An Audio Processing Library for Game Development in Flash

August 27th, 2009

Ray Migneco, Travis Doll, Jeff Scott, Youngmoo Kim, Christian Hahn and Paul Diefenbach
Motivation

• Recent popularity of music-based games
  ▸ Guitar Hero, Rock Band and Dance, Dance Revolution
  ▸ Console platforms offer:
    - Rich graphics
    - Innovative control interfaces
    - Tight synchronization with audio processing

• Music-based games add an unprecedented amount of culture to the gaming experience

• Limitations of these games?
  ▸ Most are bundled with pre-prepared tracks
    - What if you don’t like the music?
Motivation

• Rise in popularity of web-based games
  ▸ Wide availability of broadband connections
  ▸ Improved client processing power

• Adobe Flash
  ▸ Allows for rapid game development and deployment
  ▸ Cross-platform support
  ▸ Accommodates programmers of many skill levels
Music-centric Flash games

• **Super Crazy Guitar Maniac 2**
  - “Guitar Hero”-style game
    - Players press keys in response to derived beats

• **JamLegend**
  - Players can upload their own tracks
  - Limited audio processing
    - No control over instrument sound when notes “miss”

• **Music in Motion**
  - Side scrolling game: obstacles generated in response to music
  - Audio tracks are fixed and levels are preprocessed

• ...many more
Limiting factors

- Audio support
  - Previous versions restricted to just audio clip playback
  - Dynamic, buffer-based audio was recently added (ver 10)

- Processing limitations
  - ActionScript was not intended for computationally intensive algorithms
  - Some existing Flash-audio libraries authored in AS:
    - StandingWave - sound generation library
    - SoundTouch - time compression/expansion, pitch transposition
    - ActionScript Math Library - contains FFT functions
• What is Alchemy?
  ▶ Allows C/C++ libraries to be integrated into Flash projects
    - C/C++ code is compiled to byte code, optimized for the ActionScript Virtual Machine (AVM2)
    - Minimal performance degradation on AVM2 from native compilation
  ▶ Alchemy compiler generates a .swc file from C/C++ code, which is integrated into the Flash project
    - .swc : an archive file containing components and resources representing the C/C++ library
Alchemy performance

FFT Computation Times for Web-based platforms (msec)

<table>
<thead>
<tr>
<th>Target</th>
<th>FFT Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16384</td>
</tr>
</tbody>
</table>

- FFT computation time
  - Core function in DSP analysis/synthesis algorithms
  - 10,000 iterations for each size on each platform
  - Average elapsed times reported
Alchemy performance

<table>
<thead>
<tr>
<th>Target</th>
<th>FFT Size</th>
<th>16384</th>
<th>8192</th>
<th>4096</th>
<th>2048</th>
<th>1024</th>
<th>512</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActionScript (AS2 Math)</td>
<td></td>
<td>96.377</td>
<td>45.157</td>
<td>20.818</td>
<td>9.276</td>
<td>4.460</td>
<td>2.041</td>
<td>0.925</td>
</tr>
</tbody>
</table>

- **FFT computation time**
  - Core function in DSP analysis/synthesis algorithms
  - 10,000 iterations for each size on each platform
  - Average elapsed times reported
## Alchemy performance

### FFT Computation Times for Web-based platforms (msec)

<table>
<thead>
<tr>
<th>Target</th>
<th>FFT Size</th>
<th>16384</th>
<th>8192</th>
<th>4096</th>
<th>2048</th>
<th>1024</th>
<th>512</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActionScript (AS2 Math)</td>
<td>96.377</td>
<td>45.157</td>
<td>20.818</td>
<td>9.276</td>
<td>4.460</td>
<td>2.041</td>
<td>0.925</td>
<td></td>
</tr>
<tr>
<td>Java (JTransforms)</td>
<td>29.517</td>
<td>20.703</td>
<td>9.393</td>
<td>4.345</td>
<td>1.956</td>
<td>0.901</td>
<td>0.385</td>
<td></td>
</tr>
</tbody>
</table>

- **FFT computation time**
  - Core function in DSP analysis/synthesis algorithms
  - 10,000 iterations for each size on each platform
  - Average elapsed times reported
# Alchemy performance

<table>
<thead>
<tr>
<th>Target</th>
<th>FFT Size</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ActionScript (AS2 Math)</td>
<td>16384</td>
<td>8192</td>
<td>4096</td>
<td>2048</td>
<td>1024</td>
<td>512</td>
<td>256</td>
</tr>
<tr>
<td></td>
<td>96.377</td>
<td>45.157</td>
<td>20.818</td>
<td>9.276</td>
<td>4.460</td>
<td>2.041</td>
<td>0.925</td>
</tr>
<tr>
<td>Java (JTransforms)</td>
<td>29.517</td>
<td>20.703</td>
<td>9.393</td>
<td>4.345</td>
<td>1.956</td>
<td>0.901</td>
<td>0.385</td>
</tr>
<tr>
<td>C (Alchemy)</td>
<td>3.009</td>
<td>1.371</td>
<td>0.628</td>
<td>0.297</td>
<td>0.139</td>
<td>0.067</td>
<td>0.034</td>
</tr>
</tbody>
</table>

- **FFT computation time**
  - Core function in DSP analysis/synthesis algorithms
  - 10,000 iterations for each size on each platform
  - Average elapsed times reported
The goal

- A fast, versatile and open source audio processing library for Flash game development

- Requirements:
  - Optimized for maximum performance
  - Flexibility for developers
  - Ease of implementation

- The result:
  - ALF - Audio processing Library for Flash
Alchemy is used to compile a .swc for a C/C++ DSP-based library

DSP Audio Toolkit for Flash (DATF)
  - AS3-based wrapper for .swc file

ALF is the top layer of abstraction
  - AS3 wrapper for DATF

Open-source flexibility at all levels

Varying developer interaction at each level
Alchemy is used to compile a .swc for a C/C++ DSP-based library

DSP Audio Toolkit for Flash (DATF)
  - AS3-based wrapper for .swc file

ALF is the top layer of abstraction
  - AS3 wrapper for DATF

Open-source flexibility at all levels

Varying developer interaction at each level
Alchemy is used to compile a .swc for a C/C++ DSP-based library

DSP Audio Toolkit for Flash (DATF)

- AS3-based wrapper for .swc file
Alchemy is used to compile a .swc for a C/C++ DSP-based library

DSP Audio Toolkit for Flash (DATF)
  - AS3-based wrapper for .swc file

ALF is the top layer of abstraction
  - AS3 wrapper for DATF

Open-source flexibility at all levels

Varying developer interaction at each level
Alchemy Layer

Alchemy DSP Library
Alchemy Layer

- Foundation of audio processing library
  - Alchemy compiled C/C++ code

- Contains:
  - Audio buffers
  - DSP library functions

- Developer responsibilities
  - Edit/add audio processing functions in C/C++
  - Compile .swc file and integrate
DATF Layer

Alchemy DSP Library
DATF Layer

- DSP Audio Toolkit for Flash (DATF)
  - AS3-based code

- Interfaces with Alchemy DSP Library
  - Establishes shared AS3/Alchemy memory

- Developer responsibilities
  - Allocate audio buffers
  - Read and write buffers
  - Call DATF library functions
ALF Layer

- AS3-based code

- Developer specifies:
  - Audio file for analysis/playback
  - Desired ALF functions to use
  - Audio processing rate

- Simple interfacing
  - No DSP knowledge required
  - No memory management
  - No direct access to audio buffers
Audio Support

• Flash 10 limitations
  ▸ Compatible with .mp3 and .wav file types
  ▸ 22.05 kHz and 44.1 kHz sample rates

• Event listener is used to update the application when an audio frame has been computed
Available functions

- getSpectrum
- getIntensity
- getBandwidth
- getFlux
- getBrightness
- getHarmonics
- filter
- reverb
Available functions

- getSpectrum
- getIntensity
- getBandwidth
- getFlux
- getBrightness
- getHarmonics
- filter
- reverb
• Players upload .mp3 tracks from their music library
• Game environment is rendered in real-time based on game audio
  ▪ Features are extracted using ALF and mapped to game parameters
• Development
  ▪ Programmed and design with Adobe CS4 components
  ▪ Deployed via Adobe Air 1.5: Flash for the Desktop
    - Avoids network latency for file uploads
Audio-driven environment
Audio-driven environment

Pulse

SICK MOVES!!

SCORE: 4360

Dream Theater - 3 Minutes of Tuscany

Tuesday, October 6, 2009
Audio-driven environment

Platform Slope:
getFlux
Audio-driven environment

Enemies: getIntensity

SCORE: 4360

Dream Theater - 3 Minutes of Tuscany
Audio-driven environment

Cloud Color: getCentroid

Tuesday, October 6, 2009
Audio-driven environment

Cloud Size:
getBandwidth

Tuesday, October 6, 2009
Pulse 2 Demo
Conclusions

• Developed an efficient, open source, high-level interface for audio processing in Flash applications
  ▪ Utilized computational benefits of Alchemy
  ▪ Accommodates developers of varying skill levels
  ▪ Capable of real-time processing

• Future work:
  ▪ Expand the functional palette of ALF
    - Phase vocoder for time and/or pitch-scale modification
    - Additive synthesis methods for sound effect generation
    - Beat tracker
More info and downloads

• http://music.ece.drexel.edu

• Game links:
  ‣ Pulse 2
  ‣ Educational activities: Tone Bender and Hide & Speak

• ALF
  ‣ Preview documentation
  ‣ Sources coming soon
Questions?
• show the game Architecture using ALF

• utilized functions, etc;
  ▸ getFlux, getIntensity, getCentroid, getBandwidth