IMAGE SONIFICATION

❖Overview

The Image Sonification Tool uses various mapping techniques to convert visual data from images into auditory experiences. This innovative approach enhances accessibility and offers a unique way to interpret and experience visual data through sound. The project encompasses pixel extraction, the application of different sonification mappings, and the integration of these processes into a user-friendly Command-Line Interface (CLI).

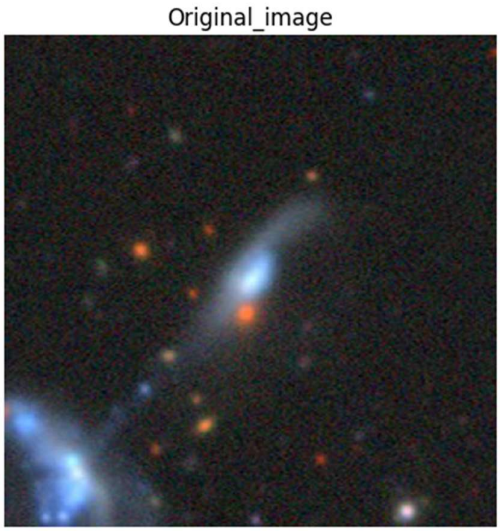
❖Pixel Extraction

Pixel extraction is the foundational step in the sonification process. It transforms an image into numerical data representing each pixel's colour values. This data is essential for further processing and mapping.

Steps:

1. **Loading the Image:**
    - The image is loaded using an image processing library, such as OpenCV, and converted into a NumPy array for easier manipulation.
    - The image is converted to RGB format if needed, ensuring consistency in colour values.
2. **Extracting Pixel Data:**
    - Each pixel's colour values (Red, Green, and Blue) are extracted.
    - The pixel data is stored in a DataFrame, making it accessible for subsequent processing steps.

❖Sample Image (6th image from images.npy)



❖Sample Pixel Data

	R	G	B
1	64	30	29
2	42	38	35
3	17	36	32
4	27	37	29
5	36	22	19
6	39	20	24
7	43	40	49
8	23	31	34
9	19	24	20
10	24	19	16
11	32	27	33
12	44	41	50
13	41	35	37
14	37	30	22
15	37	37	29
16	24	33	30
17	44	30	27
18	58	42	27
19	48	35	18
20	36	36	28

❖Sonification Mappings

Various sonification mappings are applied to the extracted pixel data to generate sound. Each mapping technique offers a distinct way to translate visual information into auditory signals.

1. Colour-Based Sound Effects

This mapping generates sound waves based on the average colour values of the pixels. It provides a direct correlation between colour intensity and sound frequency.

Process:

- **Calculate Average Colour Values:** For each row of pixels, the average values of Red, Green, and Blue are calculated.
- **Generate Sine Waves:** Sine waves are generated for each colour channel with specific frequencies (Red: 440 Hz, Green: 660 Hz, Blue: 880 Hz).
- **Combine Sine Waves:** The sine waves for each colour channel are combined to create a single wave for each row of pixels.

2. Pitch Mapping Based on Brightness

This mapping translates the brightness values of pixels into corresponding pitches, creating sound based on light intensity.

Process:

- **Convert Colour to Brightness:** The brightness of each pixel is calculated using a weighted sum of the red, green, and blue values.
- **Map Brightness to Pitch:** The brightness values are mapped to specific pitches, creating a range of tones based on light intensity.
- **Generate Sine Waves:** Sine waves are generated based on the calculated pitches, producing a sound that reflects the image's brightness.

3. Frequency Mapping

This mapping converts RGB values into specific frequency ranges, producing a diverse range of sounds that correlate with colour intensity.

Process:

- **Map RGB to Frequency:** Each colour value (Red, Green, Blue) is mapped to a specific frequency range.
- **Generate Sine Waves:** Sine waves are created for each frequency, corresponding to the colour values.
- **Combine Frequencies:** The sine waves for each colour channel are combined, resulting in a complex and rich auditory representation of the image.

4. HSV Mapping

This mapping uses the HSV (Hue, Saturation, Value) colour space to determine sound properties such as frequency, amplitude, and duration.

Process:

- **Convert RGB to HSV:** The RGB values are converted to HSV colour space.
- **Map Hue to Frequency:** The hue value is mapped to frequency, determining the pitch of the sound.
- **Map Saturation to Amplitude:** The saturation value is mapped to amplitude, affecting the volume of the sound.
- **Map Value to Duration:** The value (brightness) is mapped to duration, controlling the length of the sound.
- **Generate Sine Waves:** Sine waves are created based on the HSV values, producing a sound that encapsulates the colour's characteristics.

❖Combining Code and Creating CLI

To enhance usability, all the mappings and pixel extraction functions are integrated into a single script with a Command-Line Interface (CLI). This interface allows users to interact with the tool seamlessly, selecting images and sonification modes through simple prompts.

Integration:

- **Backend Functions:** All pixel extraction and sonification mapping functions are defined in a backend module.
- **CLI Implementation:** The CLI script prompts the user to select an image and a sonification mode, then calls the appropriate backend functions to generate and play the sound.

❖Example Workflow:

1. **Image Selection:** The user is prompted to enter the index of the image they wish to process.
2. **Sonification Mode Selection:** The user selects a sonification mode from the available options.
3. **Sound Generation:** The selected sonification mapping is applied to the pixel data, generating the corresponding sound.
4. **Output:** The generated sound is saved and played, providing an auditory experience based on the image's visual data.

❖Conclusion

The Image Sonification Tool offers a novel approach to experiencing visual data through sound. By leveraging various sonification mappings, the tool translates the intricate details of images into rich auditory signals. This project demonstrates the potential of sonification in enhancing accessibility and providing new ways to interpret and appreciate visual information.