FAST SIMILARITY SEARCH ON VIDEO SIGNATURES

Sen-ching S. Cheung *

Center for Applied Scientific Computing
Lawrence Livermore National Laboratory
P.O. Box 808, L-561, Livermore, CA 94551
sccheung@llnl.gov

Avideh Zakhor

Department of EECS
University of California
Berkeley, CA 94720
avz@eecs.berkeley.edu

ABSTRACT

Video signatures are compact representations of video sequences designed for efficient similarity measurement. In this paper, we propose a feature extraction technique to support fast similarity search on large databases of video signatures. Our proposed technique transforms the high dimensional video signatures into low dimensional vectors where similarity search can be efficiently performed. We exploit both the upper and lower bounds of the triangle inequalities in approximating the high-dimensional metric, and combine this approximation with the classical PCA to achieve the target dimension. Experimental results on a large set of web video sequences show that our technique outperforms Fastmap, Haar wavelet, PCA, and Triangle-Inequality Pruning.

1. INTRODUCTION

Thanks to widespread availability of broadband connections and decreasing cost of disk storage, it is now commonplace to publish, broadcast, or stream video sequences over the Internet. As video content becomes more popular on the web, there is a growing need to develop tools for analyzing, searching, and organizing visually similar video sequences. In the development of such tools, we are faced with two major algorithmic challenges: how to efficiently measure the similarity between two video sequences, and how to identify video sequences similar to a given query out of possibly millions of entries on the web. In [1], we introduce a class of techniques called ViSig for efficient video similarity measurement. The ViSig method summarizes a video sequence into a compact video signature, consisting of a small number of representative feature vectors from the video. Compared to other summarization techniques, video signatures are simple to compute, robust against temporal re-ordering, and capable of identifying similar video sequences regardless of their length. In this paper, we consider the problem of searching for signatures similar to a user-defined query in a very large database.

The naive approach of sequential search is typically too slow to handle large databases. Faster-than-sequential solutions have been extensively studied by the database community. Elaborate data structures, collectively known as the Spatial Access Methods (SAM), have been proposed to facilitate similarity search [2, 3]. Most of these methods, however, do not scale well to high dimensional metric spaces [4]. One strategy to mitigate this problem is to design a feature extraction mapping to map the original metric space to a low-dimensional space where a SAM structure can be efficiently applied. The approach of combining feature extraction with SAM is called GEneric Multimedia INdexIng (GEMINI) [3].

*This work was supported by NSF grant ANI-9905799, AFOSR contract F49620-00-1-0327, and ARO contract DAAD19-00-1-0352. This work was done while S.-C. Cheung was with University of California at Berkeley. Part of this work was also performed under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-Eng-48.

In this paper, we propose a novel feature extraction mapping to be used in GEMINI for fast similarity search on signature data.

The most commonly used feature extraction is the Principal Component Analysis (PCA), which is optimal in approximating Euclidean distance [5]. If the underlying metric is not Euclidean, PCA is no longer optimal and more general schemes need to be used. One such technique is the Fastmap, a heuristics algorithm that approximates general metric by Euclidean distance [6]. Another class of techniques construct mappings based on distances between the high-dimensional vectors and a set of random vectors [7, 8, 9, 10]. These kinds of "random mappings" have been shown to possess certain favorable theoretical properties [7, 8]. Such mappings, however, are very complex, and effectively require the computations of all pairwise distances between entries in the database. A more practical version has been proposed in [9] for protein matching. An even simpler version, called the Triangle-Inequality Pruning (TIP), has been proposed for similarity search on image databases [10]. TIP exploits the lower bound of the triangle inequality in approximating the high-dimensional metric. Our proposed technique improves upon TIP by taking into account both the upper and lower bounds offered by the triangle-inequality. In addition, it takes advantage of the classical PCA technique to achieve any user-defined target dimension.

This paper is organized as follows: in Section 2, we briefly review the ViSig method and the GEMINI approach. The proposed feature extraction mapping and its performance evaluation on a large database of signatures are presented in Section 3.

2. REVIEW OF VISIG AND GEMINI

We begin with a brief overview of the ViSig method [1]. We assume that each video is represented by a set of high-dimensional feature vectors, X, from a metric space $(F, d(\cdot,\cdot))$[1]. The metric function $d(\cdot,\cdot)$ is used to measure the visual dis-similarity between two feature vectors. In this paper, we use four concatenated 178-bin HSV color histograms as our feature vector, each representing a quadrant of a video frame, and l_1 as the metric between two histograms. In order to reduce the complexity in comparing two video sequences, the ViSig method summarizes each video X in the database into a signature X_S, which consists of the feature vectors in X that are closest to a set of *seed vectors* $S = \{s_1, s_2, \ldots, s_m\}$:

$$X_S = (g_X(s_1), g_X(s_2), \ldots, g_X(s_m)) \quad \text{where } g_X(s) = \arg\min_{x \in X} d(x, s). \tag{1}$$

The central idea behind the ViSig method is that if two video clips share a large fraction of similar feature vectors, their signature vectors with respect to the same seed vectors are likely to be similar as well. The seed vectors are feature vectors randomly sampled

[1] In the remainder of this paper, we refer to video and its feature vectors interchangeably.

from a training set that resembles the target video data under consideration. The robustness of an individual signature vector can be evaluated by the following function [1]:

$$Q(g_X(s)) = \min_{x \in X,\, d(x, g_X(s)) > \epsilon_C} d(x, s) - d(g_X(s), s), \quad (2)$$

where ϵ_C is the maximum distance between similar vectors within the video. In order to guarantee the existence of robust signature vectors, we typically set the number of signature vectors in a signature, m, to be fairly large, but use only the most robust, or highest-ranked, m' vectors in comparing two signatures. Specifically, two signature X_S and Y_S are compared using the following asymmetric *signature distance*:

$$d_{\text{sig}}(X_S, Y_S) = \text{median}_{i=1,\ldots,m'} d(g_X(s_{j[i]}), g_Y(s_{j[i]})), \quad (3)$$

where $g_X(s_{j[1]}), \ldots, g_X(s_{j[m']})$ have the m' largest $Q(\cdot)$ values among all the signature vectors in X_S. We declare Y_S to be similar to X_S if $d_{\text{sig}}(X_S, Y_S)$ is less than or equal to the *similarity threshold* ϵ. Given a query signature X_S, the goal of a *signature similarity search* is to identify all signatures Y_S in a given database that are similar to X_S. To provide a fast solution to this problem, we propose an approach based on a generic technique called GEMINI which tackles the specific problem of *metric-space similarity search* [3, ch. 7].

Given a query x and a database, D, of feature vectors, the goal of a metric-space similarity search is to identify the following:

$$A(x; \epsilon) = \{y \in D : d(x, y) \leq \epsilon\} \quad (4)$$

It is easy to see that the signature similarity search on X_S can be solved by applying the metric-space similarity search on each of the m' top-ranked signature vectors in X_S [11, ch. 4]. Rather than computing $A(x; \epsilon)$ by a sequential search, GEMINI first uses a *feature extraction mapping* $\mathcal{T}$ to map feature vectors into a very low dimensional *range metric space* $(F', d'(\cdot, \cdot))$. A similarity search on the transformed query $\mathcal{T}(x)$ is performed to identify the *candidate set* $C(x; \epsilon')$ defined below:

$$C(x; \epsilon') = \{y \in D : d'(\mathcal{T}(x), \mathcal{T}(y)) \leq \epsilon'\} \quad (5)$$

ϵ' is called a *pruning threshold*, which depends on ϵ, $\mathcal{T}$ and the data. As mentioned in Section 1, such a low-dimensional search problem can be efficiently solved by any SAM method. The final step of GEMINI to identify those vectors in $C(x, \epsilon')$ that are truly within ϵ of x:

$$A'(x; \epsilon, \epsilon') = \{y \in C(x; \epsilon') : d(x, y) \leq \epsilon\}. \quad (6)$$

By applying GEMINI to each top-ranked vectors in X_S, we can define $A_S(X_S; \epsilon)$, $C_S(X_S; \epsilon')$, and $A'_S(X_S; \epsilon, \epsilon')$ for similarity search on a database D_S of signatures that are analogous to those defined in (4), (5), and (6) respectively.

GEMINI is more efficient than sequential search if a typical candidate set is small enough so that few full metric computations are required in the last step of GEMINI. To assess the average complexity reduction of GEMINI over a large set of query signatures R, we measure the *Pruning* parameter defined below. It is based on the relative difference in the total number of full metric computations between GEMINI and sequential search:

$$\text{Pruning}(\epsilon') = 1 - \frac{\sum_{X_S \in R} |C_S(X_S; \epsilon')|}{(|R| \cdot |D_S|)}. \quad (7)$$

As suggested in Equation (7), a high level of pruning can be achieved by making candidate sets small. On the other hand, small candidate sets may adversely affect the *accuracy* of GEMINI, which is defined below:

$$\text{Accuracy}(\epsilon') = \frac{\sum_{X_S \in R} |A'_S(X_S; \epsilon, \epsilon')|}{\sum_{X_S \in R} |A_S(X_S; \epsilon)|}. \quad (8)$$

Our goal is to design a feature extraction mapping $\mathcal{T}$ that provides a reasonable trade-off between pruning and accuracy. In the next section, we introduce a novel design of $\mathcal{T}$ for signature data.

3. FEATURE EXTRACTION FOR SIGNATURE

Our proposed mapping consists of two steps: first, each signature vector is mapped into a particular form of low-dimensional range vector called a *projection vector*. Second, classical PCA is applied to transform the projection vector into an *index vector* of even lower dimension, as specified by the user. The motivation behind our proposed mapping is explained in Section 3.1. In Section 3.2, we present experimental results to compare our scheme with other techniques proposed in the literature.

3.1. Proposed feature extraction

Let x_s and y_s be the signature vectors in signatures X_S and Y_S that correspond to the same seed vector $s \in S$. Consider the following m-dimensional vector,

$$\mathcal{T}(x_s) = (d(x_s, s_1), d(x_s, s_2), \ldots, d(x_s, s_m)), \quad (9)$$

as a feature extraction mapping of x_s. We are interested in this particular formulation because of two reasons: first, it makes use of quantities that have already been computed in (1). Second, the distance $d(x_s, y_s)$ can be related to the coordinates of $\mathcal{T}(x_s)$ and $\mathcal{T}(y_s)$ by the triangle inequalities:

$$|d(x_s, s_i) - d(y_s, s_i)| \leq d(x_s, y_s) \leq d(x_s, s_i) + d(y_s, s_i), \quad i = 1, 2, \ldots, m \quad (10)$$

The above inequalities are instrumental in designing the feature extraction mapping. The mapping $\mathcal{T}(\cdot)$ and its variations have been previously proposed in the literature for feature extraction [7, 8, 9, 10]. These techniques typically use a l_p-metric as the range metric between $\mathcal{T}(x_s)$ and $\mathcal{T}(y_s)$. For $p = 1, 2, \ldots$, the l_p metric is defined as follows:

$$l_p(\mathcal{T}(x_s), \mathcal{T}(y_s)) = \left(\tfrac{1}{m} \sum_{i=1}^{m} |d(x_s, s_i) - d(y_s, s_i)|^p\right)^{1/p} \quad (11)$$

On the other hand, l_∞ is defined as,

$$l_\infty(\mathcal{T}(x_s), \mathcal{T}(y_s)) = \max_{i=1,\ldots,m} |d(x_s, s_i) - d(y_s, s_i)| \quad (12)$$

We use a normalization factor of $1/m$ in the definition of l_p so that it has the same order of magnitude as the l_∞-metric. All the l_p-metric functions are composed of different powers of the absolute differences between the coordinates of $\mathcal{T}(x_s)$ and $\mathcal{T}(y_s)$, i.e. $|d(x_s, s_i) - d(y_s, s_i)|$ for $i = 1, \ldots, m$. These absolute differences appear only in the lower-bound half of the triangle inequalities in (10). By using a simple experiment, we can demonstrate that better pruning-accuracy trade-off can be achieved by combining both the upper and lower bounds of the triangle inequalities.

Our experiment is based on sampling random pairs of video sequences from a database of 46,331 web video sequences called SIGDB [11, ch. 4]. We sample 100,000 pairs and generate their signature vectors with respect to a random s chosen from a set of $m = 100$ diverse seed vectors, also sampled from SIGDB using the seed vector generation algorithm described in [1]. For each pair of signature vectors x_s and y_s, we compute $d(x_s, y_s)$ and their distances with respect to all m seed vectors. The distribution of $d(x_s, y_s)$ is shown in Figure 1 as a function of a single lower and upper bound, defined as follows: for the lower bound, we take the maximum over all the individual lower bounds in (10), which is identical to $l_\infty(\mathcal{T}(x_s), \mathcal{T}(y_s))$. For the upper bound, we use a similar approach and take the minimum of the individual upper bounds in (10) to form an $\alpha(\cdot)$ function:

$$\alpha(\mathcal{T}(x_s), \mathcal{T}(y_s)) = \min_{i=1,2,\ldots,m} [d(x_s, s_i) + d(y_s, s_i)] \quad (13)$$

In Figure 1, data points of different shapes and shades represent metric values in different ranges. We separate all data points into two classes: the crosses and circles are the *small-metric* points, and the dots in different shades correspond to the *large-metric*

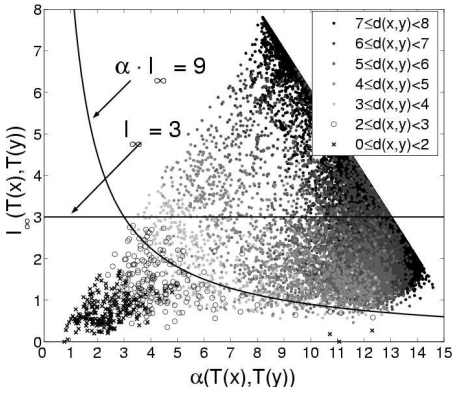

Fig. 1. *Distribution of the metric $d(x_s, y_s)$ for 100,000 random pairs of signature vectors in the coordinates of $\alpha(\mathcal{T}(x_s), \mathcal{T}(y_s))$ and $l_\infty(\mathcal{T}(x_s), \mathcal{T}(y_s))$.*

points. Small-metric points are those with metric values smaller than $\epsilon = 3.0$, which we experimentally find to be a reasonable value to identify visually similar video clips. The goal of a similarity search is to separate the small-metric points from the large-metric ones. If we use $l_\infty(\cdot)$ as the range metric, a typical candidate set based on the inequality $l_\infty(\mathcal{T}(x_s), \mathcal{T}(y_s)) \leq \epsilon'$ will include all the points below a horizontal line at level ϵ'. This is, as a matter of fact, the TIP scheme proposed in [10]. An example of such a set with $\epsilon' = 3$ is shown in Figure 1. Even though all the small-metric points are within the candidate set, many of the large-metric points are also erroneously included as they have small $l_\infty(\cdot)$ values. It is clear, based on the shape of the distribution of the small-metric points, that a better separating function should combine both $l_\infty(\cdot)$ and $\alpha(\cdot)$. One possible choice is to base on their product, $\beta(\cdot)$:

$$\beta(\mathcal{T}(x_s), \mathcal{T}(y_s)) = \alpha(\mathcal{T}(x_s), \mathcal{T}(y_s)) \cdot l_\infty(\mathcal{T}(x_s), \mathcal{T}(y_s)) \quad (14)$$

As shown in Figure 1, even though the candidate set defined by $\beta(\mathcal{T}(x_s), \mathcal{T}(y_s)) \leq 9$ misses a few small-metric points, it excludes a much larger set of large-metric points than $l_\infty(\cdot)$. Nevertheless, we cannot directly use $\beta(\cdot)$ because it is not a true metric function.

The $\beta(\cdot)$ function in (14) is defined as the product of $\alpha(\cdot)$ and $l_\infty(\cdot)$, which represent the aggregate bounds of all the inequalities in (10). Rather than using the two aggregate bounds, it is simpler to form a metric by using the product of the bounds from the individual inequalities as follows:

$$[d(x_s, s_i) + d(y_s, s_i)] \cdot |d(x_s, s_i) - d(y_s, s_i)| = |d(x_s, s_i)^2 - d(y_s, s_i)^2|, \quad i = 1, 2, \ldots, m \quad (15)$$

Note that Equation (15) is in the form of an absolute difference. While absolute differences also appear in the definitions of l_p metrics in Equation (11), the one in (15) is the absolute difference of *squares* of $\mathcal{T}(\cdot)$'s coordinates. Thus, it is conceivable to propose a new metric $\zeta(\cdot)$ that combines l_p with this absolute difference of *squares* of coordinates as follows:

$$\zeta(\mathcal{T}(x_s), \mathcal{T}(y_s)) = \left(\frac{1}{m} \cdot \sum_{i=1}^{m} [d(x_s, s_i)^2 - d(y_s, s_i)^2]^p \right)^{1/p} = l_p(\mathcal{P}(x_s), \mathcal{P}(y_s)) \quad (16)$$

where $\mathcal{P}(x_s)$ is defined as:

$$\mathcal{P}(x_s) = (d(x_s, s_1)^2, d(x_s, s_2)^2, \ldots, d(x_s, s_m)^2) \quad (17)$$

We call $\mathcal{P}(x_s)$ the projection vector of x_s, and the collection of projection vectors for all the signature vectors in a signature a *projection*. In Section 3.2, we demonstrate experimentally that using the l_2 metric on $\mathcal{P}(x_s)$ produces much better pruning and accuracy trade-off than using the l_∞ metric or the $\beta(\cdot)$ function in (14) on $\mathcal{T}(x_s)$.

Besides the superior pruning and accuracy performance, there is another reason for choosing to apply the l_2 metric on the projection vectors. The dimension of a projection vector is $m = 100$ in the case of SIGDB. Despite the fact that m is smaller than the dimension of our original feature vectors, i.e. $178 \times 4 = 712$, it is still much larger than what most SAM structures can handle. If l_2 metric is used between two projection vectors, we can reduce the dimension of the projection vectors with minimum distortion by applying the classical PCA technique [5]. We call the resulting lower-dimensional vector the *index vector*, and the collection of index vectors for all the signature vectors in a signature the *Index*. In the next section, we compare our proposed indices with other dimension reduction schemes proposed in the literature.

3.2. Experimental results

We first justify the use of the projection mapping with l_2 metric based on the experimental results on full signature data. The database consists of signatures of all the video sequences in SIGDB with respect to the same set of $m = 100$ seed vectors. A random query set of 1000 signatures are drawn from the database, and used as the set R in Equations (7) and (8) for computing pruning and accuracy for different values of ϵ'. For the signature similarity search, we set ϵ to be 3.0 and use $m' = 6$ for computing the signature distance in (3). We measure the accuracy and pruning for three different feature extraction mappings used within GEMINI: a) the "lower-bound" scheme based on the mapping $\mathcal{T}(\cdot)$ in (9) and the l_∞-metric; (b) the "product" scheme based on $\mathcal{T}(\cdot)$ and the $\beta(\cdot)$ function defined in (14); and (c) the projection mapping scheme based on $\mathcal{P}(\cdot)$ in (17) and the l_2-metric. The resulting plots of pruning versus accuracy, with ϵ' varying across each plot, are shown in Figure 2. A good feature extraction mapping should achieve pruning and accuracy that are as close to one as possible. As shown in the figure, our proposed scheme clearly out-performs both the "lower-bound" and "product" schemes by achieving much higher pruning at the same accuracy level. Also, as expected, the "product" scheme out-performs the "lower-bound" scheme as the "product" scheme exploits both the upper and lower bounds of the triangle inequality.

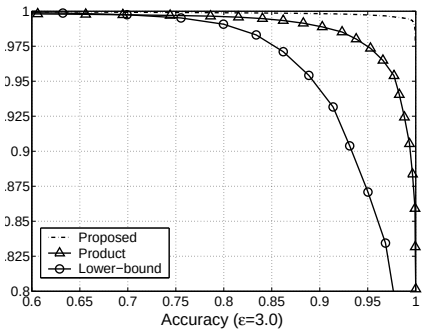

Fig. 2. *Pruning-versus-Accuracy plots for the "lower-bound", the "product" and the proposed schemes.*

In the following experiments, we apply PCA to the projection vectors, and compare the resulting index vectors with other approaches in terms of their accuracy and pruning trade-off. These approaches include: a) PCA – while in our proposed scheme, PCA is applied on the projection vectors, it can also be directly applied onto the 712-dimensional color histogram feature vectors

for dimension reduction; b) Fastmap, as described in [6]; and c) Haar wavelet on color histogram as described in the MPEG-7 standard [12]. Since most of the schemes require training data to generate the mappings, we arbitrary split SIGDB into two halves – we call one half the "training" SIGDB, which is used to build the mapping, and the other half the "testing" SIGDB, which is used for the actual testing. In order to ensure the suitability of incorporating these schemes into GEMINI, we focus on very low dimensional index vectors. We test all the schemes for dimensions two, four and eight. The corresponding pruning-accuracy plots are shown in Figures 3(a) through (c). These plots are generated by the same procedure used in the first experiment. As seen, our proposed scheme results in the best performance in all the dimensions tested, followed by Haar, Fastmap and PCA. The gain of the proposed scheme over the second best scheme, however, diminishes as the dimension increases.

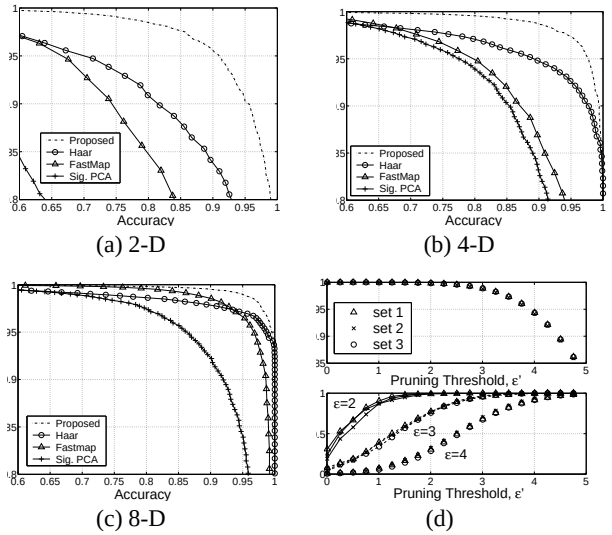

Fig. 3. *(a)-(c) Pruning-versus-accuracy plots for two-, four- and eight-dimensional spaces. (d) Pruning and Accuracy versus pruning threshold for three independent sets of queries.*

In applying the feature extraction scheme in a fast similarity search, we need to choose a particular value of pruning threshold ϵ' in order to compute the candidate set. Given the target dimension, accuracy, and pruning, one possible approach is to set ϵ' to a value that attains the particular level of performance in a previously completed experiment. Thus, an important question to answer is whether the relationship between ϵ' and the corresponding pruning and accuracy extends to other queries. To answer this question, we measure the pruning and accuracy, as defined in Equations (7) and (8), for three independent sets of random queries. Each set has 1000 signatures randomly drawn from the testing SIGDB. For each set of queries, different values of pruning and accuracy are measured by varying ϵ'. The experiment is also repeated for three different values of ϵ, namely 2, 3, and 4. The resulting plots of pruning and accuracy versus ϵ' for the three query sets and different values of ϵ are shown in Figure 3(d). As shown in the figure, there is little variation in the amount of pruning among the three sets. There is some variation in the accuracy for small ϵ, but the variation diminishes as ϵ becomes larger. The maximum differences in accuracy among the three sets over all possible values of ϵ' are 0.12, 0.06, and 0.04 for $\epsilon = 2$, 3, and 4 respectively. These fluctuations are small compared to the high accuracy required by typical applications.

We conclude this section with a number of speed measurements for the above four schemes on a particular platform. The experiments are run on a Dell PowerEdge 6300 Server with four 550MHz Intel Xeon processors and 1 Giga-bytes of memory. As all the tests are run under a single thread, only a single processor is used. The testing SIGDB, which contains 23,206 signatures, each consisting of 100 vectors, and their corresponding 8-dimensional indices are first loaded inside the memory. 100 queries are randomly sampled for testing. No SAM structure is implemented and a simple sequential search is used for the indices. Pruning thresholds are chosen, based on the previous experiments, to hit the 90% accuracy level for similarity searches at $\epsilon = 3.0$. As a reference, we also measure the performance of sequential search on signatures with no feature extraction. The results are tabulated below:

Schemes	Sequential	Proposed	Fastmap	Haar	PCA
Accuracy	1.00	0.89	0.91	0.92	0.89
Index time (ms)	-	131 $\pm$ 0.8	131 $\pm$ 1.5	152 $\pm$ 1.3	130 $\pm$ 1.4
Refine time (ms)	6730 $\pm$ 35	33 $\pm$ 8	75 $\pm$ 11	123 $\pm$ 28	401 $\pm$ 75
Candidate Size/query	-	109 $\pm$ 27	262 $\pm$ 39	428 $\pm$ 97	1386 $\pm$ 257

The Index time is the time required for the sequential search on indices to identify the candidate sets. The averages and their standard error at 95% confidence interval are shown. As the Sequential scheme does not use any indices, no number is reported. The proposed scheme, Fastmap, and PCA all use the l_2 distance on range vectors and thus, result in roughly the same index time. Haar requires slightly larger index time for its l_1 distance computation. The refine time is the time required to perform the full signature distance computations on the candidate sets, and is proportional to the size of the candidate sets as shown in the last row. Our proposed scheme outperforms all other feature extraction schemes in refinement time. The large standard error in the refinement time is due to the variation in the size of candidate sets. Combining the index time and refinement time, the proposed scheme is roughly 41 times faster than the sequential search on signatures.

4. REFERENCES

[1] S.-C. Cheung and A. Zakhor, "Efficient video similarity measurement with video signature," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 1, pp. 59–74, Jan. 2003.

[2] H. Samet, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley, 1989.

[3] C. Faloutsos, *Searching Multimedia Databases by Content*, Kluwer Academic Publishers, 1996.

[4] R. Weber, H.-J. Schek, and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces," in *Proceedings of the 24th International Conference on Very-Large Databases (VLDB'98)*, New York, NY, USA, Aug. 1998, pp. 194–205.

[5] J. Hotelling, "Analysis of a complex of statistical variables into principal components," *J. of Educational Psychology*, vol. 24, pp. 417–441, 1933.

[6] C. Faloutsos and King-Ip Lin, "Fastmap: a fast algorithm for indexing, datamining and visualization of traditional and multimedia datasets," in *Proceedings of ACM-SIGMOD*, May 1995, pp. 163–174.

[7] J. Bourgain, "On lipschitz embedding of finite metric spaces in hilbert space," *Israel Journal of Mathematics*, vol. 52, pp. 46–52, 1985.

[8] N. Linial, E. London, and Y. Rabinovich, "The geometry of graphs and some of its algorithmic applictions," *Combinatorica*, vol. 15, no. 2, pp. 215–45, 1995.

[9] G. Hristescu and M. Farach-Colton, "Cluster-preserving embedding of proteins," Tech. Rep. DIMACS 99-50, Rutgers University, Piscataway, USA, 1999.

[10] A. P. Berman and L. G. Shapiro, "A flexible image database system for content-based retrieval," *Computer Vision and Image Understanding*, vol. 75, no. 1/2, pp. 175–195, July/August 1999.

[11] S.-C. Cheung, *Efficient Video Similarity Measurement and Search*, Ph.D. thesis, University of California, Berkeley, 2002, http://www-video.eecs.berkeley.edu/papers/cheungsc/PhdThesis.pdf.

[12] L. Cieplinski, S. Jeannin, M. Kim, and J.-R. Ohm, "Visual working draft 4.0," Tech. Rep. W3522, ISO/IEC JTC1/SC29/WG11, July 2000.