PROJECT1-SPARTAN

Integrantes

Carlos Andrés Mata Calderón

Manuel Bojorge Araya

Descripción de la aplicación

La aplicación es un cliente de procesamiento de señales de audio utilizando la API de JACK y la biblioteca Boost para las opciones de línea de comandos y estructuras de datos tipo circular buffer. S

Modos de operación dentro Process

Passthrough: El audio se pasa tal cual, sin cambios.

Para dicha función se utiliza es *process_passthrough()*, esta función toma de valores de entrada la cantidad *nframes*, el puntero del *input* y *output*. La función es un simple for que copia los valores del *input* al *output*

Volume Change: Permite cambiar el volumen del audio

La función se llama *process_volume_change()*, toma valores de entrada la cantidad *nframes*, el puntero del *input* y *output*. La función es un for que utiliza el atributo *volumen* para multiplicar la entrada y ponerlo en la salida.

Calculate Energy or Power: Calcula la energía o potencia tomando en cuenta una cierta cantidad de tiempo.

La función se llama *calculate_energy_and_power()*. Toma la cantidad de muestras *nframes* y la señal que *signal* que se quiere calcular la energía. Este realiza el cálculo de energía y potencia y utiliza un ventana con tamaño *window_energy* donde se guardan los valores calculados junto con un acumulador llamado *acumulate_energy* o *acumulate_power* que es el valor actual de energía o potencia. Donde la ventana se utiliza para almacenar los valores más antiguos y eliminarlos del acumulador.

Calculate Period: Calcula el periodo de un sonido después de cumplir un cierto valor energía.

La función que almacena los datos de las muestras para después calcular el periodo *get_data period()*. Esta recibe la cantidad de muestras *nframes* y la señal que *signal* que se guardará. La información sólo se guarda después de superar un valor de energía llamado *period_minlevel*. La información se guarda en *ring_buffer* que tiene una capacidad máxima de *period_ringsize*. La información dentro del ring *buffer* se borra después de una cantidad cantidad muestras que no superan la energía necesaria.

Repeater: Repite la señal de audio según el periodo.

Para esta función se utiliza la función *process_repeater(),* esta función recibe como parámetros de entrada *nframes*, y el puntero *out.* Esta función toma la frecuencia de la señal que se produce en ese momento y reproduce una señal senoidal con el mismo periodo y energía de la señal recibida

Tuner: Ayuda a afinar un instrumento musical mostrando la frecuencia y la nota más cercana. Esta implementación se hace con la función *process_tuner()*, primero verifica si hay sonido (frecuencia positiva) y de ser así compara el valor de la frecuencia obtenida con cada una de las frecuencias de cada nota para obtener la diferencia mínima, la nota más cercana y su respectiva frecuencia.

Autotune: Ajusta automáticamente la frecuencia de la señal de audio a la más cercana en una escala predeterminada. Esta funcionalidad se realiza con *process_autotune()*, básicamente realiza la misma funcionalidad de *process_repeater()* pero utilizando la frecuencia de la nota más cercana.

Modos de operación dentro Main

El switch case:

En el ciclo principal (while) se encuentra el bloque switch-case, en cada case se evalúa la tecla presionada por el usuario para activar los modos disponibles en la aplicación, algunos modos desactivan otros como es el caso del modo energía que desactiva el cálculo del periodo y viceversa, pero el modo de volumen se puede ejecutar en paralelo con el modo repetidor y el auto-tune.

Funciones que siempre se llaman en el main:

- *calculate_periode()*: Esta función primero verifica si esta en los siguientes modos Period, Repeater, Tuner, Autotune, ya que son los modos que ocupan que se calcule el periodo. Luego verifica si ya hay datos suficientes dentro del *ring buffer* para calcular la convolución. Para dicho cálculo se utiliza una ventana de tamaño *period_nwindow*, donde el nombre de la variable de la ventana es *correlation_signal*. El cálculo inicia a la mitad de *ring_buffer* y finaliza en la mitad + *period_nwindow*. Se aprovecha el mismo for para encontrar 2 picos máximos dentro del cálculo, si el valores de la frecuencia de dichos picos se encuentra en el rango de *freq_min* y *freq_max* . Se calcula el periodo de la señal dentro de *ring_buffer* .
- *process_tuner*(): Esta función necesita comprobar si se está calculando el periodo, es decir que hay sonido, de ser así no ejecuta los cálculos para encontrar la frecuencia requerida; cuando esto se cumple indica al usuario si debe subir o bajar el tono para alcanzar la nota más cercana o si la diferencia entre la frecuencia emitida y la próxima nota cumple con un rango de precisión, indica que está afinado. Está fuera del bloque switch-case pues su funcionamiento es independiente de los otros modos.