WORCESTER POLYTECHNIC INSTITUTE

The Virtual Choir

An Interactive Qualifying Project Report
May 27, 2010

Sarah Jaffer
Christopher Petrie

Table of Contents

I. Abstract

The purpose of this project was to continue the ongoing research into the development of a virtual choir with adaptive real-time tempo adjustment. A system was developed which provided the means for a user to adjust the tempo of a multi-track choral recording of the Thomas Tallis piece *Spem in Alium* in real-time, during playback. An intermediate version was presented at the American Choral Directors Association Eastern Division conference in Philadelphia, PA, and feedback was collected.

II. Introduction

The Virtual Choir has been an ongoing project at Worcester Polytechnic Institute since 2003, when, fuelled by the success of the Virtual Orchestra, James Haupt endeavored to create a similar product for choirs. Historically, the system has been designed to function with one specific choral work, with the assumption that it could be generalized in future revisions. The piece that was used previously, and again in this project was the Thomas Tallis work *Spem In Alium* - a 40-part a cappella motet written in 1570. The music alone provides a distinct challenge; the large number of parts makes the piece difficult to faithfully reproduce in any setting, be it with live singers or with digital technologies.

However, despite the complex choice of music, the purpose was not simply to reproduce an accurate digital representation, but instead to make the digital representation conductible, as with the Virtual Orchestra. To be “conductible”, the speed (e.g. tempo) of the digital music must be adjustable during playback, in response to some form of input. This input might be a person tapping a button, or, in the optimal case, a director with a baton. This flexibility would allow for the digital recording to follow the director of a choir, and in essence ‘sing’ along with live musicians. This ability would be valuable for use in rehearsal settings, providing a learning tool that would be significantly more flexible than simple full-choir recordings with a static tempo. This system would provide the means to customize the playback to suit the needs of the learners.

Tempo manipulation is not a particularly new idea. With traditional analog recording/playback systems, it is possible to speed up or slow down the rate at which the recording is played. However, changing the tempo with traditional methods also unavoidably affects the pitch, which is undesirable in most circumstances, especially with choral recordings.

The direct relationship between the speed of playback and a changing pitch using traditional methods has prevented tempo manipulation from being more widely used, up until recently.

Since then, audio manipulation technologies have come a long way, to the point where current digital methods allow the speed and pitch of playback to be changed independently of one another. While this makes tempo manipulation much more practical, these processing methods also create often-noticeable artifacts in the audio, which are undesirable. While the simple tonal structure of choral music makes these artifacts less prominent, considerations must still be made to minimize their effects.

As compared to orchestral instruments, vocal sounds do not simply include the attack, sustain, and end to a note, but also distinct vowel and consonant sounds. The added complexity makes stretching vocals more difficult to do while maintaining a natural sound. If one were to use an audio program using the standard audio-stretching method to stretch out a sung word, every part of that word would be stretched, potentially producing awkward sounding results. While standard digital sampling methods can overcome this issue for most instruments, separating a note into the attack, sustain, and decay elements, this cannot be done in a straightforward way with voice.

The question becomes: how does one stretch out a vocal recording while both maintaining the audio quality and maintaining the natural sound to the voice? This is one of the two focal points addressed both by previous Virtual Choir projects and this one. The other point was: how does one enable this tempo change to happen in real time? This paper aims to reveal how the project team dealt with these two questions.

III. Background

A. Past Projects

In 2003, James Haupt completed the first iteration of the Virtual Choir. It was a massive undertaking involving many students at each step of the process. His response to the challenge presented by the Virtual Choir project was to take a similar approach as was taken by its predecessor, the Virtual Orchestra project. His methods involved recording each of the parts and then cutting the words into parts such that they could be controlled by Musical Instrument Digital Interface (e.g. MIDI) events. This way, the consonants could stay in time while the vowel sounds were looped to fill the time in between.

This proved to be an extremely time-consuming task. Each word needed to be broken into its base parts. The following is an excerpt from James Haupt's paper, regarding this stage of the project:

> "Whereas for each note on an instrument there's a definitive beginning, middle, and end, it is much different with a human voice. Also unlike an instrument, it's impossible to use a prerecorded set of notes for multiple applications since words need to be taken into consideration. Hence the 9726 individual files. To begin we need to analyze what happens when someone goes to sing a tune. For example's sake, I will use the tune *Twinkle, Twinkle, Little Star.* So let's start with the first word: Twinkle. Obviously we can break this down into any numerous different ways from Twin-kle, to Twi-nkle, however depending on how the person sings the word will determine how it should broken up. This example presents two possibilities: either the person singing sings a long 'n' so the word becomes Twinnnnkle, or sings a short n in which case the word becomes Twi-ngkle.
>
> Now imagine that we want to be able to preserve the way the word sounds, regardless of how fast or slowly the tempo we take the piece at is. It becomes pretty apparent where the split should be made between the first and the second notes. In the first example the first note should end right before the k, and the second note should include the kle. In the second example however, we

> want to end the first note before the ng, and then include the n sound with the kle so the second note becomes nkle. If we look further into why it should be this way, all we need to consider is if the first note is cut short what is the best way to be able to preserve all of the consonants. This is my process of thinking for splicing *Spem in Alium*." (Haupt 7)

Each of those syllables would then become its own sound file to be put into a digital MIDI-controlled Sampler. Unfortunately, this step was not successful during that iteration of the project. Two more attempts were made to take these spliced sound files and successfully develop a full virtual choir, but neither of these attempts produced more than a short segment of the first choir.

IV. Methodology

A. Initial Assessment

On the initial assessment of the requirements of this project, it was determined that there were two main considerations. The first, more immediate, but simpler aspect was the recording process. The second and more complex part was developing a method to alter the tempo in real time, as to follow a conductor or a tapped beat.

For the recording process, the program *Ableton Live!* was suggested. Upon closer examination, it was discovered that this program did not quite meet the needs of the project. The workflow was found to be lacking in comparison to its competitors; being less unified and needing external programs to perform simple audio manipulation tasks. In addition, the general procedure of recording was unnecessarily complex. From there, several other programs were considered. Notably, the similar products *Sony Acid* and *Sony Vegas* were found to be preferable alternatives, having simpler controls and a more efficient workflow. These programs were found to be able to perform the entire process of recording up to creating final, processed samples. Other programs were considered for this task, but in the end, Sony Vegas was determined most suitable for the recording process.

The real-time tempo correction portion of this project was significantly more complex. When James Haupt worked on this project, he responded to this challenge by splitting up the samples into individual consonants and vowels in order to load them into an external MIDI sampler. With the words being split up this way, it was not overly difficult to trigger the correct parts of the word at the correct times. However, this method required creating and managing thousands of small files and the use of complex MIDI arrangements to deal with them. Also, the quality of sound was reduced by the discontinuity between the samples within individual phrases. To implement this method, any MIDI arrangement and playback software that supports an external MIDI interface could be used, including *Ableton Live!* or others. Because *Ableton Live!* was designed for live performances, this would likely be the best choice, as the tempo could possibly be scripted to be altered in real time to follow a conductor. Thus, if the project team had chosen to utilize this method, Ableton's software package would probably have been the program of choice.

The most obvious alternative to the method described above would be to compress and stretch the audio samples to fit the tempo, in such a way where the pitch was unaffected. This often results in undesirable effects such as stretched consonants and digital-sounding artifacts. The Sony products and *Ableton Live!* both offer a simple method of stretching and compressing audio which significantly reduces the quality of the sound even with a small change in tempo. In both of these programs, everything within the tracks is lengthened or stretched, including aspects that should not necessarily be stretched if the piece was being performed live. In addition, the stretching method does not take into account if the audio is from a single instrument or mixed instrument source, simply looping or cutting out samples to fit the desired tempo. Thus, more specific programs were sought.

Celemony's *Melodyne Studio* is an industry leading program which deals specifically with this problem. This program separates the voice modulation from the vowel sounds of phrases so it can more cleanly stretch and compress the samples without affecting the quality of the sound. After some experimentation, it was found that it could go above and beyond the requirements in that effect, even supporting infinite holds on notes, though not during normal playback. However, there was a problem with this method, as the program could not natively support this operation in real time through the usual tap-tempo method.

After some research and experimentation with *Melodyne*, it was found that the program could be manipulated by feeding it a MIDI time-code. Using this feature, tempos could be generated externally and fed into the program, either by using preprogrammed MIDI files or with a program capable of time-code generation from real-time human control. A program was sought which could supply such time-codes in real time, given a tapped tempo. MOTU's *Digital Performer*, a flexible and widely-used synchronization program, was suitable for the task. By synching this program with *Melodyne* via a MIDI time-code, it was possible to alter the tempo of the playback without interrupting playback.

B. Recording Process

1. Recording Introduction

The goal of the recording process was to have a complete set of vocal tracks covering the first 40 measures of the piece. In addition, each vocal track would contain exactly one voice part, sung by one person. From these separated tracks, the parts could easily be moved into pitch-correction software to fine tune the parts, removing any vocal anomalies such as pitch or tempo mistakes made by the singer. From this point, the tracks would be ready to be used in any tempo-mapping (time-stretching) scheme that was deemed feasible.

2. Recording Preparation

Before recording could begin, preparation had to be made for the recording sessions. To assist and aid the singers in the task of performing as accurately as possible, minimizing errors, reference tracks were created. To do this, the first step was acquiring a digital copy of the score.

One digital copy was immediately available [SEE APPENDIX], and this provided the music separated into eight tracks, one for each chorus. This file was provided as a MIDI file with 8 tracks, each of which containing the 5 vocal parts together. This would have made separating the individual parts as in the original score time consuming. The decision was made to separate the file by choruses into individual rendered waveform files at a fixed tempo of 120bpm. While tracks for each individual part of each chorus would have been ideal, compared to a mix of all parts in the chorus, this was not possible given the source file. While the original score file (created using GVOX's *Encore* editor) was found eventually [SEE APPENDIX], this was not utilized during the recording process.

Software packages considered for recording included *Digidesign Protools* (Mac OSX), *Sony Vegas 8* (Windows), *Ableton Live!* (Mac OSX, Windows), and *Audacity* (Mac OSX, Windows, Linux), all being non-linear editors. The requirements for the recording package were the ability to easily cut and manipulate recordings using non-destructive editing, the ability to record against a tempo track or another reference, compatibility with the hardware interfaces available, and compatibility with the computers available for use in the project. All software packages considered met these requirements to a varying degree, except *Audacity*, which did not have explicit support for the hardware interfaces.

After exploring each of the choices, *Sony Vegas* was decided as the program of choice. This program provided an interface that appeared easier to use than the other programs, allowing audio clips to be cut, moved, and faded much more simply. The *Ableton Live!* interface was very clumsy and highly mouse-based, while the *Digidesign Protools* interface appeared to have a steep learning curve, as the program contained so many features. Another factor in deciding to use this program was prior familiarity and because it was compatible with the Windows machines the majority of the group used on a regular basis. If *Protools* was chosen, the group would also be limited to using Digidesign hardware interfaces. Though a minor factor, this would require the entirety of the project to be recorded in the WPI Recording studio, which would require additional scheduling.

Next, the appropriate hardware equipment was determined and for the recording process. The requirements for the hardware were devices that would allow recordings of singers with a moderate dynamic range, while producing a recorded output which had little hum or noise associated with it. Because upwards of 40 tracks would need to be layered, the noise requirement was important, as noise would build as the tracks were combined. Based on these basic requirements, the equipment chosen for this project was consumer-level audio recording equipment. Specifically, the hardware used for recording included one MXL 990 (Condenser Microphone) and one *Presonus Audiobox USB* (Audio Interface). This interface provided low-noise recording with 24-bit 48 kHz precision. Again, though this was slightly lower grade equipment than that was available (if *Protools* and Digidesign hardware were to be used), it was sufficient for the purposes of this project.

Additional hardware required included a microphone stand, shock-mount, pop-filter, and XLR cable, as well as headphones (for the singer) and studio monitor speakers (for the recording engineer). A Microsoft Windows machine fulfilling the system requirements for both *Sony Vegas 8* and the recording hardware were also essential.

3. Recording

The 8 choirs of 5 parts were recorded one part at a time, grouped by choir. In the song, each choir has distinct portions where the singers are performing. As such, these active portions were recorded from top to bottom in the score. In other words, the next section to be recorded following the first entrance of Choir 1 would not be the next entrance of Choir 1, but instead the first entrance of Choir 2. Following the recording of each choir, the tracks were rendered as waveform files and then processed to correct for pitch and tempo aberrations.

To initiate the recording process, the reference tracks were imported into the timeline of Sony Vegas and aligned with each other to match the score. Choruses could then be soloed or played together to allow the singer and recording engineer to easily align recorded parts to match up exactly with the score. A click-track was added at the same tempo as the reference tracks - 120 BPM - to further assist the singer.

The singing process was performed in a small room with sufficient dampening to sufficiently reduce echo and other reverberation effects. The microphone was set up with the pop-filter between 5 and 8 inches away from the shock-mounted microphone. The singer sat in front of the microphone, with the physical score clearly visible. The singer was instructed to sing with his mouth within a few inches of the pop-filter, towards the front of the microphone. The singer was given headphones which provided a mix of their voice (from the microphone) and the reference and click tracks. A piano was provided for reference when needed.

The singer was played the reference tracks of a small portion of the score several times before the recording of that portion began. This fragment was usually a distinct phrase such as "Spem in Alium, nun quam ha bui. Spem in Alium." This phrase was decided based on where the singer would have well defined starting and stopping points in the music. When needed, the singer could play the fragment of the score on the piano prior to recording. By recording in small segments, the singer would not be required to sight-read long portions of the song or practice ahead of time. This was presumed to provide recordings which were more accurate and true to the score.

When recording each subsequent fragment of the part, the recording would start several measures prior to the singer's entrance. If the entrance was soon after the prior fragment, the singer was instructed to sing both the exit of the prior fragment in addition to the current. By doing this, if the entrance of the previous portion did not match the new entrance in some way, the newer version could be inserted in its place. The fragments were faded together at the breakpoints.

After the first part of each chorus was recorded, the reference track was often muted, with the singer using the previous recordings and the tempo track as reference. This allowed the singer to focus on fewer things while recording. In addition, this seemed to help the singer align the new recordings more precisely with the previous, with regards to entrances and overall tempo. In addition, when recording each subsequent chorus entrance, the main reference for the new reference was the prior recorded tracks.

Once the recording was completed for a chorus, the tracks were prepared for the main post-processing phases. First, the fades were verified to be in the proper places. Next, the tracks in each chorus were rendered into volume-normalized, continuous 24-bit 48k waveform files. The beginning and ending points of each track in the chorus were consistent throughout that chorus, with the start and ending of the rendered selection being snapped to a measure. Each filename included the part and the measures that were covered, so it could be easily realigned in a timeline later on. *Figure 1* and *Figure 2* in *Appendix B* show screen captures of the final recorded tracks in *Sony Vegas*.

C. Post-Recording Cleanup Process

To prepare the recordings for the tempo-mapping, the pitch and timing had to be corrected to ensure the tracks matched the score as closely as possible. This process was performed within the software package *Melodyne Studio*, by Celemony. In contrast to other software on the market such as *Anteres Autotune*, *Melodyne* was a complete standalone solution, instead of merely a plug-in used within another editor. In addition, *Melodyne* offered an easy-to-use interface, as well as time-stretching and tempo-correction features not available in other software packages.

The tracks from each chorus were imported into the *Melodyne* timeline and aligned to match the score. Next, each track was edited one-by-one to correct for any tempo and pitch errors. An example of the *Melodyne* interface to accomplish this can be viewed in *Figure 3* in *Appendix B.* First, the built-in auto-tune algorithm was performed on the tracks. Next, the pitch modulation of the voice was reduced slightly overall. The pitches of the notes were manually checked against the score to correct for mistakes made by the automatic tuning algorithm, adjusting the pitch up or down and adjusting for pitch drift when needed. Finally, each phrase was checked against the score to ensure the rhythms were correct, with individual words being stretched or shortened when required. This process was performed on each track individually, soloed in the timeline. Next, the part was checked again against other processed tracks to ensure that the pitch and rhythms were consistent. *Figure 4* in *Appendix B* shows all of the tracks aligned in *Melodyne.*

Finally, these tracks were re-rendered as 24-bit 48k waveforms, with the same start and end times as the original files. In addition, the filenames closely resembled those of the source files, with an indication that the file was in fact the tuned version.

D. Tempo Mapping

To perform real-time tempo mapping, allowing for a conductor to "conduct" the virtual choir to change speeds live, several software packages were considered. Based on the research of previous virtual choir implementations, it had been decided to simply try to stretch the files, versus using MIDI-triggered events. As such, software was considered based on its ability to change the playback speed in real time in response to some external stimuli, the audio quality of the time stretched waveform, and the potential for automation or expandability, for future projects. The only programs that were considered for this purpose were *Ableton Live!* and *Celemony's Melodyne Studio*. While many other programs have the ability to stretch audio, very few seemed to be able to do this in real-time to an arbitrary, changing tempo. While *Ableton Live!* does have support for tempo stretching via a tap-tempo system, where the user presses a button to indicate each beat, *Melodyne* only supported tempo adjustment through MIDI time-code control. Also, *Melodyne* included a system in which virtually every element of the program could be automated using MIDI events.

While both programs could stretch any arbitrary waveform, only *Melodyne* had a "melodic" mode which was designed specifically to stretch monotonic instruments, such as the voice. Instead of stretching by looping small portions of the waveform very quickly, *Melodyne's* algorithm intelligently separates the formants (or modulation) of the vocals from the pitch before applying any stretching. Through several trials where the tempo was changed to another static tempo, it was subjectively concluded that *Melodyne's* "melodic" mode produced a more natural sounding output, lacking much of the artifacts associated with *Ableton's* traditional time-stretching methods, especially when slowing down the tempo. When the all of the individual vocal tracks of a chorus were mixed into a single file before time-stretching, both programs performed similarly. Because the primary concern was sound quality, and not ease of tempo-control (since both programs could be controlled in some manner), *Melodyne* was chosen to perform the real-time time-stretching.

The pitch corrected tracks were imported into *Melodyne* and aligned in the timeline. Next, the *Melodyne* settings were adjusted such that playback was synchronized to the MIDI time-code input, with the "auto-stretch" function enabled. From this point, *Melodyne* was ready to perform the time-stretching when provided a MIDI time-code signal. Unless *Melodyne* was modified, or additional software was written, this required an external device to create this time-code.

To generate the time-code necessary to control *Melodyne*, a combination of another computer and MIDI controller keyboard were utilized. This was accomplished by using the software package *Digital Performer* by MOTU (on a Macintosh computer) and a generic MIDI keyboard. In this program, a project was made that included a MIDI track at 120bpm with a quarter note on each beat. Then, the settings were adjusted such that a key-press on the MIDI controller keyboard controlled the tempo of the project with the MIDI track. Finally, the settings were changed such that Digital Performer would output a time-code whenever the track was playing. The track would only begin playing when the keyboard was used to control the tempo, with a key press on every quarter note.

By plugging a MIDI cable between this control system and the MIDI interface selected on the *Melodyne* computer, the tempo-mapping was realized. The audio buffer length was then adjusted in *Melodyne* to achieve a tempo-mapped output which had smooth transitions between slow and fast user-defined tempos.

E. ACDA Conference

In February of 2010, a group of WPI students who had been working on various music-based projects under the advisement of John Delorey attended an American Choral Directors Association conference in Philadelphia, PA in order to present the research conducted thus far. Among the projects presented was an initial version of the Virtual Choir. The prototype presented was intended to include the first 40 measures and all choirs of *Spem in Alium,* though fewer measures were completed at the time of presentation. This prototype had limited capabilities; it could alter the tempo of the piece with minimal quality loss, but not in real time. It was also possible to change the volume of each individual part, to play only one part, or to take out one or more parts from the recording. A survey was developed to ascertain the attendees' opinions of the technology as a rehearsal tool. It was administered after first playing for them the unchanged virtual choir and then allowing then to sing along with the virtual choir with their own part played louder or softer, as per their choice. The survey-related details can be found in *Appendix A.*

V. Results

A. Overall Results

The overall results of the recording process were low-noise, clear recordings of Choirs 1-4, covering up until measure 28 in the piece. These recordings did not exhibit any apparent hiss or noticeable noise, even when the tracks were played together.

By compressing the vocal dynamics with a 4:1 ratio, the resulting files were more uniform in volume intensity. This ensured each part could be heard when played together, despite tracks being recorded with different levels. While this removed nearly every dynamic change actively performed, it was essential so each part could be heard. Also, this issue could be overlooked since the significant dynamic changes in the song occur because choirs enter and exit.

The equalizer settings were set on each track ensured that when played together, the output would not exhibit a build-up of mid-range frequencies. This produced an output that did not sound "Muddy" as multi-track vocal recordings often do, prior to processing.

The realignment of the rendered waveform files in a timeline was simplified by starting each file for each part of a chorus at the same measure. The file naming scheme that was used clearly signifies where the files fit in the score. These steps made it significantly simpler to import the recordings into Melodyne to be processed further.

1. Accuracy and Tonal Quality:

The recording process produced recordings which were generally true to the score. However the recordings suffer from problems with the overall tone in addition to the basic pitch and tempo errors.

Due to time constraints, one of the members of the project team was used to perform all five parts. Due to his vocal range, the singer was unable to sing several parts of the bass line in the proper key, requiring these sections of the recording to be tuned after the fact. As a tenor, he was unable to strongly support the lower parts, and was only able to sing the higher parts in a forced falsetto voice. In addition, this singer was not classically trained to sing in the style of the piece.

Because only one singer was used on the project, the singer was forced to sing outside of his normal range, producing more strained representations of the parts. In addition, the singer was not guided with how to sing the song, causing the tracks produced to have vocals that are much brighter than what is generally associated with this piece. Also, because of the little variation in vocal tone between the parts, it is easier to realize that the singer provided vocals for more than one track.

B. Tuning Process

1. Overall Results

After the recording process, tuning techniques were applied to the sound files to correct for pitch inaccuracies, making the recordings more accurately fit the score. Each note was snapped to the proper pitch and timing using a combination of Melodyne's automatic algorithms and hand correction. In addition, pitch drift over time, intense modulation (vibrato), and the length of slides between notes were reduced. Melodyne proved to be a simple and effective tool allowing for these aberrations to be corrected easily and quickly.

2. Tuning Artifacts

This process did, however, produce obvious tuning artifacts in the audio, diminishing the quality of sound in places. One of the most significant noticeable artifacts produced was the result of reducing the pitch modulation on notes with strong vibrato. This produced a sound that was unnatural and very noticeable, especially towards the beginning of the piece when fewer voices are present together. In addition, transposing the bass line to the proper key proved to result in an unnatural harshness to the tone which is especially noticeable when the bass tracks are played in solo.

3. Timing Correction:

In addition to tuning the tracks, many timing errors were reduced, producing a more accurate representation of the rhythms. These corrections were generally made manually by playing back the recording and manually counting beats along with the score to find problem areas. The resulting sound files were more accurately aligned than could be produced by a real choir.

4. Blending Issues:

The exactness of the pitch correction caused blending issues between the choirs as well. In a live chorus, the "chorus" effect is a result of parts being sung by different people slightly out of pitch with one another, creating a phasing effect. However, because every track was strongly pitch corrected, these phasing effects were limited. This made the combined choirs sound like fewer parts than were actually there, as parts which sang the same pitches blended together more precisely. In effect, the combined choirs did not sound like a large choir as was intended.

In addition, the choir effect was further reduced by the timing correction performed on the tracks. While the tracks were more accurate to the score, the enhanced alignment of the starts and ends of notes created a very prominent pulsing effect when played together. The rests and beats became extremely well defined, making it sound as though fewer parts were being played simultaneously. Some of these effects were mitigated by the liberal application of reverb to the combined tracks upon playback.

C. Tempo Mapping:

1. Overall Results:

The tempo-mapping system developed for this project worked well for simple tempo adjustments, allowing the user to control the tempo of the recording with relative ease. Also, the quality of the stretched audio was rather good, exhibiting few stretching artifacts, especially when the tempo was slowed from the original.

2. Varying the Tempo:

The tempo stretching performed using Melodyne did not cause a significant degradation in audio quality with "reasonable" tempo changes. More specifically, the tempo could be slowed to such extremes as 50% or as fast as 125% of the original without sounding entirely unnatural. As such, the relatively small tempo changes the user would likely be performing (+- 15%) are tolerable. It should be noted, however, that these values are merely qualitative estimates, based on trial runs by the group.

The system produced more natural sounding results when slowing the recordings versus speeding up the recordings. This was likely because consonant sounds, pitch slides, and vibrato are more natural sounding when lengthened than shortened. When sped up, the resulting recording produces sounds that cannot be physically vocalized. Live singers would not be able to duplicate the quick pitch transitions exhibited by the sped-up recordings, after a certain threshold.

3. Conducting the Tempo:

The system was able to successfully control the tempo of the recording through tempo-tapping using the MIDI controller. However, the system proved to have problems regarding the lag between tapping a new tempo and the change in the playback (hereby known as "tap-tempo response").

It has been determined that the length of the buffer has a direct relation to the tap-tempo response time, thus making the audio buffer size an important consideration to the system setup. With a short audio buffer size, the tap-tempo response time was shorter but the playback jumped forwards and backwards harshly in response to the tempo changes, especially with more extreme alterations. Conversely, with a larger buffer size, the tempo transitions were smooth, but the tradeoff was that the tap-tempo response time was significantly longer. For demonstration of the system, the buffer size was chosen to be the minimum which did not cause the audio to jump noticeably when provided +-15% tempo changes.

Based on initial testing, the user found it difficult to compensate for the tempo lag. This often caused the user to ultimately slow down the music much more than what was initially desired. Because of this, with smaller buffers, the user found easier to control the tempo as desired. On the other hand, the harsh jumps sometimes made the conducting more confusing.

4. Overall Interface:

The overall interface of the system was rather clunky, requiring at least two computers and a MIDI controller keyboard to function. In addition, if one were to separate the tracks such that each part could be sent to its own dedicated speaker, this would require one computer system running *Melodyne* for each pair of two tracks, all synchronized to the same MIDI time-code generated by another computer system. In addition, the current setup required each software program to be manually reset before playback could start again.

5. Missing Features:

While the tempo-control schema did demonstrate the ability to adjust the tempo of playback in real-time, several essential features were missing if this were to become a usable product. First of all, with the current setup, there was no way to conduct a fermata or to easily stop and start the playback. Also, there was no support for lead-in tempo-tapping measures before playback is to begin. Lastly, there is no way to conduct dynamics or to signify the conductor's intentions to speed up or slow down the playback in the measures before the intended change.

D. ACDA Conference

The attitudes towards the Virtual Choir, from the results of the surveys, were generally very positive. The choral directors who participated almost universally agreed that the ability to change the volume levels of individual parts was extremely useful and most agreed that the technology made it much easier to sing along with the music. Most responded that they thought the sound of the Virtual Choir was at least decent. This went far beyond the project team's expectations.

However, there are certain things to take into account when scrutinizing the results from the conference. Firstly, the sample size was very small and was taken from a naturally biased section of the attendees. Most of those surveyed were conference-goers initially attracted to the kiosk by the offer of free Internet access. The results might have been different had we administered the survey to those who had little or no experience with technology. From the results themselves, we know that there was only one person that responded who did not use a computer daily. However, despite the biased nature of the survey, the results indicated that choral directors who were at least minimally technology-friendly would, in fact, use a technology like the Virtual Choir as a rehearsal tool.

VI. Conclusions and Suggestions

Overall, the project was a success, utilizing new approaches and methods to produce a virtual choir prototype capable of fulfilling the basic requirements of the project. . In the end, twenty-seven measures of all relevant parts of *Spem in Alium* were recorded, and a method for altering the tempo in real-time was designed and implemented. While the project can be considered a success, there is still much work to be done before the system can distributed as a learning tool.

Though the method used to conduct the system was usable for simple demonstrations of tempo correction, it was rather simplistic and lacking important features. In the future, a more full-featured method of conducting the virtual choir will need to be developed. This will not only involve improving the tempo input method, but will also involve adding other features (such as dynamics control) that would make the system more complete and generally more natural for a conductor to operate. In addition, these improvements are closely related to needed research regarding determining optimal buffer sizes to compromise between lag and undesired playback fluctuations. Resolving these issues would likely include either a method for the conductor to make his or her tempo intensions known ahead of time, or some method of input filtering. Further research and development is definitely required to make the system more usable.

In terms of tempo-alteration, *Melodyne* was the most suitable program the team could find and did, in fact, suit the goals of the project. It maintained above-satisfactory sound quality when stretching or compressing the audio and provided a usable, though sub-optimal, control interface. With regards to the future of the software, it would likely be worthwhile to research if *Melodyne* had any published APIs that would provide a more seamless way with integrating with the application. Applying a new interface on top of a commercial application would likely be much more difficult than if the technology was licensed and a working relationship was made with the parent company. This would likely turn this project into a computer science related project in the future, which seems a reasonable transition.

Similarly, the hardware implementation produced a working system, though it was unnecessarily complex for what needed to be done. As such, the system must be reworked in future projects to produce a system which is more integrated. Currently, the tempo-mapping system requires two separate computers (one PC and one Macintosh), a MIDI keyboard, and a MIDI cable between the computers' MIDI Ports. If an integrated software application could be developed, or other methods explored, this system could be reduced to utilize a single personal computer, making it more feasible to distribute.

In the future, the recording sessions and required vocalists should be scheduled well in advance to avoid conflicts and to ensure that the proper singers are available for each part. This would go a long way towards improving audio quality, as no one would be required to sing out of range and no parts would have to be shifted into the correct key. With multiple vocalists, the strain on each would be lessened considerably, as no one singer would be required to sing all of the parts. Using multiple vocalists would bring diversity to the tone, making the mix sound more like an actual chorus. Also, Attaining or creating a more accurate reference tracks would also go a long way towards easing the process. Not having access to the needed singers was a major problem with this iteration of the project, and the use of a single singer for every part should not be repeated.

Little can be said about the preferences of the audience for this project, as the distributed surveys were not very detailed. Thus, it would be worthwhile to do more surveys to determine whether potential users would prefer a more human-sounding recording, or the accuracy and precision that the project team strived for in this iteration.

Lastly, though producing a full working virtual choir version *Spem In Alium* would be an admirable achievement, it seems as though it was somewhat of an unreasonable task for the purposes of this project. The piece put a large emphasis on the performance and recording aspects of this project, removing much of the time which should have been spent determining the best ways to stretch and conduct the audio, in essence, the core goals of the project. The same effects could have been easily studied using a simpler musical work.

Despite the need for further analysis, a working system has been designed and tested, opening a path for improvement and extension. It is the team's hope that the Virtual Choir will continue to see development over the coming years.

VII. References

1. Haupt, James R. *Spem in Alium -- Virtual Choir Project*. Diss. Worcester Polytechnic Institute, 2003.

2. Cooper, Chris; Anderson, Joshua D. *Spem in Alium – the Virtual Choir*. Diss. Worcester Polytechnic Institute, 2006.

3. Jinno, Jeremiah. *Advanced techniques in digital remixing*. Diss. Worcester Polytechnic Institute, 2007,

VIII. Appendix A: Survey

Virtual Choir Survey

(1) How many hours a day do you use a computer? __________ Hours

(2) Did you like the sound of the virtual choir?

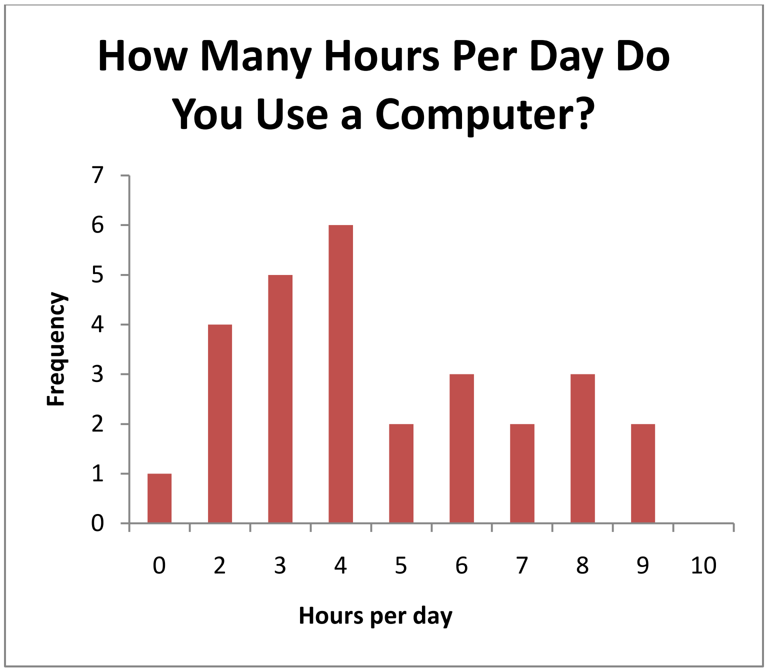
Very Much				Not at all
5	4	3	2	1

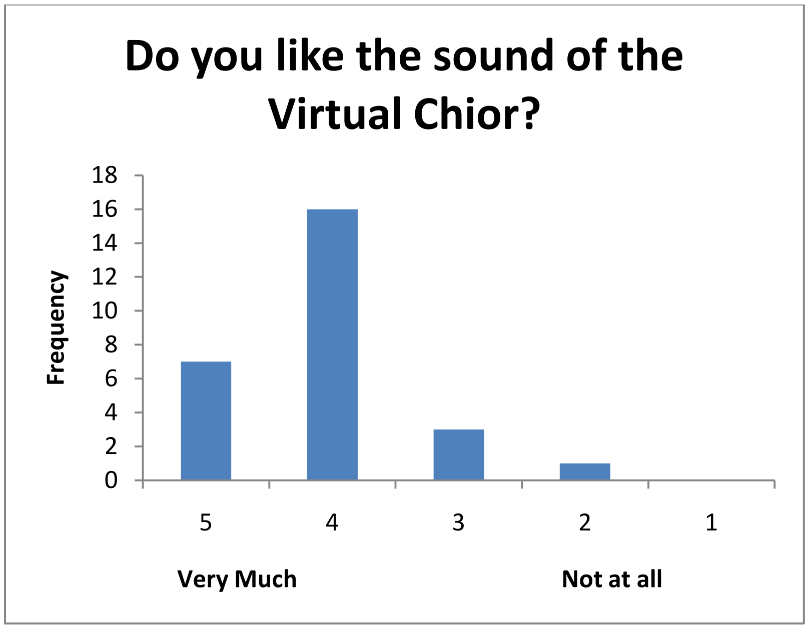
(3) How useful was the ability to manipulate volume levels?

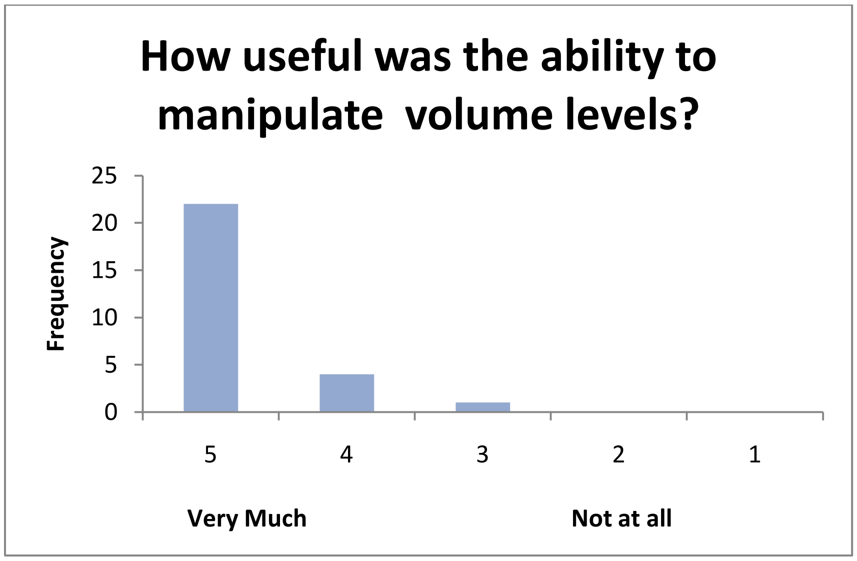
Very Much				Not at all
5	4	3	2	1

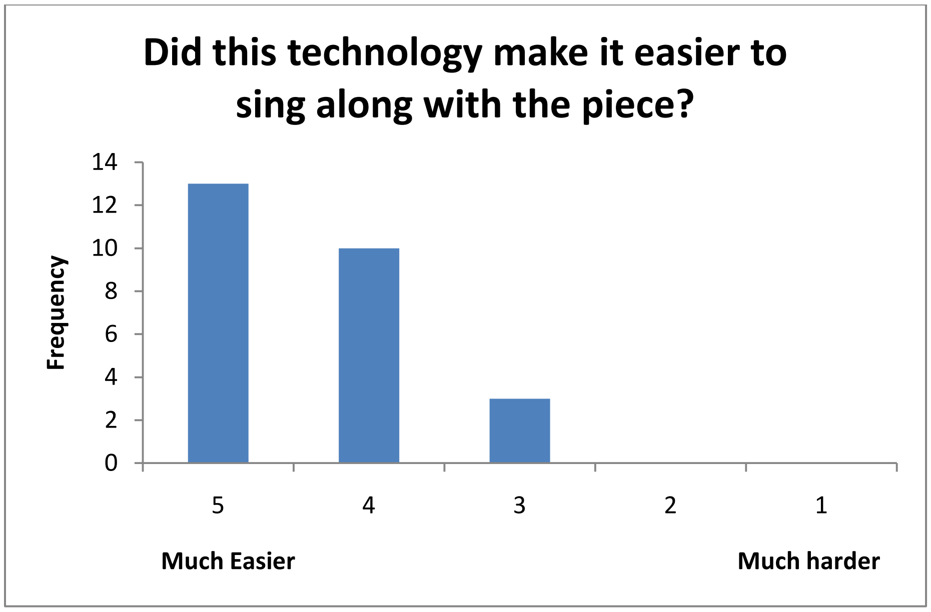
(4) Did this technology make it easier to sing along with the piece?

Much easier				Much harder
5	4	3	2	1



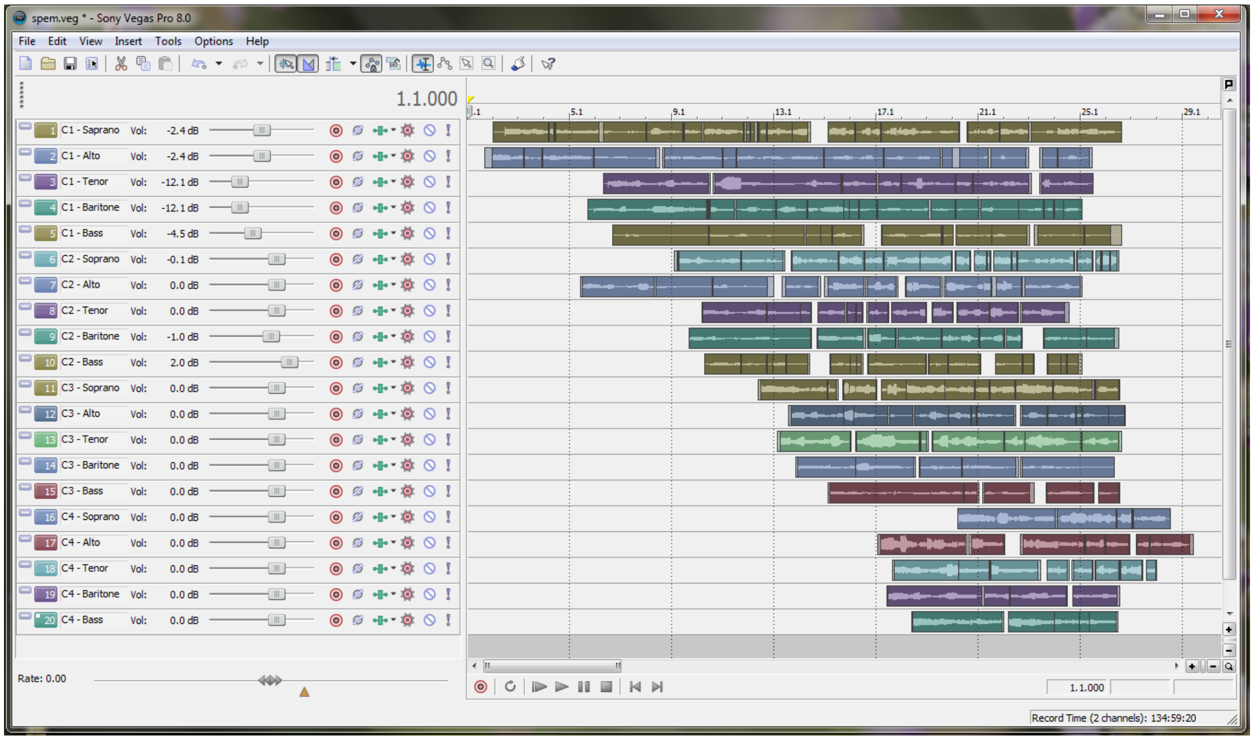

Do you like the sound of the Virtual Chior?
Frequency
18
16
14
12
10
8
6
4
2
0
5
4
3
2
1
Very Much
Not at all


How useful was the ability to manipulate volume levels?
Frequency
25
20
15
10
5
0
5
4
3
2
1
Very Much
Not at all


Did this technology make it easier to sing along with the piece?
Frequency
14
12
10
8
6
4
2
0
5
4
3
2
1
Much Easier
Much harder

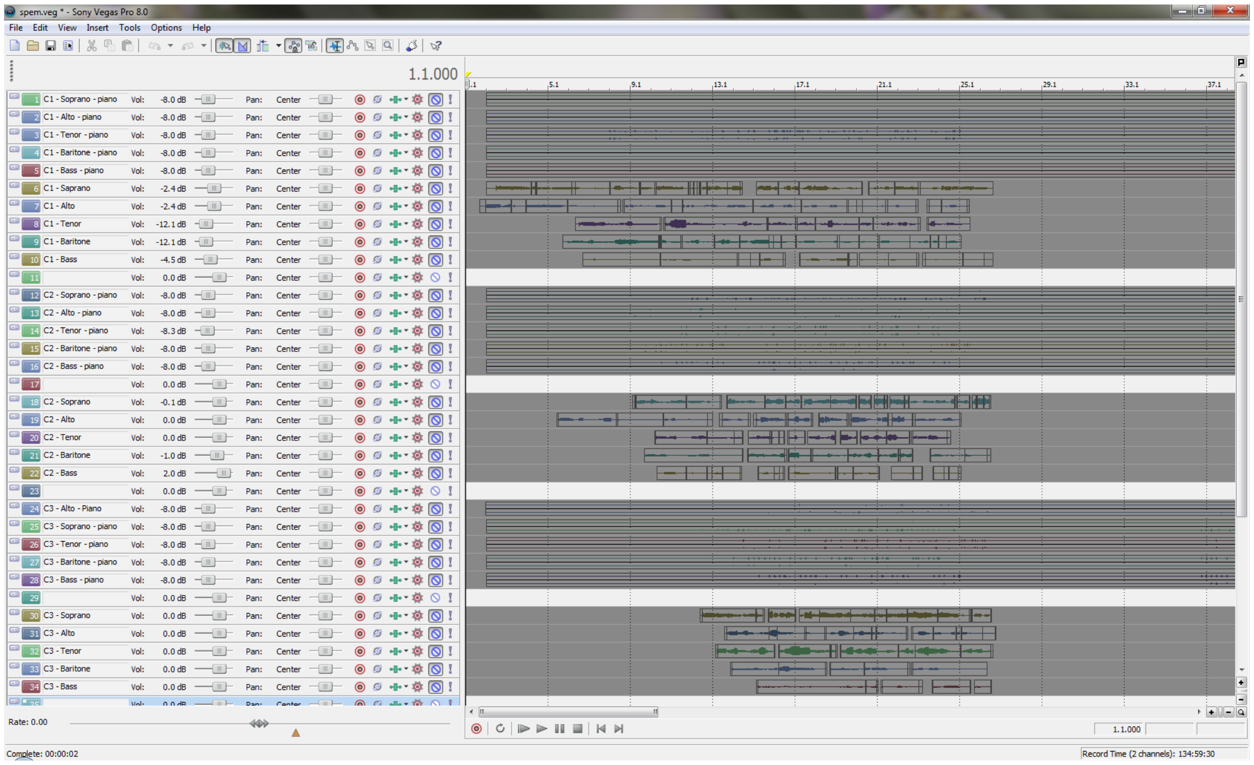
IX. Appendix B: Screen Captures

Figure 1: Sony Vegas – Choral Parts Only


spem.veg * - Sony Vegas Pro 8.0
File Edit View Insert Tools Options Help
1.1.000
.1
5.1
9.1
13.1
17.1
21.1
25.1
29.1
Rate: 0.00
1.1.000
Record Time (2 channels): 134:59:20

1	C1 - Saprano	Vol:	-2.4 dB
2	C1 - Alto	Vol:	-2.4 dB
3	C1 - Tenor	Vol:	-12.1 dB
4	C1 - Baritone	Vol:	-12.1 dB
5	C1 - Bass	Vol:	-4.5 dB
6	C2 - Soprano	Vol:	-0.1 dB
7	C2 - Alto	Vol:	0.0 dB
8	C2 - Tenor	Vol:	0.0 dB
9	C2 - Baritone	Vol:	-1.0 dB
10	C2 - Bass	Vol:	2.0 dB
11	C3 - Soprano	Vol:	0.0 dB
12	C3 - Alto	Vol:	0.0 dB
13	C3 - Tenor	Vol:	0.0 dB
14	C3 - Baritone	Vol:	0.0 dB
15	C3 - Bass	Vol:	0.0 dB
16	C4 - Soprano	Vol:	0.0 dB
17	C4 - Alto	Vol:	0.0 dB
18	C4 - Tenor	Vol:	0.0 dB
19	C4 - Baritone	Vol:	0.0 dB
20	C4 - Bass	Vol:	0.0 dB

Figure 2: Sony Vegas – Reference and Choral Tracks



spem.veg * - Sony Vegas Pro 8.0

1	C1 - Soprano - piano	Vol:	-8.0 dB	Pan:	Center
2	C1 - Alto - piano	Vol:	-8.0 dB	Pan:	Center
3	C1 - Tenor - piano	Vol:	-8.0 dB	Pan:	Center
4	C1 - Baritone - piano	Vol:	-8.0 dB	Pan:	Center
5	C1 - Bass - piano	Vol:	-8.0 dB	Pan:	Center
6	C1 - Saprano	Vol:	-2.4 dB	Pan:	Center
7	C1 - Alto	Vol:	-2.4 dB	Pan:	Center
8	C1 - Tenor	Vol:	-12.1 dB	Pan:	Center
9	C1 - Baritone	Vol:	-12.1 dB	Pan:	Center
10	C1 - Bass	Vol:	-4.5 dB	Pan:	Center
11		Vol:	0.0 dB	Pan:	Center
12	C2 - Soprano - piano	Vol:	-8.0 dB	Pan:	Center
13	C2 - Alto - piano	Vol:	-8.0 dB	Pan:	Center
14	C2 - Tenor - piano	Vol:	-8.3 dB	Pan:	Center
15	C2 - Baritone - piano	Vol:	-8.0 dB	Pan:	Center
16	C2 - Bass - piano	Vol:	-8.0 dB	Pan:	Center
17		Vol:	0.0 dB	Pan:	Center
18	C2 - Soprano	Vol:	-0.1 dB	Pan:	Center
19	C2 - Alto	Vol:	0.0 dB	Pan:	Center
20	C2 - Tenor	Vol:	0.0 dB	Pan:	Center
21	C2 - Baritone	Vol:	-1.0 dB	Pan:	Center
22	C2 - Bass	Vol:	2.0 dB	Pan:	Center
23		Vol:	0.0 dB	Pan:	Center
24	C3 - Alto - Piano	Vol:	-8.0 dB	Pan:	Center
25	C3 - Soprano - piano	Vol:	-8.0 dB	Pan:	Center
26	C3 - Tenor - piano	Vol:	-8.0 dB	Pan:	Center
27	C3 - Baritone - piano	Vol:	-8.0 dB	Pan:	Center
28	C3 - Bass - piano	Vol:	-8.0 dB	Pan:	Center
29		Vol:	0.0 dB	Pan:	Center
30	C3 - Soprano	Vol:	0.0 dB	Pan:	Center
31	C3 - Alto	Vol:	0.0 dB	Pan:	Center
32	C3 - Tenor	Vol:	0.0 dB	Pan:	Center
33	C3 - Baritone	Vol:	0.0 dB	Pan:	Center
34	C3 - Bass	Vol:	0.0 dB	Pan:	Center
35		Vol:	0.0 dB	Pan:	Center

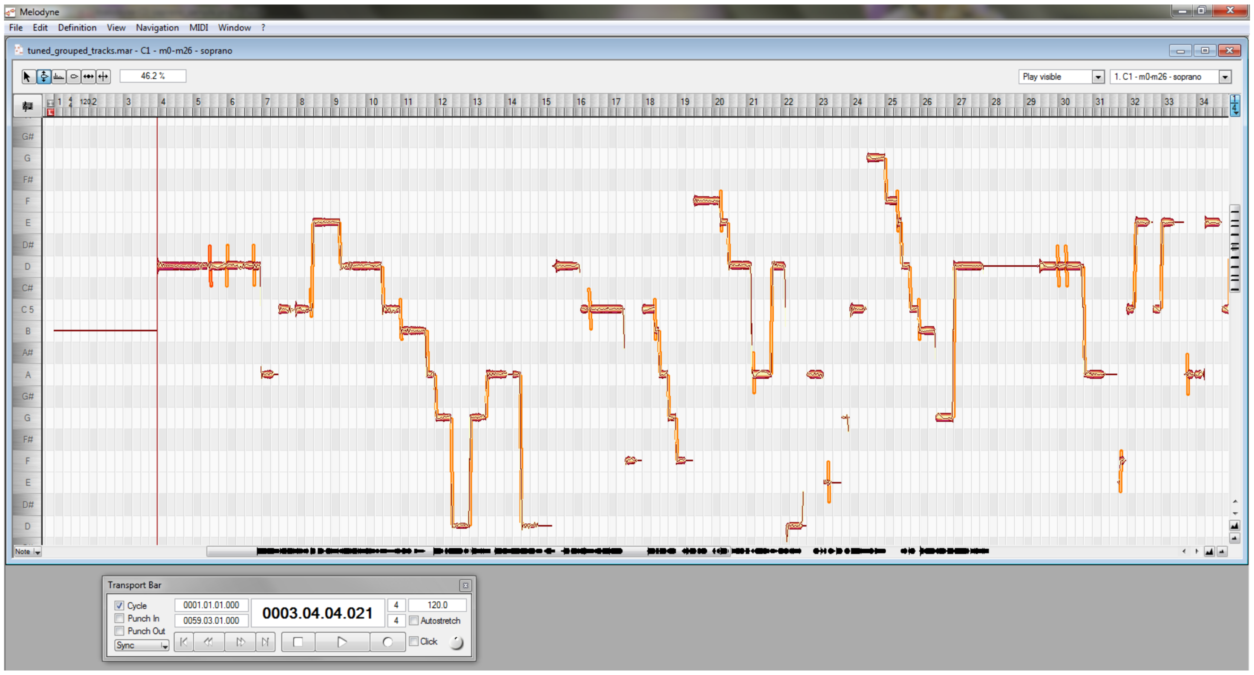
Rate: 0.00

Complete: 00:00:02

1.1.000

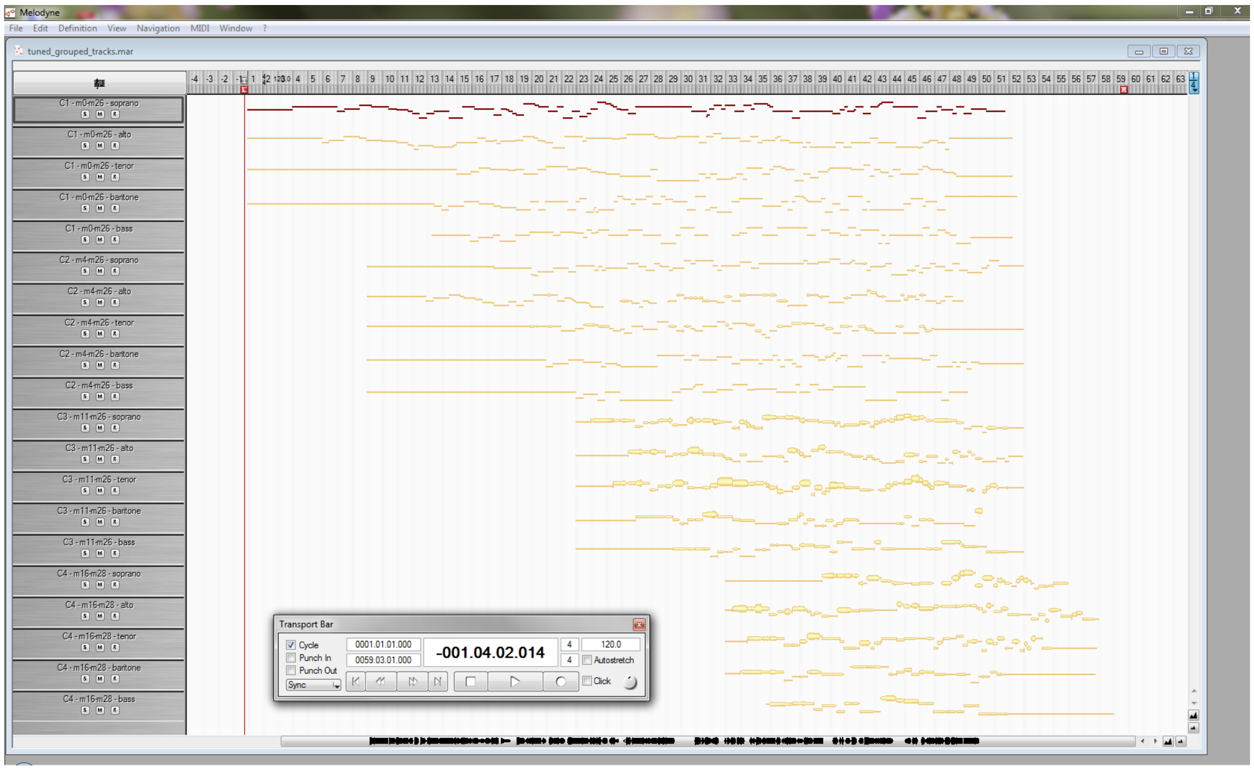
Record Time (2 channels): 134:59:30

Figure 3: Melodyne – Soprano line


Melodyne
File Edit Definition View Navigation MIDI Window ?
tuned_grouped_tracks.mar - C1 - m0-m26 - soprano
46.2 %
Play visible
1. C1 - m0-m26 - soprano
G#
G
F#
F
E
D#
D
C#
C5
B
A#
A
G#
G
F#
F
E
D#
D
Note
Transport Bar

Cycle [✓]
Punch In
Punch Out
Sync
0001.01.01.000
0059.03.01.000
0003.04.04.021
4
4
120.0
Autostretch
Click

Figure 4: Melodyne – All tracks



C1 - m0-m26 - soprano
C1 - m0-m26 - alto
C1 - m0-m26 - tenor
C1 - m0-m26 - baritone
C1 - m0-m26 - bass
C2 - m4-m26 - soprano
C2 - m4-m26 - alto
C2 - m4-m26 - tenor
C2 - m4-m26 - baritone
C2 - m4-m26 - bass
C3 - m11-m26 - soprano
C3 - m11-m26 - alto
C3 - m11-m26 - tenor
C3 - m11-m26 - baritone
C3 - m11-m26 - bass
C4 - m16-m28 - soprano
C4 - m16-m28 - alto
C4 - m16-m28 - tenor
C4 - m16-m28 - baritone
C4 - m16-m28 - bass

[✓ Cycle] ☐ Punch In ☐ Punch Out
0001.01.01.000
0059.03.01.000
-001.04.02.014
4 / 4
120.0
☐ Autostretch
☐ Click