VFerret: Content-Based Similarity Search Tool for Continuous Archived Video

Zhe Wang, Matthew D. Hoffman, Perry R. Cook, Kai Li
Computer Science Dept, Princeton University, Princeton NJ, 08540
Email: {zhewang, mdhoffma, prc, li}@cs.princeton.edu

ABSTRACT
This paper describes VFerret, a content-based similarity search tool for continuous archived video. Instead of depending on attributes or annotations to search desired data from long-time archived video, our system allows users to perform content-based similarity search using visual and audio features, and to combine content-based similarity search with traditional search methods. Our preliminary experience and evaluation shows that content-based similarity search is easy to use and can achieve 0.79 average precision on our simple benchmark. The system is constructed using Ferret toolkit and its memory footprint for metadata is small, requiring about 1.4Gbytes for one year of continuous archived video data.

Categories and Subject Descriptors
H.3.1 [Information storage and retrieval] Content analysis and retrieval.

General Terms
Algorithms, Design, Experimentation.

Keywords
Similarity search, video retrieval.

1. Introduction
A challenge in building a digital memory system is the ability to search desired information quickly and conveniently. In 1945, Vannevar Bush described his vision of Memex: "a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility." [1]. Today, commodity disks and video cameras can easily be used to implement the first part of Bush's description of the Memex---continuously capture and achieve a person's life. The challenge is to design and implement a retrieval system that "may be consulted with exceeding speed and flexibility," realizing the second part of the Memex vision.

Much of the previous work on building retrieval systems for continuous archived data is based on attributes, annotations, or automatic classifications. These approaches have limitations in different dimensions. Attributes such as time and location are helpful for information retrieval, but they do not describe the content of the archived information. Annotations of non-text data (such as audio, images, and video) can provide a text-based search engine with effective ways to retrieve information. However, generating annotations for a continuous-archived life log is a daunting and perhaps impossible task for most people. Automatic classifications try to generate annotations automatically. They tend to generate coarse-grained classes, whereas retrieving information in a life-long continuous archive require both coarse-grained and fine-grained content-based search capabilities.

Princeton's CASS (Content-Aware Search Systems) project studies how to build a retrieval system to perform content-based search of a variety of data types. We have designed a system called VFerret which provides the ability to perform content-based similarity search on unlabeled continuous archived video data. We are interested in several research issues. First, we are interested in studying how to use both audio and visual features to effectively perform content-based similarity search on continuous archived video data. Second, we would like to design a system that requires minimal metadata to handle years of continuous archived data. Third, we would like the system to allow users to combine the content-based similarity search capability with annotation/attribute-based search in retrieval tasks.

This paper describes the design and implementation of the VFerret system. The system segments the video data into clips and extracts both visual and audio features as metadata. The core component of the system is a content-based similarity search engine constructed using a toolkit called Ferret [2] that searches the metadata by a combination of filtering, indexing and ranking. The system also has a built-in attribute/annotation search engine that allows users to perform a combination of attribute/annotation-based and content-based search.

To experiment with the system, we used the DejaView Camwear model 200 as a capturing device to continuously record 6 weeks of a graduate student's life. To evaluate the search quality of our system, we have manually labeled the data and used a portion of the data as our training dataset. Our evaluation shows that the system can achieve an average precision of 0.46 by using visual features alone, 0.66 by audio features alone, and 0.79 by combining visual and audio. Our analysis shows that the current configuration of the system requires only 13.7Gbytes of storage for the metadata of 10 years of continuous archived data.

CARPE'06, October 28, 2006, Santa Barbara, California, USA.

2. System Overview

When designing the system, we first examine a number of usage scenarios of continuous archived data to identify the needed functionalities. The following is a set of sample tasks a graduate student wants:

- Find the talk given by a professor during an industrial affiliate day.
- Recall the ideas proposed by a particular team member during a project group meeting.
- Find the clip where I met some baby Canadian geese along a trail.

For the first question, if one can find the specific date when the industrial affiliate day is from her calendar, it should be easy to find the clip quickly by browsing through the clips on that day.

The second and third tasks, however, will be more difficult and time-consuming with attribute/annotation-based search method if the video clips are not rigorously annotated. What we are proposing is to use content-based search to locate these clips: we can first find some clips that look or sound similar to our memory of the desired clips and then use content-based similarity search to find the clips of interest.

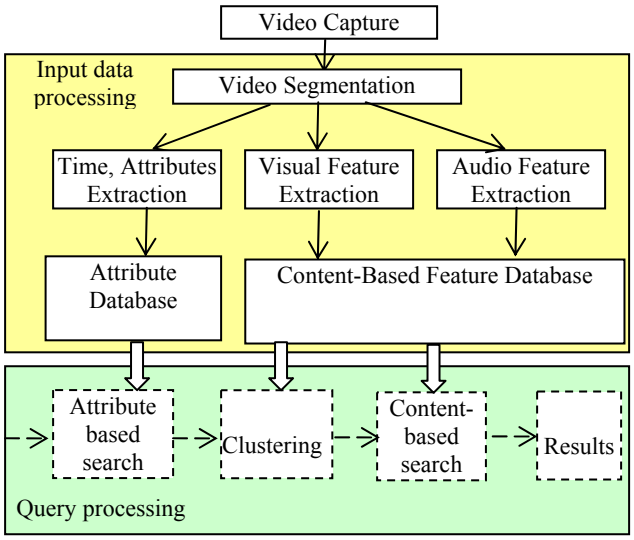

Figure 1: VFerret system architecture

With these questions in mind, we built the video search system as shown in figure 1. Once the continuously captured video is inserted into the system, it is first segmented into short video clips. We extract capture time and other possible attributes associated with the video clips and insert them into the attribute database. Meanwhile, in order to do content-based similarity search, we also extract the visual and audio features from the video clips so that we can index the clips based on their content.

At the query processing time, the user can combine the content-based search with attribute-based search to find the clips of interest. We believe these two processes are complementary: Attribute-based search can help to bootstrap a content-based search, while content-based search can search all the clips that have similar contents so that the clips that are not labeled can be found conveniently.

Our preliminary experience shows that a user typically starts with an attribute-based search with a time range; the resulting clips are then clustered based on visual and audio features. For each cluster, only one representative video clip is shown, so the user can quickly locate a video clip that is similar to the desired clip. Once a similar clip is found (this usually is much easier than finding the precise desired clip), the user can initiate the similarity search to find the clips of interest quickly. We will describe the system in more details in section 4.

3. Content-Based Similarity Search

We use content-based search as the main retrieval method to handle the unlabeled personal continuous archived video. The content-based search takes one video clip as a query clip, and returns a collection of video clips similar to the query clip. For example, using one meeting clip as a query object, one can find most other meeting clips without annotations. Our content-based video search system combines visual and audio features to determine the overall similarity of the video clips.

3.1 Video clip segmentation

The system first segments the continuous archived video into short video clips. This is a necessary step since the lengths of the original recordings vary and each recording can contain multiple activities.

We use a simple segmentation method to evenly split the video recordings into 5 minutes clips each. This is adequate for our search purpose since most of the video clips of our interests last more than 5 minutes. Although a better video segmentation tool would be more desirable, we leave this to future system improvements. We have tried several available commercial and research video scene detection and segmentation tools such as Handysaw [3], Microsoft movie maker, and the segmentation tool from [4]. These segmentation methods do not work well for continuous archived data because they depend on camera or lens movements that commercial videos tend to have for segmentation. We believe personal continuous archived videos are inherently different from commercial videos in terms of segmentations. Commercial videos have relatively clean shots and clear edited boundaries between scenes, whereas personal continuous archived videos tend to have unclean shots and no editing.

3.2 Visual feature extraction

For each segmented video clip, the system extracts a set of visual feature vectors to represent the clip. These features are used in the similarity search to determine whether two video clips look similar or not.

To extract the visual feature, we evenly sample 20 individual images from each video clip. For the video clips segmented into 5 minutes each, we extract one frame every 15 seconds. For each frame, we first convert them from RGB into HSV color space since the HSV color space distance is better for measuring human perceptual similarity. To compare images for similarity, the system uses the approach proposed by Stricker [5] which uses 3 central moments of the color distribution. The 3 moments are mean, standard deviation and skewness, which describe the average, variance and the degree of asymmetry of the color distribution. It has been shown in [6] that the color moments performs only slightly worse than the much higher-dimensional color histogram.

In our system, we take 3 color moments of each channel of HSV space. This gives us a compact 9 dimension feature vector for each image. With the training dataset, we further normalize the feature vector with their mean and standard deviation. Finally, we use L_1 distance to calculate the distance between feature vectors.

Figure 2: Visual feature extraction.

3.3 Audio feature extraction

To extract audio features, the system evenly split the audio channel of each 5-minute video clip into 20 individual 15-second segments.

For each 15-second audio segment, the system uses 154 audio features patterned after those used by Ellis and Lee [7] to describe audio segments. We begin by extracting several sets of short-time features describing 10 ms windows calculated every 5 ms over the entire segment. Then, to condense this information into a compact descriptor of the entire segment, we take the means and standard deviations of these short-time features, normalizing the standard deviations by their respective means.

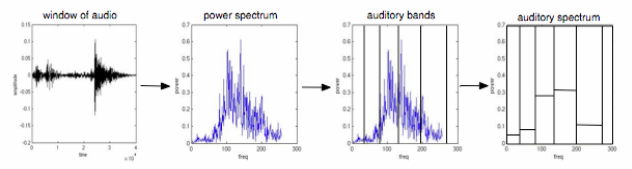

Figure 3: Audio feature extraction.

The first set of 25 short-time features measures the energy in each of 25 Bark-scale frequency bands of the window. The Bark scale divides the frequency spectrum into bands that increase in width with frequency in a way that models the bandwidth of our auditory system, as shown in figure 3. To measure the energy in one of these bands for a given window, we take the short-time Fourier transform (STFT) of the window and sum the energy in all frequency bins that fall within that band.

The next set of 25 short-time features further describe each Bark-scale band by treating the energy spectra within those bands as probability distributions and calculating their entropies. If all of the energy in a band is concentrated within one bin, its entropy will be very low, whereas if the energy is more evenly distributed the entropy will be high.

For our last set of 25 short-time features, we again treat the energy spectrum within each band as a probability distribution and calculate the Kullback-Leibler divergence for each band between subsequent windows. This provides us with information about how much the shape of the spectrum within each band is changing from window to window.

Finally, we calculate the entropy and Kullback-Leibler divergence as above for the entire short-time Bark-scale energy spectrum, yielding another 2 short-time features.

Taking the means and normalized standard deviations of each of these 25 + 25 + 25 + 2 features gives us our 154 long-time audio features:

- 50: Mean, std of energy in each of 25 Bark-scale band
- 50: Mean, std of entropy in each of 25 Bark-scale band
- 50: Mean, std of Kullback-Leibler divergence in each of 25 Bark-scale band
- 4: Mean, std of entropy and Kullback-Leibler divergence for the entire energy spectrum

L_1 distance is used to calculate the distance between feature vectors.

3.4 Combined feature vector

For each 15-second video segment, we combine the visual feature vector extracted from the sample image and the audio feature vector extracted from the corresponding audio segment to form a single feature vector. The proper weight assigned to visual and audio features are derived from the training data set as described in section 5.1. We use L_1 distance to calculate the distance between the combined feature vectors.

3.5 Similarity search

Given a query video clip, the goal of the similarity search is to find all the clips that are similar to the query video. In our system, we represent each video clip as a set of visual and audio features. So given the query video clip X, we would like the system to find all video clips Y in the collection such that the distance d(X,Y) is within a small range (also referred to as *k nearest neighbor* problem). The similarity search system will return a ranked list of video clips where the clip with smallest distance to the query clip ranks first.

Since each video clip is represented by a set of combined feature vector rather than a single combined feature vector, we need to find a proper distance between set of feature vectors. In our implementation, we use the one-to-one best match method to find the overall minimal distance between two sets of feature vectors.

As shown in figure 4, query clip X is sampled into individual images and audio clips. A set of feature vectors $<X_i>$ are extracted from the clip, one from each image and audio segment. Same applied to all the other clips in the collection. For a particular candidate Y, the distance d(X,Y) is defined as the best one-to-one match such that the sum of all distances between the underlying feature vectors is minimized:

$$d(X,Y) = \min\left\{\sum_{i=1}^{n} d(X_i, Y_{f(i)})\right\}$$

Where $f(i)$ is a function provide any permutation of [1,…,n], $d(X_i, Y_j)$ is the distance between the corresponding feature vectors.

During the similarity search, all the video clips in the collection are compared with the query video clip. They will be ranked according to their distance to the query video clip. To speed up the similarity search, our system uses *sketches* to represent feature vectors, and perform filtering and indexing to speed up the search process. A technical paper on the Ferret toolkit [2] provides more detailed information.

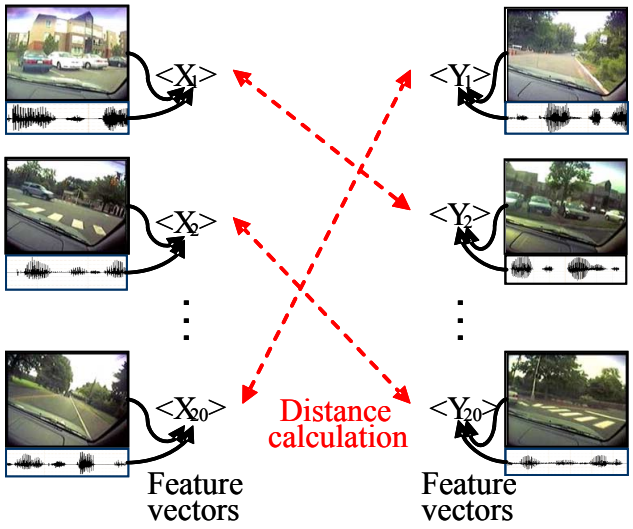

Figure 4: Illustration of distance computation.

4. Implementation

4.1 Video capture system

We adopt the commercial wearable camera [8] as shown in figure 5. The DejaView Camwear model 200 has a separate camera lens that can be attached to hat or eyeglass, and a recording device that can record up to 3-4 hours of video with a single charge of battery. It records 320*240 mpeg4 video clips with sound to the secure digital flash memory card. One hour of video will take about 0.5 GB of storage space. One of the authors carried the camwear, and recorded on average of 1 hour of video every day from May to June.

Figure 5: DejaView Camwear model 200 system.

4.2 Video search system

The video storage and search system is built using the Ferret toolkit. Our system leverages the existing Ferret infrastructure by configuring it with video segmentation, visual and audio feature extraction components.

To fully utilize the content-based similarity search, it is important to start a similarity search with a relevant query video clip. We have implemented attribute-based search methods to help bootstrap the content-based search quickly. These methods can reduce the number of video clips users need to browse through, but they still present a challenge when many video clips remain to be checked. The role of content-based similarity search is to bridge the gap between the results returned from attribute-based search and the final results.

We will use an example to illustrate the search process. Consider the example that someone wants to show a friend the clips where she saw several baby Canadian geese with their family on her way home.

4.2.1 Timeline-based search step

The timeline-based search is the most natural method to search the personal continuous archived data. Since the timestamp comes for free and people naturally anchor events with time, most systems for personal archive have this capability. In our experience, the timeline based search is effective when the event has a distinctive date (e.g.: Christmas) or is associated with some other context (e.g.: email, event saved on calendar) that is searchable via other means. The Mylifebits system [9] and the Lifelog system [10] leverage the context information and demonstrate the effectiveness of using timeline and context to retrieve contents.

On the other hand, for old events or relatively insignificant events, it is difficult to recall the exact time it occurred. In such cases, one must use a relatively wide time range, yielding many candidate video clips. A time range can be used to reduce the search range in the first step of our video search system. For our example, the user recalls that the encounter happened early this summer. So the user can limit the search from May 1st to Jun 1st, which will reduce the number of clips in the next step.

4.2.2 Clustering step

After the timeline filtering, the candidate set may still be too large for a user to browse through quickly. Our system uses a k-means clustering algorithm [11] to cluster the filtered candidates into a small set of clusters. A representative video clip is found from each cluster so that user can quickly browse the full collections.

The k-means algorithm uses the same visual and audio features as the similarity search system. The only difference is that we use the average of the 20 feature vectors, rather than using all of them. This reduces the size of the overall feature vector and greatly speeds up the clustering speed to make it interactive. This design decision is based on the observation that clustering is for users to choose candidates to perform content-based search queries instead of final results. So long as the user can identify one video clip that is similar to the desired clip, she will be able to start the similarity search with that clip.

For our example, the video clip should be an outdoor scene and is on a trail with lots of green trees. The user will look for a cluster with outdoor scenes.

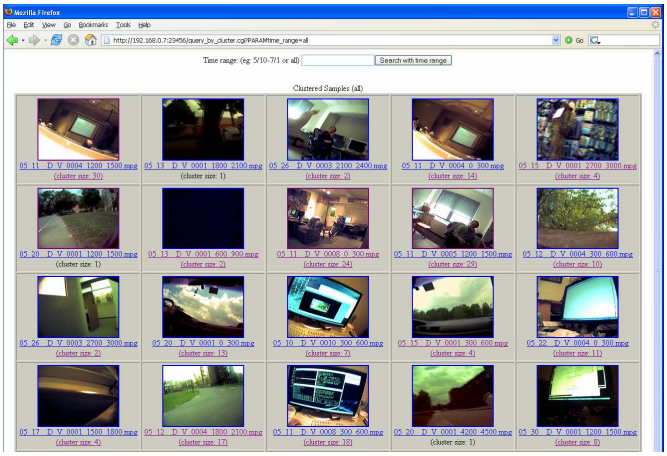

Figure 6: 20 video clusters presented to the user after the timeline and clustering steps.

4.2.3 Content-based similarity search step

The last step of the search is the content-based similarity search. Once user has a query clip, she can initiate the similarity search and iteratively refine the search to find the desired result. She can either use a new clip in the search result as the new query, or use multiple similar clips to start a new search. This process will provide higher quality results iteratively and help the user quickly pinpoint the desired clips without browsing through the entire candidate set.

For our example, the user would get a collection of similar trail video clips and find the clips of interest.

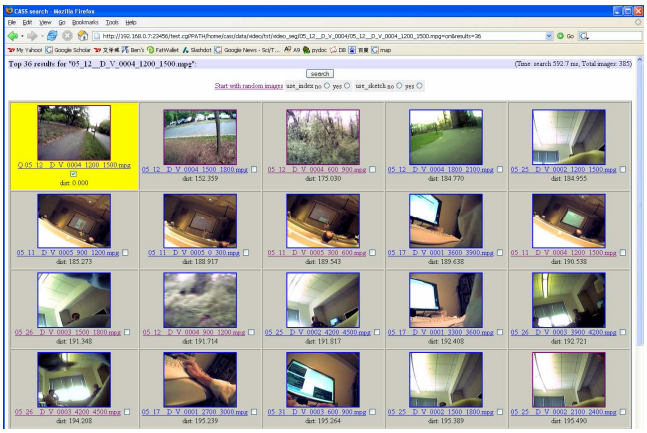

Figure 7: Results after the content-based similarity search step.

5. Initial evaluation

We have done an initial evaluation of the system to answer two questions:

- How well does the content-based similarity search produce high-quality results?
- What is the systems resource overhead for the content-based search of continuous archived video?

One of the authors recorded 6 weeks of his personal life using the Camwear gear. The video clips involve activities such as work, drive, walk, meeting, shopping, etc. The system segments the data into a total of 385 video clips, as described above.

5.1 Benchmark

We separate the video clips into two sets of about the same size: one for training and one for testing. For each set, a similarity benchmark is manually defined. The training set is used for training the system while the test set is used here to report the benchmark result.

For the benchmark, we defined the similarity sets by manually reviewing the video clips and grouping video clips together according to activity types to form similarity sets. For example, one similarity set consists of several clips recorded while walking on an outdoor trail while another similarity set consists of recordings made while driving in a car. Within a similarity set, all the clips are believed to similar to each other, thus no rank is given within the similarity set. Note that, the benchmark clips are only a subset of all the video clips and all video clips are used in the similarity search test.

We come up with 6 similarity sets for each case, labeled as below:

Activity label	Training set (Number of clips)	Test set (Number of clips)
walking outside	5	2
meeting	9	7
shopping	6	5
driving	20	17
seminar	10	10
reading	4	4

5.2 Evaluation metric

We have chosen to use average precision to measure the effectiveness of our similarity search. Given a query q with k similar clips where query q is excluded from the similarity set, let $Rank_i$ be the rank of the i_{th} retrieved relevant clip ($1 <= i <= k$) returned by the system, then average precision is defined as follows:

$$average\_precision = \frac{1}{k}\sum_{i=1}^{k}\frac{i}{Rank_i}$$

Suppose similarity set is $\{q_1, q_2, q_3\}$ and the query is q_1. If the search results returned by the system are r_1, q_2, q_3, r_4, the average precision is 1/2*(1/2+2/3) = 0.583. This measure provides a single-valued measure, simplifying comparison of the effectiveness of different systems.

5.3 Results

We compared the average precision result of our search system using visual features alone, audio features alone, visual and audio features together:

Feature vectors	Average Precision
Visual	0.46
Audio	0.66
Visual + Audio	0.79

Our results on the benchmark suggest that the audio features are contributing more to our search performance than the visual

features. This is an interesting result, and we believe that it comes from the fact that in our benchmark, audio can capture more environmental features than visual. Although the camwear lens provides 60 degree field of view, the captured video still varies a lot in the same environment as head moves around. Meanwhile, audio captures relatively stable features independent of the head position in the same environment. This gives audio more power to distinguish different environments which are associated to the activities in our benchmark.

Although audio feature works well in classifying activities, we still rely mostly on visual part to present the search interface.. The current interface allows the user to see a tile (8x8) of thumbnails created from the video clip to quickly grasp the visual content of the clip. For audio part, we do not have such capability of fast-forwarding or quick sampling of the full clip. As a result, the visual feature still plays a very important role in user's search process.

5.4 System overhead

The similarity search system only needs to store extra feature vectors in addition to the video clips for similarity search capability. Even if the user continuously record the video 24 hours a day and 7 days a week, it will only need about 1.37 GBytes extra storage space to store the feature vectors for one year worth of video.

For the search speed, the current system can return the similarity set within 600ms for a collection of 385 clips. No indexing or filtering is used for the current search since linear scan is fast enough for 385 clips. In order to search tens of years of continuous archived video, we believe with timeline based search to reduce the search range and Ferret's filtering and indexing capability to speed up search, the query should still be answered in the order of seconds.

6. Related work

Retrieving continuous archived data is an active field. Traditional methods use various kinds of attributes and annotations to aid retrieval. Mylifebits uses extensive attributes and annotations to create links between video, audio and all personal information together. Lifelog presents a system using GPS, body sensor and voice annotation to index the video.

There have also been various projects working on content-based video retrieval. Marvel [12], VideoQ [13], and most notably various projects participating TRECVID workshop [14] use visual and audio features and apply the content based search technique to retrieve video clips. These projects mostly focus on commercial video clips, which pose different kind of challenges than personal continuous archived video.

Content-based audio retrieval research (e.g. Ellis and Lee [15]) has tended to focus more on automatic classification, clustering, and segmentation problems than on generic similarity search. Some work has been done towards defining similarity spaces for shorter-timescale sound effects and instrument tones (e.g. MARSYAS3D [16], Terasawa, Slaney, and Berger [17]). The problem of developing longer-timescale similarity metrics for music is also being actively studied (see e.g. Logan and Salomon [18], Vignoli and Pauws [19]).

Content-based image retrieval research has studied various visual features to find similar images. The initial QBIC [20] and many other content-based image search system surveyed by [21] studied quality of different image feature representations. More recently, region-based image retrieval like Blobworld [22] and local feature-based image retrieval like PCA-SIFT [Ke04] demonstrated better retrieval performance. Since these methods are much more computationally intensive, we did not adopt them in our video search system.

Research on home video segmentation [23][24] and organization [25] investigated techniques to organize home video which shares some similar characteristics with continuous archived video. Further investigation is needed to apply these methods to continuously archived video.

7. Conclusions

This paper presents the design and implementation of VFerret, a system that provides content-based similarity search for unlabeled continuous archived video data. Our initial evaluation with a simple benchmark shows that the system can perform high-quality content-based similarity search. While using visual and audio features individually can achieve 0.46 and 0.66 average precisions respectively, combining both can achieve average precision of 0.79.

The metadata overhead of the system is small. The current implementation uses about 1.4GB of metadata for content-based similarity search for one-year worth of continuous archived video data. This implies that it is already practical to implement content-based similarity search in a current computing device.

We plan to improve the VFerret system in several ways. The first is to use a better segmentation algorithm for personal, continuous archived video data. The second is to explore other visual feature extraction methods and distance functions. The third is to further evaluate the system with more sophisticated benchmarks and large datasets.

8. Acknowledgements

This project is sponsored in part by NSF grants CNS-0406415 and CNS-0509447, and by research grants from Google and Microsoft. We would like to thank Qin Lv and William Josephson for co-developing the Ferret toolkit and helpful discussions. We would also like to thank the reviewers for their helpful comments.

9. REFERENCES

[1] B. Vanneva, As We May Think, *The Atlantic Monthly*, 176(1), July 1945, 101-108.

[2] Q. Lv, W. Josephson, Z. Wang, M. Chrikar, and K. Li. *Ferret: A Toolkit for Content-Based Similarity Search of Feature-Rich Data.* Leuven, Belgium, EuroSys 2006.

[3] D. Sinitsyn, Davis Handysaw software, http://www.davisr.com/

[4] C. W. Ngo, T. C. Pong & H. J. Zhang, Motion Analysis and Segmentation through Spatio-Temporal Slices Processing, *IEEE Trans. on Image Processing*, vol. 12, no. 3, 2003.

[5] M. Stricker and M. Orengo. Similarity of color images. In *SPIE Conference on Storage and Retrieval for Image and Video Databases III*, volume 2420, pages 381-392, Feb. 1995.

[6] W. Ma and H. Zhang. Benchmarking of image features for content-based retrieval. In Proc. of *IEEE 32nd Asilomar Conf. on Signals, Systems, Computers*, volume 1, pages 253-257, 1998.

[7] D. P. W. Ellis and K.-S. Lee: Features for segmenting and classifying long-duration recordings of "personal" audio. In *Proc. ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing SAPA-04*, Korea, October 2004.

[8] The DEJA VIEW CAMWEAR, http://www.mydejaview.com/

[9] J Gemmell, G Bell, R Lueder, S Drucker, C Wong, MyLifeBits: fulfilling the Memex vision, *Proceedings of the tenth ACM international conference on Multimedia*, France, 2002.

[10] D. Tancharoen, T. Yamasaki, K. Aizawa, Practical Experience Recording and Indexing of Life Log Video, *2nd ACM workshop on Capture, Archival and Retrieval of Personal Experiences*, Singapore, 2005.

[11] T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu, An efficient k-means clustering algorithm: Analysis and implementation, *IEEE Trans. Pattern Analysis and Machine Intelligence,* 24 (2002), 881-892.

[12] J. R. Smith, S. Basu, C.-Y. Lin, M. Naphade, B. Tseng, "Interactive Content-based Retrieval of Video," *IEEE Intern. Conf. on Image Processing (ICIP-2002)*, Sept., 2002.

[13] S.-F. Chang, W. Chen, H. J. Meng, H. Sundaram, and D. Zhong, VideoQ: An Automatic Content-Based Video Search System Using Visual Cues, Proceedings, *ACM Multimedia '97 Conference*, Seattle, WA, November 1997.

[14] P. Over, T. Ianeva, W. Kraaij, and A.F. Smeaton. TRECVID 2005 - An Overview. In *Proceedings of TRECVID 2005*, 2005. NIST, USA.

[15] D. Ellis and K.S. Lee Minimal-Impact Audio-Based Personal Archives *First ACM workshop on Continuous Archiving and Recording of Personal Experiences CARPE-04*, New York, Oct 2004, pp. 39-47.

[16] G. Tzanetakis and P. Cook: MARSYAS3D: a prototype audio browser-editor using a large-scale immersive visual and audio display. In *Proc. International Conference on Auditory Display*, Espoo, Finland, July 2001.

[17] H. Terasawa, M. Slaney, and J. Berger: Perceptual distance in timbre space. In *Proc. International Conference on Auditory Display*, Limerick, Ireland, 2005.

[18] B. Logan and A. Salomon: A music similarity function based on signal analysis. In *Proc. IEEE International Conference on Multimedia and Expo*, Tokyo, Japan, 2001.

[19] F. Vignoli and S. Pauws: A music retrieval system based on user-driven similarity and its evaluation. In *Proc. International Symposium on Music Information Retrieval*, England, 2005.

[20] M. Flickner, H. Sawhney,W. Niblack, etc, Query by image and video content: the QBIC system, *Computer*, Volume: 28, pages 23-32 Sept, 1995.

[21] Smeulders, A.W.M. Worring, M. Santini, S. Gupta, A. Jain, R. Content-based image retrieval at the end of the early years, *IEEE Transactions on Pattern Analysis and Machine Intelligence,,* Dec, 2000.

[22] C Carson, M Thomas, S Belongie, JM Hellerstein, etc. Blobworld: A system for region-based image indexing and retrieval, *Third International Conference on Visual Information Systems,* 1999.

[23] S. WU, Y.-F. MA and H.-J. ZHANG (2005). Video quality classification based home video segmentation, *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, 4 pp.

[24] M. ZHAO, J. BU and C. CHEN (2003). Audio and video combined for home video abstraction, *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, 1520-6149. V-620-623 vol.625.

[25] X.-S. HUA, L. LU and H.-J. ZHANG (2004). Optimization-based automated home video editing system. *Circuits and Systems for Video Technology, IEEE Transactions on* 14(5): 572-583.