Technische Universität Berlin
Fachbereich Audiokommunkation und -technologie

Master of Science

Evaluation of Pitch Shift and Time Stretch Invariant Acoustic Fingerprint Systems

Master Thesis of:
Vitali Rotteker

Advisors:
Prof. Dr. Stefan Weinzierl
Dr. Holger Kirchhoff

April 2016

Zusammenfassung

Akustische Fingerprint Systeme werden zur automatischen Erkennung von Audiosignalen anhand einer perzeptuellen Zusammenfassung verwendet. Dies perzeptuelle Zusammenfassung wird Fingerprint genannt. In dieser Masterarbeit wird ein bereits vorhandenes Fingerprint System auf seine Robustheit gegenüber Pitch Shifting, Time Scaling and anderen häufig auftretenden Signalverzerrungen überprüft. Das betrachtete System basiert auf sogenannten "Interest Points" die aus einer Zeit-Frequenz-Darstellung des Audiosignals extrahiert werden. Für die Auswertung werden sowohl verschiedene Zeit-Frequenz-Darstelliungen als auch Detektoren für Interest Points verwendet um zu testen ob man dadurch die Wiederauffindungsrate von Audiosignalen in der Datenbank erhöhen kann.

Abstract

Acoustical fingerprinting systems are used to automatically identify audio using a perceptual digest of the audio signal called a fingerprint. In this thesis the robustness of a particular fingerprinting system against time scaling, pitch shifting and some common other signal degradations is tested. The system is based on extracting interest points from a time-frequency representation of the audio signal. Both the time-frequency representation and the method to detect the interest points are modified to test if these can be improved upon to gain better retrieval results when searching for query audio in a database.

Preface

This thesis would have not been possible without the help of some generous and talented people. First of all I want to thank the whole zplane.development team for taking me on as a master's degree candidate and helping me with their good advice and interesting conversations. They were also so kind to let me use their HSS-CQT-Toolbox as well as their élastiquePro SDK for this thesis.

The algorithm used in this thesis is based on a paper by Mani Malekesmaeili and Rabab K. Ward who sent me their code for clarification of some details of the implementation which was critical to get the algorithm right.

A big "thank you" goes to my friends that kept me sane and entertained throughout the last couple of months. You know who you are.

Last but not least I want thank my parents for supporting me in everything that I do.

Vitali Rotteker
Berlin
April 2016

Contents

List of Figures

List of Tables

Glossary

pitch shifting The act of changing the tone height of an audio file while maintaining the same length.. 11

time scaling The act of changing the duration of an audio file without affecting the tone height.. 11

Acronyms

BER bit error rate. 6

CD compact disc. 2–4

CDDB Compact Disc Database. 3, 4

CQT constant Q transform. 13, 19, 26–28, 51, 52, 58, 60, 62, 65

DCT discrete cosine transform. 21, 22, 24, 61, 66

DFT discrete Fourier transform. 27

DoH determinant of Hessian. 36, 39, 40

DRM digital rights management. 2, 18

DTW dynamic time warping. 14, 19

FFT fast Fourier transform. 26, 27, 67

FN false negative. 54

FNR false negative rate. 55, 56

FP false positive. 54

FPR false positive rate. 55, 56

FWMSP frame-wise most salient peaks. 40, 47, 56, 62

HMM hidden Markov model. 15

HSS harmonic sum spectrum. 27, 28

HSS-CQT harmonic sum spectrum of the constant Q transform. 27, 28, 47, 51, 52, 56, 58, 60–62, 65, 66

IQR interquartile range. 48, 51

IR impulse response. 59, 60

IRCAM Institut de Recherche et Coordination Acoustique/Musique. 14

LGB Linde-Buzo-Gray. 10

LSH locality sensitive hashing. 3, 14, 66, 67

MFCC Mel Frequency Cepstral Coefficient. 2, 15

MIR music information retrieval. 3

NMF non-negative matrix factorization. 14

PC personal computer. 7, 60, 66

RAM random access memory. 60, 61, 66

RANSAC random sample consensus. 13

SFM spectral flatness measure. 10, 11

SIFT scale-invariant feature transform. 13, 19, 33, 67

SNR signal-to-noise ratio. 21

SSD solid-state-drive. 60

STFT short-time Fourier transform. 2, 4, 5, 7, 8, 12, 14, 21, 22

SURF speeded up robust features. 19, 27, 33, 36, 39, 40, 67

TN true negative. 55

TNR true negative rate. 55, 56

TOC table of contents. 3

TP true positive. 54

TPR true positive rate. 55, 56, 62

TSR time scale ratio. 61

WCSS within-cluster sum of squares. 23

Fakultät |
Geisteswissenschaften

Prüfungsausschüsse

Studiendekan und Vorsitzender

Prof. Dr. Friedrich Steinle

Eidesstattliche Erklärung

Ist jeder an der TU Berlin verfassten schriftlichen Arbeit eigenhändig unterzeichnet beizufügen!

Hiermit erkläre ich an Eides statt gegenüber der Fakultät I der Technischen Universität Berlin, dass die vorliegende, dieser Erklärung angefügte Arbeit selbstständig und nur unter Zuhilfenahme der im Literaturverzeichnis genannten Quellen und Hilfsmittel angefertigt wurde. Alle Stellen der Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen wurden, sind kenntlich gemacht. Ich reiche die Arbeit erstmals als Prüfungsleistung ein. Ich versichere, dass diese Arbeit oder wesentliche Teile dieser Arbeit nicht bereits dem Leistungserwerb in einer anderen Lehrveranstaltung zugrunde lagen.

Titel der schriftlichen Arbeit

VerfasserIn/VerfasserInnen*

Name Vorname Matr.-Nr.

Betreuende/r DozentIn

Name Vorname

Mit meiner Unterschrift bestätige ich, dass ich über fachübliche Zitierregeln unterrichtet worden bin und verstanden habe. Die im betroffenen Fachgebiet üblichen Zitiervorschriften sind eingehalten worden.

Eine Überprüfung der Arbeit auf Plagiate mithilfe elektronischer Hilfsmittel darf vorgenommen werden.

Ort, Datum Unterschrift**

*Bei Gruppenarbeiten sind die Unterschriften aller VerfasserInnen erforderlich.
**Durch die Unterschrift bürgen Sie für den vollumfänglichen Inhalt der Endversion dieser schriftlichen Arbeit.

Introduction 1

1.1 What is acoustic fingerprinting?

Acoustic fingerprinting is a technology used to identify audio objects. An acoustic fingerprint is a perceptual digest of an audio object by which that object can be searched and identified in a database efficiently. In their landmark paper Cano et al. (2002a) describe the process like this:

> "Audio fingerprinting or Content-based audio identification (CBID) systems extract a perceptual digest of a piece of audio content, i.e. the fingerprint and store it in a database. When presented with unlabeled audio, its fingerprint is calculated and matched against those stored in the database."

Haitsma and Kalker (2002) define the main objective of multimedia fingerprinting as follows:

> "The prime objective of multimedia fingerprinting is an efficient mechanism to establish the perceptual equality of two multimedia objects: not by comparing the (typically large) objects themselves, but by comparing the associated fingerprints (small by design)."

This already hints at the advantages of using a fingerprint system to identify audio objects. The same paper also gives three actual and compelling reasons to use fingerprint systems for content identification:

1. "Reduced memory/storage requirements as fingerprints are relatively small;
2. Efficient comparison as perceptual irrelevancies have already been removed from fingerprints;
3. Efficient searching as the dataset to be searched is smaller."

As mentioned in the quote above, a fingerprint function should convert large audio objects into small fingerprints. In this a fingerprint function is similar to a hash function. A hash

function converts data of arbitrary length (usually large) to a hash value of constant length (usually small). Because of this, hash functions allow for efficient checking for strict equality of two large objects by comparing their hash values which are far smaller. In the case of acoustic fingerprinting however we are not interested in strict equality but in perceptual equality. Two audio objects that are perceptually equal can be very different in a signal representation. It is smarter to use a hash function that yields similar hashes for similar audio content and very different hashes for dissimilar content.

The perceptual digests can be computed in many ways and are mostly based on features extracted from the audio such as maxima in the short-time Fourier transform (STFT), the chroma vector, signal statistics, Mel Frequency Cepstral Coefficients (MFCCs), etc.

Because of the aforementioned similarity to cryptographic hash functions some early publications on audio fingerprinting used the terms *perceptual hashing* and *robust hashing*.

1.1.1 Applications of audio fingerprinting

Broadcast monitoring An obvious application for acoustic fingerprinting is broadcast monitoring which is vital to royalty collection, advertisement verification and program verification. Automatic playlist recognition would improve the accuracy of the data that institutions like GEMA in Germany use to distribute royalty payments to its members. In 2014 such a new system for broadcast monitoring for night clubs and bars was tested in a night club in Berlin. [1]

Digital rights management (DRM) monitoring for file sharing platforms With fingerprinting enabled on file sharing platforms it would be possible to prevent users from distributing copyrighted material by filtering it out.

Automatic music library organization Automatic retrieval of meta data is already a built-in feature in applications like Apple's iTunes. When inserting an audio compact disc (CD) into the computer, users can fetch the meta data from a server, so they don't need to input the information themselves.

1.2 General architecture of an acoustic fingerprint system

All acoustic fingerprint systems must have, at the highest level of abstraction, two methods:

1. A method to extract a fingerprint from an audio object.

2. A method to efficiently search for a given fingerprint (extracted from a query audio object) in a database of fingerprints.

Both methods are highly dependent on each other. The second method can be further divided into two seperate sub-methods:

1. A method to save a fingerprint in a way that minimizes the required storage space, while maintaining a fast searchability. Two common formats are bit-strings and vectors in which the individual entries are weights of basic audio features.

[1] `http://www.residentadvisor.net/news.aspx?id=25053`

2. An efficient search strategy. Depending on which format is used to save the fingerprints this can encompass technologies such as lookup tables (LUT), inverse indexes, bit error rates, locality sensitive hashing (LSH) etc.

1.2.1 Parameters of a fingerprinting system

In their paper Haitsma and Kalker (2002) propose the following parameters to asses fingerprinting systems by:

Robustness Can an audio clip still be identified after severe signal degradation? This parameter depends heavily on the preprocessing and the audio features that are used to compute the fingerprint.

Reliability How often is a song correctly/incorrectly identified?

Fingerprint size How much storage is needed for a fingerprint?

Granularity How many seconds of audio is needed to identify an audio clip? How many fingerprints are extracted per second of audio? The importance of this parameter depends on the use case but for most applications only a few seconds should be enough to identify a song.

Search speed and scalability How long does it take to find a fingerprint in a fingerprint database? How much computational power is needed to compute a fingerprint? Commercial fingerprinting services only need a few milliseconds to retrieve a song from a database of several million songs.

These parameters are mostly interdependent and a trade-off has to be made between them. For example: a smaller granularity results in more storage needed for the fingerprint database and this in turn can negatively influence the speed with which the database can be searched.

1.3 Literature review

The next pages are dedicated to giving the reader an overview of the history of audio fingerprinting from the beginnings to the state of the art of today.

1.3.1 Early attempts at identifying audio

Early attempts at identifying audio were mostly not based on music information retrieval (MIR) techniques and didn't analyze the perceptual content of the audio itself.

1.3.1.1 The Compact Disc Database (CDDB)

The Compact Disc Database (CDDB) was developed in 1993 by Ti Kan as an online database to accompany his music player application *xmcd*. The system identifies whole CDs by creating a hash value from the starting times and durations of all tracks extracted from the table of contents (TOC) of the CD. Thanks to this system a user could insert a music CD into his computer and if the CD already had an entry in the database the music player would display the correct meta data such as artist name, album name, title names etc. For

this to work it relied on users to enter the correct meta data if the inserted CD couldn't be found in the database.

Since the hash depends on the length of the tracks and their order - and not on the content - this approach does work only for officially released CDs and not for custom compilations of tracks from different CDs.

Also the risk of duplicate items due to multiple CDs having the same order and length of tracks is confirmed to be a problem. (Ingebrigtsen, 2007; Hansen, 2013)

Over time and with the help of the users the CDDB grew considerably and more music applications were using the database to offer their users an automatic CD identification.

The CDDB was later acquired by a company that is now known as Gracenote which commercialized the access to the database. This led to the creation of free projects such as the freedb [2] and the MusicBrainz project [3]. The latter also has its own (open source) fingerprinting algorithm called *AcoustID*.[4]

1.3.1.2 CD-Text

Introduced by Sony, CD-Text is an extension of the Red Book CD standard that allows embedding meta data into the CD. This way a capable CD-Player can display information about the CD while playing it. (EN 60908; IEC 61866:1997)

1.3.1.3 Watermarking

Analogous to watermarking on paper, digital watermarking is a technique for embedding hidden or unperceivable information into a file that can be extracted with a predefined procedure to identify a file.

The obvious disadvantage with this approach is that the file in question has to be marked in advance to be able to identify it later. Also the watermark could be damaged by signal degradations to the extent of being unidentifiable. This makes it unsuitable for perceptual audio identification.

The advantage of the approach is the possibility to trace the paths that a watermarked file has taken. This could be used to monitor copyright infringements.

An overview on watermarking techniques for audio identification can be found in the paper by Chauhan and Rizvi (2013).

1.3.2 First Commercial Fingerprinting Systems

The first commercial fingerprinting systems emerged around the year 2001. In this subsection I concentrate on three of the first fingerprinting systems that are still in use today.

1.3.2.1 Philips

Developed by Haitsma and Kalker in 2002 (Haitsma and Kalker, 2002) this fingerprinting system is based on energy differences in neighboring STFT bins. The authors claim that a sequence of 256 fingerprints (which they call a fingerprint block) corresponding to approximately 3 seconds of audio can be uniquely identified.

[2] http://www.freedb.org/

[3] https://musicbrainz.org/

[4] https://acoustid.org/

1.3.2.1.1 Algorithm

A Scheme for the fingerprint extraction is depicted in figure 1.1.

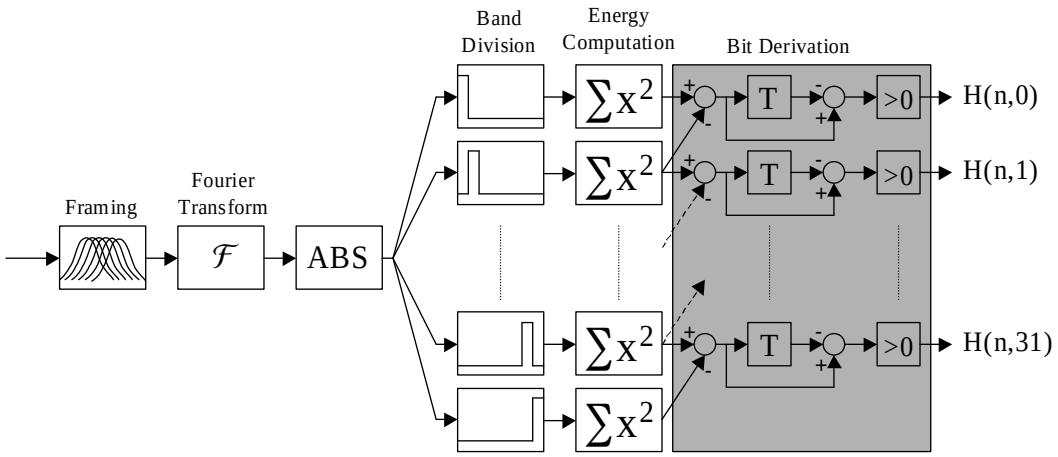


FIGURE 1.1: Overview of the fingerprint extraction scheme of the Philips system. (Haitsma and Kalker, 2002)

The algorithm for the fingerprint extraction is as follows:

1. An STFT is computed with a frame length of 0.37 s and an overlapping factor of $\frac{31}{32}$, resulting in a hop size of 11.6 ms. Only the absolute value of the STFT is used.

2. The frequency bins of the STFT are summed into 33 logarithmically spaced bands from 300 Hz to 2000 Hz.

3. from the resulting energy spectrogram E the fingerprint F is computed by comparing the energies of neighboring bins using the function

$$F(n,m) = \begin{cases} 1 \text{ if } E(n,m) - E(n,m+1) - (E(n-1,m) - E(n-1,m+1)) > 0 \\ 0 \text{ if } E(n,m) - E(n,m+1) - (E(n-1,m) - E(n-1,m+1)) \leq 0 \end{cases} \tag{1.1}$$

where n is the frame index, m the index of the frequency band and $F(n,m)$ is the resulting fingerprint value. This results in subfingerprints that are bit-strings of length 32 which are then stored sequentially in a database. These energy differences are experimentally found to be robust against a variety of signal degradations.

The retrieval stage is realized by using a lookup table that contains for every possible hash value (2^{32} possible values) the songs and positions it occurs in. Figure 1.2 shows the scheme used for the retrieval. Using the underlying assumption that at least one of the 256 fingerprints in a fingerprint block has no bit errors the following steps describe the retrieval process:

1. For every subfingerprint: use the lookup table to find the songs and positions this particular hash value occurs in.

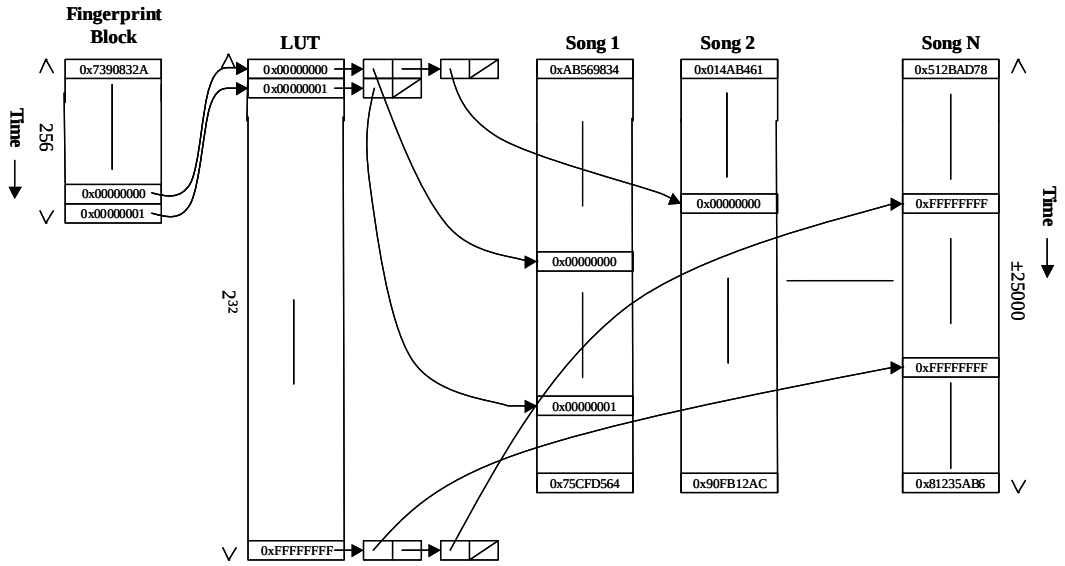


Figure 1.2: Overview of the fingerprint retrieval of the Philips System. (Haitsma and Kalker, 2002)

2. For every song and position: compute the bit error rate (BER) between the properly (time-)shifted query fingerprint block and the subfingerprints of the database.

3. If the BER is below a threshold (e.g. of 0.35) the query is accepted as a match and the song and the position are then returned as the result.
   If the BER is larger than the threshold for all possible positions and songs the song is considered to have no entry in the database.

However this works only for mild degradations so that there is at least one subfingerprint in the query that has a perfect match. For the case of heavier degradations the authors propose a method in which "unreliable" bits are flipped and also searched for in the database as described above. Reliability in this case is defined by the distance of the energy differences (used in the computation of the fingerprint values in equation 1.1) to the threshold. The higher the distance to the threshold, the higher the reliability since an energy difference close to the threshold has a high chance of being accidentally flipped. The authors call this feature of the algorithm *soft decoding*.

1.3.2.1.2 Advantages & Limitations

Advantages:

Robustness Robust against degradations such as strong mp3-encoding, equalization, compression, noise, band pass filtering.

Reliability A false positive rate of $3.6 \cdot 10^{-20}$ is achieved and controllable by the threshold for the BER

Fingerprint size Small fingerprint size of 2,7 kbit/s.

Granularity A 3 s query is enough to identify an audio object.

Search Speed and Scalability Search speeds are fast. "dozens of queries per second on a modern personal computer (PC)" (Haitsma and Kalker, 2002).

Limitations:

Robustness Not very robust against GSM encoding and linear tempo changes above $\pm 2.5\,\%$.

1.3.2.2 Shazam

Possibly one of the best known fingerprinting systems is the Shazam system. To an end user it is available as an application for smartphones and since recently for Mac OS X. The Shazam system was developed with a mobile use case in mind. The company itself summarizes the intended usage scenario in Wang et al. (2003) like this:

> "A user hears music playing in the environment. She calls up our service using her mobile phone and samples up to 15 seconds of audio. An identification is performed on the sample at our server, then the track title and artist are sent back to the user via SMS text messaging."[5]

This implies that the fingerprint has to be robust against several things such as ambient noise, reverb, mediocre microphone quality as well as GSM encoding and network dropouts.

1.3.2.2.1 Algorithm

The Shazam algorithm is described by Wang et al. (2003) and Wang and Smith (2002). To generate a fingerprint it relies on local maxima in the STFT magnitude spectrogram. A local maximum in this case is a peak in the energy spectrogram that has a higher amplitude than a region centered around the peak. The positions in the time-frequency domain of the local maxima are used to create hash values which form the fingerprint. The process of computing a fingerprint is depicted in figure 1.3.

1. Compute the STFT.
2. Find the local maxima.
3. For each local maximum: consider the maximum an anchor point and assign it a *target area*.
   For each local maximum inside the target area a hash is formed from the frequencies of the anchor point and the local maximum inside the target area and the time difference between them.

Each hash is then saved as a 32 bit unsigned integer together with the time offset from the beginning of the file of the anchor point $t1$ which is not part of the hash. A database entry consists of the 32 bit hash, the associated time offset and a track ID resulting in a 64 bit struct. The structs are then sorted by hash token value.

[5] Since smartphones are omnipresent these days (compared to the year of publication of this paper 2003) the scenario changed a little bit and it can be assumed that the analysis is done by the smartphone app and the results are then send to the server for lookup.

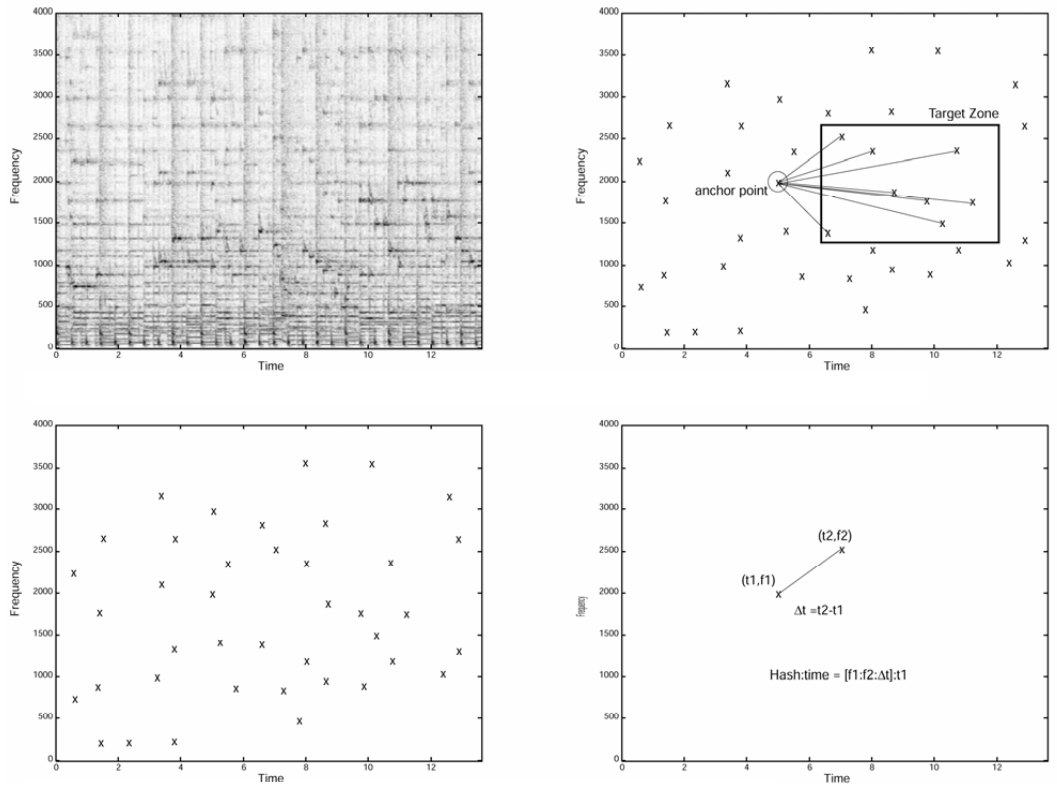


Figure 1.3: Overview of the fingerprint extraction scheme of the Shazam system. (Wang et al., 2003)

The paper and the patent are short on detailed information such as which parameters are chosen for the STFT (i.e. window length and type, hop size, sampling frequency) and how the local maxima are actually located in the spectrogram. For the latter the patent gives some examples of how this can be achieved.

The retrieval stage is realized as follows:

1. The fingerprint for the query audio is computed.

2. For every query hash a match is searched in the database.

3. The associated time offset pairs of a query hash and its match in the database are saved into bins according to the track ID associated with the matching database hash.

4. Inside each track ID bin all time offset pairs are used to compute time offset differences using the equation:

$$\delta t_k = t'_k - t_k, \tag{1.2}$$

where t'_k is the time coordinate of the feature in the matching (clean) database sound file and t_k is the time coordinate of the corresponding feature in the query.

5. The δt_k values are combined into a histogram plot. If the track ID is a match there will be a pronounced peak otherwise the histogram will be more or less flat. The height of the peak (the number of matching hashes with the same difference of time offsets) is the score of the track ID. The false positive rate can be controlled by setting a minimum threshold for the score of a hit.

1.3.2.2.2 Advantages & Limitations

The advantages of the Shazam system are:

Robustness Only 1–2 % of the hashes have to survive a signal degradation to be able to identify an audio sequence. Even drop outs / cropping is no problem.

Reliability The false positive rate of this algorithm is controlled by a threshold on the score that is needed for the algorithm to return a match.

Fingerprint size Nothing is said explicitly about the size of the fingerprint. Since in the patent a number of 5–10 landmarks per second is said to be a good number of landmarks and one fingerprint has the size if 64 bits the fingerprint size is roughly 320 bits to 640 bits per second.

Granularity It is possible to find short snippets (around 10 s) of query audio in a database of complete songs because the fingerprints consist of many local subfingerprints and not one global fingerprint for the whole song.

Search speed and scalability According to the authors a query that is part of a database of about 20.000 items can be processed in 5–500 milliseconds. In 2013 the database size was 28 million songs (Smith, 2013; SN, 2013; Phillips, 2013) and this number can be expected to have risen in the meantime. Still look-up times for queries seem to have kept very low.

Limitations:

Robustness The main disadvantage of the Shazam system is that it is not designed to cope with shifting and time stretching.

1.3.2.3 AudioID

This system is presented by the authors as more of a framework than an actual fingerprinting system and is covered in a number of a papers by Cremer et al. (2001); Hellmuth et al. (2001); Allamanche et al. (2001); Herre et al. (2001); Kastner et al. (2002); Hellmuth et al. (2003). It relies heavily on the MPEG[6]-7 standard (Lindsay and Herre, 2001) and aims to use standardized audio features. In several papers, by mostly the same authors, many audio features such as loudness, spectral crest factor and spectral flatness measure were tested for robustness against common signal degradations.

1.3.2.3.1 Algorithm

The underlying framework is summarized in figure 1.4. The algorithm for fingerprint extraction is as follows:

[6]Moving Pictures Experts Group

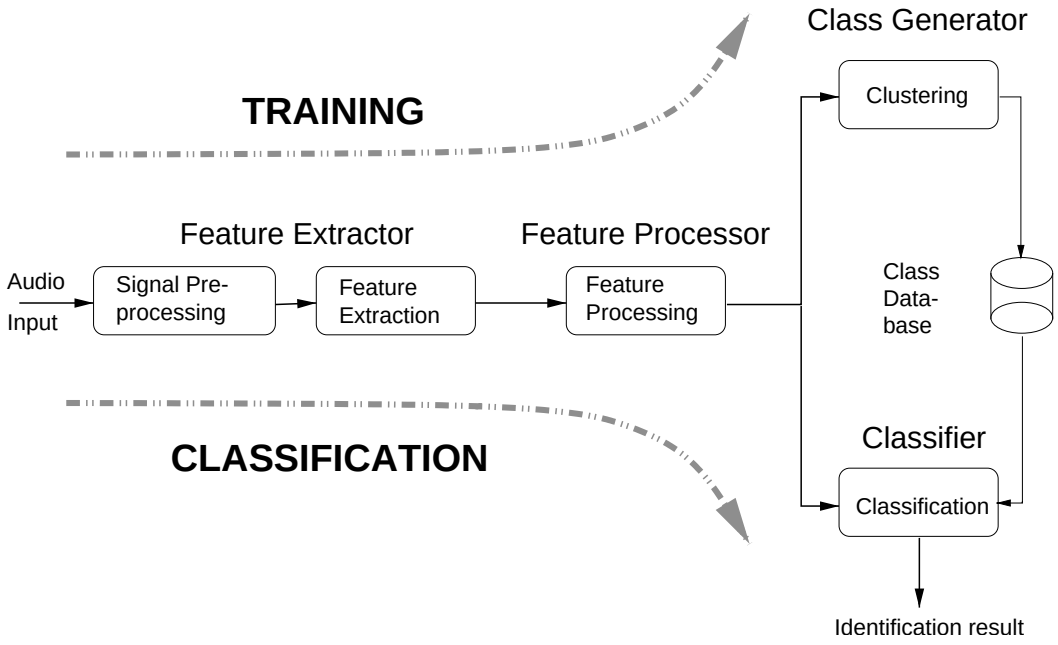


Figure 1.4: Overview of the Fraunhofer Framework. (Cremer et al., 2001)

1. The audio signal is first subjected to a preprocessor that outputs a mono signal with a sampling frequency of 44.1 kHz.

2. The following feature extraction computes the features on a frame-to-frame basis from a time frequency representation of the signal.

3. The feature processing stage tries to increase recognition performance and decrease fingerprint size by using transformation techniques and statistical data summarization. Also the fingerprints are normalized at this point.

4. The last step in the computation of a fingerprint is a clustering stage that generates a vector quantization code book that is considered the fingerprint of the audio object. This is done via a Linde-Buzo-Gray (LGB) algorithm.

The retrieval stage is organized as follows:

1. First compute some feature vectors from the query audio.

2. For each feature vector try to approximate it with each codebook in the database

3. Accumulate the approximation error for every codebook and choose the database item with the lowest approximation error that still falls under a maximum threshold.

In the corresponding patent (Allamanche et al., 2002) the authors are more specific as to what feature they actually use: the spectral flatness measure (SFM). (Peeters, 2004; Allamanche et al., 2002) This is consistent with the results of the papers Cremer et al.

(2001); Hellmuth et al. (2001); Allamanche et al. (2001); Herre et al. (2001); Kastner et al. (2002); Hellmuth et al. (2003).

This system is the basis for companies like the Mufin GmbH which is a spin-off from the Fraunhofer IDMT[7].

1.3.2.3.2 Advantages & Limitations

Advantages:

Robustness The system is very robust to the most common signal degradations. Also the system is robust to cropping since the feature vectors are not sorted in time.

Reliability False positives can be avoided by setting a threshold for the maximum allowed distance to any codebook.

Fingerprint size The fingerprint size is very small, since for every database entry only one codebook has to be saved.

Granularity Sample sizes as small as 15 s can be correctly identified in 99.6 % of the time.

Search speed and scalability The algorithm was tested with databases up to a size of 15000 items and runs at 80 times realtime speed.

Limitations:

Granularity The resulting fingerprint – the codebook – is global because there is only one codebook for every audio object. However, it is closely tied to the statistics of feature vectors that cluster around certain centroids. Given a long enough sample of the audio and the assumption that the feature vectors of a song are more similar within a song than between different songs one can retrieve the correct song from the database. In the corresponding publications samples sizes of 15 s are enough to achieve a correct identification rate of 99.6 %. This result will depend on the musical content used in the database. One cannot assume the SFM to be mostly the same throughout a song.

Robustness No results on time scaling and pitch shifting although resampling of $\pm$ 5 % is mentioned in some of the papers.

1.3.3 Subsequent literature

In the subsequent literature new systems are developed that in some cases are based on the above three systems and try to improve the performance with respect to signal degradations such as pitch shifting and time scaling.

Also in later publications some systems try to use existing image processing and computer vision technologies for audio search and fingerprinting.

[7] see `http://www.fraunhoferventure.de/de/spin-offs/institute/IDMT.html`

1.3.3.1 Modification & expansion of the Philips, AudioID & Shazam systems

The three systems discussed above are the foundation of many other papers that try to improve the performance for certain signal degradations or even extend the scope of the whole algorithm to audio similarity search or audio matching.

Many papers can be categorized as extensions or modifications of the Shazam system such as (Van Balen, 2011; Chandrasekhar et al., 2011; Grosche and Müller, 2012; Fenet, 2013; Six and Leman, 2014; Sonnleitner and Widmer, 2014; Kim and Kim, 2014).

One can just as easily see that many other papers base their algorithm on the Philips system (Haitsma and Kalker, 2003; Jang et al., 2009; Coover and Han, 2014; Yang et al., 2014; Ke et al., 2005).

1.3.3.2 Image processing techniques for acoustic fingerprint systems

1.3.3.2.1 Computer vision for music identification

In 2005 Ke, Hoiem and Sukthankar published a paper titled "Computer vision for music identification", which employed techniques that were predominantly used in the field of image processing to the task of audio identification. The approach of the authors is to treat each piece of audio as 2-D image (through it's STFT) and transform the task of music identification into a corrupted sub-image retrieval problem. This paper layed the foundation for many other algorithms that followed it.

The algorithm is loosely based on the Philips system developed by Haitsma and Kalker (2002). This is evident when looking at the parameters of the STFT or the number of filters chosen for the fingerprints. The biggest difference is that the filters in (Ke et al., 2005) are chosen by a machine learning approach to maximize classification performance.

To compute a fingerprint the authors perform the following steps:

1. Convert the audio into an image by computing the STFT with a window length of 0.372 s and a hop size of 11.6 ms

2. Divide the power between 300 Hz and 2000 Hz into 33 logarithmically spaced bands.

3. Apply the 32 previously learned Viola-Jones filters / Haar-like features to every time frame to get a 32 bit string fingerprint.

The 32 Viola-Jones filters were found by using a variant of the Adaboost algorithm on a set of around 25000 potential filters to extract only the 32 strongest filters for classification. Viola-Jones filters or Haar-like features were proposed by Viola and Jones (2001) and are 2D filters that are used in object and face detection and work by comparing the energy of an area of an image to another area of the image. So the energy band comparison used by Haitsma and Kalker (2002) can also be seen as just computing the output of a Viola-Jones filter.

The retrieval stage is also somewhat similar to the one used by Haitsma and Kalker (2002). Every possible hash value is saved in an inverted index / hash table.

1. The fingerprint of a query is extracted as described above

2. Then for each subfingerprint all matches within a hamming distance of 2 are retrieved from the database together with their positions in the databases

3. To check for correct time alignment, the authors use random sample consensus (RANSAC) (Fischler and Bolles, 1981)

1.3.3.2.2 Other algorithms using image processing techniques

Of course many other algorithms can be seen as using image processing techniques since they mostly operate on a time-frequency representation of the audio. However there are some papers that explicitly state that they use image processing techniques to achieve audio identification such as (Rafii et al., 2014) or (Zhu et al., 2010). The former is based on a constant Q transform (CQT) of the audio which is then converted into a binary image by a comparison to the median of a sliding two-dimensional window. The retrieval is then performed by calculating the hamming distance of every query frame to every database frame saving the results in a matrix of dimensions # of query Frames $\times$ # of database frames and searching for a match by detecting a $45°$ line with maximum hamming similarity with a hough transform[8]. The latter is based on an patented algorithm called scale-invariant feature transform (SIFT) which extracts – as the name implies – scaling invariant features points from an image that can be matched between two different versions of an image (see figure 1.5). In the computer vision domain SIFT is used among other things to stitch together a panorama image from several input pictures by finding corresponding feature points. A detailed description of the SIFT-Algorithm can be found in (Lowe, 1999).

Another notable fingerprinting system is the AcoustID system that was developed for the *MusicBrainz*-project by Lukáš Lalinský (Lalinský, 2011). It operates on a chroma representation of the audio and is based on the two papers (Jang et al., 2009) and (Ke et al., 2005) and by that is also similar to (Haitsma and Kalker, 2002). There are 16 filters similar to the ones in (Ke et al., 2005) which are applied to the chroma image of the audio. These yield 32 bit integers which are then used to retrieve a query by comparing bit errors like it is done in (Haitsma and Kalker, 2002).

1.3.3.2.3 Possible gains and pitfalls when using image processing techniques

A compelling reason for using image processing techniques is that there is a wealth of tested and efficient algorithms to compare images. There are feature point extractors, stable region extractors and algorithms capable of identifying two pictures of the same object taken from different angles and distances.

A possible drawback of this approach is that all these algorithms are optimized to work with similarities and features present in real world pictures which don't necessarily have to be the similarities and features one would want to use when dealing with audio. Nevertheless the authors of the algorithms that use image processing techniques report that they perform just as well if not better than many "conventional" audio identification algorithms.

1.3.3.3 Other approaches

There are of course entirely different approaches for audio identification than the ones presented here so far.

[8]The hough transform is a technique to detect objects and patterns in images such as lines, circles and basically everything that can be parameterized. A good summary on how to find lines in an image with the hough transform can be found in Duda and Hart (1972).

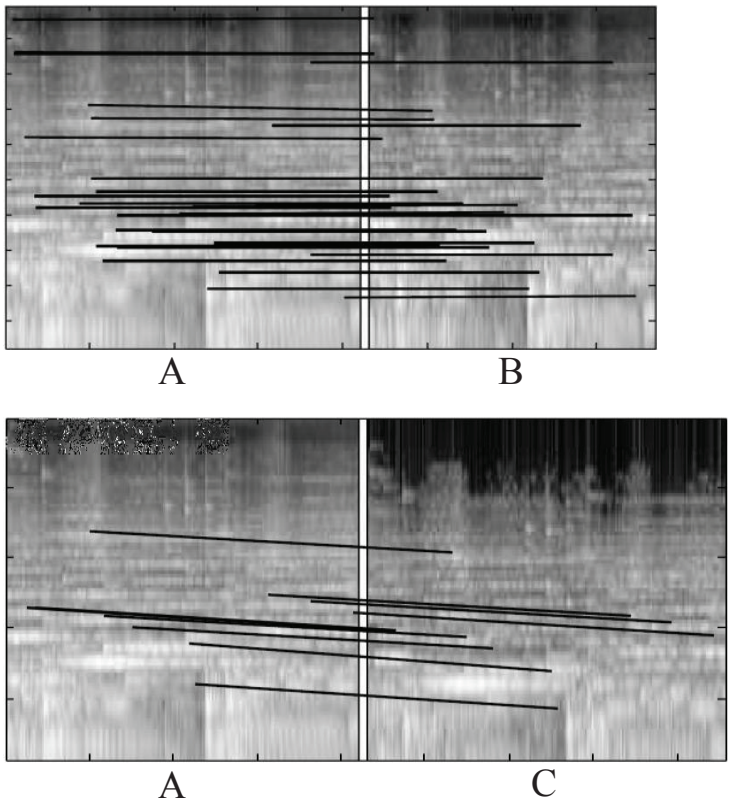


Figure 1.5: Matching Points in time scaled (B) and pitch shifted (C) versions of the original excerpt (A). (Zhu et al., 2010)

One example is the *waveprint* system by Baluja and Covell (2006) which is based on the wavelet transform introduced by Graps (1995). The approach is to compute an STFT with the parameters from Haitsma and Kalker (2002) and then also summarize the frequency axis into 32 logarithmically spaced frequency bins. On this time-frequency image the authors then apply a wavelet transform and keep only the sign of the top t magnitude wavelets. To enable an efficient nearest neighbor search with LSH in the retrieval stage these sparse wavelet representations are then converted to MinHash values (Rajaraman et al., 2012). In the retrieval stage LSH is used to retrieve only a few candidate database entries that can be then compared by using the more computationally expensive technique of dynamic time warping (DTW). This system was patented by Google in 2006 (Baluja and Covell, 2013).

In 2011 Ramona and Peeters developed the *Audioprint* system at the Institut de Recherche et Coordination Acoustique/Musique (IRCAM) and in 2013 updated it to cope better with pitch shifts (Ramona and Peeters, 2011, 2013). The system is different from many other systems in that uses two nested STFTs instead of relying on spectral peaks or energy differences.

This section could be expanded to cover all possible algorithms but to wrap this section up: there is a wealth of approaches to the problem of robust audio identification. There are systems that use …

- …a non-negative matrix factorization (NMF) (Deng et al., 2011)

- …onset detection to segment audio and gain time scale invariance (Bardeli and Kurth, 2004).
- …no tonal features at all and instead compute a Cyclic Beat Spectrum (Kurth et al., 2006).
- …machine learning to find the most robust features for fingerprinting instead of an heuristic approach (Burges et al., 2003).
- …MFCC in combination with hidden Markov model (HMM) in AudioDNA (Cano et al., 2002b).
- …global fingerprints that desribe the audio object as a whole (Kim and Narayanan, 2008; Kim et al., 2008; Guzman-Zavaleta et al., 2014).
- and many more.

One can even turn to neighboring disciplines to get some inspiration for a new fingerprint system.

A related research area is that of cover song identification which deals with not only finding the query song in a database but to also return cover versions of that song. The definition of a cover version of course is debatable as a cover version of a song could be transposed, use different instruments, change the structure of the song itself, etc. Due to the broader scope of this problem it is difficult to solve this task efficiently and therefore one cannot expect to get computationally efficient solutions from this research area.

A slightly easier task is that of the live version identification which aims to find a live version of the some recorded piece of music. In this case one can be rather certain that the live version is rather close to the original record and only differs slightly in timing, instrumentation and general structure. In this it is closer related to the task of audio identification.

1.3.4 Problems & limitations of existing fingerprint systems

As can be seen from this chapter there are a lot of fingerprint systems in the literature that yield good results for the use cases that they were designed for. Yet every system has its downsides doesn't and doesn't fulfill all the requirements that I establish for this thesis in section 2.1.

The most common problem is the lack of scalability of the retrieval process. Modern music databases like that of the Shazam system have several million songs in them and a good fingerprinting system should be able to perform fast searches without sacrificing accuracy.

Employed algorithm 2

2.1 Requirements

In theory, a well designed audio fingerprinting system should be able to compete with and outperform a real human being in the task of identifying previously heard audio due to the processing power available on computers and nearly unlimited storage space, given of course that the accuracy of retrieval is the same. Computers allow for parallel lookups of multiple queries in sub-realtime – since they don't need to listen to the songs and operate on a signal representation of the audio – and the databases can contain more songs than any human could possibly listen to in one life. What this implies is that signal degradations that pose no problem to a human's ability to identify audio shouldn't affect the accuracy of the fingerprinting algorithm.

At the time of writing, no study known to the author exists that tests the human ability to recognize sequences of audio that are subjected to various signal degradations. Such a study is necessary to compare the algorithm's performance against that of a human being and identify the signal degradations and alterations that don't affect the ability of a human to recognize audio. These degradations and their intensity could then be used to build a standardized evaluation environment for fingerprinting systems. However from personal experience one can assume that a wide variety of signal degradations don't influence our ability to recognize a piece of music. Be it noise and low dynamic range as in the case of the earliest recordings ever made or elaborate time stretches and pitch shifts as in the latest popular record releases. The website *whosampled.com* and the size of its database[1] is a good indicator that the human brain and auditory system is well equipped to recognize even strongly altered sequences of music.

The requirements for an audio identification algorithm can be manifold but as hinted in the title of this thesis the system used in this thesis should be above all invariant to pitch shifts and time scalings that stay the same over time. *Time scaling*, sometimes called time stretching, is the process of changing the duration/playback speed of an audio signal

[1] whosampled.com is a website where users can post sources of sampled music in the fashion "the section beginning at 1:15 of song A is sampled by song B at 2:45". The database contains direct connections among over 358,000 songs and 124,000 artists according to its website.

without changing the pitch. *Pitch shifting* is the process of changing the pitch of an audio signal without affecting it's duration/speed. A third signal alteration that is closely related to the former two is the *tempo change* which shifts the pitches *and* changes the time scale of an audio signal at the same time. It is the effect that one gets when playing back an audio signal at a different speed than the one it was recorded in. This phenomenon can be observed in analogue media such as vinyl records and cassette tapes or by resampling a digital audio file and playing it back with it's original sampling rate. By speeding up the playback speed (reducing the duration of the audio file) the pitches move up by the same percentage of the speed-up compared to the original speed and similarly for slowing down. Thus a tempo change can be seen as a time scaling and a pitch shift combined. The time scaling and pitch shift ratios in this alteration are inversely related, meaning that a time scale ratio of $\frac{1}{2} = 0.5 = 50$ % (equivalent to doubling the playback speed) leads to a pitch shift ratio of 2, causing all pitches to be shifted by one octave.

This thesis focusses on the former two signal alterations because they are commonly used in radio broadcasts to shorten or lengthen a song to fit a schedule or by users of services like *youtube.com* to circumvent the DRM algorithms when uploading a video / song to the service. Both are also important for modern DJ applications that enable users to either match the tempos of two records without affecting the pitch or deliberately changing the pitch of one piece of music to fit another piece of music better or both at the same time. The possibility of independently setting the pitch shift and the time scaling was not possible in the past because the only way to influence the tempo of a song on a turntable was the (unfittingly named) pitch control which controlled the physical playback speed and with it the pitch shift at the same time. Time scale modification and pitch shifting are now available in most audio editors and music production applications resulting in sample based music that uses time scaled and/or pitch shifted samples of other music. Yet, they are not satisfyingly detected by many algorithms including the previously discussed Wang et al. (2003); Haitsma and Kalker (2002); Cremer et al. (2001).

A fingerprinting system that is robust to time scale modifications and pitch shifts is of great use for broadcast monitoring and DRM as well as finding songs in DJ sets that are highly time scaled and/or pitch shifted. Of course the typical order of magnitude of time scaling and pitch shift needs to be defined. For the broadcasting case it depends on wether the audio is subjected to a tempo change or a time scaling. Since a tempo change of 6 % results in a pitch shift of roughly one half tone it is prohibitive to use tempo changes much larger than this to prevent a noticeable and unpleasant signal degradation. The most common turntable used by DJ is the Technics SL-1200/SL-1210 which allows the user to change the tempo of playback by ± 8% by default. However the pitch control can be adjusted to cover a larger or smaller range with ± 12 % being the largest possible range. Modern turntables are capable of an even larger range of tempo change going up to 50 % or even higher. The sense of which is debatable.

Of course the system also has to be somewhat robust against basic signal degradations such as compression, noise, spectral filtering & inharmonic distortion to be of any use in practice, since these are very common degradations.

In order to be able to find short snippets of query audio in a database and not only whole files, the fingerprints have to be local. This means that every analysis frame gets an own fingerprint that is searchable in the database rather than one fingerprint for a whole file.

Another important requirement is that the algorithm must not violate any patents or

copyrights since the system was developed to be freely usable for commercial purposes. This renders many research papers unusable for this thesis. All major fingerprinting algorithms (Shazam, Philips, Fraunhofer) are patented as well as some underlying algorithms in other systems such as SIFT and speeded up robust features (SURF).

There are few algorithms that meet all the requirements:

- The dynamic chroma approach by Kim and Narayanan (2008) can be used for fingerprinting time scaled and pitch shifted audio but generates global fingerprints so one has to find a way to seperate the global fingerprint into local fingerprints.

- Fenet et al. (2013) proposed a fingerprinting system that is also capable of finding similar pieces of audio. It is based on a chroma representation of the signal on which a note onset detection is performed to gain robustness against time scale modifications. The chroma representation gets rid of timbral information and sums all pitch information into one octave. Pitch shifts that are an integer number of half tones shift the chroma cyclically by the number of half tones. The onset detection is based on the paper by Alonso et al. (2005). The fingerprinting system uses a two-level approach to retrieval meaning that in a first stage a fast but coarse search for the M most probable candidate songs is performed and in a second stage those M candidates are subjected to a more detailed and computationally more expensive DTW based search.

  The downside to this approach to fingerprinting is that the chroma is a very tonality-centric feature which doesn't work that well for mostly percussive music like techno or music that doesn't use the western note system. Also the whole system depends on the performance of the onset detection algorithm and depending on the number M the DTW can be a bottleneck.

- Malekesmaeili and Ward (2014) is based on extracting local maxima from a modified chroma representation that resembles a CQT-spectrogram. The local maxima are analyzed for stability to be able to assign a stable region with a distinct time width to the local maxima. This way the chroma representation is chosen to make the algorithm invariant to pitch shifts and the stability testing is responsible for the time scale invariance that this algorithm is supposed to have. From all the fingerprinting systems this is tested to work with the largest pitch shifts and time scalings in the literature.

All other fingerprinting systems that are known to the author either use patented algorithms or don't account for (large enough) pitch shifts and time scalings.

2.2 Underlying algorithm

The algorithm of choice is the one by Malekesmaeili and Ward (2014) and will be summarized here together with the implemented modifications. It is chosen because it doesn't violate any patents and according to the authors is very robust against even large time scalings and pitch shifts. It is not so tonality centric as the approaches by Kim and Narayanan (2008) and Fenet et al. (2013) by using an extended version of the chroma spectrum that resembles a CQT and is not reliant on an onset detection.

Figure 2.1 shows an overview for the fingerprint extraction scheme taken from Malekesmaeili and Ward (2014).

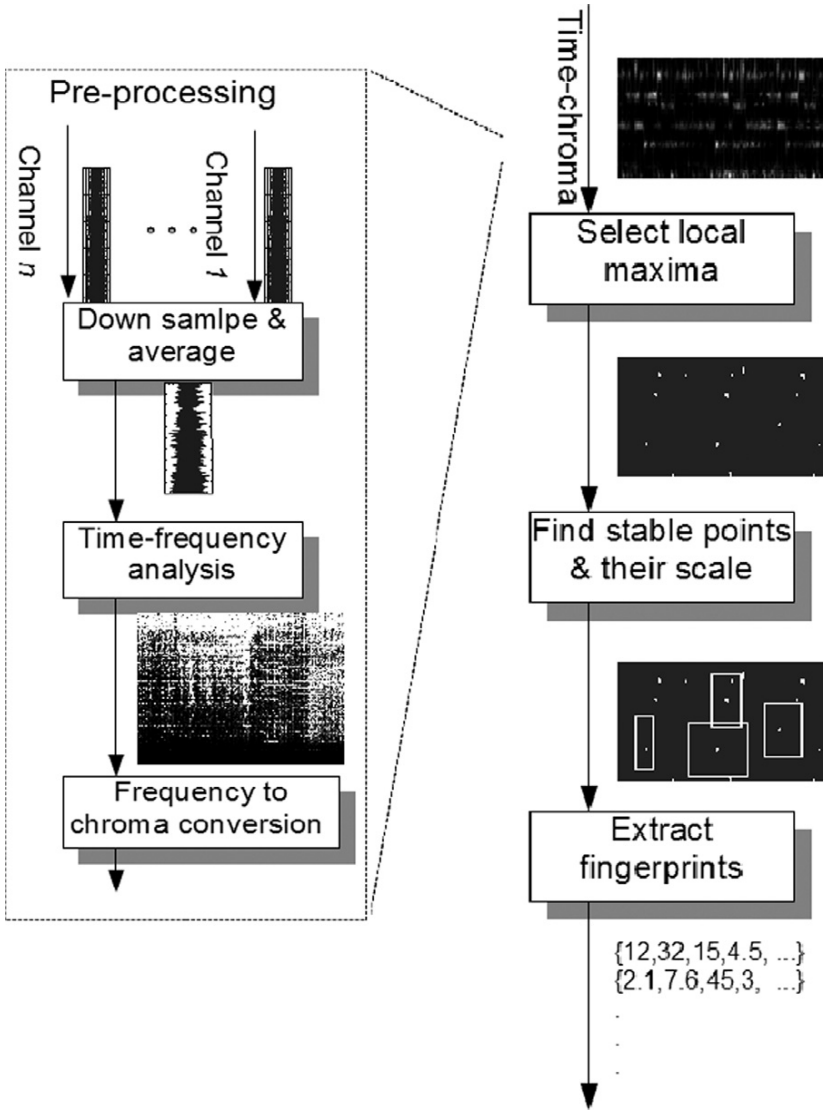


Figure 2.1: Overview of the fingerprint extraction scheme taken from (Malekesmaeili and Ward, 2014).

In short, the algorithm transforms the audio to an expanded time-chroma image from which it extracts local maxima and tests them for stability. The stable points then have a stable region / a time scale associated with them. The final fingerprints are the low-frequency coefficients of the two-dimensional discrete cosine transform (DCT) of the stable regions.

The next subsections contain a more detailed description of the algorithm.

2.2.1 Preprocessing

First the input audio is pre-processed by averaging all channels to one mono channel and resampling this channel to a sampling frequency of $f_s = 8820$ Hz (which is one fifth of the CD standard of $f_s = 44.1$ kHz).

2.2.2 Conversion to time-frequency domain

The next step is to perform a STFT on the audio. In their paper the authors found that using a Hanning window of length $l = 0.1$ s and an overlap of $l_o = 0.75 \times l = 0.075$ s (a hop size of $l_{hop} = 0.025$ s respectively) yields the best results regarding the robustness of local maxima.

This was tested by taking one long audio file comprised of short snippets ($10 - 20$ s) from different songs and generating altered versions with:

- 2 pitch-shifted version with $\pm$ 1 whole tone
- 2 time-scaled versions with a factor of $1.1^{\pm 3}$
- 1 noisy version with a signal-to-noise ratio (SNR) of 40 dB

Then for each local maximum[2] in the original version it was checked if there is a local maximum in the corresponding position in the altered version (of course the pitch shift and time scaling had to be accounted for). If this was true, the pair of local maxima was saved as a matched pair. Also to see how similar the vicinities of matching pairs are, rectangular patches of constant height and time scale corrected width were extracted and since the patches can have different widths their low frequency DCT coefficients are compared via correlation. Doing this for several combinations of l and l_o the authors then took the combination which yielded the highest number of matched pairs and the highest average correlation scores.

From this time-frequency representation the authors generate a new kind of time-chroma representation which they call chroma^n. This is basically a standard time-chroma-representation that is spread out over more than one octave in the frequency axis. The reason for this approach is that the conventional time-chroma is not specific enough for a fingerprinting system due to the fact that all pitches/frequencies are compressed into one octave. The way that chroma^n works is that one sets the lowest frequency f_0 to be analyzed and associates a pitch p_0 with it. Any other pitch above p_0 is then associated with a frequency

$$p_i \sim f_i = 2^{\frac{i}{m}} \cdot f_0, \text{ with } m \in \mathbb{N}, \tag{2.1}$$

where m is the number of bins per octave. The n in chroma^n stands for the number of octaves that the chroma representation will be expanded to. For given m and n, the frequencies

[2] The local maxima were extracted as described in section 2.2.3

belonging to a pitch p_i are: $\{f_i, 2^n f_i, 2^{2n} f_i, \ldots\}$. So, for $m = 12$ and $n = 1$, chroma^n is equivalent to the conventional chroma representation. The chroma^n set is then computed by applying logarithmically spaced filters to the STFT coefficients that capture the energy for each pitch. To find values parameters for n, m and f_0 the authors used a similar approach as for finding good values for l and l_0 but with the two latter set to the previously found values. This yields the values $n = 4$, $m = 72$ and $f_0 = 80$ Hz for which over 66 % of the original maxima are also detected in the altered versions.

2.2.3 Extraction of interest points

Once the chroma^n is computed the next step is extracting local maxima from the image. These are called interest points. The extraction is performed by comparing the values of every pixel to its neighboring pixels in a window of size 5×5 centered around the considered pixel and if the pixel in the center has the largest value it is considered an interest point.

2.2.4 Stability analysis

As a next step the extracted interest points are analyzed for stability. This is done for two reasons: to discard points that aren't robust enough and secondly to assign time scales to the points in which the area around the points is not changing much. The analysis is performed by extracting 60 patches of constant height (72 Bins $\equiv$ 1 octave) and varying width in time ($40 - 160$ Bins $\equiv 1 - 4$ seconds) from the chroma^n-image around each interest point. An interest point is considered stable if most of the patches around that point are similar. To compare the similarity between two patches the authors use a dictionary of 10 representative patches; every extracted patch is compared to each dictionary patch by computing the correlation of the lowest 12×12 two-dimensional DCT coefficients. If more than half of the 30 patches are most similar to one and the same dictionary patch the interest point is selected as a stable feature point and assigned the most similar dictionary pattern as the *type* and the time scale with the highest correlation score as the *scale*.

2.2.4.1 Computation of the dictionary

The dictionary is generated by extracting patches of one octave ($\equiv$ 72 bins) in heigth and 2 s ($\equiv$ 80 bins) in width from a subset of songs from the audio database. These patches are then analyzed using a 2D DCT and only the 12×12 lowest frequency coefficients (minus the DC component) are flattened into 1×143 vectors and saved into a matrix in which every row corresponds to one window. This matrix is then fed into a k-means clustering algorithm (see section 2.2.4.1.1) to extract the 10 representative DCT vectors. When converted back to the time-frequency domain by reshaping the resulting 10 row vectors into 12×12 matrices[3] and using an inverse 2D DCT one gets the 10 representative chroma^n patches that form the dictionary. The 10 resulting patches from Malekesmaeili and Ward (2014) can be found in figure 2.2. In general the dictionary will be influenced by the algorithm used to extract interest points and the used time-frequency representation when used on the same audio data set.

[3] setting the DC component to 0.

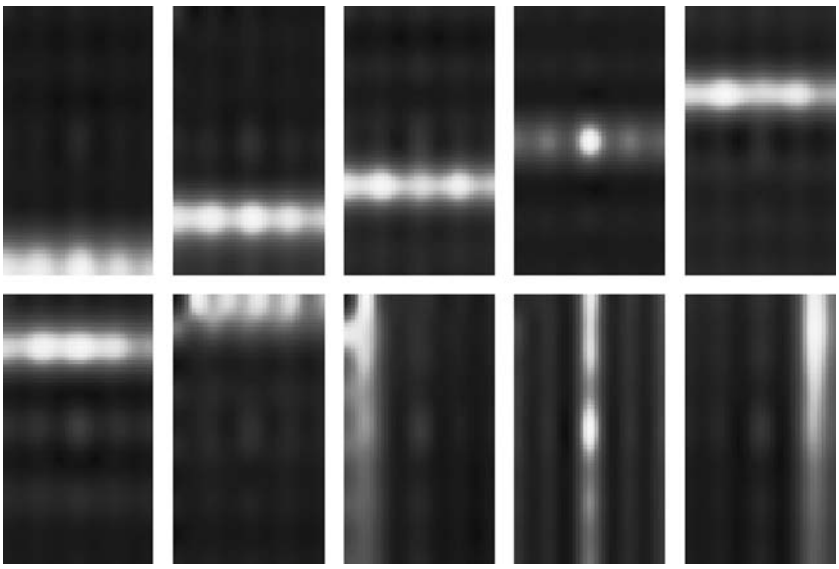


FIGURE 2.2: Dictionary of representative time-chroma patches from Malekesmaeili and Ward (2014).

2.2.4.1.1 Clustering with k-Means

The k-Means algorithm is a relatively simple algorithm to partition input data into k clusters with the goal to minimize the within-cluster sum of squares (WCSS) which are the squared distances from the data points (in the form of vectors) to one another. It is based on the paper by Steinhaus (1956) and was given it's name by MacQueen (1967). However the standard algorithm was described by Lloyd (1982).

The algorithm consists of three basic steps:

1. **Initialize** k means in the data set. This can be done in many different ways but for the sake of simplicity we just assume that for every mean a random data point is picked from the data set.

2. **Assign** every data point to its closest mean. The distance measure used here is the euclidian distance but this can be changed to any other distance measure.

3. **Update** the means by calculating the new means from the updated sets of which point belongs to which mean.

Steps 2 and 3 are executed either until convergence is reached (meaning the change in total sum of WCSS from one iteration of steps 2 and 3 to the next is below a certain threshold) or until a predefined number of iterations is reached. Figure 2.3 shows a simplified example of how the algorithm works.

The implementation used in this thesis is the k-Means++ extension of the algorithm that uses a more elaborate initialization procedure. Optimally chosen initial means allow for a faster convergence and maximize the chances of the total sum of WCSS reaching a global minimum instead of a local one. Other than the initiliaziation stage everything else is equivalent to the standard algorithm by Lloyd (1982). The maximum number of iterations is set to 500 to allow for convergence.

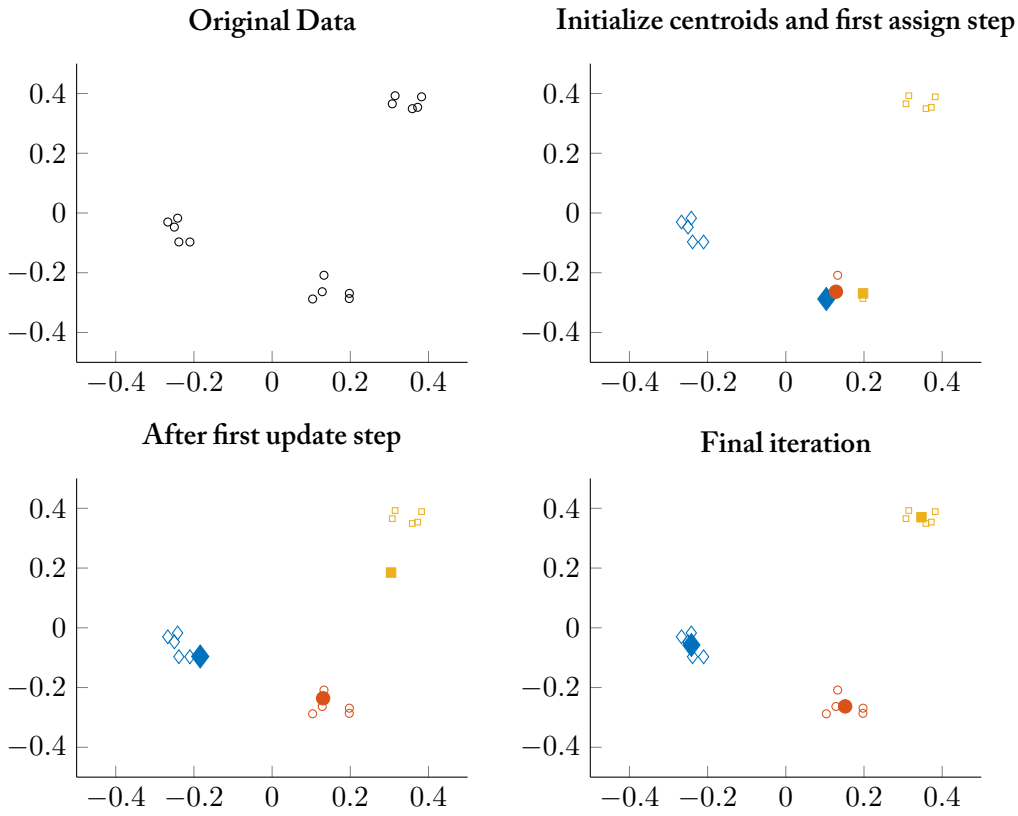


Figure 2.3: Visualization of the k-Means algorithm.

2.2.5 Convert stable points to fingerprints

From every stable point one fingerprint is computed. Every stable point has a patch associated with it of size $m \times scale$ with m being one octave ($\equiv$ 72 bins) and $scale$ being the scale that was found in the stability analysis. A two-dimensional DCT is computed from this patch and the lowest 12×12 DCT coefficients are transformed into a vector with zero mean and unit length[4] discarding the DC value (yielding a vector of length 143). These vectors are the local fingerprints. Together with their *scale*, *type* and time and frequency coordinates they are saved in the database with an unique song ID.

2.3 Lookup in the database

The lookup of a query in the database of fingerprints is performed by using an exhaustive search meaning that every fingerprint from the query is compared to every fingerprint in the database. Since the fingerprints can be considered vectors of unit length, the distance from one fingerprint to the other can be quantified by calculating the angle between the two as described in Malekesmaeili and Ward (2014):

[4] In the paper the authors write "zero mean unit variance" which clearly is a mistake since equation (1) in (Malekesmaeili and Ward, 2014, p. 314) would then be lacking a normalization with the vectors lengths. Also, database items with a large length would always score higher than items with a small length in the retrieval.

"Let X be the set of fingerprints extracted from the original song and Y be the set of fingerprints extracted from an attacked version[5]. A fingerprint $x \in X$ is matched to $y \in Y$ if it is closer to y than any other fingerprint in X by a factor of α. To measure the distance between two fingerprints we consider them as two vectors and measure the angle between them. For example, for the proposed algorithm, x is matched to y if

$$\arccos\left(y^T x\right) \leq \alpha \arccos\left(y^T x'\right) \; \forall x' \in X - \{x\}" \tag{2.2}$$

The factor α was set to $\alpha = 0.6$ as in Zhu et al. (2010).

For the evaluation of the their system Malekesmaeili and Ward (2014) use one long query comprised of short snippets ($10 - 20$ s) from different songs in a database of 250 items. The goal then is to identify all of the 100 snippets and their position in the query and in the database. A secondary goal is to correctly estimate the time scaling ratio and/or pitch shift if one was applied to the query. The output from the system is of the form "…the section t'_0 to t'_1 in the query is copied from t_0 to t_1 of song s in the database …" This is a somewhat different scenario from that of a conventional fingerprint system in which a user has one query that is mostly comprised of only one song and wants to know the ID of that song.

Nevertheless the authors propose a method to assign the most likely database item to a time interval which can be used to develop a threshold for a match depending on how much of the query has been assigned to the correct database item.

This system consists of two stages: a coarse estimation of which database item is most likely in which time interval of the query and a refinement stage that aims at improving the estimates from the pre-processing step and estimates the time scaling and pitch shifting ratios as well.

The pre-processing stage works as follows: Every query fingerprint that has been successfully matched to a database item in the exhaustive search has a song ID assigned to them. Then a sliding window of length 10 s is moved in a step size of 1 s over the song IDs to find dominant IDs among them. If over 70 % of the song IDs are one and the same the window gets assigned this song ID. This way continuous passages of one song can be detected in the query. The output of the pre-processing stage is a list of songs with the time intervals (t_i, d_i)[6] they occur in the query associated with them.

The refinement stage takes as inputs:

- The detected songs s_i and time intervals (t_i, d_i) from the pre-processing stage.
- The query's fingerprints $f \in F$ alongside with their following attributes:

  $ID(f)$: The song ID from the database item the fingerprint was matched with

  $\hat{a}(f)$: the ratio between the scale assigned to the query fingerprint and the scale assigned to the match in the database

  $\hat{\Delta p}(f)$: the difference between the chroma coordinate of the query fingerprint and that of its match

  $t_q(f)$: the time coordinate of the query fingerprint in the fingerprint itself

[5] "Attacked version" meaning an altered or degraded signal

[6] t_i being the start time of an interval and d_i being its duration.

$t_{db}(f)$: the time coordinate of the matching fingerprint in the database song

The algorithm then performs these steps for every detected song s_i:

1. Keep only fingerprints that are associated with song s_i and lie inside the time interval (t_i, d_i)
2. Use the $\hat{a}(f)$ to exclude outliers from the further computations. This is achieved by computing a histogram from the $\hat{a}(f)$ values, smoothing them with a gaussian kernel of size 21 and a standard deviation of $\sigma = 2$ and discarding all fingerprints that contribute $\hat{a}(f)$ that are outside a radius of $\delta_a = 0.1$ around the maximum. if less than 60 % of all scale ratio lie in a region of ± 0.1 around the most frequent ratio the function exits with no parameters estimated.
3. Subtract the mean value of $t_q(f)$ from the remaining fingerprints f and do the same for $t_{db}(f)$. Compute the offsets between the $t_q(f)$ and $t_{db}(f)$ values $\hat{b} = t_q(f) - t_{db}(f)$ and further remove outliers by the same method as for $\hat{a}$ but with a radius of $delta_b = max(0.1 \cdot max(\hat{b}), 0.01)$.
4. Having eliminated fingerprints that are most likely outliers w.r.t. a and b, the most likely combination of scale ratio (= time scaling ratio) a and offset b is found by applying a linear regression minimizing the sum of squared errors:

$$(a, b) = argmin_{a,b} \sum_{f \in F_{pruned}} (t_q(f) - (a \cdot t_{db} + b))^2 \tag{2.3}$$

5. The pitch shift is estimated by searching for the most frequent pitch shift of the remaining $\hat{\Delta p}(f)$.
6. To further refine the detected boundaries, the first and the last fingerprints in the time interval (t_i, d_i) that are compatible with the estimated values of a and b. These two points along with their scales define the boundaries of the detected snippet.

The measure to evaluate the lookup is the ratio of correctly identified time intervals to the total length of the query file. This way it is possible to derive a sensible threshold for a "hit" afterwards which is not given by the underlying paper.

2.4 Modifications

The underlying algorithm is modified in two main ways to test and evaluate the influence of

1. the "input image", i.e. the time frequency representation
2. and the algorithm used to extract interest points from the time-frequency representation.

The former is motivated by the fact that the chroma^n representation is similar to a CQT in that it sums fast Fourier transform (FFT) bins into logarithmically spaced frequency bins. Since this summing of bins is just an approximation of a CQT it will be evaluated if using

the actual CQT yields better performance at the price of being insignificantly more computationally expensive. Trying to reduce the complexity of the CQT-spectrum, a harmonic sum spectrum (HSS) is used to reduce most harmonic to their fundamental.

The latter is done because the original interest point method used in the paper is fairly simple which raises the question if it could be improved by using a different approach that extracts more robust and stable interest points. Interest point detectors are commonly used in the field of computer vision and many are available in the *Computer Vision System Toolbox* for Matlab from which two are used in this thesis (Harris corner detector and SURF).

Since this is the main contribution in this thesis the employed input images and interest point detectors are described in more detail in the following subsections.

2.4.1 Input images

2.4.1.1 Constant Q transform

The chroma^n approach can be seen as an approximation of a CQT with the same parameters (hop size, number of bins per octave, number of octaves, etc.). The CQT was introduced by Brown (1991) to propose a more musical transform than the widely used discrete Fourier transform (DFT). A DFT yields linearly spaced frequency bins that have constant resolution for all center frequencies. However the tones that make up western music are spaced logarithmically in the frequency domain which means that a DFT will not map the tones efficiently to frequency bins. A CQT on the other hand has frequency bins with a constant Q factor meaning a constant ratio of center frequency to resolution. This results in narrow frequency bins (high resolution) for low frequencies and wider bins (low resolution) for high frequencies on a linear frequency scale.

The CQT has the advantage that for a tone with harmonics a frequency shift is a linear displacement on the frequency axis.

One downside to the CQT is that is it computationally more expensive than a DFT. This issue is discussed and solved by Brown and Puckette (1992). However another important limitation of the CQT is that the time resolution is frequency dependent and thus the time resolution for low frequencies is worse than for high frequencies.

The parameters used to compute the CQT are chosen to be as close to the parameters in Malekesmaeili and Ward (2014) as possible:

- The hop size is also chosen to be $l_{hopsize} = 0.025\ s$
- The minimum frequency is $f_0 = 80$ Hz
- The number of octaves is $n = 4$ resulting in a maximum frequency of $f_{max} = 80$ Hz$\times 2^4 = 1280$ Hz
- The number of bins per octave is $m = 72$

The computation of the CQT is performed using the company-owned CQT/harmonic sum spectrum of the constant Q transform (HSS-CQT) Matlab toolbox provided by Holger Kirchhoff which is an implementation of the fast CQT algorithm exploiting the efficiency of the FFT from Brown and Puckette (1992).

2.4.1.2 Harmonic sum spectrum for the CQT

The HSS was introduced by Noll (1969) to determine the pitch in human speech and later was used by Klapuri (2006) to implement a multiple fundamental frequency estimator.

The idea behind the HSS is to reduce harmonic tones to their fundamental so as a result one gets a spectrum with less or no harmonics present and pronounced fundamental frequencies. This can be advantageous for the task of interest point detection because the spectrum gets simplified and the fundamentals more clear in comparison to the rest of the spectrum making them salient local maxima.

The basis for the HSS-CQT is the CQT from above. The algorithm takes as parameters:

- Number of partials to consider `numPartials`
- The ratio by which the fundamental and partials should be reduced `partialReduction`
- Minimum frequency to consider f_0
- Maximum frequency to consider f_{max}

Then the HSS is computed from that CQT by following these steps:

1. Initialize an array the size of the frequency axis `HSSspectrum` in which the HSS spectrum will be saved.
2. Repeat the next steps until either the sum of the spectrum has decreased below a threshold or a specified number of iterations is reached:
   a) Consider every CQT frequency bin within the range of $[f_0, f_{max}]$ a fundamental frequency and compute the sum of the fundamental frequency bin and `numPartials` of harmonics.
   b) Choose the fundamental with the highest sum from the previous steps and reduce all of its harmonics by the factor `partialReduction`.
   c) Save the sum of the subtracted amplitudes in the `HSSspectrum` at the index of the fundamental

By using the energy of the higher frequency partials of a fundamental tone one gains a better time resolution for lower frequencies that was lost by using the CQT.

The computation of the HSS-CQT is performed using the same toolbox as for the CQT.

Figure 2.4 shows an example of the three input images for an excerpt of a song. As can be seen the CQT is blurry for low frequencies and the HSS-CQT is a very much reduced version of the CQT.

2.4.2 Interest point methods

2.4.2.1 Original method by Malekesmaeli & Ward

The method used by Malekesmaeili and Ward (2014) works by checking for every entry in the matrix if it is the largest value in a given neighborhood and if that is true it is saved as an interest point. By default a window of size 5×5 centered around the considered point is used as the neighborhood.

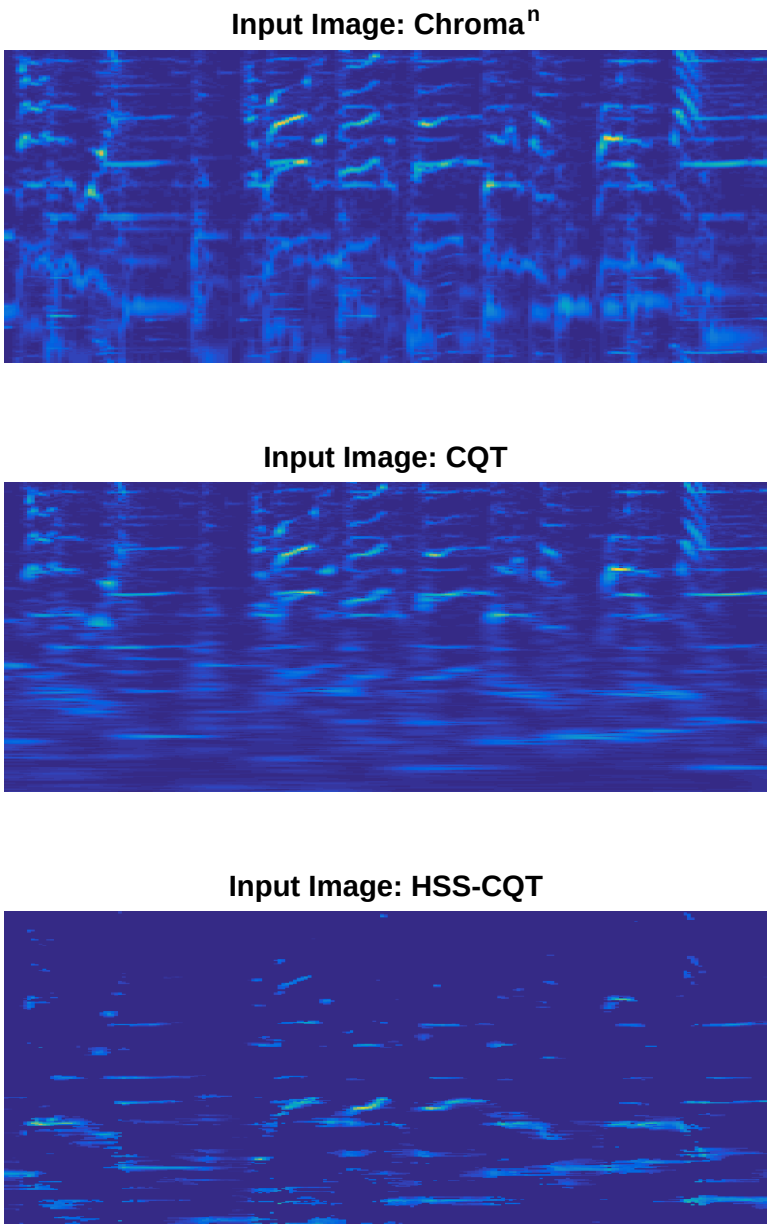


Figure 2.4: Comparison of the three input images for an excerpt of a song.

2.4.2.2 Method from the Shazam implementation of Ellis

Based on the paper by Wang et al. (2003) Dan Ellis implemented his own version of the Shazam fingerprinting system in Matlab which he made available online[7] (Ellis, 2009).

Since neither the paper by Wang et al. (2003) nor the patent by Wang and Smith (2002) elaborate on how to extract the local maxima from the spectrum, Ellis had to implement his own method, which is based on a decaying masking threshold for every frequency bin.

1. Initialize the masking threshold:
   a) Extract for every frequency bin the maximum value in the first 10 frames.
   b) Every maximum gets convolved it with a Gaussian kernel in order to generate a masking threshold around the maximum along the frequency axis.
2. For every time frame from left to right:
   a) Apply a decay constant in time to the masking threshold.
   b) Check if any points in the frame are above the masking threshold and if so mark the point as a potential local maximum and apply the Gaussian kernel to the point to spread it out along the frequency axis. Then continue with the next frame using the actualized masking threshold.

   The resulting maxima and masking thresholds for every frame are depicted in figure 2.5.
3. As we can see from figure 2.5 the masking thresholds are only decaying in one direction. To correct that repeat the last step for a reverse order of the time frames so that the masking is performed symmetrically.
4. The remaining local maxima are the interest points.

One could get similar results by convolving the spectrum with a gaussian kernel of the right dimensions and then picking the remaining local maxima with a method like the one in section 2.4.2.1. The convolution smoothes the image to get rid of local maxima that are not salient enough.

2.4.2.3 MinMax filter method

This method uses two nonlinear 2D filters to find local maxima in predefined area: a *max filter* and a *min filter*. A max filter substitutes a pixel's value by the maximum value in a predefined area around this pixel and thus dilates the area of local maxima values. A min filter does the same with the minimum values. In general the area around the considered pixel can be of arbitrary shape. The algorithm is as follows:

1. First the image is normalized so that all vales of the image are in the range $[0, 1]$ resulting in an image I_{norm}.
2. Next two images are derived form the normalized image:

[7] http://labrosa.ee.columbia.edu/matlab/fingerprint/

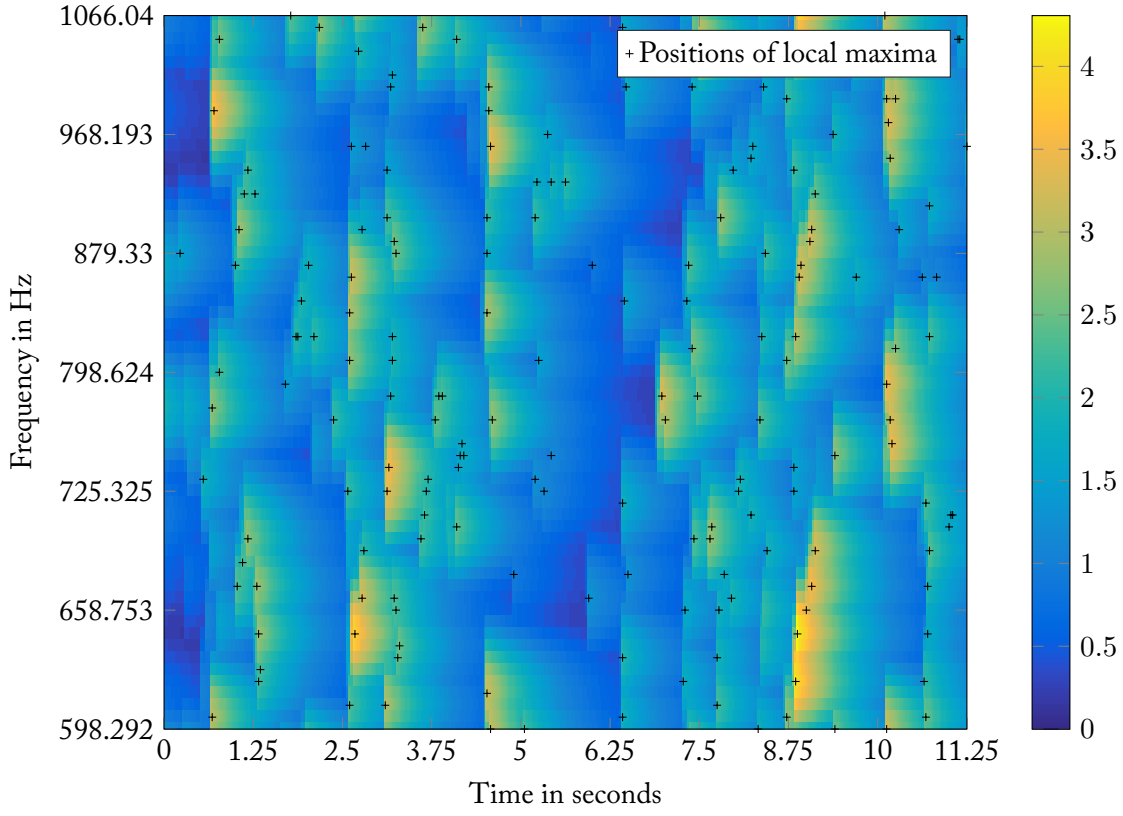


FIGURE 2.5: Example of some local maxima and their masking thresholds found by the algorithm by Ellis (2009) before the time frames are analyzed in reverse order to guarantee a symmetrical masking.

a) A max filtered version of the normalized image, $I_{maxFilt}$.

b) A min filtered version of the normalized image, $I_{minFilt}$.

3. Local maxima can already be extracted by picking the values in I_{norm} that are equal to the values of $I_{maxFilt}$ at the same pixel. However that would result in too many local maxima that are not salient enough to be robust against signal degradations. Therefore one last step is performed.

4. The min filtered version is subtracted from the max filtered version to check for prominent peaks. A salient peak has a large distance to the local minimum around it and so a threshold is introduced to filter out shallow local maxima.

By varying the threshold and the area around the pixel that is used to collect the statistical data one can influence to some extent the density of the local maxima. Setting the threshold too high results in only few local maxima which are very pronounced. Setting it too low will yield many local maxima, some of which are not very robust. Also increasing the area results in less local maxima and vice versa. In this work a threshold value of 0.01 and a rectangular area of size 18×3 pixels is used.

2.4.2.4 Harris corner detection method

In computer vision corners play a special role in several applications such as panorama stitching, video tracking and everywhere else one wants to match the contents of two similar images. Corners have the convenient property of being invariant to rotation and translation. A corner can be formally defined as the intersection of two edges. By this definition there has to be a large intensity change around a corner in all directions. In the computer vision literature the term "corner" is often used interchangeably with "interest point".

In an audio spectrum corners can be points in which notes either start or begin as well as very short notes or fast note changes and at percussive sounds.

One method to extract corners from an image is the Harris corner detection algorithm as proposed by Harris and Stephens (1988). The theory behind the algorithm is the following: Use a sliding window to search for corners. If there is a corner inside the window there will

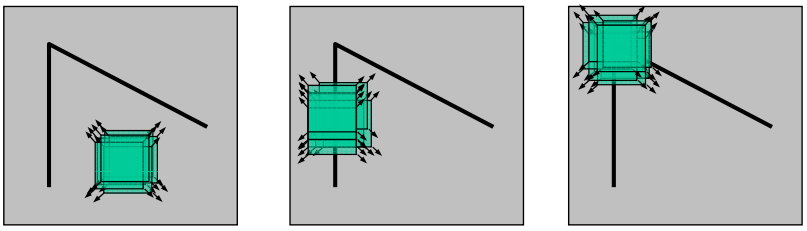


Figure 2.6: Basic idea of the Harris corner detector. (Frolova and Simakov, 2004)

be a large change in intensity values for small displacements in multiple directions. This is visualized in figure 2.6. Putting this in the form of an equation this yields:

$$E(u,v) = \sum_{x,y} w(x,y)\,(I(x+u,y+v) - I(x,y))^2 \tag{2.4}$$

with the intensity I, the intensity difference E, the window function $w(x,y)$ and a small displacement (u,v). For the detector to be isotropic the window function should be circularly weighted, e.g. using a Gaussian kernel. If there is a corner in the window E will be large. Given the displacement is small we can apply a Taylor expansion, keep only the first order approximation and rewrite the equation as:

$$E(u,v) \approx [u\ v]\, M \begin{bmatrix} u \\ v \end{bmatrix} \tag{2.5}$$

with

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \tag{2.6}$$

I_x and I_y are partial derivatives in the x- and y-direction. By analyzing the eigenvalues of M we see that:

- for a window with a corner in it M has two large eigenvalues,
- or an edge it would have one large and one small eigenvalue,
- for a flat surface both eigenvalues are small.

This is also summarized in figure 2.7

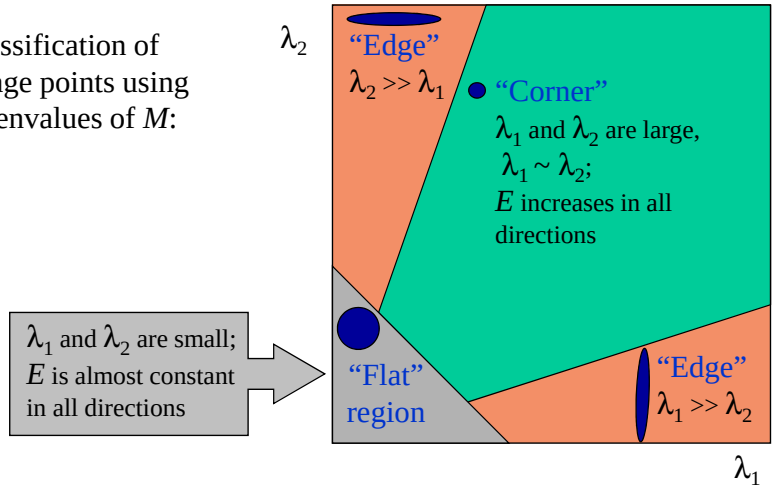


FIGURE 2.7: Classification of corners, edges and flat surfaces based on the eigenvalues of M in the Harris corner detector. (Frolova and Simakov, 2004)

Since the computation of the eigenvalue decomposition of a matrix is computationally expensive, Harris and Stephens (1988) suggest using the following scoring function

$$R = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2 = det(M) - k(trace(M))^2 \tag{2.7}$$

with eigenvalues λ_1 and λ_2 to determine if a corner is present in the current window or not. This measure exploits the fact that the determinant of a square matrix is equal to the product of the eigenvalues and the trace of a square matrix is equal to the sum of the eigenvalues. k is a tunable sensitivity parameter that has to be determined experimentally and usually lies in the range of $[0.04, 0.06]$ (Frolova and Simakov, 2004). R will be large for corners, negative for edges and if $|R|$ is small the area in the window is flat. The only thing left to do is to define thresholds that capture these three cases, which can be done experimentally. Figure 2.8 shows an example of the corner measure R for a chess board.

2.4.2.5 SURF method

The SURF algorithm is an image interest point detector and descriptor that is very similar to the SIFT algorithm. It was developed by Bay et al. (2006) and is patented (Funayama et al., 2009). For this thesis only the interest point detector is used. The detector is based on an approximation of a Hessian matrix that is applied to a Gaussian scale space representation of the analyzed image. These concepts will be presented in the following paragraphs and subsections.

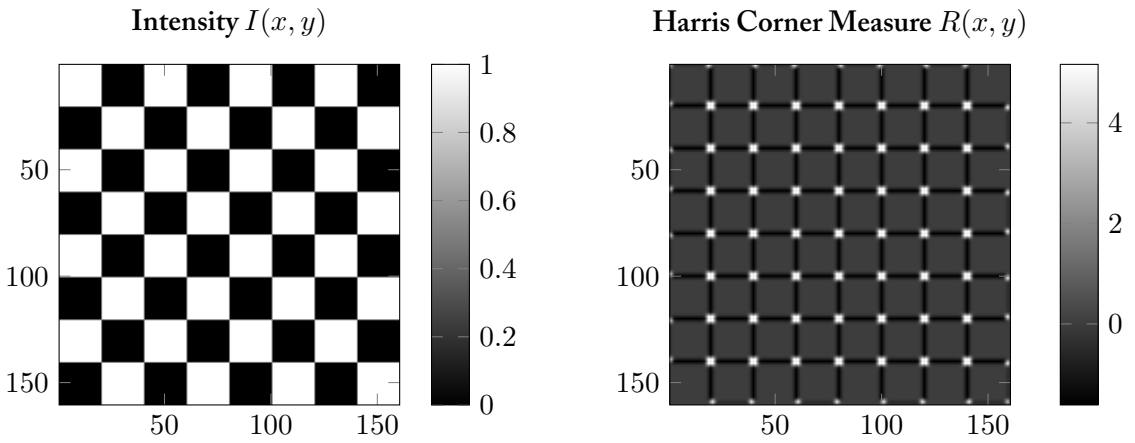


Figure 2.8: Intensity I (left) and Harris corner measure R (right) for a chess board pattern.

The Hessian matrix that the detector is based on is defined as:

$$H(x, y, \sigma) = \begin{bmatrix} L_{xx}(x, y, \sigma) & L_{xy}(x, y, \sigma) \\ L_{xy}(x, y, \sigma) & L_{yy}(x, y, \sigma) \end{bmatrix} \tag{2.8}$$

$L_{xx}(x, y, \sigma)$ being the result of the convolution of the image $I(x, y)$ with the Gaussian second partial derivative in the x-direction:

$$L_{xx} = I(x, y) * G_x x(x, y) \tag{2.9}$$

with the two dimensional Gaussian kernel:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.10}$$

and it's second order derivative:

$$G_{xx}(x, y) = \frac{\partial^2}{\partial x^2} G(x, y, \sigma) = \left(\frac{x^2 - \sigma^2}{2\pi\sigma^6} \right) e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.11}$$

and similarly for $L_{xy}(x, y, \sigma)$ and $L_{yy}(x, y, \sigma)$. Figure 2.9 depicts an example of a two-dimensional gaussian kernel $G(x, y)$, its second derivative in the x-direction $G_{xx}(x, y)$ and the mixed second derivative $G_{xy}(x, y)$.

The Hessian matrix is based on second derivatives of an image which can be used to detect edges and corners. The principle behind this is explained in section 2.4.2.5.1. The convolution with a gaussian kernel is performed for two reasons: to filter out noise and to extract local maxima on different scales of an image (see section 2.4.2.5.3).

Following the example of Lindeberg (1998) the authors use the determinant of the Hessian matrix

$$det(H(x, y, \sigma)) = L_{xx}(x, y, \sigma), L_{yy}(x, y, \sigma) - L_{xx}(x, y, \sigma)^2 \tag{2.12}$$

as a measure of local change around a point (x, y) in the image. Interest points are points around which the local change is maximal. Figure 2.10 shows an example of an image

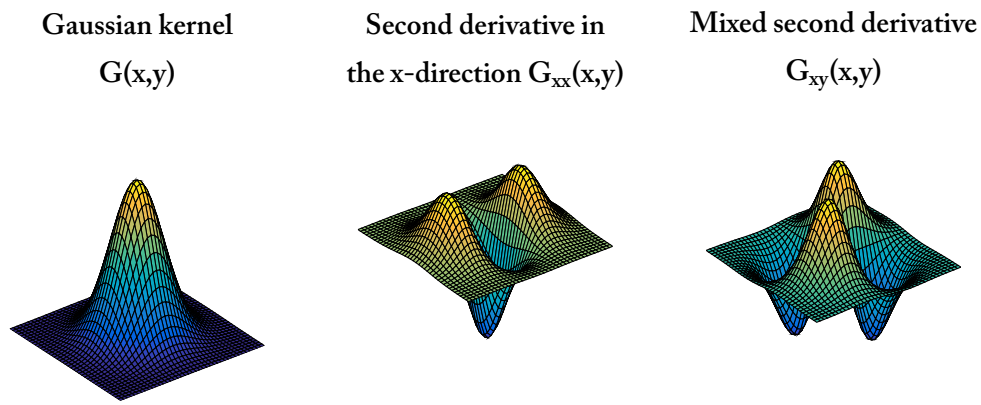
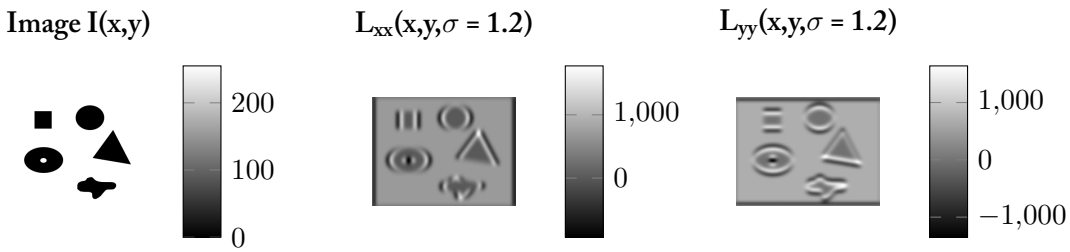


FIGURE 2.9: A two dimensional Gaussian kernel and its second derivative in the x-direction and the mixed second derivative.



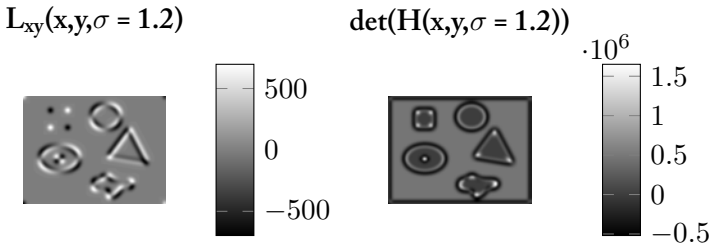


FIGURE 2.10: Image $I(x, y)$, its L_{xx}, L_{xy}, L_{yy} and its DoH response map.

I alongside its L_{xx}, L_{xy}, L_{yy} and its determinant of Hessian (DoH) response map which basically is just the determinant of the Hessian matrix evaluated for every point of the image. As can be seen in this in the DoH response map the determinant is very large for corners and has very large negative values for edges.

Since convolutions are computationally expensive the Gaussian second partial derivatives are approximated with box filters that allow the use of integral images to speed up the computation (see section 2.4.2.5.2). This approximated matrix is used in combination with a Gaussian scale space representation of the image to extract interest point on different scales.

2.4.2.5.1 Using second order derivatives to find edges and corners

Second (and first) order derivatives are widely used in the field of image processing to detect edges.

The principle behind using first and second derivatives to detect edges is shown in figure 2.11 for the one dimensional case: In the top plot we see a signal with two edges. Applying the first derivative to this signal yields the signal that is depicted in the middle and applying the second derivative to the signal on the left yields the signal in the bottom plot. The edge can be found by either searching for a positive maxima or negative minima in the first derivative of the signal or looking for zero crossings in the second derivative. In the SURF interest point detector the latter is used to find edges and corners. The downside of using the derivatives to detect edges is the sensitivity to noise. This is why in image processing it is more common to use the derivative in conjunction with a Gaussian smoothing filter to get rid of the noise. The two operations can be combined into one by applying the second order derivative to the Gaussian kernel as done in equation 2.11 above. Using different standard deviations σ for the Gaussian kernel allows for analyzing an image on different levels of detail (see paragraph 2.4.2.5.3).

To use the smoothed second derivates in digital image processing they have to be discretized and approximated. In the SURF algorithm the authors use box filters as shown in figure 2.12.

Utilizing these box like filters allows for the use of integral images to speed up the computation of the convolution significantly.

2.4.2.5.2 Integral images

In their paper Viola and Jones (2001) showed that it is possible to speed up the computation of the result of applying rectangular filters to an image.[8] This is achieved by first computing an intermediate representation of the input image $I(x, y)$ which the authors call the integral image. The integral image $I_\Sigma(x, y)$ has the same dimensions as the input image and at position x, y contains the sum of all pixels with indices smaller or equal than x, y:

$$I_\Sigma = \sum_{\substack{x' \leq x \\ y' \leq y}} I(x', y') \tag{2.13}$$

This way computing the sum of all intensity values in a rectangular area such as in figure 2.13 only takes 4 additions/subtractions regardless of the size of the area:

[8] The concept of summed-area tables was first introduced in another context by Crow (1984) though.

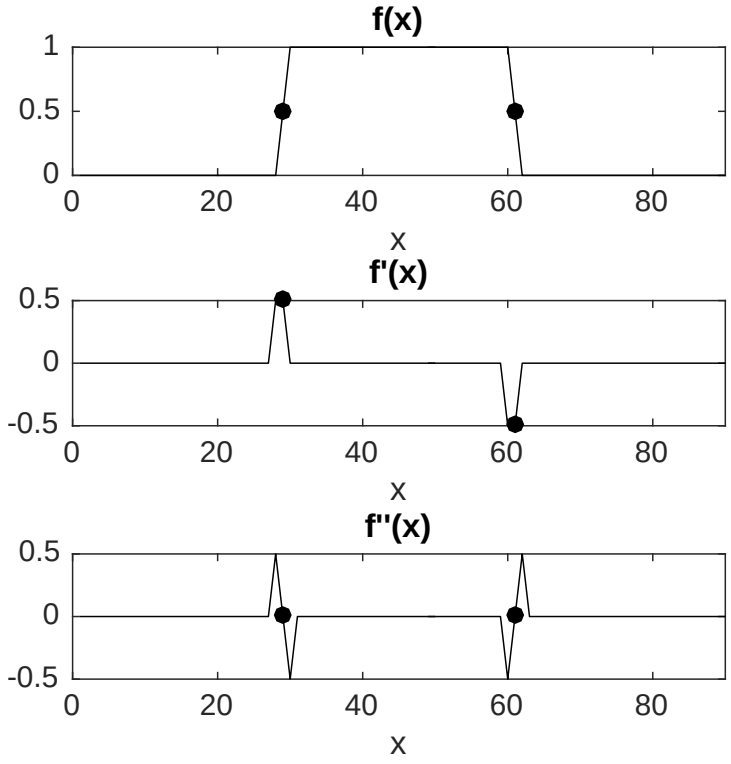


FIGURE 2.11: Signal with two edges and its first and second derivative.

$$\sum_{\substack{x_0<x\leq x_1\\ y_0<y\leq y_1}} I(x,y) = I_\Sigma(x_1,y_1) - I_\Sigma(x_0,y_1) - I_\Sigma(x_1,y_0) + I_\Sigma(x_0,y_0) \tag{2.14}$$

The advantage of using integral images is that the integral image has to be computed only once and then allows for a constant computational cost independent of the size of the rectangular area to analyze, whereas in the convolution with the unapproximated kernel the computational cost is dependent on the size of the kernel.

2.4.2.5.3 Gaussian scale space representation of an image

As mentioned in paragraph 2.4.2.5.1 using different standard deviations σ for the Gaussian kernel allows for the detection of local maxima on different scales in the image. The scale space representation of an image is a set of Gaussian filtered versions of that image with different σ. In the literature σ is often called the scale parameter t. The intuition is that structures in the image that are smaller than $\sqrt{t}$ are removed from the image.

In their paper Bay et al. (2006) use 9×9 filters which approximate a scale of $\sigma = 1.2$ as their lowest scale. Other scale levels are generated by up-scaling the filter sizes such that their structure of weights is preserved as shown in figure 2.14. This is results in filter sizes of 9×9, 15×15, 21×21 etc. Another concept of scale space is that the filters are separated in scale levels as described above and in scale octaves. Higher scale octaves use larger filters

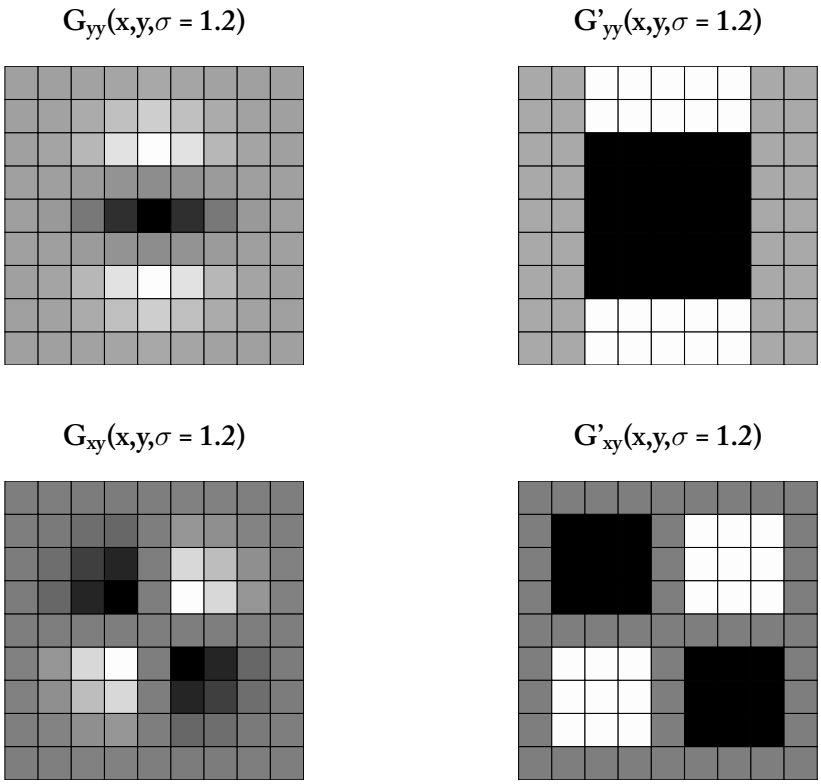


Figure 2.12: Two 9×9 smoothed second derivative filters (G_{yy} and G_{xy}) and their approximations (G'_{yy} and G'_{xy}) as used in the SURF algorithm corresponding to a standard deviation of $\sigma = 1.2$.

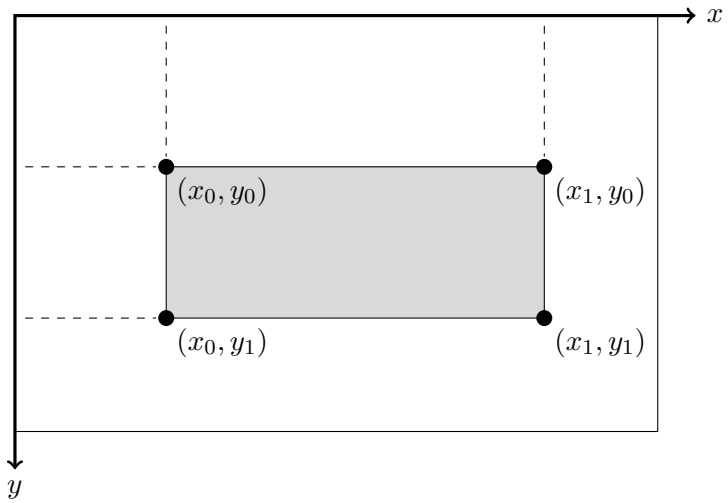


Figure 2.13: Looking up the sum of a rectangular area with an integral image. The sum of the intensity values inside the gray area can be computed using equation 2.14.

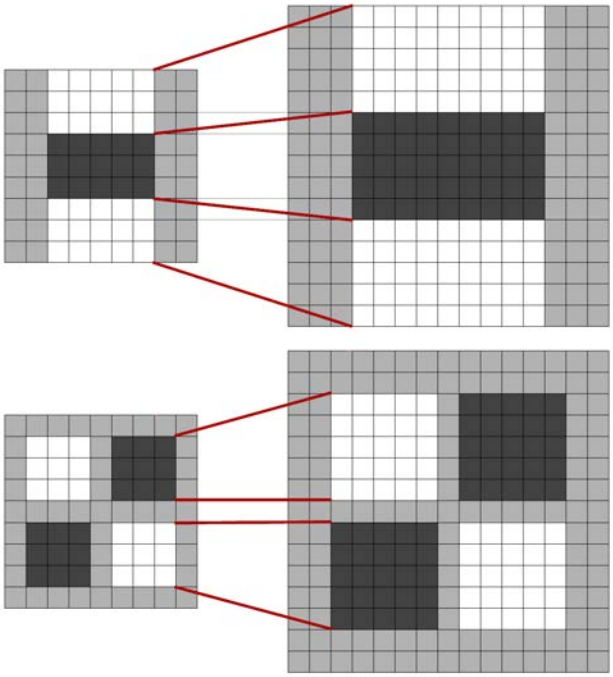


FIGURE 2.14: Filters G_{yy} (top) and G_{xy} (bottom) for two successive scale levels (9×9 and 15×15). (Bay et al., 2008)

with a larger spacing in between. The second octave for example uses double the spacing between the filters resulting in filter sizes of 15×15, 27×27, 39×39 etc. In general the filter sizes correspond to the standard deviation of a Gaussian kernel by the equation:

$$\sigma = \frac{\text{filter size}}{9} \times 1.2 \tag{2.15}$$

The results of applying the approximated and discrete kernels to the image are referred to as $D_{xx}(x, y, \sigma)$, $D_{xy}(x, y, \sigma)$ and $D_{yy}(x, y, \sigma)$.

The Matlab implementation that is used in this thesis uses 3 scale octaves and 4 scale levels per octave as default.

2.4.2.5.4 Summary of the SURF interest point extractor

The SURF interest point extractor follows these steps to extract interest points:

1. For every scale level on every scale octave:
   a) Compute the integral image.
   b) Apply the approximated box filters to the whole image using the integral image.
   c) Generate the DoH response map by computing the determinant of the approximated Hessian matrix at every point in the image:

$$det(H_{approx}(x, y, \sigma)) = D_{xx}(x, y, \sigma)D_{yy}(x, y, \sigma) - (0.9D_{xy}(x, y, \sigma))^2. \tag{2.16}$$

2. Find interest points by thresholding the DoH response maps.

3. Use non-maximal suppression[9] in a $3 \times 3 \times 3$ area centered around every interest point in the middle 2 scale levels in every octave to keep only the main local maxima.

4. Since the scale levels are quite coarse an interpolation is performed in scale and image space by using the approach from Brown and Lowe (2002, p. 255).

2.4.2.6 Frame-wise most salient peaks

The frame-wise most salient peaks (FWMSP) algorithm extracts up to 3 local maxima above a threshold per time frame. The steps to this algorithm are:

1. First the image is normalized so that all values of the image are in the range $[0, 1]$ resulting in an image I_{norm}.

2. Then local maxima are found by checking for each pixel if it is larger than the pixel above and below it.

3. Local maxima below a threshold are discarded and for every time frame only the 3 strongest maxima and their locations in the image are saved.

A good value for the threshold was found to be -10 dBFS[10].

[9] Non-maximal suppression as the name suggests suppresses all values that are not local maxima. An efficient way of doing this which is also applied in the SURF algorithm can be found in Neubeck and Van Gool (2006).

[10] dB **F**ull **S**cale, meaning a value of 1 is set as the 0 dB reference.

Evaluation 3

This chapter describes the evaluation that is carried out on the fingerprinting system from the last chapter using the modifications described in section 2.4. To evaluate a fingerprinting system an audio data set is needed. In this thesis the fingerprint system is evaluated using a duplicate-free subset of the *GTZAN Genre Collection*[1] audio data set.

The evaluation carried out in this thesis is an attempt at answering these three principal questions:

1. Which combination of input image and interest point method yields the best performance regarding the robustness of interest / stable points?
2. How robust is the fingerprinting system against common signal degradations like dynamic compression, noise, etc.?
3. How robust is the fingerprint system against time scalings and pitch shifts?

These three question give rise to several subsidiary questions such as:

- Which interest point method produces the most robust interest points?
- Is it possible to predict the displacement of interest points due to time scalings and pitch shifts?

Since the system mainly depends on robust interest points, first an analysis of the influence of the different input images and interest point methods on the number of interest and stable points is performed. This is followed by an analysis of how reliably one can predict the displacements of interest points due to time scalings and pitch shifts.

For every combination of input image and interest point method a separate dictionary is computed and used accordingly without further mentioning it in the following. The dictionaries were generated using always the same 250 audio files from the audio data set (see next section). These 250 files are later excluded from the possible queries to separate the training set from the test set. The resulting dictionaries can be found in the appendix A.

[1] Available at `http://marsyasweb.appspot.com/download/data_sets/`

All the time stretching and pitch shifting in this thesis is done by using the élastiquePro SDK[2] developed by zplane. It takes as input the ratio of time scales $TSR = \frac{\text{length of target audio}}{\text{length of original audio}}$ and the the ratio of pitch shift PSR that would result from a tempo change of the same ratio without a change in time scale. A pitch shift of one octave therefore is equivalent to a pitch shift ratio of 2 and a pitch shift ratio of $2^{\frac{1}{12}}$ would result in shift by one semitone.

3.1 The audio data set

The audio data set used in this thesis is the *GTZAN Genre Collection* introduced and used by Tzanetakis and Cook (2002). This data set consist of 1000 audio tracks, each 30 seconds long and saved in an uncompressed, mono .au format with a sampling rate of $f_s = 22050\,\text{Hz}$ and a bit depth of 16 bit. It was and still is originally used as a data set for automatic musical genre classification and to this end is divided equally into into 10 folders corresponding to 10 different genres: blues, country, hiphop, metal, reggae, classical, disco, jazz, pop & rock. This data set is chosen for several reasons:

- It covers a broad range of different musical genres, so one can minimize the possibility of training a fingerprint system to one style of music.
- All the songs are of equal length, so that it is easy to compare the numbers of extracted interest points.
- It's of manageable size.
- It's freely available and contains the raw audio material.

Apart from these positive points the data set also has its issues. Thanks to the popularity of this dataset there are two papers that investigate the shortcomings of this particular data set Sturm (2012, 2013). The only problem relevant to the task of fingerprinting are duplicates in the database which would falsely lower the recognition rate. In Sturm (2013, p. 9) one can find a table of all the duplicates in the *GTZAN* data set. Prior to any evaluations all the duplicates were taken care of by removing all but the first occurrence (in alphanumerical order) of a song from the data set which yields a clean data set of 933 items.

From this dataset 50 queries are chosen at random as a test set while taking care that none of the queries were previously used to generate the dictionaries. The choice of queries was kept constant throughout all evaluations.

3.2 Distribution of scales and types

In this section an overview over the statistical distribution of scales and types (see 2.2.4) as assigned by the employed algorithm is given. The expected result for the distribution of scales is for it to be a normal distribution so that the most common time scales are within the range of the chosen lower and upper bounds of $1 - 4$ s.

The distribution of assigned dictionary types should be somewhat uniform since anything else would indicate that some of the representative patches of the dictionaries are barely present in the signals in practice. If the latter is the case it could be because the number of clusters is too high or because of a bad training set for the k-Means algorithm.

[2] `http://licensing.zplane.de/index.php?page=description-elastique`

3.2.1 Distribution of scales and types in the fingerprint databases

3.2.1.1 Overall distribution of scales and types

Figures 3.1 and 3.1 show the overall distribution of scales and types averaged over all possible input image types and interest point methods. From this it is evident that in the overall

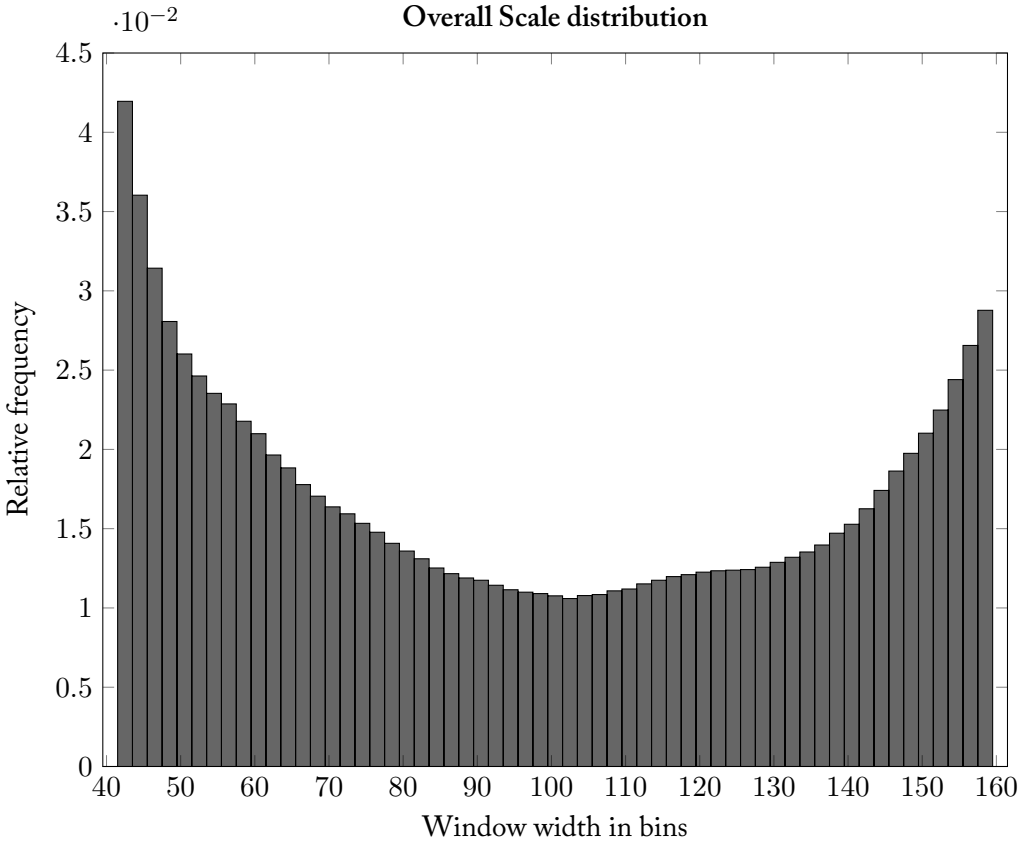


FIGURE 3.1: Distribution of scales averaged over all database items, input images and interest point methods.

distribution the types are uniformly distributed which cannot be said about the distribution of the scales; small and large scale values are much more common than the medium ones which can be a sign for a bias in the stability analysis of the algorithm. Since the time scales tend to be either very small or very large it would be interesting to increase the possible range of scale values and see where the maximum values for the distribution really are.

3.2.1.2 Distribution for all combinations of input image and interest point method

To refine the evaluation of scale and type distribution the histograms are recomputed for all the different combinations of input images and interest point methods individually averaging the data over all the database items. Figures B.1, B.2 and B.3 show the results for the scales for all possible combinations of input images and interest point methods; Figures B.4, B.5 and B.6 show the results for the assigned dictionary types.

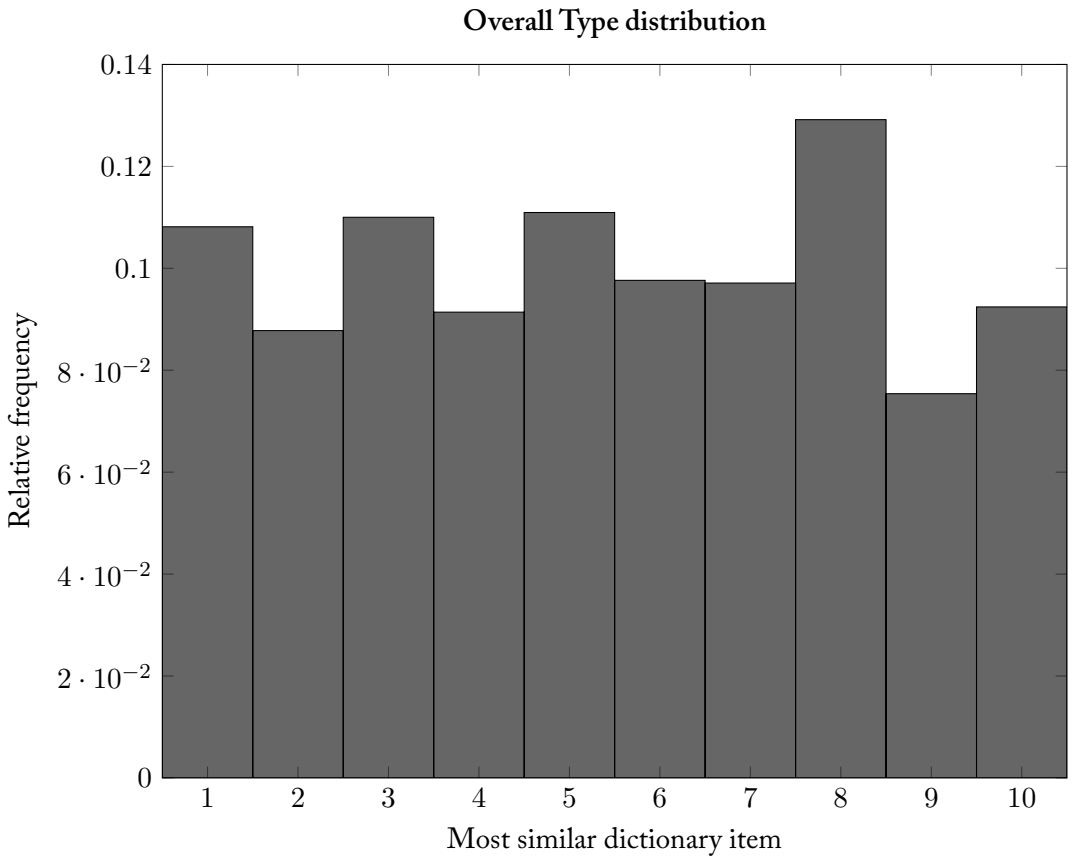


Figure 3.2: Distribution of types averaged over all input images and interest point methods.

The first thing that catches one attention when looking at the distributions (figures B.1-B.3) is that for the chroma^n input image the scale sizes are more biased towards larger scales in contrast to the other two input images which indicates that the chroma^n representation of an audio file tends to favor interest points that are stable over a longer time period.

The type distributions vary wildly depending on the input images and interest point methods, which is also evident in the dictionaries. For example: The original combination of input image and interest point method (chroma^n & original method), the types 1, 3, 5, 8 and 10 are much more likely than the other half of the types, which can be a symptom of too many clusters or a bad choice for the training set. If some dictionary types are barely used at all there is no point in having them. The type distributions that most resemble a uniform distribution are that of the Ellis interest point method.

Lowering the number of clusters can be a way to achieve more uniformly distributed dictionary types. It is also a far-reaching step to take since it would directly influence the stability analysis of the algorithm and the retrieval phase in which a sliding window voting scheme is used based on the dictionary types.

3.3 Number of Interest and stable points

In this section a statistical analysis will be given of how many interest and stable points are generated per second of audio by the six different interest point methods applied to the three different input images. This will enable us to partly answer the question which combination of input image and interest point method is best suited for the task of audio identification. The number of extracted interest points is important because extracting too many points is computationally expensive but extracting too few points will negatively influence the granularity and robustness of the system. Another important property of the extraction algorithm is the uniformity of the distribution over time and frequency. The latter is not tested here since it would require a very controlled dataset with labeled sections of mostly noise-like and mostly tonal audio signals to produce meaningful results.

The parameters of all extraction algorithms are set to produce enough interest points to yield at least 1 stable point in the further processing. This can only be done experimentally since there is no simple "interest point density" parameter for all the algorithms.

3.3.1 Number of interest points in the test set

Interest points are the points that are extracted by the 6 different algorithms applied to the 3 different input images. Every interest point detector chooses slightly different points from the images. These are then analyzed further by the stability analysis to keep only the points that are most likely to survive a signal alteration. An ideal interest point detector will detect only stable points. The worst case scenario is a detector that finds too few points or very many points that are not stable.

For this evaluation the interest points of the 50 test set items were extracted.

3.3.1.1 Results

- As can be seen in figures 3.3-3.4 the input image has a great effect on the number of interest points extracted from it for most of the interest point detectors.
- The FWMSP detector is the only one that produces roughly the same number of interest points for all 3 input images with an average interquartile range from 109 to 117 interest points per second.
- Using the HSS-CQT image produces the least number of detected interest points for most of the detectors (except the Ellis detector) which makes sense because the whole spectrogram is condensed to its fundamental frequencies and so less local maxima are spread out in the spectrogram.
- For the chroma^n the original method detects the most interest points with an interquartile range from 172 to 203 interest points per second.

3.3.2 Number of stable points in the test set

Stable points are the result of applying the stability analysis from section 2.2.4 to the interest points. The ratio of stable points to interest points is thus a measure for the quality of the interest point extraction with respect to the stability condition of Malekesmaeili and Ward (2014) (see section 2.2.4). A perfect interest point extraction algorithm will only extract

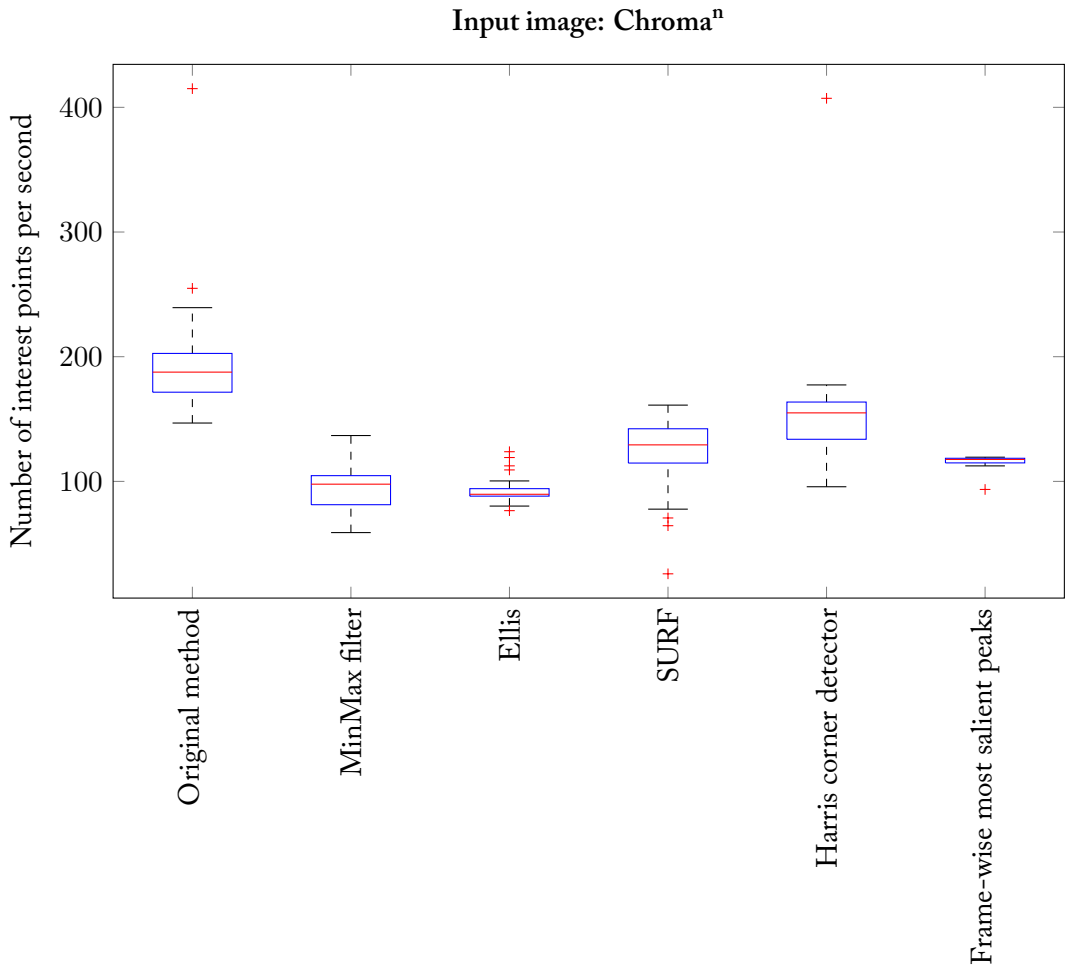


Figure 3.3: Number of interest points per second for the input image chroma^n and all interest point methods for no signal alteration.

stable interest points. Since the stability testing is rather elaborate and expensive to compute the goal is to extract as few interest points as possible which are stable while maintaining the desired minimum granularity.

3.3.2.1 Results

The results can be seen in figures 3.6-3.8 and in the tables 3.1-3.3. The tables show the quartiles of the distributions of the number of interest and stable points for the combinations of input images and interest point detectors. They also feature a ratio of stable points to interest points to give a measure of how many interest points are actually chosen as stable points. To calculate this measure the medians of the distributions are used. Additionally the ratio of interquartile ranges (IQRs) is given to see if the variance around the median

Chroma^n						
	Interest points					
	Original	MinMax	Ellis	SURF	Harris	FWMSP
1. Quartile	172	81	88	115	134	115
Median	188	98	90	129	155	118
3. Quartile	203	105	94	142	164	119
	Stable points					
	Original	MinMax	Ellis	SURF	Harris	FWMSP
1. Quartile	32	14	18	19	24	25
Median	47	21	23	28	35	32
3. Quartile	63	31	30	36	51	45
Conversion rate (Median)	0.25	0.21	0.26	0.22	0.23	0.27
Conversion rate (Interquartile range)	1.01	0.72	2.06	0.62	0.89	5.47

Table 3.1: Quartiles for the distribution of number of points per second using the chroma^n input image.

CQT						
	Interest points					
	Original	MinMax	Ellis	SURF	Harris	FWMSP
1. Quartile	126	52	92	156	128	114
Median	136	58	95	178	148	116
3. Quartile	144	65	98	193	155	118
	Stable points					
	Original	MinMax	Ellis	SURF	Harris	FWMSP
1. Quartile	17	6	10	19	11	14
Median	24	10	16	28	22	18
3. Quartile	31	13	21	40	29	27
Conversion rate (Median)	0.18	0.17	0.17	0.16	0.15	0.15
Conversion rate (Interquartile range)	0.81	0.60	1.72	0.60	0.70	3.46

Table 3.2: Quartiles for the distribution of number of points per second using the CQT input image.

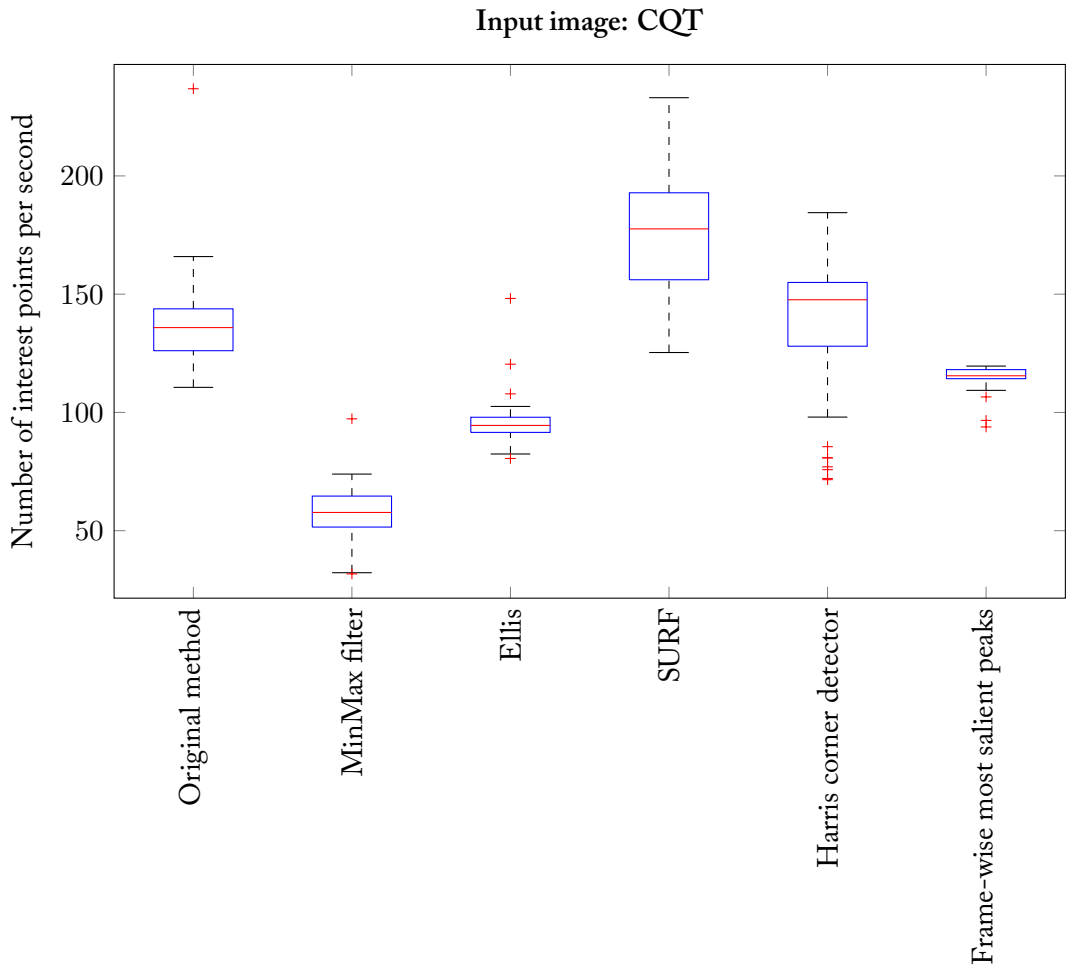


Figure 3.4: Number of interest points per second for the input image chroma^n and all interest point methods for no signal alteration.

values is changed by the stability analysis. The equation used is the following:

$$\text{Interquartile ratio} = \frac{Q_{3,\text{stable}} - Q_{3,\text{stable}}}{Q_{3,\text{interest}} - Q_{3,\text{interest}}} \tag{3.1}$$

with Q_n being the n-th quartile.

The results are:

- Compared to the number of interest points the number of stable points is lower. This is an expected behavior because the interest point detectors are not ideal.
- For the chroma^n input image the number of stable points are in a similar range (medians are ranging from $21 - 35$ stable points per second) for all interest point detectors except for the original method which has a higher median of stable points of 47.
- The ratios of the medians of stable points to the medians of interest points are roughly the same for all interest point detectors in a given input image. The original method

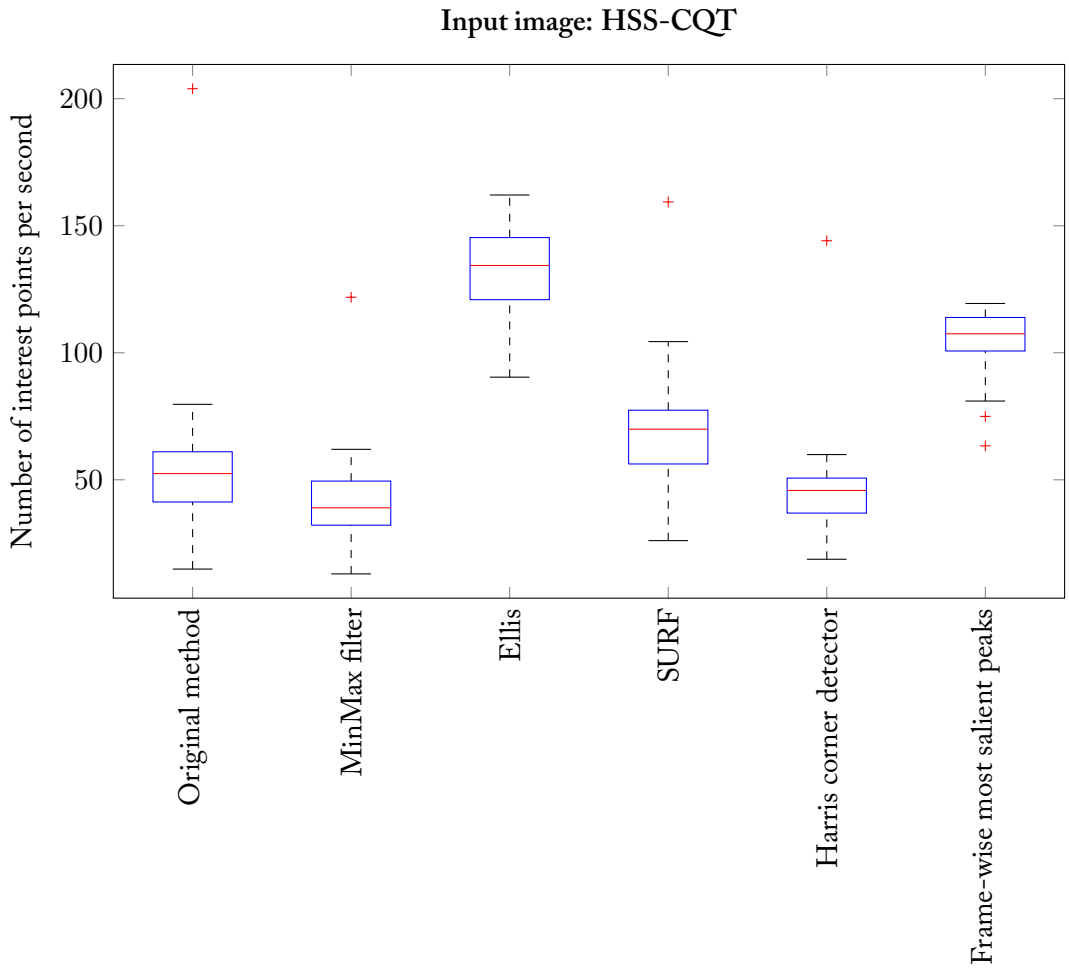


FIGURE 3.5: Number of interest points per second for the input image chroma^n and all interest point methods for no signal alteration.

has the highest conversion rates except for the chroma^n image where it lies in the middle field.

- The conversion rates for the median are the highest for the HSS-CQT input image and the lowest for the CQT input image. This can be due to the fact, that the HSS-CQT produces the least interest points to begin with. However the difference between the CQT and the chroma^n is not that large and so it can be said that using the CQT input image leads to the detection of less stable interest points.
- In most cases the IQR is decreased in the stable points compared to their interest point equivalent.

3.3.3 Results

The results from this section can be summarized as:

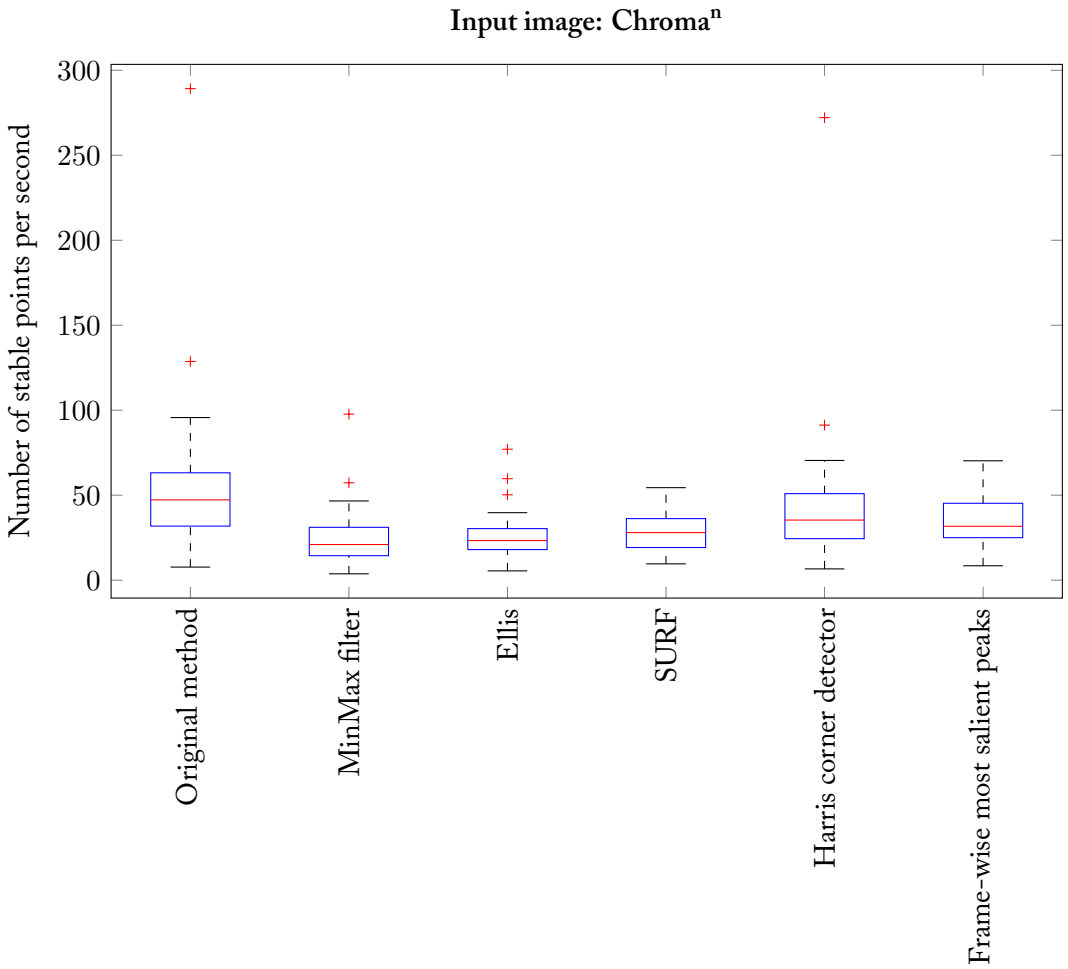


Figure 3.6: Number of stable points per second for the input image chroma^n and all interest point methods for no signal alteration.

- The chroma^n input image generates the most interest points for most of the interest point detectors and has an average median conversion rate of 24 % meaning that around this percentage of interest points are classified as stable points.

- The HSS-CQT input image generates the least interest points for most of the interest point detectors and has an average median conversion rate of 31.5 %.

- The CQT input image generates a slightly smaller amount of interest points for most of the interest point detectors as the chroma^n input image and has an average median conversion rate of 16.3 %.

- The original method of interest points detection has the highest median conversion rate for the CQT and HSS-CQT input images and a medium conversion rate for the chroma^n input image the only other detector that rivals its performance is the

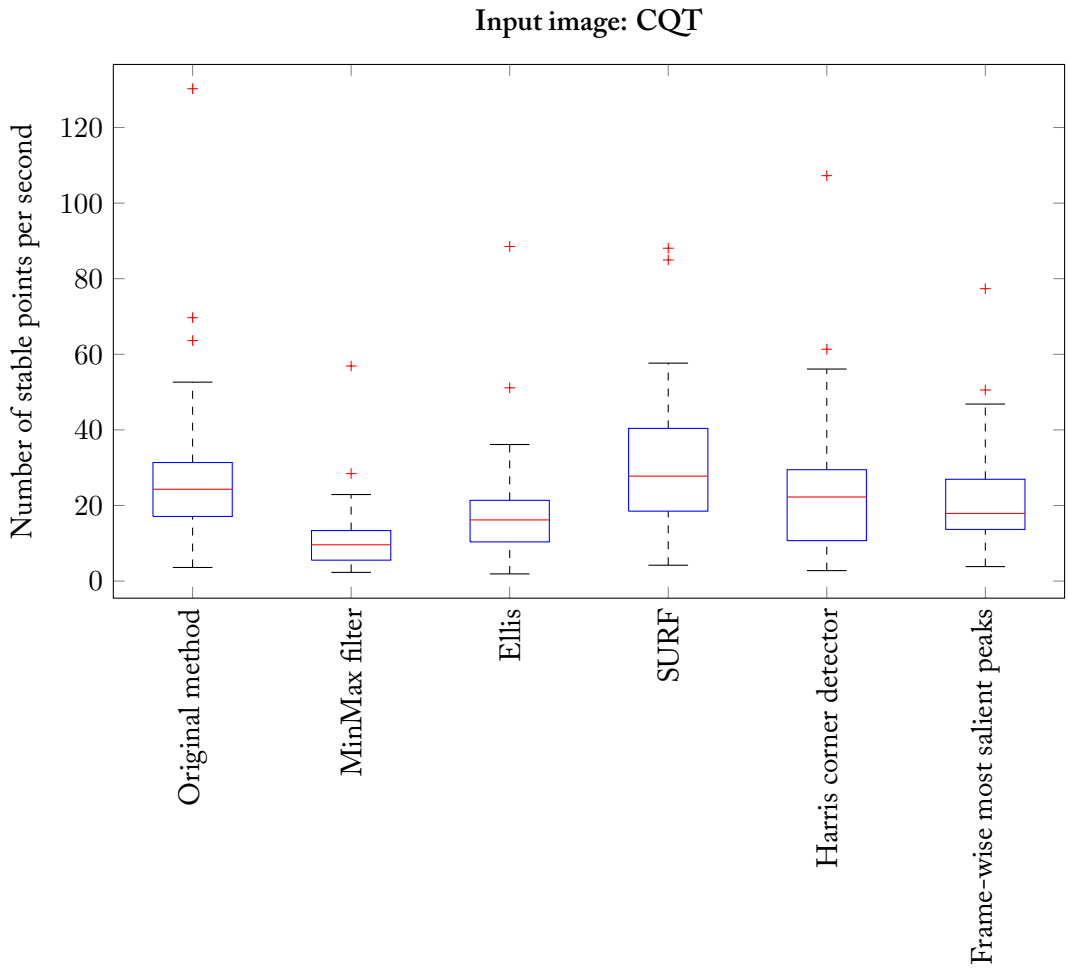


FIGURE 3.7: Number of stable points per second for the input image chroma^n and all interest point methods for no signal alteration.

Ellis detector that produces less interest points on average and has almost as good conversion rates.

3.4 Predictability of point displacements due to pitch shifting and time scaling

In the last section it was shown how many interest points are considered stable depending on the used interest point method. In this section a more thorough analysis is conducted to answer the question: Do the interest points and stable points get displaced to where they should be in theory after applying time scaling and pitch shifting to the audio?

With the time scaling ratio TSR and pitch shift ratio PSR respectively it is possible compute the theoretical displacement in the time-chroma image. Let $p = (t, f)$ be a point in the original position with time position t and frequency position f. For a time scaling

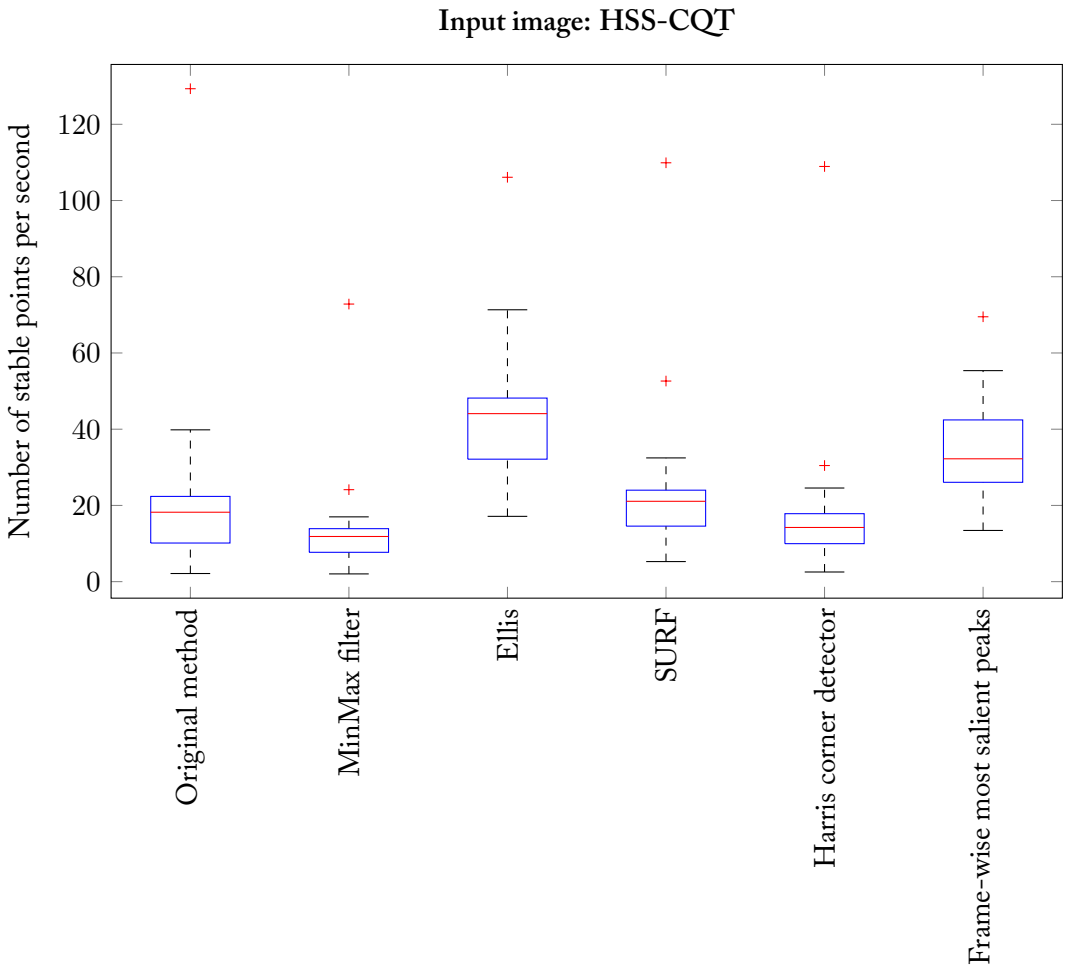


Figure 3.8: Number of stable points per second for the input image chroma^n and all interest point methods for no signal alteration.

with ratio TSR p will be displaced to the point $p' = (t \cdot TSR, f)$. For a pitch shift with ratio PSR the shifted point will be at the position $p' = (t, f + \lfloor m \cdot \log_2(PSR) + 0.5 \rfloor)$. In the case of the pitch shift all points that are displaced beyond the frequency limits have to be withdrawn from the evaluation. This way one gets a list of interest / stable points in the unaltered audio `originalPoints`, a list of predicted points for the pitch shift / time scaling ratio `predictedPoints` and a list of points that are the actual points in the altered version `actualPoints`.

From `predictedPoints` and `actualPoints` it is possible to extract the four basic measures of a binary classifier:

True positive (TP) Points that are common to `predictedPoints` and `actualPoints`.

False positive (FP) Points that are present in `actualPoints` but aren't in `predictedPoints`.

False negative (FN) Points that are present in `predictedPoints` but aren't in `actualPoints`.

HSS-CQT						
	Interest points					
	Original	MinMax	Ellis	SURF	Harris	FWMSP
1. Quartile	41	32	121	56	37	101
Median	52	39	134	70	46	107
3. Quartile	61	50	145	77	51	114
	Stable points					
	Original	MinMax	Ellis	SURF	Harris	FWMSP
1. Quartile	10	8	32	15	10	26
Median	18	12	44	21	14	32
3. Quartile	22	14	48	24	18	42
Conversion rate (Median)	0.35	0.30	0.33	0.30	0.31	0.30
Conversion rate (Interquartile range)	0.62	0.36	0.66	0.45	0.57	1.24

Table 3.3: Quartiles for the distribution of number of points per second using the HSS-CQT input image.

True negative (TN) Points that are neither in `predictedPoints` nor in `actualPoints`.

Since these are absolute measures it will be difficult to compare them between queries, different input images and interest point detectors which is why the following relative measures are used to evaluate the predictability of displacements:

True positive rate (TPR) $TPR = \frac{TP}{P} = \frac{TP}{TP+FN}$

False positive rate (FPR) $FPR = \frac{FP}{N} = \frac{FP}{TN+FP}$ Is expected to be close to 0 most of the time because of the sparsity of points in the spectrum.

False negative rate (FNR) $FNR = \frac{FN}{P} = \frac{FN}{FN+TP}$

True negative rate (TNR) $TNR = \frac{TN}{N} = \frac{TN}{TN+FP}$ Is expected to be close to 1 most of the time because of the sparsity of points in the spectrum.

F_1 Score $F_1 = 2\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}} = \frac{2\cdot TP}{2\cdot TP+FN+FP}$ Is a measure for the accuracy of the retrieval is the harmonic mean of the precision and recall. The precision is the ratio of true positives to all the points that are retrieved as positives. The recall is the ratio of true positives to all positives.

To save space in the appendix only the plots for the TPR and F_1 score are shown because as expected the FPR plots are practically 0 as well as the TNR plots are always 1 and the FNR is $1 - TPR$ and doesn't provide an information.

3.4.1 Results

The resulting plots can be found in the appendix in chapter C. The results can be summarized as:

- The highest TPR occurs for the chroma^n input image and the FWMSP detector.
- The HSS-CQT input image has the lowest TPRs and the chroma^n input image has the highest TPRs.
- The TPR for the FWMSP detector is better for larger time scalings than for larger pitch shifts.
- The FPR is always in the order of magnitude of 10^{-2} and thus doesn't provide much information just as the TNR is always above 0.97.
- The FNR is just $1 - TPR$ which also represents redundant information.

However these results have to be taken with a grain of salt, since a interest point detector that finds a lot of interest points is almost bound be have better results in it's TPR because the probability for a any random point being an interest point is higher. In other words the point that is found in the predicted position is not the displaced point but another that is in this position by pure chance. Since the chroma^n input image leads to the most interest to be extracted it is also more likely to perform better in this evaluation.

The FWMSP detector has been experimentally found to follow the trajectory of a sustained note resulting in many points for the same frequency bin right after each other in time. This explains the fact that it performs better for time scalings than for pitch shifts.

3.5 Evaluation of the Robustness against Pitch-Shifting & Time-Scaling

For this evaluation the whole test set of 50 queries is used. Every query audio file is first time scaled or pitch shifted by applying one of the ratios described in the following section 3.5.1 and then searched for in the database as described in section 2.3. The results are displayed with box plots that show the distribution of the found length of the queries in percent in the database. Because of the high number of plots all the resulting plots can be found in the appendix in chapter D.

3.5.1 Used Pitch Shifts and Time Scalings

The pitch shifts and time scalings used to evaluate the fingerprint system are:

- ± 1 %
- ± 12 %
- ± 50 %

These values are used to test …

- …very small deviations from the original query and since the pitch resolution of the input images is $2^{\frac{1}{72}} \approx 0.97$ % a value of 1 % is used.
- …the largest speed changes that common turntables are capable of, hence the 12 %. This value also corresponds to a pitch shift of 1 whole tone.

- …the limits of the algorithm by using some very large alteration.

Pitch shifts and time scalings are never applied simultaneously which results in 12 different degradations.

3.5.2 Results

The results of the retrieval are summarized in the form of boxplots which show the distribution of percentages of the queries total time found in the fingerprint database. An example is shown in figure 3.9 for the input image / interest point detector combination of the original paper by Malekesmaeili and Ward (2014). The rest of the results can be found in the appendix in chapter D. The evaluation is done separately for the coarse time estimation and

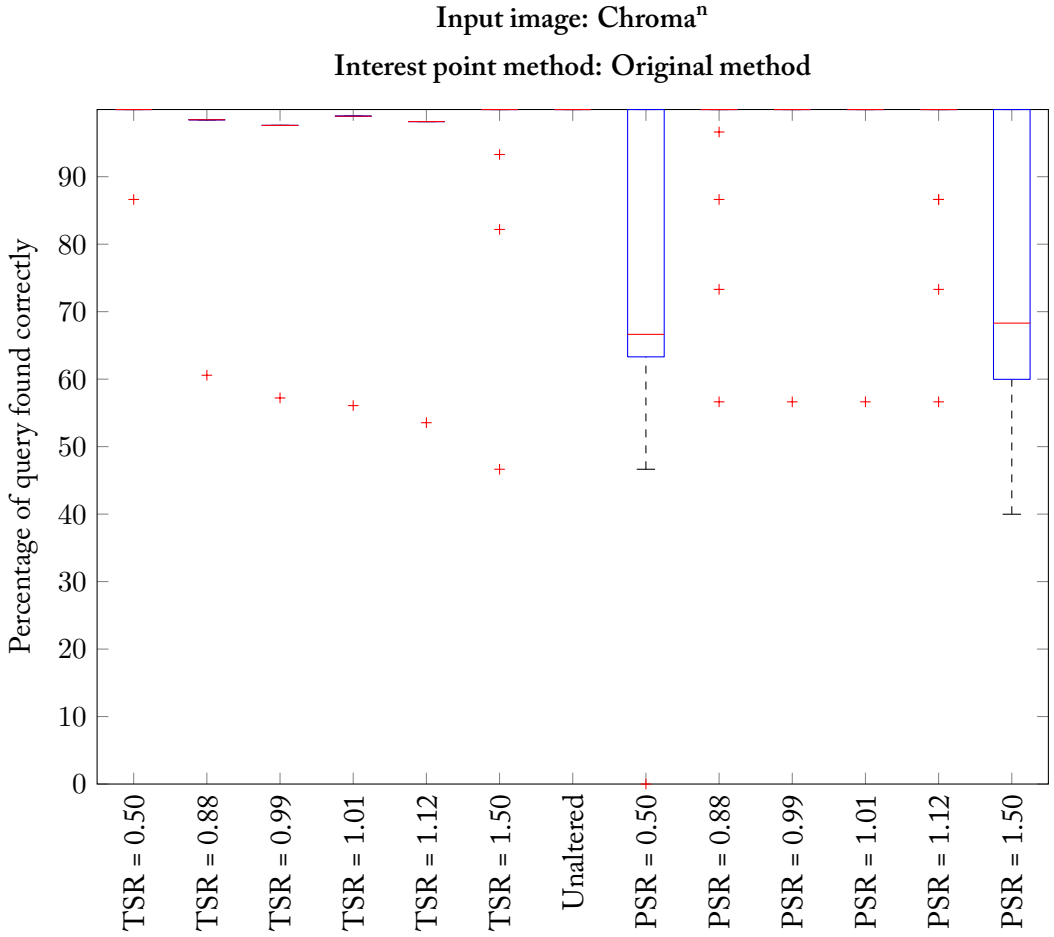


Figure 3.9: Averaged retrieval ratios for the coarse time slot estimation, input image: chroma^n and interest point method: Original method

the fine time estimation from section 2.3.

3.5.2.1 Coarse time slot estimation vs. fine time slot estimation

The results for the fine time estimation are consistently worse than for the coarse time estimation. This is rather unsurprising since the fine time estimation takes as input the coarse time estimations and in the last step reduces the found coarse time intervals to the time intervals in which the stable points are compatible with the estimated time scaling and offset. This also explains the at first sight rather confusing result that even for an unaltered query not the whole length of the query was found in the database. This is just a symptom of a sub-optimal estimation of the time scale and offset by the fine time interval estimation. The whole point of the fine estimation is to make an estimate of the time scaling and time offset between query and database item from the coarse time intervals and scale information of the stable points and then eliminating those stable points that don't fit these estimates of time scaling and time offset. This is why the time intervals that are returned by the fine estimation can only be equal or smaller than the coarse estimates. If the estimation algorithm is perfect this leads to more trustworthy estimates of the found time intervals compared to the coarse estimations. In practice however we see that the estimates are not perfect; otherwise the unaltered queries would have been found in their entire length, all of the time. Given this insight, applying the fine time estimation for audio identification seems to be of limited use – as long as you don't rely on the estimates for pitch shift, time scaling. The results from the fine time estimation are ignored for the rest of the evaluation because of the sub-optimal estimation of parameters and lack of comparability due to the unaltered queries not being found entirely.

3.5.2.2 Time scalings vs. pitch shifts

The results for the most extreme pitch shifts of ± 50 % are by far the worst for all input images and interest point methods. The equivalent time scalings are no problem for the algorithm and for the input images CQT and chroma^n have an almost perfect retrieval rate. The rest of the pitch shifts have retrieval ratios close to 100 % except for the input image HSS-CQT.

3.5.2.3 Chroma^n vs. CQT vs. HSS-CQT

From the three input images the HSS-CQT is easily identified as the worst choice for audio identification. The identification ratios are very much the same for all three input images for the time scaling ratios 0.88, 0.99, 1.01, 1.12 and for the pitch shift ratios 0.99 and 1.01. For the remaining two time scale ratios and pitch shift ratios the HSS-CQT has worse retrieval ratios.

From the remaining two input images the chroma^n is the slightly better one for almost all pitch shifts and time scalings.

3.5.2.4 Interest point detectors

The best interest point detectors in terms of highest medians and interquartile area overall is the original method used by Malekesmaeili and Ward (2014) followed by the detector used by Ellis (2009).

3.6 Robustness against standard degradations

A good fingerprinting system should be somewhat robust to common real life signal degradations. Even if the system is only used to identify audio files on a hard drive it has to be robust against the signal degradations introduced by compression such as mp3 encoding. To evaluate the robustness the *audio degradation toolbox*[3] by Mauch and Ewert (2013) for Matlab is used.

For this evaluation the 10 first queries form the test set are used. Every query audio file is first degraded by applying one of the degradations described in the following section 3.6.1 and then searched for in the database as described in section 2.3. The results are again displayed with box plots that show the distribution of the found length of the queries in percent in the database.

3.6.1 Used degradations

The audio degradation toolbox comes with six degradations that cover a wide variety of real world scenarios:

- Live recording
    1. Apply the impulse response (IR) of a large room (Great Hall).
    2. Add light pink noise.
- Radio broadcast
    1. Add dynamic range compression at a medium level to emulate the high loudness characteristic of many radio stations.
    2. Speed-up, by 2 %, which is commonly applied to music in commercial radio stations to shorten the music to create more advertisement time.
- Smartphone playback
    1. Apply the IR of a smartphone speaker (Google Nexus One), which has a high-pass characteristic and a cutoff at $f = 500$ Hz.
    2. Add light pink noise.
- Smartphone recording
    1. Apply the IR of a smartphone microphone (Google Nexus One).
    2. Add dynamic range compression, to simulate the phone's auto-gain.
    3. Add clipping to the extent that 3 % of samples are affected.
    4. Add medium pink noise.
- Strong mp3 compression
    1. Encode the audio to an mp3 file with a bitrate of 64 kbps
- Vinyl

[3] `https://code.soundsoftware.ac.uk/projects/audio-degradation-toolbox`

1. Apply a typical IR of a turntable.
2. Add some record player crackle noise.
3. Add wow-and-flutter (pitch variations of different modulation frequencies).
4. Add light pink noise.

3.6.2 Results

The results can be found in the appendix in chapter E. The evaluation for the fine time estimation technique is omitted because of it's poor performance in the evaluation of the last section.

For every combination of input image and interest point detector the degradation "strong mp3 compression" has a perfect retrieval of 100 %.

3.6.2.1 Chroma^n vs. CQT vs. HSS-CQT

The chroma^n image has the highest medians and interquartile range for the most combinations with interest point detectors and for most degradations. The worst of the input images is the HSS-CQT.

The chroma^n image has very good (first quartile > 95 %) to perfect retrieval for the degradations *Radio Broadcast*, *Strong mp3 compression* and *Vinyl*.

3.6.2.2 Problematic degradations

The two degradations that have by far the lowest retrieval rates are the *Live Recording* and the *Smartphone playback*. These two degradations are very harsh on the spectrogram since the *Smartphone playback* degradation practically deletes the lower half of the used spectrum (which spans $80 - 1280$ Hz) and the *Live Recording* degradation simulates a large hall with lots of reverberation that "smudges' spectral maxima in time.

3.6.2.3 Interest point detectors

The interest point detectors have different strengths and weaknesses for the different degradations but if a general purpose interest point detector is needed the original method by Malekesmaeili and Ward (2014) is a good choice for the CQT input image and the Harris corner detector and original method are both good choices for the chroma^n input image.

3.7 Summary of results

Before summarizing the results gained in this chapter it has to be noted that an ideal evaluation would of course have been done using more queries. However this was not possible in the time constraints of a master thesis because the computation of the fingerprints as well as the exhaustive search in the retrieval is computationally very expensive. Computing a fingerprint of a 30 s clip of audio and subsequent retrieval from the database takes takes 3.85 minutes on average on a modern PC with a quadcore featuring a 3.30 GHz quad core processor, 24 GB of random access memory (RAM) and a solid-state-drive (SSD) hard drive. This means that to test all combinations of the 3 input images, 6 interest point detectors and 13 signal alterations for the pitch shift and time scaling evaluation for just *one*

query takes 3.85 minutes $\times$ 3 $\times$ 6 $\times$ 13 $\approx$ 900 minutes $=$ 15 hours to complete. The two most inefficient steps in the algorithm are the stability analysis of interest points and the retrieval which is basically one large matrix multiplication both take about half the . The stability analysis is a bottle neck for the performance because for every interest point it extracts 60 patches around it with different time widths and then computes a two dimensional DCT for every patch. Considering that the average median of number of interest points is 109 $\frac{\text{interest points}}{\text{s}}$ and all the audio in the audio data set is 30 s long this amounts to 30 s $\times$ 109 $\frac{\text{interest points}}{s}$ $\times$ 60 time widths $=$ 196,200 patches to which a 2D DCT is applied per query. The retrieval is prohibitively expensive, especially for larger databases. It involves multiplying the whole database of fingerprints with the query fingerprints and sorting the whole result to find the most similar items. Taking the example of this thesis the audio data set is 933 items large, each 30 s long and the average median of number of fingerprints per second of audio is 25 $\frac{\text{fingerprints}}{\text{s}}$ each fingerprint being a vector of length 143. Every fingerprint is stored as an array of 64 bit floating point numbers which amounts to a file size of 64 bits $\times$ 143 $=$ 9,125 bits for one fingerprint. This allows for an estimate of the approximate database size: 9,125 bits $\times$ 25 $\frac{\text{fingerprints}}{\text{s}}$ $\times$ 30 s $\times$ 933 $=$ 798 MB. Multiplying these amounts of data requires a lot of RAM in the system and can prove nearly impossible to achieve if the size of the fingerprint database is so large that it cannot be saved in its entirety in the RAM because then the computer is forced to use the much slower hard drive to swap files in and out of the RAM.

That being said the following points summarize the overall results of the evaluations:

1. The distribution of scales shows a bias towards shorter and longer time scales. This means that probably some of the stable points are assigned a wrong scale and would be assigned a short or longer scale if possible. This has consequences for the performance of the estimation of the time scale ratio (TSR) performed by the fine time estimation in the retrieval.

2. The distribution of assigned dictionary types shows that some dictionary items are rarely assigned to stable points which is a sign for a too high number of clusters in the k-Means clustering algorithm or a bad initialization of centroids that results in a sub-optimal partition of the vector space. The latter is not very probable because the k-Means++ initialization is applied which minimizes the chances of this happening. Fewer clusters or a whole different clustering algorithm will directly influence the stability analysis because it is based on computing the the correlation of every one of the 60 patches of different time widths to every dictionary item and assigning every patch the dictionary item with the highest correlation. If most of the extracted patches have the same dictionary type assigned to them, the point is considered stable.

3. The HSS-CQT has the lowest number of detected interest points for most of the detectors and has the highest conversion ratio of stable points to interest points (around 31.5 %). This a a desirable property because it reduces the number of points that are used for expensive the stability analysis and the found interest points have higher probability of being a stable point.

4. The behavior of the interest point detectors is influenced heavily by the input images. For example the original method detector produces a median of 188 $\frac{\text{interest points}}{\text{s}}$ for

the chroma^n input image and a median of 136 $\frac{\text{interest points}}{\text{s}}$ for the CQT but drops down to 52 $\frac{\text{interest points}}{\text{s}}$ for the HSS-CQT.

5. The predictability of the displacements due to pitch shifting and time scaling is the highest for the FWMSP interest point detector for all input images but especially high for the chroma^n input image. Although it has to be noted that this evaluation is error-prone and tends to favor interest point detectors that produce a lot of points. The MinMax filter detector and the original method detector are two detectors with a high TPR.

6. The fine time estimation is far from perfect and in some cases has problems finding unaltered queries in the database which is why it is not suited for audio identification.

7. The combination of the chroma^n input image and the original method of interest point detection is the most robust against time scalings and pitch shifts and works very well for time scalings with an average median found time ratio of 0.988 and almost perfectly for pitch shifts except in the cases of the most extreme pitch shifts ± 50 %.

8. The *strong mp3 compression* degradation is retrieved correctly all the time with all input images and interest point detectors.

9. The *radio broadcast degradation* is almost always retrieved correctly with similar medians of around 0.986 for all combinations of input images and interest point methods.

10. For most combinations of input image and interest point detector the live recording and smartphone playback degradations have the worst retrieval rates.

11. The retrieval rates for all other standard degradations vary depending on the input images and interest point detectors and it is difficult to name one single best combination for all degradations. This being said, there are three combinations using the chroma^n input image that give reasonable results for most of the degradations. These are using the interest point detectors: Harris corner detector and the MinMax filter. Both have perfect or near perfect retrieval for the degradations *radio broadcast degradation*, *smartphone recording*, *strong mp3 compression* and *vinyl* and only differ for the degradations *live recording* and *smartphone playback*. The first having the best score for the *live recording* degradation but mediocre results for the *smartphone playback*. The MinMax filter detector has the opposite qualities.

Conclusion 4

In this thesis the robustness of the fingerprinting system by Malekesmaeili and Ward (2014) against time scaling, pitch shifting and the standard degradations from the audio degradation toolbox by Mauch and Ewert (2013) was tested. It was also tested if using a different type of spectral representation (called input image in this thesis) and using different interest point extractors influence the performance of the retrieval (see section 2.4 for a summary of all input images and interest point detectors). To this end all different combinations of input images and interest point detectors were evaluated with respect to the number of produced interest and stable points and the predictability of displacement of these points due to pitch shifts and time scalings were evaluated.

The results of the evaluation show that the HSS-CQT is the input image that produces the least interest points has the highest rate of interest point that are stable to total found interest points. This sounds like a good combination for a good basis for the further processing that is done to compute a fingerprint. Yet the retrieval results for pitch shifts and time scalings as well as for the standard degradations are the worst for the HSS-CQT input image. Looking at the predictability of displacements, the HSS-CQT is also the worst of all three input images.

The chroma^n input image which is originally used by the employed algorithm leads to the highest number of detected interest points and the highest predictability of point displacements. It also ranks highest for the retrieval of pitch shifted and time scaled queries. In the end the original combination of input image and interest point detector by Malekesmaeili and Ward (2014) is the best with respect to time scalings and pitch shifts and very decent for the standard degradations. For the standard degradations only the combinations of chroma^n & Harris corner detector and chroma^n & MinMaxFilter are better than the original method detector.

The CQT lies somewhere between the two in all evaluations and is thus not further discusses here.

One reason for the confusingly bad performance of the HSS-CQT seems to be that is produces too few points. The reason for using the HSS-CQT was to simplify the spectrum and create more salient peaks by summing up the harmonics of all fundamentals. This only makes sense if the interest point detectors are tuned for exactly these kinds of features that

the HSS-CQT amplifies. Apparently the HSS-CQT deletes more of the relevant interest points from the spectrum than it gains by summing up the harmonics to the fundamentals.

4.0.1 Strengths

The algorithm used by Malekesmaeili and Ward (2014) is indeed very robust against large time scale ratios and pitch shifts (see figure D.2) and also against most of the standard degradations from Mauch and Ewert (2013) (see figure E.14).

4.0.2 Limitations

The biggest drawback of the employed algorithm by Malekesmaeili and Ward (2014) is that it is computationally very expensive and takes almost 4 minutes to compute a fingerprint and retrieve it from a database of 933 items on a modern PC. About half of the time is spent on computing the fingerprint (checking the stability of interest points in particular) and the other half is spent on searching the fingerprint.

The stability analysis in the computation of the fingerprints (see section 2.2.4) is slow because as I calculate in section 3.7 the algorithm has to compute on average 6,540 2D-DCTs per second. This number can be lowered by either extracting less interest points at the expense of the robustness and granularity of the retrieval or by taking less different time widths for the analysis at the expense of a proper estimate of the time scale. Since the assigned time scales seem to be distributed inefficiently anyway my suggestion would be to lower the number of different time widths and at the same time expand the range of possible time widths. Of course nothing stops us from writing a completely new stability analysis which main goal is to output a time scale and discard unrobust interest points at a reasonable computational cost.

The search is performed in a brute force approach by multiplying all the queries fingerprints with all the fingerprints in the database then applying the $arccos$ to the results to get the angle between the multiplied fingerprint vectors and then take for every query fingerprint the database fingerprint with the smallest angle as a match. This is very inefficient and is not practical for larger databases that don't fit into the RAM of the system used. Fast search strategies that are based on the angular distance between vectors exist such as the LSH methods *random projection* described in Andoni and Indyk (2008) and Charikar (2002) and the *spherical LSH* described in Terasawa and Tanaka (2007). LSH is an umbrella term for techniques that reduce the dimensionality of some input data and at the same time clustering it by mapping similar items to the same hashes – which are often called"buckets". This way to find an item one only has to check the items in the same bucket rather than all items. The *random projections* LSH technique approximates the cosine similarity when hashing. It works by generating a number k of random hyperplanes of dimension $n - 1$ with n being the dimension of the data to hash. The hash of a data point is then constructed by initializing a bit string of length k and going through every hyperplane and checking if the point is on the positive side of the hyperplane or the negative one (by multiplying the data vector with the normal vector of the hyperplane) and depending on which side it is setting the bit value to 1 or 0. The *spherical LSH* is specifically designed for data that is on a unit sphere such as the fingerprints that are used in this thesis. It partitions the area on the surface with Voronoi cells and also preserves angular distance. Using one of these techniques could decrease search times significantly and could also be used to analyze the

statistical properties of the fingerprints by looking at how many fingerprints are in the individual buckets. A thorough survey of some of the many different LSH methods can be found in Wang et al. (2014).

The fine time slot estimation for found songs (see section 2.3) has proven to be rather unreliable since it is not able find an unaltered query in its entirety in the database which is unfortunate for an audio identification system.

4.0.3 Outlook & future research

Getting the algorithm to perform more efficiently is paramount to all further research using it. Using a system that slow is very hard to maintain since mistakes in the process are very time consuming to debug and it hinders spontaneous experimentation. The most difficult part in this will be speeding up the stability analysis since the requirement is to reliably associate a time scale with every stable point. The term stability can be interpreted differently and leaves room for different definitions of what a stable local maximum in the spectrum looks like. Implementing a faster search should be a straightforward task since fast search algorithms for this type of data exist and are well documented in the literature as discussed in the previous section.

Since the system used in this thesis is based on an expanded chroma spectrum it is slightly more tonality-centered than the FFT. It is possible that this fingerprinting system performs worse for mainly percussive and noise-like music like techno which is to be tested in a further investigation of the quality of the system.

The interest point detectors taken from the field of computer vision (Harris corner detector & SURF) didn't improve the retrieval rates despite adding a lot more computational complexity compared to the very simple original method detector by Malekesmaeili and Ward (2014). That's no reason to give up on using detectors from this field because there are a lot of different detectors that can be tested for the task of audio identification. Algorithms like SIFT and SURF have been successfully used to identify matching photographs and for panorama stitching which begs the question why this shouldn't work for spectra in a modified way or another. Some computer vision interest point detectors also associate a scale with a found interest point or blob[1] which could render the inefficient stability analysis obsolete.

If one was to design a new fingerprint system it is important to ensure the scalability of the system because a fingerprinting system is only worth as much as the database it operates with. The Shazam fingerprinting system features a database of over 28 million songs which grew steadily over the last years. With databases that large one has to take great measures to keep search times at a minimum while preserving the accuracy of the retrieval. This requires knowledge of state-of-the-art data mining techniques such as MinHashing and is an exciting field of study.

[1]The Wikipedia page for "Blob detection" (`https://www.wikiwand.com/en/Blob_detection`) offers the following (informal) definition of a Blob: "Informally, a blob is a region of an image in which some properties are constant or approximately constant; all the points in a blob can be considered in some sense to be similar to each other."

Dictionaries A

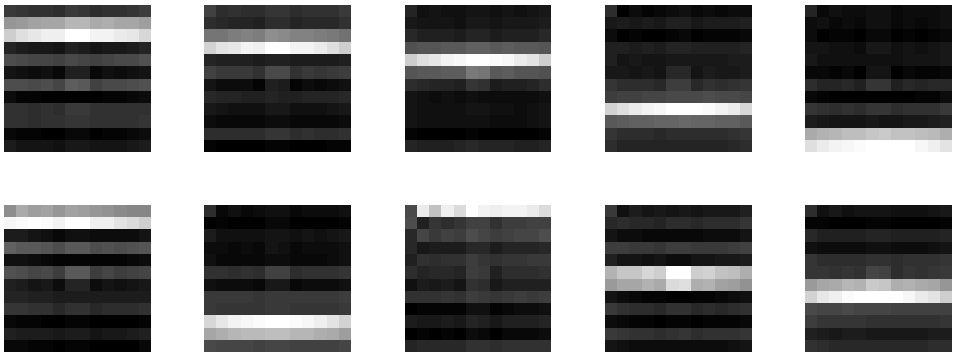


Figure A.1: Dictionary for the chroma type: CQT and the interest point method: Harris corner detector

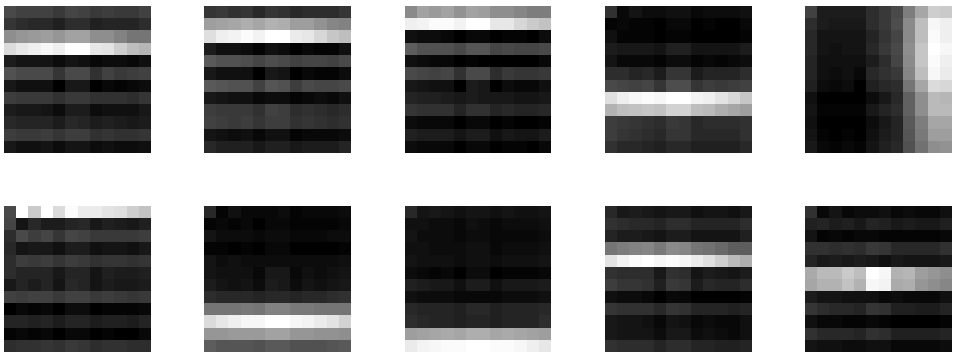


Figure A.2: Dictionary for the chroma type: CQT and the interest point method: Original Method

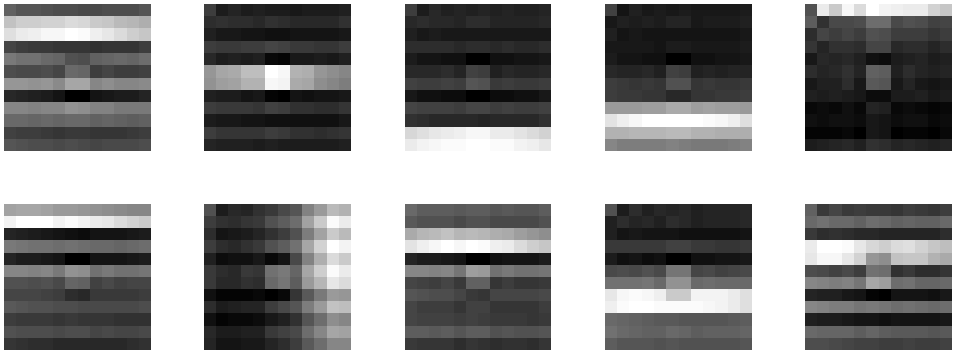


Figure A.3: Dictionary for the chroma type: CQT and the interest point method: MinMax filter

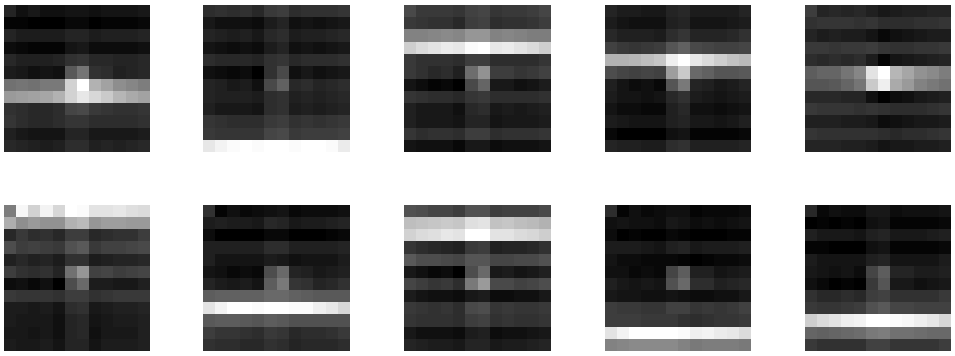


Figure A.4: Dictionary for the chroma type: CQT and the interest point method: Ellis

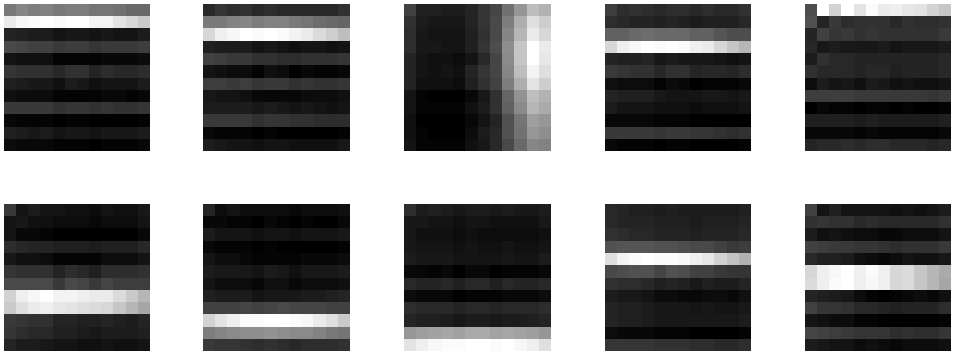


Figure A.5: Dictionary for the chroma type: CQT and the interest point method: SURF

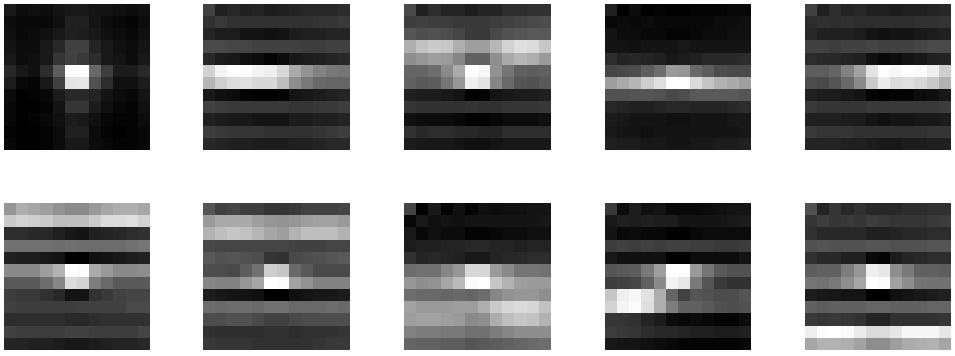


Figure A.6: Dictionary for the chroma type: CQT and the interest point method: Frame-wise most salient peaks

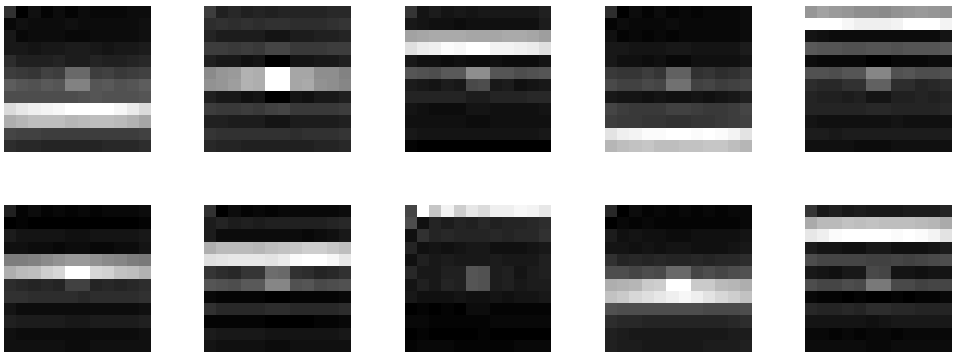


Figure A.7: Dictionary for the chroma type: HSS-CQT and the interest point method: Harris corner detector

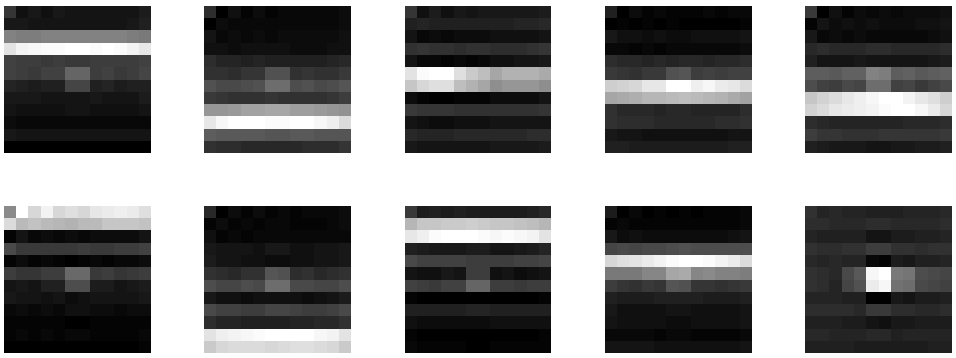


Figure A.8: Dictionary for the chroma type: HSS-CQT and the interest point method: Original Method

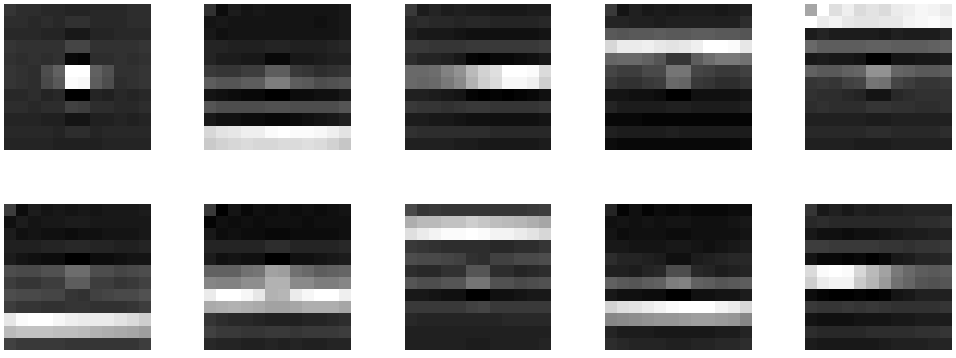


Figure A.9: Dictionary for the chroma type: HSS-CQT and the interest point method: MinMax filter

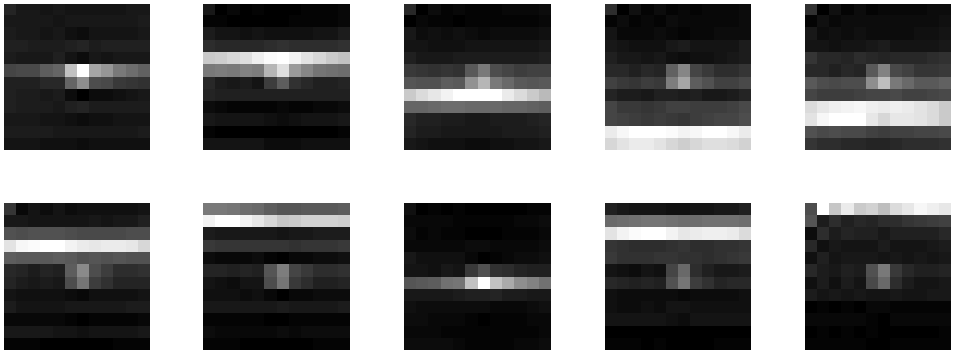


Figure A.10: Dictionary for the chroma type: HSS-CQT and the interest point method: Ellis

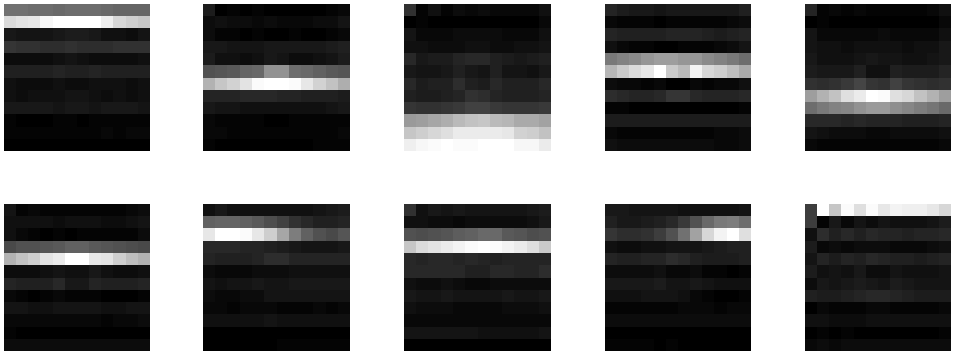


Figure A.11: Dictionary for the chroma type: HSS-CQT and the interest point method: SURF

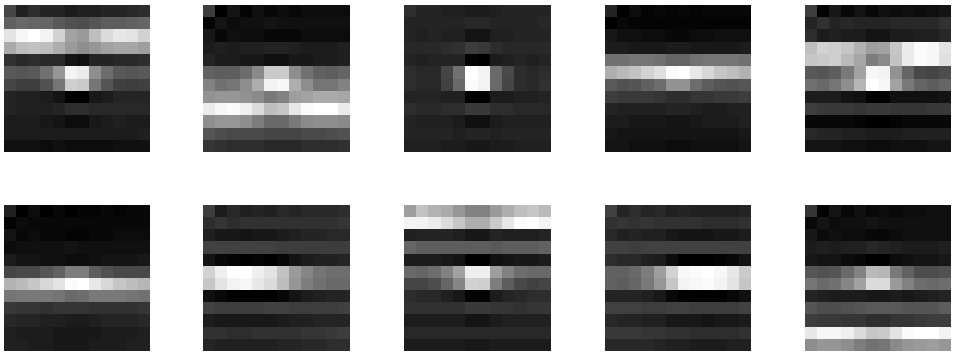


Figure A.12: Dictionary for the chroma type: HSS-CQT and the interest point method: Frame-wise most salient peaks

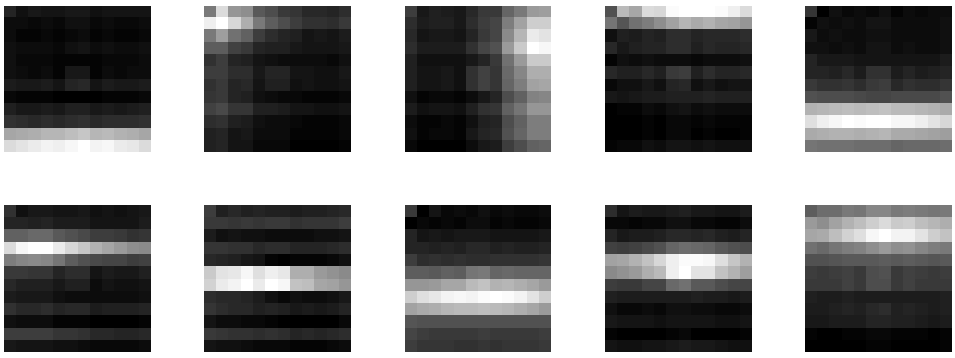


Figure A.13: Dictionary for the chroma type: Original Method and the interest point method: Harris corner detector

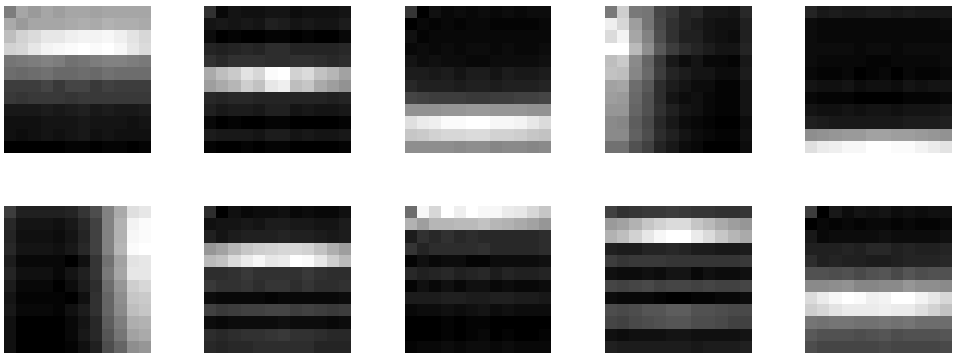


Figure A.14: Dictionary for the chroma type: Original Method and the interest point method: Original Method

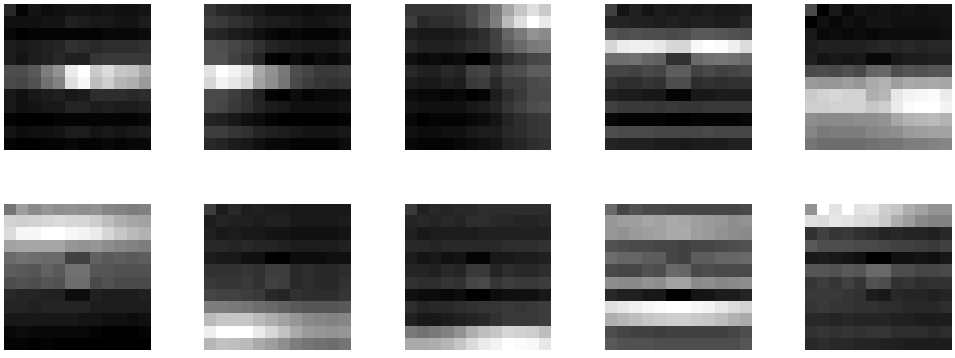


Figure A.15: Dictionary for the chroma type: Original Method and the interest point method: MinMax filter

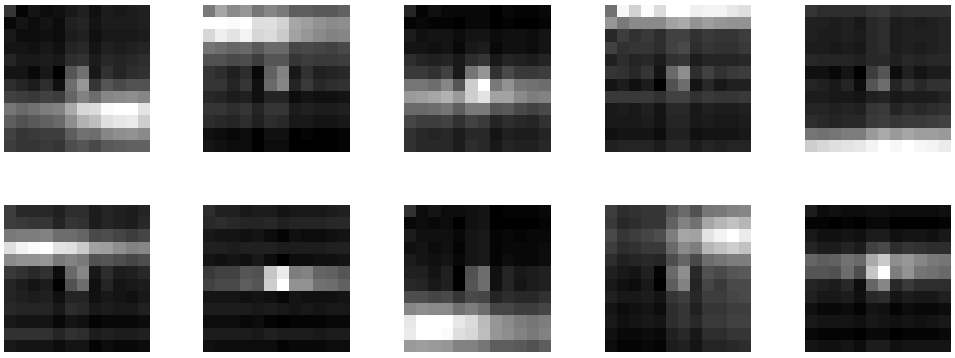


Figure A.16: Dictionary for the chroma type: Original Method and the interest point method: Ellis

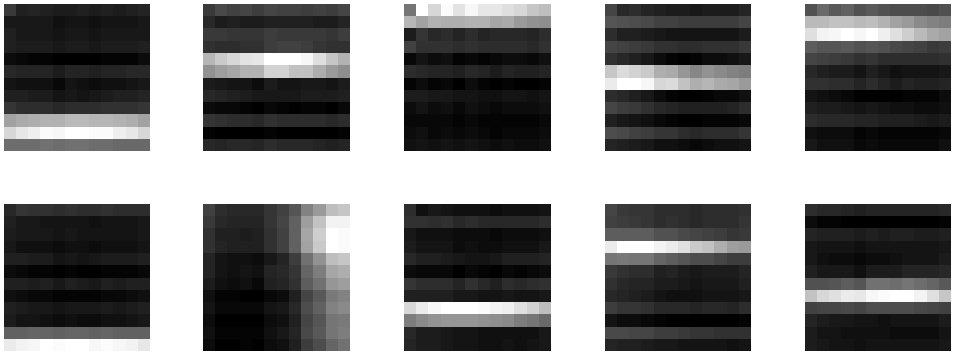


Figure A.17: Dictionary for the chroma type: Original Method and the interest point method: SURF

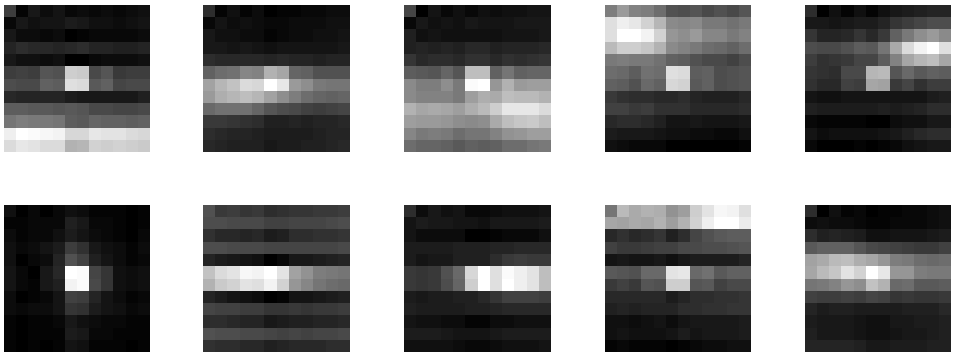


Figure A.18: Dictionary for the chroma type: Original Method and the interest point method: Frame-wise most salient peaks

Scale and dictionary type distributions in the fingerprint database B

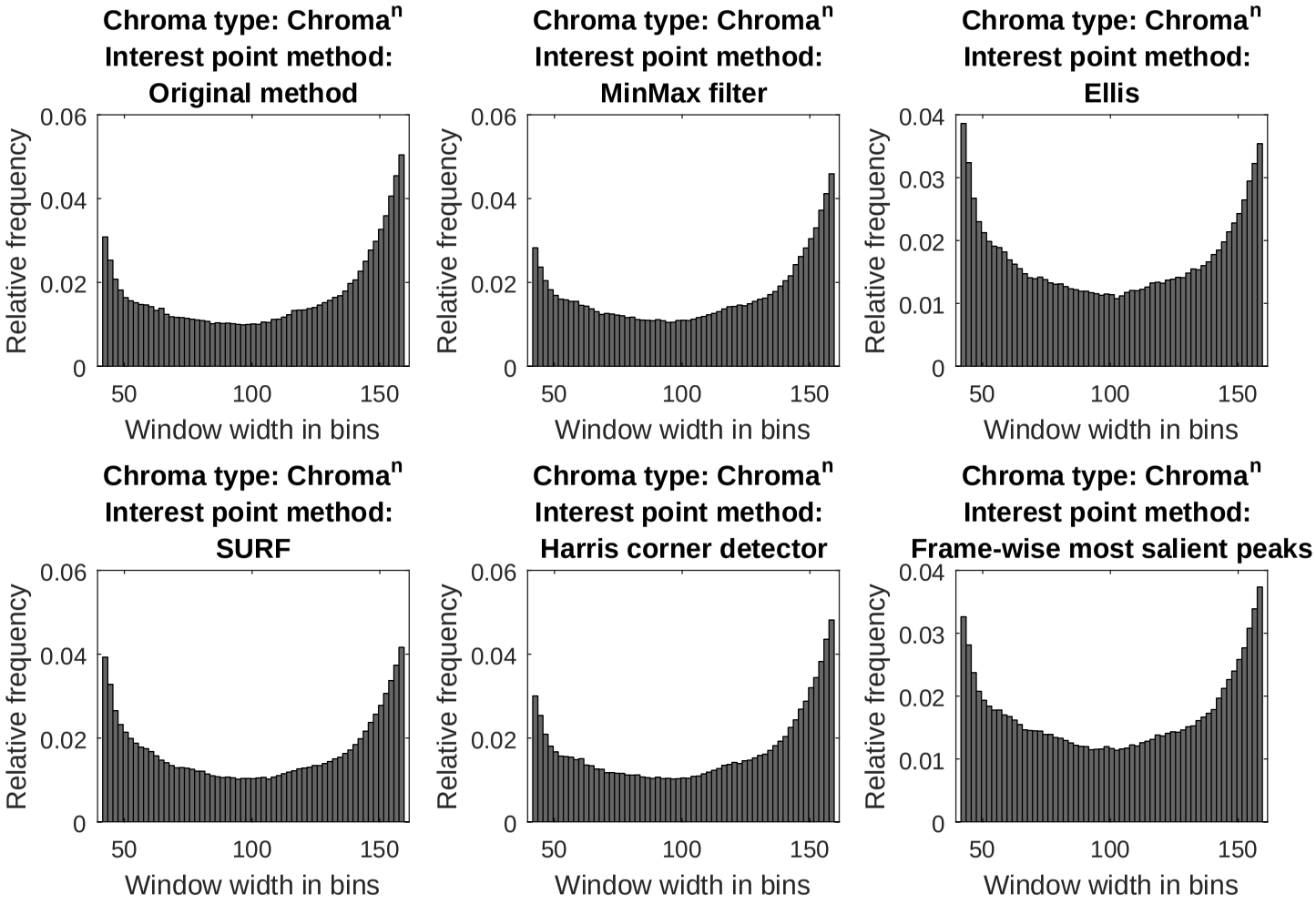


Figure B.1: Distribution of scales averaged over all database items, separately plotted for all interest point methods and input image chroma^n.

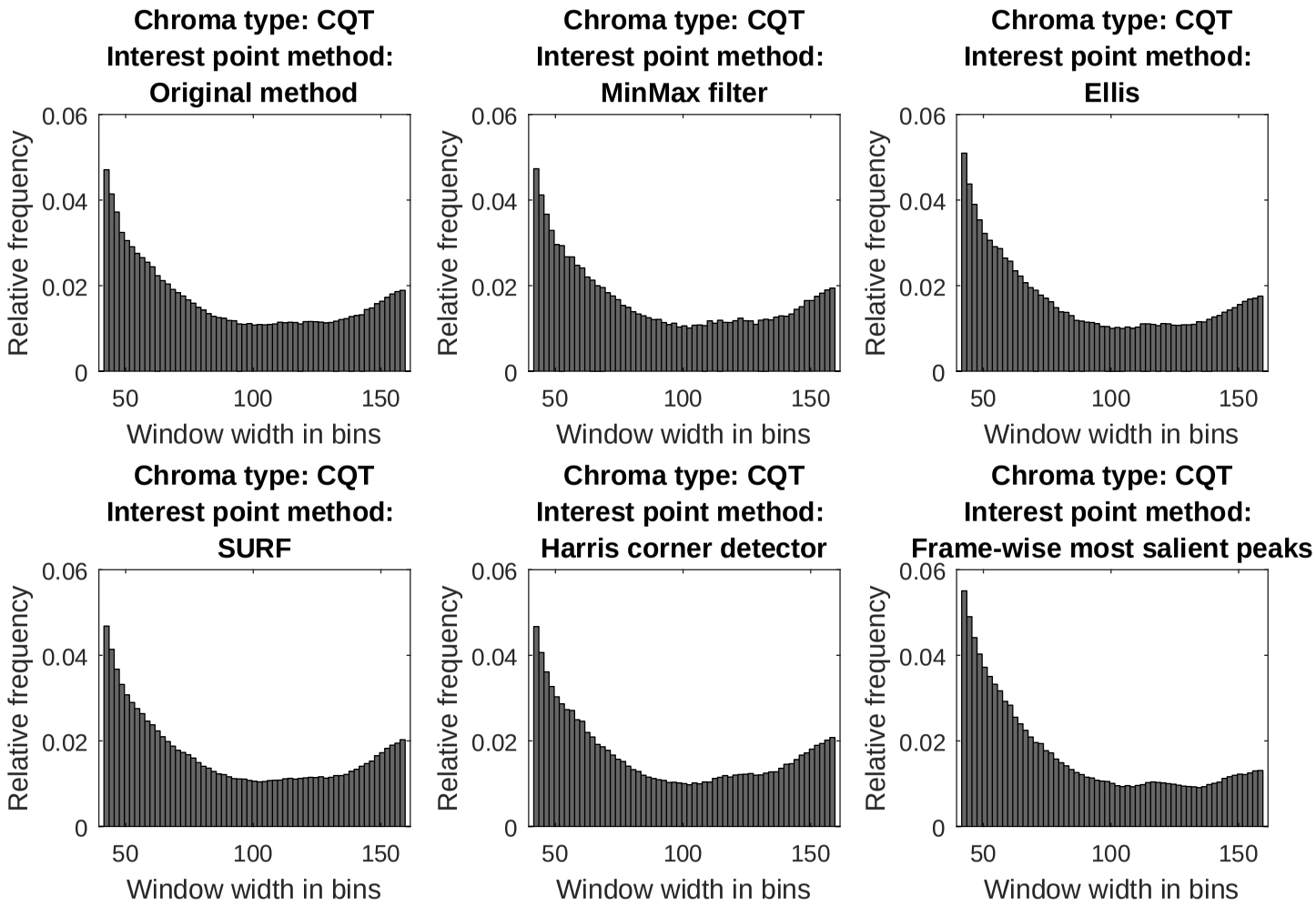


FIGURE B.2: Distribution of scales averaged over all database items, separately plotted for all interest point methods and input image CQT.

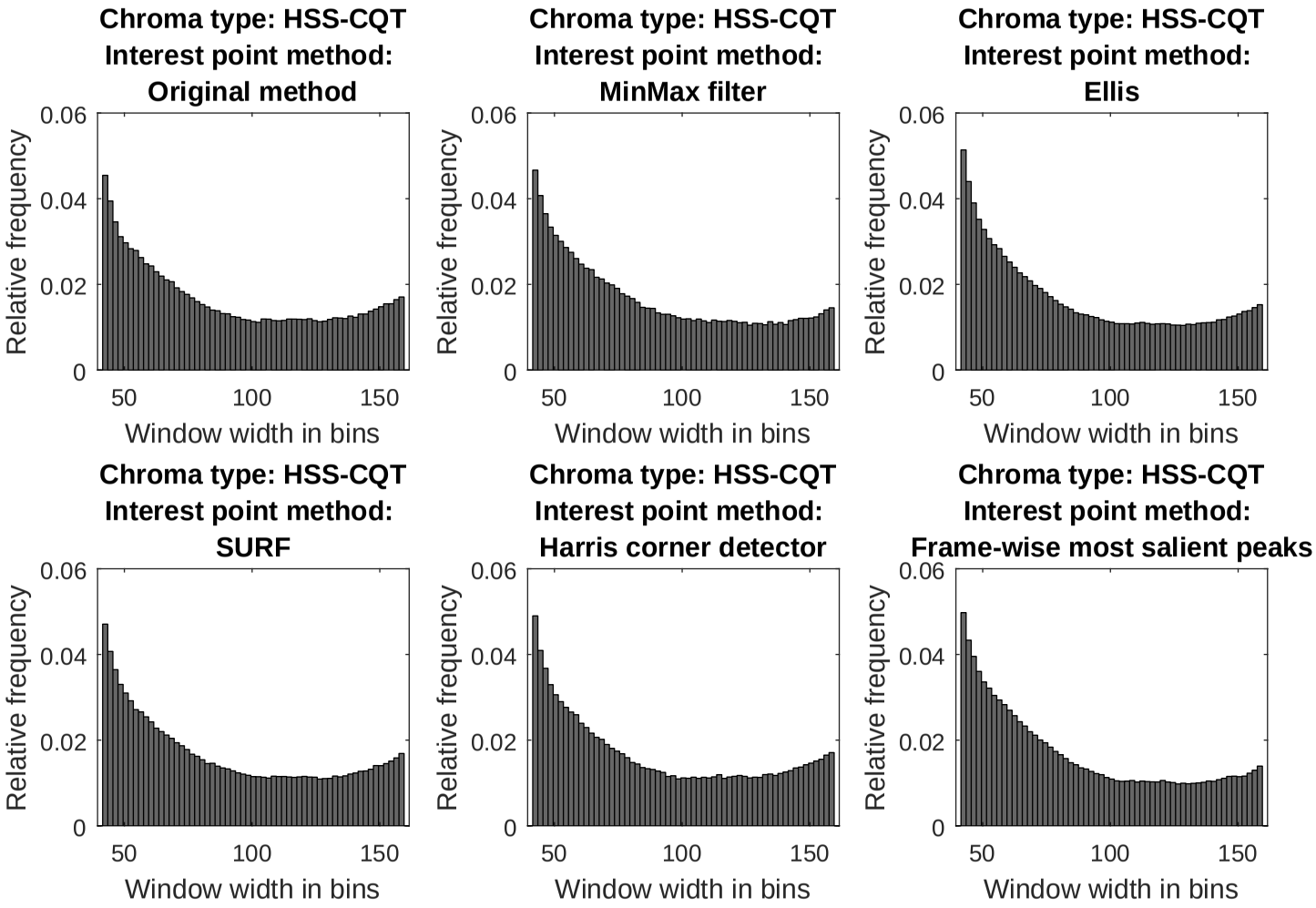


Figure B.3: Distribution of scales averaged over all database items, separately plotted for all interest point methods and input image HSS-CQT.

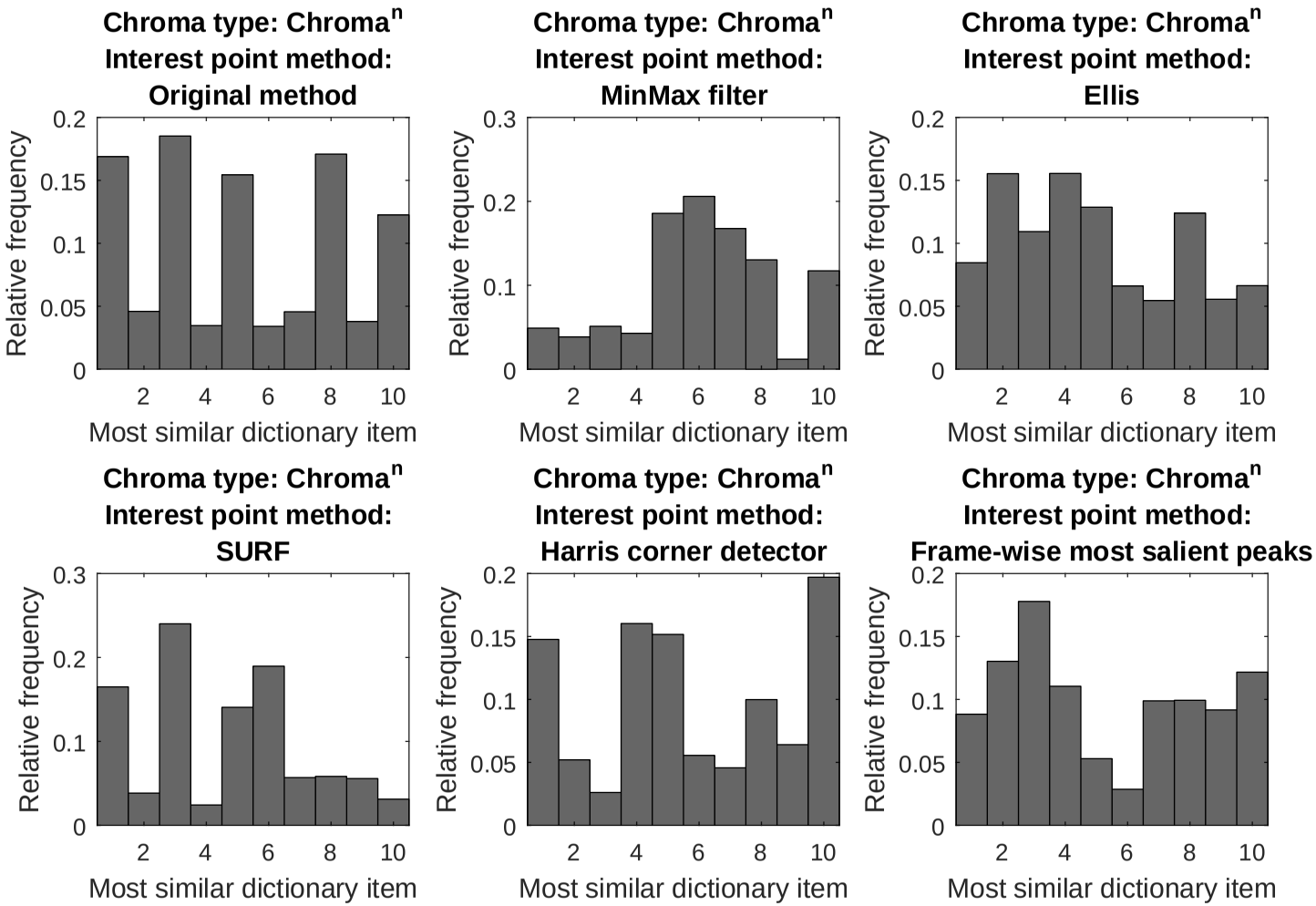


FIGURE B.4: Distribution of types averaged over all database items, separately plotted for all interest point methods and input image chroma^n.

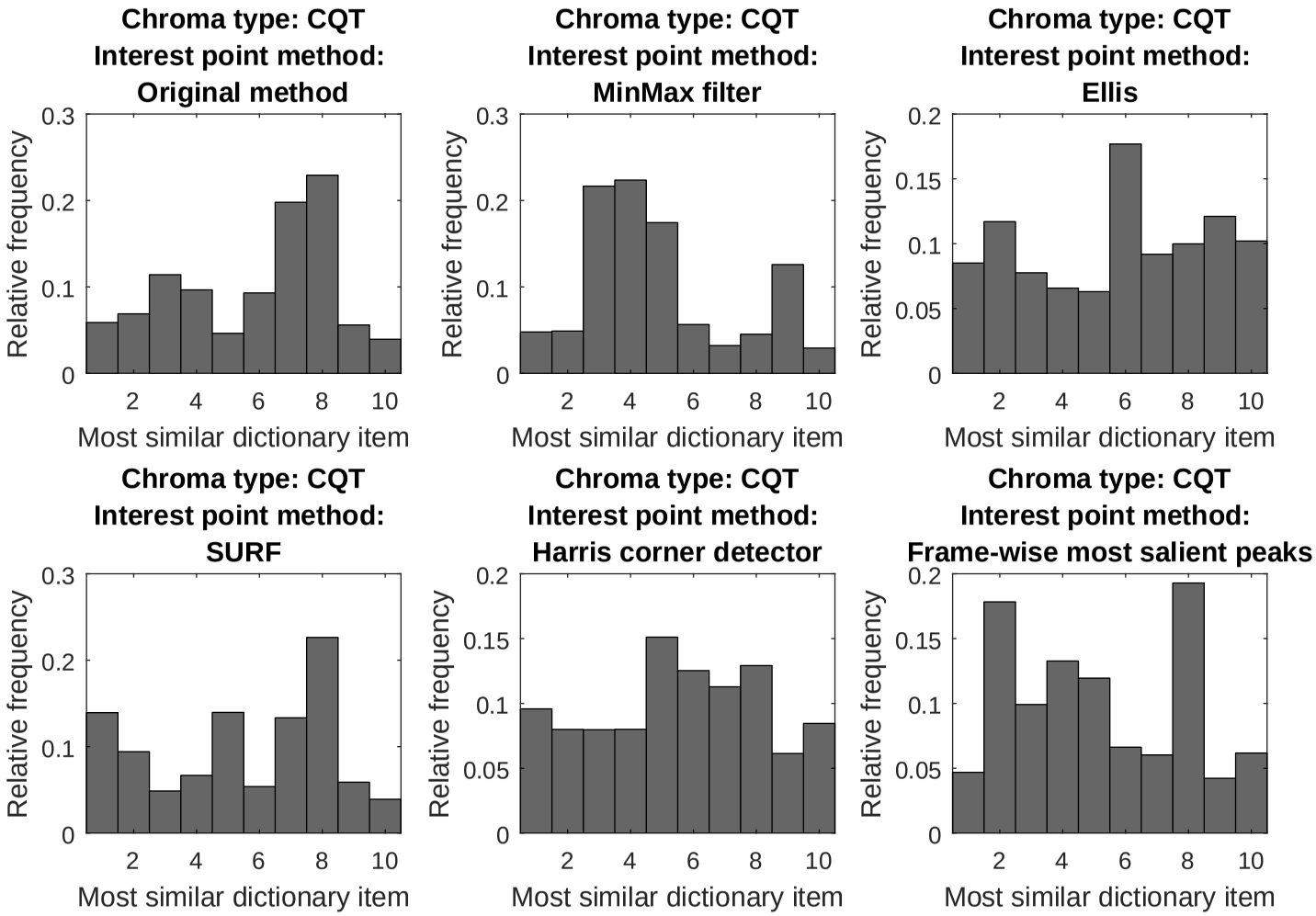


Figure B.5: Distribution of types averaged over all database items, separately plotted for all interest point methods and input image CQT.

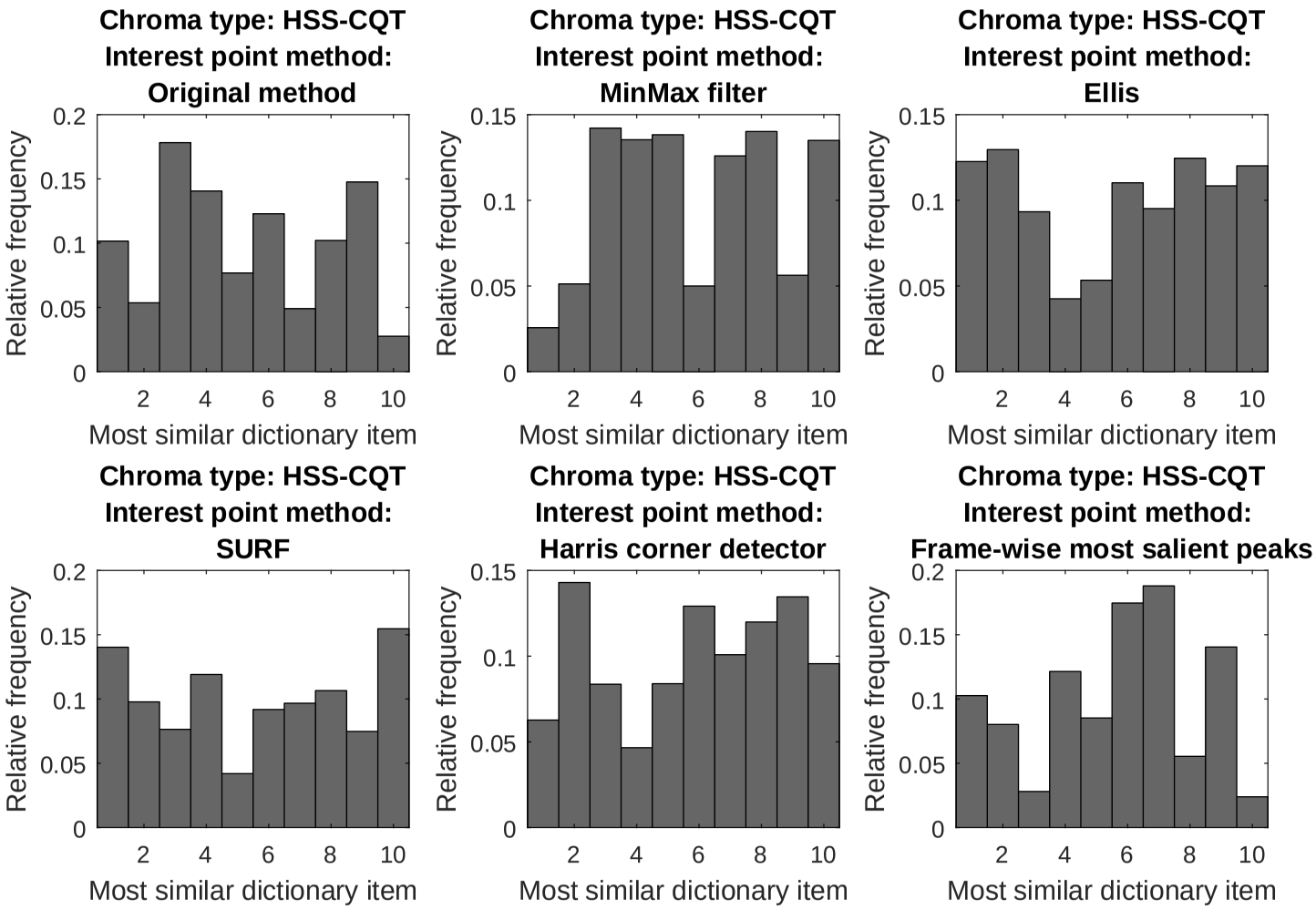


FIGURE B.6: Distribution of types averaged over all database items, separately plotted for all interest point methods and input image HSS-CQT.

Predictability of Point displacements due to Pitch Shifting and Time Scaling C

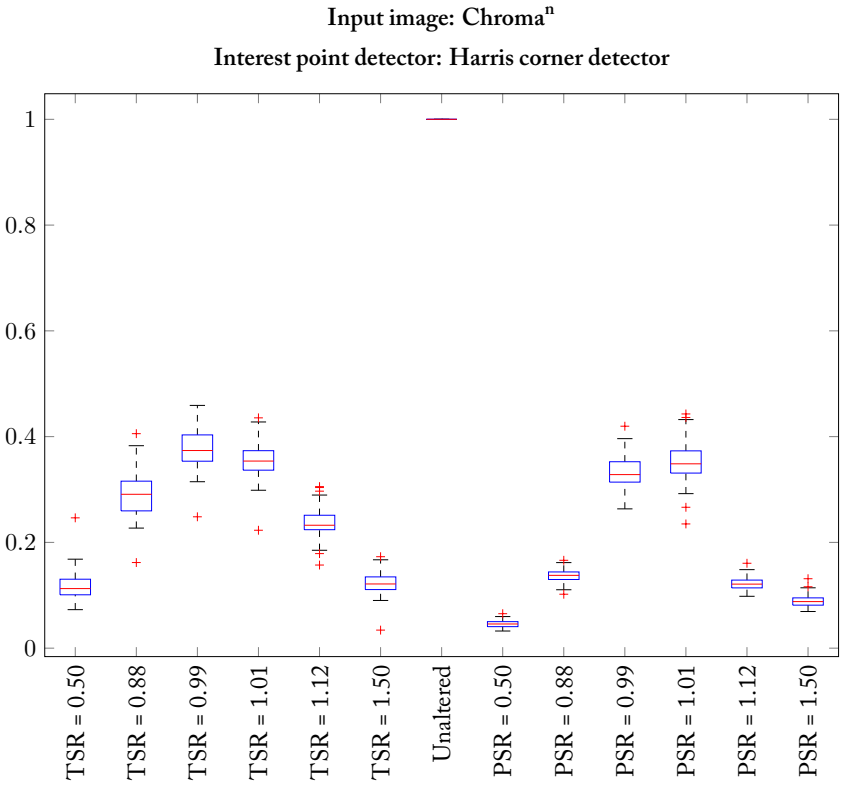


Figure C.1: True Positive Rate for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: Harris corner detector

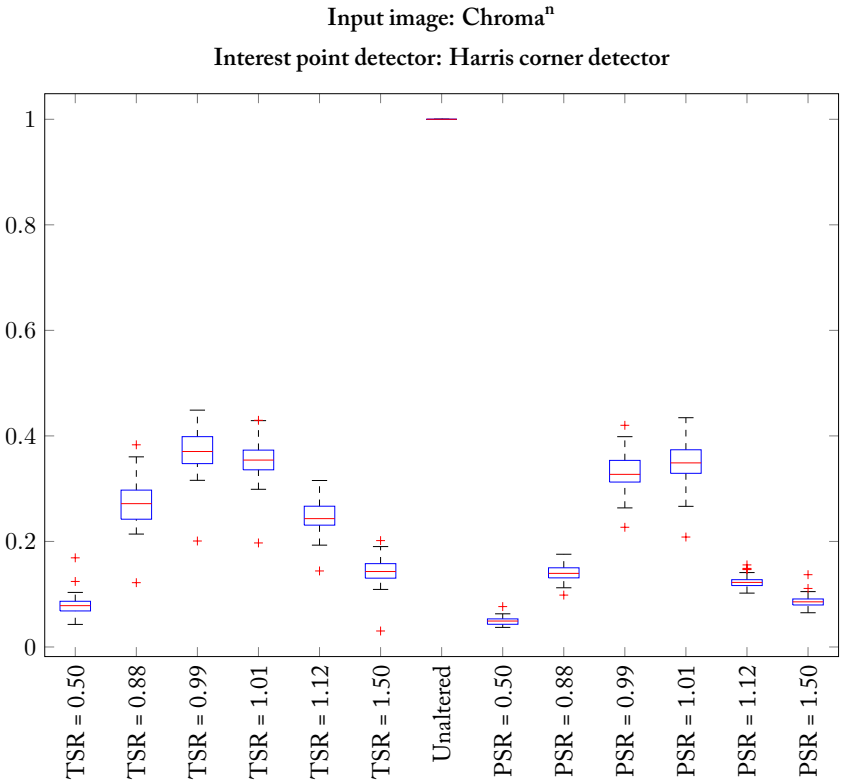


Figure C.2: F_1 score for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: Harris corner detector

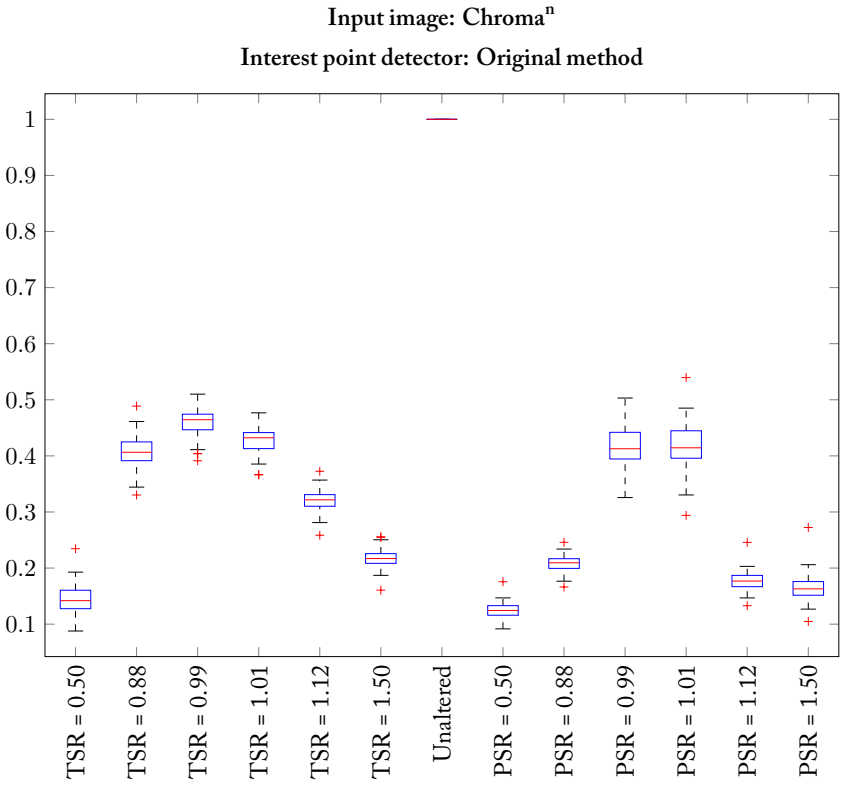


FIGURE C.3: True Positive Rate for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: Original Method

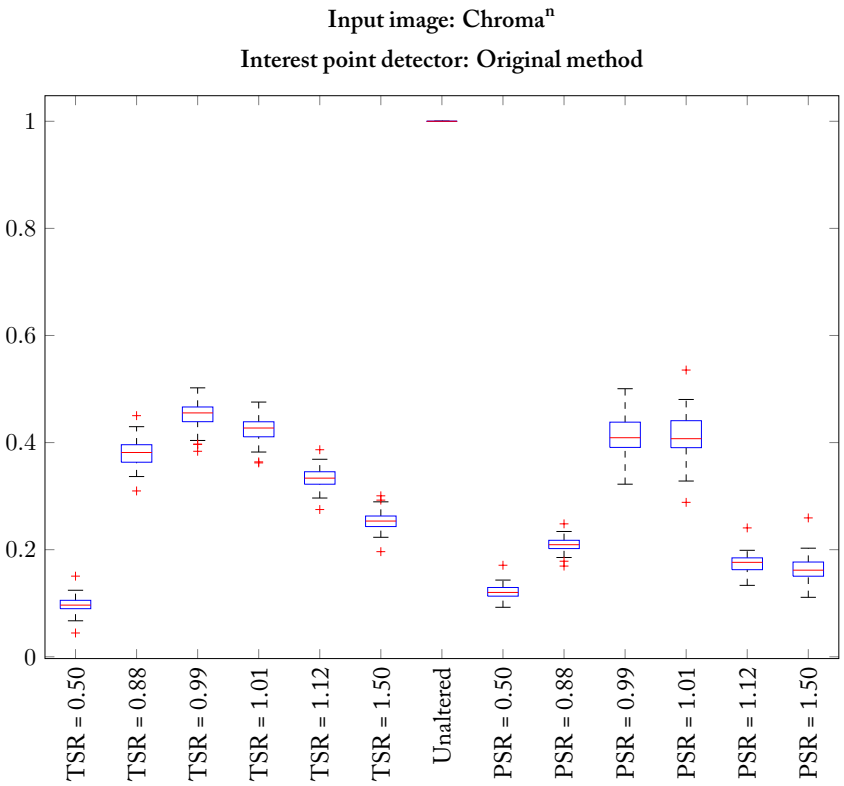


FIGURE C.4: F_1 score for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: Original Method

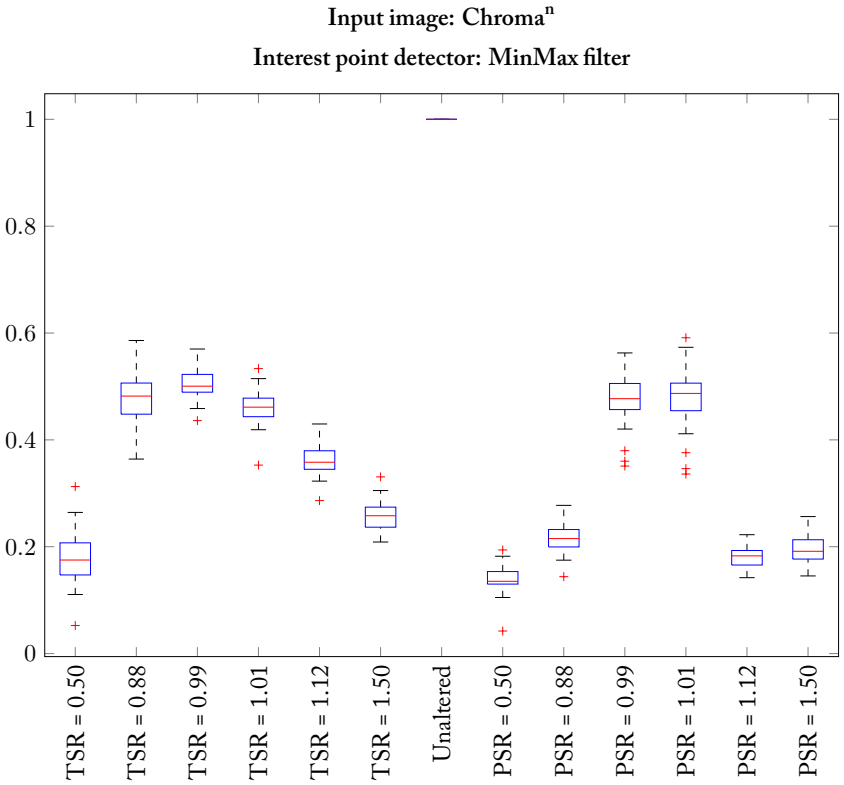


Figure C.5: True Positive Rate for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: MinMax filter

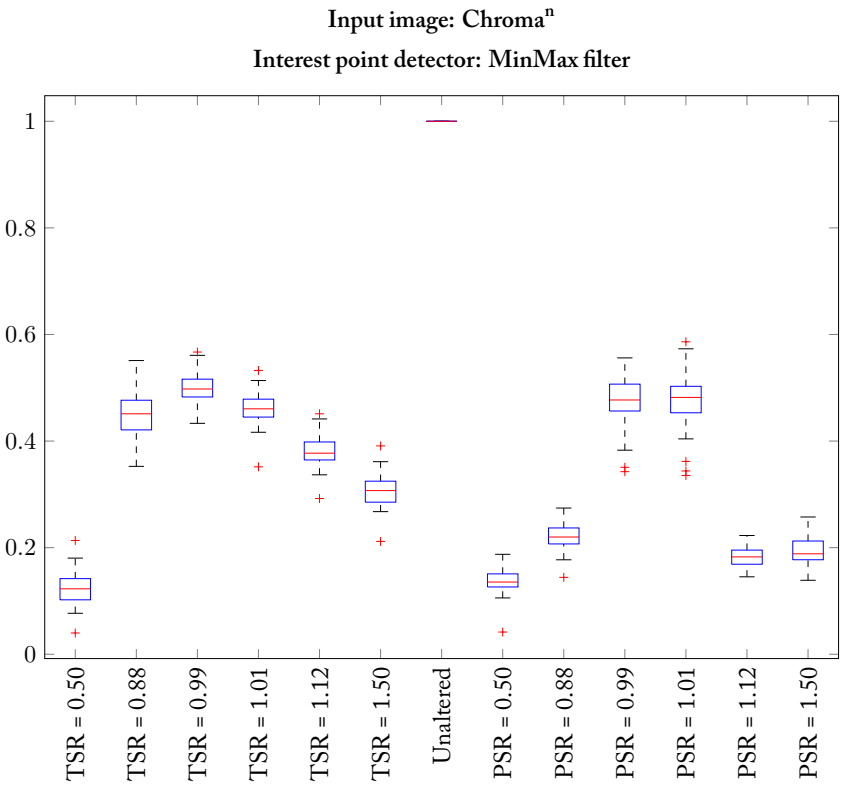


Figure C.6: F_1 score for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: MinMax filter

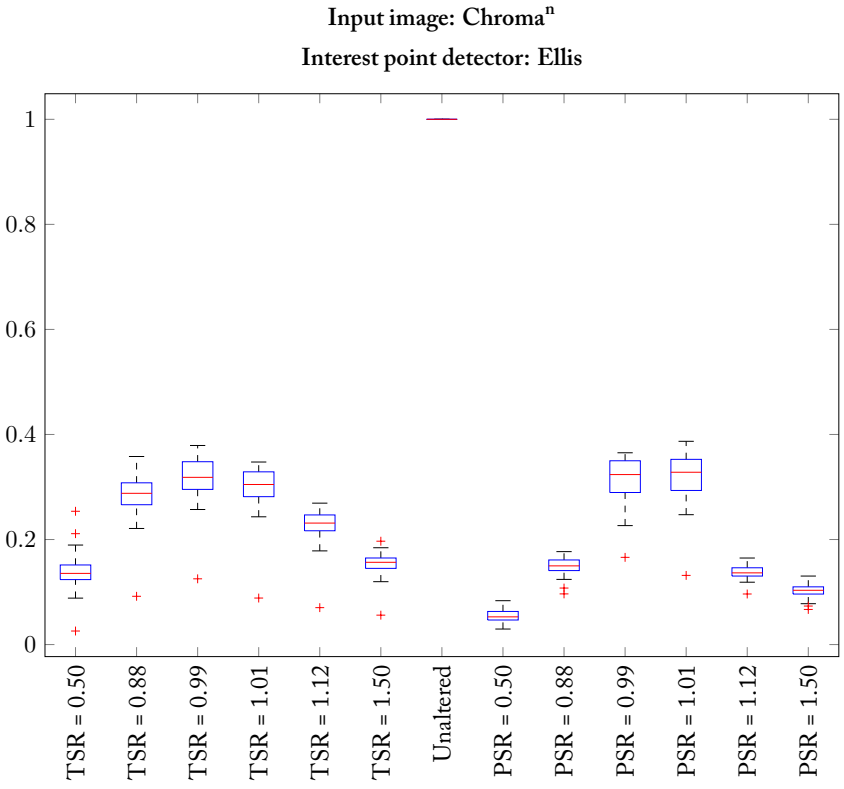


FIGURE C.7: True Positive Rate for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: Ellis

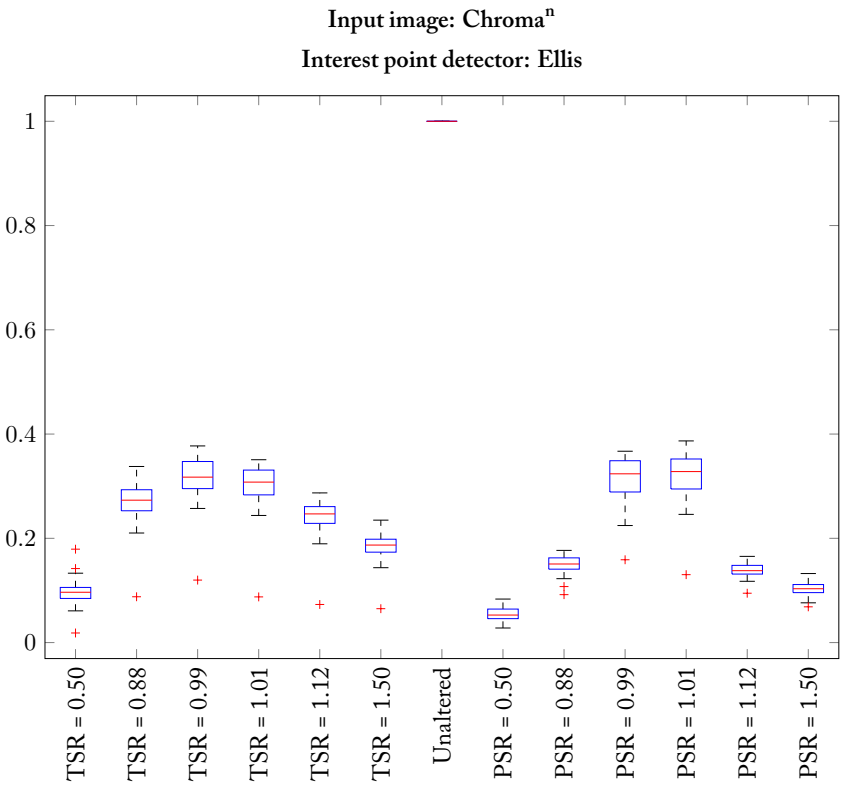


FIGURE C.8: F_1 score for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: Ellis

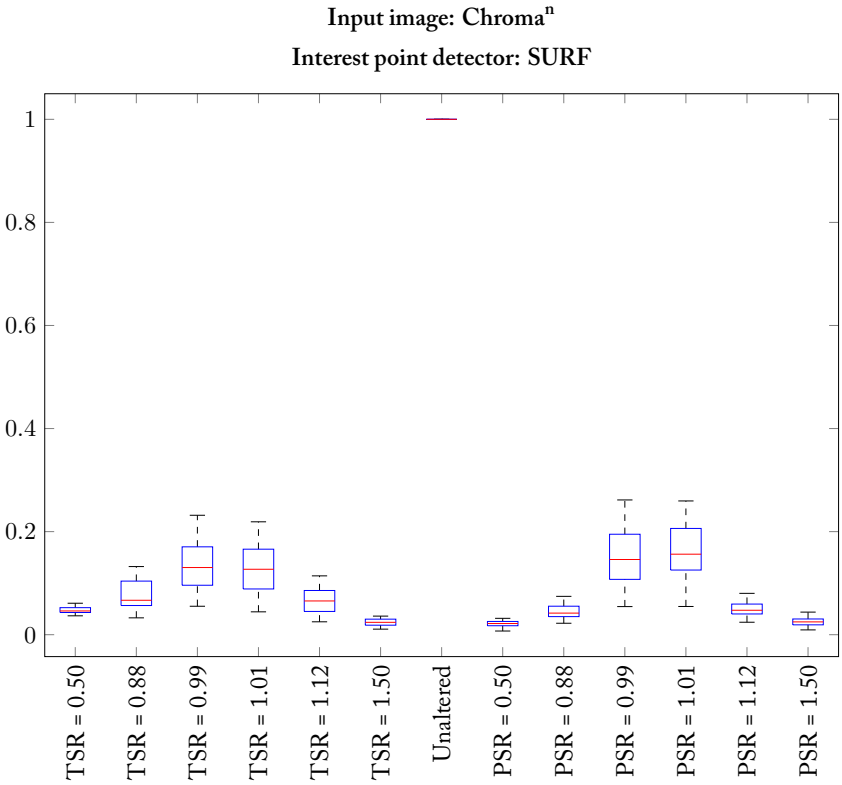


Figure C.9: True Positive Rate for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: SURF

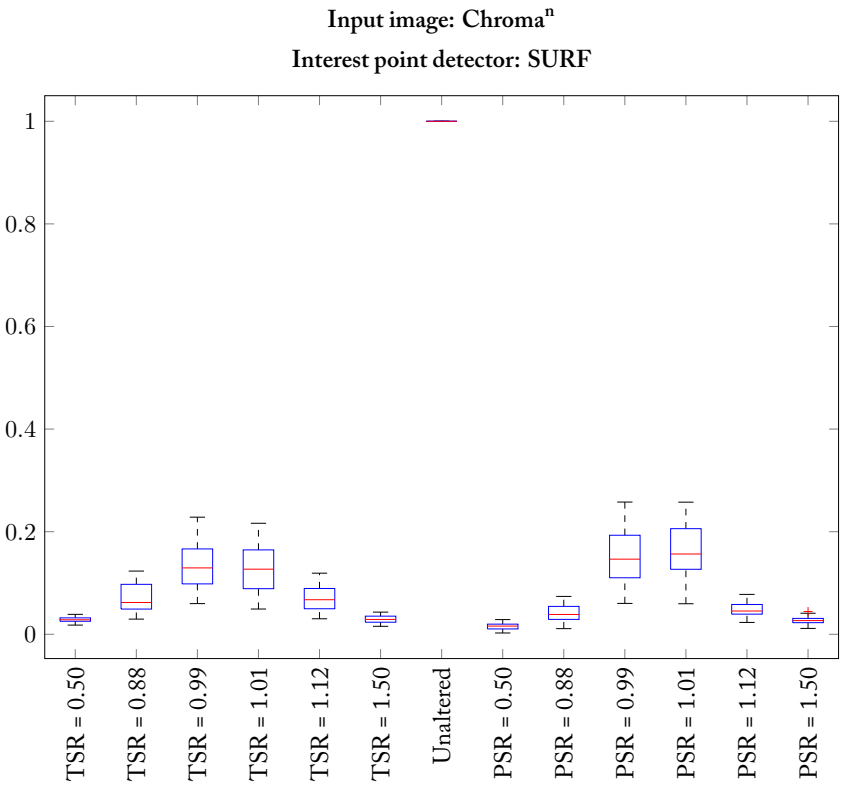


Figure C.10: F_1 score for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: SURF

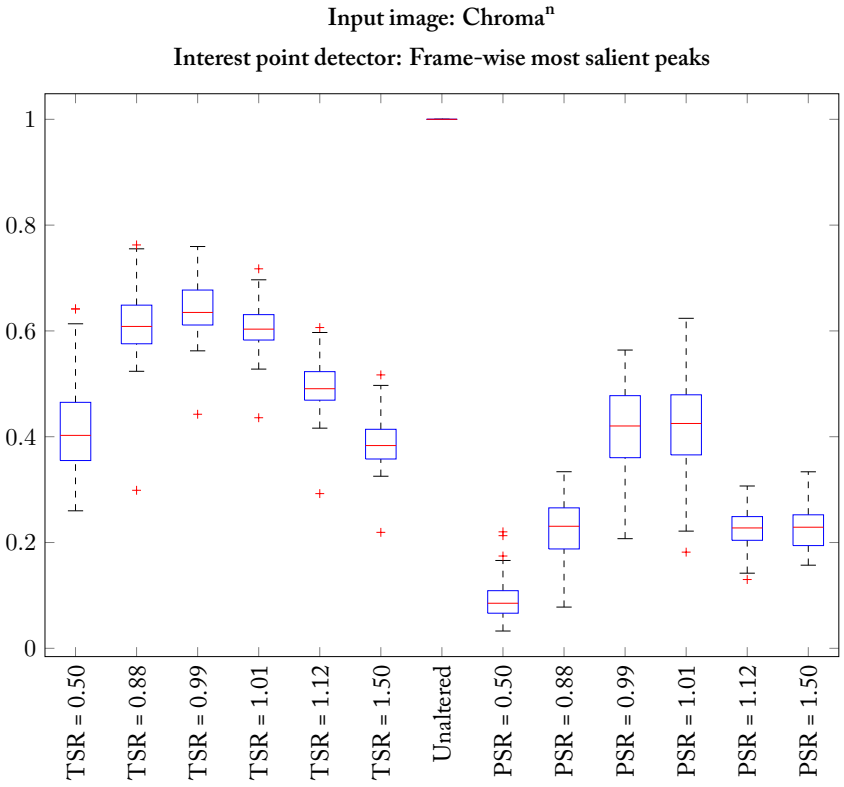


FIGURE C.11: True Positive Rate for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: Frame-wise most salient peaks

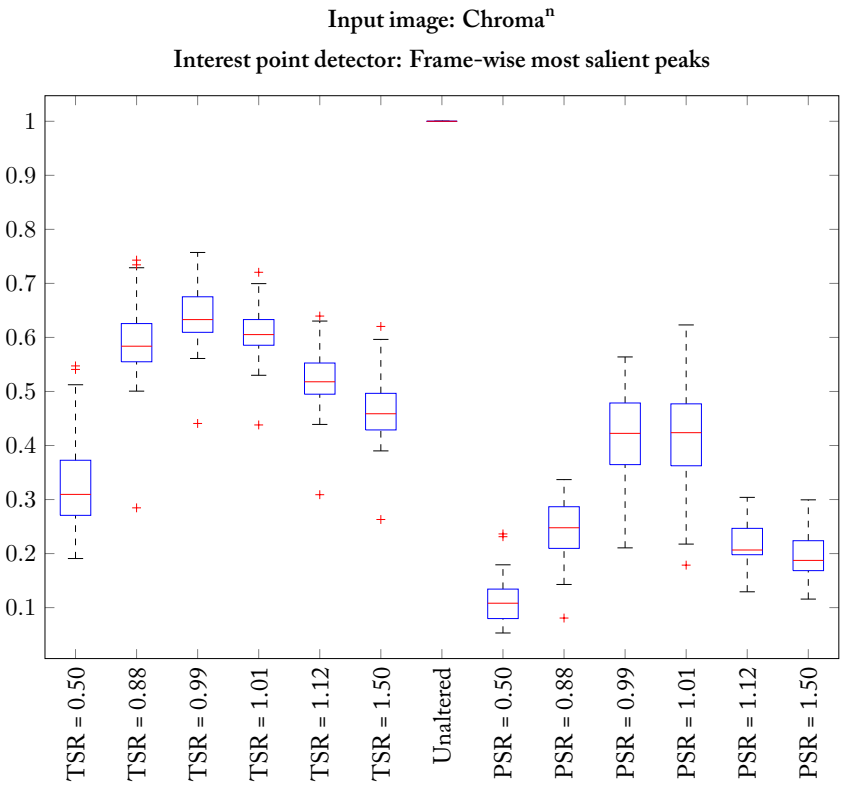


FIGURE C.12: F_1 score for all time scaling and pitch shift ratios, input image: Chroma^n and interest point method: Frame-wise most salient peaks

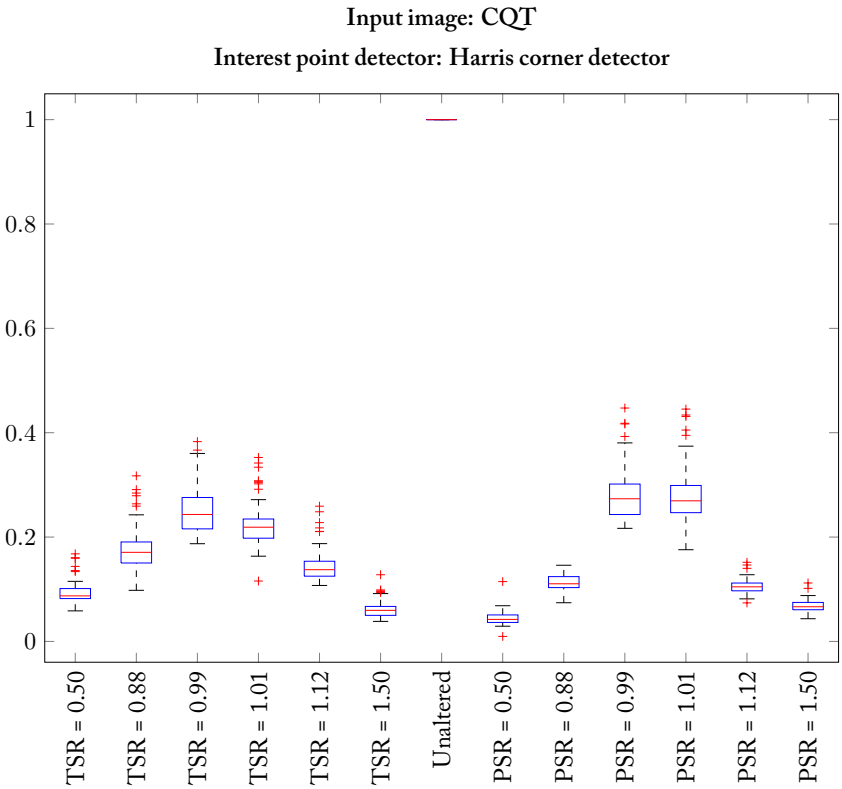


Figure C.13: True Positive Rate for all time scaling and pitch shift ratios, input image: CQT and interest point method: Harris corner detector

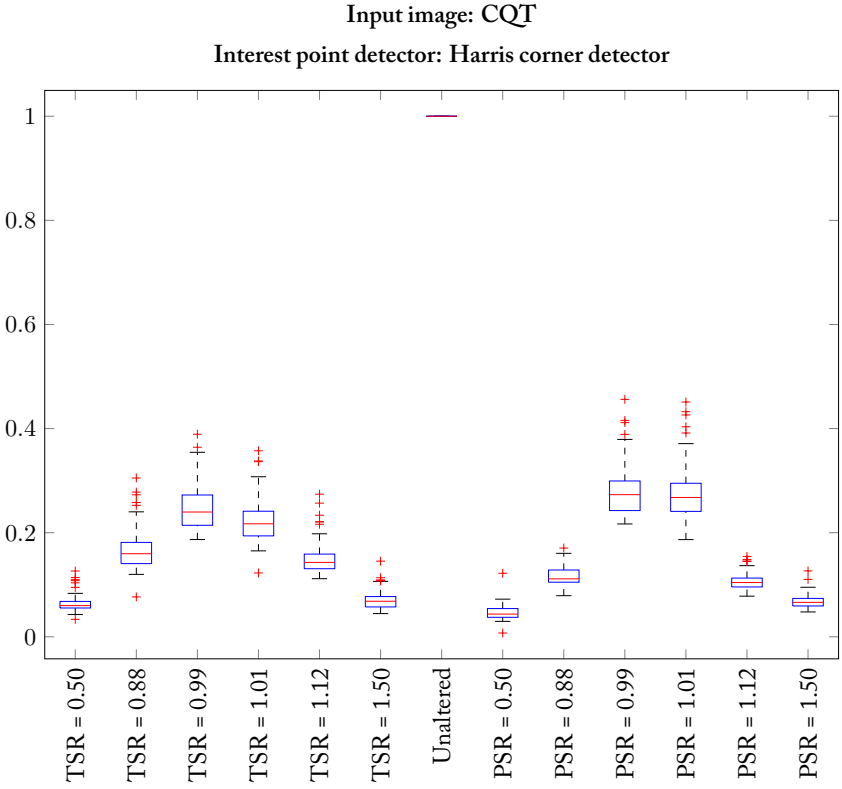


Figure C.14: F_1 score for all time scaling and pitch shift ratios, input image: CQT and interest point method: Harris corner detector

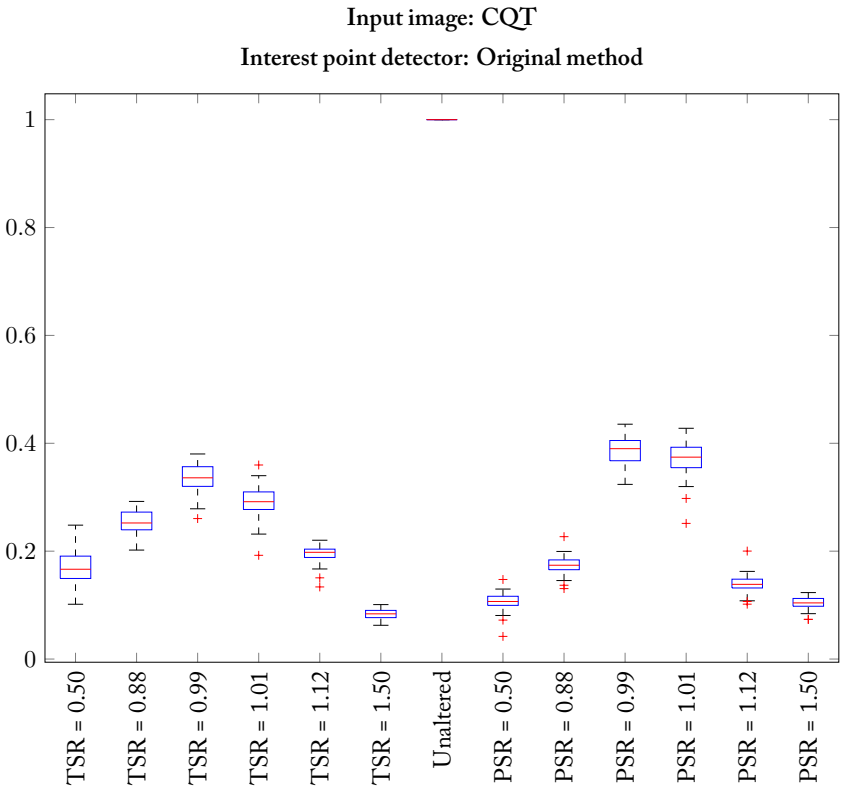


FIGURE C.15: True Positive Rate for all time scaling and pitch shift ratios, input image: CQT and interest point method: Original Method

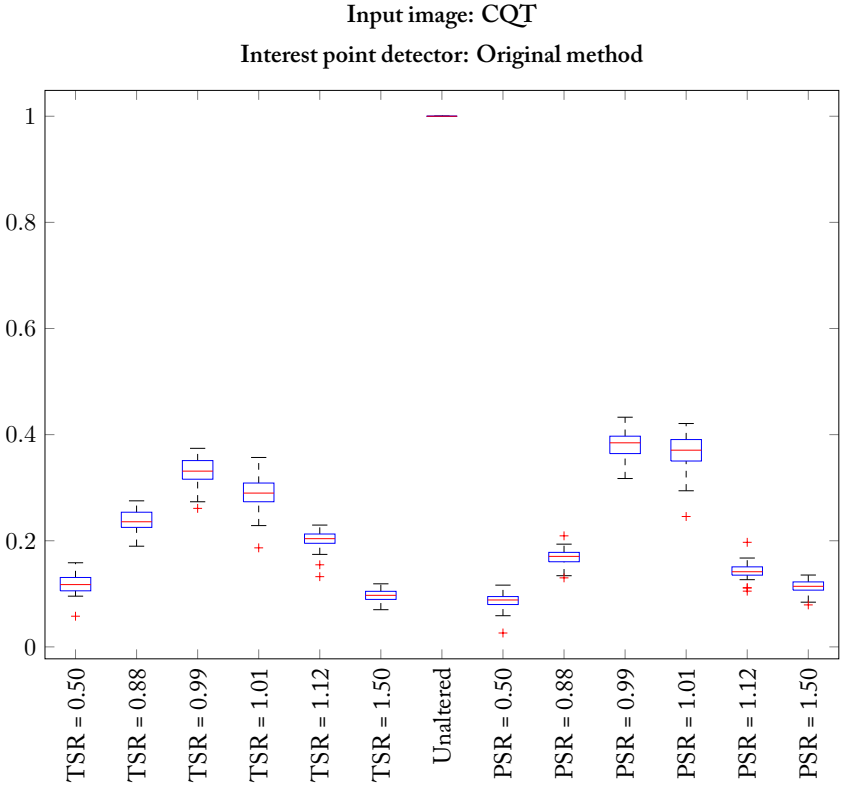


FIGURE C.16: F_1 score for all time scaling and pitch shift ratios, input image: CQT and interest point method: Original Method

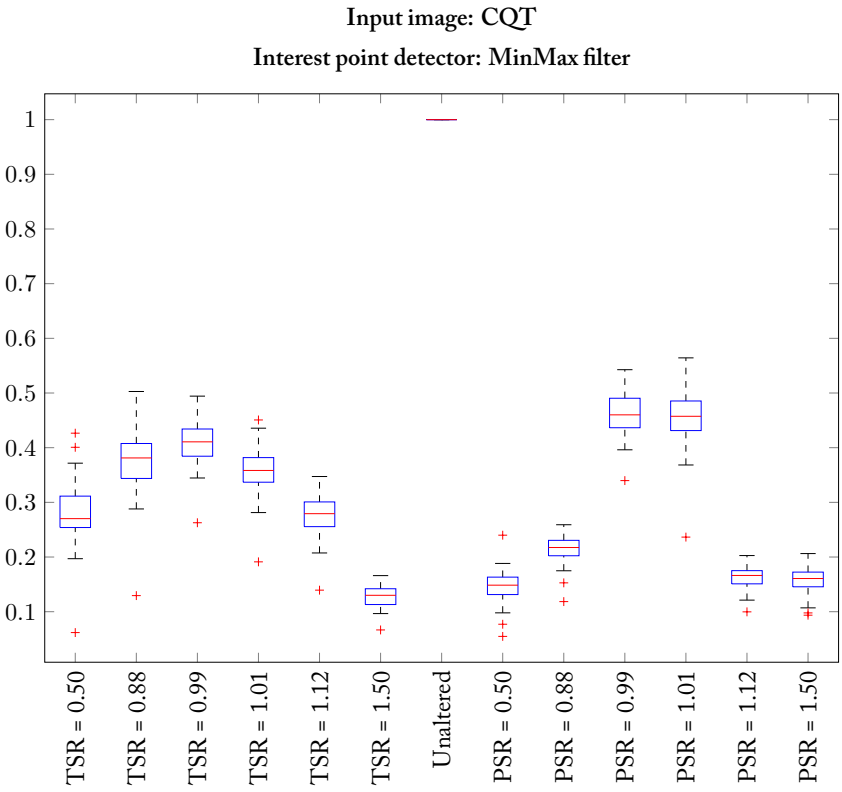


Figure C.17: True Positive Rate for all time scaling and pitch shift ratios, input image: CQT and interest point method: MinMax filter

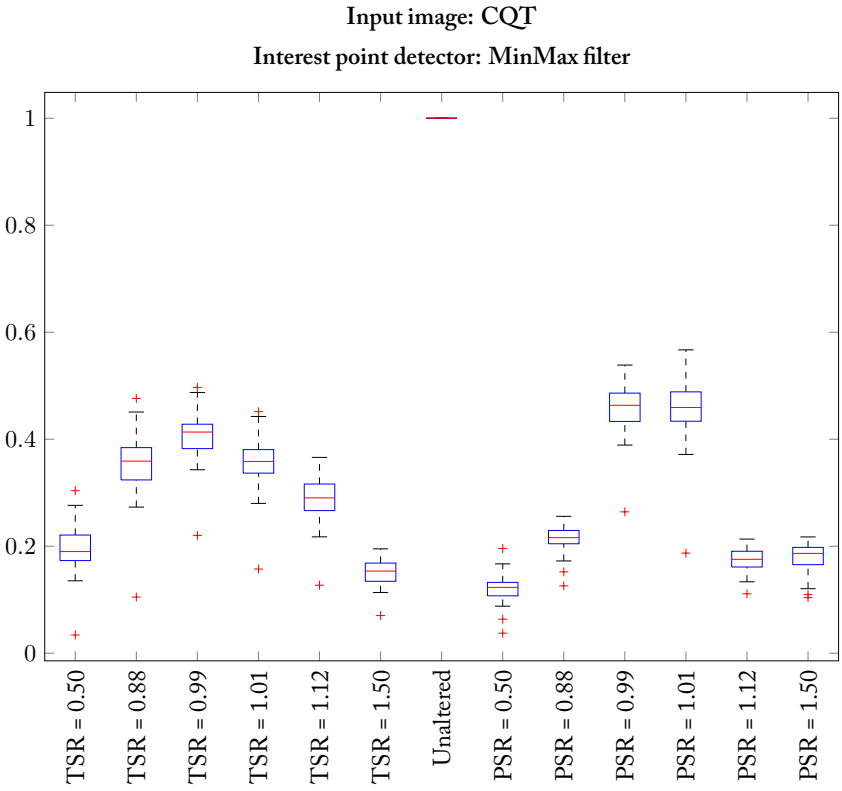


 Figure C.18: F_1 score for all time scaling and pitch shift ratios, input image: CQT and interest point method: MinMax filter

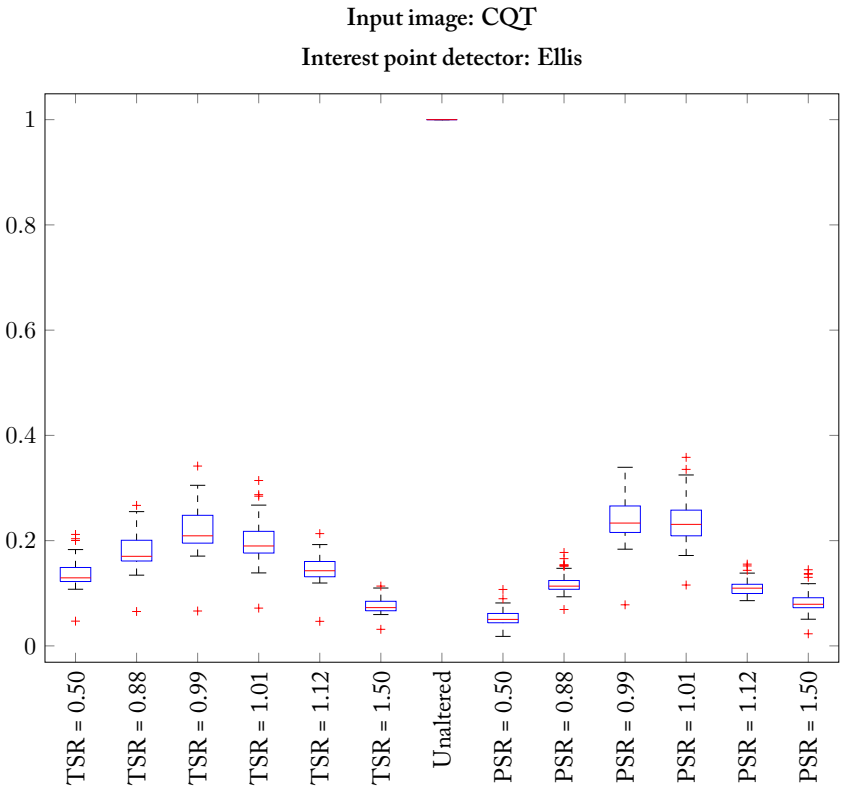


FIGURE C.19: True Positive Rate for all time scaling and pitch shift ratios, input image: CQT and interest point method: Ellis

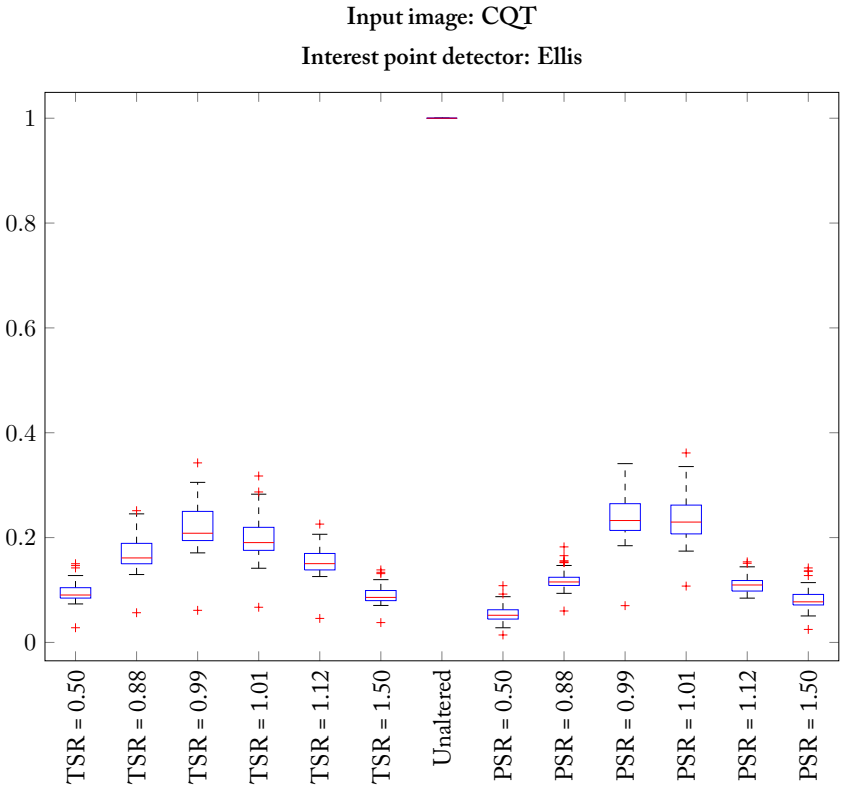


FIGURE C.20: F_1 score for all time scaling and pitch shift ratios, input image: CQT and interest point method: Ellis

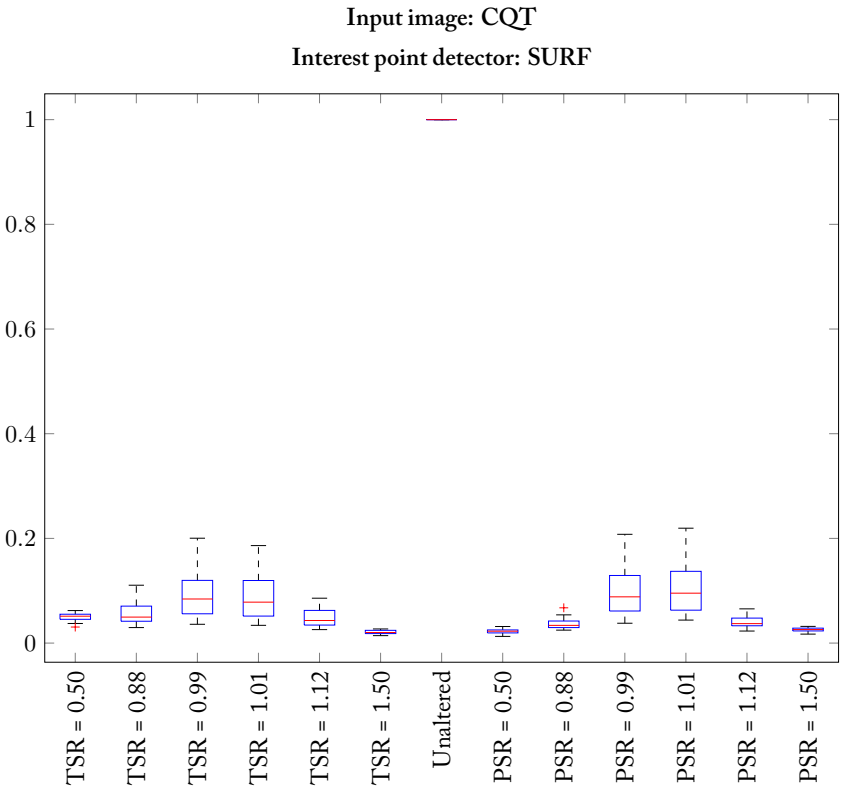


Figure C.21: True Positive Rate for all time scaling and pitch shift ratios, input image: CQT and interest point method: SURF

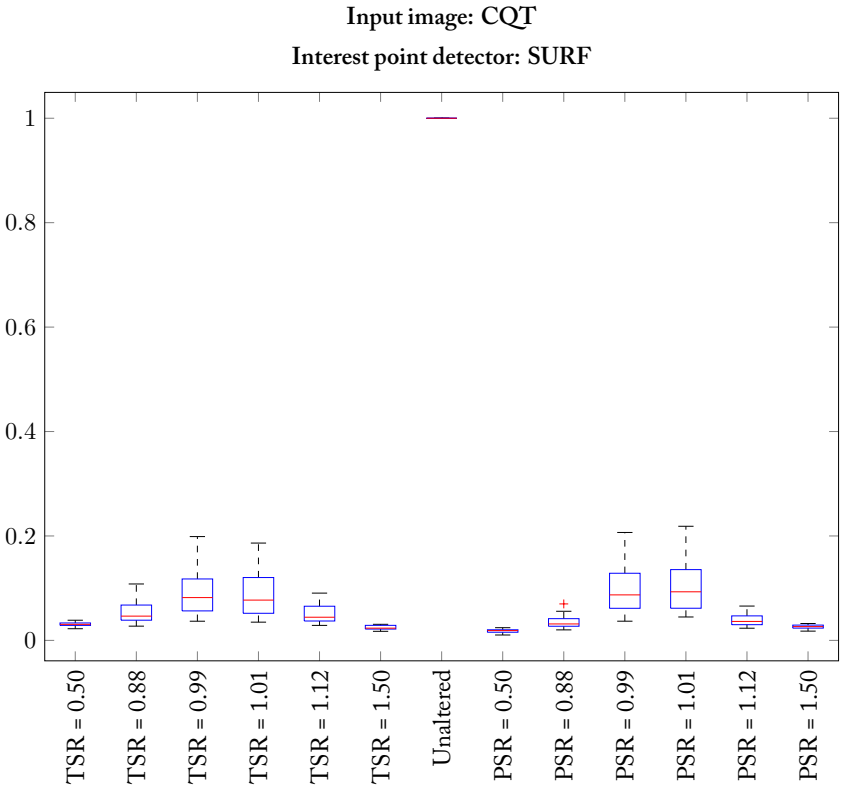


 Figure C.22: F_1 score for all time scaling and pitch shift ratios, input image: CQT and interest point method: SURF

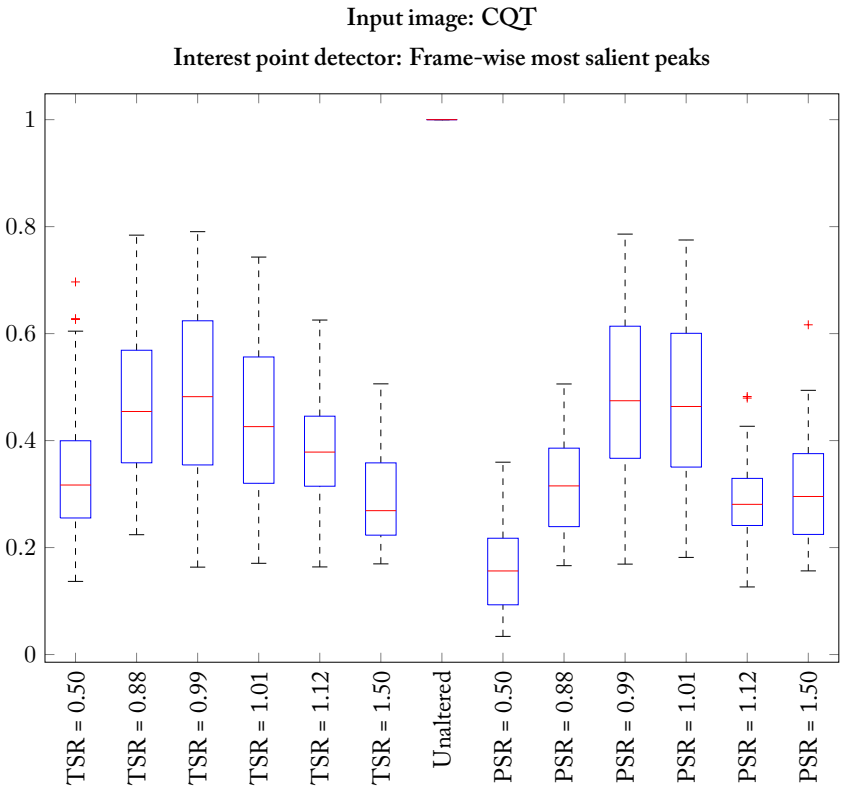


FIGURE C.23: True Positive Rate for all time scaling and pitch shift ratios, input image: CQT and interest point method: Frame-wise most salient peaks

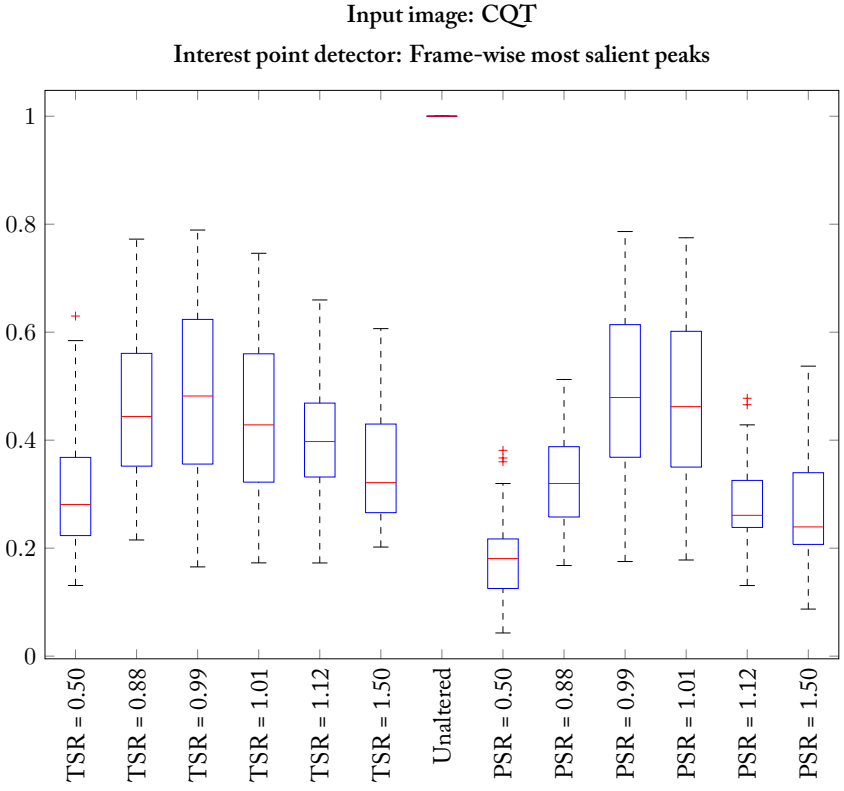


FIGURE C.24: F_1 score for all time scaling and pitch shift ratios, input image: CQT and interest point method: Frame-wise most salient peaks

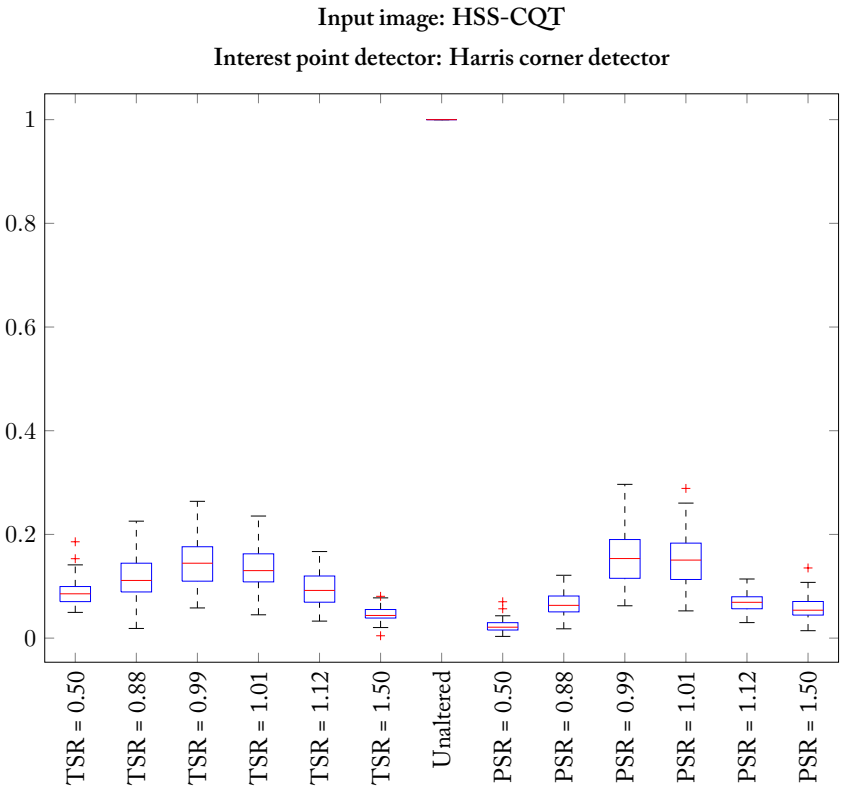


Figure C.25: True Positive Rate for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: Harris corner detector

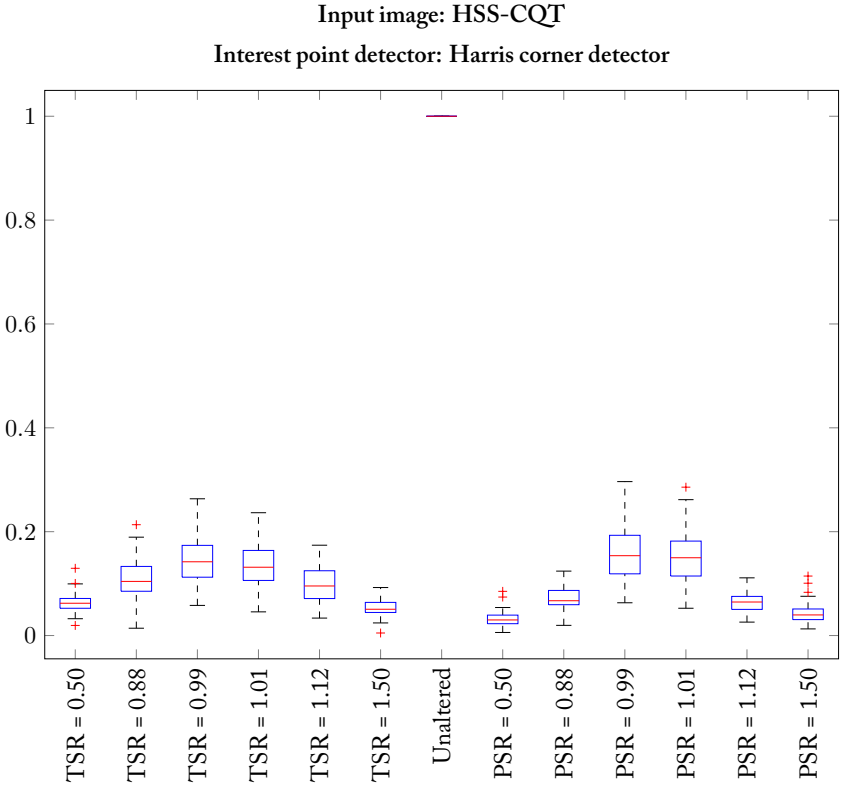


Figure C.26: F_1 score for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: Harris corner detector

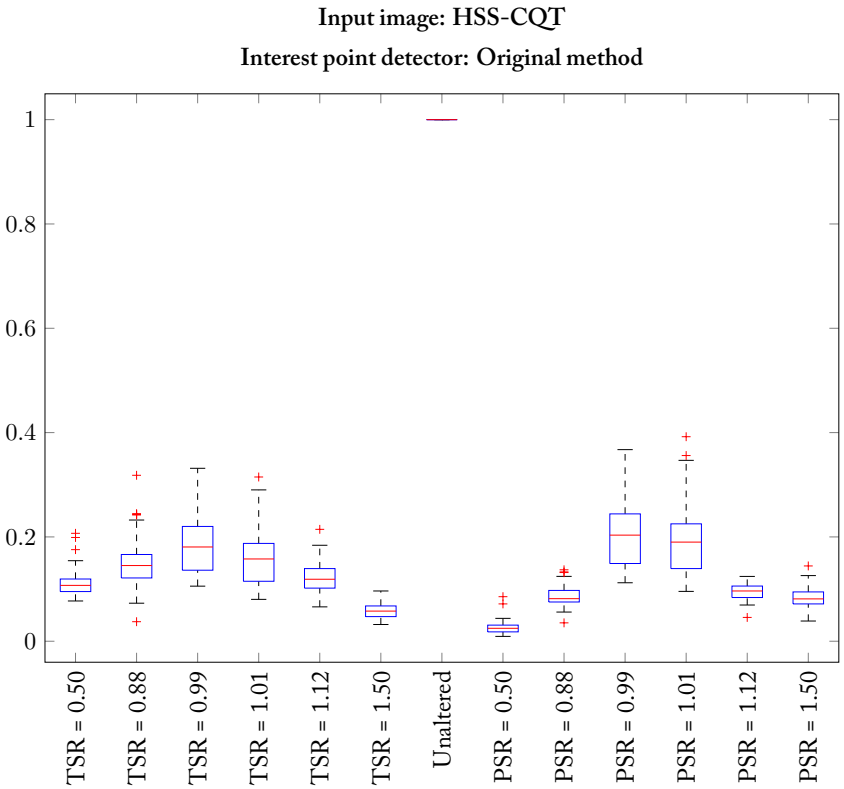


FIGURE C.27: True Positive Rate for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: Original Method

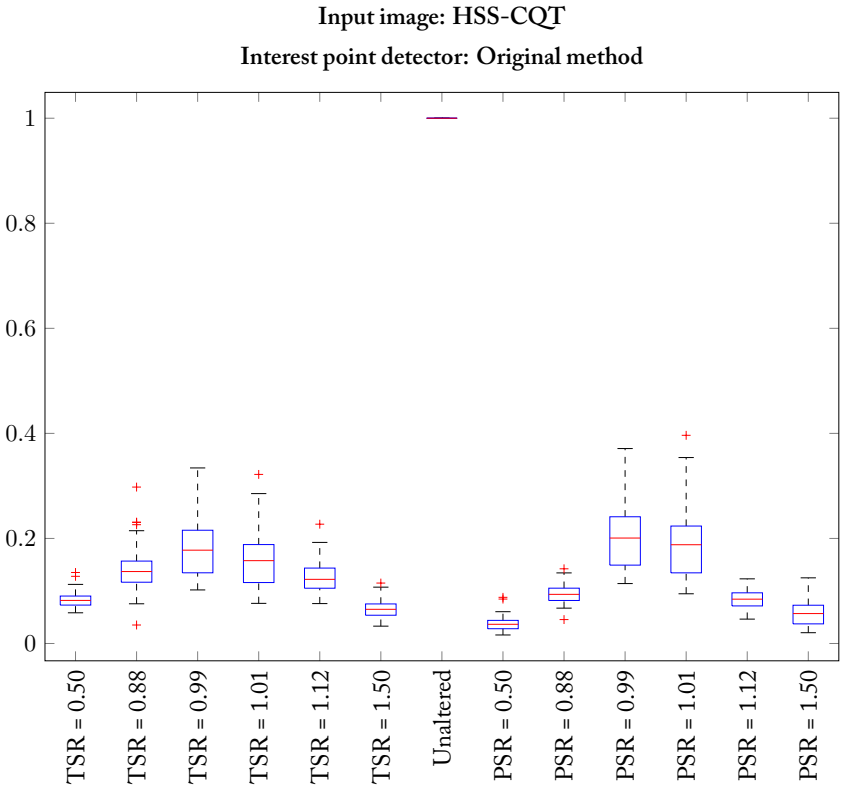


FIGURE C.28: F_1 score for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: Original Method

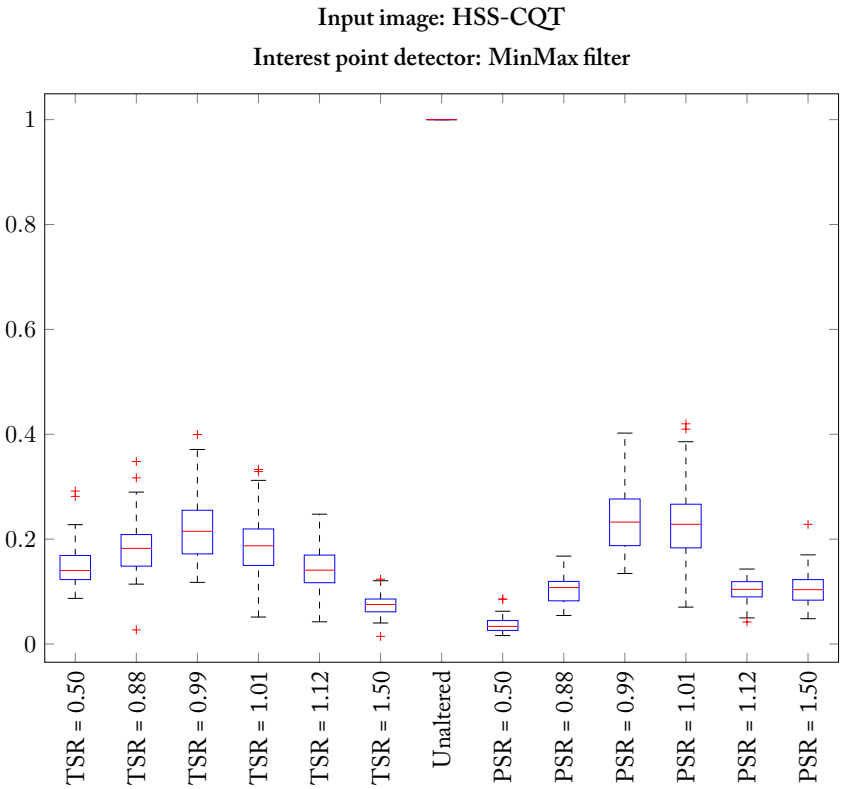


Figure C.29: True Positive Rate for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: MinMax filter

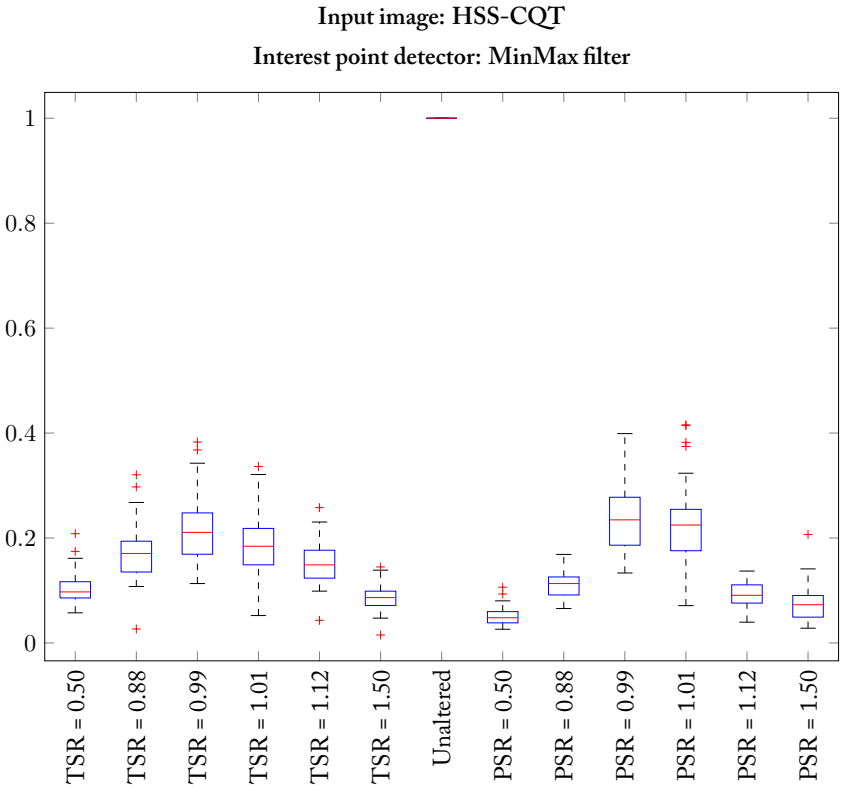


 Figure C.30: F_1 score for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: MinMax filter

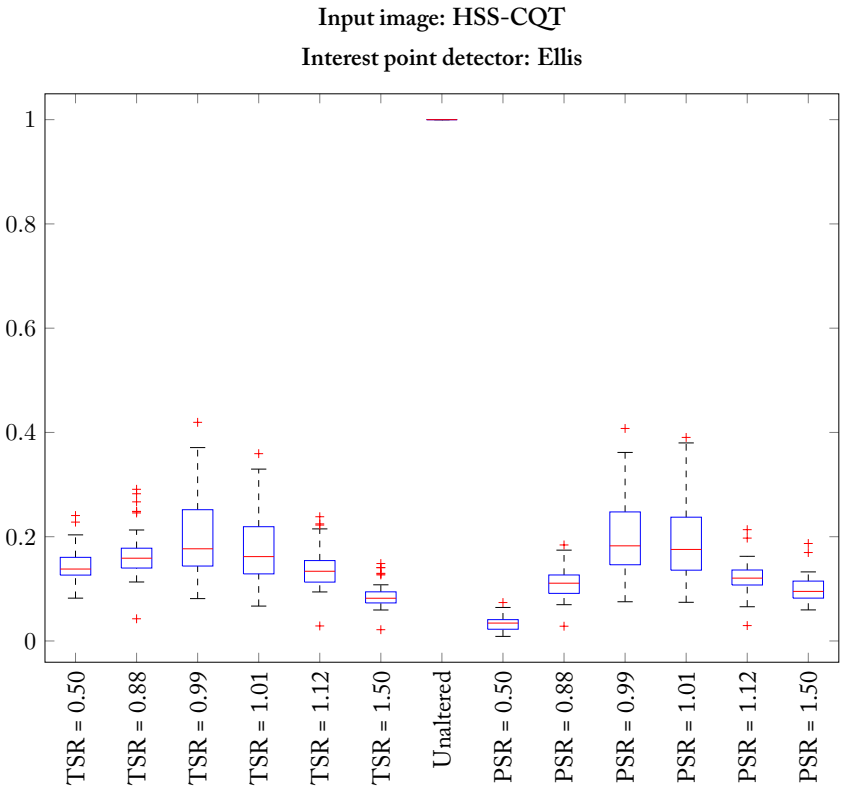


FIGURE C.31: True Positive Rate for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: Ellis

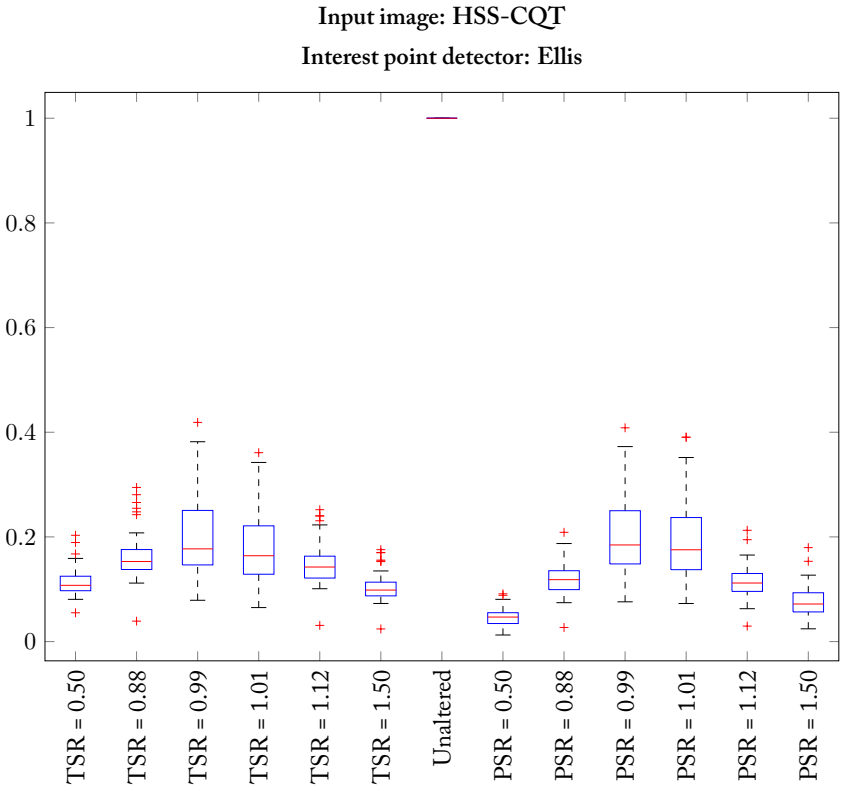


FIGURE C.32: F_1 score for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: Ellis

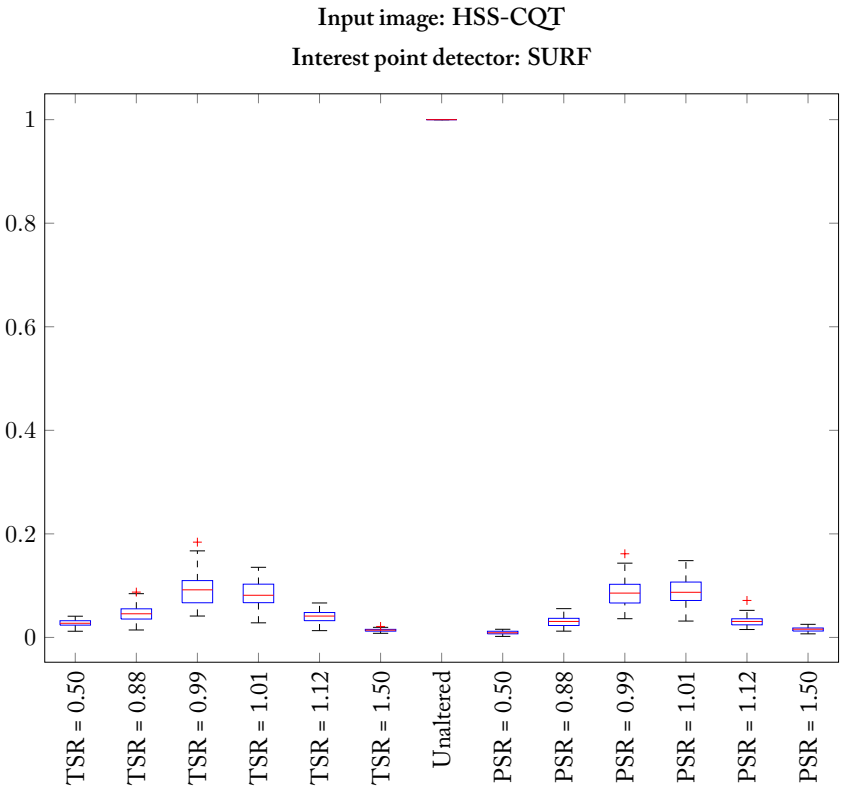


Figure C.33: True Positive Rate for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: SURF

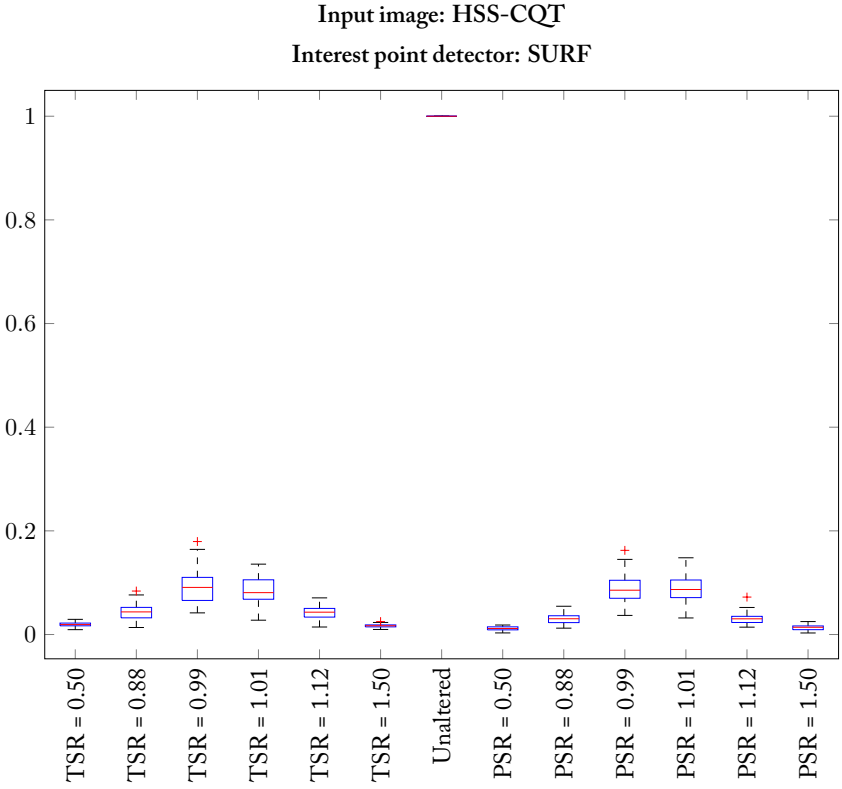


 Figure C.34: F_1 score for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: SURF

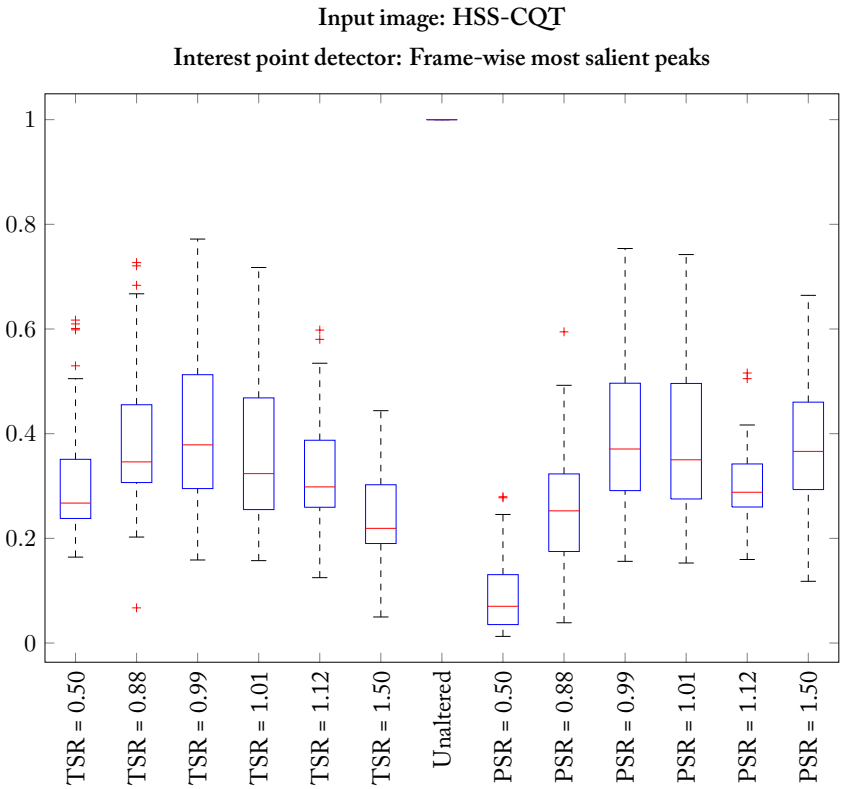


FIGURE C.35: True Positive Rate for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: Frame-wise most salient peaks

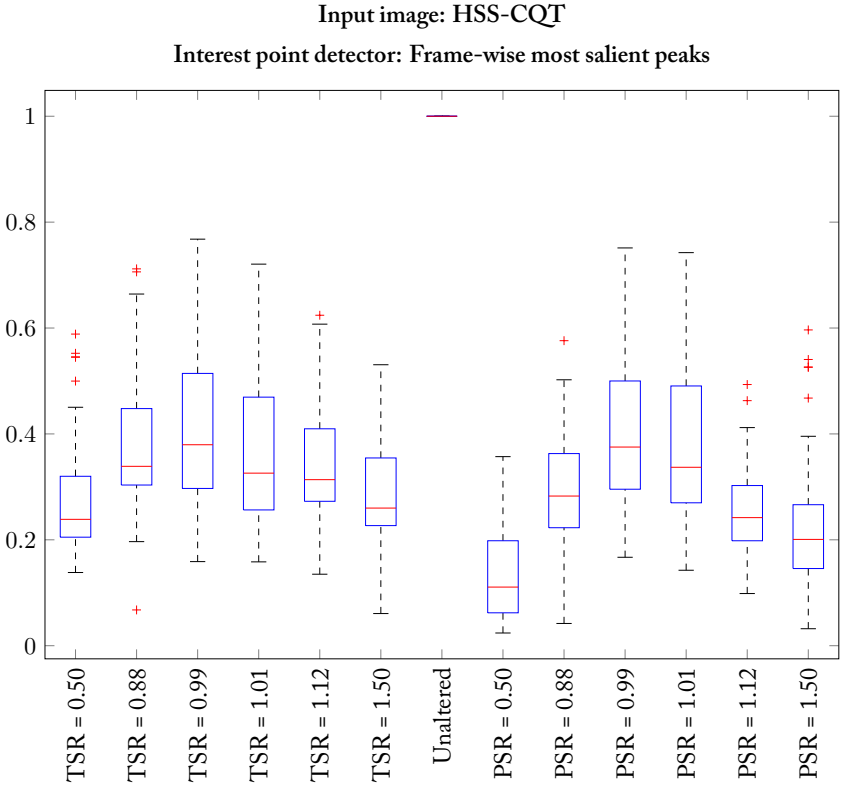


FIGURE C.36: F_1 score for all time scaling and pitch shift ratios, input image: HSS-CQT and interest point method: Frame-wise most salient peaks

Results of the retrieval with pitch shifts and time scalings
D

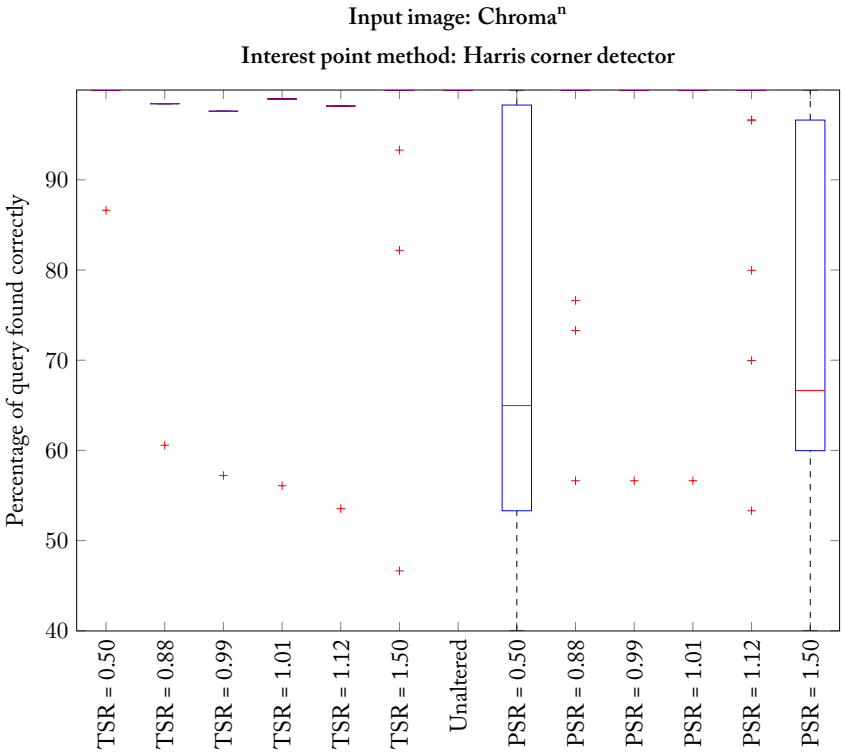


Figure D.1: Distributions of the retrieval ratios for the coarse time slot estimation, input image: Chroma^n and interest point method: Harris corner detector

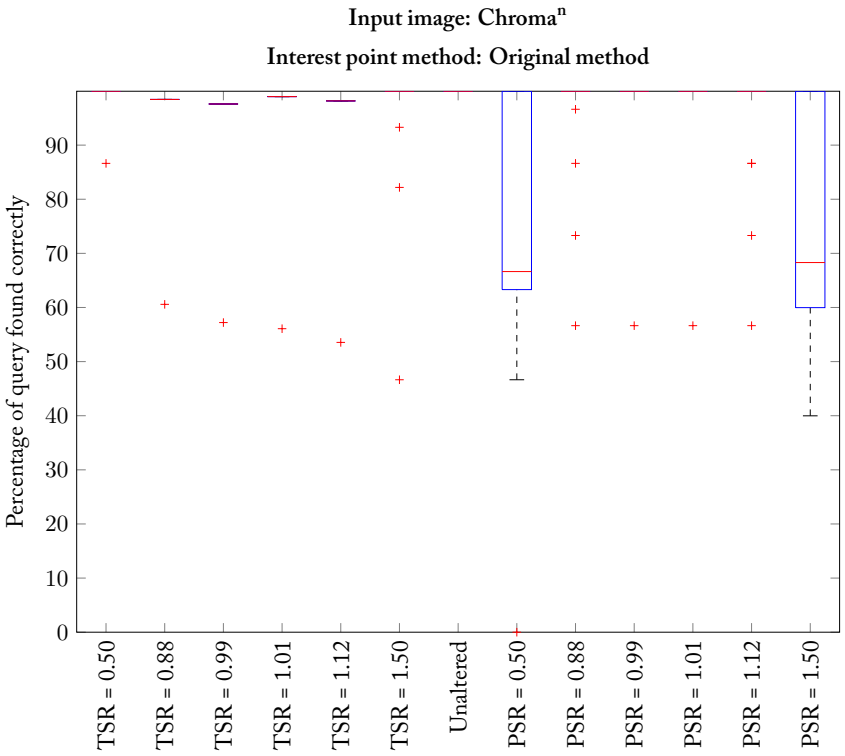


Figure D.2: Distributions of the retrieval ratios for the coarse time slot estimation, input image: Chroma^n and interest point method: Original method

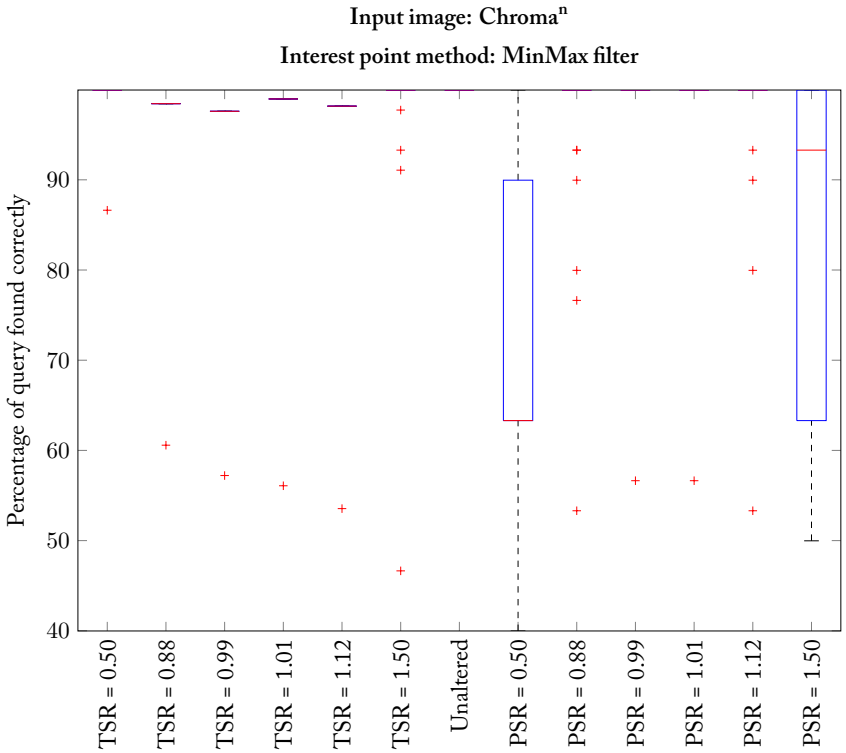


FIGURE D.3: Distributions of the retrieval ratios for the coarse time slot estimation, input image: Chroma^n and interest point method: MinMax filter

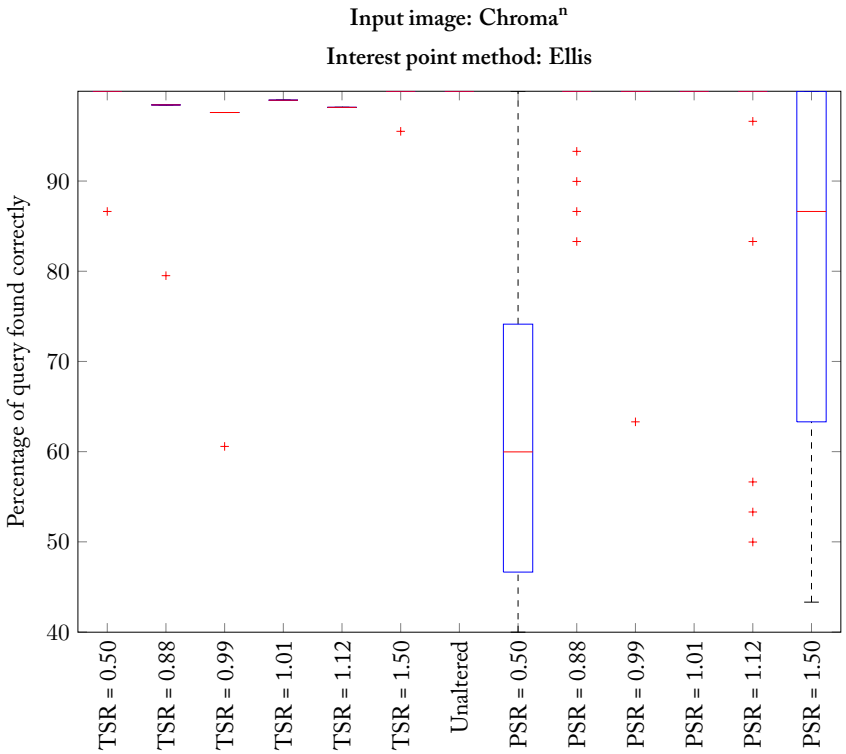


FIGURE D.4: Distributions of the retrieval ratios for the coarse time slot estimation, input image: Chroma^n and interest point method: Ellis

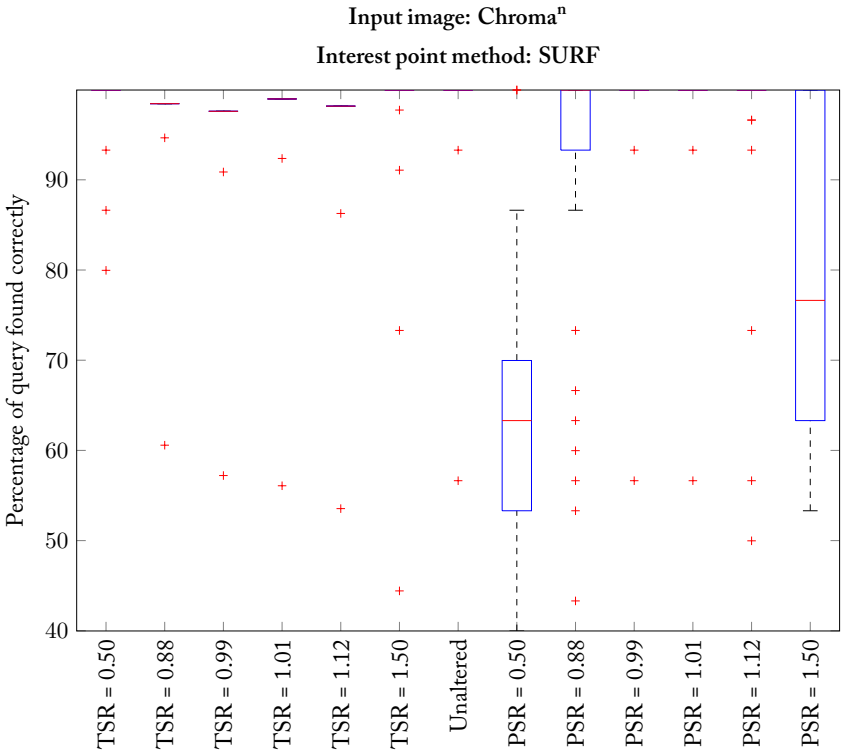


Figure D.5: Distributions of the retrieval ratios for the coarse time slot estimation, input image: Chroma^n and interest point method: SURF

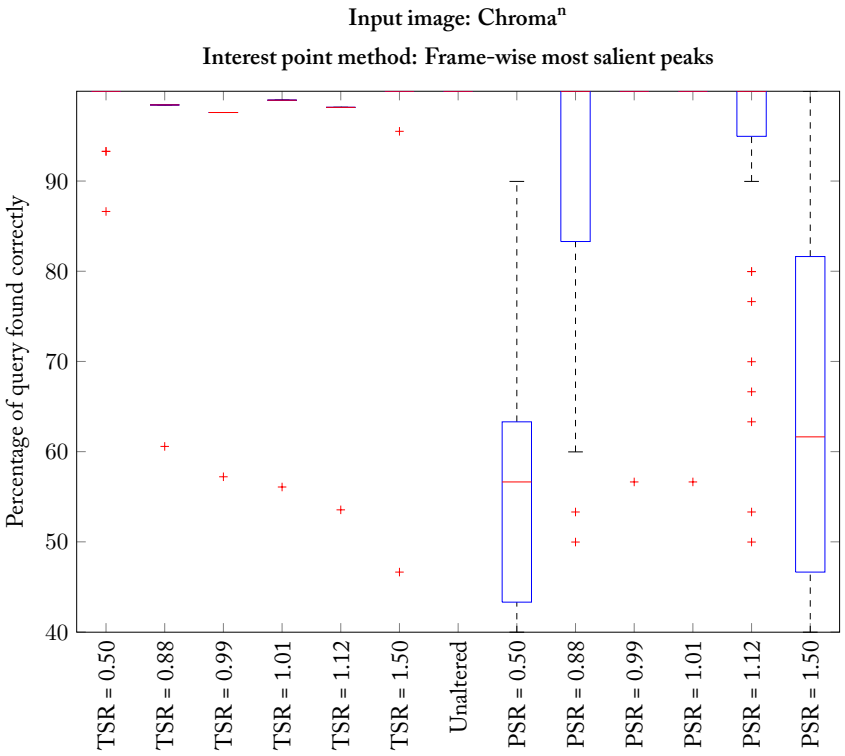


Figure D.6: Distributions of the retrieval ratios for the coarse time slot estimation, input image: Chroma^n and interest point method: Frame-wise most salient peaks

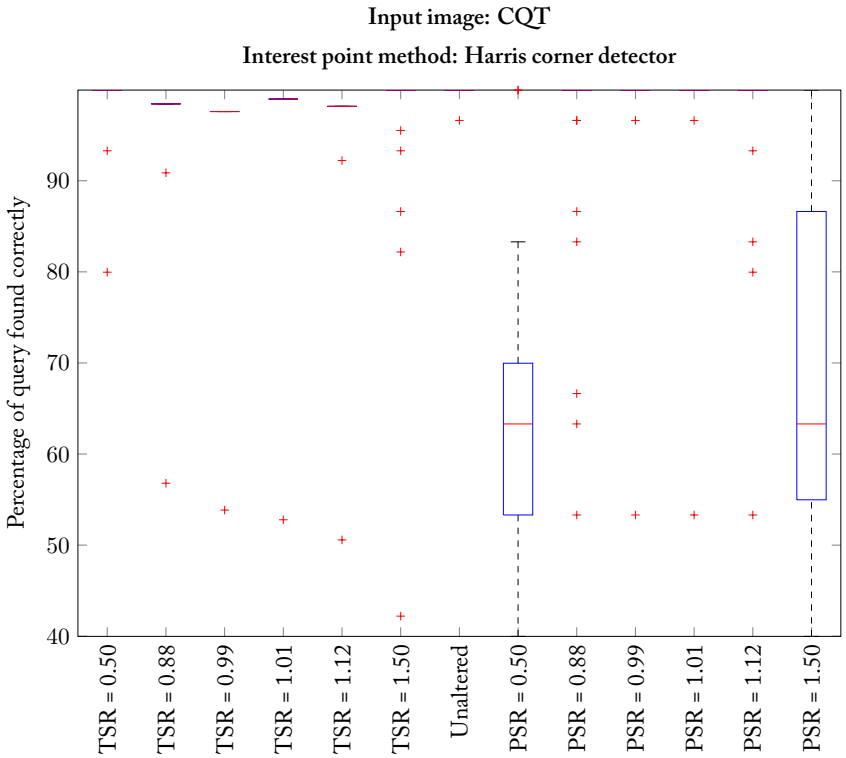


FIGURE D.7: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: Harris corner detector

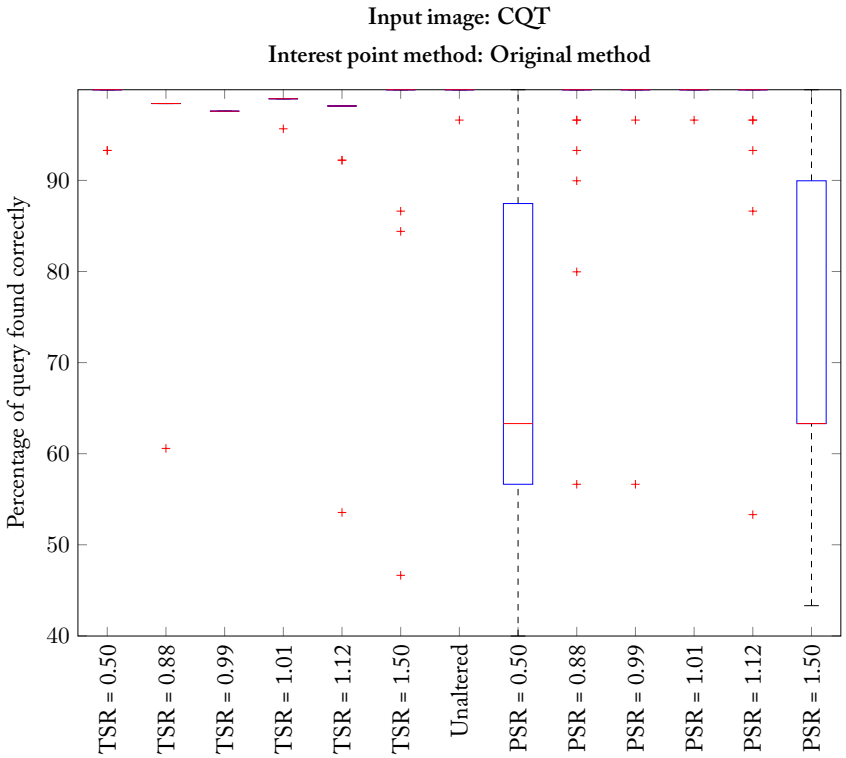


FIGURE D.8: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: Original method

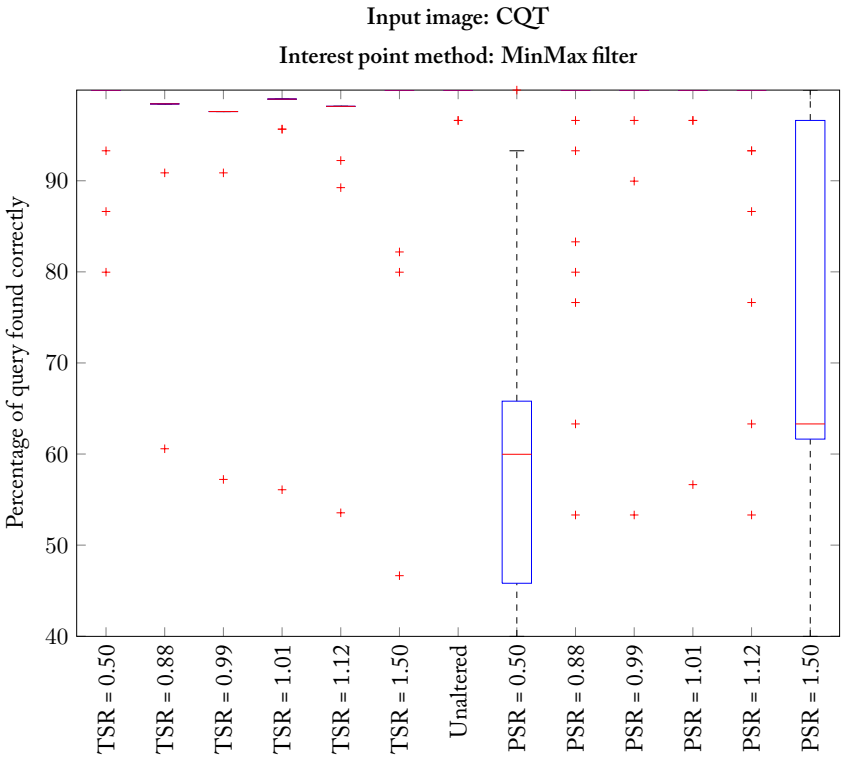


Figure D.9: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: MinMax filter

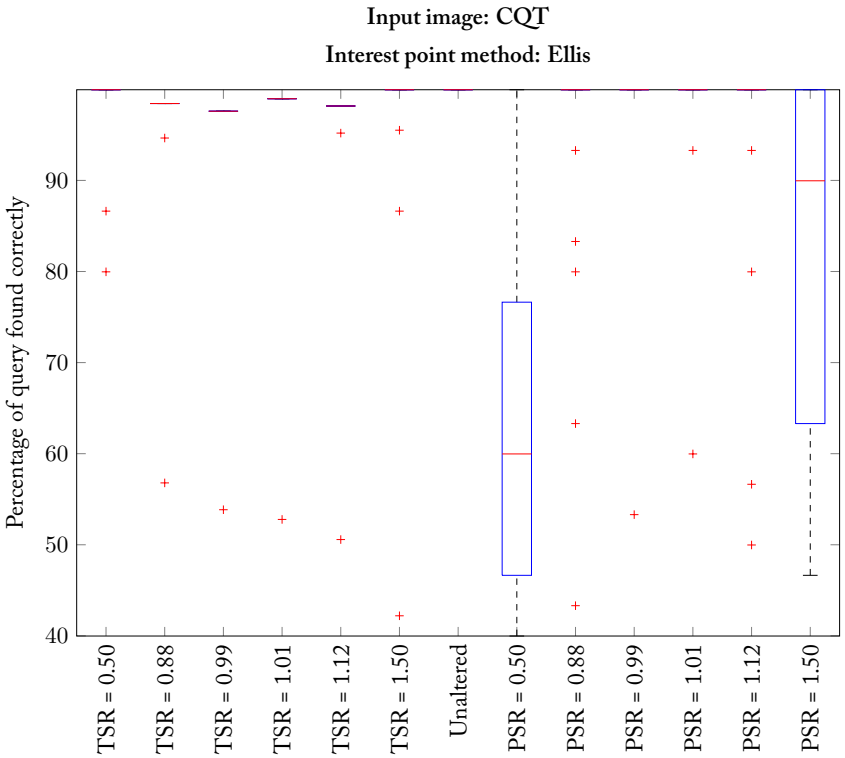


Figure D.10: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: Ellis

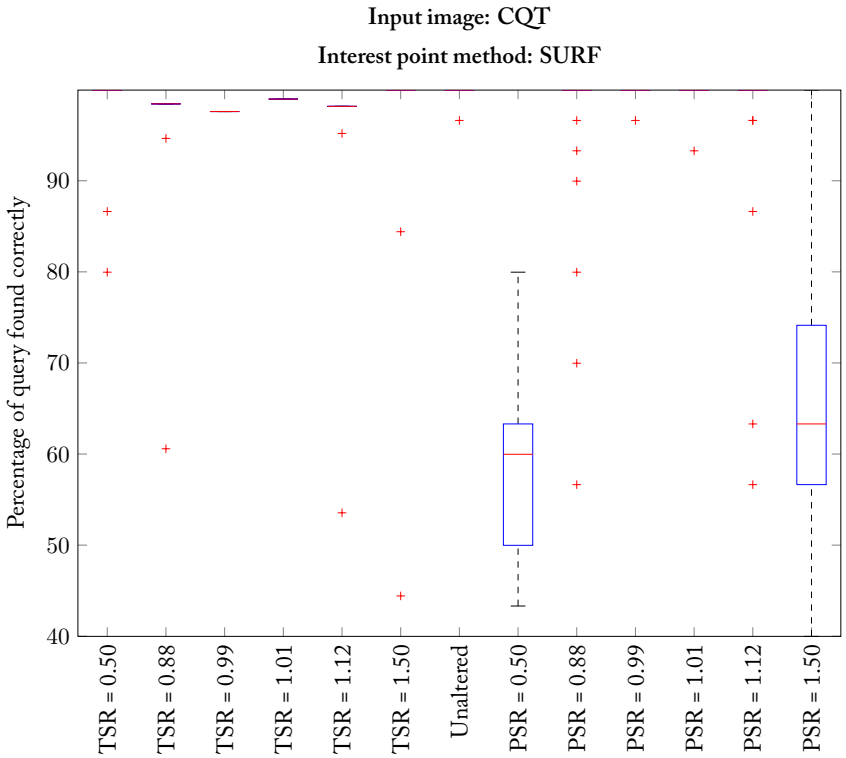


FIGURE D.11: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: SURF

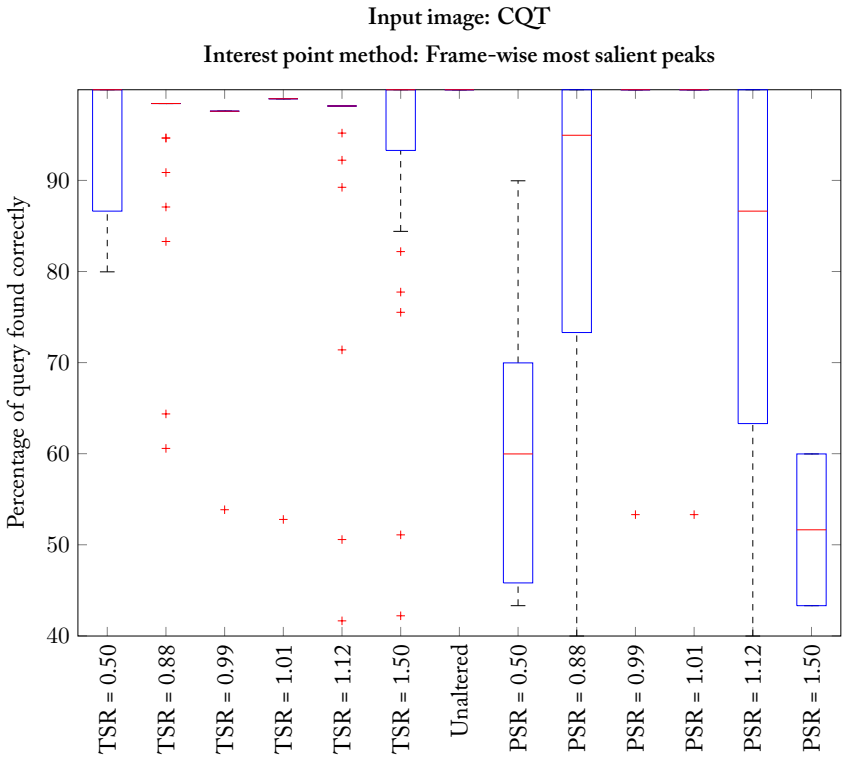


FIGURE D.12: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: Frame-wise most salient peaks

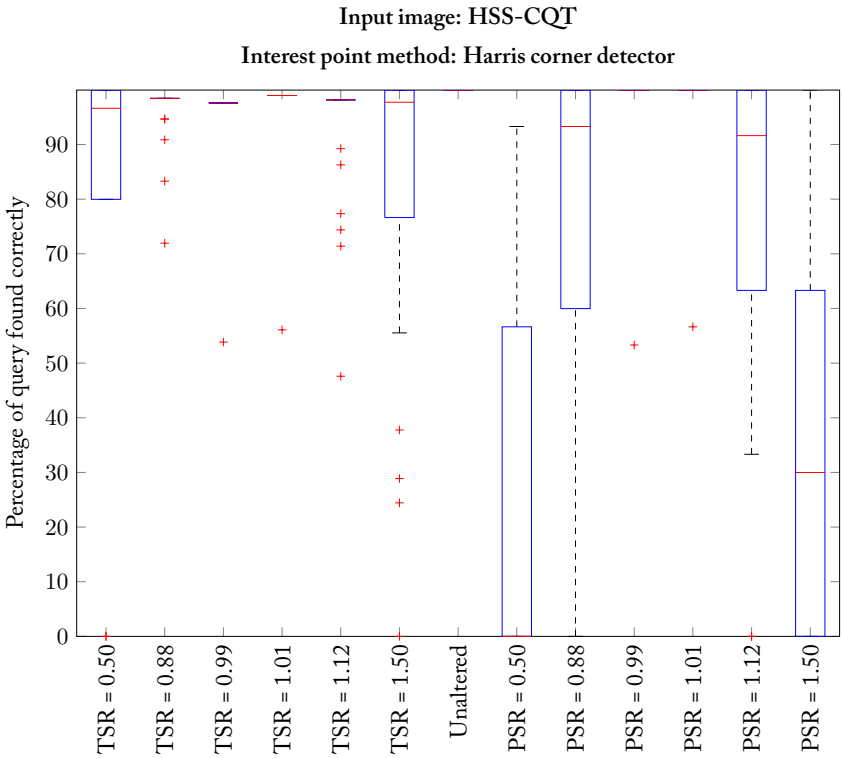


Figure D.13: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: Harris corner detector

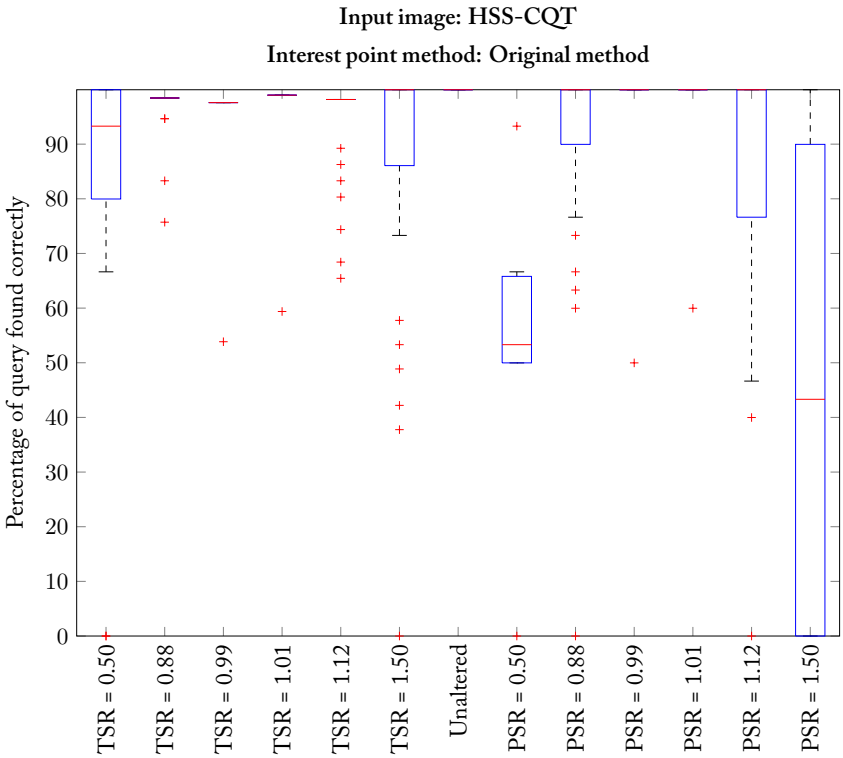


Figure D.14: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: Original method

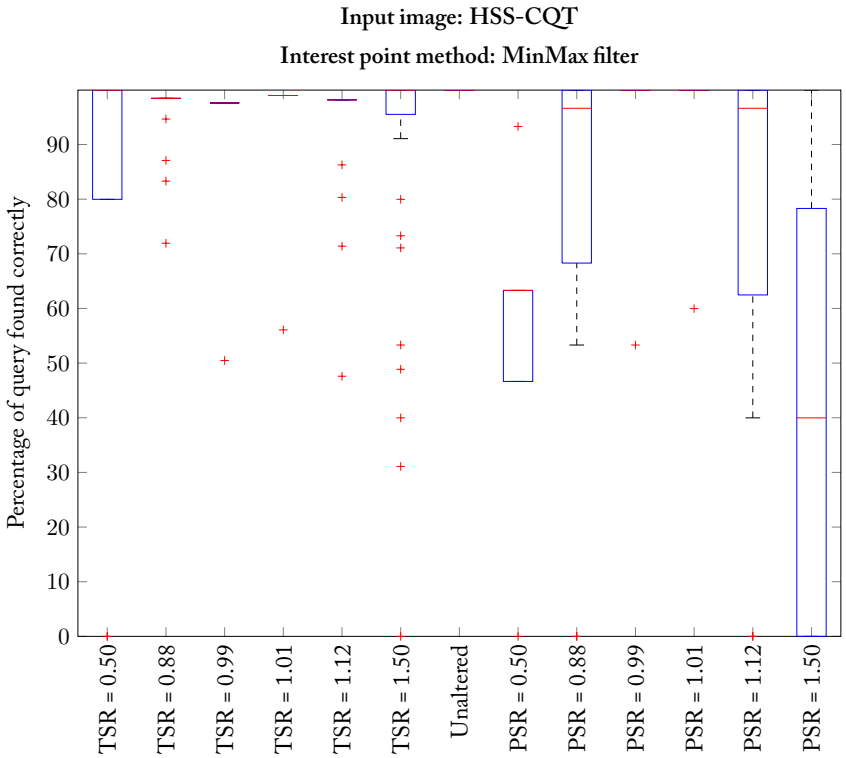


FIGURE D.15: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: MinMax filter

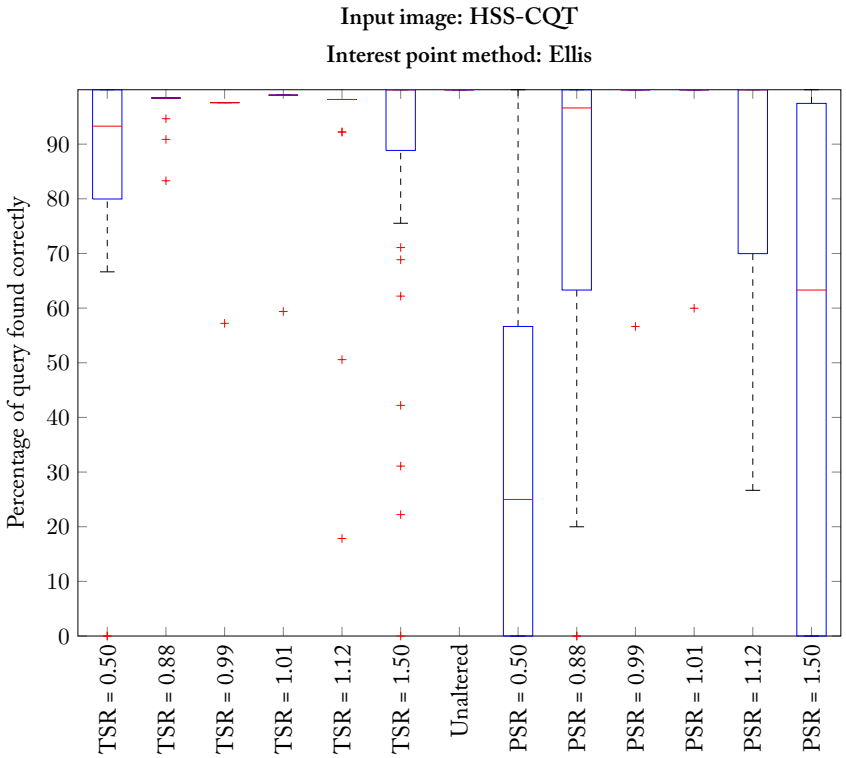


FIGURE D.16: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: Ellis

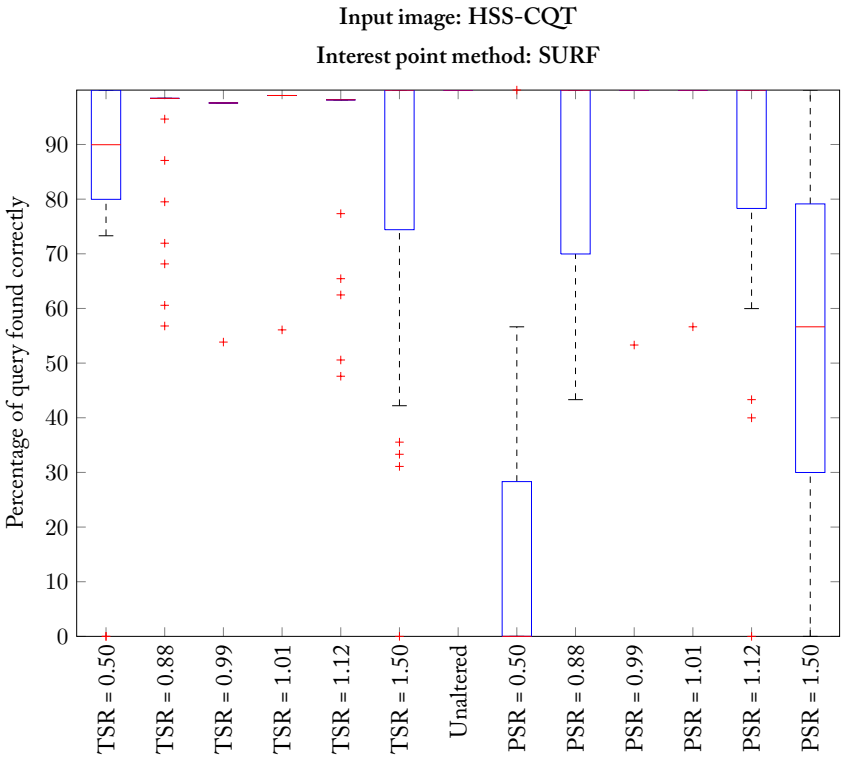


Figure D.17: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: SURF

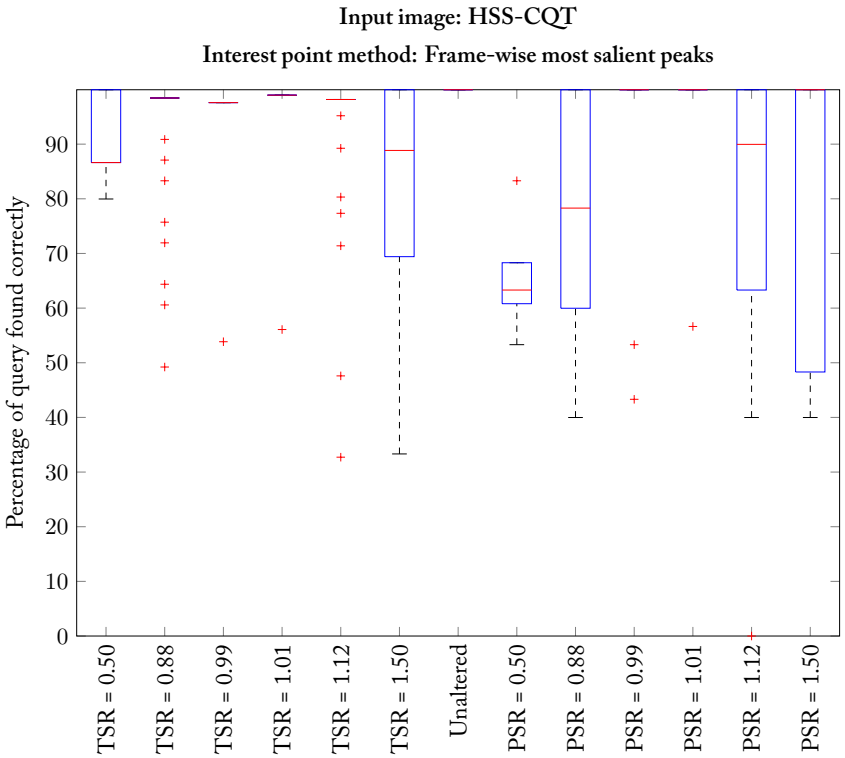


Figure D.18: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: Frame-wise most salient peaks

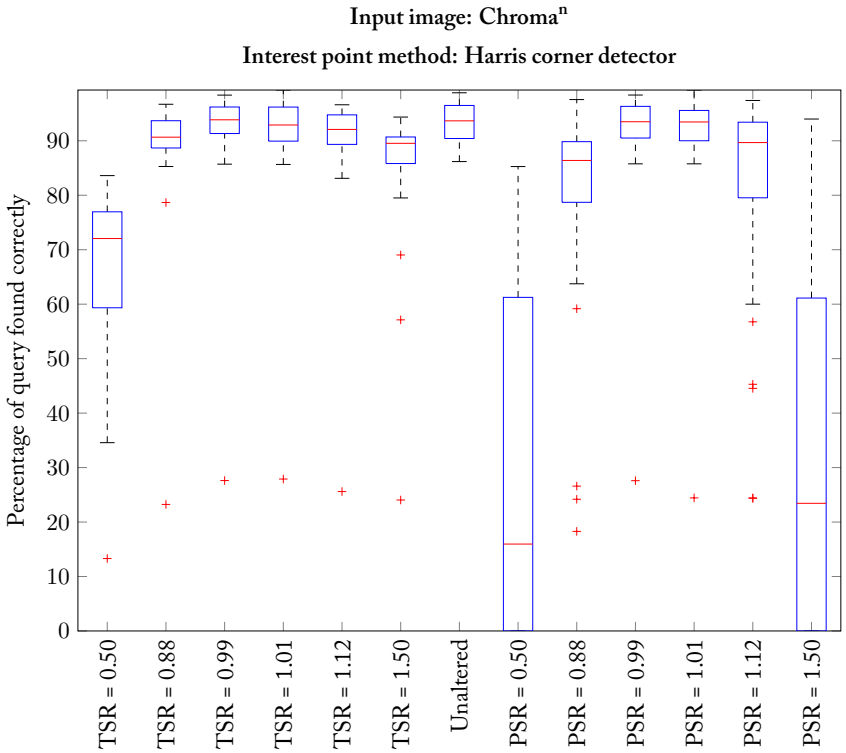


FIGURE D.19: Distributions of the retrieval ratios for the fine time slot estimation, input image: Chroma^n and interest point method: Harris corner detector

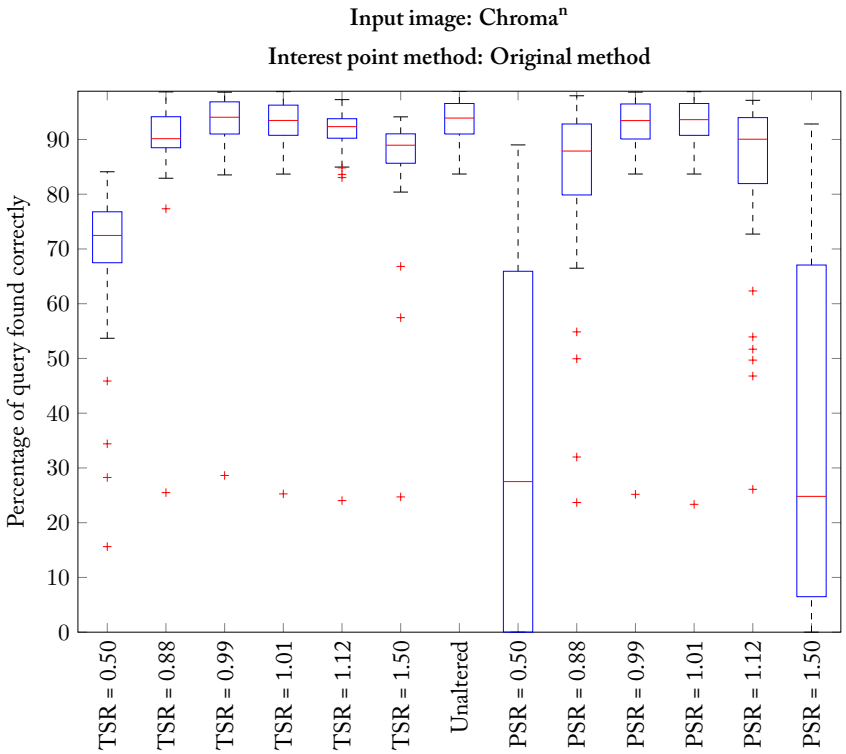


FIGURE D.20: Distributions of the retrieval ratios for the fine time slot estimation, input image: Chroma^n and interest point method: Original method

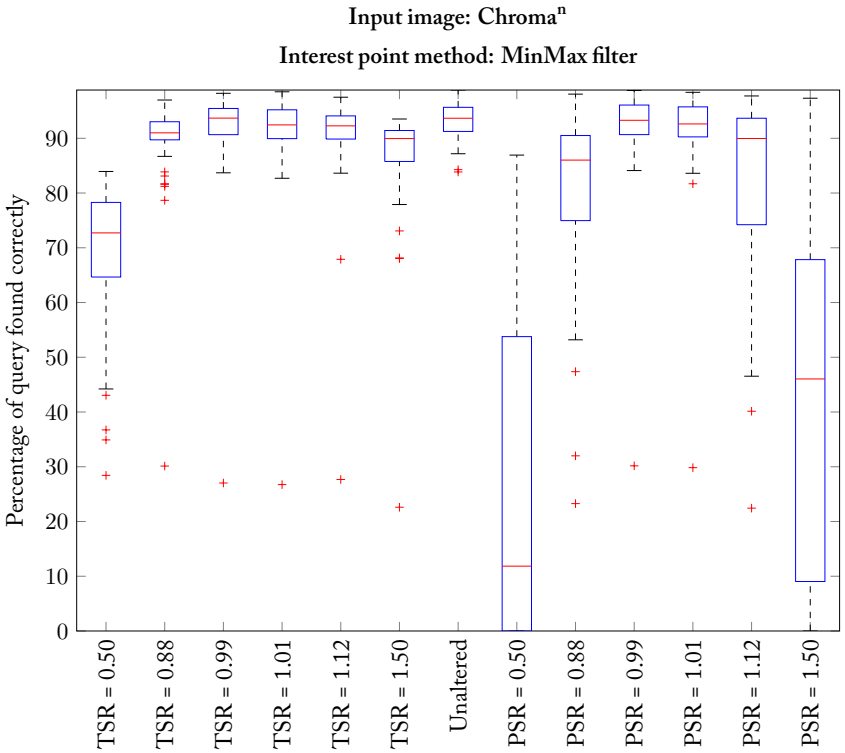


Figure D.21: Distributions of the retrieval ratios for the fine time slot estimation, input image: Chroma^n and interest point method: MinMax filter

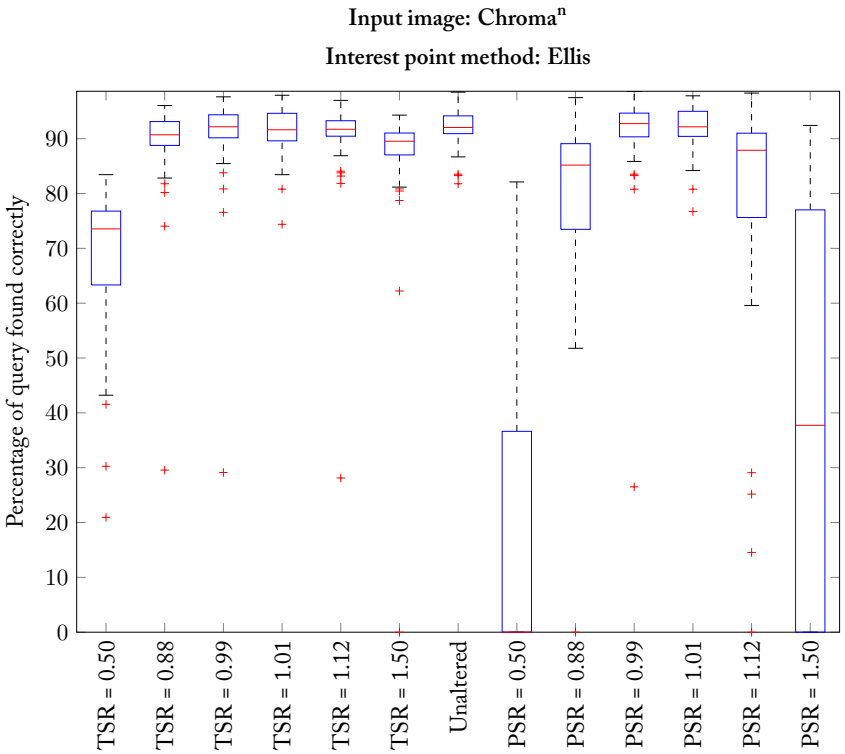


Figure D.22: Distributions of the retrieval ratios for the fine time slot estimation, input image: Chroma^n and interest point method: Ellis

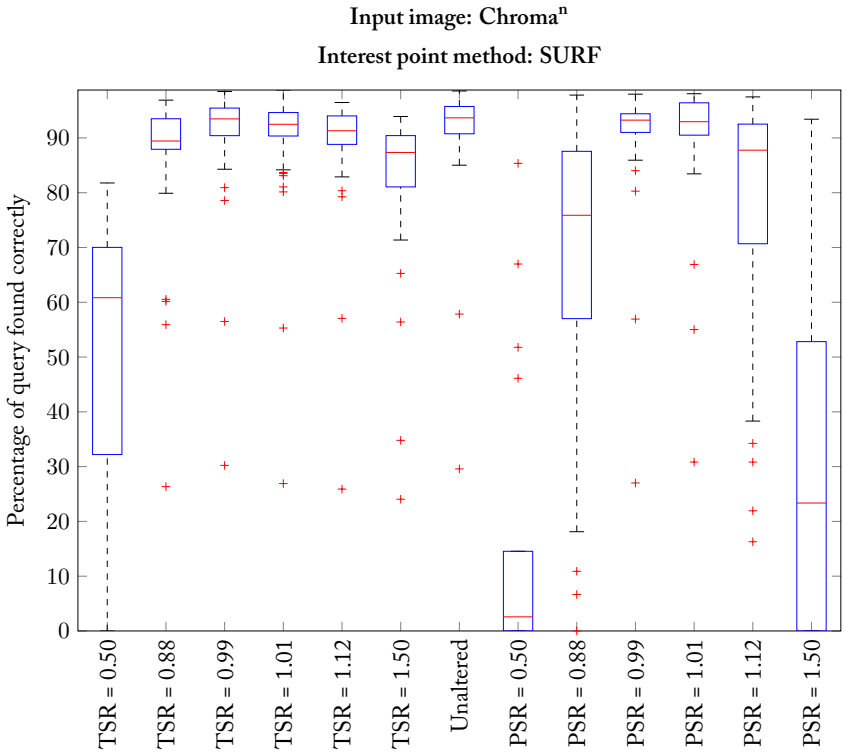


FIGURE D.23: Distributions of the retrieval ratios for the fine time slot estimation, input image: Chroma^n and interest point method: SURF

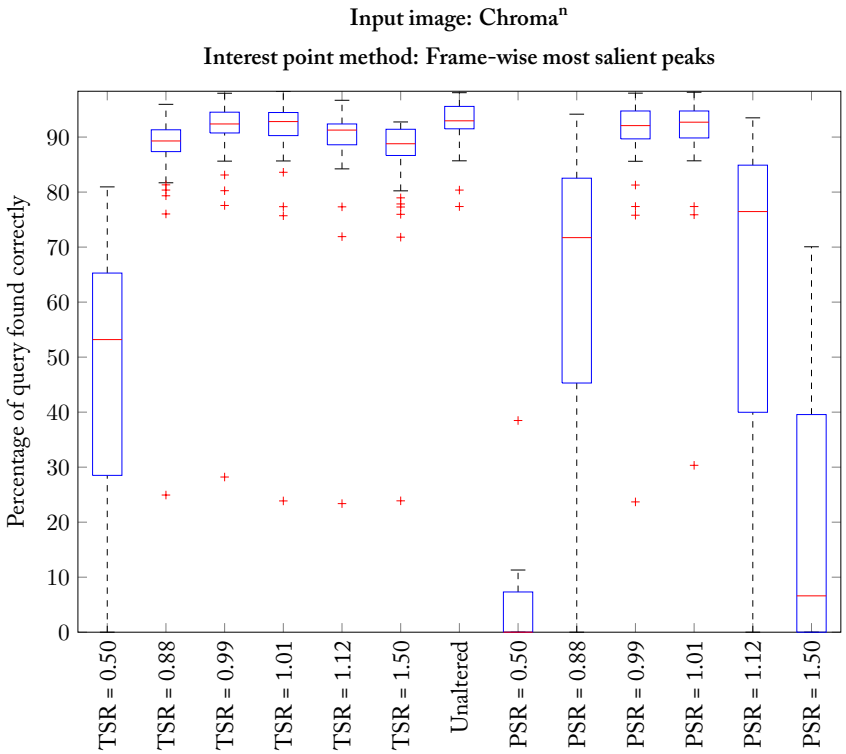


FIGURE D.24: Distributions of the retrieval ratios for the fine time slot estimation, input image: Chroma^n and interest point method: Frame-wise most salient peaks

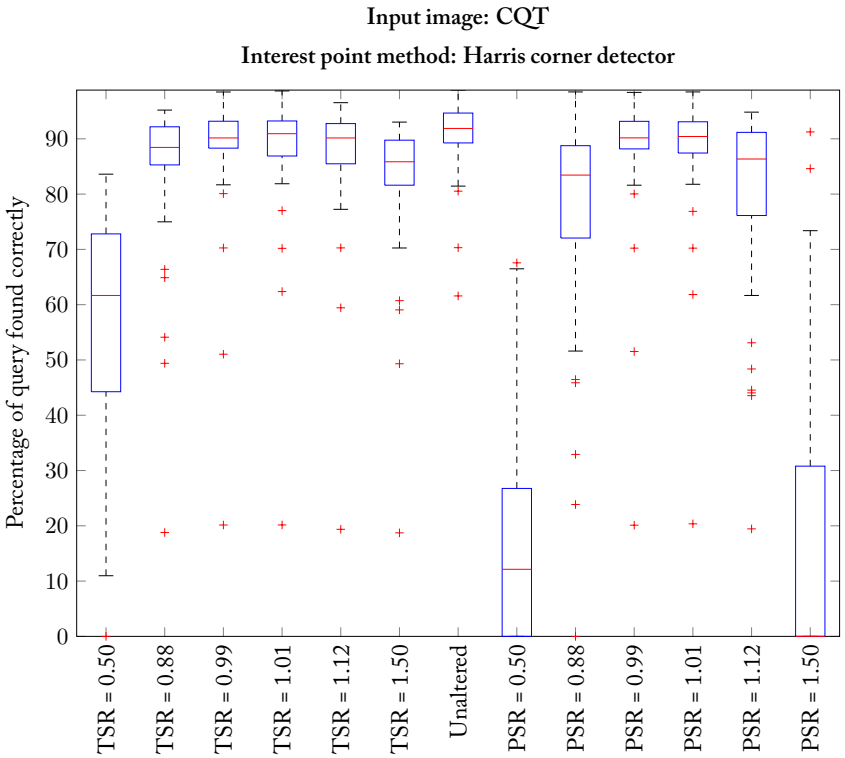


Figure D.25: Distributions of the retrieval ratios for the fine time slot estimation, input image: CQT and interest point method: Harris corner detector

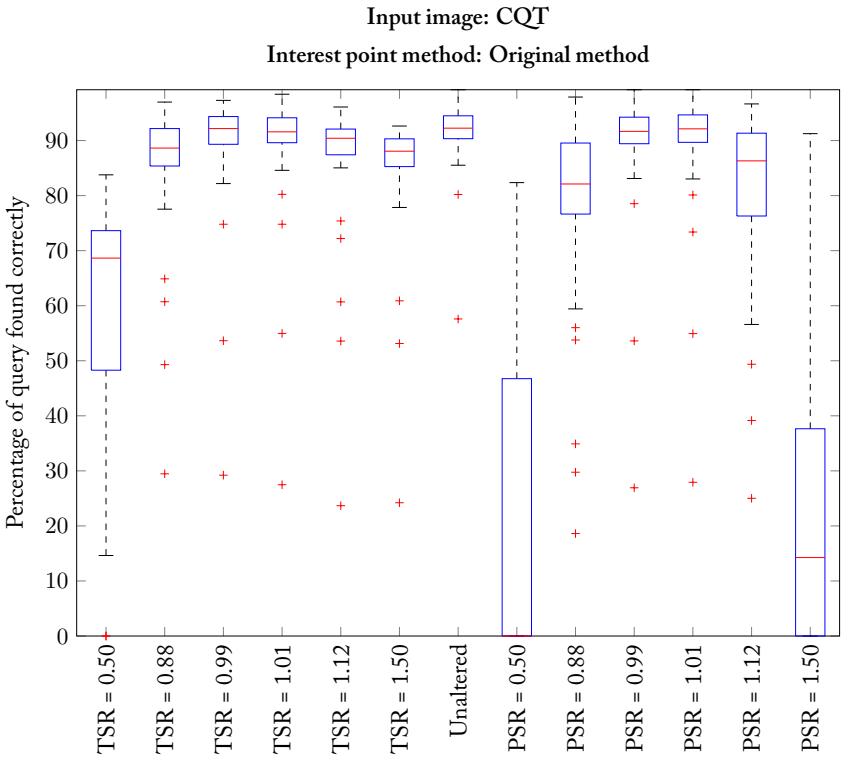


Figure D.26: Distributions of the retrieval ratios for the fine time slot estimation, input image: CQT and interest point method: Original method

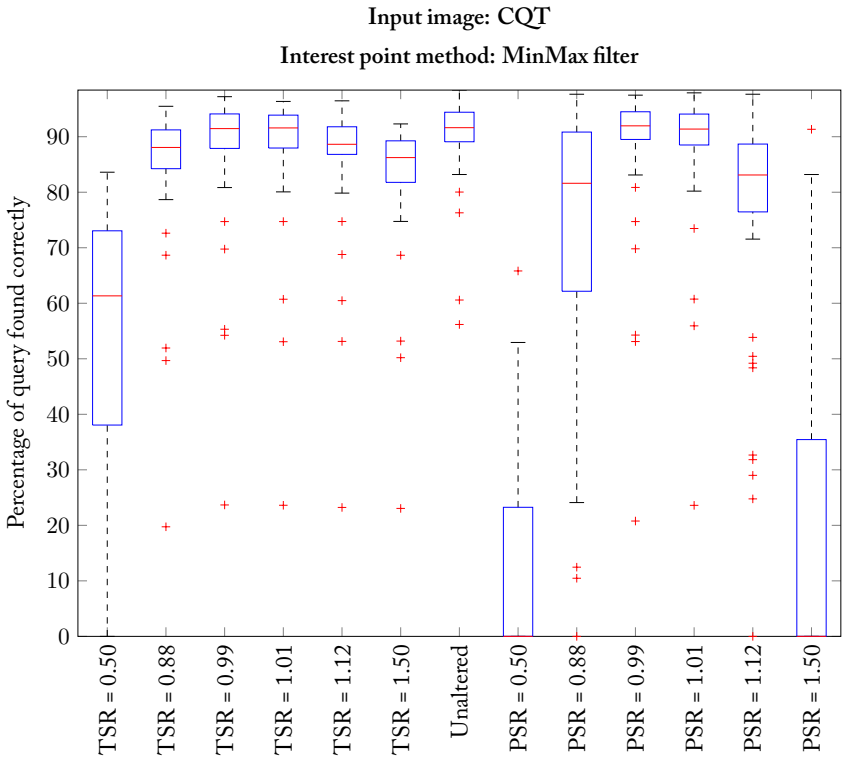


FIGURE D.27: Distributions of the retrieval ratios for the fine time slot estimation, input image: CQT and interest point method: MinMax filter

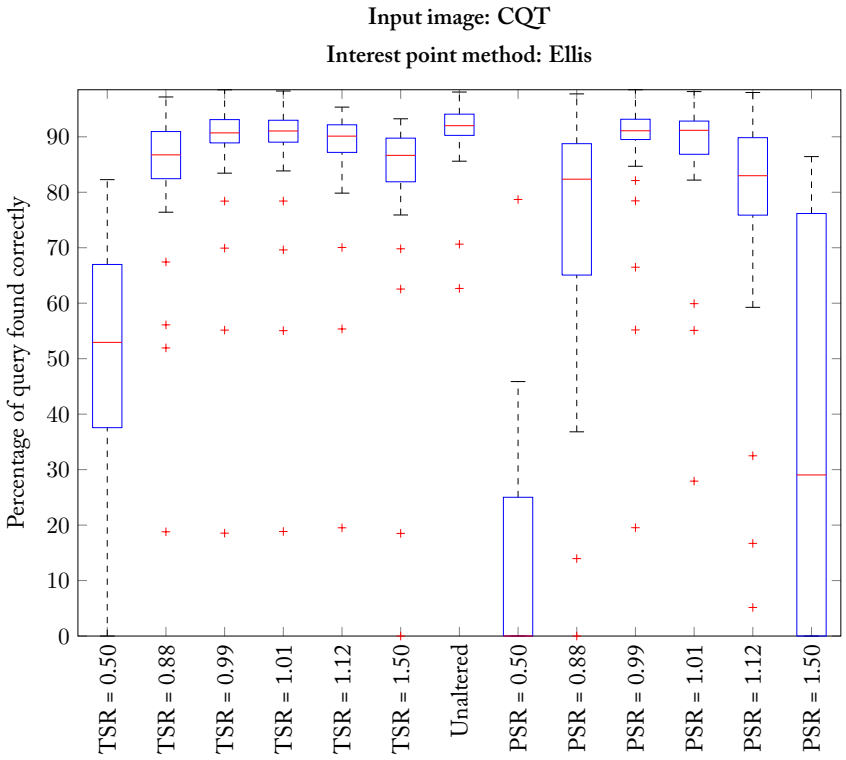


FIGURE D.28: Distributions of the retrieval ratios for the fine time slot estimation, input image: CQT and interest point method: Ellis

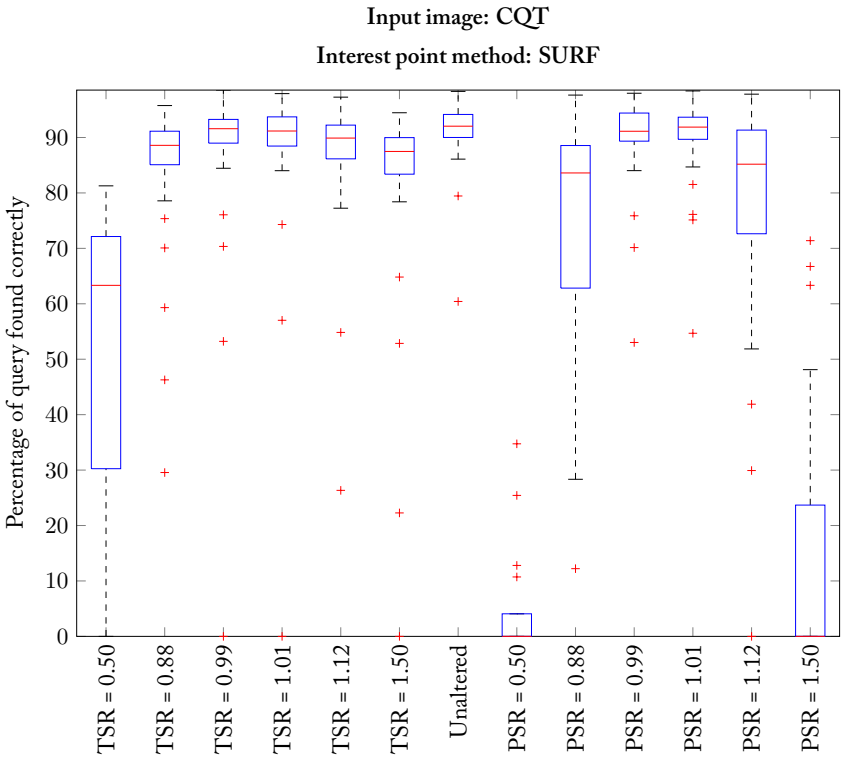


Figure D.29: Distributions of the retrieval ratios for the fine time slot estimation, input image: CQT and interest point method: SURF

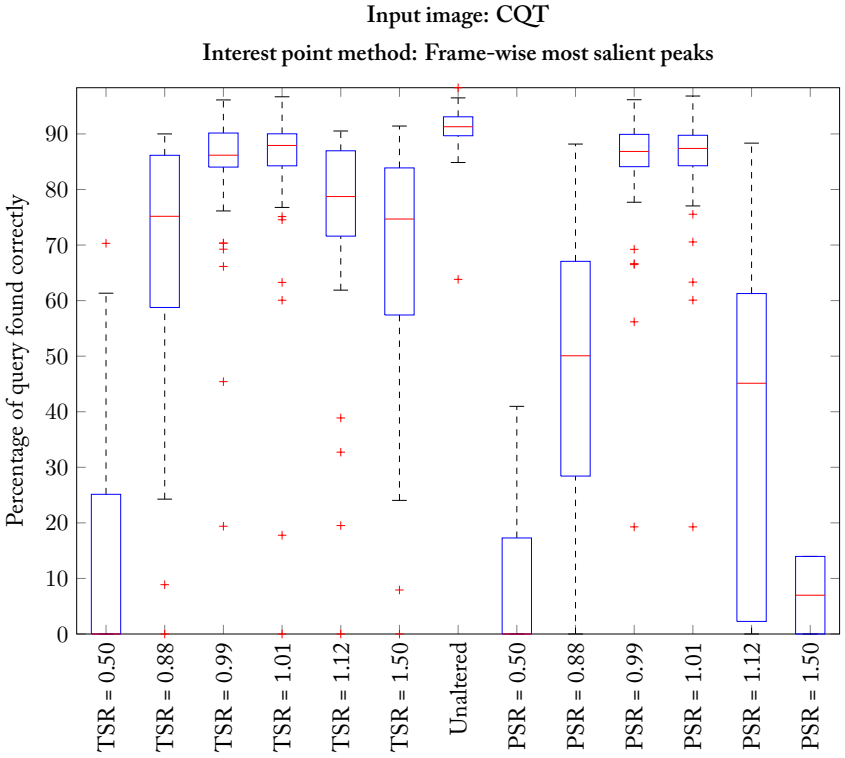


Figure D.30: Distributions of the retrieval ratios for the fine time slot estimation, input image: CQT and interest point method: Frame-wise most salient peaks

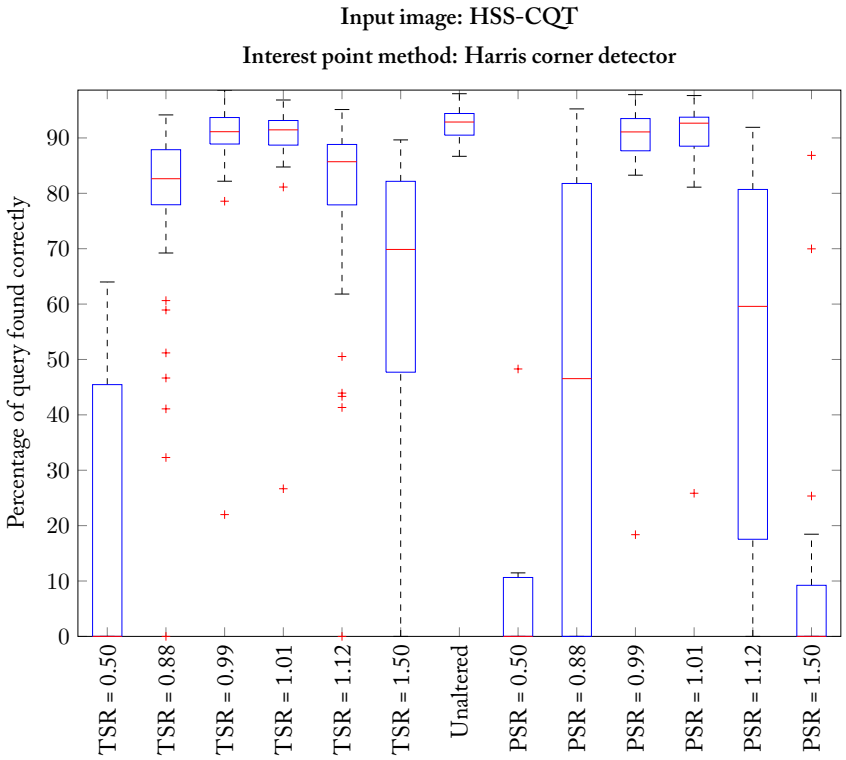


FIGURE D.31: Distributions of the retrieval ratios for the fine time slot estimation, input image: HSS-CQT and interest point method: Harris corner detector

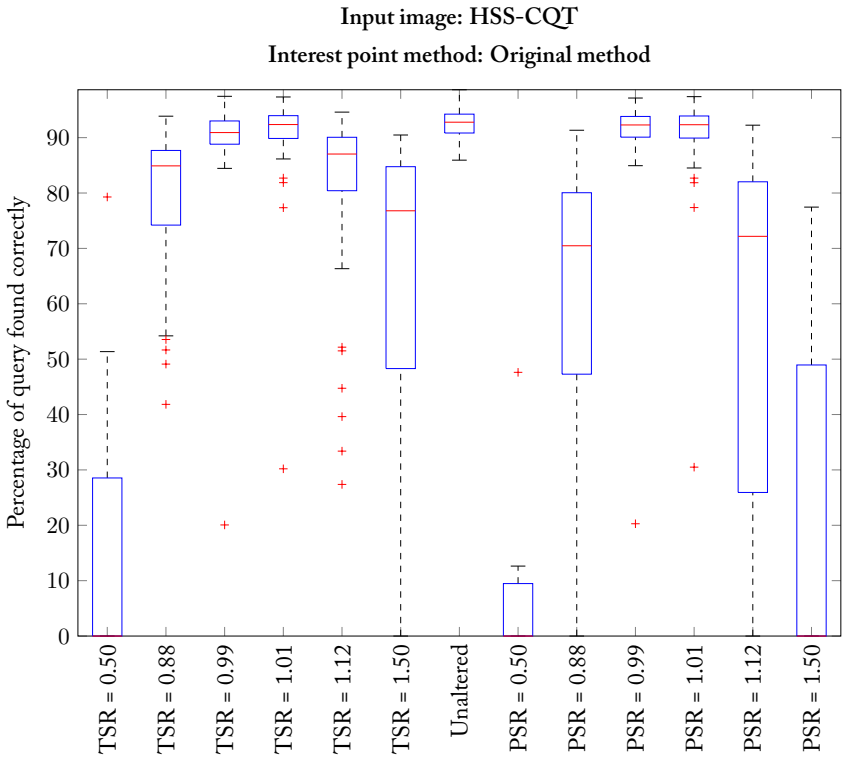


FIGURE D.32: Distributions of the retrieval ratios for the fine time slot estimation, input image: HSS-CQT and interest point method: Original method

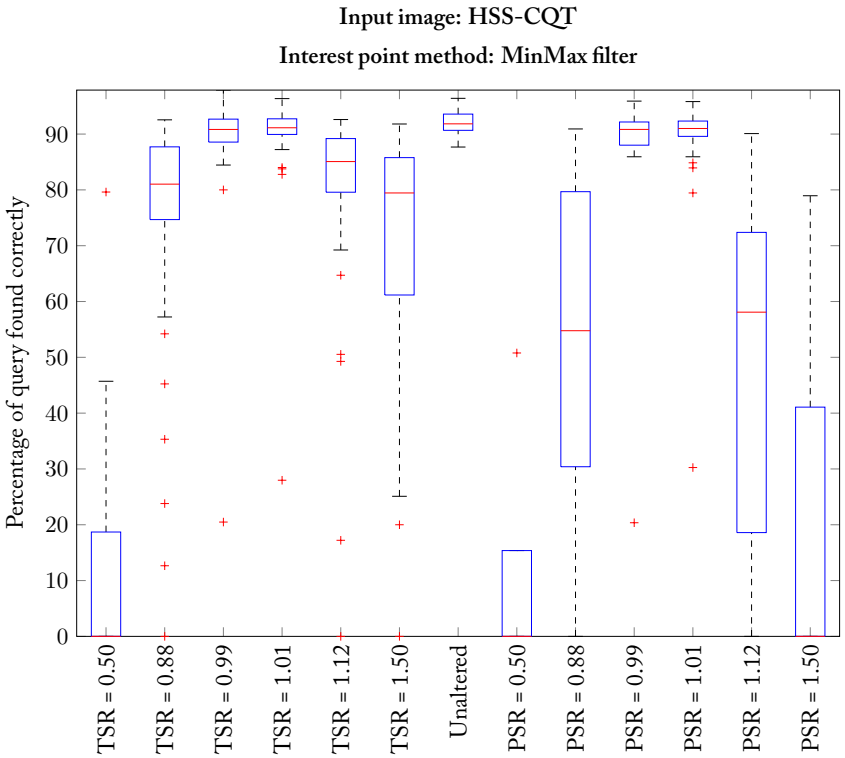


Figure D.33: Distributions of the retrieval ratios for the fine time slot estimation, input image: HSS-CQT and interest point method: MinMax filter

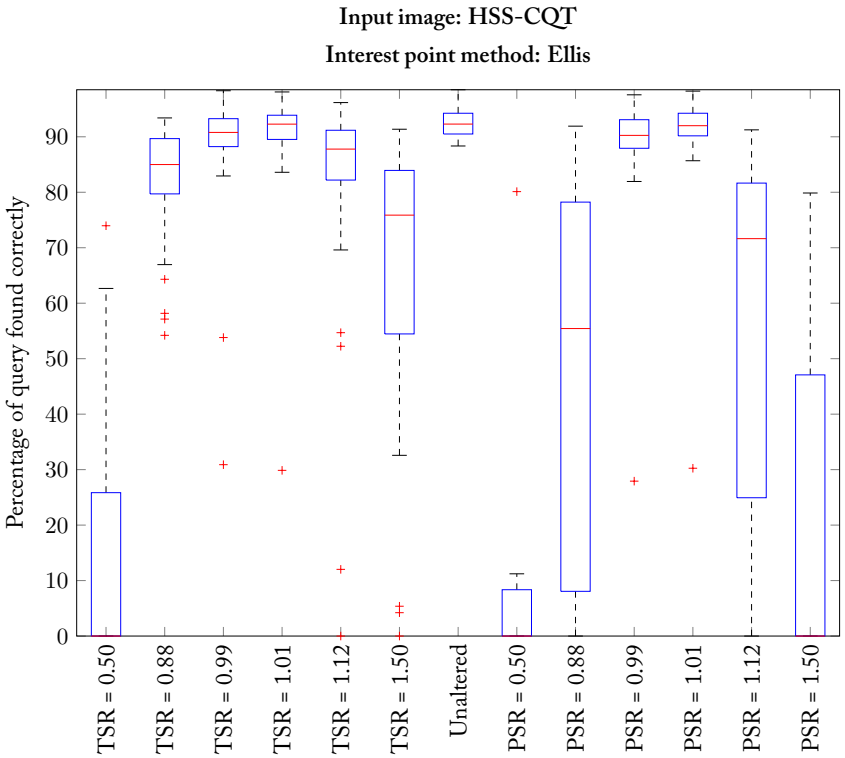


Figure D.34: Distributions of the retrieval ratios for the fine time slot estimation, input image: HSS-CQT and interest point method: Ellis

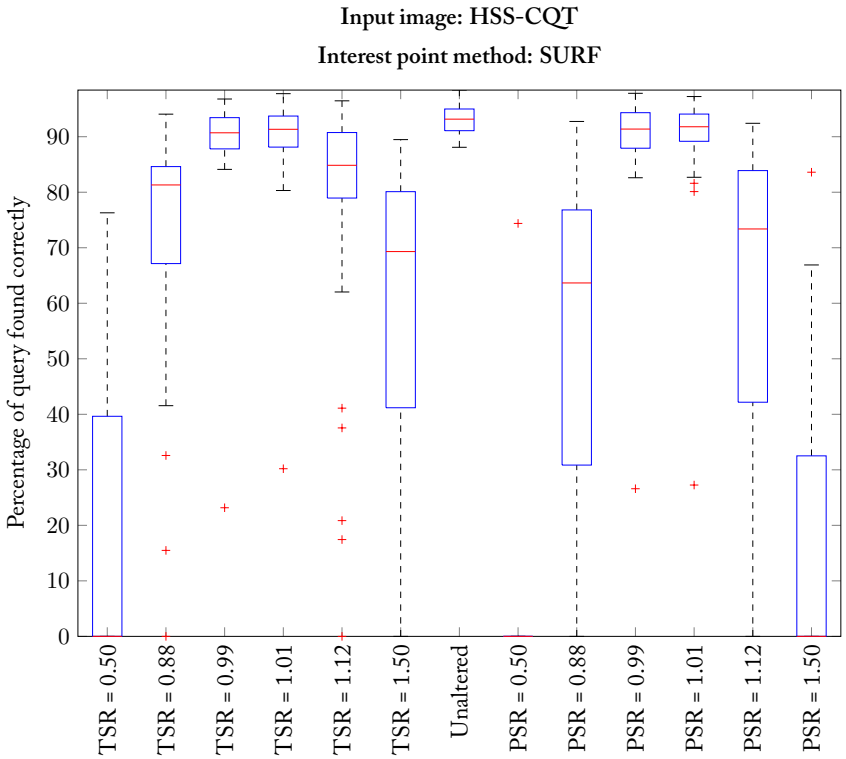


FIGURE D.35: Distributions of the retrieval ratios for the fine time slot estimation, input image: HSS-CQT and interest point method: SURF

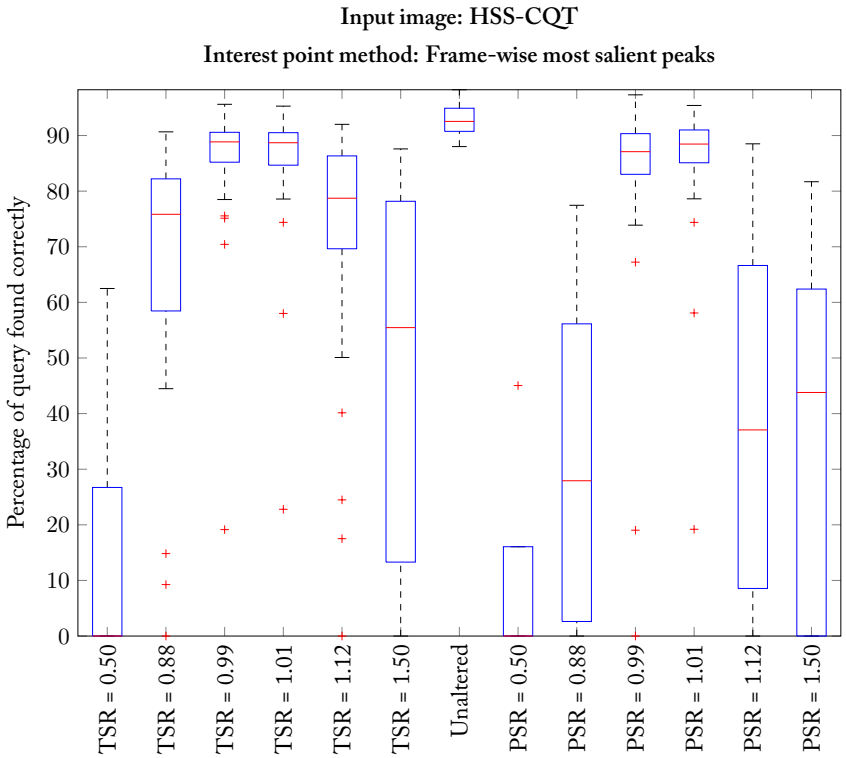


FIGURE D.36: Distributions of the retrieval ratios for the fine time slot estimation, input image: HSS-CQT and interest point method: Frame-wise most salient peaks

Results of the retrieval with standard degradations E

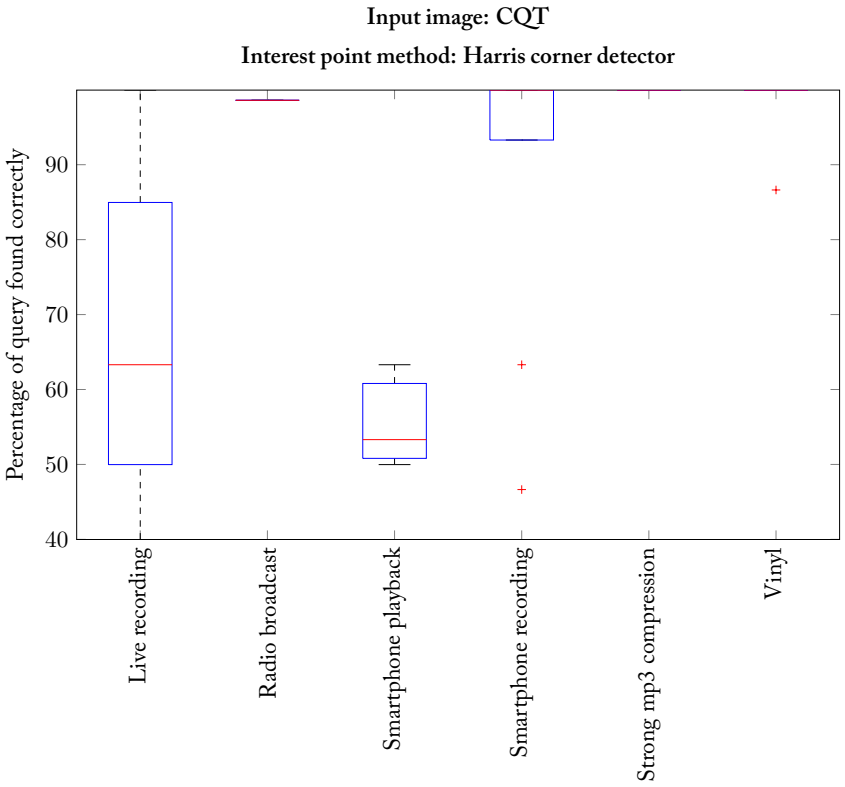


Figure E.1: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: Harris corner detector

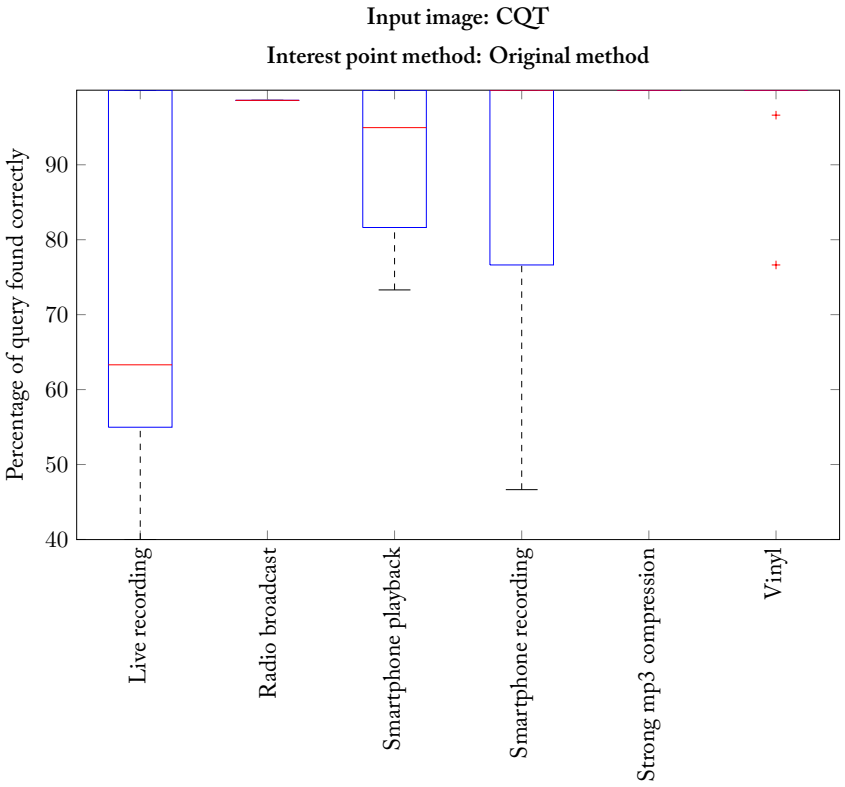


Figure E.2: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: Original method

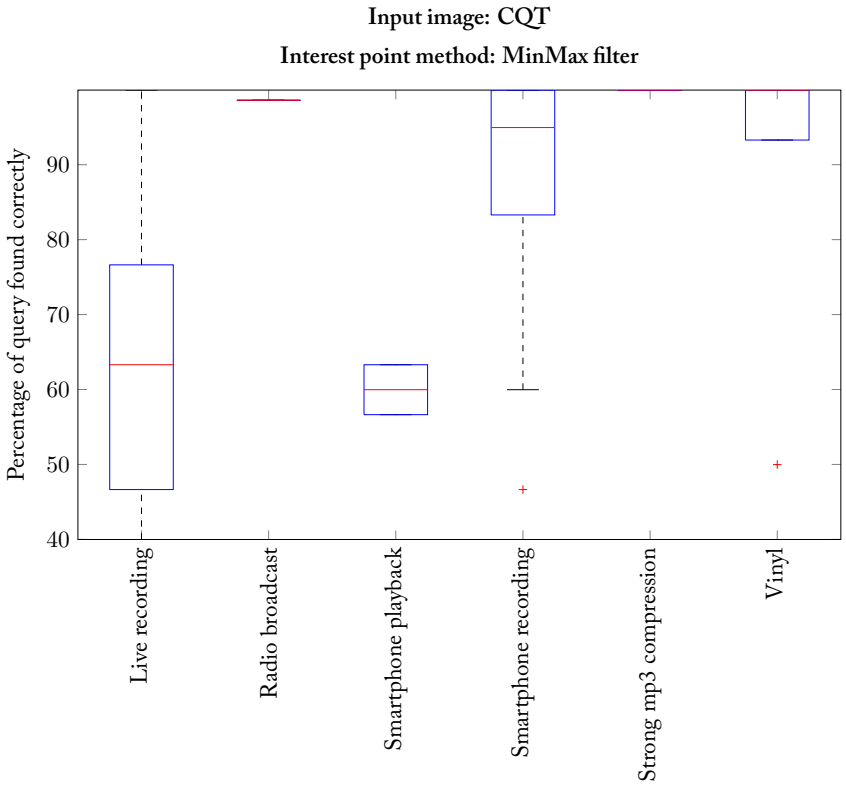


FIGURE E.3: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: MinMax filter

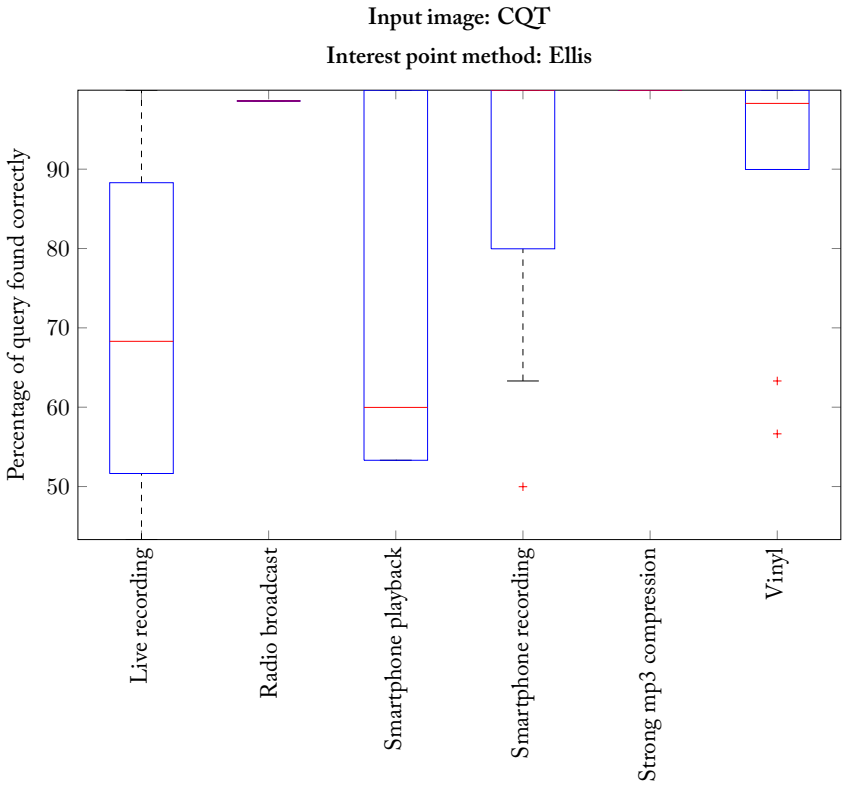


FIGURE E.4: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: Ellis

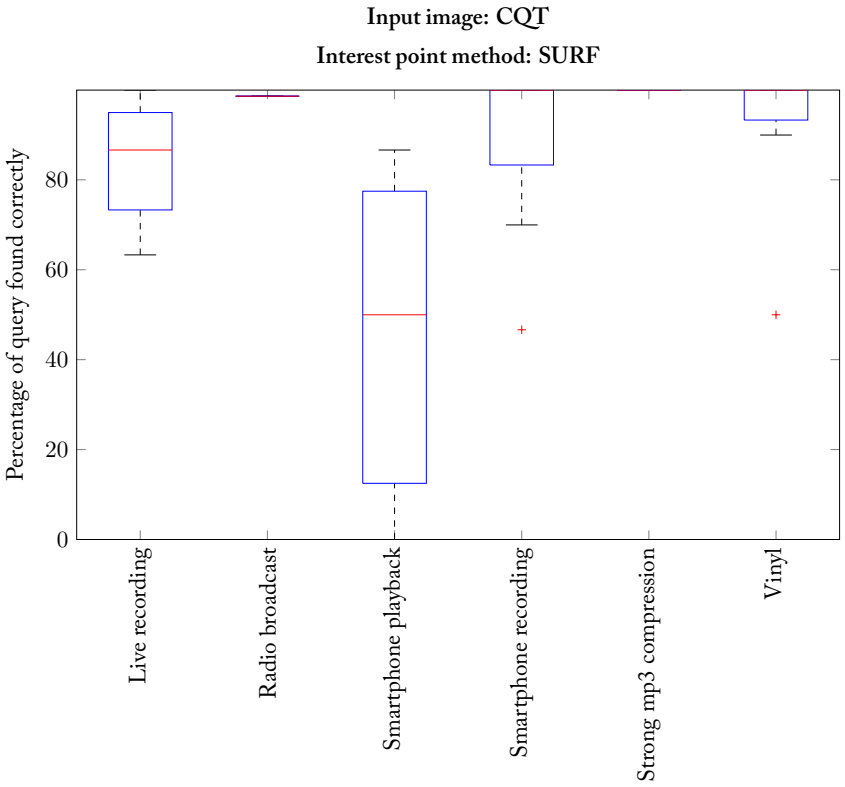


Figure E.5: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: SURF

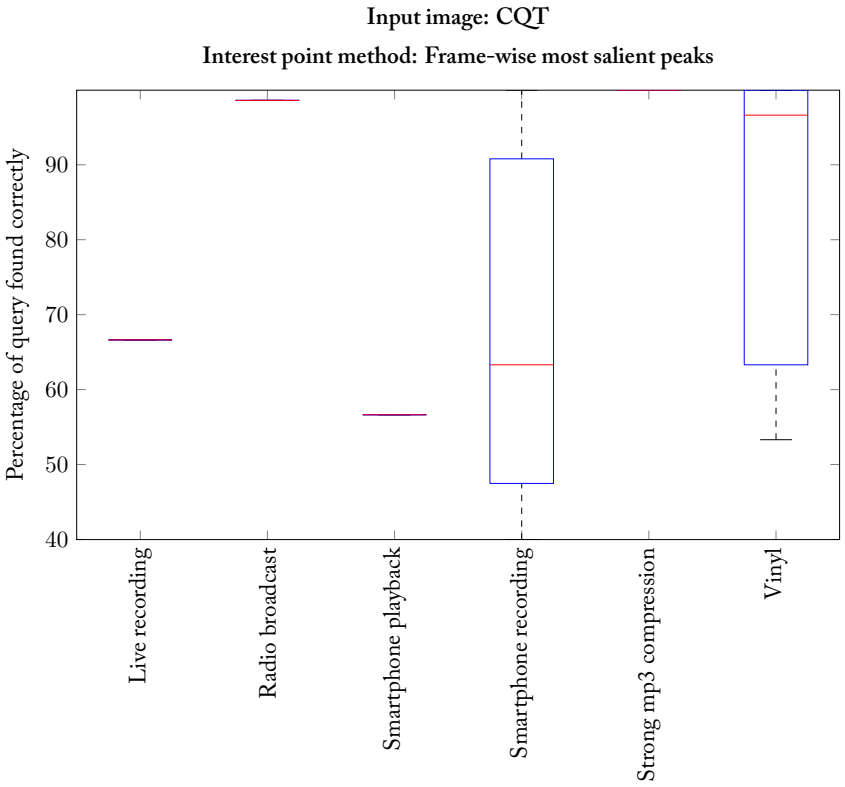


Figure E.6: Distributions of the retrieval ratios for the coarse time slot estimation, input image: CQT and interest point method: Frame-wise most salient peaks

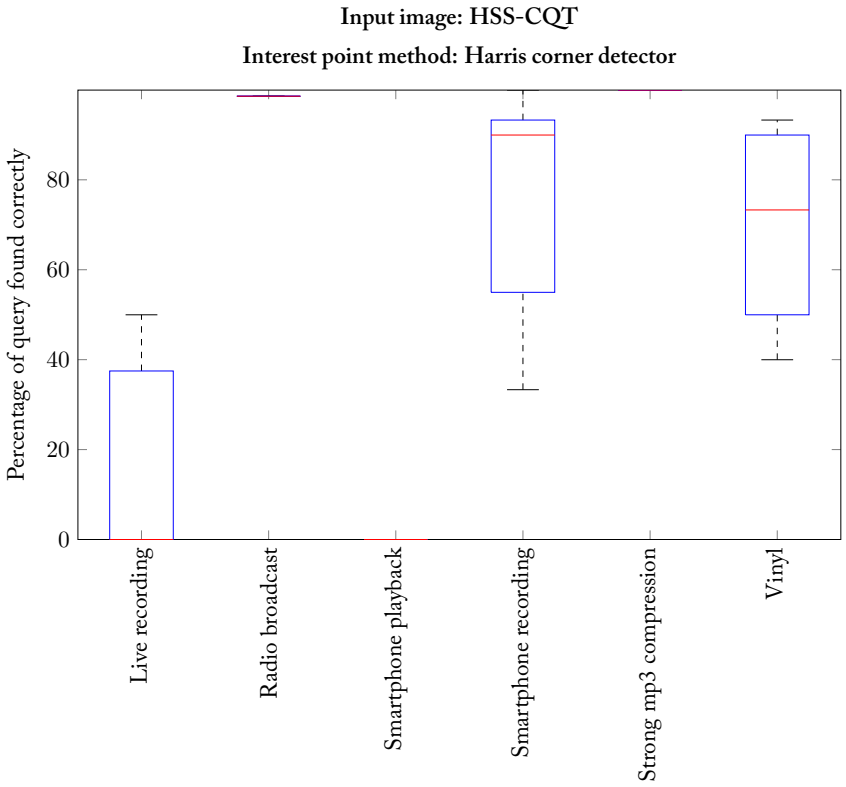


FIGURE E.7: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: Harris corner detector

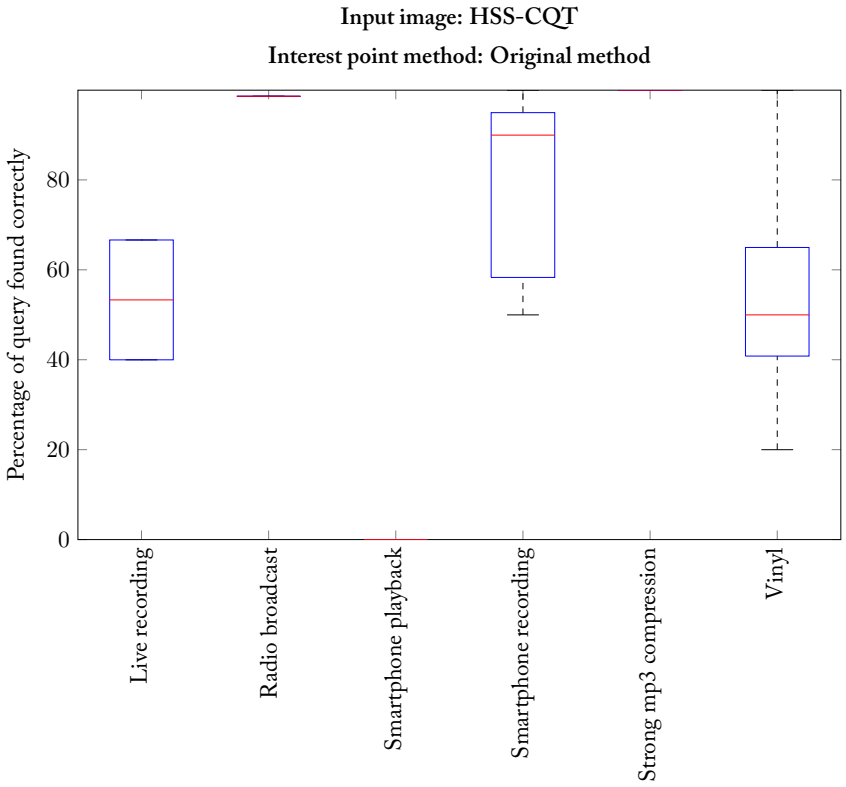


FIGURE E.8: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: Original method

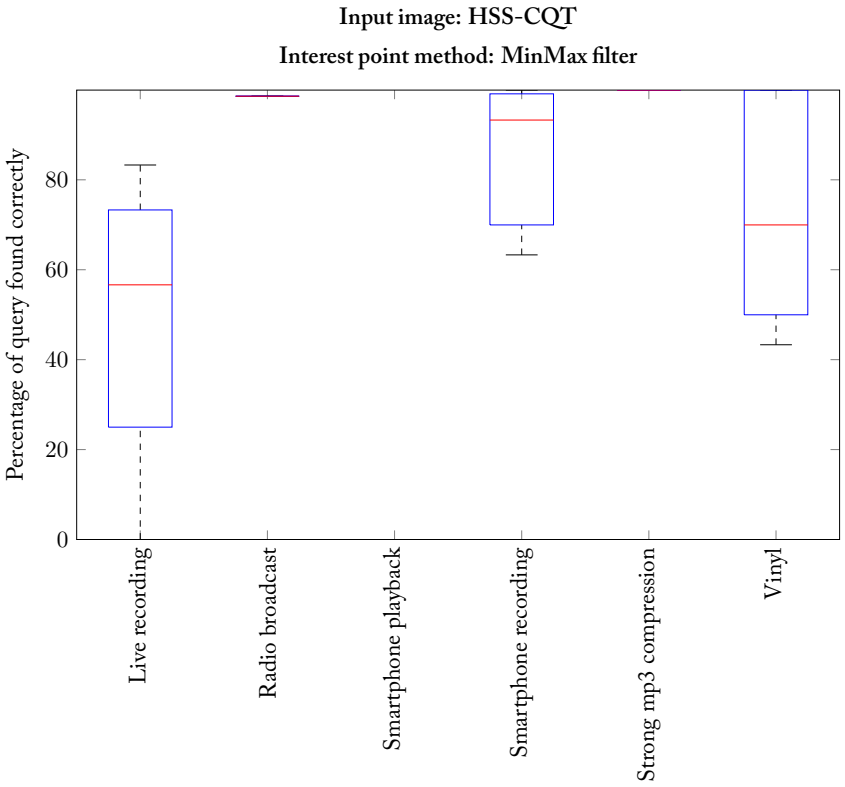


Figure E.9: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: MinMax filter

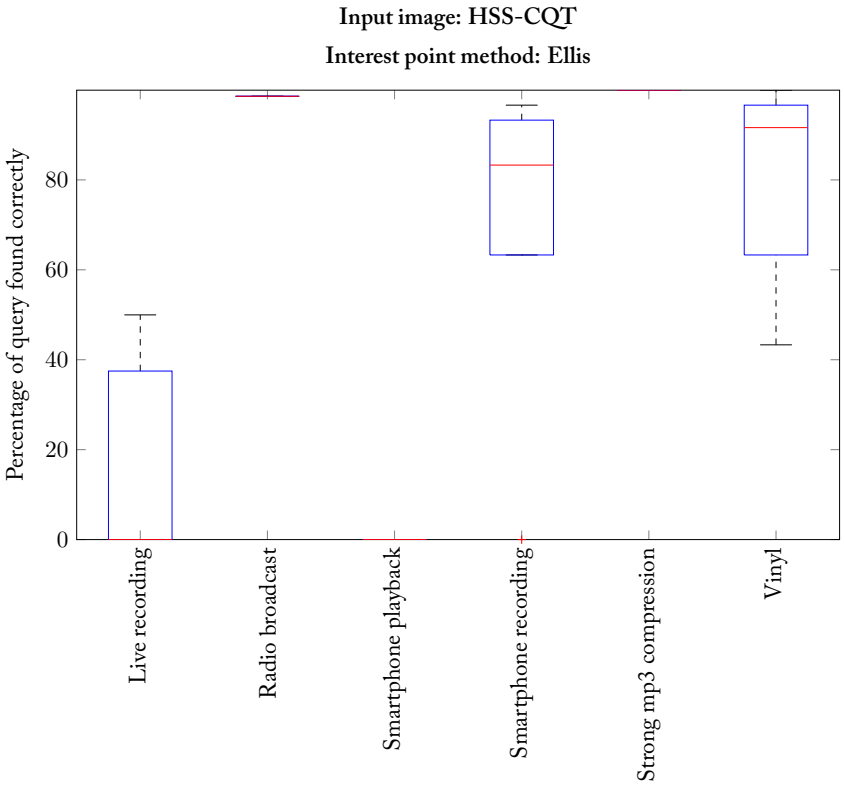


Figure E.10: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: Ellis

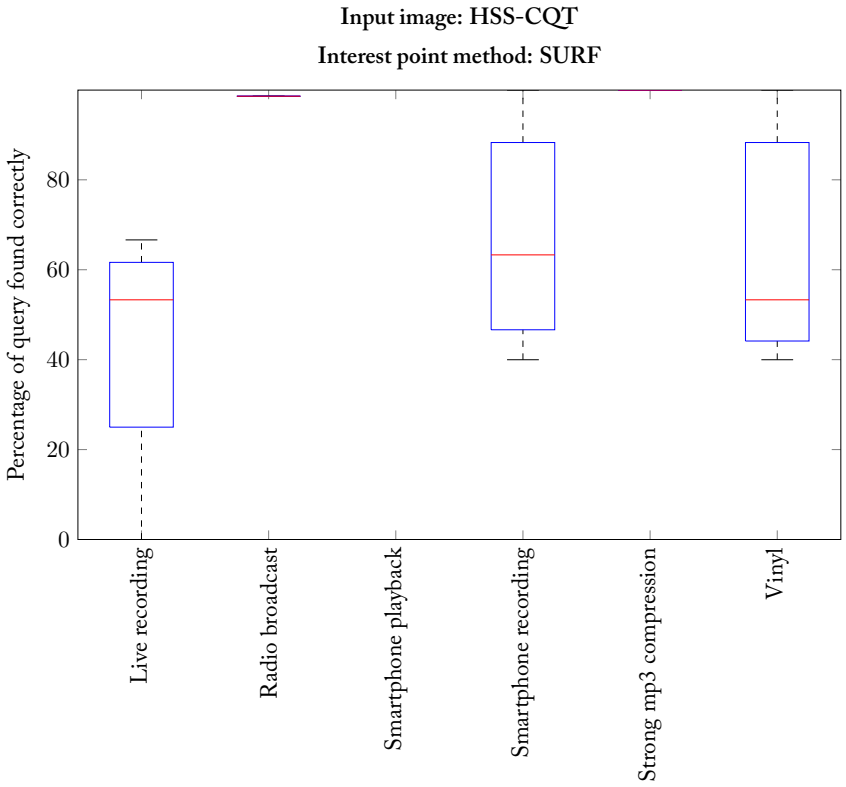


FIGURE E.11: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: SURF

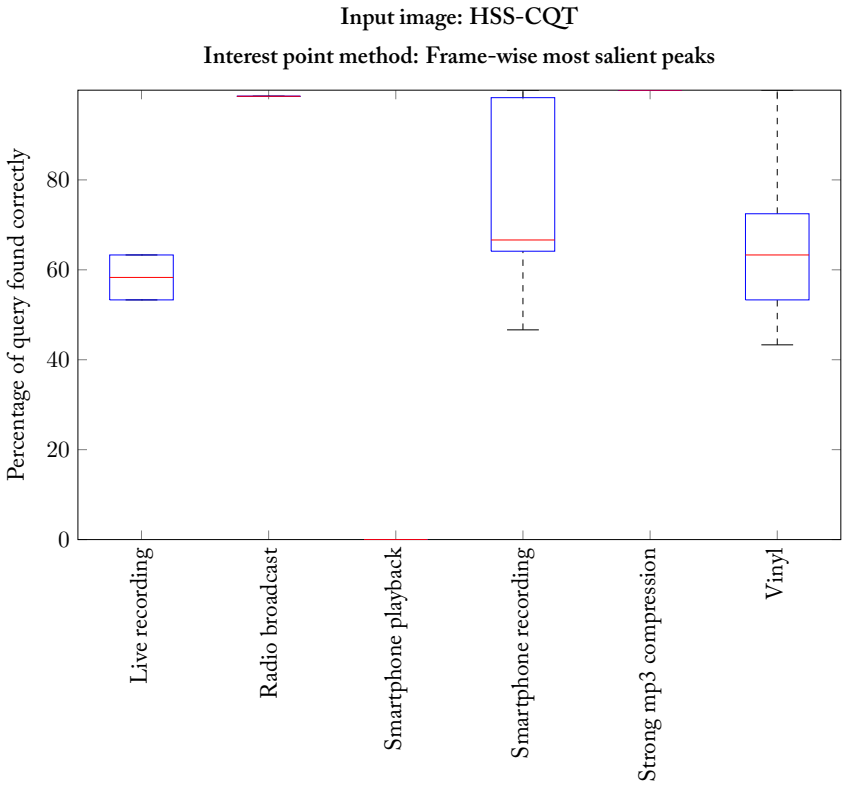


FIGURE E.12: Distributions of the retrieval ratios for the coarse time slot estimation, input image: HSS-CQT and interest point method: Frame-wise most salient peaks

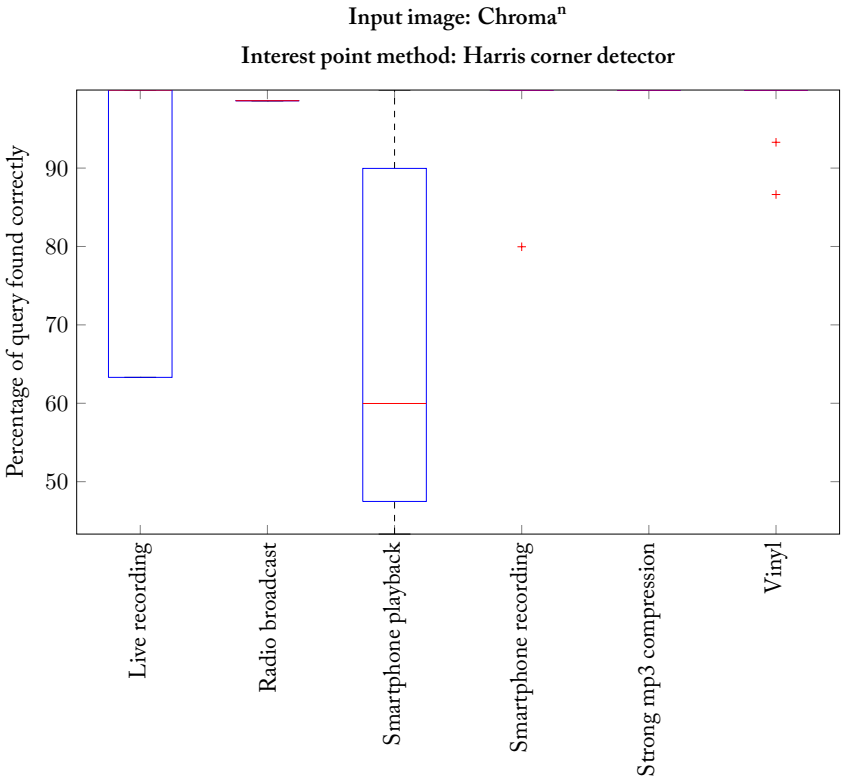


Figure E.13: Distributions of the retrieval ratios for the coarse time slot estimation, input image: chroma^n and interest point method: Harris corner detector

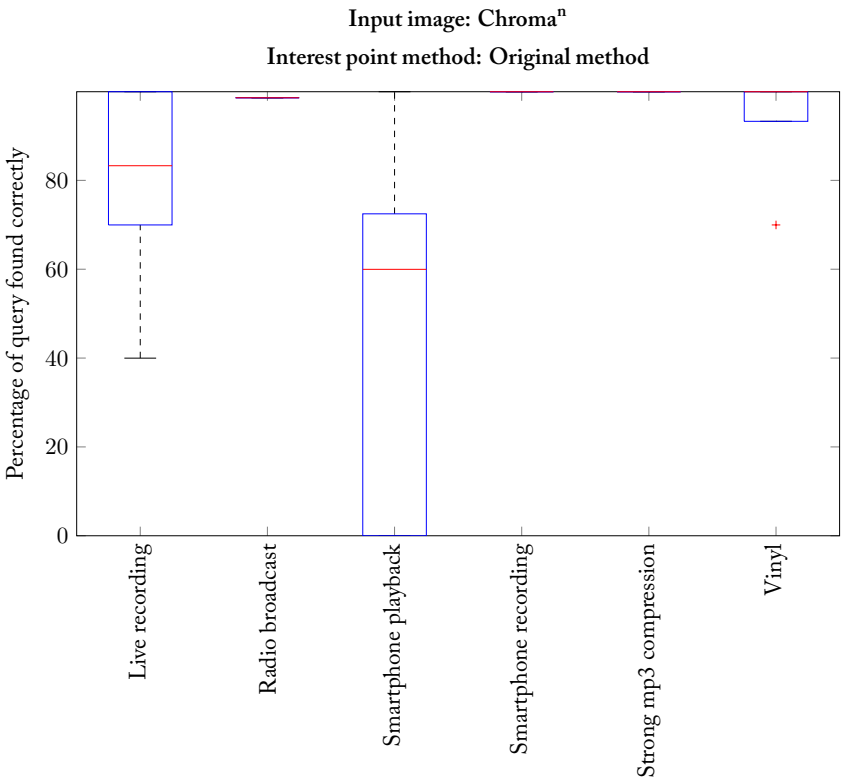


Figure E.14: Distributions of the retrieval ratios for the coarse time slot estimation, input image: chroma^n and interest point method: Original method

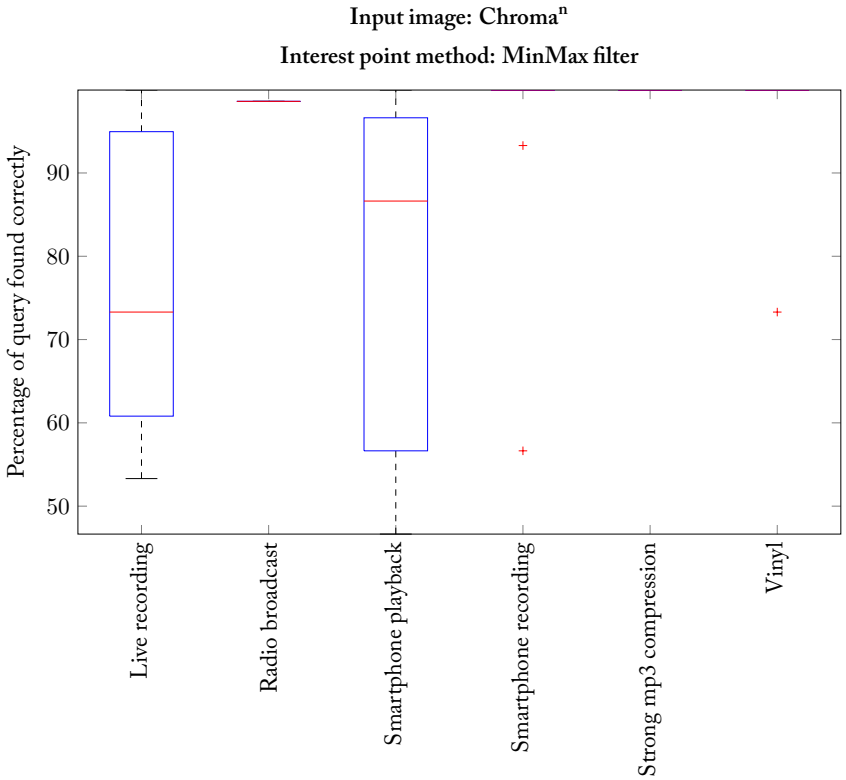


FIGURE E.15: Distributions of the retrieval ratios for the coarse time slot estimation, input image: chroma^n and interest point method: MinMax filter

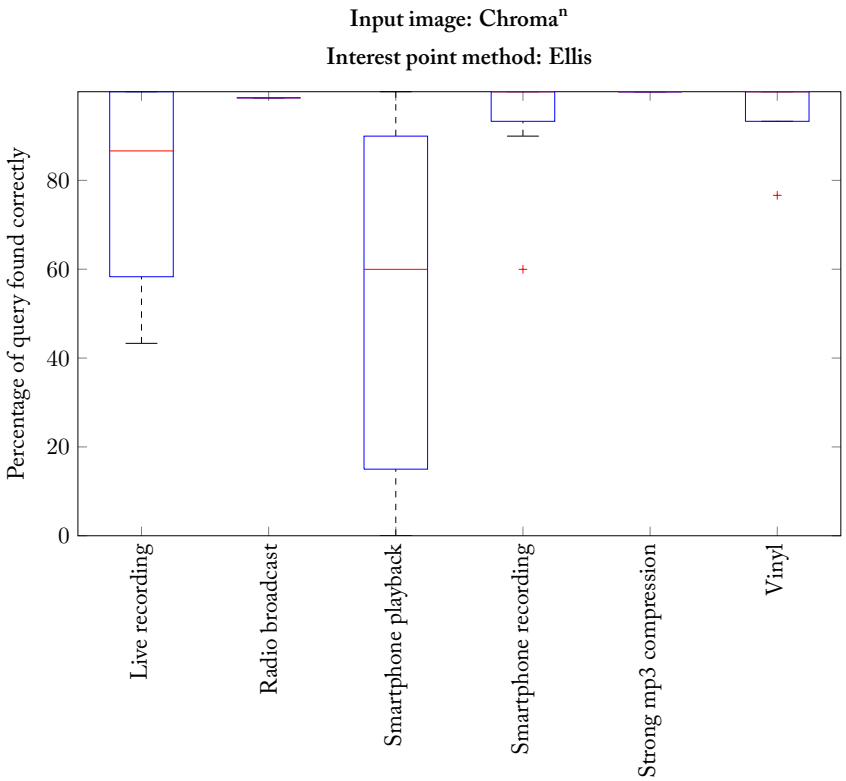


FIGURE E.16: Distributions of the retrieval ratios for the coarse time slot estimation, input image: chroma^n and interest point method: Ellis

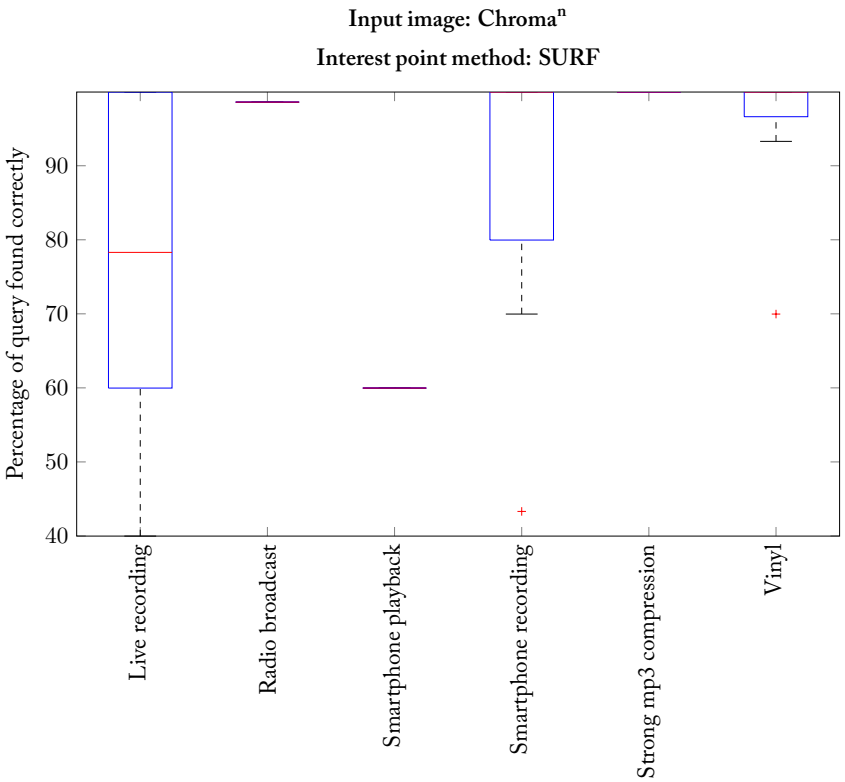


Figure E.17: Distributions of the retrieval ratios for the coarse time slot estimation, input image: chroma^n and interest point method: SURF

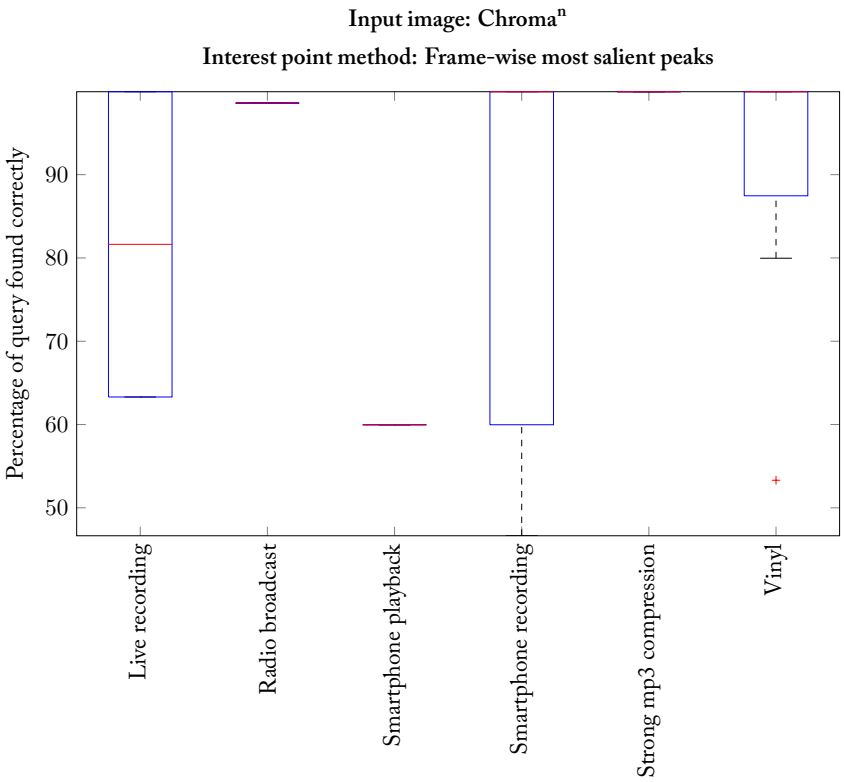


Figure E.18: Distributions of the retrieval ratios for the coarse time slot estimation, input image: chroma^n and interest point method: Frame-wise most salient peaks

Bibliography

Allamanche, E.; J. Herre; O. Hellmuth; and B. Froeba (2002): "Verfahren und Vorrichtung zum Charakterisieren eines Signals und Verfahren und Vorrichtung zum Erzeugen eines indexierten Signals." URL http://www.google.ch/patents/WO2002073592A2?cl=de. WO Patent App. PCT/EP2002/002,005.

Allamanche, Eric; et al. (2001): "Content-based Identification of Audio Material Using MPEG-7 Low Level Description." In: *Proceedings of the 2nd Annual International Symposium Music Information Retrieval (ISMIR)*. Bloomington, Indian, USA, pp. 197–204.

Alonso, Miguel; Gaël Richard; and Bertrand David (2005): "Extracting note onsets from musical recordings." In: *IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, pp. 4–8.

Andoni, Alexandr and Piotr Indyk (2008): "Near-optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions." In: *Commun. ACM*, **51**(1), pp. 117–122. doi:10.1145/1327452.1327494. URL http://doi.acm.org/10.1145/1327452.1327494.

Baluja, S. and M. Covell (2013): "Audio identification using wavelet-based signatures." URL https://www.google.com/patents/US8411977. US Patent 8,411,977.

Baluja, Shumeet and Michele Covell (2006): "Content fingerprinting using wavelets." In: *IET Conference Proceedings*, pp. 198–207. URL http://digital-library.theiet.org/content/conferences/10.1049/cp_20061964.

Bardeli, Rolf and Frank Kurth (2004): "Robust identification of time-scaled audio." In: *Proceedings of the AES 25th International Conference on Metadata for Audio*. Audio Engineering Society, pp. 1–12.

Bay, Herbert; Andreas Ess; Tinne Tuytelaars; and Luc Van Gool (2008): "Speeded-up robust features (SURF)." In: *Computer Vision and Image Understanding (CVIU)*, **110**(3), pp. 346–359.

Bay, Herbert; Tinne Tuytelaars; and Luc Van Gool (2006): "Surf: Speeded up robust features." In: *Proceedings of the 9th European Conference on Computer Vision (ECCV)*. Springer, pp. 404–417.

Brown, Judith C (1991): "Calculation of a constant Q spectral transform." In: *The Journal of the Acoustical Society of America*, **89**(1), pp. 425–434.

Brown, Judith C and Miller S Puckette (1992): "An efficient algorithm for the calculation of a constant Q transform." In: *The Journal of the Acoustical Society of America*, **92**(5), pp. 2698–2701.

Brown, Matthew and David G Lowe (2002): "Invariant Features from Interest Point Groups." In: *Proceedings of the 19th British Machine Vision Conference (BMVC)*. pp. 253–262.

Burges, Christopher JC; John C Platt; and Soumya Jana (2003): "Distortion discriminant analysis for audio fingerprinting." In: *IEEE Transactions on Speech and Audio Processing*, **11**(3), pp. 165–174.

Cano, Pedro; E Batle; Ton Kalker; and Jaap Haitsma (2002a): "A review of algorithms for audio fingerprinting." In: *Proceedings of the 2002 IEEE Workshop on Multimedia Signal Processing*. IEEE, pp. 169–173.

Cano, Pedro; Eloi Batlle; Harald Mayer; and Helmut Neuschmied (2002b): "Robust sound modeling for song detection in broadcast audio." In: *Proceedings of the 112th Audio Engineering Society International Convention*, pp. 1–7.

Chandrasekhar, Vijay; Matt Sharifi; and David A Ross (2011): "Survey and Evaluation of Audio Fingerprinting Schemes for Mobile Query-by-Example Applications." In: *Proceedings of the 12th International Society for Music Information Retrieval (ISMIR) Conference*, vol. 20. pp. 801–806.

Charikar, Moses S (2002): "Similarity estimation techniques from rounding algorithms." In: *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. ACM, pp. 380–388.

Chauhan, Sanjay Pratap Singh and SAM Rizvi (2013): "A survey: Digital audio watermarking techniques and applications." In: *Proceedings of the 4th International Conference on Computer and Communication Technology (ICCCT)*. IEEE, pp. 185–192.

Coover, Bob and Jinyu Han (2014): "A Power Mask based audio fingerprint." In: *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, pp. 1394–1398.

Cremer, Markus; Bernhard Froba; Oliver Hellmuth; Jurgen Herre; and Eric Allamanche (2001): "AudioID: Towards Content-Based Identification of Audio Material." In: *Audio Engineering Society Convention 110*. Audio Engineering Society.

Crow, Franklin C (1984): "Summed-area tables for texture mapping." In: *ACM SIGGRAPH computer graphics*, **18**(3), pp. 207–212.

Deng, Jijun; Wanggen Wan; XiaoQing Yu; and Wei Yang (2011): "Audio fingerprinting based on spectral energy structure and NMF." In: *Proceedings of the 13th IEEE International Conference on Communication Technology (ICCT)*. IEEE, pp. 1103–1106.

Duda, Richard O and Peter E Hart (1972): "Use of the Hough transformation to detect lines and curves in pictures." In: *Communications of the ACM*, **15**(1), pp. 11–15.

Ellis, Dan (2009): "Robust Landmark-Based Audio Fingerprinting." URL `http://labrosa.ee.columbia.edu/matlab/fingerprint/`.

EN 60908 (1999): "Audio recording - Compact disc digital audio system."

Fenet, Sébastien (2013): *Audio-Fingerprints and Associated Indexing Strategies for the Purpose of Large-Scale Audio-Identification*. Ph.D. thesis, Télécom ParisTech.

Fenet, Sébastien; Yves Grenier; and Gael Richard (2013): "An Extended Audio Fingerprint Method with Capabilities for Similar Music Detection." In: *Proceedings of the 14th International Society for Music Information Retrieval (ISMIR) Conference*. pp. 569–574.

Fischler, Martin A and Robert C Bolles (1981): "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." In: *Communications of the ACM*, **24**(6), pp. 381–395.

Frolova, Darya and Denis Simakov (2004): *Matching with invariant features*. Tech. rep., The Weizmann Institute of Science.

Funayama, R.; H. Yanagihara; L. Van Gool; T. Tuytelaars; and H. Bay (2009): "Robust interest point detector and descriptor." URL `http://www.google.co.ve/patents/US20090238460`. US Patent App. 12/298,879.

Graps, Amara (1995): "An introduction to wavelets." In: *IEEE Computational Science & Engineering*, **2**(2), pp. 50–61.

Grosche, Peter and Meinard Müller (2012): "Toward musically-motivated audio fingerprints." In: *Proceedings of the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Kyoto, Japan: IEEE, pp. 93–96.

Guzman-Zavaleta, Z Jezabel; Claudia Feregrino-Uribe; Alejandra Menendez-Ortiz; and Jose Juan Garcia-Hernandez (2014): "A robust audio fingerprinting method using spectrograms saliency maps." In: *Proceedings of the 9th International Conference for Internet Technology and Secured Transactions (ICITST)*. IEEE, pp. 47–52.

Haitsma, Jaap and Ton Kalker (2002): "A Highly Robust Audio Fingerprinting System." In: *ISMIR*, vol. 2002. pp. 107–115.

Haitsma, Jaap and Ton Kalker (2003): "Speed-change resistant audio fingerprinting using auto-correlation." In: *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4. IEEE, pp. 728–731.

Hansen, Sven (2013): "Wirbel um Weihnachts-CD der SOS-Kinderdörfer." URL `http://www.heise.de/newsticker/meldung/Wirbel-um-Weihnachts-CD-der-SOS-Kinderdoerfer-1784486.html`.

Harris, Chris and Mike Stephens (1988): "A combined corner and edge detector." In: *Proceedings of the Alvey Vision Conference*, vol. 15. pp. 147–152.

Hellmuth, Oliver; et al. (2001): "Advanced audio identification using MPEG-7 content description." In: *Audio Engineering Society Convention 111*. Audio Engineering Society.

Hellmuth, Oliver; et al. (2003): "Using mpeg-7 audio fingerprinting in real-world applications." In: *Audio Engineering Society Convention 115*. Audio Engineering Society.

Herre, Jurgen; Eric Allamanche; and Oliver Hellmuth (2001): "Robust matching of audio signals using spectral flatness features." In: *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*. IEEE, pp. 127–130.

IEC 61866:1997 (1997): "Audiovisual systems - Interactive text transmission system (ITTS)."

Ingebrigtsen, Lars Magne (2007): "CDDB ID: A Tale of Woe." URL `http://quimby.gnus.org/circus/notes/cddb.html`.

Jang, Dalwon; Chang D Yoo; Sunil Lee; Sungwoong Kim; and Ton Kalker (2009): "Pairwise boosted audio fingerprint." In: *IEEE Transactions on Information Forensics and Security*, **4**(4), pp. 995–1004.

Kastner, Thorsten; et al. (2002): "MPEG-7 scalable robust audio fingerprinting." In: *Proceedings of the 112th Audio Engineering Society Convention*. Audio Engineering Society, pp. 1–4.

Ke, Yan; Derek Hoiem; and Rahul Sukthankar (2005): "Computer vision for music identification." In: *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, pp. 597–604.

Kim, Hyoung-Gook and Jin Young Kim (2014): "Robust audio fingerprinting method using prominent peak pair based on modulated complex lapped transform." In: *ETRI Journal*, **36**(6), pp. 999–1007.

Kim, Samuel and Shrikanth Narayanan (2008): "Dynamic chroma feature vectors with applications to cover song identification." In: *Proceedings of the 10th IEEE Workshop onMultimedia Signal Processing*. IEEE, pp. 984–987.

Kim, Samuel; Erdem Unal; and Shrikanth Narayanan (2008): "Music fingerprint extraction for classical music cover song identification." In: *Proceedings of the 2008 IEEE International Conference on Multimedia and Expo*. IEEE, pp. 1261–1264.

Klapuri, Anssi (2006): "Multiple Fundamental Frequency Estimation by Summing Harmonic Amplitudes." In: *Proceedings of the 7th International Conference on Music Information Retrieval*. pp. 216–221.

Kurth, Frank; Thorsten Gehrmann; and Meinard Müller (2006): "The Cyclic Beat Spectrum: Tempo-Related Audio Features for Time-Scale Invariant Audio Identification." In: *Proceedings of the 7th International Conference on Music Information Retrieval*. pp. 35–40.

Lalinský, Lukáš (2011): "How does Chromaprint work?" URL `https://oxygene.sk/2011/01/how-does-chromaprint-work/`.

Lindeberg, Tony (1998): "Feature detection with automatic scale selection." In: *International Journal of Computer Vision*, **30**(2), pp. 79–116.

Lindsay, Adam T and Jürgen Herre (2001): "MPEG-7 and MPEG-7 Audio – An Overview." In: *Journal of the Audio Engineering Society*, **49**(7/8), pp. 589–594.

Lloyd, Stuart P (1982): "Least squares quantization in PCM." In: *IEEE Transactions on Information Theory*, **28**(2), pp. 129–137.

Lowe, David G (1999): "Object recognition from local scale-invariant features." In: *Proceedings of the 7th IEEE International Conference on Computer Vision*, vol. 2. IEEE, pp. 1150–1157.

MacQueen, James (1967): "Some methods for classification and analysis of multivariate observations." In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 14. Oakland, CA, USA., pp. 281–297.

Malekesmaeili, Mani and Rabab K Ward (2014): "A local fingerprinting approach for audio copy detection." In: *Signal Processing*, **98**, pp. 308–321.

Mauch, Matthias and Sebastian Ewert (2013): "The Audio Degradation Toolbox and its Application to Robustness Evaluation." In: *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*. International Society for Music Information Retrieval, pp. 83–88.

Neubeck, Alexander and Luc Van Gool (2006): "Efficient non-maximum suppression." In: *Proceedings of the 18th International Conference on Pattern Recognition (ICPR)*, vol. 3. IEEE, pp. 850–855.

Noll, A Michael (1969): "Pitch determination of human speech by the harmonic product spectrum, the harmonic sum spectrum, and a maximum likelihood estimate." In: *Proceedings of the Symposium on Computer Processing in Communications*, vol. 19. Polytechnic Press: Brooklyn, New York, pp. 779–797.

Peeters, G. (2004): *A large set of audio features for sound description (similarity and classification) in the CUIDADO project*. Tech. rep., IRCAM.

Phillips, Erica E. (2013): "Shazam Broadens Its Horizons." URL `http://blogs.wsj.com/speakeasy/2013/04/03/shazam-broadens-its-horizons/`.

Rafii, Zafar; Bob Coover; and Jinyu Han (2014): "An audio fingerprinting system for live version identification using image processing techniques." In: *Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 644–648.

Rajaraman, Anand; Jeffrey David Ullman; and Jure Leskovec (2012): *Mining of massive datasets*, vol. 1. Cambridge University Press Cambridge.

Ramona, Mathieu and Geoffroy Peeters (2011): "Audio identification based on spectral modeling of bark-bands energy and synchronization through onset detection." In: *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 477–480.

Ramona, Mathieu and Geoffroy Peeters (2013): "AudioPrint: An efficient audio fingerprint system based on a novel cost-less synchronization scheme." In: *Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 818–822.

Six, Joren and Marc Leman (2014): "Panako: a scalable acoustic fingerprinting system handling time-scale and pitch modification." In: *Proceedings of the 15th International Society for Music Information Retrieval (ISMIR) Conference*. pp. 259–264.

Smith, Oliver (2013): "Shazam Partners With The 'Spotify Of India', Saavn, To Improve Its South Asian Music Recognition." URL `http://techcrunch.com/2013/04/03/shazam-partners-with-the-spotify-of-india-saavn-to-improve-its-south-asian-music-recognition/`.

SN, Vikas (2013): "Updated: Shazam Ties Up With Saavn To Identify Hindi and Regional Music." URL `http://www.medianama.com/2013/04/223-shazam-saavn-tieup/`.

Sonnleitner, Reinhard and Gerhard Widmer (2014): "Quad-Based Audio Fingerprinting Robust to Time and Frequency Scaling." In: *Proceedings of the 17th International Conference on Digital Audio Effects (DAFx-14)*. pp. 173–180.

Steinhaus, Hugo (1956): "Sur la division des corp matériels en parties." In: *Bulletin L'Académie Polonaise des Science*, **1**, pp. 801–804.

Sturm, Bob L (2012): "An analysis of the GTZAN music genre dataset." In: *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies*. ACM, pp. 7–12.

Sturm, Bob L (2013): "The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use." In: *arXiv preprint arXiv:1306.1461*.

Terasawa, Kengo and Yuzuru Tanaka (2007): "Spherical lsh for approximate nearest neighbor search on unit hypersphere." In: *Algorithms and Data Structures*. Springer, pp. 27–38.

Tzanetakis, George and Perry Cook (2002): "Musical genre classification of audio signals." In: *IEEE transactions on Speech and Audio Processing*, **10**(5), pp. 293–302.

Van Balen, Jan (2011): *Automatic recognition of samples in musical audio*. Master's thesis, Universitat Pompeu Fabra.

Viola, Paul and Michael Jones (2001): "Rapid object detection using a boosted cascade of simple features." In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, pp. 511–518.

Wang, Avery et al. (2003): "An Industrial Strength Audio Search Algorithm." In: *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*. pp. 7–13.

Wang, Avery Lee Chang and Julius Orion Smith (2002): "System and Methods For Recognizing Sound and Music Signals in High Noise and Distortion." URL `https://www.google.de/patents/WO2002011123A2?cl=de`. WO Patent App. PCT/EP2001/008,709.

Wang, Jingdong; Heng Tao Shen; Jingkuan Song; and Jianqiu Ji (2014): "Hashing for similarity search: A survey." In: *arXiv preprint arXiv:1408.2927*.

Yang, Guang; Xiaoou Chen; and Deshun Yang (2014): "Efficient music identification by utilizing space-saving audio fingerprinting system." In: *Proceedings of the 2014 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, pp. 1–6.

Zhu, Bilei; Wei Li; Zhurong Wang; and Xiangyang Xue (2010): "A novel audio fingerprinting method robust to time scale modification and pitch shifting." In: *Proceedings of the International Conference on Multimedia*. ACM, pp. 987–990.