REAL-TIME MUSIC VISUALIZATION USING RESPONSIVE IMAGERY

Robyn Taylor robyn@cs.ualberta.ca
Pierre Boulanger pierreb@cs.ualberta.ca
Daniel Torres dtorres@cs.ualberta.ca

Advanced Man-Machine Interface Laboratory,
Department of Computing Science, University of Alberta
T6G 2E8 Edmonton, Alberta. Canada

ABSTRACT

We present a music visualization system that allows musical feature data to be extracted in real-time from live performance and mapped to responsive imagery. We have created three example mappings between music and imagery, illustrating how music can be visualized through responsive video, virtual character behaviour, and interactive features inside an immersive virtual space. The system is implemented using a visual programming paradigm, enhancing its ease-of-use and making it suitable for use by collaborative teams containing both artists and scientists.

1. INTRODUCTION

We have created a real-time system capable of visualizing live music using responsive imagery. A musician can interact with a virtual environment in a natural and intuitive way, using his or her voice as input to a visualization experience.

In this paper, we will describe our system which allows us to visualize live musical performance using responsive imagery. We will present our musical feature data extraction routine and describe three examples of mappings between music and responsive imagery that have been created using our system. Example mappings include:

- vocal timbre and piano chord data visualized through responsive video
- melodic information visualized through the responses of a virtual character
- vocal dynamics visualized through interactive aspects of an immersive virtual space

Our music visualization system has been used to create an audio-visual performance piece (see Figure 1), and is currently being used to develop additional multimedia applications. It is designed to facilitate artist/scientist collaboration, thus visual programming platforms are used whenever possible to develop the audio-visual processing components.

Section 2 of this paper discusses our motivation in creating a music visualization system, and Section 3 provides an overview of the system architecture. Section 4 explains our musical feature data extraction and organization routines, while Sections 5, 6, and 7 present examples of music visualization techniques that have been created using this system.

Figure 1 – A performer interacts with a responsive video visualization

2. MOTIVATION

New Media art often combines modern technologies with traditional art forms to create new platforms for artistic expression. Virtual and augmented reality technologies can be used to visualize live music for the purpose of creating an artistic experience.

Examples of existing music visualization artworks include immersive installations that respond to vocal or instrumental input. Ox's *Color Organ* [9] is one example of such an installation, allowing users to navigate virtual landscapes generated by assigning geometric and colour data to characteristics found within input musical streams. *The Singing Tree* created by Oliver et al. [8] allows users to immerse themselves in an environment containing both physical and virtual elements, inside which their vocalizations result in auditory and visual feedback.

Virtual characters can be used to illustrate aspects of musical performance. *Dancing Gregor* (created by Singer et al. [11]) and Goto's *Cindy the Virtual Dancer* [6] are examples of virtual characters that synchronize their movements to input provided by live musicians.

Levin and Lieberman's *Messa di Voce* [7] is a concert performance piece that generates glyphs in order to augment live performers' vocalizations.

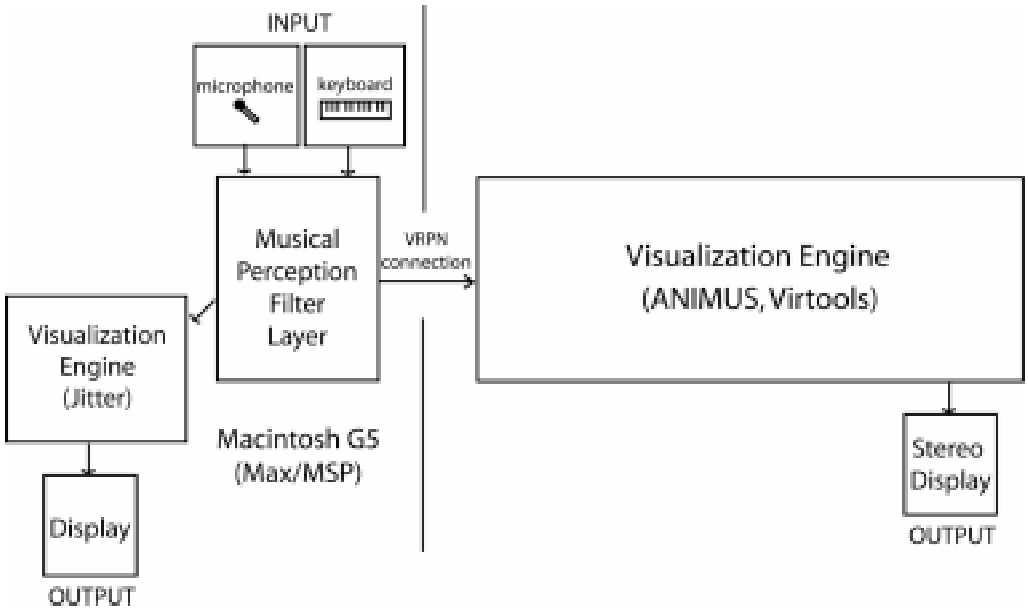

Figure 2 – System Architecture

Each of these visualization systems uses a different mapping between musical and visual content and illustrates the interactivity between musician and visualization in a different way.

Our music visualization system is flexible enough to visualize musical data through responsive video imagery, virtual character behaviour, and responsive virtual environments. We have designed our system to facilitate multiple mappings between sound and imagery, so that it may be expanded in the future to create more music visualization projects.

The distributed nature of our system encourages code re-use and task delegation. Since New Media artwork is often created by interdisciplinary teams, our system attempts to maximize ease-of-use in the creative process so that individuals without formal training in computer programming can participate in the design and development of audio-visual applications.

3. SYSTEM ARCHITECTURE

Our music visualization creation system is developed in a distributed fashion (see Figure 2) allowing the tasks of musical feature extraction and analysis to be separated from the visualization system. This simplifies the task of creating multiple mappings between music and imagery, as additional visualization front-ends may be introduced in order to create new applications.

3.1 Musical Input

The input to the system comes from a digital keyboard and vocal microphone. The keyboard and microphone are connected to a Macintosh G5 which handles the task of parameterizing and organizing musical input. Musical input is processed inside the Musical Perception Filter Layer, which is implemented inside the visual programming environment, Max/MSP [3]. The Max/MSP environment is specifically designed to simplify the creation of musical processing applications. It provides numerous tools and functions that can be used to analyze and manipulate analog and MIDI data.

3.2 Visualization

The parameterized input is used to control the appearance of the visualization environments. We have used the system to create visualizations using three different engines – Jitter, ANIMUS, and Virtools.

Jitter [2] is a video processing environment that we use to create a music visualization illustrating vocal timbre through manipulated video parameters.

The ANIMUS Framework [15][16] is a virtual character creation system. We use it to visualize melodies through the emotional responses of animated characters.

Virtools [4] is a virtual environment simulator. We illustrate vocal dynamics in Virtools by manipulating aspects of the virtual space in response to live musical input.

The Jitter video processing environment runs on a Macintosh, and is built into the Max/MSP processing environment. The ANIMUS Framework and Virtools environments run on remote PC's running the Windows XP operating system.

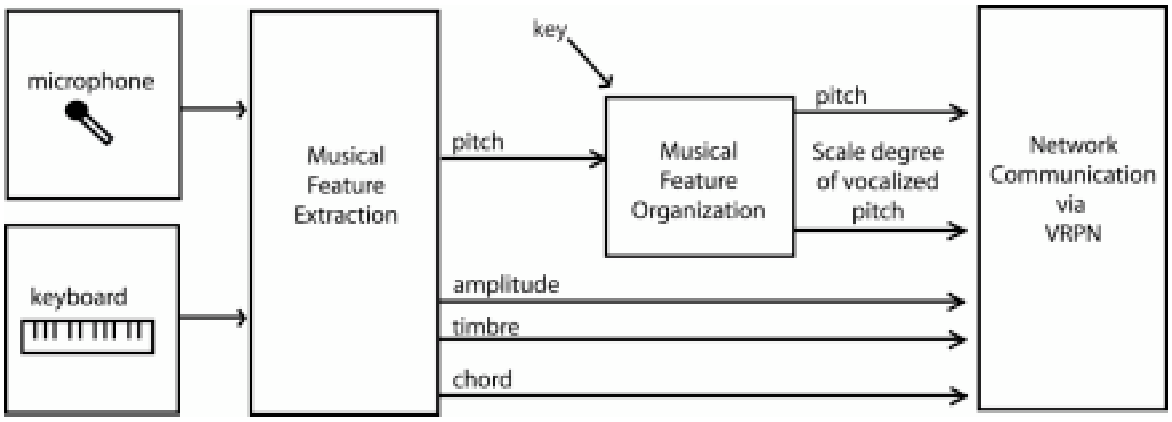

Figure 3 – The Musical Perception Filter Layer

3.3 Displayed Imagery

The output of all the visualization engines can be displayed upon large-scale projection screens. Additionally, the ANIMUS and Virtools simulation engines are capable of generating life-sized stereoscopic imagery. Virtools can be configured to control stereoscopic visualizations inside three-walled immersion rooms.

3.4 Networked Communications

Communication between the musical feature extraction system and the remote computers used to visualize the musical data (the computers running the ANIMUS and Virtools engines) is performed using the Virtual Reality Peripheral Network (VRPN) library [14]. VPRN is designed to facilitate generic distributed communications for virtual reality applications. VRPN is highly flexible, allowing our system to be capable of communicating with visualization clients housed on Windows, OS X, Linux or IRIX machines.

4. REAL-TIME MAX/MSP SOUND EXTRACTION MODULE

In order to visualize live music, a stream of musical input must first be parsed into discrete parameters. We use Cycling `74's sound processing environment Max/MSP [3] to encapsulate this process into a module called the Musical Perception Filter Layer (see Figure 3).

4.1 Max/MSP

Max/MSP allows musicians to create music processing applications by describing the dataflow between hierarchical submodules, known as *objects*. Max/MSP's ease of use and modularity have made it an industry standard in electronic music development. Numerous users create and share their objects with the large Max/MSP user community.

4.2 Vocal Feature Extraction

To analyze a musician's singing, we use one such user-created Max/MSP object called `fiddle~` [10]. `fiddle~`, created by Puckette et al., performs a Fourier analysis of the incoming sound signal and outputs information about the singer's vocalization.

4.2.1 Pitch and Amplitude Extraction

`fiddle~` is designed for use as a pitch and amplitude tracker. It outputs a constant stream of pitch and amplitude information. We use these values to track the pitch and loudness of the live musician's singing.

4.2.2 Pitch Organization

Since the objective of this application is to facilitate the creation of music visualizations that are aesthetically pleasing to humans familiar with Western tonal music (the tonal system familiar to listeners of modern popular or folk music, as well as pre-twentieth century classical music), an existing schema for musical data representation [5] which is congruent with the rules of Western musical tonality has been modified in order to organize the extracted vocal pitch data. Extracted pitches are encoded in terms of their intervallic relationship to the (pre-determined) key signature of the input melody, simplifying any later music-theoretical processing of the melodic data.

4.2.3 Assessment of Vocal Timbre

In addition to outputting estimations of the singer's pitch and loudness, `fiddle~` also makes available data describing the frequencies and amplitudes of all the harmonic components that are contained within the harmonic spectrum resulting from the Fourier analysis performed on the incoming sound stream.

Our system assesses the weighting of energy amongst the partials in the sound, creating a parameter that a vocalist can control by modifying her vocal timbre.

Vocal timbre refers to the characteristics of a vocalized sound that make it recognizably different from other vocalizations uttered at the same pitch and loudness. A voice's timbral character is determined by the way energy is distributed amongst the partial frequencies in the sound. Literally meaning tone colour, a vocalist's timbre is what is being addressed when a voice is described using terms such as dark, bright, rich, or strident.

Our system creates a description of the singer's vocal timbre by examining the harmonic spectrum output by the `fiddle~` object. Vowel choices are roughly identified by comparing the reported amplitude at each harmonic in the vocalized spectrum to known data characterizing vowel formation.

4.4 Piano Chord Identification

In addition to interpreting analogue vocal data, MIDI data from the digital piano keyboard is also assessed. A sub-module of the Musical Perception Filter Layer is used to monitor MIDI events and identify the chords played on the digital piano by comparing them to a list of "known" major and minor chords. Any inversion of the chords is recognized, and this module could easily be expanded to incorporate other types of chord data.

4.5 Broadcasting the Musical Feature Data

After the musical feature data (vocal pitch, amplitude, and timbral information as well as keyboard chord data) has been identified and extracted, it is then transmitted to the visualization engines to be represented through responsive imagery. As described in Section 3.3, this task is facilitated by the use of the VRPN library [14].

We have created a Max/MSP object encapsulating the VRPN technology required to run a server on the Macintosh system. This object, `vrpnserver`, broadcasts the extracted musical feature data so that our visualization engines may receive information about the musical performance.

5. VISUALIZATION THROUGH RESPONSIVE VIDEO IMAGERY

Our first visualization environment allows a musician to interact with a responsive video visualization by manipulating her vocal timbre and playing chords on a digital piano.

By mapping responsive video parameters to aspects of the musician's live performance, we used our music visualization system to create a multimedia piece called *Deep Surrender* that has been performed live in concert.

The piece was created using Cycling `74's video processing engine, Jitter [2] to manipulate the responsive video space.

5.1 Jitter

Jitter is an add-on to the Max/MSP system. As such, its visually programmed interface is consistent with that of Max/MSP, and Jitter operations are executed inside of the Max/MSP processing loop. Jitter's extensive library of image processing functions allows users to perform matrix-based image manipulation operations on still images and video files. We make use of two such Jitter functions, the `jit.scalebias` and `jit.chromakey` operations in order to create the effects seen in the *Deep Surrender* performance (see Figure 4.)

5.2 Visualizing Piano Chords

In the production, chords played on the piano keyboard affect the colour balance of the video imagery.

To map piano chords to colours, we use a strategy similar to the one used by Jack Ox in her *Color Organ* installation [9]. A music theoretical structure, the Circle of Fifths, is mapped to a standard colour wheel, associating a colour value with each chord.

The Circle of Fifths relates chords to one another in terms of their harmonic similarities. Chords that are closer to one another on the Circle are more similar to one another than chords that are located further away. Mapping a colour wheel to the Circle of Fifths makes chords that are musically similar appear similar in colour.

Jitter's `jit.scalebias` operation adjusts the colour balance of a displayed video, using matrix multiplication to scale the hue of the moving images.

In the *Deep Surrender* performance, the live performer manipulates the colour balance of the video playback by playing different chords on the keyboard.

5.3 Visualizing Vocal Timbre

The vocalist's singing also affects the visual environment. Image layering is used to allow the singer's vocalization to introduce coloured images into the video stream.

The `jit.chromakey` operation is used to superimpose images into the displayed video scenes. Chroma-keying (also known as blue- or green-screening) is commonly used in film and video productions. The process allows elements from one video to be layered into a second video, creating a

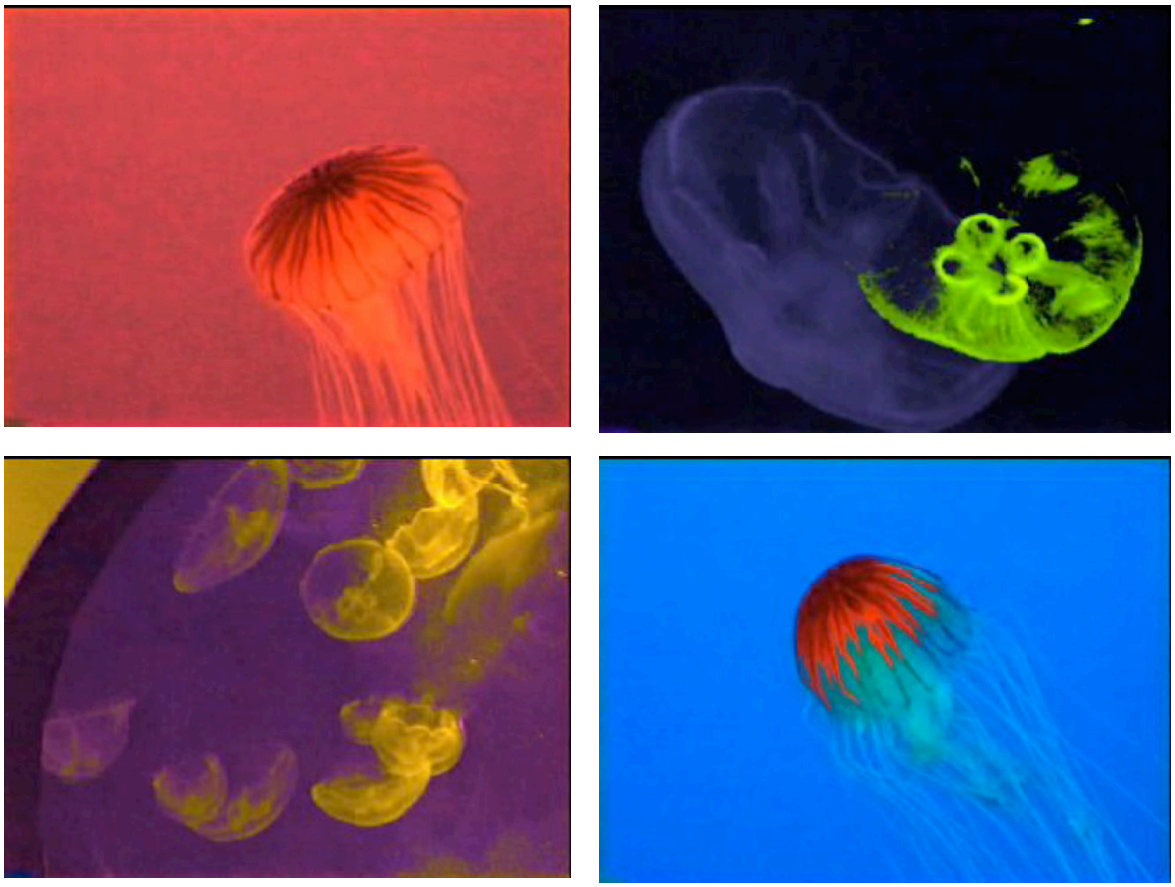

Figure 4 – Images from *Deep Surrender*

composite video stream containing elements from two separately filmed video segments.

In *Deep Surrender*, a soprano chroma-keys images from one video stream into another by making sounds with her voice.

She controls the colour of the chroma-keyed imagery by manipulating her vocal timbre.

By mapping the amplitudes found at each partial frequency of the analogue vocal input to an RGB colour selection function, we assign a colour to the singer's timbre.

The amplitude of the energy found at the fundamental frequency of the sound affects the red component of the colour selection, while the amplitude of the second and third partial frequencies control the blue and green components.

Focused sounds (like /i:/ or /u:/ vowels or extremely high pitches above soprano High C) have a high intensity of tone amplitude weighting at the fundamental frequency. Our mapping yields red-orange colours in these cases. If the soprano produces a spread sound at a moderate pitch (like /a:/ sung near the pitch of A440) there is increased amplitude at the second and third partial frequencies in her harmonic spectrum. This results in a blue-green colour value.

By making sounds, the singer introduces new objects into the scene, and by choosing the type of sound she makes, she determines their colour.

5.4 The *Deep Surrender* Performance

The intention of the *Deep Surrender* piece is to illustrate how an artist can harness anxiety and adrenalin to produce a beautiful performance. This is achieved by utilising the visual metaphor of a jellyfish -- a creature both beautiful and terrifying. The artist's musical performance manipulates the jellyfish representation, in order to convey how the artist interacts with and overcomes her anxiety.

The interaction techniques (piano playing and singing) are used to manipulate the jellyfish imagery in order to convey the musician's emotional state to the audience. Different video segments are manipulated during each section of the piece, and the performer adjusts her vocalizations as the performance progresses, illustrating her growing confidence as she overcomes her anxiety.

We have used our system to perform this piece in concert, and often perform it in the laboratory setting in order to show visiting tour groups an example of an artistic usage of visualization technology.

Figure 5 – A singer interacting with a virtual character called Alebrije

6. VISUALIZATION THROUGH THE BEHAVIOUR OF A RESPONSIVE VIRTUAL CHARACTER

A second way of visualizing music using our system illustrates emotional content in sung melodies through the responsive behaviour of a virtual character [12] [13].

6.1 The ANIMUS Framework

The virtual character used in this implementation was created using Torres and Boulanger's ANIMUS Framework [15] [16].

The ANIMUS Framework supports the creation of responsive virtual characters using a three-layered process. Musically responsive character behaviour is defined in three layers: the perception layer, the cognition layer, and the expression layer.

6.1.1 Perception Layer

In the perception layer, the virtual character perceives the musical feature data which is extracted from the live musical input and communicated by the Musical Perception Filter Layer.

6.1.2 Cognition Layer

In the cognition layer, the virtual character's simulated "personality" is defined. In this layer, the way in which the character's simulated "emotional state" is affected by musical stimuli is specified.

6.1.3 Expression Layer

In the expression layer the virtual character's internal state is expressed to the viewing audience through animations. Animations are created at run-time using DirectX functionality to interpolate between emotive keyframe poses and generate character movement on the fly.

Each layer in the ANIMUS Framework is defined by a designer who defines the functionality and animation parameters through an XML scripting language, then implemented by a developer who creates the code required to fulfill the designer's specifications. This encourages the close collaboration between designers and developers that is essential when creating an artistic application.

6.2 An Example of a Virtual Character

Our example of a virtual character music visualization illustrates sung melodies through the behaviours of Alebrije, a lizard-like character (see Figure 5.)

Alebrije's perception layer receives information from our Musical Filter Perception Layer. He is aware of the pitches the singer sings, both in terms of their raw pitch values, and in terms of their intervallic context with respect to the key signature of the sung piece.

To implement Alebrije's cognitive layer, we base his simulated emotional state upon a metric devised by Deryck Cooke [1]. Cooke's study correlates musical emotion with features in tonal melodies. Cooke's metric assigns an emotional meaning to each interval in the scale (as an example, he states that minor thirds signify despair, while major thirds signify joy.)

Each note in an incoming sung melody has a specific intervallic tension (relative to the tonic note of the key signature), and as each note is sung we affect Alebrije's emotional state in response to this tension. His simulated emotional state becomes more or less happy based on Cooke's interpretation of the significance of the perceived melodic data.

To express Alebrije's emotional state in a way that is visible to the viewing audience, his posture transitions between "happy" and "sad" keyframe poses.

6.3 The Resulting Animation

The Alebrije visualization is capable of expressing appropriate animated responses to the emotional content in tonal melodies. When presented with the folk song *Greensleeves*, his emotional state registers "sadness" in response to its wistful melody, which is characterized by minor thirds and minor sixths. *Twinkle Twinkle Little Star*, with its major melody and prominent perfect fifths, makes his emotional state transition towards "happiness". These states are expressed through emotive animations, which allow him to express his simulated emotions in a way that is visible to the audience.

We display Alebrije on a life-sized stereoscopic screen (see Figure 5) so that the viewing audience may perceive both the virtual character and the human performer on the same scale. This enhances the realism of the visualization and makes it particularly suitable for use in virtual theatre productions, since the human and virtual actors appear in a unified setting.

We intend to develop this simulation further in order to create musically responsive virtual characters with greater expressive capabilities. We are currently working with a media artist who is creating a 3D character with a wide library of emotive poses so that we may develop a compelling artistic performance incorporating the musically responsive virtual characters.

7. VISUALIZATION INSIDE A VIRTUAL SPACE

The third method we have implemented to visualize music using our system uses the Virtools development environment to create immersive virtual spaces that are responsive to musical input.

7.1 Virtools

The Virtools' authoring application [4] is a visual programming environment which allows designers of virtual reality applications to create immersive visualizations by defining Virtools Behaviours and describing how they affect the properties of objects in the virtual environment.

Connecting our Musical Perception Filter Layer's musical control system with Virtools' intuitive authoring environment allows us to rapidly develop music visualization applications. Using Virtools' visual programming environment to create visualizations allows different musical imaging strategies to be quickly and easily defined, tested, and modified.

The connection between Max/MSP's music processing environment and Virtools' virtual reality simulator allows both the musical and visual aspects of immersive music visualization projects to be implemented using specialized development environments that expedite the process of audio-visual application development.

7.2 Immersive Spaces

The Virtools simulator is capable of retargeting our music visualizations so that they may be displayed inside life-sized immersion rooms. Immersion rooms are CAVE-like three-walled structures comprised of large stereoscopic display screens. When a virtual environment is displayed in an immersion room, the immersed users experience a realistic and believable sense of actually being inside the virtual space.

7.3 Musical Control of the Virtual Environment

We interact with the Virtools simulator by connecting the Musical Perception Filter Layer to the Virtools event loop. We have built a Virtools building block that connects to our musical feature data server.

Our building block, called `MusicController`, receives information about the performer's pitch, loudness, and timbre, as well as information about any chords that are played on the digital piano. These musical features can then be used to control aspects of the Virtools simulation.

We have created a responsive environment inside Virtools that allows a user to modify aspects of the virtual space using his or her voice.

To illustrate the singer's vocal dynamics, clouds of particle fog are generated in response to singing. The colour of the clouds is controlled by the pitch the user sings (higher pitches are visualized with red-orange colours while lower pitches are visualized with blue-green colours) and the size of the clouds increases as the singer's loudness increases (see Figure 6.)

The particle cloud is a particularly responsive form of visual feedback, as the fog is evocative of the breath the user uses to create a vocalized sound.

Figure 6 – Vocalization visualized with fog

The Virtools visualization environment is particularly user-friendly, as its visually programmed authoring environment provides users with a large library of pre-built functionality. The particle fog visual metaphor was created using Virtools' built-in particle cloud simulation routines.

Enhancements to the responsive virtual room are currently being developed, so that visitors to our laboratory may experiment with the musical input mechanism and experience visual feedback in the immersive virtual space.

8. CONCLUSIONS

This system is designed to facilitate the creation of artistic applications that use musical control to interact with responsive virtual environments.

We have created this system in a way that makes the process of artist/scientist collaboration as easy as possible.

For this reason, we chose to use visual programming to develop system components whenever possible. Max/MSP, Jitter, and Virtools use visual techniques to describe the flow of data within an application, allowing those without training in traditional computer programming to participate in the development process with greater ease.

While the ANIMUS Framework requires developers to create their applications using traditional coding methods, its character-creation processes encourage task delegation, making extensive use of scripting languages to allow non-technical team members to participate in the design process of responsive animation applications.

We have used our system to develop three music visualization applications. One of these applications (the responsive video visualization) has been used to create an audio-visual work that has been performed in a live concert setting. The other applications (the virtual characters and the responsive room) are currently being used to develop additional performance pieces and installations.

Modern visualization technologies can be used to produce compelling imagery and responsive interaction. We look forward to using this system to continue our development of New Media artwork facilitated by computer science research.

ACKNOWLEDGEMENTS

The use of the VRPN library was made possible by the NIH National Research Resource in Molecular Graphics and Microscopy at the University of North Carolina at Chapel Hill, supported by the NIH National Center for Research Resources and the NIH National Institute of Biomedical Imaging and Bioengineering.

The source video footage for the *Deep Surrender* video production was filmed by Melanie Gall.

The textures on the models used in the Virtools simulation are from http://www.ktn3d.com/.

REFERENCES

[1] Deryck Cooke. *The Language of Music*. New York: Oxford University Press, 1959.

[2] Cycling '74. Jitter, 2004.

[3] Cycling '74. Max/MSP, 2004.

[4] Dassault Systémes. Virtools, 2005.

[5] Diana Deutsch and J. Feroe. The internal representation of pitch sequences in tonal music. *Psychological Review*, 88:503-522, 1981.

[6] Masataka Goto and Yoichi Muraoka. Interactive Performance of a Music-Controlled CG Dancer. http://staff.aist.go.jp/m.goto/PROJ/ip-j.html.

[7] Golan Levin and Zachary Lieberman. In-situ speech visualization in real-time interactive installation and performance. In *Proceedings of The 3rd International Symposium on Non-Photorealistic Animation and Rendering*, pages 7-14. ACM Press, 2004.

[8] William Oliver, John Yu, and Eric Metois. The Singing Tree: design of an interactive musical interface. In *DIS '97: Proceedings of the conference on Designing interactive systems: processes, practices, methods, and techniques*, pages 261-264. ACM Press, 1997.

[9] Jack Ox. 2 performances in the 21st Century Virtual Color Organ. In *Proceedings of the fourth conference on Creativity & Cognition*, pages 20-24. ACM Press, 2002.

[10] M. Puckette, T. Apel, and D. Zicarelli. Real-time audio analysis tools for Pd and MSP. In *Proceedings of the International Computer Music Conference*, pages 109-112. International Computer Music Association, 1998.

[11] Eric Singer, Athomas Goldberg, Ken Perlin, Clilly Castiglia, and Sabrina Liao. Improv: Interactive improvisational animation and music. In *Proceedings of the International Society for the Electronic Arts (ISEA) Annual Conference*, 1996.

[12] Robyn Taylor, Pierre Boulanger and Daniel Torres. Visualizing emotion in musical performance using a virtual character. In *Proceedings of the Fifth International Symposium On Smart Graphics*, pages 13-24. Springer LNCS, 2005.

[13] Robyn Taylor, Daniel Torres and Pierre Boulanger. Using music to interact with a virtual character. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 220-223, 2005.

[14] Russell M. Taylor II, Thomas C. Hudson, Adam Seeger, Hans Weber, Jeffrey Juliano, and Aron T. Helser. VRPN: A device-independent, network transparent VR peripheral system. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 55-61. ACM Press, 2001.

[15] D. Torres and P. Boulanger. The ANIMUS Project: a framework for the creation of interactive creatures in immersed environments. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 91-99. ACM Press, 2003.

[16] D. Torres and P. Boulanger. A perception and selective attention system for synthetic creatures. In *Proceedings of the Third International Symposium On Smart Graphics*, pages 141-150, 2003.