INTRODUCTION
We have been exploring the use of the general-purpose high-performance computing capabilities of GPUs to perform sound synthesis using compute-intensive physics-based models in real-time. Until now, real-time synthesis using these models has not been practical using only CPUs. While others have used these physics-based models to generate audio, none have executed in real-time. Real-time sound synthesis using these physics-based models will allow the creation of new audio synthesizer instruments. Our proof-of-concept project discussed here shows that it is possible to use these compute-intensive models to generate sound in real-time using GPUs.

GENERATING AUDIO FROM A SYNTHESIZED MEMBRANE
To simulate a membrane, we use a finite-difference scheme, using a truncated second-order Taylor series expansion of the wave equation with dissipation in two dimensions (Equation 1).

\[ u^{n+1}_{i,j} = \left[ 1 + \Delta t \left( \frac{\rho \omega^2}{\Delta x^2} + \frac{\rho \omega^2}{\Delta y^2} \right) \right] u^n_{i,j} - \Delta t \left( \frac{\rho \omega^2}{\Delta x^2} u^n_{i+1,j} + \frac{\rho \omega^2}{\Delta y^2} u^n_{i,j+1} \right) - \Delta t \left( \frac{\rho \omega^2}{\Delta x^2} u^n_{i-1,j} + \frac{\rho \omega^2}{\Delta y^2} u^n_{i,j-1} \right) + \frac{\Delta t}{\Delta x^2} \left( \frac{\rho \omega^2}{\Delta y^2} u^n_{i,j+1} + \frac{\rho \omega^2}{\Delta x^2} u^n_{i,j+1} \right) - \frac{\Delta t}{\Delta y^2} \left( \frac{\rho \omega^2}{\Delta x^2} u^n_{i+1,j} + \frac{\rho \omega^2}{\Delta y^2} u^n_{i,j+1} \right). \]  

(1)

A point is selected on the membrane (Figure 1). The vertical displacement of this point is captured at regular intervals over time. The change in vertical displacement of the membrane then corresponds to the vertical displacement over time of an audio signal. Therefore the motion of the membrane over time determines the audio output. For computational efficiency, audio output samples for consecutive timesteps are buffered before playback.

For each time step, one grid must be updated. To be processed efficiently by the GPU, the membrane (grid) must be divided into tiles, each of which can then be processed independently on the GPU (Figure 2).

EXPERIMENTAL SETUP
We implemented our Finite Difference Synthesizer (FDS) software in C and C++ using CUDA. We tested FDS on a MacBook Pro with a 2.66 GHz Intel Core i7, with a GeForce GT 330M GPU running OS 10.6.6. Table 2 shows the grid and kernel buffer size configurations that were tested. Setup I held grid size constant while varying the kernel buffer size. Setup II held the kernel buffer size constant while varying the grid size. We ran and timed the following sequence five hundred times:
1. run the excitation kernel
2. check ring buffer space
3. perform the FD simulation
4. copy the FD simulation output to the ring buffer.
The timings were then averaged. The results are presented in Table 2 and Table 3.

EXPERIMENTAL RESULTS
Table 2 and Table 3 summarize our results. Buffer sizes of 8, 512, and 4096 samples correspond to audio output durations of 0.181 ms, 11.6 ms, and 92.8 ms at 44.1kHz. For the real-time audio tests, all kernel output buffer and grid configurations produced no audio output buffer underruns.

FINITE DERRERESE FOR REAL TIME AUDIO
To be considered useful as a real-time audio instrument, jitter and latency must be within acceptable limits. This is known as responsiveness.

Table 2. Results of varying buffer size with a constant grid size of 21x21 points.

<table>
<thead>
<tr>
<th>Grid Size (points)</th>
<th>Excitation Time (ms)</th>
<th>Finite Difference Time (ms)</th>
<th>Memory Transfer Time (ms)</th>
<th>Total Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15x15</td>
<td>0.03</td>
<td>34.37</td>
<td>0.03</td>
<td>34.37</td>
</tr>
<tr>
<td>18x18</td>
<td>0.03</td>
<td>31.81</td>
<td>0.03</td>
<td>31.87</td>
</tr>
<tr>
<td>21x21</td>
<td>0.03</td>
<td>34.73</td>
<td>0.03</td>
<td>34.73</td>
</tr>
</tbody>
</table>

Table 3. Results of varying grid size with a constant buffer size of 4096 samples.

Satisfactory percussive sounds were produced in qualitative testing. An experimental Open Sound Control (OSC) controller interface (Figure 5) running on an Apple iPad2 was used as the user input controller. It was found that the FDS’s output was especially sensitive to changes in the FD parameters η and p. Sample recordings of some of these tests are available at http://userwww.sfsu.edu/~whsu/FDGPU.

CONCLUSIONS
- It is possible to generate real-time audio using GPUs and finite-difference simulations.
- Larger grids better leverage GPU computing power.
- Choice of buffer and grid sizes is important to responsiveness.
- Memory bandwidth is not a major consideration, especially with more advanced graphics cards.
- By using GPUs it is possible to create a responsive, real-time audio synthesizer instrument using compute-intensive physics-based models.