CSound: software libre de síntesis sonora

Síntesis aditiva, substractiva y FM con CSound.

En la primera parte de esta práctica abordamos la técnica de la síntesis aditiva, creando nuestros sonidos por medio de la adición de componentes puras.

La síntesis substractiva basa la creación de nuevos sonidos en el filtrado de una fuente rica en componentes puras, dejando sólo las que interesen.

Para realizar este proceso necesitaremos una fuente que contenga gran cantidad de componentes. Muchos tipos de onda son apropiadas para este propósito, como son las ondas cuadradas, triangulares o dientes de sierra. Nosotros usaremos, sin embargo, una fuente de ruido blanco.

El opcode **rand** nos permitirá contar con una fuente de números aleatorios distribuidos uniformemente en el rango -iamplitud..+iamplitud
ej.

```
anoise rand iamplitud
```

El filtrado digital puede realizarse de múltiples maneras y el lenguaje CSound ofrece varias de ellas. En concreto usaremos los filtros Butterworth implementados.
Su uso es muy sencillo, sólo tendremos que especificar las frecuencias donde trabajan y el ancho de banda en el caso del filtro pasa-banda.
Los opcodes son **butterbp** (pasa-banda), **butterlp** (pasa-bajo) y **butterhp** (pasa-alto).

ej.

```
; filtra la fuente anoise, dejando pasar un ancho de banda kbandwidth
; en torno a la frecuencia kfreq
afiltered butterbp anoise, kfreq , kbandwidth
```

Se observa que los parámetros kfreq y kbandwidth son de tipo k, esto es se pueden actualizar a k-rate. Esto significa que podemos cambiarlos durante la duración de una nota. Necesitaremos el opcode **line.**

Supongamos este fragmento de código CSound:

```
idur = p3
iamplitud = p4
ifreq1 = p5
ifreq2 = p6

kfreq line ifreq1, idur, ifreq2 ; pasa linealmente de ifreq1 a ifreq2 durante idur
anoise rand p4
afiltered butterbp anoise, 50
```

Filtrará ruido blanco en un ancho de banda de 50 Hz desde ifreq1 a ifreq2 en un tiempo idur.

Listado: "ejemplo3substractiva.csd"

```
<CsoundSynthesizer>
<CsOptions>
</CsOptions>
<CsInstruments>
sr = 44100 ;sample rate
ksmps = 1 ;control rate
nchnls = 2 ; salida de dos canales, estéreo!

;definición del instrumento 1
instr  1
idur = p3
iamplitude = p4
ifreqInicial = p5
ifreqFinal = p6
ibandwidth = p7
;segmento lineal
kfreq line ifreqInicial, idur, ifreqFinal
;ruido blanco
anoise rand iamplitude
;filtramos
afiltered butterbp anoise, kfreq, ibandwidth

;de 0 a 1 en idur segundos para la posición estéreo
kpan line 0, idur, 1

;envolvente simple de 0 a 1 y otra vez a cero en idur segundos
;mira el opcode linseg en el manual!
kenv linseg 0, idur / 2, 1, idur / 2, 0

;aplicamos una función de 1 a 0 en el canal izquierdo
; y de 0 a 1 en el canal derecho
aleft = afiltered * (1 - kpan)
aright = afiltered * kpan
        ;salidas izquierda y derecha
        ;fíjate que multiplicamos por la envolvente
        outs kenv * aleft, kenv * aright
endin
</CsInstruments>
;y ahora la partitura:
<CsScore>
;p1     p2      p3      p4      p5      p6      p7
;ins    start   idur    iampli  ifreqin ifreqfi ibandwidth
i1      0       10      20000       5000        100     150
i1      8       10      20000       20      10000       250
e
</CsScore>
</CsoundSynthesizer>
```

Tarea 1

1) Como siempre pegaremos el código en nuestro editor favorito y guardaremos el archivo con la extensión .csd

Compilar con CSound5 GUI y probar a cambiar con otras notas con otros parámetros p3, p4, p5, p6, p7. ¿Qué pasa si el ancho de banda (p7) se acerca a 20000 Hz?

2) Con cambios de este tipo crear un sonido nuevo distinto al del ejemplo. Grabar el fichero .csd. Grabar el fichero WAV resultante.

3) Duplicar el archivo .csd y darle el nombre de "ejemplo3substractivaplus.csd" (por ej.). Modificar el instrumento para que:

{ el sonido vaya de un canal a otro y vuelva (usar **linseg**)

ó

cada nota tenga un comienzo y final en el campo estéreo
(valor de comienzo de 0 a 1 y valor de final de 0 a 1 }

y

el ancho de banda sea dinámico (cambie)
durante la duración de la nota (**line**)

EXTRA: En vez de **line** usar un opcode que proporcione una curva exponencial de un valor a otro. Pista: ¡está en el manual!

[OPCIONAL

Quiero producir un eco dentro del instrumento ¿cómo lo hago?

Respuesta: Los opcodes **delay** y **delay1**

Muchas veces es necesario mezclar una señal de audio con si misma tras un cierto retraso (delay). Esto se puede hacer con el opcode **delay** cuyo uso es:

adelayed **delay** asound, idelaytime

¿Qué significa la "i" delante de idelaytime? indica que el tiempo de retraso debe ser fijado al principio de la nota y no se puede cambiar durante la ejecución.

El opcode **delay1** retrasa la señal de audio una sola muestra (o sea 1/**sr** = 1 / sample rate segundos):

adelayed1 **delay1** asound

Con estos opcodes y un poco de trabajo se pueden implementar los filtros de media móvil.

]

¿CÓMO LEO AUDIO DE UN ARCHIVO EXTERNO?

Muy fácil, hay un opcode **diskin** que lee desde cualquier fichero.
Uso:

asound **diskin** path, kvelocidad

Ejemplos:

```
;reproduce el fichero "loop.wav" a velocidad normal
;"loop.wav" debería estar en el mismo directorio que el fichero .csd
asound diskin "loop.wav", 1

;reproduce el fichero "loop.wav" a doble velocidad
asound diskin "loop.wav", 1

;reproduce el fichero "loop.wav" al revés a la mitad de velocidad
asound diskin "loop.wav", -0,5
```

Un ejemplo completo:

```
<CsoundSynthesizer>
<CsOptions>
</CsOptions>
<CsInstruments>
sr = 44100
ksmps = 1
nchnls = 1
;Instrument #1 - play an audio file.
instr 1
kvelocidad = p4
    ;Lee "yoursound.wav" a kvelocidad y lo asigna a asound
    asound diskin "yoursound.wav", kvelocidad
    ;envía asound a la salida
    out asound
;fin de definición
endin
</CsInstruments>
<CsScore>
;Una nota durante 4 segundos a velocidad normal
;luego una nota durante 5 segundos al reves.
;p1     p2      p3      p4
i1      0       4       1
i1      4       5       -1
e
</CsScore>
</CsoundSynthesizer>
```

Tarea 2

1) Crea un instrumento CSound que use el opcode **diskin** y que lea un fichero de audio(en formato WAVE, por ejemplo) con alto contenido espectral (puede ser ruido rosa, una grabación orquestal o de trash-metal, olas del mar, viento o sonido de tormenta, etc.). El fichero debe contener al menos 20 segundos de sonido.

El sonido leido (por ejemplo en la variable asource) debe pasar por un filtro pasabanda tipo Butterworth como el usado en la tarea 1.

Añade variables tipo k para que el resultado sea variable con el tiempo.

Los parámetros que puedes cambiar podrían ser:
-velocidad de lectura del fichero WAV (usa **line** para modificar kvelocidad)
-frecuencia central del filtro
-ancho de banda del filtro
-amplitud (usando una envolvente general)
-posición estereofónica [el instrumento debe tener nchnls = 2]

Si quieres tu instrumento puede tener dos o más filtros pasa-banda.

¿Qué pasa si usas muchos filtros (10 o más) con anchos de banda muy estrechos? En este caso diríamos que el sonido original está pasando por un banco de filtros en paralelo.

Hay múltiples opciones para crear un sonido interesante (¿y si uso un filtro pasa-bajo y otro pasa-alto en vez de uno solo pasa-banda?).

Una idea: la nota o notas pueden ser muy largas (por ejemplo 20 segundos o mucho más) para que se oiga bien la evolución del sonido.

3/11/2007