Tarea Corta I - Autrum

Implementar un analizador de espectros simple para audio

Tecnológico de Costa Rica Escuela de Ingeniería en Computación Redes (IC 7602)

Estudiantes:

- Luis Oswaldo Ramírez Fernández
- Max Richard Lee Chung
- Kelvin Núñez Barrantes
- Kenny Vega Obando
- Kaled Sánchez Vargas

Profesor:

- Gerardo Nereo Campos Araya

Índice

Introducción

Autrum es una aplicación desarrollada en Python que sirve como analizador y reproductor de espectros de audio. La herramienta tiene dos modos de operación: uno para analizar señales de audio en tiempo real o desde archivos, y otro para reproducir archivos con extensiones .atm, mostrando gráficos en el dominio del tiempo y frecuencia. La implementación utiliza customTkinter para la interfaz de usuario, matplotlib para la representación gráfica, y numpy para el manejo de datos. Este documento tiene como objetivo describir detalladamente las características, objetivos y modos de operación de Autrum, además de explicar cómo la aplicación puede contribuir al entendimiento de la diversidad en las características vocales. Se incluirá la respuesta de la pregunta ¿Por qué las voces de los integrantes son diferentes? junto con la justificación de la misma en base a ejemplos en la aplicación.

Versiones utilizadas

Python: 3.11.4 Matplotlib: 3.7.2 Numpy: 1.25.2 CustomTKinter: 5.2.0 Sounddevice: 0.4.6 Scipy: 1.11.2

Forma de trabajo

El equipo de desarrollo optó por una metodología de trabajo basada en la asignación de tareas específicas, en la cual se segmentan los componentes y funcionalidades del proyecto para ser ejecutados de manera individualizada. A fin de facilitar una comunicación eficaz y un seguimiento riguroso del progreso, se empleó la plataforma Trello. Esta herramienta permitió no solo monitorear el estado de completitud de cada tarea asignada, sino también identificar y abordar cualquier inconveniente que pudiera surgir durante el proceso de desarrollo.

Componentes Implementados

Interfaz gráfica

La implementación del UI se dividió en 1 ventana y 3 subventanas, las cuales vamos a profundizar a continuación: el analizador, analizador con wav y el reproductor.

- *La ventana principal:* Lo importante de destacar en esta parte es el side_frame a la izquierda y la forma de restablecer cualquier subventana. Dentro de este frame se encuentran los 3 botones principales para navegar las ventanas y el botón “salir” para cerrar la aplicación, si se quiere descartar una grabación en el analizador o reiniciar el reproductor se debe seleccionar estos 3 botones. También se debe mencionar las 3 opciones de configuración para el analizador y la app en general:
    - Sample Rate: Es la tasa de muestreo que utiliza el analizador para tomar audio.
    - Refresh Rate: Es la tasa de refresco que se utiliza en el gráfico en tiempo real y frecuencia del analizador.
    - Appearance Mode: Esta es una opcion para cambiar el esquema de color de aplicacion, entre claro, oscuro y el seleccionado en el sistema.
- *Analizador:* En esta subventana se muestran 2 gráficos y 4 botones. Los gráficos se actualizan y mostrarán información una vez se presione el botón “Start”, por otra parte el botón de “Pause” pausa la toma de sonido y “Resume” lo continuará desde el punto pausado. Si se presiona “Pause” cuando ya se está pausado no pasara nada, al igual que “Resume” si ya está tomando sonido y “Start” no funcionara como “resume”. Por último, el botón “Stop&Save” terminará la grabación y guardará el archivo .atm en el escritorio (ubicación predeterminada). Se debe comentar que los gráficos de esta ventana se puede hacer zoom, desplazarse y guardar como imagen, sin embargo, se recomienda que si se hace zoom o se desplaza se debe utilizar la función “Back to previous view” y no “reset original view” ya que provocará interacciones no deseadas.
- *Analizador con wav:* Esta subventana solo posee un botón que abre el navegador de archivos y se debe seleccionar el archivo .wav deseado. Una vez seleccionado se analizará y generará el archivo .atm en el escritorio del usuario donde se podrá utilizar posteriormente en el reproductor
- *Reproductor:* Dentro de esta última subventana, al igual que en el analizador, se muestran 2 gráficos y 4 botones. Los gráficos se actualizarán una vez se seleccione el archivo .atm con el botón “Subir .atm” y este ejecutará el WAV inmediatamente junto con los gráficos analizados previamente. Además, se muestra el botón de “Pause” para pausar la reproducción y “Resume” para resumirla. Por último, se encuentra el botón de “Stop” para detener el reproductor.

Analizador

Dentro del analizador se encuentran dos funciones, captar una entrada de audio (ventana analizador) o analizar los espectros de un archivo .wav cualquiera (ventana analizador wav) con el mismo fin de procesar los datos y crear el archivo .atm en la carpeta “Desktop”, el archivo con extensión ".atm" se utiliza para almacenar

los datos de audio grabados junto con la información necesaria para su posterior análisis y reproducción. El proceso de guardar datos de audio como un archivo ".atm" se realiza de la siguiente manera:

- Se crea un diccionario llamado "atm_data," que contiene dos elementos clave "audio" que almacena los datos de audio grabados y "sample_rate": que almacena la frecuencia de muestreo de los datos de audio.
- El diccionario "atm_data" se guarda en un archivo con extensión ".atm" en el directorio del escritorio del usuario, esto se logra utilizando la biblioteca pickle.
- Una vez que los datos se han guardado correctamente en el archivo ".atm," se muestra un mensaje de información que indica que el audio se ha guardado exitosamente junto a su respectivo gráfico.

Para la captura de audio se utiliza la librería Sounddevice de esta manera se puede capturar el flujo de audio en un arreglo que podemos controlar con un estado booleano por si se quiere iniciar, pausar, resumir o terminar el flujo. Además, mientras se captura y analiza los espectros, se actualiza a tiempo real los gráficos con los respectivos valores en términos de su dominio en el tiempo y su frecuencia. Cuando se desee ingresar un archivo .wav, el analizador tendrá un botón para subir el archivo que procesa automáticamente los datos internamente para terminar con una ventana mostrando los gráficos con toda su información al usuario.

Reproductor

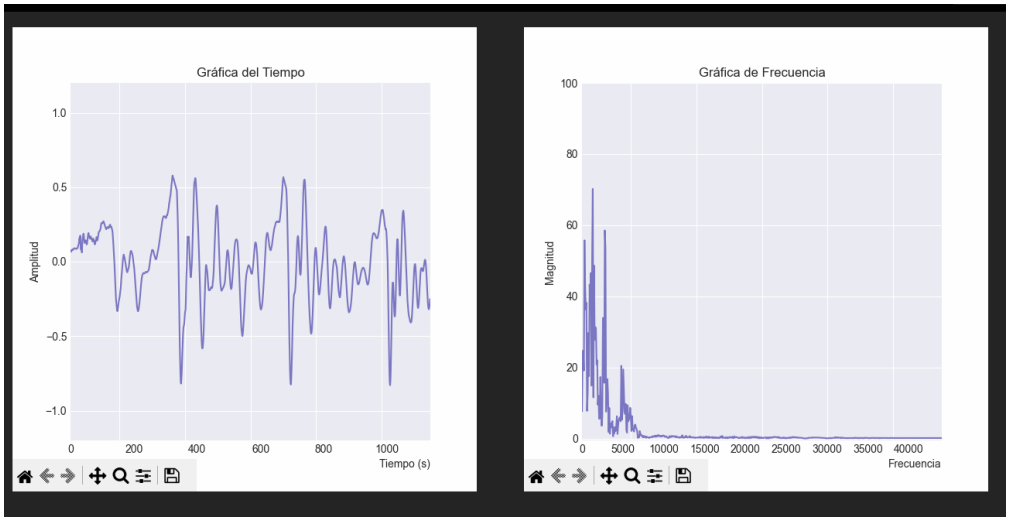
Dentro del reproductor, la función principal es de reproducir el análisis realizado previamente, extrayendo los datos del archivo .atm, en donde se verifica los datos para inmediatamente ejecutar el wav almacenado seguido de sus correspondientes datos.

Pruebas realizadas

El programa se ejecuta desde la carpeta ./Tarea Corta I - Autrum usando python Code/main.py

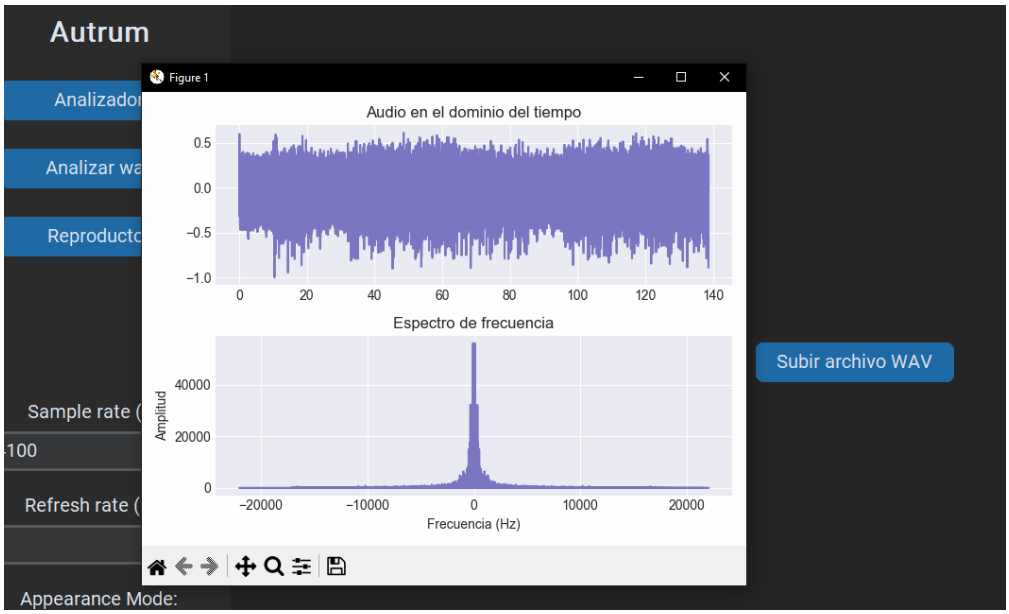
1.Prueba del analizador en tiempo real:

En esta prueba, se evaluó el funcionamiento del analizador en tiempo real, el objetivo fue verificar la capacidad de la aplicación para capturar y analizar señales de audio que se captan al momento de estar grabando el audio.



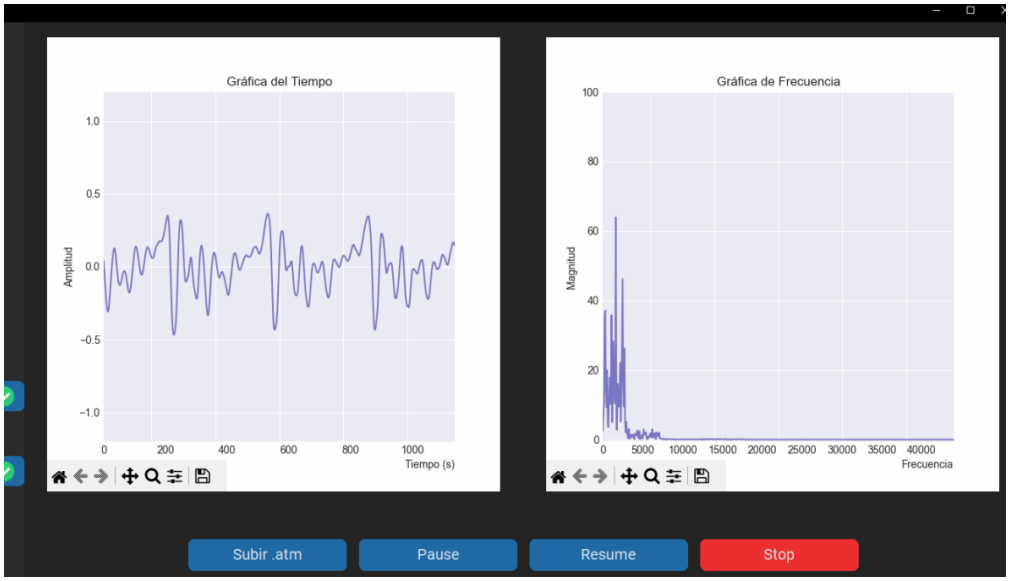
2.Prueba del analizador de WAV’s

Se evaluó la capacidad del analizador de la aplicación para procesar archivos de audio en formato .WAV, el objetivo principal fue verificar que la aplicación pueda cargar, analizar y visualizar adecuadamente el análisis de las grabaciones de voz almacenadas en archivos .WAV.



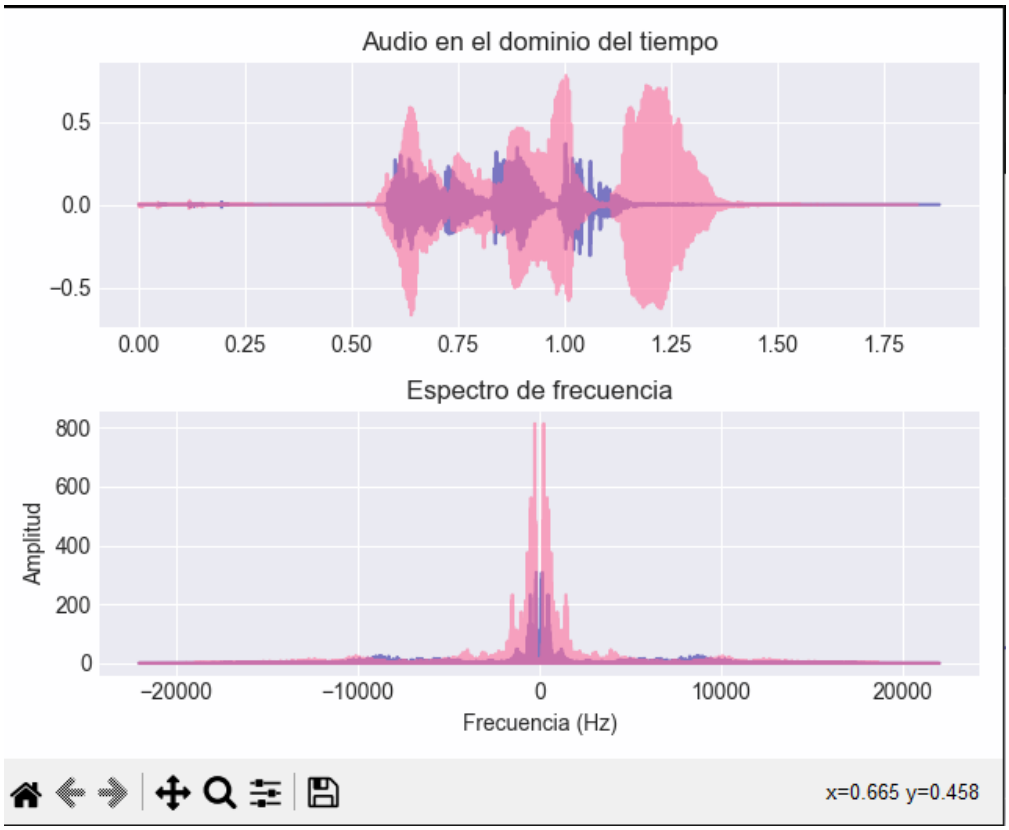
3. Prueba Subida de Archivo .ATM:

Se evaluó la capacidad del reproductor y del analizador de la aplicación para procesar archivos de audio en formato .ATM, el objetivo principal fue verificar que la aplicación pueda cargar, analizar y visualizar adecuadamente las grabaciones de voz almacenadas en archivos .ATM.



4. Análisis de múltiples grabaciones superpuestas:

En esta prueba se analizaron múltiples grabaciones de audio a la misma vez, se verificó que el programa es capaz de gestionar el análisis de múltiples grabaciones en una misma gráfica, en el caso de ejemplo se usaron hasta 4 grabaciones al mismo tiempo, cada color diferente corresponde a una grabación diferente.



Conclusiones

- Se concluye que el analizar una onda de audio mediante una transformada de Fourier da la capacidad de visualizar características de la onda muy fácilmente, tales como su gráfica de tiempo y frecuencia, esto nos permite ver, en una grabación, cuales frecuencias son las que predominan más (en el caso de furier) y ver la forma que tienen las ondas (tiempo).

En el caso de la música esto nos permite poder observar las notas/frecuencias que se están tocando, y algo interesante, es que al reproducir música en 8bits, como la usada en consolas antiguas, podemos ver los diferentes tipos de “instrumentos” que tenían disponibles: square waves, sin waves y saw waves (ondas con forma de seno, ondas escalonadas, y picos).

- Se concluye que Python y sus múltiples librerías tales como: Matplotlib, Numpy, Pickle y entre otros, son muy útiles para el manejo de señales de audio y la creación de gráficos de una manera eficiente y eficaz.
- Esta tarea sirve de gran ayuda para entender el comportamiento de las ondas y cómo estas cambian sus gráficos de frecuencia y de tiempo dependiendo de cómo sea la grabación.
- Nos brindó la oportunidad de familiarizarnos con los temas vistos en clase, como las ondas, sus características, entender qué son los armónicos de una onda y cómo hacer la transformada de fourier y comprenderla.

Referencias Bibliográficas

- python-sounddevice (2020). Streams using NumPy Arrays. Recuperado de python-sounddevice
- CustomTkinter(2023). complex_example.py. Recuperado de CustomTkinter
- Matplotlib.org. (2023). Matplotlib: Matplotlib.org. https://matplotlib.org/