

## The State of the Art in "Sound/Music Computing"

---

CS Colloquium--February 7, 1996

Stephen Travis Pope

Composer

Editor, *Computer Music Journal*

Research Director, CCMRC,

Dept. of Music, UCSB

stp@ccmrc.ucsb.edu

<http://www.ccmrc.ucsb.edu/~stp/>

## Name/Taxonomy/History

---

The Name of the Field

- Computer Music
- Digital Audio Signal Processing
- Sound and Music Computing
- Multimedia Computing
- ...others

## Outline

---

Musical Example 1

Name/Taxonomy/Historical Overview

Topics/Methods/Technologies

The Style Issue, Musical Example 2

Selected Computer Music Topics

Musical Prospects

Trends and the Future

Further References

Conclusions

Musical Example 3

## Methods/Technologies of SMC

---

(Influences, Background)

Physics (acoustics, instruments, methods)

Psychology (psychoacoustics, perception)

Electrical Engineering (hardware instruments, processors)

Computer Science (software, representations, comp. methods)

Mechanical Engineering (instruments, transducers)

Mathematics (composition and analysis methods)

Philosophy (aesthetics)

and of course Music theory, performance practise, composition

## Clients of SMC Methods/Techniques

- Music performance, recording
- Music broadcasting, distribution
- Music education
- Computer-human interaction, games, VR
- “Multimedia Computing”

### CM Artifacts

- Differences between Tools and Instruments (studio vs stage)
  - Dimensions for Taxonomical Groupings
    - task-oriented -- performance, recording, composition
    - user-oriented -- performer, composer, producer
    - technology-oriented -- DSP, MIL, HW-synth
- References: Pennycook, Loy, Pope

## History--The Dark Ages

### Late 1950s

- Max Mathews et al. at Bell Labs  
(Composers—E. Ghent, L. Spiegel)
- 1961 “Acoustical Compiler” Publications

### 1960s

- Music V distributed  
(HLL, portable, free)
- Bob Moog and Modular Analog Synthesizers  
(also Don Buchla and A. R. Pearlman)  
(Composers—W. Carlos, M. Subotnik)
- Mathews *Technology of CM* Book (1969)

## Topical Survey

- Software Sound Synthesis (direct synthesis)
- DSP Hardware and Software (synth. and proc.)
- Electro-Psychoacoustics (study and application)
- Timbre Modeling/Representation and Control
- UIs for Composers and Performers
- Real-time Protocols and Control
- Real-time Synthesis and Processing
- Alternative Performance Interfaces
- Music Representation/Notation Systems
- AI/OOP/NN/GA/... and Musical Applications

## History, and then...

### Early 1970s

- SAIL PDP-10 version Music10
- Chowning FM Paper in *JAES*  
(Technique for the control of complex timbres with few parameters)
- Many CM Centers Established--Universities, Conservatories, Radio Studios, Independent

### Mid-1970s

- Proliferation of Music-N languages  
(for N = 360, 11, 4B, etc.)
- Early Real-time Digital Hardware Systems
- Many Advancements in DSP Techniques

## History, in my career

---

### Late-1970s

- PDP-11s become universal, UNIX gains popularity over RT-11/RSX-11M
- Real-time Digital Synthesis Hardware  
--FRMBox, SamsonBox, IRCAM 4n
- Graphical Interactive User Interfaces  
--SSSP, MIT, SAIL
- Common-Practise Western Notation on Computer  
--SCORE, Mockingbird
- Annual Conference/Festivals  
--ICMC, Bourges, NMA, SEAMUS

## The Style Issue

---

Like the pipe organ, amplified guitar, or banjo, the computer is a “neutral” instrument that has been associated with a particular musical style (“radical contemporary ugly stuff”) through an accident of history.

These examples demonstrate its use for a variety of musical styles. (Examples from Stuart Favilla, Heinrich Taube, Peter Langston, and Stephen Pope)

## History, more recently

---

### 1980s

- Commercial Hybrid/Digital Synthesizers
- MIDI--Musical Instruments Digital Interface
- Better (2nd generation) Music UIs

### Early-1990s

- Physical Modeling Techniques
- Better real-time techniques and performance
- Affordable DSPs
- “Sound blasters” in all PCs
- Better OO/AI/NN/GA/VR... techniques

## Computer Music R&D Topics

---

- Synthesis Methods
- DASP Algorithms and Techniques
- DASP IC and System Architecture
- Representation of Musical Data and Knowledge
- Score Presentation and Visual I/O
- Gestural Input Devices (instruments)
- Integrated Computer Music Systems (CM Workstations--tools and instruments)
- CM Applications

# Selected Research Areas (Survey)

## Synthesis Techniques

### Physical Modeling

- Complex models -- Complex systems, lots of details
- Control methods
- Fast calculation

## DSP Techniques

### Vocoders

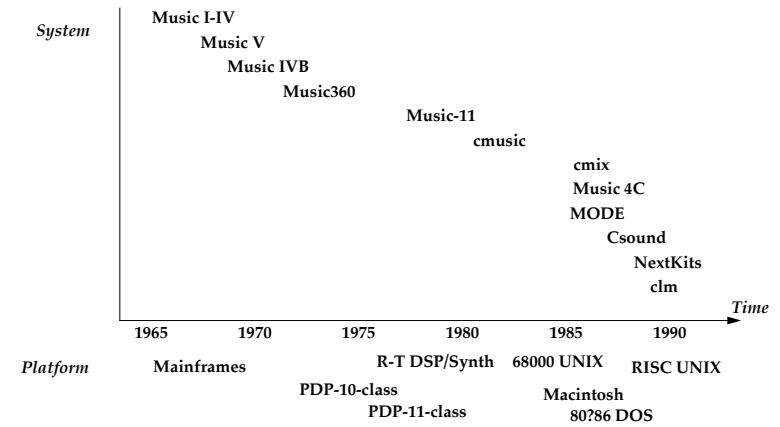
- Real-time vocoders (HW, SW)
- Complex texture mapping
- New techniques (e.g., wavelet)

### Pitch/timbre recognition

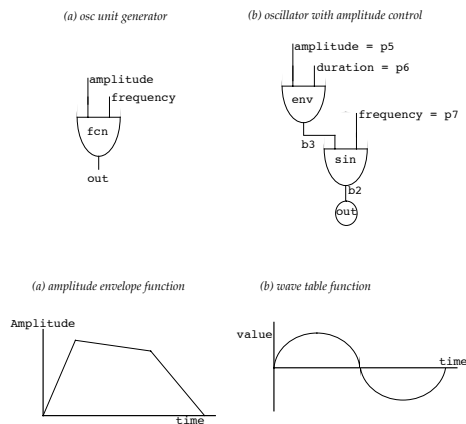
- Real-time robust pitch detection
- Expressive mapping via timbre recognition
- New techniques



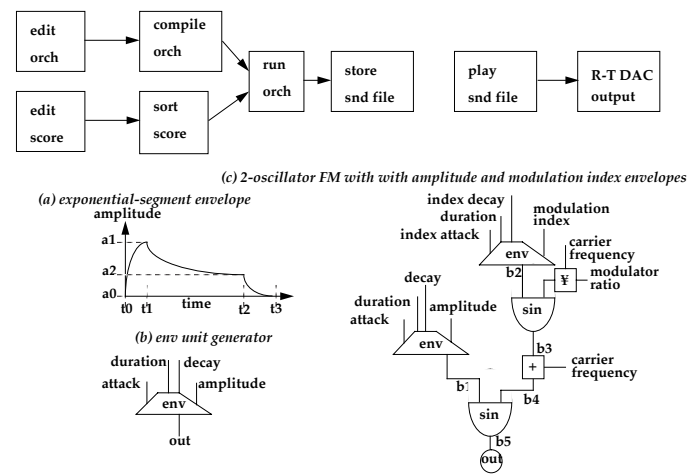
# Survey--Direct Synthesis Langs.



# Survey--Direct Synthesis Langs.



# Survey--Direct Synthesis Langs.



## Direct Synthesis Langs.--cmusic

```

{ Cmusic FM instrument          }
{ p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 }
{ note 0 fm1 dur amp freq c:m ind att dec iAtt iDec }

{ include cmusic definitions }
#include <carl/cmusic.h>

{ set important globals }
set rate = 44100;
set channels = 1;

{ names make instruments easier to read }
# define DUR p4
# define AMP p5
# define FREQ p6
# define RATIO p7
# define MODF b10
# define IND p8
# define ATT p9
# define DEC p10
# define IATT p12
# define IDEC p12

{ instrument named fm1 defined at time 0 }
ins 0 fm1; { b1 = amp1 env }
seg b1 AMP f2 d ATT 0 DEC;
{ b2 = index env }
seg b2 IND f2 d IATT 0 IDEC;

{ scale frequency }
mult MODF FREQ RATIO;

{ b3 = modulator }
osc b3 b2 MODF f1 d;

{ add base freq and mod }
adn b4 b3 p6;

{ b5 = carrier }
osc b5 b1 b4 f1 d;

out b5;
end;

{ function declarations }
SINE(f1); { this is pre-defined }

{ GEN4 uses exponential segments }
GEN4(f2) 0,0,-1 0.1,1,0 0.8,1,-1 1, 0;

{ note list }
{ 0 fm1 dur amp freq c:m ind att dec iAtt iDec }
note 0 fm1 2 -2dB 100Hz 1 1 .2 .1 .05 .4;
note 2 fm1 2 -6dB 100Hz 1 3 .2 .1 .05 .4;

ter;

```



## Direct Synthesis Langs.-CLM

```

(definstrument drum (start-time frequency amplitude
;; NB there is no duration arg -- this is computed later,
;; as a function of amplitude and frequency
&key
;; the m-ratio is set to the very 'un-integer-like' value of 1.4
(c-ratio 1.0) (m-ratio 1.4)
(index-min 0.0) (index-scl 1.0)
;; The amp-env and index-env envelopes are defined
(amp-env '(0.000 0.831 17.372 1.000 30.073 0.488 57.810
0.164 100.000 0.000))
(index-env '(0.000 1.000 0.219 0.670 0.696 0.307 10.007
0.000 100.000 0.000))
(degree 0.0) (distance 1.0) (reverb-amount 0.005))
(let* (
;; duration is in this way dependent on frequency and amplitude:
;; the higher above 130 Hz, the shorter the duration;
;; the lower the freq below 130 Hz, the longer the duration;
;; and the greater the amplitude the longer the duration.
(duration (* 0.75 (/ 130 frequency) (+ 0.9 amplitude)))
(beg (floor (* start-time sampling-rate)))
(end (+ beg (floor (* duration sampling-rate))))
;; again the value of the modulation index is given a dependency
;; with respect to amplitude and frequency: the frequency
;; relationship is as in the duration calculation, and the amplitude
;; relation is now a multiple.
(index-max (* index-scl 8 (* 1.6 amplitude) (/ 130 frequency)))
;; Simple FM instrument
(carrier (make-oscil :frequency (* c-ratio frequency)))
(modulator (make-oscil :frequency (* m-ratio frequency)))
(amp-env (make-env :envelope amp-env
:scaler amplitude
:start-time start-time
:duration duration))
(index-env (make-env :envelope index-env
:offset (in-Hz (* index-min m-ratio frequency))
:scaler (in-Hz (* (- index-max index-min)
m-ratio frequency))
:start-time start-time
:duration duration))
(loc (make-locsig :degree degree
:distance distance
:revscale reverb-amount)))
(Run
(loop for i from beg to end do
(locsig loc i (* (env amp-env)
(oscil carrier (* (env index-env)
(oscil modulator))))))))

```



## Survey--Music Representation

### Abstract representation

- AI/KR/Theoretical CS methods
- Compact interchange formats
- Expressive languages for performance description
- Flexible interfaces between formats

### Hypermedia

- "Score as document" systems
- "HyperScore" browsers and tools

### Composition Tools

- Representation of "middle-level" structures
- Scalable tools for composition and performance
- New compositional formalisms and methods



## Alg. Comp. and H-L Comp. Tools

### Algorithmic Composition

- Where's the model?
- The machine's role vs. the human's role (My bias)

### Interactive Composition

- How much is done before-hand? (My biases...)

### High-Level (Intelligent) Tools for Composers

- What's *smart*?
- What's a *higher* level?



## Proc., Stoch., and K-B Systems

### Procedural Representations

Procedural vs. declarative representations

Procedural description as a model (??)

### Stochastic Models

Stochastic data vs. stochastic structure

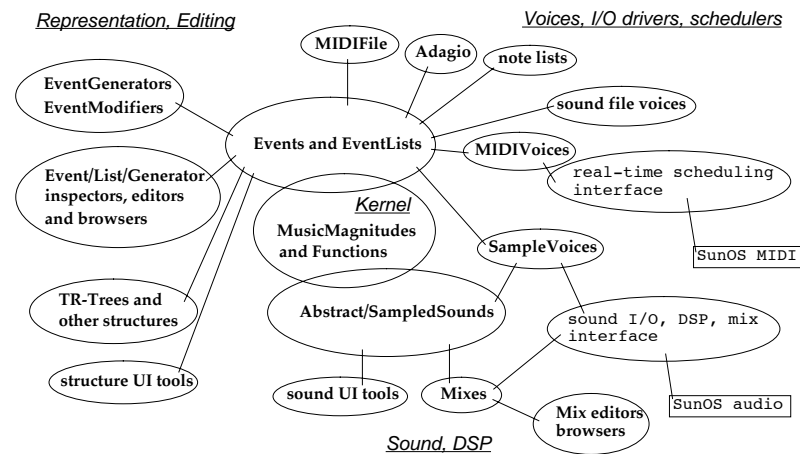
Bounded/unbounded/directed/selection stochasm

### Knowledge-based Systems

Forward- vs. backward-chaining

(analysis vs. composition)

## MODE Software Components



## The MODE/Smoke System

A Smalltalk-80-based framework and tool kit  
for music composition and performance

- **SmOKE music representation**
  - Music Magnitudes
  - Events, Event Lists
  - Generators, Modifiers, and Structures
  - Functions, Sounds and DSP
- **MODE voices and I/O**
  - Voices as device drivers
  - Voices as property-to-parameter mappers
- **MODE user interface components**
  - MVC Framework and Navigator
  - MODE MVC Support and Applications

## SmOKE--Verbose and Terse

```
"Verbose MusicMagnitude Creation
and Coercion Messages"
"Answers Duration 62 msec."
(Duration value: 1/16) asMsec
"Answers Pitch 261.623 Hz."
(Pitch value: 60) asHertz
"Answers Amplitude 106."
(Amplitude value: 'ff') asMIDI
```

```
"Event Creation Messages"
"Create a 'generic' event."
Event dur: 1/4 pitch: 'c3' ampl: 'mf'
"Create one with added props."
(Event dur: 1/4 pitch: 'c3')
color: #green; accent: #sfz
```

```
"EventList Usage"
"Create a named event list."
el := EventList newNamed: #demol.
"Add an event to it at time 0."
el add: (Event dur: 1 pitch: 36 ampl: 'mf');
"Add an event after the first."
add: (Event dur: 1 pitch: 40 ampl: 'mf');
"Add a sublist with 3 events"
add: (EventList new"a chord."
add: (Event dur: 1 pitch: 36) at: 0;
add: (Event dur: 1 pitch: 40) at: 0;
add: (Event dur: 1 pitch: 43) at: 0)
```

Verbose SmOKE Examples

```
"Terse MusicMagnitude Creation using post-ops"
440 Hz 250 msec
1/4 beat 'c#3' pitch
```

```
"Terse Events using concat. of music mags"
440 Hz, 1/4 beat, -12 dB, (#voice ->
#flute).
38 key, 280 ticks, 56 vel.
(#c4 pitch, 0.21 sec, 0.37 ampl).
```

```
"Terse EventLists using concat. of events or
(duration -> event) associations"
(440 Hz, (1/1 beat), 44.7 dB), "comma"
(1 => ((1.396 sec, 0.714 ampl) word: #xu))
```

```
"Bach Example--First measure of Fugue 2 from
WTK (ignoring the initial rest)."
```

```
((0 beat) => (1/16 beat, 'c3' pitch)),
((1/16 beat) => ('b2' pitch)),
((1/8 beat) => (1/8 beat, 'c3' pitch)),
((1/4 beat) => ('g2' pitch)),
((3/8 beat) => ('a-flat2' pitch)),
((1/2 beat) => (1/16 beat, 'c3' pitch)),
((1/16 beat) => ('b2' pitch)),
((1/8 beat) => (1/8 beat, 'c3' pitch)),
((3/4 beat) => ('d3' pitch)),
((7/8 beat) => ('g2' pitch))
```

(Can be abbreviated further...)

Terse SmOKE Examples

# MODE Event List Usage

## Event List Creation

- Text, graphical input
- Procedural generation
- Reading data from other applications
- Built-in creation methods

EventList fromSelectors:data:

EventGenerators

## EventGenerators and EventModifiers

(see also [ICMC 1989])

## TR-Trees and other structures

(see also [ICMC 1991])

## Persistency, links and hypermedia

(HyperScore idea and tools)



# MODE UI Examples

## Event, EventList and Sound Inspectors (low-level GUI)

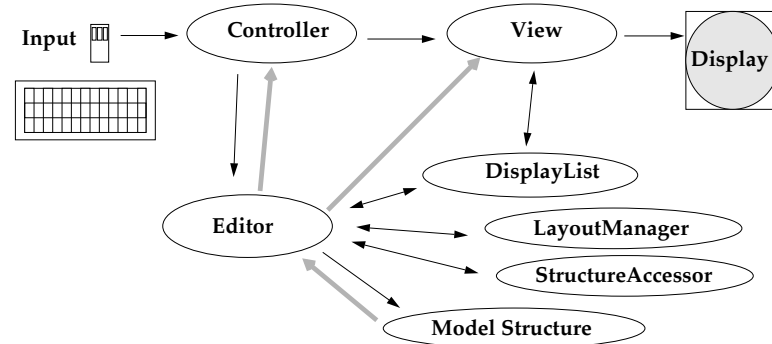
The image shows four windows from a low-level GUI:

- NoteEvent:** A window with a 'self' field and a list of properties: duration (1/4), pitch: 'c3', voice: #flute, loudness: #mf.
- EventList:** A window showing a list of 7 events with their durations and phonemes. Example: (0 msec) => (595 msec) (L: 0.8) (phoneme: #dun).
- SampledSound:** A window with properties like rate (44100), format (#linear16bit), channels (2), and a list of sample values.
- EventList (second):** A window showing a list of 4 events with their durations and properties. Example: 0ms => (NoteEvent d: 1000ms. k:36 l: 100).



# MODE User Interfaces

## Model/View/Controller Programming and Navigator



# MODE Sampled Sound UI

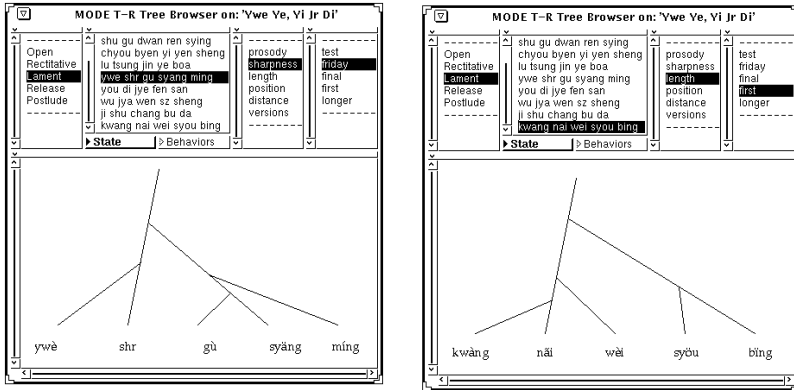
The screenshot shows the SoundBrowser application interface:

- File List:** A list of sound files including 'st', 'sun', 't1', 'tmyb', 'yy', 'yypid', 'bells', 'ec', 'instruments.i', 'instruments.s', 'sz', 'dete2.l.snd', 'dete4.l.snd', 'errs', 'Makefile', 'read.l.l.snd', 'y2cr4b.snd', 'y2cs2.snd', 'ywe.m', 'ywe1.snd', 'ywe2.snd', 'ywe2a.snd'.
- Waveform:** A visual representation of the selected sound's amplitude over time.
- Properties Panel:** Shows metadata for the selected sound: gate: 44100, fmt: lin16bit, dur: 0.534, samps: 22703, word: 'ywe', date: 22 April 1991, onat: '2d tone, left, B&K mic'.
- Actions:** Buttons for Play, File, Inspect, and Clear, along with a context menu for the waveform (copy, cut, paste, accept, cancel, spawn, inspect, vertical scale, horizontal scale, fade in, fade out).

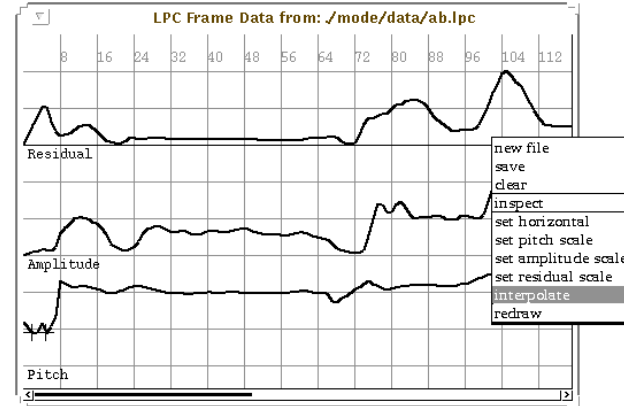


# MODE Structure Editors

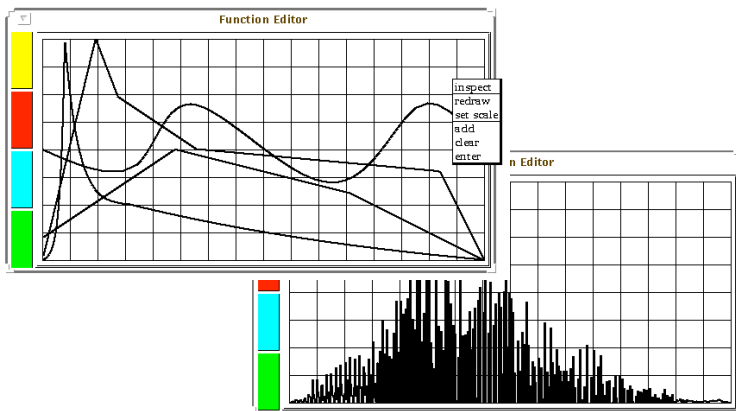
## TRTreeBrowsers



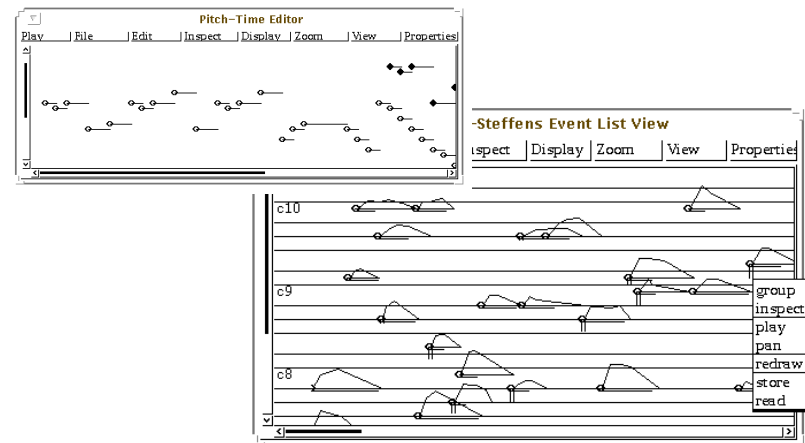
# LPC Vocoding Tools



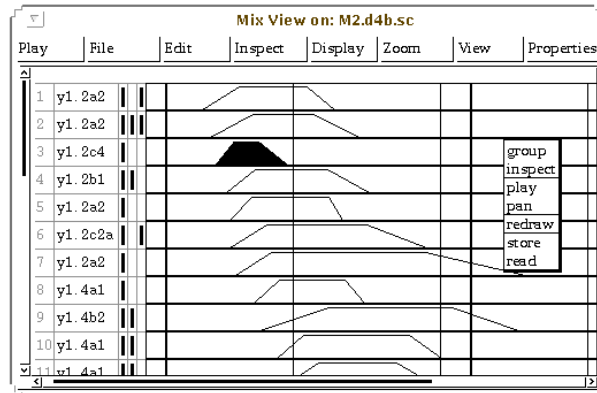
# Function Editors



# EventListEditors



## Mixer View



## Trends

- More is possible in real-time every day.  
(We cannot control everything in real-time, however.)
- Synthesis via physical models is coming.  
(It's all the way not here yet.)
- Abstract music representations are needed for composition  
(The "market" doesn't care.)
- Abstract representations for DSP are important.  
(The "market" doesn't understand this.)
- New controllers and performance interfaces are coming.  
(Through most people want keyboards or "wind controllers.")

## The Future

- Using New Hardware and Software Technologies
  - Parallel (multi-) Processing
  - PDP/NeuralNets
  - New UI SW/HW Technology
  - (etc.)
- Post-MIDI MIDIs
  - ZIPI or the TRON Music LAN
- More FLOPs in Real-time Algorithms
  - DSPs for the Masses (a la NeXT)

## Musical Prospects

- The Computer as a Composer's Tool
- The Computer as a Composer's Assistant
- The Computer as a Performer's Instrument
- The Computer as a Performer
- "Art" Music and "Pop" Music
- Results (Manifold and Masterful)

## The (large) Printed Literature

---

- Periodicals
  - *Computer Music Journal* (V1-20)
  - Int'l CM Association *Array*
  - Int'l CM Conference *Proceedings*
- Books
  - *Foundations of CM and The Music Machine*
  - MIT Press and A-R Editions Series
- On-line
  - <http://www-mitpress.mit.edu/Computer-Music-Journal/>
  - <http://www.ccmrc.ucsb.edu>
  - <news://comp.music.{research,midi,...}>
  - <http://www.ccmrc.ucsb.edu/~stp/publs.html>

## Summary

---

- Many sub-fields  
(DSP, AI, HW architecture, languages, etc.)
- Many areas of expertise  
(SW, HW, psych., math., etc.)
- Many unfinished tasks  
(Real-time issues, abstraction issues, scaling issues, UI issues, etc.)
- So what's up at CCMRC (now and in the future)?

## The Recorded Literature

---

### Lots of Available (Accessible) Music

- Wergo CD Series
- Centaur CDCM CD Series
- GMEB/UNESCO CD Series
- ICMC CD(s)
- CCRMA Cassettes
- PNM Cassettes
- Many more Records/CDs/Tapes