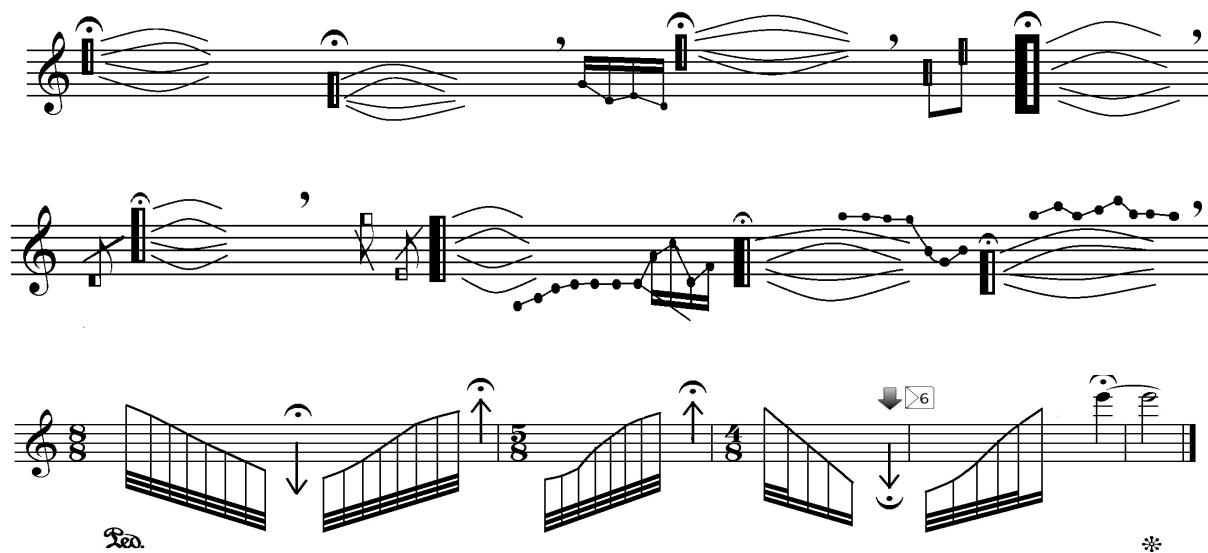


Seção IV: Gestos Interativos

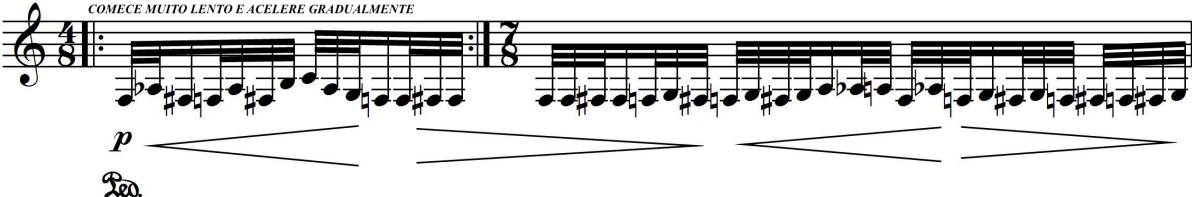


Seção V: Muito leve e sereno

The musical score for Seção V: Muito leve e sereno consists of two staves. The first staff is marked with a treble clef and a 6/8 time signature, showing a series of chords and single notes with dynamic markings like *pp* and *mp*, and a final asterisk. The second staff continues this pattern with more complex rhythmic markings and a series of dots above the notes. The third staff is marked with a treble clef and a 3/8 time signature, showing a series of chords and single notes with dynamic markings like *pp* and *mp*, and a final asterisk.

Seção VI: *Improvise ligando uma célula melódica à outra*

COMECE MUITO LENTO E ACELERE GRADUALMENTE



p
Reo.

IMPROVISE LIGANDO UMA CÉLULA MELÓDICA A OUTRA



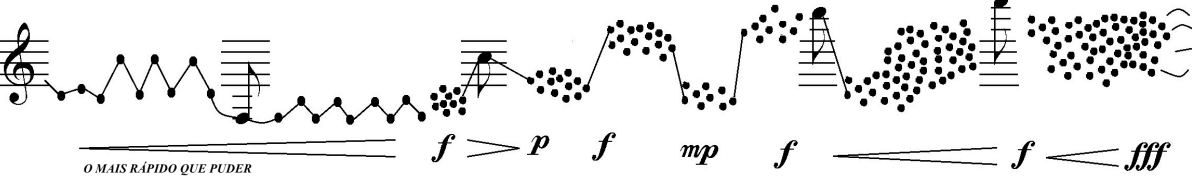
mf

SEMPRE RÁPIDO E CRESCENDO

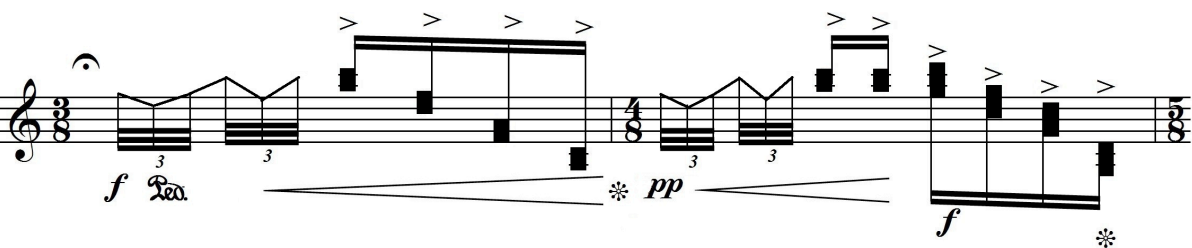


f

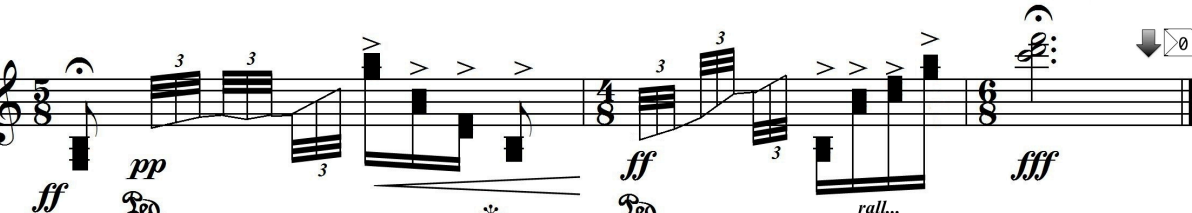
O MAIS RÁPIDO QUE PUDER



f *p* *f* *mp* *f* *f* *fff*
Reo.



f *pp* *f*
Reo.



ff *pp* *ff* *fff*
Reo. rall...

**APÊNDICE B- CD ROOM: PATHC NÉVOAS & CRISTAIS/ PARTITURA
REVISADA/ GRAVAÇÃO DE DANIELA LEITE (2012, 2013)/ GRAVAÇÃO DE
ANDRÉ JUAREZ (1997)**

APÊNDICE C- PROGRAMA DO RECITAL APRESENTADO NA DEFESA

1. Iluminura - 2010

Carlos Menezes

2. Namíbia - 2011

Cesar Traldi

3. Sxueak - 2008

Matthew Burtner

Participação: Cesar Traldi

4. Névoas e Cristais - 1995

Jônatas Manzolli

ANEXO A- Partitura original- Névoas & Cristais (1995) – Jônatas Manzolli

Névoas & Cristais

Jônatas Manzolli

*Composição Interativa para Vibrafone e
Computador*

para André Juarez

Nota de Programa

Névoas & Cristais explora o Vibrafone em dois contextos. Sons rápidos e curtos - densidades que formam nuvens sonoras. Acordes pontuais, clusters e brilho - fragmentos que geram cristais sonoros.

A interação entre o músico e o computador é usada para multiplicar a sonoridade da peça. Partindo dos dois elementos básicos, ela potencializa gestos musicais através do diálogo entre o intérprete e o computador.

Especificações

. **Vibrafone:** deverá ser um instrumento comum com um adaptador MIDI, ou seja, as notas tocadas serão transformadas em código MIDI e enviadas ao computador através de um cabo MIDI.

. **Computador:** o programa "Névoas & Cristais" foi escrito pelo autor em LISP. Ele roda em qualquer computador que tenha compilador COMMOM LISP e placa MIDI.

. **Módulo de Som:** qualquer módulo MIDI que tenha um bom som de Vibrafone.

. **Processador:** qualquer módulo que tenha um DEEP REVERBER.

. **Conexões:** serão feitas de acordo com o diagrama abaixo

Performance & Notação

A explicação será dada por seções da peça:

I - COMO SE ESTIVESSE VINDO DE LONGE

Introduz o Tema das Névoas na peça. As repetições podem ser tocadas ou não.

II - COM O BRILHO DE UM CRISTAL

Solo de Vibrafone que introduz a sonoridade dos cristais. O computador ficará em silêncio.

III- DIALOGANDO COM O COMPUTADOR

Este trecho deve ser tocado como se houvesse uma única linha melódica entre o Vibrafone e o som do computador. O número de notas da resposta do computador está descrito a cada compasso. De modo geral, todas as notas tocadas pelo vibrafone e pelo computador terão a Duração de 01 Colcheia.

IV - GESTOS INTERATIVOS

Neste trecho o intérprete estará livre para improvisar acordes e cluster. Há uma série de gestos sugeridos na partitura que podem ser mudados e não há necessidade de tocá-los sequencialmente. A cada quatro notas tocadas pelo músico, o computador reage gerando uma trajetória complexa. O músico deve ouvi-la e criar um novo gesto após o término da sequência computacional. Movimentos contextuais que criem elementos cênicos são bem vindos.

V - MUITO LEVE E SERENO

Trecho lírico onde os acordes evocam os cristais sonoros. Cada acorde deve ser tocado cuidadosamente de modo a criar uma atmosfera de brilho e luz.

VI - IMPROVISE LIGANDO UMA CELULA MELODICA A OUTRA

Final da obra e seu clímax. O intérprete deverá desenvolver uma grande nuvem sonora, intensa e forte. Partindo do Tema das Névoas, ele deverá improvisar um crescendo gradual de intensidade e de massa sonora. O computador auxiliará tal tarefa multiplicando o som de cada evento sonoro. As trajetórias sugeridas para improvisação deverão ser seguidas como referência de estudo e ensaio. O músico fica livre para criar a sua própria versão de Nébulas Sonoras em performance. O importante é manter a sonoridade em intenso crescendo até o seu clímax.

Mudança de Programa

A seta colocada na parte superior da partitura indica Mudança de Programa do Computador. O intérprete deverá apertar o pedal de mudança e esperar até que o computador indique o novo programa. O número do programa é indicado na parte superior da partitura em algarismos romanos. Em caso de ensaio, o intérprete pode escolher aleatoriamente o programa equivalente à seção da peça.

Névoas & Cristais

PROGRAMA I
COMO SE ESTIVESSE SOB UMA CORTINA DE FUMAÇA

João Carlos Nancollini

pp
ppp
p

pp
f
mp

COM O BRILHO DE UM CRISTAL

PROGRAMA II

fz
pp
rall...

ff
fz
pp
f
pp

fz
p
fz
p
fz
p

COM ACENTOS SUAVES

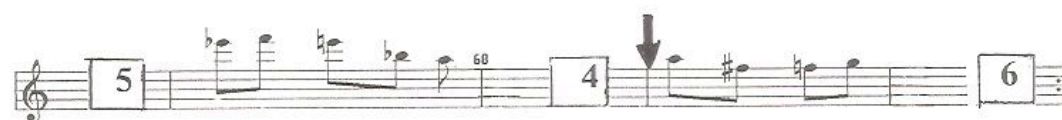
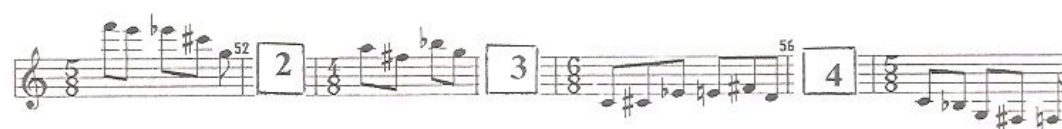
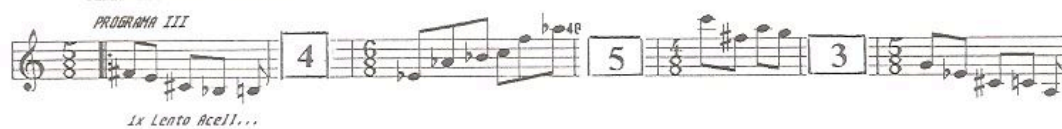
fz
mf
f
p

ACENTUANDO COM MAIS VIGOR

mf
f
rall...
nota rall...
pp

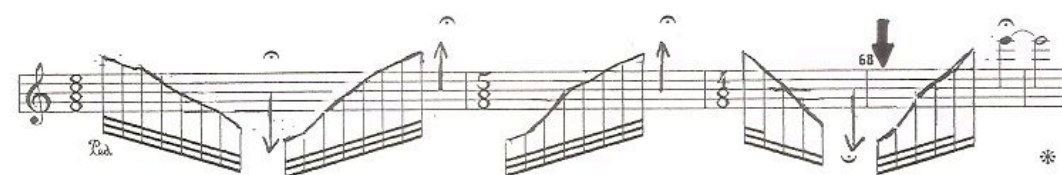
DIALOGANDO COM O COMPUTADOR

PROGRAMA III



GESTOS INTERATIVOS

PROGRAMA IV



MUITO LEVE E SERENO

PROGRAMA V

72

pp

mp

pp

76

CONCEE MUITO LENTO E ACELERE GRADUALMENTE

PROGRAMA VI

pp

ppp

p

IMPROVISE LIGANDO UMA CELULA MELODICA A OUTRA

mf

SEMPRE RAPIDO E CRESCENDO

ff

pp

ff

fff

rall ...

ANEXO B- PROGRAMAÇÃO ORIGINAL EM *KC-LISP*

```

; 01/02/96
; Jonatas Manzolli
; Programa para performance atraves de reconhecimento
; de Padroes tipo Strokes

; ===== Funcoes para MIDI IN and OUT =====

(global '(notebuffer velocitybuffer rhythmbuffer))

(defun alea(x y)
  (mapvalue (random) 0 16777215 x y))

; ===== Manipulacoes de buffers

(defun defaultBuffers()
  (put 'midi 'size 32)
  (setq notebuffer (mkvect 31))
  (setq velocitybuffer (mkvect 31))
  (setq rhythmbuffer (mkvect 31)))

(defun resetbuffers(size)
  (when (> size 0)
    (writenote (nth (alea 0 10) '(64 32 54 96 68 74 66 27 65 72 54 34 63 62)))
    (writevelocity (nth (alea 0 10) '(13 92 12 32 56 14 32 123 18 23 12 67 18)))
    (writerhythm (manypulses))
    (resetbuffers (- size 1))))

(defun resetperformance(timing)
  (put 'performance 'ciclo '(playciclo 0.57))
  (put 'performance 'retrociclo '(retroplaynote 0.33))
  (put 'performance 'arpejo '(playarpejo 0.27))
  (put 'performance 'aleaciclo '(aleaplaynote 0.53))
  (put 'performance 'inverciclo '(inverplaynote 0.47)))

(defun resetpointers()
  (resetperformance)
  (put 'control 'arpejo 15)
  (put 'midi 'ciclo 'off)
  (put 'midi 'retrociclo 'off)
  (put 'midi 'arpejo 'off)
  (put 'midi 'aleaciclo 'off)
  (put 'midi 'inverciclo 'off)
  (put 'midi 'bend 'off)
  (put 'midi 'modula 'off)
  (put 'midi 'volume 'off)
  (put 'midi 'pan 'off)
  (put 'midi 'length 250)
  (put 'midi 'flag 3)
  (put 'midi 'filtro '[144])
  (put 'midi 'channel 1)
  (put 'midi 'mouse 2)
  (put 'midi 'notein 0)
  (put 'midi 'noteout 64)
  (put 'midi 'velin 0)
  (put 'midi 'velout 64)
  (put 'midi 'rhyin 0)
  (put 'midi 'rhyout 64))

(defun changeindex(index)
  (put 'midi index (rem (+ (get 'midi index) 1)(get 'midi 'size))))

```

```

===== rotinas de escrita

(defun writenote(x)
  (putv notebuffer (get 'midi 'notein) x)
  (changeindex 'notein))

(defun writevelocity(x)
  (putv velocitybuffer (get 'midi 'velin) x)
  (changeindex 'velin))

(defun writerrhythm(x)
  (putv rhythmbuffer (get 'midi 'rhyin) x)
  (changeindex 'rhyin))

; ===== rotinas de leitura

(defun retroreadnote()
  (changeindex 'noteout)
  (getv notebuffer (- 32 (get 'midi 'noteout))))

(defun readnote()
  (changeindex 'noteout)
  (getv notebuffer (get 'midi 'noteout)))

(defun readvelocity()
  (changeindex 'velout)
  (getv velocitybuffer (get 'midi 'velout)))

(defun readrhythm()
  (changeindex 'rhyout)
  (/ (- (getv rhythmbuffer (get 'midi 'rhyin))
        (getv rhythmbuffer (get 'midi 'rhyout))) 200.0))

; ===== MIDI =====

(defun clearMIDIBuffer()
  (when (some-midi)
    (look-at-status)
    (look-at-first-data-byte)
    (look-at-second-data-byte)
    (clearMIDIBuffer)))

(defun readMIDI(flag filtro channel)
  (when (some-desired-midi filtro)
    (if (or (= flag 1) (= flag 3))
        (readMIDIecho flag (look-at-first-data-byte)
                      (look-at-second-data-byte) channel)
        (readMIDI* flag (look-at-first-data-byte)
                      (look-at-second-data-byte) channel))))

(defun readMIDIecho(flag note velocity channel)
  (midiout (+ 143 channel))
  (midiout note)
  (midiout velocity)
  (readMIDI* flag note velocity channel))

(defun readMIDI*(flag note velocity channel)
  (when (> velocity 0)
    (when (or (= flag 2) (= flag 3))

```

```

    (playmouse channel (* note 5) (* velocity 3))
    (playnotebuffer channel))
  (writenote note)
  (writevelocity velocity)
  (writerhythm (manypulses)))
  (list note velocity))

(defun playnotebuffer(channel)
  (midiout (+ 143 channel))
  (midiout (readnote))
  (midiout (readvelocity)))

(defun playciclo()
  (playnotebuffer 2))

(defun retroplaynote()
  (midiout (+ 143 3))
  (midiout (retroreadnote))
  (midiout (readvelocity)))

(defun playarpejo()
  (if (> (get 'control 'arpejo) 15) (put 'control 'arpejo 0))
  (playarpejo* (get 'control 'arpejo)))

(defun playarpejo*(x)
  (if (< x 7)
    (inverplaynote 4))
  (put 'control 'arpejo (+ x 1)))

(defun aleaplaynote
  (midiout (+ 143 5))
  (midiout (getv notebuffer (alea 0 31)))
  (midiout (getv velocitybuffer (alea 0 31))))

(defun inverplaynote
  (midiout (+ 143 6))
  (midiout (invertenote (readnote)))
  (midiout (readvelocity)))

(defun invertenote(x)
  (invertenote* (rem x 12) (/ x 12)))

(defun invertenote*(pitch oct)
  (+ (- 12 pitch) (* 12 oct)))

(defun playmouse(channel x y)
  (modulamouse channel (/ x 6) (/ y 4))
  (midiout (+ 143 channel))
  (midiout (getv notebuffer (/ x 20)))
  (midiout (getv velocitybuffer (/ y 13))))

(defun modulamouse(channel x y)
  (if (= (get 'midi 'bend) 'on)
    (pitchbend channel (abs (- x y))))
  (if (= (get 'midi 'modula) 'on)
    (pitchmodula channel (/ (+ x y) 2)))
  (if (= (get 'midi 'volume) 'on)
    (pitchvolume channel (- 128 y)))
  (if (= (get 'midi 'pan) 'on)
    (pitchpan channel x)))

```

```

; funcoes de modulacao

(defun pitchmodula(channel x)
  (midicommand (+ 175 channel) 1 x))

(defun pitchvolume(channel x)
  (midicommand (+ 175 channel) 7 x))

(defun pitchpan(channel x)
  (midicommand (+ 175 channel) 10 x))

(defun pitchbend(channel x)
  (midicommand (+ 223 channel) 64 x))

; ----- Base de Dados -----
; lista de padroes e similaridades
; todos os padroes que forem considerados iguais
; pertence a mesma sub-lista
; cada sub-lista sera entao chamada de stroke

(defun midicontrollers()
  (list
    (get 'midi 'circular)
    (get 'midi 'convexo)
    (get 'midi 'letra_c)
    (get 'midi 'ferradura)
    (get 'midi 'letra_u)
    (get 'midi 'cruz_direita_sobe)
    (get 'midi 'cruz_desce_direita)
    (get 'midi 'letra_z)
    (get 'midi 'letra_l_deitada)
    (get 'midi 'moldura_inferior)
    (get 'midi 'letra_l)
    (get 'midi 'moldura_superior)
    (get 'midi 'letra_j)
    (get 'midi 'letra_v)
    (get 'midi 'diagonal_cima)
    (get 'midi 'diagonal_baixo)
    (get 'midi 'vertical)
    (get 'midi 'horizontal)))

(defun userstrokespadroes()
  (list
    (get 'userpattern 'circular)
    (get 'userpattern 'convexo)
    (get 'userpattern 'letra_c)
    (get 'userpattern 'ferradura)
    (get 'userpattern 'letra_u)
    (get 'userpattern 'cruz_direita_sobe)
    (get 'userpattern 'cruz_desce_direita)
    (get 'userpattern 'letra_z)
    (get 'userpattern 'letra_l_deitada)
    (get 'userpattern 'moldura_inferior)
    (get 'userpattern 'letra_l)
    (get 'userpattern 'moldura_superior)
    (get 'userpattern 'letra_j)
    (get 'userpattern 'letra_v)
    (get 'userpattern 'diagonal_cima)
    (get 'userpattern 'diagonal_baixo)

```

```

(get 'userpattern 'vertical)
(get 'userpattern 'horizontal)))

(defun strokespadroes()
  '(
    ; linhas horizontais
    ((0 1 2)(3 4 5)(6 7 8)(0 1)(1 2)(3 4)(4 5)(0 1 4 5)(3 0 1 2))
    ; linhas verticais
    ((0 3 6)(1 4 7)(2 5 8)(0 3)(3 6)(1 4)(4 7)(2 5)(5 8)(0 3 6 7))
    ; diagonais
    ((0 1 5)(0 3 4 5 8)(0 1 4 7 8)(0 1 4 5 8)(0 3 4 7 8)(0 1 4 8)(0 4 5 8)
      (0 4 7 8)(0 4 8)(3 7)(1 5)(0 3 4 5)(0 4 5)(0 3 4 8)(0 1 5)(0 4 7)
      (0 3 4 7))
      ((6 3 4 1)(6 3 4 5 2)(6 7 4 1 2)(6 7 4 5 2)(6 3 4 1 2)(6 7 4 2)(6 3 4 2)
        (6 4 1 2)(2 4 6)(1 3)(5 7)(6 3 1 2)(3 1 2)(3 0 1 2)(3 4 1 2)(3 0 1)
        (3 0 1 2)(3 4 1 2))
    ; padroes V
    ((0 3 6 4 2)(0 3 6 4 1)(0 3 6 7 5 2)(3 6 4 2)(3 6 4)(1 4 7 5)(0 3 6 7 4 2)
      (1 4 8 5 2)(1 4 2)(0 4 8 5 2)(0 4 2)(3 7 5)(3 7 5 2)(3 7 4 2))
    ; padroes J
    ((0 1 2 4 7 6 3)(1 4 7 6 3)(2 5 8 7 4)(0 1 2 1 4 7 6 3))
    ; padroes ortogonais
    ((2 1 4 0 3 7)(2 1 0 3 6)(2 1 4)(1 0 3)(4 3 6)(5 4 7)(2 1 4 7)
      (1 0 3 6))
      ((0 3 6 7 4 5)(0 3 6 7 8)(0 3 6 7)(0 3 4)(1 4 5)(4 7 8)(3 6 7)(1 4 7 8))
      ((6 7 8 5 2)(6 7 4 1)(6 7 4)(7 8 5)(3 4 1)(4 5 2)(7 8 5 2)(6 7 8 5)
        (6 7 8 4 1 2)(6 7 4 1 2)(3 6 7 8 4 1 2)(6 7 4 2))
      ((8 5 2 1 0)(5 2 1 0)(4 1 0)(7 4 3)(5 2 1)(8 5 4)(7 4 1 0)(8 5 2 1)
        (5 1 0)(8 5 1 0)(7 5 2 1 0)(8 5 4 1 0))
    ; padroes zorro
    ((2 1 4 3 6 7 8 5 8)(1 2 1 4 3 6 7 8 5)(0 1 2 1 4 3 6 7 8)(0 1 2 1 3 6 7 8)
      (0 1 2 4 6 7 8)(0 1 2 1 4 3 6 7 8)(0 1 2 4 3 6 7 8)(0 1 2 1 4 3 6 7 5)
      (0 1 2 4 6 7 8)(0 1 3 6 7 8 5)(0 1 4 3 6 7 8)(0 1 3 6 7 5)(0 1 3 7 8))
    ; padroes cruces
    ((1 4 7 3 4 5)(1 4 7 4 3 4 5)(1 4 7 4 3 0 1 2)(0 3 6 3 4 5)
      (1 3 6 7 6 3 6 7 8)(1 4 7 4 1 3 4 5)(1 4 7 4 3 1 2)(4 7 4 1 4 3 4 1 2))
      ((3 4 5 1 4 7)(3 4 5 1 4 7)(3 4 5 4 1 0 3 4 7)(0 3 4 5 4 1 0 3 6)
        (3 4 5 2 5 4 1 4 7))
    ; padroes U
    ((0 3 6 7 8 5 2)(1 4 7 8 5 2)(0 3 6 7 8 5 2 1)(0 3 6 7 4 5 2)(0 3 6 7 5 2)
      (0 3 7 4 1 2)(0 3 4 5 2)(0 3 4 1)(0 3 6 7 4 5 2 1)(0 3 7 8 5 2))
    ; padroes ferradura
    ((6 3 0 1 2 5 8)(7 4 1 2 5 8)(6 3 0 1 2 5)(6 3 0 1 5 8)(3 0 1 4)(3 0 1 4 5)
      (3 0 1 4 5 8)(6 3 0 1 2 5)(3 0 1 2 5)(6 3 0 1 4 5 8)(6 3 0 1 5)(3 0 1 5 8))
    ; padroes C
    ((2 1 0 3 6 7 8)(2 1 4 7 8)(2 1 0 3 7 8)(1 0 3 6 7 8)(2 1 0 3 6 7)(1 0 3 4)
      (1 0 3 4 5)(2 1 0 3 7)(1 0 3 7 8)(2 1 0 3 7 8 5)(2 1 0 3 6 7 8 5)
      (0 3 6 7 8)(0 1 3 6 7))
    ; padroes convexos
    ((0 1 2 5 8 7 6)(0 1 4 7 6)(0 1 2 5 4 7 6)(0 1 2 5 7 6)(0 1 4 3)
      (0 1 4 3 6)(0 1 5 4 7 6)(0 1 2 5 4 7 6 3)(0 1 2 5 4 6)(0 1 4 6)
      (0 1 4 5 4 7 6)(0 1 5 8 7 6)(0 1 2 5 8 7))
    ; padroes circulares
    ((0 1 2 5 8 7 6 3)(0 1 2 5 8 7 6 3 0)(3 0 1 2 5 7 6 3 0)
      (1 2 5 8 7 6 3 0)(1 2 5 8 7 6 3 0 1)(1 2 5 4 7 6 3 0 1)
      (2 5 8 7 6 3 0 1)(2 5 8 7 6 3 0 1 2)
      (5 8 7 6 3 0 1 2)(5 8 7 6 3 0 1 2 5)
      (8 7 6 3 0 1 2 5)(8 7 6 3 0 1 2 5 8)
      (7 6 3 0 1 2 5 8)(7 6 3 0 1 2 5 8 7)
      (6 3 0 1 2 5 8 7)(6 3 0 1 2 5 8 7 6))
  )

```

```

(3 0 1 2 5 8 7 6)(3 0 1 2 5 8 7 6 3)
(3 0 1 2 5 4 7 6 3)(3 0 1 2 5 4 7 6)
(0 1 2 5 4 7 6 3 0)(0 1 2 5 7 6 3 0)
(0 1 4 3 0)(1 2 5 4 1)(4 5 8 7 4)(3 4 7 6 3))
))

(defun identificapadiao(pattern)
  (put 'padiao 'original (reverse pattern))
  (put 'padiao 'busca 'nada)
  (identificapadroses pattern))

(defun identificapadroses(pattern)
  (when (and (> (length pattern) 2)(= (get 'padiao 'busca) 'nada))
    (identificapadiao* pattern)
    (identificapadroses (rest pattern))))

(defun identificapadiao*(pattern)
  (writeidentificacao (identificapadiao**
    (strokespadroses)(userstrokespadroses)(reverse pattern)) " "))
  (if (= (get 'padiao 'busca) 'nada)
    (writeidentificacao (identificapadiao**
      (strokespadroses)(userstrokespadroses) pattern) " reverse")))

(defun writeidentificacao(x y)
  (when x
    (put 'midi 'padiao x)
    (put 'padiao 'busca (stringconcat x y))))

(defun identificapadiao**(strokes userstrokes padiao)
  (when strokes
    (cond
      ((or (member padiao (first strokes))(member padiao (first userstrokes)))
        (identificador (- (length strokes) 1)))
      (t (identificapadiao** (rest strokes) (rest userstrokes) padiao)))))

; esta funcao da nome aos bois

(defun identificador(n)
  (nth n '(
    "circular"
    "convexo"
    "letra_C"
    "ferradura"
    "letra_U"
    "cruz_direita_sobe"
    "cruz_desce_direita"
    "letra_2"
    "letra_L_deitada"
    "moldura_inferior"
    "letra_L"
    "moldura_superior"
    "letra_J"
    "letra_V"
    "diagonal_cima"
    "diagonal_baixo"
    "vertical"
    "horizontal"
  )))

(defun writemenu()

```

```

(writetext 50 60 "I N T E R F A C E   G E S T U A L")
(writetext 220 85 "1996 (c) Jonas Manzolli")
(writetext 50 120 "a- Treinamento Visual")
(writetext 50 145 "b- Treinamento Sonoro")
(writetext 50 170 "c- Programa Interface")
(writetext 50 195 "d- Performance")
(writetext 50 220 "e- Salva Treinamento")
(writetext 50 245 "f- Busca Treinamento")
(writetext 50 270 "g- Limpa Base de Conhecimento")
(writetext 50 295 "h- Termina"))

(defun GESTUAL()
  (cls)
  (whitescreen)
  (resetpointers)
  (defaultBuffers)
  (supervisormode)
  (resetbuffers (get 'midi 'size))
  (usermode)
  (writemenu)
  (menu* 2)
  (refreshBase)
  (blackscreen))

(defun menu*(option)
  (cond
    ((and (> option 96)(< option 105))
     (cond
       ((= option 97)(treinamento_visual)(menu* 2))
       ((= option 98)(treinamento_sonoro)(menu* 2))
       ((= option 99)(programa_interface)(menu* 2))
       ((= option 100)(performance)(menu* 2))
       ((= option 101)(salvar)(menu* 2))
       ((= option 102)(buscar)(menu* 2))
       ((= option 103)(refreshBase)(menu* 2))
       ((= option 104)(cls))))
    (t (menu* (internalcode (readonekey))))))

(defun refreshBase()
  (refreshBase* 0))

(defun refreshBase*(x)
  (when (< x 18)
    (remprop 'userpattern (turn-into-atom (identificador x)))
    (remprop 'midi (turn-into-atom (identificador x)))
    (refreshBase* (+ x 1))))

;-----
; Leitura de arquivos

(defun salvar()
  (salvar* (filename) (maketreinofile))
  (writemenu))

(defun maketreinofile()
  (list
    (userstrokespadroes)
    (midicontrollers)))

```

```

(defun salvar*(name treinamento)
  (when (= (second name) 1)
    (let ((oldstream (wrs (open (first name) 'output))))
      (prin treinamento)
      (wrs)
      (close (first name)))))

(defun buscar()
  (buscar* (filename))
  (writemenu))

(defun buscar*(name)
  (when (= (second name) 1)
    (let ((oldstream (rds (open (first name) 'input))))
      (buscatreinofile (read))
      (rds)
      (close (first name)))))

(defun buscatreinofile(stream)
  (makestreamtreino 'userpattern (first stream))
  (makestreamtreino 'midi (second stream)))

(defun makestreamtreino(index stream)
  (when stream
    (put index
      (turn-into-atom (identificador (- (length stream) 1)))
      (first stream))
    (makestreamtreino index (rest stream))))

(defun filename()
  (file-selection "A:\\*.GST" "SCORE.GST"))

; a funcao busca verifica se ha movimento no mouse
; e renova continuamente as variaveis
; n variavel que da o numero maximo de pontos no stroke
; minx miny maxx maxy que sao os minimos e maximos das coordenadas
; qx qy sao maxx-minx div 3, ou seja o quantum divisor
; coord lista com as ultimas coord x e y do mouse
; mouse lista com os parametros do mouse
; coord contem a seguinte estrutura
; [[x0 x1 x2...] [y0 y1 y2...] minx miny maxx maxy]

(defun programa_interface()
  (cls)
  (writeopcoesprogramas)
  (writeopcoespadrao)
  (programa_interfaces* 2)
  (writemenu))

(defun programa_interfaces*(opcao)
  (writetext 50 200 "Escolha a stroke [a/b/c/d/e/f/g/h/i/j/k/l/m/n/o/p/q/r]")
  (cond
    ((= opcao 115)(cls))
    ((and (> opcao 96)(< opcao 115))
     (programa_interfa (turn-into-atom (identificador (- opcao 97)))))
    (t (programa_interfaces* (internalcode (readonekey)))))

(defun programa_interfa(stroke)
  (writetext 50 200 "
  (writetext 50 200 "Escreva lista de funcoes para")
  ")

```

```

(writetext 250 200 stroke)
(cursoron)
(put 'midi stroke (read))
(cursoroff)
(programa_interfaces* 2 (getmouseinfo)))

(defun writepocooprogramas()
  (writetext 200 60 "Programas para mudanca atraves de strokes")
  (writetext 200 80 "-----")
  (writetext 200 100 "mudaprg[x]   identifica[x]   ciclo[]   inverciclo[]")
  (writetext 200 120 "mudacanal[x] arpejos[]   aleaciclo[] resetbuffers[]")
  (writetext 200 140 "perfomer[x]   superarpejos[] retrociclo[] volume[]")
  (writetext 200 160 "bend[]   modula[]   pan[]   tecle S para sair"))

(defun treinamento_sonoro()
  (acertaTela)
  (put 'midi 'length 15)
  (setup-midi-in)
  (put 'control 'button 0)
  (busca 1 15 (iniciacoord)(getmouseinfo)(getmidiinfo))
  (usermode)
  (writemenu))

(defun getmidiinfo()
  (processaMidi
   (readMIDI (get 'midi 'flag)(get 'midi 'filtro)(get 'midi 'channel))))

(defun processaMidi(midi)
  (when midi
    (list
     (* (rem (second Midi) 3)(rem (first Midi) 3))(/ (first Midi) 13))
     (* (rem (first Midi) 23)(/ (first Midi) 17)))))

(defun processaMidi*(omega midi)
  (when midi
    (list
     (+ 150 (round (* 150 (sin (* omega (first Midi))))))
     (+ 70 (round (* 69 (sin (* omega (second Midi)))))))))

(defun performance()
  (reclaim)
  (cls)
  (resetpointers)
  (writeperformance)
  (setup-midi-in)
  (put 'midi 'length 250)
  (put 'control 'button 0)
  (busca 2 250 (iniciacoord)(getmouseinfo)(getmidiinfo))
  (usermode)
  (writemenu))

(defun writeperformance()
  (writetext 180 20 "P E R F O R M A N C E")
  (writetext 170 35 "-----"))

(defun treinamento_visual()
  (acertaTela)
  (setup-midi-in)
  (put 'midi 'length 500)
  (put 'control 'button 0)

```

```

(busca 0 500 (iniciacoord)(getmouseinfo) 'nil)
(usernode)
(writemenu))

(defun acertaTela()
  (cls)
  (changecursor 0)
  (writeopcoespadrao)
  (showcursor))

(defun writeopcoespadrao()
  (drawline 0 252 639 252)
  (drawline 0 255 639 255)
  (writetext 25 270 "a- circular f- cruz dir sobe k- letra L p- diag baixo")
  (writetext 25 285 "b- convexo g- cruz desc dir l- mold sup q- vertical")
  (writetext 25 300 "c- letra C h- letra Z m- letra J r- horizontal")
  (writetext 25 315 "d- ferradura i- letra L deit n- letra V")
  (writetext 25 330 "e- letra U j- mold inf o- diag cima"))

(defun treinamento*(flag)
  (writetext 50 50 " ")
  (writetext 50 80 " ")
  (cond
    ((/= flag 2)
     (allnotesoff)
     (writetext 20 15 (get 'padrao 'original))
     (writetext 20 30 (get 'padrao 'busca))
     (classificacao 'w))
    (t (executa (get 'midi (turn-into-atom (get 'midi 'padrao)))))))

; =====
; funcao que muda o status de performance em funcao da
; identificacao do stroke

(defun executa(midi)
  (when (/= (get 'padrao 'busca) 'nada)
    (writetext 50 50 (stringconcat "Stroke Action: " (get 'padrao 'busca))))
  (when midi
    (writetext (+ 50 (* 25 (length midi))) 80 (first midi))
    (allnotesoff)
    (funcall (first midi)(second midi))
    (executa (nthcdr 2 midi))))

(defun programas(x)
  (nth x
    '(mudaprg
      mudacanal
      performer
      identifica
      arpejos
      superarpejos
      ciclo
      aleaciclo
      retrociclo
      inverciclo
      resetbuffers
      bend
      modula
      volume
      pan

```

```

(busca 0 500 (iniciacoord)(getmouseinfo) 'nil)
(usermode)
(writemenu))

(defun acertaTela()
  (cls)
  (changecursor 0)
  (writeopcoespadrao)
  (showcursor))

(defun writeopcoespadrao()
  (drawline 0 252 639 252)
  (drawline 0 255 639 255)
  (writetext 25 270 "a- circular f- cruz dir sobe k- letra L p- diag baixo")
  (writetext 25 285 "b- convexo g- cruz desc dir l- mold sup q- vertical")
  (writetext 25 300 "c- letra C h- letra Z m- letra J r- horizontal")
  (writetext 25 315 "d- ferradura i- letra L deit n- letra V")
  (writetext 25 330 "e- letra U j- mold inf o- diag cima"))

(defun treinamento*(flag)
  (writetext 50 50 " ")
  (writetext 50 80 " ")
  (cond
    ((/= flag 2)
     (allnotesoff)
     (writetext 20 15 (get 'padrao 'original))
     (writetext 20 30 (get 'padrao 'busca))
     (classificacao 'w))
    (t (executa (get 'midi (turn-into-atom (get 'midi 'padrao)))))))

; =====
; funcao que muda o status de performance em funcao da
; identificacao do stroke

(defun executa(midi)
  (when (/= (get 'padrao 'busca) 'nada)
    (writetext 50 50 (stringconcat "Stroke Action: " (get 'padrao 'busca))))
  (when midi
    (writetext (+ 50 (* 25 (length midi))) 80 (first midi))
    (allnotesoff)
    (funcall (first midi)(second midi))
    (executa (nthcdr 2 midi))))

(defun programas(x)
  (nth x
    '(mudaprg
      mudacanal
      performer
      identifica
      arpejos
      superarpejos
      ciclo
      aleaciclo
      retrociclo
      inverciclo
      resetbuffers
      bend
      modula
      volume
      pan

```

```

(put 'performance 'retrociclo (list 'retroplaynote (readrhythm)))
(changeswitch 'retrociclo))

; cria ciclo com intervalos invertidos

(defun invertciclo()
  (put 'invertplaynote 'oldtempo (manypulses))
  (put 'performance 'invertciclo (list 'invertplaynote (readrhythm)))
  (changeswitch 'invertciclo))

; pitch bend

(defun bend()
  (changeswitch 'bend))

(defun changeswitch(index)
  (if (= (get 'midi index) 'on)
    (put 'midi index 'off)
    (put 'midi index 'on)))

; modulacao LFO

(defun modula()
  (changeswitch 'modula))

; volume midi

(defun volume()
  (changeswitch 'volume))

; especializacao

(defun pan()
  (changeswitch 'pan))

; =====
; funcoes para apresentacao do programa
; telas etc...

(defun limpatela()
  (writetext 20 15 " ")
  (writetext 20 30 " ")
  (writetext 20 45 " "))

(defun classificacao(opiniao)
  (writetext 20 45 "Voce concorda com a classificacao? [s/n]")
  (cond
    ((= opiniao 's)(limpatela))
    ((= opiniao 'n) (classificacao* 2)(limpatela))
    (t (classificacao (readonekey)))))

(defun classificacao*(opcao)
  (writetext 20 45 "Escolha o seu [a/b/c/d/e/f/g/h/i/j/k/l/m/n/o/p/q/r]")
  (cond
    ((and (> opcao 96)(< opcao 115))
     (padraousuario (turn-into-atom (identificador (- opcao 97)))))
    (t (classificacao* (internalcode (readonekey)))))

(defun padraousuario(index)
  (put 'userpattern index

```

```

    (cons (get 'padrao 'original)(get 'userpattern index))))

(defun iniciacoord()
  '(nil nil 9999 9999 -9999 -9999))

; =====
; funcoes para controlar real-time performance

(defun activefunctions()
  (activefunctions* '(ciclo retrociclo arpejo aleaciclo inverciclo)))

(defun activefunctions*(routines)
  (when routines
    (print routines)
    (if (= (get 'midi (first routines)) 'on)
      (cons (get 'performance (first routines))
            (activefunctions* (rest routines))))))

(defun makeperformanceroutine(funcao)
  (list
    (first funcao)
    (get funcao 'oldtempo)
    (second funcao)))

(defun scheduler()
  (scheduler*
   (activefunctions)))

(defun scheduler*(functions)
  (when functions
    (print functions)
    (puttime (first functions)
              (playrealtime (first (first functions))
                             (timerseconds (second (first functions))
                                             (third (first functions)))))
    (scheduler* (rest functions))))

(defun puttime(funcao tempo)
  (when tempo
    (put funcao 'oldtempo tempo)))

(defun playrealtime(index funcao time delay)
  (getmidiinfo)
  (when (> time delay)
    (apply funcao 'nil)
    (manypulses)))

(defun busca (flag n coord mouse midi)
  (if (> (timerseconds (get 'midi 'timer)) 3.0)
    (put 'control 'button 1))
  (scheduler)
  (cond
    ((= (third mouse) 3)
     (allnotesoff)
     (cls)
     (allnotesoff))
    ((= (third mouse) 2)
     (cls)
     (cond
      ((or (= flag 0)(= flag 1))

```

```

    (allnotesoff)
    (writeopcoespadrao))
    ((= flag 2)(writeperformance)))
    (busca flag n coord (getmouseinfo)(getmidiinfo)))
    ((> (length (first coord)) n)
     (identifica coord)
     (treinamento* flag)
     (clearMidiBuffer)
     (busca flag (get 'midi 'length) (iniciacoord)(getmouseinfo)(getmidiinfo)))
    ((and midi (/= flag 0))
     (put 'midi 'timer (manypulses))
     (busca flag n (makecoord coord (first midi)(second midi))
      (getmouseinfo)(getmidiinfo)))
    ((= (third mouse) 1)
     (put 'control 'button 1)
     (hidecursor)
     (drawline (first mouse)(second mouse)(+ (first mouse) 2)(second mouse))
     (drawline (first mouse)(second mouse)(first mouse)(- (second mouse) 2))
     (playmouse (get 'midi 'channel) (first mouse)(second mouse))
     (showcursor)
     (busca flag n (makecoord coord (first mouse) (second mouse))
      (getmouseinfo)(getmidiinfo)))
    ((and (> (length (first coord)) 2) (/= flag 1)(= (get 'control 'button) 1))
     (put 'control 'button 0)
     (put 'midi 'timer 'nil)
     (identifica coord)
     (treinamento* flag)
     (clearMidiBuffer)
     (busca flag n (iniciacoord)(getmouseinfo)(getmidiinfo)))
    (t (busca flag n coord (getmouseinfo)(getmidiinfo))))

; =====
; funcao para controlar o delay de identificacao das notas
; tocadas pelo keyboard

(defun Timerseconds(old)
  (if old
    (/ (- (manypulses) old) 200.0) 0.0))

; funcao que controla iterativamente a lista de coordenadas

(defun makecoord(coord x y)
  (when coord
    (list
     (cons x (first coord))
     (cons y (second coord))
     (menor x (third coord))
     (menor y (fourth coord))
     (maior x (fifth coord))
     (maior y (sixth coord)))))

; funcao que retorna maior de dois numeros

(defun maior(x y)
  (if (> x y) x y))

; funcao que retorna menor de dois numeros

(defun menor(x y)
  (if (<= x y) x y))

```

```

; funcao que identifica o stroke de acordo com uma lista de padroes

(defun identifica(coord)
  (identificapadrao
    (constroipadrao (first coord)(second coord)
      (quantum (fifth coord)(third coord) 5)
      (quantum (sixth coord)(fourth coord) 5)
      (third coord)(fourth coord))))

; identificapadrao esta acima

; quantum acha o divisor da lista de coordenadas

(defun quantum(x y resolucacao)
  (+ resolucacao (/ (- x y) 3)))

; constroi o padrao normalizando as coordenadas e verificando
; quais a variacao de area
; cx contem os valores das coordenadas x
; cy contem os valores das coordenadas y
; qx e qy sao os quantuns divisores

(defun constroipadrao(cx cy qx qy minx miny)
  (constroipadrao* 0 (rest cx) (rest cy) qx qy
    (area (first cx) (second cy) qx qy minx miny) -9999 minx miny))

(defun constroipadrao*(n cx cy qx qy newarea oldarea minx miny)
  (when (and (< n 9) cx)
    (cond
      ((/= newarea oldarea)
        (cons newarea (constroipadrao* (+ n 1) (rest cx)(rest cy) qx qy
          (area (first cx) (first cy) qx qy minx miny) newarea minx miny)))
      (t (constroipadrao* n (rest cx)(rest cy) qx qy
        (area (first cx)(first cy) qx qy minx miny) newarea minx miny)))))

; funcao que da o valor da area a que pertence a coord x y
; ja normalizada

(defun area(x y qx qy minx miny)
  (+ (/ (- x minx) qx)(* 3 (/ (- y miny) qy))))

```

**ANEXO C- PROGRAMA DO CONCURSO ELDORADO REALIZADO POR ANDRÉ
JUAREZ**



2^a

ELIMINATÓRIA

14 de agosto de 95 - Teatro Cultura Artística - Sala Rubens Sverner - 20 horas

ANDRÉ JUAREZ - VIBRAFONE

Nascido em São Paulo, 51 anos, é bacharel em música (percussão) pela Unesp, tendo estudado também no CLAM (bateria), Berkless College of Music (jazz-vibrafone), Unicamp (Mestrado em Artes) e Festival Internacional de Gramado (percussão e música de câmara).

Destacam-se entre seus diversos orientadores no Brasil:

- Amilson Godoy e Pietro Maranca (piano).
- John Boudler (percussão e vibrafone).
- Almeida Prado, Damiano Cozela, H. J. Koellreutter (Teoria e História da Música).
- John Boudler e Ayrton Pinto (música de câmara).

e nos EUA:

- Richard Thompson, Fred Lipsius, Scott Milles (piano).
- Gary Burton, Jean Baptista Boele, Toshiro Akamatsu e Mathias Haus (vibrafone).

Tem desenvolvido intensa atividade artística como percussionista e vibrafonista.

Atualmente ocupa o cargo de Professor Assistente do Departamento de Música do Instituto de Artes da Unicamp.

PROGRAMA

H. J. Koellreutter / André Juarez	- Panra Rhei*
Jônatas Manzolli / André Juarez	- Névoas e Cristais*
Lívio Tragtenberg	- Rose Bed*
Oswaldo Lacerda	- Cançoneta*
Almeida Prado	- Harmonias*
Nelson Ayres	- Simple Melody*
E. Villani Côrtes	- Ritmata n.º 3*
Raul do Valle / André Juarez	- Imagens

* primeira audição mundial

JÚRI

Presidente: J. J. de Moraes

JÚRI PERMANENTE

Maria José Carrasqueira

Lorenzo Mammi

Marcos Lacerda

CONVIDADO PARA ESTA APRESENTAÇÃO

Paulo Porto Alêgre

PRÓXIMA APRESENTAÇÃO: 16 DE AGOSTO DE 95

A escolha do repertório é de responsabilidade dos candidatos.

Pede-se ao público a gentileza de manter o máximo silêncio durante as apresentações, para que os candidatos não sejam prejudicados.

TROFÉU IDEALIZADO POR MARIA BONOMI**Patrocínio****Realização**