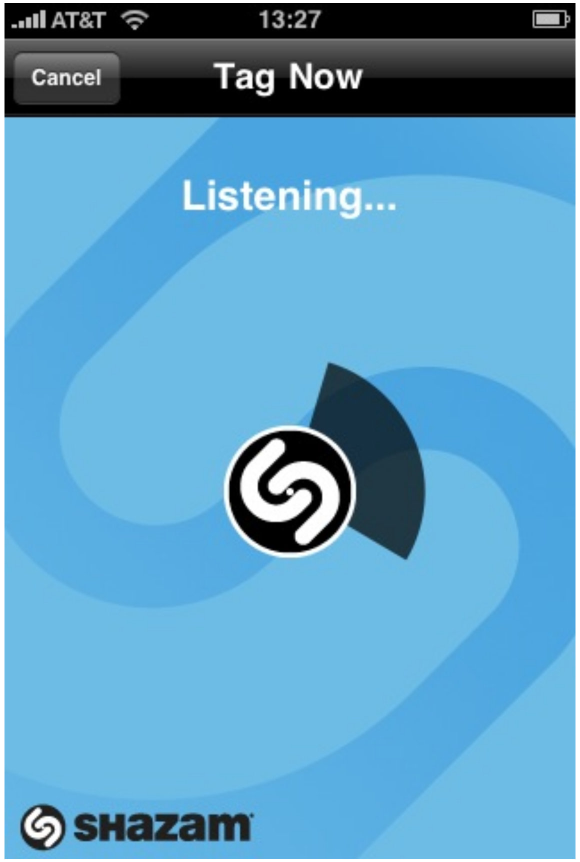
Lecture 13: Audio Fingerprinting

1. The Fingerprinting Problem
2. Frame-Based Approach
3. Landmark Approach

Dan Ellis
Dept. Electrical Engineering, Columbia University
dpwe@ee.columbia.edu http://www.ee.columbia.edu/~dpwe/e4896/

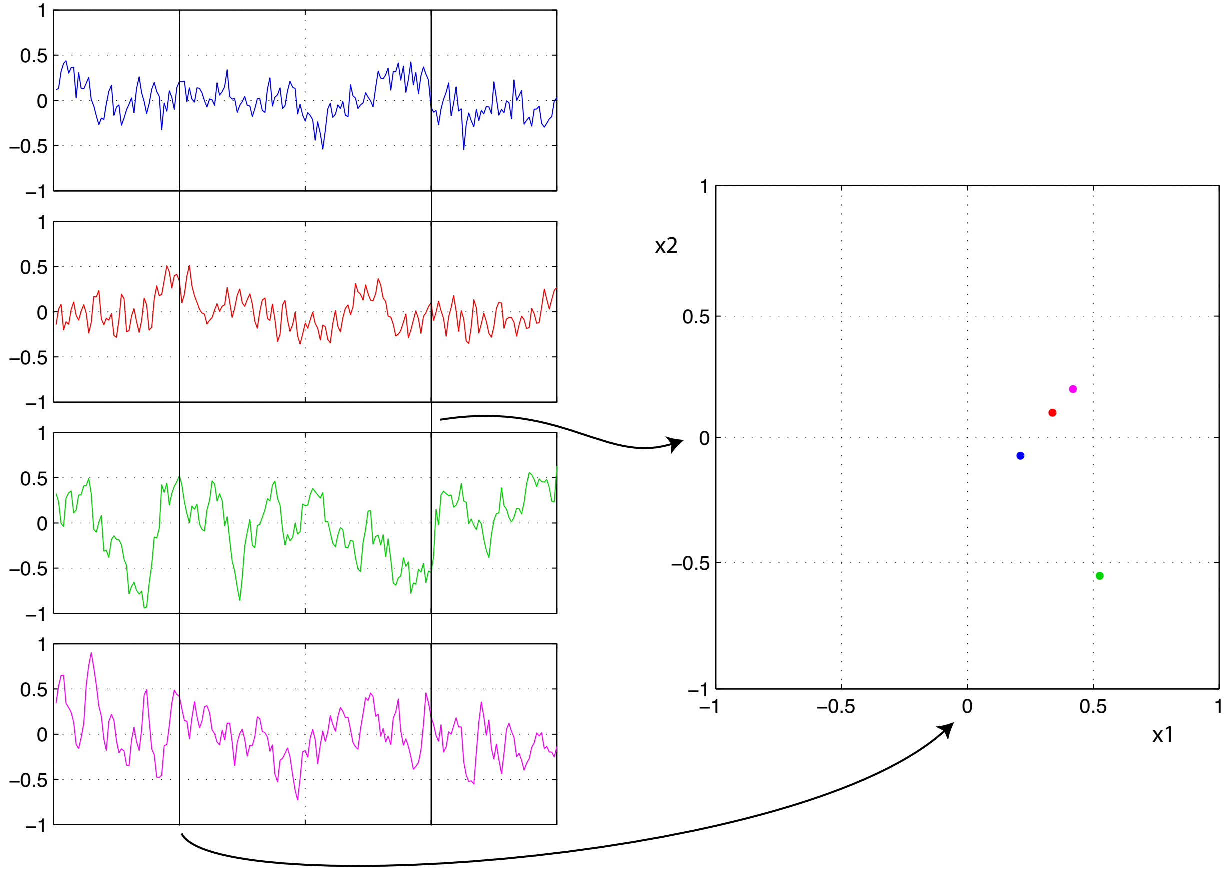
1. The Fingerprinting Problem

- **Audio Fingerprinting: Known-Item search**
  - for the exact same performance (no "cover versions")
  - despite differences in audio channel, encoding, noise etc.
- **Applications**
  - media monitoring
  - metadata reconciliation
  - "what's that song?"



A Simple Fingerprint

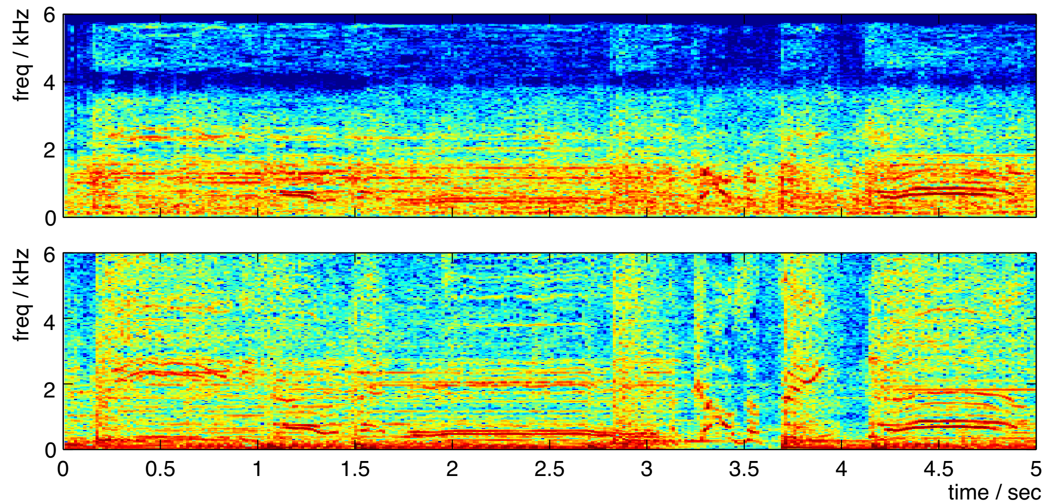
- "Fingerprint" is a compact record sufficient to uniquely identify an example
  - difficulty depends on item density, noise



  - hash functions?

Fingerprinting Challenges

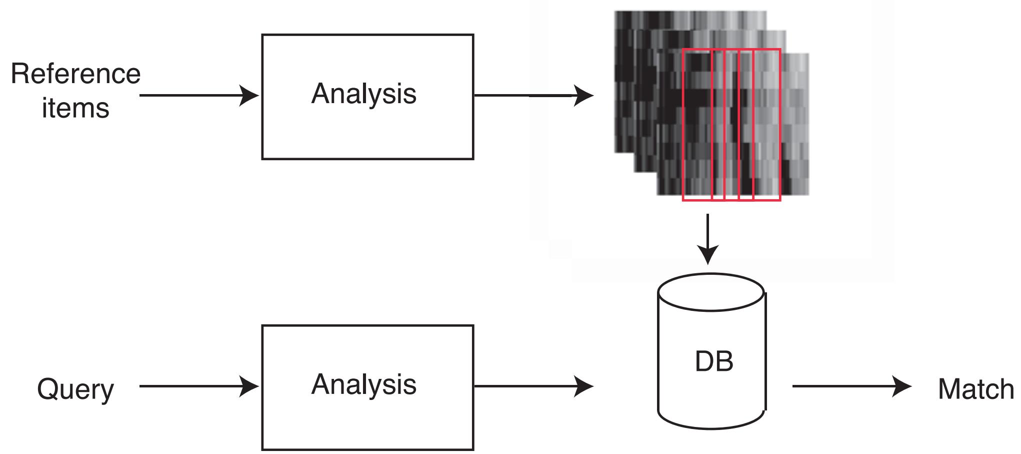
- Immunity to channel (speaker/mic), added noise
  - the "coffeeshop" problem



- Recognize fragments from anywhere in the track
  - the shorter the better
- Large corpus of reference items
- False alarm vs. false reject

2. Frame-Based Approaches

- Standard audio-processing paradigm
  - chop-up waveform into frames
  - each frame → feature vector
  - match on a sequence of feature vectors



- Challenges
  - make the features invariant to channel variations
  - make features insensitive to timing skew / offset
  - computational efficiency

Channel Immunity

Haitsma & Kalker 2003

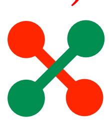
- **Audio matching should be invariant to**
  - lossy encoding (low-bitrate MP3)
  - dynamic range compression (per band?)
  - added noise (quantization, environment noise)

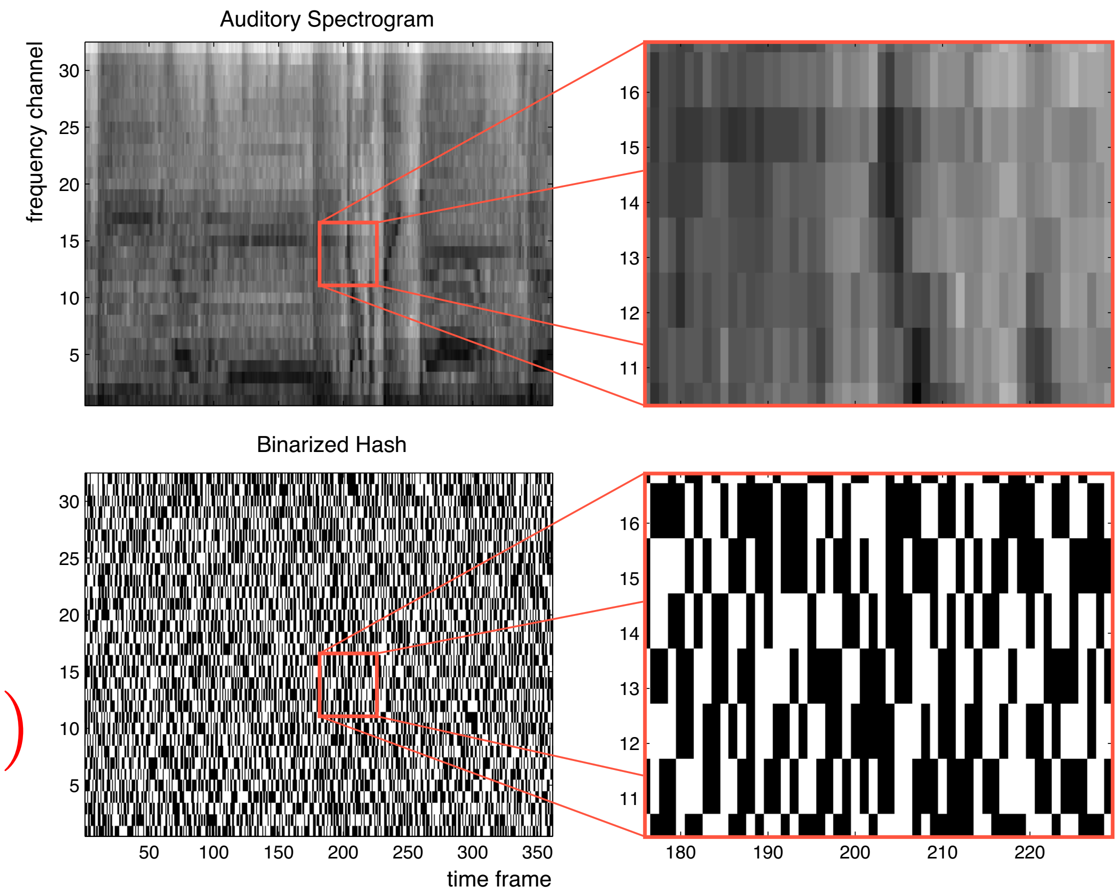
- **Local threshold**
  - auditory magnitude-spectrogram $X(t, f)$
  - bitmask "hash":

$$B(t, f) = 1 \quad \text{iff}$$

$$X(t, f) + X(t+1, f+1)$$

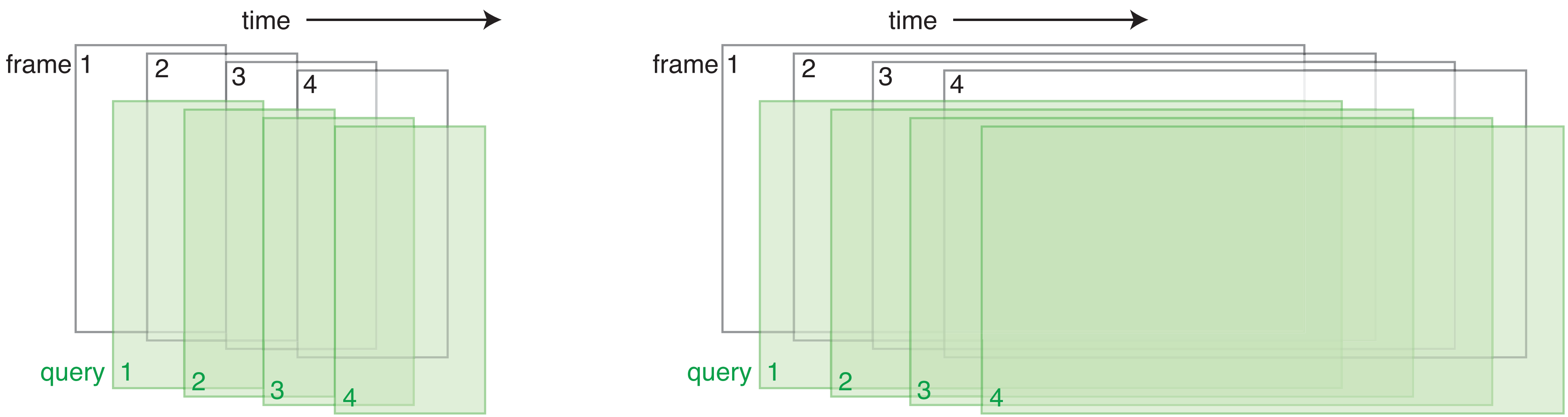
$$> X(t, f+1) + X(t+1, f)$$





Timing Skew

- What happens if reference frames are out of sync with test frames?
  - make frame length much longer than "hop"

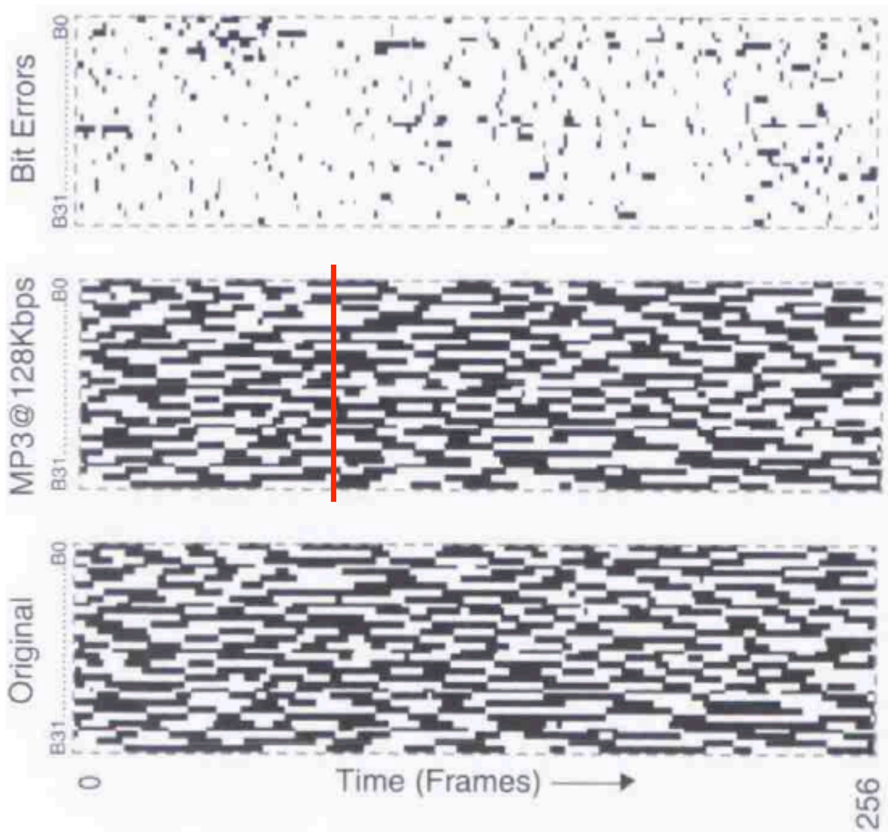


  - → features are very smooth in time

Retrieval & Matching

Haitsma & Kalker 2003

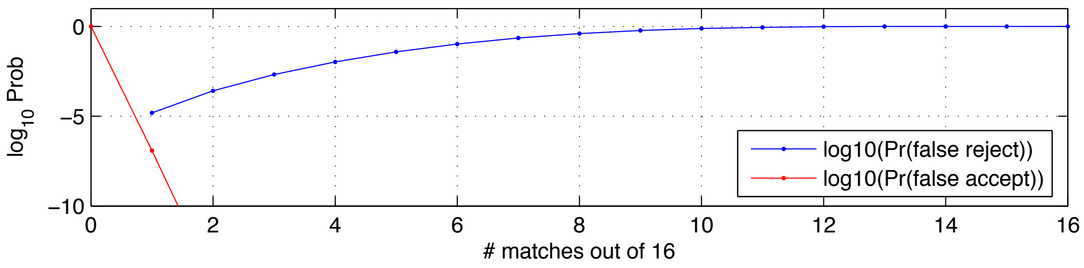
- Matching is by Hamming Distance between query & ref
  - use 256 x 32 bit frames (3 sec @ 11.6 ms frames)
  - 10k tracks ~ 200M frames



- Only check near exact match of one 32-bit word
  - hash table index of occurrences of all 2^{32} = 4G values
  - repeat for all 256 columns
  - (can also test all 32 one-bit differences)

False Alarms vs. False Reject

- **One 32-bit hash**
  - a = Pr(same hash | same audio) ~ 0.5 (dep. noise?)
  - b = Pr(same hash | different audio) ~ $1/2^{32}$
    (or $33/2^{32}$ ~ $1/2^{27}$ = 10^{-8} for 1-bit diffs)
  - Pr(false match in L frames) = $1 - (1 - b)^L$ (~Lb)
  - L = 200M → Pr(false match) = 0.78
- **K 32-bit hashes (e.g. K=8)**
  - Pr(all match | diff audio) = b^K ~ 10^{-65}
  - Pr(all match | same audio) = a^K ~ 1/256
- **K matches out of N (e.g. 4 of 16 – Binomial)**
  - Pr(false reject) = Pr(B(16, a) < 4) = 0.0106
  - Pr(false accept)
    = Pr(B(16, b) ≥ 4)
    ~ 10^{-29} (~b^4)



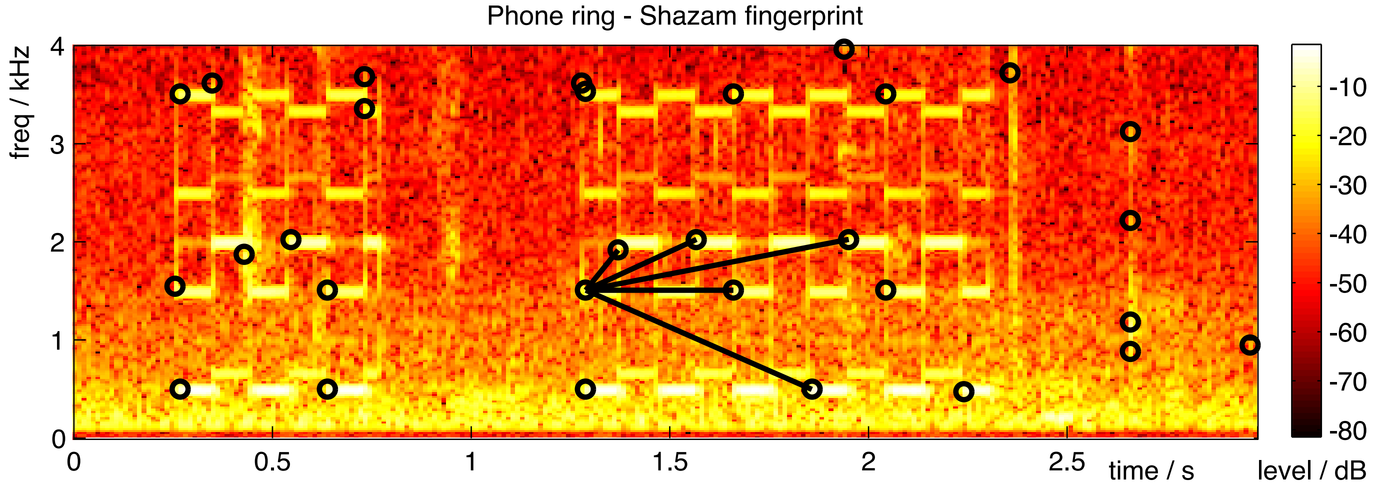
3. Landmark Approach

Wang 2003, 2006

- Idea:
  Use structures in audio as time reference instead of arbitrary time frames
  - eliminates "framing errors"
- Another idea:
  Use individual, spectrally-local structures as component hashes
  - robustness to missing frequency bands
- Use time-frequency peaks
  - i.e. onsets of specific harmonics
  - highest energy → most robust to noise
  - the Shazam algorithm

Shazam Landmarks

- Find local peaks in spectrogram
  - but only ~256 different frequencies - common
- Join them into pairs
  - look for 2nd peak within some window

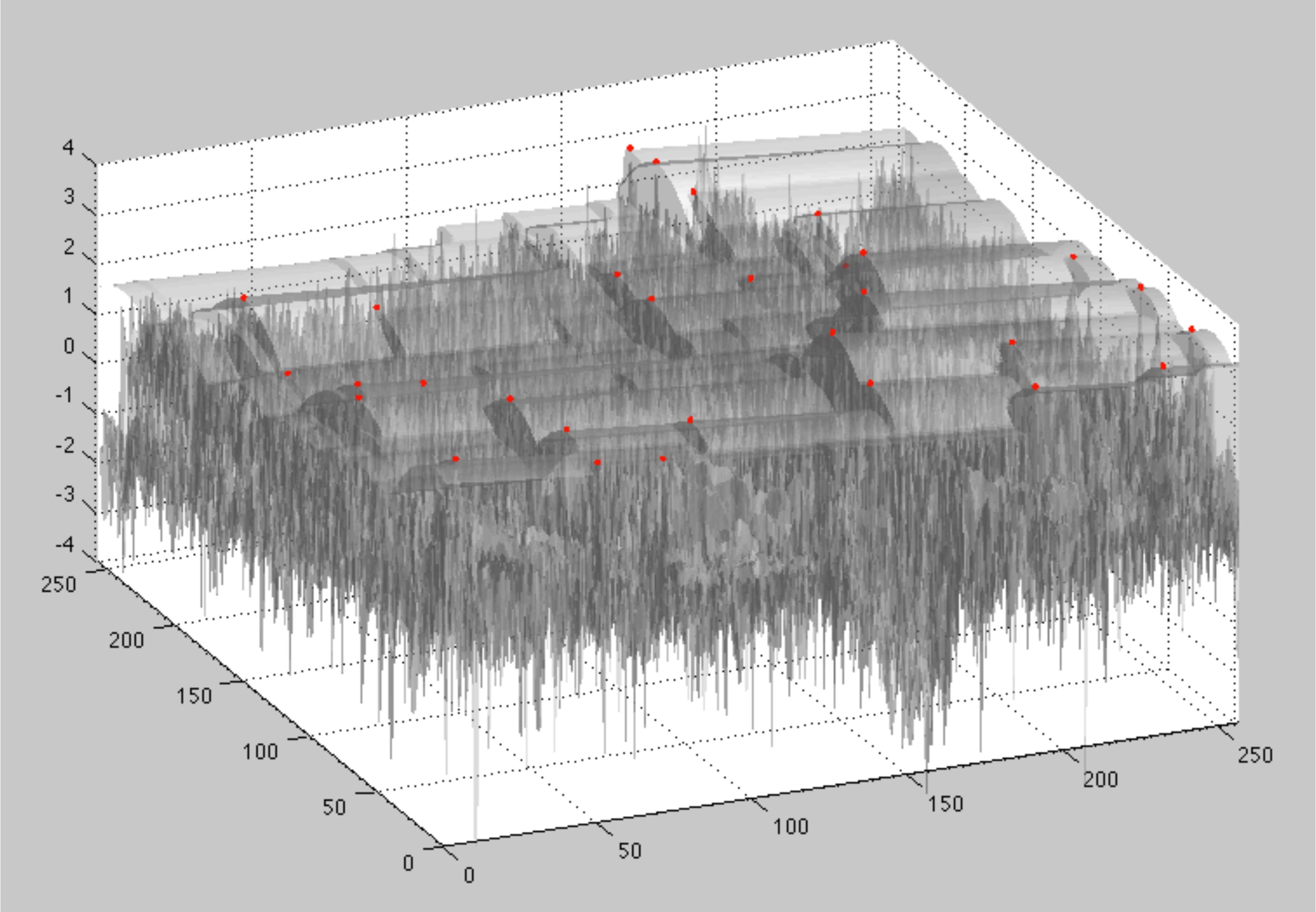


  - hash {start freq, end freq, time diff}
    = 256 x 256 x 64 = 4M distinct patterns
  - build index: [hash → {track_ID, offset_sec}]

Selecting Peaks

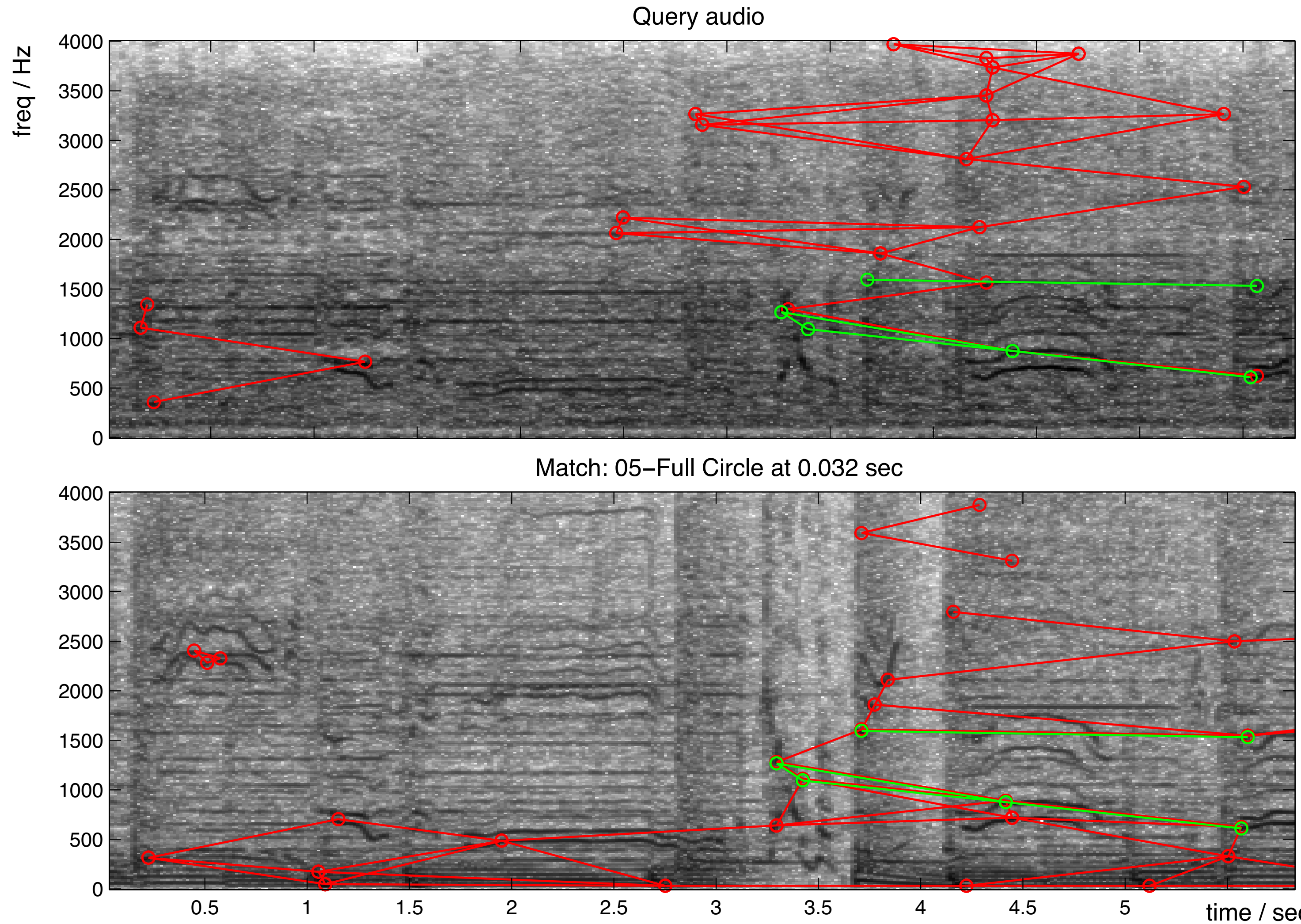
- Landmark Density controls match chance, required query size
  - control by density of peaks

- Pick peaks based on local decaying surface
  - width, rate of decay → peak density



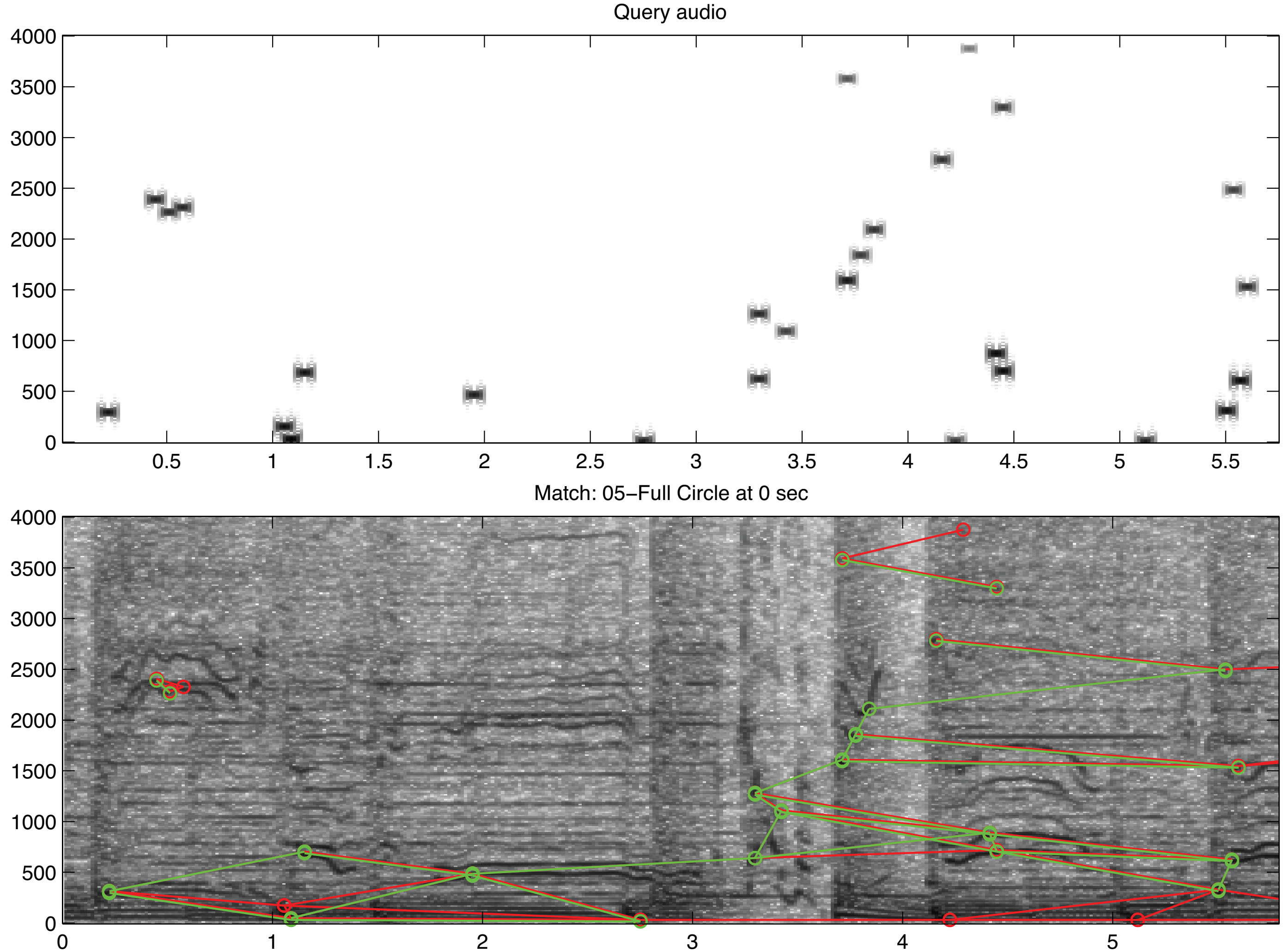
Shazam Matching

- Nearby pairs of peaks → hashes
- Each query hash → list of matching ref items
- Any subset of hashes is sufficient
- Check temporal sequence for ref items with multiple hits



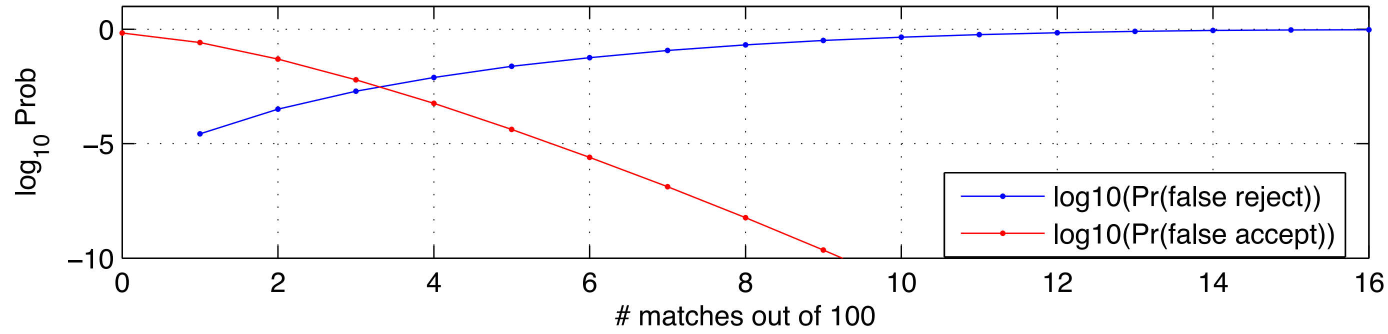
Shazam Decoy

- Only a tiny part of the signal needs to be preserved



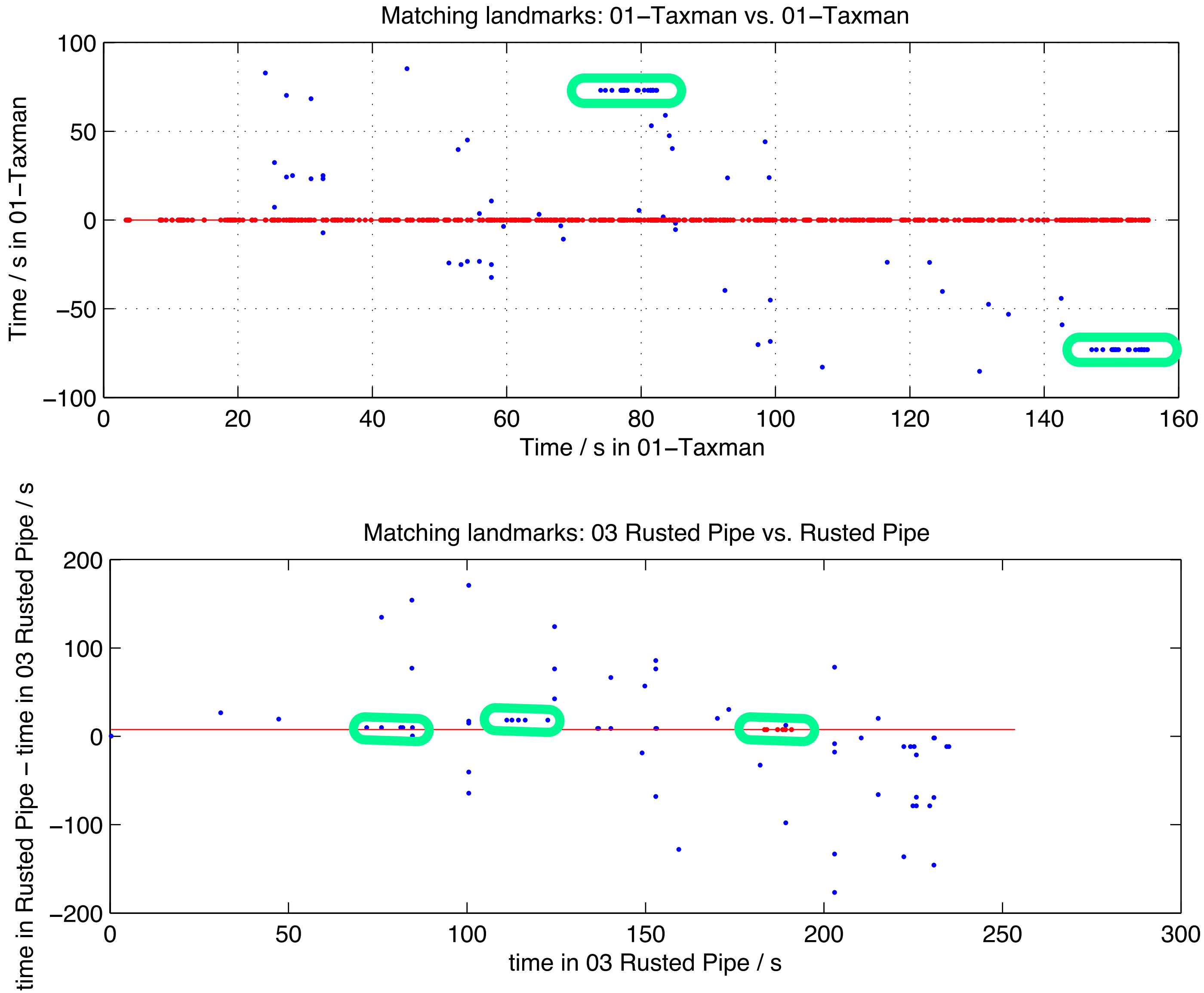
Error Analysis

- **~20 hashes / sec → ~4000 hashes / track**
  - 10k tracks → 40M hash entries (160MB)
  - 20 bit space → 10^6 distinct hashes
  - b = Pr(hash | wrong audio) = $(1 - 10^{-6})^{4000}$ = 0.4%
  - a = Pr(hash | right audio) = 0.1 ?
- **5 sec query → K =100 hashes (Binomial)**
  - Pr(N chance matches) = ${}^{K}C_{N}\, b^{N}(1-b)^{K-N}$
    N = 6 → Pr ~ $10^9 \cdot 3\times10^{-15} \cdot 0.7 = 2.5\times10^{-6}$
  - Pr(true matches < N) = $\Sigma\, {}^{K}C_{N}\, a^{N}(1-a)^{K-N}$ ~ 6%



Diagnostic Uses

- Recurring Landmarks show reused audio



Summary

- Fingerprinting
  Match same recording despite channel

- Frame-based
  Local features with fuzzy match

- Landmark based
  Highly tolerant of added noise

References

- J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system with an efficient search strategy," *J. New Music Research* 32(2), 211-221, 2003.
- A. Wang, "An Industrial-Strength Audio Search Algorithm," *Proc. Int. Symp. on Music Info. Retrieval*, 7-13, 2003.
- A. Wang, "The Shazam music recognition service," *Comm. ACM* 49(8), 44-48, 2006.