Fingerprinting

Audio and Video Detection Systems

Speaker: Matteo Pozzetti

Supervisor: Laura Peer

01 April 2015

Overview

Introduction

Fingerprinting

Content based fingerprinting:

- Algorithm that maps a multimedia object into a compact digital summary (set of hashes)

Fingerprinting

Content based fingerprinting:

- Algorithm that maps a multimedia object into a compact digital summary (set of hashes)
- Used to retrieve such an object in a database

Fingerprinting

Content based fingerprinting:

- Algorithm that maps a multimedia object into a compact digital summary (set of hashes)
- Used to retrieve such an object in a database
- Applications:

Fingerprinting

Content based fingerprinting:

- Algorithm that maps a multimedia object into a compact digital summary (set of hashes)
- Used to retrieve such an object in a database
- Applications:
  - Search engines (Shazam)

Fingerprinting

Content based fingerprinting:

- Algorithm that maps a multimedia object into a compact digital summary (set of hashes)
- Used to retrieve such an object in a database
- Applications:
  - Search engines (Shazam)
  - Copyright protection (YouTube)

Fingerprinting

Content based fingerprinting:

- Algorithm that maps a multimedia object into a compact digital summary (set of hashes)
- Used to retrieve such an object in a database
- Applications:
  - Search engines (Shazam)
  - Copyright protection (YouTube)

Requirements

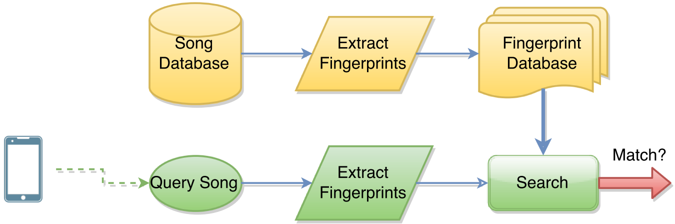
A fingerprint algorithm must be noise resistant and computationally efficient.

Table of Contents

Introduction

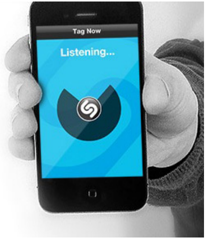

SHAZAM®
AT&T
11:32 AM
92%
Listening

Process


Song Database
Extract Fingerprints
Fingerprint Database
Query Song
Extract Fingerprints
Search
Match?

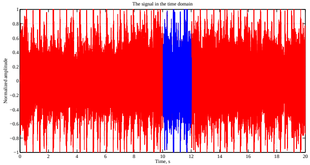
Goals

- The length of the recorded sample should not be more than 15 seconds long



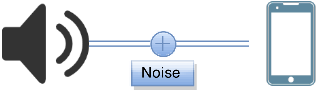
Goals

- The length of the recorded sample should not be more than 15 seconds long
- The sample can come from any portion of the original audio track



Goals

- The length of the recorded sample should not be more than 15 seconds long
- The sample can come from any portion of the original audio track
- Robust



Hashing

Transformations to the frequency domain

Fourier transform

Fourier transform: maps the signal $x(t) : \mathbb{R} \to \mathbb{C}$ into

$$X(f) = \int_{\infty}^{\infty} x(t) e^{-j2\pi ft} dt$$

which decomposes the function into its frequencies.

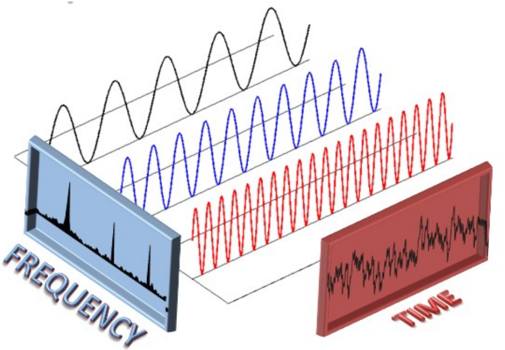
Transformations to the frequency domain

Fourier transform

Fourier transform: maps the signal $x(t) : \mathbb{R} \to \mathbb{C}$ into

$$X(f) = \int_{\infty}^{\infty} x(t) e^{-j2\pi ft} dt$$

which decomposes the function into its frequencies.



Short-time Fourier Transform

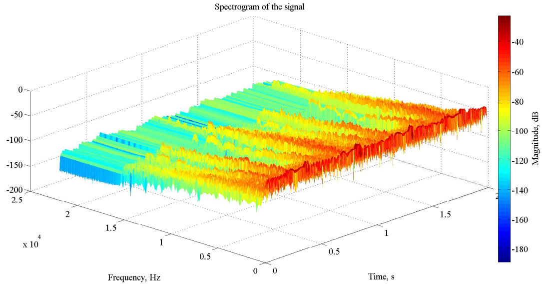
STFT is a Fourier-related transform used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time.

$$x(t) \xmapsto{STFT} X(\tau, f) = \int_{-\infty}^{\infty} x(t) w(t-\tau) e^{-j2\pi ft} dt$$

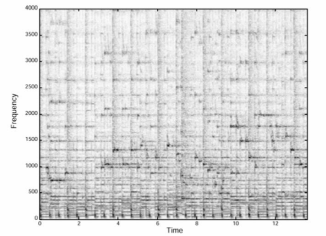
Short-time Fourier Transform

STFT is a Fourier-related transform used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time.

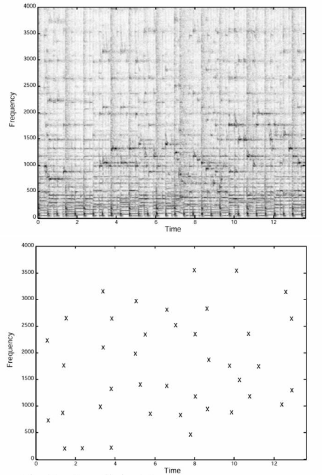
$$x(t) \xrightarrow{STFT} X(\tau, f) = \int_{-\infty}^{\infty} x(t)w(t-\tau)e^{-j2\pi ft}dt$$



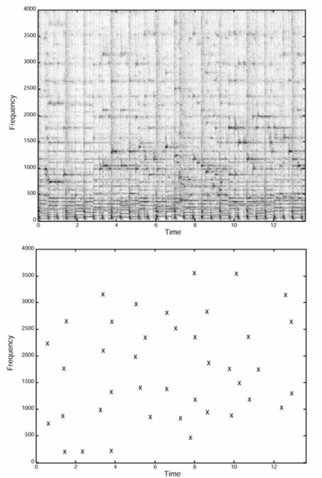
Constellation and Hashing

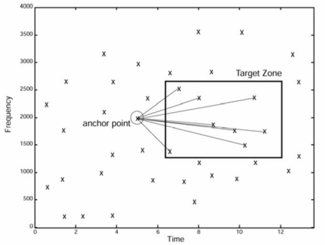


Constellation and Hashing

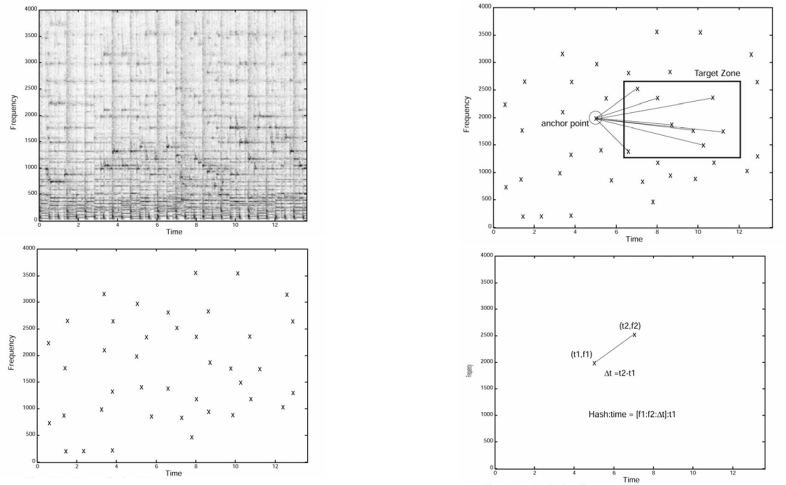


Constellation and Hashing





Constellation and Hashing

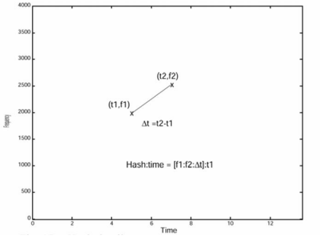


Hashing

Each anchor point is sequentially paired with points within its target zone.

Hashing

Each anchor point is sequentially paired with points within its target zone.



$$(f_1, f_2, \Delta t) \rightarrow \text{hash}$$

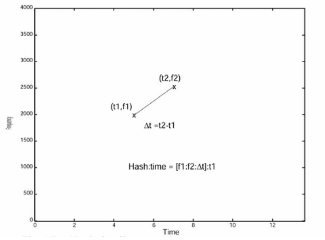
30 bits hash:

- 10 BITS frequency anchor
- 10 BITS frequency target
- 10 BITS time difference

HASHING

Each anchor point is sequentially paired with points within its target zone.

$$(f_1, f_2, \Delta t) \rightarrow \text{hash}$$



30 bits hash:

- 10 BITS frequency anchor
- 10 BITS frequency target
- 10 BITS time difference

Each hash is associated with the time offset:

$$\text{hash:offset} = (f_1, f_2, \Delta t) : t_1$$

Searching and Scoring

Searching and Scoring

We take all matching hashes of a given song and we plot their time offset

Searching and Scoring

We take all matching hashes of a given song and we plot their time offset

DB song	Query song
h:t	h:τ

Searching and Scoring

We take all matching hashes of a given song and we plot their time offset

DB song	Query song
h:t	h:τ

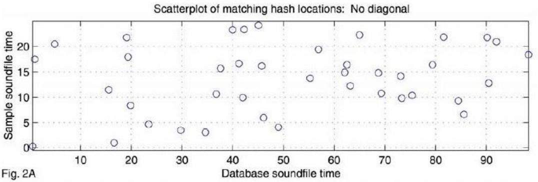


Fig. 2A

Searching and Scoring

We take all matching hashes of a given song and we plot their time offset

DB song	Query song
h:t	h:τ

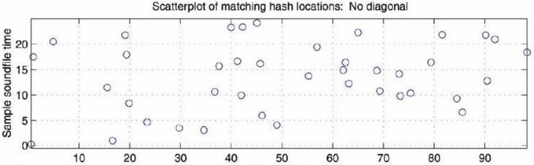


Fig. 2A

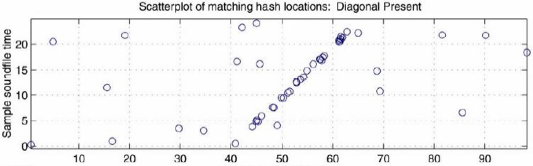


Fig. 3A

Scoring

- Matching features: $t_k = \tau_k + \text{constant}$

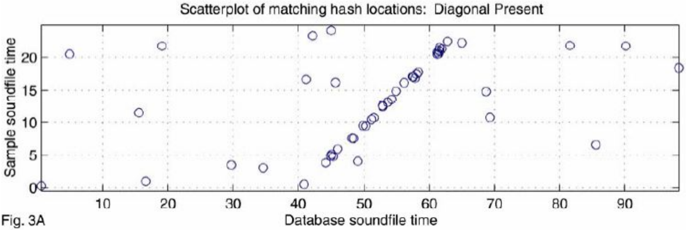


Fig. 3A

Scoring

- Matching features: $t_k = \tau_k + \text{constant}$
- For each (t_k, τ_k) in the scatter-plot, we calculate:

$$\Delta_k = t_k - \tau_k$$

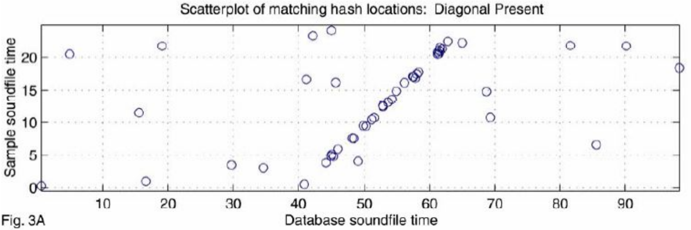


Fig. 3A

Scoring

- Matching features: $t_k = \tau_k + \text{constant}$
- For each (t_k, τ_k) in the scatter-plot, we calculate:

$$\Delta_k = t_k - \tau_k$$

- Then we plot a histogram of these Δ_k

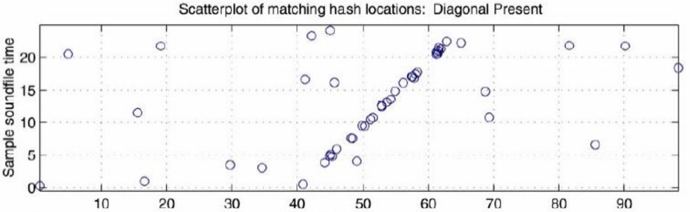


Fig. 3A

Scoring

Track that does not match the sample

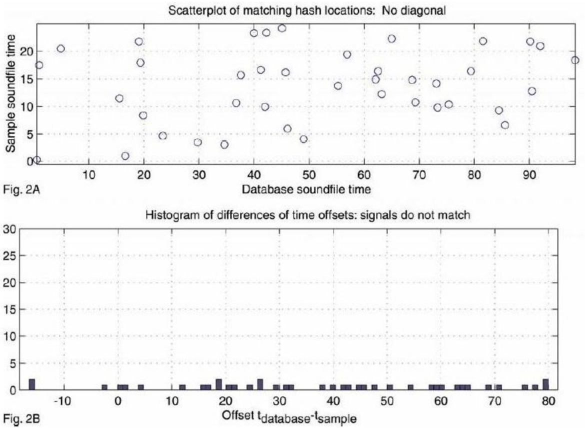


Figure : No diagonal in the scatter-plot.

Scoring

Track that matches the sample

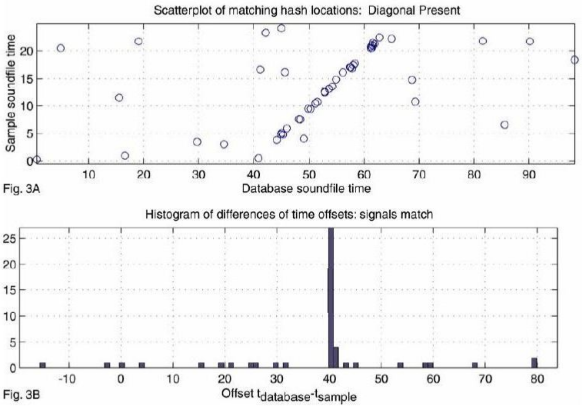
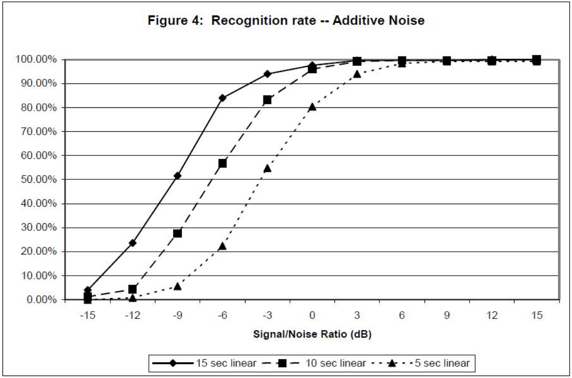


Figure : Diagonal in the scatter-plot: cluster in the histogram.

Results

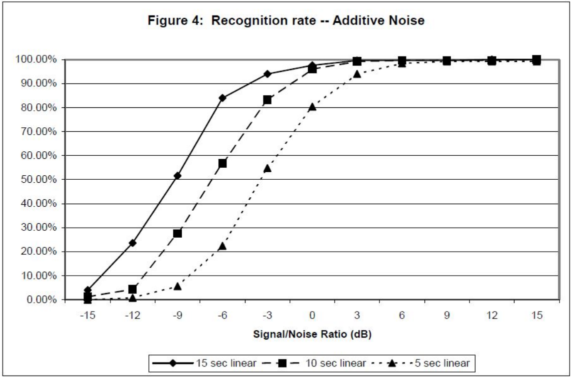
Results

- The algorithm performs well with significant levels of noise



Results

- The algorithm performs well with significant levels of noise



- For a database of 20 about thousand tracks the search time is in the order of 5 – 500 milliseconds.

Table of Contents

Introduction

Introduction

Copy detection system

The purpose of this system is to detect copyrights violations.

Introduction

Copy detection system

The purpose of this system is to detect copyrights violations.

A fingerprint should be

- Robust to content-preserving distortions

Introduction

Copy detection system

The purpose of this system is to detect copyrights violations.

A fingerprint should be

- Robust to content-preserving distortions
- Discriminant

Introduction

Copy detection system

The purpose of this system is to detect copyrights violations.

A fingerprint should be

- Robust to content-preserving distortions
- Discriminant
- Easy to compute

Introduction

Copy detection system

The purpose of this system is to detect copyrights violations.

A fingerprint should be

- Robust to content-preserving distortions
- Discriminant
- Easy to compute
- Compact

Image and Video Representations

- RBG

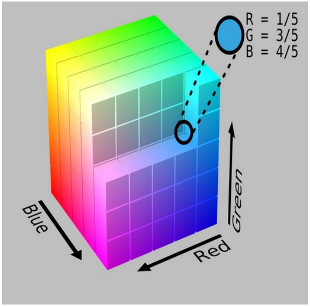


Image and Video Representations

- RBG
  - Colors represented as a combination of red, green and blue (3D-space)

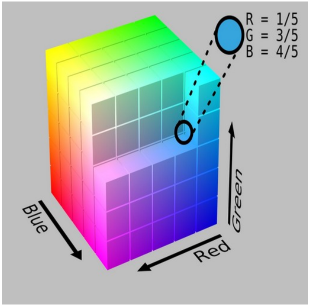


Image and Video Representations

- RBG
  - Colors represented as a combination of red, green and blue (3D-space)
  - Each component is an integer value in $[0, 255]$

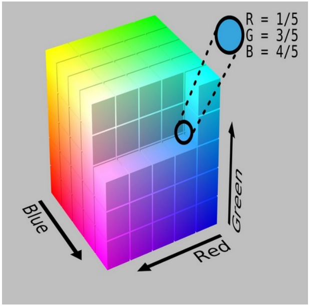


Image and Video Representations

- RBG
  - Colors represented as a combination of red, green and blue (3D-space)
  - Each component is an integer value in $[0, 255]$
- YUV

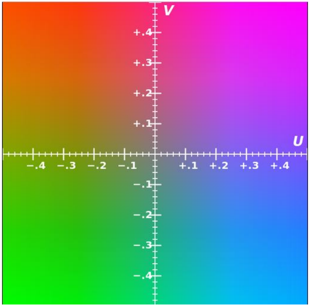
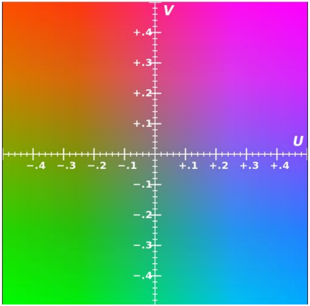


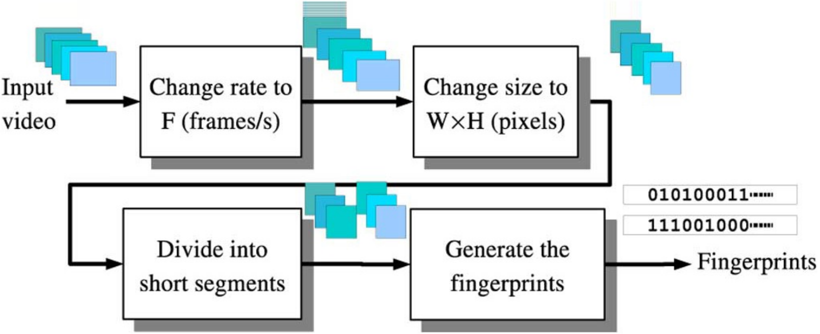
Image and Video Representations

- RBG
  - Colors represented as a combination of red, green and blue (3D-space)
  - Each component is an integer value in $[0, 255]$
- YUV
  - Y is the luminance component, U and V are the chrominance components.



Preprocessing

Preprocessing



Discrete cosine transform

Discrete cosine transform

- The DCT is similar to the discrete Fourier transform: it transforms a signal or image from the spatial domain to the frequency domain.

Discrete cosine transform

- The DCT is similar to the discrete Fourier transform: it transforms a signal or image from the spatial domain to the frequency domain.
- It is a widely used tool in image and video processing and compression!

Discrete cosine transform

- The DCT is similar to the discrete Fourier transform: it transforms a signal or image from the spatial domain to the frequency domain.
- It is a widely used tool in image and video processing and compression!
- It is possible to extract some relevant information from the frequency representation

Discrete cosine transform
Formula

The two-dimensional DCT of an M-by-N matrix A is defined as follow:

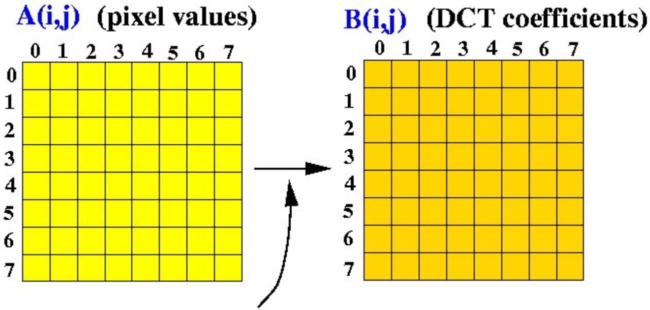
$$B_{pq} = \alpha_p \alpha_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} A_{mn} \cos \frac{\pi(2m+1)p}{2M} \cos \frac{\pi(2n+1)q}{2N}$$

$$\alpha_p = \begin{cases} \frac{1}{\sqrt{M}}, & p = 0 \\ \sqrt{\frac{2}{M}}, & 1 \leq p \leq M-1 \end{cases} \qquad \alpha_q = \begin{cases} \frac{1}{\sqrt{N}}, & q = 0 \\ \sqrt{\frac{2}{N}}, & 1 \leq q \leq M-1 \end{cases}$$

where $0 \leq p \leq M-1$ and $0 \leq q \leq N-1$ are the matrix indexes.

Discrete cosine transform

Example



$$B(k_1,k_2) = \frac{1}{4} C(k_1)C(k_2) \sum_{i=0}^{7} \sum_{j=0}^{7} A(i,j) \cos\left[\frac{\pi \times k_1}{2 \times 8}(2i+1)\right] \cos\left[\frac{\pi \times k_2}{2 \times 8}(2j+1)\right]$$

$$C(k) = \frac{1}{\sqrt{2}} \quad \text{if } k = 0$$

$$C(k) = 1, \text{ otherwise}$$

3D-DCT

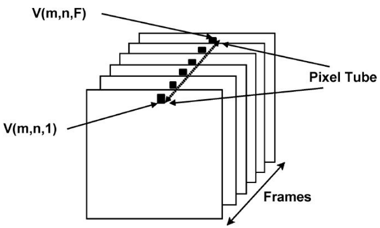
3D Discrete cosine transform

3D-DCT is a DCT applied to a 3-dimensional matrix.

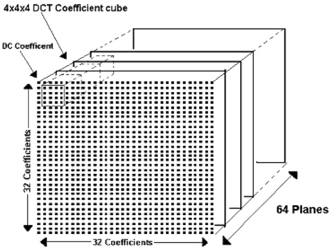
3D Discrete cosine transform

3D-DCT is a DCT applied to a 3-dimensional matrix.

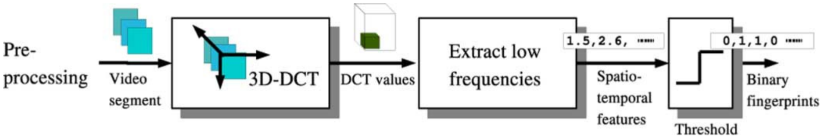
Time domain



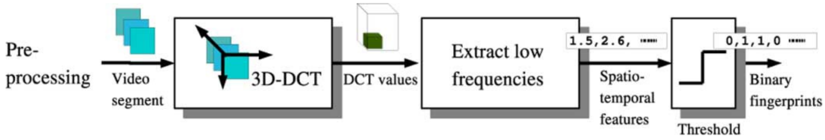
Frequency domain

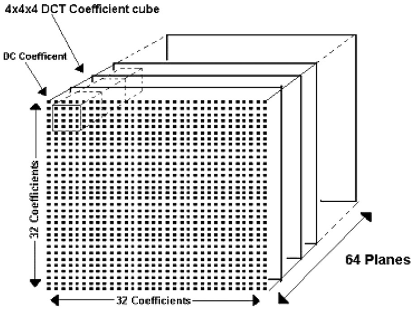


Fingerprinting using 3D-DCT

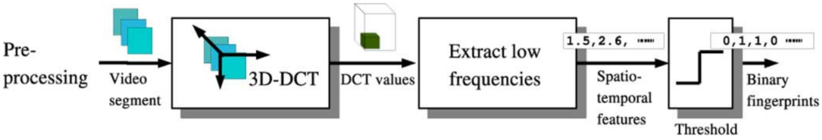


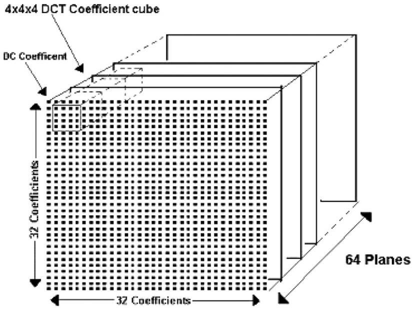
Fingerprinting using 3D-DCT





Fingerprinting using 3D-DCT





Binary fingerprints are derived by thresholding the low-frequency coefficients of the transform; the threshold is the median value of the selected coefficients.

Fingerprinting using TIRIs

TIRI

Temporally Informative Representative Image

A TIRI contains spatial and temporal information of a short segment of a video sequence.

TIRI

Temporally Informative Representative Image

A TIRI contains spatial and temporal information of a short segment of a video sequence.

$$I'_{m,n} = \sum_{k=1}^{J} w_k I_{m,n,k}$$

where $I_{m,n,k}$ is the luminance value of the (m,n)th pixel in a set of J frames and w_k is a weighting factor.

TIRI

Weighting factor

Experiments were made with different w: constant, linear, exponential.

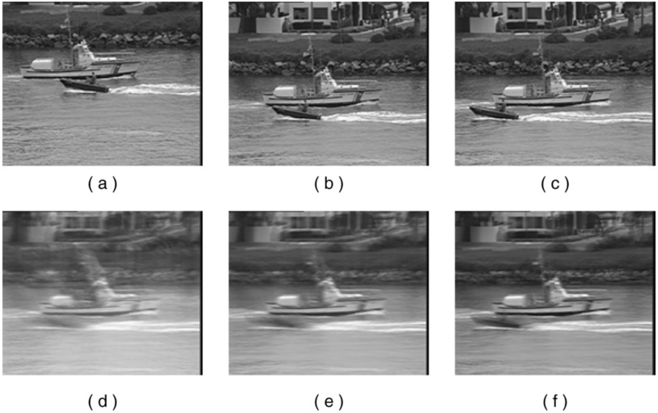
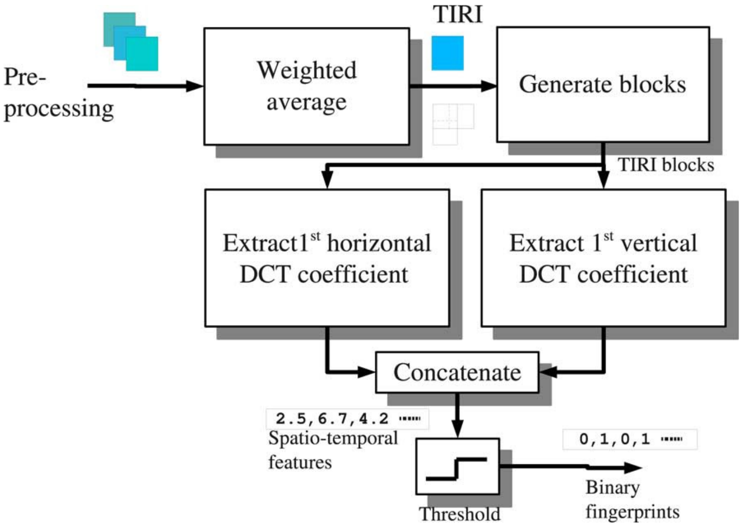


Figure : (d) $w_k = 1$ (costant), (e) $w_k = k$ (linear), (f) $w_k = 1.2^k$ (exponential)

TIRI-DCT


Pre-
processing
Weighted
average
TIRI
Generate blocks
TIRI blocks
Extract1st horizontal
DCT coefficient
Extract 1st vertical
DCT coefficient
Concatenate
2.5,6.7,4.2 ......
Spatio-temporal
features
Threshold
0,1,0,1 ......
Binary
fingerprints

Searching

Searching requirements

After the fingerprint is extracted, the database is searched for the closest fingerprint.

Searching requirements

After the fingerprint is extracted, the database is searched for the closest fingerprint.

Nearest neighbor problem

Fingerprints of two different copies of the same video are similar but not necessarily identical. We then search for the most similar fingerprint in the binary space.

Searching requirements

After the fingerprint is extracted, the database is searched for the closest fingerprint.

Nearest neighbor problem

Fingerprints of two different copies of the same video are similar but not necessarily identical. We then search for the most similar fingerprint in the binary space.

Since the database is very large, the algorithm must be fast. A fast *approximate* algorithm is preferred over an *exact* slow one.

Searching requirements

After the fingerprint is extracted, the database is searched for the closest fingerprint.

Nearest neighbor problem

Fingerprints of two different copies of the same video are similar but not necessarily identical. We then search for the most similar fingerprint in the binary space.

Since the database is very large, the algorithm must be fast. A fast *approximate* algorithm is preferred over an *exact* slow one.

Hamming distance

The process used to create binary fingerprints allow us to use the Hamming distance as a metric of similarity.

Inverted-File-Based Similarity search

$$\underbrace{0110001}_{1^{st}\ \text{word}}\ \underbrace{0010110}_{2^{nd}\ \text{word}} \dots\ \underbrace{1110001}_{(n-1)^{th}\ \text{word}}\ \underbrace{0010101}_{n^{th}\ \text{word}}$$

$m =$ word length

$$n = \frac{L}{m}$$

Inverted-File-Based Similarity search

$$\underbrace{0110001}_{1^{st}\text{ word}}\underbrace{0010110}_{2^{nd}\text{ word}}\ldots\underbrace{1110001}_{(n-1)^{th}\text{ word}}\;\underbrace{0010101}_{n^{th}\text{ word}}$$

$m =$ word length

$$n = \frac{L}{m}$$

Position of a word in a fingerprint

	1	2	3	...	n-1	n
1	1, 44, 250	10, 14, 99	21, 250			
2	2, 86	5, 170, 241				
⋮				⋱		
⋮						
2^m				...	61, 26	4

Inverted-File-Based Similarity search

To find a query fingerprint in a database:

- The fingerprint is divided into *n* words (of *m* bits)

Position of a word in a fingerprint

	1	2	3	...	n-1	n
1	1, 44, 250	10, 14, 99	21, 250			
2	2, 86	5, 170, 241				
⋮				⋱		
⋮						
2^m				...	61, 26	4

Inverted-File-Based Similarity search

To find a query fingerprint in a database:

- The fingerprint is divided into n words (of m bits)
- The query is compared with fingerprints that starts with the same word.

Position of a word in a fingerprint

	1	2	3	...	n-1	n
1	1, 44, 250	10, 14, 99	21, 250			
2	2, 86	5, 170, 241				
⋮				⋱		
⋮						
2^m				...	61, 26	4

Inverted-File-Based Similarity search

To find a query fingerprint in a database:

- The fingerprint is divided into *n* words (of *m* bits)
- The query is compared with fingerprints that starts with the same word.
- If the Hamming distance is below a certain threshold, the match is found.

Position of a word in a fingerprint

	1	2	3	...	n-1	n
1	1, 44, 250	10, 14, 99	21, 250			
2	2, 86	5, 170, 241				
⋮				⋱		
⋮						
2^m				...	61, 26	4

Inverted-File-Based Similarity search

To find a query fingerprint in a database:

- The fingerprint is divided into n words (of m bits)
- The query is compared with fingerprints that starts with the same word.
- If the Hamming distance is below a certain threshold, the match is found.
- Otherwise the procedure is repeated with the second word and so on.

Position of a word in a fingerprint

	1	2	3	...	n-1	n
1	1, 44, 250	10, 14, 99	21, 250			
2	2, 86	5, 170, 241				
⋮				⋱		
2^m				...	61, 26	4

Cluster-based similarity search

- The database is clustered into K non-overlapping groups; each cluster has a centroid called cluster head.

Cluster-based similarity search

- The database is clustered into K non-overlapping groups; each cluster has a centroid called cluster head.
- The cluster head closest to the query fingerprint is found

$$\underbrace{0100001}_{\substack{0 \\ \text{dist}_0 = 2}}\ \underbrace{0010110}_{\substack{0 \\ \text{dist}_0 = 3}}\ \underbrace{1010111}_{\substack{1 \\ \text{dist}_1 = 2}}\ \underbrace{0000001}_{\substack{0 \\ \text{dist}_0 = 1}}\ \underbrace{1111101}_{\substack{1 \\ \text{dist}_1 = 1}}$$

Cluster head = 00101

Cluster-based similarity search

- The database is clustered into K non-overlapping groups; each cluster has a centroid called cluster head.
- The cluster head closest to the query fingerprint is found

$$\underbrace{0100001}_{\substack{0 \\ \text{dist}_0 = 2}}\ \underbrace{0010110}_{\substack{0 \\ \text{dist}_0 = 3}}\ \underbrace{1010111}_{\substack{1 \\ \text{dist}_1 = 2}}\ \underbrace{0000001}_{\substack{0 \\ \text{dist}_0 = 1}}\ \underbrace{1111101}_{\substack{1 \\ \text{dist}_1 = 1}}$$

$$\text{Cluster head} = 00101$$

- Every fingerprint belonging to this cluster are searched to find a match

Cluster-based similarity search

- The database is clustered into K non-overlapping groups; each cluster has a centroid called cluster head.
- The cluster head closest to the query fingerprint is found

$$\underbrace{0100001}_{\substack{0\\ \text{dist}_0\ =\ 2}}\ \underbrace{0010110}_{\substack{0\\ \text{dist}_0\ =\ 3}}\ \underbrace{1010111}_{\substack{1\\ \text{dist}_1\ =\ 2}}\ \underbrace{0000001}_{\substack{0\\ \text{dist}_0\ =\ 1}}\ \underbrace{1111101}_{\substack{1\\ \text{dist}_1\ =\ 1}}$$

Cluster head = 00101

- Every fingerprint belonging to this cluster are searched to find a match
- If no match is found, then the cluster that is the second closest to the query is examined, then the third and so on

Results

Results

- A database of 200 videos was created
- Videos were attacked (distorted) to create query videos

Results

- A database of 200 videos was created
- Videos were attacked (distorted) to create query videos

Metrics

Recall It's the true positive rate. It is a measure of robustness.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Results

- A database of 200 videos was created
- Videos were attacked (distorted) to create query videos

Metrics

Recall It's the true positive rate. It is a measure of robustness.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Precision Percentage of correct hits within all detected copies.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Results

- A database of 200 videos was created
- Videos were attacked (distorted) to create query videos

Metrics

Recall It's the true positive rate. It is a measure of robustness.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Precision Percentage of correct hits within all detected copies.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

F-score It is the harmonic mean of precision and recall:

$$F = 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Results

Attack	F-score	
	3D-DCT	TIRI-DCT
Noise	1.00	0.99
Brightness	1.00	0.99
Contrast	1.00	0.99
Rotation	1.00	1.00
Time shift	0.91	0.98
Spatial shift	1.00	0.98
Frame loss	0.98	0.99
Average	0.99	0.99

Results

Attack	F-score	
	3D-DCT	**TIRI-DCT**
Noise	1.00	0.99
Brightness	1.00	0.99
Contrast	1.00	0.99
Rotation	1.00	1.00
Time shift	0.91	0.98
Spatial shift	1.00	0.98
Frame loss	0.98	0.99
Average	0.99	0.99

Speed

- TIRI-DCT is more than 3 times faster than 3D-DCT

Results

Attack	F-score	
	3D-DCT	**TIRI-DCT**
Noise	1.00	0.99
Brightness	1.00	0.99
Contrast	1.00	0.99
Rotation	1.00	1.00
Time shift	0.91	0.98
Spatial shift	1.00	0.98
Frame loss	0.98	0.99
Average	0.99	0.99

Speed

- TIRI-DCT is more than 3 times faster than 3D-DCT
- Approximate search algorithms are much faster than the exhaustive search when the error rates are low.

Results

Attack	F-score	
	3D-DCT	**TIRI-DCT**
Noise	1.00	0.99
Brightness	1.00	0.99
Contrast	1.00	0.99
Rotation	1.00	1.00
Time shift	0.91	0.98
Spatial shift	1.00	0.98
Frame loss	0.98	0.99
Average	0.99	0.99

Speed

- TIRI-DCT is more than 3 times faster than 3D-DCT
- Approximate search algorithms are much faster than the exhaustive search when the error rates are low. Cluster-based approach is faster.

Table of Contents

Shazam

- The algorithm is noise resistant and efficient. It can identify a short segment of music out of a database of over a million tracks. However the algorithm is not meant to be resistant to general attacks (e.g time compression, pitch alterations, ...).

Shazam

- The algorithm is noise resistant and efficient. It can identify a short segment of music out of a database of over a million tracks. However the algorithm is not meant to be resistant to general attacks (e.g time compression, pitch alterations, . . .).
- Shazam was founded in 1999 and launched its first identification service in 2002

Shazam

- The algorithm is noise resistant and efficient. It can identify a short segment of music out of a database of over a million tracks. However the algorithm is not meant to be resistant to general attacks (e.g time compression, pitch alterations, . . .).
- Shazam was founded in 1999 and launched its first identification service in 2002
- Shazam has now 400 million users in 200 countries and was used to identify 15 billion songs

Shazam

- The algorithm is noise resistant and efficient. It can identify a short segment of music out of a database of over a million tracks. However the algorithm is not meant to be resistant to general attacks (e.g time compression, pitch alterations, . . .).
- Shazam was founded in 1999 and launched its first identification service in 2002
- Shazam has now 400 million users in 200 countries and was used to identify 15 billion songs
- The value of the company is now half a billion dollars

Video fingerprinting

- The proposed system is robust and it is able to find a match in a database of tens of millions of fingerprints in a few seconds

Video fingerprinting

- The proposed system is robust and it is able to find a match in a database of tens of millions of fingerprints in a few seconds
- It yields a high average true positive rate of 97.6% and a low average false positive rate of 1%.

YouTube testing

Results of Scott Smitelli tests:

Modification Performed	Result
Song reversed	Pass
Pitch was lowered 6%	Pass
Pitch was lowered 5%	Fail
Pitch was raised 5%	Fail
Pitch was raised 6%	Pass
Time was expanded 6%	Pass
Time was expanded 5%	Fail
Time was compressed 5%	Fail
Time was compressed 6%	Pass
44% noise, 56% song	Fail
45% noise, 55% song	Pass

Modification Performed	Result
Volume was reduced 48 dB	Fail
Volume was reduced 18 dB	Fail
Volume was reduced 6 dB	Fail
Volume was increased 6 dB	Fail
Volume was increased 18 dB	Fail
Volume was increased 48 dB	Fail
Slow down 4%	Pass
Slow down 3%	Fail
Speed up 3%	Fail
Speed up 4%	Fail
Speed up 5%	Pass

Questions?