Fully Automatic Methodology for Human Action Recognition Incorporating Dynamic Information

Ana González, Marcos Ortega Hortas, and Manuel G. Penedo

University of A Coruña, VARPA group, A Coruña 15071, Spain
{ana.gonzalez,mortega,mgpenedo}@udc.es

Abstract. In this paper, a star-skeleton-based methodology is described for analyzing the motion of a human target in a video sequence. Star skeleton is a fast skeletonization technique by connecting centroid of target object to its contour extremes. We represent the skeleton as a five-dimensional vector, which includes information about the positions of head and four limbs of a human shape in a given frame. In this manner, an action is composed of a sequence of star skeletons. With the purpose of use an HMM which allows model the actions, a posture codebook is built integrating star skeleton and motion information. With this last information we can distinct better between actions. Supervised (manual) and No-supervised methods (clustering-based methodology) have been used to create the posture codebook. The codebook is dependently of the actions to represent (We choose four actions as example: walk, jump, wave and jack). Obtained results show, firstly, including motion information is important to get a correctly differentiation between actions. On the other hand, using a clustering methodology to create the codebook causes a substantial improvement in results.

Keywords: Human action recognition, Star skeleton, Clustering, Hidden Markov Models.

1 Introduction

Vision-based human action recognition is currently a significant research topic, since it can be useful for a wide variety of applications, such as video indexing and browsing, virtual or augmented reality.

Several human action recognition methods were proposed in the past few years: model-based methods, eigenspace technique and Hidden Markov Model. HMM has been used successfully in speech recognition and is a training based recognition technique. HMM transforms the problem of action recognition into the problem of pattern recognition. Yamato et al. [6] are the first researchers who applied HMM for action recognition. Some of the recent works [1], [2], [4] and [7] have shown that HMM performs well in human action recognition as well.

This paper [1] proposes an action recognition method based on HMM using star skeleton as the recognition feature, with a symbol codebook built using

C. San Martin and S.-W. Kim (Eds.): CIARP 2011, LNCS 7042, pp. 173–180, 2011.

clustering-based methodology, in an automatic way. In addition, each symbol of our codebook includes information of the motion direction of the target in order to consider dynamic information.

The paper is organized as follows. In section 2, we make a review of the system proposed by [1] and [2]. In section 3 we introduce the automatic generation of the codebook. We also introduce the need of including dynamic information (direction vector) in the posture of a human in base to complete the description of each symbol and improve the action recognition. Section 4 shows some experimental results and, finally, section 5 offers final conclusions and future work discussion.

2 Action Recognition Using Star Skeletonization

The system architecture consists of three parts: feature extraction, mapping features to symbols and action recognition.

A frame is processed to extract the contours from a human silhouette. Then, the associated star skeleton is obtained according to a distance defined in the skeleton space. A posture codebook is built containing representative star skeletons (symbols) of each action. When the system gets a new star skeleton, it is mapped to the most similar symbol in the codebook. HMMs are used to model the different actions. They receive a symbol sequence and give a probability value associated to the action each HMM is trained for.

2.1 Feature Extraction

Obviously, there is no perfect motion detection algorithm. In this approach, a very simple method is used: background subtraction. There will be spurious pixels detected and other anomalies, so a preprocessing step is needed. The difference between the background image and the current frame is binarized and morphological dilation and erosion are used to extract a high quality border contour.

The method proposed by [2] provides a real-time method for detecting extremal points on the boundary of the target to get the star skeleton. This structure consists of the head and four limbs of a human joined to its centroid. This is described as follows:

– Considering (x_i, y_i) points from border contour and N_b the total number of these points, the centroid (x_c, y_c) of the target is determined by:

$$x_c = \frac{1}{N_b}\sum_{i=1}^{N_b} x_i \quad y_c = \frac{1}{N_b}\sum_{i=1}^{N_b} y_i \tag{1}$$

– The distances d_i (d_i is expressed as a one dimensional signal $d(i) = d_i$) from each border point to centroid are calculated:

$$d_i = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \tag{2}$$

- d_i is smoothed (d'_i) to reduce noise using a smoothing filter.
- The extremal points come by the local maxima of (d'_i). The star structure is built connecting them to the target centroid.
- In order to make the process valid independently of the person characteristics, normalization should be made to get relative distribution of the feature vectors. This is achieved dividing vectors by human dimensions (width and height).

Figure 1 shows examples of different skeletons obtained from person contours.

Fig. 1. Star skeletons obtained for distinct human targets

2.2 Mapping Features to Symbols

Once we have the star skeleton, we must define the concept of distance between feature vectors. The star skeleton is made up of five sub-vectors, so star distance between two feature vectors U and V ($d^*(U,V)$) could be defined as the sum of the Euclidean distances of the five sub-vectors.

$$d^*(U,V) = \sum_{i=1}^{5} ||U_i - V_i|| \tag{3}$$

However, it is necessary to consider that the particular position of each human extremal is unknown, so a greedy mapping is needed to compare two feature vectors. The star distance is defined by minimal sum of the five sub-vectors in U in all permutations, where K is the total number of possible permutations and U_i^p is the sub-vector i in the permutation p in the feature vector U:

$$d^*(U,V) = \arg\min_{p} \sum_{i=1}^{5} ||U_i^p - V_i|| \quad p \in 1..K \tag{4}$$

With the star distance defined, we can build a codebook with the most representative star skeletons for all the actions considered. Star skeletons obtained of each frame will be mapped to a symbol in the codebook to make the action recognition possible. Using a finite set of symbols the codebook is built manually: they choose subjectively symbols are subjectively chosen to be representative in a visual way.

2.3 Action Recognition

In order to achieve the action recognition, one HMM is designed for each action to be considered. The number of states was empirically determined. Once we

have each action model trained (with symbols from the codebook), we calculate the probability of each model of generate the observation posture sequence. Considering the set of h HMMs of the model (where P_i is the HMM for action i) and a given symbol sequence s, the action associated to s will be calculated as:

$$\arg\max_i P_i(s), i = 1..h \tag{5}$$

where $P_i(s)$ is the probability of s in P_i. Also, it is required that the given sequence ends in a final state of P_i

3 Automatic Generation of Codebook

The process of constructing the codebook is manual in the current model. It carries some disadvantages, such as the subjective selection of symbols by a human, as well as the fact that symbols may not be representative enough in the codebook. Also, in a manual selection the time of the processing is high and tedious.

All these limitations can be improved choosing the most representative star skeletons, using an automatic method. In this paper we propose a way of building the codebook using a clustering-based method to automatically select the components in the codebook.

The method used to get the codebook symbols is the K-means, where the number of clusters is prefixed before the clustering. This number was chosen in base of the number of symbols used by [1] and considering a sufficient number to modeling the actions successfully. However, K-means algorithm has some limitations. In this case, the main inconvenient is the initialization-dependency inherent to the algorithm. This may lead to bad results, because in the training set there could be symbols more frequent than others. If in the initialization step, many clusters are assigned to very similar symbols, at the end of the process, some clusters will not have any symbol of the training set assigned. In addition, the rest of the clusters will not represent successfully all the possible symbols.

In order to find a solution for the problem of the K-means initialization, we introduce a modification at this point. First clusters must be really representative of the training set. The variance in the group is considered in order to extract the most different symbols from the set, so clusters are initialized as follows:

1. We choose cluster c as the symbol from the training set which minimizes the variance in the symbols group $G - c$:

$$\arg\min_c \texttt{Var}(G - c), c \in G \tag{6}$$

2. We compare each symbol in the training set with the first cluster using (4). Symbols $c' \in G$ too similar to the first cluster (up to a threshold λ, i.e. $|c - c'| \leq \lambda$) are eliminated from the training set: $G' = G - c'$.

3. If $|C \cup c| < n$ `and` $|G'| > 0$ then $G = G'$ and go to step 1; else $n = |C|$.

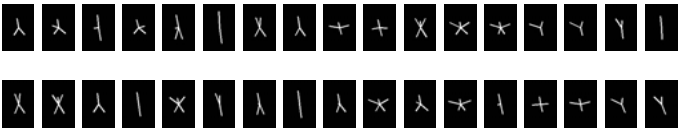


Fig. 2. 17-cluster codebook generated using manual method (above) and K-means algorithm (below) with initial $n = 20$ and $\lambda = 0.95$

Fig. 3. Sequence of star skeletons for action *walk* (left) and *jump* (right)

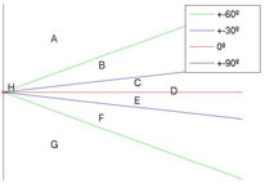
The main problem in this step is choose λ to discriminate the similarity between symbols. λ was selected empirically. High values of λ make elimination process too aggressive, leaving some clusters with no symbols assigned. This situation is also possible if the variance between symbols is very low. In these cases, the final number of clusters is automatically reduced by K-means. Figure 2 shows examples of codebooks, manual and K-means-generated by. Pattern recognition techniques present problems to distinguish very similar patterns. In our case, some symbols in the codebook are shared by many actions; even a sequence of an action can be a subsequence in another one, as Figure 3 shows. If we want to differentiate them, it is necessary to represent them in a different way. Using position of each human extremal to represent a posture is not enough, so we add dynamic information to the symbols in the codebook. This information includes the direction of motion of the human target between consecutive frames. In this way, we can know if the star skeleton calculated in a frame is displaced respect the star skeleton in the previous.

The most important is to know the variation respect Y-axis: this can be a difference between the symbols in jump and walk sequence. It is also important know if the human target is still or moving, but system must work independently if he is moving left or right in the scene. Motions are accordingly transformed from left to right in any case.

The displacement of a human target is defined as the variation of the center of human target in the Y-axis over the X-axis during a sequence of frames. For an easier processing, we work with the angle of variation (measured in degrees) to evaluate the magnitude and direction of motion. This angle α of motion in the frame i respect to previous frame is computed as:

$$\alpha = \arctan((y_i - y_{i-1})/(x_i - x_{i-1})) \tag{7}$$

Where x_i and y_i are the pixels representing the mean point of human target dimensions in the frame i. The motion direction in function of α is shown in Figure 4. A color-coding is used to represent it in the skeleton figures. This dynamic component must be added to each symbol in the codebook, so the final



Legend	Angle of motion(Degrees)
	$90 \leq \alpha > 60$
	$60 \geq \alpha > 30$
	$30 \geq \alpha > 0$
	$\alpha = 0$
	$-30 \leq \alpha < 0$
	$-60 \leq \alpha < -30$
	$-90 \leq \alpha < -60$
	No motion

Fig. 4. Left: Diagram with the different types of possible direction in motion sequences. Right: Coding of the different directions of motion.

codebook will be the Cartesian product of the star skeletons (symbols obtained by K-means method) with all possible directions of displacement A calculated as (7) shows. A symbol s in the codebook is now defined as (8). When a new star skeleton comes we look for the most similar symbol in the new codebook, which is now a Cartesian product of skeleton symbols and direction symbols.

$$s = (c, \alpha) \quad c \in C, \alpha \in A \tag{8}$$

4 Results

Video sequences used in training and test phases have been token from [8] database, some frames of them are shown as example in Figure 5. Many sequences are extracted from each video and The number of sequences in the training set for each considered action is 206 (walk), 172 (jump), 98 (wave) and 104 (jack). Table 1 shows the confusion matrix of testing data. The left side is the ground truth of action types and the upper side is the recognition action types. It is possible to know in which type of action the system misclassifies sequences. From Table 1, left and center, we can see that a 91% of sequences are classified successfully with the manual method, while, with the K-means-based methodology this rate is over 97%. K-means achieved a significant improvement.

Empirically, we observed that if a sequence mapped to codebook symbols has an odd or not frequent symbol, the associated HMM can produce a high probability but, in some cases, it does not have to finish in a Markov final state. If the rest HMMs produce very low probability, we can use a threshold t to discriminate when it is neccesary to take the HMM with highest probability even if it had not finished in a final state. Thus, we follow the criterion given by (5) but introducing a new constraint: even in the case that model P_i does not end processing the sequence in a final state, the sequence is assigned to P_i if

$$P_i(s) \geq P_i^{'}(s) + t \qquad i = 1..h, \quad t \in [0\,, 1] \tag{9}$$

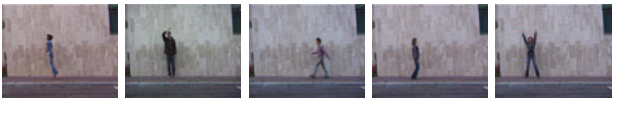


Fig. 5. Examples of action video sequences from used database

Table 1. Confusion matrix for recognition of testing data using manual codebook (left) and K-means codebook (center). Confusion matrix for recognition of testing data using K-means codebook and HMMs with $t = 0.65$ (right).

	Walk	Jump	Wave	Jack
Walk	27	1	0	3
Jump	0	26	0	0
Wave	0	7	63	0
Jack	0	2	0	20

	Walk	Jump	Wave	Jack
Walk	28	2	0	1
Jump	0	26	0	0
Wave	0	0	70	0
Jack	0	1	0	21

	Walk	Jump	Wave	Jack
Walk	30	1	0	0
Jump	0	26	0	0
Wave	0	0	70	0
Jack	0	0	0	22

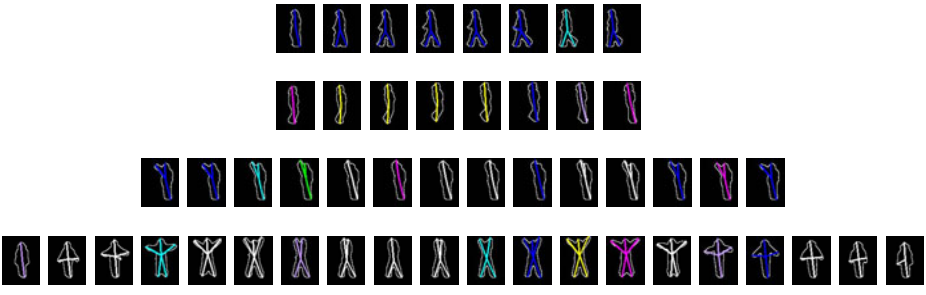


Fig. 6. Stars skeleton mapped to walk, jump, wave and jack action sequences

Adding this constraint improves the performance rendered by our HMMs, as shown in table 1, right. Finally, Figure 6 presents examples of sequences for different actions. It shows the contour of human targets and the symbols in the codebook matched with them. The direction obtained is also indicated in the skeletons with the color coding shown in Figure 4. It is simple to see that, although the sequences share many symbols, specially jump ones, the direction associated is different in each action and problems of shared symbols and subsequences are solved.

5 Conclusions and Future Work

In this paper improvements on action recognition based in star skeleton has been presented. Codebook selection is fully automatic, avoiding human intervention and its subjectivity. In addition, manual method chooses clusters among a finite symbols set, while the proposed clustering-based methodology chooses a optimal representation of the symbols, independently of set size and its variety.

K-means-method chooses initial clusters considering the variance in the symbols set, so the symbols chosen are the most representative. The number of clusters is enough to represent successfully all the actions presented as example.

Other methodologies can also be used to build the codebook in an automatic fashion, as Neural Networks. With this methodology more complex actions than those presented here could be considered. For more realistic applications it would

be possible the continued recognition of successive sequences with different actions using dynamic HMM.

Acknowledgements. This paper has been partly funded by the Xunta de Galicia through the 10TIC009CT and 10/CSA918054PR grant contracts.

References

1. Chen, H.-S., Chen, H.-T., Chen, Y.-W., Lee, S.-Y.: Human Action Recognition Using Star Skeleton. In: Proc. VSSN 2006, Santa Barbara, CA, USA, pp. 171–178 (October 2006)
2. Fujiyoshi, H., Lipton, J., Kanade, T.: Real-Time Human Motion Analysis by Image Skeletonization. IEICE Transactions on Information and Systems E87-d(1), 113–119 (2004)
3. Aggarwal, J.K., Cai, Q.: Human motion analysis: A review. Computer Vision Image Understanding 73(3), 428–440 (1999)
4. Aloysius, L.H.W., Dong, G., Huang, Z., Tan, T.: Human Posture Recognition in Video Sequence using Pseudo 2-D Hidden Markov Models. In: Proc. International Conference on Control, Automation, Robotics and Vision Conference, vol. 1, pp. 712–716 (2004)
5. Kellokumpu, V., Pietikäinen, M., Heikkilä, J.: Human Activity Recognition Using Sequences of Postures. In: Proc. IAPR Conference on Machine Vision Application, pp. 570–573 (2005)
6. Yamato, J., Ohya, J., Ishii, K.: Recognizing Human Action in Time-Sequential Images using Hidden Markov Model. In: Proc. IEEE International Conference on Computer Vision and Pattern Recognition, pp. 379–385 (1992)
7. Cucchiara, R., Grana, C., Prati, A., Vezzani, R.: Probabilistic posture classification for Human-behavior analysis. IEEE Transactions on Systems, Man and Cybernetics 35(1), 42–54 (2005)
8. Classification Database (2005), `http://www.wisdom.weizmann.ac.il/~vision/SpaceTimeActions.html`