UNIVERSIDAD
DE GRANADA

Práctica 5: Experimentación con el sistema de sonido

Autor: Juan José Martínez Águila

GRADO EN INGENIERÍA INFORMÁTICA (SISTEMAS DE LA INFORMACIÓN)

—

Curso 2022 - 2023

Índice

1. Leer dos ficheros

Para esta práctica utilizaremos la librería de python pygame para poder reproducir los ficheros, que es una librería para programar videojuegos pero también nos sirve para trabajar con sonidos. Para realizar esta parte de l práctica será muy sencillo, tendremos que leer los archivos y luego utilizando la librería reproducirlos, sería con este código:

```python
import pygame

pygame.init()

archivo1 = "/home/juanjo/4aniocarrera/SEGUNDO_CUATRI/PDIH NO GIT/PDIH/P5/nombre.wav"
archivo2 = "/home/juanjo/4aniocarrera/SEGUNDO_CUATRI/PDIH NO GIT/PDIH/P5/apellidos.wav"

pygame.mixer.init()

nombre = pygame.mixer.Sound(archivo1)
apellidos = pygame.mixer.Sound(archivo2)


nombre.play()
pygame.time.wait(int(nombre.get_length() * 1000))

apellidos.play()
pygame.time.wait(int(apellidos.get_length() * 1000))


pygame.mixer.quit()
pygame.quit()
```

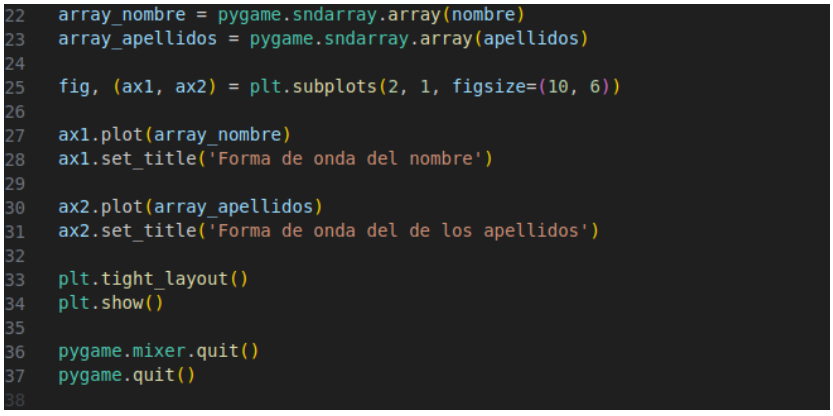
Figura 1: Código para leer y reproducir dos archivos.wav

2. Dibujar la forma de onda

Par ello vamos a necesitar dos librerias más:

- **Numpy**: Para obtener los valores númericos de los sonidos y poder representarlos gráficamente.
- **matplotlib**: Para poder realizar laas gráficas

El código que deberíamos hacer sería convertir los sonidos obtenidos anteriormente en formato numpy y con esos datos representar las gráficas con matplotlib.



```python
array_nombre = pygame.sndarray.array(nombre)
array_apellidos = pygame.sndarray.array(apellidos)

fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(10, 6))

ax1.plot(array_nombre)
ax1.set_title('Forma de onda del nombre')

ax2.plot(array_apellidos)
ax2.set_title('Forma de onda del de los apellidos')

plt.tight_layout()
plt.show()

pygame.mixer.quit()
pygame.quit()
```

Figura 2: Código para generar las gráficas

Esta sería la imagen de las gráficas:

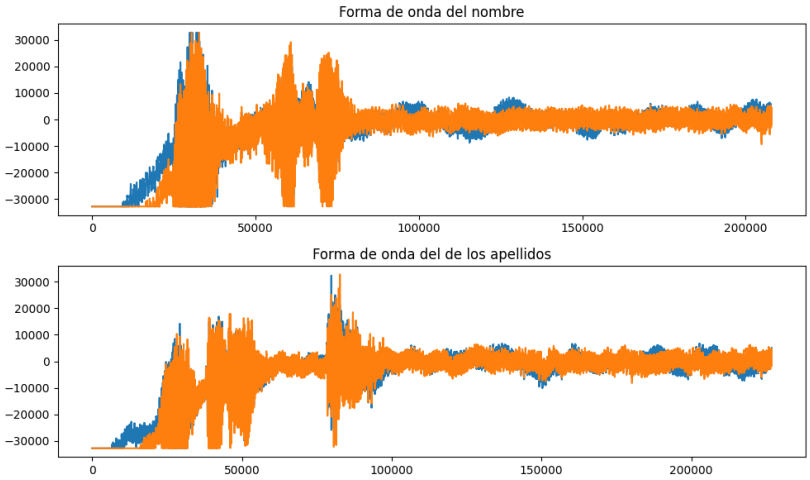


Figura 3: Gráfico de ondas

3. Obtener la información de las cabeceras de ambos sonidos

Para obtener la cabecera pygame no nos incorporá funciones como tal, así que utilizaremos la librería wave que también sirve para trabajar con sonidos.

```
#OBTENER CABECERAS DE LOS SONIDOS
with wave.open(archivo1, 'rb') as sonido:
    cabecera1 = sonido.getparams()

with wave.open(archivo2, 'rb') as sonido:
    cabecera2 = sonido.getparams()

print("Información del sonido del nombre: \n {}".format(cabecera1))
print("Información del sonido de los apellidos: \n {}".format(cabecera2))
```

Figura 4: Código para obtener las cabeceras

```
juanjo@juanjo-hpomen:~/4aniocarrera/SEGUNDO_CUATRI/PDIH NO GIT/PDIH/P5$ python3 sonido.py
pygame 2.4.0 (SDL 2.26.4, Python 3.8.10)
Hello from the pygame community. https://www.pygame.org/contribute.html
Información del sonido del nombre:
_wave_params(nchannels=2, sampwidth=2, framerate=44100, nframes=207979, comptype='NONE', compname='not compressed')
Información del sonido de los apellidos:
_wave_params(nchannels=2, sampwidth=2, framerate=44100, nframes=226843, comptype='NONE', compname='not compressed')
```

Figura 5: Información de las cabeceras

4. Unir ambos sonidos

Para unir ambos sonidos seguiremos utilizando la librería **wave** y nos quedará el siguiente código.

```
infiles = [archivo1, archivo2]
outfile = "sounds.wav"

data= []
for infile in infiles:
    w = wave.open(infile, 'rb')
    data.append( [w.getparams(), w.readframes(w.getnframes())] )
    w.close()

output = wave.open(outfile, 'wb')
output.setparams(data[0][0])
output.writeframes(data[0][1])
output.writeframes(data[1][1])
output.close()
```

Figura 6: Código para obtener la combinación de los sonidos

5. Dibujar la forma de onda de la señal resultante

Seguiremos la misma idea que para el ejercicio 2, transformar los datos a numpy y con eso dibujar las gráficas.

```python
#OBTENER FORMA DE ONDA
archivo = "/home/juanjo/4aniocarrera/SEGUNDO_CUATRI/PDIH NO GIT/PDIH/P5/sounds.wav"

with wave.open(archivo, 'rb') as wav:
    canales = wav.getnchannels()
    profundidad_bits = wav.getsampwidth()
    frecuencia_muestreo = wav.getframerate()
    frames = wav.getnframes()

    frames_audio = wav.readframes(frames)

audio_array = np.frombuffer(frames_audio, dtype=np.int16)

tiempo = np.linspace(0, len(audio_array) / frecuencia_muestreo, num=len(audio_array))

fig, ax = plt.subplots()
ax.plot(tiempo, audio_array)

ax.set(xlabel='Tiempo (s)', ylabel='Amplitud',
       title='Forma de la combinacion de ambos sonidos')
ax.grid()

plt.show()
```

Figura 7: Código de las gráficas

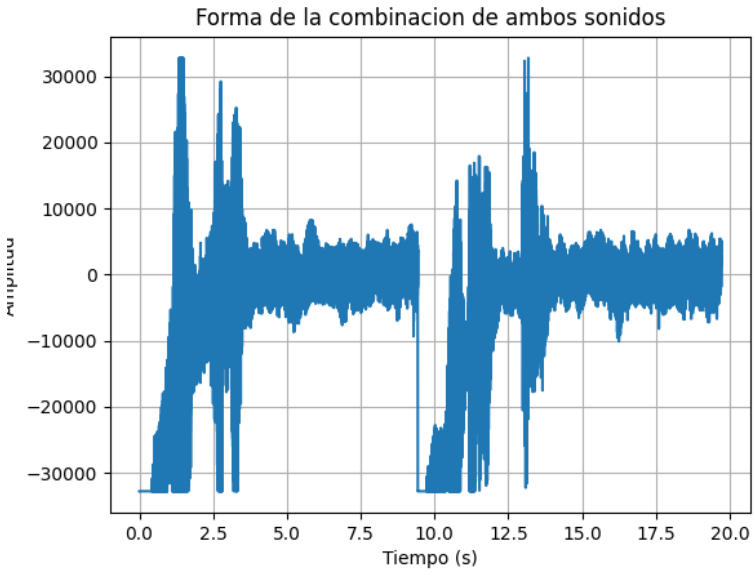
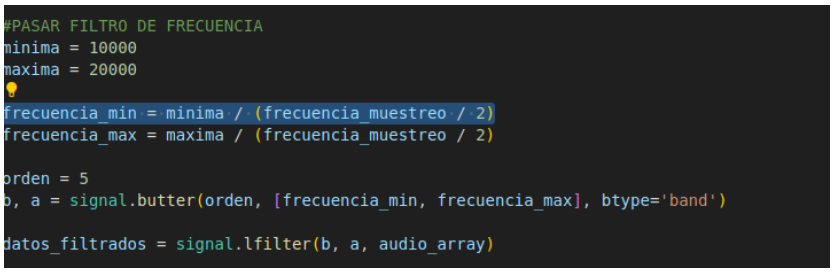


Figura 8: Forma de onda

6. Pasar filtro de frecuencia

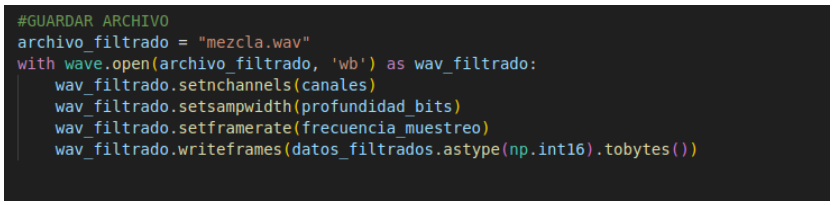
Para ello hay que normalizar la frecuencia del sonido combinado con las respectivas frecuencias mínimas y máximas y crear un nuevo wav con esa normalización.



```
#PASAR FILTRO DE FRECUENCIA
minima = 10000
maxima = 20000

frecuencia_min = minima / (frecuencia_muestreo / 2)
frecuencia_max = maxima / (frecuencia_muestreo / 2)

orden = 5
b, a = signal.butter(orden, [frecuencia_min, frecuencia_max], btype='band')

datos_filtrados = signal.lfilter(b, a, audio_array)
```

Figura 9: Aplicación de filtro de frecuencia

7. Almacenar señal obtenida



```
#GUARDAR ARCHIVO
archivo_filtrado = "mezcla.wav"
with wave.open(archivo_filtrado, 'wb') as wav_filtrado:
    wav_filtrado.setnchannels(canales)
    wav_filtrado.setsampwidth(profundidad_bits)
    wav_filtrado.setframerate(frecuencia_muestreo)
    wav_filtrado.writeframes(datos_filtrados.astype(np.int16).tobytes())
```

Figura 10: Código para almacenar la señal obtenida

8. Invertir audio

Para ello iba a utilizar la librería audioop que te proporciona una función llamada reverse pero no he sabido utilizarla.

9. Bibliografía

- https://parzibyte.me/blog/2020/10/31/reproducir-sonido-python-pygame/
- https://docs.python.org/es/3/library/wave.html
- https://es.stackoverflow.com/questions/447925/unir-audio-wav-en-python
- https://docs.scipy.org/doc/scipy/tutorial/signal.html
- https://docs.scipy.org/doc/scipy/reference/signal.html
- https://realpython.com/playing-and-recording-sound-python/
- https://programacionpython80889555.wordpress.com/2023/02/02/graficando-senales-de-audio-con-wave-y-matplotlib/
- https://docs.python.org/es/3.10/library/audioop.html