pd-faust: An integrated environment for running Faust objects in Pd

Albert Gräf
Department of Music Informatics
Johannes Gutenberg University Mainz

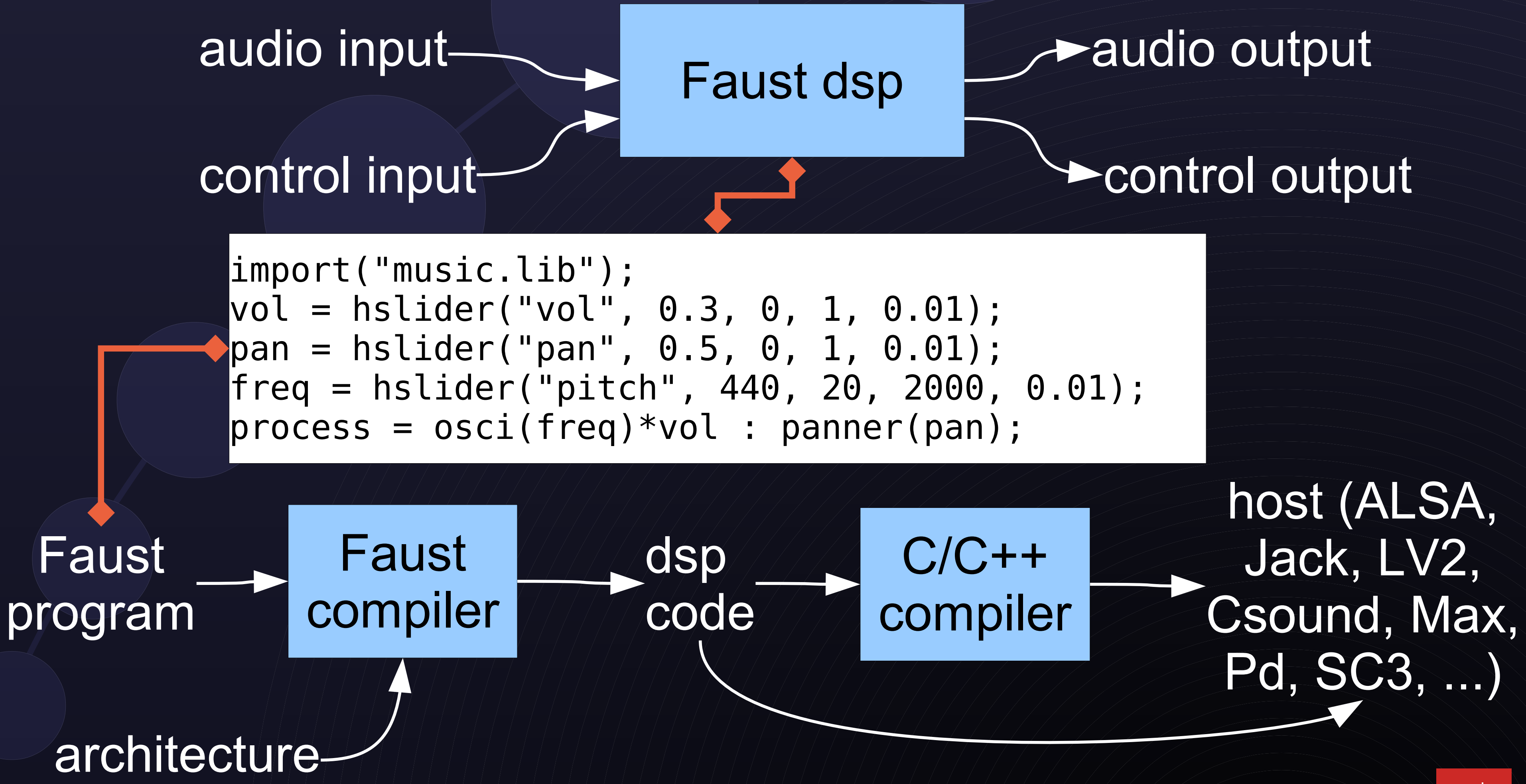
Contents

- Brief introduction to Faust
- Old Pd-Faust interface
- New interface
- Implementation
- Demo

Learn more:

Pure and Faust: Functional Programming for Media Applications, Tue, 04/17/2012 - 5:15pm – 6:30pm, CCRMA Classroom [Knoll 217]

Faust



audio input

control input

Faust dsp

audio output

control output

```
import("music.lib");
vol = hslider("vol", 0.3, 0, 1, 0.01);
pan = hslider("pan", 0.5, 0, 1, 0.01);
freq = hslider("pitch", 440, 20, 2000, 0.01);
process = osci(freq)*vol : panner(pan);
```

Faust program

Faust compiler

architecture

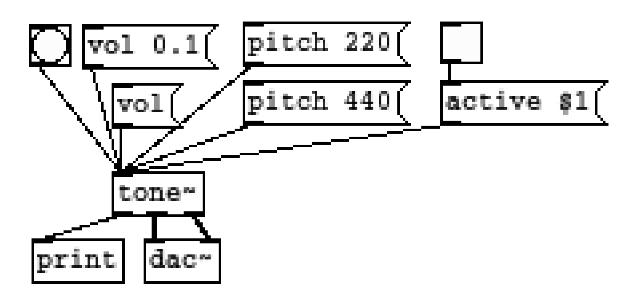
dsp code

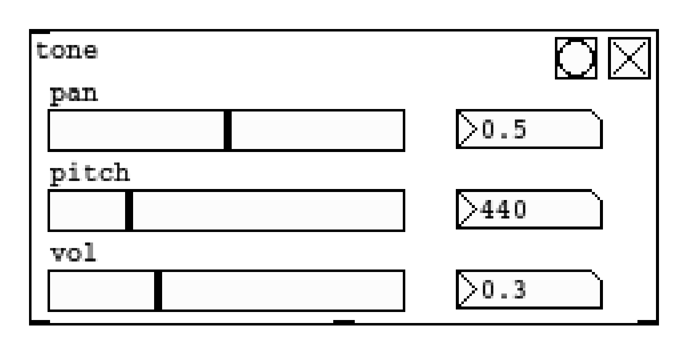
C/C++ compiler

host (ALSA, Jack, LV2, Csound, Max, Pd, SC3, ...)

Old Pd-Faust interface

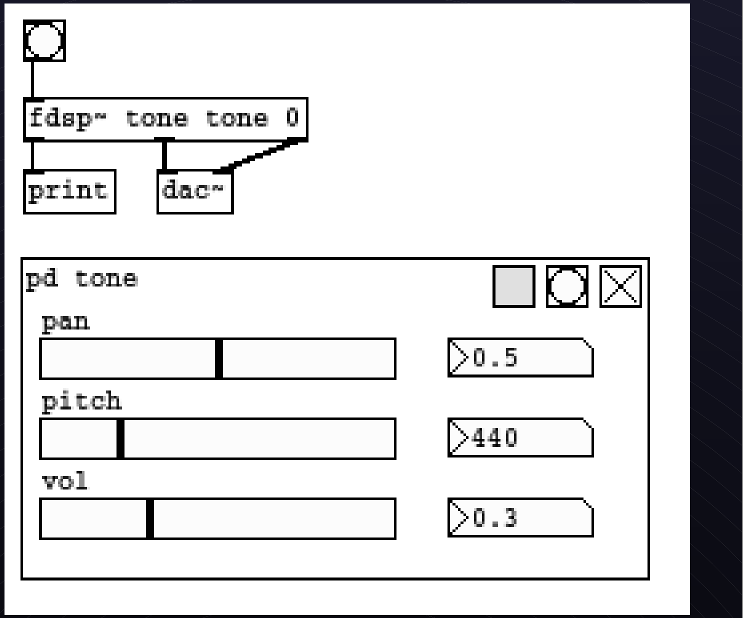
- **puredata.cpp** architecture turns Faust dsp into Pd external
- **faust2pd** script generates Pd GUI for the control variables
- **static:** GUI must be regenerated and patch reloaded after changes





New Pd-Faust interface

- **dynamic**: Faust dsp can be reloaded while patch is running, GUI gets regenerated instantly
- **idea:** leverage Pure-Faust interface [LAC 2011] which is capable of hot-swapping Faust modules
- use Pd's **FUDI** protocol to create GUI inside subpatch



```
fdsp~ tone tone 0
print
dac~
```

pd tone

pan 0.5

pitch 440

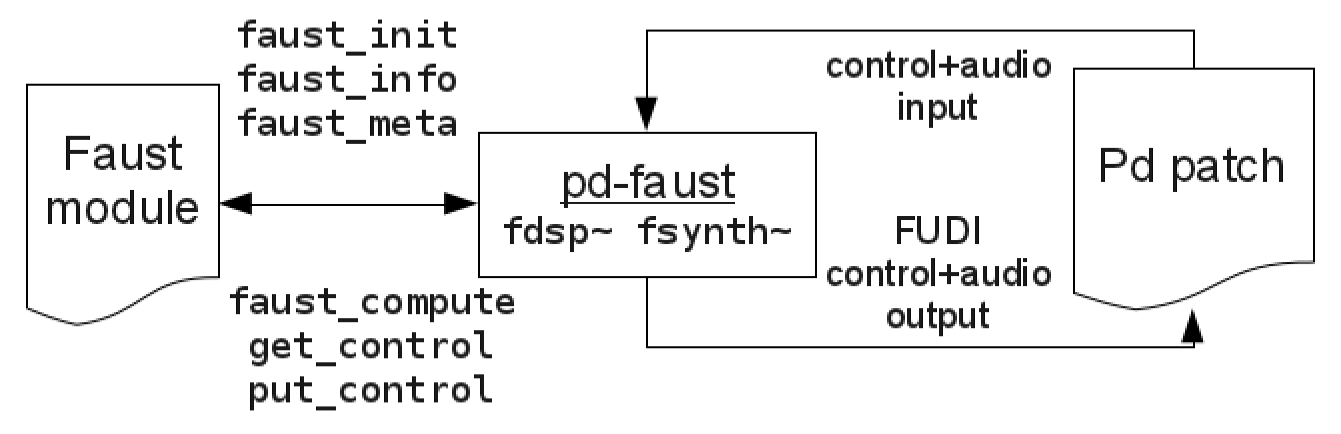
vol 0.3

Additional goodies

- all control logic implemented in Pure
- both native Faust modules (.so) and LLVM bitcode modules (.bc) are supported
- more options to configure the GUI layout
- built-in MIDI playback, MMC sync
- MIDI and OSC controller mappings
- automation
- livecoding

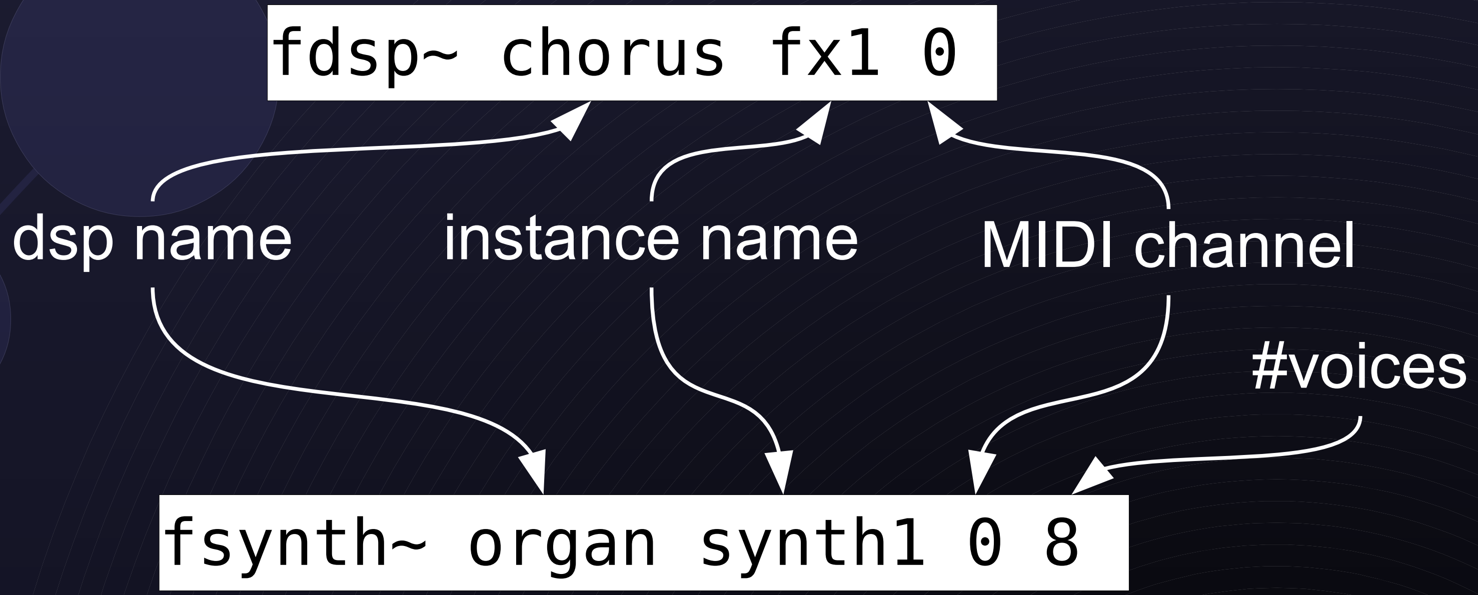
Implementation

- pd-faust: written in Pure, gets compiled to native library of Pd externals (pd -lib)
- requires pd-pure (Pd Pure script loader) and pure-faust (Pure-Faust interface)
- fdsp~ and fsynth~ objects for effect and polyphonic synth units



fdsp~ and fsynth~ objects

inlets: 1 for control input, 1 per audio input channel



```
fdsp~ chorus fx1 0
```

dsp name

instance name

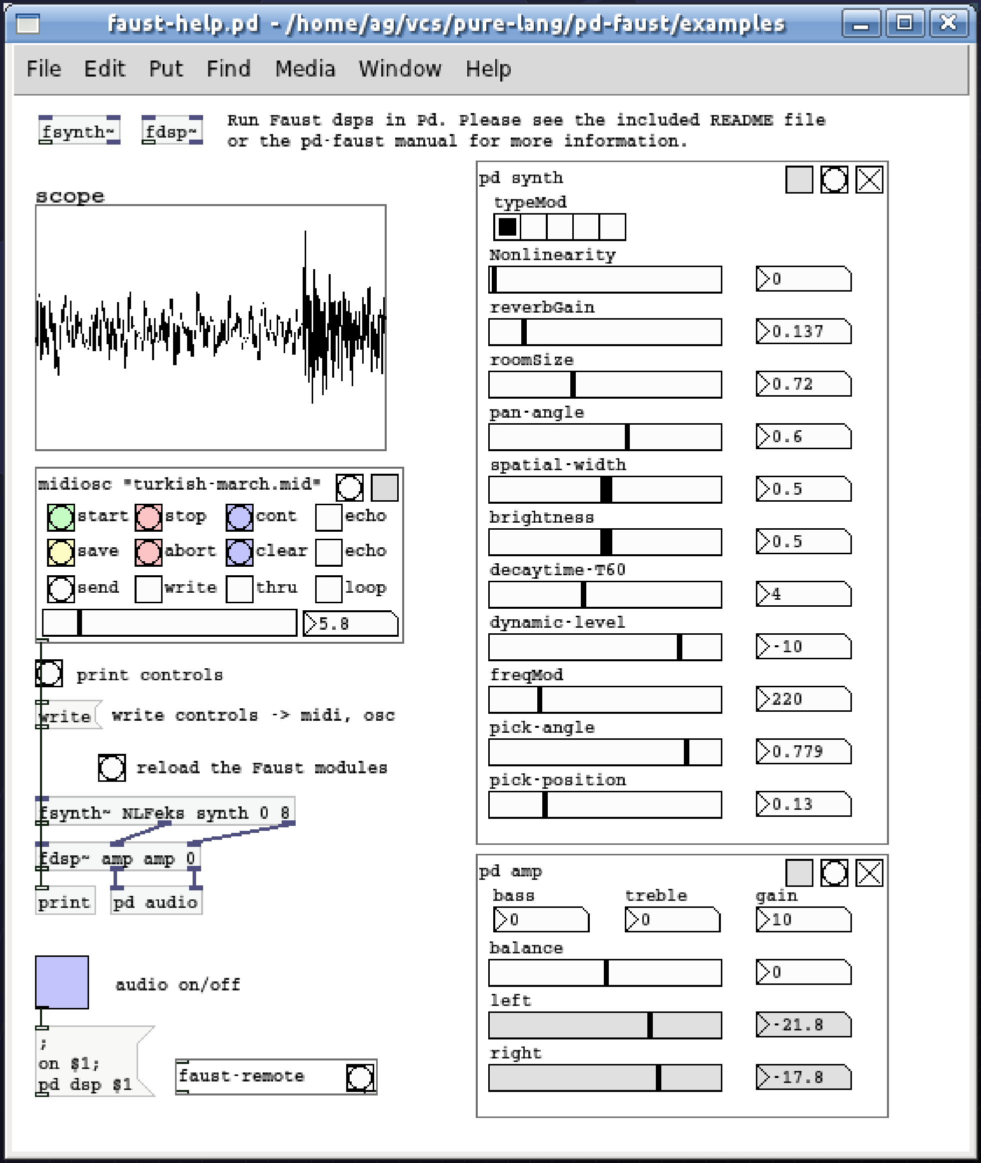
MIDI channel

#voices

```
fsynth~ organ synth1 0 8
```

outlets: 1 for control output, 1 per audio output channel

Demo


faust-help.pd - /home/ag/vcs/pure-lang/pd-faust/examples
File Edit Put Find Media Window Help
fsynth~
fdsp~
scope

Run Faust dsps in Pd. Please see the included README file or the pd-faust manual for more information.

```
midiosc "turkish-march.mid"
start  stop   cont   echo
save   abort  clear  echo
send   write  thru   loop
5.8

print controls
write   write controls -> midi, osc
reload the Faust modules
fsynth~ NLFeks synth 0 8
fdsp~ amp amp 0
print   pd audio
audio on/off
;
on $1;
pd dsp $1
faust-remote
```

```
pd synth
typeMod
Nonlinearity      0
reverbGain        0.137
roomSize          0.72
pan-angle         0.6
spatial-width     0.5
brightness        0.5
decaytime-T60     4
dynamic-level     -10
freqMod           220
pick-angle        0.779
pick-position     0.13

pd amp
bass   treble   gain
0      0        10
balance           0
left              -21.8
right             -17.8
```

Future Work

- Better OSC sequencing (preferably via external DAW/sequencer software such as Ardour3 or Qtractor)
- Port pd-faust to other environments (Jack, LV2, SuperCollider...)