CONSONANCE

A Computer Generated Composition

Joaquim Ribeiro Freire-Neto

A dissertation submitted in partial fulfillment of the
requirements for the degree of

Doctor of Musical Arts
University of Washington
2017
Reading Committee:
Huck Hodge, Chair
Joël-François Durand
Charles Corey

Program Authorized to offer Degree:
Music

University of Washington

Abstract

Consonance

A Computer Generated Composition

Joaquim R. Freire-Neto

Chair of Supervisory Committee:

Associate Professor Huck Hodge

Music

Consonance is a computer generated composition using Csound not only to construct the instruments but also to explore some of its innumerous possibilities to modeling timbres, dynamics, spatialization and reverberation. The sound events were done using blue, a time track environment for Csound and python scripts inside blue.

The process of producing the piece had part of its source of inspiration from scales and tuning used by inhabitant people of the Northeastern Brazilian countryside and part from the author`s conceptual method of composing music directly related with sound produced from the environment.

A supplementary file (Consonance) in WAV format is part of this dissertation. It is the output of a computer generated Csound score built in blue.

ACKNOWLEDGEMENTS

During all those past years I have been at the University of Washington as graduate student, nothing was more important than the new possibilities I discovered in the process of composing, by exploring new ways of expressing myself with music. There I had the opportunity to study with professors Joël-François Durand, Richard Karpen, William Smith and Kenneth Benshoof. They made a great impact in my way of thinking of music as expression of art.

Special thanks to Huck Hodge who accepted being the chair of my committee, and Charles Corey for being a member of my final exam in such short notice.

I would like to thank Steven Yi for his support in grasping the way of working with the Csound environment called “blue” and for his help in understanding python script inside blue.

I also would like to express my gratitude to those who helped me to achieve this goal of great significance:

To Dr. Brenda Banks who was always prompt to help me in several ways during the last arduous process of being a student far away from campus and for her help in acquainting me with new procedures at the UW.

To my wife, Orlania M. Freire for her patience, understanding and dedication during the months I spent working on this project.

To my aunt, D`Alva Stella N. Freire who has been encouraging me in the music field ever since my youth.

To my colleagues at Federal University of Piauí, who made a great effort to adjust situations in my favor and to motivate me in all steps of this work.

POSTHUMUS MEMORIAM

In loving memory of my parents, Joaquim N. Freire and Maria Anísia de S. Freire and my brother Walter de S. Freire. This work is dedicated to them.

CONTENTS

	Page
LIST OF FIGURES	4
INTRODUCTION	5
CHAPTER 1: HISTORICAL ASPECT	6
CHAPTER 2: USE OF FRACTALS AS THE SOURCE OF MATERIAL IN THE COMPOSITION	7
CHAPTER 3: PROCEDURES TO GENERATE THE SCORE	9
CHAPTER 4: DESCRIPTION OF THE PIECE	12
CHAPTER 5: THE ORCHESTRA	14
CHAPTER 6: GEN FUNCTIONS AND MOST USED WAVE AND ENVELOP TABLE DIAGRAMS	23
CHAPTER 7: PYTHON SCRIPTS AND LIST OF CSOUND EVENTS	29

LIST OF FIGURES

Page

Figure 1: the first four levels of the Koch curve 8

Figure 2: chords of the piece 10

Figure 3: diagram of the plucked instruments 1 and 2 12

Figure 4: sine wave for buzz opcode - instrument 3 23

Figure 5: buzz-like wave for instruments 1 and 2 24

Figure 6: cosine wave - instrument 4 26

Figure 7: short attack long decay envelope 27

Figure 8: ADSR envelope 27

Figure 9: ADR envelope 28

Figure 10: short attack and decay exponential release 28

INTRODUCTION

My interest in algorithmic composition started when I took classes as DMA graduate student at the University of Washington. During those years I got acquainted with Csound, a powerful music programming language with almost endless possibilities of modeling sound. From that moment, I decided to compose part of my works with Csound, considering the possibility of creating synthetic sounds to suit my ideas of composing a particular musical piece.

Consonance is a computer-generated composition with five instruments and three global reverb instruments. Part of the score was made with python script to process lists of sound events being some of them based on the process proposed by Helge von Koch known as the Koch curve, a concept of repeated self-similar smaller forms of the first form. Inspired by this recursive principle, this work departs from the process of generating ascending pitch intervals having equal duration and their smaller repetitions. The score was implemented using blue, a time-track Csound environment, allowing the writing of score with part of them with python objects.

In this paper it is discussed the process, starting from the techniques and coding of the Csound instruments and the selection of theme and rhythm to generate the list of sound events. The following sections are related to the coding and techniques of the instruments used in the composition and aspects that played major roles in decisions of some specific synthesis techniques.

CHAPTER 1: HISTORICAL ASPECT

All musical aspects of the composition are attempts to depict guitarists playing their guitars in an open market of a small town located on the Brazilian Northeastern region. As observed by the author in his youth, some guitarists played a scale on their out of tune hand-made guitars, alternated with chords, in such a way that each sequence of chords was not related with the key of the scale, but in exquisite and exotic way resembling chord progression from another tune. The sequence of chords, in the key of C major, was either played with ternary pulse or without any rhythmic implication. The scale, lost in mind, started on the open G-string played on a steady pulse. All notes of the scale were plucked one at a time, with the exception of the open G. That string was struck several times, being muted by the left fingers two or three times during the execution. The conjunction of the scale played in a frantic manner seems to be an attempt to make the tune depart from it, with alternations of a calm and contemplative part made out of chords. That somehow chaotic performance made that musical environment unique and original.

CHAPTER 2: USE OF FRACTALS AS THE SOURCE OF MATERIAL IN THE COMPOSITION

Fractal geometry has been used to represent objects, such as trees, mountain ranges, snowflakes and other objects that cannot be represented by means of the classical Euclidean geometry. Like many aspects in music, fractals are self-similar on multiple levels. How a part is represented is a similar replica of the whole. Forms are achieved by repeating the initial step over and over.

THE KOCH CURVE

The Koch curve was idealized by a Swedish mathematician, named Helge von Koch (180-1924) as a way of representing a curve that has no tangent at any point. He describes in his work published in 1904 an iterative process of placing triangles on the side of other triangles, as shown in Figure 1.1 that shows the first four levels. The Koch curve results if the process is carried to higher levels, ad infinitum. It will result in a line segment with an infinite length. The Koch curve is the logarithm of the lengths of four segments over the logarithm of the distance of three segments. Its dimension is log 4/log3 or ≈ 1.26[1]. The construction of the Koch curve is started by dividing a line segment into three equal parts. A fourth line segment, equal in length to the other three, is inserted in the middle so that the shape is that of an equilateral triangle. The first seed level is now achieved. For the next level, this process is done for each straight line segment of level one. Iteratively, this process could be carried on, ad infinitum, to achieve higher levels. The process of breaking the whole into smaller self-similar parts was applied in Consonance to represent duration, starting with a three-note ascending motif, each of the three notes being four seconds. Throughout the piece, superposing layers of the motif and its variations, from the first to the fourth level, are heard until the very end. At that moment, the initial motif is heard for the last time.

[1] Wahl, Bernt. *Exploring Fractals on the Macintosh.* Reading, Massachusetts: Addison-Wesley Publishing Company, 1995, 45-48.

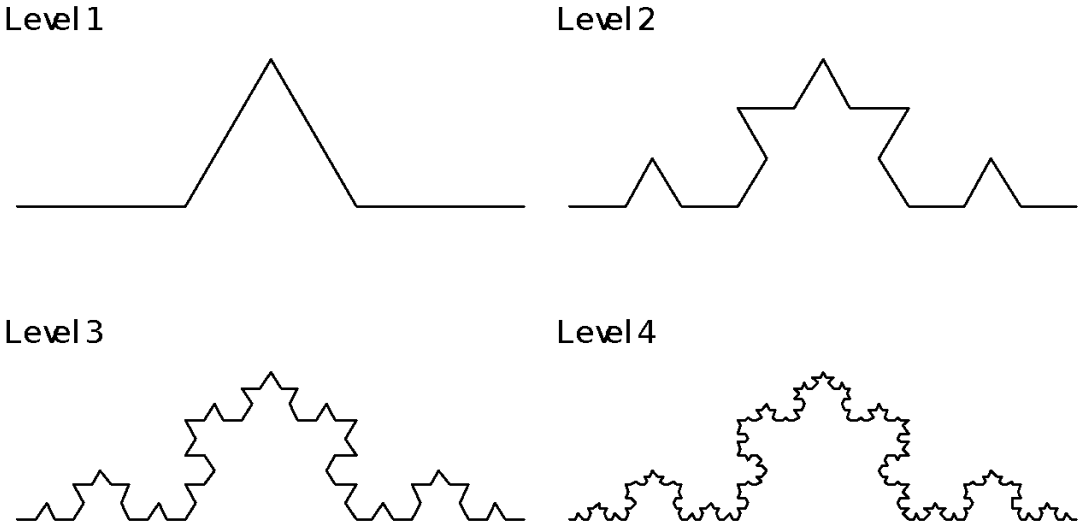

Figure 1: the first four levels of the Koch curve[2]

[2] The fractals in Figures 1 was drawn using the Macintosh program FractaSketch.

CHAPTER 3: PROCEDURES TO GENERATE THE SCORE

MELODIC MATERIAL

The melodic motif was set up having in mind the melodic pattern played by guitarists mentioned earlier. They departing from G reaching the first half of the scale an augmented fourth, G - C#, as in the first half of the lydian mode, while the second half a perfect fourth, D - G, as in the second half of the mixolydian mode.

In order to establish the melodic material in Consonance there are two sets of melodic material in combination with a third set, written in python scripts. The first set, played by instrument 1, departs from E3 followed by an augmented fourth.

```
pitch = 164.81; # E3
increment = 68.27; # Init A4th
```

The second set, played by instrument 2, departs from G4 followed by a perfect fourth.

```
pitch = 392; # G4
increment = 131.25; # Init P4th
```

The third set, played by instrument 5 in conjunction with either instrument 1 or 2 has D4 as the initial note, followed by a perfect fourth.

```
pitch = 293.66; # D4
increment = 98.34; # Init P4th
```

Python scripts are used to make ascending pitches, being the first interval either a perfect or augmented fourth. As the pitches progress they become less accurate due to the fact that each python script has fixed increment related to frequency. Since frequency is logarithmic related, ascending or descending intervals with fixed increments makes the three-note motif and its derived sets not related to equal temperament, Therefore the adopted procedure is more closely related to rustic instruments that do not take this logarithm principle into account.

CHORDS

Chords, played by instruments 3 and 5, are based on diminished, augmented and perfect fourth and fifth intervals. Chords 1 and 4 are combination of perfect fifth intervals displaced in two and three octaves respectively, while chord 2 and 3 are the result of perfect, diminished and augmented intervals, displaced in octaves as well.

CHORD CHART

Chord 1		**Chord 2**		**Chord 3**		**Chord 4**	
D_4	293.66	C_4	261.63	B_4	493.88	F_3	174.61
A_3	220.00	B_3	246.94	$F^{\#}_4$	369.99	B_2	123.47
G_3	196.00	F_3	174.61	E_4	329.63	E_2	82.41
D_3	146.83	E_3	164.81	$A^{\#}_3$	233.08	B_1	61.74
G_2	98.00	C_3	130.81	E_3	164.81	E_0	20.60

Figure 2: chords of the piece.

Instrument 3 also plays pitch undefined lines representing gentle wind blowing, which intermingles with the sound coming out of the instruments and from the crowd. Instrument 4 plays a bass line with harmonics, functioning as a pedal point from the very beginning to the end of the piece.

Having established the sources of the composition here comes the development. The macro material is used in the process of generating all the structural chords throughout the piece. Chords and their derivation have their order shifted, sometimes arpeggiated and sometimes struck blocks of notes.

The alternation between chords and melodic materials represent initial iteration of the set as well as the chaotic way they change from scale to chord. The initial sequence departs from the level one of the Koch curve. Sequences derived from the first set are related with the melodic parts that form the piece.

The whole piece is divided into three parts, the last being a recurrence of the first with a more intense subdivision of the Koch curve. The middle section functions as a contrast of the two outer sections. Instrument 5 plays a leading part alternating in timbres sectioned by chords produced by the plucked instruments.

RHYTHM STRUCTURE

The rhythmic sequences are related to the melodic parts as in the establishment of the intervals between note events, therefore it creates the rhythmic structure associated with theme and its development. The first set of durations outputs a list of three pitches, all having the same duration. The sum of the three durations establishes the timing of the subsequent derived sets. Consecutive sets have the ⅓ ratio subdivision in duration but multiplied by three and multiples of three the number of pitches. The sum of durations of the set, whether the three-note set or its derived ones are all the same, except when the derived sets are shortened by one third in the number of pitches.

CHAPTER 4: DESCRIPTION OF THE PIECE

The composition starts with a bass line, followed by decaying plucked string instruments on a steady pulse, doubled by instrument 5 and echoed by one of the global instruments. The first pitches are heard in the middle upper and lower registers. Instruments 1 and 2 were coded, each with three connected modules based on the Karplus-Strong algorithms. The second module is multiplied by 1.009 and the third one by .9984. This gives the quality of an instrument slightly detuned. In both instruments the tables are created from the sound files generated by instrument 3. Each sound file was assigned a different center frequency and bandwidth. GEN tables are from 20 to 29.

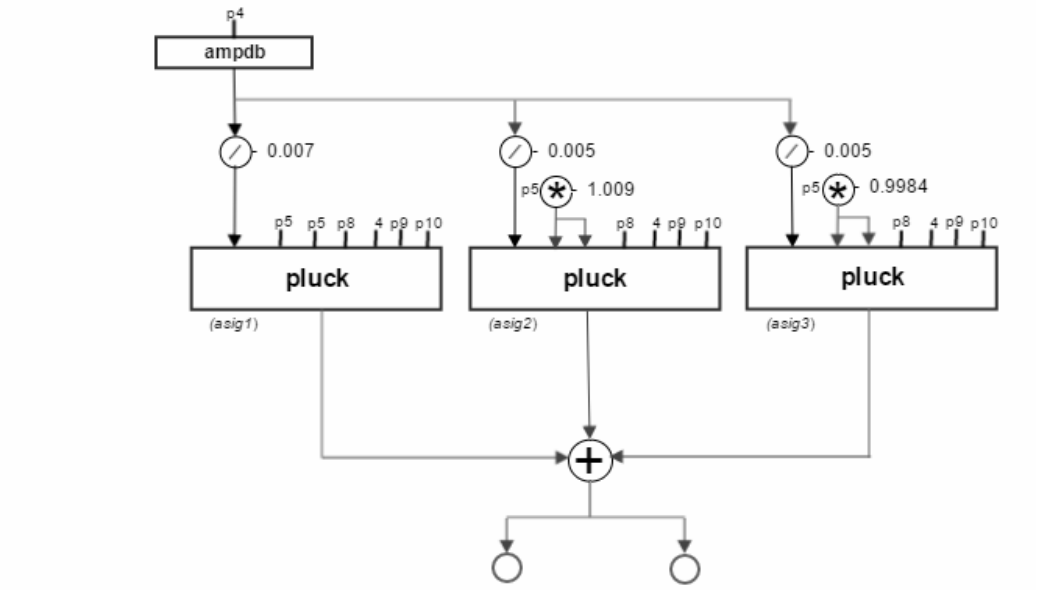

Figure 3: diagram of the plucked instruments 1 and 2.

The higher the center frequency and bandwidth, the more open and brighter the quality of the sound. The main characteristic of a plucked string instrument is preserved by placing some recurrences of the iteration with different tables, which achieves a change in timber. A background sound in a lower register is brought to the foreground to accompany the development of the theme. From the opening statements, the parameters of pitch deviation and pitch displacement are set to small numbers. It lasts until the statements of an agitated plucked passage and its echoed repetition, which announce the beginning of a more chordal, calm and contemplative section. Portions of the melodic line are present in the background. Instrument 5 is predominant in this section. With its different GEN f-table numbers 31 to 38 that shape the waveforms, and its envelopes, GEN f-table numbers 41 to 48 that model the sound. This middle section introduces an orchestral-like passage with a variety of timbers and chord attacks. This section ends right after the chordal arpeggio that announces the returning of the iteration. This time, the iteration is introduced by instrument 3. On this latter instrument, not only center frequencies and bandwidths can be varied dynamically, but also its iteration along with the note's duration. The iteration starts on various registers of G on five-second long notes. As the piece progresses, the decrescendo-like iterations diminish in number. Iterations on the plucked instrument are heard in the background and are rapidly brought to the foreground and again returned to the background. At this time all the instruments join in, playing parts generated by both levels of the Koch curve in a circulatory way, each transitioning from background to foreground. All five Csound instruments coded for this piece have the opcodes for locating sound and spatialization, written by Richard Karpen[3]. The last part, the reappearance of the theme is heard in the background, bringing the piece to a coda before the final statement of the Koch-curve set.

[3] Csound opcodes *locsig* and *locsend*. Author: Richard Karpen.

CHAPTER 5: THE ORCHESTRA

```
sr = 44100
kr = 4410
ksmps = 10
nchnls = 2

gareverb        init 0
gareverb1       init 0
gareverb2       init 0
gadelayleft     init 0
gadelayrght     init 0

;-----------------------------------------------------------------------------------------------
; Detuned pluck with different center frequencies and bandwidth tables - method 4
;-----------------------------------------------------------------------------------------------
          instr 1
iamp              = ampdb(p4)
ibend             = p6
iexpfn  = p7
ifn               = p8
iparm1 = p9
iparm2 = p10
idegrstrt         = p11
idegrend          = p12
idiststrt         = p13
idistend          = p14
ieffect  = p15
          ipch1   = p5
          ipch2   = ipch1 * 1.009
          ipch3   = ipch1 * .9984
          kfq1 init ipch1
          kfq2 init ipch2
          kfq3 init ipch3
kbend  oscil1i 0.15, ibend, p3, iexpfn
khz1    = (kfq1 + kbend)
khz2    = (kfq2 + kbend)
khz3    = (kfq3 + kbend)
kvib oscil 1/120, ipch1/50, 1
```

```
apluck1         pluck (p4/2.3), (khz1+kvib), ipch1, ifn, 4, iparm1, iparm2
apluck2         pluck (p4/1.6), khz2, ipch2, ifn, 4, iparm1, iparm2
apluck3         pluck (p4/1.6), khz3, ipch3, ifn, 4, iparm1, iparm2
        asignal = apluck1+apluck2+apluck3
        asig    envlpx asignal, .001, p3, (p3*.008), 4, .52, .08
kdegree         line idegrstrt, p3, idegrend
kdistance       line idiststrt, p3, idistend
asig1, asig2 locsig asig, kdegree, kdistance, .1
asig12, asig22 locsend

outs asig1, asig2

        if ( p15 != 1 ) goto reverb97
        gareverb = gareverb + asig12
        gareverb = gareverb + asig22
reverb97:
        if ( p15 != 2 ) goto reverb98
        gareverb1 = gareverb1 + asig12
        gareverb2 = gareverb2 + asig22
reverb98:
        if ( p15 != 3 ) goto delay99
        gadelayleft = gadelayleft + asig12
        gadelayrght = gadelayrght + asig22
delay99:

endin
```

```
;-----------------------------------------------------------------------------------------------
; Detuned pluck with different center frequencies and bandwidth tables - method 5
;-----------------------------------------------------------------------------------------------

        instr 2

iamp            = ampdb(p4)
ibend           = p6
ibndfn  = p7
ifn             = p8
iparm1 = p9
iparm2 = p10
idegrstrt       = p11
idegrend        = p12
idiststrt       = p13
idistend        = p14
ieffect  = p15
        ipch1   = p5
        ipch2   = ipch1 * 1.008
        ipch3   = ipch1 * .9994
        kfq1 init ipch1
        kfq2 init ipch2
        kfq3 init ipch3
kbend  oscil1i 0.15, ibend, p3, ibndfn
khz1    = (kfq1 + kbend)
khz2    = (kfq2 + kbend)
khz3    = (kfq3 + kbend)
kvib oscil 1/120, ipch1/50, 1
apluck1         pluck (p4/2.3), (khz1+kvib), ipch1, ifn, 5, iparm1, iparm2
apluck2         pluck (p4/1.6), khz2, ipch2, ifn, 5, iparm1, iparm2
apluck3         pluck (p4/1.6), khz3, ipch3, ifn, 5, iparm1, iparm2
        asignal = apluck1+apluck2+apluck3
        asig    envlpx asignal, .001, p3, (p3 * .008), 4, .52, .08

kdegree         line idegrstrt, p3, idegrend
kdistance       line idiststrt, p3, idistend
asig1, asig2 locsig asig, kdegree, kdistance, .1
asig12, asig22 locsend
outs asig1, asig2
```

```
        if ( p15 != 1 ) goto reverb97
        gareverb = gareverb + asig12
        gareverb = gareverb + asig22
reverb97:
        if ( p15 != 2 ) goto reverb98
        gareverb1 = gareverb1 + asig12
        gareverb2 = gareverb2 + asig22
reverb98:
        if ( p15 != 3 ) goto delay99
        gadelayleft = gadelayleft + asig12
        gadelayrght = gadelayrght + asig22
delay99:

endin


;-----------------------------------------------------------------------------------------------
; Iteration buzz filtered with center frequency and band width dinamic variations
;-----------------------------------------------------------------------------------------------

        instr 3

idur      = abs(p3)
iamp      = ampdb(p4)
ifqc      = p5
ifn       = p6
icf1      = p7
icf2      = p8
ikbw1     = p9
ikbw2     = p10
irate1    = p11
irate2    = p12
idgstrt   = p13
idgend    = p14
idststrt  = p15
idstend   = p16
ieffect   = p17
irevgain init 2
knh       = int((sr*.5)/ifqc)
krate     line irate1, idur, irate2
kenv      oscil iamp, (1/idur)*krate, ifn
```

```
kcf       expon  icf1, idur, icf2
kbw       line ikbw1, idur, ikbw2
abuzz     buzz 1,ifqc, knh, 1
afilt     reson abuzz, kcf, kbw, 1
asig      balance afilt, abuzz
asig = asig*kenv
kdegree          line idgstrt, p3, idgend
kdistance        line idststrt, p3, idstend
asig1, asig2 locsig asig, kdegree, kdistance, .1
asig12, asig22 locsend
outs asig1, asig2
          if ( p17 != 1 ) goto reverb97
          gareverb = gareverb + asig12
          gareverb = gareverb + asig22
reverb97:
          if ( p17 != 2 ) goto reverb98
          gareverb1 = gareverb1 * irevgain + asig12
          gareverb2 = gareverb2 * irevgain + asig22
reverb98:
          if ( p17 != 3 ) goto delay99
          gadelayleft = gadelayleft * irevgain + asig12
          gadelayrght = gadelayrght * irevgain + asig22
delay99:

endin

;-----------------------------------------------------------------------------------------------
;Gbuzz bass with  harmonics
;-----------------------------------------------------------------------------------------------

          instr 4

iamp      = ampdb(p4)
ifqc      = p5
knh       = int((sr*.5)/ifqc)
ilh       = p6
idegrstrt = p7
idegrend  = p8
idiststrt = p9
idistend  = p10
ieffect   = p11
```

```
kenv            expseg 100, p3*.6, 300, p3*.4, 200
kfreq           envlpx kenv, 1, p3, .001, 11, .6, .01
aenv            linen iamp, .5, p3, 1
abuzz           gbuzz aenv,ifqc, knh, ilh, 1, 12
afilt1   butterbp abuzz, 55+kfreq, 20
afilt2   butterbp afilt1, 110+kfreq, 30
afilt3   butterbp afilt2, 164.8+kfreq, 40
afilt4   butterbp afilt3, 220+kfreq, 50
asig            balance afilt4, abuzz

kdegree         line idegrstrt, p3, idegrend
kdistance       line idiststrt, p3, idistend
asig1, asig2 locsig asig, kdegree, kdistance, .1
asig12, asig22 locsend

outs asig1, asig2

        if ( p11 != 1 ) goto reverb97
        gareverb = gareverb + asig12
        gareverb = gareverb + asig22
reverb97:
        if ( p11 != 2 ) goto reverb98
        gareverb1 = gareverb1 + asig12
        gareverb2 = gareverb2 + asig22
reverb98:
        if ( p11 != 3 ) goto delay99
        gadelayleft = gadelayleft + asig12
        gadelayrght = gadelayrght + asig22
delay99:

endin
```

```
;-----------------------------------------------------------------------------------------------
; Random oscilator
;-----------------------------------------------------------------------------------------------

        instr 5

iamp    = ampdb(p4)
ipitch  = p5
idev    = p6
idspl   = p7
ienvfn = p8
ifn     = p9
idgrst  = p10
idgrend = p11
idstst  = p12
idstend = p13
ieffect = p14

krandh randh idev, idspl

awave  oscili  iamp, 1/p3, ifn
aosc1  oscili .2 * awave, .9999 * ipitch + krandh, ienvfn;35 36 37 38
aosc2  oscili .3 * awave, 1.0001 * ipitch + krandh, ienvfn
asig = aosc1 + aosc2

kdegree         line idgrst, p3, idgrend
kdistance       line idstst, p3, idstend
asig1, asig2 locsig asig, kdegree, kdistance, .1
asig12, asig22 locsend

outs asig1, asig2

        if (p14 != 1) goto reverb97
        gareverb = gareverb + asig12
        gareverb = gareverb + asig22
reverb97:
        if (p14 != 2) goto reverb98
        gareverb1 = gareverb1 + asig12
        gareverb2 = gareverb2 + asig22
```

```
reverb98:
        if (p14 != 3 ) goto delay99
        gadelayleft = gadelayleft + asig12
        gadelayrght = gadelayrght + asig22
delay99:

endin

;-----------------------------------------------------------------------------------------------
; Global reverberation
;-----------------------------------------------------------------------------------------------

instr 97
irvrbtime       = p4
istart  = p5
iend            = p6

kbalance        line istart, p3, iend
areverb nreverb gareverb, irvrbtime, 1
outs    areverb * sqrt(kbalance), areverb * sqrt(1- kbalance)
gareverb = 0
endin


;-----------------------------------------------------------------------------------------------
; Global reverberation with delayed echos
;-----------------------------------------------------------------------------------------------

        instr 98

irevtime        = p4
idif            = p5
itime           = p6
areverb1        nreverb gareverb1, irevtime, idif
areverb2        nreverb gareverb2, irevtime, idif
asig1 delay areverb1, itime
asig2 delay asig1, itime
asig3 delay asig2, itime
asig4 delay asig3, itime
asig5 delay areverb2, itime/.2
asig6 delay asig5, itime
asig7 delay asig6, itime
asig8 delay asig7, itime
```

```
outs (asig1 + asig3 + asig5 + asig7)/4, (asig2 + asig4 + asig6 + asig8)/4

gareverb1 = 0
gareverb2 = 0
endin

;-----------------------------------------------------------------------------------------------
; Global reverb delay
;-----------------------------------------------------------------------------------------------

          instr 99

ifn       = p4
istart = p5
iend      = p6

kbalance        line istart, p3, iend
areverb1 nreverb gadelayleft, 3, .5
aline oscili 2, 1/p3, ifn
a1        delayr 1
adl1      deltapi aline
adl2      deltapi aline * 1.5
          delayw areverb1

areverb2 nreverb gadelayrght, 3, .5
a2        delayr 1
adl3      deltapi aline * 1.9
adl4      deltapi aline * 2.8
          delayw areverb2

aenv      linen    2, .1, p3, .1
asig1 = (areverb1 + adl1 + adl2) * aenv
asig2 = (areverb1 + adl3 + adl4) * aenv
outs      asig1 * sqrt(kbalance), asig1 * sqrt(1- kbalance)
outs      asig2 * sqrt(kbalance), asig2 * sqrt(1- kbalance)
gadelayleft       = 0
gadelayrght       = 0
endin
```

CHAPTER 6: GEN FUNCTIONS AND MOST USED WAVE AND ENVELOP TABLE DIAGRAMS

Tables of the instruments are presented according to the numbers they received.

Sine wave - Instrument 3.

f1 0 8192 10 1

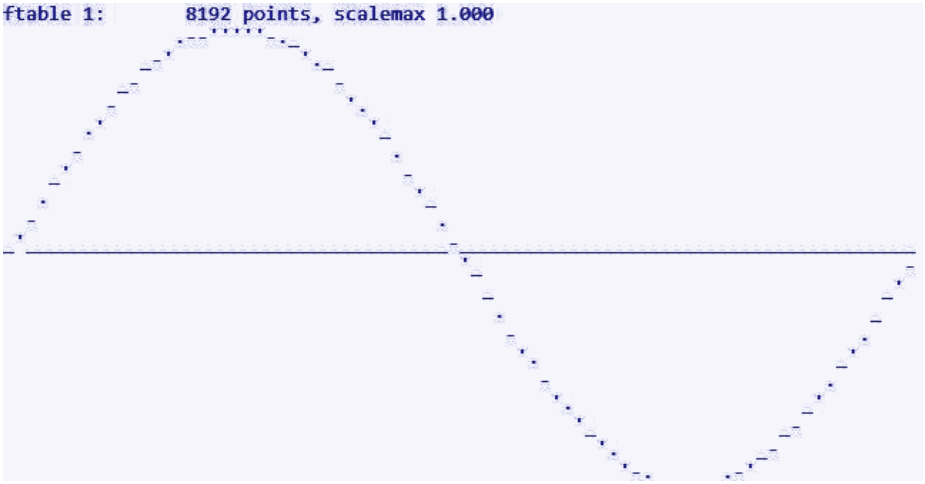

Figure 4: sine wave for buzz opcode - instrument 3

Pitch bend tables, used in instruments 1 & 2.

Linear rise

f2 0.0 513 7 0 513 1

Linear fall

f3 0.0 513 7 1 513 0

Exponential rise

f4 0.0 513 5 .001 300 1 213 1

Exponential fall

f5 0.0 513 5 1 513 .001

Buzz-like wave

f6 0.0 512 10 1 1 1 1 1 1 1 1

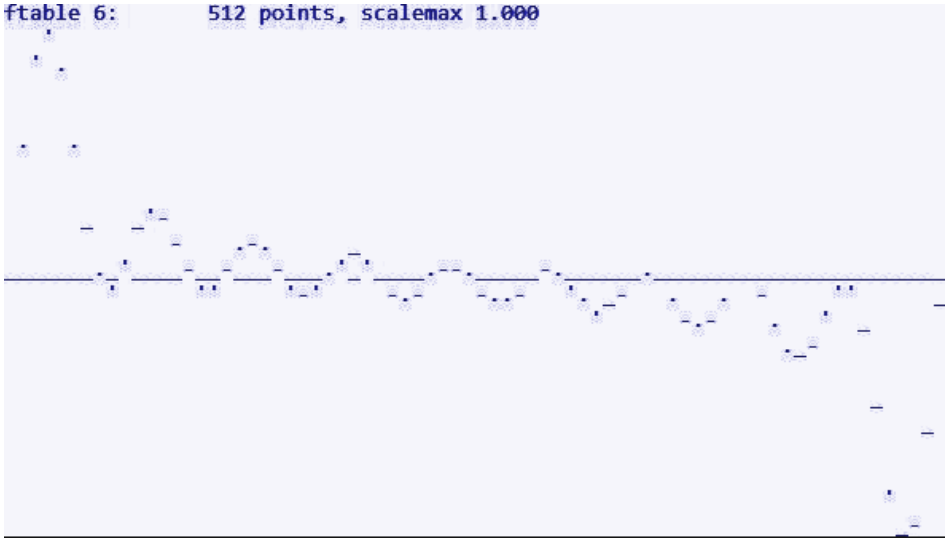

Figure 5: buzz-like wave for instruments 1 & 2.

Instruments 1 and 2 - pluck fn decay buffer

```
f20  0  1024  1   "instr3.aif"    .1    0  0; cf = 6000 bw = 1000
f21  0  1024  1   "instr3.aif"   1.1    0  0; cf = 3000 bw =  750
f22  0  1024  1   "instr3.aif"   2.1    0  0; cf = 2500 bw =  500
f23  0  1024  1   "instr3.aif"   3.1    0  0; cf = 2000 bw =  250
f24  0  1024  1   "instr3.aif"   4.1    0  0; cf = 1500 bw =  200
f25  0  1024  1   "instr3.aif"   5.1    0  0; cf = 1000 bw =  150
f26  0  1024  1   "instr3.aif"   6.1    0  0; cf =  750 bw =  100
f27  0  1024  1   "instr3.aif"   7.1    0  0; cf =  500 bw =   75
f28  0  1024  1   "instr3.aif"   8.1    0  0; cf =  400 bw =   50
f29  0  1024  1   "instr3.aif"   9.1    0  0; cf =  300 bw =   25
```

Kenv tables of instrument 3

```
f9 0 512 7 0 16 1 496 0
f10 0 512 7 0 256 1 256 0
```

Tables of instrument 4

Exponential rise

```
f11 0.0 513 5   .01 513 .5
```

Single cosine wave for gbuzz

f12 0 8193 9 1. 1. 90

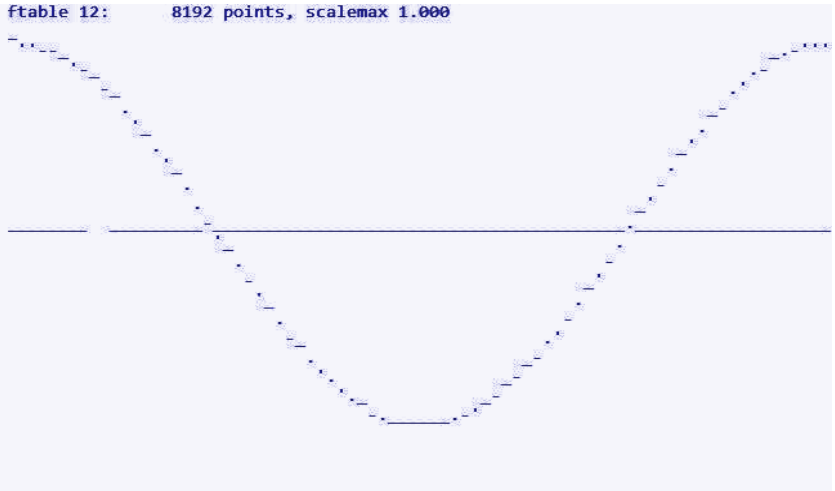

Figure 6: cosine wave - instrument 4

Oscil tables of instrument 5

f31 0 2048 10 0 1 1 0 0 0 0

f32 0 2048 10 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

f33 0 2048 10 1 1 1 1

f34 0 2048 10 0 1 2 3

f35 0 1024 7 0 128 10 256 10 256 -10 256 -10 128 0

f36 0 1024 7 0 6 1 200 1 12 -1 800 -1 6 0

f37 0 1024 7 0 500 1 24 -1 500 0

f38 0 1024 7 0 256 1 256 0 256 -1 256 0

Envelop tables for instrument 5

f45 0 512 7 0 22 1 490 0

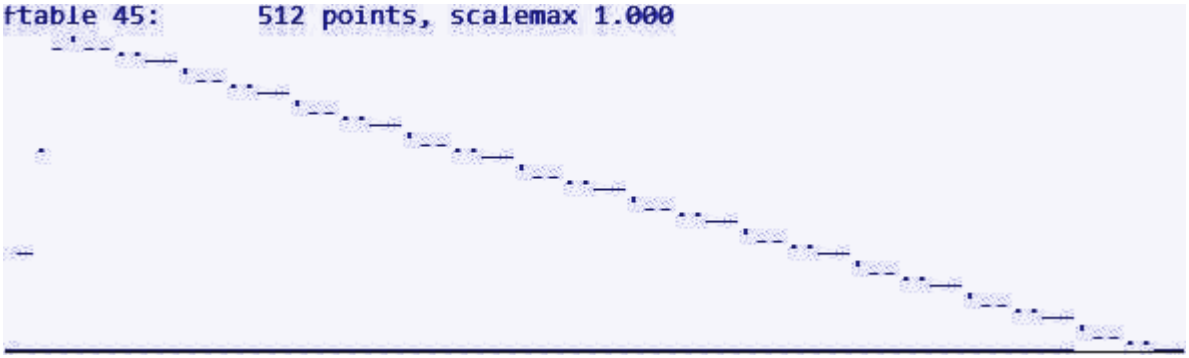

Figure 7: short attack long decay envelope.

f46 0 512 7 0 50 1 50 0.5 300 0.5 112 0

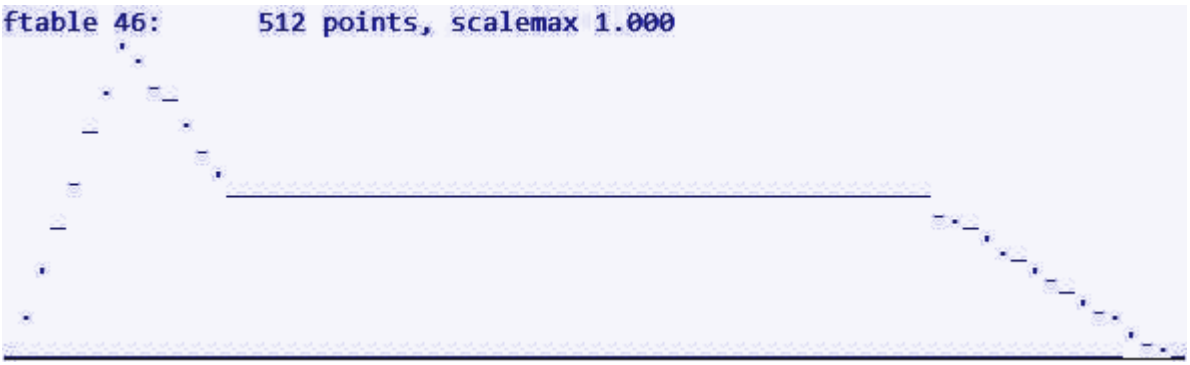

Figure 8: ADSR envelope

f47 0 512 7 0 30 1 32 0.5 450 0

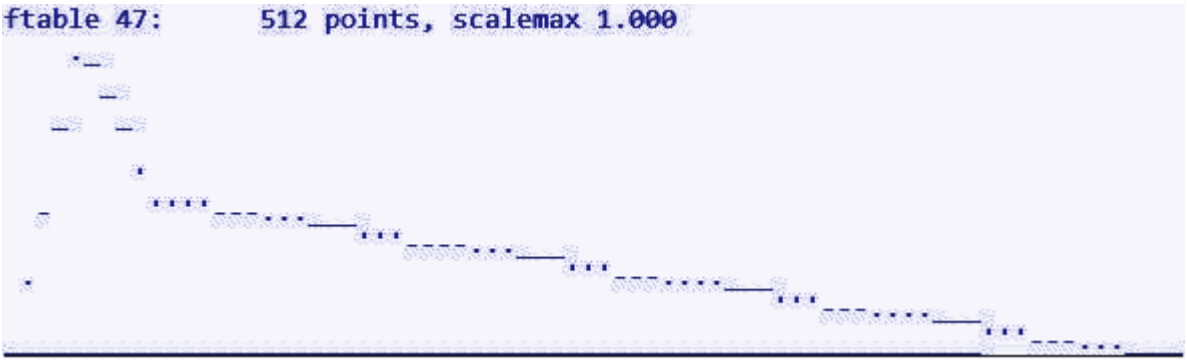

Figure 9: ADR envelope

f48 0 512 7 0 25 1 25 0.4 55 0.3 55 0.2 176 0.1 176 0

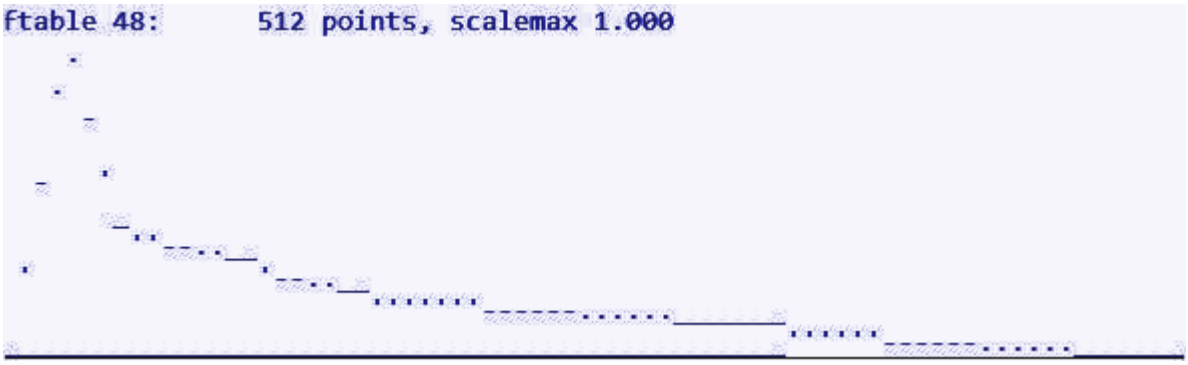

Figure 10: short attack and decay exponential release

Tables for instr 98

f7 0 4096 -5 .001 4096 .2

f8 0 4096 -7 .001 4096 .1

CHAPTER 7: PYTHON SCRIPTS AND LIST OF CSOUND EVENTS

```
#Koch curve score – Python script score generator
def gen_koch_curve1 (numOfNotes):
scoreText = "";
pitch = 164.81; # E3
increment = 68.27; # Init A4th
for i in range(numOfNotes):
scoreText += "i1 " + str(i) + " 1 140 "+str(("%.2f" % pitch)) +" 1 4 23 0 1.9 1 50 6 30 1 \n";
scoreText = scoreText.replace('\\n', '\n');
pitch = pitch+increment;
return(scoreText);

#i1 0 1.2 150 496.0 3 5 23 0.04 4.95 1 50 1 50 3
#score = gen_koch_curve1 (3)

def gen_koch_curve2 (numOfNotes):
scoreText = "";
pitch = 392; # G4
increment = 131.25; # Init P4th
for i in range(numOfNotes):
scoreText += "i2 " + str(i) + " 1 180 "+str(("%.2f" % pitch)) +" 1 1 20 0.6 0.4 60 10 5 1 3 \n";
scoreText = scoreText.replace('\\n', '\n');
pitch = pitch+increment;
return(scoreText);
```

```
def gen_koch_curve3 (numOfNotes):
scoreText = "";
pitch = 293.66; # D4
increment = 98.34; # Init P4th
for i in range(numOfNotes):
scoreText += "i5 " + str(i) + " 1 90 "+str(("%.2f" % pitch)) +" 1 4 31 45 55 10 10 1 2 \n";
scoreText = scoreText.replace('\\n', '\n');
pitch = pitch+increment;
return(scoreText);

#Python objects (examples):

score = gen_koch_curve1 (3)
score += gen_koch_curve3 (3)

i1      262.4848 1.0050964355 140 465.81  1 4 23  0 1.9    1 50 6  30 1
i1      263.4899 1.0050964355 140  534.08 1 4 23  0 1.9    1 50 6  30 1
i1      264.495   1.0050964355 140  602.35 1 4 23  0 1.9    1 50 6  30 1
i5      262.4848 1.0050964355   90  594.66003 1 4 31 45   55 10   10 1 2
i5      263.4899 1.0050964355   90  693.0        1 4 31 45   55 10   10 1 2
i5      264.495   1.0050964355   90  791.33997 1 4 31 45   55 10   10 1 2
```

```
score = gen_koch_curve2 (6)
score += gen_koch_curve3 (6)

i2      677.0991 1.0092595816   180 392.00   1 1 20 0.6 0.4 60 10 5 1 3
i2      678.1084 1.0092595816   180 523.25   1 1 20 0.6 0.4 60 10 5 1 3
i2      679.1176 1.0092595816   180 654.50   1 1 20 0.6 0.4 60 10 5 1 3
i2      680.1269 1.0092595816   180 785.75   1 1 20 0.6 0.4 60 10 5 1 3
i2      681.13617 1.0092595816 180 917.00   1 1 20 0.6 0.4 60 10 5 1 3
i2      682.14545 1.0092595816 180 1048.25 1 1 20 0.6 0.4 60 10 5 1 3
i5      677.0991   1.0092595816 90   293.66   1 4 31 45  55  10 10 1 2
i5      678.1084   1.0092595816 90  392.00    1 4 31 45  55  10  10 1 2
i5      679.1176   1.0092595816 90  490.34    1 4 31 45  55  10  10 1 2
i5      680.1269   1.0092595816 90  588.68    1 4 31 45  55  10  10 1 2
i5      681.13617 1.0092595816 90  687.02    1 4 31 45  55  10  10 1 2
i5      682.14545 1.0092595816 90  785.36    1 4 31 45  55  10  10 1 2
```

VITA

University of Washington Joaquim R. Freire-Neto studied music and had private piano lessons in his youth at the Conservatory of Music Alberto Nepomuceno in Fortaleza, Brazil. After completing High School he decided to pursue a career in music. His Undergraduate degree in Music is from the State University of Ceará, Brazil. His Master of Fine Arts is from the University of Io wa where he studied composition with Richard Hervig and Donald Genni. He received his Doctorate of Musical Arts from the University of Washington where he studied composition with Joël-François Durand, William Smith and Kenneth Benshoof. He also studied computer music and composition with Richard Karpen. He began his music career as conductor of the Technical School of Piauí Choir in Teresina, Brazil. Later, he held the Choral director Position at the Federal University of Piauí (UFPI), where he currently is on the music faculty. At this institution, he has been teaching Tonal Harmony, Tonal Counterpoint, Musical Analysis and Music Technology. He was the editor of the periodical *Educação e Compromisso* (Education and Compromise) during the time he held the position of Vice-director of the Center of Science and Education at UFPI. He has given private composition lessons and has produced compositions for a variety of media, among them pieces for voice and computer generated sound, string quartet, duos for flute and clarinet, piano and percussion ensembles, and saxophone quartet. He has been composing algorithmic compositions using Csound to generate instruments and Python for score coding. His music has been performed in cities throughout Brazil and the U.S.A.