Musically Understanding the Emotional Content of Speech

Kevin D. Deisz

College of William and Mary

Follow this and additional works at: https://scholarworks.wm.edu/honorstheses

Recommended Citation
https://scholarworks.wm.edu/honorstheses/615

This Honors Thesis is brought to you for free and open access by the Theses, Dissertations, & Master Projects at W&M ScholarWorks. It has been accepted for inclusion in Undergraduate Honors Theses by an authorized administrator of W&M ScholarWorks. For more information, please contact scholarworks@wm.edu.
Musically Understanding the Emotional Content of Speech

A thesis submitted in partial fulfillment of the requirement for the degree of Bachelors of Science in Computer Science and Music from The College of William and Mary

by

Kevin Daniel Deisz

Accepted for High Honors
(Honors, High Honors, Highest Honors)

Dr. Greg Bowers, Advisor

Timothy Mauthe

Dr. Paul Bhasin

Dr. Evgenia Smirni

Williamsburg, VA
April 12, 2013
Introduction

The purpose of this project is to explore the relationship between affect in speech and music in the context of a live, interactive system. I posit that through the creation of a relationship between the emotional content of speech and music, intrinsic information can be gathered with respect to both the emotion in the speech and the created music, i.e., more can be understood about the speech itself and there is a depth added to the musical composition. Broadly, this is a study in cognitive science; it is an application that searches for a method that can represent emotions in speech both parametrically (through measuring and quantifying input) and musically (through synthesis and composition).

Within cognitive science, this project falls under three distinct fields: psychology and neuroscience, computer science, and music. Within the fields of psychology and neuroscience, this project requires the understanding of the implications of inflection and affect in speech. Within the field of computer science, this project creates an input for speech into the program, processes and translates the speech, and outputs the music. Finally, the field of music defines terms by which the parameters given by the program can be translated into an approachable and understandable auditory composition.

For this project, I have created an installation in Max/MSP - a graphical programming language used for multimedia development. The installation takes input from a microphone of one person speaking. The speech is analyzed and quantified into many different variables. Those variables are then used to create an auditory landscape reflective of the emotional content found in the speech, which is then played back in real-time. The program has the additional functionality of being able to process recorded
speech and apply the same methodology. The result is an interactive system that provides a unique experience for each user every time it is used.

In this paper I will discuss the basis for this installation in the various fields of study under which it falls, both in terms of similar projects and the historical attributes of the fields that directly relate to this project. I will describe the variables that I extracted and measured from the inputted speech, as well as the artistic decisions of how those variables could be translated into sound. Finally, I will discuss the overall state of the installation, as well as the possibilities that it holds for further research on this project or any other in a related field.

**Basis in Psychology and Neuroscience**

Composed music has no intrinsic emotional value. Producers of music translate their compositional ideas into seemingly meaningless notation, which can be interpreted and performed by anyone who understands the method by which their ideas were encoded. After centuries of repeating this pattern, cultures develop an affinity for the sounds of their society. Neurological studies show that subjects internalize their culture’s musical tendencies, and that Western cultures exhibit brain activity similar to that of experiencing anger when they are exposed to severe dissonance in music (Blood, Zatorre, Bermudez, & Evans, 1999). As a culture, we internalize these rules at an early age: children ages 6-8 are already able to assess the affect of a composition based on its mode (Bella, Peretz, Rousseau, & Gosselin, 2001). Composers have played off of this response for ages, using the tension in music to make the listener uncomfortable in order to create

> Every tone which is added to a beginning tone makes the meaning of that tone doubtful ... In this manner there is produced a state of unrest, of imbalance which grows throughout most of the piece ... The method by which balance is restored seems to me the real *idea* of the composition (p. 49).

As Schoenberg states, the resolution of the tension of the piece is incredibly important to the musical structure of the piece, but it is also important to the listener. Similar neurological studies to that of cultural musical tendencies show that both the anticipation and the experience of this resolution (musical expectancy) result in endogenous dopamine release in the brain (Salimpoor, Benovoy, Larcher, Dagher, & Zatorre, 2011). Listeners subconsciously treat the resolution of dissonance as an abstract physiological reward. In other words, large builds and releases in music enable the listener to feel intense pleasure as a direct result of the music.

This manipulation of the listener’s emotions, while having implications for musicians and composers, is indelibly linked to the psychology of emotion. Although communication through music is a small subset of the much broader fields of the psychology of communication and the communication of emotion, it has extensive implications about both. When listening to music, a person’s response moves beyond cognitive appraisal to include many other factors such as episodic memory, brain stem reflexes, and, as mentioned before, musical expectancy (Juslin & Västfjäll, 2008). As a result of music having the ability to manipulate emotions through these extrinsic “mechanisms,” the study of emotion in music can serve as a basis for the study of emotion as a whole, and a model for “studying the effects of manipulating perceptual
features on the emotional content of a sensory stimulus” (Hailstone et al., 2009, p. 2142).

This link between music and emotional communication is largely the basis for this project: I use the perception of emotion in music as a model for displaying the variability of emotion in speech.

**Basis in Computer Science**

This application falls under the subfield of computer science known as “affective computing.” This field of study is defined as computing that relates to, arises from, or deliberately influences emotion or other affective phenomena. More simply, it is the field of computer science that has to do with computation and emotion. The field of affective computing arose in 1997, with the publication of Rosalind Picard’s book, *Affective Computing*. In it, she postulates that computers would be a more effective means of facilitating productivity if they were able to recognize and act on the emotional state of the user. An example of such a benefit would be the ability of the computer to recognize behavioral patterns. One such pattern would be whenever a user consistently manifests some clear emotional state; he or she always requests that the computer perform some task. The computer would then recognize that emotional state and proceed to preload the information required for that task. This is just one of many examples that Picard uses to describe the potential benefits of computers with emotional intelligence - and what she posits will be the future of affective computing.

The Massachusetts Institute of Technology Media Lab has a subgroup entitled the Affective Computing Lab, which does research in this field, headed by Picard herself. This lab has published research on many different applications that relate to music
including: a guitar that changes pitch based on the physical gestures of the player with the instrument, a carpet that gathers emotional information from a performer based on weight-shift patterns, and a jacket worn by a conductor that interprets the information he or she is conveying in a beat pattern through sensors on the arm and chest muscles. The lab has also produced material on the psychological study of affect in speech, including an ongoing project on the recognition of affect and a project on assembling a database of speech patterns that demonstrate affective variability.

In relation to this project, affective computing describes the manner by which I attempt to quantify the speech input to the program. The ability to quantify this information is based on the fact that fluctuations in the autonomic nervous system indirectly alter speech (Sobin & Alpert, 1999). When a person experiences emotions that stimulate the sympathetic nervous system (e.g., stress, anxiety, anger), he or she also experiences physical manifestations of those emotions such as increased heart rate. This results in speech that embodies a higher average pitch, a wider pitch range, and an increase in speech rate (Breazeal & Aryananda, 2002). Contrastingly, when a person experiences emotions that stimulate the parasympathetic nervous system (e.g., sadness, boredom, exhaustion), the speech typically exhibits a lower average pitch, a narrower pitch range, and a decrease in speech rate (Breazeal & Aryananda, 2002). It is therefore possible to infer the current emotional state of the speaker from the various characteristics of his or her speech pattern.
**Basis in Music**

The understanding of the history of music software and the computers running that software is requisite to the understanding of the installation. Max/MSP is an object-oriented programming language, meaning it stores data and functions that operate on that data in small sections of code known as objects. The objects themselves each have a historical significance attached to them and their functionality. For example, the simplest input-output stream in Max/MSP is one analog-to-digital converter object (ADC) attached to a digital-to-analog converter object (DAC). While this seemingly simple system is readily programmed in Max/MSP now, this task would have taken many engineers many months in the earliest age of computers. In order to add context to the project, and to more deeply understand the installation, it is therefore necessary to understand the significant points of the history of computer music.

Computers have been making noises since they were first invented in the late 1940s. The sounds they made were originally used as an indication of the current state of their computation. A common engineering paradigm was to connect different parts of the processor to speakers that would output tones whenever they were fed data. In this way programmers could tell what a computer was doing without having to view the printed output that was the norm at the time. The first truly musical program was MUSIC, created in 1957 at Bell Labs by Max Mathews (Mattis, 2011). MUSIC allowed for a single voice to be generated and then played. This program spawned an entire family of programs identified by increasing roman numerals after the title. MUSIC II expanded on the original by allowing for four-part polyphony. MUSIC III and IV both came with many more improvements that significantly bolstered both the functionality and
reputation of the software. With the advent of MUSIC III, Mathews had added the ability to program subroutines called unit generators, which notably eased the creative process (Mathews, Miller, Moore, Pierce, & Risset, 1969). As a result, many composers began to take notice and compose using this software, including James Tenney, Jean-Claude Risset, and Charles Dodge.

In the years that followed, the MUSIC software spread to many universities and research centers. Jean-Claude Risset went on to found the Institut de Recherche et Coordination Acoustic/Music in Paris (IRCAM). While there, Risset became notable within the field for his analysis of traditional musical timbres. John Chowning, a professor at Stanford University, became the founder of the Center for Computer Research in Music and Acoustics (CCRMA) at Stanford University. Chowning, while serving as the director of CCRMA, became famous for his development of frequency modulation synthesis. The University of Illinois at Urbana-Champaign continued and improved on MUSIC, having earlier developed the ILLIAC computer, which composed the Illiac Suite. The Illiac Suite was founded deeply in computer science roots, using a combination of probabilities and generative grammars known as Markov Chains (Hiller & Isaacson, 1959).

When MUSIC IV reached Princeton, Godfrey Winham and Hubert Howe expanded it to run on an IBM mainframe, naming it MUSIC IVB. It was then further expanded into MUSIC IVBF, which was written in FORTRAN for portability. Following the success of MUSIC IV, Mathews developed the final Bell Labs product, MUSIC V. After its publication, John Gardner and Jean-Louis Richer further developed it at IRCAM to allow it to process digitized sounds as well as synthesized sounds. Further descending
from MUSIC IVBF was MUSIC 11, written at the Massachusetts Institute of Technology by Barry Vercoe (a founding member of the M.I.T. media lab). After a couple more years, Vercoe developed his most lasting addition to the computer music field: Csound, a modular, audio programming language that has had an enormous impact on the computer music community (Vercoe, 1982). Its ability to be extended by its user base has allowed it to grow to encompass some 1200 unit generators.

Csound continues to be developed today. For ease of composition, many companies and individuals have created user-friendly environments that are based on Csound’s capabilities but which slightly ease the burden of having to understand the underlying computer science. Notable among these languages is SuperCollider, a language for real-time audio synthesis (McCartney, 2013). Another is Reaktor, a sound synthesis environment that allows users to create their own instruments. Finally, another advancement on Csound is Max/MSP, the multimedia development platform in which this program was built.

Computers deeply impact the public digestion of modern musical performance. Now, in the context of popular music, the modern period ear expects to hear at minimum some semblance of artificial reverb and a varying amount of autotune on the microphone of the lead singer. In the context of classical Western art music, most publishers ship a recording of birds to be played on a computer at the end of the third movement of Pines of Rome. None of these examples, however, use computers as prominently as the much more modern genre of interactive music, under which this project falls.
Machine Musicianship

One major problem in interactive music is machine musicianship, or the ability of the installation to “perform” with sensitivity to great musical qualities such as melody, form, and phrasing (Rowe, 1996, p. 50). Most technical applications require the implementation of some kind of algorithm or process that when correctly executed produces the desired output after each run. The ability for a computer to create sound with true musicianship is not necessarily such a deterministic behavior. Two different professional instrumentalists will each have their own unique interpretation of a score. Similarly, two well-programmed interactive installations could interpret whatever input (sonic, gestural, etc.) completely differently. As a result of this difference, each program must define its own parameters by which it will operate, interpret and create. Each must also be sure that those parameters are never broken, unless a truly random output is desired.

The expressive abilities of any of the major classical instruments have been measured, quantified, and taught for centuries. After learning the notes, students are immediately taught to play with many different styles of expression. The vast range of affect that can be expressed from any of these instruments is tremendous because of the variability of the input (Nakra, 2000). A slight change in embouchure, air support, bow strength, or wrist pressure can immensely change the color of the output. Any of these variables and more provide the nearly infinite dimensions by which the input can vary. In classical orchestras, therefore, the construction of the instruments become the parameters by which they will operate, while the various musicians provide specific, finely tuned input. This installation, therefore, has to necessarily extend beyond the classical orchestra
in order to contribute to the field of music. The instruments used in this installation are synthesized in real-time, based on the input of the user. The instruments are therefore far more dynamic than would be physically possible for any classical instrument.

Sound synthesis (and more specifically, additive synthesis), the method by which this project creates the instruments that “play” the composed sound, is a core concept within interactive music. Whatever tones are used to create the final product of the installation generally move beyond simple sinusoidal waves. One popular method of creating more complex waves is additive synthesis. Additive synthesis is the process of combining a sound wave at some fundamental frequency \( f \) with many other waves at different volumes that vibrate at frequencies that fall on the fundamental’s harmonic series. The harmonic series is the set of pitches that are whole integer multiples of \( f \), i.e., \( kf, k \in N \). This combination of waves, which can simply be represented as a set of ratios of the volume at which each individual wave is being sounded as compared to the fundamental, create what musicians refer to as timbre. The sets, which describe each timbre being used by the program, become analogous to an orchestra’s instruments under the direction of a skilled composer. Depending on the composition, each “instrument” has the capacity to be explored in different registers, in combination with other “instruments”, played at different volumes, and most interestingly - from the context of computer music - to have its timbre manipulated.

The interplay of the different characteristics that constitute the synthesized instruments that make up this installation is precisely the value of computers in interactive music. The ability of these programs to operate on the fundamental essence of instruments is the advantage that they have over what would otherwise be physical
limitations. A clarinet being played by the most virtuosic performer in the world will never be able to vibrate with the even harmonics of whatever fundamental frequency is being played. The physical construction of a clarinet’s cylindrical bore limits its timbral range in that way. Under the assumption that timbral variety has a direct impact on emotional output possibilities, a program created for the express purpose of mimicking and expanding on the clarinet would be more effective than a physical clarinet because it would be able to manipulate its timbre in ways a professional clarinet player could not (Hailstone et al., 2009).

**Variables and Their Measurements**

The goal of this project is to establish a relationship between emotional input and created sonic output. As a result, the technical limitations of measuring every dimension of the input is one the largest difficulties. While some variables are relatively trivial to measure, some are much more arduous. All of these variables are made all the more difficult to measure because of the added dimension of time. Each variable can be measured to provide data for the current state of the program, but until the variable is measured over a stretch of time it cannot provide any information as to where the emotional content of the speech is going or what it is implying about the speaker. Each variable provides its own unique information, and each plays a role in the artistic interpretation of the speech.

The first variable measured in my program is volume, which is easily measured as an instantaneous value by measuring the amplitude of the sound wave of the speech. When placed in a temporal context and measured in small phrases, the program measures
the additional variables of average volume, minimum volume, maximum volume, and variance of volume. Average volume, especially when compared to known thresholds (zero volume, average speaking volume, and average yelling volume) yields information about the emotional state of the speaker for that phrase, but does not give any indication as to the overall affect of the speaker. Minimum volume and maximum volume give slightly more information, as the difference between them denotes a volume range. When the range is significantly wide and the variance is large, this is an indication of a more ecstatic speaker. Contrastingly, when the range is significantly narrow and the variance is small, this is an indication of a more reserved speaker. When the maximum and minimum volumes create a large range yet the variance is small, the three variables do not yield any useable information because one or both of the minimum and maximum are outliers.

All information pertaining to measured input comes into the program simultaneously because of the nature of speech. Phrases are consecutive and continuous. As a result, all of these variables must be calculated concurrently with execution. Minimum and maximum values are the simplest: by storing a value and comparing it to each input, the program maintains the current minimum and maximum. At the beginning of a new phrase, the program resets. Average is somewhat more difficult: the program must maintain a running sum of volume samples, as well as the total number of samples that have been processed. The program then divides to determine the average. The variance is calculated by maintaining the same running sum as average, as well as the running sum of the squares:

\[
\begin{align*}
  n_1 &= \sum_{i=1}^{N} x_i \\
  n_2 &= \sum_{i=1}^{N} x_i^2
\end{align*}
\]
The variance is then calculated at the end of the phrase, by evaluating the following equation:

$$\sigma^2 = \frac{n_2}{N} - \frac{n_1^2}{N}$$

In this manner, no buffer is ever maintained to be processed later. Each variable is calculated throughout the regular functioning of the program.

When measuring the average volume over a period of multiple phrases, patterns can begin to emerge if the user is increasing or decreasing in volume over an extended period of time. To measure this, the program differentiates volume with respect to time over many phrases. This derivative of volume is useful for determining the immediate changes in volume, and therefore emotion. In order to determine a deeper-set pattern of increasing or decreasing volume, the program differentiates once more with respect to time. This second differentiation calculates the second derivative, from which the program can infer larger trends in volume changes over the square of the number of phrases that were measured for the first derivative. In other words, if the program measured the volume changes over the course of three phrases in order to determine the first derivative, it would then measure the second derivative over the course of nine phrases by taking the three values of the first derivative.

Pitch is another variable that the program measures that contributes to the outputted sound from the installation. Measuring pitch is significantly more difficult than volume, however. The algorithm typically used is a Fourier transform, which is a mathematical function that essentially breaks up any continuous signal into its component sinusoids by outputting each pair of amplitudes and frequencies. The manner by which it does this is based on the fact that as analog signals are converted to digital, they are
stored as complex numbers that make up the wave. In order to calculate the discrete Fourier transform (DFT), each complex number $x_0, ..., x_{N-1}$ is fed into the formula:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{i2\pi kn/N}$$

This formula transforms the complex numbers into an $N$-periodic sequence of complex numbers. Each resulting complex number has frequency, amplitude and phase encoded into it. Each sinusoid’s frequency is $k/N$ cycles per sample; its amplitude and phase are, respectively:

$$\frac{|X_k|}{N} = \frac{\sqrt{Re(X_k)^2 + Im(X_k)^2}}{N}$$

$$\arg(X_k) = atan2(Im(X_k), Re(X_k))$$

The $Re$ and $Im$ functions are simply extracting the real and imaginary portions of the complex number. Within the phase function, the $atan2$ function determines the arctangent of the input, the second argument being used to determine the quadrant. The $arg$ function is a function from complex analysis that gives the angle between the line joining the point to the origin and the real axis, which is the phase. With the wave broken down into individual sinusoids, it is then a somewhat trivial matter of finding the wave with the highest amplitude.

The object in Max/MSP that the program uses for determining pitch is called the pitch object, developed by Tristan Jehan, based off of the fiddle~ object created by Miller Puckette. The object utilizes a fast Fourier transform (FFT) algorithm, which is related to the formulas that determine the DFT, except that they are optimized for efficient calculation. Within the pitch object, the FFT algorithm is performed which generates the Fourier series. The frequency of each of the resulting sinusoids is an integer multiple of
the fundamental, i.e., each wave is either the fundamental or a harmonic. The sinusoid with the largest amplitude is taken to be the fundamental pitch (Mitre, Queiroz, & Faria, 2006). Thanks to hardware acceleration and the huge improvements in technology over the last couple of decades, this conversion takes place within a few milliseconds of input. Once pitch has been determined, like volume, it can be averaged over a period of time. Also like volume, maximum, minimum, and variance values are determined for each phrase. Finally, the derivative of average pitch (known musically as contour) and the second derivative of average pitch (musically akin to form) are calculated. Contour is a particularly interesting variable, which we subconsciously use to evaluate the speech of others every day. For example: mothers who speak with a descending pitch contour are more likely to be comforting infants, whereas a bell-shaped contour may serve to indicate approval or a prompt for attention (Spence & Moore, 2002). In this way, the contour of a pitch value serves as another indicator of emotion.

A third set of variables that the program measures encompass the rate of speech of the speaker. Since the program is constantly attempting to evaluate the pitch of the speaker, the number of discrete pitches that the voice hits within a half second period determines speech rate. In order to simplify the measurement, the pitch that is normally measured in Hertz is instead measured in MIDI values using the conversion equation:

\[ p = \left[ 69 + 12 \times \log_2 \left( \frac{f}{440 \text{ Hz}} \right) \right] \]

This equation could be manipulated to use any two values used on the MIDI scale. I chose A = 440 Hz, which corresponds to the MIDI value of 69. The log function determines the octave, multiplies by 12 to determine how many octaves above A the pitch is, and then adds the base value. The value of \( p \) is then taken as the current pitch.
After the number of pitches encountered in a phrase is known, similar variables to
volume and pitch can be determined for speech rate including average number of pitches
per second, variance, and the first and second derivative. In this case the derivatives give
information on the rate at which the speech is increasing or decreasing and how quickly
that increasing or decreasing is happening.

The final set of variables that the program measures is timbre. Timbre is by far the
most difficult and time-consuming variable to measure and quantify. In order to “measure”
it, the program determines the relative amplitudes of the incoming complex wave at each
of the harmonic frequencies above the fundamental. The set of these amplitudes, known
as Fourier series coefficients, is a rough estimation of the incoming timbre, under the
assumption that there is only one speaker. In summary, the values that the program keeps
track of in order to calculate all necessary values are as follows:

number of volume samples $v_i = N_v$

sum and square sum of $v_i = \sum_{i=1}^{N_v} v_i \sum_{i=1}^{N_v} v_i^2$

min and max $v_i$ of a phrase $= v_{max} \ v_{min}$

number of volume samples $p_i = N_p$

sum and square sum of $p_i = \sum_{i=1}^{N_p} p_i \sum_{i=1}^{N_p} p_i^2$

min and max $p_i$ of a phrase $= p_{max} \ p_{min}$

number of rate of speech samples $r_i = N_r$

sum and square sum of $r_i = \sum_{i=1}^{N_r} r_i \sum_{i=1}^{N_r} r_i^2$
Translation from Measurements Into Sound

In David Rokeby’s article “Transforming Mirrors: Subjectivity and Control in Interactive Media,” (1998) he describes an interactive system as an aesthetic relationship between the user and the interface and the interactive artist as a creator of these relationships. In that light, my installation is an aesthetic relationship that links the emotional content of a speaker to the sonic landscape of the installation. While the speech of the speaker is not reflected directly, it is refracted, decomposed, and recomposed as part of the landscape.

Rokeby describes feedback loops that govern the interaction between the user and the program. In this context, the feedback loop can be either positive or negative. In a positive loop, the emotional content of the user is reflected back and amplified, lending new weight to the effect of the music. In a negative loop, the emotional content could be refracted in such a way as to negate or redirect the original gesture from the user. In essence, this project is a system that allows the computer to reinforce or redirect the emotional impact of the music. In composition, this system allows for the creation of pieces of music that affect the audience in ways they could not have experience with physical instruments.

The installation uses a system of many probabilities to determine its output. The background sounds include wind (many noise objects with filters), low rumblings (noise and cycle objects played together with more filters), and high flutters (more noise and cycle objects with filters). The background sounds increase in volume and intensity (wider gates with the filters and more objects used) in step with the speech of the user. The volumes used for each of these objects also stays within the range of $v_{max} - v_{min}$ to
stay proportional to the speech, while also taking into account the variance of the volume samples. Finally, the entrances and duration of each of these noises is proportional to the rate of speech of the user, and the variance of that rate of speech.

The foreground sounds are comprised of many different “instruments” of my design. Some are based on recordings of actual orchestral instruments (a flute, an oboe, a saxophone, and a trumpet) while others are designed within the program (such as the bottom subpatch). The structure of when and how they are played is more built on the actual pitches of the speech. There are probabilities built in for each “instrument” to play pitches within the range of $p_{\max} - p_{\min}$ but not necessarily mirroring the pitches of the speech. The variance also plays a role, as pitches that stay within one standard deviation are more highly favored than within two, and much more heavily favored than those pitches extending beyond two.

Finally, the derivatives of each of the measurements come into play. The first derivative of volume is used to indicate a sudden increase, which is reflected in one or multiple “accented” notes in the background sounds. The second derivative of volume can similarly trigger a swell in the background volumes that, while subtle, greatly affects the texture of the landscape. The first derivative of pitch sends information to the background as to which it should favor, higher or lower sounds. For example, a high first derivative of pitch indicates that the most recent phrase was much higher than the previous, indicating a sudden shift in emotional content, which is reflected by the probabilities of the background sounds. The second derivative of pitch functions in a somewhat similar manner, in that it can trigger swells in the background and foreground sounds based on how significant the difference is.
Results and Conclusions

The purpose of this project was to explore the relationship between affect and music in the context of a live, interactive system built in Max/MSP. By creating this installation, the user gains a unique experience: a new perspective on the emotion in their speech, music, or both. In this paper, the basis for this installation in various fields of study is explained, as well as the significant portions of each field that directly relate to this project. This paper examined each variable that the program measures, as well as how each of those measurements contributes to the music that is being created. Finally, this paper discussed the current state of the installation, and its various capabilities.

In a lot of ways, the development of this installation mirrored its actual purpose. As the project grew into fruition, the sounds that were being output from the program were constantly changing and evolving. As the sounds changed, the direction of the development of the project changed as well. In this way, the conception of the project and the actual output began to inform each other to the point that they eventually met. Consequently, the output of the program lent more information to the both the music and the emotion in speech: the music grew deeper and more complex, and the emotion was able to be practiced and measured in a more empirical fashion.

This project holds many possibilities for further research in both the fields of psychology and music. In terms of psychology, this project could be used as a basis for a study of whether or not people can create similar aural landscapes through their speech, and what it means to effectively communicate emotion. Multiple empirical experiments in that field could be conducted by measuring the pitch and volume ranges sounded in relation to different emotional inputs. In terms of music, this installation is consistently
generating new and interesting compositional gestures that would not be possible in other programs. Composers could use this project as either a simple means of inspiration (by hearing sounds and themes they would not otherwise hear), or as a vessel by which to record and create (using the program to create unique compositions). Personally, this project has changed my view of the definition of music, and I hope that it can similarly impact any potential user of this software.
References


