TEC | Tecnológico de Costa Rica

Tecnológico de Costa Rica

Licenciatura en Ingeniería en Electrónica

Curso:

Elementos de computación

Profesor:

Ing. Juan Manuel Sánchez Corrales

Investigación proyecto 01

Grupo:

20

Estudiante:

Brayan Barquero Madrigal

Introducción

En el proyecto asignado, se utilizan ciertas bibliotecas y funciones que se da a la tarea para que se investiguen y se utilicen el la implementación del código necesario para la funcionalidad del programa, la finalidad del proyecto es lograr filtrar archivos *.wav* de una frecuencia determinada, con ciertos coeficientes dados, para ello se propone el uso de ciertas bibliotecas como: *Scipy.io.wavfile, Windsound y Matplotlib,* la finalidad de esta investigación es conocer y aprender sobre cómo utilizar cada una de las bibliotecas y sus diferentes funciones.

Scipy.io.wavfile

Es la biblioteca encargada de manejar los archivos .wav, es capaz tanto de leer como de escribir archivos .wav. Para utilizarla debemos primeramente importarla, entonces escribimos *import scipy.io.wavfile as waves,* en este caso se importa y se le coloca el nombre *waves*, con este nombre es que se utiliza en donde se necesite en el resto del código, cuando la vamos a utilizar para leer un archivo .wav, se coloca de la siguiente manera: *frecuencia*,*x=waves.read (archivo_a_filtrar)*, en esta estructura se utiliza *frecuencia* como el resultado que nos entrega la biblioteca sobre la frecuencia del archivo .wav que se le indique, se utiliza *x* para almacenar los datos que trae ese archivo .wav, la biblioteca se encarga de darnos valores numéricos, el *waves* es el nombre que se utilizó al importar la biblioteca y el *.read* funciona para indicar que se quiere que se lea el archivo que le vamos a indicar, que en este caso se llama *archivo_a_filtrar*. Para escribir un archivo .wav mediante valores numéricos utilizamos *waves.write*, que tiene la siguiente estructura *waves.write(nombre, frecuencia, audio_filtrado),* donde *nombre* es el nombre que se le quiera asignar al archivo saliente, *frecuencia* es la frecuencia del archivo que se va a generar y *audio_filtrado* es el conjunto de valores numéricos que se toman en cuenta al crear el archivo, cabe recalcar que en el nombre deseado se le puede agregar una ruta de acceso donde el usuario desee guardar el archivo.

NumPy

Es una librería utilizada para trabajar con arreglos (conjunto de datos homogéneos), permite crear y modificar matrices y realizar operaciones matemáticas mucho más fácil y rápido de lo que se haría con una lista, para utilizarla se debe importar así *import numpy as np* se le da el nombre de *np* en el que se usa en el resto del código, en la implementación en el proyecto solo se utiliza para convertir un conjunto numérico de una lista a un arreglo de esta forma, *audio_filtrado= np.asarray (y,dtype="int16"),* donde *audio_filtrado* es el resultado, *np.asarray* es para indicarle que se quiere que se genere un arreglo (array) y entre paréntesis se coloca primero el dato que se quiera convertir (la lista *y* en este caso), seguido de una coma y luego al tipo de dato que se quiere convertir que en este caso es *dtype="int16"*.

En el código se utiliza *np.linspace,* esta nos va a ayudar en la graficación de las señales, ya que, se le indica un término de inicio (start), un término final (stop), y la cantidad de datos que quiere que se muestren en la gráfica, entonces cuando el usuario indica la cantidad de muestras que desea graficar, esta tiene la función de delimitar el eje x para que inicie y termine en un número determinado, y así lograr el resultado deseado según la cantidad de muestras que indique el usuario.

Windsound

Permite la reproducción de sonidos en la plataforma Windows, en el proyecto se utiliza para reproducir los archivos .wav, tanto el ingresado por el usuario como el archivo que se genera luego de haber filtrado el ingresado, se importa la biblioteca

como import *winsound* y cuando se desea escuchar un archivo de audio se coloca de la siguiente manera, *winsound.PlaySound ((archivo_a_filtrar), winsound.SND_FILENAME)*, *winsound.PlaySound* le indica a la biblioteca que deseamos reproducir un archivo, se coloca el paréntesis y dentro se coloca el nombre del archivo que deseamos escuchar que en este ejemplo es *archivo_a_filtrar* se coloca una coma y luego *winsound.SND_FILENAME*, esto último le indica a la biblioteca que estamos ingresando el nombre del archivo que deseamos escuchar.

Matplotlib

Es la biblioteca que nos va a servir para graficar los audios, tanto del audio ingresado por el usuario como el audio generado por el filtro, se importa así *import matplotlib.pyplot as plt*, esta función se utiliza con varios comandos como:

plt.title('Gráfica de audios'), esto se utiliza para ponerle un título a la gráfica.

plt.xlabel('Tiempo(s)'), se le pone un nombre a los valores que se ven en el eje x.

plt.ylabel('Amplitud'), se le coloca un nombre a los valores del eje y.

plt.plot(tiempo, grafica_original), tiempo es la función *np.linspace* que se definió anteriormente, utilizada de la siguiente manera, *tiempo=np.linspace(start=0, stop=muestras/frecuencia, num=muestras),* esto funciona para delimitar al eje x según el número de muestras que indique el usuario, luego se coloca el arreglo que deseamos graficar.

plt.plot(tiempo, grafica_filtrada, color='green'), esta es igual a la anterior, solo que, se grafica otro arreglo y se le define un color (verde).

plt.legend(['Gráfica original', 'Gráfica filtrada']), se utiliza para generar un recuadro pequeño dentro de la gráfica para conocer que representa cada línea.

plt.show(), utilizado para que se muestre la gráfica previamente generada.

Resultados

Como se menciona anteriormente la utilidad esperada para el código creado es lograr filtrar un audio formato *.wav*, el código recibe un archivo de audio y su finalidad es filtrarlo, que el usuario pueda escuchar ambos, el ingresado y el resultante del filtro y que se grafiquen ambos audios, el código cumple con los requerimientos deseados y se nota la diferencia en el tono del sonido al reproducir un archivo o el otro, esto también se denota en la gráfica entre como varían los audios uno del otro.

Conclusión

Se investigó y se aprendió sobre el uso de las diferentes bibliotecas y funciones necesarias para la implementación del proyecto, herramientas muy importantes tanto en la implementación y funcionamiento del código utilizado en el proyecto, como también para poder utilizarlas en otros momentos donde se necesiten para implementar una función o programa que se desee y ya con el conocimiento de tales bibliotecas nos pueden facilitar el trabajo implementando un código. Es muy buena experiencia, ya que, al investigar sobre algunas funciones o bibliotecas vemos la cantidad que existen y la cantidad de cosas que se pueden realizar conociendo bien cada una de las funciones sería una gran herramienta al querer desarrollar un programa que deseemos implementar.

Referencias

scipy.io.wavfile.write. (s. f.). SciPy.org. https://docs.scipy.org/doc/scipy/reference/generated/scipy.io.wavfile.write.html

SciPy.org., (s.f) *scipy.io.wavfile.write*. Recuperado de: https://docs.scipy.org/doc/scipy/reference/generated/scipy.io.wavfile.write.html

SciPy.org., (s.f.) scipy.io.wavfile.read. Recuperado de: https://docs.scipy.org/doc/scipy/reference/generated/scipy.io.wavfile.read.html

Del Rosario, E. (2018, 27 febrero). Audio en formato .wav. Recuperado de: http://blog.espol.edu.ec/telg1001/audio-en-formato-wav/

Numpy.org. (s. f.). numpy.asarray. Recuperado de: https://numpy.org/doc/stable/reference/generated/numpy.asarray.html

NumPy.org. (s. f.). numpy.linspace. Recuperado de: https://numpy.org/doc/stable/reference/generated/numpy.linspace.html

Python.org. (s. f.). winsound. Recuperado de: https://docs.python.org/es/3.10/library/winsound.html

Solomon, B. (s. f.). *Python Plotting With Matplotlib (Guide)*. Recuperado de: https://realpython.com/python-matplotlib-guide/