An Audio Fingerprinting System for Live Version Identification using Image Processing Techniques

(Dr.?) Zafar Rafii

Northwestern University

EECS department

Acknowledgments

- Work done during internship at Gracenote, a company doing audio and video identification, with co-authors Bob Coover and Jinyu Han.

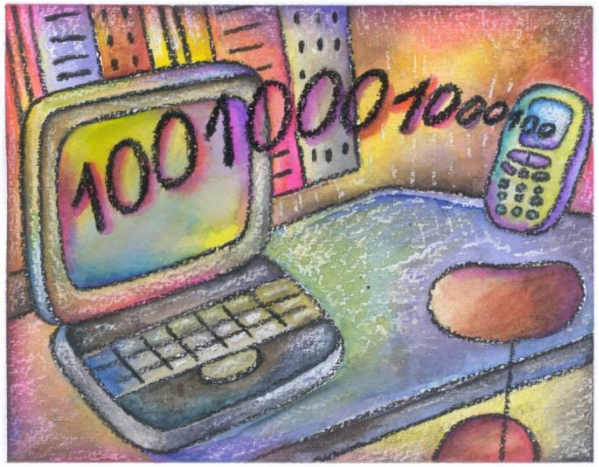
- Work presented at the 39th IEEE International Conference on Acoustics, Speech and Signal Processing, Florence, Italy, May 4-9, 2014.

Context

- You are at a concert
  - You know the artist who is playing
  - You want to know about the song being played
  - You have a smart device (e.g., an iPhone)

Idea

- You can use a music identification system
  - You record an excerpt using your smart device
  - It is processed and compared against a database
  - You get information about the song (e.g., title)



Principle

- Audio fingerprinting systems
  - Transform the audio into a compact fingerprint
  - Compare the query against a database for a match
  - Typically index fingerprints to speed up matching

Limitations

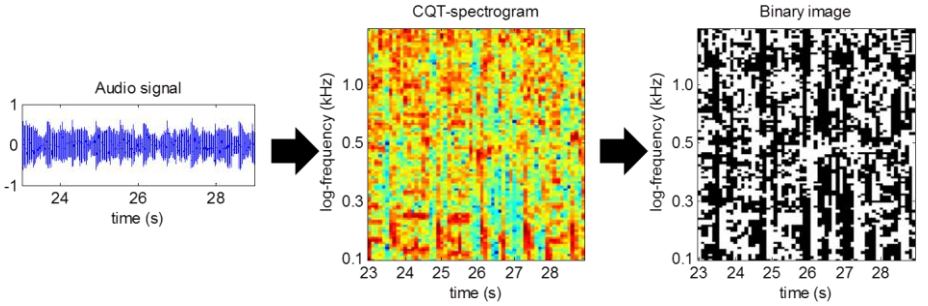
- Does not work with cover versions (e.g., live)
  - Variations in tempo (e.g., faster renditions)
  - Variations in key (e.g., higher pitch)
  - Variations in instrumentations, etc.

Solution

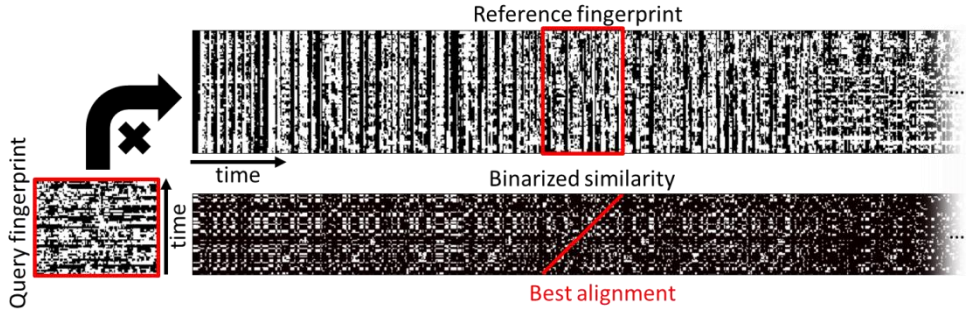
- A novel system that can handle
  - Short excerpt quickly (i.e., less than 10 seconds)
  - Audio degradations (e.g., noise, encoding, etc.)
  - Audio variations (e.g., different tempo, key, etc.)

Approach

- Fingerprinting stage
  - Constant Q Transform
  - Adaptive thresholding

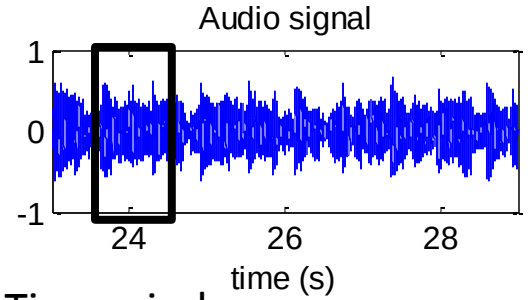


- Matching stage
  - Hamming similarity
  - Hough Transform

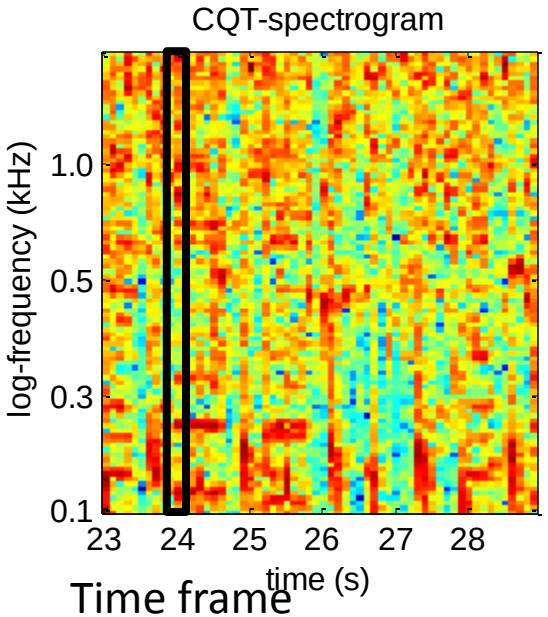


Fingerprinting

- Constant Q Transform (CQT)
  - We first transform the audio signal into a time-frequency representation using the CQT



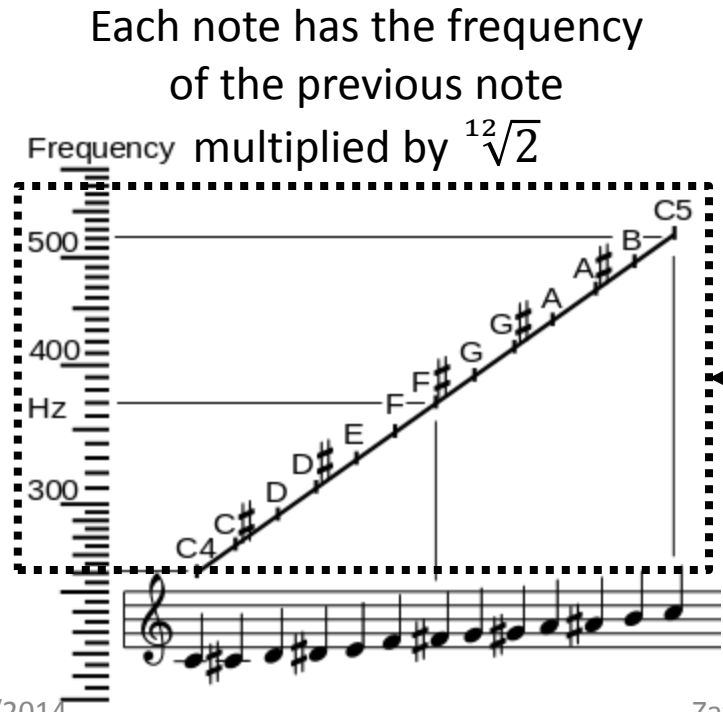
Time window centered at 24 seconds

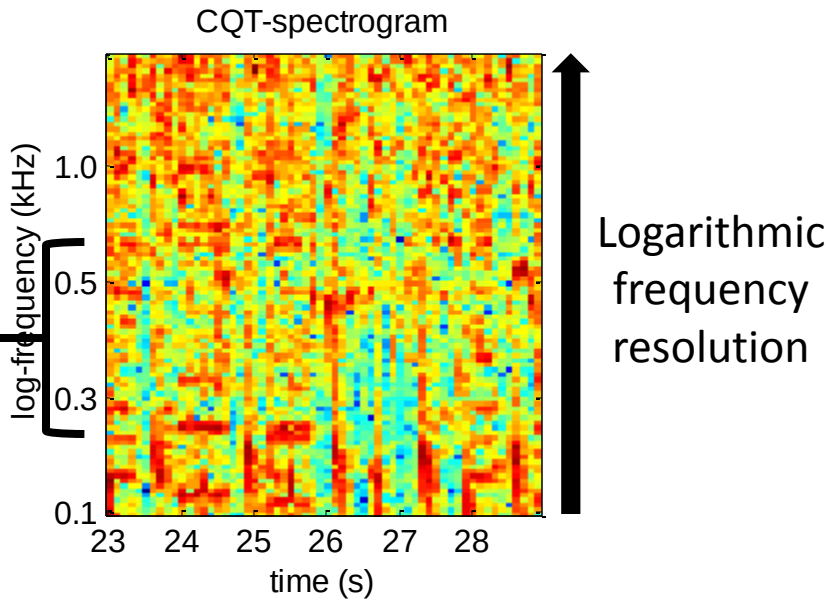


Time frame centered at 24 seconds

Fingerprinting

- Constant Q Transform (CQT)
  - The CQT has a log-frequency resolution, matching the notes of the chromatic scale (i.e., C, C#, etc.)



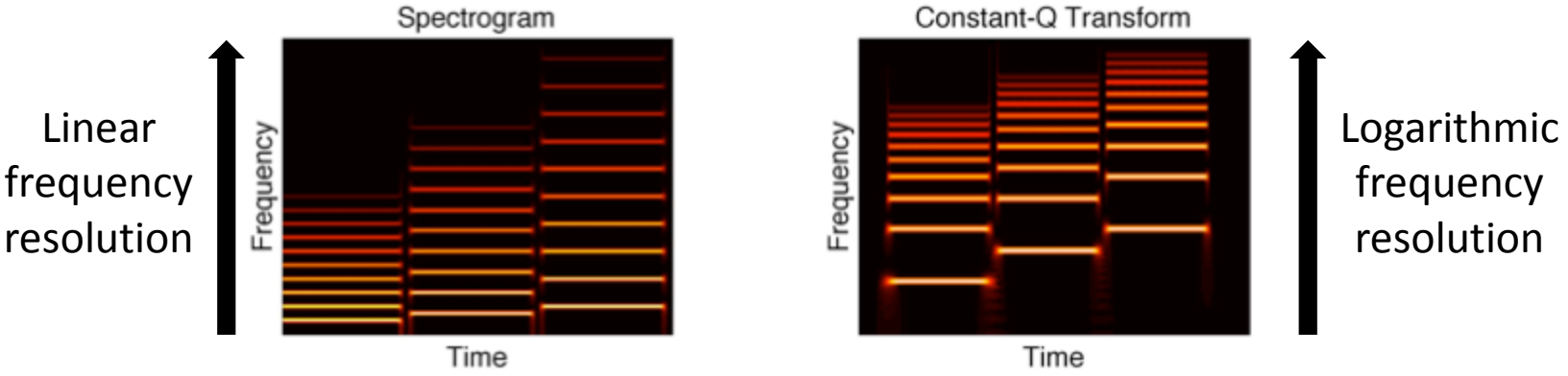


http://en.wikipedia.org/wiki/Note

Fingerprinting

- Constant Q Transform (CQT)
  - Unlike the FT, the CQT is more compact and better adapted to music (vertical shift = pitch shift)

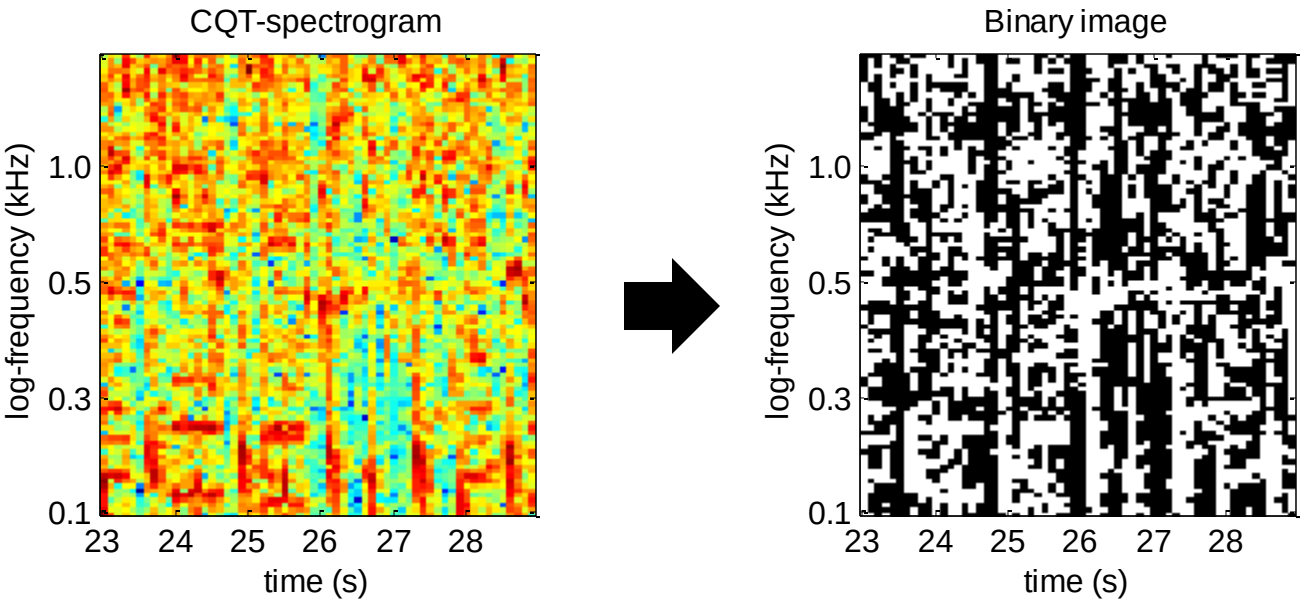
Three notes played at different pitches in
the FT-spectrogram (left) and the CQT-spectrogram (right)



https://ccrma.stanford.edu/~gautham/

Fingerprinting

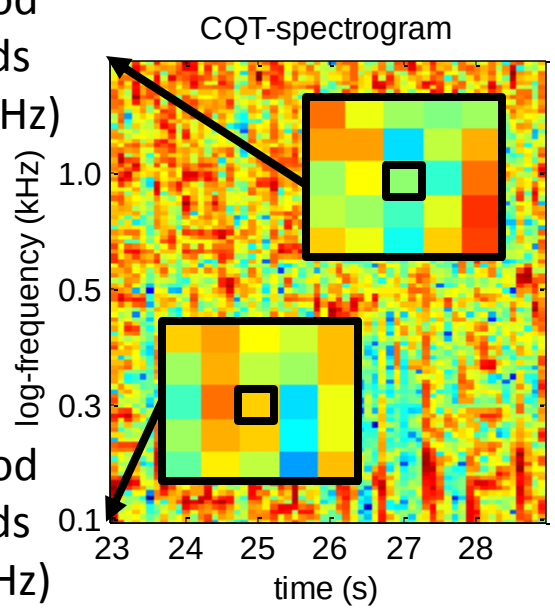
- Adaptive thresholding
  - We transform the CQT-spectrogram into a binary image using an adaptive thresholding method



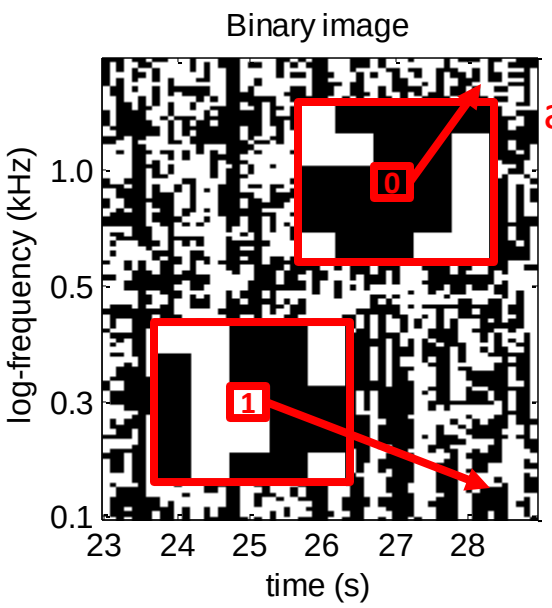
Fingerprinting

- Adaptive thresholding
  - For each bin in the spectrogram, we assign 1 if the bin is higher than the median of the neighborhood

Neighborhood at 27 seconds and C6 (1.0 kHz)



Neighborhood at 25 seconds and C4 (262 Hz)

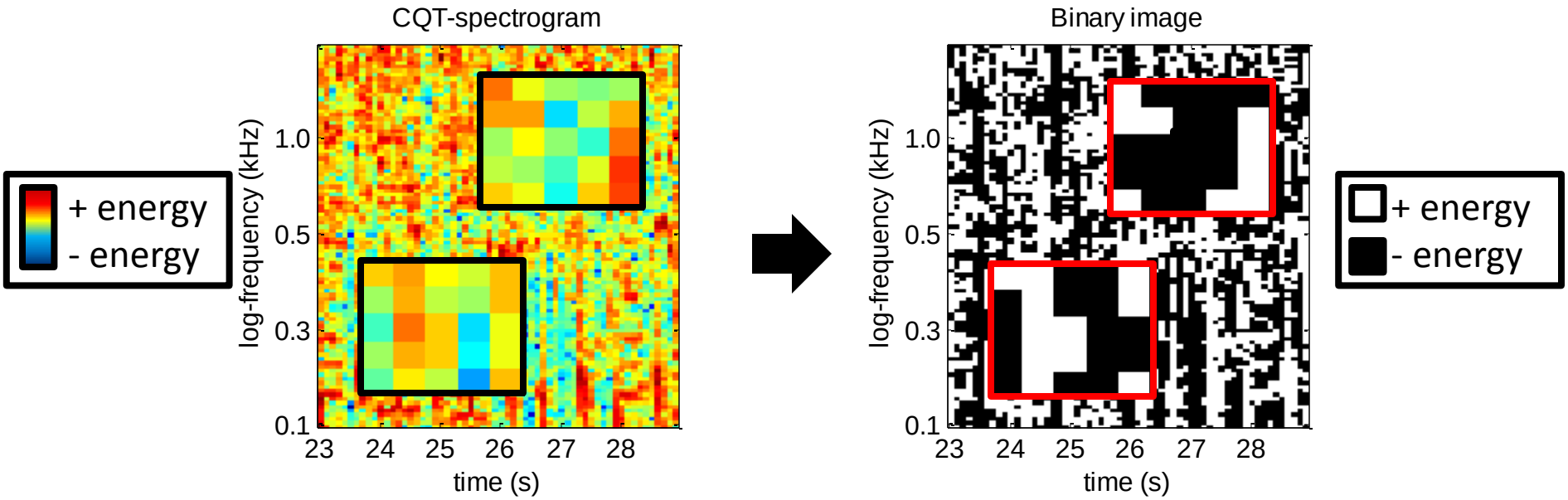


Bin at 27 seconds and C6 (1.0 kHz) < than median

Bin at 25 seconds and C4 (262 Hz) > than median

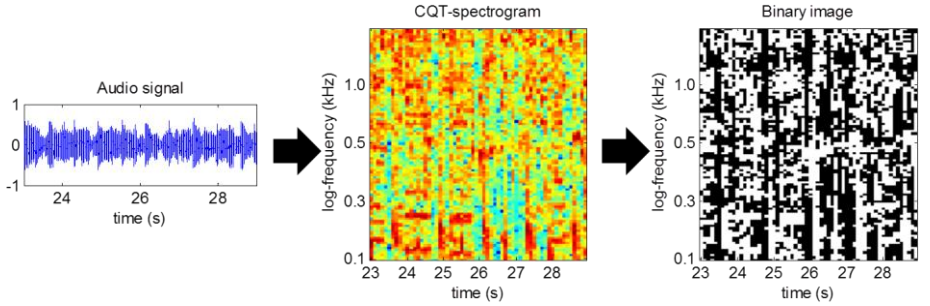
Fingerprinting

- Adaptive thresholding
  - We get a fingerprint that reduces the spectrogram into 2 components, of locally low and high energy

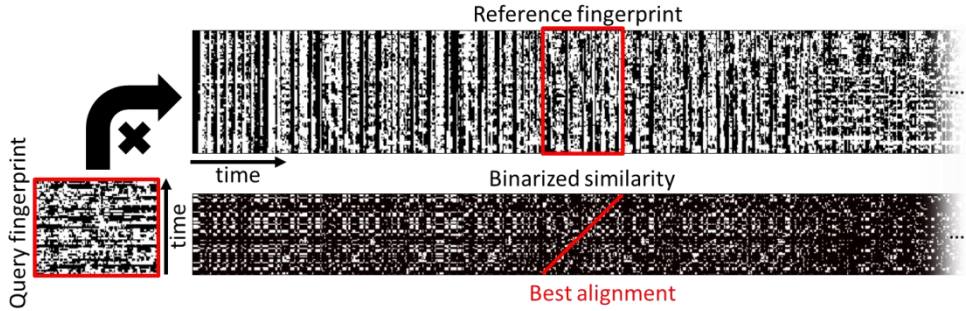


Approach

- Fingerprinting stage
  - Constant Q Transform
  - Adaptive thresholding

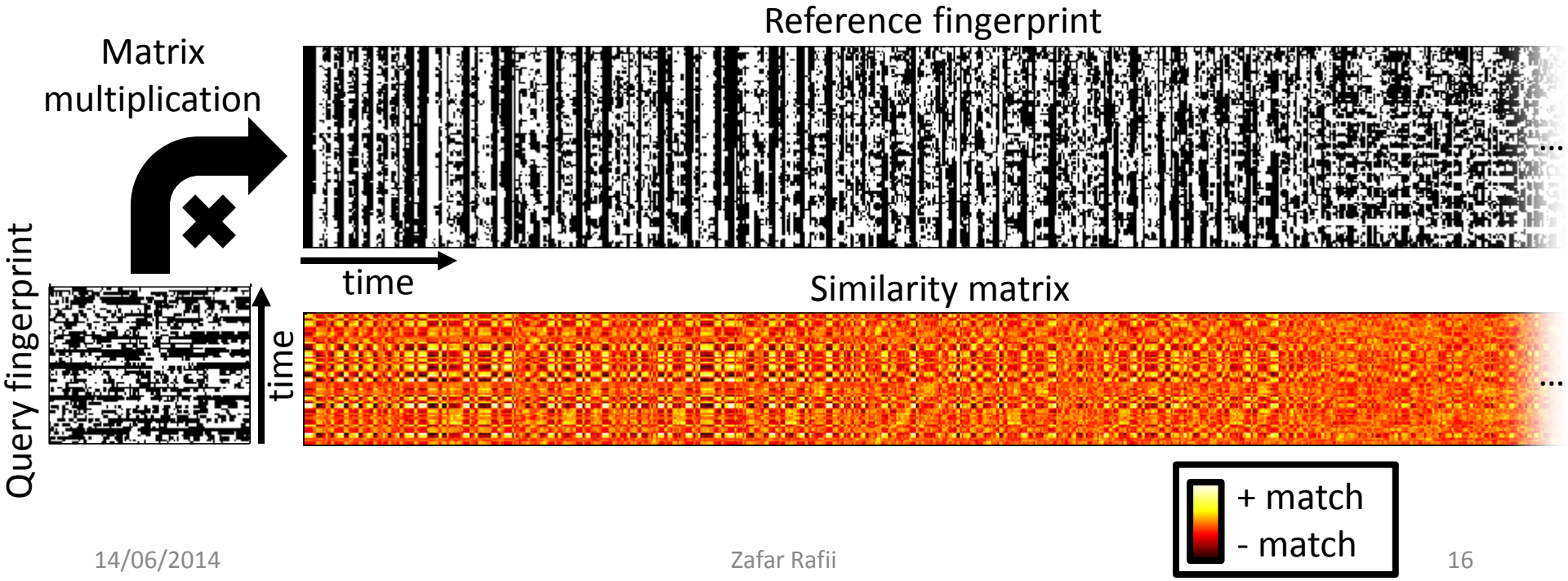


- Matching stage
  - Hamming similarity
  - Hough Transform



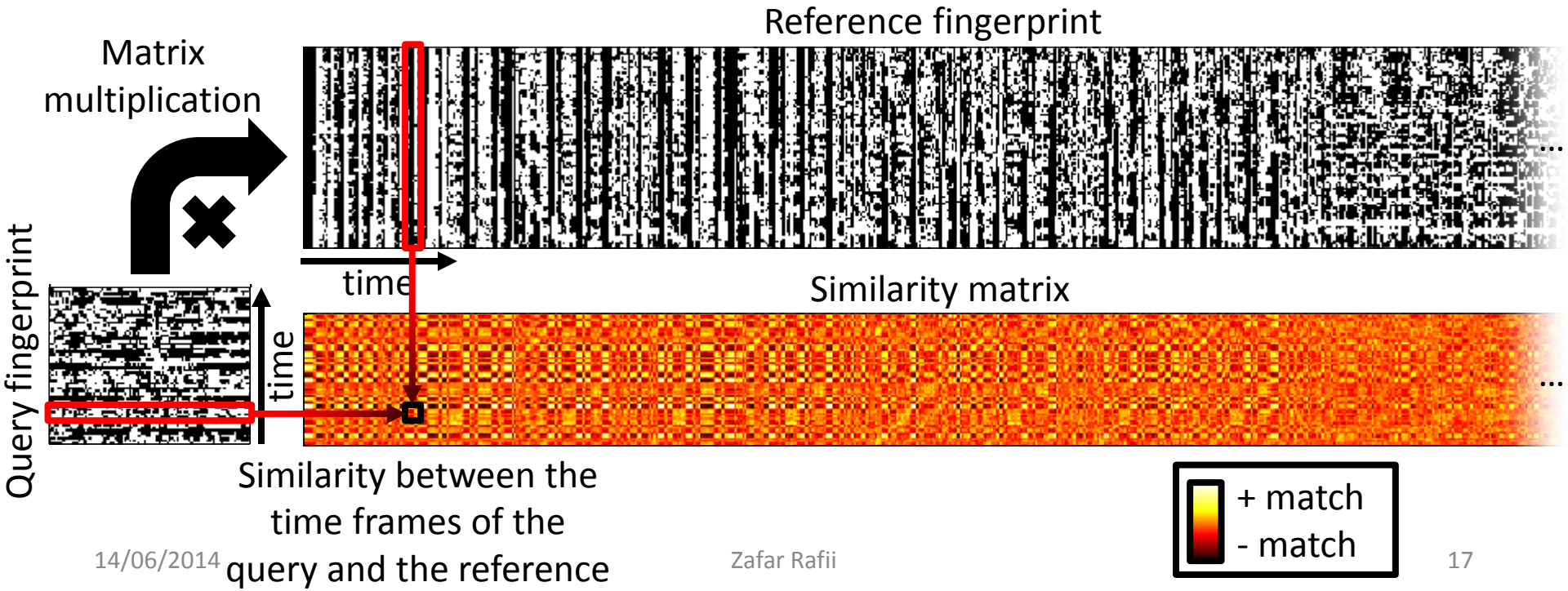
Matching

- Hamming similarity
  - We then compute a similarity matrix between the fingerprints of a query and each of the references



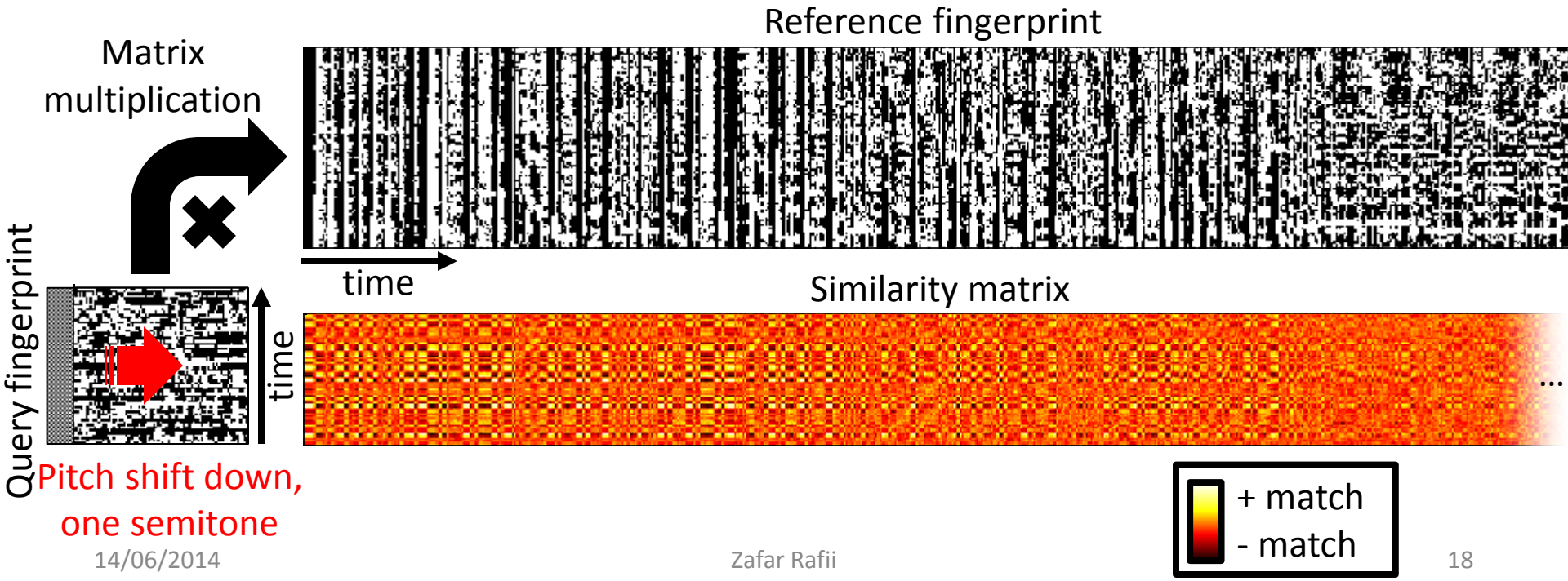
Matching

- Hamming similarity
  - We use the Hamming similarity between all pairs of time frames (= percentage of bins that match)



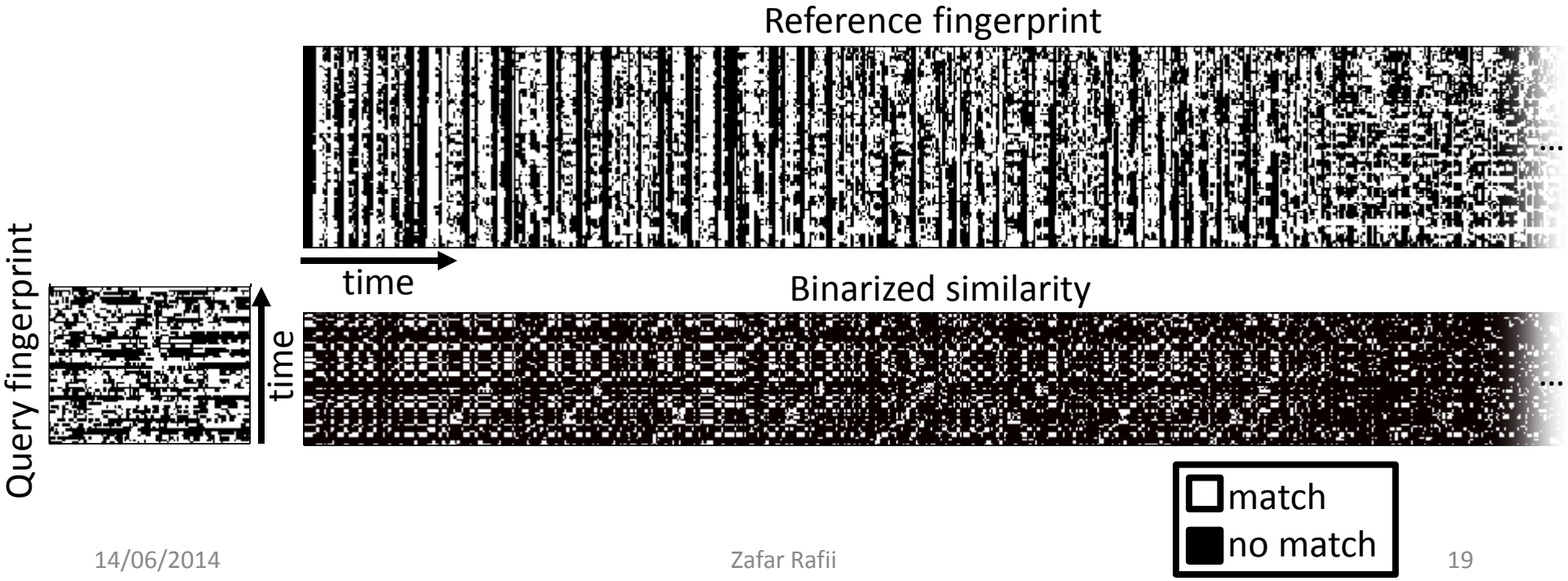
Matching

- Hamming similarity
  - We compute the similarity matrix for different pitch shifts between the query and the references



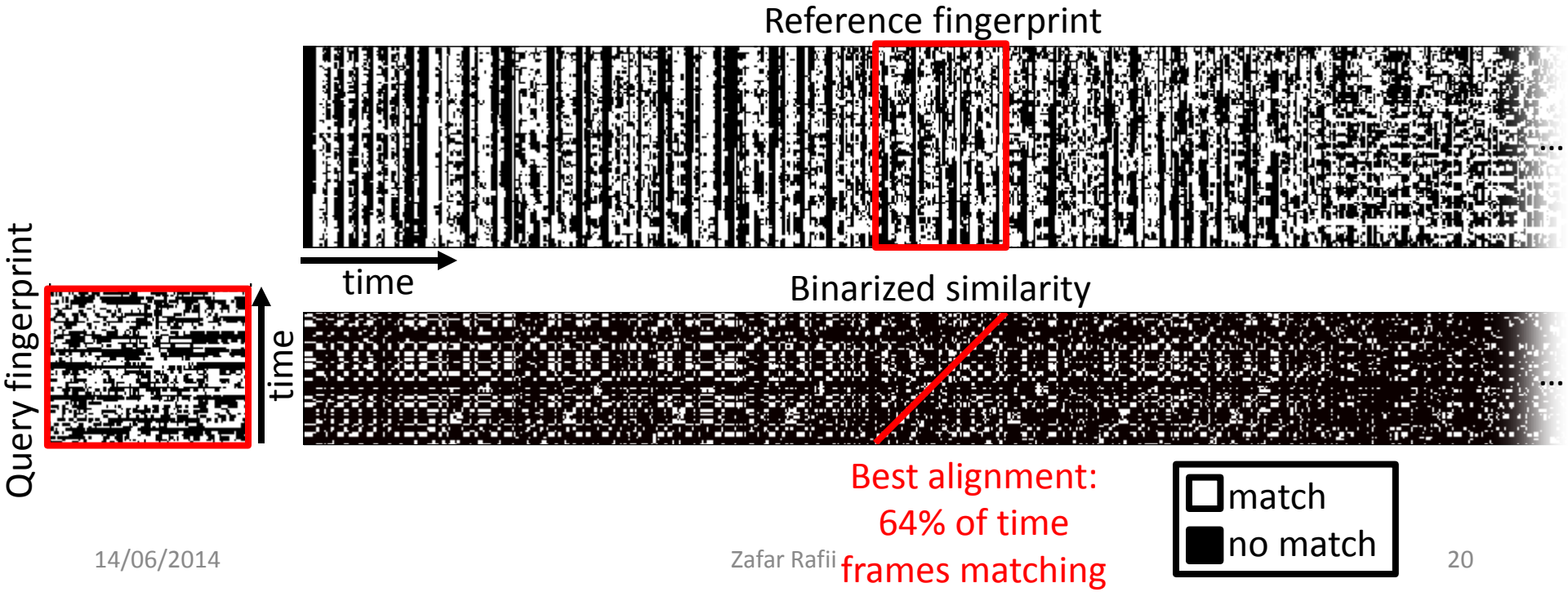
Matching

- Hough Transform (HT)
  - We binarize the similarity matrix via a threshold to have pairs of time frames that match (1) or not (0)



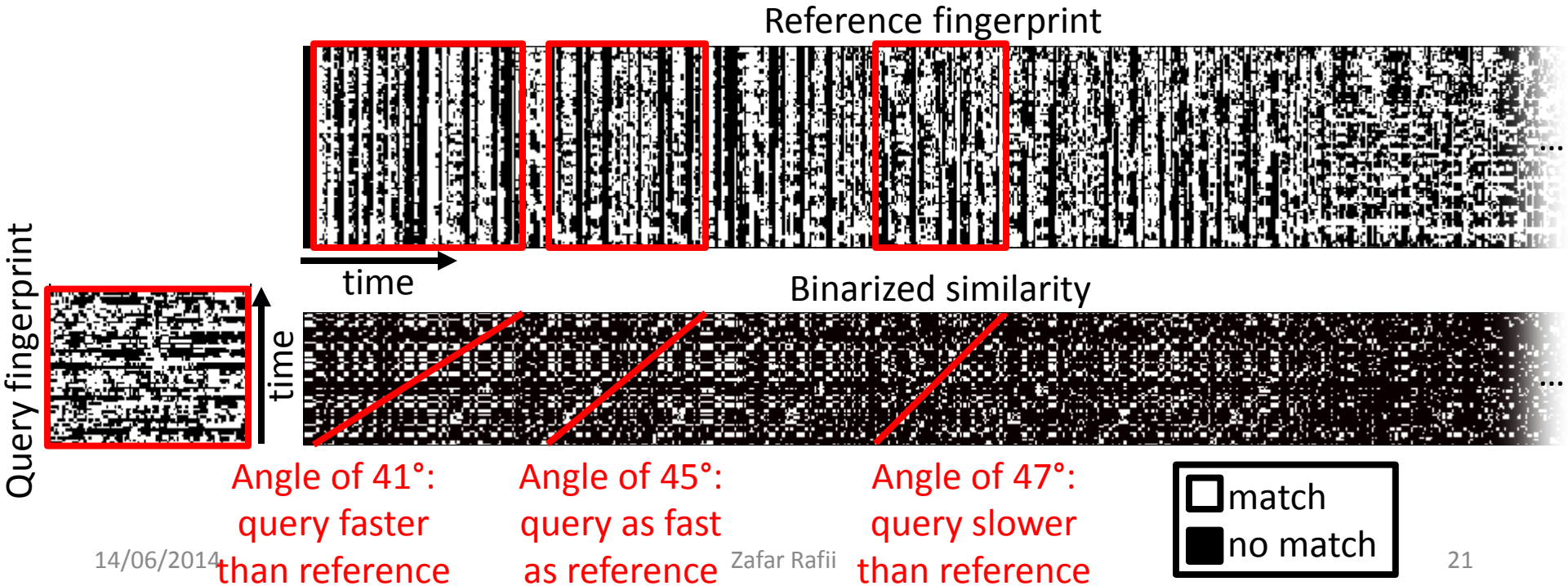
Method

- Hough Transform (HT)
  - We use the HT to identify the best alignment between the query and the reference fingerprints



Matching

- Hough Transform (HT)
  - The HT helps to take into account potential tempo deviations, by trying different angles for a line



Evaluation

- References
  - 10 different artists of varied genres
  - 389 full tracks from studio albums
  - Durations from 01’04’’ to 11’06’’
- Queries
  - 87 full tracks from live albums (experiment 1)
  - 87 audio tracks from smart devices (experiment 2)
  - 10 queries per tracks, 6 and 9 second length

Data set

artist	genre	#references	#queries
AC/DC	hard rock	36	60
Arcade Fire	indie rock	33	100
Bonobo	electronic	42	100
Eagles	rock	32	90
Foreigner	rock	29	100
Jefferson Airplane	psychedelic rock	65	40
Led Zeppelin	rock	40	80
Phoenix	alternative rock	38	100
Portishead	electronic	33	100
Suprême NTM	French hip hop	41	100
all	-	389	870

Live albums (9 seconds)

Top-k matches	k=1	k=2	k=3	k=4	k=5
AC/DC	0.92	0.95	0.97	0.97	0.97
Arcade Fire	0.84	0.92	0.94	0.96	0.97
Bonobo	0.83	0.89	0.92	0.92	0.96
Eagles	**0.93**	**0.97**	**0.98**	**0.99**	**0.99**
Foreigner	0.88	0.93	0.93	0.95	0.97
Jefferson Airplane	0.60	0.68	0.78	0.78	0.80
Led Zeppelin	0.74	0.81	0.84	0.85	0.90
Phoenix	0.88	0.92	0.93	0.97	0.98
Portishead	0.92	0.93	0.93	0.93	0.93
Suprême NTM	0.87	0.95	0.96	0.97	0.97
all	0.86	0.91	0.92	0.94	0.95

Smart devices (9 seconds)

Top-k matches	k=1	k=2	k=3	k=4	k=5
AC/DC	0.70	0.83	0.85	0.87	**0.93**
Arcade Fire	0.79	**0.86**	**0.89**	**0.91**	**0.93**
Bonobo	0.60	0.75	0.83	0.89	**0.93**
Eagles	0.70	0.77	0.88	**0.91**	0.91
Foreigner	0.68	0.83	0.86	0.86	0.88
Jefferson Airplane	0.40	0.53	0.55	0.60	0.63
Led Zeppelin	0.28	0.39	0.48	0.53	0.54
Phoenix	0.67	0.76	0.82	0.86	0.87
Portishead	**0.80**	**0.86**	0.87	0.87	0.87
Suprême NTM	0.30	0.42	0.45	0.51	0.55
all	0.61	0.71	0.76	0.79	0.81