Audio/Visual Experiments with Python

Radio Free Quasar and Ergo

Tim Thompson
tjt@nosuch.com

Outline

- Radio Free Quasar = audio experiments
- Ergo = MIDI-driven visual experiments
- Hoops = Looping MIDI to drive Ergo

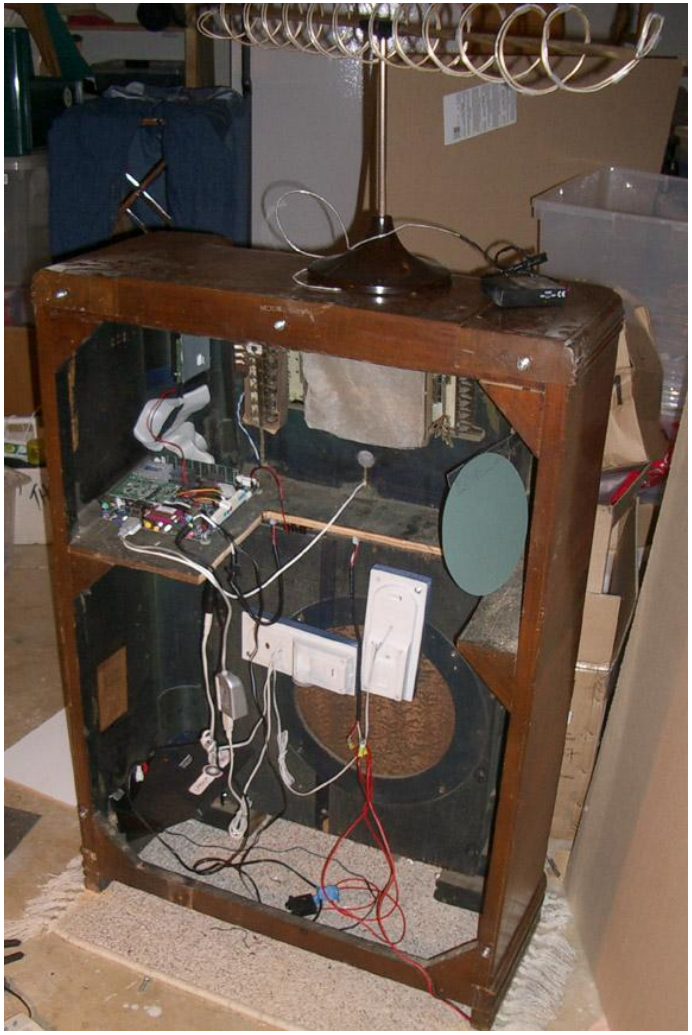
Intentions

- Intended to demonstrate:
  - Fun and interesting apps
  - Ease of developing with Python
  - Ease of using pyrex to integrate with C
  - Combination with other languages
- Not intended to demonstrate:
  - Sophisticated Python
  - Particularly good code or object design

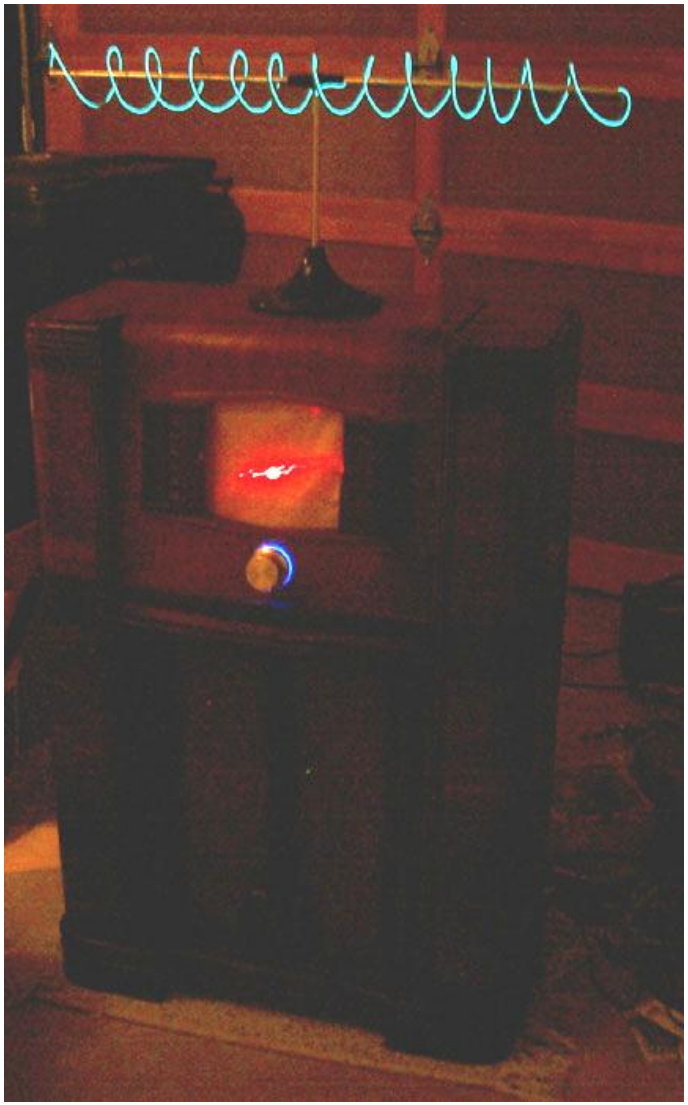
Radio Free Quasar

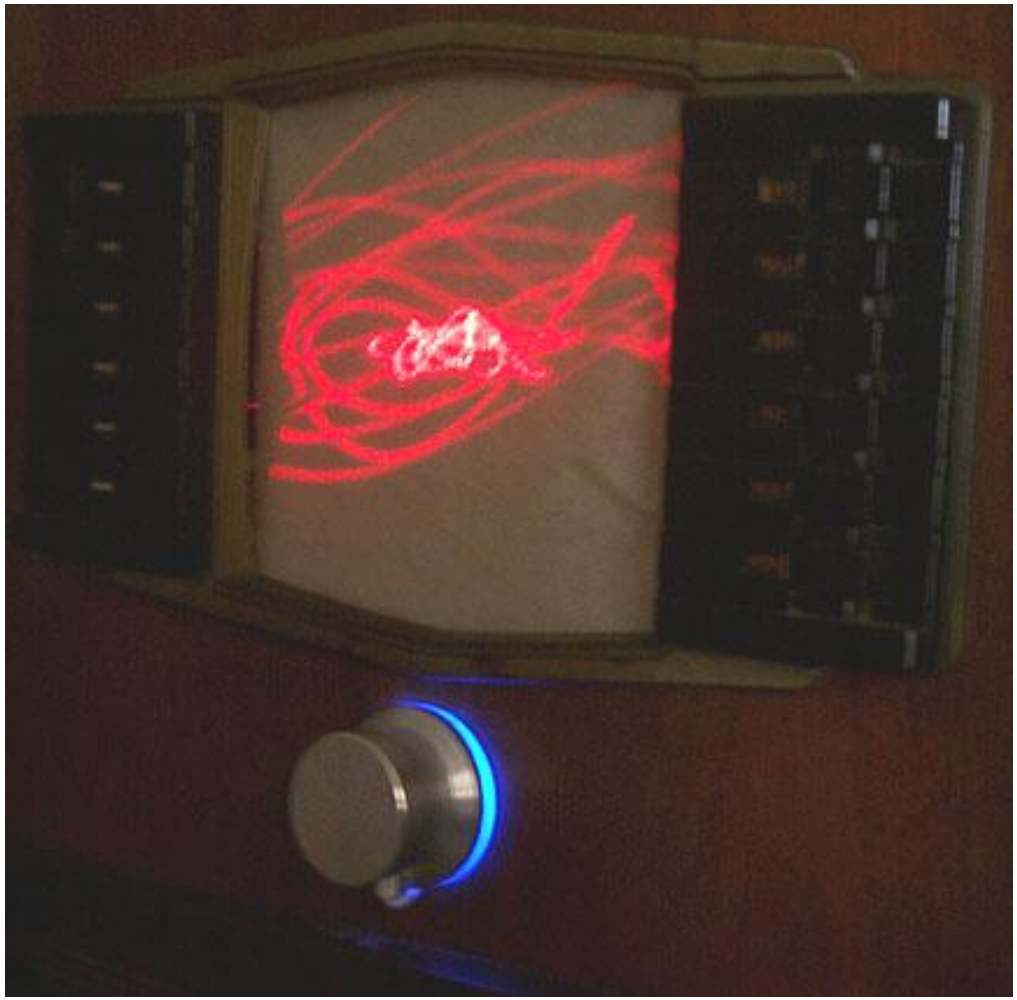
- Art installation for Burning Man 2004
- Antique radio
- Computer generating audio
- Laser generating graphics
- Big knob for control

Radio Free Quasar

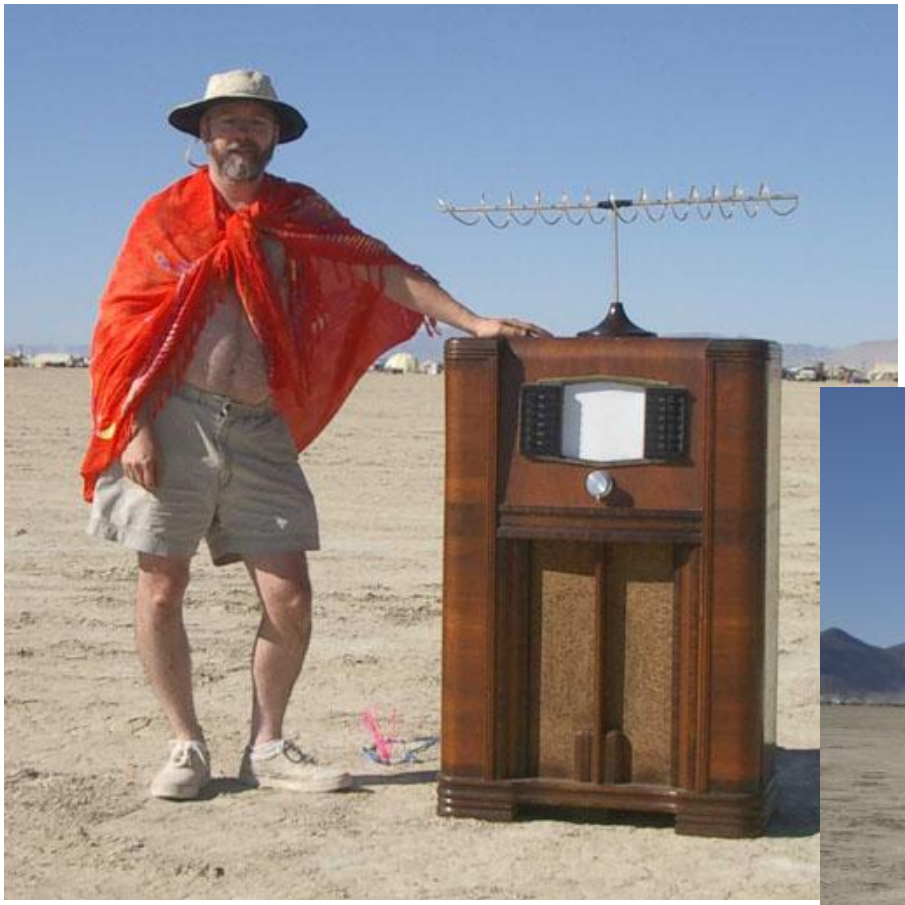


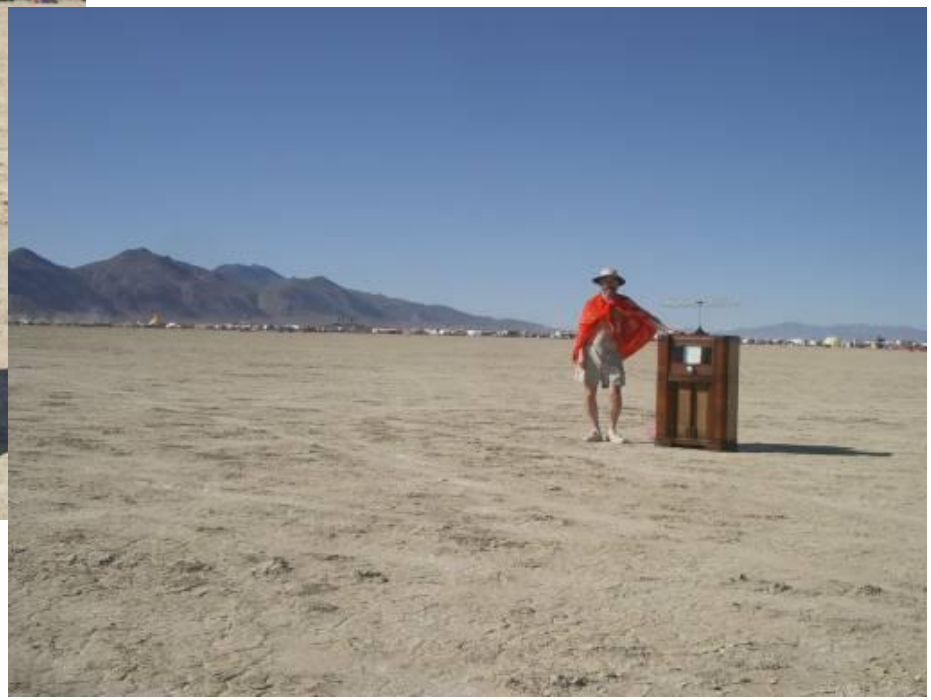
Radio Free Quasar



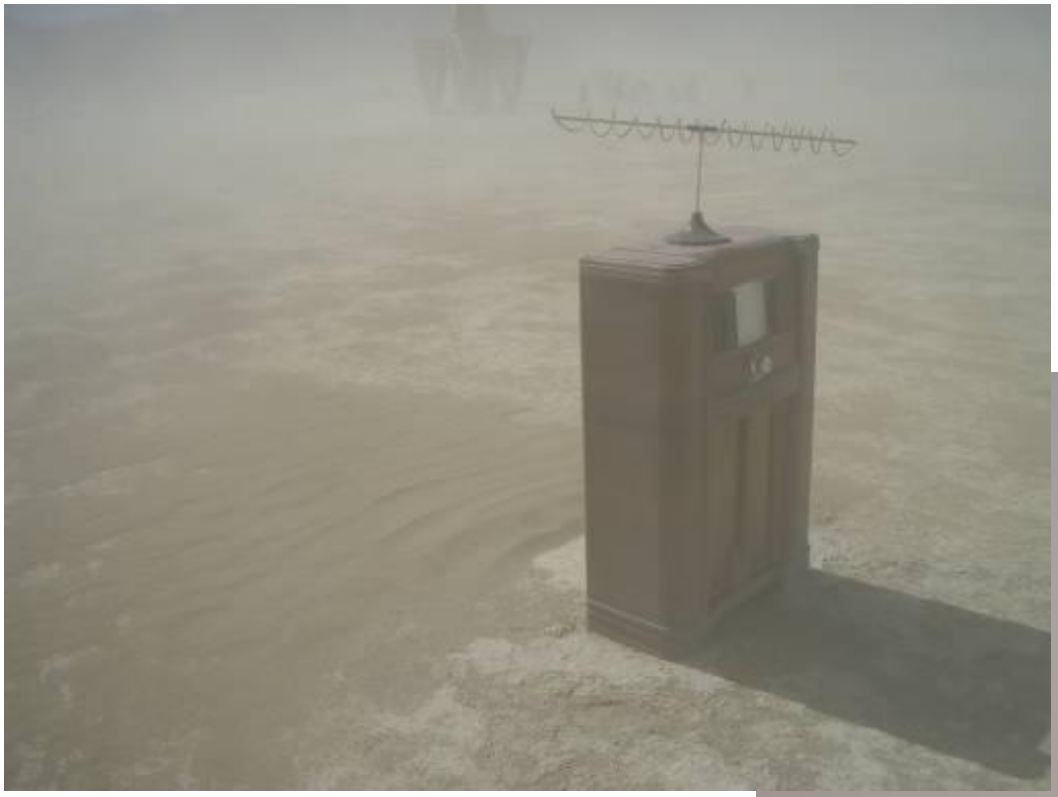


Radio Free Quasar at Burning Man

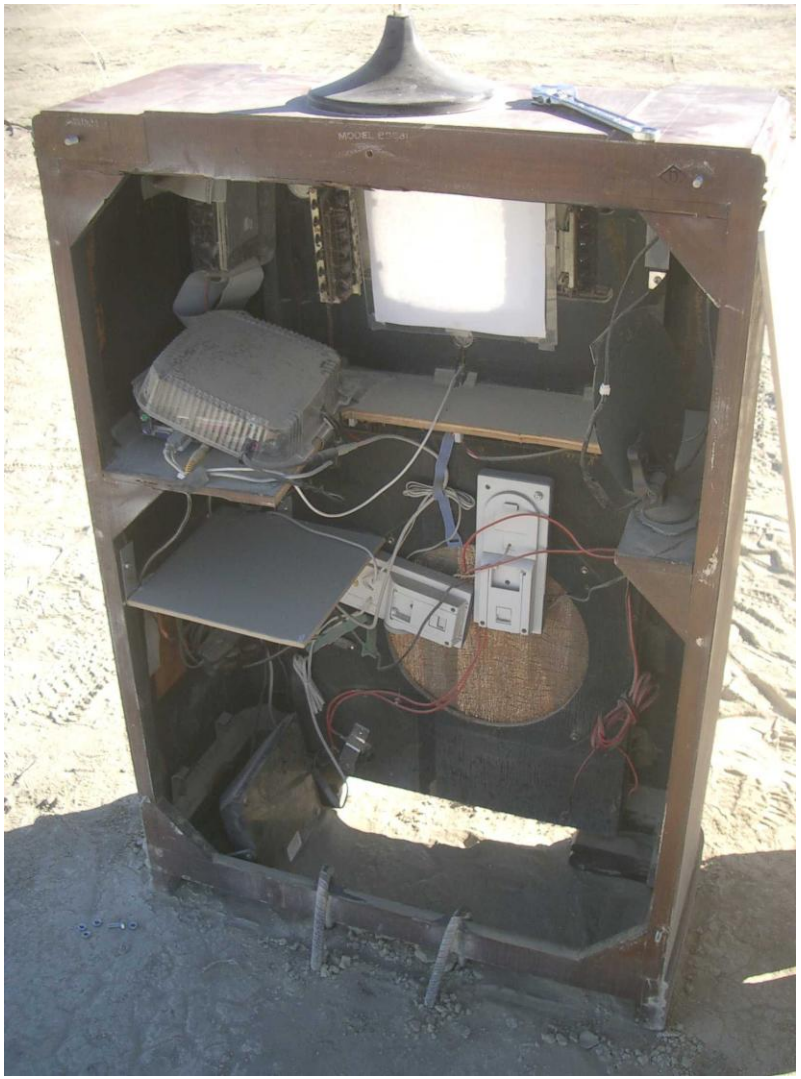


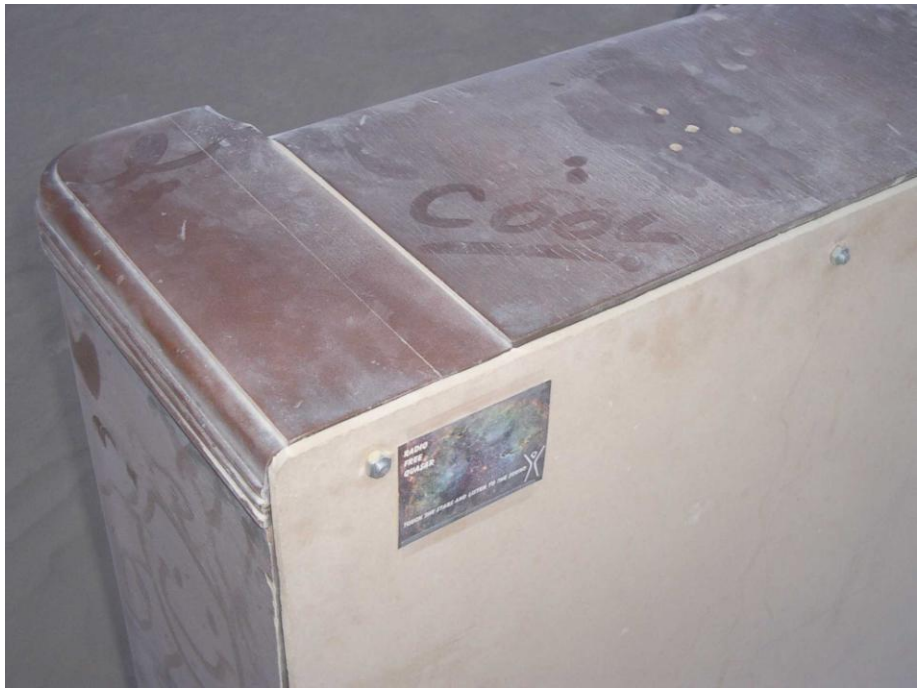


Radio Free Quasar at Burning Man



Radio Free Quasar at Burning Man





Radio Free Quasar at Burning Man

But what does it sound like?

Radio Free Quasar – hardware

- Windows XP
- Mini-ITX motherboard (fanless, low power)
- DC power supply
- Power sequencer
- USB-powered speakers
- USB knob (Griffin Powermate)
- Laservibe laser (driven by computer's audio output)
- EL-wire antenna

Radio Free Quasar – software components

- Python
- VST plugins
- Pyrex-based VST wrapper
- Pyrex-based Portaudio wrapper

Radio Free Quasar – example code 1

```
#!/usr/bin/env python
import time
import snip
import sys
a = snip.snipaudiodevice()
sound1 = snip.snippet(filename="winsound1.wav")
loop1 = snip.sniploop(sound1)
a.open()
a.start()
a.attach(loop1)
time.sleep(10)
a.stop()
a.close()
sys.exit(0)
```

Radio Free Quasar – example code 2

```
# Load a VST plugin and randomize its parameters

ring = snip.snipvst(dll=vstdir+"\\BJ Ringmodulator.dll")
for i in range(ring.numParams()):
        ring.set_param(i,random.random())
```

Radio Free Quasar – example code 3

```
ring = snip.snipvst(dll=vstdir+"\\BJ Ringmodulator.dll")
vsyn = snip.snipvst(dll=vstdir+"\\StrataVar.dll")

# Connect output of vsyn VST to ring modulator VST
ring.setinput(vsyn)

# Open audio output and connect ring modulator output
a.open()
a.start()
a.attach(ring)

# Send random MIDI notes to vsyn VST
# while randomizing parameters of both VSTs
for i in range(100):
   time.sleep(2.0)
   pitch = int(random.random() * 128) % 128
   vstrandparam(vsyn)
   vstrandparam(ring)
   vsyn.send_midi(1,pitch,8000,80)
```

Radio Free Quasar – example code 4

```
def vstrandparam(v):
   for i in range(v.numParams()):
       f = random.random()
       v.set_param(i,f)

def vstinfo(v):
   print "Is VST2 = ",v.is_vst2()
   print "Is synth = ",v.is_synth()
   print "numParams = ",v.numParams()
   print "numInputs = ",v.numInputs()
   print "numOutputs = ",v.numOutputs()
   print "can_receive_midi = ",v.can_receive_midi()
   print "can_send_midi = ",v.can_send_midi()
```

Radio Free Quasar – final program

- Collection of WAV files
- Ten robust VST plugins
- Python program:
  - selects wave files
  - enables/disables/randomizes VST plugins
  - allows interactive control from keyboard
- Big knob on radio sends keypresses
- Automatic randomization if no user input

Radio Free Quasar – interactive control

r	: randomize wave file and plugin parameters
w	: change to a different wave file
0-9	: randomize parameters of plugin N
d	: disable all plugins
e	: enable all plugins
s	: save current parameters
p	: select previously saved parameters

Ergo

E – Events

R – Routed to

G – Graphical

O – Objects

Ergo

- Used in dud, a multimedia improvised art ensemble
- **`http://dudland.com`**
- Drummer uses DrumKAT drums
- Drum MIDI output processed by KeyKit program
- Events sent to Python over TCP/IP socket
- Python does OpenGL graphics in response
- Sliders and buttons (from a MIDI controller) affect what kind of graphics are generated

Ergo – interactive control

- Sliders:
  - object's initial width/height, final width/height
  - speed in x/y direction, speed of rotation
  - fade time
  - initial x/y position
  - velocity threshold
- Buttons:
  - color shift, color randomizing
  - initial position control
  - sprite enable/disable
  - patch control

Ergo – sprites

- vector (horizontal and vertical)
- triangle
- square (hollow and filled)
- bar (horizontal and vertical)
- image (selected interactively)
- text (entered interactively)

Ergo – image control

- Wanted images related to content of unpredictable musical improvisation
- Solution = library of 300,000 images
- Filenames describe image
- Type in a word and it select all files whose name contains that word
- Retrieval of images interleaved with graphical display to avoid pauses

Ergo Version 2 ideas

- Rework with cleaner object model
- Envelopes, LFO’s, etc
- Patch represented entirely by connection of event processors
- Randomized patch generation

Hoops

- MIDI application written in Keykit
- Designed in collaboration with Herb Heinz
- Loops MIDI in realtime
- Handles multiple MIDI loopers simultaneously
- Animated graphical display of loops
- If ergo is running, MIDI events are sent to it
- Used at last Saturday's Y2K5 International Looping Festival in Santa Cruz

Audio/Visual Experiments with Python

Radio Free Quasar and Ergo

Tim Thompson
tjt@nosuch.com