实验2报告

于新雨 计25 2022010841

思路

首先以 0.1 为阈值，判断当前音频是否静默
接着做傅里叶变换，从频谱中找出 600Hz ~ 1000Hz，1100Hz ~ 1600Hz 内对应幅度值最大的频率，并且找到和该频率最接近的按键低频或者高频
最终由找到的按键低频或者高频找到对应的按键

代码

解析频率找到按键

```
def get_closest_frequency(freq, freq_list):
    return min(freq_list, key=lambda x: abs(x - freq))

def detect_key(low_freq, high_freq):

    low = get_closest_frequency(low_freq, low_freqs)
    high = get_closest_frequency(high_freq, high_freqs)
    return dtmf_freqs.get((low, high), -1)
```

找低频最大值和高频最大值

```
def key_tone_recognition(audio_array):
    # Unpack audio and sample rate
    signal, sr = audio_array
    frame_size = int(sr / 64)  # 750 samples per frame
    num_frames = len(signal) // frame_size

    # RMS threshold for silence
    rms_threshold = 0.1

    output = []

    for i in range(num_frames):
        # Extract current frame
        frame = signal[i * frame_size:(i + 1) * frame_size]
        # Calculate RMS energy
        rms = np.sqrt(np.mean(frame**2))
        if rms < rms_threshold:
            # Silence detected
            output.append('-1')
            continue
```

```
        # Perform FFT
        fft_result = np.fft.rfft(frame)
        freqs = np.fft.rfftfreq(len(frame), 1 / sr)
        magnitudes = np.abs(fft_result)

        # get the highest peak in the low frequency band
        # convert freqs and magnitudes to map
        freqs_map = dict(zip(freqs, magnitudes))
        # get the key of the max value in the freqs_map where the key is between
600 and 1000
        low_peak_key = max({k: v for k, v in freqs_map.items() if 600 <= k <=
1000}.items(), key=lambda x: x[1])[0]
        high_peak_key = max({k: v for k, v in freqs_map.items() if 1100 <= k <=
1600}.items(), key=lambda x: x[1])[0]
        # Detect key based on frequency peaks
        detected_key = detect_key(low_peak_key, high_peak_key)
        output.append(str(detected_key))

    # Combine the results into the expected format
    return ' '.join(output)
```