

Sistemas Musicales Interactivos

Sergi Jordà

Master en Artes Digitales
Especialidad Música
Febrero-junio, 2003

Sesión 3: Introducción a PD y a MAX

Indice

- Historia
- Descripción del entorno
- Instalación
- Principios básicos
- Tipos de objetos
- Conexiones y prioridades
- Objetos básicos, objetos de interfaz gráfico y mensajes
- PD y el MIDI
- Documentación
- Ejercicios propuestos

MAX - Descripción

- MAX es un lenguaje gestado a inicios de los 80s para control en t.real de uno de los primeros sintetizadores digitales, fabricado por IRCAM
- Inicialmente es pues un sistema de control, sin posibilidad sonora propia (ni visuales)
- Conforme aumenta la potencia de los ordenadores, MAX va incorporando otras posibilidades
 - Entrada/salida de audio (~1996)
 - Entrada/salida de imagen (~1999)

Historia de MAX

- Varios sistemas previos, desarrollados en IRCAM por Giuseppe di Giugno, para desembocar en MAX: 4A (1976) → 4B → 4C → 4X (1981)
- Estos sistemas, aunque flexibles a nivel de software utilizaban hardware específico: el 4X vale 100.000\$!
- Con 4X, Pierre Boulez realiza Répons (1981) para solistas, cto. instrumental, ordenador 4X (que procesa el sonido instrumental) y dispositivo de espacialización (*fragmento*)
- En 1987, Miller Puckette desarrolla un nuevo lenguaje para controlar la 4X. Funciona en un Macintosh y se bautiza como MAX (en honor de M. Mathews).
- En 1991, Opcode comercializa MAX (sólo MIDI, no audio)
- En 1996, MAX/MSP incorpora ya audio (sin necesidad de hardware ad.)
- Por estas fechas surgen, las nuevas versiones de MAX (PD, desarrollada personalmente por M.Puckette y JMAX desarrollada en el IRCAM)
- Putting Max in Perspective (Peter Desain, Henkjan Honing, Robert Rowe, Brad Garton, with comments by Roger Dannenberg, Dean Jabobs, Cort Lippe and Zack Settel, Stephen Travis Pope, Miller Puckette, and George Lewis, CMJ 17:2, 1993)

Puckette y PD

- Uno años después de crear MAX, y de que Opcode se hiciera cargo de la versión comercial para Macintosh, Puckette inicia el proyecto PD (libre, código abierto, y disponible para Linux y Windows)
- Puckette, M. 1991. "FTS: A Real-time Monitor for Multiprocessor Music Synthesis." Computer Music Journal 15(3): pp. 58-67.
- Puckette, M. 1991. "Combining Event and Signal Processing in the MAX Graphical Programming Environment." Computer Music Journal 15(3): 68-77.
- Puckette, M. 1991. "Something Digital." (invited paper, Colloquium in Honor of John Pierce's 80th Birthday, Nov. 1990, Evanston, Ill) Computer Music Journal 15(4): pp. 65-69.
- Puckette, M. 1996. "Pure Data: another integrated computer music environment." Proceedings, Second Intercollege Computer Music Concerts, Tachikawa, Japan, pp. 37-41.
- Puckette, M. 1996. "Pure Data." Proceedings, International Computer Music Conference. San Francisco: International Computer Music Association, pp. 269-272

La familia MAX

- **MAX** (cycling74) para Macintosh (comercial)
 - Audio con MSP (comercial)
 - Imagen con NATO o Jitter (ambos comerciales)
- **JMAX** (IRCAM) para Linux (freeware & open source)
 - Incorpora audio
- **PD** (M. Puckette) para Linux y Windows (freeware & open source)
 - Incorpora audio
 - Imagen con GEM y Framestein (ambos libres y open source)

Interactividad – MAX/PD - Posibilidades generales

- De forma general, un ordenador puede responder a:
 - Datos de Control (teclados, ratones, joysticks, sensores, red...)
 - Sonido (procesarlo o escucharlo-analizarlo-interpretarlo)
 - Imagen (procesar ficheros, entrada cámaras...)
 - Cualquier combinación (e.g. audio + videocámara + sensores...)
 -
 - El ordenador puede responder con:
 - Sonido nuevo (MIDI o audio: composición algorítmica, síntesis a t. real, secuencias pregrabadas, alteradas...)
 - Sonido procesado (a partir del sonido de entrada)
 - Imágenes (generadas, disparadas, procesadas.../ 2D, 3D, vídeo...)
 - Control de dispositivos (proyectors, luces, robots...)
 -
- Un lenguaje como MAX (ó PD...) (con algunos plugins o librerías adicionales)
- Admite entradas de tipo:
 - Audio
 - MIDI
 - Mouse + computer keyboard + joystick...
 - Webcam (u otra entrada de vídeo)
 - Puertos serie, paralelo...
 - TCP/IP
 -
 - Admite salidas de tipo:
 - Audio
 - MIDI
 - Puertos serie, paralelo...(control de dispositivos)
 - TCP/IP
 - Imagen:
 - Disparo y/o procesado de secuencias vídeo
 - Disparo y/o procesado de imágenes fijas
 - Generación animaciones sintéticas 2D, 3D
 -
- Así pues un entorno como MAX es a la vez
- EL "supermapeador"
 - Un sintetizador/procesador de audio
 - Un sintetizador/procesador de vídeo

Descripción del entorno

TO DO

PD en Internet

Instalación

- Bajar y descomprimir pd en cualquier directorio
- Esto instalará el ejecutable (pd.exe), librerías básicas, objetos adicionales, ejemplos, documentación, ayudas así como todo el código fuente (por si deseásemos modificar o añadir objetos en C)

PD FLAGS

- | | | |
|-------------------|--|--|
| • r | specify sample rate | |
| • inchannels | number of audio input channels (08) | |
| • outchannels | number of audio output channels (08) | |
| • audiobuf | specify size of audio buffer in msec | |
| • sleepgrain | specify number of milliseconds to sleep when idle | |
| • nodac | suppress audio output | |
| • noadc | suppress audio input | |
| • nosound | suppress audio input and output | |
| • nomidiout | suppress MIDI output | |
| • nomidiin | suppress MIDI input | |
| • nomidi | suppress MIDI input and output | |
| • path | add to file search path | |
| • open | open file(s) on startup | |
| • lib | load object library(s) | |
| • font | specify default font size in points | |
| • verbose | extra printout on startup and when searching for files | |
| • d | specify debug level | |
| • noloadbang | suppress all loadbangs | |
| • nogui | suppress starting the GUI | |
| • guicmd "cmd..." | substitute another GUI program (e.g., rsh) | |
| • send "msg..." | send a message | |
- with additional options for Windows:
- | | | |
|---------------|------------------------------------|--|
| • listdev | list audio and MIDI devices | |
| • soundindev | specify audio input device number | |
| • soundoutdev | specify audio output device number | |
| • midiindev | specify MIDI input device number | |
| • midioutdev | specify MIDI output device number | |
- Para ejecutar PD es conveniente crear un fichero .BAT (e.g. PD.bat)
 - En este fichero, además de llamar al ejecutable PD.exe se incluyen varias opciones de entorno (flags) (en amarillo algunas de las más utilizadas)
 - *path* nos permite indicar los directorios en los que PD buscará patches personalizados
 - *lib* se utiliza para cargar librería externas (e.g. GEM...)
 - Las opciones inferiores permiten seleccionar de entre los puertos MIDI y de audio disponibles en la máquina
 - A continuación un ejemplo de PD.bat

```
bin\pd.exe -lib extra/gem -listdev -midiindev 1 -soundindev 1 -soundoutdev 1 -path c:\aplic\pd\mypatches
```

Principios básicos

- Entradas y salidas de los objetos...
- Las entradas de la izquierda son las que disparan salidas
- Las salidas se emiten ordenadas, de derecha a izquierda
- Para crear patches propios con entradas y salidas, se utilizan los objetos inlet y outlet
- Para que PD encuentre los objetos que hemos creado, éstos deberán:
 - Estar en un directorio indicado en el flag `-path` (forma más flexible) ó
 - Estar en el mismo directorio que el patch que los utiliza (forma más sencilla y menos flexible). Para que un patch encuentre estos objetos, este patch contenedor deberá estar ya salvado (en ese mismo directorio)
- Estudiar los ejemplos siguientes que tratan estos principios.

Objetos PD

- La información que sigue está extraída del manual oficial (*doc\1.manual\index.htm*)
- Se incluyen para consulta rápida
- TO DO...

Control básico

- | | |
|----------------|---|
| • bang | output a bang message |
| • float | store and recall a number |
| • symbol | store and recall a symbol |
| • int | store and recall an integer |
| • send | send a message to a named object |
| • receive | catch "sent" messages |
| • select | test for matching numbers or symbols |
| • route | route messages according to first element |
| • pack | make compound messages |
| • unpack | get elements of compound messages |
| • trigger | sequence and convert messages |
| • spigot | interruptible message connection |
| • moses | part a numeric stream |
| • until | looping mechanism |
| • print | print out messages |
| • makefilename | format a symbol with a variable field |
| • change | remove repeated numbers from a stream |
| • swap | swap two numbers |
| • value | shared numeric value |

Tiempos

- | | |
|------------|---|
| • delay | send a message after a time delay |
| • metro | send a message periodically |
| • line | send a series of linearly stepped numbers |
| • timer | measure time intervals |
| • cputime | measure CPU time |
| • realtime | measure real time |
| • pipe | dynamically growable delay line for numbers |

TO DO...

PD vs. MAX

- El funcionamiento de ambos programas es muy similar
- Tutorial oficial de MAX (de cycling74) (PDF)
- TO DO...

Ejercicios propuestos

- Estudio de la documentación incluida en *doc\1.manual\index.htm*, hasta el punto 2.4 (excl.)
- Estudio detallado de los ejemplos disponibles desde el menú *Help|PD Documentation|Control Examples*, hasta el ejemplo *04.messages.pd* (incl.)
- Realizar un patch PD que funcione como contador, de forma que cada vez que el usuario clique sobre un bang, muestre un valor numérico incrementado en uno (para ello se debe usar el objeto int o el objeto float y el objeto + que se usará para ir sumando 1 cada vez)
- Estudiar la documentación interactiva del objeto metro y realizar un cronómetro digital que cuente los segundos
- La solución a los dos ejercicios propuestos está en el ejemplo *05.counter.pd*.
- Estudiar los ejemplos siguientes, hasta el *11.review.pd* (incl.).
- Estudiar la documentación interactiva del objeto random y realizar un patch que genere números aleatorios entre min y max (donde min y max son entradas interactivamente por el usuario)
- Estudiar el objeto makenote que permite generar notas, a partir de 3 valores (altura, velocidad y duración en ms). Cada vez que recibe una nota (con altura, velocidad y duración) este objeto se encarga de mandar el mensaje de NOTE OFF correspondiente, transcurrida la duración indicada. Para que los resultados sean audibles, la salida de makenote debe conectarse a un objeto noteout
- Estudio de los ejemplos incluidos en el tutorial de Max (*Max4TutorialsAndTopics.pdf*), hasta el ejemplo 11. Estos ejemplos no deberán verse antes de los propuestos más arriba, ya que Max presenta algunas diferencias con PD, que conviene conocer previamente.

Ejercicios

- Realizar un patch que convierta valores numéricos de B.P.M (beats/minuto) a ms
 - Este objeto tendrá una entrada/inlet (los BPMs) y una salida/outlet (los milisegundos correspondientes)
 - Por ejemplo, si BPM fuera 60, la duración deberá ser 1000 (ya que 60 significa 60 beats/minuto, cada beat durará 1 segundo o 1000 ms)
- A partir del objeto metro (que manda bangs a intervalos regulares), y del objeto makenote, se puede construir un metrónomo audible
- Utilizar el objeto random (y algunas ideas algorítmicas sencillas) para experimentar con generadores de notas aleatorios, utilizando este metrónomo...
- El programa se puede ir complicando de varias formas:
 - Alturas aleatorias pero duraciones (pulsación) constantes (utilizando nuestro metrónomo)
 - Duraciones variables (por ejemplo múltiplos del metrónomo)
 - Cambios de timbres (con pgmout), polifonía, varios canales MIDI, etc...
 - Control interactivo de las tesituras aleatorias (e.g. con sliders controlados por el usuario)
- Estos ejercicios NO son triviales. Algunas cuestiones pueden asimismo requerir mayores conocimientos musicales que otras, por lo que se aconseja el trabajo en grupo, discusión de ideas, etc... así como la integración, combinación de patches diferentes, producidos por grupos diferentes...
- Se incluye un patch a modo de pista

Armonizador : Ejercicio propuesto

- Hemos visto algunos pequeños ejemplos que “preparan” o trucan un teclado MIDI
- Una propuesta más compleja es la realización de un patch que funcione como un armonizador (de momento para la tonalidad de Do Mayor)
(este patch es bastante más complejo, por lo que no se recomienda abordarlo hasta haber realizado los ejercicios indicados previamente)
- Cuando el usuario introduce por teclado (MIDI) una nota de la escala de Do Mayor (i.e. una nota “blanca”) el programa le añade un acorde triada coherente con la nota introducida. A continuación se dan indicaciones musicales para los

Armonizador : Indicaciones Musicales

- Las notas de la escala de Do mayor, son: Do,Re,Mi,Fa,Sol,La,Si
- Las notas MIDI Do de cualquier octava son múltiplos de 12 (0,12,24,36,48,60...)
- Por consiguiente, los valores (semitonos) que hay que sumar a cualquier Do (i.e. un múltiplo de 12) para obtener otra de las notas de la escala, son:

Do	0
Re	2
Mi	4
Fa	5
Sol	7
La	9
Si	11
- Por ejemplo, la nota 45 vale $36+9$, por lo tanto es un La
- A continuación se indican los acordes triadas que corresponden a cada una de las notas de la escala

Notas que tienen acordes (triadas)	mayores:	Do,Fa,Sol	(0,5,7)
Notas	menores:	Re,Mi,La	(2,4,9)
Notas	disminuidos:	Si	(11)
- Que son los acordes mayores, menores o disminuidos?
 - En las triadas mayores hay que sumar un intervalo de 4 semitonos (tercera mayor) y uno de 7 (quinta)
 - En las triadas menores hay que sumar 3 (tercera menor) y 7 (quinta)
 - En las triadas disminuidas hay que sumar 3 (tercera menor) y 6 (quinta disminuida)

Armonizador : Pistas

- Para detectar el nombre (i.e. Do, Re,...) de una nota MIDI de una octava cualquiera, se puede usar el operador %, que da el resto de la división entera.
Ejemplo: $62\%12$ vale 2
- Si miramos el resto de dividir cualquier nota MIDI [0-127] por 12, obtendremos valores entre 0 (Do) y 11 (Si), que nos indicarán el “nombre” de la nota en cuestión
- A continuación podemos utilizar un objeto sel con varios argumentos (0, 2,4,5,7,9) (mirar la documentación de sel). Esto nos va a permitir tratar por separado cada caso. Es decir cuando entremos un Mi (resto 4), saldrá por la puerta correspondiente al 4...
- A partir de allí aplicaremos la casuística descrita en la imagen anterior...
- Una vez terminado, este armonizador se podría colocar en algún punto del generador de notas anterior, para que las notas correspondientes a la escala de Do Mayor, sonaran en acordes. Las notas que no sean de la escala, pueden (a) mutearse o simplemente (b) no armonizarse
- Si os parece demasiado complicado, no os desaniméis... ¡Lo es un poco! (al menos para empezar...)