Fluid Simulation as Full Body Audio-Visual Instrument

Andrew Johnston
Creativity & Cognition Studios
School of Software
University of Technology Sydney
andrew.johnston@uts.edu.au

ABSTRACT

This paper describes an audio-visual performance system based on real-time fluid simulation. The aim is to provide a rich environment for works which blur the boundaries between dance and instrumental performance – and sound and visuals – while maintaining transparency for audiences and new performers. The system uses infra-red motion tracking to allow performers to manipulate a real-time fluid simulation, which in turn provides control data for computer-generated audio and visuals. It also provides a control and configuration system which allows the behaviour of the interactive system to be changed over time, enabling the structure within which interactions take place to be 'composed'.

Keywords

performance, dance, fluid simulation, composition

Figure 1: *Encoded* in performance, November 2012.

1. INTRODUCTION

In this paper an interactive audio-visual instrument based on real-time fluid simulation is described. The motivation for this project is to explore ways of providing intimate control of real-time sounds and visuals which are simple to understand and transparent in operation, but also rich and complex enough to be engaging and interesting for both performers and audiences.

The system itself was developed in collaboration with the dance/physical theatre company Stalker Theatre, and was used in an hour-long dance work, *Encoded*,[1] which premiered in November 2012 (figure 1). In that work, the system was used largely as a visual instrument, but we have more recently been working to integrate interactive audio into our work in order to explore (and blur) the boundaries between dance and instrumental performance.

Encoded explored how notions of digitised space alter our perceptions of physical space. It involved the use of both large and small scale interactive projections onto architectural features of the performance space and the performers themselves. In order to play with the boundaries between the real-world and digital representations of the real-world, we developed a system which uses what could be termed a 'natural user interface' or 'reality-based interaction' [3].

The term 'natural' is potentially controversial in this context [8]. The use of this term is quite specific here. It is used to refer to interfaces based on simulations of physical processes – interfaces which respond to user gestures in ways which are intuitively understandable because they are based on the laws of Newtonian physics.

Of course, simulations of physical systems have a long history in sound synthesis, but they also have interesting and useful characteristics when used in musical interfaces. Momeni and Henry, for example, propose the use of physical models as an intermediate mapping layer [7], arguing that they effectively support 'rich and intuitive exploration' of audio-visual synthesis and provide an 'intrinsic link' between sound and visuals.

The author has previously created and evaluated a series of virtual musical instruments which made use of mass-spring models to simultaneously visualise acoustic sounds and provide control data for audio visual synthesis [4, 5] Mass-spring models were found to provide a good balance between 'legibility' and complexity. Because they behaved like actual physical objects, their behaviour was intuitively understandable – performers could see that the behaviour of the physical system was a direct and physically plausible result of their actions. However, even very simple mass-spring models exhibit complex, emergent behaviour which meant that the generated sounds and visuals, while clearly a result of the gestures of the performer, could not in practice be precisely predicted.

William Hsu has used fluid simulations, rather than mass-spring models, in his real-time performance systems [2]. In his work, the fluid is manipulated by performer gestures sensed by tablets, touchscreens or cameras, and also by

[1] `http://vimeo.com/55150853` (5 minute highlights); `http://vimeo.com/54920989` (full show)

acoustic sounds. A key difference with our work is that Hsu does not use the fluid as a source for sound generation. Rather, sounds produced by his system are linked to the performer's gesture directly.

There are also a number of predominantly visual uses of fluid simulations in interactive artworks. A common theme in these works is the use of the fluid as a kind of liquid container for image pixels. Works such as Graeme Murphy's *Fluid Bodies*, Mehmet Atkin's *Gold* and Hsu's *Interstices* function at times as a kind of interactive mirror in which images form and dissolve, exploring the line between representation and abstraction. In addition, the fluid provides an interaction mechanism which is playful and intuitively understandable.

2. SYSTEM DESCRIPTION

The overall structure of the system, primarily written in C++ using OpenFrameworks[2], and Pure Data,[3] can be seen in figure 2. The movements of the dancer are captured via a Point Grey FireFly camera[4] fitted with a visible light filter (Lee #87), which blocks visible light while allowing infra-red light to pass through. Infra-red lights are used to illuminate the performance space so that the performers can be tracked independently of the stage lighting.

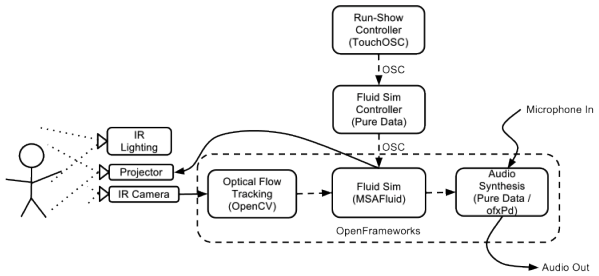

Figure 2: An overview of the system.

The camera images are passed through an OpenCV[5]-based optical-flow motion tracking system which tracks the movements of pixels between frames of video [6]. We found this approach preferable to a more complex 'blob tracking' system which attempts to identify and group together coherent regions of moving pixels, because it was more responsive and also less prone to error. The blob-tracking system ran into difficulty when the blobs it was attempting to track changed size or shape, resulting in less predictable, non-intuitive behaviour.

The pixel movements identified by the motion tracking system were linked to a real-time fluid simulation, a modified version of `MSAFluid` by Mehmet Atken.[6] The movements of the performers' bodies effectively 'stirs' the fluid and may also add particles and/or colour to it. The fluid simulation is extremely flexible and there are numerous parameters which can be adjusted during performance. These include parameters which affect the response of the fluid simulation to dancers' movements (viscosity, for example) as well as settings which affect how the fluid is visualised (eg. colour or black and white, particles or lines, etc). All settings are set from a Pure Data patch which sends Open Sound Control (OSC) [14] control messages to the fluid simulation. This allows parameters for multiple fluid simulations to be set from a single controller. This is important as for some performances we have had up to seven separate instances of the fluid system running on separate computers simultaneously.

[2] `http://www.openframeworks.cc/`
[3] `http://puredata.info/`
[4] `http://www.ptgrey.com/`
[5] `http://opencv.org/`
[6] `http://www.memo.tv/msafluid/`

2.1 Audio Synthesis

In order to bring sound to what was previously a purely visual interactive system, it was of course necessary to decide how exactly to map the state of the fluid simulation to audio synthesis parameters. The simulation is realised by breaking the fluid up into a grid of cells (we have been using a 150 x 116 grid) which store velocity and density values [13]. While this is a rich source of information about the state of the fluid, it is also a large amount of data to manage at 60 frames per second. Our solution was to simply average the velocity values of neighbouring cells so that the simulation output a more manageable 400 values each frame.

We found even this to be too much data to effectively transmit over OSC to Pure Data (our preferred audio processing environment), so `ofxPd`,[7] an OpenFrameworks addon by Dan Wilcox which wraps an instance of the Pure Data audio environment inside an OpenFrameworks application, was used. This meant that the fluid simulation state information passed directly to our Pure Data audio patch without having to pass through the network stack.

Our first approach to creating live audio with the fluid simulation was to map the averaged velocity values of the cells to the gains of band-pass filters in a filter bank (figure 3). Pre-recorded audio files were then passed through the fluid-simulation-controlled filter bank. Using this technique a wide range of sounds could reliably be produced and affected by the movements of performers. However, the fact that movement in particular regions of the fluid simulation always produced sounds with particular frequencies (eg. high sounds to left of screen, low sounds to the right, etc) quickly became tedious. We sought a richer sound space and more complex and subtle mappings.

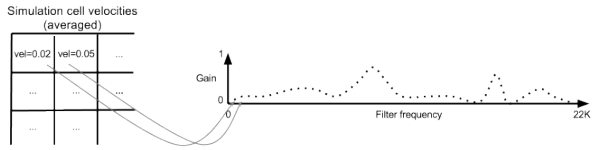

Figure 3: Initial mapping of fluid cell velocities to filter frequencies.

For this reason, for the next iteration we chose to use techniques from concatenative synthesis [10, 15, 11] to create a sound space which was controlled by the fluid simulation. The sound synthesis system, based on the `timbreID` toolkit for Pure Data [1], first analyses an audio file to identify sound onsets/attacks. A short segment of audio at each onset, the size of which is adjustable but typically set to around 2000 samples, is analysed to produce a set of Bark frequency cepstrum coefficients – perceptual descriptors of the timbre of the attacks. The `timbreId` external is then used to reorder the attacks so that sounds are grouped together by timbral similarity.

The first 400 segments of audio, now grouped together by timbral similarity, are used as the source for a granular synthesis engine. Each audio segment (grain) plays continually, at a volume set by the velocity of fluid in associated

[7] `https://github.com/danomatika/ofxPd`

cells of the simulation (figure 4). Additionally, grain size and a random 'deviation factor' (amount of delay between grain playback loops) [9] can be set via sliders on the user interface.

Effectively, this means that the fluid is broken up into a grid of 400 cells, each with their own velocity, which is affected by the gestures of the performer. The velocity of each cell is mapped to the volume of an individual synth unit which continuously outputs a looped sample from a pre-analysed sound file. Because the samples have been grouped together by timbral similarity, the effect is that neighbouring regions of the fluid generate similar sounds.

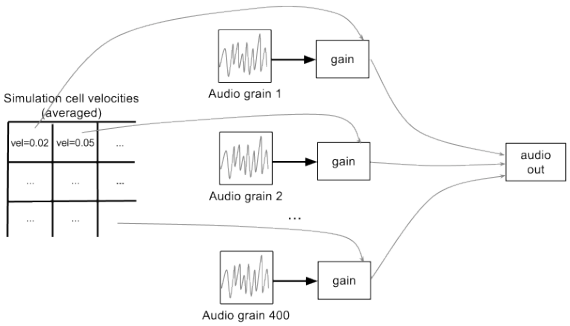

Figure 4: Mapping fluid simulation cell velocities to granular synthesis.

This basic approach to creating and exploring a sound space draws on the approach taken by CataRT [12], although simplified in the sense that while perceptually similar sounds are grouped together, they are uniformly distributed over the 2D grid of fluid simulation cells, rather than being positioned closer together or further apart based on their *degree* of similarity.

The granular/concatenative approach to audio allows a broader range of sounds to be produced, and the links between gesture and sound can be richer and more complex than we were able to achieve using the simpler filter-bank approach. Any audio file can be loaded into the synthesis engine, analysed and then played back via the fluid simulation, so the range of sounds produced is essentially unlimited. In addition, the system allows for the results of the audio analysis to be saved and reloaded. This means that it is possible to load and transition between any number of sound spaces during performance, allowing the sounds produced by the system to change significantly over time if desired.

2.2 Incorporating Acoustic Sounds

Ultimately, the system we have developed will be used in performances which include vocalists and musicians playing acoustic instruments For this reason we wanted to explore ways of incorporating these acoustic sounds into the fluid instrument. The primary mechanism which was devised for this was to bring the live acoustic sounds into the granular synthesis system. This technique works as follows:

1. The number of audio snippets to be recorded (up to the maximum of 400) is specified and recording is triggered.
2. The cells in the fluid simulation with the highest velocity are sent to the audio patch.
3. The audio patch records the specified number of audio snippets into the grains which are associated with the cells identified in step 2.

The live audio recorded in this way is not analysed using `timbreId`, primarily because we wished to maintain a strong relationship between gesture and sound during recording. If a live sound was analysed and then associated with a fluid simulation cell which was not currently active then the immediate effect of the live sound on the fluid may not be apparent. However, if instead it is associated with a cell which is active (because it is being affected by the movements of performers) then the recorded sound is likely to be immediately heard. We feel that this behaviour is likely to be more comprehensible to performers and audiences.

2.3 Fluid Manipulation Objects

For previous versions of *Encoded*, the movement of the fluid is affected only by physical gestures of dancers. This was intended to ensure that the human performers were the audience's primary focus, with the system playing a supportive role, and avoid a situation where the system distracted the audience with technical 'magic' which was unrelated to the performers' actions. As can be seen in the video, this meant that while performers were aware of the fluid and at times approached it instrumentally, the interaction was predominantly 'ornamental' [5, 4], in that the fluid augmented and supported their movements without significantly affecting them during performance.

While we wanted to retain a clear link between performer actions and the behaviour of the interactive system, and to continue to support instrumental and ornamental interactions, we were keen to explore ways of encouraging more complex, 'conversational' interactions to develop. In addition to linking the fluid behaviour to sounds, as discussed in previous sections, we therefore added an additional mechanism for affecting fluid behaviour: fluid manipulation objects. These are objects that can be positioned in the fluid simulation and add forces at their position. The manipulation objects are placed in the fluid simulation using the mouse and keyboard. Once in place they can be moved around directly with the mouse. Alternatively they can be coupled to particles in the fluid simulation, in which case they will flow about the screen in response to gestures.

Once the manipulation objects are in place they can be set to apply attractive or repulsive forces to the fluid at their location. The objects may be also be set to pulse, in which case the forces ramp up and down at an adjustable rate. Finally, they may be linked to audio, in which case the volume of incoming audio sets the level of force to be applied to the fluid. Using this mechanism, acoustic musicians, for example, can poke and prod the fluid with the sound of their instruments. This is particularly effective when a manipulation object linked to audio is used in conjunction with the live audio recording feature described in section 2.2.

2.4 Control System

The large number of parameters which affect the behaviour of the fluid simulation and associated visualisations and sonifications mean that the overall system provides an extensive expressive pallet. This, of course, is desirable, but maintaining some degree of control over the system during performance requires practical features such as the ability to save parameters into presets which can be loaded quickly during performance, and methods to smoothly transition between states.

This is provided by a relatively simple run-show system consisting of a set of Pure Data patches which send OSC messages to the fluid simulation. The patches allow all parameters to be saved so that effective states can be easily reloaded during performances. In addition, gradual tran-

sitions between states can be triggered by specifying that parameters ramp to preset values over a particular time period. Thus it may be specified that the preset 'scene 2' be transitioned to over a period of 15 seconds for example. If greater control is desired, it is possible to specify that certain parameters change at different rates: viscosity might transition to a new value over 10 seconds, while a new audio file is loaded after only 2 seconds, for example.

We noticed that as *Encoded* developed the amount of improvisation during rehearsals gradually decreased. Extensive improvisation did, however, occur during the early stages. Successful elements which worked well during development workshops were video recorded and saved as presets and then polished for inclusion in later performances. As effective fluid simulation states gradually became part of the final show, it became desirable for control of the system to be given to choreographers and the director. A simple TouchOSC[8] layout was created and used in final stage rehearsals and performances. The TouchOSC controller was also used to control other software used during performance, providing a unified control system for the full show.

3. CONCLUSION

This paper outlines our attempts at creating an interactive system which blurs the boundaries between instrumental performance and dance, and between sounds and visuals. The system we have devised is centred on a real-time fluid simulation which we believe offers intuitive controllability and transparency of operation, while providing behaviours of sufficient richness and complexity to sustain a large-scale performance. The visual components of the system have been successfully used in an hour-long dance performance, but the audio aspects of the system are new and still being refined.

With this technical foundation in place we plan to further develop and refine this system in performance. Areas for further exploration include considering more sophisticated layouts of the sound grains in the fluid space and finding which source sounds are most effective for this kind of work.

In addition, we have conducted a series of interviews with the performers, choreographers and other creative and technical staff involved in *Encoded* about their experiences with the original (visual only) fluid system during creative development, rehearsals and performances. The next round of interviews will focus on the introduction of audio and the impacts that it has on performers' experiences and practices.

4. ACKNOWLEDGEMENTS

Thanks to Andrew Bluff and Lukasz Karluk for their work on the fluid manipulation objects, and motion tracking system respectively. This work was supported by the Australia Council for the Arts, Creative New Zealand and a University of Technology Sydney Industry and Innovation grant.

5. REFERENCES

[1] W. Brent. A timbre analysis and classification toolkit for pure data, 2008.

[2] W. Hsu. On movement and structure and abstraction in generative audiovisual improvisation. In A. R. Jensenius, A. Tveit, R. I. Godøy, and D. Overholt, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 417–420, Oslo, Norway, 2011.

[8] `http://hexler.net/software/touchosc`

[3] R. J. Jacob, A. Girouard, L. M. Hirshfield, M. S. Horn, O. Shaer, E. T. Solovey, and J. Zigelbaum. Reality-based interaction: a framework for post-wimp interfaces. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 201–210, New York, NY, USA, 2008. ACM.

[4] A. Johnston, L. Candy, and E. Edmonds. Designing and evaluating virtual musical instruments: facilitating conversational user interaction. *Design Studies*, 29(6):556–571, 2008.

[5] A. Johnston, L. Candy, and E. Edmonds. Designing for conversational interaction. In *Proceedings of New Interfaces for Musical Expression (NIME)*, 2009.

[6] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[7] A. Momeni and C. Henry. Dynamic independent mapping layers for concurrent control of audio and video synthesis. *Computer Music Journal*, 30(1):49–66, 2006.

[8] D. A. Norman. Natural user interfaces are not natural. *Interactions*, 17(3):6–10, May 2010.

[9] C. Roads. *Microsound*. MIT Press, 2001.

[10] D. Schwarz. A system for data-driven concatenative sound synthesis. In *Digital Audio Effects (DAFx)*, pages 97–102, Verona, Italy, December 2000.

[11] D. Schwarz. The sound space as musical instrument: Playing corpus-based concatenative synthesis. In G. Essl, B. Gillespie, M. Gurevich, and S. O'Modhrain, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Ann Arbor, Michigan, May 21-23 2012. University of Michigan.

[12] D. Schwarz, G. Beller, B. Verbrugghe, and S. Britton. Real-time corpus-based concatenative synthesis with catart. In *Proceedings of the International Conference on Digital Audio Effects (DAFx-06)*, pages 279–282, 2006.

[13] J. Stam. Real-time fluid dynamics for games. In *Proceedings of the Game Developer Conference*, volume 18, 2003.

[14] M. Wright and A. Freed. Open sound control: A new protocol for communicating with sound synthesizers. In *International Computer Music Conference*, pages 101–104, Thessaloniki, Hellas, 1997. International Computer Music Association.

[15] A. Zils and F. Pachet. Musical mosaicing. In *Proceedings of DAFX 01*. University of Limerick, December 2001.