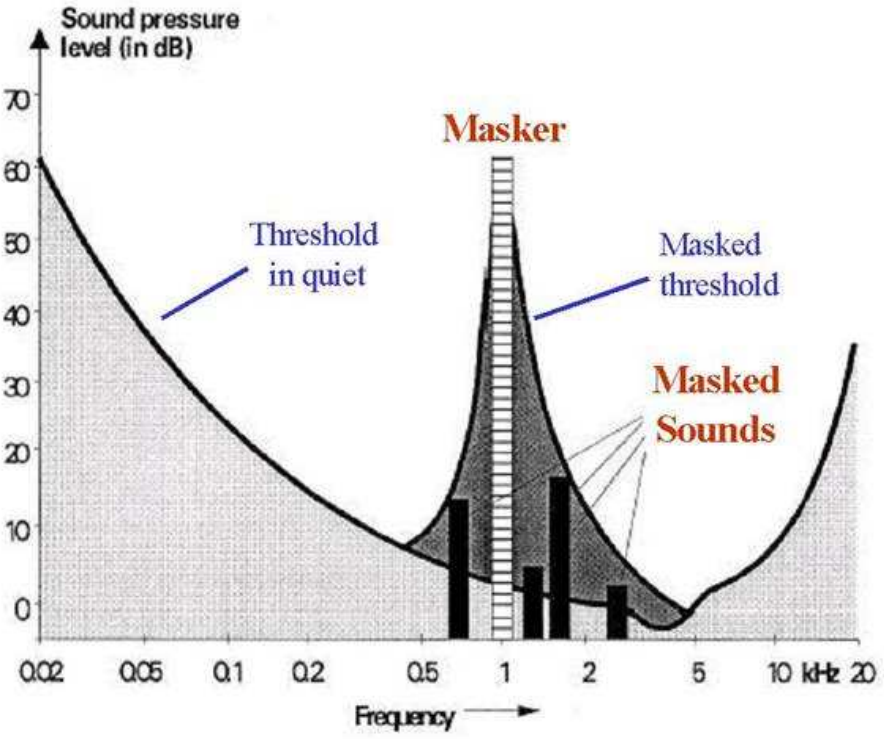
Modelo psicoacústico de enmascaramiento para implantes cocleares



Universidad Politécnica de Cartagena

Mayo 2006

LUIS ARMADA DORDA

CONTENIDOS

1 INTRODUCCIÓN

En el laboratorio de Tecnología de la Información de la Universidad de Hannover se diseñan nuevas estrategias de procesamiento de señal. Y en el Centro de Audición (Hörzentrum) de la Universidad de Medicina de Hannover, los nuevos algoritmos se implementan y prueban en pacientes con implantes cocleares.

Un implante coclear es un aparato electrónico usado para recuperar parte del sentido del oído a personas con profunda sordera. Algunos individuos con implantes cocleares pueden ahora comunicarse sin leer los labios o mediante signos, y algunos incluso por teléfono.

Generalmente, estos aparatos constan de un micrófono, un procesador de voz, un transmisor, un receptor y un array de electrodos colocado en el interior de la cóclea. El procesador de voz es responsable de descomponer la señal de audio entrante en diferentes bandas frecuenciales o canales y decidir el modelo de estimulación más adecuado para los electrodos.

Cuando se usan estrategias de procesamiento de señal como Continuous Interleaved Sampling (CIS) o Advanced Combinational encoder (ACE), los electrodos cercanos a la cóclea representan la información de altas frecuencias, mientras que los cercanos al otro extremo transmiten la información de baja frecuencia. Las estrategias de codificación de voz juegan un papel muy importante a la hora de maximizar el potencial comunicativo de los usuarios, y han sido desarrolladas distintas estrategias de procesamiento de voz en las últimas dos décadas para imitar patrones de excitación en el interior de la cóclea tan naturalmente como se pueda.

Estrategias “NofM” tales como la ACE fueron desarrolladas en los noventa. Estas estrategias descomponen las señales de voz en M sub-bandas y obtienen la información de la envolvente de cada banda de la señal. Se seleccionan las N bandas con las mayores amplitudes para la estimulación (N de M). El objetivo básico aquí es incrementar la resolución temporal rechazando los componentes frecuenciales menos significativos, concentrándose en las características más representativas. Estas estrategias han demostrado una mejora significativa o al menos una preferencia de uso respecto a las estrategias convencionales como la CIS.

De todas formas, el reconocimiento de voz en implantes cocleares en condiciones de ruido, y para algunos individuos, incluso sin ruido, sigue siendo todo un reto. Para mejorar aún más la percepción de la voz en implantes cocleares, se diseñó una nueva estrategia llamada Psychoacoustic Advanced Combinational Encoder (PACE). Esta estrategia modifica el algoritmo de selección de bandas de la estrategia de codificación de voz ACE. Esta nueva estrategia describe por tanto un nuevo método para seleccionar las N bandas en estrategias “NofM”.

Como ha sido mencionado arriba, las estrategias “NofM” tradicionales seleccionan las N bandas con las mayores amplitudes de entre las M salidas o bandas del banco de filtros. Con el nuevo modelo las N bandas son seleccionadas usando un modelo psicoacústico de enmascaramiento. La estrategia PACE ha sido probada con pacientes y ha obtenido mejoras significativas bajo algunas condiciones respecto a la estrategia comercial ACE.

Sin embargo, el modelo psicoacústico implementado en la estrategia PACE sólo incorpora efectos de enmascaramiento frecuencial que tienen lugar en la membrana basilar. El modelo psicoacústico implementado en la estrategia PACE no tiene en cuenta los efectos de enmascaramiento en el dominio temporal del sistema auditivo humano. Además, los efectos de enmascaramiento no están sujetos a las características de las señales de audio, ya que no se distinguen componentes tonales o no tonales a la hora de calcular los umbrales de enmascaramiento.

2 OBJETIVO Y FASES DEL PROYECTO

El objetivo del proyecto consiste en mejorar la percepción de la voz en los pacientes con implantes cocleares. Para alcanzar este objetivo se debería diseñar una versión mejorada del modelo psicoacústico implementado en la estrategia PACE.

Fases del Proyecto

La mejora se desarrollará en las siguientes fases:

1) Incluir el modelo de los efectos de enmascaramiento temporal en una versión en C++ de la estrategia PACE.

2) Después de la descomposición en el banco de filtros, se debería detectar si cada banda del banco de filtros puede considerarse componente tonal o componte de ruido. Dependiendo del comportamiento de cada banda de frecuencia, será aplicada una función de ensanchamiento específica para calcular el umbral de enmascaramiento individual.

3) Verificación del algoritmo.

4) El nuevo modelo psicoacústico debería ser probado en pacientes de implantes cocleares usando un hardware especialmente diseñado para ello. El nuevo algoritmo debería ser probado al menos con 2 pacientes.

3 FUNDAMENTOS

En esta parte del proyecto serán descritos tanto el sistema auditivo humano como el funcionamiento de los implantes cocleares. Primero veremos las partes del sistema auditivo humano, cuya comprensión es necesaria para entender el funcionamiento de los implantes cocleares. Después explicaremos cómo funciona un implante coclear. Y por último, serán descritas algunas de las estrategias de procesado de señal que se utilizan hoy en día en este tipo de implantes.

3.1 Sistema Auditivo Humano

3.1.1 Partes del Sistema Auditivo Humano

En la figura 1 podemos ver las partes de las qué consta el oído humano: Oído externo, oído medio y oído interno.

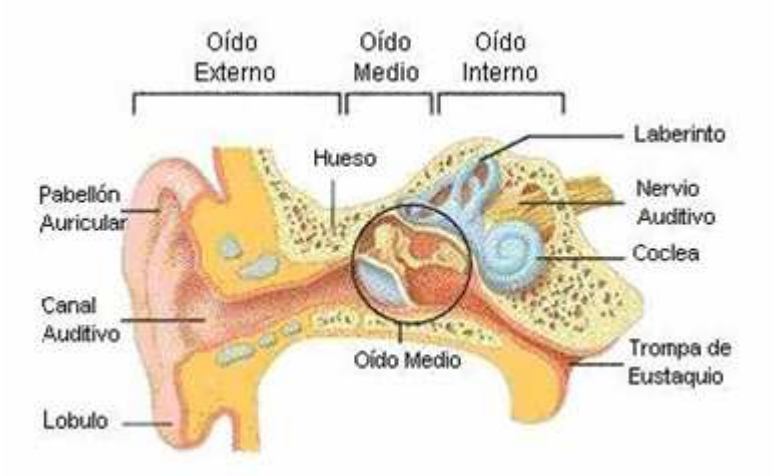


Figura 1. Oído humano

El oído externo es la parte del sistema auditivo que recoge las ondas de presión entrantes (energía acústica) y las transmite hasta la cóclea, órgano del oído interno encargado de la audición.

La energía acústica viaja a través del canal auditivo hacia el oído medio causando la vibración del tímpano, el cual a su vez, produce la vibración de tres pequeños huesos (martillo, yunque y estribo). La energía acústica es transformada así en vibraciones mecánicas de estos tres huesecillos. A continuación, estas vibraciones mecánicas llegan al oído interno.

En el oído interno podemos distinguir dos partes bien diferenciadas: los canales semicirculares (o laberinto) y un órgano llamado cóclea. Los implantes cocleares reciben este nombre porque están basados, precisamente, en el funcionamiento de la cóclea.

A continuación vamos a pasar a describir este órgano y cómo funciona.
Vemos, en la figura 1, que la cóclea tiene una forma que recuerda a la concha de un caracol. En su interior está la membrana basilar, que juega un papel muy importante en el fenómeno de la audición, ya que es, en última instancia, el órgano que envía la información al cerebro a través del nervio auditivo.

Vemos en la siguiente figura, todos los elementos interrelacionados con la cóclea.

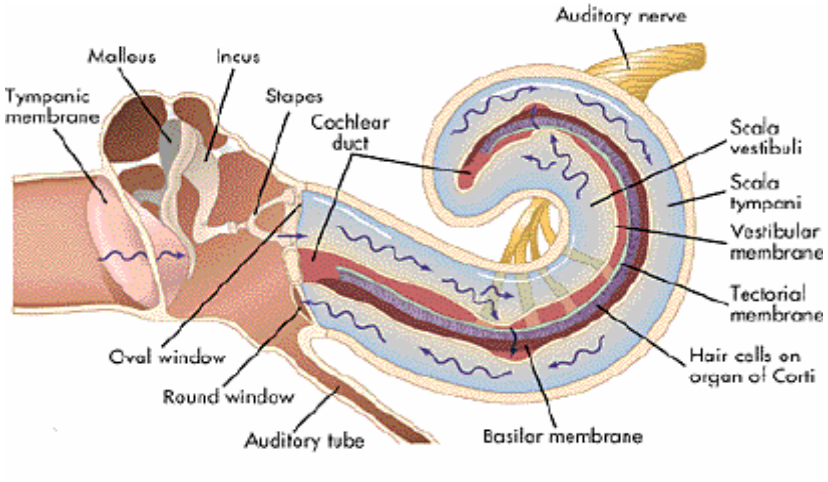


Figura 2. Cóclea

El interior de la cóclea está relleno de un líquido linfático (azul en el dibujo). Inmersa en este líquido se encuentra la membrana basilar, que recorre longitudinalmente la cóclea y sobre la cual se asientan los filamentos terminales del nervio auditivo.

La ventana oval está unida al estribo y recibe de él sus vibraciones. Estas vibraciones aumentan y disminuyen la presión del líquido contenido encima de la membrana basilar, dando lugar a la aparición de una onda que se desplaza de izquierda a derecha a lo largo de la membrana.
Dependiendo de la frecuencia de los sonidos que entran a nuestro oído, un determinado punto u otro de la membrana basilar, absorberá la energía de la onda producida en la membrana debida a las vibraciones del estribo.
Esto significa que cada frecuencia corresponde con un determinado punto en la membrana basilar, pudiendo así el cerebro interpretar además de la intensidad del sonido, su frecuencia. Así, el oído interno se comporta como un analizador de frecuencias.

Para ilustrar este concepto, fijémonos en la figura que vemos a continuación.

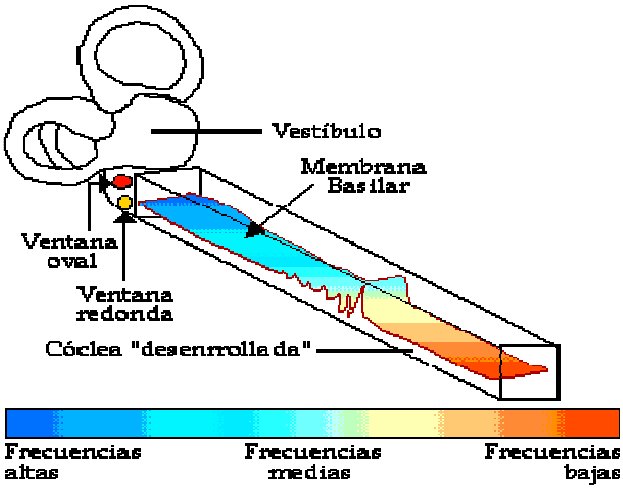


Figura 3. Membrana basilar

Observamos el interior de una cóclea “desenrollada”. Vemos de colores la membrana basilar estirada, correspondiendo, los sonidos con frecuencias altas al extremo de la membrana más pegado a la ventana oval y las bajas frecuencias al extremo más alejado. Podríamos decir que la cóclea se comporta como un sistema de mapeado frecuencia-posición.

Finalmente, las señales producidas en la cóclea generan, a través de la membrana basilar, la excitación de determinadas terminaciones nerviosas del nervio auditivo, causadas éstas por reacciones químicas que se traducen en impulsos eléctricos transmitidos al cerebro. El cerebro interpretará la frecuencia del sonido simplemente identificando qué terminaciones nerviosas fueron excitadas.

Pero a veces la membrana basilar deja de funcionar correctamente o bien nunca ha funcionado. La causa de la sordera es el fallo en la conversión del movimiento mecánico producido en la membrana basilar en impulsos eléctricos que tienen lugar en las células ciliadas internas, así como la pérdida de funcionalidad de las células ciliadas externas. Diversas causas producen este fallo (tumores, meningitis, accidentes, etc). En algunos de estos casos, la sordera puede ser corregida en parte, mediante el uso de uno de los mencionados implantes cocleares.
Como veremos más adelante, los implantes cocleares se basan, entre otros aspectos, en el principio arriba mencionado de mapeado frecuencia-posición.

3.1.2 Efectos de Enmascaramiento

El enmascaramiento cae dentro de los estudios psicoacústicos que buscan determinar de qué manera la presencia de un sonido afecta a la percepción de otro sonido. Hablamos de enmascaramiento cuando un sonido impide la percepción de otro, es decir, lo enmascara. Se produce una modificación del umbral de audibilidad del oyente.

Podemos definir el umbral de audibilidad como “el mínimo nivel de presión sonora a partir del cual somos capaces de oír, en condiciones no ruidosas”. Varía con la frecuencia y está representado en la siguiente figura:

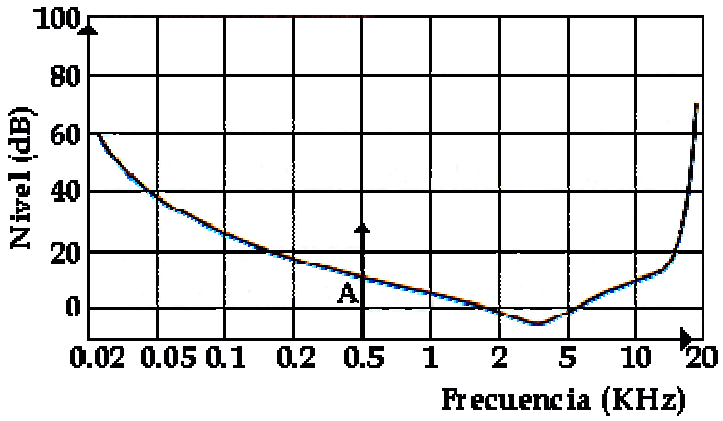


Figura 4. Umbral de audibilidad

Sonidos cuyo nivel de presión sonora esté por encima de este umbral, como la señal A, serán percibidos por el oído humano; y sonidos por debajo del umbral no se percibirán. Producirse enmascaramiento equivale a decir que el umbral de audibilidad ha cambiado.

Existen dos tipos básicos de enmascaramiento: el enmascaramiento simultáneo o frecuencial, y el enmascaramiento no simultáneo o temporal.

3.1.2.1 Enmascaramiento Frecuencial o Simultáneo

En el caso de dos señales de frecuencias relativamente cercanas, la señal más fuerte hace subir el umbral de audición en sus proximidades, cuyo efecto es disminuir la sensibilidad del oído alrededor de estas frecuencias. La subida o variación del umbral de audición es lo que denominamos umbral de enmascaramiento.

La figura 5 representa este caso, donde la señal A, antes audible, es ahora enmascarada por la cercana señal B, más potente que A, al generar B el mencionado umbral de enmascaramiento. Este efecto recibe el nombre de enmascaramiento frecuencial.

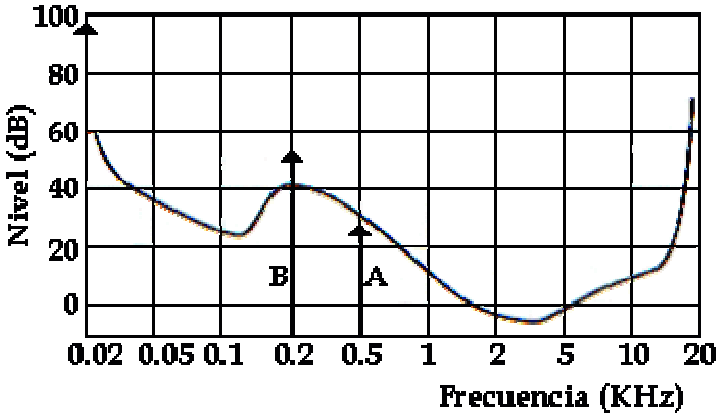


Figura 5. Enmascaramiento frecuencial

El fenómeno de enmascaramiento frecuencial se divide en un pre-enmascaramiento y un post-enmascaramiento frecuencial, que se da en frecuencias inferiores y superiores a la de la señal enmascaradora, respectivamente. Más adelante veremos detenidamente los parámetros que modelan el umbral de enmascaramiento. Ahora simplemente vamos a esbozar cómo cambia el umbral de enmascaramiento dependiendo de la naturaleza de la señal enmascaradora.

Estudios actuales han demostrado que existen grandes diferencias entre la forma de enmascarar que producen sonidos de distinta naturaleza. Esto es, el umbral de enmascaramiento tendrá distinta variación dependiendo de si la señal que enmascara proviene de un ruido o proviene, por ejemplo, de la voz de una persona.

Distinguiremos los dos extremos entre los que puede variar el umbral de enmascaramiento:

- Por un lado, cuando la **señal** enmascaradora es de naturaleza **tonal**, el umbral de enmascaramiento generado será el menos enmascarador, ya que la diferencia entre la amplitud del tono enmascarador y el nivel máximo del umbral de enmascaramiento es muy grande.

- En cambio, si la señal enmascaradora se trata de una **señal** de naturaleza **ruidosa** (componentes no tonales), el efecto enmascarador será bastante más pronunciado. En este caso, la diferencia entre la amplitud del ruido enmascarador y el nivel máximo del umbral de enmascaramiento es menor.

A modo orientativo vemos en la siguiente figura la citada diferencia de amplitud:

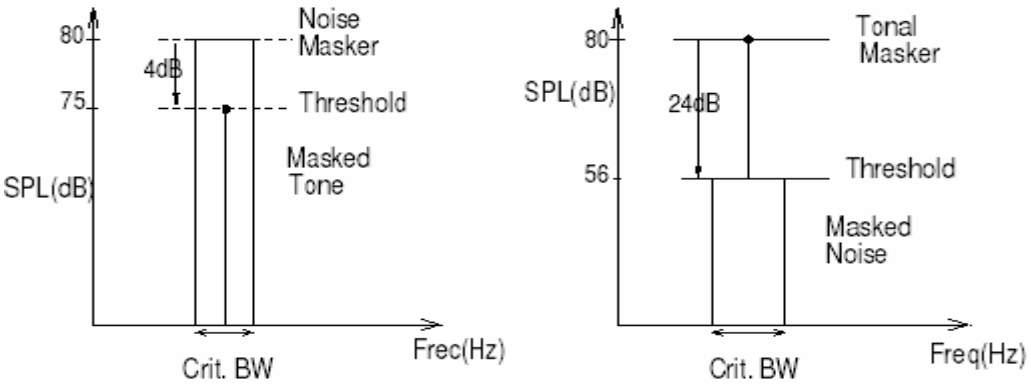


Figura 6. Variación del umbral de enmascaramiento

En la figura de la izquierda tenemos un ruido como señal enmascaradora. Vemos como la diferencia hasta donde comienza el umbral de enmascaramiento es de apenas 4 dB. En cambio, a la derecha observamos como esta diferencia asciende a 24 dB, ya que en este caso tenemos como señal enmascaradora a un tono. Así, señales de la misma amplitud, podrán ser enmascaradas o no, dependiendo de si la señal enmascaradora tiene componentes tonales o no.

3.1.2.2 Enmascaramiento Temporal o no Simultáneo

Al igual que en el fenómeno de enmascaramiento frecuencial, las señales producen también un efecto de enmascaramiento temporal, esto es, señales que se dan muy próximas en el tiempo, podrían ocultarse entre ellas, haciéndose así inapreciables por el oído humano.

A continuación vemos como se modela este tipo de enmascaramiento:

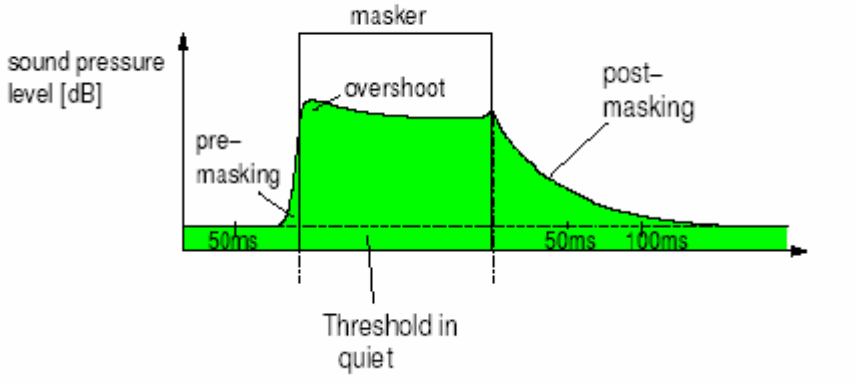


Figura 7. Enmascaramiento temporal

Se presenta cuando un tono suave está muy cercano en el dominio del tiempo (unos cuantos milisegundos) a un tono fuerte. Si se está escuchando un tono suave y aparece un tono fuerte, el suave será enmascarado por el fuerte, de hecho, antes de que el tono fuerte aparezca (pre-enmascaramiento).

Posteriormente, cuando el tono fuerte desaparece, el oído necesita un pequeño intervalo de tiempo (entre 50 y 300 ms) para poder seguir escuchando el tono suave (post-enmascaramiento). El efecto de post-enmascaramiento se puede entender de una forma intuitiva. Sin embargo, el pre-enmascaramiento sugiere que un tono será enmascarado por otro tono, antes de que el tono enmascarador realmente aparezca, atentando contra el buen juicio de cualquier oyente.

Para este fenómeno, se han presentado dos posibles explicaciones**:**

1) El cerebro integra el sonido sobre un período de tiempo, y procesa la información por ráfagas en la corteza auditiva, o;

2) Simplemente, el cerebro procesa los sonidos fuertes más rápido que los sonidos suaves.

Se puede demostrar que el efecto de pre-enmascaramiento temporal tiene una duración considerablemente menor que la del post-enmascaramiento (aproximadamente 30 ms).

En un sonido complejo cualquiera, se pueden presentar ambos tipos de enmascaramiento. Superponiendo ambos enmascaramientos en una sola gráfica que presente tres ejes, se puede ver una curva bajo la cual están todos los sonidos que no pueden ser percibidos por nuestro sistema auditivo.

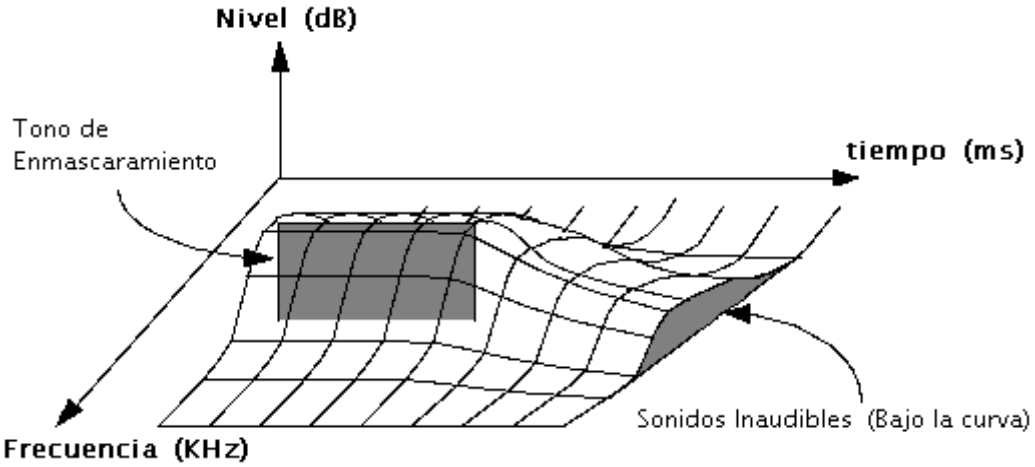


Figura 8. Enmascaramiento total

3.2 Implantes cocleares

En la Figura 9 se muestran los elementos que componen un implante coclear.

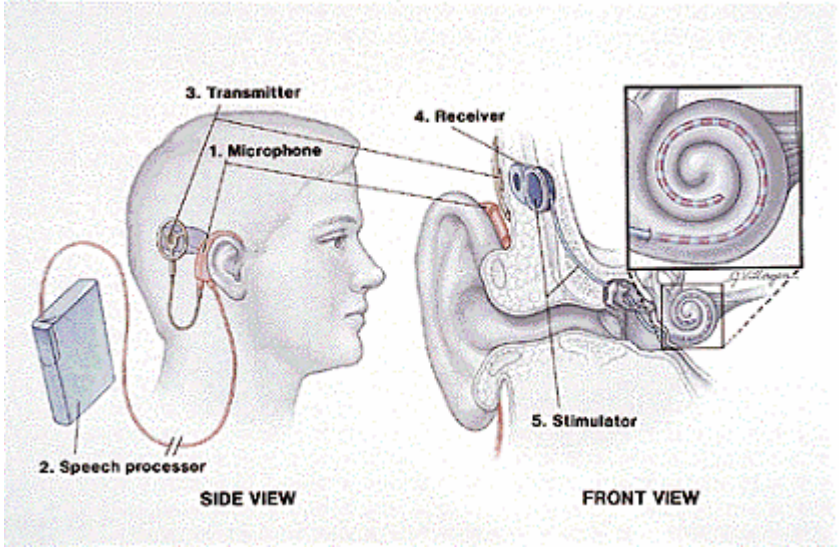


Figura 9. Implante coclear

El implante consta básicamente de una parte exterior o parte visible, formada por un micrófono, un procesador de voz y un transmisor; y de una parte interna formada por un receptor/estimulador, un cable y un array de electrodos implantados a lo largo de toda la cóclea, como observamos en la parte derecha de la imagen.

Vamos a explicar a continuación el funcionamiento de los implantes cocleares, desde que el sonido alcanza el micrófono, hasta que llega la información al nervio auditivo y es interpretada por el cerebro, produciendo la capacidad de oír en personas con sordera.

Empecemos por el micrófono. Éste va colocado detrás de oreja y recoge el sonido entrante que llegaría al oído, por ejemplo voz, mandándolo directamente al procesador. Aquí, mediante los algoritmos de procesado de señal adecuados, se selecciona la información relevante que más tarde se enviará al cerebro.
Más adelante veremos cuáles son estos algoritmos, cuál es la información relevante y cómo se envía ésta al nervio auditivo.

Una vez procesada y seleccionada la información por el procesador, ésta se manda al transmisor, que como su nombre indica, transmite la información al receptor, que se encuentra ya en el interior de la cabeza. El receptor, en base a la información recibida, estimula eléctricamente los electrodos (array de electrodos) situados en el interior de la cóclea.

La estimulación de los electrodos elegidos excita las terminaciones nerviosas de sus entornos, y a través del nervio auditivo, se traslada esta información al cerebro, donde es interpretada.

Resumiendo, el micrófono recoge las ondas de sonido, se procesan, y se codifican de tal forma que, dependiendo de la naturaleza del sonido (intensidad y frecuencia), se seleccionan unos u otros electrodos, que son estimulados eléctricamente cada uno con su respectiva intensidad. Este estímulo llega al cerebro mediante el nervio auditivo, produciéndose la sensación de oír en el paciente.

Llegados a este punto, cabe preguntarse sobre las limitaciones que presentan los implantes cocleares. Recordar, antes de nada, que la membrana basilar situada en el interior de la cóclea, era la responsable de determinar a que frecuencia o frecuencias correspondían los sonidos que iban llegando al oído.

Tendremos las siguientes limitaciones:

1) Por un lado, una baja resolución frecuencial. Esto es debido a que los implantes cocleares presentan un número finito (normalmente 22) de electrodos implantados a lo largo de toda la cóclea. Con esto estamos sustituyendo miles de terminaciones nerviosas con un máximo de 22 electrodos.

Figura 10. Electrodos en cóclea

Además, no todos los electrodos pueden ser estimulados simultáneamente, debido en parte, a la otra gran limitación;

2) Baja resolución temporal. Esta otra gran restricción se refiere a que, por limitaciones intrínsecas del implante, no se pueden estimular a la vez todos los electrodos seleccionados, sino que se debe dejar un tiempo entre la estimulación de un electrodo y el siguiente.

Por esto no podemos seleccionar todos los electrodos, ya que no podríamos llevar a cabo fielmente todo el proceso en tiempo real. Visto de otro modo, cuántos más electrodos seleccionemos, menor es el tiempo de espera entre la estimulación de dos electrodos consecutivos. Hasta que llega un momento en que este tiempo es tan pequeño que hace que el implante no funcione correctamente.

Como consecuencia, surge la necesidad de crear estrategias de procesamiento de señal que hagan frente a estas limitaciones. Vamos a ver a continuación algunas de estas técnicas.

3.3 Estrategias de procesamiento de señal para implantes cocleares

Vamos a describir algunas de estas técnicas, lo que nos ayudará a comprender mejor la función del procesador de voz, que vimos cuando describíamos los elementos que formaban un implante coclear. Prestaremos especial atención a la segunda de estas técnicas, en la cual se basa este proyecto.

1) La primera de estas técnicas es la denominada **ACE** (Advanced Combinational Encoder), cuyo esquema vemos en la siguiente figura:

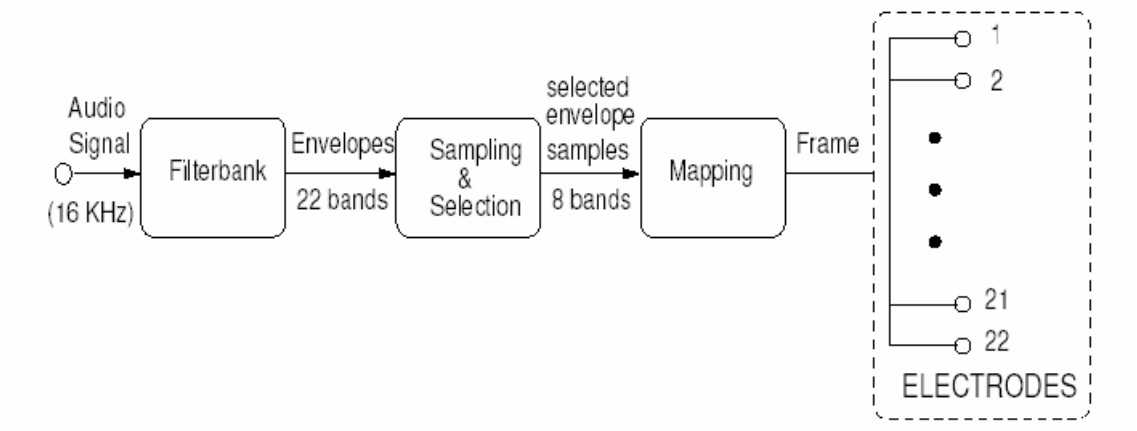


Figura 11. Estrategia ACE

Se trabaja con la señal de audio ya convertida a digital con una frecuencia de muestreo de, por ejemplo, 16 KHz. A continuación se pasa por un banco de filtros que divide el espectro de la señal en **22 bandas frecuenciales**. Después llegamos al bloque de Sampling and Selection, donde **se seleccionan las 8 bandas** frecuenciales **con** una **mayor amplitud**. Dado que cada banda

frecuencial esta asociada directamente con un electrodo (en nuestro caso, la banda uno con el electrodo veintidós, la segunda banda con el electrodo veintiuno, y así sucesivamente, tenemos ya identificados que electrodos se van a estimular. De esto se encarga el bloque de Mapping, además de asignar valores adecuados de intensidad de estimulación (amplitudes) a cada una de las estimulaciones de los 8 electrodos seleccionados.

Esta estrategia se usa en muchos de los implantes cocleares de hoy en día.

2) Estrategia **PACE** (Psychoacoustic Advanced Combinational Encoder).

Diagrama de bloques:

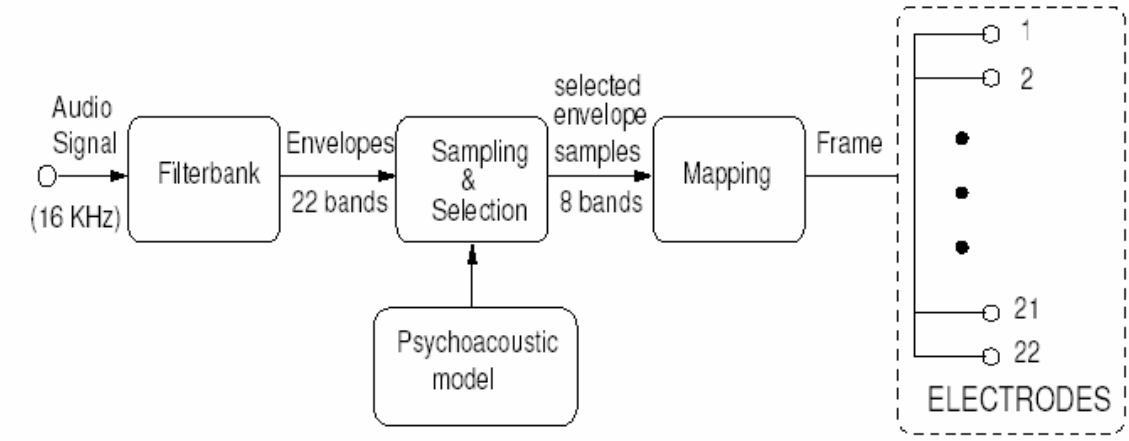


Figura 12. Estrategia PACE

De la misma forma, tenemos una señal digital muestreada a 16 KHz que hacemos pasar por un banco de filtros, dividiendo el espectro en **22 bandas**. Igualmente tenemos el bloque de Sampling and Selection, pero esta vez, las bandas seleccionadas vienen determinadas por un modelo psicoacústico, que explicaremos detalladamente más adelante.
Una vez seleccionadas **las 8 bandas** consideradas como las **más relevantes**, **de acuerdo con el modelo psicoacústico**, se procede a estimular los 8 electrodos con los valores adecuados de intensidad, mediante el bloque Mapping, al igual que en la estrategia ACE.

Para terminar, veremos una breve comparación entre las dos estrategias que hemos visto.

Podemos ver en la figura 13 como las bandas seleccionadas por uno u otro algoritmo son muy diferentes, eligiendo la estrategia PACE las bandas frecuenciales más relevantes, ya que se han elegido de acuerdo con el modelo psicoacústico.
Además, también vemos que las bandas seleccionadas por la PACE tienen una menor amplitud, lo que hace que se **ahorre energía** a la hora de la estimulación. Este ahorro de energía es importante si pensamos que los implantes cocleares funcionan con baterías.
Y además, las bandas en la PACE están más separadas, lo que produce que haya una **menor interacción** a la hora de estimular los electrodos.

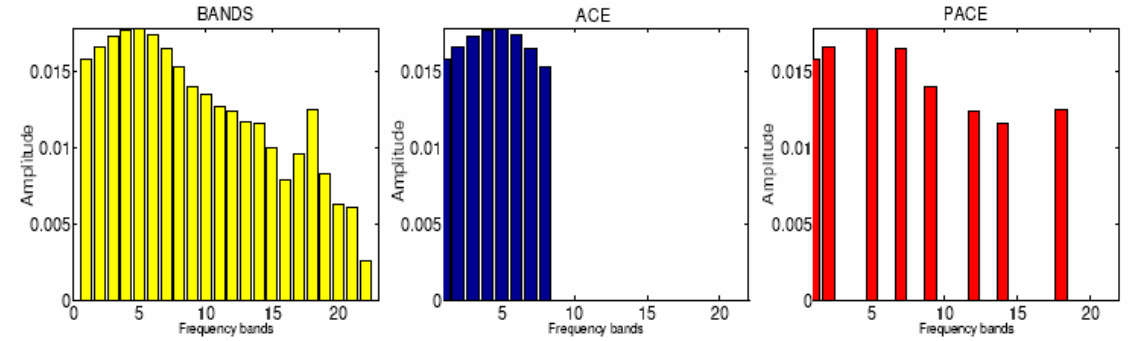


Figura 13. PACE vs ACE

El ahorro de energía y la menor interacción entre bandas, así como tests de inteligibilidad que han demostrado que PACE da mejor resultado que ACE, convierten a la PACE en mejor estrategia.

4 MODELO PSICOACÚSTICO EN ESTRATEGIAS “NdeM” PARA IMPLANTES COCLEARES

Implementar un modelo psicoacústico para seleccionar las bandas más relevantes consiste en tener en cuenta ciertos aspectos intrínsecos del sonido. En el caso de la PACE, el aspecto que se tiene en cuenta se llama “enmascaramiento frecuencial”.

Básicamente consiste en lo siguiente: se sabe por numerosos estudios que, si se tienen señales próximas en la frecuencia, aquella con una mayor amplitud puede enmascarar a las señales adyacentes de menor amplitud, causando que las señales enmascaradas de menor amplitud no sean captadas por el oído humano.
Así, lo que intenta hacer el modelo psicoacústico es tener en cuenta este efecto a la hora de seleccionar las 8 bandas más relevantes de entre las 22 posibles, para conseguir representar lo más fielmente posible lo que nuestro oído captaría.

Este fenómeno se ve ilustrado en la figura 8, dónde vemos además, que está representado otro efecto importante a la hora de implementar este modelo, denominado “threshold in quiet” (“umbral de audibilidad”), que podemos definir, como el mínimo nivel de presión sonora a partir del cual somos capaces de oír, en condiciones no ruidosas. Como vemos en esta figura, este umbral depende de la frecuencia.

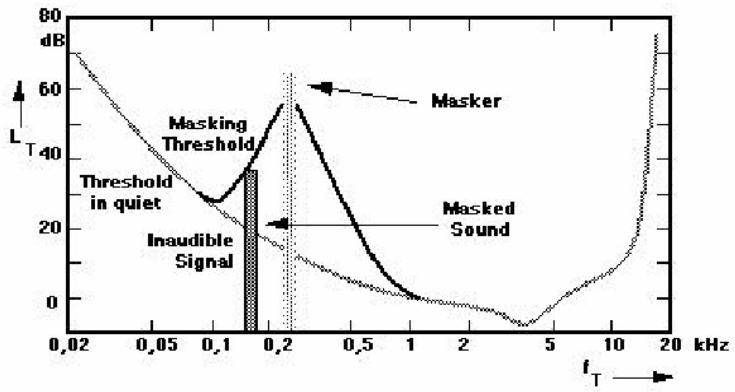


Figura 14. Modelo Psicoacústico

Observamos como la señal de mayor amplitud (masker) genera un umbral de enmascaramiento (masking threshold) que sumado al threshold in quiet generan un nuevo umbral de enmascaramiento en el que ninguna señal que se halle dentro de él podrá ser oída (en la figura, inaudible signal).

Teniendo en cuenta lo anterior, el modelo psicoacústico se desarrolla en el **algoritmo de selección** de bandas, que describimos a continuación.

Vemos en la gráfica inferior izquierda que tenemos el espectro ya dividido en veintidós bandas. Se selecciona una primera banda, que es la que tiene una mayor amplitud con respecto al threshold in quiet, y se multiplica por una función de ensanchamiento (spreading function) definida por ciertos parámetros, y que representa el efecto del enmascaramiento frecuencial comentado anteriormente (gráfica de la derecha). La función de ensanchamiento está definida por una serie de parámetros, obtenidos experimentalmente, como la pendiente de bajada (left y right slope) y el valor máximo.

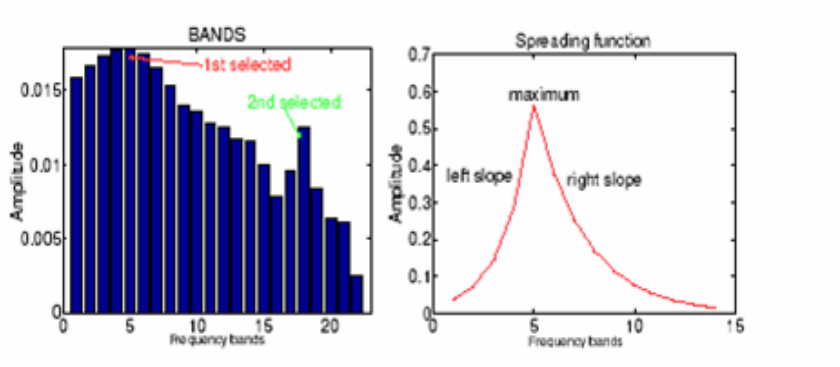


Figura 15. Algoritmo de Selección de la PACE

Tras multiplicar el valor de amplitud de la banda elegida por la función de ensanchamiento, se suma el resultado de la multiplicación con el threshold in quiet, obteniendo de esta forma un nuevo umbral, que en la gráfica inferior izquierda de la figura 16 vemos en rojo con el nombre de “Threshold 1”.

Hasta aquí ya hemos seleccionado la primera banda y hemos calculado su efecto enmascarador. El siguiente paso es seleccionar la siguiente banda, que de la misma forma, será la banda de mayor amplitud, pero ahora, con respecto del “Threshold 1”. Elegida la segunda banda la multiplicamos por la misma función de ensanchamiento y sumamos el resultado con el Threshold 1 obteniendo un nuevo umbral, representado de verde en la figura inferior derecha denominada “Threshold 2”.

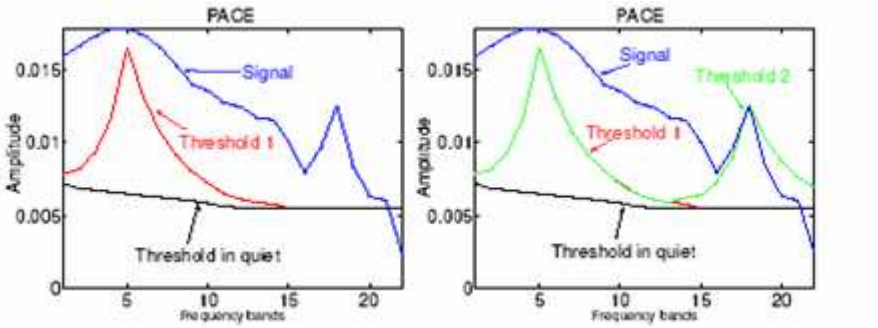


Figura 16. Algoritmo de Selección de la PACE

El proceso se repite hasta seleccionar el número de bandas deseado, en nuestro caso 8.

Estas 8 bandas junto con sus respectivas amplitudes calculadas en el proceso, corresponden con los 8 electrodos a estimular y sus respectivas intensidades de estimulación adecuadas al implante, que son calculadas en el último bloque de la estrategia PACE, el bloque “Mapping”, que, por otra parte, no se abordará aquí por ser transparente en lo que refiere al modelo psicoacústico y al desarrollo de este proyecto [6].

5 MODELO PSICOACÚSTICO MEJORADO PARA ESTRATEGIAS "NdeM" PARA IMPLANTES COCLEARES

Los avances que en los últimos años ha habido en el campo de la codificación de audio, han hecho posible que se adopten nuevas técnicas basadas en los nuevos fenómenos descubiertos. En el apartado 1 del capítulo 3 Fundamentos se vieron ya algunos de estos fenómenos, pero ahora vamos a identificarlos dentro de la estrategia PACE y a explicar cuáles son las mejoras planteadas para tener en cuenta estos fenómenos intrínsecos del sonido.

Básicamente se trata de intentar mejorar el modelo psicoacústico de la estrategia PACE. Y es que esta estrategia usa un modelo psicoacústico muy simple. Veamos por qué causas:

- PACE sólo tiene en cuenta efectos de enmascaramiento en el dominio de la frecuencia. A este fenómeno también se le denomina enmascaramiento simultáneo, que sucede en la frecuencia para un mismo instante de tiempo.

- La misma función de ensanchamiento o spreading function es usada para componentes de tipo tonal (voz, por ejemplo) y componentes de tipo ruidoso, a la hora de calcular el enmascaramiento y selección de las bandas.

Así, el objetivo del proyecto es la mejora del modelo psicoacústico de la estrategia PACE, afrontando las dos cuestiones anteriores.

5.1 Implementación de Enmascaramiento Temporal

En primer lugar, decíamos que el modelo de la estrategia PACE tenía una limitación en cuanto que sólo consideraba efectos de enmascaramiento en la frecuencia. Pues bien, numerosos estudios han demostrado que este fenómeno también sucede en el dominio del tiempo. Por tanto, en el nuevo modelo consideraremos **enmascaramiento temporal**, que tiene el mismo efecto que el enmascaramiento frecuencial pero que se produce en el eje temporal. Este fenómeno es también conocido con el nombre de **enmascaramiento no simultáneo**, ya que se produce para instantes de tiempo distintos.

Se puede demostrar que el enmascaramiento temporal tiene mayor relevancia cuando se presenta un transiente. Y, ¿qué es un transiente? Un transiente es una **señal** o **forma de onda** que empieza con una amplitud baja, para pasar en un muy breve espacio de tiempo, a una amplitud considerable. Ejemplos son el sonido de un disparo de un rifle, o la vibración de un golpe de un martillo. Sin saber mucho de psicoacústica, todo el mundo sabe, que si estamos hablando con una persona, y de pronto se produce un fuerte ruido, este ruido no nos permitirá oír la conversación. Este es un claro ejemplo de enmascaramiento temporal donde el fuerte ruido es el transiente. El fuerte ruido será, en este caso, la señal enmascaradora y la conversación la señal enmascarada.

Vemos a continuación la secuencia de audio “castanets”, comúnmente utilizada como referencia para el desarrollo de diversos estándares de codificación de audio [3,13]. Esta señal de audio que se representa abajo, en la figura 17, corresponde al sonido de unas castañuelas.

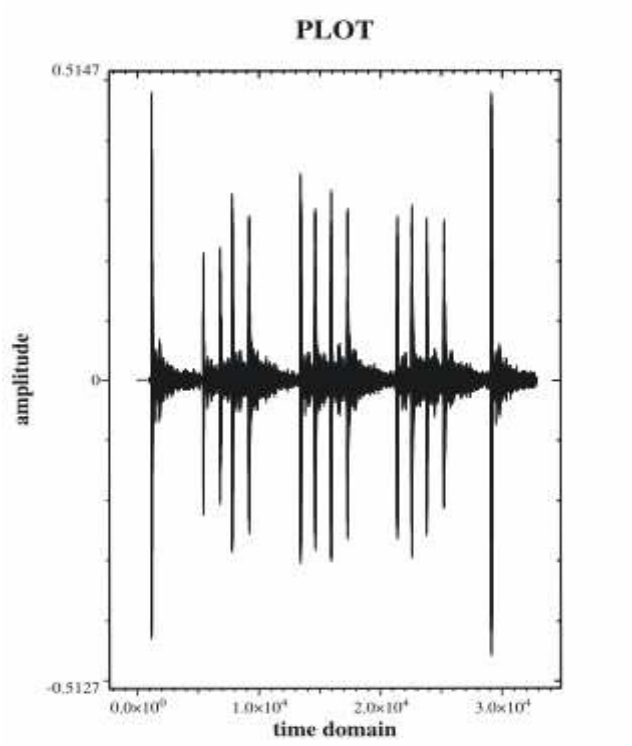


Figura 17. Transientes

Se aprecian muy bien los grandes picos que representan los transientes. Todo lo contrario de lo que sucede en una representación en el dominio frecuencial, donde los transientes generan un espectro suave, en el que la energía está distribuida sobre todo el rango de frecuencias.

No es intención aquí la de explicar como se ha llevado a cabo la detección de transientes utilizada en este proyecto. Sólo decir que ésta fue implementada y diseñada por una compañera de laboratorio como parte de su proyecto fin de carrera, que se centraba en el banco de filtros anterior al bloque de “Sampling and Selection”.

La implementación en nuestro modelo de enmascaramiento temporal parte del conocimiento de los transientes que pueda haber en la señal de audio a analizar, aunque esta detección de transientes se haga, por supuesto, en tiempo real.

La implementación de enmascaramiento temporal consistirá en una primera fase de inicialización, donde se definirá y creará la función de ensanchamiento temporal, y una segunda fase, donde se calculará el umbral de enmascaramiento producido por este fenómeno, haciendo uso de la función de ensanchamiento.

5.1.1 Función de ensanchamiento temporal (Inicialización)

Observamos en la siguiente figura una función general de ensanchamiento temporal. Vemos que consta de una etapa de pre-masking, otra llamada simultaneous masking y una última de post-masking.

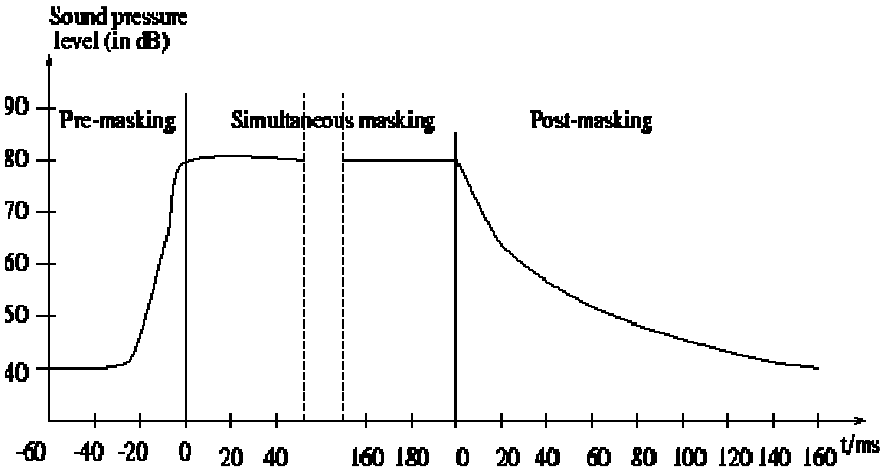


Figura 18. Enmascaramiento temporal

Como ya dijimos anteriormente, la etapa de pre-masking consiste en el enmascaramiento de señales debido a una señal posterior. Hoy en día no se sabe mucho a ciencia cierta de este fenómeno, aunque si se ha demostrado que se produce.
De todas formas, se decidió al principio del proyecto que no se tendría en cuenta este fenómeno anterior a la señal enmascaradora debido a la mínima duración de sus efectos (apenas unos milisegundos).
La siguiente etapa, la de simultaneous masking, se refiere al enmascaramiento que sufren las señales para ese mismo instante de tiempo, esto es, enmascaramiento frecuencial. Este enmascaramiento frecuencial es tratado aparte por la estrategia. Con lo que nuestra función de ensanchamiento se ceñirá al fenómeno de post-masking.

Cabe preguntarse, **¿se ve afectado realmente nuestro modelo por el fenómeno de post-masking?** La respuesta es sí, en teoría sí. Explicación:

En nuestro modelo, el periodo de estimulación del implante es de 2 mseg, ya que 500 Hz es la frecuencia de estimulación (1/500 = 0.002). Esto quiere decir que cada 2 mseg se están estimulando 8 electrodos. Por otra parte, de la figura 18, observamos que los efectos de enmascaramiento temporal pueden llegar incluso a durar hasta pasados los 100 mseg. Con esto, podemos afirmar, sin lugar a dudas, que este modelo se verá afectado por el efecto de post-masking.

Teniendo en cuenta todo lo anterior, adelantamos que la función de ensanchamiento usada será básicamente una función decreciente con el tiempo, pero, ¿qué parámetros definen nuestra función?
Tendremos dos parámetros que definen nuestra función, uno nos marcará la pendiente de bajada y el otro será un valor de amplitud, normalmente llamado "peak Offset", que guarda relación con la amplitud de la función de ensanchamiento, que es lo mismo que decir, con la cantidad de enmascaramiento que se producirá.

Para la elección de estos valores, nos hemos servido de un trabajo de codificación de voz [14], que tiene en cuenta los efectos psicoacústicos de enmascaramiento, tanto en el tiempo como en la frecuencia. Se han decidido los siguientes valores:

Pendiente de bajada (dBs/mseg) = 15.

Peak Offset (dBs) = 10.

Con estos datos, la función de ensanchamiento generada es la siguiente:

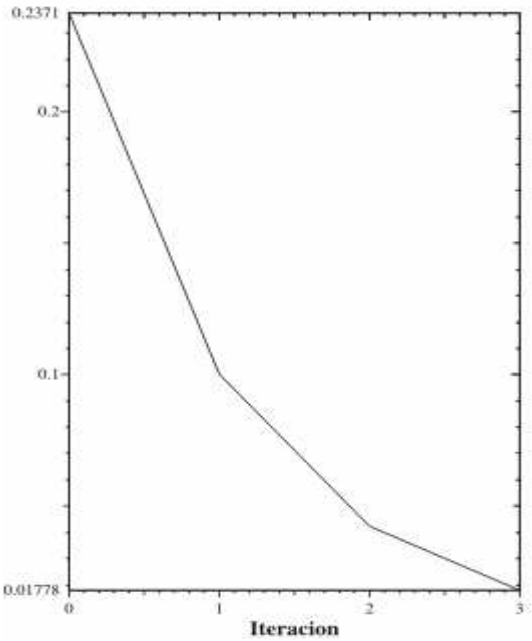


Figura 19. Función de ensanchamiento temporal

El código que implementa esta función se muestra a continuación.

```
void TPsychoacousticProc::TimeGenSpreadTable(TParams *p)
{
  int j;
  short numright;
  double value,value1;

  numright= 0;

  for(j=1; j<=10; j++)
   {
    value= TimeSpreadFunction(j, p->TpeakOffset, p->postMaskSlope, p->exp);
    value1= db2exp(-(p->levelmin), p->exp);
    if(value < value1)
    break;
   }

  numright=j-1;
  p->TimeLenSpreadTable= numright;

  p->TimeSpreadTable=(double *)malloc(numright*sizeof(double));

  for(j=1; j<=numright; j++)
  {
   p->TimeSpreadTable[j-1]=TimeSpreadFunction(j, p->TpeakOffset, p->postMaskSlope, p->exp);
  }

//Pv(numright,"xyd","Iteracion","Amplitud", p->TimeSpreadTable);

}



double TPsychoacousticProc::TimeSpreadFunction(short j, short TpeakOffset, short postMaskSlope,
double expo)
{
 double value2;

  value2=db2exp(-TpeakOffset-j*postMaskSlope, expo);

 return value2;
}


double TPsychoacousticProc::db2exp(short dbx, double exp)
{
  double q;

  q=pow(pow(10,(double)(dbx)/(double)(10)),exp);

  return q;
}
```

Hasta aquí hemos obtenido la función de ensanchamiento temporal. A continuación veremos cómo se calcula el umbral de enmascaramiento temporal a partir de ella.

5.1.2 Cálculo del umbral de enmascaramiento

Antes de nada, vamos a repasar brevemente dónde nos encontramos. Seleccionamos 128 muestras de la señal de audio en cada iteración. Después del banco de filtros nos encontramos con la señal ya transformada en 22 bandas frecuenciales. Estas bandas ocupan todo el rango de frecuencias que nuestro oído es capaz de percibir, desde los pocos Hertzios hasta los 20 KHz.

De entre estas 22 bandas, aquellas en las que se ha detectado transiente, son las que, como ya se ha dicho, producirán enmascaramiento, y debemos por tanto de aplicar enmascaramiento temporal.

Es importante mencionar, que el algoritmo de selección de bandas, se mantiene igual a aquél que se explicó en el apartado 4 de este escrito, correspondiente al modelo psicoacústico de la estrategia PACE [6]. La diferencia es que, el umbral de partida, a la hora de calcular la primera banda significativa respecto a dicho umbral, será, la suma del umbral de audibilidad más el umbral de enmascaramiento temporal, generado del cálculo de la multiplicación de la función de ensanchamiento temporal por el valor de amplitud de aquellas bandas dónde se haya detectado transiente.
Podemos ver esto de manera ilustrativa en la siguiente figura.

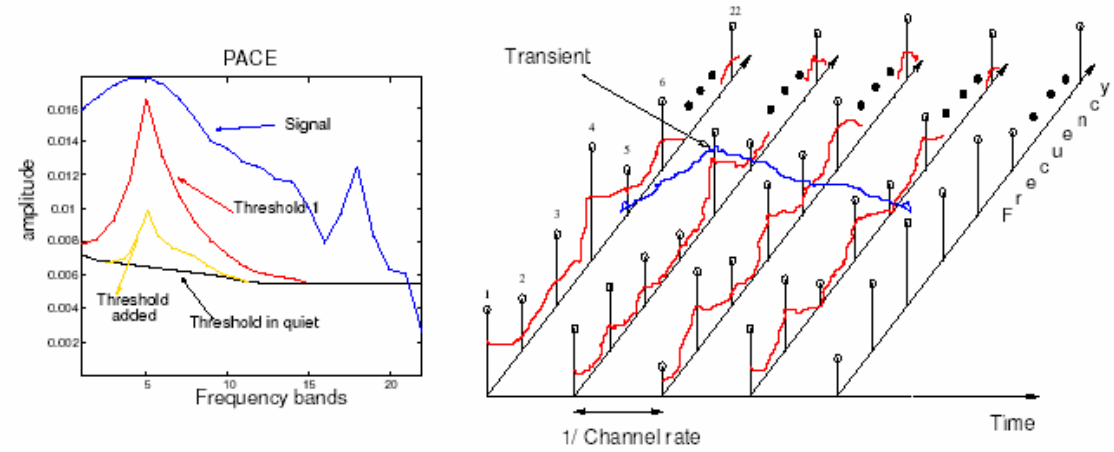


Figura 20. Cálculo de enmascaramiento temporal

A la derecha tenemos una gráfica con el tiempo y la frecuencia por ejes. Vemos las distintas bandas frecuenciales en el eje de la frecuencia. En el eje del tiempo, el tiempo de estimulación. El enmascaramiento frecuencial se representa con el color rojo y el temporal de color azul. Entonces, si por ejemplo se detecta un transiente en la quinta banda frecuencial de la segunda trama temporal (iteración), como vemos en el dibujo, se calculará el umbral de enmascaramiento temporal generado (color azul) y se sumará al umbral de enmascaramiento frecuencial (ver gráfica de la izquierda: “Threshold added”). Así, las bandas seleccionadas serán distintas ya que se tienen en cuenta, en este caso, además de los efectos psicoacústicos frecuenciales, los efectos psicoacústicos temporales.

Para ello, se creó la función “TDetectTrans”, cuyo código vemos a continuación:

```
void TPsychoacousticProc::TDetecTrans(TParams *p, double *final)
{
  int i;
  short len;
  double *valueTransVector;

  len=p->TimeLenSpreadTable;

  valueTransVector=(double *)malloc(p->numBands*sizeof(double));

  if(p->cont==1)
   {
    for(i=0; i<p->numBands; i++)
     {
      if(p->trans[i]==1)
       {
        p->acu[i]= len;
        valueTransVector[i]= pow(p->env[i], p->exp);
        p->tempThr[i]= valueTransVector[i] * p->TimeSpreadTable[0];
        p->acu[i]--;
        final[i]= 0;
       }
      else
       {
        final[i]= 0;
        p->tempThr[i]= 0;
       }
     }
   }
  else
   {
    for(i=0; i<p->numBands; i++)
     {
      valueTransVector[i]= pow(p->env[i],p->exp);
      final[i]= p->tempThr[i];

      if (p->trans[i]==1)
       {
        if(p->acu[i]>0)
         {
          p->tempThr[i]= valueTransVector[i] * p->TimeSpreadTable[len-p->acu[i]];
          p->acu[i]--;
         }
        else
         {
          p->acu[i]= len;
          p->tempThr[i]= valueTransVector[i] * p->TimeSpreadTable[len-p->acu[i]];
          p->acu[i]--;
         }
       }
      else if(p->trans[i]==0)
       {
        p->tempThr[i]= valueTransVector[i] * p->TimeSpreadTable[len-p->acu[i]];

        if(p->acu[i]<=0)
         p->tempThr[i]= 0;

        p->acu[i]--;
       }
     }
   }
  free(valueTransVector);
}
```

Así, esta función es la encargada de, en las bandas donde se ha detectado la existencia de un **transiente** (p->trans[i]=1), **multiplicar el valor de amplitud de esa banda por el valor correspondiente de la función de ensanchamiento temporal**. Hay que tener en cuenta que en cada iteración (trama), se ha de calcular el enmascaramiento debido a transientes en tramas anteriores y en la actual. Por ejemplo, si se detecta un transiente en una determinada banda en la iteración 18, el transiente afectará a esa banda en las tramas siguientes, dependiendo el alcance, de la función de ensanchamiento definida.

Lo primero que hace esta función es guardar en final [i] el umbral de enmascaramiento temporal de las 20 bandas calculado en la trama anterior y calcular el nuevo umbral de enmascaramiento temporal de la presente iteración (debido a transientes anteriores y actuales), p->tempThr[i].

Tras esta función, en el algoritmo de selección de las bandas, se sumará **al inicio**, final[i] más el umbral de audibilidad(qe ya tiene en cuenta enmascaramiento frecuencial), obteniendo así un nuevo umbral de partida, que llamaremos por ejemplo umbral de audibilidad modificado. Esto significa, que el algoritmo de selección de bandas se mantiene exactamente igual que el de la estrategia PACE, pero que tiene en cuenta desde un principio, los efectos de enmascaramiento temporales.
Así, se selecciona una primera banda, que será aquella que tiene una mayor amplitud respecto, y aquí está la diferencia, al umbral modificado. A partir de aquí, nada varía, excepto las bandas seleccionadas, que ya no tendrán por qué coincidir con las seleccionadas por el modelo anterior, que sólo tenía en cuenta el enmascaramiento frecuencial.
Se multiplica la banda seleccionada por la función de ensanchamiento frecuencial, para obtener el umbral de enmascaramiento frecuencial. Este umbral de enmascaramiento frecuencial calculado se sumará al umbral de enmascaramiento modificado, obteniendo un nuevo umbral, respecto del cuál se selecciona la segunda banda. Así sucesivamente hasta seleccionar las 8 bandas deseadas.

5.2 Implementación de Detección de Tonalidad

Recordemos primero qué es esto de la tonalidad. A groso modo, diremos que la tonalidad hace referencia a la periodicidad de la señal en el tiempo. Es decir, hay sonidos que por su propia naturaleza son más o menos periódicos, o lo que es lo mismo, son más o menos tonales.
Así por ejemplo las vocales son siempre sonidos de carácter tonal (cuasi-periódicos) y por consiguiente de espectro discreto. En cambio, las consonantes son sonidos que pueden ser tonales o no tonales, dependiendo de si las cuerdas vocales están vibrando o no. Consonantes tonales son por ejemplo: “b”, “d”, “m”, etc. Y consonantes no tonales son: “s”, “z”, “j”, “f”.
Por otro lado el ruido se caracteriza por no presentar periodicidad alguna, por consiguiente su espectro es plano. Por esto el ruido tiene carácter no tonal.

A partir de ahora, sólo trabajaremos en el dominio frecuencial. Se ha comprobado que el cerebro humano actúa enmascarando (frecuencialmente) de diferente forma sonidos con mayor o menor tonalidad. Así, si tenemos un sonido tonal, el efecto de enmascaramiento frecuencial será menor que el producido si en el sonido predominan los componentes no tonales.
El no tener en cuenta este detalle, debería de significar la pérdida de comprensión auditiva por parte del paciente. Por ello se propuso la segunda mejora al modelo psicoacústico de la estrategia PACE, un detector de tonalidad.

Centrándonos, como hemos dicho, en el **dominio frecuencial**, implementaremos un **detector de tonalidad**, que detecte en cada una de las 20 bandas frecuenciales si la señal procesada tiene componentes tonales o no tonales (ruido), y en base a ello, se utilice una u otra función de ensanchamiento frecuencial más adecuada para cada caso. Se conseguirá de este modo una **función de ensanchamiento** frecuencial **adaptativa**. De esta forma, los efectos de enmascaramiento calculados se aproximarán más a la realidad.

Para la implementación del detector de tonalidad se han propuesto 2 métodos distintos, ya que el primero de ellos no daba resultados correctos en todos los casos, como veremos a continuación.

5.2.1 Implementación de la medida “Spectral Flatness Meassure”

Este método trabaja sobre el espectro de la señal. Como su propio nombre indica, calcula una medida de la “planicidad” del espectro de la señal. Y es que dependiendo de cómo de plano sea el espectro, se puede estimar la tonalidad de la señal a la que pertenece dicho espectro. Veamos cómo.

El SFM se calcula según la siguiente ecuación:

$$SFM = \frac{\text{Media geométrica}}{\text{Media aritmética}}$$

Para el cálculo de la media geométrica y aritmética hemos, primero, de conocer con un poco más de detalle, algunos conceptos usados en la estrategia PACE.

El banco de filtros realmente no descompone la señal de audio digital directamente en 22 bandas frecuenciales, sino que se hace una primera descomposición obteniéndose un total de 64 bins. Es luego, cuando se

agrupan un cierto número de bins por cada banda, de acuerdo con la teoría del Bark.

Esta teoría nos dice, básicamente, el número de bins que corresponden a cada banda. Según esta teoría, las bandas correspondientes a frecuencias inferiores están formadas por un solo bin, número que va aumentando en bandas superiores, donde el número de bins asciende hasta los 7 u 8 bins.

En la siguiente figura se ilustra en que consiste la mencionada teoría del bark.

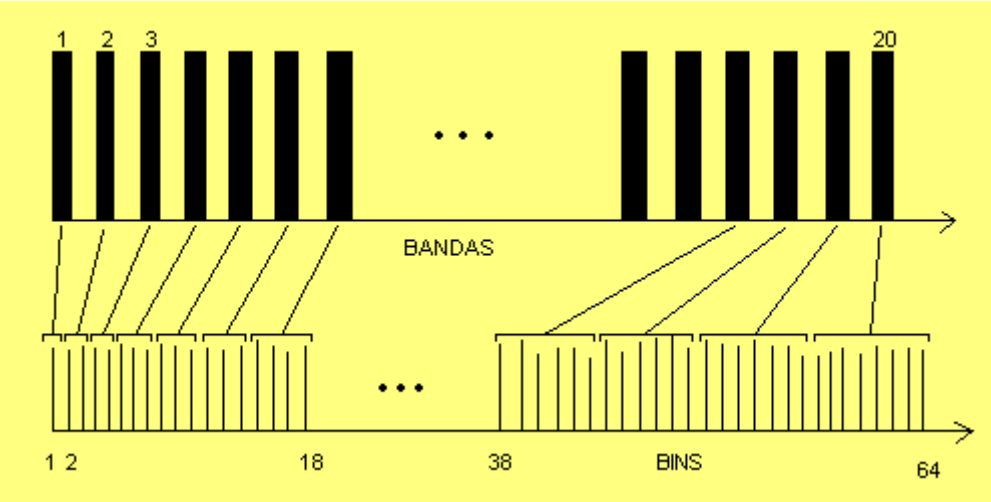


Figura 21. Teoría del Bark

Vemos como decíamos, que a la primera banda le corresponde un sólo bin, a la segunda y tercera dos bins, y así hasta llegar a la última banda que está formada por 8 bins. Estos datos no son exactos, sólo se intenta en este punto, entender en que consiste esta parte del procesado, ya que el cálculo de la media geométrica y aritmética se hace respecto a los bins de cada una de las 20 bandas.

El SFM se calculará para cada una de las 20 bandas frecuenciales, obteniendo así un valor de tonalidad para cada banda, y es que para una misma trama puede haber bandas formadas fundamentalmente por componentes tonales y bandas que no.
Las ecuaciones que rigen el cálculo de la media aritmética y geométrica se escriben a continuación:

Media geométrica $= \left(\prod x^2 \right)^n$

donde $n = 1/K$
$K = n^o$ de bins
x = densidad espectral de energía del bin k

Media aritmética $= \frac{1}{K} \sum x^2$

Donde x = densidad espectral de energía del bin k
K = nº de bins

Así pues, ya hemos calculado el SFM, a partir la media aritmética y geométrica de los bins que forman cada banda.
Veremos que el SFM varía en el rango de 0 a 1, correspondiendo el valor 1 a un espectro totalmente plano y valores próximos a 0, a espectros no planos.
Se sabe, por otro lado, que el ruido siempre va a generar un espectro plano, y, que cuánto menos plano sea el espectro, menos ruidosa será la señal. Pero lo que nos interesa es la correspondencia de la señal que tenemos con el valor de tonalidad. Para ello, el siguiente paso es calcular la tonalidad en cada banda a partir de la medida del SFM.

El valor de tonalidad estará en el rango de 0 a 1 también, correspondiendo valores próximos a 0 para señales no tonales (ruido) y valores próximos a 1 para señales muy tonales. No se debe confundir el SFM con la tonalidad, tienen correspondencia inversa.

Para ello, nos fijamos en un documento “Tonality and its Aplication to Perceptual-Based Speech Enhancment” [2], cuyo autor hace uso precisamente de la medida del SFM para calcular la tonalidad de cada banda frecuencial. Y lo hace de la siguiente forma:

$$\textbf{SFM}(i)\textbf{(dB)} = 10 * \log\left(\frac{M.\ geo}{M\ artm}\right) \quad y\ ;$$

$$\textbf{Tonalidad}(i) = \min\left(\frac{SFM(i)(dB)}{-60}, 1\right)$$

Pero se pudo comprobar que los valores de tonalidad que se obtenían, no eran totalmente satisfactorios, ya que, por ejemplo, para una señal formada por un tono puro, un seno a cierta frecuencia por ejemplo, para la que se obtenía un valor de SFM de 0.0001 (espectro nada plano, hasta aquí todo bien), se correspondía con un valor de tonalidad de cómo máximo 0.7 sobre 1, cuando teóricamente debiera haber dado 1 de tonalidad, ya que un seno es una señal tonal pura.

Nos reafirmamos todavía más cuando comprobamos que efectivamente, con el método de predicción que veremos a continuación, el valor obtenido de tonalidad para la misma señal sinusoidal era, efectivamente 0.999. Así que, a nuestro modo de ver, la correspondencia que hacía el autor entre SFM y

tonalidad no nos servía para nuestro modelo. Se decidió entonces calcular la tonalidad para este método directamente del valor del SFM de la siguiente forma:

Tonalidad(i) = 1 – SFM(i)

Esto se pudo hacer ya que cuánto más plano es el espectro (ruido), el SFM más tiende a 1 y menos tonal es la señal (ya que vimos que el ruido tiene solamente componentes no tonales), con lo que la tonalidad tiende a 0.

Hasta aquí hemos calculado el valor de tonalidad de cada una de las 20 bandas frecuenciales. El último paso que resta es calcular la función de ensanchamiento específica para cada una de las 8 bandas que van a ir siendo seleccionadas en el bucle de selección de bandas. Así, en cada paso, cuando se selecciona una banda, por ser la más representativa de la señal, se calculará el enmascaramiento que produce, multiplicando esa banda por la función de ensanchamiento específica que hemos calculado para esa banda en cuestión (en base al valor de tonalidad calculado para esa banda).

Hemos conseguido de esta forma implementar una función de ensanchamiento adaptativa, intentando acercarnos un poco más a al forma de procesar del sistema auditivo humano.

Sólo quedaría por detallar mediante que expresiones se calcula la función de ensanchamiento. En un principio, y considerando como estamos considerando, enmascaramiento frecuencial, trabajábamos con una función de ensanchamiento frecuencial definida por los siguientes tres parámetros, uno más que para la función de ensanchamiento temporal, las pendientes de bajada (left y right slope), y el valor de amplitud máximo (o peak Offset).
El valor de tonalidad obtenido, modifica únicamente el valor de peak Offset a la hora del cálculo de la función de ensanchamiento. Las expresiones para obtener el peak Offset a partir de la tonalidad que se han decidido son las siguientes [2]:

Offset(i) = ton(i)*18 + (1- ton(i))*6 (dB), hasta 1.5 KHz y;

Offset(i) = ton(i)*(18+i) + (1-ton(i))*6 (dB).

Así por ejemplo, estando por debajo de los 1500 Hz, a un valor 1 de tonalidad le corresponderá un offset de 18 dB, y a un valor 0 de tonalidad le corresponderán 6 dB de offset, a la hora de calcular la función de ensanchamiento para cada banda de las escogidas.
Entre estos dos extremos, tenemos el correspondiente rango de valores continuo. El offset puede variar de 6 a 18 dBs, para frecuencias inferiores a 1500 Hz, y tiene un mayor rango superior para frecuencias superiores a 1500 Hz.

El código que implementa todo lo anterior:

```
if(p->bandBins[i]>3)
 {

            //********** Spectral Flatness Measure **********//


  GM[i]=1;
  AM[i]=0;

  for(s=0;s<p->bandBins[i];s++)
   {

    GM[i]= GM[i]* p->abs[k];
    AM[i]= AM[i]+ p->abs[k];

    k++;

   }


  GM[i]= pow(GM[i],((double)(1)/(double)(p->bandBins[i])));
  AM[i]= AM[i]/p->bandBins[i];

  SFM[i]=GM[i]/AM[i];


  //******************************

  //SFMdb[i]=10*log10(SFM[i]);
  //ton[i]=SFMdb[i]/(-60);

  //******************************


  ton[i]=1-SFM[i];


  if(p->cont>=3)
   {
    if(p->crossOverFreqs[i+1]<1500)
     p->offset[i]=ton[i]*18+(1-ton[i])*6;

    else
     p->offset[i]=ton[i]*(18+i)+(1-ton[i])*6;
   }

 }
```

Después de todo, probando el algoritmo SFM, se percibió que no se obtenían valores de tonalidad correctos para todos los casos. En aquellas bandas formadas por tres o menos bins, los valores de tonalidad no eran del todo correctos. Esto se debía a que al estar formadas estas bandas por muy pocos bins, al calcular las medias (geométrica y aritmética) no se podía obtener un resultado muy preciso. Surgió entonces la necesidad de implementar un nuevo algoritmo para calcular el valor de tonalidad en aquellas bandas formadas por pocos bins, donde los resultados eran poco precisos mediante este método.

5.2.2 Implementación de predicción para detección de componentes tonales

Este método se basa en la predicción para calcular la tonalidad. En cada banda frecuencial, el valor de tonalidad se calcula basándose en cómo de predecible es la señal en la banda en cuestión. Así tendremos un vector de tonalidades, cuya longitud será el número de bandas frecuenciales en que se descompone la señal para ser procesada.

El método predictivo que se ha implementado [Chapter 11: MPEG audio, del libro “Introduction to Digital Audio Coding and Standards”] consiste, primero, en predecir mediante extrapolación lineal, la amplitud y fase de todas y cada una de las bandas frecuenciales de la trama actual, a partir de los valores de las dos tramas anteriores. Una vez calculados la amplitud y la fase predichos en cada una de las 20 bandas, sólo quedará llevar estos valores a una ecuación que calcula una medida de “cantidad de predicción” que nos dirá cómo de predecible es la banda “i” de la trama actual de nuestra señal de audio. Gracias a ello, se obtendrá finalmente el vector de tonalidades, a partir de la relación existente entre la medida de “cantidad de predicción” y la tonalidad, relación que veremos más adelante.

Antes de nada, comentar el fundamento de este método. Como hemos visto, en la anterior introducción, el método trabaja con la señal de audio principalmente en el dominio temporal, ya que hace una estimación a partir de valores anteriores. Analizando las propiedades de una señal de voz en el eje del tiempo, se puede observar, como ya dijimos anteriormente, el carácter periódico (tonal) que presentaban las vocales y algunas consonantes, y el carácter no periódico que siempre presentaba una señal ruidosa.
Así, este método aprovecha esa periodicidad, para, a partir del valor predicho (estimado) y, comparándolo con el valor real actual, calcular cuánto de periódica es la señal, o lo que es lo mismo, la tonalidad de la señal.

Veamos esto de la periodicidad en algunas gráficas. Se va a analizar en el tiempo la señal formada por la palabra “asa”. En primer lugar representaremos la vocal “a”.

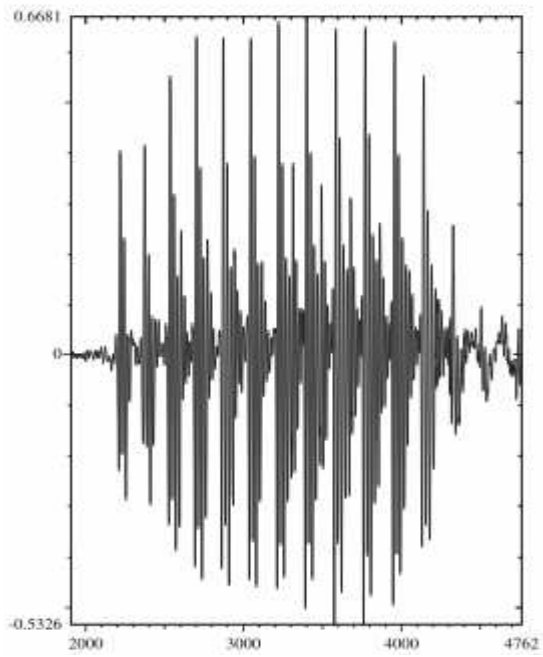


Figura 21. Vocal “a”

Se observa claramente una cierta periodicidad en la señal. La “a” dará como resultado un valor grande de tonalidad. En cambio, si nos fijamos en el carácter que viene a continuación, la consonante “s”:

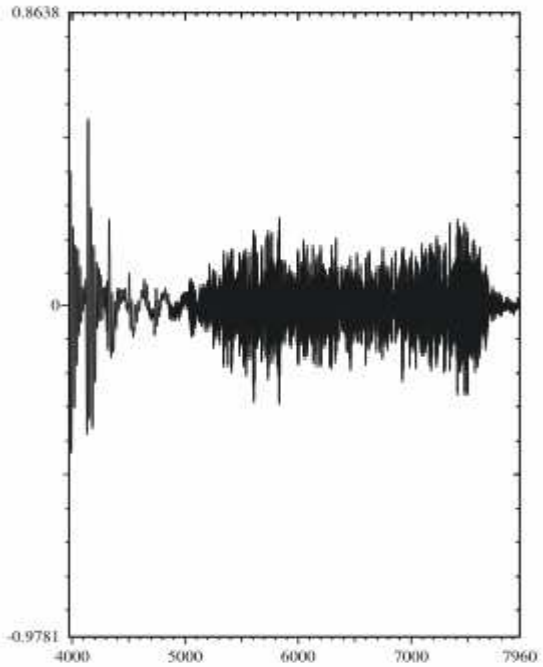


Figura 22. Consonante “s”

Se ve que la forma de la señal es totalmente distinta. Ya no tiene nada de periódica y por lo tanto esta señal será pobremente predicha, lo que conllevará, un valor de tonalidad pequeño.

En la siguiente figura está representada la señal de audio correspondiente a la palabra “asa”.

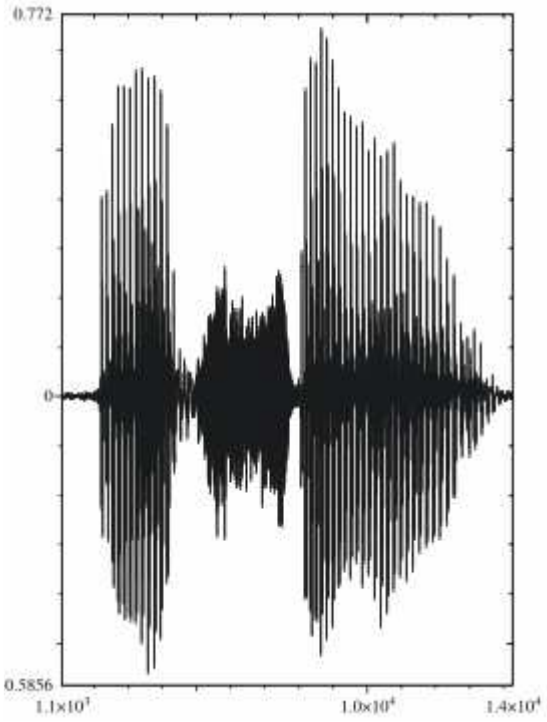


Figura 23. Palabra “asa”

Se puede observar cierta periodicidad al principio y al final, correspondiendo con las vocales “a”. Esta periodicidad, nos da en última instancia, el valor de tonalidad buscado. Ya que cuánto más periódica sea la señal, mejor predicha será, y un mayor valor de tonalidad dará.

Vayamos por pasos. En primer lugar, cómo se calculan los datos predichos:

$$A_m'[k] = A_{m-1}[k] + \{ A_{m-1}[k] - A_{m-2}[k] \}$$

$$\varphi_m'[k] = \varphi_{m-1}[k] + \{ \varphi_{m-1}[k] - \varphi_{m-2}[k] \}$$

o lo que es lo mismo:

$$A_m'[k] = 2*A_{m-1}[k] - A_{m-2}[k]$$

$$\varphi_m'[k] = 2*\varphi_{m-1}[k] - \varphi_{m-2}[k]$$

Donde $A_m'[k]$ y $\varphi_m'[k]$ representan los valores predichos de amplitud y fase, respectivamente, a partir de los valores de amplitud y fase en las dos tramas anteriores m-1 y m-2, para cada una de las bandas frecuenciales k.

Veamos como se ha programado esto para el caso de aquellas **bandas frecuenciales formadas sólo por un bin**. Veremos después que para el caso de más de un bin la programación es algo más compleja.

```
for(i=0; i<p->numBands; i++)
 {

 // ********** Tonality Index ********** //

 if(p->bandBins[i]==1)
  {

  if(p->cont==1)
   {
   p->amplitude2[i]=sqrt(p->abs[k]);
   p->fase2[i]=atan2(p->cbuf[k].im,p->cbuf[k].re);
   }

  if(p->cont==2)
   {
   p->amplitude1[i]=sqrt(p->abs[k]);
   p->fase1[i]=atan2(p->cbuf[k].im,p->cbuf[k].re);
   }

  if(p->cont>2)
   {

   amplitudeE[i]= p->amplitude1[i]+p->amplitude1[i]-p->amplitude2[i];
   //valor de amplitud estimado, predicho

   if(amplitudeE[i]<0)
   amplitudeE[i]=-amplitudeE[i];              //valor absoluto

   faseE[i]= p->fase1[i]+p->fase1[i]-p->fase2[i];
   //valor de fase estimado, predicho

   :
   :
   :
```

Seguidamente se calculan los valores reales de amplitud y fase de la iteración actual y, entre aquellos estimados y estos reales se calcula la medida de “cantidad de predicción” según las siguientes ecuaciones:

```
   p->amplitude[i]=sqrt(p->abs[k]);                  //amplitud real actual
   p->fase[i]=atan2(p->cbuf[k].im,p->cbuf[k].re);    //fase real actual


   test[i]=sqrt(( pow((amplitudeE[i]*cos(faseE[i]))-(p->amplitude[i]*cos(p->fase[i])),2) +
pow((amplitudeE[i]*sin(faseE[i]))-(p->amplitude[i]*sin(p->fase[i])),2)))/(p->amplitude[i]+amplitudeE[i]);

   //test, medida de cantidad de predicción
```

La variable test nos da, como hemos dicho, una medida de la cantidad de predicción, donde test es igual a 0 cuando el valor actual es exactamente igual

al predicho, e igual a 1 cuando los valores actual y predicho son dramáticamente distintos. Si pensamos que cuando mejor predicha sea, es porque más tonal es la señal, tendremos que valores de test→0 se corresponden con valores de tonalidad→1. Así, tendremos una relación de la forma:

```
ton[i]=1-test[i];
```

Lo único que restaría es actualizar los valores de amplitud y fase de las variables de las dos tramas anteriores, con vistas a la siguiente iteración.

```
    p->amplitude2[i]=p->amplitude1[i];
    p->amplitude1[i]=p->amplitude[i];

    p->fase2[i]=p->fase1[i];
    p->fase1[i]=p->fase[i];
    }

  k++;
  }                          //fin de if(p->bandBins[i]==1)


if(p->bandBins[i]==2)
  {

    :
    :
    :
```

Bien, este sería el procedimiento para los casos en los que la banda frecuencial esté formada por un solo bin. Queda entonces por ver aquellos casos en los que la banda está formada por más de 1 bin, que son la mayoría.

Para aquellas bandas formadas por más de un bin, de alguna forma, había que tener en cuenta todos los bins a la hora de calcular el valor de tonalidad correspondiente a dicha banda. Se decidió repetir el procedimiento explicado antes para cada uno de los bins, con lo que se obtendría una medida de la "cantidad de predicción" para cada bin. Luego habría que hacer una media ponderada de la energía de cada bin por su valor de cantidad de predicción. Así aquellos bins con una mayor energía, repercutirán con un mayor peso en la medida de la tonalidad. De esta forma se obtendría el valor de tonalidad correspondiente a la banda en cuestión. De esta forma, si tuviéramos una banda formada por 4 bins, el cálculo de la tonalidad se haría de la siguiente forma:

```
        if(p->bandBins[i]==4)
          {
```

```
:
:          // Cálculo de cantidad de predicción en cada bin
:


absRaiz[k-4]=sqrt(p->abs[k-4]);
absRaiz[k-3]=sqrt(p->abs[k-3]);
absRaiz[k-2]=sqrt(p->abs[k-2]);
absRaiz[k-1]=sqrt(p->abs[k-1]);


ton[i]=1-((test[i]*absRaiz[k-4]+testbin2[i]*absRaiz[k-3]+testbin3[i]*absRaiz[k-
2]+testbin4[i]*absRaiz[k-1])/(absRaiz[k-4]+absRaiz[k-3]+absRaiz[k-2]+absRaiz[k-1]));


} //fin de if(p->bandBins[i]==4)


:
:
:
```

Pues bien, hecho esto, tenemos en un vector los valores de tonalidad correspondientes a cada una de las bandas. Por último, quedaría de nuevo por ver la cuestión de cómo calcular, a partir del valor de tonalidad obtenido, el "offset", dato con el que se calculará directamente la función de ensanchamiento frecuencial específica para cada banda. Esto se hace exactamente igual a como ya vimos para el método SFM, esto es:

Offset(i) = ton(i)*18 + (1- ton(i))*6 (dB), hasta 1.5 KHz y;

Offset(i) = ton(i)*(18+i) + (1-ton(i))*6 (dB).

Se ha podido comprobar que el método de predicción aquí implementado funciona correctamente y, a diferencia del anterior SFM, este método funciona de forma correcta para todas las bandas sin excepción. Así que, aunque en un principio se pensó en usar los dos métodos simultáneamente (SFM para altas frecuencias, muchos bins; Predicción para bajas frecuencias, pocos bins), quizá no sea ésta buena idea, y sea mejor utilizar un único método que se sabe que funciona bien en todos los casos.

6 EXPERIMENTOS

En el presente apartado de esta memoria se van a describir una serie de experimentos, cuya única finalidad es constatar que el código implementado da resultados coherentes. Así pues, comprobaremos que, tanto la implementación del detector de tonalidad como la de enmascaramiento temporal presentan los resultados esperados.

6.1 Detector de tonalidad. Experimentos.

En cuanto al detector de tonalidad se debería comprobar que una señal tonal da mayor valor de tonalidad que una señal ruidosa. Para ello se procesará el algoritmo con dos señales diferentes, una señal de voz y una señal ruidosa.

Como ya dijimos anteriormente todas las vocales son señales tonales (debido a su periodicidad), en cambio, las consonantes pueden comportarse de forma tonal o no tonal. La cuestión es, entonces, procesar los dos tipos de señales y comparar los resultados, esperando que el valor de tonalidad obtenido sea mayor para los sonidos que contienen vocales que para los formados por consonantes, donde el valor de tonalidad es más bien aleatorio.

El primer experimento que se realizó para comprobar el funcionamiento del detector de tonalidad consiste en lo siguiente. Simplemente procesar como señal de audio una señal sinusoidal, que es la señal más tonal que puede haber. Si el detector funcionaba correctamente, debía de obtenerse la mayor tonalidad, esto es, 1.

Pues bien, se introdujo como señal de audio la siguiente señal sinusoidal:

```
for(i=1; i<bs; i++)

   {
     ale=rand();
     SeñalAudio[i]=200*sin(zpi*50/128*i)+(ale/2147483648);
   }
```

Se trata de una señal formada por ruido en todos los bins excepto en el bin 50, donde se ha introducido un seno, tal y como se ve en negrita arriba. Esto quiere decir que al procesar esta señal de audio, el valor de tonalidad debería ser aleatorio en todos los bins, excepto en el 50. Como sabemos, un seno en el tiempo equivale a una delta en el espectro. Lo vemos:

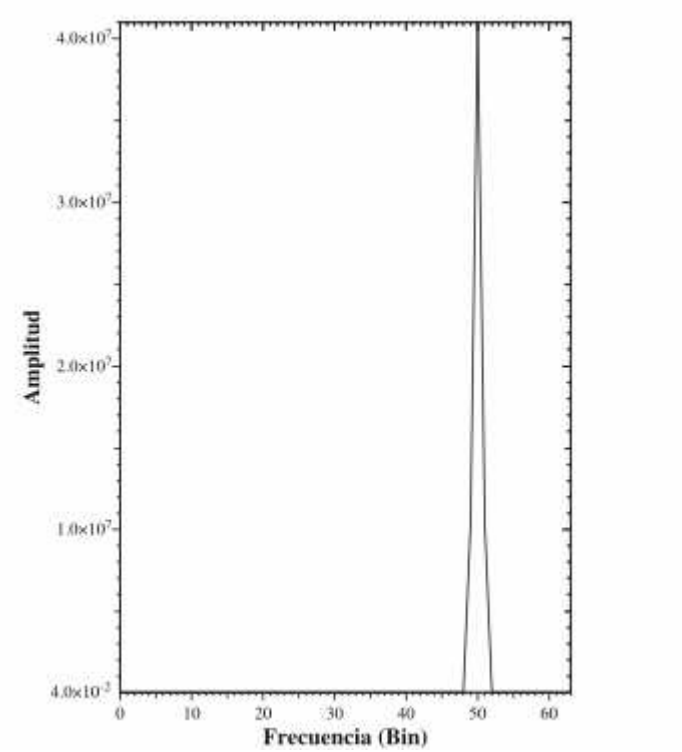


Figura 24. Delta en el espectro

Pues bien. Los valores de tonalidad obtenidos son:

Iteracion99 TonPred1 0= 0.004208
Iteracion99 TonPred1 1= 0.009505
Iteracion99 TonPred1 2= 0.600845
Iteracion99 TonPred1 3= 0.740089
Iteracion99 TonPred1 4= 0.010446
Iteracion99 TonPred1 5= 0.003217
Iteracion99 TonPred1 6= 0.471573
Iteracion99 TonPred1 7= 0.051309
Iteracion99 TonPred2 8= 0.406046
Iteracion99 TonPred2 9= 0.119324
Iteracion99 TonPred2 10= 0.360402
Iteracion99 TonPred3 11= 0.271996
Iteracion99 TonPred3 12= 0.089663
Iteracion99 TonPred4 13= 0.240447
Iteracion99 TonPred4 14= 0.456782
Iteracion99 TonPred5 15= 0.073165
Iteracion99 TonPred6 16= 0.277687
Iteracion99 TonPred7 17= 0.214261
Iteracion99 TonPred8 18= 0.999310
Iteracion99 TonPred8 19= 0.266906

Tenemos aquí los valores de tonalidad correspondientes a las veinte bandas frecuenciales. Recordando lo que dijimos acerca de la teoría del Bark, el bin 50 cae dentro de la banda 19, marcada en negrita, y cuyo valor de tonalidad es, como se predijo, de prácticamente 1, muy tonal.

El siguiente paso a realizar es comprobar que una señal de características tonales daba mayores valores de tonalidad que una señal de características ruidosas. Para ello, se utilizará la señal “asa” que vimos representada en el punto anterior de este escrito. Veremos si el valor de tonalidad cuando se está procesando la vocal “a” es mayor o no, que cuando se está procesando la consonante “s” por ejemplo, o una señal ruidosa, que debiera dar valores aleatorios de tonalidad.

Pero antes de ver los resultados, hay que destacar lo siguiente. El rango de frecuencias que nuestro programa procesa, abarca, desde los pocos herzios hasta los casi 7000 Hz. Así, las bandas inferiores tendrán frecuencias menores que las de bandas superiores. En concreto, aquí tenemos el rango superior de cada una de las veinte bandas:

Frecuencia superior de la banda 0 = 187.500000
Frecuencia superior de la banda 1 = 312.500000
Frecuencia superior de la banda 2 = 437.500000
Frecuencia superior de la banda 3 = 562.500000
Frecuencia superior de la banda 4 = 687.500000
Frecuencia superior de la banda 5 = 812.500000
Frecuencia superior de la banda 6 = 937.500000
Frecuencia superior de la banda 7 = 1062.500000
Frecuencia superior de la banda 8 = 1187.500000
Frecuencia superior de la banda 9 = 1437.500000
Frecuencia superior de la banda 10 = 1687.500000
Frecuencia superior de la banda 11 = 1937.500000
Frecuencia superior de la banda 12 = 2312.500000
Frecuencia superior de la banda 13 = 2687.500000
Frecuencia superior de la banda 14 = 3187.500000
Frecuencia superior de la banda 15 = 3687.500000
Frecuencia superior de la banda 16 = 4312.500000
Frecuencia superior de la banda 17 = 5062.500000
Frecuencia superior de la banda 18 = 5937.500000
Frecuencia superior de la banda 19 = 6937.500000

Con esto, la primera banda (banda 0) va de 1 a 187.5 Hz, la segunda (banda 1) de187.5 a 312.5 Hz, y así sucesivamente hasta la veinteava banda (banda 19) que va, desde los 5937.5 a los 6937.5 Hz.

También hay que tener en cuenta, que el espectro de la voz humana va de los 500 a los 2000 Hz, y, que normalmente, las vocales se corresponden con las menores frecuencias dentro de este rango, es decir, desde los 500 hasta los 1200 Hz aproximadamente (bandas 3-8), y las consonantes se centran en las frecuencias superiores, o sea, de 1200 a 2000 Hz (bandas 9-12).

Todo esto es para saber en que bandas hay que fijarse a la hora de ver los valores de tonalidad de unas u otras señales. Por ello, cuando busquemos resultados para la vocal “a” se deberá mirar en las bandas desde la segunda o tercera hasta la séptima u octava, y, cuando lo que procesamos es la consonante “s”, deberemos centrarnos en las bandas desde la octava hasta la onceava o doceava.

Sin más explicaciones veamos primero los valores de tonalidad procesando la vocal “a”. Se han escogido dos iteraciones cualesquiera de entre todas las que dura el procesamiento de esta vocal:

Iteracion99 TonPred1 0= 0.588559
Iteracion99 TonPred1 1= 0.382257
Iteracion99 TonPred1 2= 0.478690
Iteracion99 TonPred1 3= 0.805318
Iteracion99 TonPred1 4= 0.818140
Iteracion99 TonPred1 5= 0.454851
Iteracion99 TonPred1 6= 0.407794
Iteracion99 TonPred1 7= 0.583950
Iteracion99 TonPred2 8= 0.560019
Iteracion99 TonPred2 9= 0.435861
Iteracion99 TonPred2 10= 0.467012
Iteracion99 TonPred3 11= 0.733846
Iteracion99 TonPred3 12= 0.480941
Iteracion99 TonPred4 13= 0.665175
Iteracion99 TonPred4 14= 0.599124
Iteracion99 TonPred5 15= 0.630581
Iteracion99 TonPred6 16= 0.347960
Iteracion99 TonPred7 17= 0.344312
Iteracion99 TonPred8 18= 0.420590
Iteracion99 TonPred8 19= 0.194254

Iteracion100 TonPred1 0= 0.456207
Iteracion100 TonPred1 1= 0.050667
Iteracion100 TonPred1 2= 0.449890
Iteracion100 TonPred1 3= 0.716214
Iteracion100 TonPred1 4= 0.757232
Iteracion100 TonPred1 5= 0.328727
Iteracion100 TonPred1 6= 0.320706
Iteracion100 TonPred1 7= 0.706522
Iteracion100 TonPred2 8= 0.563827
Iteracion100 TonPred2 9= 0.494617
Iteracion100 TonPred2 10= 0.310099
Iteracion100 TonPred3 11= 0.534547
Iteracion100 TonPred3 12= 0.643495
Iteracion100 TonPred4 13= 0.547957
Iteracion100 TonPred4 14= 0.546697
Iteracion100 TonPred5 15= 0.471394
Iteracion100 TonPred6 16= 0.359194
Iteracion100 TonPred7 17= 0.336235
Iteracion100 TonPred8 18= 0.343597
Iteracion100 TonPred8 19= 0.286996

Se ha marcado en negrita aquellas bandas frecuenciales que dan un valor de tonalidad mayor que el resto para **todas las iteraciones** mientras dura la vocal “a”. Se observa que el valor de tonalidad obtenido para estas bandas es mayor que el resto. Además, se puede comprobar que para todas las iteraciones, no sólo para estas dos, esto se cumple, es decir, los valores de tonalidad mantienen cierto patrón para estas bandas mencionadas.
Esto es, obviamente, porque al pasar a la frecuencia la vocal “a” siempre recaerá en determinadas frecuencias, que son siempre las mismas, intrínsecamente propias de esta vocal. Si no fuera así, se trataría de otro carácter.

En cuanto al resto de bandas, ya en dos iteraciones, que se diferencian temporalmente por sólo 2 mseg, no se observa ningún tipo de patrón, sino más bien aleatoriedad. Quizá no tanto para las bandas 5, 6 y 8, que corresponden también al espectro reservado para las vocales, pero sí para el resto de bandas. En este resto de bandas, los valores de tonalidad no superan, por lo general el 0.5, tal y como se esperaba. De todas formas, hay que decir que las vocales, como señales periódicas que son, presentan armónicos y en ciertos casos, se producen valores atípicos de tonalidad, que podría ser consecuencia de, tanto los armónicos producidos, como de la aleatoriedad debida al ruido, que siempre está presente.

Seguidamente vemos los valores de tonalidad en el momento en que se está procesando la consonante “s” de nuestra señal de audio “asa”:

Iteracion189 TonPred1 0= 0.576929
Iteracion189 TonPred1 1= 0.052418
Iteracion189 TonPred1 2= 0.128483
Iteracion189 TonPred1 3= 0.228689
Iteracion189 TonPred1 4= 0.081992
Iteracion189 TonPred1 5= 0.688301
Iteracion189 TonPred1 6= 0.374517
Iteracion189 TonPred1 7= 0.599479
Iteracion189 TonPred2 8= 0.523044
Iteracion189 TonPred2 9= 0.468676
Iteracion189 TonPred2 10= 0.452585
Iteracion189 TonPred3 11= 0.450081
Iteracion189 TonPred3 12= 0.240833
Iteracion189 TonPred4 13= 0.705889
Iteracion189 TonPred4 14= 0.772339
Iteracion189 TonPred5 15= 0.589600
Iteracion189 TonPred6 16= 0.205573
Iteracion189 TonPred7 17= 0.378197
Iteracion189 TonPred8 18= 0.568962
Iteracion189 TonPred8 19= 0.245491

Iteracion190 TonPred1 0= 0.796604
Iteracion190 TonPred1 1= 0.064838
Iteracion190 TonPred1 2= 0.712791
Iteracion190 TonPred1 3= 0.619867
Iteracion190 TonPred1 4= 0.237520
Iteracion190 TonPred1 5= 0.641422
Iteracion190 TonPred1 6= 0.629865
Iteracion190 TonPred1 7= 0.156899
Iteracion190 TonPred2 8= 0.172699
Iteracion190 TonPred2 9= 0.411393
Iteracion190 TonPred2 10= 0.479995
Iteracion190 TonPred3 11= 0.605855
Iteracion190 TonPred3 12= 0.666808
Iteracion190 TonPred4 13= 0.519253
Iteracion190 TonPred4 14= 0.519862
Iteracion190 TonPred5 15= 0.618678
Iteracion190 TonPred6 16= 0.272194
Iteracion190 TonPred7 17= 0.368991
Iteracion190 TonPred8 18= 0.551538
Iteracion190 TonPred8 19= 0.347081

Se han escogido de nuevo 2 iteraciones al azar y consecutivas. Se han marcado en negrita esta vez las bandas correspondientes al rango de frecuencias correspondientes a las consonantes, que como dijimos iba de 1200 a 2000 Hz aproximadamente.
No se puede apreciar, en estas cuatro bandas, un valor alto de tonalidad, ni mayor a los valores de tonalidad que se obtenían para el caso de la vocal ”a”. Por supuesto, ahora, las bandas antes marcadas en negrita (bandas 3, 4 y 7), no presentan ninguna relación y como se ve, los valores de tonalidad en iteraciones consecutivas han pasado a ser más bien aleatorios.

Pues bien, ya se ha comprobado que el detector de tonalidad funciona correctamente, detectando alta tonalidad donde debería y detectando mayor tonalidad en segmentos de voz vocálicos que en consonánticos.

Pasamos entonces a la segunda parte de este punto, referido a los experimentos realizados para comprobar el buen funcionamiento de la implementación de enmascaramiento temporal.

6.2 Enmascaramiento temporal. Experimentos.

Para demostrar que el modelo de enmascaramiento temporal funciona correctamente se debe, en primer lugar, comprobar que efectivamente, se detecta transiente cuando se presenta éste en la señal de audio, y que, cuando esto sucede, se calcula adecuadamente la función de ensanchamiento temporal.

Para ello, lo que se debía hacer era procesar una señal de audio que presentara algunos de estos transientes y ver si, efectivamente, se detectaban correctamente por el programa. Como señal de audio con transientes vamos a utilizar aquella que ya vimos anteriormente denominada “castanets”, que representa el sonido de unas castañuelas y como vimos contenía numerosos transientes. Veremos cuántas bandas detectan transiente en las tramas alrededor de uno de estos transientes.

Aquí tenemos de nuevo la señal:

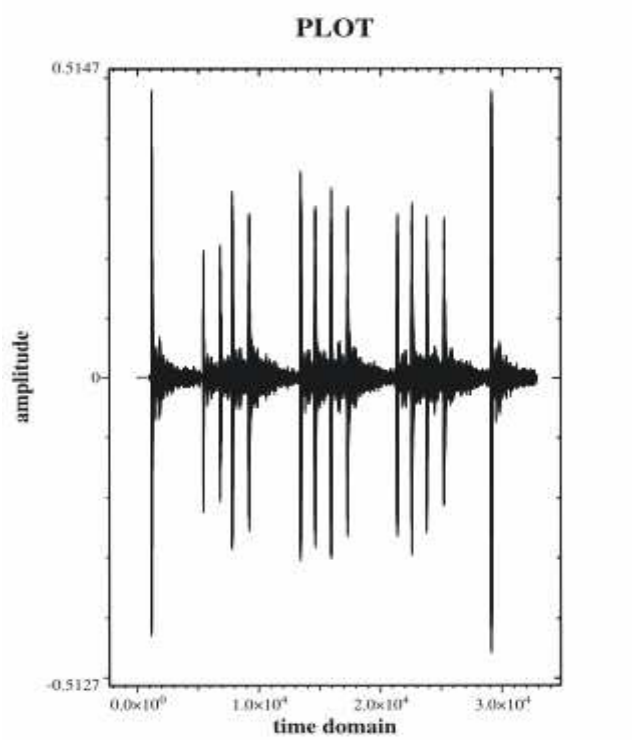


Figura 25. Señal de audio temporal “castanets”

Vemos un gran transiente al principio, luego una serie de transientes de un poco menos de amplitud y otro gran transiente al final. Veamos, por ejemplo que pasa alrededor de uno de esos transientes del medio. El procedimiento consiste en imprimir el vector que guarda si se ha detectado o no transiente en esa banda y trama, es por tanto, un vector de tamaño veinte. Entonces para cada iteración o trama tendremos, para cada banda, un 1 si se ha detectado transiente y un 0 si no. En nuestra señal, exactamente, se produce un transiente en las iteraciones 246 y 247. Veamos ya los resultados.

Iteracion 243
Banda 0 --> 1
Banda 1 --> 1
Banda 2 --> 1
Banda 3 --> 0
Banda 4 --> 0
Banda 5 --> 0
Banda 6 --> 0
Banda 7 --> 0
Banda 8 --> 0
Banda 9 --> 0
Banda 10 --> 0
Banda 11 --> 0
Banda 12 --> 1
Banda 13 --> 0
Banda 14 --> 1
Banda 15 --> 1
Banda 16 --> 0
Banda 17 --> 0
Banda 18 --> 0
Banda 19 --> 0
Numero total de bandas que detectan transiente: **6**

Iteracion 244
Banda 0 --> 0
Banda 1 --> 1
Banda 2 --> 0
Banda 3 --> 0
Banda 4 --> 0
Banda 5 --> 1
Banda 6 --> 1
Banda 7 --> 1
Banda 8 --> 1
Banda 9 --> 1
Banda 10 --> 1
Banda 11 --> 0
Banda 12 --> 0
Banda 13 --> 0
Banda 14 --> 0
Banda 15 --> 0
Banda 16 --> 0
Banda 17 --> 1
Banda 18 --> 0
Banda 19 --> 0
Numero total de bandas que detectan transiente: **8**

Iteracion 245
Banda 0 --> 0
Banda 1 --> 1
Banda 2 --> 0
Banda 3 --> 0
Banda 4 --> 0
Banda 5 --> 1
Banda 6 --> 1
Banda 7 --> 1
Banda 8 --> 1
Banda 9 --> 0
Banda 10 --> 0
Banda 11 --> 0
Banda 12 --> 0
Banda 13 --> 1
Banda 14 --> 0
Banda 15 --> 0
Banda 16 --> 0
Banda 17 --> 1
Banda 18 --> 0
Banda 19 --> 0
Numero total de bandas que detectan transiente: **7**

Iteracion 246
Banda 0 --> 1
Banda 1 --> 1
Banda 2 --> 1
Banda 3 --> 1
Banda 4 --> 1
Banda 5 --> 1
Banda 6 --> 1
Banda 7 --> 1
Banda 8 --> 1
Banda 9 --> 1
Banda 10 --> 1
Banda 11 --> 1
Banda 12 --> 1
Banda 13 --> 1
Banda 14 --> 1
Banda 15 --> 1
Banda 16 --> 1
Banda 17 --> 1
Banda 18 --> 1
Banda 19 --> 1
Numero total de bandas que detectan transiente: **20**

Iteracion 247
Banda 0 --> 0
Banda 1 --> 1
Banda 2 --> 1
Banda 3 --> 1
Banda 4 --> 1
Banda 5 --> 1
Banda 6 --> 1
Banda 7 --> 1
Banda 8 --> 1
Banda 9 --> 1
Banda 10 --> 1
Banda 11 --> 1
Banda 12 --> 1
Banda 13 --> 0
Banda 14 --> 0
Banda 15 --> 0
Banda 16 --> 1
Banda 17 --> 1
Banda 18 --> 1
Banda 19 --> 1
Numero total de bandas que detectan transiente: **16**

Iteracion 248
Banda 0 --> 0
Banda 1 --> 1
Banda 2 --> 0
Banda 3 --> 0
Banda 4 --> 0
Banda 5 --> 0
Banda 6 --> 0
Banda 7 --> 0
Banda 8 --> 0
Banda 9 --> 0
Banda 10 --> 0
Banda 11 --> 0
Banda 12 --> 0
Banda 13 --> 0
Banda 14 --> 0
Banda 15 --> 0
Banda 16 --> 1
Banda 17 --> 0
Banda 18 --> 0
Banda 19 --> 0
Numero total de bandas que detectan transiente: **2**

Iteracion 249
Banda 0 --> 0
Banda 1 --> 1
Banda 2 --> 0
Banda 3 --> 0
Banda 4 --> 0
Banda 5 --> 0
Banda 6 --> 0
Banda 7 --> 0
Banda 8 --> 0
Banda 9 --> 0
Banda 10 --> 0
Banda 11 --> 1
Banda 12 --> 0
Banda 13 --> 0
Banda 14 --> 1
Banda 15 --> 0
Banda 16 --> 1
Banda 17 --> 0
Banda 18 --> 0
Banda 19 --> 0
Numero total de bandas que detectan transiente: **4**

.
.
.

Ahí queda demostrado que se detectan bien los transientes, ya que justamente en las iteraciones 246 y 247, cuando se produce el transiente, es cuando casi todas e incluso todas las bandas, detectan transiente.

Antes y después sólo hay algunas bandas que detectan transiente, cosa que es totalmente normal. En bandas donde no hay transientes tan destacados como el analizado aquí, puede haber transientes menores, y esto puede producir la detección de transientes en algunas bandas.

Ya sólo queda por mostrar, que efectivamente, cuando se detecta transiente, se calcula la función de ensanchamiento temporal. Y así es, más abajo tenemos representada la función de ensanchamiento temporal calculada para aquellas bandas en las que se ha detectado transiente.

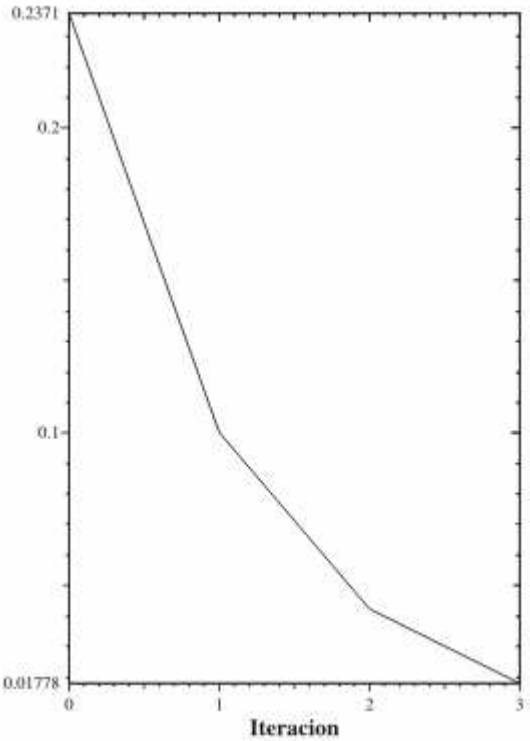


Figura 26. Función de ensanchamiento temporal

6.3 Otras comprobaciones.

En los dos puntos anteriores, se trataba de comprobar que, tanto el detector de tonalidad como el enmascaramiento temporal producían, por separado, resultados coherentes. El objetivo de este último punto del capítulo es comprobar que el programa en su conjunto funciona correctamente. Y es que, para poder probar el algoritmo en pacientes, finalidad última de este proyecto, era necesario hacer una última comprobación, que a continuación se explica.

Como ya dijimos, el algoritmo que se ha programado en este proyecto, deriva de la estrategia PACE (Psychoacoustic Advanced Combinational Encoder), y ésta a su vez, de la ACE (Adavanced Combinational Encoder). La diferencia entre la ACE y la PACE radica en la implementación del modelo psicoacústico. Y, entre la PACE y la nueva versión mejorada que llamaremos TPACE, justamente, en la implementación de enmascaramiento temporal y el detector de tonalidad. Así pues, había que comprobar que esta diferencia no perjudicaba en nada a los resultados que se obtenían normalmente con la estrategia PACE o ACE. Dicho de otro modo, se trata de comprobar que los resultados de la estrategia PACE o ACE y de la versión mejorada TPACE coinciden. Por supuesto, para unas mismas entradas y bajo unas mismas condiciones de ejecución.

Los requisitos para poder efectuar la citada comparación de una manera fiel, son, como hemos dicho: mismas entradas y mismas condiciones de ejecución.

El requisito de tener entradas iguales, significa, que deberemos procesar, obviamente, el mismo fichero de audio, pero además, deberemos tener la misma frecuencia de estimulación de canal y un igual número de bandas a seleccionar para ambas estrategias. Éstas son las tres únicas entradas o parámetros de entrada que solicita la estrategia original ACE, y algunas de las que solicita la TPACE:

- Nombre del archivo de audio
- Frecuencia de estimulación del canal
- Nº de bandas a seleccionar

La estrategia TPACE, además de estas entradas, que serán estrictamente idénticas para ambas estrategias a la hora de compararlas, necesita otras más:

- Parámetros de la función de ensanchamiento frecuencial:
    - Valor de pico
    - Pendiente de bajada (izquierda)
    - Pendiente de bajada (derecha)
    - Valor mínimo
- Parámetros de la función de ensanchamiento temporal:
    - Valor de pico

- Pendiente de bajada (derecha)
- Valor mínimo

Estos parámetros de entrada correspondientes a la estrategia TPACE, determinan las condiciones de ejecución, segundo requisito. Decíamos que los algoritmos debían compararse bajo las mismas condiciones de ejecución.

Así, estos siete parámetros adicionales que necesita la estrategia TPACE respecto de la ACE, han de ser modificados, de tal forma que, suponiendo que no haya ningún fallo de programación en la estrategia TPACE, se obtengan idénticos resultados tras la ejecución de ambas estrategias ACE y TPACE. Es decir, se obtendrán, para todas y cada una de las tramas de que se compone la señal de audio, las mismas bandas seleccionadas con sus respectivos valores de amplitud, para ambas estrategias. Sólo así se podrá asegurar que no existe fallo alguno de programación en la nueva estrategia TPACE.

La forma en la que han de ser modificadas las entradas de la TPACE para conseguir unas mismas condiciones de ejecución respecto de la ACE se describe a continuación.

1) Se intenta omitir de la estrategia TPACE todo aquello que la haga diferente de la ACE. Para ello, todos los parámetros de la función de ensanchamiento frecuencial se han de poner a cero, ya que la estrategia ACE no tiene modelo psicoacústico:

- Valor de pico = 0
- Pendiente de bajada (izquierda) = 0
- Pendiente de bajada (derecha) = 0
- Valor mínimo = 0

Con estos valores, se desactiva, como si dijéramos, el modelo psicoacústico original de la estrategia PACE.

2) En cuanto a la función de ensanchamiento temporal, se puede activar o desactivar su funcionamiento, al inicio de la ejecución del programa, como parámetro booleano de entrada. Así pues, desactivaremos el funcionamiento del enmascaramiento temporal.

Realizados estos dos puntos, sólo quedaba introducir los parámetros de entrada adecuados e iguales para las dos estrategias y comparar los resultados.

Valores parámetros de entrada:

- Nombre del archivo de audio = “prueba.au”
- Frecuencia de estimulación del canal = 500 Khz
- Nº de bandas a seleccionar = 8

Teniendo en cuenta todo lo anterior, se procedió a comparar los resultados, con la sorpresa de que éstos no coincidían. Al final se cayó en la cuenta del posible motivo por el cual no coincidían los resultados de las dos estrategias. Nos habíamos olvidado del detector de tonalidad. Recordemos cual era su función.

El detector de tonalidad, a grandes rasgos, calculaba un valor de pico. Este valor de pico, es uno de los parámetros que definen la función de ensanchamiento frecuencial.

Dado que se seguía calculando este valor de pico durante la ejecución del programa por el detector de tonalidad, aunque se pusieran a cero todos los parámetros de la función de ensanchamiento frecuencial en la inicialización del mismo, el programa usaba el valor de pico calculado en tiempo de ejecución por el detector, dando como resultado la no coincidencia de resultados anteriormente mencionados. ¿Solución? En el mismo código del programa, igualar, al final de la función que implementa el detector de tonalidad, el valor de pico a cero, sobrescribiendo el calculado por el detector de tonalidad, que no nos interesaba en este caso.

Se pudo comprobar entonces, la total semejanza de resultados entre las estrategias ACE y TPACE, para unas mismas entradas y bajo las mismas condiciones de ejecución, rechazando así, un fallo en la programación, y consecuentemente, que los diferentes resultados que la estrategia TPACE pudiera dar respecto a las otras dos originales, ya en condiciones normales de simulación (no para comparar resultados), iban a ser producidos, exclusivamente, por el detector de tonalidad y por el enmascaramiento temporal, instrumentos de este proyecto, que como se pudo comprobar anteriormente, presentan un funcionamiento correcto.

Asimismo, al concluir la igualdad de resultados entre las estrategias ACE y TPACE, ya no hacía falta realizar la misma comprobación entre las estrategias PACE y TPACE, ya que ambas derivan de la ACE.

Terminado este capítulo, pasamos ya a los resultados obtenidos en pacientes.

7 RESULTADOS

7.1 Resultados objetivos

En primer lugar y antes de ver los resultados obtenidos en pacientes, comparemos brevemente las bandas seleccionadas para las estrategias ACE y TPACE. Se procesará la señal castanets vista anteriormente.

En la figura 27 vemos las bandas que selecciona cada estrategia. Se ha escogido al azar la iteración 100.

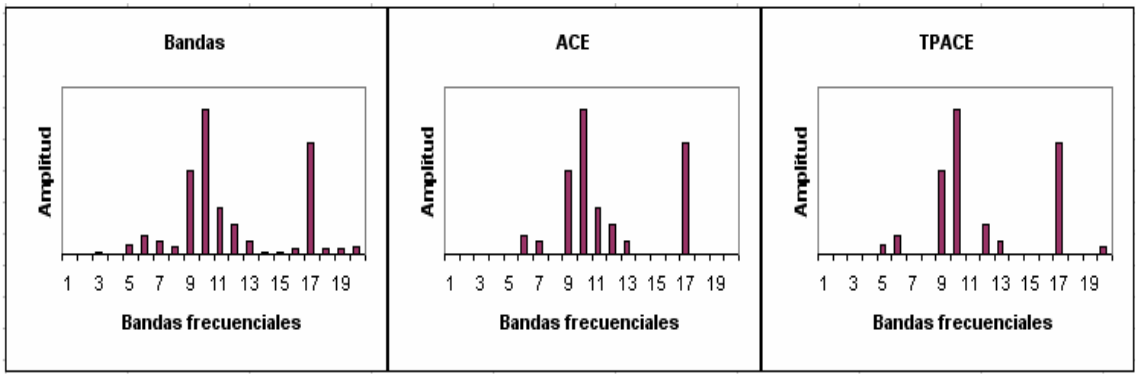


Figura 27. Bandas seleccionadas ACE vs TPACE

La primera diferencia apreciable es que las bandas seleccionadas por la TPACE se encuentran más separadas entre si. Esto supone la ventaja, al igual que sucedía en la PACE, de una **menor interacción** entre canales. También se puede apreciar que las bandas seleccionadas por la TPACE tienen una menor amplitud. Esto también es favorable ya que, como dijimos anteriormente, los implantes cocleares funcionan con baterías, y una menor amplitud equivale a una menor intensidad de estimulación, que significa un menor consumo de las mencionadas baterías (**ahorro de energía**).

Así, estos primeros resultados objetivos son favorables.

7.2 Tests subjetivos con pacientes

En el Centro de Audición (Hörzentrum) de la Universidad de Medicina de Hannover, los nuevos algoritmos son implementados y probados en pacientes con implantes cocleares. Bajo un entorno Matlab, a los pacientes se les somete a diversas pruebas para medir la posible mejora en la percepción del habla que puedan experimentar con los nuevos algoritmos implementados.

Estas pruebas se realizan mediante un hardware específico desarrollado por la Universidad de Hannover. Entre las diversas pruebas que se pueden realizar tenemos las realizadas “in quiet” (en silencio), es decir, con ausencia total de ruido; y las efectuadas en condiciones ruidosas, que supone una mayor dificultad para el paciente, aunque también unas condiciones más próximas a la realidad.

En el caso de este proyecto y para medir el grado de mejora, se decidió que era suficiente probarlo con dos pacientes. Y dado que se trataba de los primeros experimentos llevados a cabo con el nuevo algoritmo, se decidió hacerlo bajo condiciones no ruidosas. Tras la obtención de resultados positivos en el marco de este proyecto, sería recomendable aprovechar su potencial de desarrollo y proseguir la investigación con una más extensa experimentación, con más pacientes, y otras diversas condiciones.

Los experimentos específicos realizados a estos dos pacientes, consistieron básicamente en presentar al paciente una serie de listas de palabras. Cada una de estas listas se compone de 20 palabras. Los pacientes oirán las distintas palabras de cada serie procesadas independientemente por la estrategia ACE y por la estrategia TPACE. De este modo, y conforme a las palabras que el paciente identifique correctamente, se podrá medir el grado de mejora que supone la nueva estrategia a testar (en nuestro caso la TPACE), respecto a la ACE, estrategia que como hemos dicho se usa actualmente en personas con implantes cocleares.

Veamos ya los resultados obtenidos para el primer paciente:

Paciente 1:

	Estrategia ACE	Estrategia TPACE
Lista 1	78.3	73.18
Lista 2	74.52	72.32
Lista 3	68.86	79.23
Lista 4	74.52	70.89
Media (%)	**74.05%**	**73.905%**
Desviación	3.88	3.23

Figura 27. Resultados paciente 1

En el caso de este paciente se han procesado cuatro listas de palabras. Se puede apreciar que los resultados están muy igualados, obteniéndose un 74.05 % de palabras entendidas con la estrategia ACE y un 73.90 % para la estrategia TPACE, con unas desviaciones de 3.88 y 3.23 respectivamente.

Seguidamente, se muestran los resultados del segundo paciente:

Paciente 2:

	Estrategia ACE	Estrategia TPACE
Lista 1	55.5	66
Lista 2	43.26	60
Lista 3	65.25	74.2
Media (%)	**54.67%**	**66.73%**
Desviación	8.99	5.82

Figura 28. Resultados paciente 2

Como vemos, para esta paciente se han procesado tres listas de palabras, obteniéndose un 54.67 % de palabras entendidas para la estrategia ACE y un 66.73 % para la estrategia TPACE. Asimismo, la desviación resulta 8.99 y 5.82, respectivamente.

Por consiguiente, los resultados globales calculados, al hacer la media entre los resultados de los dos pacientes, y en ausencia de ruido (in quiet) son:

Experimento 1		Media	Mejora
ACE		**64.36%**	
TPACE		**70.319%**	***+ 9.26%***

Figura 29. Resultados globales del experimento

7.3 Análisis y conclusión de los resultados

Analicemos en primer lugar los resultados del primer paciente.

	Estrategia ACE	Estrategia TPACE
Lista 1	78.3	73.18
Lista 2	74.52	72.32
Lista 3	68.86	79.23
Lista 4	74.52	70.89
Media (%)	**74.05%**	**73.905%**
Desviación	3.88	3.23

Llama la atención el alto grado de entendimiento de este paciente. En torno al 74% de palabras entendidas con ambas estrategias. Al coincidir prácticamente el número de palabras entendidas, la única mejora apreciable es la disminución de la desviación estándar, lo cual indica que los resultados en la estrategia TPACE son algo más homogéneos, presentando menor dispersión.

Para el segundo paciente en cambio, los resultados son mucho más alentadores.

	Estrategia ACE	Estrategia TPACE
Lista 1	55.5	66
Lista 2	43.26	60
Lista 3	65.25	74.2
Media (%)	**54.67%**	**66.73%**
Desviación	8.99	5.82

Se aprecia la gran mejoría que presenta esta paciente, pasando del 54% de palabras entendidas con la estrategia ACE, a un 66% con la TPACE. Esto significa una mejora de un 22% en la estrategia TPACE respecto a la ACE. Además, teniendo en cuenta que la dispersión vuelve a disminuir, ésta vez, significativamente, podemos concluir que los resultados obtenidos con esta paciente han sido muy satisfactorios.

Para concluir veremos los resultados globales de ambos pacientes. Hallando la media del porcentaje obtenido para cada paciente con cada una de las dos estrategias, se obtuvo la siguiente tabla:

Experimento 1		Media	Mejora
ACE		**64.36%**	
TPACE		**70.319%**	***+ 9.26%***

Se observa que en conjunto se obtiene una mejora. Aumenta en un 9.26% el entendimiento o percepción del habla de los pacientes, usando la estrategia TPACE respecto de la ACE. Además la dispersión también disminuye.

8 DISCUSIÓN

La idea de este proyecto surge de la necesidad de mejorar e innovar nuevas técnicas de procesamiento de señal para mejorar la calidad de vida de personas con sordera. Se decide pues, mejorar una de las estrategias más actuales y con mejores resultados hoy en día. Diseñada en Laboratorio de tecnologías de la información de la Universidad de Hannover denominada PACE. [6]. Esta estrategia esta basada en la idea utilizada en el mp3 en que sólo las componentes más audibles o más importantes para la percepción auditiva son enviadas a los electrodos que posteriormente se utilizan para estimular el nervio auditivo. La decisión sobre que bandas se deben seleccionar recae en un modelo psicoacústico que modela el conocido fenómeno de enmascaramiento frecuencial que tiene lugar en el sistema auditivo humano [6]. La mejora recae sobre el modelo psicoacústico implementado en esta estrategia. Dos son los puntos clave, la adición de enmascaramiento temporal y la implementación del detector de tonalidad [1], [2].

Se comprobó en el capítulo de experimentos el correcto funcionamiento de ambas propuestas, lo que indicaba que la mejora era factible. Además, y lo más importante, se han obtenido buenos resultados al probar el algoritmo en pacientes. Los resultados, utilizando tests especialmente diseñados para medir la inteligibilidad del habla en personas implantadas, han mostrado que la nueva estrategia TPACE presenta un aumento de la inteligibilidad del 9.26% en comparación con la estrategia comercial ACE que utilizan los pacientes diariamente. Hay que remarcar que estos resultados son sólo resultados preliminares realizados con dos pacientes hasta la fecha.

Por otra parte, hay que destacar otros puntos a favor del método implementado no menos importantes que vimos en el capítulo 7 apartado 1 de experimentos objetivos. Se produce, por lo general, una menor interacción entre canales, ya que los canales seleccionados por la TPACE están más separados entre sí. El otro punto a favor es el ahorro de energía. Estos cuasi-objetivos son muy importantes pues proporcionan mayor fiabilidad al implante, y aumentarán aún más su posible preferencia de uso.

Sin duda, hubiera sido deseable experimentar más y con más pacientes, pero hay que destacar la dificultad que conlleva realizar tests con pacientes. Entre otras cosas, se ha estado con cada uno de ellos aproximadamente 2 horas. Por ello, hay que asegurar que el algoritmo que se quiere experimentar en el paciente, funciona bien en todos sus aspectos, realizando antes una estricta verificación del código y comprobación de resultados.

Ya para terminar, y en lo que concierne a este proyecto, en definitiva podemos afirmar que la introducción del modelo psicoacústico ha contribuido a una mejora de la inteligibilidad en pacientes con implantes cocleares. Éste era el objetivo final del proyecto, y se ha alcanzado. Por tanto, se abre la puerta a nuevos experimentos y a nuevas mejoras. Experimentos en condiciones ruidosas o el perfeccionamiento del enmascaramiento temporal, introduciendo pre-enmascaramiento además de post-enmascaramiento, deberían de plantearse y desarrollarse en un futuro, para intentar acercarnos cada vez más a la forma de funcionar del cerebro junto con el sistema auditivo humano**.**

9 BIBLIOGRAFÍA

[1] Libro “Introduction to Digital Audio Coding and Standards

[2] Publicación “Tonality and its Aplication to Perceptual-Based Speech Enhancement”, Jessica A. Rossi-Katz & Ajay Natarajan.

[3] “MPEG-4 CELP Coding”, Masayuki Nishiguchi y Bernd Edler.

[4] Publicación “An Experimental High Fidelity Perceptual Audio Coder”, Bosse Lincoln. 1998.

[5] Libro “General Audio Coding”, Jürgen Herre and Heiko Purnhagen.

[6] Publicación “A Psychoacoustic “NofM”-type Speech Coding Strategy for Cochlear Implants”, Waldo Nogueira, Andreas Büchner, Thomas Lenarz, Bernd Edler. 2004.

[7] B. S. Wilson, C. C. Finley, D. T. Lawson, R. D. Wolford, D. K. Eddington, and W.M. Rabinowitz, “Better speech recognition with cochlear implants,” *Nature*, vol. 352, no. 6332, pp. 236–238, 1991.

[8] “ACE Speech Coding Strategy,” *Nucleus Technical Reference Manual*, Z43470 Issue 3, Cochlear Corporation, Lane Cove, New SouthWales, Australia, December 2002.

[9] P. C. Loizou, “Signal-processing techniques for cochlear implants,” *IEEE Eng. Med. Biol. Mag.*, vol. 18, no. 3, pp. 34–46, 1999.

[10] http://members.fortunecity.com/alex1944/mp3coding/analisis.html#

[11] http://citeseer.ist.psu.edu/context/331900/0

[12] http://www.aes.org/e-lib/browse.cfm?elib=13231

[13] http://www.hwupgrade.com/audio/diamond_rio/index2.html

[14] PERCEPTUAL SPEECH CODING USING TIME AND FREQUENCY MASKING CONSTRAINTS. Benito Carnero, Andrzej Drygajlol. Signal Processing Laboratory, Swiss Federal Institute of Technology of Lausanne, CH-1015 Lausanne, Switzerland