AN ARTISTIC TECHNIQUE FOR AUDIO-TO-VIDEO TRANSLATION ON A MUSIC PERCEPTION STUDY

Eugene Mikyung Kim

Department of Music Technology, Korea National University of Arts
eugene@u.northwestern.edu

ABSTRACT

The paper presents an audio-to-visual instrument that allows sound-to-image transformation based on an empirical investigation of the relationship between four auditory parameters – pitch, amplitude, timbre, and duration - and four visual parameters – color, location, shape, and size - in the multimedia context. Implementing the audio-to-visual instruments involves real-time sound analysis by using a constant-Q transform and image generation in a Max/MSP/Jitter environment.

KEYWORDS

Audio-to-video Interface, Constant-Q Transform, Visualization, Real-time Sound Analysis, Image Generation, Algorithm

1. INTRODUCTION

The idea of the unity of audition and vision has been a topic of interest since the time of ancient Greece. Investigation of this idea has continued to grow in many areas, especially in psychology, music, visual art, and computer science, most notably since the invention of the world's earliest color organ. Recent developments of digital computing systems have made the concept more tangible. Currently, many audio-to-visual performance software applications are available, but they tend to depend on an arbitrary or personal association rather than perceptually significant information in transforming auditory properties into visual characteristics. Thus this paper attempts to create audio-to-visual instruments based on the results of a previous empirical study on perceived match between auditory and visual parameters [1].

2. AN EMPIRICAL STUDY ON THE RELATIONSHIP BETWEEN AUDIO AND VISUAL PARAMETERS

The previous experiment involved matching estimation of four auditory elements – pitch, amplitude, timbre, and duration – and four visual aspects – color, location, shape, and size. The study was based on a primary research question: When changes in auditory and visual correlates are presented together, are some pairings always perceived as a better match than other combinations? If so, which specific visual attribute is "best match" for each of the selected auditory aspects?

The experimental results revealed that there are, in fact, differences in the degree of "match" perceived by subjects, depending on the audio and visual components of an A-V composite, as illustrated in Figure 1.

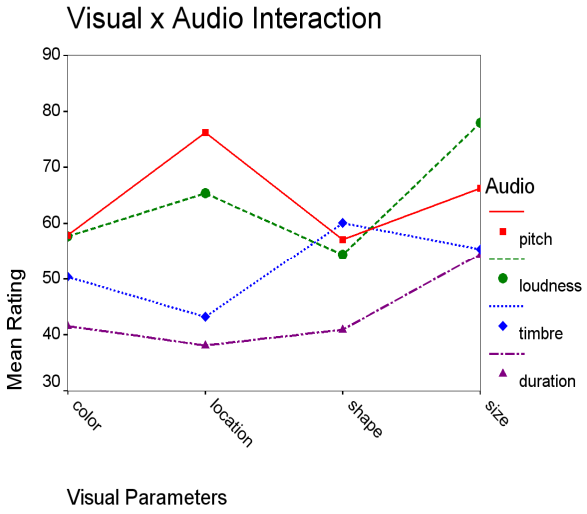

Figure 1. Diagram of subject mean responses

The highest ratings of perceived match suggested the following pairings: pitch-location, loudness-size, and timbre-shape. Also indicated is the presence of non-unitary relationships. For instance, the visual quality of color matched equally well with both pitch and loudness in the auditory domain. It is worthy to note that duration did not pair as "best match" with any visual element. Finally, as well as the relationship of both pitch and loudness to color mentioned before, there are several cases in which secondary relationships propose that the primary relationships cited previously do not present a singular appropriate "matched" combination. Therefore, although the primary combination obtains a higher mean score, secondary relationships such as pitch-size and loudness-location may supply sources of variations that can be incorporated into the audio-to-visual algorithm. Moreover, the auditory aspects of timbre, pitch, and loudness may all provide an acceptable matched pairing for the visual element of shape.

3. IMPLEMENTING AUDIO-TO-VIDEO INTERFACES

Figure 2 depicts how to extract the four audio elements - pitch, amplitude, timbre, and duration - and how to match them to the visual parameters to generate moving images in the Max/MSP/Jitter environment in practice.

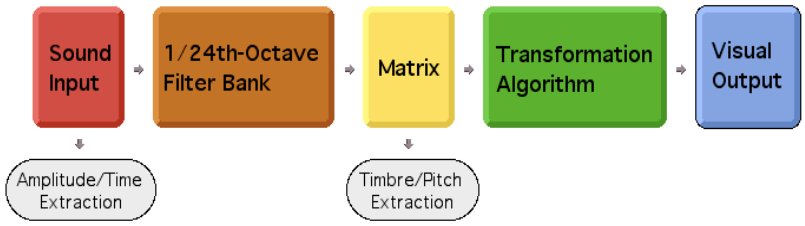

Figure 2. Sequential scheme for an implementing audio-to-visual interface

3.1. Amplitude and Time Extraction

Amplitude is fairly easy to extract with MSP objects. These include *number~*, *meter~*, *average~*, and *avg~*. The number~ produces the instantaneous amplitude of the signal, whereas the *meter~* shows the peak amplitude it has received since the last display. However, the averaged amplitude is usually used to match the perception of human beings. Both the *average~* and the *avg~* objects

generate the mean sample value over a brief period, but the latter is easier to use in image synchronization because it outputs a "float" rather than a "signal" when "banged".

The update interval should be set to fit the input signal because a slight difference in the averaging time may cause different values. As a result, it affects detecting variations in the amplitude. A different amplitude value activates a different event in visual response. In general, a shorter update time interval outputs a more accurate averaged value, but an expeditious updating causes the visual output to blink [2].

Figure 3 illustrates a simple implementation of amplitude and time detection. The update rate is determined by the argument (in ms) of the *metro* object. A *number* object in the main patcher is connected to the metro object to adjust the rate.

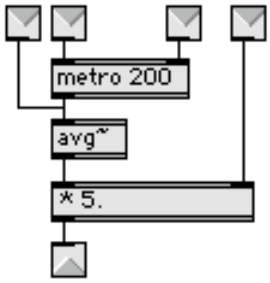

Figure 3. Amplitude and time extraction

3.2. Constant-Q Filter Bank Analysis

A constant-Q filter bank has considerable benefits for the analysis of musical signals because the center frequency of each filter can be set to that of the equal tempered musical tones, whose frequencies are logarithmically spaced, as opposed to the linear interval that is produced in a FFT. Moreover, a constant-Q technique makes timbre identification much easier than an FFT. In constant-Q algorithms, the harmonics of musical sounds played with the same musical instruments form a constant pattern in the frequency domain. The absolute positions of the harmonic frequency components differ in accordance with the fundamental frequency, but the relative positions are fixed. Thus, the pattern differences of the spectral components manifest the timbre differences of the sounds analyzed [3] [4] [5].

The constant Q transform in the present work is the same as a 1/24th-octave filter bank with center frequencies between 175 Hz (F3, MIDI note 53), a frequency just below that of the G string (196 Hz) of a violin, and 13,432 Hz (MIDI note 128), selected to be below the Nyquist frequency with a sampling rate of 44.1 KHz. The method supplies exact frequency information corresponding to quartertone distance of the equal tempered scale that is sufficient to distinguish adjacent musical notes. Furthermore, it yields a constant pattern with harmonic frequency components for timbre detection.

The preference of quartertone spacing leads to a total of 150 channels in order to cover the whole frequency range. An *fffb~* object can have up to only 50 bands, and three of the *fffb~*, each starting at 175 Hz, 762 Hz, and 3,232 Hz, are required.

3.3. Timbre and Pitch Extraction

The lists of the amplitude values generated from the constant-Q filter are filled in the matrix bands with the *jit.fill* objects, illustrated in Figure 4.

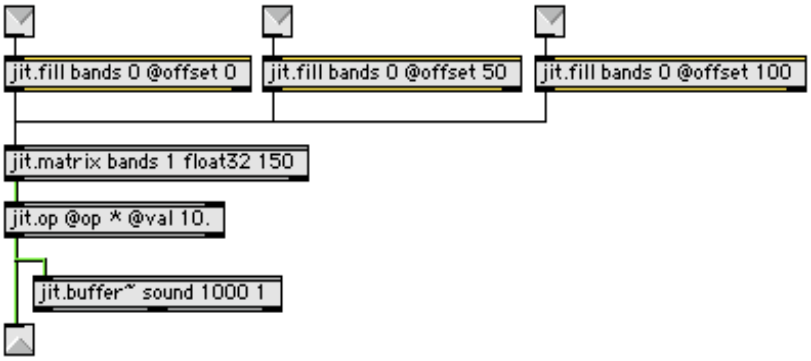

Figure 4. Storing the amplitude values into matrices

An individual cell address of the matrix identifies the index number of the center frequency of a filter from 0 to 149. To retrieve each amplitude value and its frequency number simultaneously, the data are also passed into a *jit.buffer~* object.

Timbre information – the combination of frequencies and their amplitudes - is already obtained with the constant-Q filter bank analysis and stored in the buffer. By employing a *uzi* and a *peek~* object, the spectrum data are easily taken out from the buffer and drawn with a jitter object, displayed in Figure 5.

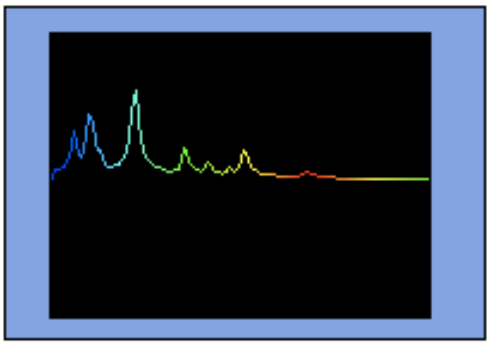

Figure 5. A waveform generated with a *jit.lcd* object

The waveform in Figure 5 shows the strengths of the various frequencies contained in the signal. The goal of the pitch extraction is to find peaks to isolate the dominant frequencies of the spectrum as shown in Figure 6. Such pitch recognition can be done with a *jit.iter* object.

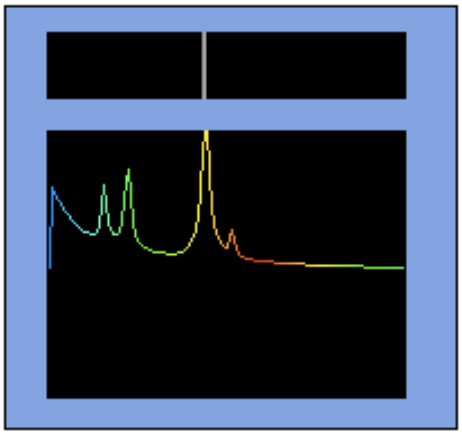

Figure 6. Pitch extraction from a waveform

3.4. An Artistic Transformation Algorithm

Now, by modifying the method of generating the x and y coordinates, it is possible to create other shapes than a simple waveform. In the following paragraphs,, the creation of polar roses with a waveform generated depending on the audio parameters and the software simulation of a video feedback will be discussed.

In Max, the *poltocar~* object requires a radius in pixels and an angle in radians. The angles are expressed as $2\pi /k$, where the k may be provided by the value of one of the selected audio parameters. The radius can be expressed as $a \cos(n\emptyset) * \cos(n\emptyset)$. The length of the petals of the rose is determined by the variable a, which can be supplied by the audio signal. Figure 7 shows a simple pattern of the polar roses. Increasing the n makes the pattern rotate in a counter-clockwise direction and have more petals. When the n is an integer, a more complex pattern is generated.

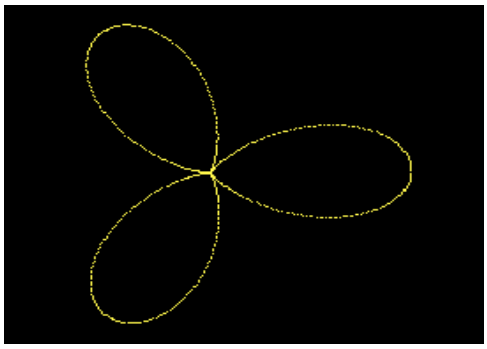

Figure 7. A simple pattern of the polar roses

It is possible to make the amplitude control the size of the rose and the waveform control the fluctuation of the pattern as illustrated in Figure 8.

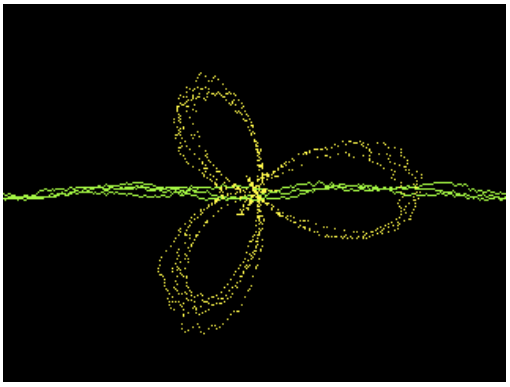

Figure 8. Polar roses controlled by audio parameters

This pattern is possibly rotated, spatially magnified, and tinted in a video feedback system. Video feedback is the procedure of pointing a camera at a monitor that is showing the output of the camera. The camera transforms visual information on the monitor into an electronic signal that is then transformed by the monitor into an image on its screen. The image is then electronically transformed and displayed on the monitor. This dynamical flow of information creates an endless looping of the information, which results in interesting patterns.

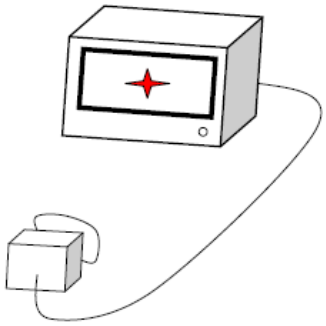

Figure 9. Video feedback setup

The patterns depend on the parameters of a video feedback system. Although there are many potential controls which have an impact on what will be displayed on the monitor screen, in the most typical video feedback system there are only a few controls: zoom, focus, and rotation for the camera, and brightness, contrast, and color for the monitor, as well as the position of the camera with respect to the monitor [6] [7] [8].

Figure 10 demonstrates the simulation of a video feedback system in Jitter. The data stored in the *videofeedback* matrix represent the old image stored in a camera. The image is zoomed and rotated by the *jit.rota* object and displayed in the *jit.window* object that functions like a monitor screen. The *jit.rota* object zooms and rotates the image based on its attribute settings.

The color values of the zoomed and rotated image are then modified with the *jit.op* object. The *jit.op* object operates on two matrices or the left input matrix. It takes 4 plane char values ranging from 0 to 255 and converts them to floating-point values. It also interprets each plane as alpha, red, green, and blue channels. A different operator may be applied for each plane of the incoming matrix with the *val* attribute. If the only one value is set, the *jit.op* object uses it for all planes. If multiple values are specified, the *jit.op* object applies them for each plane in order. Various combinations of the jit.op's operators produce various image effects.

Figure 10. Simulation of video feedback

Afterwards, the image data are modified by the *jit.wake* object that performs a temporal feedback, as well as a spatial convolution to the matrix, producing a variety of motion and spatial blur effects.

Finally, the image data are altered by the *jit.hue* object performs a hue rotation without changing the luminance values. The hue rotation is stated in degrees.

In the video feedback systems, different initial conditions usually result in different patterns. In addition, slightly disturbing the system by continuously changing the image of the virtual camera introduces complex and striking imagery.

This system supports not only the slow change of spatial and temporal dynamics, but also the synchronization of the input sound and its visual representation possible. Figure 11 shows some examples of the video feedback system with polar roses.

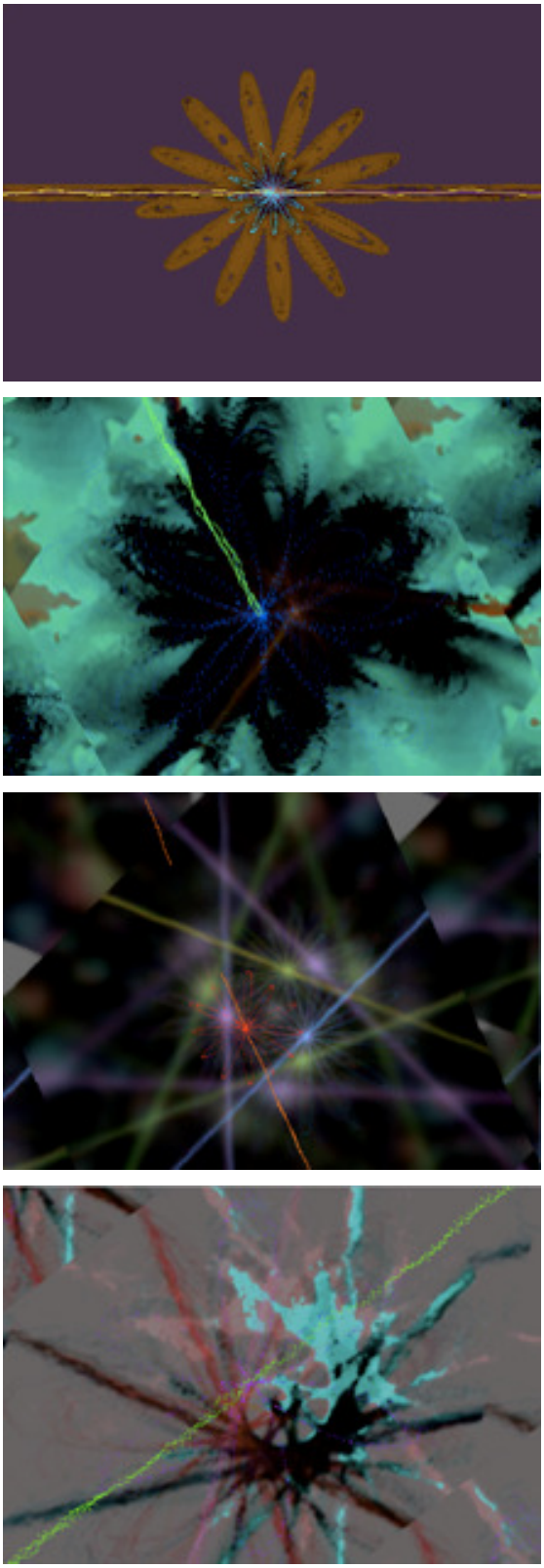

Figure 11. some examples of the video feedback system

ACKNOWLEDGEMENTS

I would like to thank everyone, just everyone!

REFERENCES

[1] Lipscomb, S.D. & Kim, E. M. (2004) "Perceived match between visual parameters and auditory correlates: An experimental multimedia investigation", *The 8th International Conference on Music Perception and Cognition.*

[2] Elsea, P. (2004) *Visual audio* [Electronic Version], ftp://arts.ucsc.edu/pub/ems/.

[3] Brown, J.C. (1991) "Calculation of a constant Q spectral transform", *Journal of the Acoustical Society of America*, 89(1), pp425-434.

[4] Brown, J. C., & Puckett, M. S. (1992) "An efficient algorithm for the calculation of a constant Q transform", *Journal of the Acoustical Society of America, 92*(5), pp2698-2701.

[5] FitzGeral, D., Cranitch, M., & Cychowski, M. T. (2006) "Towards and Inverse Constant Q Transform", Paper presented at the Audio Engineering Society 120th Convention.

[6] Crutchfield, J. P. (1984) "Space-time dynamics in video feedback", *Physica*, pp191 - 207.

[7] Edwards, K. D., Finnewy, C. E. A., Nguyen, K., & Daw, C. S. (2000) *Application of nonlinear feedback control to enhanced the performance of a pulsed combustor* [Electronic Version], http://www-chaos.engr.utk.edu/pap/crg-cssci2000tpc.pdf.

[8] Essevaz-Roulet, B., Petitjeans, P., Rosen, M., & Wesfreid, J. E. (2000), " Farey sequences of spatiotemporal patterns in video feedback", *The American Physical Sociery, 61*(4), pp3743 -3749.

Authors

Instructor, Department of Music Technology at Korea National University of Arts

Doctor of Philosophy *in Music Technology* at Northwestern University