On Movement, Structure and Abstraction in Generative Audiovisual Improvisation

William Hsu
San Francisco State University
1600 Holloway Avenue
San Francisco CA 94132, USA
whsu@sfsu.edu

ABSTRACT
This paper overviews audiovisual performance systems that form the basis for my recent collaborations with improvising musicians. Simulations of natural processes, such as fluid dynamics and flocking, provide the foundations for "organic"-looking movement and evolution of abstract visual components. In addition, visual components can morph between abstract non-referential configurations and pre-defined images, symbols or shapes. High-level behavioral characteristics of the visual components are influenced by real-time gestural or audio input; each system constitutes a responsive environment that participating musicians interact with during a performance.

Keywords
Improvisation, interactive, generative, animation, audio-visual

1. MOTIVATIONS
In the last few years, I have been working on a series of audio-visual pieces for performance with improvising musicians. Each piece is an interactive animation environment that responds to gestural input and real-time audio. The animations are projected on stage with the musicians. The video becomes an additional component in the interaction between the musicians during the performance. The musicians cannot directly control the details of the animations; they are improvising with each other, and with the animations. Each system is primarily generative; the initial specification of a minimal amount of source material (usually a few images or shapes) will result in a wide range of dynamically evolving structures and behavior for different performances. Hence, my work takes a different approach from systems such as VERSUM [1], which is primarily an audiovisual sequencer with powerful spatialization capabilities.

While the concept and "look" of my pieces can vary significantly, there are some unifying themes that inform much of my work in this area. As a performer, system designer, and listener, I have noted the associations we often make between abstract gestures and forms, and natural phenomena. I have tried to build performance environments that capture some of these associations.

My animation environments are designed primarily for working with free improvisers. Hence, I work mostly with abstract visual components, such as particle clusters, lines and curves. Gestural and timbral cross-referencing between sound and visuals evokes the tactile, nuanced, timbrally rich gestures that I enjoy in improvised music. The complex behavior observed in natural processes, such as fluid dynamics and flocking behavior, seemed promising to me for evoking complex and dynamic gestures. A number of my pieces are built around simulations of such processes. The resulting visual phenomena evolve in a complex and highly detailed manner, comparable with the gestures of free improvisers.

While the basic components of my animation environments are abstract particles and shapes, I am interested in setting up tensions between abstract elements and configurations that reference or evoke concrete objects or symbols. While a specific visual stimulus can fundamentally be interpreted in multiple ways, the human visual system tends to prefer one clear interpretation. One of my goals is to encourage situations where this interpretation is highly unstable, shifting from an unstructured configuration such as a pseudo-random point cloud, to a well-defined image with a fairly unambiguous interpretation, such as a human skull. In all the environments and pieces described in this paper, the abstract elements are able to coalesce into well-defined images, patterns or symbols, and eventually scatter into non-referential configurations. These evolutions in structure occur in the context of the simulated process that is the basis for each environment; they can be influenced by gestural input or real-time audio.

In this paper, I will describe two groups of pieces that incorporate improvising musicians and interactive animation. These are the *Interstices* pieces, which are based on particles in a fluid system, and the *Flayed/Flock* pieces, which are based on flocking simulations. I will focus on 1) using simulations of natural processes, such as fluid dynamics and flocking behavior, to achieve an overall framework for generating motion of visual components that evoke natural processes, 2) adapting the simulations to incorporate referential visuals, and handling transitions between abstract elements and images, and 3) interaction with gestural input and real-time audio in the context of live performance. Versions of these pieces have been performed at Sound and Music Computing 2009 (Porto), Steim (Amsterdam), and other venues in the United States and Germany.

2. PAINTING WITH PARTICLES
The *Interstices* group of pieces are based on the motion of particles in a fluid system. My starting point for the fluid simulation is Glen Murphy's Fluid code [7]. Murphy's code was also used in his *Fluid Bodies* installation [8]; viewers interact with the fluid system through a camera and projection setup. Viewers' movements cause changes in particle brightness and density, and "reflections" of the viewers form in the particle system.

2.1 Motion Generation

In the *Interstices* pieces, a system of up to hundreds of thousands of particles is manipulated with gestural controllers, such as a graphics tablet or a multi-touch device. Synthetic components such as attractors, repulsors and large tidal generators can be dropped into the system and set in motion. In addition, images (pre-loaded or captured in real-time) can be placed in the particle system, to be scattered apart by the simulated fluid movement; particles may also swirl and coalesce into images. The end-result resembles somewhat asymmetric, constantly morphing versions of the Rorschach inkblots used in psychological evaluations; the goal is to open up a wide range of visual associations.

My reworking of Murphy's code mostly involved increasing its efficiency to support large numbers of particles and a fine-grain 800 x 600 simulation grid; I found that the latter especially enhanced the highly-detailed look of the animation that I was aiming for. To enhance performance, the simulation grid is subdivided into 32 x 32 tiles; fluid velocity and pressure updates are skipped for tiles that contain very few particles.

In addition, I built a number of classes to implement high-level behavioral components in the fluid system. *Attractors* and *repulsors* are simply centers of positive or negative gravity in the fluid system, that influence the motion of the particles. These components themselves can be set in motion within the system. *Tidal generators* correspond to large sweeping gestures that influence the motion of particles in a large area; a *scheduler* object manages the aleatoric generation of sequences of actions to create tidal currents in the fluid system. With my customization and tuning of fluid simulation, these currents result in large "splashing" gestures that evoke breaking waves or painterly splatters. Such gestures can evolve for extended periods of time, through multiple shape configurations, before finally damping out. Interactions between tidal currents and attractors are also complex and unpredictable. A number of examples can be seen in the video clips at http://userwww.sfsu.edu/~whsu/PSHIVA. Figure 1 shows snapshots of the evolution of the particle system, over about 16 seconds, under the influence of a single tidal current, from a section in my video *A Way*. The ring-like shape is a final state that the system sometimes settles into, after many frames of chaotic behavior.

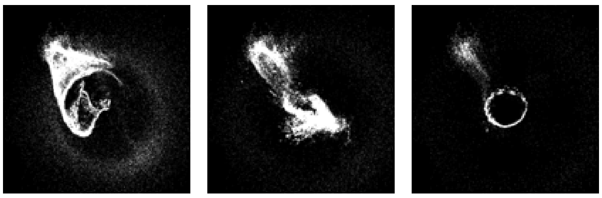

Figure 1: Examples of particle configurations as a result of tidal currents

2.2 Incorporating Pre-defined Structures

The particle system is also able to coalesce from a pseudo-random spatial distribution to an image or other well-defined referential visual structure, which I will call a *morph target*. A morph target might be pre-loaded, or captured from real-time camera input. Morph targets can be chosen and positioned dynamically during a performance.

In installations such as Murphy's *Fluid Bodies* [8], the shapes of viewers cause changes in particle brightness and density; concrete shapes are introduced to pseudo-random particle clusters by essentially fading in the shapes. This works reasonably well in some contexts; however, I felt that "fading in" an image results in an effect that clashed with the overall look of my particle-based pieces. I felt that having pre-existing particles move and collectively coalesce into an image was more compatible with the underlying fluid-driven motion.

To enable transition from an unstructured distribution to a morph target, the list of particles close to a morph target are simply mapped in a straightforward manner to the pixels in the morph target. The necessary linear trajectories are calculated for each particle to reach its corresponding pixel in the morph target. Over the next few seconds, each particle involved in the morphing activity would follow its pre-defined trajectory; when all such particles have traversed their trajectories, they will have coalesced into the morph target.

This is a very simple but efficient approach to handling the transition from unstructured particle clusters to images. Because of the simple mapping of particle to pixel, unnatural motion artifacts are possible; however, these tend to be obscured by the complex particle motion that is usually present. There are often over 100,000 particles in the system, so more sophisticated optimization techniques to minimize motion artifacts were too compute-intensive for our current development platform (an Intel Core 2 Duo with a low-end graphics processor); we do plan to explore other mapping strategies in the future.

Once a particle cluster has coalesced into a morph target, it is simple to let the underlying fluid simulation take over motion control for the particles. The fluid-based motion usually breaks up the morph target, and the particles return to a pseudo-random unstructured state. Figure 2 shows snapshots of a particle cluster coalescing into an image (a skull).

Figure 2: Particle cluster coalesces into a skull

2.3 Interaction with Gestural or Audio Input

I have tried several strategies for managing the audio environment for *Interstices*. In solo performances, with a single performer manipulating the particle system and audio generation simultaneously, my primary concern is to enable direct and detailed control of the particle system through gestural controllers. Hence, the performer's gestures correspond directly to events in the fluid simulation, "stirring" the underlying fluid to move particle clusters, placing or removing attractors or repulsors, guiding the formation of tidal currents, triggering the coalescing of particles into morph targets, etc. The same physical gestures are "interpreted" and loosely mapped to generative sonic gestures; there are several high-level options for the interpretation and mapping, which can be chosen by the performer to build contrasting sections of a piece.

Sonic gestures are synthesized by specifying high-level sound synthesis parameters such as duration, loudness, brightness, amplitude modulation etc. In a particular section, large physical gestures may result in loud, bright sonic gestures of long duration; in another section, sonic gestures may be restricted to shorter durations with very low brightness.

In addition, the particle system can respond to real-time audio descriptors. For audio-reactive performances, I have used a customized version of Jehan's analyzer~ [4], and the Zsa descriptors [5] to generate audio descriptors. Estimates of activity level or timbral characteristics are extracted in real-time, and communicated via Open Sound Control to the animation environment. My preference is to avoid straightforward mappings of audio to animation parameters, such as brightness to position, etc, in favor of more open "interpretations" with multiple degrees of freedom. For example, the onset and continuation of a slow and loud sonic gesture may trigger a large tidal current in the animation; if the roughness of a sonic gesture is maintained above a threshold for a minimum time, a particle cluster will be triggered to coalesce into an image.

2.4 Implementation and Performances

The animation components of *Interstices* were developed in the Processing environment (http://www.processing.org). The sound analysis/synthesis component is a Max/MSP patch (http://www.cycling74.com). Gestural input from a tablet, touch screen or camera is captured and interpreted by Processing components, and sent to the sound synthesis components via Open Sound Control messages. Audio descriptors and other information captured by the Max/MSP audio analyzers is also communicated to the animation via Open Sound Control messages.

The first installment of the *Interstices* series was premiered at Sound and Music Computing 2009 in Porto. Subsequent performances have been at Steim (Amsterdam) in 2010, and at various venues in San Francisco and Germany, with musicians such as John Butcher, Chris Heenan, Gino Robair, Moe Staiano, and Birgit Ulher (using Ulher's drawings as morph targets in one section). A proposal for *Interstices AP*, a solo version with multitouch controller, has been submitted to NIME 2011.

3. FLOCKING FILAMENTS

Swarming or flocking has been widely observed in the collective behavior of migratory birds, ant colonies, schools of fish, etc. Flocking simulations have been used in generative art and music. Most of these projects simulate relatively small flocks of tens to a few hundred agents. [2] and [3] discuss the use of flocking/swarming agents to generate music.

My initial experience with flocking systems in the context of music improvisation was at the Live Algorithms for Music workshop in August 2009 in London. Tim Blackwell [3], Tom Mudd and I set up a chain of systems that improvised with percussionist Eddie Prevost. Prevost's live sound is analyzed, and descriptors are generated and distributed through software by Sebastian Lexer and Ollie Bown. The descriptors map to movement parameters of Blackwell's flocking simulation. My software module detected the presence, position and size of clusters in the flock; these were mapped to timbral space parameters in Mudd's software synthesizer.

Rowe and Singer's *A Flock of Words* [9] is a flocking animation of 10-30 words; flocking parameters are influenced by the performance of a chamber ensemble. My subsequent *Flayed/Flock* pieces utilize one or more flocks of thousands of particles that appear to "draw" abstract scribbles in space. In response to real-time audio, the flock formations evolve, and flocks are able to coalesce into well-defined shapes and symbols. To trace well-defined curves and patterns, dense flocks of thousands of particles are necessary; great attention must be paid to computational efficiency.

3.1 Motion Generation

Subsequently, I built a piece based on flocking, initially for use with Birgit Ulher and Gino Robair in San Francisco in February 2010. My starting point for the flocking simulation was Kyle McDonald's implementation [6], which simulates a single flocking population of particles. A particle moves through space based on Perlin noise; its motion is the result of forces affected by high-level parameters such as *speed* (an overall scaling factor for the velocity of each particle), *neighborhood* (essentially the extent to which a particle is influenced by nearby particles), and *spread* (the strength of an attractive force toward the centroid of the flock). I rewrote McDonald's code to support multiple flocks, improve its framerate significantly, and added the ability for the flock to coalesce into pre-defined patterns. Figure 3 shows two examples of curves traced by my flocking implementation.

Figure 3: Examples of curves traced by simulated flocks

3.2 Incorporating Pre-defined Structures

The complex looping curves and lines traced by simulated flocks seemed to me to be a good match with simple referential shapes or symbols, such as circles, crescents or stars. For my current implementation, I restricted my symbols to relatively simple closed shapes. Black-and-white *masks* of the shapes, with one color assigned to the inside and one to the outside, are stored in image files that are loaded during performance.

As in the *Interstices* particle-based pieces in Section 2, I opted for having the flock coalesce collectively into well-defined target shapes, rather than resorting any fade-in effects. Again, the decision-making process for instigating the movement toward the referential forms must be highly efficient. To coax a flock into a pre-defined mask, I used the following algorithm:

1) determine the centroid of the flock (this is already part of the flocking simulation)

2) center the mask at the flock centroid

3) if a particle is outside the mask, it experiences an attractive force towards the centroid

4) if a particle is inside the mask, it experiences a repulsive force away from the centroid

This set of simple and efficient rules is sufficient to push particles to settle near the outline/boundary of the mask. For complex asymmetrical shapes, it is also possible to manually place multiple attractors within the shape, one for each section. However, the particles are often distributed in a very uneven manner around the outline. Hence, part of the outline might be clearly visible, having attracted many particles, but a significant part of the outline may be missing, having attracted few or no particles at all.

To even out the distribution of the particles around the boundary, I was inspired by a suggestion from Kazunori Okada (Okada, personal communication), resulting in a fifth rule:

5) if a particle is on the boundary of the mask, it will move from a region of high particle density to one of low particle density

With the last rule in place, the flock reliably settled into a number of pre-defined test patterns, distributing evenly around the outline of each pattern. Figure 4 shows snapshots of a flock, initially in a non-referential configuration, then slowly transitioning into a crescent shape. More examples of these transitions can be seen at http://userwww.sfsu.edu/~whsu/ARFlock.

Once the flock has settled into a pre-defined image or pattern, simply removing the constraints of the pattern will cause the flock to return to its previous abstract flocking behavior. (This can also be seen for example as part of McDonald's Janus Machine installation.)

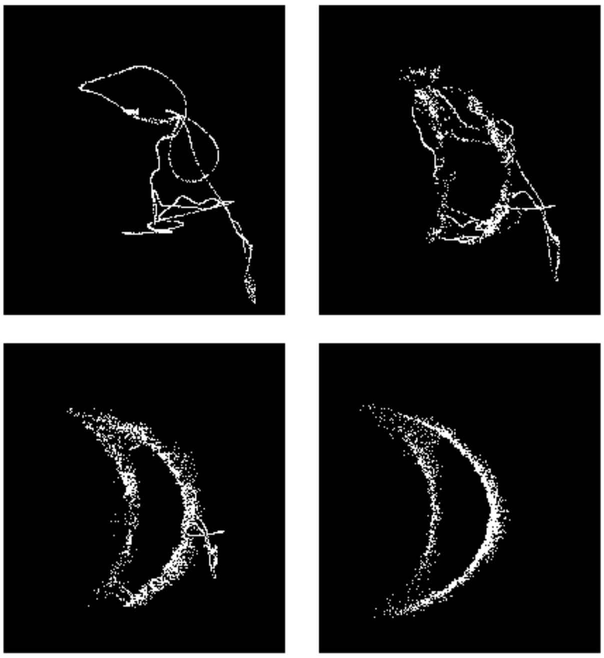

Figure 4: Transition of simulated flock from non-referential configuration to a pre-defined crescent shape.

3.3 Interaction with Gestural or Audio Input

In my design for flock's interaction with real-time audio, I wanted to avoid overly obvious mappings of audio parameters to spatial parameters in the flock's movement. My intention is to allow the flocking simulation to evolve based on its own rules, with the audio influencing high-level behavioral trends only. In earlier versions, I have again used simple activity measures for the audio, based on spectral flux [5]. Greater activity would increase the number of particles in the population, the speed of the population, and encourage the formation of coherent lines (by varying the *neighborhood* parameter); lower activity levels would decrease the number of particles and their speed, and encourage the flock to disperse in a random-seeming fashion. This can be clearly observed in the video clips available online. The details of the flocking behavior still evolve in a complex and unpredictable manner, but the high-level trends make clear references to the real-time audio.

For a flock to coalesce into pre-determined patterns and symbols, longer time intervals are required than are served by moment-to-moment monitoring of real-time audio descriptors. One approach I have explored uses an activity measure over a larger time window; as a musician remains active over a period of time, the analogy is to "energy" building up in a system. When a threshold is crossed, the flock undergoes a process of coalescing into a target pattern. As the activity level decreases over a period (i.e., energy levels dissipate), the particles abandon their previous target pattern, and return to the basic flocking behavior.

3.4 Implementation and Performances

Software for the *Flayed/Flock* pieces is structured in a similar manner as the *Interstices* pieces. Visual components are in Processing, audio components in Max/MSP, and gestural information and audio descriptors are exchanged via Open Sound Control.

Flayed/Flock has been performed with a number of free improvisers with varied insruments and approaches, such as its premiere at Artists Television Access gallery in San Francisco in February 2010, with Gino Robair (percussion, electronics), Birgit Ulher (trumpet) and myself (electronics), and at Steim (Amsterdam) in May 2010, with Gareth Davis (bass clarinet), Anne Laberge (flute) and myself. A proposal for *Flayed/Flock*, in collaboration with Oslo-based musicians Håvard Skaset and Guro Skumsnes Moe, has been submitted to NIME 2011.

4. SUMMARY

I have described the concepts and technology behind some audiovisual performance systems that I have built in the last few years. These were designed for my work with musicians, largely in a non-idiomatic free improvisation context. The generative visuals constitute complex, highly detailed gestures and textures, with unstable forms that encourage constantly shifting viewer interpretations. The systems have been used in a number of solo and collaborative performances, with musicians such as John Butcher (saxophone) and Gino Robair (percussion, electronics).

Video excerpts of the *Interstices* particle-based pieces can be found at

http://userwww.sfsu.edu/~whsu/PSHIVA/

Excerpts of the *Flayed/Flock* pieces can be found at http://userwww.sfsu.edu/~whsu/ARFlock/

5. REFERENCES

[1] Barri, T. Versum: audiovisual composing in 3d. In *Proceedings of the 15th International Conference on Auditory Display (ICAD2009)* (Copenhagen, Denmark, May 18-21, 2009).

[2] Bisig, B. and Neukom, M. Swarm Based Computer Music – Towards a Repertory of Strategies. In *Proceedings of the 11th Generative Art Conference* (GA 2008).

[3] Blackwell, T. Swarming and Music. In *Evolutionary Computer Music*, 2007, pp. 194 – 217, Springer-Verlag.

[4] Jehan, T., and Schoner, B. An Audio-Driven Perceptually Meaningful Timbre Synthesizer. In *Proceedings of the International Computer Music Conference* (2001).

[5] Malt, M. and Jourdan, E. Real-Time Uses of Low Level Sound Descriptors as Event Detection Functions Using the Max/MSP Zsa Descriptors Library. In *Proceedings of the 12th Brazilian Smposium on Computer Music* (September 2009).

[6] McDonald, K. Clouds are Looming, http://openprocessing.org/visuals/?visualID=6753

[7] Murphy, G. Smoke 2, http://bodytag.org/smoke2/

[8] Murphy, G. Fluid Bodies, http://bodytag.org/fluid_bodies/

[9] Rowe, R. and Singer, E. Two Highly Integrated Real-Time Music and Graphics Performance Systems. In *Proceedings of the International Computer Music Conference* (1997).