Published in *Proceedings of the 1997 C onnecticut College Symposium on Arts and Technology*

Creating Interactive Dance with the Very Nervous System

Todd Winkler
Brown University

ABSTRACT

Since the invention of the Theremin, composers and choreographers imaginations have been piqued by the possibility of turning dancer's movements into sound. One of the more recent inventions, David Rokeby's *Very Nervous System*, offers a sophisticated level of computer control for detecting accurate location and movement information via video camera. The computer then interprets this movement data, mapping it to sounds and other musical parameters.

This paper will discuss the artistic and technical processes used to create a new work for dance, an ongoing collaboration between choreographer Walter Ferrero and composer Todd Winkler. The dance space is divided into various regions that report presence, stillness, and speed of motion. The size and function of these regions may be reconfigured during a performance. The various regions may act in one of two ways: they may use the body as a musical instrument by responding immediately to produce a sound; and they may be used to start, stop, or influence larger musical processes. Our working method seeks to identify music and sounds that seem appropriate and responsive to particular movements.

Background

Computer music researchers have grappled for many years with the primary artistic and technical problem of how humans can interact and perform with computers. My primary interest as a composer has been to use interactive computer technology to enhance a performer's range of expression, by writing computer music and software that creates a believable and seamless response to musical gestures. More recently, I have been interested in expanding the notion of interactive performance to include dance movements as the primary input to a computer music system, creating music and sound that is immediately responsive to a dancer's location and physical gestures.

I began working with choreographer Walter Ferrero in the summer of 1996 on a solo work for interactive dance. For motion analysis, I used the "Very Nervous System" (VNS) a motion sensing computer input device that provides information about location and speed of motion. The VNS was invented by David Rokby who has used motion tracking systems in his installations since 1983 (Rokby, 1997). I started out by linking the VNS to a large program that I had previously written for interactive music, called *FollowPlay* (Winkler, 1992). Some of the techniques used for interactive musical performances translated well into the motion sensing world, since the primary information sent by most MIDI controllers, such as a keyboard, includes location (fingers located on a particular key) and speed (finger velocity, usually controlling dynamics). However, instrumental paradigms were often inadequate to produce music spawned by human movement. Thus, the most exciting aspects of the project were discovering new music and techniques that responded in an interesting way to specific movements, making the resultant music appear as "physical" as the dance (for a more theoretical discussion, see my paper, "Making Motion Musical," published in the Proceedings of the 1995 Computer Music Conference).

At the onset, the project was divided into four phases: information gathering through experimentation, testing, and improvisation; selecting successful improvisations for further development and exploration; structuring and refining the various sections of the work leading to a work-in-progress critique; and finalizing the work for performance. At the time of writing this paper, we have completed the first two phases of the project. A premiere date is set for April 1997.

The Collaborative Process

Our initial studies had two purposes: to discover the capabilities, limitations and idiosyncrasies of the VNS, and to explore a wide variety of ideas regarding combinations of movement and sound. Rather than approach the project with a preconceived notion of a type of music that the dancer would "play," we began with several months of improvisation so that our artistic decisions would naturally evolve out of a spontaneous, physical understanding of the system. We hoped to become familiar with the space to the point of playing on an intuitive level. Improvisational software, designed to produce music with unpredictable results, heightened the quality of interaction.

The VNS proved to be accurate, dependable, and surprisingly stable. In each session we explored various camera angles, camera lenses, grid systems, movements, and sounds. Sometimes props were brought in to trigger the VNS, such as scarves, balls, and sticks. Software was often written on the spot or modified in real time to test variations. This was a time of play and experimentation, with ideas tested out at a rapid rate. In this way, we enjoyed a collaborative process where we discovered specific movements and sounds that seemed especially responsive to and appropriate for the system.

Each improvisational session was videotaped and the various sections were "slated" and indexed to the software used that day. After many tapes had accumulated, we sat down and edited them, looking for particularly striking moments. We were able to clearly identify several distinct ideas that reappeared throughout the improvisations, and edited the master copies down to four tapes based on the system setup, mood, movement, and sound. The four sections we identified were as follows: machine movements (robotic or repetitive movements with each part of the body triggering a single percussive or machine sound, with complex rhythms generated by cyclical body movements); slow, fluid movements producing long evolving sounds, with the system responding often to overall motion; live mixing with various locations used to start and stop different tracks of a multitrack sequence; and theatrical movements characterized by humor and clowning using single triggers to toggle selected processes on and off.

Our current stage involves relearning these video-taped segments, and putting together sequences of events, often taken from several different improvisational sessions. These reconstructed sections will be further explored and refined.

We also investigated ideas of performer control. From a theatrical point of view, we played with the idea of the performer appearing to be confidently in control of the space, playing each sound with great skill; versus the performer acting machine-like, with movement and sounds that seemed involuntary. We also experimented with the amount that the performer actually controlled the computer system. Since our configuration of the VNS imposed certain limitations on the dancer, taking control away from the performer at times allowed for freer overall movement.

Technical Background Regarding the Very Nervous System
(see http://www.interlog.com/~drokeby/vnsII.html)

A basic understanding of the VNS is required to understand the project. The VNS is a computer SCSI device designed to analyze movement within a space using one or two video cameras (the original VNS is a single camera system, the VNSII can take one or two camera inputs). Anything moving within a camera's view can be analyzed. The video image can be mapped onto a user definable grid, with each square of the grid an active "region." The amount of motion is analyzed for each region, as well as a total for the entire video field. A region may also be irregularly shaped, the size and shape defined with simple drawing tools that come with the VNS software. Up to 240 regions may be active at a time, although in practice, fewer regions are often more practical and very effective. Numerous parameters make the system adjustable for specific locations and projects.

The VNS doesn't actually measure motion, it measures changes in light. By comparing the light in one video frame to previous frames, it determines what part of the video image has changed, and by how much. The device analyzes black and white images (color video is converted to black and white) and the gray-scale resolution is 6 bits (64 shades of gray). Each region is defined by a group of pixels, and the total of the gray-scale values for all of the pixels in a region are compared frame to frame. The image resolution is 128 (horizontal) by 240 (vertical) per field.
The VNS has two basic modes of analysis: motion and presence. Simply put, motion analysis compares the current frame only to the most recent previous frames, representing continuous motion. Presence analysis compares the current frame to a video image captured sometime in the past (for instance, comparing a current image to a stored image of a blank stage). For the purposes of this project, only motion analysis was used, and further discussions will refer only to that mode of analysis.

If nothing changes in a region from one frame to the next the VNS reports a value of zero. Faster movements within a region will usually produce higher values than slower movements, since there are more changes in light values per frame. Movement, however, is not the only determinant of reported values. Background color, clothing color, lighting, and proximity to the camera all have an impact on the analysis. For example, all other things being equal, if the VNS analyzed the movement of a person wearing white in front of a black curtain, it would result in higher values than a person wearing black moving in front of the same curtain, since there would be greater changes in light per frame. Also, if a bright light was suddenly turned on in an empty room, all active regions would temporarily report tremendous "activity" with the light levels changing dramatically from one frame to the next. Since the VNS is constantly updating itself (most often at 30 frames per second) these high values would quickly reset back to zero when the background image became steady again. As for proximity, moving closer to a camera usually result in higher values, since the body occupies more of a region and even small movements cause large changes in the image. In this example, something as trivial the color or pattern of a shirt could alter the overall reading of the movements. These issues become especially important considerations in designing sets, costumes, and lighting for dance.

Software Design

Data from the VNS streams into the Max programming language via a custom software object (Max is a graphical programming language available from Opcode, Inc.). I created software to handle further analysis and interpretation of this data, music generation, and a user interface to facilitate rehearsals and performances (figure 1). The raw values from the VNS object are displayed graphically to represent the grid. These values are then scaled, mapped, or otherwise prepared to enter the system's response modules. The response modules are a collection of self-contained programs that are designed to produce music based on location triggers and continuous motion. Any number of these modules may be active at a given time using on/off toggle switches. They range from very simple data structures designed to map regions to specific MIDI note numbers, to highly complex algorithms, such as using overall movement to continuously change parameters representing tempo, register, and timbre.

A master "preset" object is used to automatically store and recall all possible changes in the system duringrehearsals and performances. All of these changes are available in real time through on-screen graphics, or may be automated in response to particular movements. Specialized modules are all linked to the master "preset" object. Thus, going from one section to the next entails calling up the next preset, with each section having its own behavior and response. This strategy was especially helpful in our initial working sessions, since everything from the selection of sounds to the configuration of the VNS could be quickly stored and recalled at a later time (figure 2).

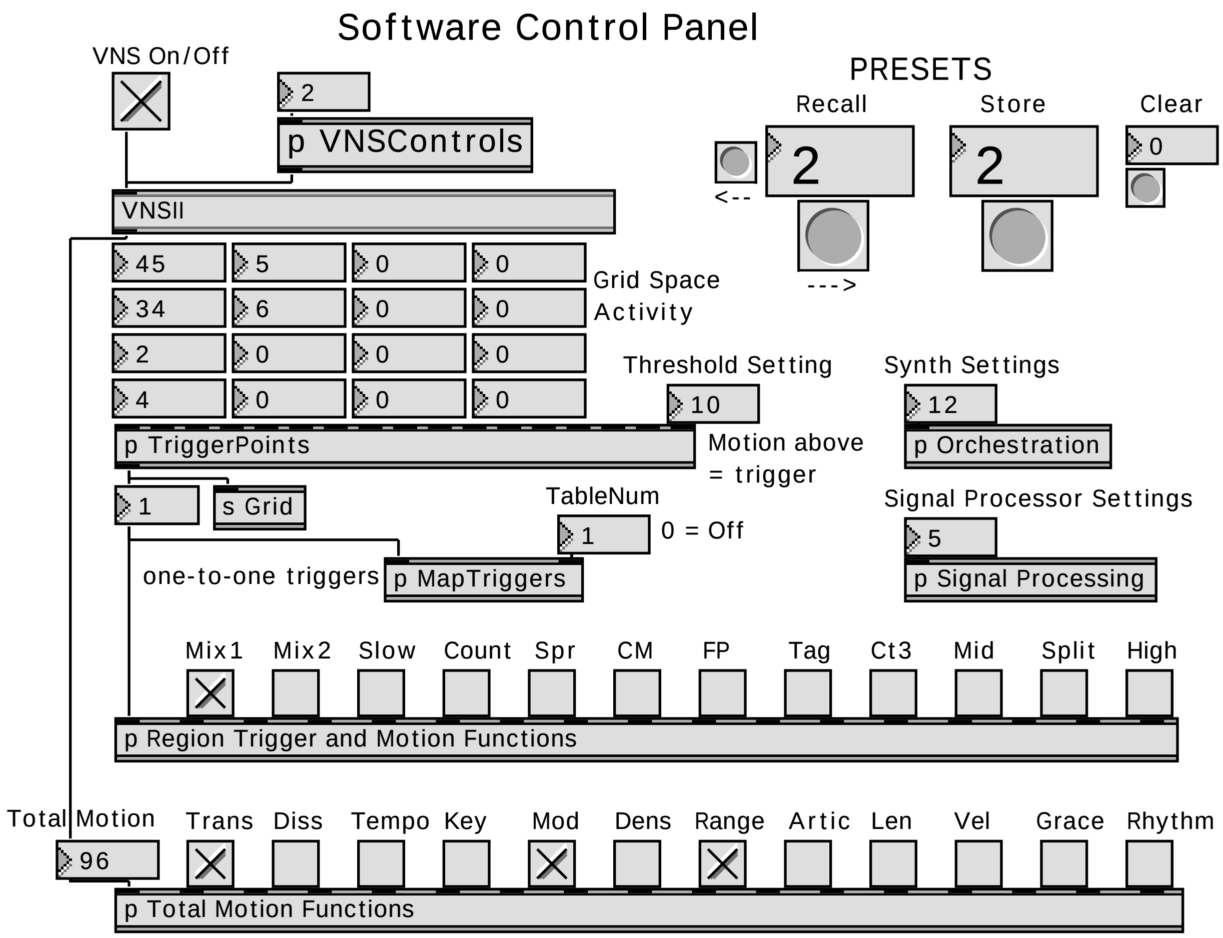

Figure 1. Software interface showing activity in the top-left area of the grid.

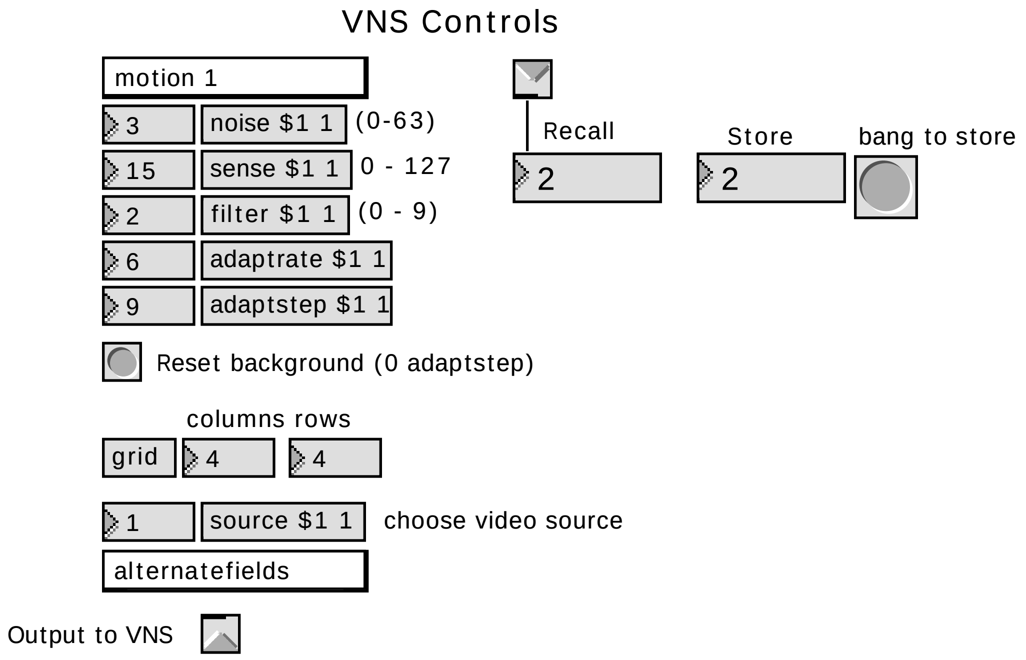

Figure 2. VNS Controls Module with a link to the Master Preset

Technical Specifications of the Project

Since the VNS has numerous parameters which make it highly flexible in its analysis of movement, we decide to settle on a single configuration, so that it could be learned as spatial "instrument." After trying out various configurations, we eventually settled on using a single camera with a wide angle lens, placed on the floor in the front of the stage, with a simple 4 x 4 grid. We spent enough time with this setup so that we could perform and improvise with some degree of expertise. The sixteen active grid areas (regions) provide plenty of challenges and variety, particularly since the size of the active areas changes as the performer moves upstage and down stage, and the function of one or more grid points can change through software. Thus, each square has the ability to trigger a particular MIDI note or set of notes, report changes in movement, and start or stop a computer process used to create music (or to reconfigure the software). MIDI boxes used include the Kurzweil K2500R synthesizer with 32MB of sampling memory and an Ensonique DP/4 signal processor.

The software can combine two or more regions to form a single, larger active space, or to form an area that the system will ignore. (Although the VNS is capable of combining grid spaces into single regions, and blocking out or "masking," various areas, it proved easier to automate and structure these changes via Max). In this way, further variations of the initial 4 x 4 set-up were created, such as using all of the middle regions of the grid to determine the overall tempo, while using the outer regions to start and stop specific layers of music. For other purposes, the space was divided into left/right sides, or high/low regions (figure 3).

The response characteristics of each region remained constant, so that a presence trigger was reported immediately upon entering a defined space, and reset only after leaving the space or by becoming motionless within the space. In this way, constant movement in one space avoided sending a trigger, but a short pause, or leaving a region and returning, would allow for the next trigger. The "retrigger" parameter (Threshold Setting in figure 1) could be set high so that only large movements acted as triggers, or low so that any small movements acted as triggers. At the same time, motion data was used, as reported to the interface.

Our familiarity with a single set-up paid off in our ability to try out new ideas quickly. By understanding and going deeply into a single set-up, we avoided the "kitchen sink" approach, attempting to use all of the possibilities (and the additional complexity of changing grids, light sensitivity, etc.). Once we had accurate numbers representing movement, our simplified set-up enabled us to focus more of our time on creating compelling connections between sound and movement.

Examples

Linking a single sound to a specific area on stage was an obvious starting point, turning the space into the equivalent of a spatial MIDI keyboard. These spaces could then become less predictable through the controlled randomization of MIDI notes or channels, so that a mixture of sounds could be produced in one region. A very effective technique was achieved using three related sounds layered in one region, with the continuous motion data used to determine sample selection. In this way, the amount of energy expended in one region could be reflected in the resulting sound. Similarly, a musical process could be continuously shaped over time, such as transposing or speeding up the music in response to increased physical activity. Each time a region was triggered, a corresponding number entered the program. The "map" module allowed for real-time note assignments during our improvisations, by recording sixteen notes from a MIDI keyboard. Collections of these maps were stored and any number of them could be activated at a time.

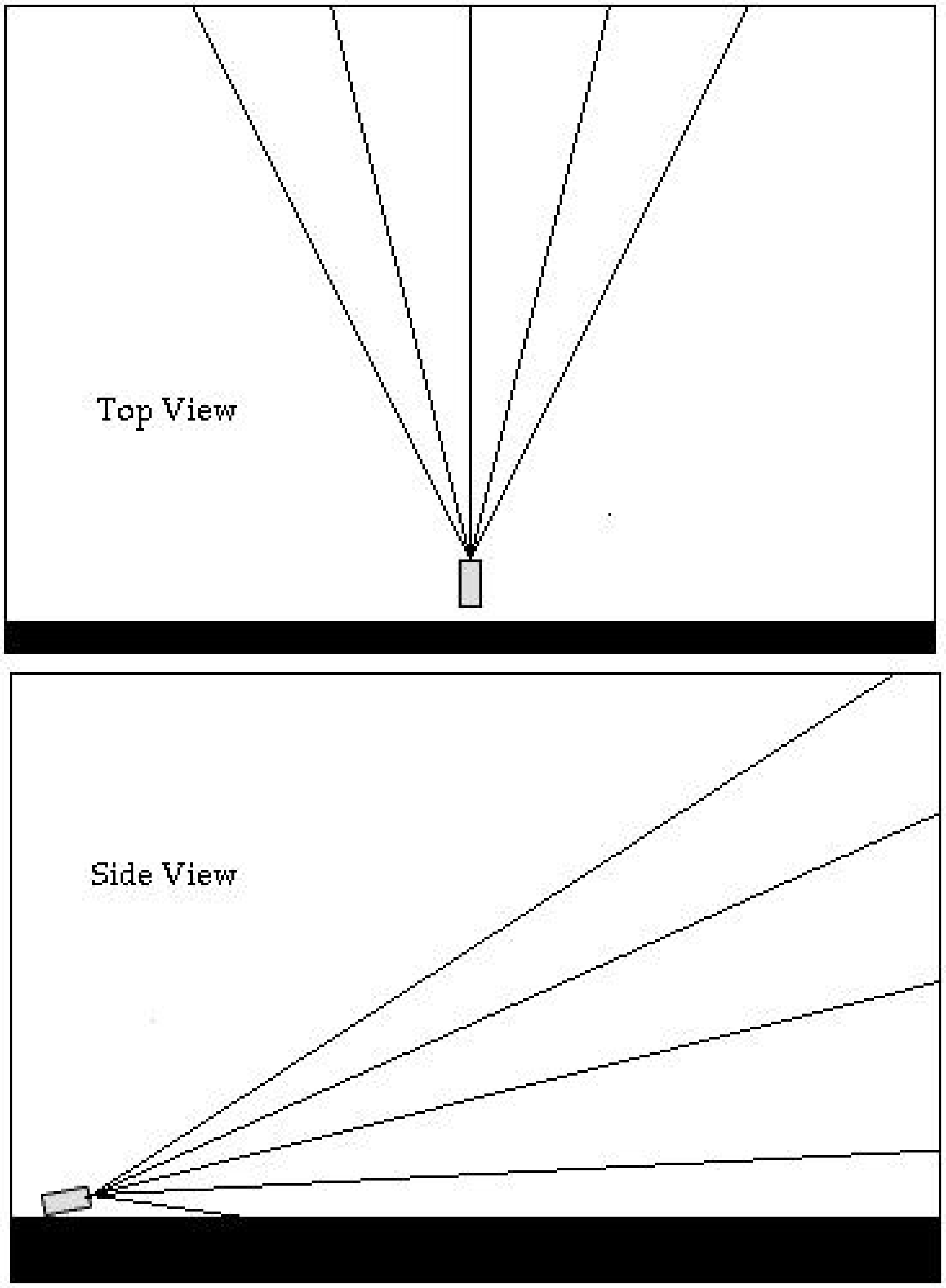

Figure 3. Projected 4 x 4 grid space (sixteen regions) showing the active horizontal and vertical areas.

Although we tried out a few ideas using synthesized sounds with fixed pitch, we gravitated towards samples of percussive sounds, vocal sounds, and machine sounds. We experimented with associating these sounds with a particular part of the body and the force and weight used to create the movement. Thus, a heavy jump onto one leg had the weight of a large, low sound. Small head movements or flicks of the hand were lighter with less energy, and seemed to fit smaller sounds like a bell or small cymbal. While this approach was often too literal, it made the sounds "feel right" to the performer for particular movements. Being aware of the underlying physics of movement, however, did not necessarily imply an obvious musical solution. Tampering with the apparent laws of physics, a luxury made possible in virtual environments, also provided successful results. So, we also played with seemingly contradictory and unpredictable results, creating models of response unique to the computer. More furious and strenuous activity, for example, could result in quieter sounds and silence. Or, a small yet deliberate nod of the head could set off an explosion of sound. Such "unnatural" correlations often made motion all the more meaningful.

The one-to-one approach was extended using prestored sequences of notes or algorithms to produce a different single sound or pitch each time a particular region was triggered. This was an especially successful approach when using pitched sounds, since a single area could be retriggered to generate melodies with continuous variation. A single trigger was also used to play short melodic phrases.

"Physical mixing" employed longer multitrack sequences, with the mix controlled by speed and location. Each grid space acted as an on/off toggle switch to enable or disable tracks. In an improvisational setting, this method worked best with an 6 - 8 track maximum, since it was difficult to keep track of the on/off state of more tracks. With practice and rehearsal, a larger number of tracks would be possible to control in a choreographed work.

Perhaps more interesting, but no less effective, were the sections using speed to control musical processes. Certain processes, such as transposition, worked well on a continuous basis, with the music moving high and low corresponding to the amount of movement on stage. Continuous changes in tempo were a bit chaotic, and became more predictable when divided to create discreet selections of three or four related speeds. Other processes influenced by overall movement included steadiness of the pulse, amount of dissonance, range of melodies, phrase length, timbre, note density, selection of intervals, melodic contour, and articulation. A smoothing algorithm was often used to average these values to avoid abrupt changes. Motion data averaged over a longer period of time would reflect general types of movement (fast/slow) within a section. This resulted in interesting delayed processing that showed the accumulative results of movement rather than an immediate response.

Conclusion

This paper describes some of our current techniques and ideas used for interactive dance. Imaginative relationships between sound and movement are established by viewing the body and space as input to a responsive computer music system. The interactive loop is completed when the dancer then responds to the resulting sound. This new "musical instrument," free from the associations of traditional instruments, still has a distinct character based on its physical and technical limitations. The VNS allows for experimentation with the physical and psychological properties of movement, and their musical interpretation.

References

Rokby, David. Personal website. http://www.interlog.com/~drokeby, 1997.

Winkler, Todd. "Making Motion Musical: Gesture Mapping Strategies for Interactive Computer Music." In *Proceedings for the 1995 International Computer Music Conference*. San Francisco, CA: Computer Music Association, 1995.

Winkler, Todd. "*Followplay*: A MAX Program for Interactive Composition." In *Proceedings for the 1992 International Computer Music Conference*. San Francisco, CA: Computer Music Association, 1992.