Overview (Audio Retrieval)

- Audio identification (audio fingerprinting)
- Audio matching
- Cover song identification

Full Disclosure: These slides are by Meinhard Muller, from ISMIR 2012, with minimal changes by me.
Overview (Audio Retrieval)

- Audio identification (audio fingerprinting)
- Audio matching
- Cover song identification

Audio 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 are not identifying a piece of music but a specific recording of the piece
Many Applications!

- “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, ...
- **Audio Database Retrieval** (“Query by Humming”)

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

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
- Pitching (audio played faster or slower)
- Equalization
- Compression artifacts
- …

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

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

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

Literature (Audio Identification)

- Allamanche et al. (AES 2001)
- Cano et al. (AES 2002)
- Haitsma/Kalker (ISMIR 2002)
- Kurth/Clausen/Ribbrock (AES 2002)
- Wang (ISMIR 2003)
- Dupraz/Richard (ICASSP 2010)
- Ramona/Peeters (ICASSP 2011)
Literature (Audio Identification)

- Allamanche et al. (AES 2001)
- Cano et al. (AES 2002)
- Haitsma/Kalker (ISMIR 2002)
- Kurth/Clausen/Ribbrock (AES 2002)
- Wang (ISMIR 2003)
- Dupraz/Richard (ICASSP 2010)
- Ramona/Peeters (ICASSP 2011)

Fingerprints (Shazam)

Steps:
1. Spectrogram
2. Peaks
   (local maxima)

- Efficiently computable
- Standard transform
- Robust
Fingerprints (Shazam)

Steps:
1. Spectrogram
2. Peaks

Robustness:
- Noise, reverb, room acoustics, equalization
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

(constellation map)
1. Shift query across database document
2. Count matching peaks
Matching Fingerprints (Shazam)

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

Database document (constellation map)
Query document (constellation map)

#(matching peaks)

Shift (seconds)
Matching Fingerprints (Shazam)

1. Shift query across database document
2. Count matching peaks
Matching Fingerprints (Shazam)

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 (quantized) frequencies
- 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

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 pairs 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).
Indexing (Shazam)

Definitions:
- $N$ = number of spectral peaks
- $p$ = probability that a spectral peak can be found in (noisy and distorted) query
- $F$ = fan-out of target zone, e.g. $F = 10$
- $B$ = #bits used to encode spectral peaks and time difference

Consequences:
- $F \cdot N$ = #tokens to be indexed
- $2^{B+B}$ = increase of specificity ($2^{B+B}$ instead of $2^B$)
- $p^2$ = probability of a hash to survive
- $p \cdot (1-(1-p)^F)$ = probability that, at least, one hash survives per anchor point

Example: $F = 10$ and $B = 10$
- Memory requirements: $F \cdot N = 10 \cdot N$
- Speedup factor: $2^{B+B} / F^2 \sim 10^6 / 10^2 = 10000$ ($F$ times as many tokens in query and database, respectively)

Conclusions (Shazam)

Many parameters to choose:
- Temporal and spectral resolution in spectrogram
- Peak picking strategy
- Target zone and fan-out parameter
- Hash function
- …
Literature (Audio Identification)

- Allamanche et al. (AES 2001)
- Cano et al. (AES 2002)
- Haitsma/Kalker (ISMIR 2002)
- Kurth/Clausen/Ribbrock (AES 2002)
- Wang (ISMIR 2003)
- Dupraz/Richard (ICASSP 2010)
- Ramona/Peeters (ICASSP 2011)

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

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 (simultaneously along time and frequency axes)
- Sequence of 32-bit vectors
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)

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

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

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

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)

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

Conclusions (Audio Identification)

- Basic techniques used in Shazam and Philip systems
- Many more ways to define robust audio fingerprints
- Delicate trade-off between specificity, robustness, and efficiency
- Audio recording is identified (not a piece of music)
- Does not allow for identifying studio recording using a query taken from live recordings
- Does not generalize to identify different interpretations or versions of the same piece of music