Structuring Continuous Video Recordings of Everyday Life Using Time-Constrained Clustering

Wei-Hao Lin and Alexander Hauptmann

Language Technologies Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213 U.S.A.

ABSTRACT

As personal wearable devices become more powerful and ubiquitous, soon everyone will be capable to continuously record video of everyday life. The archive of continuous recordings need to be segmented into manageable units so that they can be efficiently browsed and indexed by any video retrieval systems. Many researchers approach the problem in two-pass methods: segmenting the continuous recordings into chunks, followed by clustering chunks. In this paper we propose a novel one-pass algorithm to accomplish both tasks at the same time by imposing time constraints on the K-Means clustering algorithm. We evaluate the proposed algorithm on 62.5 hours of continuous recordings, and the experiment results show that time-constrained clustering algorithm substantially outperforms the unconstrained version.

Keywords: video segmentation, video clustering, time-constrained clustering, continuously recorded video

1. INTRODUCTION

As personal wearable devices become more powerful and ubiquitous, capturing everyday life without losing any precious moments will soon become commonplace. Mobile phones nowadays can record short video, and in the foreseeable future they will be able to continuously record video for days, as already demonstrated by many research prototypes.[1–3] This is a step towards the Vannevar Bush's vision of Memex, "in which an individual stores all his books, records, and communications".[4] With the complete recording of one's life available, the personal video archive allows the owner to retrieve episodes of past events,[5–8] to automatically create biographies,[9] and to share memories with others.

In order to support efficient indexing, browsing, and searching provided by any video retrieval systems, continuous media steams need to be segmented into manageable processing units in order to be processed for any video retrieval systems. Each browsing unit must have a start and end point to begin playing video and to end the playback. As text retrieval systems define documents as the unit of retrieval, video retrieval for continuous recordings needs an equivalent. Unlike segmenting structured video (e.g. broadcast news and TV sitcoms[10]), continuously recorded video contains no pre-defined units, which makes segmentation and clustering continuous recordings nontrivial tasks.

Most researchers approach the problem of structuring continuously recorded personal memory archive in two passes,[11, 12] as illustrated in 1. Firstly, a continuous recording is segmented into chunks based on a pre-defined coherence criterion. In the second pass, a clustering algorithm is performed to group similar chunks together. As pointed out by,[13] the reason to take the segment-then-cluster approach is that clustering algorithms usually make no attempts to assign adjacent chunks into the same cluster, resulting in often unsatisfactory clustering results.

However, the segment-then-cluster approaches make no effort correcting segmentation errors. In Figure 1, the recording where the wearer walked on a sky bridge was wrongly segmented into three chunks, i.e. over-segmentation. A unconstrained clustering algorithm puts two of them in the correct "sky bridge" cluster but puts the middle segment in the wrong cluster, which is not satisfactory given the surrounding segments were

---

E-mail: {whlin,alex}@cs.cmu.edu

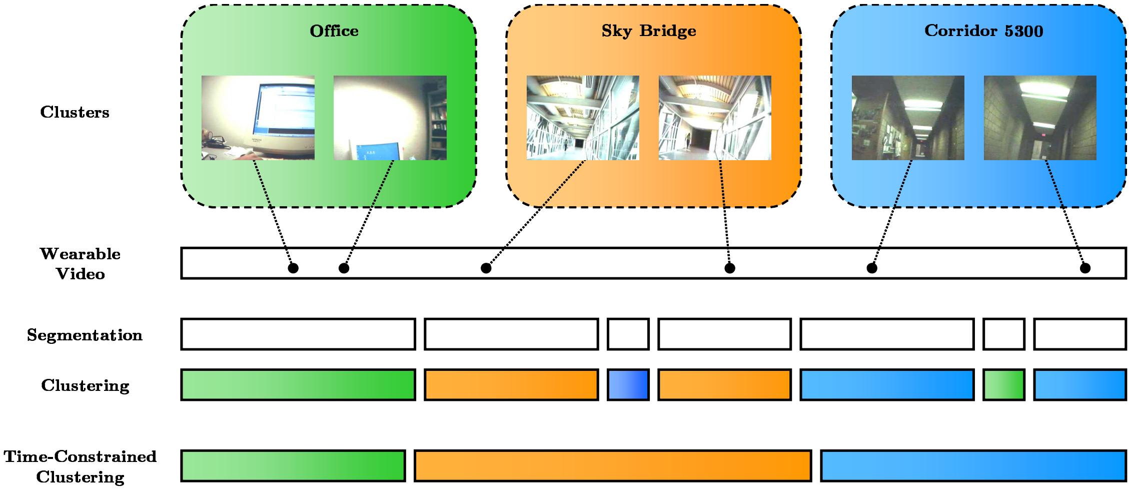


Figure 1. An example continuous video illustrates time-constrained clustering. Inside the "Sky Bridge" and "Corridor 5300" clusters there are errors caused by the first segmentation step, and the second clustering step fails to correct the errors. On the contrary, by imposing time constraints we not only accomplish two tasks in one step but also eliminate the segmentation errors.

all in the "sky bridge" cluster. In this paper we address the problem and propose a novel one-pass method to accomplishing segmentation and clustering at the same time, which saves the trouble of running two separate steps over a large collection of continuous recordings. Moreover, by imposing time constraints on a clustering algorithm, K-Means, we encourage contiguous video frames to be clustered together, and thus segmentation errors can be corrected. The experiment results show that time-constrained K-Means algorithm substantially outperforms the unconstrained version.

Note that our use of time constraints is very different from Yeung and Yeo's work.[10] They utilized the linearly narrative structure of TV sitcoms to constrain the members of a cluster to be close in time, i.e. the same story unit, while our motivation for imposing time constraints is to reduce segmentation and clustering errors due to spurious light change or body-induced camera motions, and thus we constrained the cluster assignment of a video frame to be as consistent as the cluster assignments of the neighboring frames, not the members in the same cluster. The video frames of a cluster, say, "office" cluster, can occur multiple times in the continuous video. Imposing time constraints in Yeung and Yeo's way would have many local "office" clusters, while our method will generate a single, global "office" clusters, enabling the user to quickly find all "office" segments.

The rest of the paper is organized as follows. We briefly review the K-Means clustering algorithm in Section 2, and discuss why K-Means often results in poor clustering results on continuous recordings. We describe the proposed Time-Constrained K-Means algorithm in Section 3. In Section 4, we present experiment results that objectively evaluate the proposed time-constrained clustering algorithm on 62.5 hours of continuous recordings. Finally we conclude and discuss future work in Section 5.

2. K-MEANS CLUSTERING ALGORITHM

K-Means is a one of the most popular clustering algorithms. The K-Means algorithm attempts to partition data into groups by minimizing the within-group squared Euclidean distance. More formally speaking, K-Means partitions data set $\mathcal{X} = \{x_i\}_{i=1}^n$ into k disjoint clusters $\mathcal{C} = \{\mathcal{C}_i\}_{i=1}^k$, where every data points x belongs to only one of the cluster $\mathcal{C}_i$, i.e. $\bigcap_{1\le i,j\le k} \mathcal{C}_i, \mathcal{C}_j = \emptyset$. K-Means minimizes the following objective function to find the best partition function on $\mathcal{X}$,

$$J_{\text{K-means}}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{x_j \in \mathcal{C}_i} ||x_j - \mu_i||^2 \tag{1}$$

where μ_i is the mean of the data points belong to the cluster $\mathcal{C}_i$, i.e. $\mu_i = |\mathcal{C}_i|^{-1} \sum_{x_i \in \mathcal{C}_i} x_i$.

The minimal value of (1) can be found in an iterative fashion, as shown in Figure 2. The convergence can be easily verified by observing that both "cluster assignment" and "mean re-estimation" steps in the loop decrease the value of the objective function in (1).[14] The solution found by the iterative algorithm, however, is not guaranteed to be globally optimal, and the problem is usually partially alleviated in practice by trying different initial values and selecting the one with the lowest objective value. The time complexity of the K-Means algorithm for each iteration is $O(nkp)$, where n is the number of examples, p is the dimension of the feature vector, and k is the number of the clusters. Our implementation of K-Means algorithm is based on.[15]

```
Data: data points $\{x_i\}_{i=1}^{n}$
Result: $k$ clusters $\{\mathcal{C}_i\}_{i=1}^{k}$
$t \leftarrow 0$;
initialize cluster means $\{\mu_i^{(0)}\}_{i=1}^{k}$;
repeat
    cluster assignment: assign each data point $x_i$ to the closest cluster $\mathcal{C}_j^{(t+1)}$, i.e. $\arg\min_j ||x_i - \mu_j^{(t)}||^2$;
    mean re-estimation: update the cluster means, i.e. $\mu_i^{(t+1)} = |\mathcal{C}_i^{(t+1)}|^{-1} \sum_{x_j \in \mathcal{C}_i^{(t+1)}} x_j$;
    $t \leftarrow t + 1$;
until convergence;
```

Figure 2: K-Means Algorithm

The difficulty of naïvely applying K-Means to structuring continuous video recordings can be attributed to the independence assumptions made on data. For continuous captured video, the video frames, in fact, are very dependent in time. The K-Means algorithm implicitly assumes that data points are independently sampled from an unknown distribution with k components, which clearly contradicts with the realty that individual video frames are closely related to surrounding frames in time. Therefore, we should exploit the temporal relationship to constraint the clustering process such that adjacent frames are not assigned into different clusters.

3. TIME-CONSTRAINED K-MEANS CLUSTERING ALGORITHM

In order not to ignore temporal relationship between data points in K-Means, we extend the K-Means algorithm to incorporate temporal relationship between data points, and call this extended version Time-Constrained K-Means, TCK-Means.

The temporal constraints is captured as an additional penalty term in the objective function of the TCK-Means algorithm , as shown in the following equation,

$$J_{\text{TCK-Means}} = \sum_{i=1}^{k} \sum_{x_j \in \mathcal{C}_i} \left(||x_j - \mu_i||^2 + \sum_{x_j \notin C_i} w(x_i, x_j) \right) \tag{2}$$

where $w(x_i, x_j)$ is the cost function that determines the penalty of clustering adjacent frames x_i and x_j into the different clusters.

By designing a proper cost function that penalizes the cluster assignment of one video frame into a cluster different from adjacent video frames, TCK-Means encourages video frames that are close in time to be clustered together.

The (local) optimal solution to (2) can be found in an iterative manner similar to the K-Means algorithm, as listed in Figure 3.

The TCK-Means algorithm looks similar to the K-Means algorithm, except for the cluster assignment step. TCK-Means takes the temporal relationship into consideration, and thus any data points are preferably assigned to a cluster not only close in the feature space but also close in the temporal domain.

The convergence of the TCK-Means can be argued in the same way as K-means because the objective in (2) keeps decreasing or stays the same in the alternate steps.[16] The time complexity at the first sight seems to jump to $O(n^2k)$ due to the cost summation step. However, if we restrict the scope of the time constraints within

```
Data: data points $\{x_i\}_{i=1}^n$
Result: $k$ clusters $\{\mathcal{C}_i\}_{i=1}^k$
$t \leftarrow 0$;
initialize cluster means $\{\mu_i^t\}_{i=1}^k$;
repeat
    cluster assignment: assign each data point $x_i$ to the cluster $\mathcal{C}_j^{(t+1)}$ with the lowest cost, i.e.
    $\arg\min_j ||x_i - \mu_j^{(t)}||^2 + \sum_{x_j \notin \mathcal{C}_j^{(t)}} w(x_i, x_j)$;
    mean re-estimation: update the cluster means, i.e. $\mu_i^{(t+1)} = |\mathcal{C}_i^{t+1}|^{-1} \sum_{x_j \in \mathcal{C}_i^{t+1}} x_j$;
    $t \leftarrow t + 1$;
until convergence;
```

Figure 3: TCK-Means Algorithm

the small number of neighbors that is much smaller than p, as shown later in Section 3.1, the time complexity of TCK-Means can be kept the same as K-means in the order of $O(nkp)$.

TCK-Means is inspired by a class of clustering algorithms called pairwise-constrained clustering.[17, 18] Usually pairwise constraints are specified as "must-link" constraints, i.e. two points must be clustered together, and "cannot-link" constraints, i.e. two points must not be clustered together. Here time constraints can be interpreted as the soft version of the must-link and cannot-link constraints. When two video frames are close in time, they are encouraged to be in the same cluster, which is like relaxed "must-link" constraints. When two video frames are far away, they can be discouraged to be in the same cluster, which is like relaxed "cannot-link" constraints.

3.1. Cost Function in TCK-Means

The time cost function $w(x_i, x_j)$ plays a crucial role in the TCK-Means algorithm. The function determines quantitatively how strongly time constraints are imposed on the clustering algorithm in addition to the Euclidean distances in the feature space. In this paper we consider the following cost function,

$$w(x_i^{t_i}, x_j^{t_j}) = \alpha \mathbb{I}(|t_i - t_j| < d) \tag{3}$$

where the superscripts t_i and t_j are the time offsets of the x_i and x_j in the continuous video recording, respectively, $\mathbb{I}$ is an indicator function, α is a constant cost, and d is the window size.

If two data points are close in time within the window of w but are not clustered in the same group, that will incur α cost. If we set α to zero, TCK-Means will behave exactly like K-Means because the constraint term in (2) disappears. Similarly, TCK-Means rolls back to K-Means when window d is zero. When d is small, clustering results will tend to over-segment because there is little constraints on temporal consistency. On the other hand, when d is large, the clustering results will turn to under-segment because of strong constraints on temporal consistency with neighbors. Therefore, d is the trade-off between strong and weak temporal consistencies.

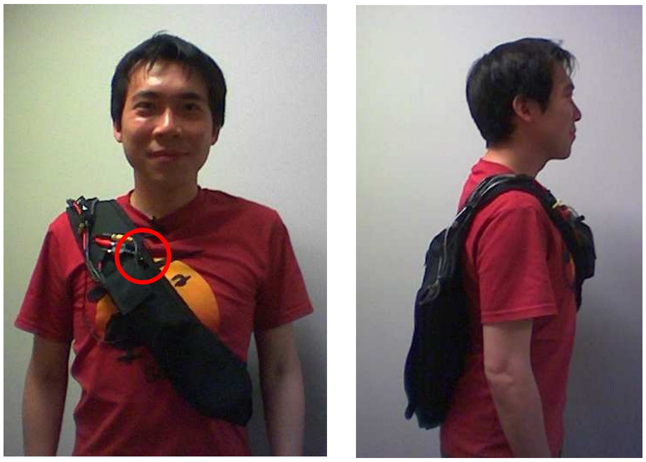
The cost constant α is likely to vary from data to data , and it is unreasonable to expect users specify the parameters externally. Our solution is to set α proportional to the average squared distance of the data set, resulting in equal emphasis on within-cluster coherence and temporal consistency in (2). Users are therefore free from the burden of setting α.

4. EXPERIMENTS

4.1. Data Collection

One of the authors wore a video a recording device to continuously capture his everyday life for two to six hours on weekdays, and the experiment lasted for a month. The wearable recording device consisted of a small, wearable camera and a 1.6 GHz laptop, as shown in Figure 4. The laptop was equipped with an extra battery pack such that it could operate continuously for at least six hours without re-charging.

The high-fidelity video recordings pose a significant challenge to the wearable storage system. In order to provide playback quality for offline browsing and searching, we opt for 320 × 240 pixels video resolution and



(a) The wearable camera, circled in red, is positioned in the front chest, which is one of the best sites on the front of the upper body to attach optical device.[19]

(b) A video camera is connected to the notebook in the backpack. The laptop computer performs real-time video compression and stores video recordings.

Figure 4. Wearable Continuous Video Recording Device

capture 29.412 frames per second in the 24-bit color depth. Without any compression, raw video continuous recordings would quickly overflow the hard-disk space with data rates as high as 24 Gigabytes per hour for the chosen video resolution. In order to acquire high-quality video and meet the constraints of the storage capacity of the laptop at the same time, we take a two-step approach to reduce the size of the video recordings. All raw video recordings are compressed online by a fast video encoder. After finishing recording at the end of a day, video recordings are uploaded to the server and then compressed offline by a slower two-pass MPEG-4 encoder with higher compression rate. In the end, a total of 62.5 hours of continuous video of everyday life were collected and took a total of 37.9 Gigabytes of disk space.

We represent a continuous video as a series of one-second units, and extract color features from the middle frame of each unit. 5 by 5 by 5 3-D color histogram in RGB color space* is calculated, resulting a 125-dimensional feature vector. Visual content within one second does not vary dramatically.

Every second of the everyday recordings is annotated with locations. Location is arguably one of the most important user's context,[20] and is of particular interest since positioning technologies like GPS do not work indoors. While granularity of locations can range from meters to kilometers, we set the functionally useful granularity at the room level, partially due to the nature of the wearer's life as a graduate student. In addition to office rooms, the corridors, staircases, and elevators in different building are labeled . All outdoor scenes are labeled as "campus outdoors." There are total 34 locations in the 62.6 hours of recordings. Note that data are annotated here for the evaluation purpose, and the goal of this study is not to learn to identify place as in the supervised learning framework.[21] The sheer amount of recordings make it unrealistic to ask a user to annotate hours of training data.

---

*While determining the best feature set for continuous video is a very important research question, it is not the main goal in our paper. For illustration purposes we use RGB color histogram as the instance of any number of available features. Our algorithm is definitely not limited by color histogram, and color histogram can be replaced with more advanced features.

4.2. Evaluation Metrics

Clustering can be seen as the process of recovering underlying true location labels (clusters) from the data in a unsupervised fashion. Therefore, a clustering algorithm performs well if the clustering results show high degree of consistency with human-annotated labels on the data. We calculate how often a pair of data points that a clustering algorithm put into the same group are indeed in the same labeled group, i.e. precision, and how many pairs of data points that are labeled as the same group are recovered by the clustering algorithm, i.e. recall, and the harmonic mean of precision and recall, i.e. F1, in the following formulae,

$$\begin{aligned} \text{precision} &= \frac{\sum_{x_i} \sum_{x_j} I(x_i, x_j \in \mathcal{C}_k \text{ and } x_i, x_j \in \mathcal{T}_{k'})}{\sum_{x_j} \sum_{x_j} I(x_i, x_j \in C_k)} \\ \text{recall} &= \frac{\sum_{x_i} \sum_{x_j} I(x_i, x_j \in \mathcal{C}_k \text{ and } x_i, x_j \in \mathcal{T}_{k'})}{\sum_{x_i} \sum_{x_j} I(x_i, x_j \in \mathcal{T} k')} \\ \text{F1} &= \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} \times \text{recall}} \end{aligned}$$

where $\mathcal{T}_{k'}$ is a set of data points with the true label k'. All of the metrics range between 0 and 1, and the higher the score, the better the performance.

4.3. TCK-Means vs. K-Means

We compare our proposed TCK-Means with K-Means[†] on the continuous recordings, and the results are shown in Table 1. Moreover, we provide three random baselines for comparisons, including clustering all video frames into a single cluster (All Same), clustering each video frame into its own cluster (All Different), and randomly assign each video frame into clusters (Random). The relative improvements in percentage are shown in the last row of the table.

Methods	Precision	Recall	F1
Baseline (All Same)	0.1710	1.0	0.2921
Baseline (All Different)	N/A	0	N/A
Baseline (Random)	0.1697	0.1674	0.1600
K-Means	0.7952	0.4101	0.5239
TCK-Means	0.8060	0.4872	0.5930
Improvement	+1%	+19%	+13%

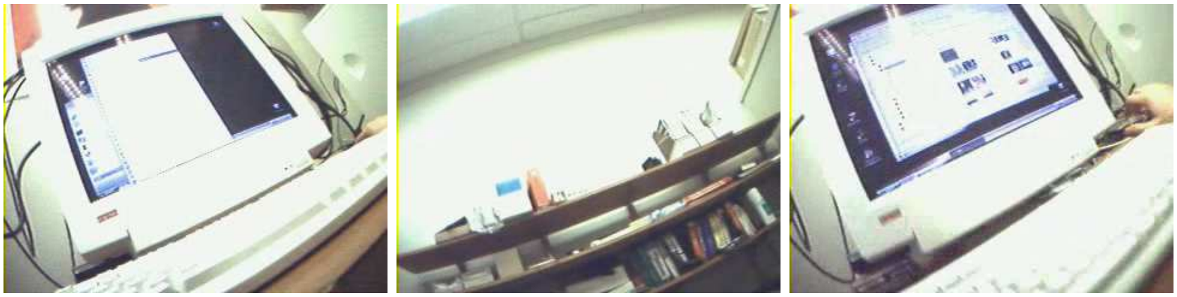
Table 1. Experiment Results: TCK-Means vs. K-Means with $d = 1$

Both the K-Means and TCK-Means algorithms clearly outperform the three baselines, which suggests that simple RGB color features can capture large potion of the visual characteristics of various locations. The high precision shows that color similarity is often sufficient to correctly cluster together two video frames captured at the same place . However, low recalls suggests that color similarity is sometimes not adequate to cluster dissimilar video frames from the same environment. Take Figure 5 as an example. While a user is typing in front of a computer screen in the office, s/he may suddenly turns to a bookshelf to fetch a book, and then comes back to the seat. While both typing and book fetching events occur in the office, they look very differently in color histogram, and any unconstrained clustering algorithms will put the two events into different clusters. However, two events occur close in time, and thus TCK-Means encourages the book fetching event to be in the same cluster with the preceding and following frames, i.e. the office cluster. The substantial improvement in recall supports our ideas with empirical evidence.

4.4. TCK-Means with Varied Window Sizes

We vary the window size to look into how window size influence the clustering performance, and the results are shown in Table 2.

[†] In order to control the experiment and compare two algorithms fairly without an extra confounding factor, we fix the cluster number k in K-Means and TCK-Means with the true number of clusters from annotation truth. In practice we do not know k in advance, and techniques such as "gap statistics"[14] can be used to estimate the best k for the data.



(a) typing in front the screen (b) turning to the bookshelf (c) back to the computer

Figure 5. An example shows how time constraints can cluster color dissimilar video frames into the same cluster.

Method	Precision	Recall	F1
Baseline (All Same)	0.1710	1.0	0.2921
Baseline (All Different)	N/A	0	N/A
Baseline (Random)	0.1697	0.1674	0.1600
K-Means ($d = 0$)	0.7952	0.4101	0.5239
TCK-Means ($d = 1$)	0.8060	0.4872	0.5930
TCK-Means ($d = 2$)	0.8141	0.4999	0.5883
TCK-Means ($d = 4$)	0.8165	0.5110	0.5909
TCK-Means ($d = 6$)	0.8175	0.5104	0.5979
TCK-Means ($d = 8$)	0.8207	0.5174	0.6029
TCK-Means ($d = 10$)	0.8262	0.5313	0.6061
TCK-Means ($d = 12$)	0.8235	0.5297	0.6094
TCK-Means ($d = 14$)	0.8319	0.5307	0.6169
Improvement	+5%	+29%	+18%

Table 2. Experiment Results: TCK-Means with varied window size (in seconds) vs. K-Means.

When the window size d is increased, we impose greater time constraints, and therefore each video frame is asked to be as consistent with more surrounding frames as possible. Therefore, small camera motions or transient movement will not be wrongly clustered into different clusters from the locations immediately before and after. By enlarging the window size, the recall can improve from 19% (small window size $d = 1$) to 29% (large window size $d = 14$)!

When d is set to be greater than 14, the time constraints were so strong that TCK-Means cannot construct the same numbers of clusters as oracle anymore, which would confound our comparisons between K-Means and TCK-Means. Therefore we discard the clustering results that are less than true numbers of clusters and only report the results less or equal to 14. Eventually the clustering performance of TCK-Means falls back to "Baseline (All Same)" when the window size is as large as half of the the recording length.

5. CONCLUSIONS AND FUTURE WORKS

With advance of storage capacity and wearable sensors, it will soon become feasible to capture the whole life of human experiences in digital video. However, before any video retrieval systems can index the recordings, continuous video need to be segmented into manageable units. In the paper we propose a time-constrained K-means clustering algorithm to perform both segmentation and clustering in one step. The experiment results show that TCK-Means achieves substantially better clustering performance than the unconstrained version.

While the experiment results show that TCK-Means outperforms K-Means, it is still not clear to what extent the performance gain can transfer to continuous video retrieval. We plan to investigate the issue by conducting video retrial experiments with a set of queries, and evaluate the effectiveness of the video retrieval systems with our proposed clustering algorithms.

K-Means was chosen in our paper for its simplicity such that we can illustrate the idea of imposing time constraints on K-Means more clearly than other more complicated clustering algorithms. In future work we plan to extend our work to other clustering algorithms that can relax the constraint of hard clustering of K-Mean. Soft clustering algorithms such as Gaussian mixture models allow us to assign probabilities on multiple cluster assignments, which may improve segmentation and clustering of continuously recorded video.

ACKNOWLEDGMENTS

This material is based on work supported by the National Science Foundation under grant number IIS-0121641.

REFERENCES

1. S. Reich, L. Goldberg, and S. Hudek, "Deja view camwear model 100," in *Proceedings of the First ACM Workshop on Continuous Archival and Retrieval of Personal Experiences*.[22]
2. J. Gemmell, L. Williams, K. Wood, R. Lueder, and G. Bell, "Passive capture and ensuing issues for a personal lifetime store," in *Proceedings of the First ACM Workshop on Continuous Archival and Retrieval of Personal Experiences*,[22] pp. 48–55.
3. S. Mann, "Continuous lifelong capture of personal experience with eyetap," in *Proceedings of the First ACM Workshop on Continuous Archival and Retrieval of Personal Experiences*,[22] pp. 1–21.
4. V. Bush, "As we may think," *The Atlantic Monthly* **176**, pp. 101–108, July 1945.
5. J. Gemmel, G. Bell, R. Lueder, S. Drucker, and C. Wong, "Mylifebits: fulfilling the memex vision," in *Proceedings of the Tenth ACM International Conference on Multimedia*, pp. 236–238, 2002.
6. T. Hori and K. Aizawa, "Context-based video retrieval system for the life-log applications," in *Proceedings of the 5th ACM SIGMM International Workshop on Multimedia Information Retrieval*, pp. 31–38, 2003.
7. W.-H. Lin and A. Hauptmann, "A wearable digital library of personal conversations," in *Proceedings of the Second ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 277–278, 2002.
8. B. Clarkson, K. Mase, and A. Pentland, "Recognizing user context via wearable sensors," in *Proceedings of the Fourth International Symposium on Wearable Computers*, pp. 69–75, 2000.
9. H. D. Wactlar, M. G. Christel, A. G. Hauptmann, and Y. Gong, "Informedia experience-on-demand: Capturing, integrating and communicating experiences across people, time and space," *ACM Computing Survey* **31**, June 1999.
10. M. M. Yeung and B.-L. Yeo, "Time-constrained clustering for segmentation of video into story units," in *Proceedings of the 13th International Conference on Pattern Recognition*, **3**, pp. 375–380, 1996.
11. K. Aizawa, K.-I. Ishijima, and M. Shiina, "Summarizing wearable video," in *Proceedings of the 2001 International Conference on Image Processing*, 2001.
12. D. P. Ellis and K. Lee, "Features for segmenting and classifying long-duration recordings of personal audio," 2004.
13. D. P. Ellis and K. Lee, "Minimal-impact audio-based personal archives," in *Proceedings of the First ACM Workshop on Continuous Archival and Retrieval of Personal Experiences*.[22]
14. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer-Verlag, 2001.
15. J. A. Hartigan and M. A. Wang, "Algorithm as 136: A k-means clustering algorithm," *Applied Statistics* **28**(1), pp. 100–108, 1979.
16. S. Basu, A. Banerjee, and R. J. Mooney, "Active semi-supervision for pairwise constrained clustering," in *Proceedings of the SIAM International Conference on Data Mining (SDM-2004)*, pp. 333–344, 2004.
17. K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl, "Constrained k-means clustering with background knowledge," in *Proceedings of the 18th International Conference on Machine Learning (ICML-01)*, 2001.
18. S. Basu, M. Bilenko, and R. J. Mooney, "A probabilistic framework for semi-supervised clustering," in *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 59–68, 2004.

19. W. W. Mayol, B. Tordoff, and D. W. Murray, "On the positioning of wearable optical devices," Tech. Rep. OUEL2241/01, Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK, 2001.
20. E. Kaasinen, "User needs for location-aware mobile services," *Personal and Ubiquitous Computing* **7**(1), pp. 70–79, 2003.
21. A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin, "Context-based vision system for place and object recognition," in *Proceedings of the Ninth IEEE International Conference on Computer Vision*, 2003.
22. *Proceedings of the First ACM Workshop on Continuous Archival and Retrieval of Personal Experiences*, ACM Press, 2004.