MUSICAL COMPUTER GAMES PLAYED BY SINGING

Perttu Hämäläinen, Teemu Mäki-Patola

Telecommunications Software
and Multimedia Laboratory
Helsinki Univ. of Technology, Espoo, Finland
{pjhamala|tmakipat}@tml.hut.fi

Ville Pulkki, Matti Airas

Laboratory of Acoustics
and Audio Signal Processing
Helsinki Univ. of Technology, Espoo, Finland
{ville.pulkki|matti.airas}@hut.fi

ABSTRACT

Although voice has been used as an input modality in various user interfaces, there are no reports of using pitch of the user's voice for real-time control of computer games. This paper explores pitch-based control for novel games for musical education. Mapping pitch to the position of a game character provides visual feedback that helps you to learn to control your voice and sing in tune. As demonstrated by two example games in this paper, the approach can be applied to both single and two-player games even with just one microphone.

1. INTRODUCTION

Voice has been used as an input modality in various user interfaces. Speech recognition is nowadays used in games and other applications for wide audiences. The technology has matured so that simple recognition of a limited set of spoken commands can be implemented with little technological expertise using tools like the Microsoft Speech Application SDK [1].

An alternative to recognizing spoken commands is using sound pressure level, pitch, and other features of voice for immediate, real-time control. Igarashi and Hughes call this approach "Voice as Sound" [2] and give examples, such as a television volume control where the user says "Volume up, ahhh" and the volume continues to increase as long as the "ahhh" continues. Hämäläinen and Höysniemi have used the approach in a multimodal perceptual game user interface, where you control a dragon that flies when you flap your hands like wings and breathes fire when you shout [3]. Earlier, similar features of sound have mainly been used to control musical user interfaces. For example, in the Swim Swan performance in 1993, the pitch and loudness of a clarinet were used to control musical synthesis [4]. Speech input without actual speech recognition has been in used toys, such as a parrot that repeats words [5], for which it is necessary only to segment the input signal into words or phrases.

This paper describes two computer games that apply the "Voice as Sound" approach in the context of musical edutainment (education and entertainment). The main idea is to use the pitch of your voice to control a game so that you learn to control your voice and sing correctly with help of immediate graphical feedback, created by mapping pitch to the position of an avatar or other visual objects in the game.

Pitch has been defined as the characteristic of a sound that makes it sound high or low or determines its position on a scale [6]. Human perception of pitch is a complex process, but in case of singing, pitch corresponds to the frequency at which the vocal cords vibrate. However, the concept of pitch and the related musical vocabulary are not clear for many people. For example, if you instruct a person with no musical background to sing "higher", he or she may just continue to sing at the same pitch, but use a different vowel.

In the following, we first describe the two example games, shown in Figures 1 and 2. We then discuss the technological aspects and our experiences from testing the games.

2. THE HEDGEHOG GAME

The Hedgehog game in Figure 1 is a part of Soittopeli (PlaySingMusic), a musical edutainment CD-ROM released in Finland in 2000 by Elmorex Ltd. PlaySingMusic also features other games played with a MIDI keyboard, but they are beyond the scope of this paper. The game is aimed at 5 to 10 year old children.

The game is designed to provide a fun way of learning to sing. A song is played and the scenery scrolls from right to left in sync with the music. The player has to sing so that the hedgehog stays on the path that twists along the melody. As the vertical position of the hedgehog corresponds directly to pitch, the player gets immediate visual feedback that helps in finding the right pitch.

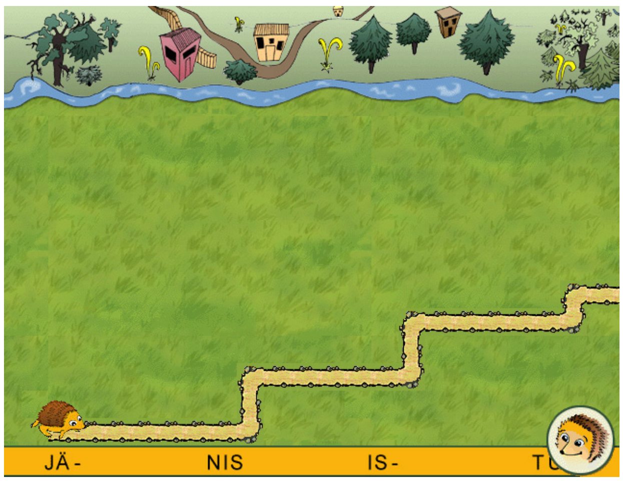

Figure 1: *A screenshot of the hedgehog game. The vertical position of the hedgehog is controlled by the pitch of the user's voice. You sing the song correctly if the hedgehog stays on the path twisting along the melody. The words of the song are shown at the bottom of the screen.*

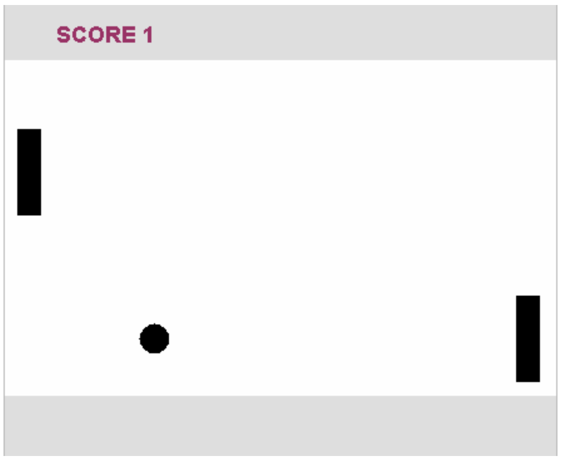

Figure 2: *A pitch-controlled Pong-style game. The bats on the left and on the right move vertically according to the pitch of the user's voice. The screenshot is from one-player mode, where the right bat is controlled by the computer.*

The feedback helps particularly when the player has some idea of going up and down with the melody, but the intervals and key are not right. The game grades the player's performance as the average distance from the correct notes and the player is awarded with different fanfares and prizes after the song. The expression of the hedgehog also changes according to the performance, shown in the bottom-right corner of the screen.

3. PITCH-CONTROLLED PONG

Pitch can be used to control practically any game with one-dimensional input. We created the pitch-controlled Pong game in Figure 2 to provide a simple game for learning the concept of pitch and to experiment with a two-player game. Pitch controls the vertically moving bats and the goal is to intercept a bouncing projectile with your bat, similar to the original Pong and other "paddle games" derived from it [7].

The history of computer games features several examples of multiplayer games played on a single computer. Generally speaking, you have to either use multiple control devices, or take turns using a shared control. Examples of the former are Tekken and other martial arts games played with several gamepads. Stewart et al. have also researched this kind of interfaces for collaborative work [8]. Having two microphones would make it easy to implement a two-player game, because the pitch detection usually outputs the pitch of the loudest voice. However, few users have the equipment needed to connect two microphones into the left and right input channels of a soundcard.

The main reason for choosing Pong as the starting point was that it combines both real-time and turn-based control so that it works using a single microphone. In two-player mode, pitch controls both bats so that they are always at the same vertical position. The players take turns in singing or humming so that once a player hits the ball, it's the other player's turn to react and sing his or her bat to intercept the ball. In one-player mode, the computer controls the other bat.

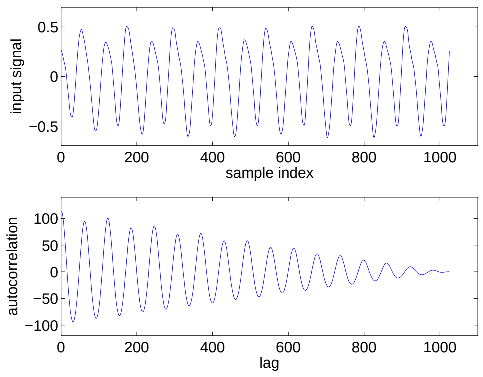

Figure 3: *A frame of the vowel 'u' and the corresponding autocorrelation. The strongest peak of the autocorrelation is located at the signal period of 124 samples.*

4. PITCH DETECTION TECHNOLOGY

Pitch detection (fundamental frequency estimation, *f0* estimation) has been researched widely since the 70's. It is perhaps most widely used as part of speech recognition and coding. Estimating pitch can be implemented as estimating the periodicity of voice, as seen in Figure 3. A periodic waveform repeats itself after the period *T*, the reciprocal of the fundamental frequency *f0*. In Figure 3, the period of 124 samples is clearly visible.

There are several algorithms for pitch detection. We have mainly experimented with autocorrelation based methods. The approach was found suitable for speech signals already in the 70's [9] and several improved variants of the approach have been developed, one of the latest being YIN by de Cheveigné and Kawahara [10], who report error rates three times lower than the best competing methods.

Comparing pitch detection algorithms is difficult, since they are often tested with different data. Recently, the Keele pitch detection reference database [11] has been gaining popularity. It has been used to evaluate YIN as well as methods by Wang and Seneff [12] and Tabrikian *et al.* [13] to name a few.

In autocorrelation based methods, sound input signal is divided into short frames and the period of a frame is estimated by computing its autocorrelation function (ACF), as shown in Figure 3. ACF estimates the similarity of a signal with the same signal delayed by some time lag. If the signal is periodic, the strongest peak of its ACF is usually located at the lag equal to the period. Note that the ACF computed from a frame of finite length is only an estimate, since the autocorrelation of a random process $x(k)$ is defined as the expectation $E[x(k)x^*(l)]$, where * denotes complex conjugate. For wide-sense stationary (WSS) processes, autocorrelation depends only on the lag $k-l$ [14].

Autocorrelation of a WSS process can also be defined as the inverse Fourier transform of the signal's power spectrum (squared magnitude spectrum). In practice, the fastest way to compute the ACF of a signal frame is to assume stationarity within the frame, compute an estimate of the power spectrum using FFT and then apply inverse FFT.

In a pitch-controlled application, you often need to know whether there is voice or just background noise or non-voiced phonemes, such as 's' or 't'. One suitable estimate for voicedness is the relative signal power at the signal period, estimated as the ratio of the ACF at the period and at zero lag [9].

4.1. Latency

For the game or interaction designer, it is important to note that no pitch detection method is perfect. The performance is affected by differences in people's voices, the quality of the microphones and soundcards, and interfering sounds from the environment. Figure 4 compares YIN to a basic autocorrelation based method that simply finds the maximum value of ACF within a range of suitable periods. YIN performs better, but the output still has some glitches. The errors can be reduced, for example, by using a median filter or analyzing longer signal frames, but in general, more reliable methods cause more delay in the detection and thus result in a less responsive user interface.

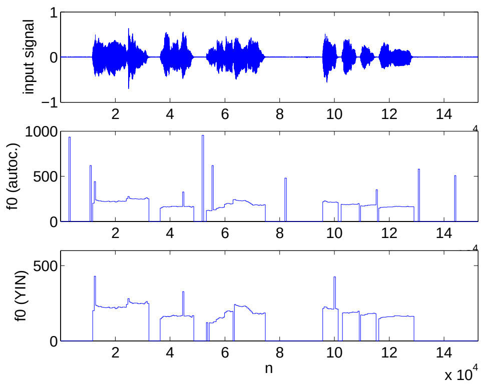

Figure 4: *Comparison of pitch detection using basic autocorrelation based method and YIN. The test signal is male voice with sampling rate 11025Hz and frame size 512 samples.*

There are many studies on the effect on latency on user interfaces. A classical experiment conducted by Michotte and reported by Card, Moran and Newell [15] shows that users perceive two events as connected by immediate causality if the delay between the events is less than 50ms. Dahl and Bresin [16] suggest that over 55ms of latency degrades use of a percussion instrument without tactile feedback while playing along with a metronome. Finney has shown that delay in auditory response caused large errors in performance of pianists [17]. A study by Sawchuk et al. [18] showed that latency tolerance is highly dependent on the piece of music and the instrumentation. Collaborative playing over a networked system was researched. A bit surprisingly the performers tolerated 100ms of latency on a piano sound but only 20ms on an accordion sound in the same piece.

Less than 10ms latencies are often suggested for musical controllers, as professional piano players might already notice that much [19][20]. However, the amount of tolerable latency depends on the music played, the instrument and the presence or absence of tactile feedback. For an extreme perspective, let us remind that latencies as high as several hundred milliseconds are not rare for church organs and yet they can be played when also practiced with the same latency.

In our games, the user and the computer form a control system where the user tries to adjust his or her voice according to both aural and visual feedback. In general, the more delay there is in the feedback loop, the slower the control and the more there's overshoots if pitch is adjusted rapidly. However, depending on experience and musical talent, the user does not rely entirely on feedback. When you sing, you are usually able to start a note roughly at the correct pitch, after which you make small adjustments according to the feedback. The main benefit of the visual feedback is that it helps you train your ear and singing.

Latency is not an issue in the Pong game, where the player has several seconds to search for the correct pitch. However, in the Hedgehog game, the tempos of the songs are limited by latency. Delays in the movements of the hedgehog are not disturbing as long as they are small compared to the lengths of the notes. There's no problem with most slow and medium-paced songs, but the visual feedback can start to feel ambiguous in fast songs, such as Polly Wolly Doodle.

The total delay between voice input and graphical feedback consists of the following components:

- The pitch detection algorithm. The frame size of pitch detection should be at least double the longest signal period to be detected. If the lowest note to be expected is a2 (55Hz), this results in minimum frame size of 36ms. If the sampling rate is 11025Hz, the minimum power of two for the frame size is 512 samples (46.4 ms).
- Video hardware and drivers. Games are usually double-buffered, meaning that graphics are first drawn to an off-screen surface that is made visible on the next screen refresh. This is essential to prevent tearing in the scrolling scenery of the hedgehog game. Depending on the timing of screen refreshes and incoming audio frames, double buffering can cause a delay of one refresh. At a typical refresh rate of 70Hz this yields 14ms.
- Audio hardware and drivers. MacMillan et al. report audio input-output latencies in the range of 60-120ms in the Windows operating system using DirectSound drivers commonly supported by consumer audio cards [21]. Windows with ASIO drivers, Linux with ALSA drivers, and MacOS X with CoreAudio can provide lower latencies, but form a lot smaller target audience. In our experience, most of the latency is due to the sound input part, which is logical since games and other consumer software typically only require low output latency for sound effects and music. Our games run on Windows computers and we use DirectSound for the sound input.

If we take the song Polly Wolly Doodle as an example, the shortest notes in the melody are eighths. If the tempo is 180 beats per minute, the duration of an eighth is 1/6 s. According to our experience, the latency should be at most half that, resulting in 83ms. Thus, even the delays of the audio and video hardware and drivers alone can be too much and this should be taken into account when selecting the songs. The pitch detection algorithm is typically not the major cause of latency in a consumer PC system, if no filtering is applied after the frame-based analysis.

4.2. Octave Errors and Transposing

As seen from Figure 4, octave errors are common in *f0* estimation, that is, ACF peaks at $2^N f0$ are selected, where N is an integer. This happens particularly with low quality microphones. Fortunately, octave errors are not an issue in our games, since the detected pitch is transposed in octave steps into a suitable range.

In the hedgehog game, different players may want to sing the notes from different octaves so we first designed the game so that the visible tone scale equals one octave and the pitch is simply transposed inside that octave. People sing at different octaves and specifying the octave before the game would make the user interface more complex. Most of the songs children sing in elementary schools in Finland have tone scales less than one octave.

However, the simple transposing breaks the visual feedback system of the game, since the mapping from the pitch to the vertical position is many-to-one instead of one-to-one. For example, if the desired note is at the bottom of the screen, singing only a little below it causes the hedgehog swing to the top of the screen, that is, above the path. Another drawback of the simple transposing is that small fluctuations about the octave border result in the hedgehog appearing randomly at the top and bottom of the screen.

Fortunately, the problems can be solved by transposing the pitch into the octave closest to the desired note and then clipping the corresponding vertical position into the visible range. This way, the Hedgehog appears in the direction where it is closer to the path so that the feedback is correct in errors smaller than a half octave. Singing below a note at the bottom of the screen presses the hedgehog against the bottom so that you know to correct the pitch upwards. The improved transposing also allows wider tone scales than one octave.

The vertical range of the Pong game also equals one octave and pitch is transposed inside it. The transposing is required to allow people with differently pitched voices to play against each other.

5. DISCUSSION

During development, the hedgehog game was tested informally with several adults, 6 third-graders (9 to 10-year-olds), 3 preschoolers and 2 14-year-olds. Preschoolers seemed to need adult guidance in using the game, but are able to play using songs they already know, such as Mary had a little lamb. The third-graders all played the game successfully, but two non-musical children needed 4 to 5 times practicing until they figured out the mapping from their voice to the movement. It helped when we instructed them that the loudness of the voice has no effect and that "you should make your voice go up and down like the siren of a firetruck", giving a concrete example of what 'up' and 'down' mean.

In general, the game works best if the player already knows the song. Learning a new song based on the visual feedback is difficult and requires a really slow tempo or long notes so that you have time to find the notes.

The pitch-controlled Pong has been tested informally in several parties. The game can also be played in a web browser at the address `http://www.elmorex.com/products/pong.html`. Because the loudest voice dominates, it is easy to play dirty and disturb your opponent, but the same applies also to other turn-based games played with a shared input device. Engaging in a game usually means that you agree on the rules, but on the other hand, a shouting contest is often as entertaining as fair play.

The game seems to satisfy the goal of teaching the concept of pitch, but people sometimes need instruction despite the seemingly simple control. Since the game does not have a musical background to sing to, you learn the tone scale of the display only by experimenting. We found out that trying to make your voice to go higher without a clear target can result in problematic results. Instead of pitch, some people only change the vowel they are voicing, for example, from 'a' to 'e', which can be explained in terms of spectral changes that contribute to the perception of pitch [6]. Occasionally, people have made their voices go up a full octave, effectively making the bat stay at the same location because of the octave transposing. One solution for this is to lowpass filter the pitch before the transposing so that the bat can be seen going upwards and wrapping back down, but this makes octave errors visible and increases latency.

It can be questioned whether games controlled by singing might face the same doom as brainwave sensors and other interaction devices found uncomfortable and tedious in prolonged game use, as described by Crawford [22]. We agree with Crawford in that developing a new control method for an existing game is usually not a very fruitful approach, especially if you could play better with a traditional controller. The Pong game is an example of this and although described fun and comical by the users, it loses its appeal quickly. On the other hand, this can be attributed to the overall simplicity of the game and the new interface can be considered successful for reviving the game for party use.

In general, the control mechanism should not be viewed as a separate entity, but as a part of the user experience and gameplay. In the hedgehog game, the interface makes gaming a spectator sport – you can show off your musical skills much the same way as when singing karaoke or performing on stage in a concert. This way, it is related to the dancing games that first gained popularity in Asia and have become a hit also in the western world thanks to recent PlayStation 2 and X-Box titles, such as Dancing Stage Fever. People dance and sing for fun also without computers, and the games add a possibility for improving your skills and competing based on quantitative feedback.

Staraoke, a TV series based on a remake of the hedgehog game was broadcasted in Finland in 2003. In the show, schoolchildren competed in the game. Staraoke was the fourth most popular program on Finland's commercial television channels among 4 to 9 year old children [23]. In an upcoming version, you can apply for the show by playing the game at home and then uploading your recorded performance to the show's website.

Lately, interactive karaoke games in the style of the hedgehog game have been becoming a new game genre of their own. Time Magazine ranked Konami Karaoke Revolution as the best game of 2003 [24]. Instead of a story-based avatar and graphics, the game uses an arrow as a pitch indicator and horizontal bars as notes, similar to the piano roll views of many sequencers. In May 2004, Sony Computer Entertainment Europe released a rival karaoke game called SingStar, also with a piano-roll view.

6. CONCLUSIONS AND FUTURE WORK

We have introduced a novel approach to musical edutainment based on using pitch of the user's voice as a game control mechanism. The games can help you to learn to sing and to control the pitch of your voice. In the future, we will carry out more rigorous testing to inspect the educational effects of the games. For example, it can be questioned whether the learning is only contextual, since the player can learn to rely on game graphics instead of hearing.

The pitch detection sometimes reacts to the background music so that in the hedgehog game, the music controls the hedgehog when the user is not singing. This also prevents adding a reference melody for learning new songs, since the reference could make the hedgehog stay on path even if the user does not sing at all. There is no problem when using headphones or low output volume, but a general solution would be to use acoustic echo cancellation (AEC) methods to remove the music from the input signal. According to preliminary tests, basic approaches, such as LMS adaptive filters do not completely cancel the music and remove the errors. Echo cancellation methods often assume that there is not much double-talk (simultaneous output and input activity) and that the output and the desired input are not correlated. These assumptions do not hold in musical games, especially if the musical background includes melody and there are only few pauses. On the other hand, echo cancellation can be optimized because the games only need the pitch information and the processed sound does not have to be output for listening.

Considering other games that could be controlled by singing, it should be noted that pitch can also be mapped to other variables than vertical position. Using pitch together with loudness or other features could also enable control in two or more dimensions. This kind of interface could be used to teach dynamics and articulation in addition to singing in tune.

Increasing the dimensionality of control complicates the task of displaying the desired input in a predictable way. In the case of the hedgehog game, the basic method is to use an N+1 –dimensional display for an N-dimensional input signal, using the extra dimension for time. However, we are also researching other methods.

7. ACKNOWLEDGEMENTS

This work has been partly supported by HeCSE graduate school of Academy of Finland. We also want to acknowledge the contributions of our colleagues at Elmorex, especially Jyrki Kohonen (CEO), Niina Saarelainen (music production), Tero Tolonen (audio technology), Erno Soinila (programming), and Samuli Siltanen (idea of the Hedgehog game with Ville Pulkki).

8. REFERENCES

[1] Microsoft Speech Application SDK documentation, `http://msdn.microsoft.com/library/default.asp?url=/library/en-us/getstarted_beta2/html/SDK_GetStarted.asp`, Oct. 6, 2003.

[2] T. Igarashi, J. F. Hughes, "Voice as Sound: using non-verbal voice input for interactive control," *Proc. Symp. on User Interface Software and Technology (UIST'01)*, ACM Press, 2001.

[3] P. Hämäläinen, J. Höysniemi, "A Computer Vision and Hearing Based User Interface for a Computer Game for Children," *Proc. 7th ERCIM Workshop "User Interfaces For All,"* France, Oct. 2002.

[4] K. Furukawa, P. Dutilleux, "Live-electronics Algorithms in the Multimedia Work 'Swim Swan'," *Proc. Int. Conf. on Digital Audio Effects (DAFx-02)*, Germany, Sept. 2002.

[5] T. Selker, "New Paradigms for Using Computers," *Communications of the ACM*, Aug. 1996, vol. 39, no. 8.

[6] T. D. Rossing, *Science of Sound*, 2nd ed., Addison-Wesley, 1990.

[7] C. Crawford, *The Art of Computer Game Design*. Osborne/McGraw-Hill, 1982. Available at `http://www.mindsim.com/MindSim/Corporate/artCGD.pdf`

[8] J. Stewart, B. B. Bederson, A. Druin, "Single Display Groupware: A Model for Co-present Collaboration," *Proc. CHI'99,* ACM Press, 1999

[9] L. R. Rabiner, M. J. Cheng, A. E. Rosenberg, and C. A. McGonegal, "A Comparative Performance Study of Several Pitch Detection Algorithms," *IEEE Trans. on Acoustics, Speech, and Signal Proc.*, vol. 24, no. 5, Oct. 1976

[10] A. de Cheveigné, H. Kawahara, "YIN, a fundamental frequency estimator for speech and music," *J. Acoust. Soc. Am.*, vol. 111, no. 4, April 2002.

[11] F. Plante, G. Meyer, W. A. Ainsworth, "A pitch extraction reference database," *Proc. EUROSPEECH'95*, Madrid, Spain, 1995, pp. 827–840.

[12] C. Wang, S. Seneff, "Robust Pitch Tracking for Prosodic Modeling in Telephone Speech," *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Proc. (ICASSP '00)*, vol. 3, 2000, pp.1343–1346.

[13] J. Tabrikian, S. Dubnov, Y. Dickalov, "Maximum *A-Posteriori* Probability Pitch Tracking in Noisy Environments Using Harmonic Model," *IEEE Trans. on Speech and Audio Proc.,* vol. 12, no. 1, Jan 2004.

[14] M. S. Hayes, *Statistical Digital Signal Processing and Modeling*. John Wiley & Sons, 1996.

[15] S. Card, T. Moran, A. Newell, *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates, 1983.

[16] S. Dahl, R. Bresin, "Is the Player More Influenced by the Auditory Than the Tactile Feedback from the Instrument?" *Proc. Int. Conf. on Digital Audio Effects (DAFx-01)*, 2001.

[17] S. A. Finney, "Auditory Feedback and Musical Keyboard Performance," *Music Perception,* vol. 15, no. 2, pp. 153–174, 1997.

[18] A. A. Sawchuk, E. Chew, R. Zimmermann, C. Papadopoulos, C. Kyriakakis, "From Remote Media Immersion to Distributed Immersive Performance," *Proc. 2003 ACM SIGMM Workshop on Experiential Telepresence*, 2003.

[19] A. Freed, A. Chaudhary, B. Davila, "Operating Systems Latency Measurement and Analysis for Sound Synthesis and Processing Applications," *Proc. Int. Computer Music Conf. (ICMC 1997),* 1997.

[20] J. Wright, E. Brandt, "System-Level MIDI Performance Testing," *Proc. Int. Computer Music Conf. (ICMC 2001),* 2001.

[21] K. MacMillan, M. Droettboom, I. Fujinaga, "Audio Latency Measurements of Desktop Operating Systems," *Proc. Int. Computer Music Conf. (ICMC 2001)*, Cuba, Sept. 2001.

[22] C. Crawford, *Chris Crawford on Game Design*. New Riders Publishing, 2003.

[23] MTV 3 Oy, `http://spotti.mtv3.fi/jutut/liitteet/116.pdf`, June 29, 2004.

[24] Time Top 10 Video Games 2003, `http://www.time.com/time/techtime/200311/videogames/karaoke.html`, June 28, 2004.