 an example for selecting edge vector

2. The proposed Algorithm

The distance between two pose is defined as picture 2, but how about the distance between two sequence? In most times, two sequences will have different length. They apply DTW to solve this problem, get a cumulative cost as the distance of the two input sequence.

1) Training

Then how to train? A simple idea is to use SVM. But it will cose highly because the cost to calculate a distance between two sample is not a constant time(but O(m + n), where m and n is the length of the two sequence). Therefore, they propose a new method to train a model. Set the shortest sequence as the initial model, apply DTW to warp with each other sequence, the new model is the mean of these

warped sequences, and loop until the model convergence. They say that the iteration is usually less than 15 in practice, so it is efficient.

With this representation, dissimilarity between two poses $P : \{\{e_P^i\}_{i=1}^n, \{t_P^i\}_{i=1}^n\}$, $Q : \{\{e_Q^i\}_{i=1}^n, \{t_Q^i\}_{i=1}^n\}$ can be computed as follows:

$$distance(P,Q) = \sum_{i=1}^{n} \| e_P^i - e_Q^i \|_2 + \alpha \sum_{i=1}^{n} \| t_P^i - t_Q^i \|_2 \quad (1)$$

where e_P^i, e_Q^i are edge vectors and t_P^i, t_Q^i are trajectory vectors of poses P and Q respectively. α is the relative weight given to trajectory vectors.

Picture 2

2) Action Recognition

After training, each action class has a model sequence. Therefore, action recognition is a simple process that just calculate the distance between the input sequnce and each model sequence, select the smallest one as the result of classification. The accuracy is rather high! It says that they have 98.33% on MHAD.

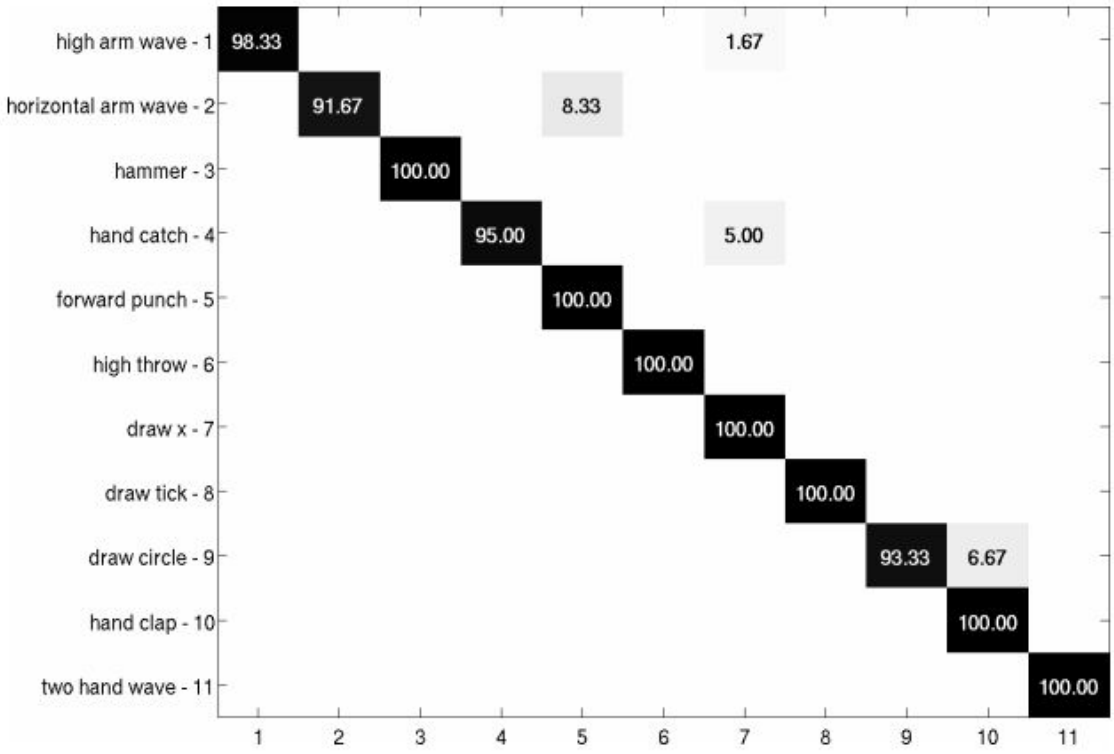
B. New things

1) Edge vector focus on the frequent-move-part is a pretty good idea.

2) Add a factor alpha bigger than one (like 10) when calculate the distance between two poses, because the trajectory's change is rather small in a small time slot. This care may often be ignored by me.

3) The result is presented as a form like picture 3, which I think is rather intuitive for others to get the accuracy of this method.

4) Using DTW.



Picture 3

C. Shortcomings

1) This method may not be able to perform in real-time environment because it need to capture the whole sequence to caculate the temporal variance to get the edge vectors.

D. Reference

1. DTW algorithm: DTW, wiki.