Análisis Espectogramas

October 5, 2023

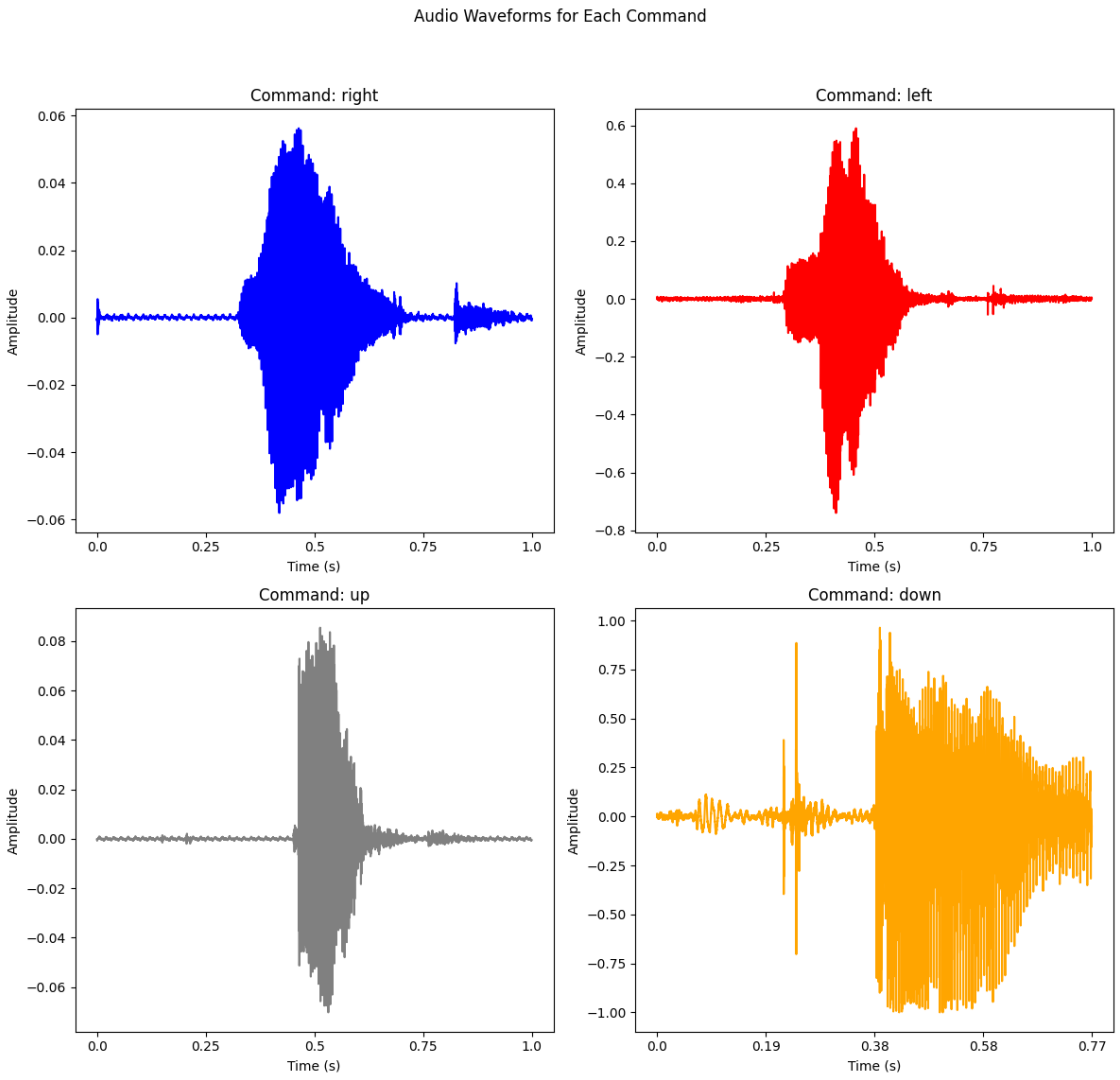
[1]:
```
import tarfile
import os
import librosa
import librosa.display
import matplotlib.pyplot as plt
import numpy as np
from sklearn.decomposition import PCA
```

[5]:
```
# Define the extraction path
extraction_path_directions = "/Users/marcgrayson/Downloads/direction_commands.
 ↪extracted"

# Extract the content of the tar file
with tarfile.open("/Users/marcgrayson/Downloads/direction_commands1.tar", 'r')␣
 ↪as tar:
    tar.extractall(path=extraction_path_directions)

# List the extracted folders to verify
commands = os.listdir(extraction_path_directions)
print(commands[:])
```

```
['right', 'left', 'up', 'down']
```

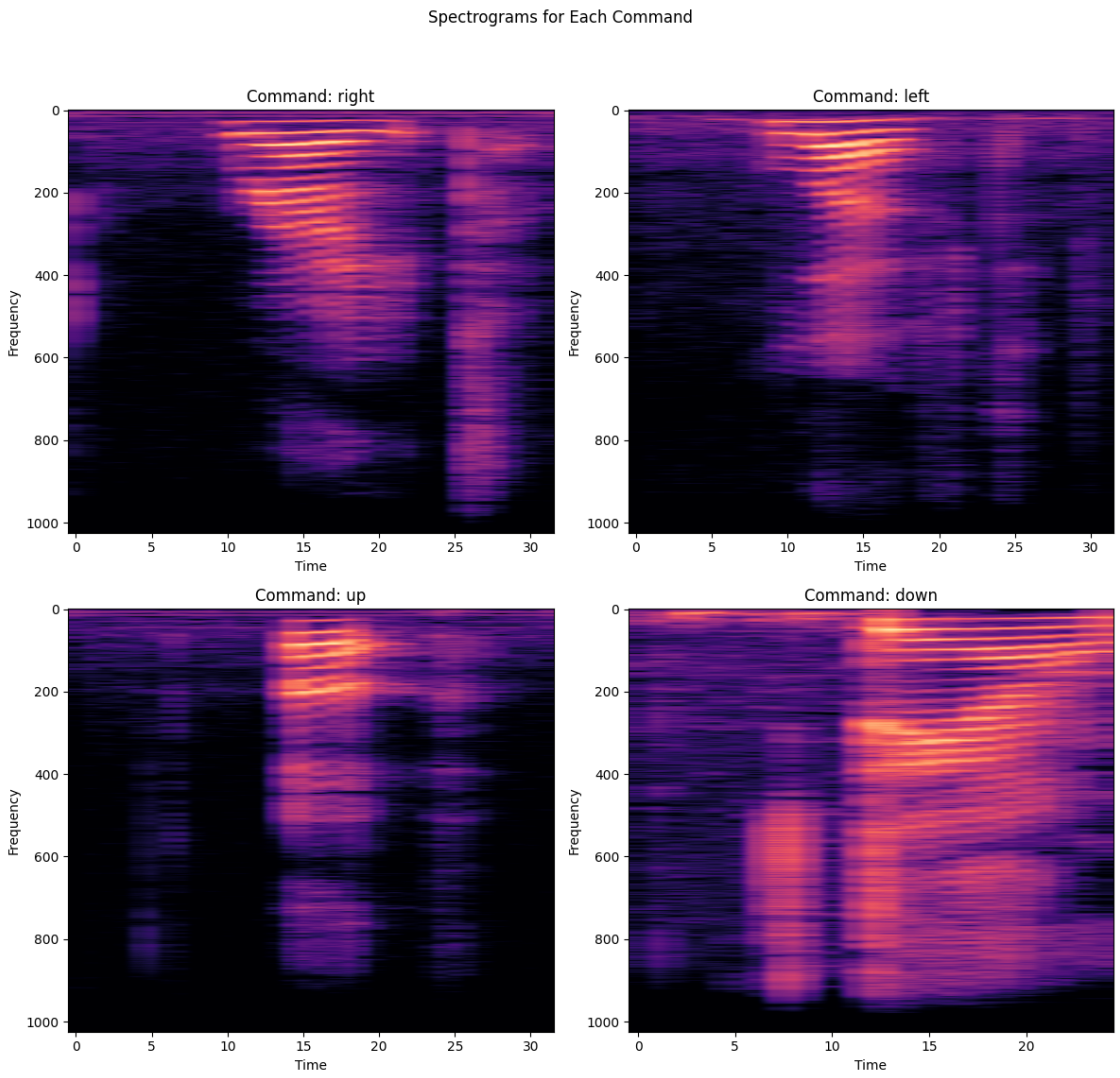
[41]:
```
# Colors for waveforms
colors = ['b', 'r', 'grey', 'orange']

# For demonstration, take the first audio file in each command folder
sample_audio_paths = {command: os.listdir(os.path.
 ↪join(extraction_path_directions, command))[0] for command in commands}

# Create a 2x2 grid for Audio Waveforms
fig, axs = plt.subplots(2, 2, figsize=(12, 12))
fig.suptitle('Audio Waveforms for Each Command')
for idx, command in enumerate(commands):
    row = idx // 2
    col = idx % 2
```

```
    audio_data, sample_rate = librosa.load(os.path.
 ↪join(extraction_path_directions, command, sample_audio_paths[command]),␣
 ↪sr=None)
    axs[row, col].plot(audio_data, color=colors[idx])
    axs[row, col].set_title(f'Command: {command}')
    axs[row, col].set_xlabel('Time (s)')
    axs[row, col].set_ylabel('Amplitude')
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()
```

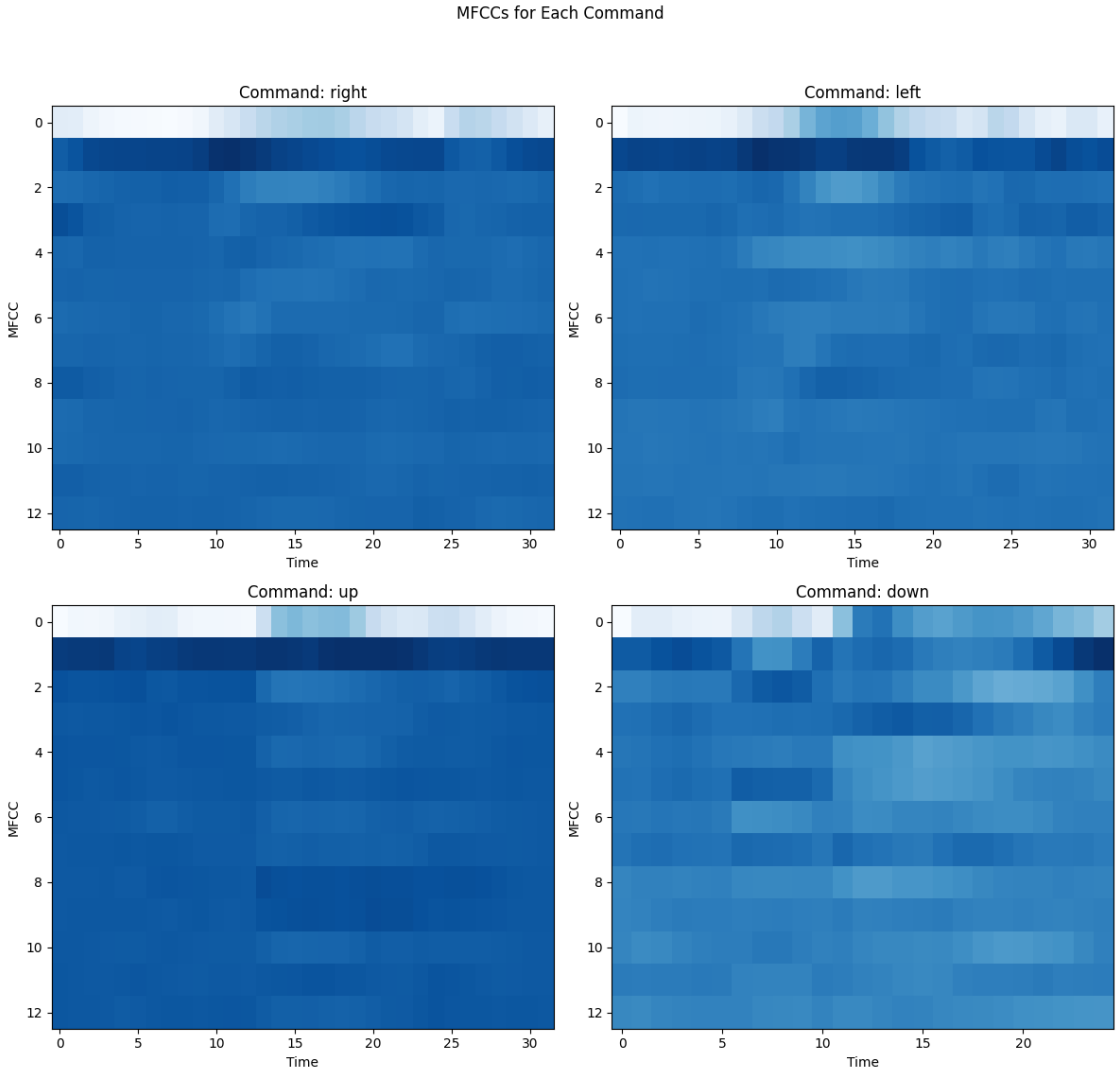


[52]:
```
# 2 Create a 2x2 grid for Spectrograms
fig, axs = plt.subplots(2, 2, figsize=(12, 12))
fig.suptitle('Spectrograms for Each Command')
for idx, command in enumerate(commands):
```

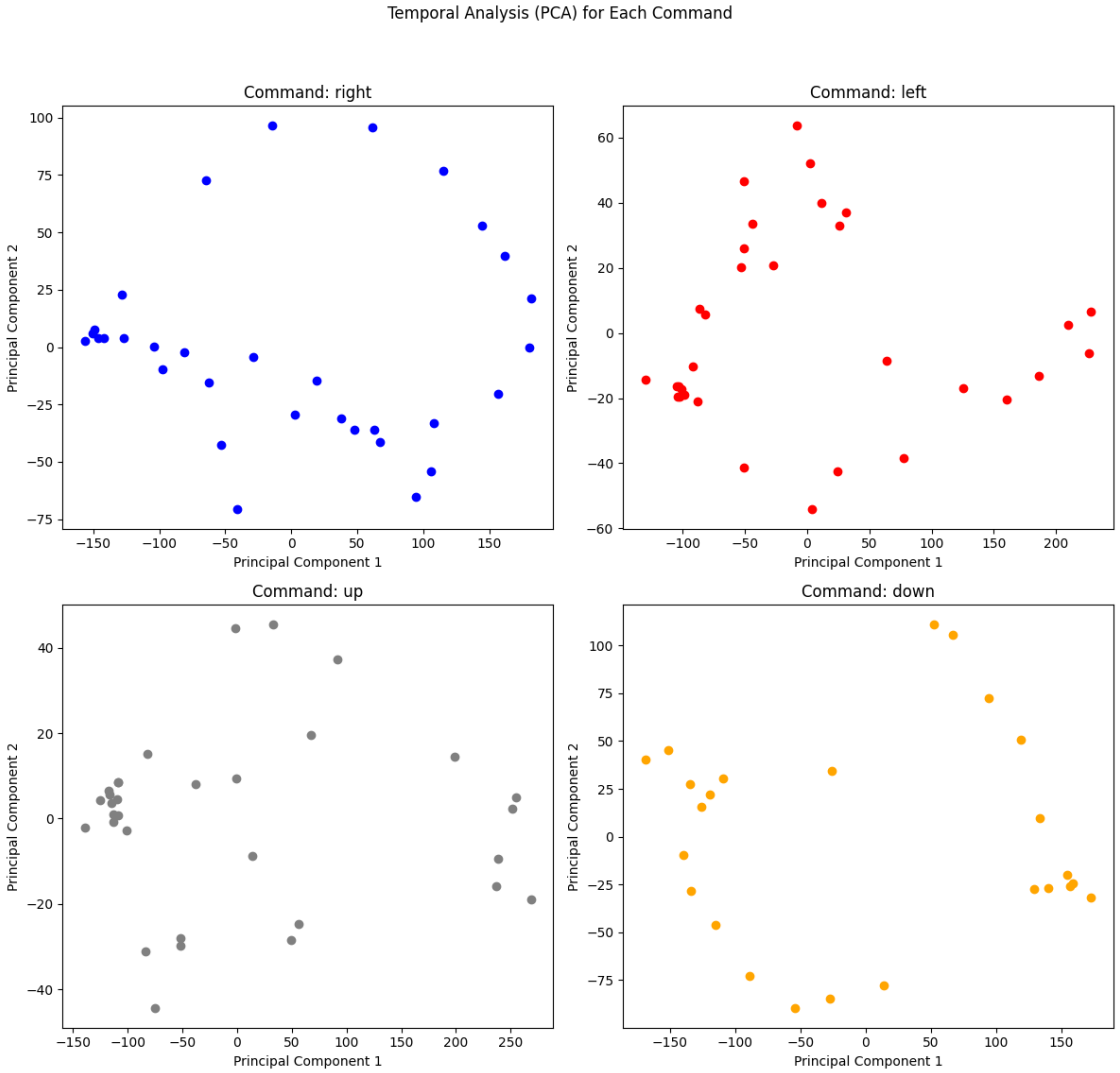
```
    row = idx // 2
    col = idx % 2
    audio_data, sample_rate = librosa.load(os.path.
 ↪join(extraction_path_directions, command, sample_audio_paths[command]),␣
 ↪sr=None)
    D = librosa.amplitude_to_db(np.abs(librosa.stft(audio_data)), ref=np.max)
    axs[row, col].imshow(D, aspect='auto', cmap='magma')
    axs[row, col].set_title(f'Command: {command}')
    axs[row, col].set_xlabel('Time')
    axs[row, col].set_ylabel('Frequency')
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()
```



[49]:
```
# 3 Create a 2x2 grid for MFCCs
fig, axs = plt.subplots(2, 2, figsize=(12, 12))
fig.suptitle('MFCCs for Each Command')
for idx, command in enumerate(commands):
    row = idx // 2
    col = idx % 2
    audio_data, sample_rate = librosa.load(os.path.
 ↪join(extraction_path_directions, command, sample_audio_paths[command]),␣
 ↪sr=None)
    mfccs = librosa.feature.mfcc(y=audio_data, sr=sample_rate, n_mfcc=13)
    axs[row, col].imshow(mfccs, aspect='auto', cmap='Blues')
    axs[row, col].set_title(f'Command: {command}')
    axs[row, col].set_xlabel('Time')
    axs[row, col].set_ylabel('MFCC')
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()
```



[15]:
```
for command in commands:
    # 4. Temporal Analysis using PCA
    pca = PCA(n_components=2)
    mfccs_pca = pca.fit_transform(mfccs.T)
    plt.figure(figsize=(12, 3))
    plt.scatter(mfccs_pca[:, 0], mfccs_pca[:, 1])
    plt.title(f'Temporal Analysis (PCA) for Command: "{command}"')
    plt.xlabel('Principal Component 1')
    plt.ylabel('Principal Component 2')
    plt.show()
```



[ ]: