多媒體系統與應用Project2-Speech Processing

F74094017 李昆翰

一、Waveform output

1. code:

import numpy as np

import matplotlib.pyplot as plt

import math

import scipy.io.wavfile as wav

import parselmouth # making Pitch contour

'''

global variables

'''

sample\_rate, audio = wav.read("../audio/audio.wav")

data = np.array(audio).T[1] # normalize the signal

frame\_size = int(round(sample\_rate \* 0.01)) # the frame size is 10 ms

frame\_num = math.ceil(len(data) / frame\_size)

'''

waveform plotting

'''

plt.figure(figsize=(18, 7))

plt.plot(data)

plt.title("Waveform")

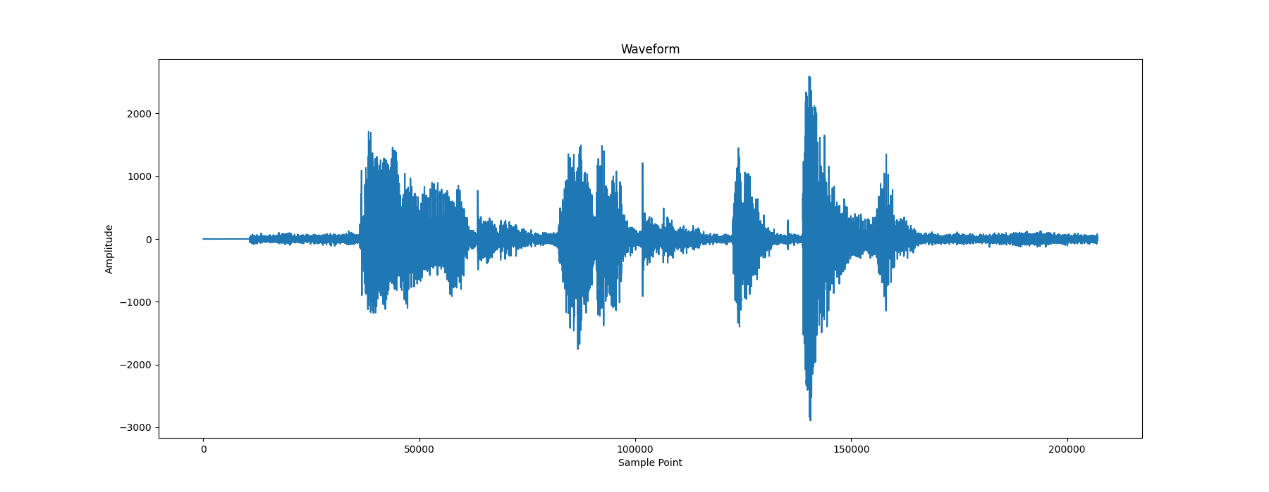
plt.xlabel("Sample Point")

plt.ylabel("Amplitude")

plt.savefig('../figure/wavform.png')

2. 說明:

這裡是使用 scipy.io.wavfile 去讀取 .wav 檔案，並將讀取的資料先做一些處理，比如說定義frame\_size、frame\_num 等，之後將 .wav圖表做輸出。



二、Energy contour output:

1. code:

'''

Energy contour plotting

'''

def cal\_energy\_contour() :

energy = np.zeros(frame\_num)

for i in range(frame\_num) :

curr\_frame = np.array(data)[np.arange(i \* frame\_size, min((i + 1) \* frame\_size, len(data)))]

total = 0

for val in curr\_frame :

total += pow(val, 2)

energy[i] = total

return energy

energy = cal\_energy\_contour()

plt.figure(figsize=(18, 7))

plt.plot(energy)

plt.title("Energy Contour")

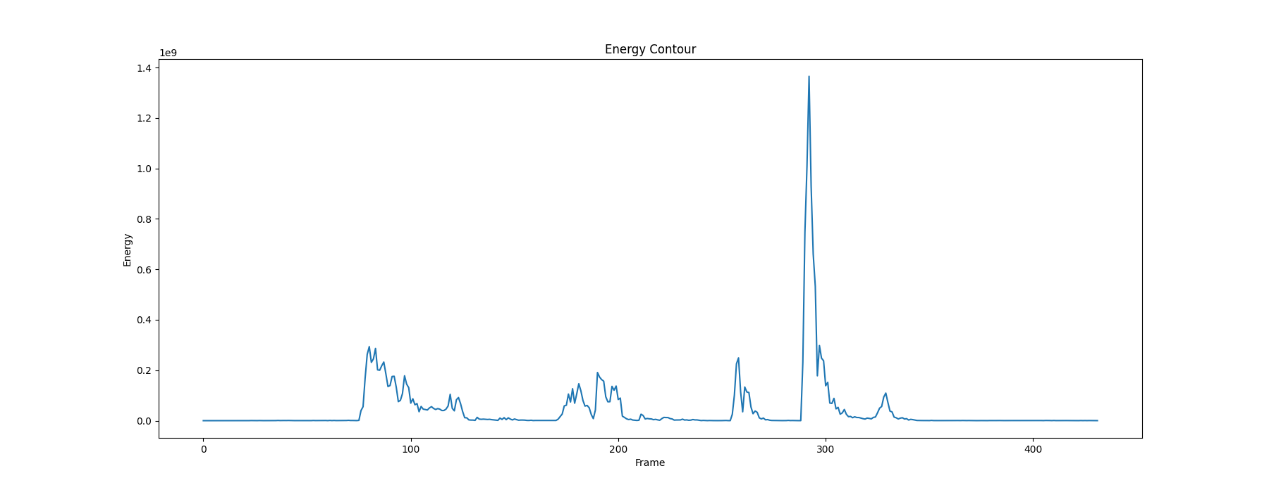
plt.xlabel("Frame")

plt.ylabel("Energy")

plt.savefig('../figure/energy\_contour.png')

2. 說明:

這裡主要是用 short-time energy 的做法來製作相關的資料。計算的方法是對每個 data 中的 frame，去算所有 sample point 的平方和，最後將算完的結果存入 energy list 中。



三、Zero-crossing rate contour output

1. code:

def zero\_cross\_rate\_contour():

zero\_cross = np.zeros(frame\_num)

for i in range(frame\_num) :

curr\_frame = np.sign(np.array(data)[np.arange(i \* frame\_size, min((i +

1) \* frame\_size,

len(data)))])

zero\_cross[i] = sum(np.abs(np.diff(curr\_frame))) / len(curr\_frame)

return zero\_cross

zero\_cross = zero\_cross\_rate\_contour()

plt.figure(figsize=(18, 7))

plt.plot(zero\_cross)

plt.title("Zero-crossing Rate Contour")

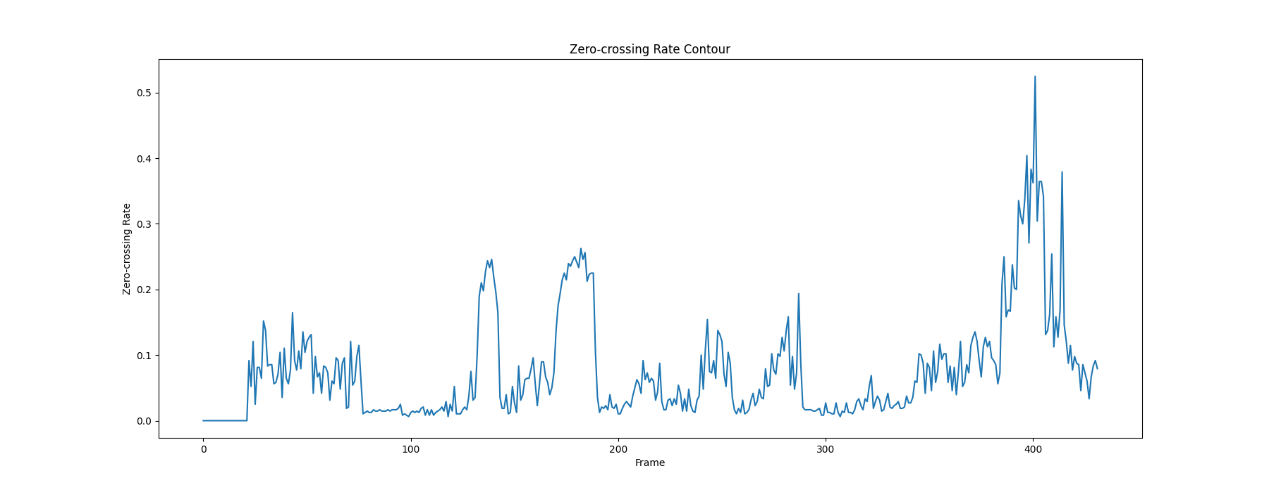
plt.xlabel("Frame")

plt.ylabel("Zero-crossing Rate")

plt.savefig('../figure/zero-crossing\_rate\_contour.png')

2. 說明:

這裡的作法是對每一個 frame 的全部 sample point 的相鄰 sample point 互相做差值計算。由於我們使用的資料 data 已經被 normalize 過了，所以這裡是使用 np.diff 來計算。根據np.diff，它會沿著軸去計算和回傳離散值，所以我們可以將這些數值全部加總，然後去和 frame 的長度相除，得到平均值，也就是我們要的 zero crossing rate



四、Pitch contour output

1. code:

pitch = parselmouth.Sound('../audio/audio.wav').to\_pitch()

pitchValue = pitch.selected\_array['frequency']

pitchValue[pitchValue == 0] = np.nan

plt.figure(figsize=(18, 7))

plt.plot(pitch.xs(), pitchValue)

plt.title("Pitch Contour")

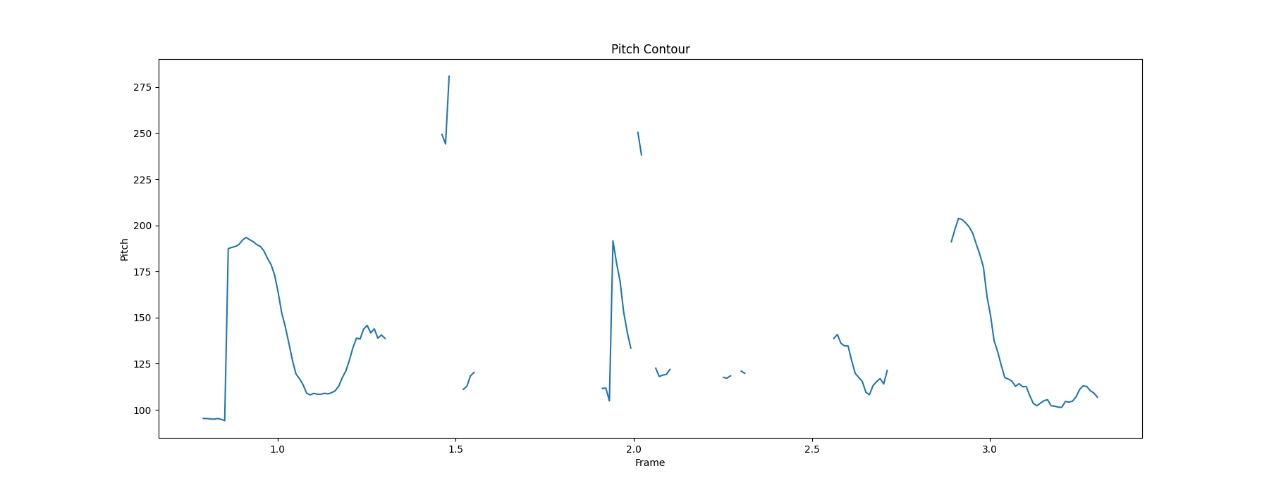
plt.xlabel("Frame")

plt.ylabel("Pitch")

plt.savefig('../figure/pitch\_contour.png')

2. 說明:

這裡我使用的是 praat- parselmouth 這個 python 套件來去幫我們從音檔中取得 pitch。



五、End-point detection output

1. code:

def end\_point\_detec() :

start\_point = []

end\_point = []

# thres = 10 -> at least the begining top ten frame doesn't have anything

izct = 0.6 # depend on results

itl = sum(energy) / len(energy) / 1.5 # depend on results

itu = 2.5 \* itl

is\_end = 0

curr = 0

spec = 0

for en\_x, en\_y in enumerate(energy):

if is\_end == 0:

if spec == 0:

if en\_y >= itl:

spec = 1

curr = en\_x

elif spec == 1:

for chk\_x, chk\_y in enumerate(energy[np.arange(curr, len(energy))]):

if chk\_y >= itu:

spec = 2

break

elif chk\_y < itl:

spec = 0

break

else:

over\_zero\_thres = 0

for chk\_x, chk\_y in enumerate(zero\_cross[np.arange(max(0, curr - 10), curr)]):

if chk\_y >= izct:

over\_zero\_thres += 1

if over\_zero\_thres >= 3: # by ppt

curr = chk\_x

start\_point.append(curr)

is\_end = 1

spec = 0

else:

if spec == 0:

if en\_y < itl:

spec = 1

curr = en\_x

else:

over\_zero\_thres = 0

for chk\_x, chk\_y in enumerate(zero\_cross[np.arange(curr, min(curr + 10, len(zero\_cross)))]):

if chk\_y >= izct:

over\_zero\_thres += 1

if over\_zero\_thres >= 3: # by ppt

curr = chk\_x

end\_point.append(curr)

is\_end = 0

spec = 0

return start\_point, end\_point

start\_point, end\_point = end\_point\_detec()

points = np.array([np.array(start\_point).dot(frame\_size),

np.array(end\_point).dot(frame\_size)]).T

# extend to the point in waveform

plt.figure(figsize=(18, 7))

plt.plot(data, label = 'Waveform')

for p in points:

plt.axvline(x = p[0], color = "#2ca02c") # start point -> green

plt.axvline(x = p[1], color = "#d62728") # end point -> red

plt.title("End Point Detection")

plt.xlabel("Sample Point")

plt.ylabel("Amplitude")

plt.savefig('../figure/end\_point\_detection.png')

2. 說明:

這裡主要還是以 ppt 的內容為準、網路上的資料為輔去做的。從 ppt 中的概念來說，end point detection 最主要是依靠 energy contour 和 zero crossing rate detection 聯合達成的。因此，在這支 code 中我們用到了 energy contour 以及 zero crossing rate contour 的結果。

根據 ppt 的論述，在紀錄 start\_point 中，我們會用 loop 的方式看 energy contour 的 sample frame 的值是否是大於 itl。若是的話再看往後的frame 有沒有超過 itu。若又超過的話，先記錄 start\_point 的值，最後再看 zero crossing rate contour 的指定範圍內的 frame 有沒有超過 izct，如果超過izct 三次貨以上會將當下紀錄的數值做更新。

反過來，end\_point 的更新比起 start\_point 就少了要看 itu 的環節，只有小於 itl 時才會開始記錄 end\_point，並再看 zero crossing rate contour 的指定範圍內的 frame 有沒有超過 izct。若超過就會對當下記錄的數值做更新。

如果我們看這支程式的輸出的話，對比我們的 energy contour 結果，可以看到有趣的一點是，因為我們的錄音在 energy contour 中有部分是連在一起的關係（連起來的字詞就是一開始的 “多媒體”），所以在 end point detection 的輸出中，可以發現它把這三個字的 wave 認定成了一個字的開始及結束。也因此，我們才會看到最終的輸出只有六個區塊，而非八個區塊。

