[[1]](#footnote-1)

Preparation of Papers for IEEE TRANSACTIONS and JOURNALS

First A. Author, Second B. Author, Jr., and Third C. Author, Member, IEEE

# Separación espectral

## Objetivo y resumen

Se tomó como objetivo agregar al programa principal una funcionalidad que permita obtener un archivo de audio únicamente con los instrumentos de percusión y otro con los instrumentos armónicos a partir de un único archivo de audio con una canción.

Dicho objetivo pudo cumplirse empleando el algoritmo obtenido de [1]. El algoritmo consiste en la aplicación de filtros sobre el espectrograma del audio original para crear otros dos espectrogramas a partir de los cuales se generan dos mascaras binarias utilizadas para obtener los audios deseados. La funcionalidad lograda puede utilizarse simplemente indicando el archivo de audio que se quiere separar en parte armónica o parte percusiva. Adicionalmente, cuenta con cuatro parámetros que pueden ser configurados por el usuario para ajustar los resultados obtenidos. Los parámetros con mayor efecto en los resultados fueron el factor de separación β y el tamaño de muestras a tomar para cada Fast Fourier Transform(FFT) del espectrograma.

## Sonidos percusivos y armónicos

Para comenzar a analizar como cumplir nuestro objetivo hace falta primero definir las características de los elementos percusivos de un audio, así como también de los armónicos.

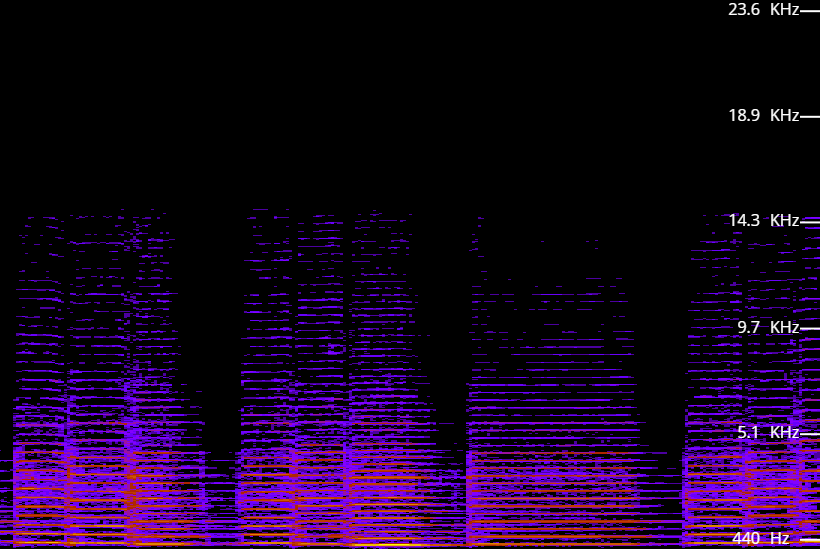


Figure 1. Espectrograma de un violín

Como puede verse en ‘Figure 1’ el espectro de un violín en el tiempo forma estructuras horizontales en el tiempo, lo que quiere decir que su espectrograma es continuo en el tiempo pero no en frecuencia.

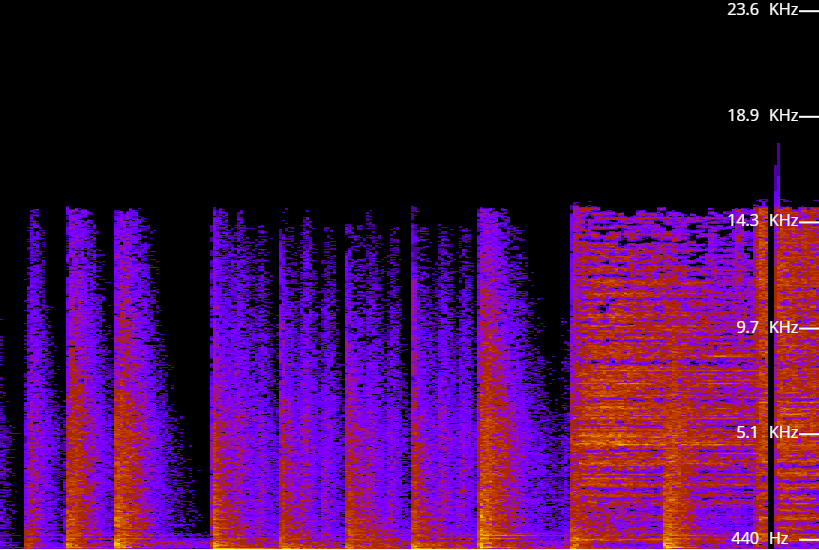
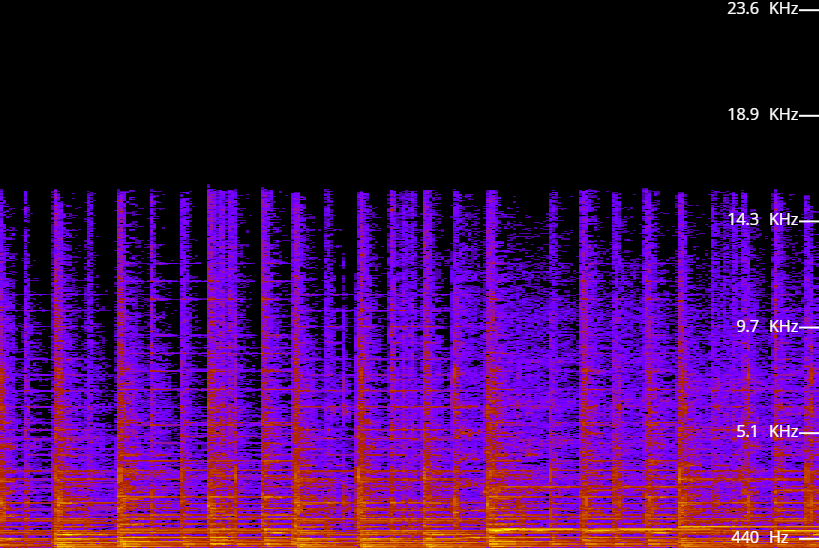


Figure 2. Espectrograma de un solo de batería

Por otro lado, como se puede ver en ‘Figure 2’, un instrumento percusivo como lo es una batería presenta estructuras verticales en su espectrograma, que significa que presenta continuidad en frecuencia pero no en el tiempo. Esto se debe a que los instrumentos percusivos tienen una duración muy corta en el tiempo lo que significa que se extienden en el dominio de la frecuencia. Esta es la razón por la que los instrumentos percusivos se denominan también instrumentos inarmónicos debido a que como contienen una banda muy amplia de frecuencias no es posible caracterizarlos por las frecuencias o armónicos que poseen.



Espectrograma original del audio

|  |  |  |
| --- | --- | --- |
| β | Armónico | Percusivo |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |

Figure 3. Espectrogramas obtenidos utilizando N=1024, Lp = La = 15

## Algoritmo

### Calculo de espectrograma y filtrado

En primer lugar, se debe obtener el espectrograma del vector con las muestras de la canción. El número de muestras a utilizar para cada FFT de la Short Time Fourier Transform(STFT) es un parámetro importante del algoritmo y se denotara con el símbolo N.

La elección de los demás parámetros de la STFT como el tipo de ventana y el overlap no tienen un efecto apreciable en el resultado final, por lo que simplemente se utiliza la ventana de haming con un overlap del 50%. Una vez realizada la STFT del vector con muestras se toma el modulo del resultado y se obtiene el espectrograma del audio original.

El paso siguiente es la aplicación del filtro armónico y del filtro percusivo al espectrograma calculado previamente. Al aplicar dichos filtros se obtienen dos nuevos espectrogramas. Los filtros utilizados se definen de la siguiente manera:

SP[t, k]= Median( S[t, k - Lp/2], … , S[t, k + Lp/2] )

Sa [t, k ]= Median( S[t– La/2, k], … , S[t+ La/2, k] )

Donde t es el índice para los bloques de tiempo y k es el índice para los bins de frecuencia. La función Median devuelve la mediana del conjunto de valores recibidos. Las variables Lp y La  son las longitudes de los filtros y son dos de los cuatro parámetros del algoritmo.

Los valores de Lp y La no afectan drásticamente los resultados finales mientras no se tomen valores extremos. En [1] se sugiere tomar Lp equivalente a un ancho de banda de 500Hz y La equivalente a 200ms. Dicha selección depende de la frecuencia de muestreo del audio utilizado pero por lo general ronda en valores de Lp y La entre 10 y 15.

### Aplicación de mascaras

Una vez que se tienen los espectrogramas filtrados SP y Sa, se generan dos mascaras binarias definidas de la siguiente manera:

Ma[t,k] = ( Sa [t, k ] / SP[t, k]) > β

Mp[t,k] = ( Sp [t, k ] / Sa[t, k]) ≥ β

El símbolo β es el factor de separación y es el último parámetro del algoritmo de separación. Multiplicando una máscara con la STFT del audio original se obtiene una nueva STFT correspondiente a la parte percusiva o armónica del audio dependiendo de la máscara utilizada.

Finalmente, para obtener los audios deseados solo se debe aplicar la Inverse Short Time Fourier Transform(ISTFT) a cada uno de los resultados obtenidos luego de aplicar las máscaras a la STFT del audio original.

## Implementación y resultados[[2]](#footnote-2)

El algoritmo explicado previamente se implementó en Python como una funcionalidad más dentro de un programa que realiza varias funciones relacionados con archivos de audio.

Se puede observar de “Figure 3” los espectrogramas de la canción utilizada así como de los resultados obtenidos de la separación. Se puede ver que mientras mayor el valor de β se distinguen mejor las estructuras horizontales en el espectrograma armónico, mientras que en el percusivo se distinguen mejor las estructuras verticales. Esto se debe a que el criterio de separación es más estricto por lo que hay una menor fuga de elementos percusivos al espectrograma armónico y viceversa.

Asimismo, a mayor β se puede ver que se pierde una energía notable en el audio resultante, esto tiene sentido ya que al tener una separación más estricta se incrementa la cantidad de elementos que no son ni armónicos ni percusivos y son descartados.

Referencias

1. Driedger, M. Muller and S. Disch, *Extending harmonic-percussive separation of audio signals*. Erlangen, Germany, 2014.

1. [↑](#footnote-ref-1)
2. Todos los audios utilizados y generados se encuentran en https://tinyurl.com/ve5phd5 [↑](#footnote-ref-2)