CS 583– Computational Audio

Wayne Snyder
Computer Science Department
Boston University

Lecture 19

Content-Based Audio Retrieval (Fingerprinting)

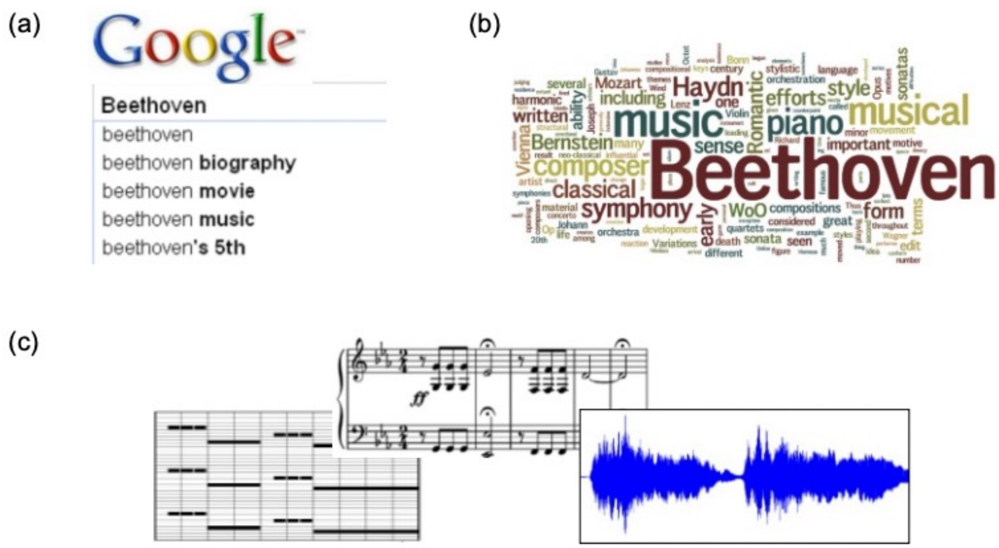
Overview (Audio Retrieval)

Retrieval of audio (essentially database search) can be done in at least three ways:

(a) Traditional retrieval using textual metadata (e.g., artist, title) and a web search engine.

(b) Retrieval based on rich and expressive metadata given by tags.

(c) Content-based retrieval using audio, MIDI, or score information



Audio Content-Based Identification

Database: Huge collection consisting of all audio recordings (encoded by feature representations) to be potentially identified.

Goal: Given a short query audio fragment, identify the original audio recording the query is taken from.

Notes:

- Instance of fragment-based retrieval
- High specificity: we can not only identify a piece of music, but even a specific recording of the piece

Many Applications!

- Audio Database Retrieval (“Query by Humming”)
- “What’s that song?” – user hears a song in noisy environment, wants to identify it (and perhaps buy it right there on his/her iPhone)
- Connected Audio (audio triggers changes to your environment): Screensavers, web ads, graphical displays on audio devices
- Music recommender systems (“here’s another one just like that one”)
- Broadcast Monitoring: identify music being played for royalty collection, surveys, filtering of copyrighted material, ...

Application Scenario: What‘s that song?

- User hears music playing in the environment
- User records music fragment (3 – 5 seconds) with mobile phone
- Audio fingerprints are extracted from the recording and sent to an audio identification service
- Service identifies audio recording based on fingerprints
- Service sends back metadata (track title, artist) to user

Audio Fingerprints

An audio fingerprint is a content-based compact signature that summarizes some specific audio content.

Requirements:

- Discriminative power
- Invariance to distortions
- Compactness:
- Computational simplicity

Audio Fingerprints

An audio fingerprint is a content-based compact signature that summarizes a piece of audio content

Requirements:

- Discriminative power
- Invariance to distortions
- Compactness
- Computational simplicity

- *Ability to accurately identify an item within a huge number of other items (informative, characteristic)*
- *Low probability of false positives*
- *Recorded query excerpt only a few seconds*
- *Large audio collection on the server side (millions of songs)*

Audio Fingerprints

An audio fingerprint is a content-based compact signature that summarizes a piece of audio content

Requirements:

- Discriminative power
- Invariance to distortions
- Compactness
- Computational simplicity

- *Recorded query may be distorted and superimposed with other audio sources*
- *Background noise*
- *Pitch and tempo may vary (audio played faster or slower or in a different key)*
- *Codec artifacts (e.g., MP3 compression)*
- *…*

Audio Fingerprints

An audio fingerprint is a content-based compact signature that summarizes a piece of audio content

Requirements:

- Discriminative power
- Invariance to distortions
- Compactness
- Computational simplicity

- *Reduction of complex multimedia objects*
- *Reduction of dimensionality*
- *Making indexing feasible*
- *Allowing for fast search*

Audio Fingerprints

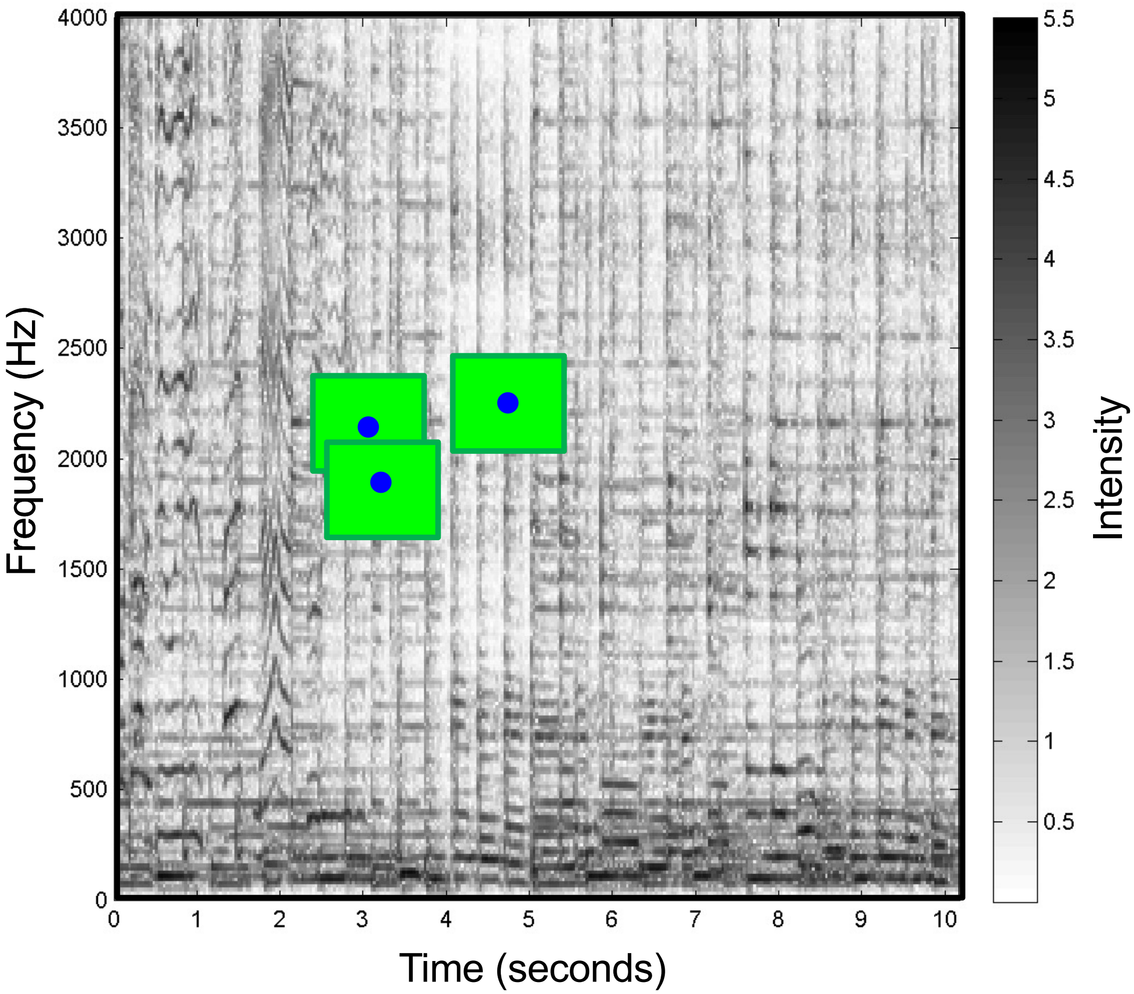
An audio fingerprint is a content-based compact signature that summarizes a piece of audio content

Requirements:

- Discriminative power
- Invariance to distortions
- Compactness
- Computational simplicity

- *Computational efficiency*
- *Extraction of fingerprint should be simple*
- *Size of fingerprints should be small*

Fingerprints (Shazam)

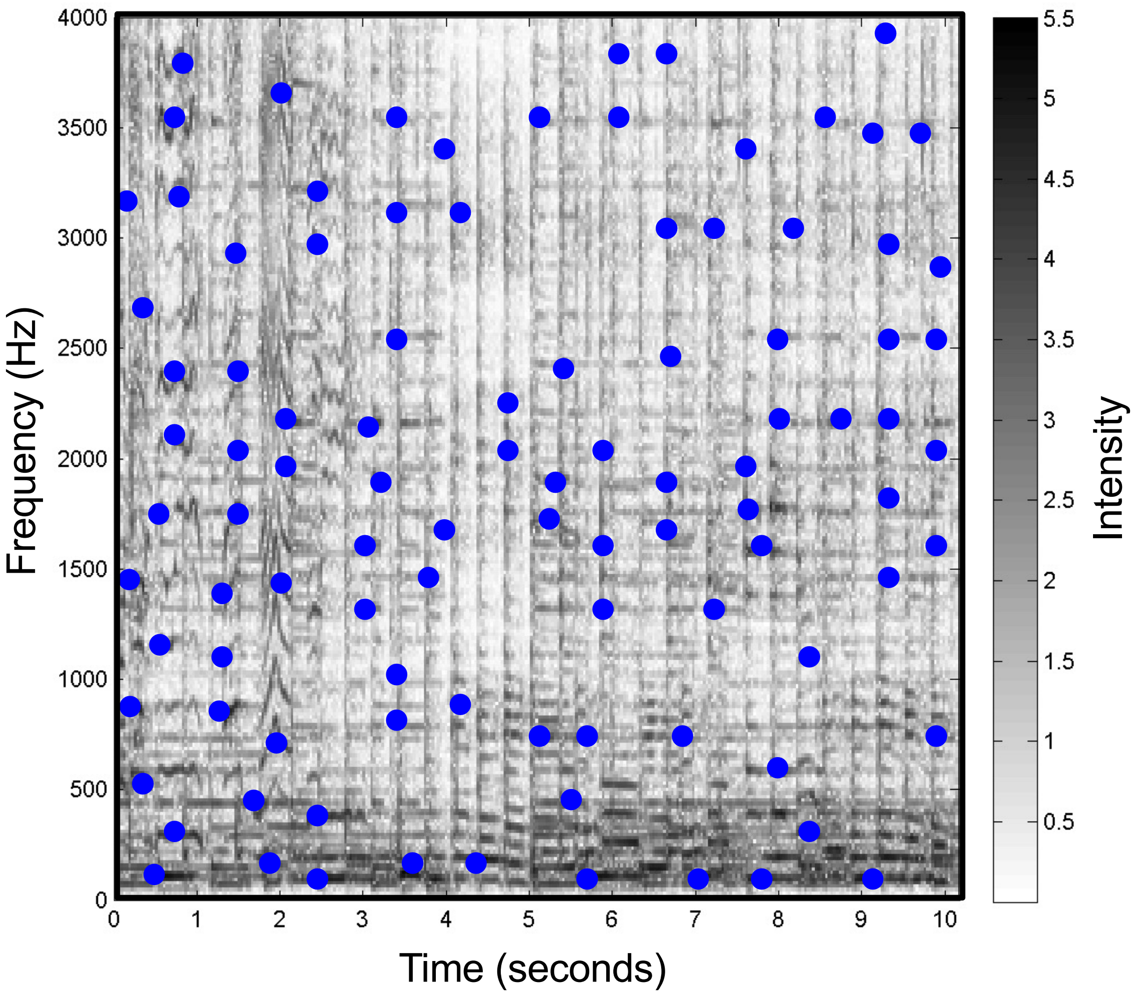


Steps:

1. Spectrogram
2. Peaks
   (local maxima –
   subtract local mean)

- *Efficiently computable*
- *Standard transform*
- *Robust*

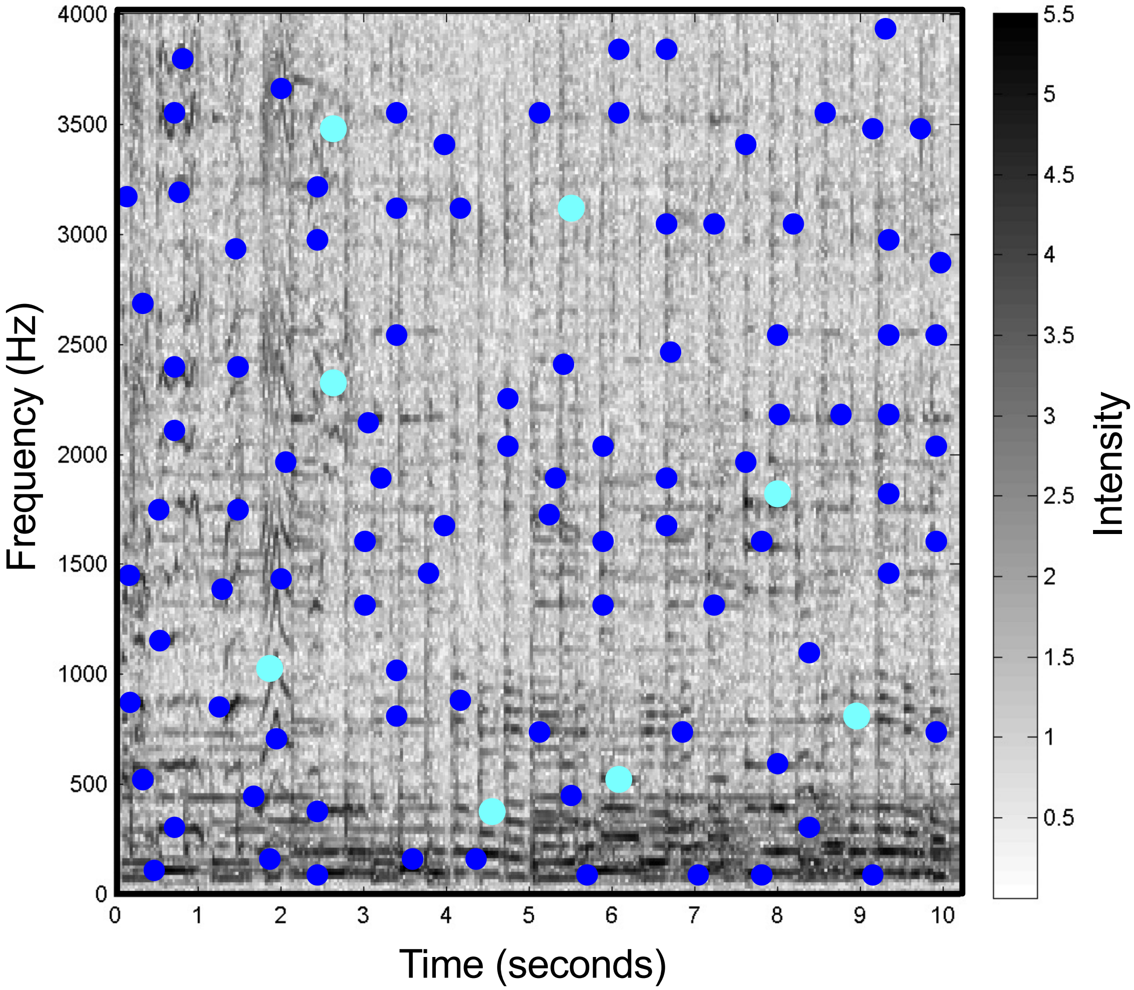
Fingerprints (Shazam)



Steps:

1. Spectrogram
2. Peaks

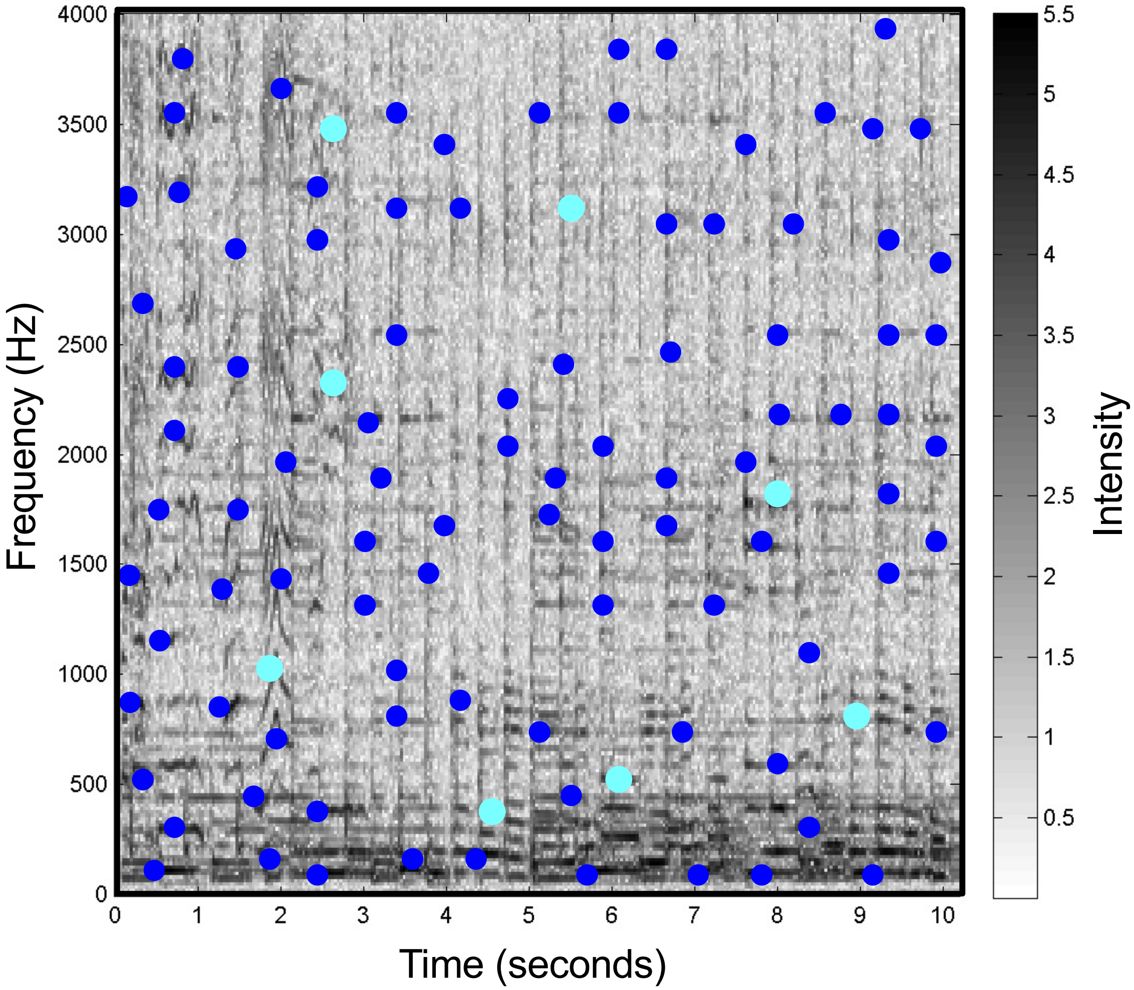
Fingerprints (Shazam)



Steps:

1. Spectrogram
2. Peaks / differing peaks

Fingerprints (Shazam)



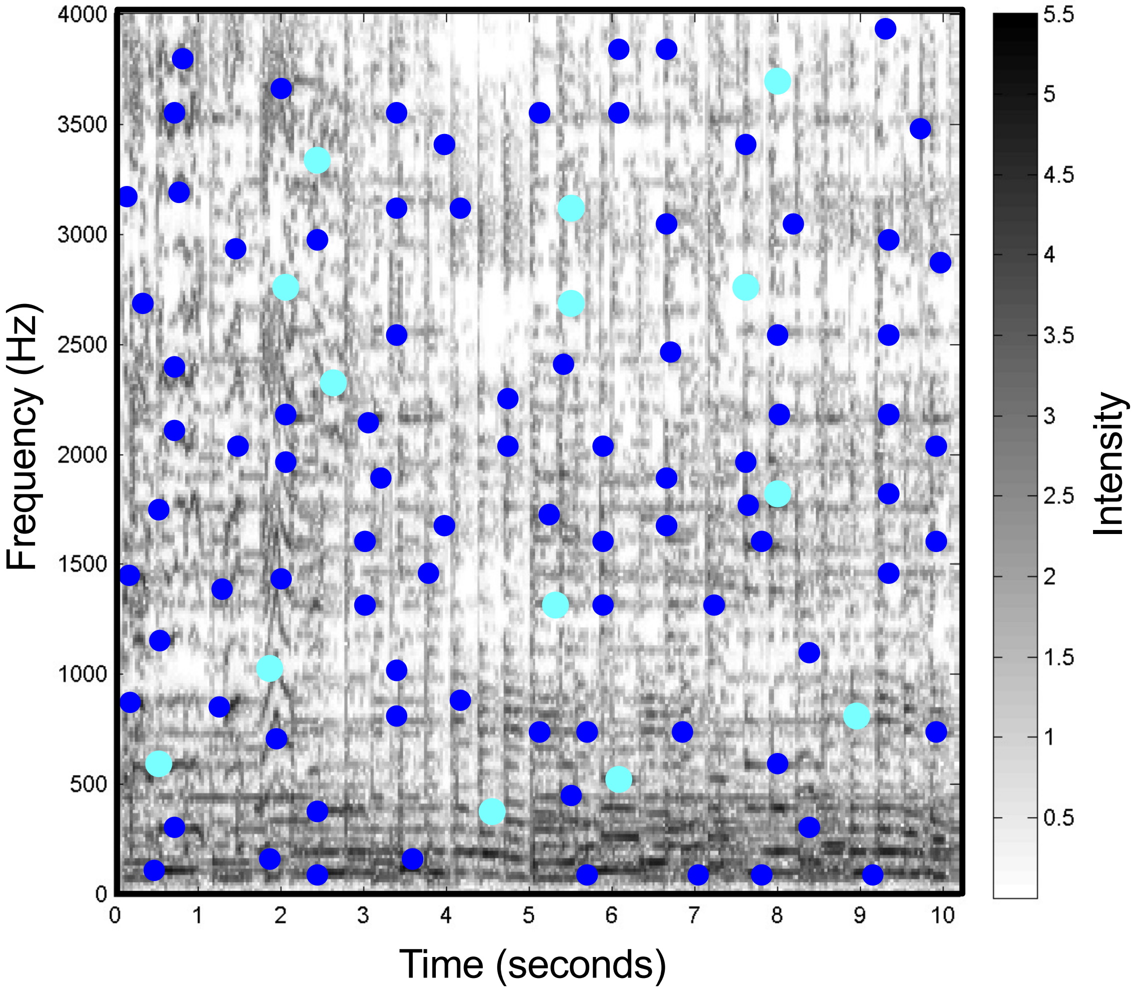
Steps:

1. Spectrogram
2. Peaks / differing peaks

Robustness:

- Noise, reverb, room acoustics, equalization

Fingerprints (Shazam)



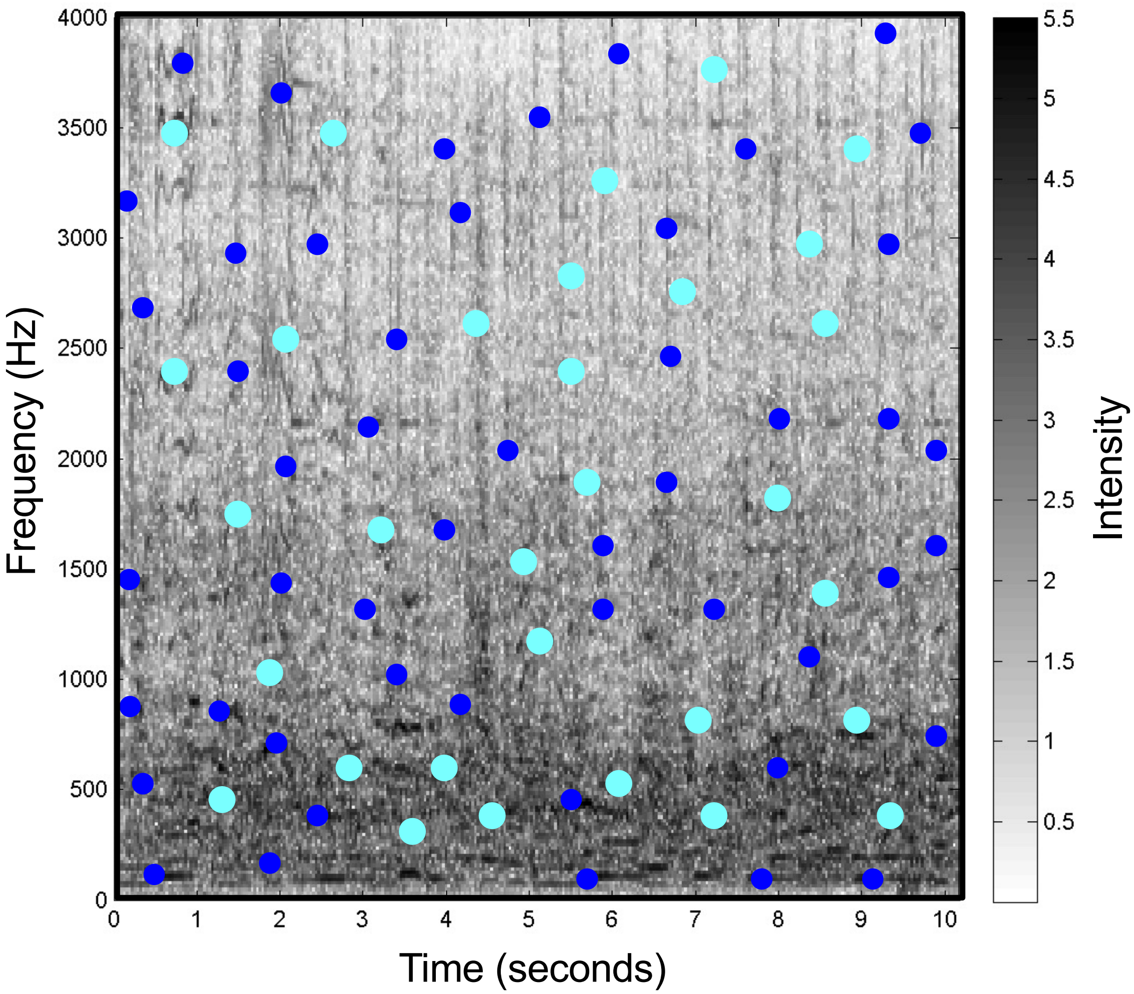
Steps:

1. Spectrogram
2. Peaks / differing peaks

Robustness:

- Noise, reverb, room acoustics, equalization
- Audio codec

Fingerprints (Shazam)



Steps:

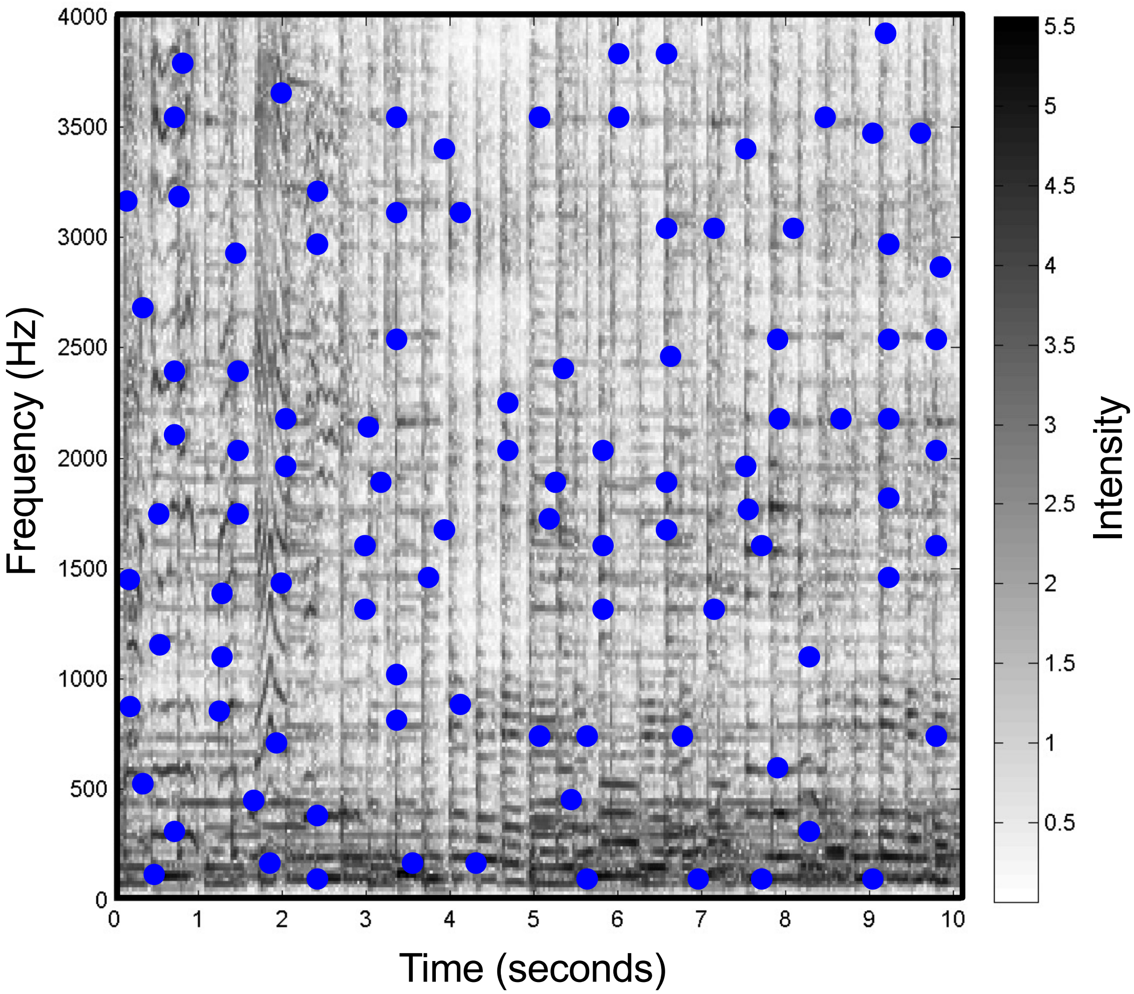
1. Spectrogram
2. Peaks / differing peaks

Robustness:

- Noise, reverb, room acoustics, equalization
- Audio codec
- Superposition of other audio sources

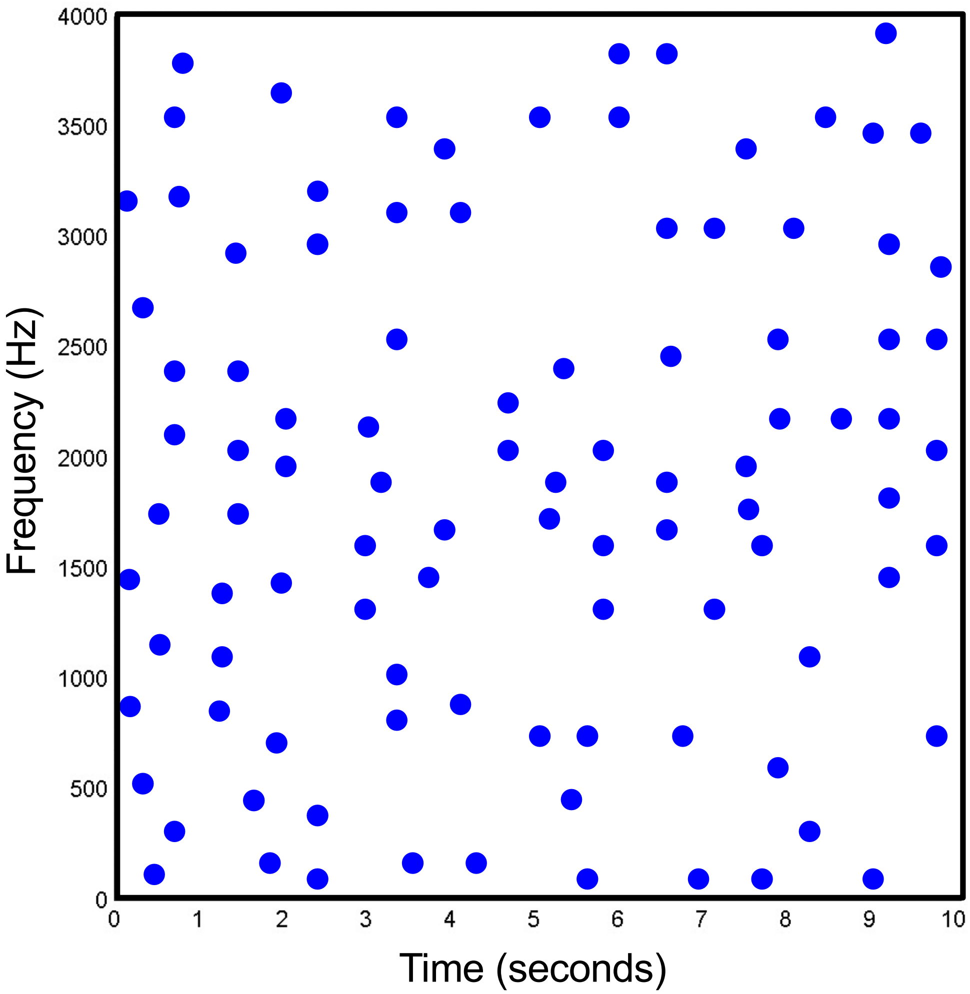
Matching Fingerprints (Shazam)

Database document



Matching Fingerprints (Shazam)

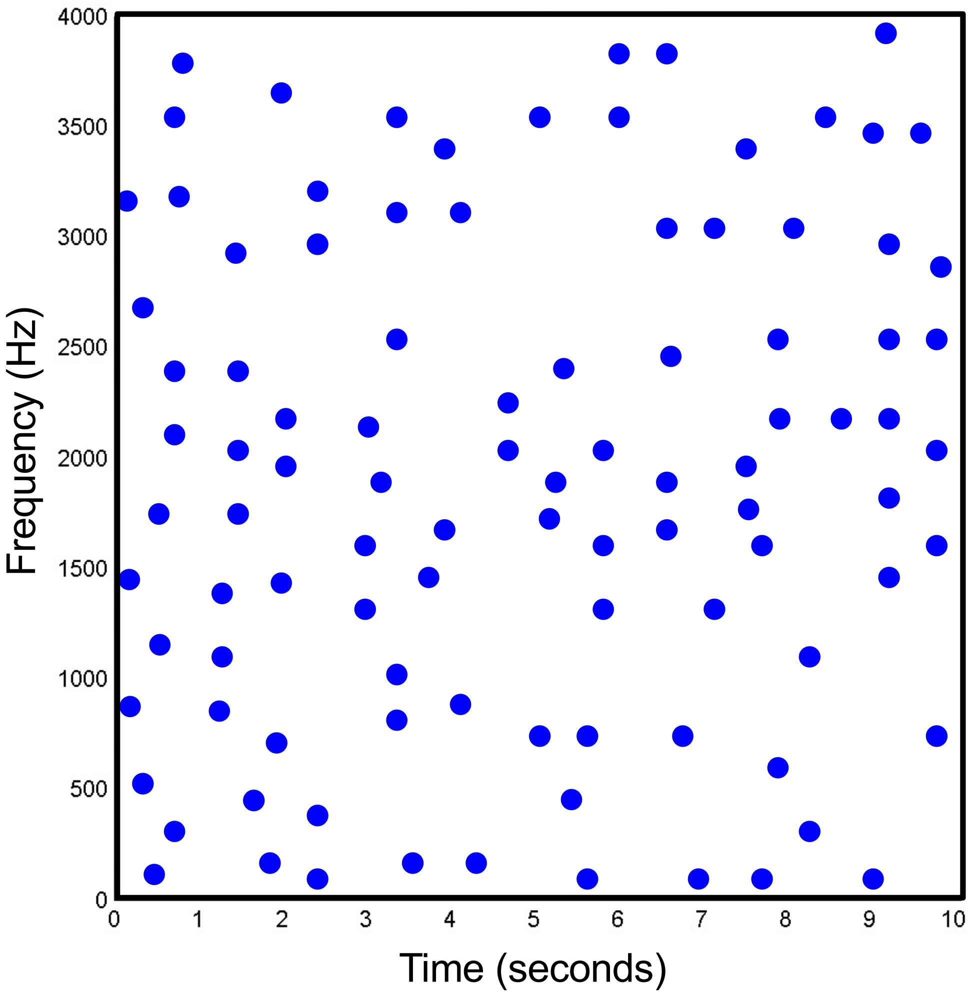
Database document (constellation map)

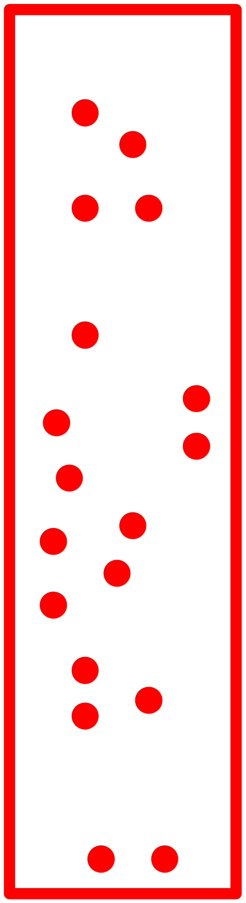


Matching Fingerprints (Shazam)

Database document
(constellation map)

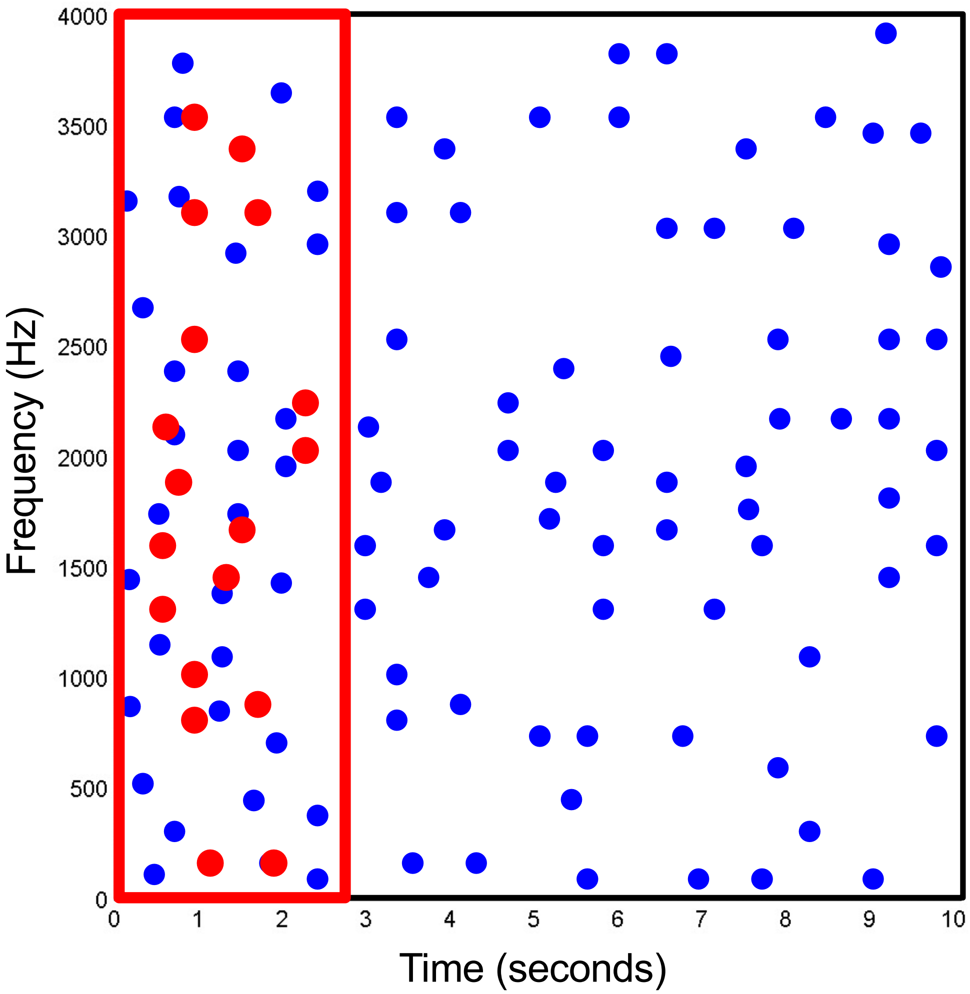
Query document
(constellation map)





Matching Fingerprints (Shazam)

Database document
(constellation map)

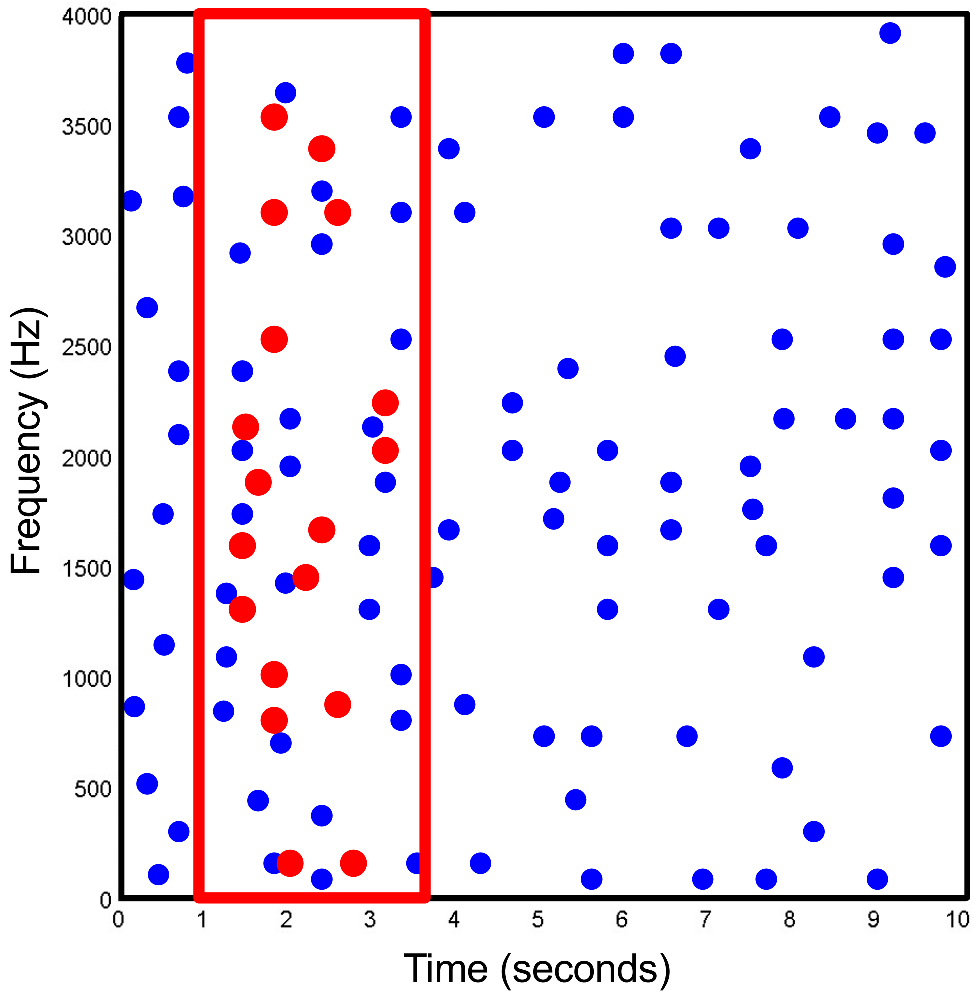


Query document
(constellation map)

1. Shift query across database document
2. Count matching peaks

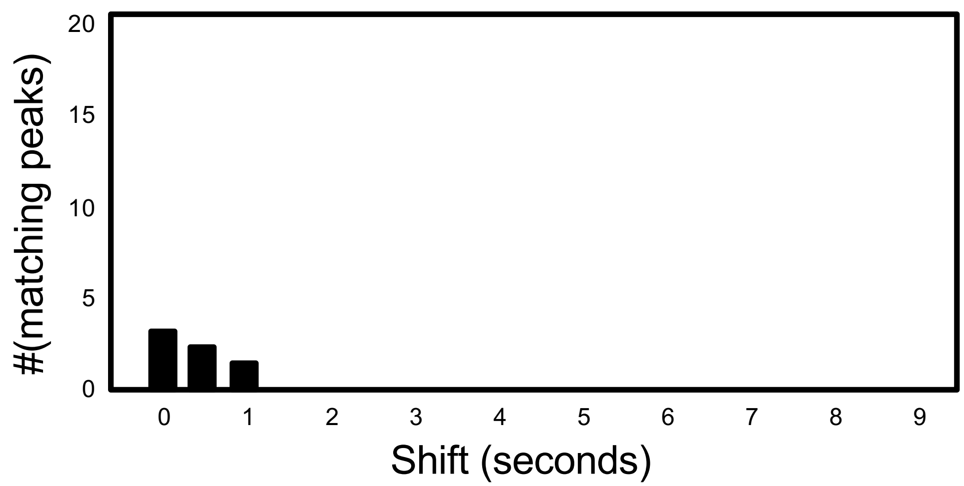
Matching Fingerprints (Shazam)

Database document
(constellation map)



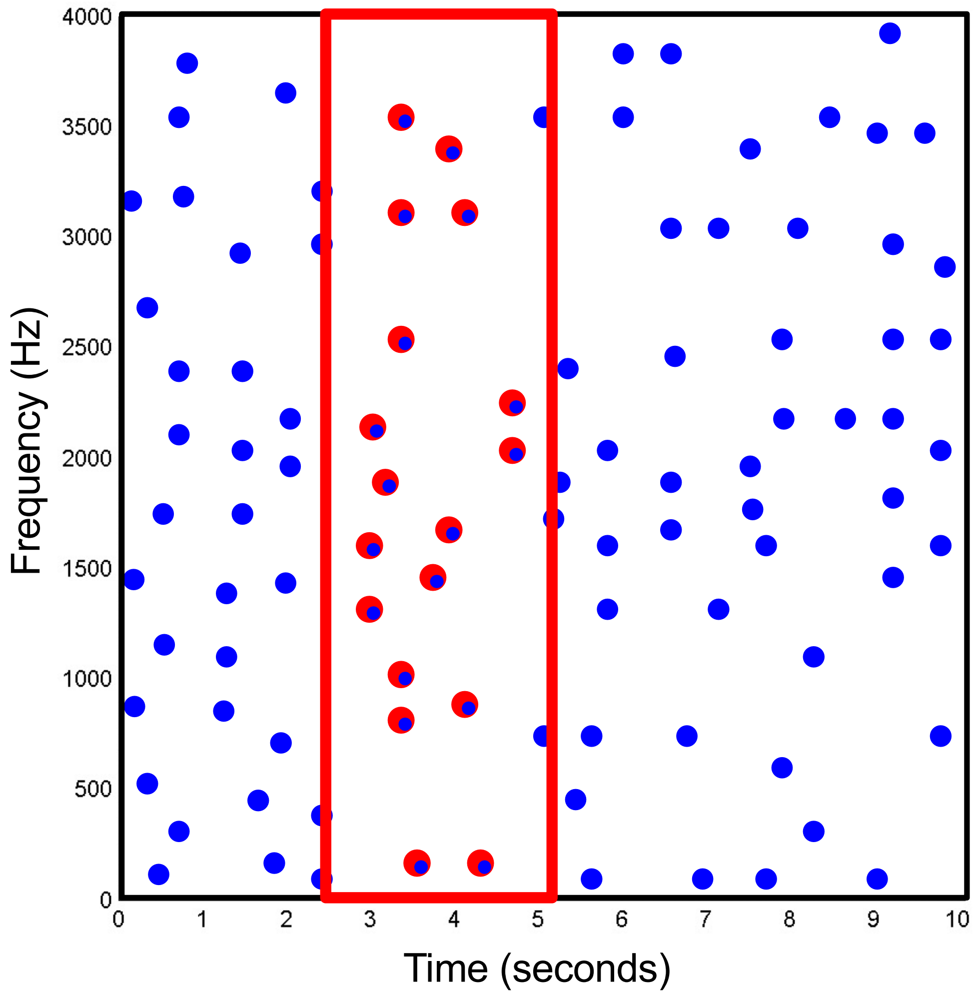
Query document
(constellation map)

1. Shift query across database document
2. Count matching peaks



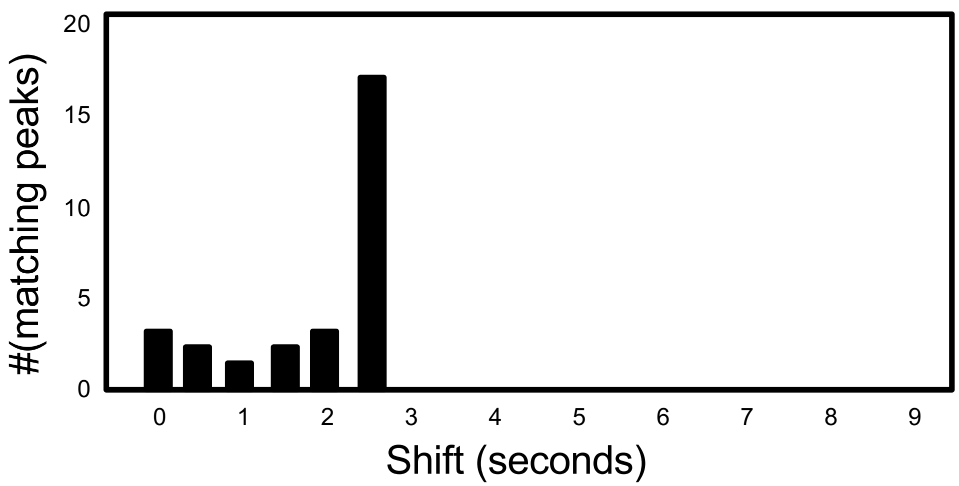
Matching Fingerprints (Shazam)

Database document
(constellation map)



Query document
(constellation map)

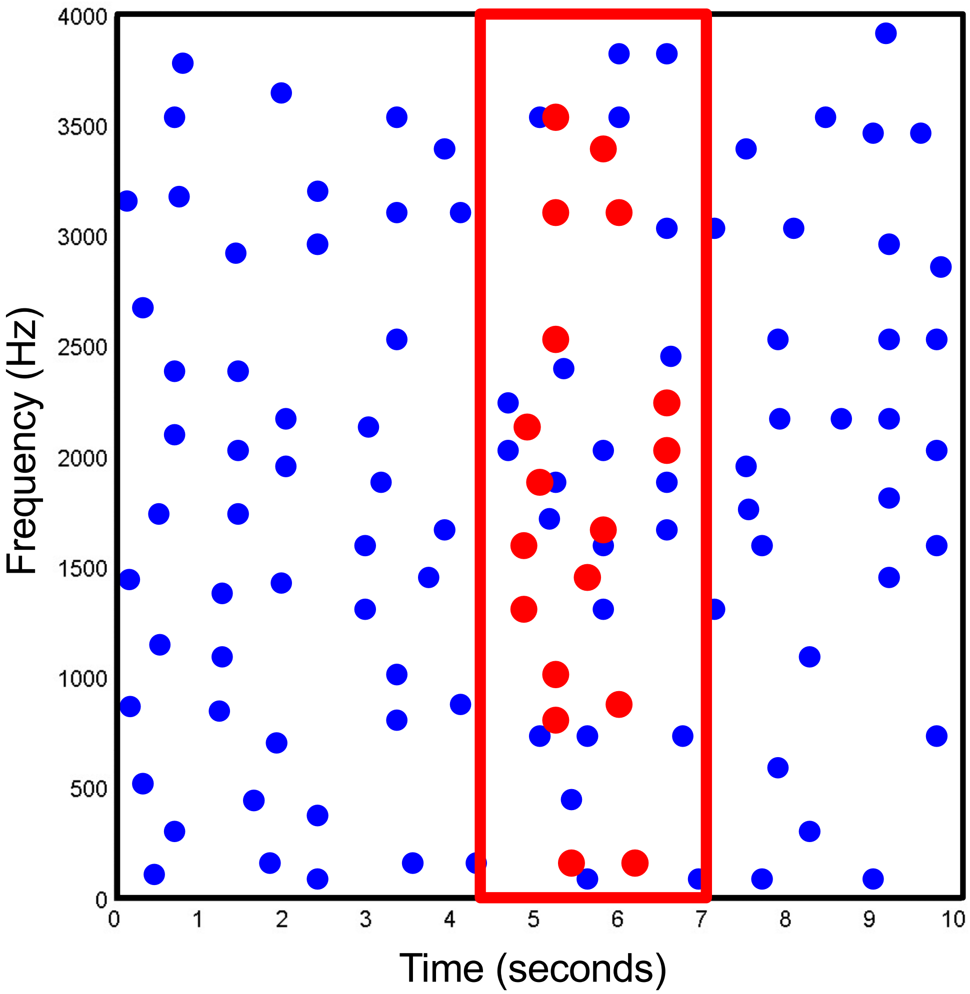
1. Shift query across database document
2. Count matching peaks



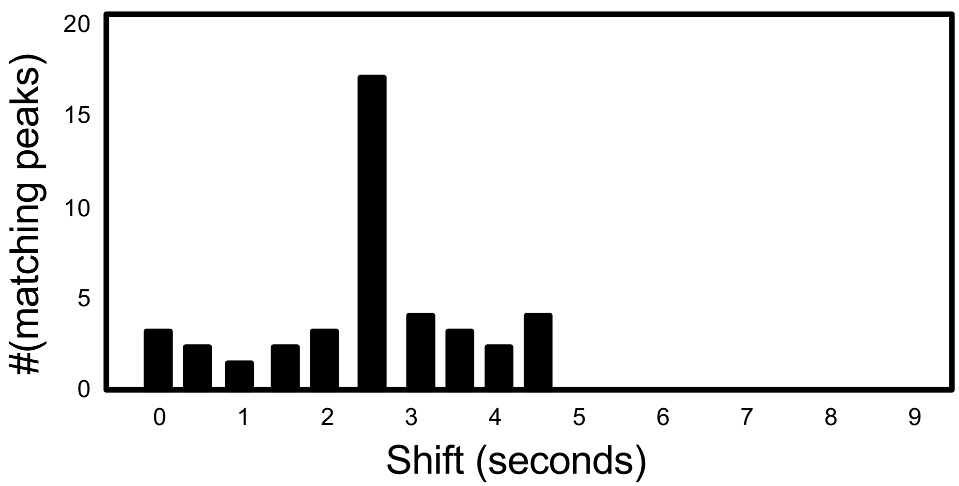
Matching Fingerprints (Shazam)

Database document
(constellation map)

Query document
(constellation map)

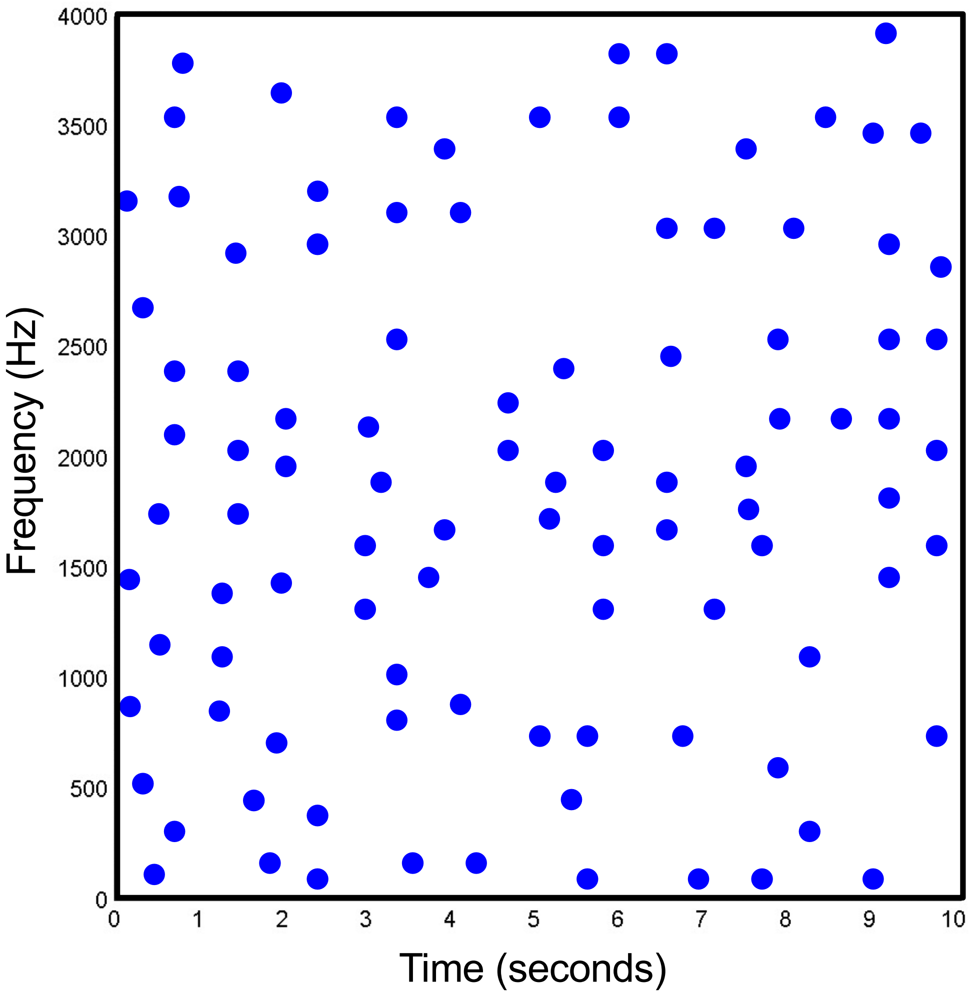


1. Shift query across database document
2. Count matching peaks



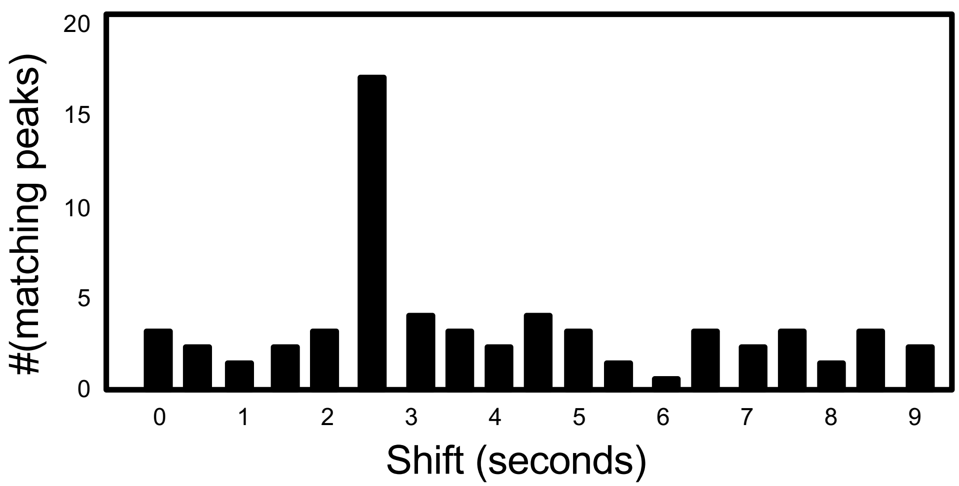
Matching Fingerprints (Shazam)

Database document
(constellation map)



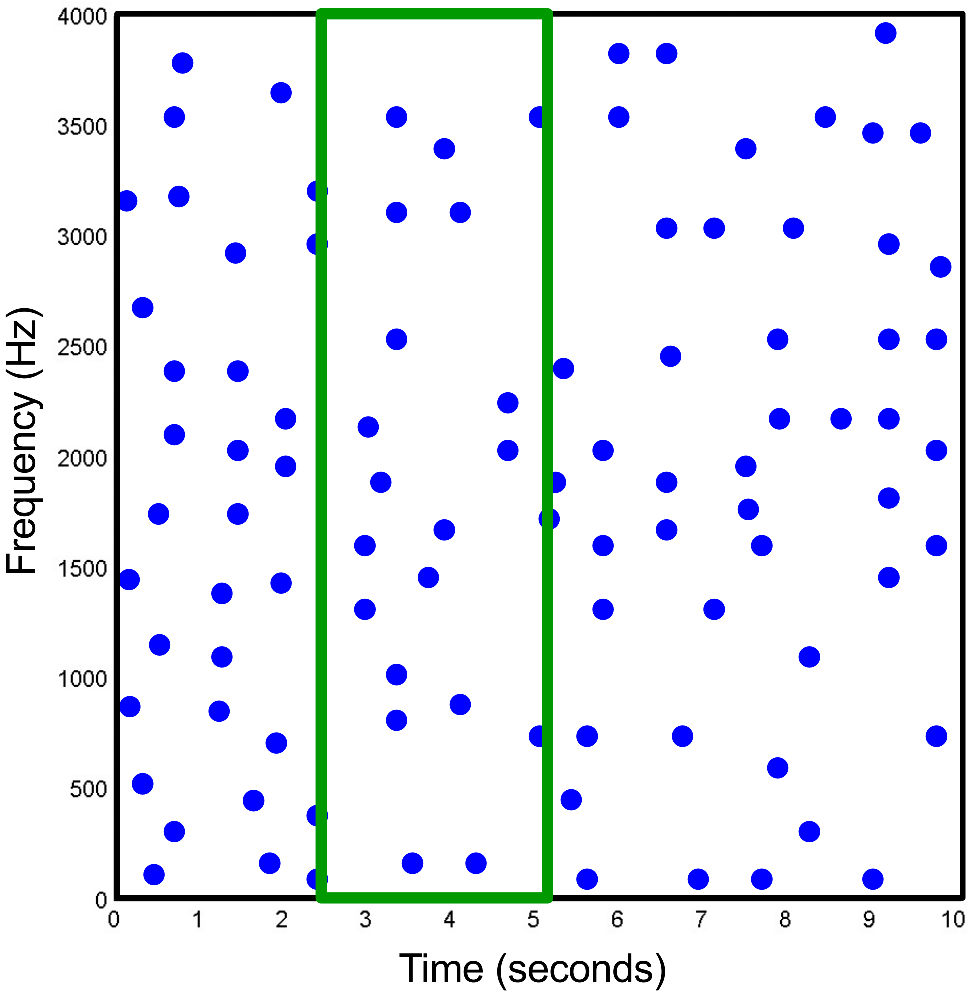
Query document
(constellation map)

1. Shift query across database document
2. Count matching peaks



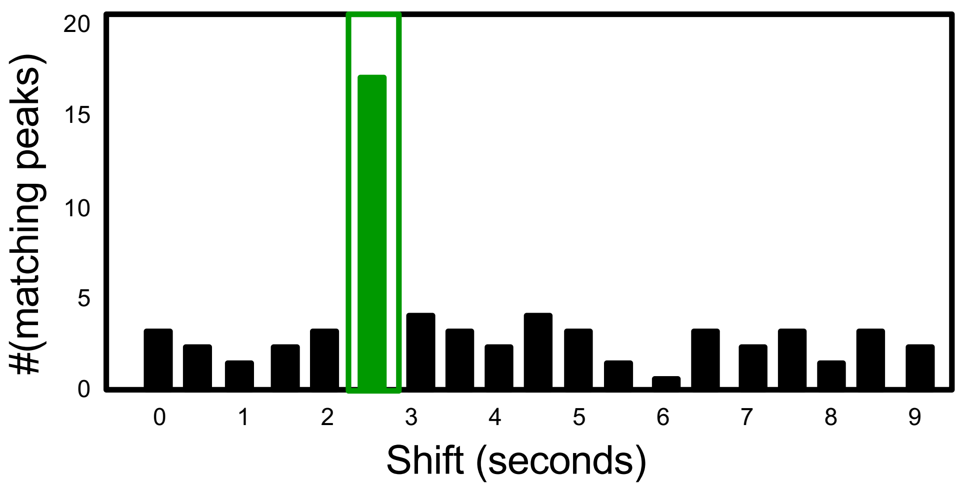
Matching Fingerprints (Shazam)

Database document
(constellation map)



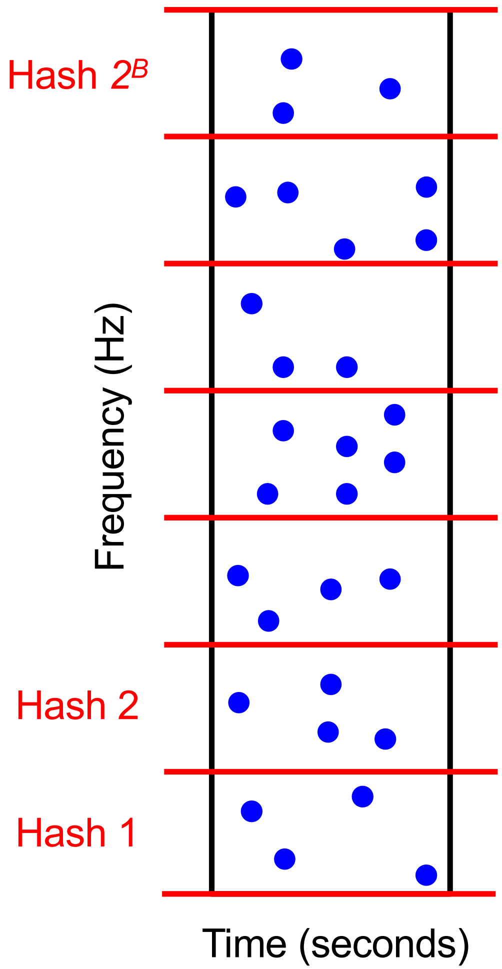
Query document
(constellation map)

1. Shift query across database document
2. Count matching peaks
3. High count indicates a hit (document ID & position)



Indexing (Shazam)

- Index the fingerprints using hash lists
- Hashes correspond to (quantized) frequencies



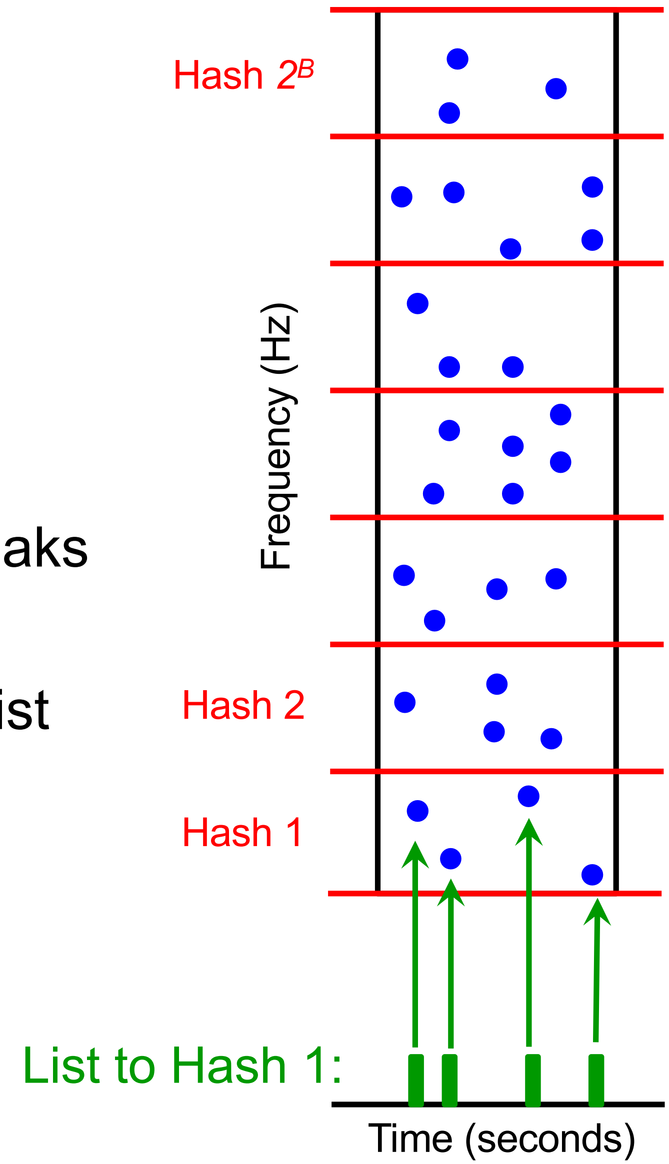
Indexing (Shazam)

- Index the fingerprints using hash lists
- Hashes correspond to frequency bins
- Hash list consists of time positions (and document IDs)

- N = number of spectral peaks
- B = #(bits) used to encode spectral peaks
- 2^B = number of hash lists
- $N / 2^B$ = average number of elements per list

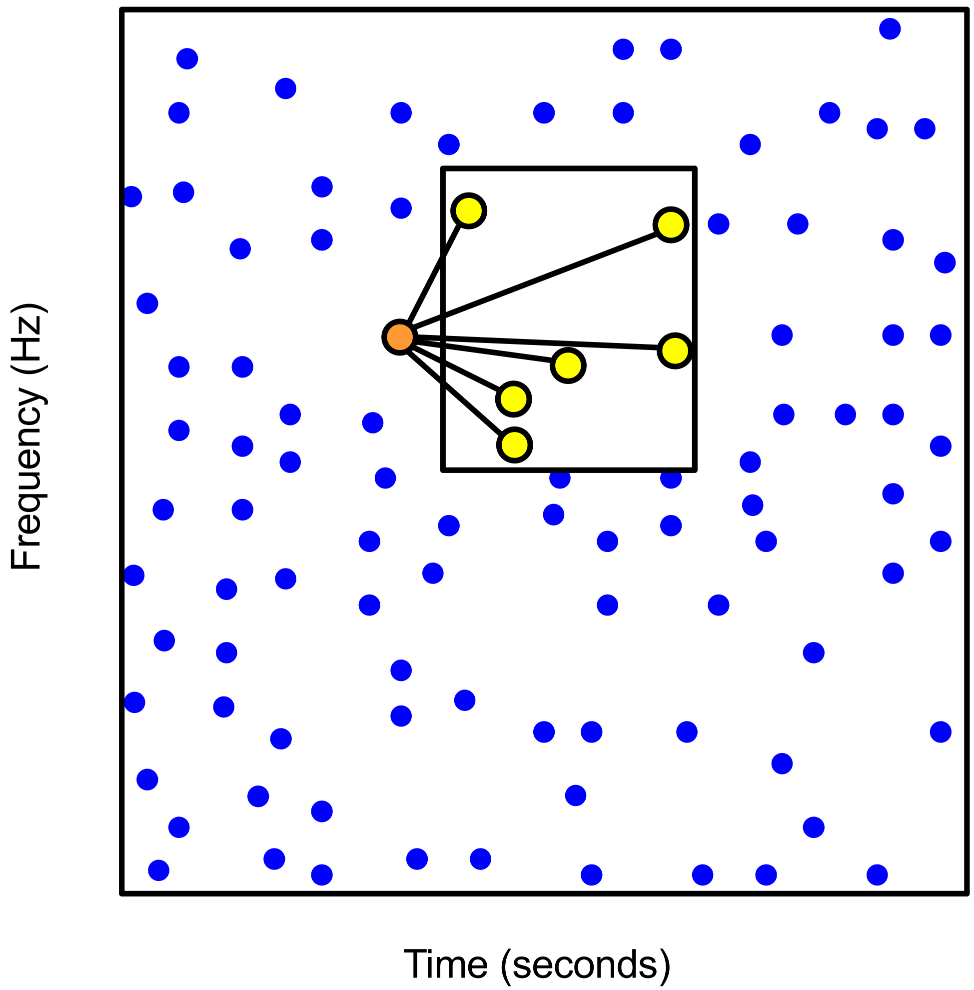
Problem:

- Individual peaks are not characteristic
- Hash lists may be very long
- Not suitable for indexing



Indexing (Shazam)

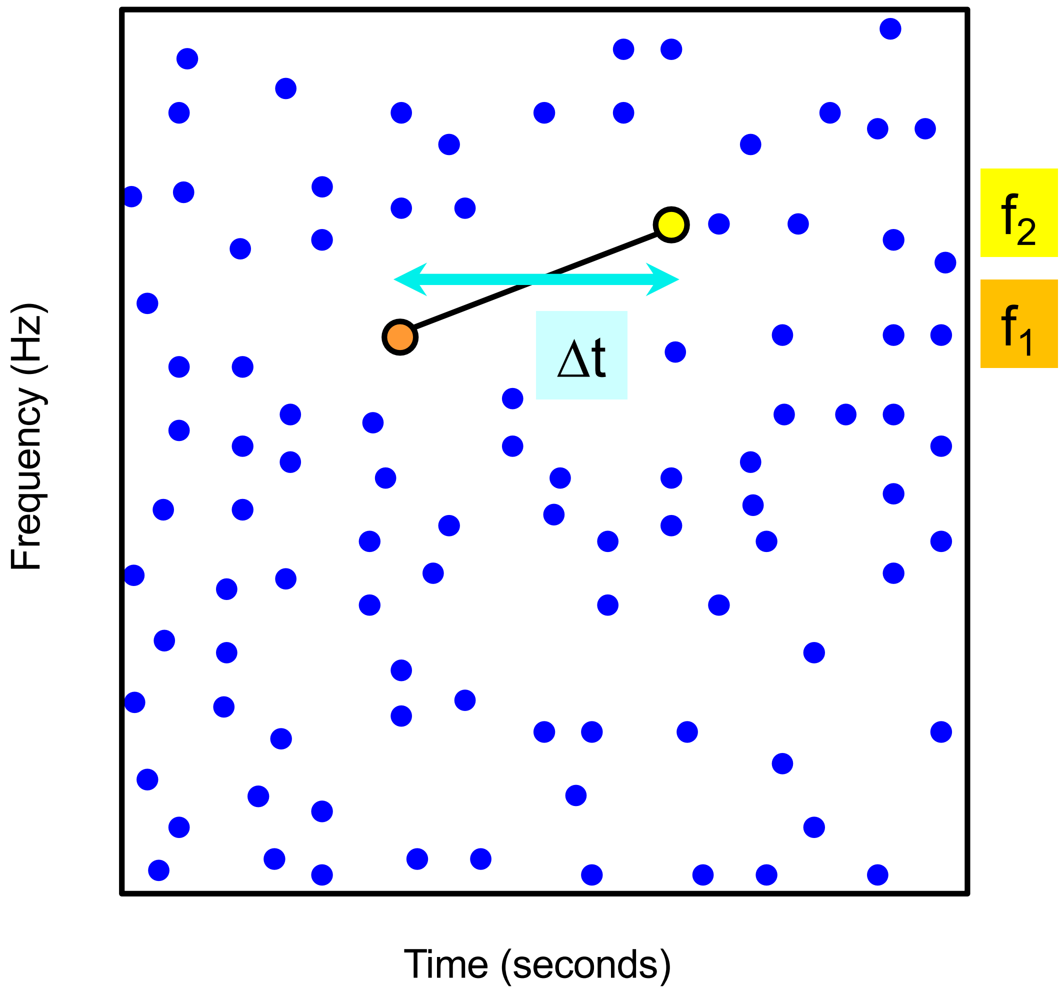
Idea: Use pairs of peaks to increase specificity of hashes



1. Peaks
2. Fix anchor point
3. Define target zone
4. Use pairs of points
5. Use every point as anchor point

Indexing (Shazam)

Idea: Use pairs of peaks to increase specificity of hashes



1. Peaks
2. Fix anchor point
3. Define target zone
4. Use paris of points
5. Use every point as anchor point

New hash:

Consists of two frequency values and a time difference:

$$(f_1 , f_2 , \Delta t)$$

Indexing (Shazam)

- A hash is formed between an anchor point and each point in the target zone using two frequency values and a time difference.

- Fan-out (taking pairs of peaks) may cause a combinatorial explosion in the number of tokens. However, this can be controlled by the size of the target zone.

- Using more complex hashes increases specificity (leading to much smaller hash lists) and speed (making the retrieval much faster).

Conclusions (Shazam)

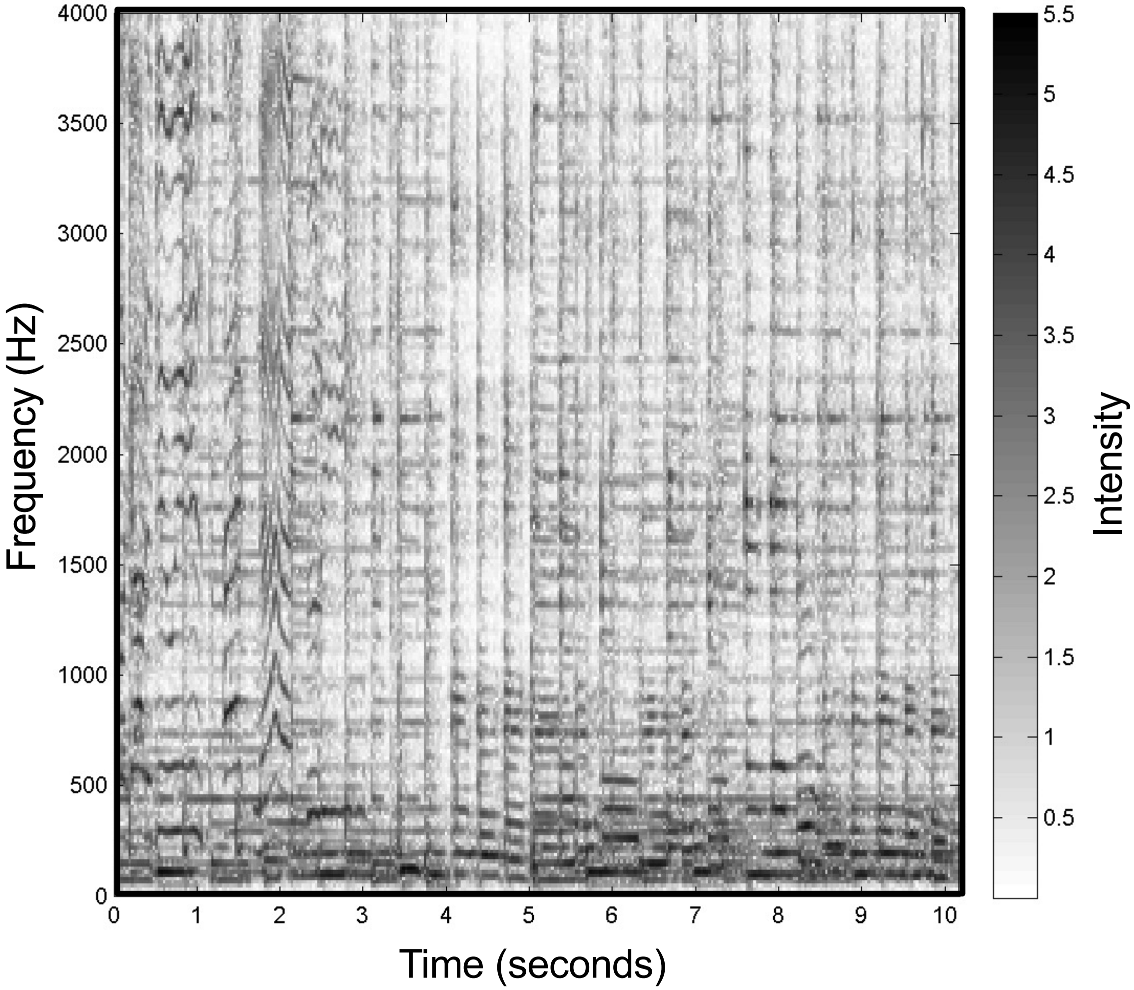
Many parameters to choose:

- Temporal and spectral resolution in spectrogram
- Peak picking strategy
- Target zone and fan-out parameter
- Hash function
- …

Fingerprints (Philips)

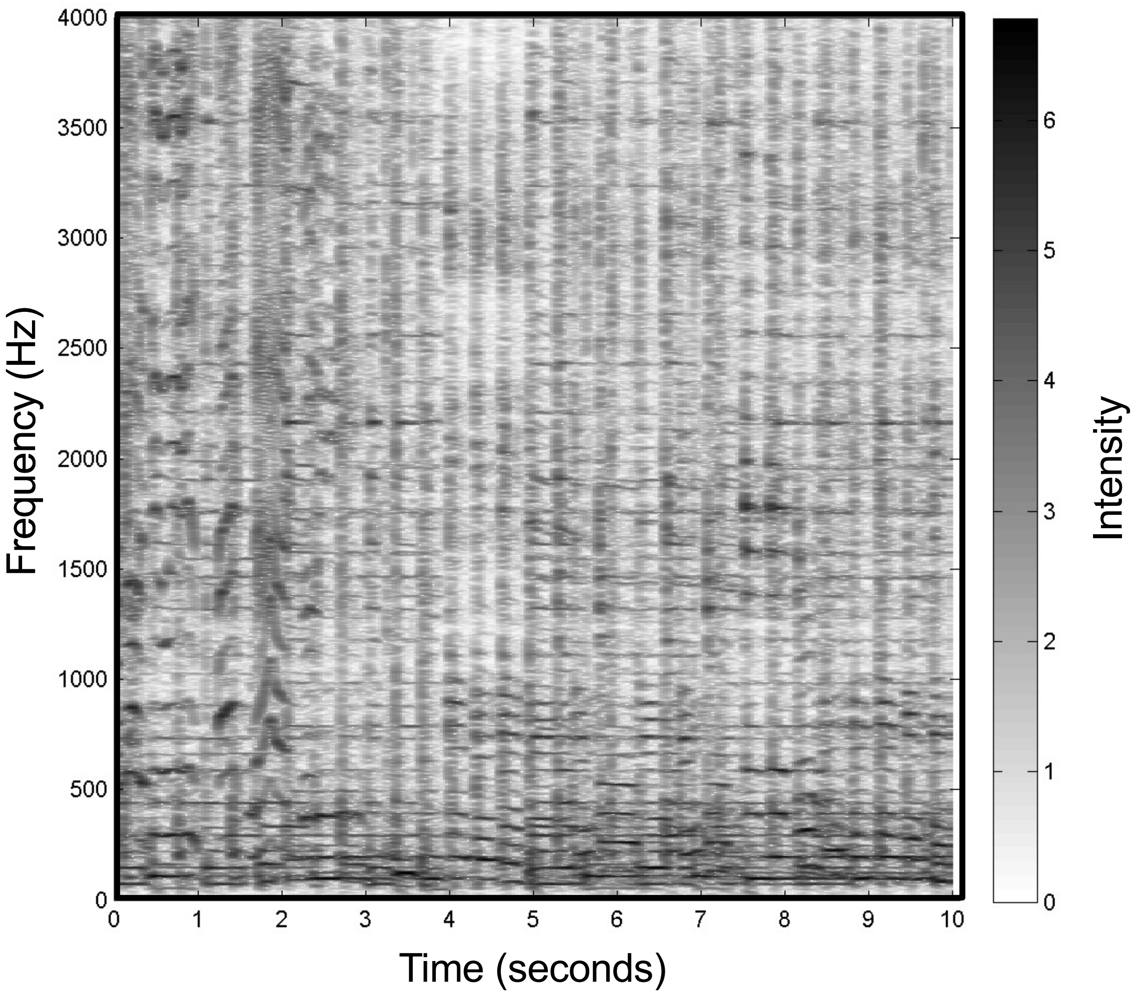
Steps:

1. Spectrogram



- *Efficiently computable*
- *Standard transform*
- *Robust*

Fingerprints (Philips)

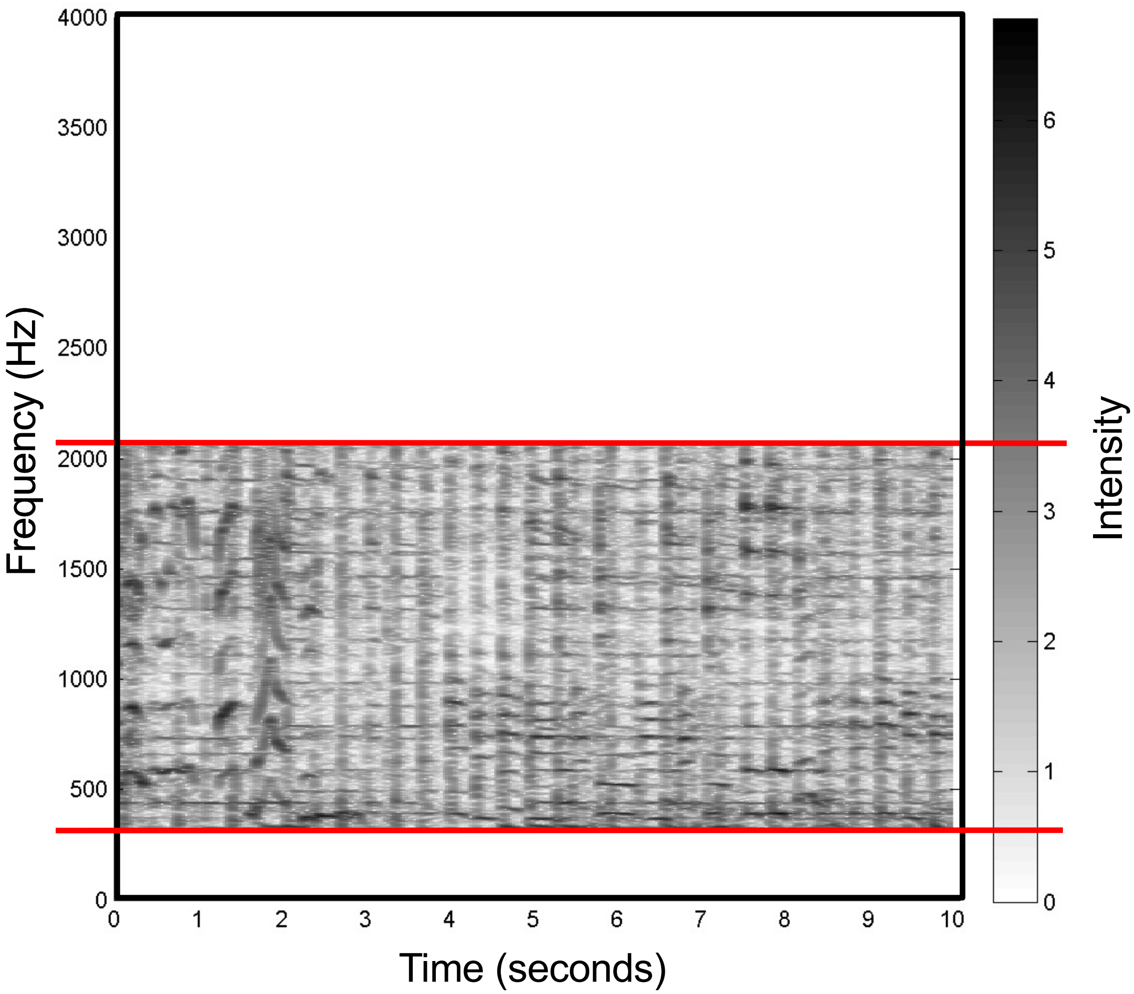


Steps:

1. Spectrogram (long window)

- *Coarse temporal resolution*
- *Large overlap of windows*
- *Robust to temporal distortion*

Fingerprints (Philips)

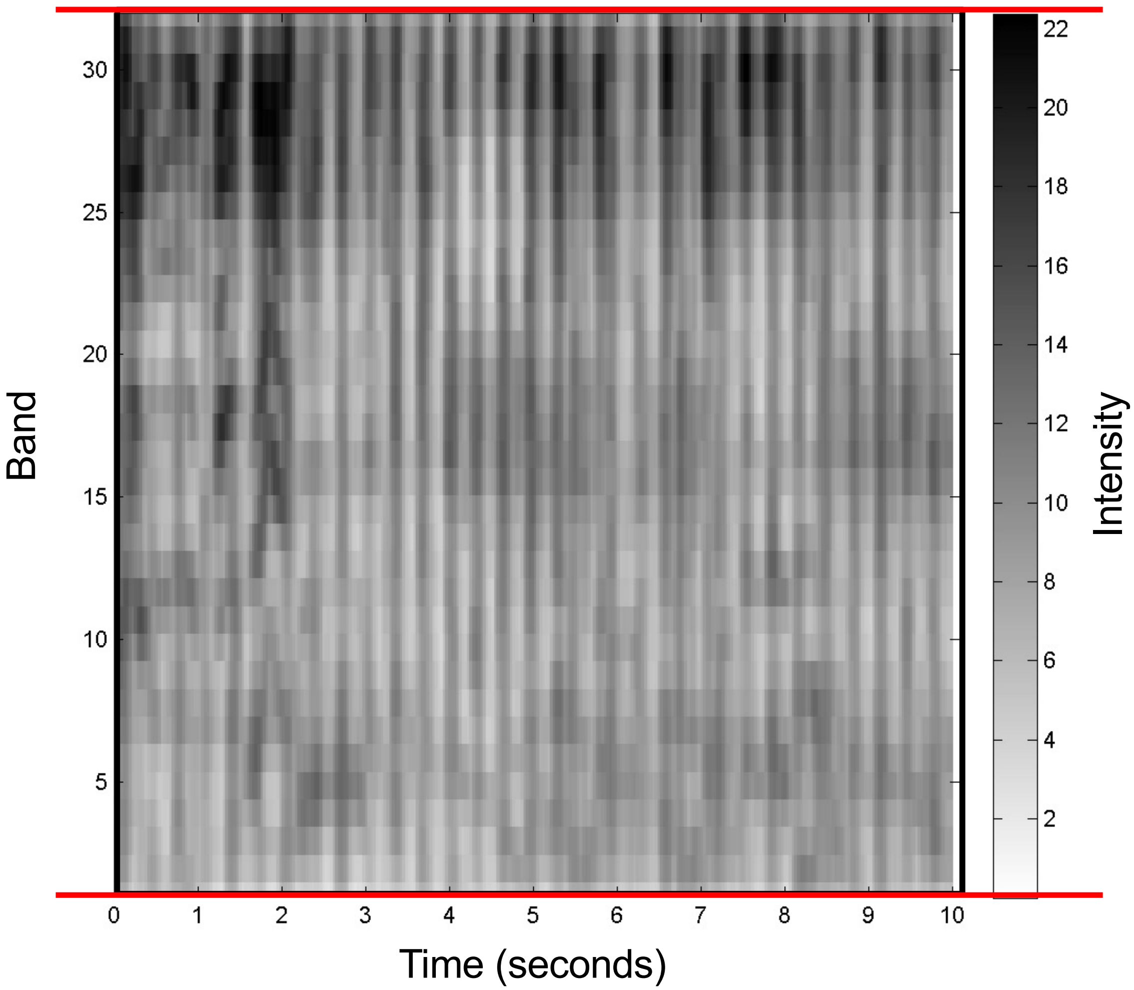


Steps:

1. Spectrogram (long window)
2. Consider limited frequency range

- *300 – 2000 Hz*
- *Most relevant spectral range (perceptually)*

Fingerprints (Philips)

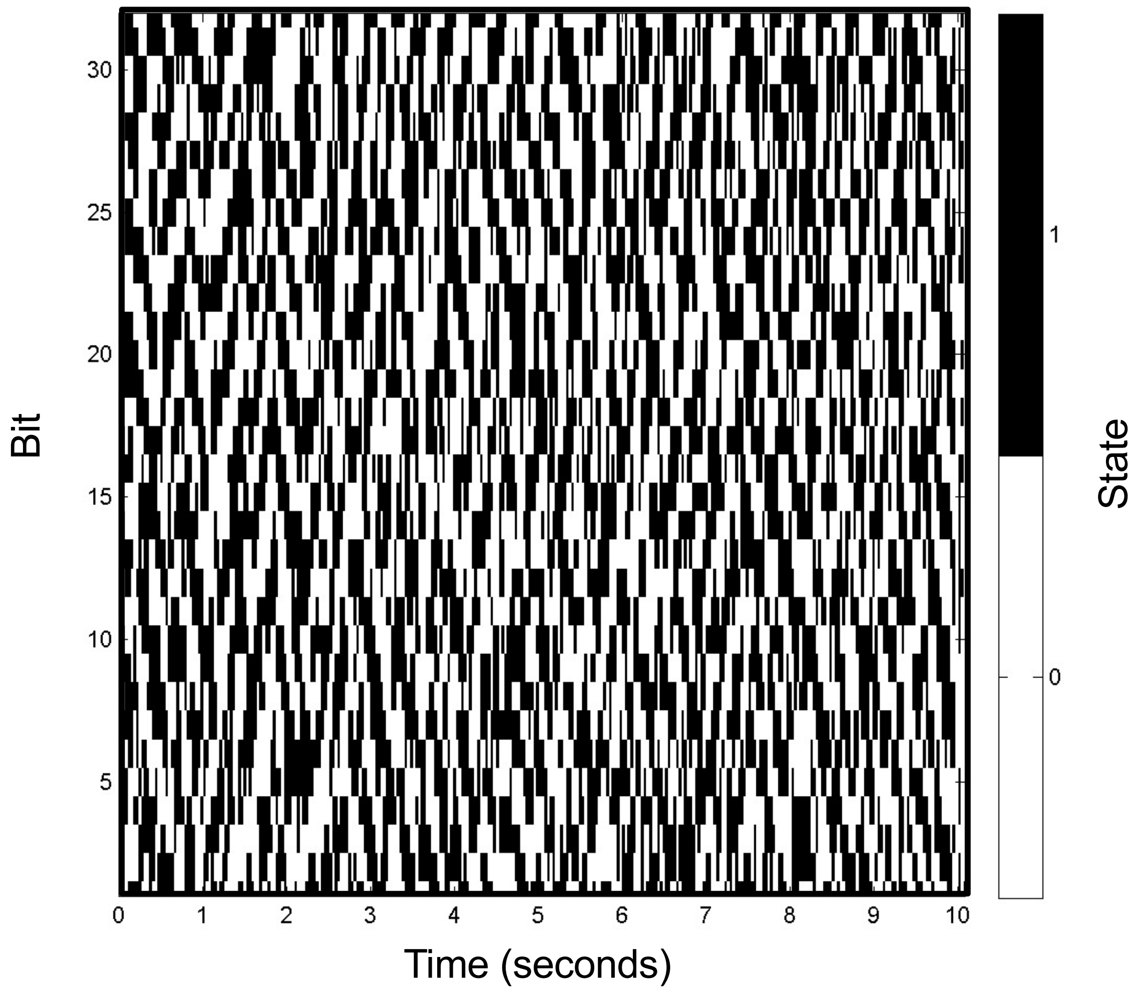


Steps:

1. Spectrogram (long window)
2. Consider limited frequency range
3. Log-frequency (Bark scale)

- *300 – 2000 Hz*
- *Most relevant spectral range (perceptually)*
- *33 bands (roughly bark scale)*
- *Coarse frequency resolution*
- *Robust to spectral distortions*

Fingerprints (Philips)

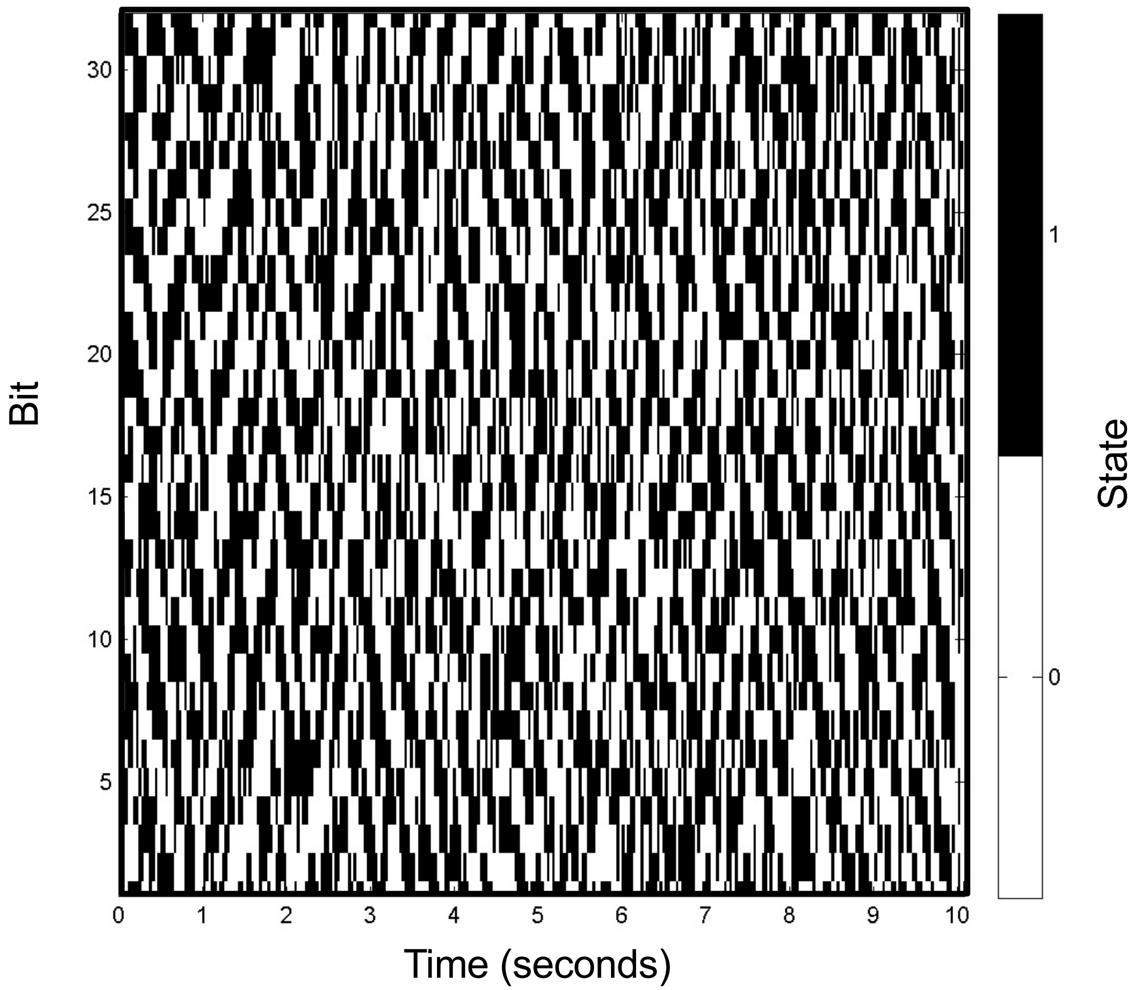


Steps:

1. Spectrogram (long window)
2. Consider limited frequency range
3. Log-frequency (Bark scale)
4. Encoded in binary

- *Local thresholding*
- *Sign of energy difference (simultanously along time and frequency axes)*
- *Sequence of 32-bit vectors*

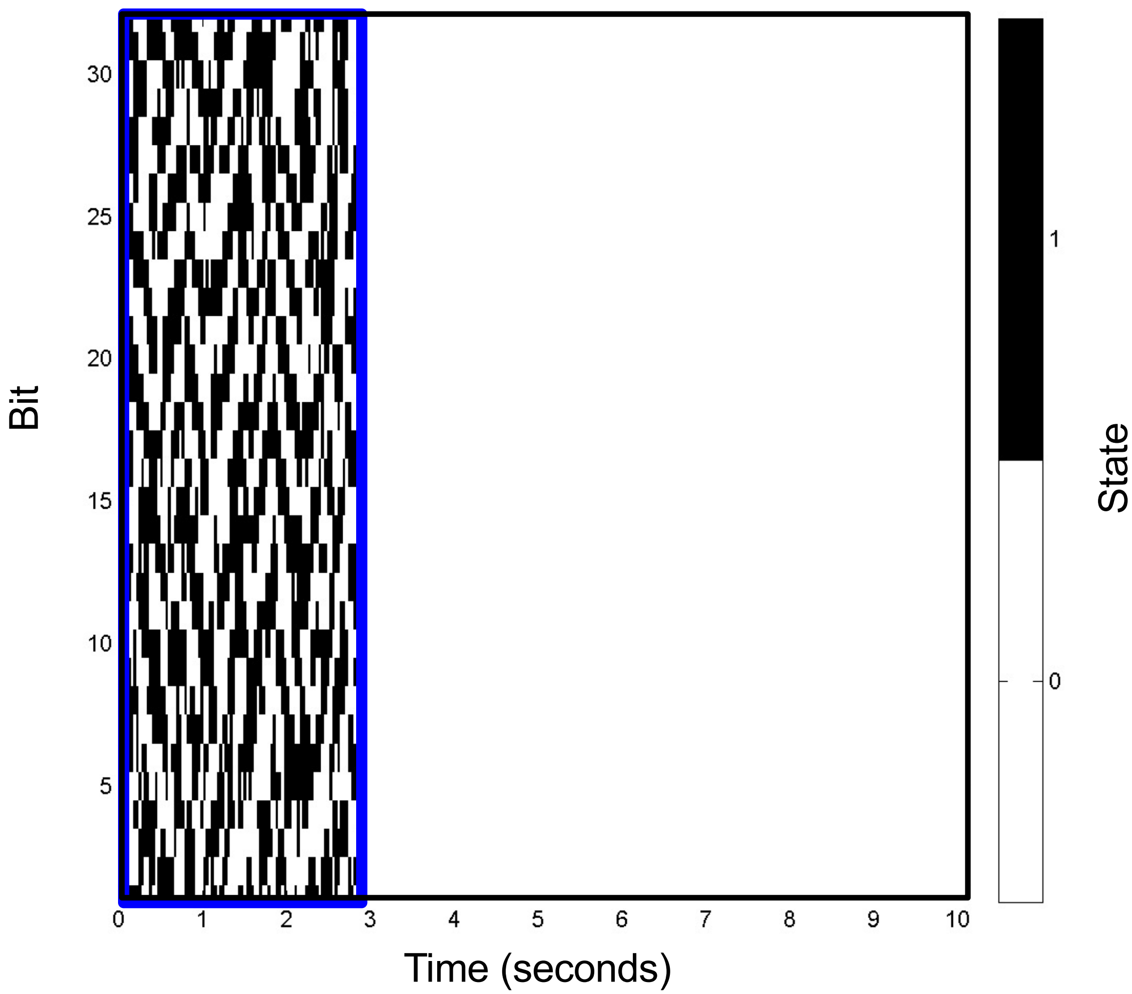
Fingerprints (Philips)



Sub-fingerprint:

- 32-bit vector
- Not characteristic enough

Fingerprints (Philips)



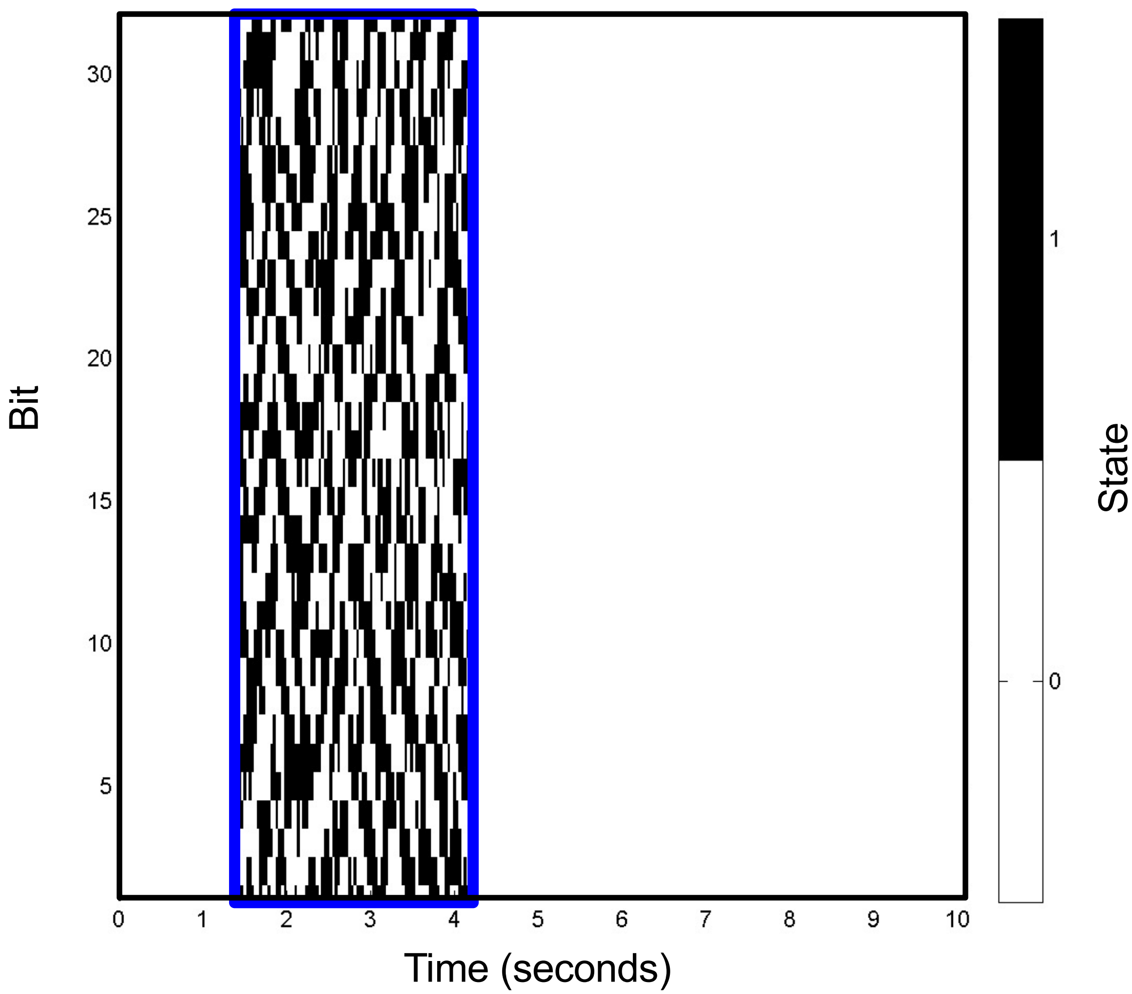
Sub-fingerprint:

- 32-bit vector
- Not characteristic enough

Fingerprint-block:

- 256 consecutive sub-fingerprints
- Covers roughly 3 seconds
- Overlapping

Fingerprints (Philips)



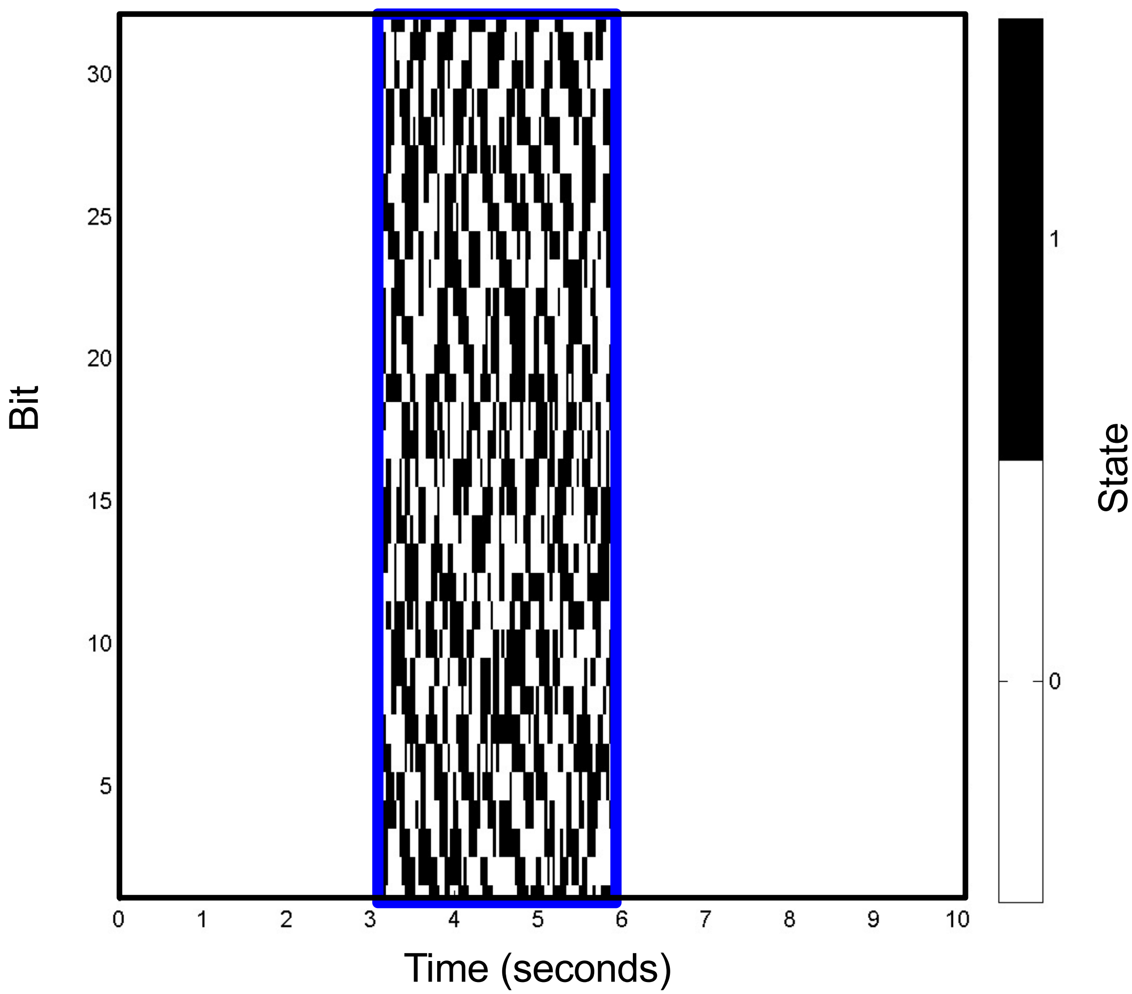
Sub-fingerprint:

- 32-bit vector
- Not characteristic enough

Fingerprint-block:

- 256 consecutive sub-fingerprints
- Covers roughly 3 seconds
- Overlapping

Fingerprints (Philips)



Sub-fingerprint:

- 32-bit vector
- Not characteristic enough

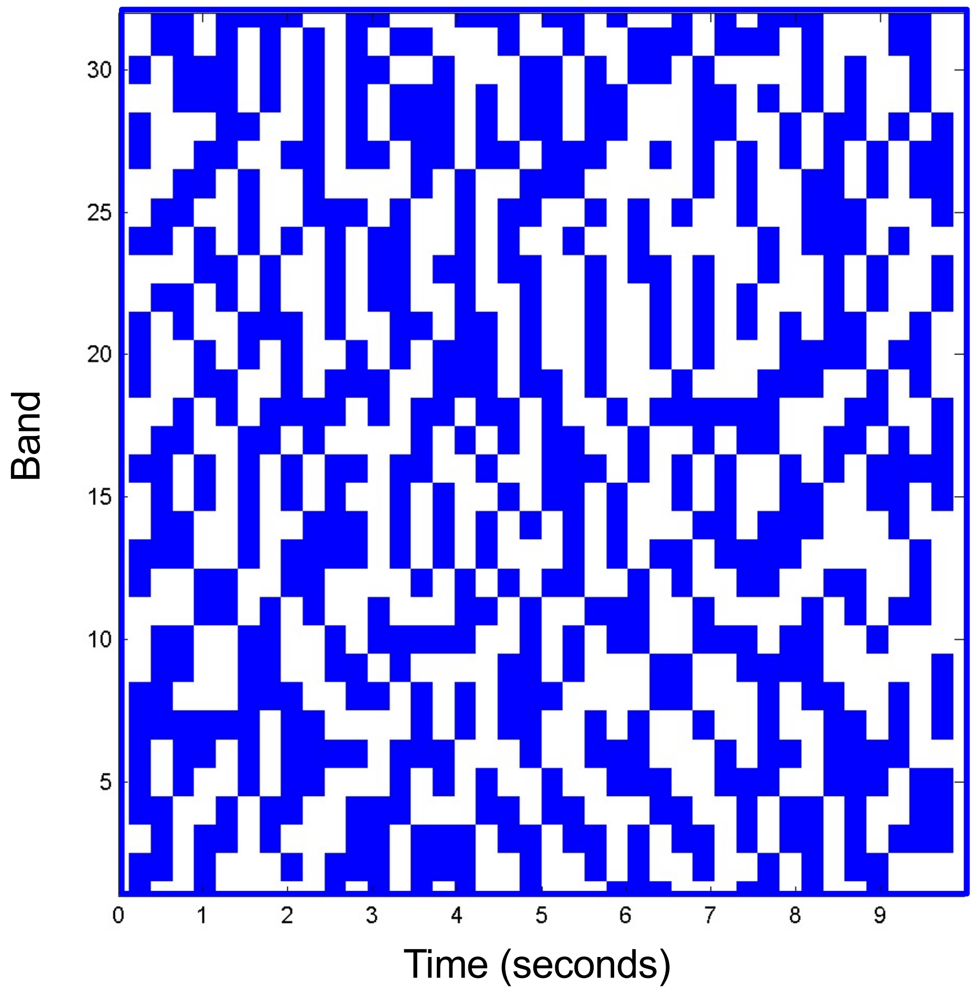
Fingerprint-block:

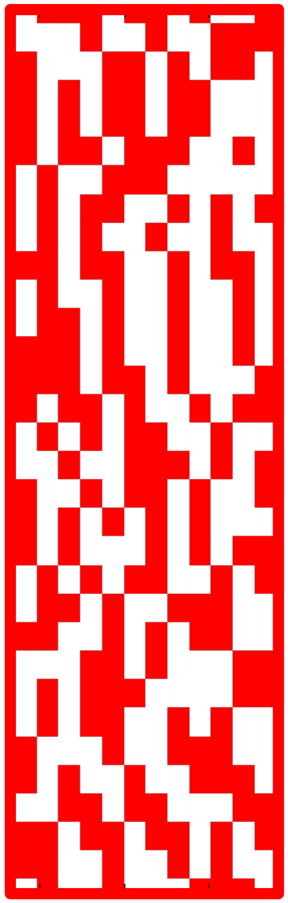
- 256 consecutive sub-fingerprints
- Covers roughly 3 seconds
- Overlapping

Matching Fingerprints (Philips)

Database document
(fingerprint-blocks)

Query document
(fingerprint-block)

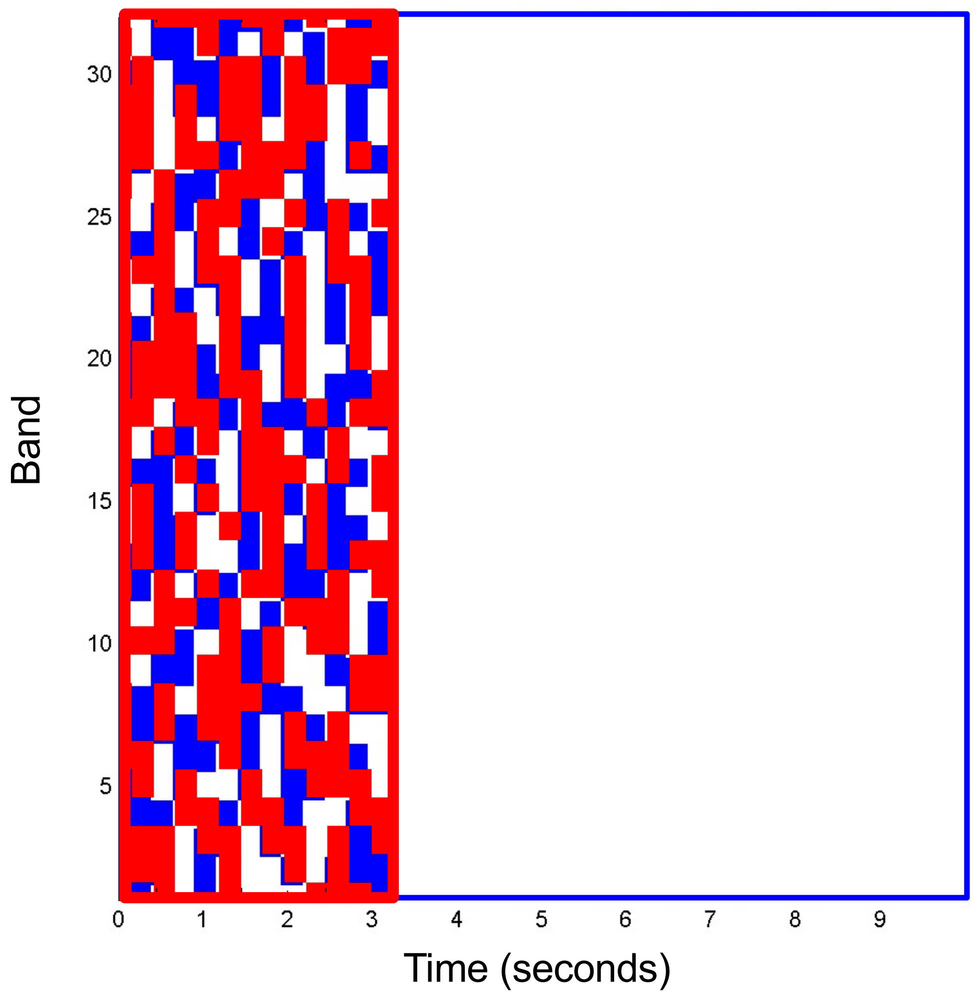




Matching Fingerprints (Philips)

Database document
(fingerprint-blocks)

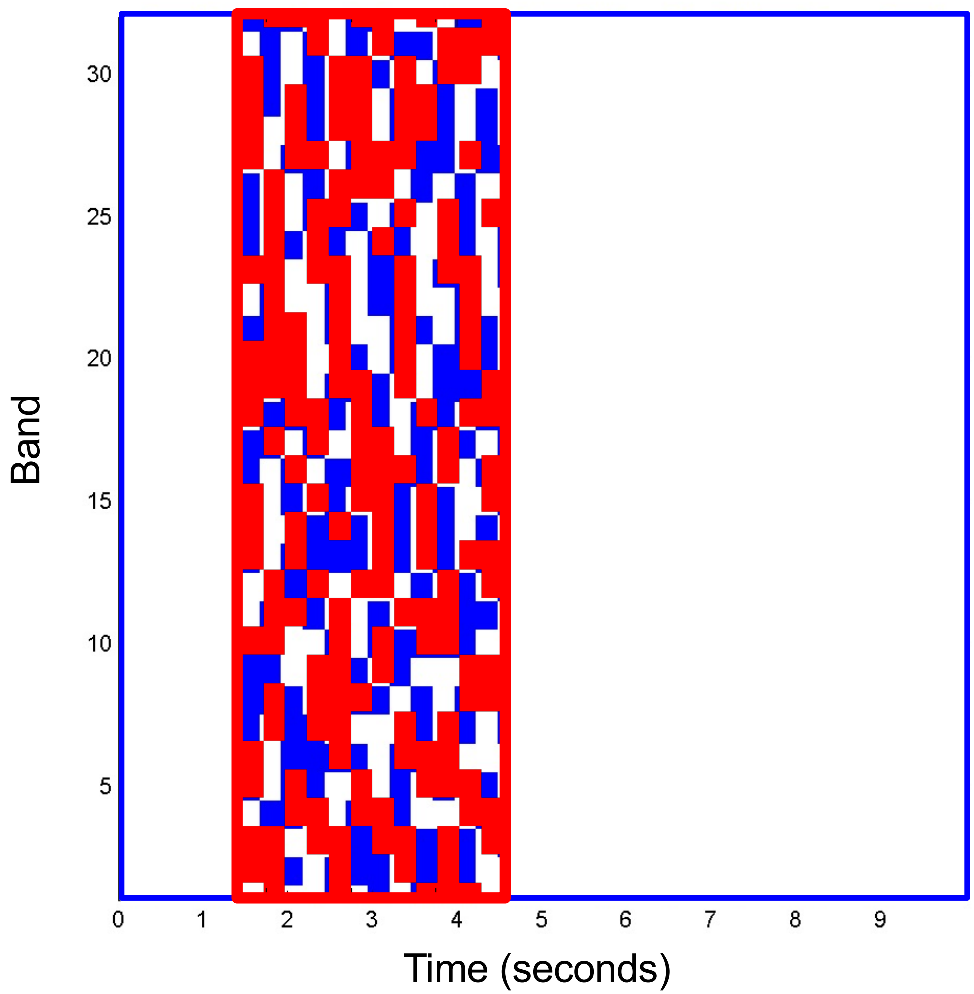
Query document
(fingerprint-block)



1. Shift query across database document
2. Calculate a block-wise bit-error-rate (BER)

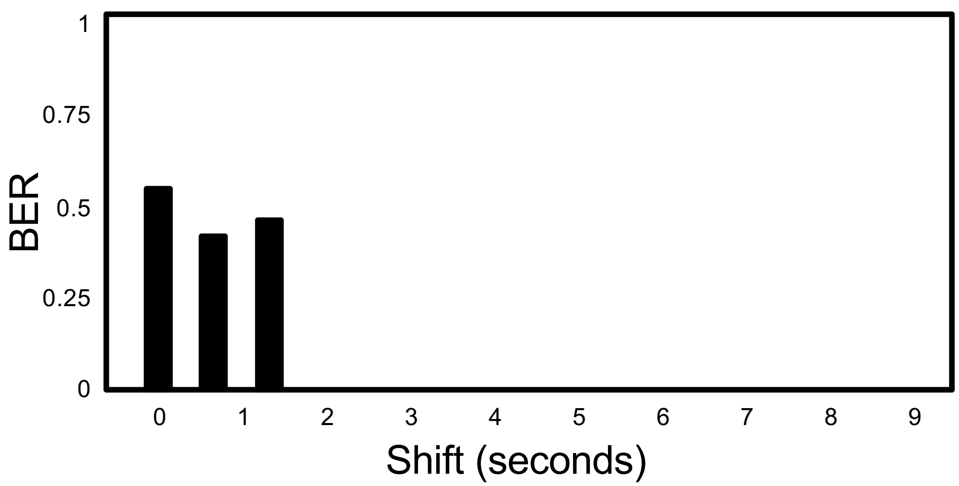
Matching Fingerprints (Philips)

Database document
(fingerprint-blocks)



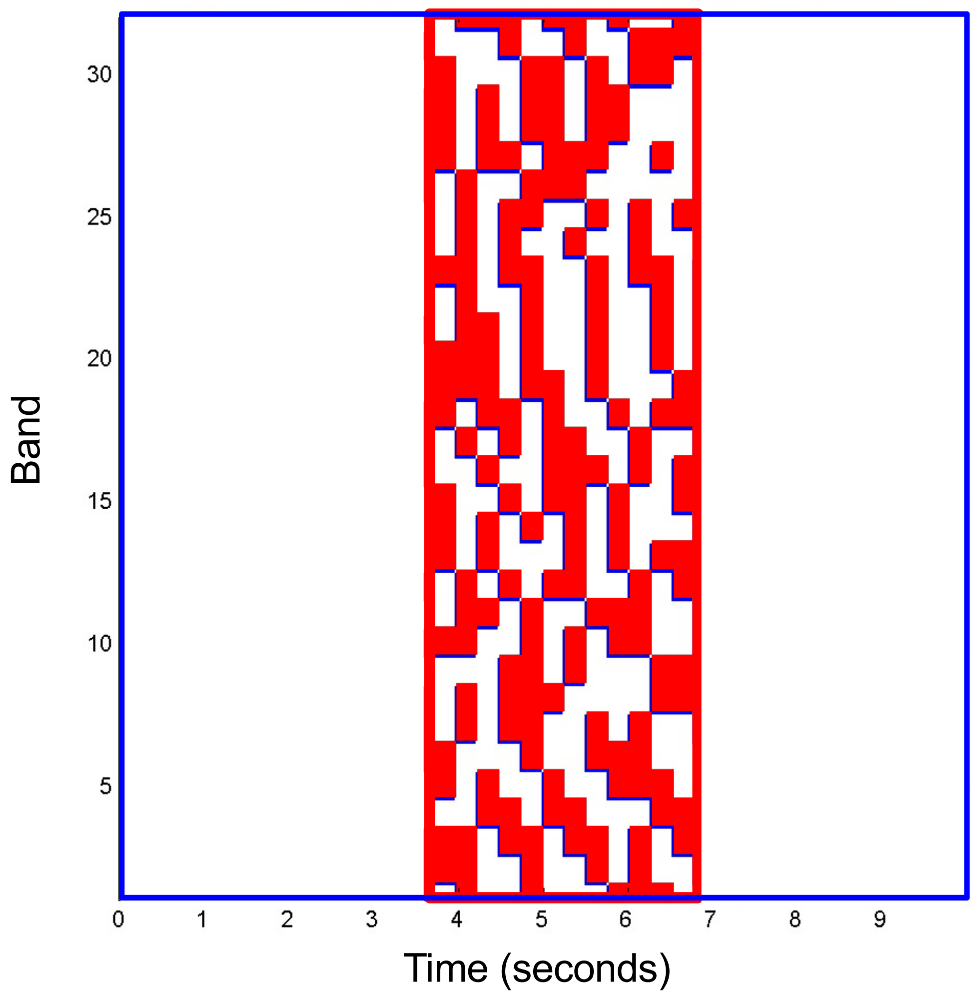
Query document
(fingerprint-block)

1. Shift query across database document
2. Calculate a block-wise bit-error-rate (BER)



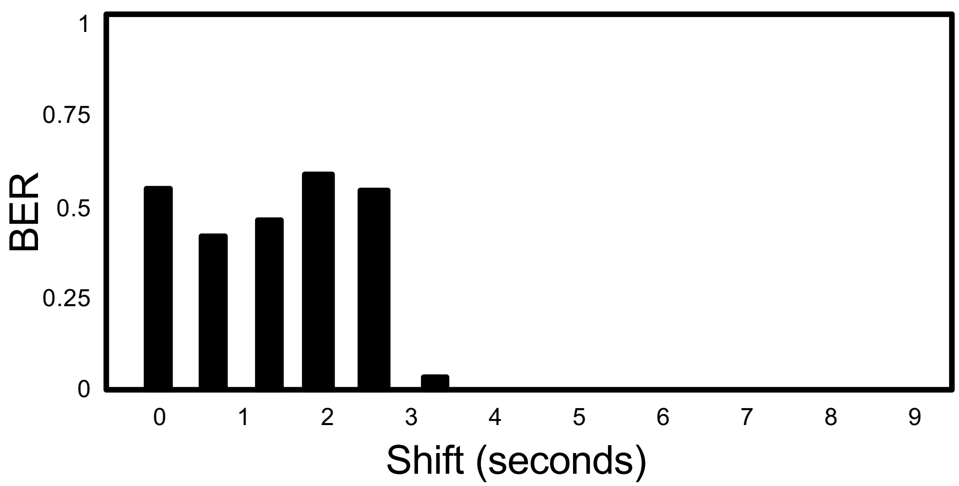
Matching Fingerprints (Philips)

Database document
(fingerprint-blocks)



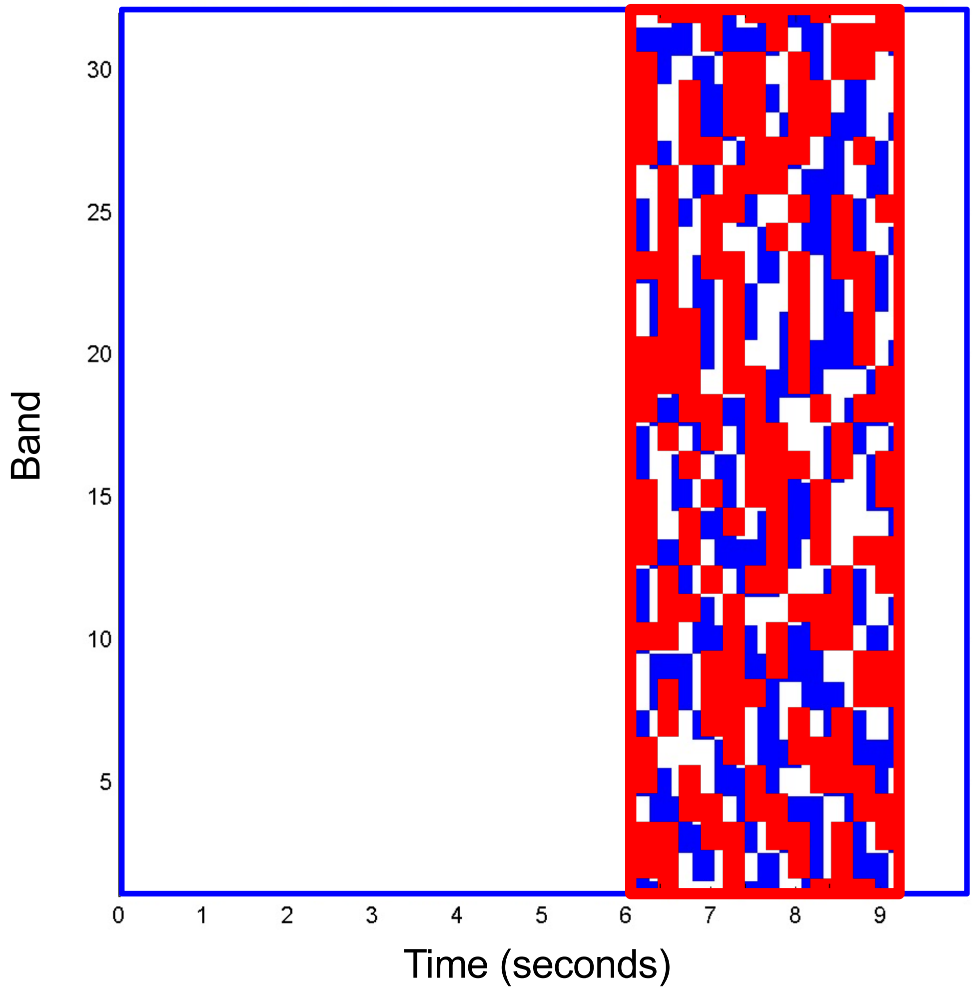
Query document
(fingerprint-block)

1. Shift query across database document
2. Calculate a block-wise bit-error-rate (BER)



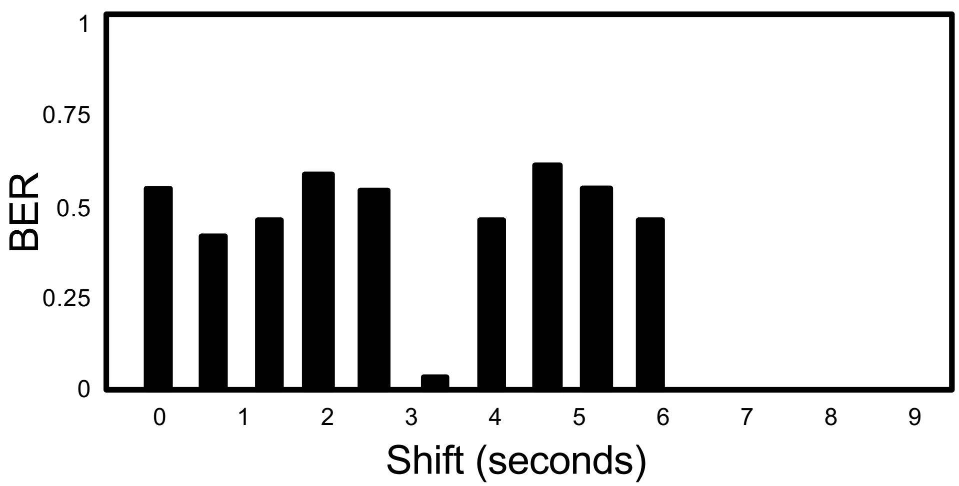
Matching Fingerprints (Philips)

Database document
(fingerprint-blocks)



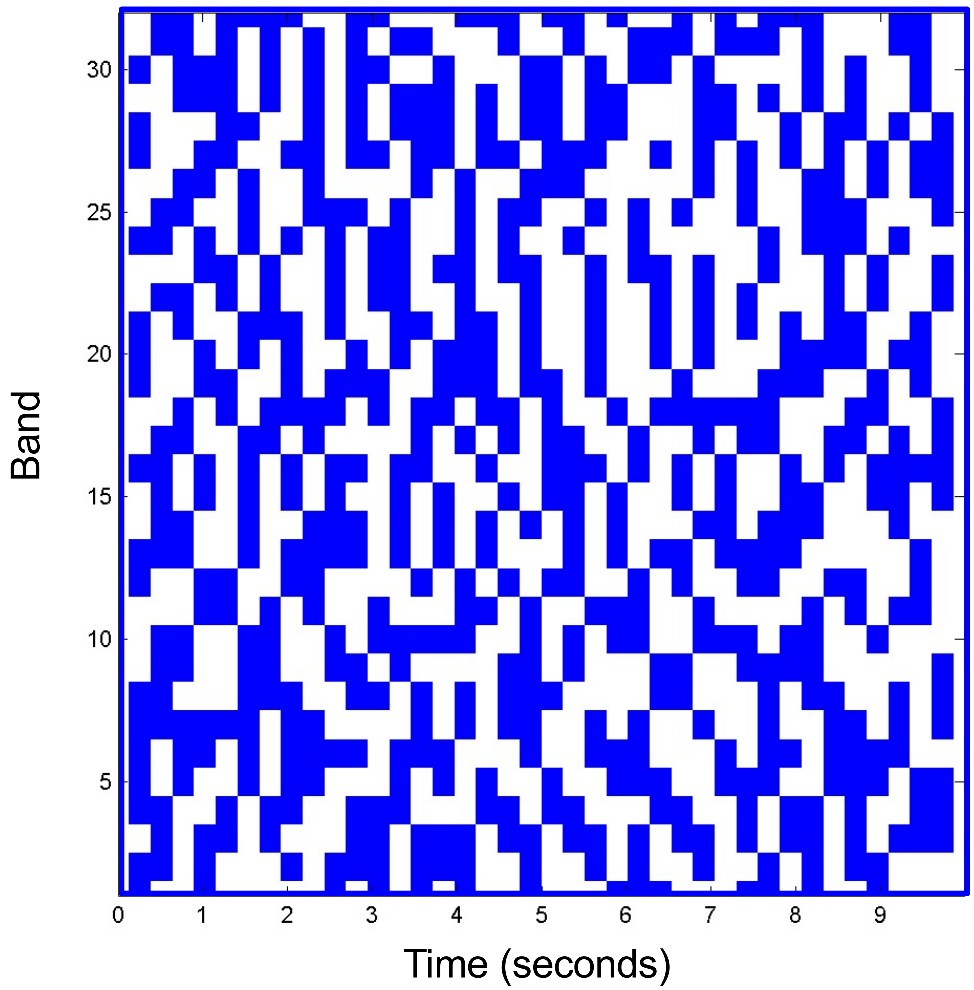
Query document
(fingerprint-block)

1. Shift query across database document
2. Calculate a block-wise bit-error-rate (BER)



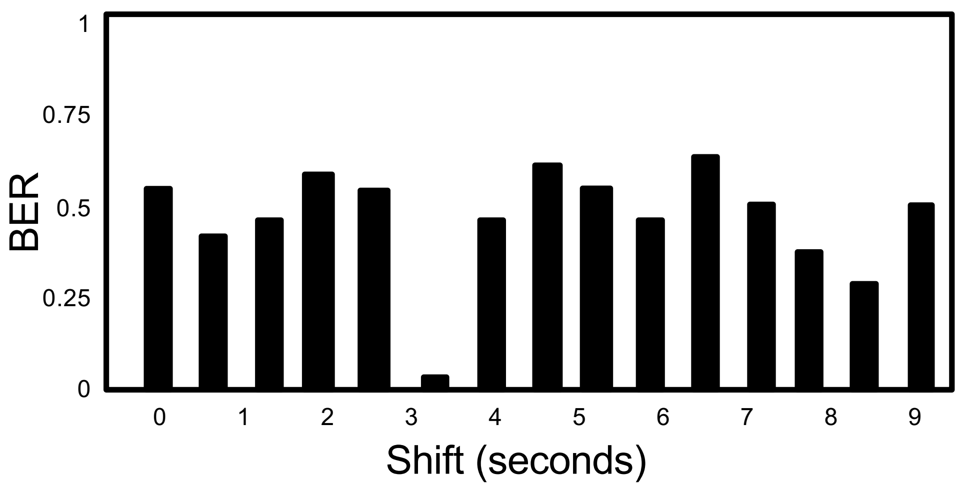
Matching Fingerprints (Philips)

Database document
(fingerprint-blocks)



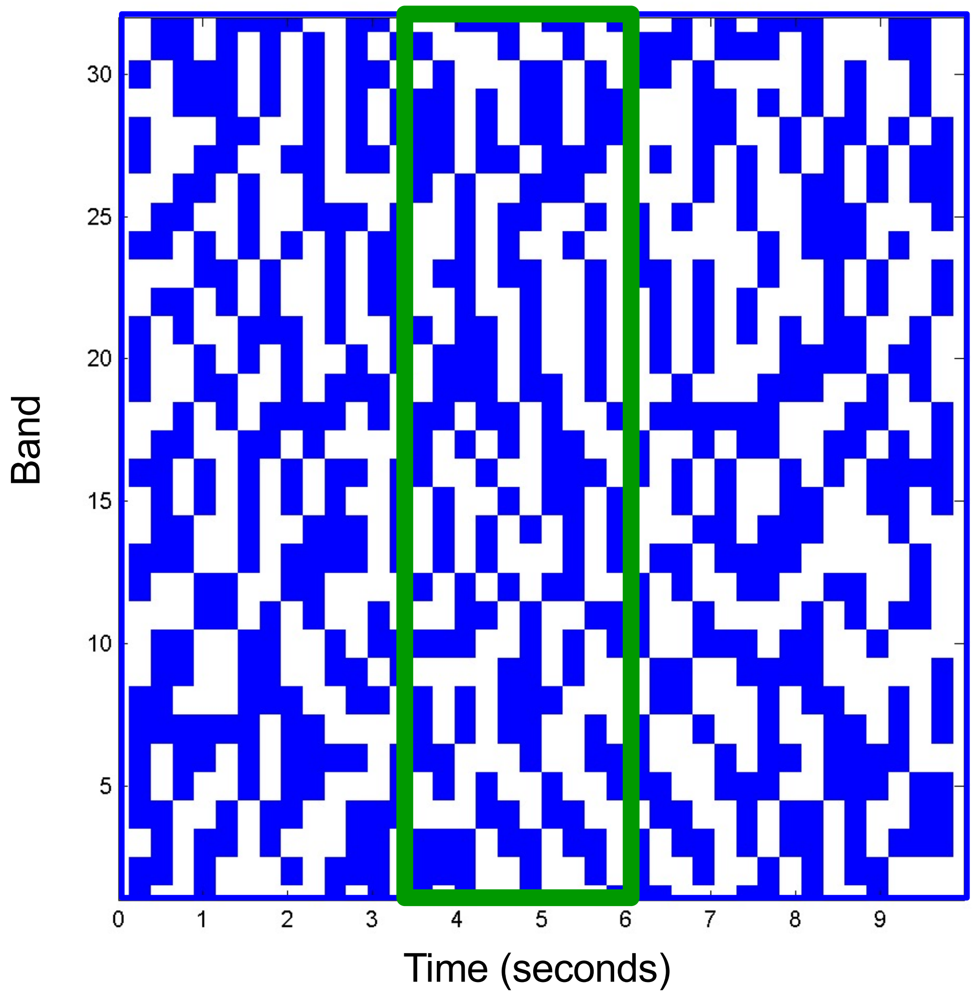
Query document
(fingerprint-block)

1. Shift query across database document
2. Calculate a block-wise bit-error-rate (BER)



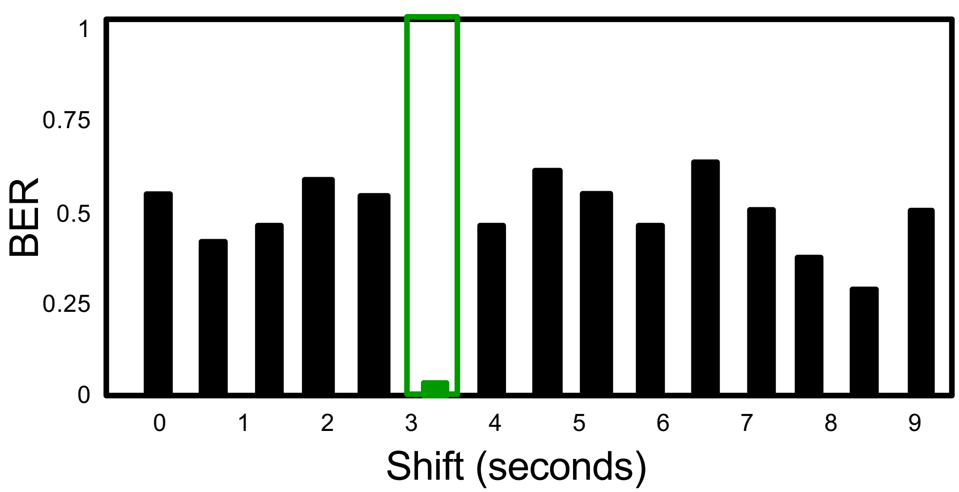
Matching Fingerprints (Philips)

Database document
(fingerprint-blocks)



Query document
(fingerprint-block)

1. Shift query across database document
2. Calculate a block-wise bit-error-rate (BER)
3. Low BER indicates hit



Indexing (Philips)

Note:

- Individual sub-fingerprints (32 bit) are not characteristic
- Fingerprint blocks (256 sub-fingerprints, 8 kbit) are used

Problem:

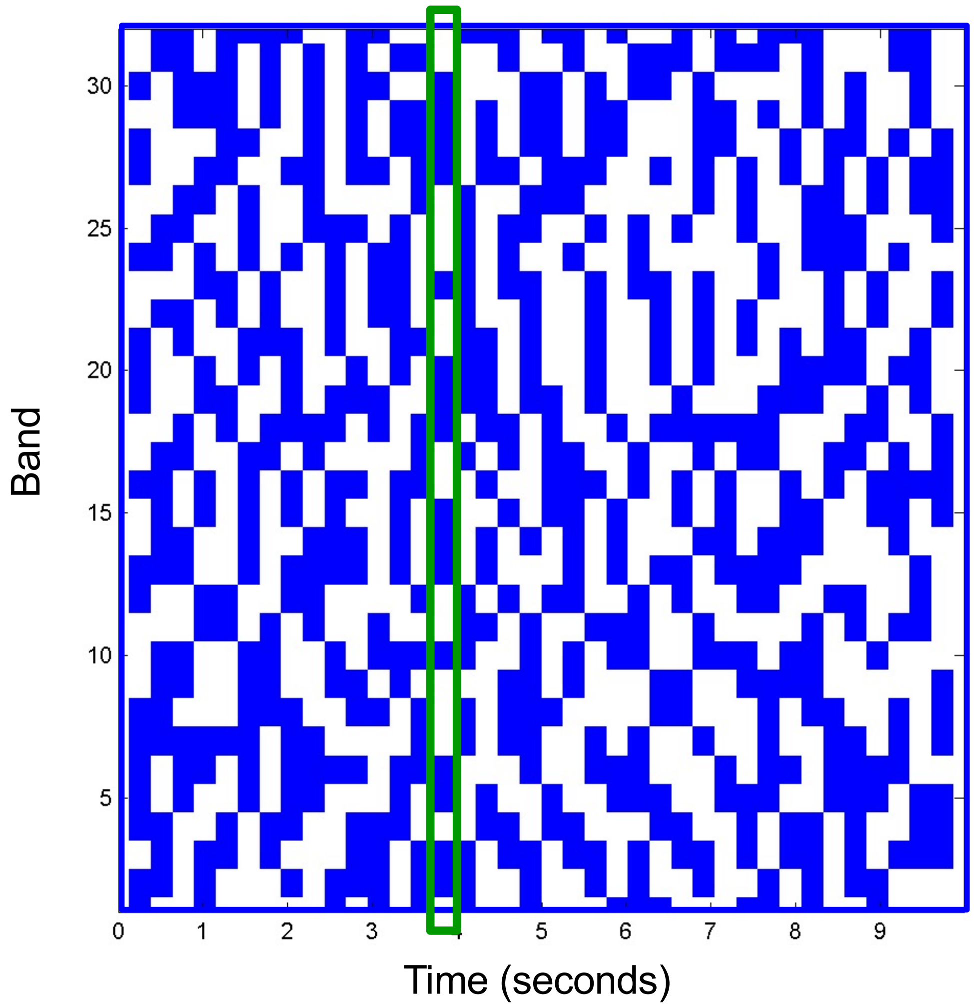
- Computation of BER between query fingerprint-block and every database fingerprint-block is expensive
- Chance that a complete fingerprint-block survives is low
- Exact hashing problematic

Strategy:

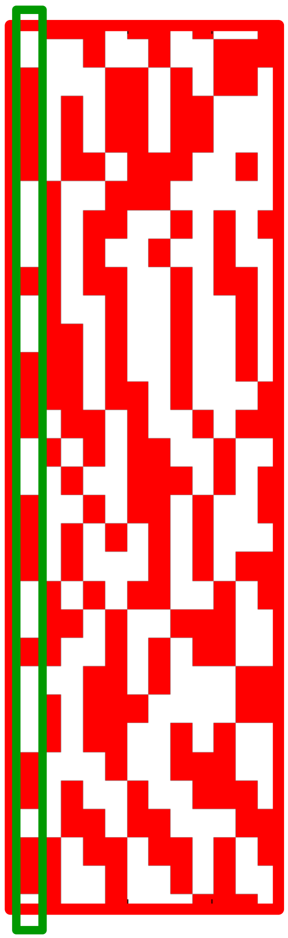
- Only sub-fingerprints are indexed using hashing
- Exact sub-fingerprint matches are used to identify candidate fingerprint-blocks in database.
- BER is only computed between query fingerprint-block and candidate fingerprint-blocks
- Procedure is terminated when database fingerprint-block is found, where BER falls below a certain threshold

Indexing (Philips)

Database document (fingerprint-blocks)



Query document (fingerprint-block)

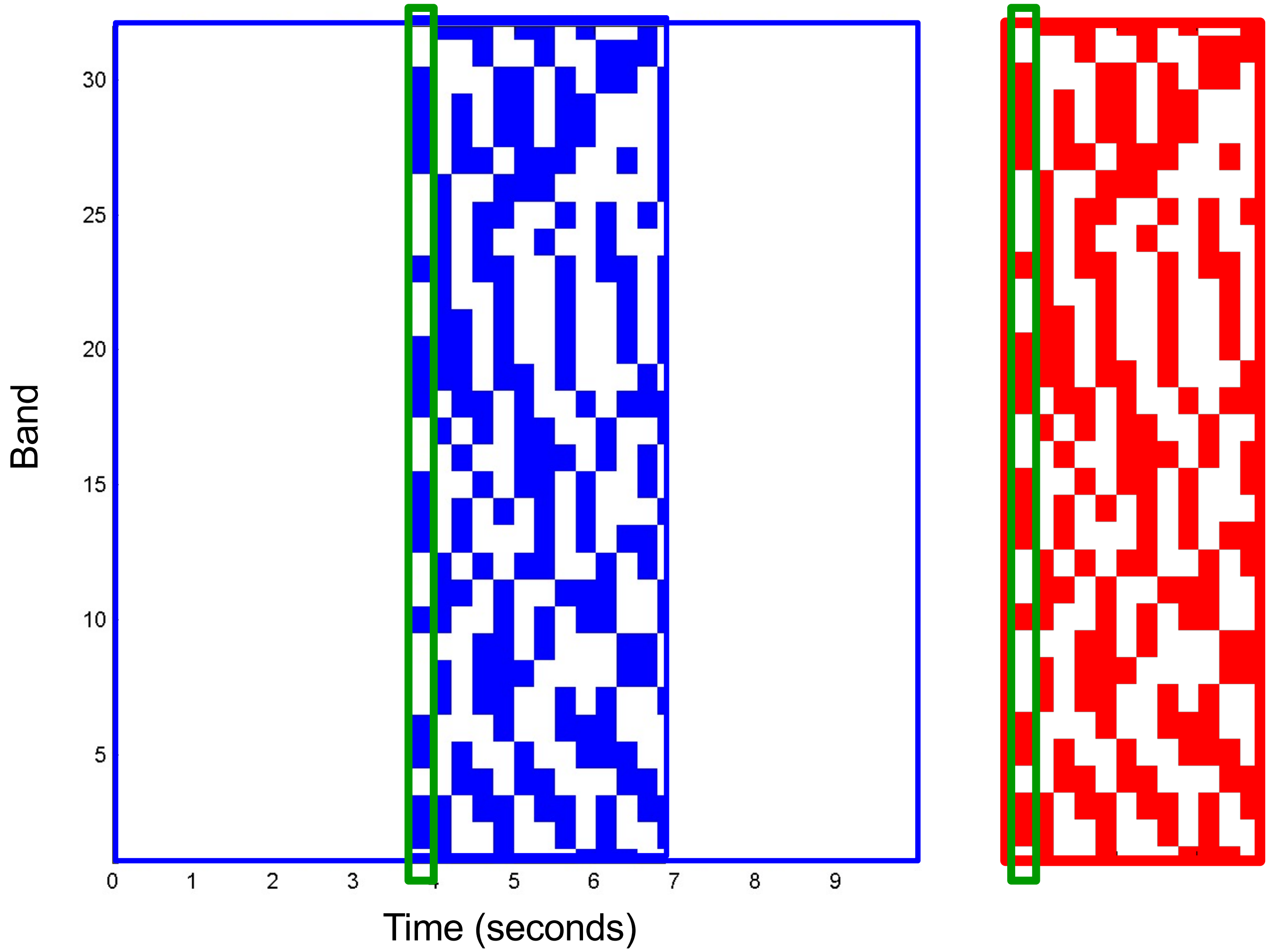


1. Efficient search for exact matches of sub-fingerprints (anchor points)

Indexing (Philips)

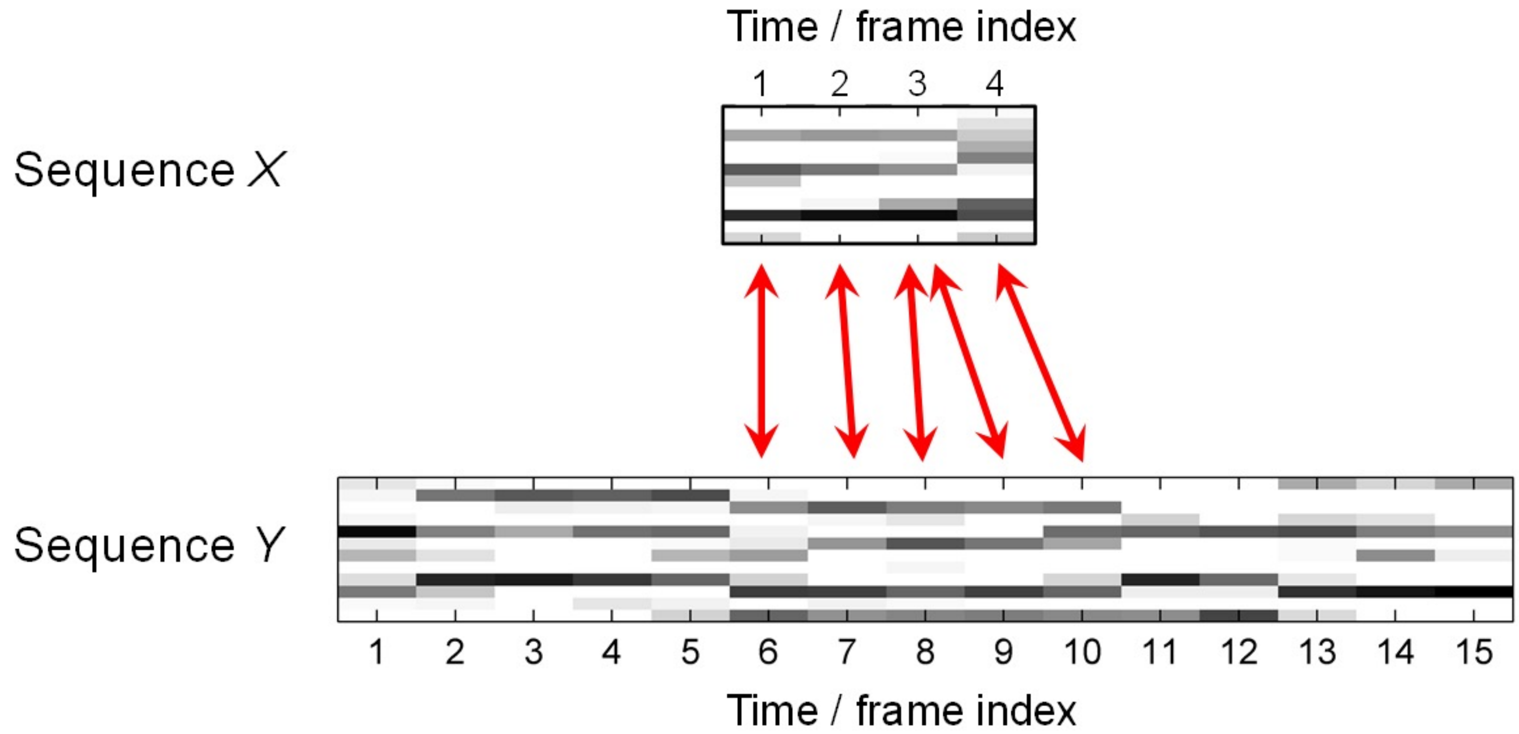
Database document (fingerprint-blocks)

Query document (fingerprint-block)



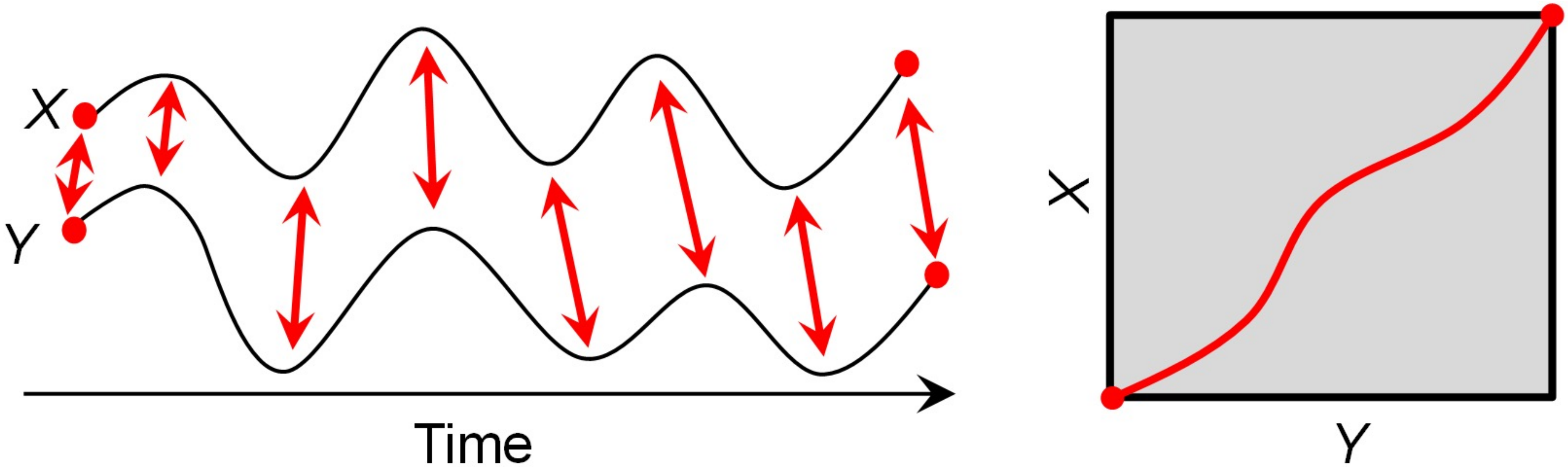
1. Efficient search for exact matches of sub-fingerprints (anchor points)
2. Calculate BER only for blocks containing anchor points

How to account for variation in time and pitch?



How to account for variation in time and pitch?

A version of the similarity-matrix method is incorporated into the match process to account for variation in time:



How to account for variation in time and pitch?

To account for differences in pitch, instead of storing frequencies as fixed, absolute values, store the first derivative (first difference) of the pitches relative to some initial pitch; for example, if C is the first note, we can indicate the number of semi-tones between each successive note:

0 2 2 1 -3 2 -4 7