Chapter 7. Meeting 7, History: Gottfried Michael Koenig

7.1. Announcements

• Test direct rendering of CSD files with Csound if ELr is not working

• Make sure you have PD-extended installed and Martingale on your system

7.2. Quiz

• 10 Minutes

7.3. Gottfried Michael Koenig

• Gottfried Michael Koenig (1926-)

• 1954-1964: Worked with Stockhausen and others at West German Radio in Cologne

• Composed for tape and acoustic instruments

• 1963-1964: Studied programming, began developing software for CAAC

• 1964-1986: Director of the Institute of Sonology in the Netherlands

• Employed CAAC at three different levels

  • Symbolic: output from computer used to transcribe notation

  • Control: create sequences of control voltage mapped to synthesis parameters

  • Direct: employed direct creation of waveforms to create non-standard synthesis techniques

7.4. Reading: Koenig: The Use of Computer Programs in Creating Music


• Koenig states that the use of the computer does not herald a new musical epoch: instead, what does he see the computer as offering?

• Koenig describes a variable function generator: what is this?
• Koenig sees work in the electronic music studio as suggesting some of the practices of algorithmic composition: how so?

• Koenig introduces the term composition theory: what might this mean?

• What role did Koenig imagine for the computer in the work of composers and music students?

• Koenig describes a technique of “sound production”: what is this?

7.5. Koenig: CAAC for Acoustic Instruments

• Two early software systems

• 1964: Project 1 (PR1)

• 1969: Project 2 (PR2)

• Favored discrete value generation and selection

7.6. PR1: Concepts

• 1964: Project 1 (PR1): Programmed in Fortran for the IBM 7090

• A closed system, providing output based on user parameters

• A user specified six tempo values, twenty-eight entry delays (rhythmic values), a random generator seed value, and the length of the composition

• Materials were algorithmically selected

  • Series: random permutations, selection without replacement

  • Alea: random selection

• Koenig saw series generation as an abstraction of twelve-tone techniques: “the need for variation is satisfied without there having to be the pretense that somewhere deep inside the work the twenty-fifth permutation is still being systematically derived from an original series” (1970a, p. 34).

• At a larger level, 12-tone rows are created and deployed and three-note aggregate completing trichords are created.

• A seven-section formal structure controls the large-scale form, defining a position in a range from regular/periodic to irregular/aperiodic.

• Output is provided for six parameters: (1) timbre (instrument or instrument group), (2) rhythm, (3) pitch, (4) sequence, (5) octave register, and (6) dynamic.
• Sequence is spare parameter, applied to chord formation or timbre component within a timbre group

• All parameters are independent

7.7. PR2: Concepts

• 1969: Project 2 (PR2): Algol, then Fortran for the PDP-15

• A closed system, but more general and user-configurable

• Eight parameters are generated: (1) instrument, (2) harmony, (3) register, (4) entry delay, (5) duration, (6) rest, (7) dynamics, and (8) mode of performance.

• Expanded tools for algorithmic selection

  • Series

  • Alea

  • Ratio: weighted random selection

  • Group: repetition of values

  • Sequence: ordered selection

  • Tendency: random selection within dynamic boundaries

7.8. PR2: The List-Table-Ensemble Principle

• Selection procedures can be used on user-specified numeric or symbolic values (lists, stockpiles, or tables), or new, algorithmically generated expansions of user-specified numeric or symbolic values (ensembles).

• Lists: raw stockpiles of data (assigned index values for access)

• Tables: user-organized collection of indexes pointing to data in Lists

• Ensembles: selection methods are used to create intermediary groups of data that are then drawn from to produce parameter values

• A techniques of meta-selection that constrains values within distinct representations (distributions and orderings)

• IterateHold: a rough analogy to the list-table-ensemble principle: select a number values from a PO, employ these for a number of times, and then regenerate a new selection

:: tpmap 120 ih,(ru,0,1),(bg,oc,(2,4,13)),(bg,oc,(10,15))
iterateHold, (randomUniform, (constant, 0), (constant, 1)), (basketGen, orderedCyclic, (2, 4, 13)), (basketGen, orderedCyclic, (10, 15)), orderedCyclic

TPmap display complete.

7.9. Listening: Koenig

• Three Asko Pieces


7.10. PR2 Selection Principles: Ratio

• Weighted randomness can be achieved by configuration of BasketGen values

• More control can be obtained by configuring a zero-order Markov chain, to be discussed later

7.11. Controlling Pitch in athenaCL

• Paths provide ordered collections of pitch groups (Multisets) with proportional durations
• A Texture is assigned a Path based on the last-created Path, an assigned Path, or an automatically
created Path (if none exist)

• The default Path is a single pitch, C4

• A Texture can transform the Path with ParameterObjects assigned to the field (transposition) and
octave (register shift) parameters

• Different TextureModules can deploy Paths in very diverse ways

7.12. PR2 Selection Principles: Group

• IterateGroup: Two POs, one generating values, the other selecting how many times those values
are repeated before a new selection is made

:: tpmap 100 ig,(bg,oc,(0,5,10)),(bg,rc,(3,5,7))
iterateGroup, (basketGen, orderedCyclic, (0,5,10)), (basketGen, randomChoice, (3,5,7))
TPmap display complete.

• Create a collection of values, select a value, and then repeat a selected number of times

• Command sequence:
  • emo m
  • tin a 6
  • tie r cs,(rb,2,2,.02,.25)
  • tie f ig,(bg,rc,(2,4,7,9,11)),(bg,rp,(2,3,5,8,13))
  • tie o ig,(bg,oc,(-2,-1,0,1)),(ru,20,30)
  • ticp a b c d
  • eln; elh
7.13. PR2 Selection Principles: Tendency Mask

- Random values selected from within dynamic minimum and maximum value range
- Can be implemented with any Generator PO that has min/max parameter
- Boundaries can be controlled by BreakPoint, Wave, or similar ParameterObjects
- A powerful technique for creating long range behavior

Here, a break-point function and a wave sine generator form the boundaries of a random beta selection to control pitch

Command sequence:

- emo m
- tin a 15
- tie r cs,(ig,(ru,.01,.25),(ru,4,12))
- tie a ru,2,(cg,u,3,9,.005)
- tie f rb,2,2,(bpl,t,l,((0,-12),(30,12))), (ws,t,29,0,0,24)
- eln; elh

A powerful technique for creating long range behavior

Here, random octave values are chosen between two wave triangle generators

Command sequence:

- emo m
- pin a d,e,g,a,b
- tin a 107
- tie r pt,(e,16),(ig,(bg,rc,(1,2,3,5,7)),(bg,rc,(3,6,9,12))), (c,1)
- tie o ru,(wt,t,25,0,-2,4),(wt,t,20,0,-3,1)
- eln; elh


- Koenig states that “... to react functionally means ... to refrain from imitation of a particular production mode in another medium”: what is he suggesting?

- What is Koenig suggesting about the use of histograms, where the composer supplies histograms and the computer program takes care of the data connections?

- What is the process of transcription that Koenig describes? How is this different than conventional transcription?

- What is aesthetic integration? Does Koenig suggest that this step can also be automated?

- Koenig talks about composer having a sense of responsibility for the aesthetic result: why is this significant?

7.15. Koenig: CAAC for Voltage Control

- Used PR1 to generate events that were encoded in voltage control data

- Voltage control data processed and translated to various musical parameters at different speeds

- Used “variable function generator” (1966) to set and deploy values from the control rate to the audio rate

• Similar methods will be employed by outputting athenaCL generators to PD Arrays

7.16. Listening: Koenig

• Employed techniques of Funktion pieces

• Koenig: “Terminus X” (1967)
7.17. Alternative Approaches to Grouping and Masking

- **BasketSelect**: Select values from a list using another PO providing values within the unit interval

  :: tpmap 100 bs, (-3, -2, -1, 0, 1, 2, 3), (ru, (bpl, e, l, ((0, .5), (100, 1))), (bpl, e, l, ((0, .5), (100, 0))))
  basketSelect, (-3, -2, -1, 0, 1, 2, 3), (randomUniform, (breakPointLinear, event, loop, ((0, 0.5), (100, 1))), (breakPointLinear, event, loop, ((0, 0.5), (100, 0)))), TPmap display complete.

- **IterateWindow**: Select from a list of POs, and then draw a selected number of values from that PO

  :: tpmap 100 iw, ((ru, .2, .8), (re, 15, 0, 1), (ws, e, 12, 0, 0, 1)), (bg, rp, (14, 20, 26)), oc
  iterateWindow, ((randomUniform, (constant, 0.2), (constant, 0.8)), (randomExponential, 15.0, (constant, 0), (constant, 1)), (waveSine, event, (constant, 12), 0, (constant, 0), (constant, 1))), (basketGen, randomPermutate, (14, 20, 26)), orderedCyclic
  TPmap display complete.