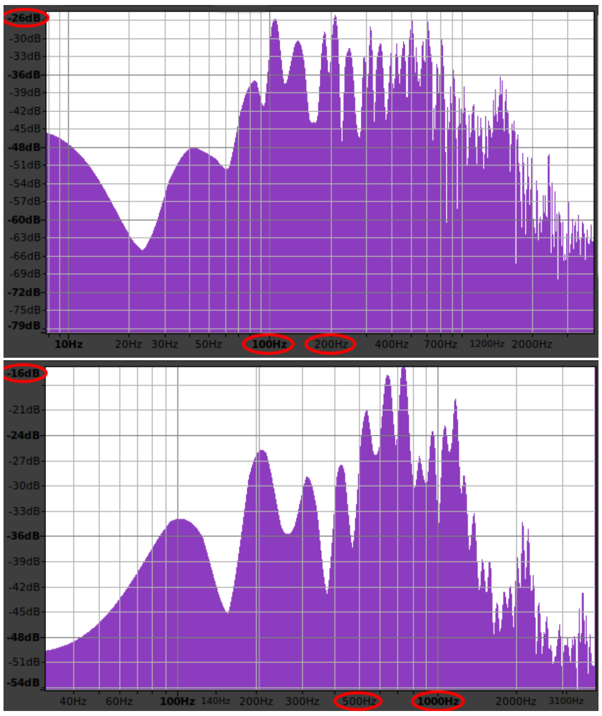
Final project report for VAD algorithm

The realization of the algorithm comes from a careful analysis of the input files through the Audacity software.

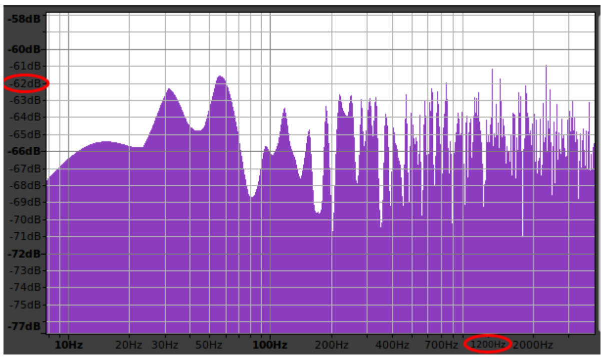
In particular, thanks to the "Show spectrum" function, I was able to analyze packets in the frequency domain.

I noticed that the frequency range was not well defined. In fact, the voice has frequencies ranging from 100 Hz to 1200 Hz depending on the voice timbre of the person (so it changes depending on the input file), but even in some intervals where only background noise is present , these are the dominant frequencies.

Some examples:



Examples of two different file entry packages

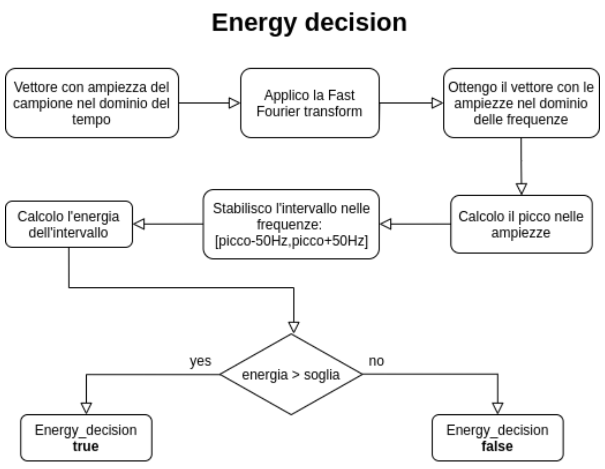


Example of a silence pack in a fileinput

I then noticed that the main difference between a packet with voice and one with silence lay in the signal strength.

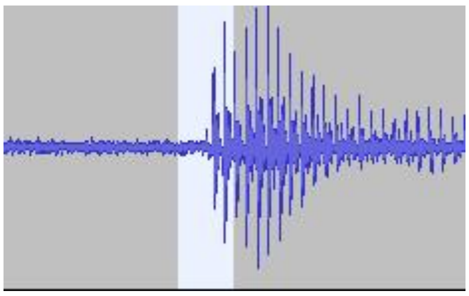
A first part of my algorithm aims to:

1. Transform the packet into consideration of the signal in the frequency domain.
2. Find its peak.
3. Establish a frequency range: [peak-50Hz, peak+50Hz].
4. Calculate the energy in the range.
5. Assess whether the energy is higher than a threshold established a priori or not.


Energy decision
Vettore con ampiezza del campione nel dominio del tempo
Applico la Fast Fourier transform
Ottengo il vettore con le ampiezze nel dominio delle frequenze
Calcolo il picco nelle ampiezze
Stabilisco l'intervallo nelle frequenze: [picco-50Hz,picco+50Hz]
Calcolo l'energia dell'intervallo
energia > soglia
yes
no
Energy_decision true
Energy_decision false

I decided to keep a high threshold ($6*10^7$) to make sure I properly evaluate a voice pack and not confuse it with high background noise.

This first algorithm I noticed was not efficient enough because in some situations (in particular at the beginning of speech and at the end) packets were discarded because they did not have sufficient energy, as in this example:

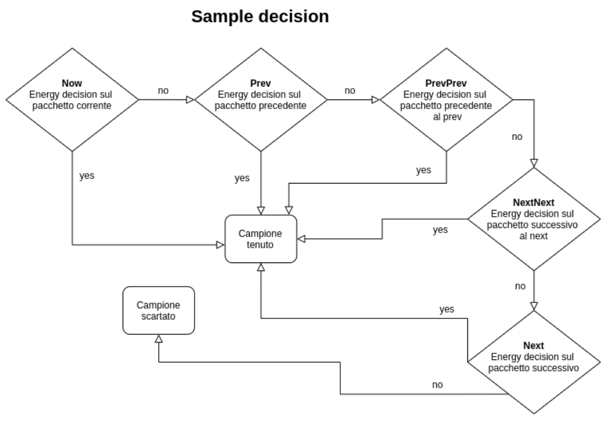


The voice was therefore too cut and in some cases not understandable.

I decided to integrate this first decision algorithm with an algorithm that was linked to the energy of the next and previous samples.

Since the overall encoding delay must remain below 50 ms, and each packet of 160 samples corresponds to 20ms, it is possible to evaluate the energy of only two successive samples.

Basically, even if the energy of a package is not enough, but it is one of the two previous or between the two future, the package is maintained. This mechanism allows to lengthen the time of speech and to avoid the effect of clipping.


Sample decision
Now
Energy decision sul pacchetto corrente
no
Prev
Energy decision sul pacchetto precedente
no
PrevPrev
Energy decision sul pacchetto precedente al prev
no
yes
yes
yes
NextNext
Energy decision sul pacchetto successivo al next
yes
Campione tenuto
no
Campione scartato
yes
Next
Energy decision sul pacchetto successivo
no

The results obtained were satisfactory.

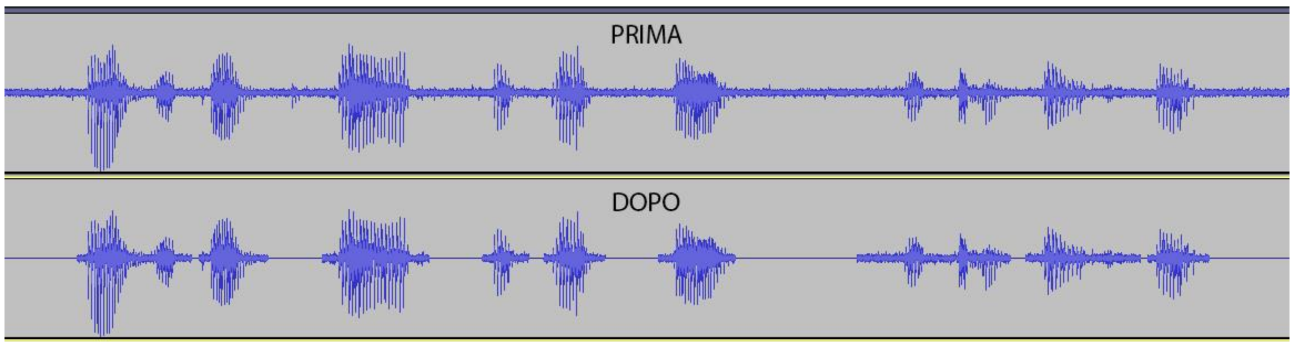
The algorithm does not handle the last packet (with less than 160 samples) if the input samples are not multiples of 160.

In order to use the FFT function in C++, only versions have been found that work if the input samples are powers of 2. For this reason, the energy decision is also made with 96 samples from the previous package (96+160 = 256). Using the previous package and not the next one you do not risk exceeding 50ms with the evaluation of the energy in the next and nextNext samples.

Only in the case of the first package are they considered samples of the next one.

When choosing the peak, the frequency 0 Hz is discarded, which is often predominant and does not allow to identify the nature of the package.

Inputaudio4:



Inputaudio5: