Hot Topics in Musical Acoustics Applied to Real-Time Sound Synthesis

Julius Smith
CCRMA, Stanford University

Acoustical Society of America 166th Meeting

Hot Topics Session

December 4, 2013

Outline

New Enabling Technologies

1. Smart-Phones and Tablets
   - High-quality audio in (mono) and out (stereo)
   - Fast multicore processors (exponentially growing speed)
   - Multitouch display screens (5 for iPhone, 11 for iPad)
2. Domain-Specific Languages
   - Functional AUdio STream (FAUST) — High-Level Signal-Processing Language (Yann Orlarey, GRAME)
   - Synth-A-Modeler — Physical Model "Diagram" to FAUST (Edgar Berdahl)
3. New Convex Optimization Techniques and Formulations (Esteban Maestre)

Smart-Phone/Tablet Example: moForte Guitar

moForte Guitar

Real time on the iPhone 4S (all written in FAUST):

- Full electric-guitar + effects:

Six vibrating strings —	general excitations
Distortion	Feedback
Compression	Wah pedal or Autowah
Phaser	Flanger
Five-band parametric equalizer	Reverb

- Responds to
  accelerometer, gyros, touches (plucks), swipes (strumming), . . .
- It is challenging to fully utilize *five* points of multitouch on the iPhone and *eleven* on the iPad!

moForte Guitar

Real time on the iPhone 4S (all written in FAUST):

- Full electric-guitar + effects:

Six vibrating strings —	general excitations
Distortion	Feedback
Compression	Wah pedal or Autowah
Phaser	Flanger
Five-band parametric equalizer	Reverb

- Responds to
  accelerometer, gyros, touches (plucks), swipes (strumming), ...
- It is challenging to fully utilize *five* points of multitouch on the iPhone and *eleven* on the iPad!

moForte Guitar

Real time on the iPhone 4S (all written in FAUST):

- Full electric-guitar + effects:

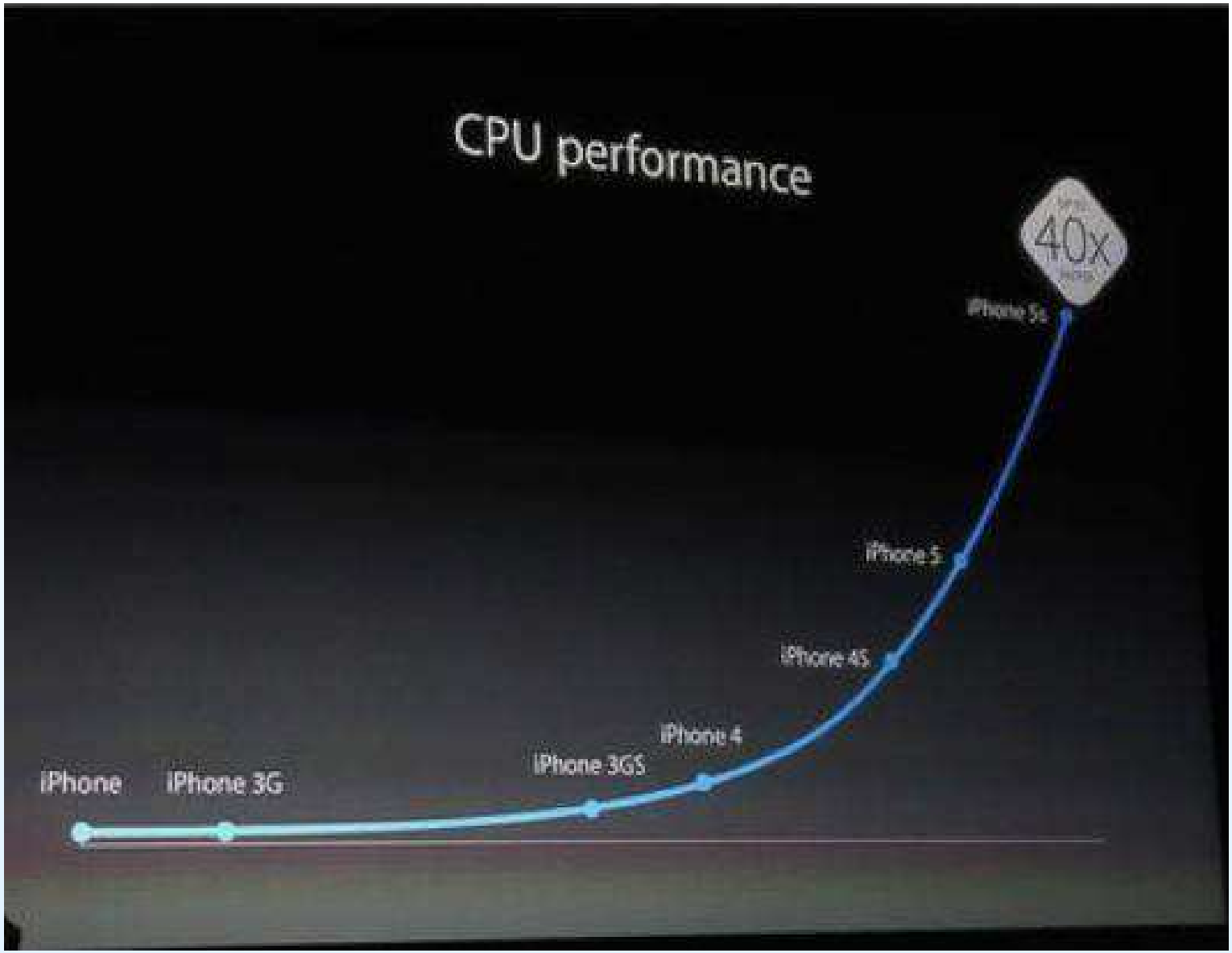
Six vibrating strings —	general excitations
Distortion	Feedback
Compression	Wah pedal or Autowah
Phaser	Flanger
Five-band parametric equalizer	Reverb

- Responds to
  accelerometer, gyros, touches (plucks), swipes (strumming),
  ...

- It is challenging to fully utilize *five* points of multitouch on the iPhone and *eleven* on the iPad!

Effects Running in Real Time on the iPhone 4S with All Six Strings Playing

iPhone CPU Performance



Distortion Guitar Sound Examples

- moForte Guitar Demo Video
- Distortion Guitar: (WAV) (MP3)
- Amplifier Feedback 1: (WAV) (MP3)
- Amplifier Feedback 2: (WAV) (MP3)

FAUST Introduction

FAUST Language

Short FAUST Program Examples:

```
process = _ ;
process = + ;
process = _,_ : + : _;
process = pole(0.9) with { pole(p) = + ~ *(p); };
```

Partial C++ Output for Last Example Above:

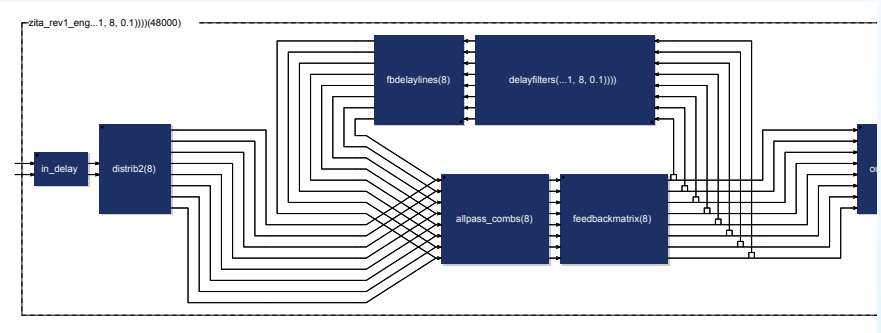
```
virtual void compute (int count, FAUSTFLOAT** input,
                                 FAUSTFLOAT** output)
{
  FAUSTFLOAT* input0 = input[0];
  FAUSTFLOAT* output0 = output[0];
  for (int i=0; i<count; i++) {
    fRec0[0] = ((float)input0[i] + (0.9f * fRec0[1]));
    output0[i] = (FAUSTFLOAT)fRec0[0];
    // post processing
    fRec0[1] = fRec0[0];
  }
}
```

FAUST Example: zita_rev1.dsp

Source Code (FAUSTeffect.lib):

```
zita_rev_fdn(f1,f2,t60dc,t60m,fsmax) =
   ((bus(2*N) :> allpass_combs(N) : feedbackmatrix(N)) ~
   (delayfilters(N,freqs,durs) : fbdelaylines(N))) ...
```

Block Diagram (drawn by the FAUST compiler):



JACK-Aware Standalone App (generated from FAUST source):

FAUST Main Programs Supported

Shell Script	Arch. File	Description
faust2jack (f2j)	jack-gtk.cpp	JACK GTK standalone application
faust2jaqt	jack-qt.cpp	JACK QT4 standalone application
faust2caqt (f2ca)	ca-qt.cpp	CoreAudio QT4 standalone application
faust2pa	pa-gtk.cpp	PortAudio GTK standalone application
faust2paqt	pa-qt.cpp	PortAudio QT4 standalone application
faust2netjackqt	netjack-qt.cpp	server with libnetjack support
faust2oss	oss-gtk.cpp	OSS GTK standalone application
faust2alsa	alsa-gtk.cpp	ALSA GTK standalone application
faust2alqt	alsa-qt.cpp	ALSA QT4 standalone application
faust2alsaconsole	alsa-console.cpp	ALSA terminal program
faust2android	android.cpp	Android phone/tablet application
faust2ios	ios-coreaudio[-jack].cpp	iOS phone/tablet application
faust2rpi*	[alsa—netjack]-console.cpp	Raspberry Pi application
	sndfile.cpp	sound file transformation command
	bench.cpp	speed benchmark
faust2octave (f2o)	octave.cpp	output signals to Octave (matlab)

FAUST Plugin Architectures Supported

Shell Script	Arch. File	Description
faust2ladspa	ladspa.cpp	Linux LADSPA effect plugin
faust2dssi	dssi.cpp	Linux synth plugin, extending LADSPAA
faust2lv2	lv2.cpp	Linux synth plugin, newer lv2 type
faust2lv2-synth	lv2synth.cpp	Linux synth plugin, newer lv2 type
faust2vst	vst2p4.cpp	VST 2.4 plugin
faust2vsti	vsti-poly.cpp	VSTi instrument
faust2au (f2au)	au-effect.cpp	Apple Audio Unit (AU) effect plugin
faust2au	au-instrument.cpp	Apple AU instrument plugin
faust2max	max-msp.cpp	Max/MSP external
faust2puredata	puredata.cpp	PD external
faust2supercollider	supercollider.cpp	SuperCollider Unit Generator
	snd-rt-gtk.cpp	Snd-RT music programming language
faust2q	q.cpp	Q language plugin
faust2csound	csound.cpp	CSOUND opcode
faust2alchemy	alchemy-as.cpp	Flash/ActionScript plugin (for Web browsers)

Getting Started with FAUST

- Faust Website: `http://faust.grame.fr/`
  (Click on "Online Examples" and type in some FAUST code!)
- Faust Intro:
  `https://ccrma.stanford.edu/~jos/aspf/`
  (Google Search: "Audio Signal Processing in Faust")
- FAUST is Free Open Source Software (FOSS) for
  Mac OS, Linux, and Windows

Ongoing Research

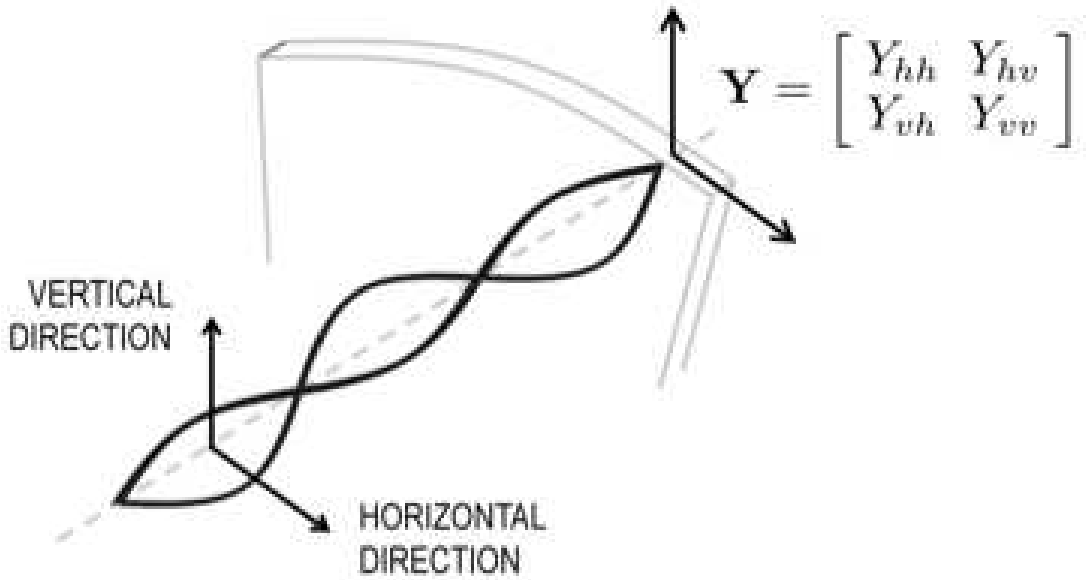
Synth-A-Modeler Block-Diagram to FAUST Translator

- FAUST is excellent for specifying platform-independent signal-processing block diagrams
- For *physical models* a higher-level front-end is helpful
- Physical objects need *bidirectional* connections
- Linux Audio Conference 2012 (LAC-12) Paper:

  `http://lac.linuxaudio.org/2012/papers/34.pdf`
  "An Introduction to the Synth-A-Modeler Compiler: Modular and Open-Source Sound Synthesis using Physical Models," Edgar Berdahl et al.
- GPL Free Software:
  `git clone https://github.com/eberdahl/SaM.git`

2D Bridge Modeling for Bowed Strings

Reference: E. Maestre, G. P. Scavone, and J. O. Smith III, "Digital modeling of bridge driving-point admittances from measurements on violin-family instruments", in Proceedings of the Stockholm Musical Acoustics Conference (SMAC-13), http://www.speech.kth.se/smac-smc-2013/.



Two-dimensional Bridge Model

Scattering Delay Networks (SDN)

"Scattering Delay Networks" by **Enzo De Sena**,
Hüseyin Hacıhabiboğlu, Zoran Cvetković, and Julius O. Smith III