Towards An Online Music Processor for Folk and Popular Musics and Its Educational Implications

(A rough draft)

Anh-Thu G. Phan, M.A., Ed.M. Music Education
Teachers College
Columbia University
New York, NY USA
agp2132@tc.columbia.edu

Center for Vietnamese Philosophy, Culture & Society
Temple University
Philadelphia, PA USA
anh.thu.phan@temple.edu

Thanh-Nhan Ngo, Ph.D. Linguistics
Courant Computer Science, Medical Language Processing
New York University
New York, NY USA
nhan@cs.nyu.edu

Center for Vietnamese Philosophy, Culture & Society
Temple University
Philadelphia, PA USA
nhan@temple.edu

***Abstract*—This paper describes a musical processor that takes a sonic object and turns its spectrographic data into a series of meaningful layers associated with different musical knowledge representations, in such a way that it can be understood, reproduced, played, compared, taught, etc. by everyone across cultures, regardless of their musical backgrounds. Any music audio file can be used as input. Within the scope of this paper, the authors focus on processing of musical audio files onto common platform of physical sound properties, in Hertz, Decibels, and milliseconds, graphically so that culturally dependent musical units such as notes, beats, measures, phrases, chords, and sections can be viewed in separate layers. Syntactic techniques, such as frequency of occurrences and adjacency are applied to musical units such as pitches and musical chords. They are key pitches in context and key chords in context. The results are then mapped onto circles of fifths which reveal the distinct patterns of each song, each section of one song, each artist, each genres, and each culture. Semi-automatic generation of layers of annotations on top of the spectrogram helps teachers to quickly discover and/or compare distinct properties of a song, while preparing lessons. Learners of all levels can choose the most most prominent patterns of the song to learn. This can also advance methods for preservation of future deeper studies of sonic objects.**

Keywords*—music processor, music representation, audio files, frequency, pitch, melody, chord, harmonic progression, musical form, circle of fifths, frequency of occurrences, distinct characteristics, key words in context, key pitches in context, key chords in context.

I. Introduction

For years, social sciences and humanities rarely involve the study of music.

Music and songs can be thought of as linear strings of pitches or syllables from performers to listeners. Music and linguistic units, such as pitches and syllables, are inherently not well-tempered. Notating non-western music in western notation has stripped away many intrinsic features. Adding these once lost features back to their context, the adjacent pitches and syllables, reveals culturally dependent characteristics of sonic artefacts.

Today, with the advent of artificial intelligence, esp. a deep understanding of natural language processing, and the massive use of digitized sonic objects, one can imagine a music processor that can put all music signals onto the same platform of spectrograms, measured uniformly by Hertz, Decibels and milliseconds, thus music can be free of well-tempered pitches. Although the well-tempered pitches are not matching all the real recorded sonic frequencies, the authors still respect their popularity and familiarity to the current music lovers. The well-tempered pitches' popularity are partially due to the ease for instrument tuning and for playing together as a groups. Thus in this paper, non-well-tempered pitches and well-tempered pitches are presented in different layers and superimposed on top of each other for further analysis and comparison.

A characteristic of well-tempered pitches is that by generating a successive intervals of fifths one after the others, one arrives at a full set of 12 chromatic notes of a western-scale. This paper will reintroduce the concept of the circle of fifths.

The linearity, adjacency, regularity, structure, unit oriented properties of music from pitches, phrases, sections are similar, but not identical, enough, to that of natural language processing which is a section of artificial intelligence. This tool helps address music in social sciences and humanities..

In this paper, the humble authors would like to present the first sketch of a music processor, associated with separate culturally dependent music knowledge bases (once called semantics) by applying different top-of-the-art computer music analyses to expedite, with extreme accuracy, the manual processes traditionally done by human.

II. Paper Organization

This paper is largely divided into two parts, one part using technologies to convert musical objects into physical data, measured by Hertz, Decibel, and millisecond (ms). From these units, a layer of musical clefs and a layer of manual insertion of beats, measures, phrases, sections, lyrics, and chords, are drawn. The second part is dedicated to display frequencies of occurrences of pitches and chords, key pitches in context to discover frequencies of recurrences of adjacent pitches, and adjacent key chords in context.

III. Obtaining Data

Data of this paper can be divided into two categories: phenomenal and conventional (or knowledge) data. Phenomenal data are data describing physical signals such as time, frequency, and intensity. Conventional data are symbols and theoretical approaches that have been used in music and languages in different cultures. Examples of conventional data are musical beats, pitches, phrases, sections, and conventional musical notation such as quarter notes, measures, treble and bass clefs, chords, etc. which exist only in the learned contexts of theoretical or cultural knowledge. To them we add frequencies of occurrences, key pitches in context, key chords in context, from which home pitches can be surmised, on relative pitch/chord relations can be mapped on a circle of fifths or/vs a circle of chromatics.

Whereas phenomenal data can be obtained solely by automated software, conventional data are obtained semi-automatically in consultation with a musical knowledge base and human experiences. Obviously, there is a role of individual human perception at play in listening to music. For example, the separation of sections may not always align with longest pauses in a musical recording. The pauses might serve other purposes such as creating tension by intentionally having the listeners wait, and be influenced by human perception.

The manual processes to obtain some conventional data are kept minimal and requires little musical training. These are tapping the beats along the recording, marking the beginning of musical phrases, marking the beginning of lyrical phrases, and marking the beginning of musical sections. Only the marking the beginning of musical sections require naming of sections for later structural analysis. These four manual processes should take four times the song length to complete, once for each.

IV. Technologies Used

Top-of-the-art technologies are chosen based on four criteria: user-friendliness, cost, and standardization.

A. Sound Analysis Software

1) Sonic Visualiser: a free standalone software (https://sonicvisualiser.org/index.html) that has been developed at the Centre for Digital Music at Queen Mary, University of London [1]. This program allows visualized representations of audio files such as soundwaves and spectrograms. It also allows overlay annotations on top of one another by manually adding new time points or using external plug-ins.

The follow plugins are used in this paper:

a) Chordino Chord Estimation (http://www.isophonics.net/nnls-chroma)

b) Melodia Melody Extraction (https://www.upf.edu/web/mtg/melodia?p=Download%20and%20installation)

c) Queen Mary Note Onset Detector (https://code.soundsoftware.ac.uk/projects/qm-vamp-plugins/files)

d) BBC Intensity https://github.com/bbc/bbc-vamp-plugins/releases)

2) MuseScore (Offline): a free standalone open-source software for music notation (https://musescore.org/en).

3) MuseScore (Online): (https://musescore.com/dashboard) an online platform allowing users upload, download, change tempo, transpose sheet music among MuseScore community members. It can also work as a mobile application, be connected to other personal websites, and converted into a Youtube video.

4) AnthemScore: a standalone or an online software (https://www.lunaverus.com/) to transcribe an audio file (MP3, WAV, etc.) into sheet music using western notations. It also has functions such as "playback at a slower tempo, transposing, setting the lowest and highest note, choosing clefs, and linking to an external sheet music editor" [2].

5) Praat: a free standalone software (http://www.fon.hum.uva.nl/praat/) that was developed at the University of Amsterdam [3]. This is designed for statistically analysis of spectrum, pitch, formant, intensity, etc. of voice. However, it also works well with non-vocal sounds and allow for precise cropping of an audio file in milliseconds.

B. Web programming language PHP & SVG

1) International, world wide web, archival, librarian multilingual standards: The online world wide web standards are, specifically, eXtensible Markup Language (XML) and its associate HyperText Markup Language (HTML) and Uniform Resource Locator (URL). The up-to-date versions of multilingual standards are Unicode and ISO/IEC 10646. The music processor also adheres to the Open Archive Initiative/Object Reuse and Exchange (OAI/ORE), with Dublin Core Metadata Initiative (DCMI), especially the Dublin Core Metadata Element Set.

2) Our main web programming language is PHP (HyperText Preprocessor) which generates scalable and controllable layers in Scalable Vector Graphics (SVG).

3) The initial music processor runs modestly on a Dell T1650 workstation, with 16MB RAM and 1TB hard drive, accessible at http://mlp.cs.nyu.edu, hosted by Courant Institute of Mathematical Sciences, Computer Science Department for the Linguistic String Project medical language processor (MLP). It is currently running on Ubuntu 18.4 with PHP 7.2.

4) The music processor currently runs as a test with about 6,000 lines of PHP codes.

V. Organizing Data

Data comes from software cited above in spreadsheet format (tab or comma separated values), XML, MusicXML, and audio formats (MP3, WAV, etc.).

A. Unifying the Data

The use of multiple software leads to multiple measurement units when obtaining data. For example, there are more than four ways to measure time when dealing with these software. The first one is frame used by Sonic Visualiser. One frame is 1/44,100 second which is a standard sampling rate when converting analogue signals to digital ones. The second way is to use millisecond (ms) as in Praat. The third way is to use groups of milliseconds as measurement units, as in AnthemScore. For instance, users can set a unit to 5 ms or to 20 ms. The fourth way is to use 1/24 of a quarter note as a measurement unit as in MuseScore which is relative and can only be converted to absolute values by multiplying the metronome marking of the section in question.

A similar complicated measurement system has also used for pitches. The authors have to unify Hertz and a partial of a note into cents. (Note that a western semitone equals to 100 cents). Only intensity is consistently used in Decibel (dB).

B. Cross-checking the Data Among Software

The use of multiple software leads to multiple representations of same elements in multiple file formats. For example, spectrogram is presented in Sonic Visualiser, AnthemScore, and Praat. Pitches and intensity are presented in all software. Instead of picking the clearest and most portable data, the authors decide to plot the data of different software onto the same two-dimensional coordinates to cross-check the data among multiple pieces of software.

VI. A Test Case: Despacito by Luis Fonsi 2017

The video version of *Despacito* by Luis Fonsi (hereafter, simply *Despacito*) was chosen to guide the design of our folk and popular music processor. The design is componential, producing different layers.

A. The Folk Music archive system

The Folk Music archive system is built using the file hierarchy as part of an open data structure. This file hierarchy structure is reflected in the file naming convention, following the British Library format. For example, if the library contains a “folk music” folder, and in turn, has a “folk song” subfolder, then we may have “despacito” as its subfolder, with version “2017” as a subsequence subfolder. The Dublin Core HTML/PHP metadata file is located here. Thus, we have

… lib/folkmusic/folksong/despacito/2017/ file structure,

… folkmusic_folksong_despacito_2017_000.php metadata

Each field of the Dublin Core (DC) metadata is tagged following the DCMI field tags, such as <dc:title>, <dc:subject>, etc. with language subfields <dc:subject_spa>, <dc:subject_eng>, following ISO 639-3 code naming standards.

B. Substructure of a song

Below each song, we may have a JPEG subdirectory for images related to the song, data subdirectory for raw data from the song, such as spectrographic data, phrase data, lyric data, etc. in ASCII format, including MusicXML, and audio data in mp3, wav, etc. These data are imported from freeware used, see above, to analyze the spectrographic data.

C. Frame,

Frame is a common feature that pairs all data collected. Frame can be converted to milliseconds. Each frame is 1/44,100 or 1,000/44.100 millisecond sample rate.

D. Spectrographic data plotted in SVG as the background layer

To obtain the song spectrogram, an audio file is imported in to AnthemScore. As a test case, the authors used Despacito by Luis Fonsi (https://youtu.be/kJQP7kiw5Fk) which is 4 minutes and 42 seconds long. AnthemScore generates a spreadsheet of spectrographic data. There are choices to obtain data from every 1 ms to every 10 ms, in terms of time; and there are choices to obtain data every 5 cents to every 25 cents, in terms of pitch. The authors chose to test two sets of data: a fine one of 1 ms by 25 cents and obtained a spreadsheet of 351 pitch columns by 281,522 one-ms rows. The other one is crude of 10 ms by 25 cents and obtained a spreadsheet of 351 pitch columns by 28,153 ten-ms rows. Each cell contains value of intensity in Db’s. The former is 774 MB and the latter, 94 MB in size.

Reading the fine spreadsheet took 130 msec processing time, data show 26.7171 to 4,308.66 Hz range, intensity range from 0 to 17,872 dBs. Reading the crude spreadsheet took 15.5 msecs, with the intensity range of 4,246.1 dBs.

The data are plotted pixel by pixel onto an SVG mat, using 255 shades of grey for Db data. The 10 pieces of the Mime format[1] of the spectrogram png (Portable Network Graphics) generated by Sonic Visualiser, 12 MB in size, for *Despacito* are patched together, with intensity mapped to colors navy (lowest dBs) to white (the highest) (see Fig. 1).

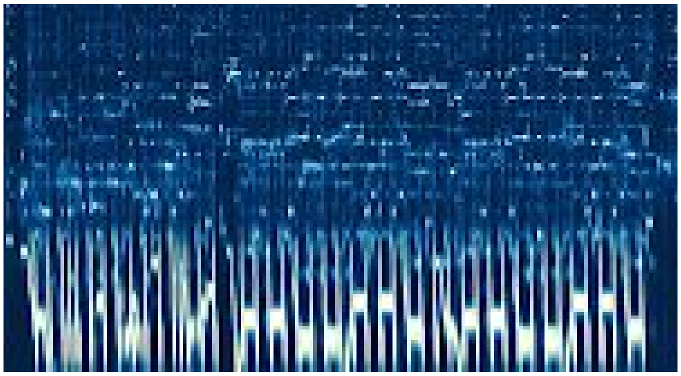

Fig. 1. An example of strectro- graphic piece of *Despacito's* recording.

E. *Musical grand staff and piano keyboard as SVG Layer 1*

The 351 columns in Sonic Visualiser's generated spectrographic spreadsheet range from C0 (0 cent at 26.7171 Hz) to B8 (10,700 cents at 4,366.08 Hz), while the piano 88 keys range from A0 (900 cents) to C8 (9,600 cents). This allows a treble and a bass clef to be drawn on top of the spectrographic Layer 0 (See Fig. 2). The staff lines are drawn horizontally from the beginning to the end of the time coordinate.

This Layer 1 helps readers to identify items on the spectrographic image to familiar western notational system.

Fig. 2. 88-key piano and the grand staff covering the pitch range of *Despacito* collected by Sonic Visualiser and AnthemScore.

F. *Layer 2: Representation of Beats, Measures, Chords, Phrases and Sections in Despacito spectrogram*

These data are generated by Sonic Visualiser in frames with manual inputs, in XML format. They are drawn and superimposed on Layer 0 and Layer 1 to show how human perception paired with the physical data.

On top of each panel of spectrogram, the bar graph of dB values at each frame in the song is drawn in red with the white dots under-foot representing the percussive onsets (see Fig. 3). The beats are yellow down arrows inside red transparent measure boxes. The chords are displayed in cyan color at their time positions (see Fig. 4).

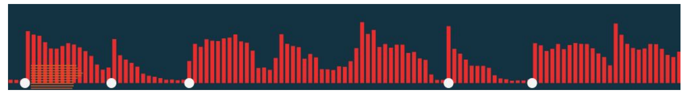

Fig. 3. An example of Despacito intensity and percussive onsets.

Fig. 4. An example of *Despacito* markings: down arrows for beats, rectangular boxes for measure, cyan letters for chord symbols.

At the bottom of the spectrogram Layer 0, the phrase orange boxes containing the lyrics are closest to the spectrogram. Last but not least, the lowest light brown bars represent sections in the song (see Fig. 5).

Fig. 5. An example of Despacito markings: orange rectangular boxes for lyrics and light brown rectangular boxes for sections.

Manual marking creates 26 sections with 11 music forms (labelled from A to L), 83 phrases, 113 lyric expressions, and 22 chords.

G. *Layers for sum of intensity of a measure, a beat, a chore, a phrase and a section.* Specifically, they are:

- Layer 3 dBs by Measures
- Layer 4 dBs by Beats
- Layer 5 dBs by Chords
- Layer 6 dBs by Phrases
- Layer 7 dBs by Sections

[1] Multipurpose Internet Mail Extensions (MIME) is an internet standard used mostly in emails, usually together with PNG images.

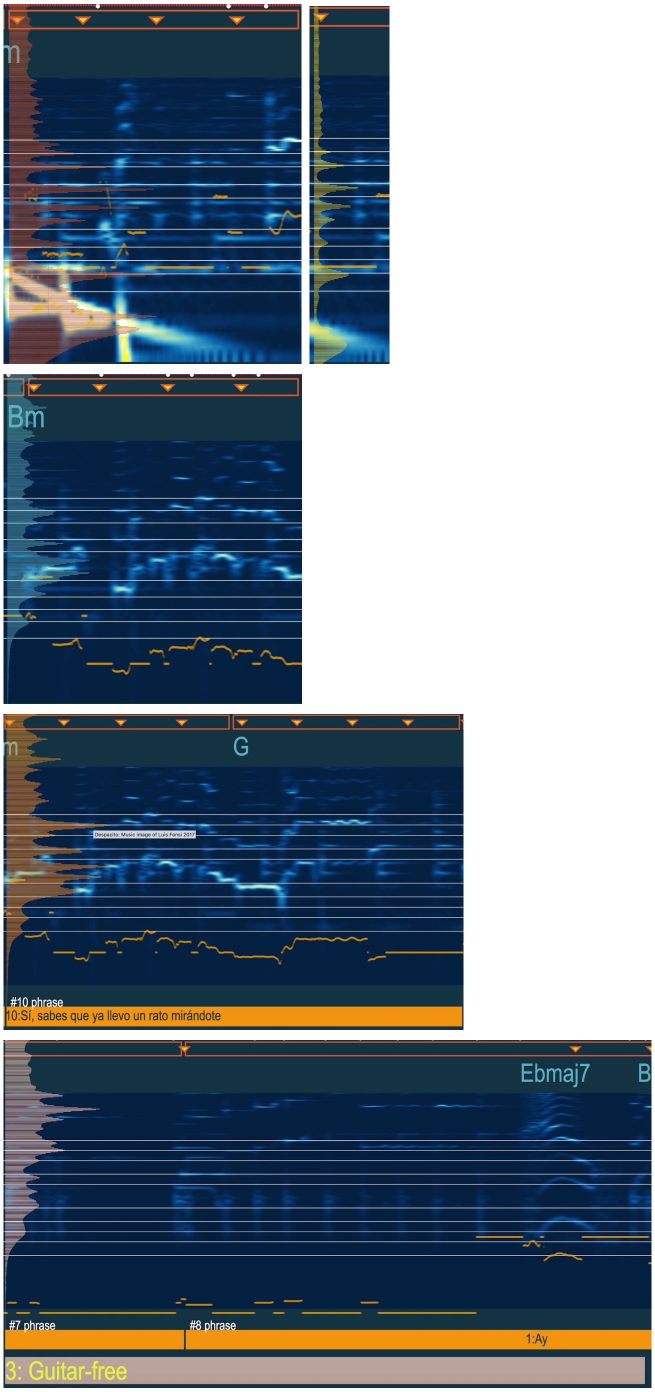

Fig. 6. Examples of layer 3 (top left), layer 4 (top right), layer 5 (second row), layer 6 (third row), layer 7 (bottom) over *Despacito* spectrogram image.

These layers collect the sums of all dB values in a beat, a measure, a chord, a phrase and a section, at their specific timestamps, to reveal the strongest pitches within, using the pitch markings of the staff and cent marking on the vertical axis.

These layers are intended to reveal differences in the manual perception against the raw data.

H. Layer 8, music sheet of Despacito according to AnthemScore

This is a estimate by AnthemScore, producing a MusicXML sheet. The notes are displayed with Layer 0, Layer 1 and Layer 2 to pair the estimates with other manual data.

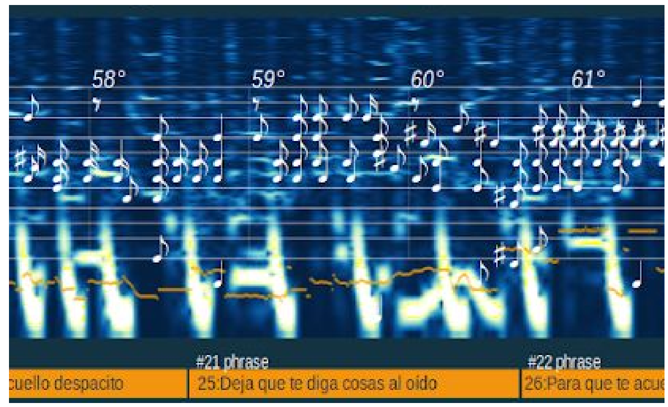

Fig. 7. An example of music notation on top of the spectrogram

I. Layer 9, AnthemScore section sheets, and MuseScore estimates are displayed with the original video playing at the section starts and ends

This layer is most active. The viewer can play Despacito 's video section by section, while MuseScore interactive music sheet generated by AnthemScore.

Layer 0 to 9, currently taken about 235 seconds by the music processor, can be viewed at

http://mlp.cs.nyu.edu/vietmusic/au2spec.php.

J. Analyses based on frequencies of occurrences, key chords in context kcic(n), and visual representations on the Circles of Fifths.

Despacito has a range of 47 pitches, from B0 to C#7. D4 occurs 508 times, the most frequent, B3, 353 times, F#4, 241 times, A3, 201 times, then E4, 173, etc. They are sorted and put on the music clefs, as well as positioned on a circle of fifths with their frequency bubbles as radii appearing immediately outside. This representation shows that even though the song is consisted of all 12 pitch classes, they are strongly divided between central (very high frequencies of occurrence) pitches, or just decorative ones. Further analysis confirms this observation (see Fig 8).

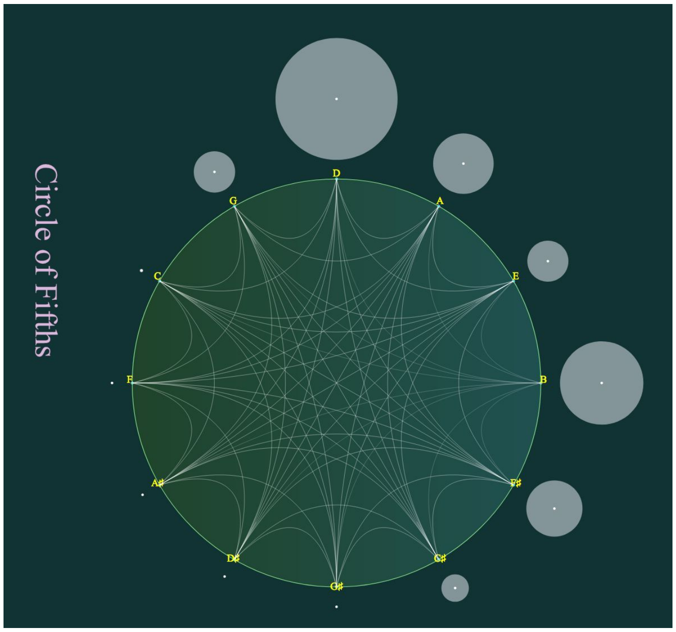

Fig. 8. Circle of fifth representing the frequency of occurrences of pitch classes of *Despacito.*

When the song is evaluated sections by sections, it turns out that the song now has the range of 60 notes, from B0 to D7. The same 4 notes are central to the song: D4, occurring 612 times, B3, 433 times, F#4, 292 times, and A3, 243 times, etc. Their class positions are more prominent in the circle of fifths.

The same examination on frequency of occurrences of pitches is done between sections. This analysis shows the tendencies of section movements towards each other as clues to identification of musical forms.

Here the concept of key pitches in context kpic(n), and key chords in context kcic(n), $1 < n < m$, where m is the total number of pitches or chords in a section. Kpic(n) and kcic(n) are the derivatives of key word in context (kwic(n)) that helps immensely to put a pitch and a chord back to their context.

A sample of kcic(2) is shown below on the circle of fifths with their frequency of occurrences. It is immediately obvious that *Despacito* has 4 prominent chords, D major, G major, A major and B minor.

Significantly, the chord progressions are B minor to G major (occurring 22 times), G major to D major (18 times), D major to A major (18 times), and finally A major to Bm major (10 times). While B minor to G major is most frequent, there is no G major to B minor progression (see Fig. 9).

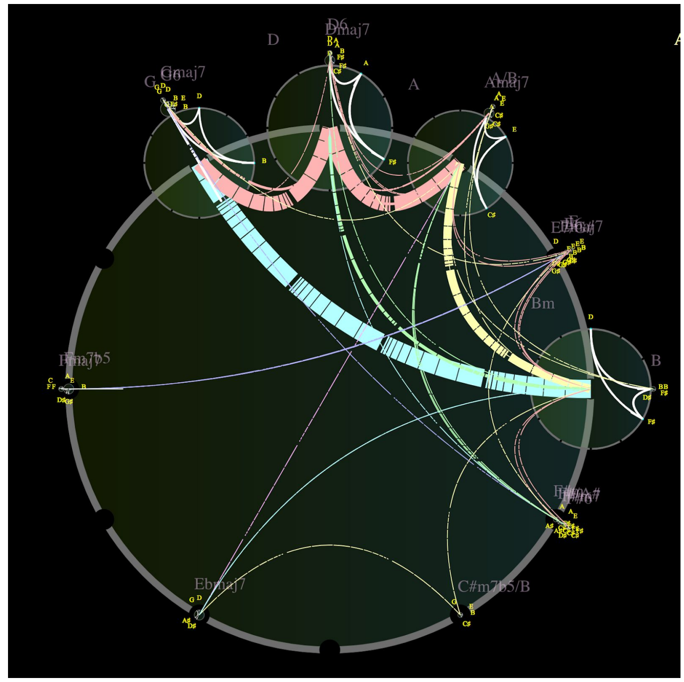

Fig. 9. A circle of fifth representing the frequency of occurrences of key chords in context (kcic(2)) of *Despacito.*

K. Song section analyses based on frequencies of occurrences, and their distributions on the circles of fifths and their music forms

In this column, the sections are displayed with frequencies of pitches on the music clefs and their proper positions, as well as their pitch classes on the circle of fifths, together with their music forms. They are divided into 26 groups, laid out sequentially from top to bottom. The changes in the circles of fifths visually help viewers to see characteristics of each section associated with their forms.

L. Lyrics in phrase bars

In this final (for now) panel, another way of data presentation can be viewed in a familiar manner, as a phrase by phrase layout, containing lyrics with chords associated with their proper lyric syllables. This presentation is a shorthand way for viewers to sing along without being bothered with music clefs and music pitch notes, measures or beats.

Items J. to L currently taken about 20 seconds by the music processor, can be viewed at

http://mlp.cs.nyu.edu/vietmusic/au2ana.php.

M. Current Issues

At the moment, the music processor adopted temporarily the use of the best spectrogram analyses of current software. This decision creates temporary problems in the discrepancies of data units, and requires data unification into milliseconds and cents. There are other issues such as best strategy for data presentation in SVG when the entire process for Despacito producing over 330MB to browsers.

1) Phrase breaks and lyric breaks are marked before the starts of the Section breaks.

The music processor is in the first design phase. The authors choose to have manual intervention begins with phrase and lyric breaks which are the scaffolding steps before identifying the section breaks. Therefore, there are chances that users would mark the beginnings of a musical phrase or the beginning of a lyrical sentences slightly ahead of the beginning of the sections. This will be fixed by an algorithm in later version of the processor.

2) Sizes of the Data.

The music processor begins to modestly scale up to over 4 minutes with 281,522 x 351 table of 98,814,222 data points. The processor is able to map them into an SVG mat. The first spectrographic and manual onset identification part of 3 layers took 235 seconds together on an SVG mat. They can be, and will be processed separately according to the preference of viewers. The separate second part now takes 20 seconds to produce together 5 subparts, which shall be presented also separately according to the preference of viewers.

3) AnthemScore

One difficulty which is obvious is the consistency of AnthemScore. It produces different scores when analyzing full *Despacito* against analyzing *Despacito* section by section, not only in the numbers of notes. The resolution for this problem may not come quickly or easily.

VII. Extracting Pertinent Characteristics For Users

A. Beginners

This paper's approach allows beginner music learners to grasp the gist of a song without years of ear training or music reading. These are a few examples. First, following the melodic contour and lyrics, beginner music learners understand where to raise or lower their voice. Using this contour to sing or play in tune without a teacher is possible. Second, learners look at the length of the phrases and will breathe or take a break appropriately while performing. If the learners change their mind, they can manually change the phrase or section breaks and obtain different analyses. Third, the structure of the song shows exactly how many times a section is repeated, thus aiding to the memorization process for learners.

B. Experts

This paper's approach presents a "fullest" sonic data into visual and aural representations. By fullest, the authors mean to preserves all the phenomenal signals that are often omitted in traditional notation. For example, using the chromatic scale (12 pitches) in western notation ignore all the sounds in between well-tempered pitches. In fact, the sounds of a songs appear continuously from the lowest to the highest frequencies. These "in-between" pitches makes Indian music sounds distinct from Vietnamese music, for example. Thus examining the spectrogram will reveal the microtones that the western notation often ignores.

Additionally, the representation of intensity also outperforms the dynamic notion (i.e., *piano* vs *forte* symbols). Because one chord could be played in infinite manners, the intensity in Layers 3 to 7 inform players to shape the chords to best match the recording.

C. Everyone in Between

For most music learners that fall between beginners and experts, this paper's approach allows them to enjoy music of their favorite in multiple learning styles. For example, they can listen to each section and play along if they are a aural learner. If they are a visual learners, they can follow the notations in *MuseScore* section.

VIII. Educational Implications

The main goal of this paper is to work toward a model of music teaching and learning that would be more generally accessible, but not in the sense that it would sacrifice the contents and quality of musical experiences. This model heavily relies on, among other pedagogical considerations, using folk and popular musics, and digital technologies. The authors have taken a test case the most viewed Youtube video, *Despacito*. The target audience is the underprivileged community who otherwise would not have time and resources to access music, for the authors believe that social justice is promoted best in this audience.

Over decades of teaching and learning musics, the authors strongly believe in the possibility of developing alternative modes of music education that reaches a wider range of students, across cultures, with lower cost, less time consuming, yet maintaining the quality of musical experiences.

The use of folk and popular musics has multiple advantages. First, folk and popular musics by its very nature has a direct appeal to people, even people without prior musical training, extensive knowledge, nor expensive musical instruments. Using folk and popular musics repertoires from the students' origins take advantages of what students and their families already know and promotes the principle of student-centered approach. Second, being able to reach out to particular community groups, especially communities that are poor, underserved, underrepresented, and disenfranchised, who do not have access to musical instruction, our proposed approach goes beyond knowledge preservation and transmission. Specifically, performing and understanding their own music well will help lessen apprehensions of the students and their families that music education is reserved only for the privileged. Finally, through folk and popular music materials, the community's social, political, and historical values are re-lived, thus cultivating the sense of pride in younger students because folk and popular musics are often distilled with the community's cultural traditions. All of these are hoped to enhance the sense of empowerment.

The second element is the use of digital technologies to facilitate teachers and student to be successful in developing the musical skills and knowledge, most efficiently in the following three ways. First, the music processor the authors develop provides a graphical analogue of music that bypasses the need for specialized musical notation and symbols. Graphical representation is especially suited for teaching musics whose tonalities do not often fit well with the western chromatically-based tonal system. In this sense, these graphs help describing the music inputs faithfully with visual aids. After that, the music processor proceeds to calculate prevailing rhythmic and melodic patterns of a songs by constantly comparing units of data based on the theory of linguistic string grammar and deep learning. These patterns serve as basic blocks for improvisation and stylistic recognition. Finally, it allows students to record their singing or playing and superimpose the graphical representation of their own version to those that they aim to model after or compare with. This live and instant feedback increases quality outside-of-the-classroom practice time and decreases the need to have teachers' presence on trivial mistakes. Teachers are then able to have more time focusing on accommodating other peculiar student needs, encouraging individual interpretations, and providing contextual knowledge.

This music processor can provide flexible arrangement of any songs, regardless of the instruments played and the levels of learners. This can be used by self-taught students, by teachers as a teaching aid, and by researchers as a researching tool.

Acknowledgment

The authors thank the Computer Science Department of the New York University Courant Institute of Mathematical Sciences and the Linguistic String Project's Medical Language Processing. The authors further thank the Center for Vietnamese Philosophy, Culture & Society of Temple University which provides computer resources and part of Henry Luce Grant No. 270380 / Expanding Opportunities for Research on Vietnam, for Vietnamese Folk Music Studies. And finally, the authors thank Ambassador Nguyễn Phương Nga and the Cultural Section of Permanent Mission of the Socialist Republic of Vietnam to the United Nation who donated a Dell workstation for the Folk Music Studies project.

References

[1] C. Cannam, C. Landone, and M. Sandler, "Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files," in Proceedings of the ACM Multimedia International Conference, Firenze, Italy, October 2010, pp. 1467-1468.

[2] Lunaverus, AnthemScore, 2018. Retrieved at https://www.lunaverus.com/

[3] P. Boersma, "Praat, a system for doing phonetics by computer," *Glot International,* vol. 5:9/10, pp. 341-345.