Real-Time Conducting Animation from Sheet Music

Andrea Salgian
Department of Computer Science
The College of New Jersey
Ewing, NJ, USA
salgian@tcnj.edu

Kseniya Rychkova
Department of Computer Science
The College of New Jersey
Ewing, NJ, USA
ksusha.rychkova@gmail.com

ABSTRACT

In this paper we present a system that extracts conducting information from sheet music encoded in a MusicXML file to generate a real-time animation that conveys the gestures that would be performed by a conductor. The information includes time signature, tempo, dynamics, and entrance cues. The approach could form the basis of an educational virtual conductor application that educates budding musicians, or could be used as a means to augment an orchestral performance with synchronized visual displays.

INTRODUCTION

Music-based human-computer interaction applications have seen increased interest recently, with both musicians and audiences showing more acceptance of computers supplementing classical performances. Technology allows people with little to no musical training to easily interact with instruments and even immerse themselves in a virtual orchestra.

Conducting is easier to address in a camera-based application than playing an instrument, since conductors move their hands freely and do not directly manipulate a rigid instrument. Various tracking technologies permit the creation of installations such as the Virtual Conductor at Vienna's House Der Musik museum [1], where even the novice user can enjoy the experience of conducting (a virtual model of) the Vienna Philharmonic.

The opposite problem, of a virtual or robotic conductor conducting a real orchestra has also been tackled, from Honda's cute ASIMO robot conducting the Detroit Symphony Orchestra performing "The Impossible Dream" in 2008 [2], to the rather creepy Android Alter 3 robot conducting a piece called "Scary Beauty" in 2020 [3]. However, these robots had their gestures preprogrammed, copying a human conductor and without using any musical knowledge. More work is needed to develop robots that could conduct any musical piece without the need for reprogramming, just based on sheet music. But what would be the applicability of such a robot? Why would a human musician want to be conducted by a robot, and, after the novelty wears off, why would such a performance be interesting to the public? Perhaps the most obvious application would be an educational one, where a robot (physical or animated) could tirelessly conduct the same piece over and over again for the benefit of students. But another, possibly even more useful application would be to augment the performance of a human conductor and their orchestra with features that can entertain and educate the audience.

Orchestral performances of classical music have been registering a significant drop in attendance [4], as younger audiences are quickly losing interest. Orchestras around the world are augmenting their performances with video projections [5]. The challenge is to find suitable image content that doesn't distract from and is synchronized with the music. The Philadelphia Orchestra has been utilizing LiveNote, a mobile app that concert goers could run on their mobile devices to receive real-time program notes, including text and translations for vocal works [6].

However, most of these examples require reprogramming or redesign for every musical piece, as systems that get information from a musical score are still very rare. One system [7] uses music stored in MIDI files. This has its limitations, as MIDI files were not intended for musical notation, and therefore have to be manually annotated with information that is not available in the notes themselves, such as dynamics, articulation, and even entrance cues.

In this paper we present a system that extracts conducting information from sheet music encoded digitally in MusicXML format, and generates an animation that conveys conducting gestures for tempo, dynamics, and cueing, in real time, synchronized with the music that is to be played. While the animation is based on the classic gestures performed by conductors, most notably the various time signature patterns, the approach represents just a starting point that can be used to create more artistic visual displays.

SYSTEM OVERVIEW

Our system creates and displays a real-time animation that is meant to accompany a musical piece, and reflects the gestures that a conductor would perform when conducting it. The animation can also be used as an artificial conducting tool. In the absence of musicians, for a better testing and viewing experience, the system also plays the music.

We use a MusicXML file to obtain conducting information. The user is asked to also convert the file to MIDI format using a third party software such as

MuseScore, so the application can play the music while displaying the animation. Both files, MusicXML and MIDI, are then supplied to the application by the user. Figure 1 shows the initial window and the MusicXML file selection window. The MIDI file selection window looks similar.

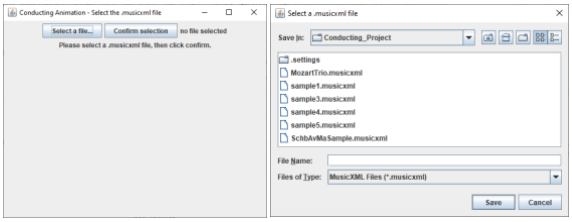

Figure 1. Left: initial window displayed to the use. Right: file selection window allowing the user to select a MusicXML file.

After the MusicXML and MIDI files have been selected and confirmed, the program quickly parses the MusicXML file and presents a JFrame window labeled "Conducting Animation". The animation begins in the paused state: the user is able to click anywhere in the window to begin playing, and can click again to pause it whenever they like. The paused state displays the piece's title and contributors at the top of the window, and a blue triangle representing a "play" symbol in the center of the window. Once the user clicks to play, the program displays the following items (see Figure 2 for reference):

- the title of the piece and contributors (authors), which remain at the top of the window throughout the animation;
- a row of colored blocks on the left, each representing an instrument part in the musical score, labeled with the name of the instrument intended to play the part; these blocks fade out whenever the instrument encounters a rest measure, and fade back in when the part has non-rest notes to play;
- yellow arrows above or blue arrows below the blocks indicating an increase or a decrease in dynamics; these arrows are labeled with the term extracted from the musical score music (pianissimo through fortissimo) and fade out for a visual effect;
- a circle moving in a pattern according to the music's time signature and tempo, displayed on the right side of the window; this display includes a fading trail to emphasize the pattern;
- the time signature, displayed at the bottom of the screen;
- the current measure number and the total number of measures, below the time signature;
- the tempo in beats per minute, below the measure number.

The MIDI audio is played alongside the animation, and is also paused when the animation is paused. Once the piece has finished, both the animation and the music stop.

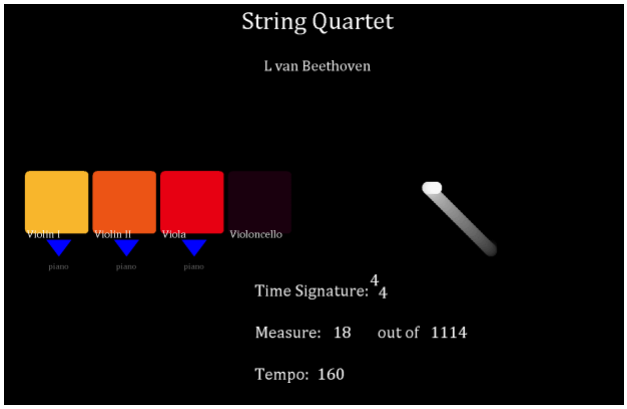

Figure 2. Screenshot of the Conducting animation for a String Quartet piece by Beethoven.

METHODOLOGY

1.1 Extracting Conducting Elements

Our algorithm uses the musical score encoded in a MusicXML file [8]. Like other XML formats, a MusicXML file contains readable text, formatted generally as follows: *<tag>value</tag>*. Specific tags are used for different elements of the musical score. General piece information is listed at the beginning of the file, and is followed by one music part after another (one for each instrument in the piece), with each part containing all of its measures in order.

Our application relies on the time signature, tempo, dynamics, and rest measures to generate an animation that conveys the information passed on by the conductor to the musicians. All of these are represented in a MusicXML file [9].

Time signature is one of the most important elements we need, as it defines the general conducting pattern. In a MusicXML file, this can be found in the first measure of each part. Beats per measure and beat value are specified separately. For example, a 3/4 time signature (meaning that there are 3 quarter beats per measure) would appear in a MusicXML as *<beats>3</beats>* followed by *<beat-type>4<beat-type>*.

Tempo is specified in a MusicXML file as one line within a measure, for example *<sound tempo="120"/>*. In this case, 120 is the number of quarter notes per minute. Notice that MusicXML specifies the tempo in quarter notes per minute, regardless of the value of the time signature, so occasionally there may be a need to convert from quarter notes to beats. The first measure will usually have the tempo listed, while later measures show tempo only if this changes from its previous value.

Dynamics representation is relative in music. Terms such as *piano* or *forte*, or wedge symbols for *crescendo* or *decrescendo*, do not correspond to a specific value, leaving the ultimate loudness or softness up to the conductor or musicians. The MusicXML representation either reflects this, or it can also use an explicit value representing the MIDI note velocity, if available.

The final conducting element that we used in our system is entrance cues after rest measures. While rest measures are obvious in sheet music because they don't have any notes, MusicXML encodes them either using a specific rest measure label for the entire measure, or by including rest notes in a regular measure. Both ways to describe rest measures are easy to work with.

1.2 Algorithm

Our system was built in Java, using SAXParser [10], a generic XML file parser, to parse the MusicXML file, several object classes to help organize the information, and ArrayLists to manage the objects. The animation is created in a JPanel, managed by a JFrame which implements Runnable in order to achieve a frame rate of exactly 100 frames per second. The time signature and tempo extracted from the MusicXML file are considered together with the frame rate to ensure the animation is synchronous with the music.

Our animation is inspired from the textbook gestures of a music conductor [11], and this is reflected in the structure of the code. Different components handle the parsing of the MusicXML file, generating an animation for the tempo using the pattern appropriate for the time signature (equivalent to the conductor's right hand gesture), and additional animations for cueing different parts of the orchestra for dynamics and entrance (this would be equivalent to the conductor's left hand). Figure 3 shows a simplified class diagram of the source code.

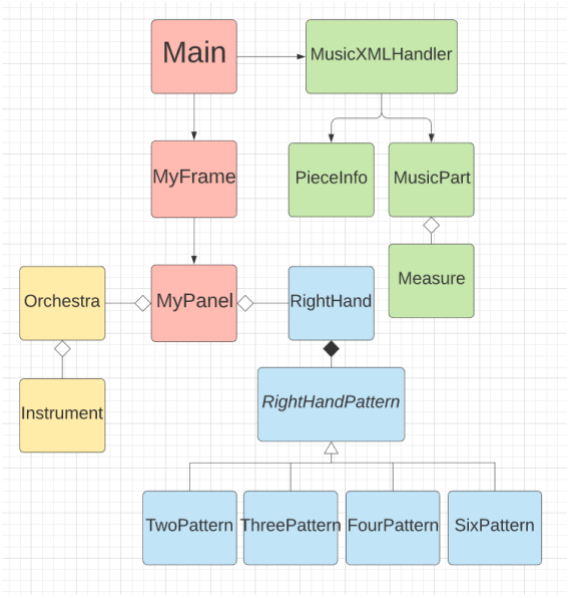

Figure 3. Simplified class diagram of the system.

The MusicXMLHandler uses the XML tags from Table 1 to parse and extract conducting information and store it in various variables in the system.

The RightHand class oversees the movement of the white circle on the right side of the animation's window, which corresponds to the right hand of the conductor, and indicates the tempo in a pattern determined by the time signature. Figure 4 shows the conducting patterns implemented in our system: 2/4, 3/4, 4/4, 6/8. It is important to keep the correct timing for each part of each pattern, while keeping the frame rate constant. This can be done by calculating the number of frames available for each beat: *(frames per second) * 60.0 / (beats per minute)*. Curved patterns are obtained using quadratic functions. Use of trigonometric functions ensures that the animation is smooth and speeds up slightly at the end of each beat, while accurately keeping the tempo.

Tag	Information
number	Measure number
beats	Number of beats per measure
beat-type	Value of one beat
sound tempo	Tempo
sound dynamics	Dynamics, written as an explicit value
rest measure	If "yes", then this is a rest measure
implicit	If "yes", then this is not a full measure and should not be counted

Table 1. MusicXML tags relevant to the system.

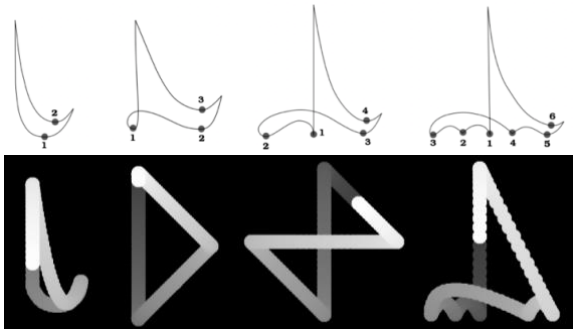

Figure 4. Conducting patterns implemented in the system. From left to right: 2/4, 3/4, 4/4, 6/8. Top: textbook illustration; bottom: screenshot from our animation

The Orchestra class handles the colored blocks on the left side of the animation. These show which instruments are playing, as well as dynamics and entrance cues. A block will gradually fade out when the corresponding instrument is resting, and will gradually fade in providing an entrance cue. Arrows labeled with dynamics terms show dynamics cues for each block. Since a conductor would signal instruments in advance, the information displayed is based on the upcoming measure. This class also displays the time signature, tempo, and current measure number.

TESTING AND RESULTS

The system was tested on a variety of sample MusicXML files obtained from the MusicXML [8] and Project

Gutenberg [12] sites to ensure that every aspect works for every variation.

Some inconsistencies were found between different files, such as a lack of title and/or contributors in some pieces, various expressions for dynamics, two different ways to describe a rest measure, the presence or absence of a pickup bar, and changes in tempo throughout the piece. These were all accounted for in our system's implementation.

The score-editing software MuseScore [13] was used to verify the validity of the sample files, as well as to export MIDI files to use for the animation soundtrack. MuseScore also displays the musical score from a MusicXML file as it would look on paper, which was used to help test and debug the program.

The system runs as expected. The animation and music are synchronous, the pause/unpause function works as expected, and the data retrieved by the parser is correct. The noticeable changes in dynamics in the audio correspond to the correct arrows display, and the instrument blocks fade out and in when appropriate. The movement of the right hand animation is similar to the conductor's motions for the appropriate time signature. The animation properly halts once the current measure number reaches the total measure number, and the music stops then, too.

CONCLUSIONS AND FUTURE WORK

In this paper we presented an application that generates a conducting animation in real time from digital sheet music stored in MusicXML format. The algorithm we developed could form the basis of an educational virtual conducting application that educates budding musicians, or could be used as a means to augment an orchestral performance with synchronized visual displays. Most previous approaches to virtual (robotic) conductors or videos accompanying musical performances require redesign or reprogramming for each new musical piece. Our approach is much more versatile: once implemented, it can be used for any piece encoded in the MusicXML format.

Our system parses the MusicXML file and extracts information including time signature, tempo, dynamics, and rest measures, and uses it to generate an animation that simulates the gestures of a musical conductor. One component simulates the right hand, following the pattern appropriate to the time signature in the tempo prescribed by the musical score, while additional components show dynamics and entrance cues for each instrument with a part in the score.

To allow for easy testing of the synchronization, our application can play the MIDI version of the music while displaying the animation. We tested on a variety of MusicXML files containing musical scores with various time signatures and tempos, and found that all the conducting information was conveyed correctly, at the correct time.

Future work includes adding listening capabilities that would allow the system to adjust to musicians playing in real time, who are bound to occasionally stray from the original score. More conducting information, such as articulation, could also be added.

REFERENCES

[1] Haus der Music. 4th Floor – The Virtual Conductor. https://www.hausdermusik.com/en/museum/4-etage-der-virtuelle-dirigent/. Last accessed 01/06/2022.

[2] D.A. Durbin, "Honda Robot Conducts Detroit Symphony," *USA Today, 5/14/2008.* http://usatoday30.usatoday.com/tech/news/robotics/2008-05-14-asimo-robot-conductor_N.htm. Last accessed 01/06/2022.

[3] H. Asprou, "Giant humanoid robot conducts human orchestra in mildly disturbing footage," *ClassicFM News, 02/25/2020.* https://www.classicfm.com/music-news/videos/giant-humanoid-robot-conducts-human-orchestra/. Last accessed 01/06/2022.

[4] Z. Giraud Voss, G. B. Voss, K. Yair, and K. Lega. "Orchestra Facts: 2006-2014," *League of American Orchestras.* November 2016. https://www.arts.gov/sites/default/files/Research-Art-Works-League.pdf. Last accessed 01/06/2022.

[5] B. Wise, "Orchestras Use New Video Technology, Courting a Younger Crowd," *WQXR Blog,* 08/12/2013. https://www.wqxr.org/story/311596-orchestras-video-technology-younger-crowd/. Last accessed 01/06/2022.

[6] The Philadelphia Orchestra. LiveNote. https://old.philorch.org/concert/livenote-2019-20#/. Last accessed 01/06/2022.

[7] Andrea Salgian, Lawrence Agina, and Teresa M. Nakra. "Teaching Robots to Conduct: Automatic Extraction of Conducting Information from Sheet Music", *Proceedings of the 2014 International Computer Music Conference*, Athens, 2014.

[8] MakeMusic, Inc, MusicXML. https://www.musicxml.com/. Last accessed 01/07/2022.

[9] MusicXML Version 4.0. Final Community Group Report. June 1, 2021. https://www.w3.org/2021/06/musicxml40/. Last accessed 01/07/2022.

[10] Java Oracle. "Class SAXParser". https://docs.oracle.com/javase/7/docs/api/javax/xml/parsers/SAXParser.html. Last accessed 01/07/2022.

[11] J. Nowak and H. Nowak, *Conducting the Music, Not the Musicians.* Carl Fischer, 2002.

[12] Project Gutenberg: Sheet Music. https://www.gutenberg.org/browse/categories/4. Last accessed 01/07/2022.

[13] MuseScore. https://musescore.org/en. Last accessed 01/07/2022.