Search Methods for Fast Matching of Video Fingerprints within a Large Database

Miss. LaxmiGupta[1], Prof. M.B Limkar[2], Prof. Sanjay. M Hundiwale[3]
[1](*M.E Student EXTC,ARMIET College Of Engineering, Sapgaon,* Mumbai , INDIA)
[2](*Associate Prof. Department Of Electronics, Terna College of Engineering, Nerul ,Mumbai, INDIA)*
[3](*Associate Prof. Department of EXTC,ARMIET, College of Engineering, Sapgaon, Mumbai ,INDIA)*

ABSTRACT: *Fingerprint identification has been a great challenge due to its complex search of database. Fingerprinting system is the ability to detect and/or reject a query video within a large database in fast & reliable fashion. This paper proposes an efficient fingerprint search algorithm for fast matching of fingerprints within a large video database. Here we evaluate the performance of proposed Inverted File based search & Cluster based search algorithm and compare with that of exhaustive search method when applied to fingerprints derived by TIRI-DCT. It can be seen that proposed Cluster based approach is faster than that the inverted file search method. We thus adopt the cluster based algorithm as the search engine for our copy detection system for secure version of proposed fingerprinting algorithm. It not only greatly speeds up the search process but also improves the retrieval accuracy.*

Keywords: *Cluster Search, Fingerprinting, Inverted search, Retrieval accuracy.*

I. INTRODUCTION

In real-world applications, the size of online video databases can reach tens of millions of videos, which translates into a very large fingerprint database size. This means that even if the fingerprint of a query video can be extracted very quickly, searching the fingerprint database to find a match may take a long time. For online applications, however, a reliable match should be found in almost real-time. Therefore fingerprint matching forms a practical bottleneck for online fingerprinting systems.There is a large body of research on fast and reliable similaritysearch. Muja *et al.* have conducted a comprehensive study on state-of-the-art similarity search algorithms in Euclidian spaces [3]. There are a number of studies on similarity search on binary spaces as well [1], [4], [5]. However, only few papers in the video fingerprinting area have considered the fast search aspect in their design. A simple exhaustive search method has a complexity of $0(N)$, where N is the number of the fingerprints in the database. As an example of a fast search algorithm, Oostveen proposed a search algorithm for their video fingerprinting algorithm, based on the inverted file technique[1][2] . Proposed search methods inverted file based similarity search and cluster based similarity search methods are modified version of search algorithm that was proposed by Oostveen [1] .

II. SEARCH METHODS FOR MATCHING OF VIDEO FINGERPRINTS

Inverted File Based Similarity Search

The binary fingerprints are divided into small non overlapping blocks of m bits. These small blocks are called as words as shown in figure 1. Words are then used to create an inverted file from the fingerprints of the database. All fingerprints have equal lengths, so the inverted file can be represented as a table of size $2^m \times n$ where n is the number of words in a fingerprint of length$n = L/m$. The horizontal dimension of this table refers to the position of a word inside a fingerprint, and the vertical dimension corresponds to possible values of the word. To generate this table, start with the first word of each fingerprint as shown in figure 2, and add the index of

the fingerprint to the entry in the first column corresponding to the value of this word. Continue this process for all the words in each fingerprint and all the columns in the inverted file table.

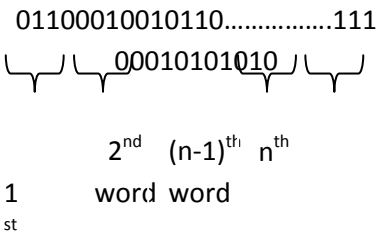


Fig.1 Dividing Fingerprint into Words

A sample of the generated inverted file is shown in figure 2. Entry $< i, j >$ in the table is a list of the indices of all the fingerprints that theirjth word is wordi. To find a query fingerprint in the database, first the fingerprint is divided into n words (of m bits). The query is then compared to all the fingerprints that start with the same word. The indices of these fingerprints are found from the corresponding entry in the first column of the inverted file table. The Hamming distance between these fingerprints and the query is then calculated. If a fingerprint has a Hamming distance of less than some predefined threshold, it will be announced as the match. If no such match is found, the procedure is repeated for the fingerprints that have exactly the same second word as the query's second word (the indices of these fingerprints are read from the corresponding entry in the second column of the inverted file table). This procedure is continued until a match is found or the last word is examined. When no match is found in the end, it is stated that the query does not belong to the database.

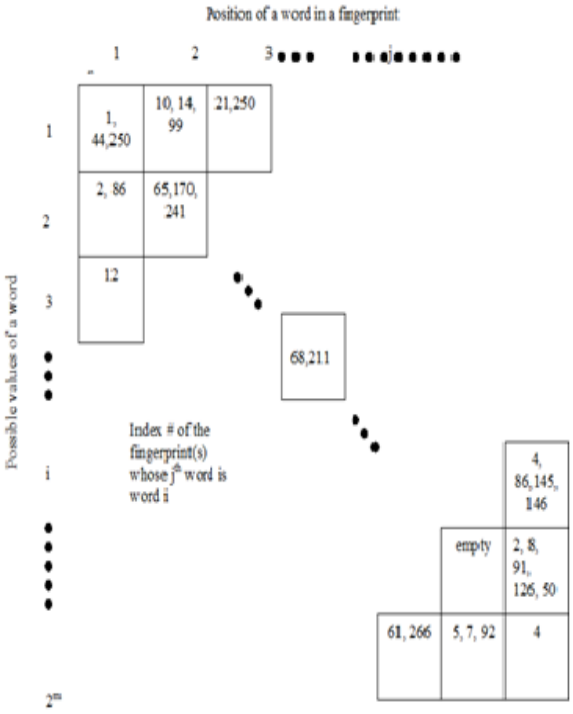


Fig.2 Sample Inverted file for finger print database

The above inverted file based similarity search algorithm can be used forfinding the nearest neighbor of any binary fingerprint within a database. To show the validity of the assumption that two similar fingerprints have at least one exactly matching word.Assuming that the fingerprinting algorithm is perfect, i.e. no two perceptually different videos have fingerprints that are closer than a Hamming distance of th the algorithm is guaranteedto find the correct match, ifth $<$ n.If th $>=$ n then the algorithm may generate false negativesdue to the fact that there may exist a mismatch in each word.To calculate the probability of false negative, we assume that afingerprint exists in the database that has a

Hamming distanceofth with the query fingerprint. Probability of false rejection P_{FR} is given by

$$P_{FR} = \sum_{k=0}^{k_{max}} -1^{k} \binom{n}{k} \frac{\binom{L-k*m}{th}}{\binom{L}{th}}$$

The steps involved in inverted file based similarity searchare as follows.

Step 1

Binary fingerprints are divided into n words of equal bits.

Step 2

The horizontal dimension of the table represents the position of words.

Step 3

The vertical dimension of the table represents the possible values of words.

Step 4

Add index for each word of the fingerprint to the entry in column corresponding to the value of the word.

Step 5

Hamming distance is calculated between fingerprints in the database and query

fingerprint.

Step 6

If the distance is less than the threshold value, then the query video will be announced as matching.

Step 7 Otherwise, it will be announced as not matching.

In inverted file based similarity search the Hamming distance is calculated between fingerprints in the database and query fingerprint. If the distance is less than the threshold value, then the query video will be announced as matching, otherwise as not matching with the database. The Hamming distance between two strings of bits (binary integers) is the number of corresponding bit positions that differ. Hamming distance is a small portion of a broader set of formulas used in information analysis. Specifically, Hamming's formulas allow computers to detect and correct errors on their own. This can be found by using XOR on corresponding bits or equivalently, by adding corresponding bits (base 2) without a carry.

For example, in the two bit strings that follow:

A	0 1 0 0 1 0 1 0 0 0
B	1 1 0 1 0 1 0 1 0 0
A XOR B	1 0 0 1 1 1 1 1 0 0

The Hamming distance (H) between these 10-bit strings is 6, the number of 1's in the XOR string. In general, steps in the calculation of hamming distance are as follows.

Step 1

Ensure the two strings are of equal length. The Hamming distance can only be calculated between two strings of equal length. String 1: "1001 0010 1101" String 2: "1010 0010 0010"

Step 2

Compare the first two bits in each string. If they are the same, record a "0" for that bit. If they are different, record a "1" for that bit. In this case, the first bit of both strings is "1," so record a "0" for the first bit.

Step 3

Compare each bit in succession and record either "1" or "0" as appropriate. String 1: "1001 0010 1101" String 2: "1010 0010 0010" Record: "0011 0000 1111"

Step 4

Add all the ones and zeros in the record together to obtain the Hamming distance.
Hamming distance =
0+0+1+1+0+0+0+0+1+1+1+1 = 6

In short for binary strings a and b the Hamming distance is equal to the number of one in a XOR b. The Hamming distance is named after Richard Hamming. Other distance measures are Euclidean distance, Chess board distance, City block distance, Camberra distance.

Cluster Based Similarity Search

Cluster based similarity search is another similarity search algorithmfor binary fingerprints. The main idea is to use clusteringto reduce the number of queries that are examined within thedatabase. By assigning each fingerprint to one and only onecluster (out ofk clusters), the fingerprints in the database will be clustered into knonoverlapping groups. To do so, a centroidis chosen for each cluster, termed the cluster head. A fingerprintwill be assigned to cluster if it is closest to this cluster's head as shown in figure 3.

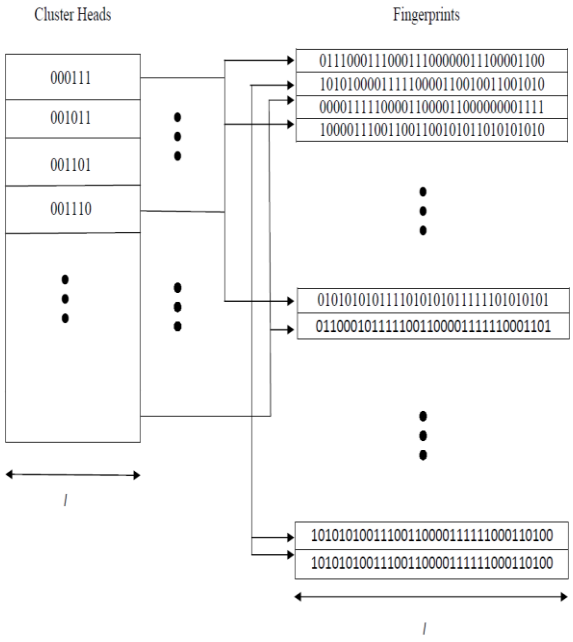


Fig. 3 Cluster Based Similarity Search for TIRI-DCT

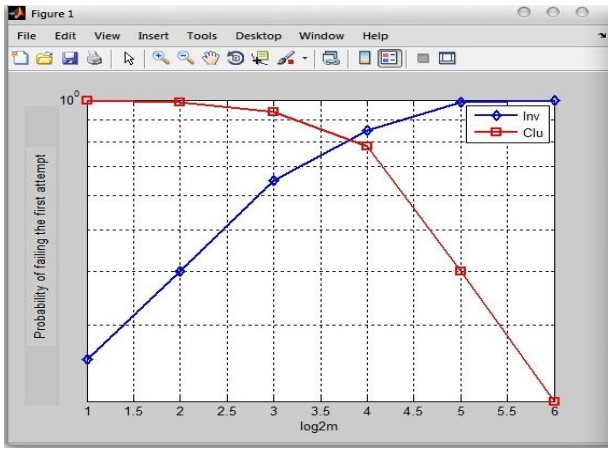
To determine if a query fingerprint matches afingerprint in the database, the cluster head closest to the queryis found. All the fingerprints (of the videos in the database) belongingto this cluster are then searched to find a match, i.e., theone which has the minimum Hamming distance (of less than acertain threshold) from the query. If a match is not found, thecluster that is the second closest to the query is examined. Thisprocess continues until a match is found or the farthest clusteris examined. In

the latter case, the query is declared to be out ofthe database. The cluster heads should be chosen such that a small changein the fingerprint does not result in the fingerprint being assignedto another cluster. In general setting, cluster heads (centers) as all the binary vectors with lengthl $\ll$ L are chosen. To assign a fingerprint to a cluster, the fingerprint is firstdivided into segments (words) of length m. Each word is then represented by one bit in thel -bit cluster head, depending on the majority of word's bit values; for example, it is represented by 1, if it has more than $m/2$ 1's and it is represented by 0, if it has less than $m/2$ 1's. Equivalently, each bit of the cluster head can be replicated m times and the Hamming distance between the expanded $L = m * l$ bit version of all the cluster heads and the fingerprint is calculated. The cluster head closest to the fingerprint is then assigned to that fingerprint. Thus the worst case complexity of cluster based similarity search is$O(N)$. Here while clustering fingerprints constraint based clustering technique is used. The constraint based clustering technique allows for specification of user constraints. Depending on the nature and application constraint clustering problems are classified into following categories.

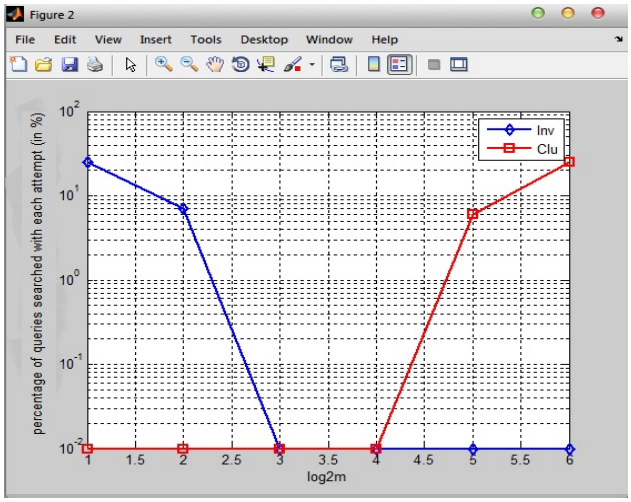
- Constraint on individual object
- Obstacle objects as constraints
- Clustering parameters as constraints
- Constraints imposed on each individual cluster

There are different clustering techniques like hierarchical algorithms, partitioning algorithms, grid based algorithm constraint based algorithm, evolutionary algorithms, scalable clustering algorithms etc. Partitioning algorithms are classified into different categories like k-means, k-medoids, density based algorithm, probabilistic algorithm etc.

III. RESULTS



(a) Probability of falling the first attempt



(b) Approximate percentage of queries searched with each attempt.

Fig. 4 Comparing the run time of the clustering based search method with that of the inverted file based method for different values of m.

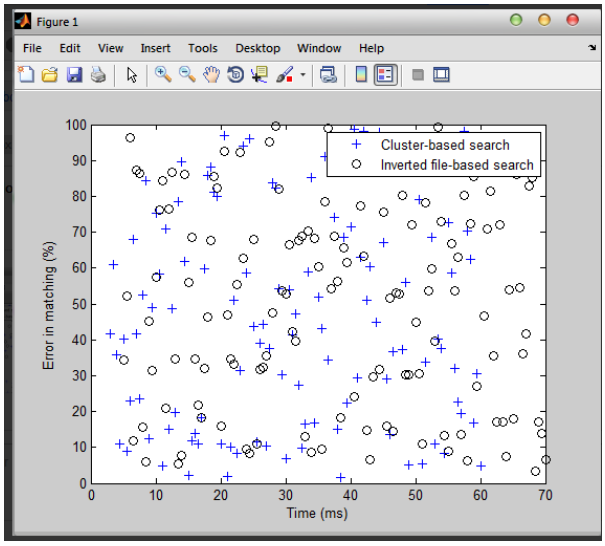


Fig. 5 Error versus search time for different search algorithms applied on fingerprints derived by TIRI-DCT.

IV. Conclusion

This paper proposes a fingerprinting system for video copy detection. This paper proposes an efficient fingerprint search algorithm for fast matching of fingerprints within a large video database. Here we evaluate the performance of proposed Inverted File based search & Cluster based search algorithm and compare search method. We will also evaluate our proposed fast search methods when applied to other fingerprinting methods.

It can be seen that proposed Cluster based approach is faster than that the inverted file search method. We thus adopt the cluster based algorithm as the search engine for our copy detection system for secure version of proposed fingerprinting algorithm. It not only greatly speeds up the search process but also improves the retrieval accuracy.& This algorithm maintains a good performance for different attacks on video signals, including noise addition, changes in brightness/contrast, rotation, spatial/temporal shift, and frame loss.

V. Future work

As part of our futurework, we will conduct a detailed analytical study of the security of fingerprinting algorithms including the one proposed in this paper. As another part of our future work, we will carry an extensive comparison study to compare our fingerprinting algorithms toother state-of-the-art algorithms.We will also evaluate our proposed fast search methods when applied to other fingerprinting methods. We also plan to study the performance of the system in the presence of some other attacks, such as cropping, and logo insertion.

REFERENCES

[1] JOostveen, T. Kalker, and J. Haitsma, "Feature extraction and a database strategy for video fingerprinting," in *Proc. Int. Conf. Recent Advances in Visual Information Systems (VISUAL)*, London, U.K., 2002, pp. 117–128, Springer-Verlag.

[2]IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 6, NO. 1,MARCH 2011 213 A Robust and Fast Video Copy Detection System Using Content-Based Fingerprinting by-Mani MalekEsmaeili, MehrdadFatourechi, and RababKreidieh Ward, *Fellow, IEEE*

[3] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. Int. Conf. Computer VisionTheory and Applications*, 2009, pp. 331–340.

[4]J. A. Haitsma, A. A. C. M. Kalker, C. P. M. J. Baggen, and J. C . Oostveen, Generating and matching hashes of multimedia content US 2002/0178410, Nov. 2002.

[5] M. L. Miller, "Audio fingerprinting: Nearest neighbour search in high dimensional binary spaces," in *IEEE Workshop on 2002, MultimediaSignal Processing, 2002*, 2002, pp. 182–185.