Pose-Based Clustering in Action Sequences

Gareth Loy, Josephine Sullivan and Stefan Carlsson
Numerical Analysis and Computing Science
Royal Institute of Technology (KTH)
Stockholm, Sweden S - 100 44

Abstract

A method is presented for automatically extracting key frames from an image sequence. The sequence is divided into clusters of frames with similar appearance, and the most central frame in each cluster defines a key frame. Clustering is done using an extension of the normalized cut segmentation technique based on the inter-frame similarities. The similarity between every pair of frames in the sequence is determined from the spatial image characteristics via a shape matching technique. Our algorithm is demonstrated successfully extracting 20 key frames for a tennis player in action over a 30 second (900 frame) video sequence.

1 Introduction

The content of a video sequence can often be characterized by the human activity taking place. This applies to a variety of sources such as feature movies, home video, surveillance systems, news and especially sports broadcast TV. The ability to automatically describe human activity in long video sequences is a crucial step towards building a general tool for the classification of video content. This has potential applications for automatic editing, archiving, browsing and education.

Ideally, we would like to classify the content of a video sequence in a totally unsupervised way without any prior knowledge about the types of actions being performed. This will be done by locating human actors in the video and measure similarity of activity across time. The key element needed here is a method to assess similarity of action independent of actor. Given such a method the activity across time can be described by the structure of an activity affinity or co-occurrence matrix. Specific reoccurring actions will be obtained by applying segmentation algorithms to this matrix. This methodology will automatically generate exemplars in the form of key-frames, to represent actions. Single or multiple exemplars will be used to characterize a specific action. These key-frames can depict the whole body of the acting person or just parts of it. In the latter case, combinations of key frames can be used to depict the whole body action.

Such an approach poses the problem of action recognition as one of pure recognition, i.e. as a general image matching problem. This is in contrast to approaches that try to decompose action by tracking human motion in space and time. Any human action has specific dynamics defined by changes of posture. Often a discrete set of postures are associated with a specific action which can be schematically broken down into transitions between elements of this set. The problem of selecting this set for a specific action is similar to the problem of selecting a set of points on a curve that optimally characterize the curve. If sequences of all postures of an action are sampled regularly in time and clustered using some measure of similarity, cluster centers will appear at postures with minimal motion. For walking this will occur at the instances when both feet are touching the ground. For a reaching movement they will typically occur at the initial and at the final posture. When imaged by a camera, this sequence of postures will project to a sequence of frames. We will denote these frames as key-frames for the specific action. The idea of using key frames for action recognition and tracking has appeared recently in various works, [11, 2] and it replaces traditional approaches based on tracking dynamics. The main argument which is expressed in [11] and [2] is that tracking has to be based on recognition, not the other way around. Note that there is still room for the idea of capturing action specific dynamics and using it as a corresponding spatio-temporal key frame for action recognition. This was done in [13] and used for unsupervised categorization of human motions. Related to our work is also the work in [7] which performs unsupervised segmentation of long video sequences based on viewed location, and [5] which uses spectral methods to cluster shots in home videos.

In this paper we present a method for automatically extracting key frames from a video sequence. We firstly devise a means of quantifying the similarity between every pair of frames in the sequence. Using this similarity measure the frames are clustered into similar groups, and the most central frame in each group is identified as the key frame. Section 2 discusses the similarity measure computed to compare every pair of frames. Section 3 describes how

the frames are clustered into groups of similar frames, and key frames extracted. Section 4 presents the results of the algorithm applied to a 30 second video of a tennis player, and Section 5 concludes with a summary of the key findings and a discussion of further work.

2 Comparing Frames

We are interested in extracting key frames that capture the body poses of a subject throughout a sequence, independent of the background or other elements of the scene.

The approximate location of the subject is isolated in each frame using a conventional tracker. Edge data is then extracted, and shape matching applied to quantify the distance between each pair of frames in shape space.

2.1 Localising the Subject

A conventional tracker is used to track the approximate locations of the head and body regions of the person, as described in [2]. The head and torso are modelled as quadrangles. A standard particle filter is used for the tracking [1], with the likelihood function based upon a sum-of-squares distance measure between a colour template and the image data.

2.2 Shape Matching

Shape matching is used to determine a correspondence field between edge points in each pair of frames in the sequence. From this correspondence field the shape deformation can be quantified.

Shape matching was performed on the portion of the image identified by the conventional tracker as containing to the upper body of the player. The procedure is described in detail in [2]. In summary, the procedure involves extracting shape context information for a set of edge points in each frame, then comparing this information between frames to determine the correspondence field linking the edge points in the different frames. The shape context information is determined by taking all combinations of four edge points (and associated tangent lines) and computing a topology-based shape context index for each combination, giving a histogram of shape contexts for every edge point. When comparing two images the likelihood of an edge point in one image matching one in the another image is estimated based on the commonality of the shape context histograms at both points.

2.3 Distance Matrix

The shape matching algorithm allows the similarity between the shapes in two images to be quantified by a single number. This can be considered as the distance between the two frames in some higher dimensional shape space. A distance matrix $\mathbf{M}$ is constructed quantifying these distances between every pair of frames in a sequence, such that

$$\mathbf{M}(i,j) = \text{distance from frame } i \text{ to frame } j \qquad (1)$$

$\mathbf{M}$ will be symmetric and $N \times N$ for an N frame sequence, and the main diagonal of $\mathbf{M}$ is zero since frames have zero distance from themselves.

Figure 1 shows $\mathbf{M}$ for a 900 frame (30 second) tennis sequence, together with several example frames and their corresponding rows and columns in the matrix.

The dark rectangular regions in the matrix correspond to periods where there is little change between frames. For the tennis sequence this equates to the player standing still in between strokes, such as in frames 324 and 931 in Figure 1. Dark diagonals (off the main diagonal) correspond to distinct repeated events, such as the forehand (407, 729) and backhand (187, 616) illustrated frames in Figure 1. Note that there is only one such dark diagonal in the rows and columns corresponding to frames 187 and 616. This is because there are only two backhands in the sequence, so there is only one repeated event. The light bands in the matrix correspond to unusual frames that do not match well with the majority of other frames.

3 Clustering Similar Frames

Knowing the distances separating all frames in the sequence we can precede to segment the sequence into clusters of similar frames. A number of distance-based clustering methods exist that can be applied to segment the sequence, for example [6, 8, 10], and our approach is by no means limited to a single clustering technique.

For the implementation reported here we apply an iterative clustering process that clusters frames together in a hierarchical manner, and is based on the method of [10]. The process is illustrated in Figure 2, and involves the following steps:

1. Compute distance matrix: Initially this is $\mathbf{M}$ (given by Equation 1), but at each subsequent iteration a new distance matrix is computed measuring the distances between the clusters computed in the previous iteration. The distance between two clusters is defined as the average distance from a frame in one cluster to a frame in the other cluster.

2. Clustering: An extension of the normalised cut [10] segmentation scheme is used to group elements into similar clusters (see Section 3.1).

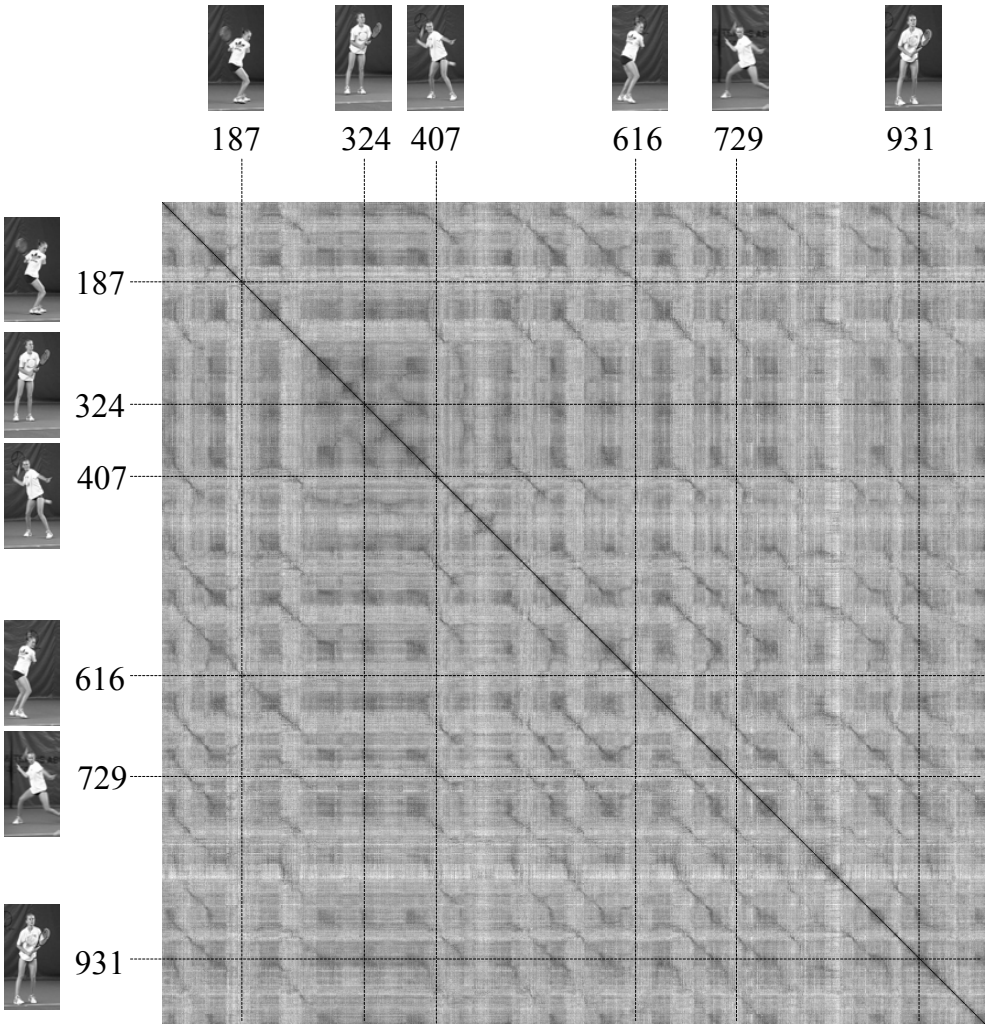


Figure 1: The distance matrix $\mathbf{M}$ for a 900 frame (30 second) tennis sequence, with several sample frames (frames are labelled from 101 to 900). Short dark diagonals correspond to forehands and backhands, and dark rectangular regions indicate periods where the player is standing still.

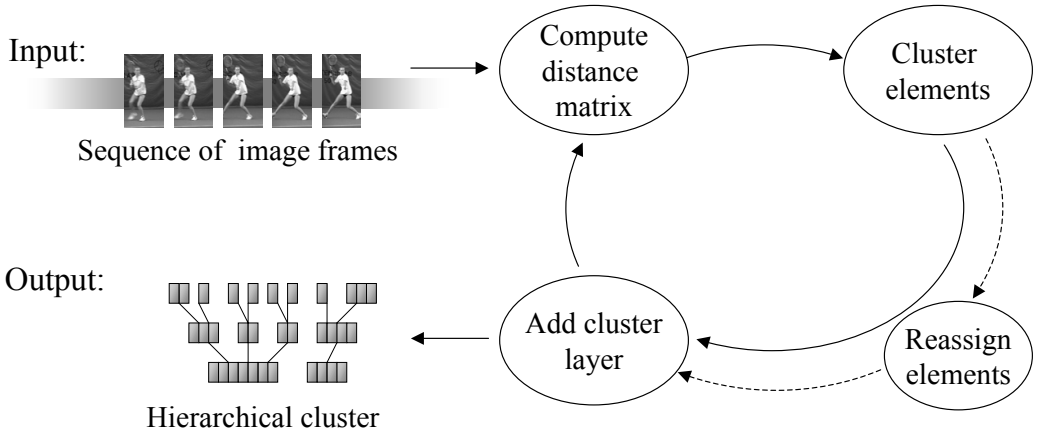


Figure 2: Clustering process.

3. Reassign elements: This step is only performed on the first iteration. First-level clusters are refined by reassigning elements among the clusters according to their average distances from each cluster (see Section 3.2).

4. Add cluster layer: The resulting clustering of the elements gives another layer in the output cluster tree.

In the first iteration of the loop individual frames are grouped into many small clusters called first level clusters. In the next iteration, these first level clusters are grouped together to form second level clusters (clusters of clusters), and the algorithm continues until the desired granularity of clustering is achieved. Typically only a small number of iterations are required.

The appeal of this architecture is its ability to establish small low-level clusters with little internal variation and thus significantly less chance of containing incorrectly clustered frames. These low-level clusters are then used as building blocks to construct larger clusters. Once the first-level clusters have been established the effect of noisy image data effecting isolated frames is reduced since subsequent clustering is done with groups rather than individual frames. In our experiments we observed that building clusters in this way facilitated the construction of larger and more accurate clusters than could be obtained by directly clustering into a small number of large clusters.

3.1 Minimizing the Normalised Cut

Clustering elements based on inter-element distances can be reposed as a graph partitioning problem. Each frame in the image sequence is treated as a node in a graph with links to all other frames (nodes). The strength of these links indicate how similar the frames are to each other, and are determined from the distance matrix $\mathbf{M}$ as

$$\mathbf{W}(i,j) = e^{\frac{-\mathbf{M}(i,j)^2}{2\sigma^2}}$$

where σ is a free variable (for our experiments σ was chosen as twice the standard deviation of the elements of $\mathbf{M}$). Segmentation is achieved by cutting edges in the graph, and dividing it into two disjoint sets. The question is: where should the graph be partitioned?

The normalised cut was introduced by Shi and Malik [9] as a partitioning criteria for subdividing graphs. Minimizing the normalised cut minimizes the affinities between different partitions, whilst maximising the affinities within each partition. Unfortunately finding the partition with the minimum normalized cut is an NP-complete problem. However, Shi and Malik formulated an efficient approach for finding a partition with a good normalised cut by applying results from spectral graph theory, that is, using the eigenvectors of the Laplacian to partition a graph. Both Shi and Malik [10] and subsequent authors [12, 4, 3] have compared the normalised cut method with various spectral clustering schemes, and several variations and improvements have been proposed. However, whilst these spectral approaches have shown impressive results they have moved away from addressing the initial goal of minimizing the cut criteria.

We use the spectral-based approach of Shi and Malik to determine an initial guess of where to partition the set (we empirically observed that this method gave an initial partitioning with a lower normalised cut than other spectral methods). The normalised cut criteria is then used to iteratatively improve this partition until a local minimum is reached.

The normalised cut is defined as follows: if V, the set of all frames in the sequence, is to be divided into two sets A and B then the normalised cut between set A and B is defined as

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

where the $cut(A,B) = \sum_{u\in A, v\in B} \mathbf{W}(u,v)$ is a measure of the total number of links severed to divide A and B, and $assoc(A,V) = \sum_{u\in A, t\in V} \mathbf{W}(u,t)$ is the total connection of the members of A to other elements.

Shi and Malik [10] show that the segmentation that minimizes the normalised cut is given by the solution to the discrete problem

$$\min_{\mathbf{y}} \frac{\mathbf{y}^\top(\mathbf{D}-\mathbf{W})\mathbf{y}}{\mathbf{y}\mathbf{D}\mathbf{y}}$$

subject to the constraints $\mathbf{y}_i \in \{1, -b\}$ and $\mathbf{y}^\top\mathbf{D}\mathbf{1} = 0$ (where b is a scalar and $\mathbf{1}$ is a vector of ones). Noting that this discrete optimisation is NP-complete they propose using the solution to the *continuous* problem version of this problem. They show that the solution to the continuous problem is given by the generalized eigenvector associated with the second smallest eigenvalue of the generalized eigensystem

$$(\mathbf{D}-\mathbf{W})\mathbf{y}_i = \lambda_i \mathbf{D}\mathbf{y}_i$$

Since the eigenvector's values are continuous (as they were not constrained by $\mathbf{y}_i \in \{1, -b\}$) a threshold is applied to generate the discrete values required for segmentation. It is straight forward to test every possible threshold point by using the magnitude of the eigenvector's elements to define an ordering. For N elements there are $N-1$ possible points where the ordering can be cut. By evaluating the normalised cut value for each of these partition points we are able to choose the optimum point at which to cut the ordered set.

As noted by Shi and Malik there is no guarantee that the solution to the continuous problem bares any resemblance to the discrete solution. Nonetheless, empirical results have

shown that the eigenvector does in general provide a good segmentation.

However, surprisingly, the solution provided by thresholding the eigenvector is in general *not* at a local minimum of the normalised cut. That is, by moving a single element from one side of the partition to the other it is possible to further decrease the $Ncut$ value of the partition. Thus the partition can be improved by the following procedure

1. Take each element in turn
   - compute the $Ncut$ values with the element on either side of the partition.
   - If necessary adjust the partition so the element lies on the side that minimizes the $Ncut$.
2. Repeat until all elements can be shifted across the partition (one at a time) without further reducing the $Ncut$.

This method is used to drive the $Ncut$ to a local minimum and increase the quality of the resulting partition.

3.2 Refining First-level Clusters

The first-level clusters of image frames obtained by minimizing the normalised cut typically contain a small number of badly clustered frames. This can be attributed to noisy image data leading to noisy distance measures and poorly separated data. To overcome this a filtered difference matrix is generated by blurring the distance matrix $\mathbf{M}$ via convolution with a 3×3 averaging kernel. This reduces high frequency noise and enforces the knowledge that since the image sequence is recorded at 25 Hz each frame must be quite similar to the frames adjacent to it in time.

Using this filtered difference matrix the clusters are refined by reassigning frames between clusters according to their average (filtered) distances from each cluster. The order in which frames are considered is important, since the clusters will change each time a frame is reassigned. With this in mind the frames more likely to be reassigned are considered first. The ordering is determined by finding the average distance from each frame to other members of its cluster, and comparing this to the mean average distance between all members of that cluster. The more standard deviations above the mean a frame lies the earlier it is considered.

This reassignment process is iterative and is repeated until a steady state is reached:

1. Determine ordering in which to consider frames.
2. For each frame:
   - Remove frame from its assigned cluster (if it is the only member of that cluster then delete the cluster) and compute the average distance between that frame and the frames in each of the clusters.
   - Reassign the frame to the cluster that has the minimum average distance to that frame.

This distance-based refinement of clusters was only necessary for first-level clusters and was not preformed for subsequent clusters in the heirarchy.

3.3 Keyframe Selection

Once the desired cluster granularity has been reached one key frame is selected to represent each cluster. The key frame is chosen as the most central frame in the cluster: for every frame in the cluster the average distance is computed to every other frame in the cluster, and the frame with the minimum average within cluster distance is the key frame.

In our experiments we used second-level clusters to define the key frames. Figure 3 shows several second-level clusters and the resulting key frames.

4 Results

Our algorithm was applied to extract key frames from a 30 second sequence of a woman playing tennis. During the sequence the woman moves about the baseline and plays several forehand and backhand strokes. The sequence was initially clustered into 100 first-level clusters and these were combined into 20 second-level clusters, and 6 third-level clusters. The key frames were extracted from the second-level clusters, giving 20 key frames for the sequence.

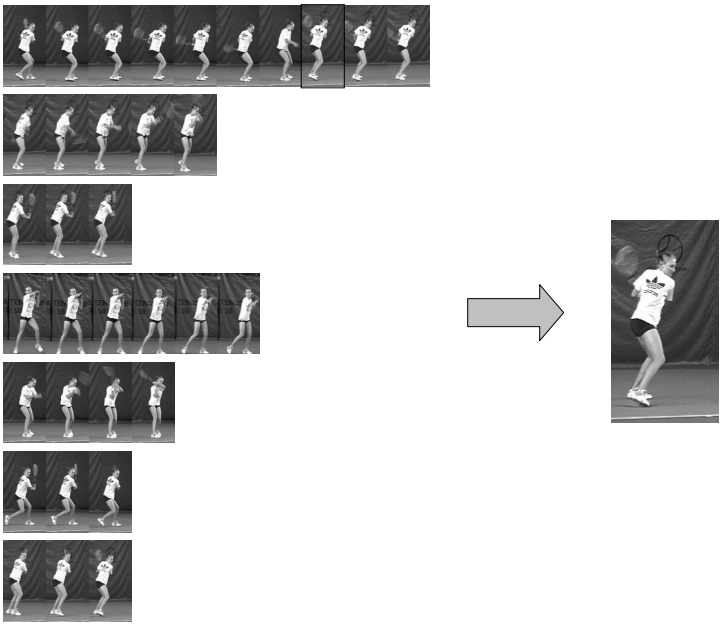
Figure 3 shows several second-level clusters and the resulting key frames. Each row of these clusters is a first-level cluster.

By representing each second-level cluster by its key frame we can examine the occurrence of different body poses throughout the sequence. Figure 4 shows all the key frames extracted from the sequence and Figure 5 shows the occurrence of the key frame's clusters throughout the sequence. The third-level clusters, which were too coarse to provide useful key frames were used to group the key frames into the clusters indicated by the grey shading in Figure 5. As noted on the righthand side of the figure, two of these clusters correspond to forehand key frames, one to backhand key frames, and the other three to the various stances adopted by the player in between shots.

The forehands and backhands are easily identified respectively by the strong peaks and troughs in the graph in Figure 5. However, there are several narrow troughs in the graph that do not correspond to backhands. These are

(a)

(b)



(c)

Figure 3: Three (of the 20) second-level clusters and the extracted key frames. Each row of the second-level clusters is a first-level cluster. The most central frame of each second-level cluster is indicated, these are taken as the key frames.

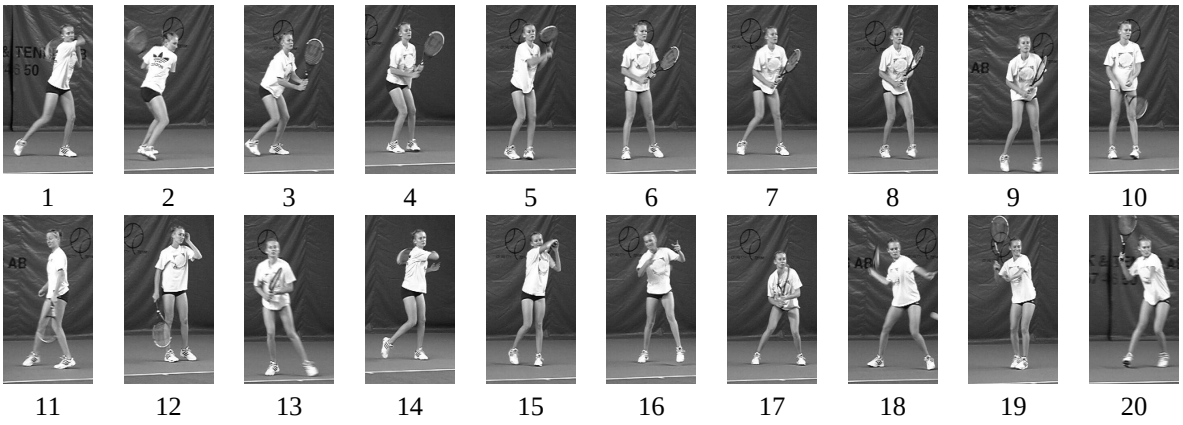


Figure 4: 20 key frames extracted from the 30 second sequence.

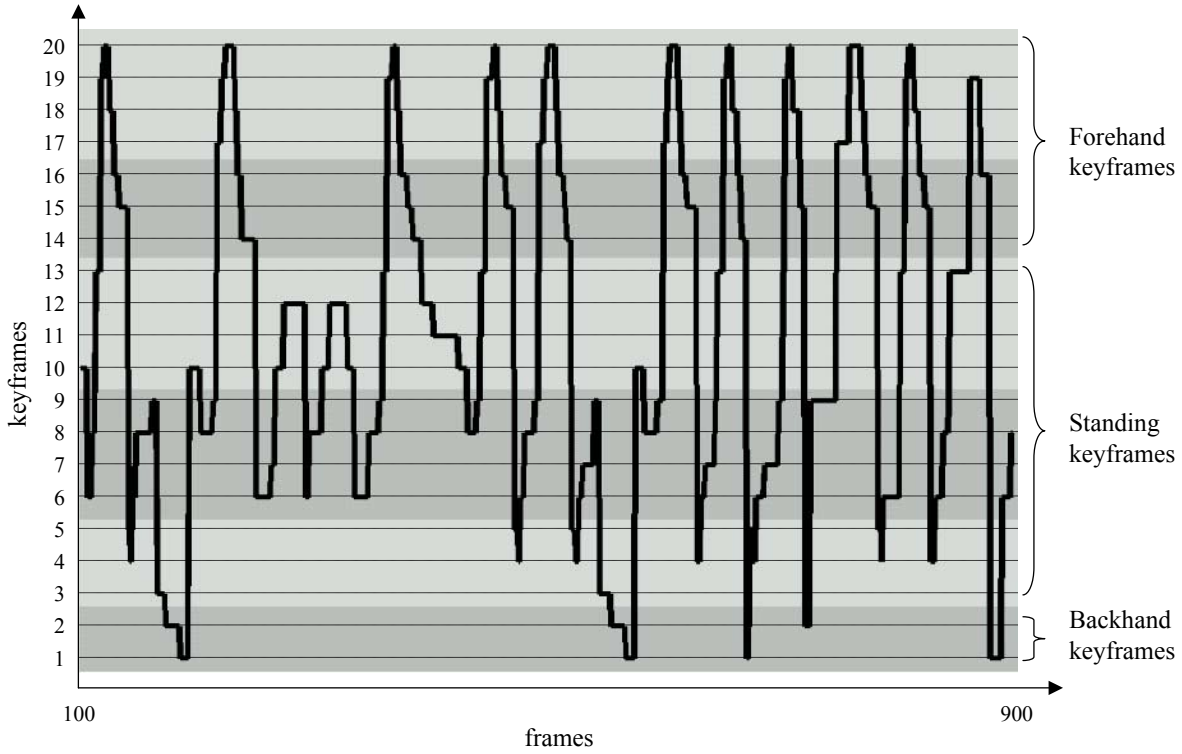


Figure 5: Occurrence of key frame clusters throughout sequence, the shading in the graph indicate the third-level clusters.

caused by follow-throughs from forehands which put the player in a similar pose to that immediately preceding a backhand stroke. The forth row of Figure 3(c) shows an example of this, where a first-level cluster corresponding to a forehand follow-through is grouped with backhand clusters. This is not a problem since the key frames still correspond to similar body poses.

Selecting the appropriate cluster granularity is very subjective and depends primarily on what the key frames are to be used for. In general the more key frames the better, however, the maximum number of key frames is typically limited by the application (if there is no limit then every frame becomes a key frame). For a given application appropriate constraints can be imposed on key frame variability and how closely each image frame must be to the nearest key frame, with such information the appropriate cluster granularity could be automatically determined.

5 Summary and Conclusions

This paper has presented a method for extracting key frames from a video sequence. A method was outlined for quantifying the similarity between every pair of frames in the sequence using shape matching. This similarity measure was then used to cluster the frames into similar groups, from which key frames could be extracted. A clustering hierarchy was used together with an extension of the normalised cut segmentation technique to obtain a small number of tightly packed clusters of image frames. The algorithm was applied to a 30 second tennis video and successfully extracted 20 key frames describing the sequence.

Further work will focus on improving the inter-frame similarity measure using additional matching techniques. Consideration is also being given to temporal information. The distinctive patterns of backhands and forehands in Figure 3 suggest that temporal information would be beneficial to the clustering process. For instance if one cluster always occurs immediately after another in the sequence it is likely the two are related and should possibly be grouped together in the next cluster layer.

References

[1] A. Blake and M. Isard. Active contours: The application of techniques from graphics, vision, control theory and statistics to visual tracking of shapes in motion. 1999.

[2] S. Carlsson and J. Sullivan. Action recognition by shape matching to key frames. In *Workshop on Models versus Exemplars in Computer Vision, Kauai, Hawaii, USA*, December 2001.

[3] Sepandar D. Kamvar, Dan Klein, and Christopher D. Manning. Spectral Learning. In *To appear in Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence IJCAI*, 2003.

[4] A. Ng, M. Jordan, and Y. Weiss. *On spectral clustering: Analysis and an algorithm*. MIT Press, Cambridge, MA, 2002.

[5] J-M. Odobez, D. Gatica-Perez, and M. Guillemot. On spectral methods and the structuring of home videos. IDIAP-RR 55, IDIAP, Martigny, Switzerland, 2002.

[6] Rmer Rosales and Brendan Frey. Learning generative models of similarity matrices. In *To appear in Proc. 19th Conference on Uncertainty in Artificial Intelligence (UAI)*, August 2003.

[7] F. Schaffalitzky and A. Zisserman. Automated scene matching in movies. In *Proceedings of the Challenge of Image and Video Retrieval, London*, LNCS 2383, pages 186–197. Springer-Verlag, 2002.

[8] E. Sharon, A. Brandt, and R. Basri. Fast multiscale image segmentation. In *Proc IEEE Conference on Computer Vision and Pattern Recognition*, volume I, pages 70–77, 2000.

[9] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. In *Proc IEEE Conference on Computer Vision and Pattern Recognition*, pages 731–737, 1997.

[10] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[11] Kentaro Toyama and Andrew Blake. Probabilistic tracking in a metric space. In *Proc International Conference on Computer Vision (ICCV)*, pages 50–59, 2001.

[12] Yair Weiss. Segmenting using eigenvectors: a unifying view. In *Seventh International Conference on Computer Vision*, volume 2, pages 975–982, September 20 - 25 1999.

[13] L. Zelnik-Manor and M. Irani. Event-based video analysis. In *Proceedingas of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, December 2001.