Williams et al. EURASIP Journal on Audio, Speech, and Music Processing (2017) 2017:17
DOI 10.1186/s13636-017-0114-4

EURASIP Journal on Audio, Speech, and Music Processing

RESEARCH Open Access

Efficient music identification using ORB descriptors of the spectrogram image

Dominic Williams*, Akash Pooransingh and Jesse Saitoo

Abstract

Audio fingerprinting has been an active research field typically used for music identification. Robust audio fingerprinting technology is used to successfully perform content-based audio identification regardless of the audio signal being subjected to various types of distortion. These distortions affect the time-frequency correlation relating to pitch and speed changes. In this paper, experiments are done using the computer vision technique ORB (Oriented FAST and Rotated BRIEF) for robust audio identification. Investigations are conducted for ORB, relating to its advantage of robustness against distortions including speed and pitch changes. The ORB prototype compares the features of the spectrogram image query to a database of spectrogram images of the songs. For the initial experiment, a Brute-Force matcher is used to compare the ORB descriptors. Results show that the ORB prototype performs robustly to real-world distortions with fast, reliable performance against distortions such as speed and pitch which justifies the research done.

Keywords: Audio fingerprinting, Music identification, Oriented FAST and Rotated BRIEF, Spectrogram

1 Introduction

In recent years, there has been an increase in digital multimedia technology, which made it simpler to share music files. Music is being compressed for portability for the ability to be transmitted between computers via the internet. This proliferation of digital multimedia including online streaming services, and online downloading, has become very popular in the past decade. Now, almost any song can be purchased or streamed digitally at the consumer's fingertips [1]. According to [2], the cost to manufacture and distribute the physical media components such as the CD's, and plastic cases, are no longer necessary, since digital music can reach a far greater audience in a shorter space of time. Music can now be released and distributed digitally [3] as soon as they are ready, and geographical accessibility has become limitless for the distribution of music. As a result, companies and private individuals have also developed various methods for searching, indexing, and matching unknown music samples to a database. This is done using audio fingerprinting [4] and can be used in application areas including broadcast monitoring, copyright protection, and digital rights management music identification.

* Correspondence: dominic.williams@my.uwi.edu
The University of the West Indies, St. Augustine, Trinidad and Tobago

Music identification [5] is typically done using an audio fingerprinting technique, which is a compact and unique digital summary of the audio content. The fingerprint is computed from the audio signal, stored in a database using a hash method (compact representation) and used for comparing and matching. It also contains the song's metadata such as artist, and song title. The techniques must be robust so that even audio content that is degraded due to distortion can still be accurately identified.

There are many algorithms such as the Philips Robust Hash (PRH) algorithm [5], which was a well-studied and verified method to be mathematically robust and efficient for searching. The PRH algorithm works by converting the audio signal from the time domain to the frequency domain and by diving the frequency spectrum into sub-bands where sub fingerprints are generated for each sub-band. WavePrint [6] proposed an algorithm using computer vision techniques for audio fingerprinting involving wavelets. The spectrogram image is determined and then broken into smaller spectral images where the top wavelets are computed using the Haar wavelet. Once completed, the top wavelets are computed and the sub-fingerprints are determined using the binary quantization. Also, the Shazam algorithm was successfully developed by [7]. This method is based on the use

of the spectrogram data where spectrogram peaks are determined and paired off and hashed to determine the robust audio fingerprint. In addition, the recently developed algorithm for music identification using the scale-invariant feature transform (SIFT) is seen to be successful in robustly identifying music based on the computer vision technique [8].

This paper proposes a computationally simple algorithm that will be robust to distortions such as noise. The system will also account for significant discrimination over a large number of fingerprints and scalability for efficient storage and comparing. The algorithm will be fast and accurate identification with minimal misidentifications and false positives. The robustness to distortions is a main factor to consider as there are various real-world distortions that affect the audio and therefore the performance of the algorithm. Some of these include interference, and environmental noise. Also, not all speakers and microphones will be high quality and therefore, this can cause certain frequencies to be lost or inaccurately recorded for analysis. Another distortion may come from the audio itself being altered due to speed and pitch synchronization changes (radio, online streaming, and etc.). This paper proposes a music identification algorithm that is highly robust to audio signal noise as well as time and frequency domain synchronization changes. Based on research, it is seen that ORB is a computationally efficient alternative to SIFT and SURF. See proof in [9].

2 Related work

An audio fingerprint is a compact content-based signature of the audio signal which the unique acoustic characteristics are stored as the fingerprint [10]. Using audio fingerprints for identification can be considered as two main parts. The first part is where the audio content is fingerprinted and stored into a database along with the related metadata. The second part is the where "unknown" audio content is fingerprinted and compared to the fingerprint database for identification [11]. If the sampled fingerprint is matched to a fingerprint in the database, then the corresponding metadata for that fingerprint is returned.

An audio fingerprint structure comprises of the components for fingerprint extraction and fingerprint identification. Fingerprint extraction entails the determination of robust features of the audio signal and then building a fingerprint representation based on those robust features [12]. Fingerprint identification is where the fingerprint is to be matched against a database to retrieve similar fingerprints. Collection of fingerprints of songs must be scaled for the purpose of efficiently storing and searching in a database, and etc. The following figure shows the general system for audio fingerprinting [12]. As shown in Fig. 1, the audio to be fingerprinted and stored in a database and the unknown audio sample to be fingerprinted and queried are both analyzed through the same process. The audio fingerprinting and matching algorithm must be robust so that the distorted audio sample can still be recognized as the same audio content stored in the fingerprint database. The audio signal is processed where it is sampled and converted from the time domain to the frequency domain. The audio fingerprints are then extracted to be compared to the audio fingerprint database of similarly analyzed music files. Once there is a match, the corresponding song information would be returned and displayed to the user.

There are various approaches of audio fingerprinting. Some discussed are the PRH, MLH, RARE, Shazam, Wavelet Transform, and SIFT algorithms. This section summarizes some of their features and limitations. According to Haitsma and Kalker [13], the major objective of audio fingerprinting is to determine the perceptual equality between two objects (multimedia) rather than comparing the actual objects directly. Digital fingerprinting technology can take advantage of this as fingerprints reduce the storage required in contrast to the size of the audio file itself. Also, audio fingerprints result in efficient searching since the dataset is smaller. In addition, perceptual discrepancies would have been removed in the process of fingerprinting the audio file. The fingerprints and the corresponding metadata would be stored in a database for comparison and retrieval. The fingerprint system can be summarized as two main components, the extraction method of the fingerprints and an efficient search and comparison method for the matching fingerprints between the query and the fingerprint database.

When dealing with audio fingerprinting, the audio signal is converted to another representation. Some typical transformations include the fast Fourier transform (FFT), the discrete cosine transform (DCT), and the Haar Transform. After the initial analysis is done on the audio signal, the features are determined. Some feature extraction methods include the mel-frequency cepstrum coefficients (MFCC) and the spectral flatness Measure (SFM), which is based on the power spectrum from the fast Fourier transform function [12].

Furthermore, Haitsma and Kalker [13] introduced and described a very robust fingerprint extraction method and a highly efficient searching strategy which was suitable for a large audio fingerprint database with limited computational resources (Philip's Robust Hashing Algorithm). The extraction process of the PRH algorithm was similar to other methods where the audio was segmented into frames with overlap of a specific length and then shifted a fraction (1/32) of the frame length [14]. The Fourier transform analysis would be computed for every frame, and the power spectral density is determined.

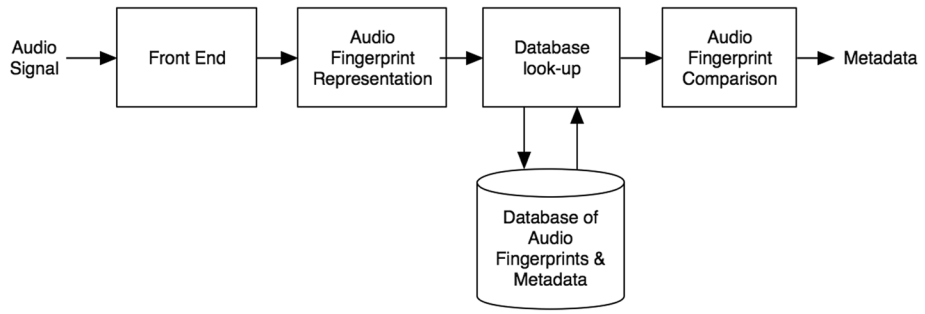

Fig. 1 Audio fingerprint system. This figure shows the general audio fingerprinting system done by previous authors. The algorithms generally followed this approach where the raw audio would be captured (by loading the audio signal or by microphone). The audio signal is then processed into a digital format where the audio fingerprints are extracted. The audio fingerprints of songs that were similarly processed and stored in a database are then compared by some algorithm to execute a database lookup. After the resulting audio fingerprint comparison is done, the corresponding song information for the highest match is returned

Sub-fingerprints would then be determined by dividing the frequency spectrum into logarithmic frequency bands for each frame. The bands would be in the human auditory system (HAS) range which was chosen as 300 to 2000 Hz [15]. The audio fingerprints were queried with the use of a look-up table. However, while dealing with low signal to noise ratio, the algorithm was not performing well. Also, the algorithm was susceptible to performance degradation, since a small change in frequency misalignment due to speed change [8].

The PRH algorithm was fast, however, the trade-off involved was accuracy. Liu, Yun, and Kim [5] proposed an algorithm using the multiple hashing method (MLH) in the DCT (discrete cosine transform) domain. The method was similar to the PRH algorithm, but with the application of the DCT to the time-based sequence of energies in each respective sub-band and a sub-fingerprint is generated for each DCT coefficient stream [5]. They claimed that the de-correlation property increased performance as it generated more sub-fingerprints, and that the robust property of compaction of the DCT will aid accuracy as the energy of the signal tends to be concentrated in certain frequency components. From their research, the MLH method performed better than the PRH method under various added noise experiments while maintaining accuracy. This improved the recognition of audio signal even if the quality of the signal is degraded. The consequence of this method was that the fingerprint size increased so the songs stored in the database would be less, and as a result, the search time would increase as well [8].

Burges et al. [16] proposed the Robust Audio Recognition Engine algorithm (RARE), which involved the use of the modulated complex lapped transform (MCLT) to convert the audio data to its time-frequency representation. Instead of determining the features based on the spectral representation, it reduces the dimensionality of the audio data by using two projections. Dimensionality reduction is where data is mapped to a lower dimensional space so that certain data variance is discarded. The projections are determined by training data using both distorted and undistorted data and applying the oriented principle component analysis (OPCA). The data is projected onto directions that increase the signal-to-noise ratio (SNR) of the training data [17]. Although this method was robust, the computation involved was the disadvantage to the algorithm.

Covell and Baluja [18] proposed a different approach to the previously stated methods, where computer vision techniques are incorporated for audio identification using wavelets. After the FFT is used to obtain the spectrogram, the spectrogram is then divided into smaller subsamples of spectral images to be decomposed using Haar wavelets. For each spectral image, the wavelet is computed. Then, the top wavelets are extracted (measured by magnitude) and the sub-fingerprints are found using the binary quantization of the major retained wavelet components using min-hash. The experimental results showed that the algorithm performed better than the PRH and MLH algorithms with additive white Gaussian noise [18]. Although the algorithm was robust, it was computationally expensive. Also, the fingerprint required more memory for storage and therefore, the database size and search time increase. Also, the algorithm was not as robust against time scale modification (TSM) and speed change [8].

In the early 2000s, Wang et al. founded Shazam Entertainment which developed a music recognition service to be accessible through the use of mobile phones. The user would be able to capture the audio sample for query by calling a simple number, and when completed,

the server would hang up the call and return the corresponding song information in an SMS text message [19]. The Shazam algorithm was developed a fingerprinting technique which involved the use of the highest amplitude peaks of the spectrogram (robust constellations), which were robust against noise. This was done by converting the audio signal from the time domain to the frequency domain using the fast Fourier transform (FFT) in order to obtain the spectrogram data of the audio (frequency spectrum for each frame) [20]. These peaks were then paired off and combinatorial hash values were computed based on time-frequency pairs, which also increased robustness. Wang also proposed an efficient searching and scoring algorithm. When a query was compared to the database, each matching hash was linked to offset time pairs between the offset times of the beginning of the query song and the song in the database. A match would be determined since the matching features would have similar differences of the relative offsets from the beginning of the song [7]. After experiments were done, it was seen that the algorithm was very robust to noise and compression but was sensitive to frequency and time synchronization alteration [8].

Other search techniques include inverted file indexing proposed by Haitsma and Kalker [13]. Once fingerprints of part of the sample query are correctly matched, a number of positions and candidate songs would then be retrieved, narrowing the search index for the rest of the query. Another set up involved separating the songs in the database into two databases. One database would contain fingerprints of more trendy songs of the moment (higher query probability), while another database would have the fingerprints rest of songs. This would decrease search times where the query would be compared to the smaller and more likely database and if there is not a match, it would be compared to the second database [11].

Further investigations involving the use of image recognition for audio identification as computer vision techniques made significant progress in recent years with the development of local invariant features. Zhang et al. [8] proposed "a novel feature for robust music identification" using scale invariant feature transform (SIFT) local descriptors on the spectrogram image. Generally, this would make feature descriptors and detectors robust to scaling, shifting, rotating, and etc. Results showed that the main advantage of this method was to deal with distortions such as pitch shifting and speed changes. These were difficult distortions for other music identification algorithms to resist, as seen in the results for algorithms such as Shazam. [8]. From a performance evaluation, Figat, Kornuta, and Kasprzak [21] stated that older vector-based descriptors such as SIFT and SURF (Speeded-Up Robust Features) were being replaced by binary feature descriptors and detectors. Some of these binary descriptors and detectors include Binary Robust Invariant Scalable Keypoints (BRISK), Binary Robust Independent Elementary Features (BRIEF), and the developed algorithm, Oriented FAST and Rotated BRIEF (ORB). From the results, it was seen that both the binary and vector descriptors had similar robustness to transformations of images while having great computational efficiency.

According to the paper on "ORB: an efficient alternative to SIFT and SURF", Rublee et al. [9] proposed a feature ORB which built on the well-liked keypoint detector algorithm, FAST (features from accelerated segment test), and recently developed BRIEF descriptor. From the results, it was seen that ORB performed better than older detectors and descriptors in both computational cost and speed. The goal was to improve image recognition on lower power devices with limited computational resources. Miksik and Mikolajczyk [22] evaluated the performance of ORB to common feature detectors and descriptors like SIFT and SURF. An optimized computer system was used with a 3.4 Ghz Intel CPU, and OpenCV libraries were used for the experiments. It was seen from the results that ORB resulted in a substantially better performance compared to the SIFT and SURF when determining keypoint detectors and descriptors. ORB also maintained high accuracy while being computationally faster making it more suitable. In general, ORB is also more robust to Gaussian noise when compared to SIFT [9]. However, there were some cases where SIFT had a higher accuracy when matching the features [21]. Also, ORB comprised of a combination of two successful modified feature detectors; the Oriented FAST keypoint detector and the Rotated BRIEF descriptor and was seen to be two orders of magnitude faster and more computationally efficiency than SIFT [23]. Hence, ORB was chosen for the proposed algorithm.

3 Proposed method

This proposes the use of the ORB method as discussed in [9]. The music signal is converted to a two-dimensional spectrogram image then ORB is used to compute the image and extract the spectrogram image features using the ORB descriptors. ORB is chosen since it is a robust and efficient computer-vision technique. This makes it a novel approach to music identification where distortions such as pitch and speed changes occur which may be difficult for other existing algorithms to be robust to. For the experimental prototype, a Brute-Force matcher is used to match the corresponding descriptors between the music query sample and the database of spectrogram images. This algorithm will be compared to the Shazam method which was a developed industrial-strength audio search algorithm [7]. Also, the

proposed system was comparable to the SIFT-based algorithm by Zhang et al. for robust music identification [8]. This can be seen in Fig. 2. The audio signal is processed where it is sampled and converted from the time domain to the frequency domain in order to obtain the spectrogram image. The image features are then extracted to be compared to the database of similarly analyzed spectrogram images. Once there is a match, the corresponding song information would be returned and displayed to the user.

3.1 Feature extraction

Firstly, pre-processing was done to convert the audio signal to the correct digital format and sample rate. For this prototype, the audio format used for testing was uncompressed (WAV), and the sample rate was 44.1 KHz. The audio is then divided into frames using the Hanning window technique of 8192 points and an overlap of 0.75. The window length was experimentally chosen so that the spectrogram representation would have a low time resolution which would increase robustness to time variations. The overlap was chosen since the signal maybe misaligned and hence this helps with the desynchronization problem. The fast Fourier transform is then applied to convert the audio from the time domain to the frequency domain in order to obtain the spectrogram [24]. The linear spectrogram is then quantized into 64 logarithmically spaced sub-bands to cover a large frequency range for more features [8]. After, the logarithmic spectrogram is converted to a gray image in order to extract the features. The spectrogram is converted to a log-magnitude representation using (1):

$$S(i,j) = \log|X(i,j)| \tag{1}$$

where S is the log magnitude spectrogram representation, X is the original spectrogram, i is the frequency sub-band, and j is the frame index [8].

Oriented Fast and Rotated BRIEF (ORB) consists of a fast keypoint detector which is based on the FAST algorithm and binary descriptor which is based on the BRIEF algorithm [9]. Keypoints can be defined as spatial locations on the image that may contain a feature that stands out. A descriptor can be defined as the way the keypoint is described. In other words, the keypoint is a detected location, and the descriptor is the details of that location. These are used as the image features to determine matches when comparing two image samples. ORB builds on variations of the FAST keypoint detector and the recent BRIEF descriptor, which are popular due to their computational efficiency and good performance [25]. The variation for the FAST keypoint detector is oFAST which adds the feature of orientation and Rotated BRIEF which adds the feature of rotation (it is rotation invariant).

Due to its good performance, FAST-9 is used (called oFAST), which is a variation of FAST that uses a circular radius of 9. A Harris corner measure is used to order the FAST keypoints, since FAST had high responses along edges. Also, since FAST does not yield multi-scale features and is not scale invariant, a scale pyramid of the imaged is employed and for each pyramid level, the FAST features are determined [9]. The intensity centroid is used for the measure of corner orientation. The following describes the moments of a patch defined by Rosin [26] as:

$$M_{\mathrm{pq}} = \sum_{x,y} x^{\mathrm{p}} y^{\mathrm{q}} I(x,y) \tag{2}$$

where $x^{\mathrm{p}}y^{\mathrm{q}}$ are the pixel location in corresponding patch location of the image, and $I(x,y)$ is the image intensity. Based on the moments, the centroid is determined by:

$$C = \left(\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}}\right) \tag{3}$$

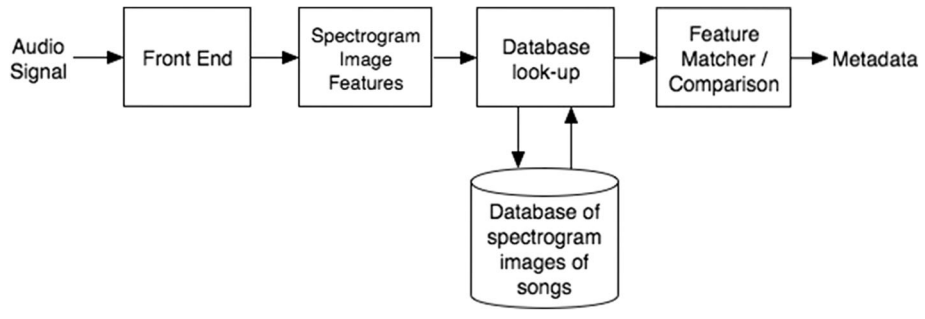

Fig. 2 Proposed system. The figure shows the proposed system where the raw audio would be captured (by loading the audio signal or by microphone). The audio signal is then processed by converting the audio signal into a spectrogram image. The spectrogram image of the sample is compared to the database of spectrogram images of the songs in the database. This is done using an image processing techniques to match the features of the spectrogram image. After the resulting audio fingerprint comparison is done, the corresponding song information for the highest match is returned

Now, from the corner's center, O, a vector could be constructed $\overrightarrow{OC}$. The patch orientation can be determined as:

$$\theta = \text{atan2}(m_{01}, m_{10}) \tag{4}$$

However, Rublee et al. [9] stated that regardless of the corner type, the angle is consistent, so it can be ignored. The rotation invariance was then improved by computing the moments with x and y staying within the circular radius r. This is done by choosing r as the patch size so that x and y run from the range $[-r, r]$.

The BRIEF descriptor is defined as a bit string of an image patch created from a set of binary intensity tests [25]. The following shows the binary test τ:

$$\tau(p; x, y) = \begin{cases} 1 & : p(x) < p(y) \\ 0 & : p(x) \geq p(y) \end{cases} \tag{5}$$

where p is the smoothed image patch, $p(x)$ is the intensity of p at point x.

The feature is defined as a vector of n binary tests:

$$f_n(p) = \sum\nolimits_{1 \leq i \leq n} 2^{i-1} \tau(p; x_i, y_i) \tag{6}$$

Due to its performance, a Gaussian distribution test is applied around the center of the patch [9]. Before the test is performed, the image is smoothened. For BRIEF to be invariant to rotation, an efficient approach was implemented to steer BRIEF in accordance to the keypoint orientation. At location (x_i, y_i), the 2 x n matrix is defined by the following:

$$S = \begin{pmatrix} x_1, \ldots, x_n \\ y_1, \ldots, y_n \end{pmatrix} \tag{7}$$

From the rotation matrix R_θ and the patch orientation θ, the "steered" S_θ:

$$S_\theta = R_\theta S \tag{8}$$

Therefore, the steered BRIEFis [9]:

$$g_n(p, \theta) = f_n(p) | (x_i, y_i) \in S_\theta \tag{9}$$

Now, the angle is discretized to 12° increments and a lookup table which includes already computed BRIEF patterns is created. An advantage of BRIEF is that each bit feature has a high variance. However, from the results from tests, the steered BRIEF has lower variance. The variation of BRIEF used by [9], rBRIEF, caused the variance of the steered BRIEF to improve.

From the experiments done by Rublee et al. [9], ORB produced high performance and computational efficiency when compared to other similar computer vision methods such as SIFT and SURF. For this experiment, the orientation and rotation aspect was not necessary as the analysis was done on spectrograms which were stationary images.

Since conditions are not ideal in the real world, it is common for audio signals to be affected by various types of audio distortion. Pitch shifting, time stretching and time scaling are some difficult distortions for music identification to resist. Pitch shifting causes the spectrogram image to remain the same on the time axis, but the frequency component is just translated vertically (up or down). Time stretching causes the time axis to lengthen or shorten while the frequency axis remains constant on the spectrogram. Time scaling can be classified as the combination of pitch shifting and time stretching.

The features are then extracted from the spectrogram image to represent the unique characteristics of the music using the ORB descriptors which are based on the corners detected for patches on the image. Generally, ORB-based features are robust to changes in image scale, illumination, and etc. It was also noted that the spectrogram is a stationary representation the orientation and rotation features were not needed to be used for computation.

3.2 Robust matching

For this module, the unknown sample is analyzed in the same manner as the audio for the database where the spectrogram image is created for the unknown audio sample. When the spectrogram image is created for the query sample, it is stored temporarily. The ORB detector is initiated, and the key-points and descriptors are detected and computed for the query audio sample (spectrogram image).

Once completed, the BFMatcher (Brute-Force Matcher from OpenCV) is used along with a Hamming windowing technique to search for matches. The BFMatcher uses a descriptor in one data set (sample) and compares it to all the other features in the second data set (in the database) with a distance calculation (Hamming distance), and the closest match is returned. This is done for all the descriptors in the first data set. In the real world, noise may exist, creating the possibility of some false positives (although not much). Usually, a threshold is put in place to the distance between the query feature and the nearest matching feature returned, rejecting matches that may have distances greater than the threshold [8]. When the descriptors are matched, a threshold was used for the distances of the matches, if the distance is far apart between descriptor matches of the query sample and a database song then it may have been a false positive detected. The amount of matches below the threshold distance would be the confidence of the corresponding match for the song. The Brute-Force matcher was used to achieve exact matches to the

spectrogram image features and to minimize false positives from being detected since the spectrogram features can be considered as fine details on the image. Tests were done and results were shown in the next section.

4 Experimental results

Testing was done for robustness [9] and other factors. For this experimental prototype, the database consisted of the songs being converted to their associated spectrogram image representation. These spectrogram images, along with their related song information would be stored for searching and retrieval of the query sample and its corresponding match. For this experiment, a database library of songs with an even spread of genres were used, including Pop, Hip-hop, Modern Rock, Soca, and Jazz. ORB would be used to determine the descriptors of the spectrogram images which would be used for comparison. These songs comprised of both similar and dissimilar chords/rhythms and various instruments to test the difficulty in song identification.

The algorithm was tested by using songs of various genres (Pop, Hip-hop, Modern Rock, Soca, and Jazz), and testing was performed on a total of 3200 sample songs. Various factors were considered in sample selection. Not only tests for robustness to various distortions, but other factors such as reliability, system speed, and granularity were experimented [27]. Robustness involves the system having the ability to accurately identify the audio when subjected to real world variations including compression and various types of distortion such as radio interference, environmental noise, pitch shifting, and time scaling. Accuracy deals with the system being able to determine the correct match instead of missed identifications and false positives (incorrect identifications). Due to the amount of songs that may or may not be in the database, a reliable system must be able to distinguish the actual match from similar audio content and be able to assess if the query is not in the database and hence should not have a match (no false positives). The granularity of the system is the ability to correctly identify the matching audio when the query is a short audio clip. This is where the searching algorithm is important. The complexity of the system processes such as the feature extraction, searching and comparison complexity, fingerprint size, and database alterations should not be computationally demanding. The Shazam algorithm and the ORB prototype were tested and compared and in addition, research for the SIFT-based music identification algorithm by Zhang et al. [8] was also considered.

4.1 Robustness

This was a key aspect of testing. The audio samples were subjected to various types of audio distortions that occur in the real world. The tests done included speed changes, pitch changes, tempo changes and the addition of noise to the audio samples. From research, Rublee et al. [9] proposed that using image processing techniques for audio identification had an advantage of robustness to distortions such as speed and pitch changes which algorithms such as Shazam was difficult to resist. This was also investigated from the results. The tests were done to test the ability of the two algorithms to accurately identify the query samples. The Shazam-based algorithm and the ORB prototype were tested, and the results were compared.

4.1.1 Speed test

The speed test was done to test the ability for the query audio sample to be accurately identified with different speed changes applied to the audio samples. A 20-s sample was taken from each song and queried to the database. The audio samples' speeds were edited using the Audacity audio editing software. The audio samples were tested in each case, and the results showed the number of queries that matched for each speed change. This can be seen in Fig. 3.

The speed test was done to apply a synchronization attack on the sample audio for query. This would cause the query clips to deform as this type of distortion alters both pitch and the original time scaling of the song. An increase in speed would increase the pitch and decrease the length of that clip (tempo) and a decrease in speed would decrease the pitch and increase the length of the clip (tempo), as it would be slower. The test was done in such a way that a random 20-s audio sample was taken for each song in the database. The distortion was then applied for various percentage ranges and the clips were queried for different test case scenarios. For each test, the resulting match rate percentage would be determined by the songs that were accurately matched. In the real world, the distortion range would be from −10 to +10%, since this is where the distortion is most likely to occur [8]. However, speed tests were also done out of this range to test the limits of the experimental prototypes.

In Fig. 3, the speed test was done for the range from −20 to +20%. The both algorithms were compared to show the effects of the speed changes on the ability for the both systems to correctly identify the distorted samples.

From the graph, it was seen that the Shazam algorithm did not perform well in accurately identifying the distorted sample queries for any speed variation. This could have been due to the pitch and time variation which would affect the time-frequency peak pairs used for the hashing aspect. Since the Shazam algorithm is based on combinatorial hashed time-frequency constellation analysis, the hashes are based on specific frequencies at its associated point in time. These are then paired off and a

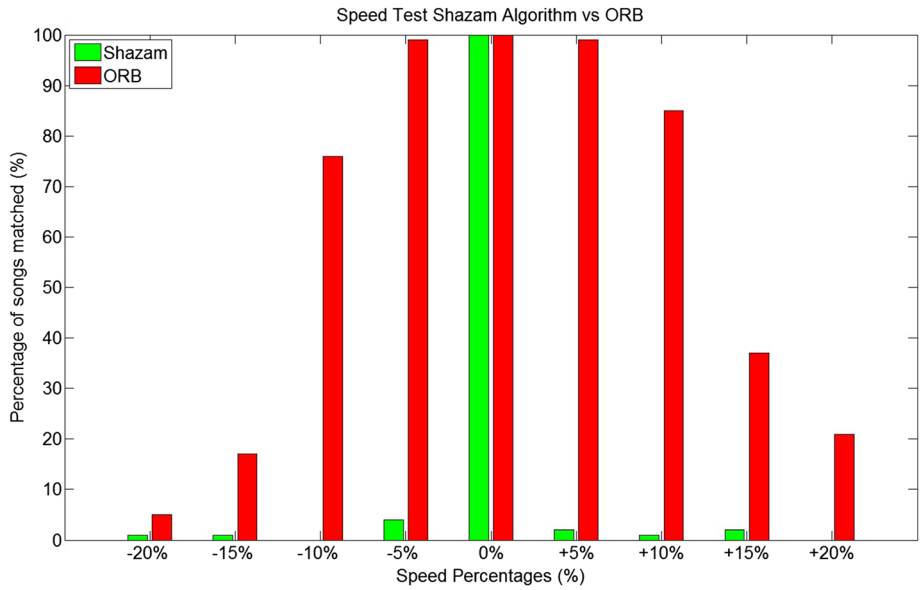

Fig. 3 Speed test for robustness (Shazam vs ORB). In this figure, the speed test was done for the range from −20 to +20%. The both algorithms were compared to show the effects of the speed changes on the ability for the both systems to correctly identify the distorted samples. The speed is the combination of pitch and tempo distortion. It was seen that the ORB prototype produced better results compared to the Shazam algorithm

hash a created based on the frequency pairs and their time difference. Therefore, once there is an alteration in speed, the pitch-changing aspect (frequency change) would cause mismatch to occur (due to different hashes generated) and hence the query would not be accurately identified. Also, it was noted that the few samples that matched the query had a low confidence.

It was seen that the ORB prototype produced better results compared to the Shazam algorithm. The ORB prototype was able to correctly identify almost all the songs for the range from −10 to +10% where the match rate fell to approximately 80% for the range from −10 to +10%. Higher percentages tested the algorithm past real-world situations which showed the ORB prototype performance decreasing but still better than the Shazam method. The experimental results showed that the ORB implementation was generally accurate for typical distortion ranges in the real world and also performed better than the Shazam method. This would be due to ORB being robust to shifting and scaling of the features of the spectrogram image query. This would justify the research done previously [8].

4.1.2 Pitch test

The pitch test involved testing a 20-s sample of each song in the library and applying various pitch changes to the audio samples. This experiment was done to test the ability for queries to be accurately identified at various pitches. The audio samples were tested in each case, and the results showed the amount of query samples that matched for each pitch change. Figure 4 shows the comparison of results for both the Shazam-based algorithm and the ORB prototype.

This experiment showed the audio samples being queried to be correctly identified when pitch distortions are applied. Pitch distortion would cause the frequencies in the audio sample to shift vertically upward on the spectrogram if the pitch is increased, and vertically downward if the pitch is decreased. The time synchronization would remain the same, and there are no temporal changes involved. The test was done similarly to the speed where a random 20-s clip was taken for each song in the database, and the pitch distortion was applied. The typical range was tested (−10 to +10%), but the limits of the prototypes were tested outside the real-world conditions (−20 to +20%) in increments of 5% change.

Similar to the speed test, it was seen in Fig. 4, that the Shazam algorithm generally did not perform well as the various pitch changes were applied. Also, the confidences of the songs that matched were relatively low (approximately 0–10). This would be due to the similar reason to the speed test where the pitch is being altered, therefore, the frequencies would change causing the audio fingerprints to misalign since they are based on the time-frequency pairs for the hashes.

The ORB method performed very well for the real-world pitch variation conditions and was seen to be robust for pitch changes up to ranges from −20 to +20%. The ORB method was generally robust to pitch distortion and outperformed the Shazam method. This would be due to ORB being robust to shifting of features (peaks) on the spectrogram images which would confirm previous research done [8].

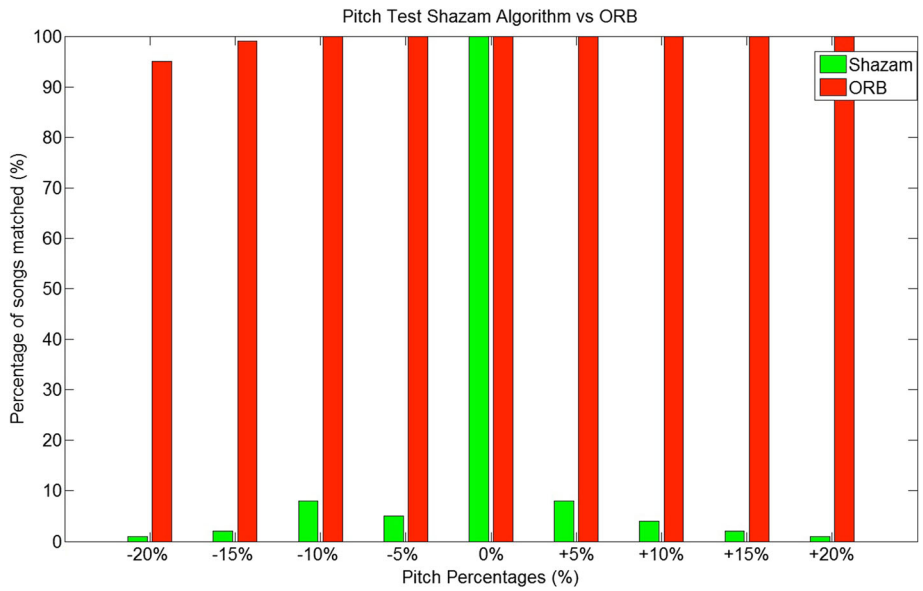

Fig. 4 Pitch test for robustness (Shazam vs ORB). This figure involves the pitch test where audio samples were queried to be correctly identified when pitch distortions are applied. Pitch distortion would cause the frequencies in the audio sample to shift vertically. The time synchronization would remain the same, and there are no temporal changes involved. The test was done similarly to the speed where a random 20-s clip was taken from each song in the database, and the pitch distortion was applied. The typical range was tested (−10 to +10%), but the limits of the prototypes were tested (−20 to +20%). It was seen that the ORB prototype produced better results compared to the Shazam algorithm

4.1.3 Tempo test

This test involved adjusting the tempo (speed while preserving the pitch) for audio samples taken from each song in the database. This tested the ability for the query to be accurately identified when tempo distortions are applied. The audio samples were tested in each case, and the results showed the amount of query samples that matched for each tempo change. The tempo distortions were tested in increments of 10% changes for a range from −50 to +50% to test the limits of the algorithms. Figure 5 shows the results of the tempo test for both the Shazam-based algorithm and the ORB prototype.

In this experiment, the tempo was changed to different extents. This type of distortion is known as time stretching or time scaling. The speed of the audio clip is altered while the pitch remains constant. On the spectrogram, this change would be seen on the time axis (temporally), but no change on the frequency axis (pitch). The experiment was carried out where a random 20-s sample was taken from each song in the database, and the tempo change was applied using the Audacity software. The distorted samples were then queried to the test if they accurately matched.

In Fig. 5, it was seen that the Shazam algorithm performed well generally for the various tempo changes applied and outperformed the ORB method. This could be due to the sample rate, the window size, and the overlap chosen (44100 Hz, 4096 and 50%, respectively). Due to a large window size and high overlap for each consecutive sub-sample determined, as the time stretch distortion occurs (along the horizontal time axis), the combinatorial hashes or audio fingerprints generated would still match when compared [7]. In addition, this could be due to the number of audio fingerprints generated because of the "neighbourhood" size in which the distance for peaks to be paired off is considered. In this case, this would cause a lot of audio fingerprints to be generated. As a result of the tempo change being constant for each test, the absolute offset difference between the query offset and the offset of the song in the database would be constant. Also, although the confidence may not be as high as the original sample without distortion, the accurate match would still have a generally high confidence.

For the results obtained in Fig. 5 for the ORB prototype, it was seen that the queries matched accurately for the typical range (−10% to +10%). The results ORB prototype dropped for higher ranges; however, further work can be done to improve this, which is suggested in Section 5. This meant that for the real-world applications, it would be robust which corresponds to research done.

4.1.4 Noise test

For the noise test, audio samples were taken from the database. The audio samples were then corrupted with additive white Gaussian noise. The signal to noise ratio was tested where the white Gaussian noise was applied to the signal for percentage amplitude ranges of noise in

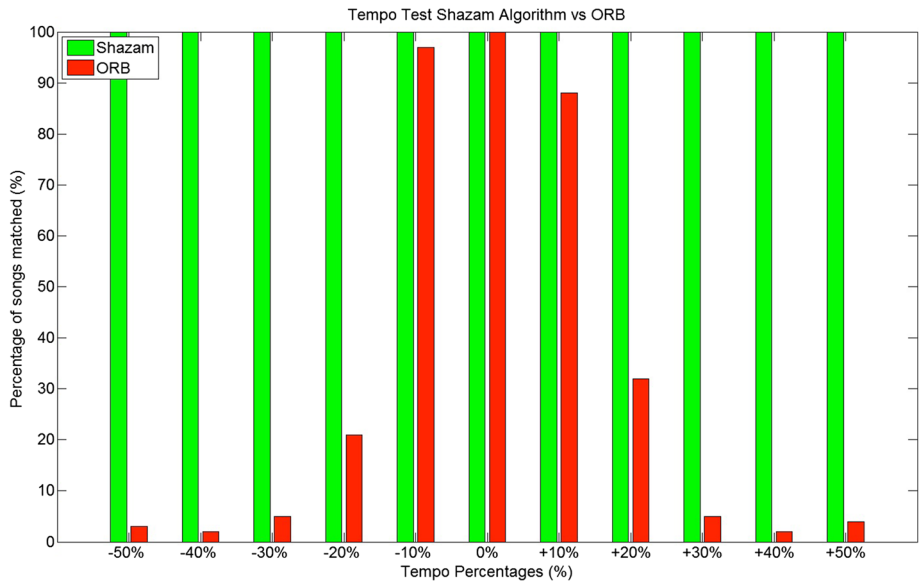

Fig. 5 Tempo test for robustness (Shazam vs ORB). This figure shows the results for the tempo being changed by a range from −50 to +50% to test the limits. This type of distortion is known as time stretching or time scaling. The speed of the audio clip is altered while the pitch remains constant. On the spectrogram, this change would be seen on the time axis (temporally) but no change on the frequency axis (pitch). The experiment was carried out where a random 20-s sample was taken from each song in the database, and the tempo change was applied. The distorted samples were then queried to the test if they accurately matched. It was seen that the Shazam algorithm produced better results compared to the ORB prototype

increments of 10% from 0 to 50% to test the limits. This test examines the ability for the corrupted audio samples to be identified accurately. The audio samples were tested in each case, and the results showed the amount of query samples that matched for each case of added noise. Figure 6 shows the results for the Shazam-based algorithm and the ORB prototype.

This experiment involved the testing the ability to accurately identify the audio sample when it is subjected to the addition of noise. In the real world, this is a common case. Additive white Gaussian noise was added which would be used to simulate background noise. The similar test setup was done where a 20-s clip was taken from each song in the database, and the various degrees of noise were added to each sample and then tested for each system.

From Fig. 6, the Shazam algorithm was very robust to the noise as seen in the results. This would be due to the peaks being acquired using the spectrogram data in the audio fingerprinting extraction process. The peaks obtained will be more likely to survive the effects of noise as the song's frequencies (from the audio sample) would have a higher amplitude when compared to the amplitude of frequencies of the noise. It was seen that the Shazam's algorithm performed slightly better than the ORB prototype for higher noise percentages. This was justified by previous research done by [7].

The ORB method involved the analysis and comparison of the query sample to the database based on the spectrogram images themselves. The performance of the ORB method with noise was well for real-world circumstances and was similarly as robust as the Shazam method. This could be due to the fact that ORB uses the spectrogram images. When there is noise present, the spectrogram image would differ (from a spectrogram with no noise) as the song's frequencies can be masked with the noise (different color seen on the spectrogram image). Due to the compression of the image, the noise may be more apparent in the spectrogram image. This can be a source for the decrease in accuracy of the identification of the query sample.

4.2 Other factors

In addition to the main experiments testing robustness to distortions of speed, pitch, tempo, and noise, further experiments were conducted to determine the performance of the algorithms on softer metrics such as compression, reliability, granularity, scalability, and system speed. These tests measured an example of the performance of a single query under varying conditions.

4.2.1 Compression

An example of a random 20-s clip of a song from the database was used for this test. It was subjected to MP3 compressions at various bitrates, where lower bitrates would result in compression loss and therefore, audio degradation can be heard. This tests the ability to accurately identify the sample query with distortion due to

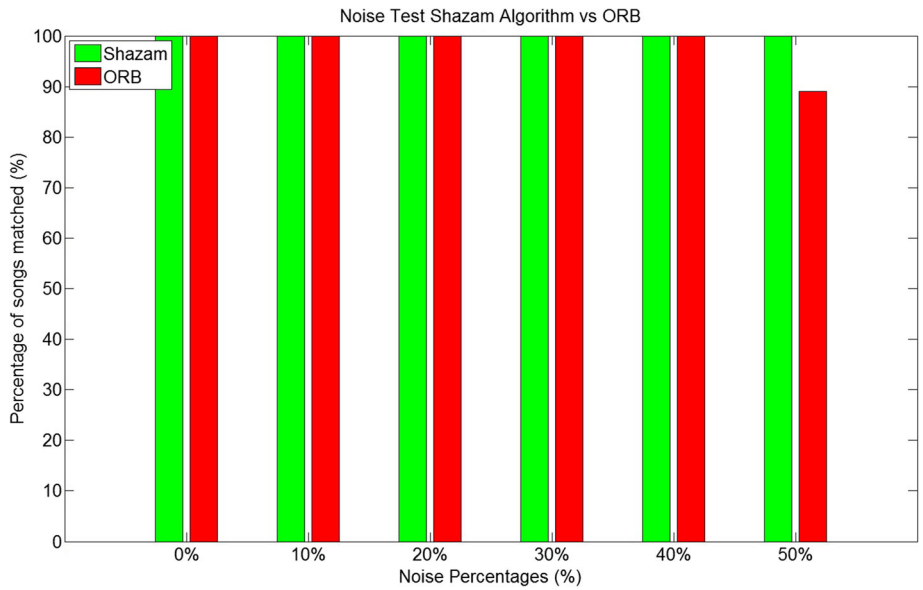

Fig. 6 Noise test for robustness (Shazam vs ORB). In this figure, the experiment involved testing the ability to accurately identify the audio sample when it is subjected to the addition of noise. In the real world, this is a common case. Additive white Gaussian noise was added which would be used to simulate background noise. The similar test setup was done where a 20-s clip was taken from each song in the database, and the various degrees of noise were added to each sample and then tested for each system. It was seen that the Shazam algorithm was very robust to the noise as seen in the results. This would be due to the peaks being acquired using the spectrogram data in the audio fingerprinting extraction process. The peaks obtained will be more likely to survive the effects of noise as the song's frequencies (from the audio sample) would have a higher amplitude when compared to the amplitude of frequencies of the noise. It was seen that the Shazam's algorithm performed better in comparison to the ORB prototype although the ORB prototype performed well to an extent

compression (quantization noise). Tables 1 and 2 show the results for the Shazam-based algorithm the ORB-based prototype.

From the results above, the number of matches for the Shazam method and the ORB method were based on two different criteria hence their results cannot be compared. The Shazam method uses audio fingerprints where an entire song will have a great amount of audio fingerprints depending on the amount of combinatorial hashes determined [7]. In the Shazam method, the time-based high amplitude peaks are determined in the frequency domain to then be paired for robust and unique identification. The ORB prototype uses the keypoints and descriptors as the features to be matched. This is dependent on the detail of the image to be captured and the quality of the image, i.e., a higher quality image with more determined descriptors would have a higher accuracy. For the experiment, an average value of 3000 descriptors were used based on the specifications of the computer hardware available, hence the number of matches would be lower.

When audio is compressed to lossy formats such as MP3 and lower bitrates, audio data can be lost. The test was used to determine if the prototype could still accurately identify the unknown sample with the songs in the database, when the unknown sample is compressed to various bitrates of the MP3 audio format. From the results in Table 1, the Shazam-based algorithm was able to accurately identify the unknown samples at different compression rates. It was observed that at very low bitrates such as 64 Kbps, the confidence of the match dropped significantly as compared to a sample at 320 Kbps, however, it was still able to identify the song correctly with a relatively high confidence compared to other songs in the library.

Table 1 Robustness test using compression (Shazam)

Compressed Sample query	Successful match	No. of matches
Sample at 320 Kbps	Yes	812
Sample at 192 Kbps	Yes	767
Sample at 128 Kbps	Yes	796
Sample at 96 Kbps	Yes	471
Sample at 64 Kbps	Yes	288

Table 2 Robustness test using compression (ORB)

Compressed Sample query	Successful match	No. of matches
Sample at 320 Kbps	Yes	169
Sample at 192 Kbps	Yes	181
Sample at 128 Kbps	Yes	184
Sample at 96 Kbps	Yes	170
Sample at 64 Kbps	Yes	126

The results from both tests were expected from research, as it was seen that when compression increased, the audio data lost increases and hence the confidence would decrease as a result. However, compression should not result in significant loss of audio data to cause mismatch to occur. For the Shazam method, a relatively high accuracy was maintained for sample rates between 128 and 320 Kbps as the number of matches were generally high for these compression rates. At 96 and 64 Kbps, the songs matched accurately but the number of matching hashes dropped by almost half for each of the lower bitrates.

The ORB prototype was then tested, and the results shown in Table 2. The compressed audio samples were converted to their respective spectrogram images and were analyzed. From the results, it was seen that the confidence was relatively high for most tested samples. A relatively high accuracy was maintained for sample rates between 96 and 320 Kbps as the number of matches were generally high for these compression rates. The sample compressed at 64 Kbps had the lowest confidence but still had a high amount of matches compared to other songs in the library.

4.2.2 Reliability

Firstly, a test was done where a random 20-s sample was taken for each song and was queried to determine if they match correctly. The results were used to determine if the system accurately returns the correct match for each sample. All samples matched correctly with the songs in the database.

Next, an example of a random 20-s sample clip from a song in the database was used to compute its match compared to the matches found in other songs in the database. This was used to show the difference between the number of matches of the correct song match to other songs in the database. The similarity for the Shazam-based algorithm is shown in Fig. 7.

Based on these results the variance and hence the standard deviation was determined to measure how spread out the numbers are from the mean. The following equation shows the variance for the t-distribution since N is the number of songs in the library.

$$s^2 = \frac{\sum (X-\mu)^2}{N-1} \tag{10}$$

where,

s^2–variance
μ–the average
X–the values from the graph
N–the number of terms in the distribution

Hence, the standard deviation, s is,

$$s = \sqrt{\frac{\sum (X-\mu)^2}{N-1}} \tag{11}$$

The variance was determined to be 26500.38, and the standard deviation is simply the square root of the variance. Therefore, the standard deviation for the Shazam method was 162.79. The standard deviation is large which means that the matches of the other songs in the database were small.

The same test involving a random 20-s sample being used to query the database was done to show its similarity of the actual match to other songs in the library. The similarity is shown in Fig. 8 for the ORB prototype.

Similar to the Shazam method, the variance and therefore, the standard deviation were calculated for the ORB results from Fig. 8. The Eqs.10 and 11 were used to calculate the variance and standard deviation, respectively. The variance for the ORB method results was calculated to be 1113.58, and the standard deviation was calculated as 33.37. The standard deviation is large which means that the matches of the other songs in the database were very small.

From experimentation, it was seen that when a random 20-s sample clip was taken from each song in the database and queried using both algorithms, all samples were accurately identified. In addition, a test was done to show and compare the number of matches a query had to each song in the library. This resulted in a significant difference in the number of matches of the actual song match compared to the number of matches for other songs in the database. In Fig. 7, a random sample clip was taken and queried using the Shazam algorithm. Results showed that the number of matches for the corresponding song was large compared to the matches of the query found in other songs in the database. This justified the reliability of the Shazam algorithm.

The ORB method was tested similarly, where the generated spectrogram image of the same random 20-s sample clip was queried to the database and the amount of matches for each song was shown in Fig. 8. It was also seen from results that the corresponding song identified had the most amount of matches when compared to the matches found in other songs in the database. Based on the results, both algorithms were reliable in the identification of songs correctly.

For the Shazam method, the variance was determined to be 26500.38, and the standard deviation was 162.79. Similar to the Shazam method, the variance and therefore the standard deviation were calculated for the ORB results. The variance for the ORB method results was determined to be 1113.58, and the standard deviation was calculated as 33.37. The standard deviation for both methods was large which confirmed that the matches of

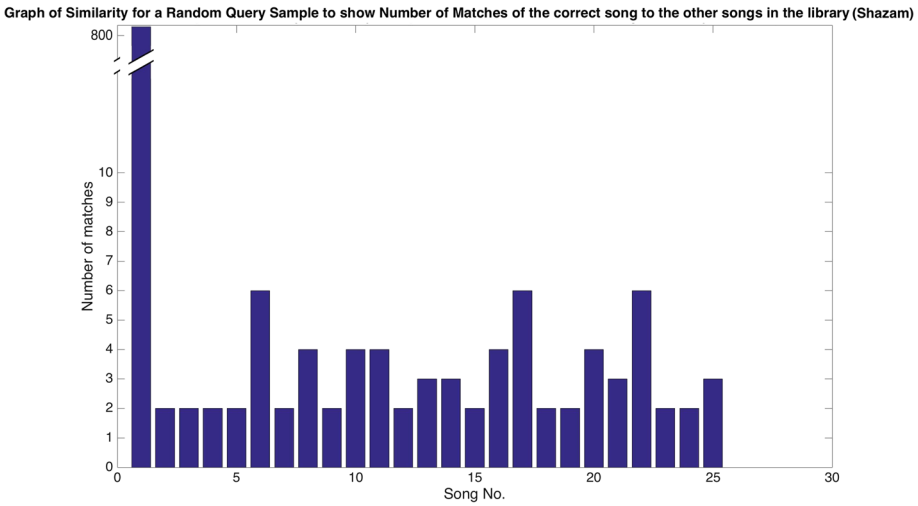

Fig. 7 Test for similarity of unknown sample to songs in the database (Shazam). In this figure, it was seen that a random 20-s sample clip was taken from a song in the database and was queried using the Shazam algorithm. This test was done to show and compare the number of matches the query had to each song in the library. Results showed that the number of matches for the corresponding song was large compared to the matches of the query found in other songs in the database. This justified the reliability of the Shazam algorithm

the other songs in the database were very small compared to the actual match.

4.2.3 Scalability

The system was tested for scalability to measure the size of the songs to the size of the database. The songs for the database were shown for MP3 and WAV formats. The Shazam-based audio fingerprint database consisted of the analyzed audio fingerprints of the songs using the algorithm developed. The ORB image database comprised of the spectrogram images of the song database. Table 3 compares the sizes of the songs for the database, the Shazam-based audio fingerprint database and the ORB image database.

This experiment involved the comparison of the various aspects of storage associated with the database and the actual audio files themselves. The storage was measured for the music files in both WAV and MP3 formats. The Shazam database's size was also measured which is based on the MySQL database consisting of two tables:

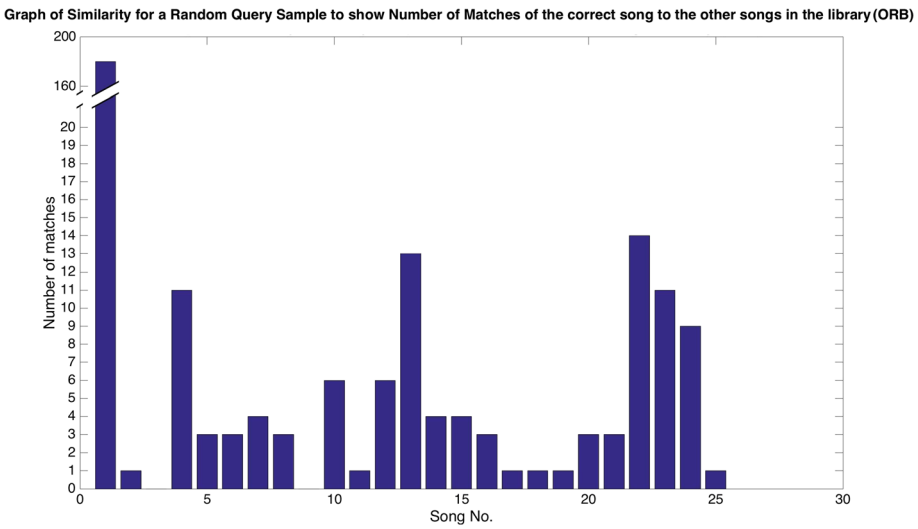

Fig. 8 Test for similarity of unknown sample to songs in the database (ORB). In this figure, it was seen that a random 20-s sample clip was taken from a song in the database and queried using the ORB prototype. This test was done to show and compare the number of matches the query had to each song in the library. Results showed that the number of matches for the corresponding song was large compared to the matches of the query found in other songs in the database. This justified the reliability of the ORB prototype

Table 3 Measurement of storage sizes for both algorithms

Audio information category for database	Total storage (size in MB)	Average storage for each song (size in MB)
WAV	1062	42.48
MP3	241	9.64
Shazam-based database	309	12.36
ORB-based database	364	14.56

the fingerprint table and the song table. The size of the ORB prototype's database was also measured which involved the storage of the spectrogram images of the songs in a folder.

From the results in Table 3, it was seen that the Shazam algorithm and the ORB algorithm both had similar database storage sizes with the ORB database being slightly larger. The reason for the size of the Shazam database being larger than the mp3 songs themselves would be because of the system design extracting and creating a large number of audio fingerprints for each song. The more the audio fingerprints, the higher the accuracy of the matching process. The reason for the ORB method having a large database is because the resolution of the spectrogram images in the database has to be high in order to properly determine the features to compare to the query for matches. The higher the resolution, the more details of the image are retained. The lower the resolution, the more data is lost due to image compression. The optimum resolution used was 3600×2700 pixels to maintain accuracy due to compression. This resolution was set experimentally due to the trade-off between processing time and accuracy based on the available hardware used for testing.

4.2.4 System speed

The speed of the system was measured to determine the time taken for the songs to be fingerprinted. For the Shazam-based algorithm, it took approximately 1147 s (approximately 19 min) to fingerprint the songs, while it took approximately 1489 s (approximately 25 min) to generate the spectrogram images for the songs for the image library. These results were based on the available hardware used for testing at the time. In addition, the time taken for a query to be searched and matched was also tested. This was done for libraries of songs. Figure 9 shows these results for the Shazam-based and ORB-based algorithms.

This experiment involved the time taken for the unknown sample to be queried and matched to the songs in the database. The test was done so that database size would be different sizes. Also, it was noted that the time taken to fingerprint the library of songs for both methods were also taken. The Shazam algorithm took approximately 1147 s while the ORB method took approximately 1489 s.

From the graph in Fig. 9, the Shazam algorithm results showed an approximately linear relationship, although five songs took a shorter time to for the audio sample to be queried. In Fig. 10, the ORB results were also linear, but it was seen that it had a larger time difference for the query process to be executed as the song library increased. This would be due to the images processing aspect relating to OpenCV's ORB library and the Brute-Force Matcher taking some time to process the image. It was seen that for smaller libraries, the ORB method worked better but for very larger libraries, the Shazam's audio fingerprinting method may be more efficient. Hence, the Shazam method may be suitable for very large databases whereas the ORB method may be better suited for smaller databases.

4.2.5 Granularity

Analyzing granularity involved testing the minimum length of the audio sample used to query the database for a correct match. Audio samples were used for a range of time intervals for the Shazam-based algorithm. The spectrogram images were generated for the same time intervals to test the ORB prototype. Table 4 shows the results.

This experiment investigated an example of the minimum time for each algorithm to accurately identify the query for various lengths of the audio sample. From the results, the Shazam algorithm accurately matched the sample audio when it was a minimum length of 2 s. However, the confidence was not as high as the number of matches the 15- or 20-s clip would typically have. This was so because the shorter the sample clip, the less the audio would be available to be fingerprinted, and therefore fewer fingerprints would be extracted and matched.

The ORB prototype was able to accurately identify an audio sample with a minimum of 10 s in length. This would be due to the shorter audio samples producing smaller spectrogram images, which would mean that there are less features to be detected and matched. The Shazam algorithm generally produced better results for this experiment and hence fine granularity. The fingerprint extraction method of the Shazam method entails the use of the raw spectrogram data to compute the audio fingerprints. The ORB method uses the spectrogram images, where the data is limited by the resolution of the image (compression). This is where the Shazam method would produce better scores as a result.

The tests were successfully executed and results were seen to justify the research done. Also, some tests compared the two algorithms performance.

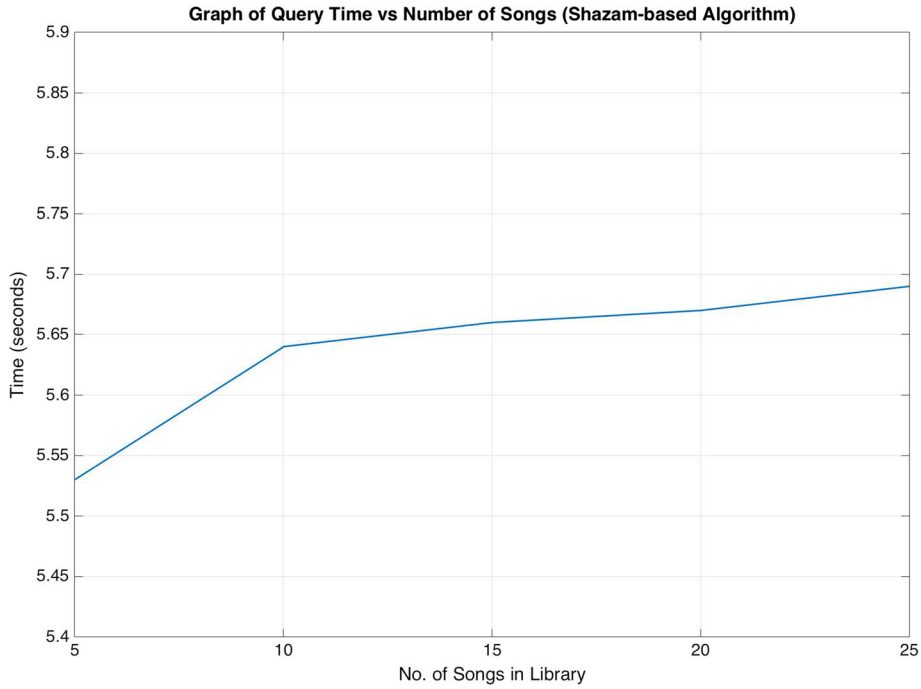

Fig. 9 Graph of time taken for query to search and match songs in database (Shazam). In this figure, results were shown for the time taken for a query to be searched and matched. This was done for libraries of songs. This was done for the Shazam algorithm. The results showed an approximately linear relationship, although five songs took a shorter time to for the audio sample to be queried

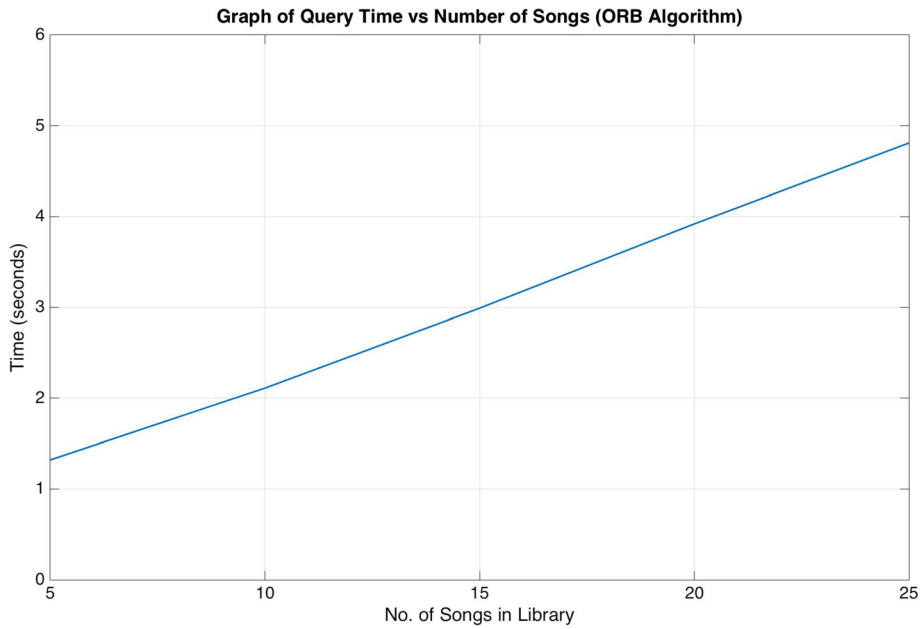

Fig. 10 Graph of time taken for query to search and match songs in database (ORB). In this figure, results were shown for the time taken for a query to be searched and matched. This was done for libraries of songs. This was done for the ORB prototype. The results showed a linear relationship, but it was seen that it had a larger time difference for the query process to be executed as the song library increased

Table 4 Test for granularity of Shazam and ORB prototypes

Sample query length (seconds)	Shazam-based algorithm		ORB-based algorithm	
	Successful match	No. of matches	Successful match	No. of matches
1 s	No	–	No	–
2 s	Yes	66	No	–
3 s	Yes	77	No	–
4 s	Yes	123	No	–
5 s	Yes	145	No	–
10 s	Yes	240	Yes	59
15 s	Yes	308	Yes	131
20 s	Yes	816	Yes	166

5 Conclusions

In this paper, a robust audio identification technique was proposed and compared to the Shazam method [7]. As proven in [9], the ORB method is more computationally efficient and faster than the SIFT method as proven by Rublee et al. [9]. As it was extensively proven by Rublee et al. [9], the computational efficiency for ORB was superior to that of SIFT with comparable accuracy. The consideration of testing the Shazam method for comparison [7] was due its current prevalence in the industry. Using a spectrogram image, the ORB descriptor was used to extract the audio signal features and a Brute-Force Matcher was used to match the sample query with the database. The main focus of this work was to highlight the robustness of this algorithm in terms of speed, pitch, tempo, and noise and to compare to the Shazam algorithm. Further experiments were conducted to determine the performance of the algorithms on softer metrics such as compression, reliability, granularity, scalability, and system speed.

The results of performance were comparable and better than the Shazam algorithm for all robustness tests including noise distortion, pitch shifting, time stretching, and time scaling. The match rates were generally high and maintained exceptional performance compared to the Shazam algorithm for the tested distortion ranges. It was also noted that the both algorithms performed similarly using other metrics such as compression, reliability, granularity, scalability, and system speed.

Further work can be done to improve to the efficiency of data storage and retrieval using the ORB method. The ORB implementation in this paper is an experimental prototype used to achieve audio identification based on the spectrogram image. Improvement in the overall performance can be investigated to increase computational efficiency and accuracy of the algorithm. This method resulted in similar accuracy with a larger dataset when tested. However, due to limited resources and time, further tests can be done using a larger database with the inclusion of the various distortions.

Other image processing techniques may be investigated to be implemented such as image thresholding and filtering to enhance features of the spectrogram image so that even if the image is further compressed, the features would still be apparent. The ORB prototype can also be improved in order to retain the robust features while recording the sample for query using the microphone as quality may be lost this way and would affect the spectrogram image. Since initial experiments conducted used the Brute-Force matching method, other more efficient matching methods can be investigated for the comparison of spectrogram images (e.g., nearest neighbor, histogram comparison). This would be a novel and innovative approach to audio fingerprinting.

6 Endnotes

The experimental setup included the available computer hardware which was a 1.7 GHz Intel i5 processor with 4 GB RAM. The software environment used was Python 2.7 and OpenCV 3.0 (for ORB) on a Linux system. The program and various tests were coded and executed using Python script modules.

Acknowledgements

The authors would like to acknowledge the contribution of Aniel Maharajh for his assistance in the experimental tests conducted.

Authors' information

DW completed his B.Sc. Degree with First Class Honors at the Department of Electrical and Computer Engineering from the University of the West Indies (UWI), St. Augustine Campus (STA) in 2016. His area of interest includes audio signal processing, image processing, software development and also in audio engineering. AP is a lecturer in Computer Systems at the Department of Electrical and Computer Engineering. He is a member of the IEEE. He completed his M.Sc. in Communications Systems with Distinction in 2005 and his Ph.D at the Department of Electrical and Computer Engineering of the University of the West Indies (UWI) St. Augustine Campus (STA) in 2012. His area of interest includes sport engineering, image processing and video analysis of sport broadcasts. JS completed his B.Sc. Degree at the Department of Electrical and Computer Engineering from the University of the West Indies (UWI), St. Augustine Campus (STA) in 2015. He has a deep interest in software development and mobile application development. He also has a keen interest in image processing and researching innovative and efficient methods in image processing applications for mobile phones.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Received: 5 December 2016 Accepted: 29 June 2017
Published online: 11 July 2017

References

1. M Huber, Music reception in the digital age—empirical research on new patterns of musical behaviour. Int. J. Music. Res. 2, 6–35 (2013)
2. A Lin, Understanding the market for digital music. Stanf. Undergr. Res. J. 4, 50–56 (2005)
3. Sen A (2010) Music in the Digital Age: Musicians and Fans Around the World "Come Together" on the Net. Glob. Media J. 9.16
4. V Venkatachalam, L Cazzanti, N Dhillon, M Wells, Automatic identification of sound recordings. IEEE Signal Process. Mag. 21, 92–99 (2004). doi:10.1109/MSP.2004.1276117
5. Y Liu, HS Yun, NS Kim, Audio fingerprinting based on multiple hashing in DCT domain. IEEE Signal Proc. Lett. 16, 525–528 (2009). doi:10.1109/LSP.2009.2016837
6. S Baluja, M Covell, Waveprint: efficient wavelet-based audio fingerprinting. Pattern Recognit. 41, 3467–3480 (2008). doi:10.1016/j.patcog.2008.05.006
7. Wang A (2003) An industrial-strength audio search algorithm. ISMIR 2003, 4th Int Conf Music Inf Retr. doi:10.1109/IITAW.2009.110
8. Zhang X, Zhu B, Li L, Li W, Li X, Wang W, Lu P, Zhang W (2015) SIFT-based local spectrogram image descriptor: a novel feature for robust music identification. EURASIP J. Audio Speech Music Proc. doi:10.1186/s13636-015-0050-0
9. Rublee E, Rabaud V, Konolige K, Bradski G (2011) ORB: An efficient alternative to SIFT or SURF. ICCV '11 Proceedings of the 2011 International Conference on Computer Vision 2564–2571. doi:10.1109/ICCV.2011.6126544
10. P Cano, E Batlle, T Kalker, J Haitsma, A review of audio fingerprinting. J. VLSI Signal Proc. 41, 271–284 (2005). doi:10.1007/s11265-005-4151-3
11. A Wang, J Smith, *System and methods for recognizing sound and music signals in high noise and distortion* (Publ, U.S. Pat. Appl, 2006)
12. Cano P, Batle E, Kalker T, Haitsma J (2002) A review of algorithms for audio fingerprinting. Proc 2002 IEEE Work Multimed Signal Process MMSP 2002 169–173. doi:10.1109/MMSP.2002.1203274
13. Haitsma J, Kalker T (2002) A highly robust audio fingerprinting system. Proc 3rd Int Soc Music Inf Retr Conf 107–115. doi:10.1.1.103.2175
14. S Kharat, S Modani, S Pujari, M Shah, Audio fingerprinting and review of its algorithms. Int. J. Comput. Technol. Appl. 5, 662–667 (2014)
15. S Lee, D Yook, S Chang, An efficient audio fingerprint search algorithm for music retrieval. IEEE Trans. Consum. Electron. 59, 652–656 (2013). doi:10.1109/TCE.2013.6626252
16. Burges C, Platt J, Jana S (2002) Extracting noise-robust features from audio data. Acoust Speech, Signal … 1:I-1021-I-1024. doi:10.1109/ICASSP.2002.5743968
17. Doets PJO, Menor Gisbert M, Lagendijk RL (2006) On the comparison of audio fingerprints for extracting quality parameters of compressed audio. Proc SPIE - Int Soc Opt Eng 60720L–60720L–12. doi:10.1117/12.642968
18. Covell M, Baluja S (2006) Content fingerprinting using wavelets. 3rd Eur Conf Vis Media Prod (CVMP 2006) Part 2nd Multimed Conf 2006 198–207. doi:10.1049/cp:20061964
19. A Wang, The Shazam music recognition service. Commun. ACM 49, 44–48 (2006). doi:10.1145/1145287.1145312
20. N Instruments, *Understanding FFT and windowing* (White Pap, In, 2015). http://www.ni.com/white-paper/4844/en/, Accessed 13 Mar 2016
21. J Figat, T Kornuta, W Kasprzak, Performance evaluation of binary descriptors of local features. Comput. Vis. Graph. (2014). doi:10.1007/1-4020-4179-9
22. Miksik O, Mikolajczyk K (2012) Evaluation of local detectors and descriptors for fast feature matching. Pattern Recognit (ICPR), 2012 21st Int Conf 2681–2684. doi:978-1-4673-2216-4
23. A Kulkarni, J Jagtap, V Harpale, *Object recognition with ORB and its Implementation on FPGA* (Theaccents, Org, 2013)
24. S Worner, *Fast Fourier Transform* (Numer. Anal, Semin, 2008)
25. M Calonder, V Lepetit, C Strecha, P Fua, BRIEF : Binary Robust Independent Elementary Features. Eur. Conf. Comput. Vis. 2010, 778–792 (2010). doi:10.1007/978-3-642-15561-1_56
26. PL Rosin, Measuring corner properties. Comput. Vis. Image Underst. 73, 291–307 (1999). doi:10.1006/cviu.1998.0719
27. HB Kekre, N Bhandari, N Nair, A review of audio fingerprinting and comparison of algorithms. Int. J. Comput. Appl. 70, 24–30 (2013)

Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH (“Springer Nature”).
Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users (“Users”), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use (“Terms”). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.
These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.
We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.
While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.
These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.
To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.
Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.
If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com