Sistem de analiză a calității vocii cântate

Kelerman Eric Bernard

Abstract

Frumusețea discursului uman se află în complexitatea diferitelor sunete care pot fi produse de câteva tuburi și mușchi. Această complexitate, totuși, face ca prelucrarea vorbirii să fie o sarcină dificilă. O caracteristică definitorie a vorbirii este pitch-ul său. Detectarea acestui pitch sau echivalent, detecția frecvenței fundamentale a unui semnal de vorbire este importantă în multe aplicații de vorbire. Detectoarele de trecere sunt utilizate în vocodere, sisteme de identificare și verificare a vorbitorilor și, de asemenea, ca ajutoare pentru persoanele cu handicap.[1] Datorită importanței sale, multe soluții pentru detectarea pitch-ului au fost propuse atât în domeniul timp cât și în cel al frecvenței. O astfel de soluție este detectarea pitch-ului prin utilizarea metodei Autocorelație și a funcției de diferență medie a amplitudinii (AMDF), metodă care este efectuată în domeniul timpului, iar cealaltă detectează natura armonică în domeniul frecvențelor.Cu toate acestea am folosit o metoda mai exacta pentru detecttia automata a frecventei fundamentale bazata pe maximele acestor functii.

Tema proiectului meu este un sistem de analiza al calitatii vocii cantate, mai exact acordarea vocii in gama majora. Pentru realizarea proiectului am folosit programul Anaconda Navigator in limbajul de asamblare python.

Cuvinte cheie: detectie, filtrare, frecventa fundamentala, gama majora

1. Introducere

Corzile vocale reprezintă două pliuri simetrice ale membranei laringelui. În timpul fonației, înălțimea sunetului rezultat este controlată de poziția acestor pliuri și de gradul de închidere/deschidere a orificiului delimitat de ele.
Periodicitatea sunetului este însă dată de vibrația liberă a unui segment al corzilor vocale și nu de mișcarea controlată a acestora cu ajutorul mușchilor conecși așa cum este de cele mai multe ori considerat.
Această oscilație a corzilor vocale determină periodicitatea semnalului vocal pe segmentele sonore. Măsura
obiectivă a acestei periodicități este definită de frecvența fundamentală (F0).
În limba engleză, însă, există o diferență între termenii ”pitch” și ”fundamental frequency”. ”Pitch” se referă la frecvența fundamentală percepută, iar ”fundamental frequency” se referă la frecvența fundamentală obiectiv calculată din semnalul vocal. Această diferență vine, printre altele și din faptul că, dacă într-un semnal audio se înlătură primele câteva armonici, frecvența percepută de ascultător va fi tot cea fundamentală, deși în spectru aceasta nu este prezentă, timbrul sunetului fiind însă ușor diferit. Astfel că, vom defini frecvența fundamentală (F0) ca fiind frecvența de oscilație a corzilor vocale și cea mai joasă frecvență prezentă în spectrul semnalului vocal. Deoarece frecvența fundamentală este rezultatul caracteristicilor fiziologice ale unei persoane, valoarea sa medie variază de la un vorbitor la altul, iar diferențe majore există și între categoriile de vorbitori. La femei, valoarea medie a F0 este de aproximativ 210Hz, iar la bărbați este de 120Hz. În cazul copiilor, valoare medie F0 urcă până la 300Hz. A nu se face confunzia între F0 din timpul vorbirii cu cea din timpul cântatului. Un cântăreț trebuie să modifice constant frecvența fundamentală pentru a putea reda notele muzicale. În cadrul aplicațiilor de prelucrare de voce, frecvența fundamentală este extrem de importantă, iar cei mai mulți algoritmi de codare a semnalului vocal sau de procesare tratează în mod independent fluxul de date F0 față de răspunsul tractului vocal. Determinarea cu acuratețe a F0 devine astfel un domeniu de studiu larg, iar metodele de determinare a acestei frecvențe sunt multiple. De exemplu, în cadrul codorului CELP (Code Excited Linear Prediction) utilizat în codarea GSM, posibilitatea estimării F0 folosind un set redus de impulsuri face ca această codare să fie extrem de eficientă.
Datorită caracterului cvasi-periodic al semnalelor sonore, cele mai simple metode de determinare a F0 din semnalul vocal se referă la estimarea acestei periodicități din forma de undă. Periodicitatea unui semnal este dată de intervalul de timp după care valorile eșantioanelor semnalului se repetă. Acest interval este denumit perioada semnalului ( ) și este egală cu . Cel mai simplu exemplu de semnal periodic este o funcție sinus. Spunem însă că semnalul vocal este cvasi-periodic deoarece chiar și în segmentele sonore, datorită fenomenelor de coarticulare (trecerea de la un fonem la altul și modificarea poziției tractului vocal în acest proces), periodicitatea semnalului nu este pură, existând oarecare diferențe între perioade consecutive ale sale[2]

2. Metoda

Funcția de autocorelație poate fi folosită pentru a determina dacă un semnal este periodic prin calcularea similarității semnalului cu versiuni întârziate ale sale. Dacă semnalul este periodic, similaritate la întârzierea va avea un maxim.[2]

```
def compute_autocorrelation(x): #functie generica autocorelatie
    result = np.correlate(x, x, mode='full')
    return result[int(result.size/2):]
```

Folosind funcțiile de autocorelație și AMDF am putut să detectăm valorile mult mai ușor decât din forma de undă a semnalului. Dar această detecție a fost realizată manual. Această metodă nu este eficientă atunci când dorim să calculăm frecvența fundamentală a mai multor semnale vocale. Astfel că, avem nevoie de o metodă automată de calcul a pornind de la maximele sau minimele funcțiilor prezentate

anterior. O valoare maximă locală poate fi detectată automat prin compararea valorii ei cu cele din imediata vecinătate.[2]

```
def peak_detection(sample, sampling_frequency, max_F0): #detectie maxime
    local_max = [sample[0]] #maxime locale
    local_max_ind = [0]    #indicii maximelor locale
    indices = [0] #construieste lista

    min_peak_distance = sampling_frequency//max_F0 #distanta minima intre varfuri (imp

    for i in range(1,len(sample)-1): #for de la 1 la nr. cadre-1
        if sample[i] > sample[i-1] and sample[i] > sample[i+1]: # maxim dintre vecini
            local_max_ind.append(i) #adauga incicele maximului la lista
            local_max.append(sample[i]) #adauga valoarea maximului

    i = 1
    while i < len(local_max): #while pana cand i ajunge la ultimul maxim local
        next_max = np.argmax(local_max[i:]) #argmax => maximul dintr-un array
        if local_max_ind[i+next_max] - local_max_ind[i-1] > min_peak_distance:
            indices.append(local_max_ind[next_max+i])
        i = i+next_max+1

    return indices #returneaza lista indici
```

Dacă încercăm să determinăm valoarea lui F0 ce se află sub 500Hz, aceasta înseamnă că restul informației din banda de frecvențe superioară poate fi ignorată. Astfel că putem filtra semnalul trece-jos și să îmbunătățim astfel performanțele algoritmului.

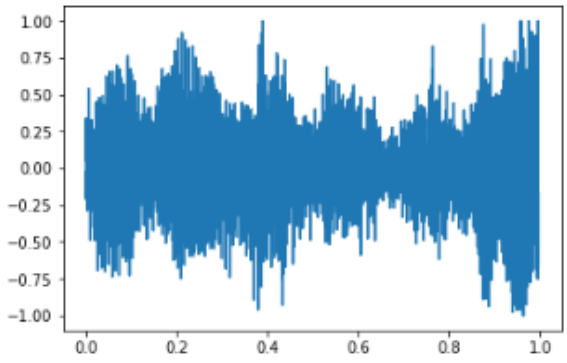
```
def _filter(this_frame): #filtrare trece jos
    cutoff = 499 #frecv de taiere
    # ordinul filtrului
    order = 5
    # frecvența Nyquist
    nyq = 0.5 * sample_rate
    # normalizăm frecvența de tăiere
    normal_cutoff = cutoff / nyq
    # calculăm coeficienții unui filtru Butterworth de tip trece-jos
    b, a = butter(order, normal_cutoff, btype='low', analog=False)
    # filtrăm datele
    filtered_data = lfilter(b, a, this_frame)
    return filtered_data
```

Am ales frecventele fundamentale ale notelor din gama majora conform site-ului Wikipedia, pentru a putea evalua calitatea vocii cantate prin compararea frecventei cantate cu cea a notei. Pentru inregistrarea de la microfon in timp real am folosit biblioteca pyaudio, instalata cu ajutorul comenzii “pipwin”, la randul ei instalata cu ajutorul comenzii “pip”. Aceasta biblioteca permite inregistrarea in format .wav la rularea codului. [3][4]

Numărul	Nota (solfegiu)	Notația cu litere	Frecvența (in Hertz)
1	Do	C	$440/(1.0595)^{9} \approx 261.6$
2	Do ♯	C♯	$440/(1.0595)^{8} \approx 277.2$
3	Re	D	$440/(1.0595)^{7} \approx 293.7$
4	Re ♯	D♯	$440/(1.0595)^{6} \approx 311.1$
5	Mi	E	$440/(1.0595)^{5} \approx 329.6$
6	Fa	F	$440/(1.0595)^{4} \approx 349.2$
7	Fa ♯	F♯	$440/(1.0595)^{3} \approx 370.0$
8	Sol	G	$440/(1.0595)^{2} \approx 392.0$
9	Sol ♯	G♯	$440/(1.0595) \approx 415.3$
10	La	A	440
11	La ♯	A♯	$440 \times 1.0595 \approx 466.2$
12	Si	B sau H	$440 \times (1.0595)^{2} \approx 493.9$

Dupa deschiderea fluxului de date, am apelat toate functiile mentionate anterior in vederea calcularii F0, frecventa fundamentala fiind ulterior comparata cu valorile frecventelor fiecarei note din gama pentru determinarea notei dorite.

```
Frecventa notei Do cantata de tine : 203 Hz
Frecventa notei Do cantata corect : 262 Hz
```



3. Discutii

Elemente ce pot fi imbunatatite in cadrul acestui sistem de analiza sunt adaugarea unui algoritm de detectie a zonelor sonore/nesonore pentru a analiza cu exactitate doar cadrul in care nota este redata cu acuratete si calitatea microfonului. In cadrul sistemului calitatea microfonului joaca un rol foarte important, calitatea sistemului de analiza depinzand intr-o mare masura de acest factor.

4. Concluzii si idei de continuare a proiectului

In concluzie, sistemul de analiza al calitatii vocii cantate bazat pe detectia frecventei fundamentale a notei cantate este o solutie interesanta si foarte utila, relativ simplu de implementat, care are mare success in cazul tunerelor de voce folosite de interpreti, acestea reusind sa realizeze aceasta detectie in timp real fara sa fie limitate la un anumit numar de esantioane.Este o metoda suficient de exacta si universala, frecventa fundamentala a notelor nefiind afectata de tonul vocii interpretilor.

6. References

[1] OSCAR MAYOR1, JORDI BONADA1, AND ALEX LOSCOS2, “PERFORMANCE ANALYSIS AND SCORING OF THE SINGING VOICE”, 2009

[2]Adriana Stan, “Laborator 4 - Pitch detection in the time domain”

[3]https://ro.wikipedia.org/wiki/Gam%C4%83_muzical%C4%83

[4] https://gist.github.com/mabdrabo/8678538