Locally Embedded Linear Subspaces for Efficient Video Indexing and Retrieval

Zhu Li[1], Li Gao[2], and Aggelos K. Katsaggelos[2]

[1]Multimedia Research Lab (MRL), Motorola Labs, Schaumburg, Illinois, USA
[2]Dept of Electrical Engineering & Computer Science (EECS), Northwestern University, Evanston, Illinois, USA

Keywords: Video Retrieval, High Dimensional Indexing, Component Analysis, Manifold Learning.

Abstract

Efficient indexing is a key in content-based video retrieval solutions. In this paper we represent video sequences as traces via scaling and linear transformation of the frame luminance field. Then an appropriate lower dimensional subspace is identified for video trace indexing. We also develop a trace geometry matching algorithm for retrieval based on average projection distance with a locally embedded distance metric. Simulation results demonstrated the high accuracy and very fast retrieval speed for the proposed solution.

1. Introduction

As video content grows exponentially, an efficient content based video shot retrieval that can handle very large size collections is becoming very important. In a typical search engine application scenario, the system would be required to identify the existence of the video shot in the collection and return the shot locations within a very short time. Extremely large size of video collections puts new challenges on video indexing/retrieval algorithms.

In previous work, various image/video features based indexing and retrieval solutions are reported, e.g, color based in [Ferman02], and [Yuan05], color and motion based in [Mezaris04], [Ngo02], [Zeng02], object level spatial-temporal feature based in [Chang98], and time interval statistics based in [Snoek05].

In video content indexing, a well-known problem is the "curse of dimensionality", when the feature space dimension is high, which is typically the case for image features like color, shape and textures, the indexing efficiency falls rapidly [Bohm01]. To resolve this problem a lower dimensional feature space with an appropriate matching metric need to be found. In [Li04], [Li05], we developed such a metric based on a trace representation of video sequence in luminance field's principal component space. However, the loss of information from dimension reduction may degrade the retrieval metric performance, therefore the conflicting requirement for indexing efficiency and retrieval accuracy on subspace dimensionality need to be addressed.

In this paper, motivated partly by various manifold learning algorithm, like principal component analysis (PCA) [Turk91], local linear embedding [Saul03], we develop an efficient indexing/retrieval solution utilizing the embedded low dimensional feature space and metric. First a global model is built to represent video sequence as traces. Then different embedded subspaces are trained for indexing and retrieval to achieve better efficiency and accuracy.

The paper is organized into the following sections, in Section 2, we briefly review the low dimensional feature space and metric developed for video retrieval, in Section 3, we develop indexing scheme based on kd-tree [Robinson81], and a fast retrieval algorithm with locally embedded metrics, in Section 4, simulation results in terms of both accuracy and speed are presented, in Section 5, we draw the conclusion and outline the future directions of our work.

2. Global Luminance Field Trace Model

In search of a lower dimensional feature space and metric for video shot indexing, instead of using the image features that may have obvious interpretation, like colour, shape and texture, we view video sequences as high dimensional traces spanned by the luminance field's variations over time. If a lower dimension representation of video trace can be found, then a space partition type indexing scheme and a trace geometry matching metric for retrieval can be developed.

We developed a global luminance field trace representation of the video sequence via scaling and PCA [Li04], [Li05], such that a video frame luminance field f_k is mapped into an d-dimensional point x_k, as,

$$x_k = A_N(S(f_k)) \quad (1)$$

where S is the scaling operator, which scales and stacks an input luminance field of W x H pixels into an image icon vector in R^{wxh}, with the desired icon size of w x h pixels. The d-dimensional PCA [Turk91] transform basis functions, $A_N = [a_1^T, a_2^T, \ldots, a_d^T]$, are obtained from the eigen vectors of the covariance matrix of video frames in a local neighbourhood, denoted as N. For global model, N consists of randomly sampled frames from a large collection. A_N identifies the d-dimensional subspace in R^{wxh} which preserves maximum amount of energy, that is,

$$A_N = \arg\max_A \sum_k (x_k - \bar{x})^2, \forall x_k \in N \quad (2)$$

The scaling and PCA transform reduce the dimensionality of the feature space, but inevitably introduce loss of energy/fidelity of the video trace. There is a trade off between the retrieval metric accuracy and the indexing and

computational efficiency obtained from dimensionality reduction.

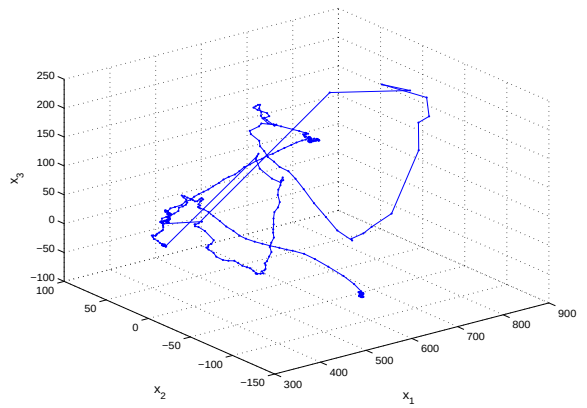

Figure 1. Video Trace Example

With a proper choice on scale [*w h*] and PCA dimension *d*, a global model can be built to map an *n*-frame video sequence into an *n*-point trace in some *d*-dimensional PCA space, preserving sufficient information in trace geometry for accurate retrieval. Example of a short 400 frame news video trace is plotted in an 3-d global space in Fig. 1. An efficient indexing and retrieval scheme is developed in the next section.

3. Indexing and Retrieval in Embedded Subspaces

The global video trace model may not be the most efficient for indexing and retrieval. For a "good" indexing space, we need the video traces to be evenly distributed in a much lower dimensional subspace, e.g., d_{indx}=2 to 6. While for good retrieval performance, we need a subspace with higher dimension, d_{rtrv}, to more accurately represent video traces. To address this conflict, we identify separate subspaces for indexing and retrieval purposes, with local optimisations.

3.1 kd-Tree Based Indexing

For indexing, we need video traces to be evenly distributed in a low d_{indx}-dimensional space, therefore a hierarchical data-partition type indexing scheme like kd-tree [Robinson81] can be applied to partition the video trace space into non-overlapping subspaces such that a binary tree structure can be built. At retrieval time, query clip trace will only need to match with a subset of video traces identified from the tree structure, instead of the whole collection, therefore improving the retrieval efficiency.

The PCA space is indeed, a good subspace for indexing purpose, because it finds the dimensions with maximum scatter of the data. For indexing purpose, we simply use the first *d* basis functions from global PCA, $A_{INDX} = [a_1^T, a_2^T, \ldots, a_d^T]$. The objective here is to partition the video traces space into hierarchical, non-overlapping subspaces such that a binary tree structure can be built and the actually matching of video shot traces can be limited to video traces in one or several leaf nodes, therefore improve the retrieval speed performance.

The covariance information obtained from global PCA process is utilized in the indexing. The indexing starts with a split of the whole collection along the maximum variance basis a_1, at the median value for all trace points projection on a_1. Then for left and right child, the maximum variance basis a_k is identified and the median value split along a_k is performed. The process is repeated in a breadth-first fashion and propagate down the tree structure until some pre-determined criteria for number of levels in the tree, and/or, number of frames in each leaf node is met. At each node, a minimum bounding box (MBB), V_{min}, $V_{max} \in R^d$ is also computed and stored with the split dimension *k*, and medium value *v*.

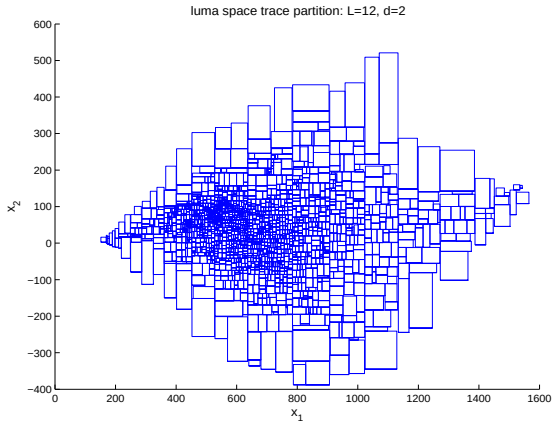

Figure 2. Indexing space partition example, *L*=12, *d*=2

An example of 200,000 frame video collection indexing space partition is illustrated in Fig. 2. The trace space dimension is *d=2*, and the depth of the kd-tree is *L=12*. There are total 2048 leaf node level MBBs, each contains roughly 98 frames.

With the kd-tree partition of the trace space, video sequences are further segmented by the MBB boundaries of each leaf node, in addition to semantically meaningful boundaries obtained from shot boundary detection. It is not surprising that a semantically meaningful shot can be further segmented by the leaf node MBB boundaries.

3.2 Retrieval with Leaf Node Embedded Metric

The retrieval consists of two steps. First, the area in the indexing space traversed by the query clip, i.e. the leaf node MBBs intersected by the querying clip trace, need to be identified. Second, for all video traces in the identified MBBs, compute the average projection distance between query clip trace and video collection traces, if the minimum projection distance is below certain threshold, then report a match ahs been found and return the location of matching, otherwise, report there is no match.

For a given point *x* in the query clip, the leaf node it belongs to is identified through the algorithm given below,

```
FindLeafNode(x, node) {
  IF IsLeafNode(node)
    //MBB check
```

```
        IF node.Vmin ≤ x ≤ node.Vmax
          RETURN(node)
        ELSE
          RETURN(NULL)
    // check the cutting plane
    k = node.cutDim; v = node.cutValue;
    IF  x_k < v
        FindLeafNode(x, node.LeftChild);
    ELSE
        FindLeafNode(x, node.RightChild);
}
```

For querying clips not exist in the collection, their trace points may or may not pass thru any leaf node MBBs. Therefore, an early rejection criteria is checked. As certain percentage of query trace points fall outside all leaf node MBBs, then the query clip is rejected as non-exist.

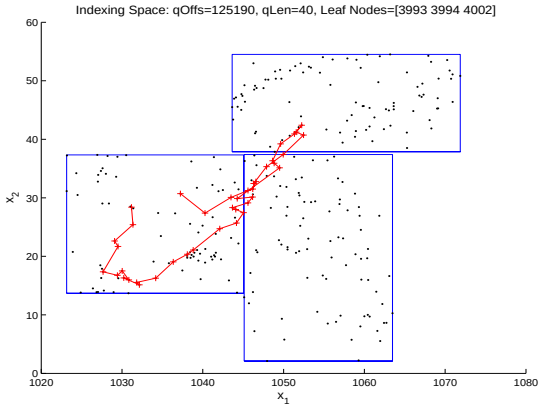

Figure 3. Leaf nodes traversed space example

An example 40-frame query clip and its traversed MBBs in the 2-d indexing space shown in Fig. 2 are plotted in Fig. 3. A total of 3 leaf nodes are traversed by this query clip, which is plotted as a connected line, while database side video frames are plotted as dots.

Once the MBBs traversed by the query clip are identified, the query clip need to be matched with all traces in those MBBs to find out if there's any match.

The trace geometry matching is based on the average projection distance metric. Let an m-frame query clip trace be $Q=[q_1, q_2, ..., q_m]$, and an n-frame video collection trace in traversed leaf nodes be $T=[t_1, t_2, ..., t_n]$, then the average projection distance, $d(Q,T)$, as a function of location k, is found by,

$$d(Q,T) = \frac{1}{m}\min_{k_1,k_2,\cdots,k_m}\sum_{j=1}^{m}\| q_j - t_{k_j} \| = d(k;Q,T) = \frac{1}{m}\min_{k}\sum_{j=1}^{m}\| q_j - t_{k+j-1} \| \quad (3)$$

Since this is a temporal sequence matching, the order of projection location indices need to be enforced, that is, k_1^*, k_2^*, …, k_m^* must be consecutive. Therefore the minimizer of Eq. (3) can be uniquely identified $k^*=k_1$.

In deciding the existence of a query clip, all traces in the traversed leaf nodes must be checked to compare the minimum average projection distance with a threshold d_{min},

$$k^* = \begin{cases} \arg\min_k d(k;Q,T^*), & if \ \min_{T_j \in N(Q)}\{d(Q,T_j)\} \le d_{\min} \\ not\ found, & else \end{cases} \quad (4)$$

where T^* is the trace with the minimum projection distance with Q.

The accuracy of the distance metric, $\|.\|$, used in Eq. (3) directly affect the retrieval accuracy. The global Euclidean distance metric in the indexing space does not offer enough accuracy, especially for large collections, because the video traces are not well separated, as shown in the example in Fig. 4. We need to find a new subspace/metric that more accurately captures the behaviour of the traces.

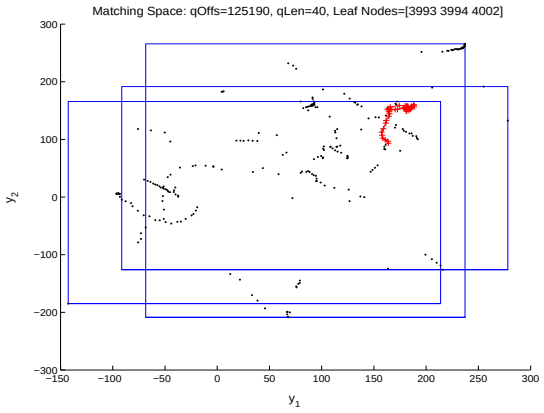

Figure 4. Embedded metric space example

Since a global model is not necessary for the retrieval, we can project video traces in each leaf node, l, to a new subspace, B_l, embedded in the global PCA model. For a given number of dimensions d_{rtrv}, $B_l=[b_1, b_2, ..., b_{d_{rtrv}}]$ is obtained thru a local PCA with frame points x in node l,

$$y = B_l x,\ \forall x \in N_l,\ s.t.\ B_l = \arg\max_B \sum_k (y_k - \bar{y})^2 . \quad (5)$$

Therefore the distance for original PCA space points x_1 and x_2 in leaf node l becomes,

$$\| y_1 - y_2 \| = (x_1 - x_2)^T B_l B_l^T (x_1 - x_2) . \quad (6)$$

By allow more dimensions in each leaf node than the indexing space, the metric in Eq. (6) is more accurate. The example embedded subspace for retrieval is shown in Fig. 4, with the same 40 frame query example in Fig. 3. Notice that in Fig. 4, the traces are better separated than in Fig. 3.

3.3 Computational Efficiency Analysis

Let the indexing efficiency of a kd-tree for a particular m-frame query clip Q be,

$$\eta(Q) = \frac{m}{N(Q)}, \quad (7)$$

where m is the length of the query clip, and $N(Q)$ is the total number of frames in leaf nodes traversed by Q. A perfect efficiency is 1.0. The time complexity comprises of two parts,

$$C(m,L,\eta) = C_1 + C_2 = t_1 O(mL) + t_2 O(m^2/\eta) . \quad (8)$$

The leaf nodes search part is roughly C_1, where m is the query clip length and L is the leaf node depth in the kd-tree, and t_1 reflects time spent on a scalar value comparison. The trace

matching part is C_2, which is a product of query clip length m, and number of leaf node trace points compared m/η . t_2 is the time to evaluate vector distance metric in Eq. (6). Notice that as video collection size doubles, to maintain roughly the same indexing efficiency, we need to double the number of leaf nodes. Therefore, L grows at $O(\log n)$, but C_2 shall stay relatively constant if indexing efficiency does not degrade drastically.

4. Simulation Results

We set up our simulation from sequences in the MPEG-7 and NIST TREC video data set. A total of 300,000 frames are used in training the global PCA model $A = [a_1^T, a_2^T, \ldots, a_d^T]$. At the luminance field scale of 8x6, we found that the global model preserves *90.15*% of energy at *d=12*. The data is divided into set 1 and 2 consist of 200K and 100K frames respectively.

For data set 1, we built two indexing structure with d_{indx}=2 and 3. Both indexing tree has L=12 levels and they took 168 and 176 seconds to built on a 2.4GHz/256MB Celeron notebook PC. Data set 2 is reserved for the negative queries.

The proposed method is very fast and robust in retrieval. For both indexing structure, we randomly set up 800 positive with query clips lengths m=15,30,45, from data set 1, and similarly 800 negative queries from data set 2. The performance is summarized in Tables 1 and 2.

Table 1. Indexing/Retrieval Performance, $L=12$, $d_{indx}=2$

Data Set	m	ErrRate (e/200)	T_1 (ms)	T_2 (ms)	mean(η)
1	*15*	*1/200*	*1.0*	*15.9*	*0.036*
1	*30*	*1/200*	*3.1*	*27.5*	*0.042*
1	*45*	*0/200*	*3.0*	*43.8*	*0.045*
1	*60*	*0/200*	*5.5*	*49.4*	*0.051*
2	*15*	*0/200*	*1.7*	*8.8*	*0.049*
2	*30*	*0/200*	*3.2*	*19.3*	*0.060*
2	*45*	*0/200*	*4.0*	*25.3*	*0.060*
2	*60*	*0/200*	*7.7*	*34.4*	*0.072*

Table 2. Indexing/Retrieval Performance, $L=12$, $d_{indx}=3$

Data Set	m	ErrRate (e/200)	T_1 (ms)	T_2 (ms)	mean(η)
1	*15*	*0/200*	*1.3*	*9.5*	*0.049*
1	*30*	*0/200*	*2.1*	*19.5*	*0.064*
1	*45*	*0/200*	*2.7*	*23.2*	*0.070*
1	*60*	*0/200*	*6.5*	*32.7*	*0.079*
2	*15*	*0/200*	*1.9*	*5.2*	*0.849*
2	*30*	*0/200*	*2.6*	*9.2*	*0.115*
2	*45*	*0/200*	*4.6*	*14.5*	*0.125*
2	*60*	*0/200*	*7.1*	*19.8*	*0.126*

Notice that the accuracy is very good in both cases, and the d_{indx}=3 case has better indexing efficiency. T_1 is the time spent on locating leaf nodes, and T_2 is the trace matching time. The embedded retrieval metric space dimension used is d_{rtrv}=2. The average total retrieval time range from 10.8 to 54.9 ms in both cases, this is a very fast performance compared with the state-of-art.

5. Conclusion & Future work

In this paper, we developed an efficient video indexing and retrieval scheme based on video traces. Locally embedded metrics are trained for better retrieval performance. The overall performance of the proposed solution is extremely fast and very accurate.

In the future, we will try larger video data size, in range of 100 hours, and add noise and corruption in query clip, to test the robustness of the solution.

References

[Bohm01] C. Bohm, S. Berchitold, and D. A. Keim, "Searching in High-Dimensional Spaces—Index Structures for Improving the Performance of Multimedia Databases", ACM Computing Survey, pp. 322-373, vol. 33(3), Sept., 2001.

[Chang98] S.-F. Chang, Chen, W., Meng, H.J., Sundaram, H., Di Zhong, "A fully automated content-based video search engine supporting spatiotemporal queryies", *IEEE Trans. on Circuits and System for Video Technology*, vol. 8, no.5, September 1998.

[Ferman02] M. Ferman, et al., "Robust color histogram descriptors for video segment retrieval and identification," *IEEE Trans. on Image Processing*, vol. 1, no. 5, May 2002

[Hsu02] Chiou-Ting Hsu, and Shang-Ju Teng "Motion trajectory based video indexing and retrieval", *Proceedings of Int'l Conference on Image Processing*, September, 2002, Rochester, NY.

[Li04] Z. Li, A. K. Katsaggelos, and B. Gandhi, "Fast video shot retrieval by trace geometry matching in principal component space", *Proceedings of IEEE Int'l Conference on Image Processing* (ICIP), pp. 24-27, October, 2004.

[Li05] Z. Li, A. K. Katsaggelos, and B. Gandhi, ."Fast video shot retrieval based on trace geometry matching", *IEE Proceedings on Vision, Image and Signal Processing*, pp. 367-373, vol. 152(3), June, 2005.

[Mezaris04] V. Mezaris, I. Kompatsiaris, N. V. Boulgouris, and M. G. Strintzis, "Real-time compressed-domain spatiotemporal segmentation and ontologies for video indexing and retrieval", *IEEE Trans. on Circuits and Systems for Video Technology*, pp. 606-621 vol. 14(5), May 2004.

[Ngo02] C.-W. Ngo, T.-C. Pong, and H.-J. Zhang, "On clustering and retrieval of video shots through temporal slices analysis", IEEE Trans. On Multimedia, vol. 4(4), pp. 446-458, Dec, 2002.

[Robinson81] J. Robinson, "The k-d-b-tree: A search structure for large multidimensional dynamic indexes", *Proc. Of ACM SIGMOD Int'l Conference on Management of Data*, pp. 10-18, 1981.

[Saul03]L.K. Saul and S.T. Roweis, "Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifolds", *J. Machine Learning Research*, vol. 4, pp. 119-155, 2003.

[Snoek05] C. G. M. Snoek, and M. Worring, "Multimedia event-based video indexing using time intervals", IEEE Trans. On Multimedia, vol. 7(4), pp. 638-647, August, 2005.

[Turk91] M. Turk and A.P. Pentland, "Face Recognition Using Eigenfaces," *IEEE Conf. Computer Vision and Pattern Recognition*, 1991.

[Yuan05] J. Yuan, L-. Y. Duan Qi Tian, and C. Xu, "Fast and Robust Short Video Clip Search Using an Index Structure", *Proceedings of 6th ACM Workshop on Multimedia Info Retrieval* (MIR), 2005.

[Zeng02] W. Zeng, W. Gao, and D. Zhao, "Video indexing by motion activity maps", *Proceedings of Int'l Conference on Image Processing*, September, 2002, Rochester, NY.