To appear in 6th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS'09), Genoa, Italy, Sept. 2-4, 2009

# Clustered Synopsis of Surveillance Video

Yael Pritch Sarit Ratovitch Avishai Hendel Shmuel Peleg
School of Computer Science and Engineering
The Hebrew University of Jerusalem
91904 Jerusalem, Israel

***Abstract*—Millions of surveillance cameras record video around the clock, producing huge video archives. Even when a video archive is known to include critical activities, finding them is like finding a needle in a haystack, making the archive almost worthless. Two main approaches were proposed to address this problem: action recognition and video summarization. Methods for automatic detection of activities still face problems in many scenarios. The video synopsis approach to video summarization is very effective, but may produce confusing summaries by the simultaneous display of multiple activities.**

**A new methodology for the generation of short and coherent video summaries is presented, based on clustering of similar activities. Objects with similar activities are easy to watch simultaneously, and outliers can be spotted instantly. Clustered synopsis is also suitable for efficient creation of ground truth data.**

## I. Introduction

Most surveillance video is recorded, but normally no one views these recordings. Recorded video is too long to watch, and automated video analysis approaches [28], [2] are still far from giving satisfying solutions. Many cases, where a human observer could have made fast and accurate decisions, are still too difficult even for the best video analytics systems.

Summarization methods enable more efficient browsing in video [21], [14], [6], [16], [9], [22], [4], [25], but create summaries that are either too long or are confusing. Most methods generate a static description as a set of keyframes [29]. Other methods use adaptive fast forward [15], [1], skipping irrelevant periods. In this paper we build upon video synopsis [18], [17], [19], [8], [4], displaying simultaneously activities which originate from different times periods. Video synopsis methods create very efficient summaries, but the summaries may be confusing when mixing together different activities.

The goal of this paper is to improve the browsing efficiency in archives of surveillance video, and make it both faster and more accurate. We propose to achieve this goal by improving video synopsis with prior clustering of activities, and displaying together only similar activities. With this approach we add three benefits for video synopsis. (i) Similar activities pack together more efficiently into shorter video summaries. (ii) These summaries are very clear, making browsing very efficient, as it is easy to view multiple similar activities. (iii) Irregular activities are easier to detect. In addition to simple video summarization, clustered summaries can help in structured browsing of objects, and in preparing ground truth data for training classifiers. The accuracy of classifiers can be checked as well on thousands of objects.

In this paper we will cover three main topics: (i) The definition of distance between activities. (ii) Clustering activities into similar clusters. (iii) Efficient presentation of video summaries using the obtained clusters.

This research was supported by the Israel Ministry of Science and by the Alberto Moscona Foundation. Video examples can be found in www.vision.huji.ac.il/synopsis/ .

## II. Activities

The basic element in our work in an activity, which is simply a dynamic object [27]. The object is detected in a sequence of frames, and each activity is represented by a sequence of object masks in those frames. In addition to the object mask in each frame, an object has a rectangular bounding box called the ROI (Region of Interest). The information for each activity $A_i$ includes the following:

$$A_i = \left(t_s^i, t_e^i, \{M_t^i, R_t^i\}_{t_s \leq t \leq t_e}\right), \tag{1}$$

where $t_s$ and $t_e$ are the start and end frames for this activity, $M_t$ is the object mask for frame $t$ which includes pixel colors, and $R_t$ is the ROI for frame $t$.

Any method that can detect activities as in Eq. (1) is suitable for clustered synopsis. We do not elaborate on this topic as there are numerous methods that segment moving objects fairly well. We use a simplification of [23] to compute activities. This method combines background subtraction together with min-cut to get segmentation of moving objects. But other methods for the detection of moving objects are adequate as well.

### A. Tubelets: Short Activity Segments

In order to enable the analysis of objects performing multiple activities, activities are broken into temporal subparts called ”tubelets”. Tubelets have a predefined maximal length (we use 50 frames), and can overlap with other tubelets (we use 50% overlap between tubelts). The division into tubelets has the following benefits:

- Activities vary substantially in length. By breaking into tubelets we compare activities of similar lengths.
- Long activities may be composed from parts having different dynamics. Tubelets are more likely to have a single, simple, motion.
- Different objects may intersect in the video frames, creating complex activities composed from different objects. Most tubelets include a single object since they are shorter.

After clustering the tubelets, overlapping tubelets that were clustered together are merged into a longer activity.

### B. Activity Features

The features used for clustering are appearance (image) features and motion features. A comparative study of several image descriptors is presented in [12], and we used the SIFT descriptors [10] as appearance features. For each object, multiple SIFT features are computed inside the object masks in the relevant frames. This large collection of SIFT features will be used to estimate appearance similarity between objects. For efficiency, we randomly select a predetermined number of features for the initial unsupervised clustering. In the examples shown in this paper we select 200 SIFT features from each activity.

For representing the motion of objects, we use the smooth trajectory of the center of the object. The trajectory of an object (activity) $A_i$ is a sequence of frame by frame features, including for every frame $t$ three features: $(x_t^i, y_t^i, r_t^i)$ which represent the $x, y$ coordinates of the centroid of the object, as well as the radius of the object. Shorter motion descriptors can be used by sampling fewer frames from the activity. Motion trajectories were clustered before for behavior analysis and possible video compression [11], and we now propose to use such clusters for video summarization.

## III. Similarity Between Activities

In order to cluster together similar activities, a distance metric between activities is needed. A symmetric distance between activities is needed for use in spectral clustering, that will be used in Sec. III-C. In our experiments we used a distance based on two components, as described in this section: (i) Appearance features that are derived from the shape of the objects (Eq. 2), and (ii) motion features that are derived from the motion of the objects (Eq. 6).

### A. Appearance Distance

For the appearance distance between two activities we use the NN (Nearest Neighbor) estimate computed from the distance between their SIFT descriptors. As a distance between SIFT descriptors we use a simple squared distance, but other distances such as the distance proposed in [10] can be used as well. Let $S_k^i$ be the $k$'s SIFT descriptor of activity $A_i$, and let $\tilde{S}_k^j$ be the SIFT descriptor in $A_j$ closest to $S_k^i$. Similarly, $\tilde{S}_k^i$ is the closest descriptor in $A_i$ to $S_k^j$.

The appearance distance $Sd_{ij}$ between activities $A_i$ and $A_j$ is

$$Sd_{ij} = \frac{1}{2N}\left(\sum_k |S_k^i - \tilde{S}_k^j| + \sum_k |S_k^j - \tilde{S}_k^i|\right), \quad (2)$$

where $N$ is the number of SIFT descriptors in an activity. This measure uses the nearest neighbor distance [3], which we found to be very effective in our experiments.

### B. Motion Distance

Motion similarity between two activities is especially useful for the construction of summaries that display simultaneously multiple objects. Given two activities $A_i$ and $A_j$, we compute a motion distance between them for all temporal shifts $k$ of $A_j$. Let $l_x$ be the time length of activity $A_x$, let $T_{ij}(k)$ be the time period common to $A_i$ and to $A_j$ after the latter has been temporally shifted by $k$, and let

$$w(k) = \frac{\min(l_i, l_j)}{T_{ij}(k)} \quad (3)$$

be a weight encouraging a long temporal overlap between temporally shifted activities.

The separation between the activities is:

$$Sep_{ij}(k) = \sum_{t \in T_{ij}(k)} \left[(x_t^i - x_{t+k}^j)^2 + (y_t^i - y_{t+k}^j)^2\right] \quad (4)$$

The motion distance between $A_i$ and the shifted $A_j$ is defined as follows:

$$Md_{ij}(k) = \frac{w(k)}{T_{ij}(k)} Sep_{ij}(k) \quad (5)$$

The elements in the motion distance $Md_{ij}(k)$ minimize the spatial separation between the activities (4), and increase the temporal overlap between the activities as represented by $w(k)$ (Eq. 3). Dividing by the temporal overlap $T_{ij}(k)$ is a normalization to a "per frame" measure.

When the motion distance between two activities should not depend on the object location in the image, the two centroids are computed for the respective activities in $T_{ij}(k)$, the time period common to the two activities. The two objects are spatially shifted to a common centroid before computing $Md_{ij}(k)$ (Eq. 5). The final motion distance between $A_i$ and $A_j$ ia a minimum over all temporal shifts $k$:

$$Md_{ij} = \min_k Md_{ij}(k) \quad (6)$$

Eq. 6 is minimized by the implementation in [13].

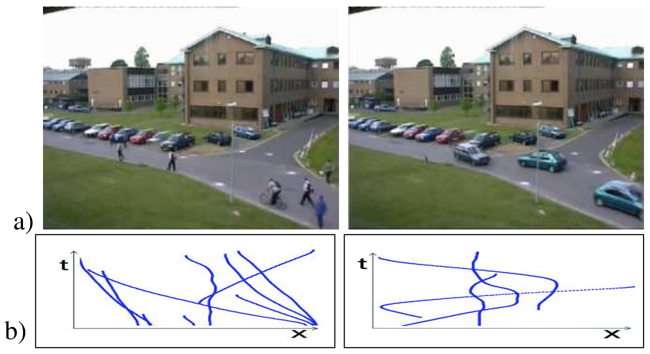


Figure 1. Unsupervised spectral clustering using appearance features for a video from the PETS database. The people and cars are separated well. (a) Frames from the two summaries (b) Motion paths of the objects in the displayed cluster, each object is shown as a curve in $x-\mathrm{t}$.

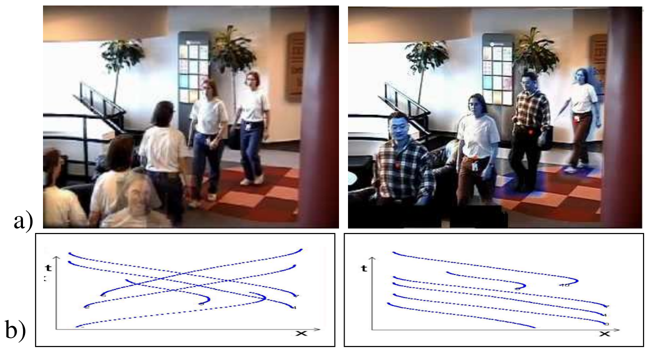


Figure 2. Unsupervised spectral clustering using appearance and motion. Only appearance features were used on left, and only motion features were used on right. (a) An image from a summary generated from one cluster. (b) Motion paths of the objects in the displayed cluster, each object is shown as a curve in $x-\mathrm{t}$. Sequence taken from [7].

## C. Unsupervised Clustering

Unsupervised clustering uses a distance measure $D_{ij}$ between activities $A_i$ and $A_j$ using the appearance distance $Sd_{ij}$ (Eq. 2) and the motion distance $Md_{ij}$ (Eq. 6).

$$D_{ij} = \alpha Sd_{ij} + (1-\alpha)Md_{ij}. \tag{7}$$

The parameter $\alpha$ balances between motion and appearance. From $D_{ij}$ an affinity matrix $M$ is generated:

$$M(i,j) = M(j,i) = \exp(-D_{ij}/\sigma), \tag{8}$$

where $\sigma$ is a constant used for normalization. The normalized-cut approach of [20], [24] is used to cluster the data given the affinity matrix $M$. We used doubly stochastic normalization of the affinity matrix to improve spectral clustering results as proposed by [26]. Examples showing the results of clustering are shown in Fig. 1 and Fig. 2.

Performing unsupervised clustering on one set of features can be followed by taking the resulting clusters, and on each cluster performing clustering using a different set of features. This is shown in Fig. 3, where two appearance clusters were first generated, and on each appearance cluster

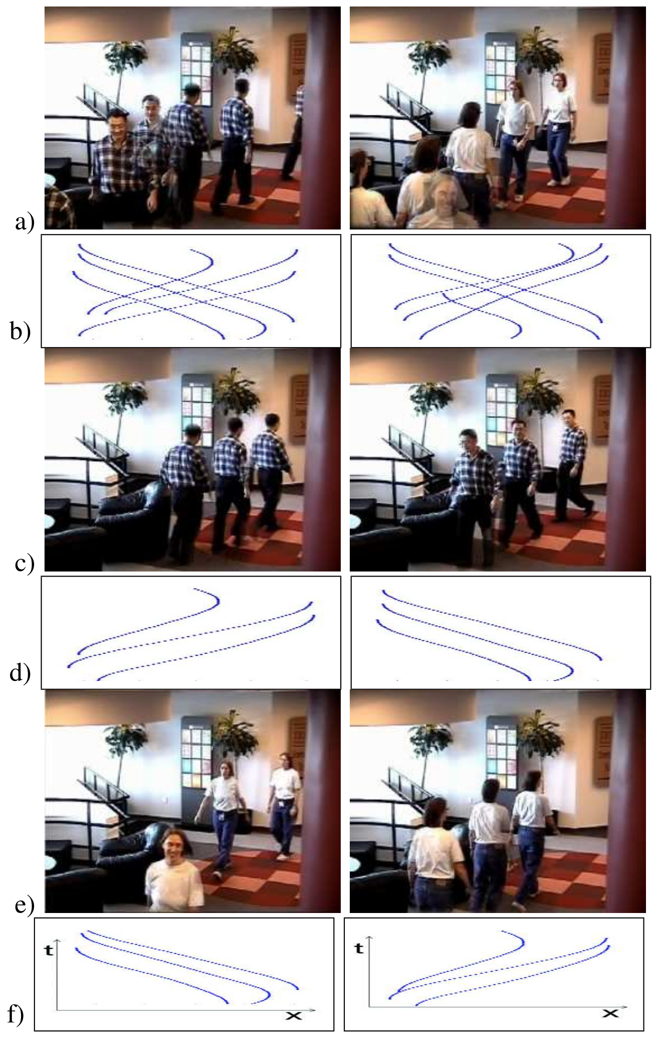


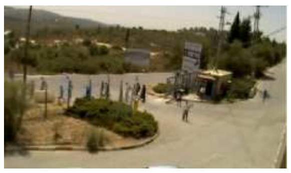
Figure 3. Performing two steps of unsupervised spectral clustering. (a) Two appearance-based clusters, with a good separation between the man and the woman. (b) The motion paths of the clusters in (a). Motion paths are shown as curves in $x-\mathrm{t}$. (c-d) Further clustering on the man cluster using motion features. Man walking left and man walking right are the two new clusters. (e-f) Further clustering on the woman cluster using motion features. Woman walking left and woman walking right are the two new clusters.

motion clustering has been applied. This resulted in four clusters, each having different appearance and motion.

## IV. Creating Summaries

Given a set of objects or activities, we would like to create a summarization video displaying these objects with minimal length and minimum collisions between them. This is done by assigning each object its start play time in the summary. This mapping from objects to play times is performed in three stages:

1) Objects are clustered based on the packing cost (Eq. 11) defined in Section IV-A.
2) Play time is assigned to objects within each cluster.

a) b) 

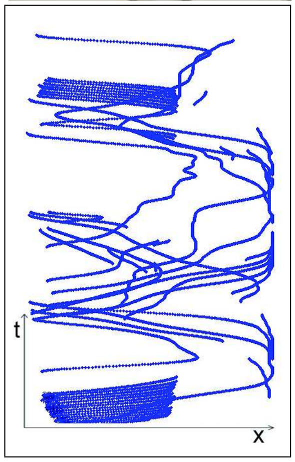
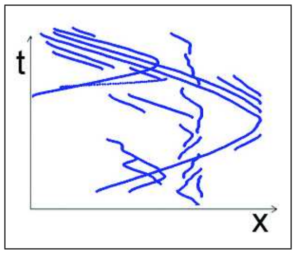
c)  d) 

Figure 4. Selecting similar objects using a nearest neighbor approach. (a) Objects found to be closest to two selected cars. (b) Objects found to be closest to two selected people. (c) Motion trajectories of the cars in the summary. (d) Motion trajectories of the people in the summary.

3) Play time is assigned to each cluster.

These steps will be described in detail in this section. Once each object is assigned its play time, the output summary can be generated by playing the objects over the background at the assigned times. All example in this paper were generated using the method described in this chapter. For example, the video used in Fig. 1 was originally 5 minutes long, and using clustered synopsis the summary including all activities was about 20 seconds long.

Another example for simple browsing of surveillance video is in Fig. 4. In viewing the video, the user prefers to watch only the people, or only the cars. The fastest approach is to select a few objects in the desired class, pick up appropriate similar objects using a nearest neighbor method, and display the objects in a video summary.

### A. *Packing Cost*

The packing cost between two activities should indicate how efficiently the activities could be played together. The activities should have similar motion, and for some temporal shift they should play simultaneously with minimal collisions and with minimal increase of the length of the video.

The packing cost is very similar to the motion distance in Sec. III-B, with the following modifications (i) There is no spatial shift of the activities. (ii) A collision cost $Col_{ij}(k)$ is added between objects, defined as follows:

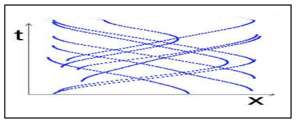
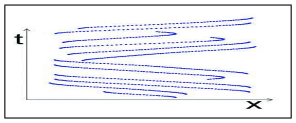
a)  b) 

Figure 5. (a) Motion trajectories of all input objects, shown as curves in $x-$t. (b) Motion trajectories in a clustered synopsis. Note that there are no confusing intersections.

$$Col_{ij}(k) = \sum_{t \in T_{ij}(k)} \left| (x_t^i - x_{t+k}^j)^2 + (y_t^i - y_{t+k}^j)^2 < (r_t^i)^2 + (r_{t+k}^j)^2 \right| \quad (9)$$

where $r_t^i$ is the radius of object $A_i$ in frame $t$, and $r_{t+k}^j$ is the radius of $A_j$ in frame $t+k$. $Col_{ij}(k)$ counts the number of collisions for the temporal shift $k$, where a collision occurs when the separation between the object centers is smaller than the sum of the radiuses of the two objects.

The packing cost for temporal shift $k$ is defined using the motion distance (5) and the collision cost (9):

$$Pk_{ij}(k) = \beta Md_{ij}(k) + (1-\beta)Col_{ij}(k). \quad (10)$$

And finally, the packing cost for the two activities is the minimum over all temporal shifts:

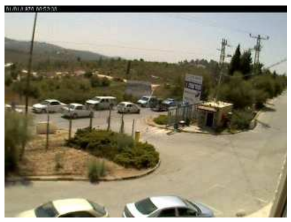
$$Pk_{ij} = \min_k Pk_{ij}(k) \quad (11)$$

The packing cost $Pk_{ij}$ between two objects is used for clustering before arrangement into the video summary. Fig. 5 is an example for the clustering into two clusters of a set of objets based on the packing cost.

### B. *Object Arrangement within Cluster*

Once the objects are clustered based on the packing cost of Eq. (11), each cluster contains objects that can be packed efficiently. In order to create a summary video from all objects in such a cluster, we need to determine the starting play times for all objets. These starting play times should generate a short but easy to watch video. Since all objects in a cluster already have a similar motion, we need to determine the play time to minimize both total playing time but also minimize collisions between objects. This is done using the packing cost as defined in (10). Since optimal packing is a difficult problem, we use the following optimization which gives good results.

We start with an empty set $G$ of arranged objects. Determining the mapping of each object into its play time starts with the longest temporal object, which is placed arbitrarily and added to $G$. We continue with the longest object outside $G$, and find the time mapping $k$ which minimizes the sum over all its frames of the packing costs $Pk_{ij}(k)$ between the current object and the closest object in $G$ in all frames.

a) b) 

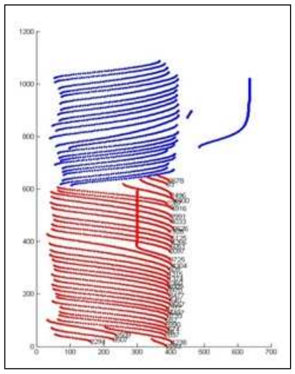
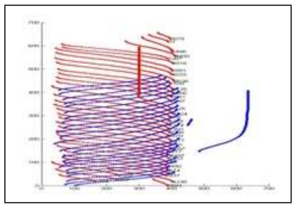
c)  d) 

Figure 6. Combining two clusters. (a,c) A sample frame and trajectories in a synopsis having a high weight to collision cost. (b,d) A shorter synopsis is obtained the collision cost has a lower weight.

In this computation, The temporal overlap $T_{ij}(k)$ is the temporal overlap with the set $G$. Every object is added to $G$ after its time mapping has been determined. The temporal arrangement continues until all objects are mapped into play time. Such temporal arrangements is shown in Fig. 5.

We compute the packing costs $Pk_{ij}(k)$, which involves computing the collisions of one object with the nearest object from a collections of objects, using the efficient approximate k-nearest neighbors algorithm and kd-tree implementation of [13]. The expected time for a NN search is logarithmic in the number of elements stored in the kd-tree.

### C. *Combining Different Clusters*

The combination of different clusters is done similarly to the combination of individual objects. While objects in a cluster have their relative playing time, we need to assign to each cluster a global playing time. This is done similar to assigning time to individual objects. An arbitrary playing time is assigned to the cluster having maximal number of objects. We continue by taking the largest cluster with unassigned playing time, and assign to it a global time minimizing collision with the clusters whose time has already been assigned. An example is shown in Fig 6.

## V. Training and Testing Supervised Classifiers

Training a supervised classifier, e.g. SVM [5], requires a large training set of tagged samples. Building such a large training set is especially time consuming for surveillance video, as there are thousand of objects to classify. Clustered summaries can make the building of the training set fast and efficient.

One possible approach for building the training set is the use of unsupervised clustering to create approximate clusters. Another approach can be the tagging of a single sample, and using a nearest neighbor approach to tag other samples. While these approached can create quickly large training sets, they have errors that need to be corrected. Clustered summaries can display in a very short time the created sets, allowing the creation of large and accurate training sets with minimal effort and time. Once a working classifier has been trained, a clustered summary is the most efficient way to test its performance. The alternative of spending many hours to watch the resulting classification is not practical.

The training set for the example in Fig. 7 has about 100 tubelets. Instead of tagging 100 tubelets individually, unsupervised clustering allowed the creation of the training set with only 10 key clicks following unsupervised clustering. Fig. 7 shows clustered summaries of the SVM classification of 100 tubelets using motion features. Simple view of the classification results, assuming 10 seconds for each tubelet, takes about 20 minutes, while the length of the clustered summary is less than 2 minutes.

In a typical experiment, 70% of objects were labeled correctly in the unsupervised clustering step, enough to give the user an indication of the type of activities in the scene. After deletion of 30% of the wrong labels in the training data, SVM reached correct classification for 93.6% of remaining objects.

## VI. Concluding Remarks

The clustered summaries methodology is proposed as an efficient method to browse and search surveillance video. Surveillance videos are very long (actually they are endless), and include many thousands of objects. Regular browsing is practically impossible. In clustered summaries, multiple objects having similar motion are shown simultaneously. This enables to view all objects in a much shorter time, without losing the ability to discriminate between different activities. Summaries of thousands of objects can be created in a few minutes (not counting object extraction time). In a small user study, 15 subjects were presented two summaries of the same time period, both containing the same objects: a normal summary and a clustered summary. Later, given an activity to search for, 13 out of the 15 chose to search in the clustered summary.

In addition to efficient viewing of all objects in the surveillance video, clustered summaries are important for creating examples for classifiers. Multiple examples can be prepared and given to the learning mechanisms very quickly using unsupervised clustering and clustered summaries. Even a simple nearest neighbor classifier can initially be used, cleaned up using clustered summaries, and the results given to learning classifiers.

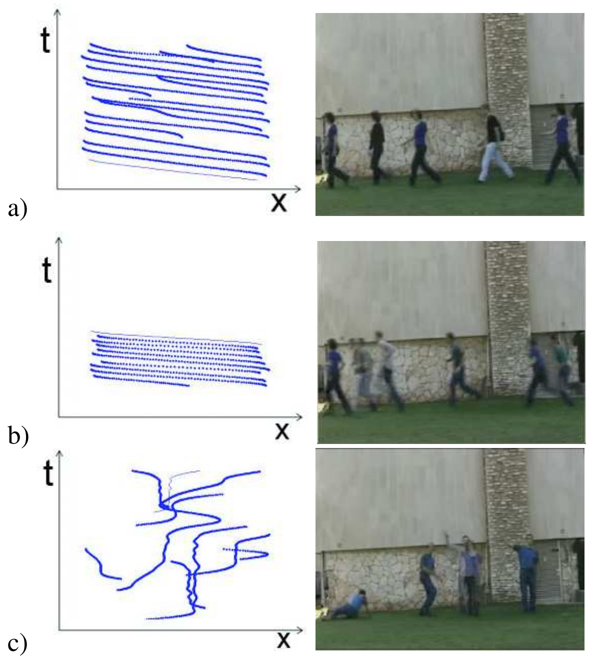


Figure 7. Clustered summaries of SVM classification. 100 tubelets were classified by SVM using motion features. Simple view of the classification, assuming 10 seconds for each tubelet, would take about 20 minutes. The length of the clustered synopsis is less than 2 minutes. The left column is the motion trajectory of the objects, and the right column is one frame from the clustered summary. The classes are as follows: (a) walking left. (b) running left. (c) standing and waving. Walking right and running right are not shown.

Clustered summaries can also be used for video browsing. Instead of spending many hours to watch the captured video, the clustered summaries methodology enables to browse the video archive quickly and efficiently, and focus on a smaller set of interesting objects. Browsing can be done by hierarchical application of clustered summaries. The user first selects an interesting cluster, and then zooms-in on this cluster to identify the interesting objects in it. Or the user can select irrelevant clusters and remove their objects from the summary. The user may continue browsing by ”cleaning” the cluster using a supervised classifier, or by simply selecting some nearest neighbors.

## REFERENCES

[1] E. Bennett and L. McMillan. Computational time-lapse video. In *SIGGRAPH'07*, 2007.

[2] O. Boiman and M. Irani. Detecting irregularities in images and in video. In *ICCV*, pages I: 462–469, Beijing, 2005.

[3] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *CVPR*, Anchorage, Alaska, 2008.

[4] B. Chen and P. Sen. Video carving. In *EUROGRAPHICS'08*, 2008.

[5] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20, 1995.

[6] A. Ferman and A. M. Tekalp. Multiscale content extraction and representation for video indexing. *Proc. of SPIE*, 3229:23–31, 1997.

[7] IBM Research. Performance evaluation of surveillance systems. In *http://www.research.ibm.com/peoplevision/performanceevaluation.html*.

[8] H. Kang, Y. Matsushita, X. Tang, and X. Chen. Space-time video montage. In *CVPR'06*, pages 1331–1338, New-York, 2006.

[9] C. Kim and J. Hwang. An integrated scheme for object-based video abstraction. In *ACM Multimedia*, pages 303–311, New York, 2000.

[10] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 30(2):91–110, 2004.

[11] D. Makris and T. Ellis. Path detection in video surveillance. *Image and Vision Computing*, 20(12):895–903, 2002.

[12] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Trans. PAMI*, 27(10):1615–1630, 2005.

[13] D. Mount and S. Arya. Ann: A library for approximate nearest neighbor searching. In *University of Maryland*, 1997.

[14] J. Oh, Q. Wen, J. lee, and S. Hwang. Video abstraction. In S. Deb, editor, *Video Data Mangement and Information Retrieval*, pages 321–346. Idea Group Inc. and IRM Press, 2004.

[15] N. Petrovic, N. Jojic, and T. Huang. Adaptive video fast forward. *Multimedia Tools and Applications*, 26(3):327–344, August 2005.

[16] A. Pope, R. Kumar, H. Sawhney, and C. Wan. Video abstraction: Summarizing video content for retrieval and visualization. In *Signals, Systems and Computers*, pages 915–919, 1998.

[17] Y. Pritch, A. Rav-Acha, A. Gutman, and S. Peleg. Webcam synopsis: Peeking around the world. In *ICCV'07*, Rio de Janiero, 2007.

[18] Y. Pritch, A. Rav-Acha, and S. Peleg. Nonchronological video synopsis and indexing. *IEEE Trans. PAMI*, 30(11):1971–1984, 2008.

[19] A. Rav-Acha, Y. Pritch, and S. Peleg. Making a long video short: Dynamic video synopsis. In *CVPR'06*, volume 1, pages 435–441, 2006.

[20] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. PAMI*, 22:888–905, 2000.

[21] D. Simakv, Y. Caspi, E. Shechtman, and M. Irani. Summarizing visual data using bidirectional similarity. In *CVPR'08*, Ancorage, 2008.

[22] A. Stefanidis, P. Partsinevelos, P. Agouris, and P. Doucette. Summarizing video datasets in the spatiotemporal domain. In *DEXA Workshop*, pages 906–912, 2000.

[23] J. Sun, W. Zhang, X. Tang, and H. Shum. Background cut. In *ECCV'06*, pages 628–641, 2006.

[24] Y. Weiss. Segmentation using eigenvectors: a unifying view. In *ICCV'99*, pages 975–982, 1999.

[25] L. Wolf, M. Guttmann, and D. Cohen-Or. Non-homogeneous content-driven video-retargetings. In *ICCV'07*, Rio de Janiero, 2007.

[26] R. Zass and A. Shashua. A unifying approach to hard and probabilistic clustering. In *ICCV'05*, volume 1, pages 294–301, 2005.

[27] L. Zelnik-Manor and M. Irani. Event-based analysis of video. In *CVPR'01*, pages 123–130, 2001.

[28] H. Zhong, J. Shi, and M. Visontai. Detecting unusual activity in video. In *CVPR'04*, pages 819–826, 2004.

[29] X. Zhu, X. Wu, J. Fan, A. K. Elmagarmid, and W. G. Aref. Exploring video content structure for hierarchical summarization. *Multimedia Syst.*, 10(2):98–115, 2004.