A Physically Based Sound Space for Procedural Agents

Benjamin Schroeder
The Ohio State University
Dept. of Computer Science and Engineering
395 Dreese Laboratories
2015 Neil Avenue
Columbus, OH 43210
benschroeder@acm.org

Marc Ainger
The Ohio State University
School of Music
110 Weigel Hall
1866 College Road
Columbus, OH 43210
ainger.1@osu.edu

Richard Parent
The Ohio State University
Dept. of Computer Science and Engineering
395 Dreese Laboratories
2015 Neil Avenue
Columbus, OH 43210
parent@cse.ohio-state.edu

ABSTRACT

Physically based sound models have a "natural" setting in dimensional space: a physical model has a shape and an extent and can be given a position relative to other models. In our experimental system, we place procedurally animated agents in a world of spatially situated physical models. The agents move in the same space as the models and can interact with them, playing the models and changing their configuration. The result is an ever-varying audiovisual landscape.

This can be seen as purely generative—as a method for creating algorithmic music—or as a way to create instruments that change autonomously as a human plays them. A third perspective is in between these two: agents and humans can cooperate or compete to produce a gamelike or interactive experience.

Keywords

Physically based sound, behavioral animation, agents

1. INTRODUCTION

Physically based sound models [4] can be used to create synthetic instruments that react in expressive ways to varied input. Such models often are defined in physical, spatial terms; for example, a plucked string might have a length defined in meters or an acoustic tube a certain diameter. It is therefore natural to think of creating an instrument by arranging these models in some kind of spatial setting. A performer might then use gestural input to play the virtual instrument [7].

Physical models produce realistic sounds, but because they are computer models, they are not limited to the strictly physical. One way to extend the capabilities of a virtual instrument is by introducing procedurally animated *agents* into the same spatial environment as the models. These agents can affect the physical constructs represented by the models in ways impossible or difficult in the real world. For example, the agents could play an instrument in algorithmic ways, but could also change the instrument over time, producing a changing experience for a human performer. Because the models and the agents are spatially situated, the changing system can be appreciated not only in terms of its sound, but also in terms of its visual appearance.

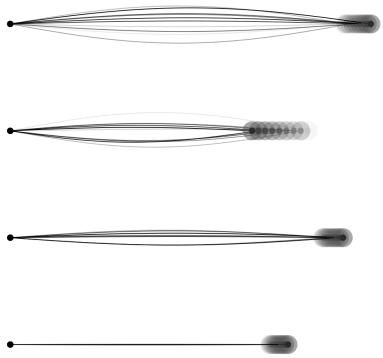

Figure 1: Agents smoothly changing the length of strings being played by a human performer, producing a *glissando* effect.

Figure 1 shows a scenario in which a human performer is playing several strings by strumming them using a mouse gesture. Meanwhile, several agents have grabbed the ends of the strings and are moving, choosing new lengths for the strings at random. This produces an unpredictable *glissando* effect as the strings' lengths, and thus their pitches, change.

In the remainder of this paper, we discuss an experimental system for situating behavioral agents and physical models in the same spatial world. After some background, we present the physical models and their embedding into space, and discuss how agents can interact with and change the models. We conclude with several example scenarios. The discussion throughout is in the context of our proof of concept implementation, which runs in real time on standard Macintosh hardware[1].

2. BACKGROUND

Other musical systems, such as the various modes of Electroplankton[2], have used behavioral motion to produce sound. A recent example is Lush [3], which assigned musical notes to individual flocking agents, giving users a movable play-

NIME'11, 30 May–1 June 2011, Oslo, Norway.

[1] A limited number of models can be simulated in real time; we are confident that increasing processing speeds and advanced simulation techniques will lead to greater capacity in the future.

[2] *Electroplankton* is a Nintendo DS game designed by Toshio Iwai; it was originally released in April 2005.

head which could capture and play the notes. In our system, the environment rather than the agents generates sound, although different agents could (for example) represent different excitation models.

There is a rich literature in computer animation regarding behavioral agents. A good starting place is the work of Craig Reynolds, inventor of the classic "boids" flocking algorithm [6]. We do not impose any particular high-level behavior on the agents in our system, but instead provide an approach for relating agents to physical models.

Cook's book [4] gives a good introduction to physical modeling synthesis. Our proof of concept implementation used finite difference time domain models; many issues surrounding this type of model are discussed in a recent book by Bilbao [1].

The recent web artwork *Conductor*[3], by Alexander Chen, represents subway traffic using physically-inspired strings drawn by moving agents. The sound played by the strings, however, is based on recorded samples.

Schroeder et al. [7] also embedded physical models in space; this work is an evolution of the system they describe. Both systems owe much to the way in which the Reactable [5] used spatial arrangement of its models, although the models in that system, and the particulars of their arrangement, are substantially different from those in the present work.

3. THE SOUND SPACE

The agents in our system interact with spatially situated physical models. Our proof of concept implementation includes models of musical strings and rectangular plates situated in 2D space. The models are based on finite differences [1] and are simulated in real time. These choices were made for the sake of simplicity, but other models could be used as long as they could be situated in space and respond to force input; similarly, the models could be embedded in 3D space.

Each model is described below in terms of its vibratory behavior, which is simulated to produce sound, and also in terms of its embedding in the agent space. The capabilities of agents to change or excite the models are described in a later section.

3.1 String Model

Our string model is an ideal string with basic damping and force-based input:

$$y_{tt} = c^2 y_{xx} - \sigma y_t + f(x,t),\ 0 \le x \le L.$$

In this notation, a subscript indicates a partial derivative; thus y_{tt} is the second time derivative or acceleration of the string.

In the equation above, the coefficient c is the speed of sound on the string; σ is a damping coefficient; force input is given by the function f, which varies in space and time. The string is of length L. Strings are assumed to have fixed boundary conditions.

A string is embedded in the space with a center position, a length, and a rotation. This implies the existence of two endpoints which agents may move; the string's length L is recalculated throughout the simulation based on its endpoints' positions.

3.2 Plate Model

Our plate model is similar to the string model, but differs in two ways. First, the motion of a plate in our system is mainly due to stiffness rather than to tension, and therefore the initial term of the equation of motion is a fourth derivative. Second, the model includes a frequency-dependent damping term meant to mimic the effects of different materials. A similar term could be included in the string model, but it has a greater effect here, allowing for mimicry of materials as distinct as metal and wood.

For the plate, then,

$$u_{tt} = -\kappa^2 \nabla^4 u - \sigma u_t + b_3 (\nabla^2 u)_t + f(x,y,t),$$
$$0 \le x \le L_x,\ 0 \le y \le L_y.$$

Here, ∇^2 is the 2D Laplacian operator, $\nabla^2(u) = u_{xx} + u_{yy}$, and ∇^4 is the biharmonic, $\nabla^4(u) = \nabla^2(\nabla^2(u))$. The coefficient κ describes the plate's stiffness and σ basic damping across all frequencies. The term with coefficient b_3 describes frequency-dependent damping. As with strings, plates are assumed to have fixed boundary conditions at all edges.

A plate is embedded in 2D space in a way similar to the string, above, except that the plate's extent is in two dimensions. Agents may change the extent of a plate.

Plates in our proof of concept implementation are always rectangular. There is no reason in general why non-rectangular shapes could not be used, perhaps through the judicious use of boundary conditions.

4. PROCEDURAL AGENTS

Procedural agents exist in the same space as a collection of sounding models. Each agent is essentially an oriented particle; it has a *position*, a *heading*, and a *velocity*. In order to allow environmental forces to act on the agents, each agent is also assigned a *mass*.

Agents' motion is simulated forward through time based on these properties. Navigation is specified at a higher level and is discussed briefly below.

An agent can affect its local environment in several different ways. First, an agent may apply *excitational force*, such as plucking or tapping, to a nearby sound model. An important variant of this is to feed some external sound signal, such as that of a microphone, into a model by adapting the signal as force input. Similarly, an agent may apply *damping* to a nearby model, allowing it to do such things as fret a string. The particular kinds of forces applied are up to the agent and may depend on factors such as its velocity as well as higher-level behavioral concepts.

An agent may also change the configuration of a nearby model. It may grab a model for *translation* in space or for *rotation* around a fixed point (such as the model's center). A grabbed model travels with the agent until it is dropped.

Finally, an agent may *create* or *destroy* models or other agents. (In our proof of concept implementation, we do not implement coupling between models, but in general an agent could also create or destroy such coupling connections.)

4.1 Model Visibility

An agent has limited access to its local environment. It can "see" and affect models at its current position or that it has crossed in the last time step. In planning navigation, high-level behaviors might make use of additional factors such as which other agents are nearby or which models are some distance ahead of a given agent.

One of the advantages of physical models is that they respond differently to forces applied at different points. A plate resonates differently, for example, with input near its edges than it does with input near its center. Therefore when an agent is given access to a model, the access is through a *proxy* that represents the particular region near the agent. Excitation and damping are then applied through the proxy to affect the model at the appropriate

[3] At http://www.mta.me; retrieved on February 2, 2011.

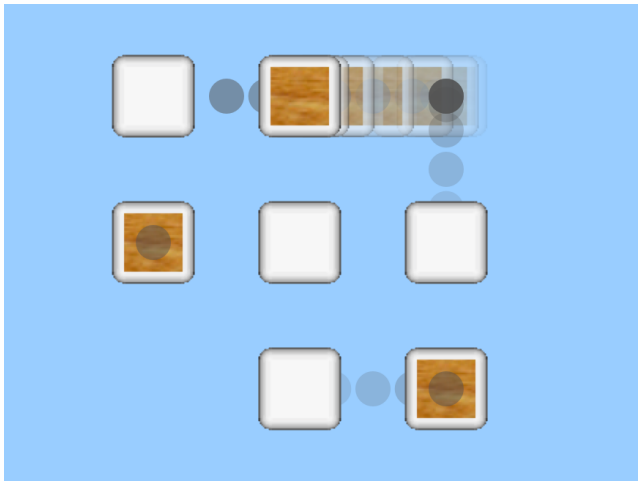

Figure 2: Agents moving and tapping rhythmically on sliding tiles.

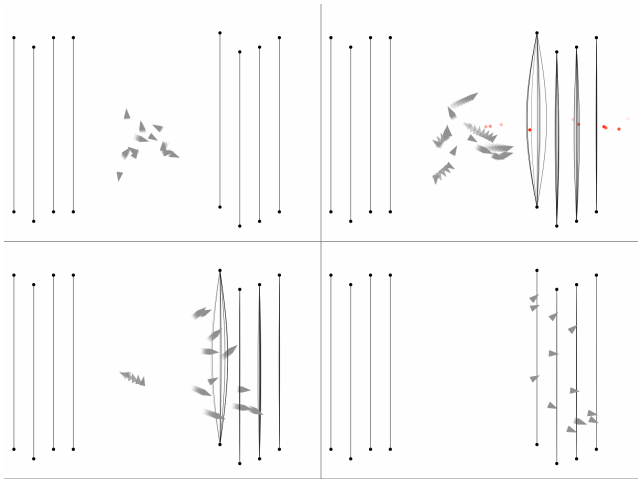

Figure 3: A system in which agents move to damp sounding strings.

location.

Shape changes are implemented as a variant of this: an agent is presented with proxies representing a string's endpoints or a plate's edges, which may then be moved as if they were entire models. Moving the proxies causes changes in the underlying models.

4.2 High-Level Navigation

Various high-level navigational strategies may be used with the agents described above; some of these are described in a later discussion of example scenarios. In general, a navigational strategy is responsible for planning the large-scale motion and behavior of an agent. A trivial example of a behavior might be that of a "water bug":

```
Given an environment with several strings,
Swing around to a random heading;
Skim across the surface at that heading.
If you cross a string while moving,
   Pluck it.
```

In this case, the navigational strategy would implement the logic above and apply forces to the bug to produce the "skimming" movement.

In general, navigational strategies might take the environment into account, sending agents towards a model, for example, or directing an agent to wander up and down the length of a string.

4.3 Environmental Forces

Environmental forces may be used to affect the agents by giving them physics-based motion. Examples of such forces might include gravity wells, repulsive barriers, and directional "wind". These forces change agents' velocities and may be applied based on proximity or globally.

If forces such as these are used in conjunction with other high-level navigation, the navigational strategy is responsible for weighting its input and that of the environment to produce whatever effect is desired.

4.4 Human Agency

A human user may interact with the environment at the same time as the agents, doing anything that an agent can do: playing sound models, moving them around or changing them, or creating or destroying models.

The human's relationship to the agents depends on the particular scenario implemented. For example, agents might change parts of an instrument that a human is playing through gestural input or through the microphone, giving the effect of playing an instrument which is changing over time. On the other hand, a human might interact with the agents more directly, putting obstacles in their way, giving them sound models to play, or guiding them toward a goal, producing a gamelike scenario.

5. EXAMPLE SCENARIOS

We have already seen a simple example of agents producing a *glissando* effect (Figure 1). Below are several other small scenarios involving agents, various behaviors, and physical models.

5.1 Rhythm on Sliding Tiles

Figure 2 shows a few agents moving rhythmically on a playing board made up of several sounding plates. The simulation proceeds as a series of *moves* made by the agents. For any move an agent may choose to

1. Stay still, doing nothing
2. Hop on its present tile, producing a sound
3. Hop to an adjacent tile
4. Slide its tile into a nearby gap
5. Change the material of its tile to be metal or wood

Agents are more likely than not to stay still (otherwise the rhythm would descend into chaos). An agent may only hop onto a tile which is not already occupied.

5.2 Sound and Silence

Figure 3 depicts two sets of strings, each tuned to produce notes from a pleasant chord. When the system is at rest, several agents wait in the middle, making only small, random movements.

The user may play any of the strings. When a string is sounding, agents will depart from the middle and latch onto the string in order to dampen the sound. Some time after a string becomes quiet, an agent damping that string will release the string and return to the middle of the screen.

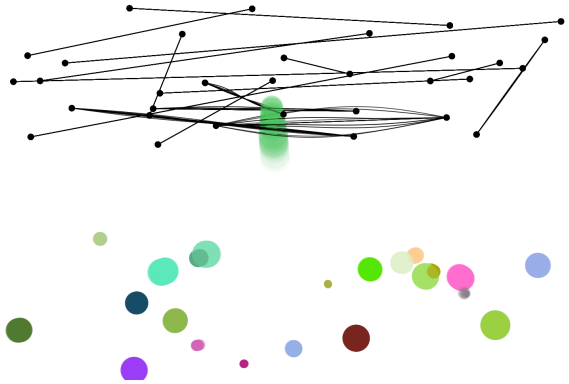

Figure 4: Bubbles representing recordings drift up, releasing their sound into resonating objects.

5.3 Resonance and Motion

Figure 4 depicts an interactive system. Users of the system speak into the microphone in order to record sound bubbles which then cluster near the bottom of the screen. Each bubble is a separate agent; when clustering, the bubbles drift randomly. Longer recordings are assigned larger bubbles.

At random times, a bubble is chosen to be released. The bubble/agent then drifts upwards; it feeds its stored sound into any nearby sound models, causing them to resonate, as it goes. Users interact with this system through the microphone, but a possible enhancement would be to give the users portable gravity and anti-gravity wells to affect the motion of the agents.

6. CONCLUSIONS

We have described an experimental system in which agents and physically based sound models are placed in the same space and allowed to interact. This extends the capabilities of the sound models, allowing for the creation of algorithmic audio or playful interactive systems.

Several questions and opportunities for future work remain. The agents have only a limited kind of interaction with the models; they can feed forces into the models, but are not themselves affected by model motion. It would be interesting to allow the agents to be affected by the sound vibration as well; in that case agents could represent, for example, elements reminiscent of those in the prepared piano [2], or could bounce off of vibrating membranes as if they were trampolines.

Our system does not yet allow for high-level, interactive programming of the behaviors. Such a programming system would be a useful addition, allowing users to experiment with different behaviors and configurations at runtime.

The agent behaviors discussed here are only a start. Further experimentation and research is needed to determine additional interesting musical uses for a behavioral system such as the one described here.

7. REFERENCES

[1] S. Bilbao. *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. Wiley Publishing, 2009.

[2] S. Bilbao and J. ffitch. Prepared Piano Sound Synthesis. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06)*, pages 77–82, Montreal, Quebec, Canada, Sept. 18–20 2006.

[3] H. Choi and G. Wang. LUSH: An Organic Eco+Music System. In *Proceedings of NIME 2010*, 2010.

[4] P. R. Cook. *Real Sound Synthesis for Interactive Applications*. A. K. Peters, Ltd., Natick, MA, USA, 2002.

[5] M. Kaltenbrunner, S. Jorda, G. Geiger, and M. Alonso. The reacTable*: A collaborative musical instrument. *Enabling Technologies, IEEE International Workshops on*, pages 406–411, 2006.

[6] C. W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, pages 25–34, New York, NY, USA, 1987. ACM.

[7] B. Schroeder, M. Ainger, and R. Parent. An Audiovisual Workspace for Physical Models. In *Proceedings of Sound and Music Computing 2010*, 2010.