Query by Humming

Wasif Jawad Hussain*,180882
Indian Institute of Technology Kanpur
wasif@iitk.ac.in

Sumit Kumar Pandey*,180797
Indian Institute of Technology Kanpur
sumitkrp@iitk.ac.in

***Abstract*— A Query by Humming(QbH) system is aimed at retrieving songs from a database given a hummed query of it. The main idea behind a QbH system is that we generally perceive melody as how the pitches of successive notes relate to each other. We used various pitch extraction methods to obtain the note sequence for target songs and hummed query and then we used sequence matching algorithm to find the closest match and report a ranked list of songs from the database. Our method is currently supported for monophonic audio only and we will be looking to add support for polyphonic audio in near future.**

I. INTRODUCTION

Query by Humming is an important research topic in the area of audio-based search engines. It enables one to retrieve melodies by only humming the tune of it. The aim of our project was to come up with a working prototype of a QbH system.
Some challenges in building a QbH system are-

- Queries hummed by users differed substantially from actual melody in terms of pitch, tempo, etc
- Presence of background noise caused significant distortions to the actual melody data
- It is quite difficult to encapsulate all the properties of a melody from its audio file.

Handel [3] points out that the melodic contour is one of the most important methods that listeners use to determine similarities between pitches. Our approach has been built upon the idea that the sequence of difference between successive notes of a target song and its hummed query should match or be very similar to one another.
We currently use an alphabet scheme to model variation between pitches [‘u’, ‘d’, ‘U’, ‘D’, and ‘S’], representing the situations where a note is slightly above, significantly above, slightly below, significantly below or the same as the previous note. We experimented with various pitch extraction algorithms which are explained in section IV.
One method presents an additional challenge, The hummed query and the target song may vary significantly in terms of their duration. Thus we need effective string matching algorithms that could be used to evaluate similarity between strings of differing lengths. For our task we used the Dynamic Programming Algorithm which is explained in Section-V.

*Indicates equal contribution

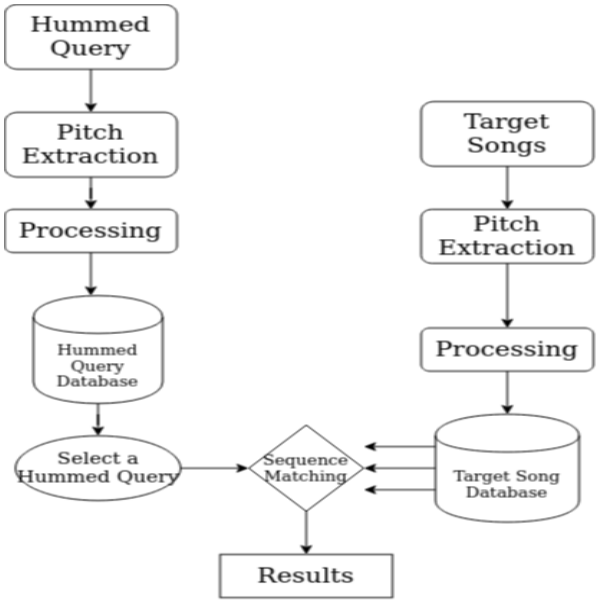

Fig. 1. Overview of our QbH pipeline

II. RELATED WORK

In the year 2003 a music retrieval system called TANSEN [1] was developed by IIT Bombay. It explains in detail about pitch extraction and the algorithm to be used for matching songs. Also in 2002 [4] , an end to end music search system was developed that could be used to support Query by Humming. A similar system was developed by Pascale Fung and Naziba Mostafa [2] in the year 2017. They used Convolutional Neural Networks to learn the features directly from raw audio signal and they used Hidden Markov Models to model the temporal aspect of note transcription. Their method performed much better than other feature engineering models. For Candidate Melody Retrieval they used Locality Sensitive Hashing to narrow down the list of candidates and then they used a combination of Dynamic Time Warping and Earth Mover Distance to do the final ranking of the candidates. Our model partly draws inspiration from their Candidate Melody Retrieval architecture.

III. METHODOLOGY

An overview of our QbH system is given in Figure 1. The notes of the query and target songs are transcribed using our transcription system. User selects a hummed query from a pre-transcribed database which is then passed onto

a candidate melody retrieval system. Here each query is compared against every other target song present in the pre-transcribed target song database. The output is the ranked list of melodies that are most similar to the input query. Our transcription and candidate melody retrieval systems are explained further in Sections IV and V respectively.

IV. Transcription System

We experimented with various pitch extraction algorithms. We highlight our observations and the salient features of each of them as follows.

A. YIN

Cheveigne [5] presented an algorithm for the estimation of the fundamental frequency F_o of speech or musical sounds. It is based on the well-known auto-correlation method with a number of modifications that combine to prevent errors. The algorithm has several desirable features. Error rates are about three times lower than the best competing methods, as evaluated over a database of speech recorded together with a laryngograph signal. There is no upper limit on the frequency search range, so the algorithm is suited for high-pitched voices and music. The algorithm is relatively simple and may be implemented efficiently and with low latency, and it involves few parameters that must be tuned. It is based on a signal model (periodic signal) that may be extended in several ways to handle various forms of aperiodicity that occur in particular applications. Finally, interesting parallels may be drawn with models of auditory processing.

B. YIN-FFT

The addtional term FFT stands for Fast Fourier Transform. This algorithm was derived from the YIN algorithm. In this implementation, a Fourier transform is used to compute a tapered square difference function, which allows spectral weighting. Because the difference function is tapered, the selection of the period is simplified. We found this method to be faster than the routine YIN algorithm.

C. Spectral Approaches(specacf)

Spectral/temporal pitch detection algorithms [6] are based upon a combination of time domain processing using an auto-correlation function such as normalized cross correlation, and frequency domain processing utilizing spectral information to identify the pitch. Then, among the candidates estimated from the two domains, a final pitch track can be computed using dynamic programming. The advantage of these approaches is that the tracking error in one domain can be reduced by the process in the other domain.

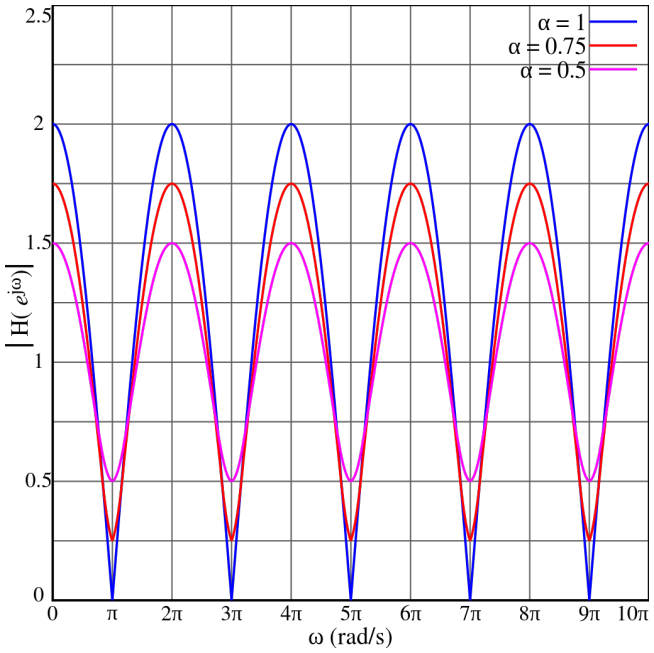

Fig. 2. Magnitude response for a feed-forward comb filter. Notice the comb like appearance

D. COMB-Filters

A comb filter is a filter implemented by adding a delayed version of a signal to itself, causing constructive and destructive interference. The frequency response of a comb filter consists of a series of regularly spaced notches, giving the appearance of a comb.
Gainza presented a technique [7] , for detecting the pitches of a polyphonic signal. The system utilises modified Infinite Impulse Response (IIR) comb filters, which are generated to ensure that a null (stop band notches) exists at multiples of note frequencies, and that a very flat pass band is present in the remain of the spectrum. Thus, the signal spectrum is not distorted after applying the filters to the audio signal, which is the case when using Finite Impulse Response (FIR) comb filters.
We experimented with two kind of COMB-Filters in our project-

1) Fast Harmonic Comb Filter(fcomb)- This was computationally very quick but the accuracy wasn't as good compared to its counterpart.
2) Multiple Comb Filter(mcomb)- This method implements spectral flattening, multi-comb filtering and peak histogramming. It gave us the best performance on all our datasets (particularly for the polyphonic target songs) and the margin with which scores differed for different target songs was the best using it.

Overall our model gave the best results using the M-COMB filter method. All the results presented were obtained using the same.

V. Dynamic Programming Algorithm for Candidate Melody Retrieval

We illustrate dynamic programming using the edit distance problem [8] . We assume a finite set of characters or letters, which we refer to as the alphabet, and we consider strings or words formed by concatenating finitely many characters from the alphabet. The edit distance between two words is the minimum number of letter insertions, letter deletions, and letter substitutions required to transform one word to the other. The edit distance of two strings, str1 and str2, is defined as the minimum number of point mutation required to change str1 into str2, where a point mutation is one of:

1) Change a letter
2) Insert a letter
3) Delete a letter

This algorithm averages pitch values within the 50 to 80 percentage of the duration of the note.

$$d_{i,j} = min \begin{cases} d_{i-1,j} + w(a_i, 0) & ;(deletion) \\ d_{i-1,j} + w(a_i, b_j) & ;(match\,/\,change) \\ d_{i,j-1} + w(0, b_j) & ;(insertion) \end{cases}$$

Fig. 3. Edit Distance Formula

The initial conditions are shown in *Fig.(4)*

$$d_{0,0} = 0$$
$$d_{i,0} = i$$
$$d_{0,j} = j$$

Fig. 4. Edit Distance Initial Conditions

A matrix $d_{i,j}$ is initialized with size [(*length of string*$_1$+1) x *length of string*$_2$ +1)]. W(a_i, 0) is the weight associated with the deletion of a_i, W(0, b_j) is the weight for insertion of b_j, and W(a_i, b_j) is the weight for replacement of element 'i' of sequence 'A' by element 'j' of sequence 'B'. The operation titled "match/change" sets W(a_i, b_j) = 0, if a_i = b_j and a value greater than 0 if a_i=b_j. The weights used here are 1 for insertion, deletion and substitution(change) and 0 for match. As an example, if two pitch contour strings *UDDSSUD and *UDDSUD are compared, the edit distance is 1.

VI. Experiments

We created three custom datasets each as follows-

1) Monophonic target songs and hummed queries both sung by the same user.
2) Monophonic target songs and hummed queries sung by different users.
3) Polyphonic target songs and hummed queries sung by different users

We tried to keep the duration of target songs and hummed query very similar to each other. This was because our string matching algorithm(DPA) depends strongly on the string length.
The results we obtained are summarised as follows-

Dataset	*YIN*	*YIN-FFT*	*SPEC-ACF*	*M-COMB*
1	80%	50%	50%	90%
2	80%	80%	20%	80%
3	20%	20%	20%	60%

We found the *YIN-FFT* method to be having the best computation time however it's accuracy for specific datasets was much lower than its counterpart M-COMB filter method. All the results presented in this paper were obtained using the M-COMB filter method.

VII. Techincal Details

For pitch extraction using the M-COMB filter method we kept the sampling rate to be the default **44100 Hz**, The hop size was fixed at **128 samples**, the buffer size was set to default at **2048**, pitch tolerance level was **not** set and the silence threshold was at default **-90 dB**

CONCLUSIONS AND SCOPE FOR FUTURE WORK

In this project we learnt about and experimented with various pitch detection algorithms and used them to extract fundamental pitches from our sound database. Then we used the Dynamic Programming Algorithm to match strings and used it to produce final rankings for our Candidate Melody Retrieval system. In the end we were able to come up with a good implementation of a Query by Humming system. Our model performed very well on songs of similar duration and in particular its performance on monophonic sounds was remarkable. In the future we plan to extend our work by researching on better string matching algorithms like Dynamic Time Warping(DTW) [9] to incorporate for inconsistencies in song duration. We also plan to use MIDI [10] file format for storing our target songs as they can better encapsulate the features of melody by storing the information from different instruments seperately.

References

[1] M Anand Raju, Bharat Sundaram, and Preeti Rao. Tansen: a query-by-humming based music retrieval system. *In Proceedings of the national conference on communications(NCC), 2003*

[2] Mostafa, Naziba & Fung, Pascale. (2017). A Note Based Query By Humming System Using Convolutional Neural Network. 3102-3106. 10.21437/Interspeech.2017-1590.

[3] Stephen Handel. Listening: An introduction to the perception of auditory events. The MIT Press, 1993

[4] Pauws, Steffen. (2002). CubyHum: A Fully Operational Query by Humming System.

[5] Cheveigné, Alain & Kawahara, Hideki. (2002). YIN, A fundamental frequency estimator for speech and music. The Journal of the Acoustical Society of America. 111. 1917-30. 10.1121/1.1458024.

[6] Zahorian, Stephen & Hu, Hongbing. (2008). A spectral/temporal method for robust fundamental frequency tracking. The Journal of the Acoustical Society of America. 123. 4559-71. 10.1121/1.2916590.

[7] Gainza, M. & Lawlor, Bob & Coyle, Eugene. (2005). Multi pitch estimation by using modified IIR comb filters. Proceedings Elmar - International Symposium Electronics in Marine. 233- 236. 10.1109/ELMAR.2005.193685.
[8] Dynamic Programming Algorithm (DPA) for Edit Distance
[9] Dynamic Time Warping
[10] Music Instrument Digital Interface
[11] All our code is publicly available and can be accessed from https://github.com/wasif1508/Query-by-Humming