Efectos de la compresión mp3 en la determinación del pitch

J. M. Garin† P. D. Agüero† J. C. Tulli† E. L. González†

†*Laboratorio de Comunicaciones, Universidad Nacional de Mar del Plata, Mar del Plata, Argentina*
jmgarin@fi.mdp.edu.ar

***Resumen*— En el desarrollo de una plataforma para ayudar en las prácticas de oralización de personas con discapacidad auditiva, la compresión de voz es un aspecto importante. La misma es necesaria para permitir tanto a especialistas como ingenieros recibir la señal de voz generada durante las prácticas usando Internet para diagnósticos y mejoras de la plataforma. En este artículo se explora la robustez de los algoritmos de extracción de frecuencia fundamental en condiciones de ruido y compresión de señal, debido a que la frecuencia fundamental es uno de los parámetros analizados durante las prácticas. Los resultados experimentales muestran que la compresión MP3 es una opción adecuada debido al pequeño incremento en los errores de extracción de frecuencia fundamental.**

***Palabras clave*— frecuencia fundamental, compresión mp3, métodos de autocorrelación, producto armónico espectral, transformada cepstrum**

1. INTRODUCCION

El pitch (característica acústica perceptual) o frecuencia fundamental (característica acústica física) es un parámetro muy importante en el análisis de las señales de voz, ya que permite obtener la frecuencia a la cual vibra la glotis [4]. Este parámetro acústico es utilizado en la mayoría de las aplicaciones de procesamiento de voz, tales como análisis del habla, codificación, reconocimiento y verificación del locutor, análisis de voces patológicas, etc. En consecuencia, es un área de gran interés en la actualidad.

La detección del pitch es una tarea difícil debido a la existencia de problemas relacionados con la calidad del audio, ocasionados por el nivel de ruido de la señal y la compresión, y otros mas graves como es el caso de los problemas de entonación presentes en las voces patológicas.

En la actualidad existen diversos algoritmos de compresión, que se pueden clasificar en dos grandes grupos: algoritmos de compresión con pérdidas y algoritmos de compresión sin pérdidas. Entre los primeros se encuentran aquellos que permiten una cierta degradación de la calidad de la señal para lograr mayores compresiones: MP2, MP3, AAC, WMA, ADPCM, etc. Entre los algoritmos que no degradan la señal y permiten recuperarla en su forma original se encuentran FLAC [9], optimFROG, Shorten, Tak, entre otros. En estos casos los niveles de compresión no superan 1:3.

En este trabajo se aborda el estudio de la influencia de la compresión MP3 en la precisión de la extracción del pitch. El objetivo final es la utilización de la compresión de audio para transmitir en forma mas compacta la señal de voz en un sistema de práctica de la oralización de personas sordas e hipoacúsicas [3]. Los algoritmos de extracción de pitch estudiados abarcan varios dominios: temporal, frecuencial y quefrecuencial (cepstral).

Existen estudios similares acerca de la influencia de la compresión en el análisis de la voz, como es el caso del artículo de Euler y Zinke [2] sobre los efectos de la codificación de la señal usando 16kBit/s CELP, 13kBit/s RPE-LTP y 4.8kBit/s CELP en el rendimiento de un sistema de reconocimiento de palabras aisladas independiente del locutor y en un sistema de verificación del locutor.

En la Sección 2 se explican los distintos algoritmos de extracción de pitch estudiados. Luego, en la Sección 3 se detallan las condiciones experimentales y se analizarán los resultados. Finalmente, en la Sección 4 se encuentran las conclusiones y direcciones futuras.

2. METODOS

Tal como se mencionó en la introducción, en este artículo se estudia la influencia de la compresión en la precisión de la extracción del pitch de diferentes algoritmos.

Los métodos bajo estudio serán los siguientes: autocorrelación (Sección 2.1), autocorrelación optimizada (Sección 2.2), producto armónico (Sección 2.3), cepstrum (Sección 2.4) y error absoluto en la envolvente (Sección 2.5). En cada uno de ellos se ha trabajado con una frecuencia de muestreo de 16KHz.

En el proceso de extracción de pitch es común la aparición de errores de estimación que resultan en un valor detectado que es el doble (*pitch doubling*) o mitad (*pitch halving*) del apropiado. Para resolver este problema se decidió utilizar un algoritmo de programación dinámica que minimice la posibilidad de aparición de tales errores (Sección 2.6).

2.1. Autocorrelación

La autocorrelación se define como la correlación de un vector de datos consigo mismo, lo cual se puede caracterizar a través de la Ec. 1. El pitch se puede determinar como el período de tiempo comprendido entre el máximo principal $R(0)$ y el primer máximo secundario [10].

$$R(\tau) = \frac{1}{L}\sum_{j=1}^{L} S_j S_{j-\tau} \quad (1)$$

La implementación de este algoritmo con una señal digitalizada requiere asegurarse la presencia de un mínimo de 3 periodos de señal dentro del fragmento analizado, lo cual corresponde a 600 muestras (37,5ms) para una frecuencia de muestreo de 16KHz, considerando una frecuencia fundamental mínima de 80Hz. Luego de separado el intervalo de señal, se realiza la operación de autocorrelación descrita en la Ec. 1, obteniendo resultados como el mostrado en la Fig. 1.

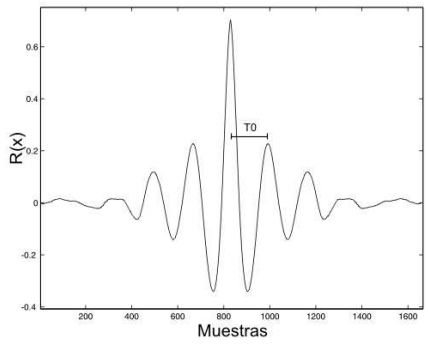


Figura 1: Autocorrelación

Una vez obtenida la autocorrelación se determina la distancia existente entre el máximo principal y el primer máximo secundario, calculando así el valor del periodo de pitch T_0.

En algunos casos los resultados obtenidos no se corresponden a la Fig. 1 sino que tienen la forma mostrada en la Fig. 2, en la cual existe un máximo secundario que enmascara al verdadero, generando así una determinación errónea del pitch. Este problema será abordado en la Sección 2.6.

2.2. Autocorrelación Optimizada

El algoritmo implementado para la autocorrelación optimizada es similar al de la Sección 2.1 pero con la diferencia que realiza un refinamiento de los valores estimados de frecuencia fundamental considerando valores no enteros de pitch.

El refinamiento planteado por Yohav Medan [6] propone la búsqueda de un valor racional entre el valor de pitch entero hallado por el algoritmo de autocorrelación y los valores enteros adyacentes.

Este método realiza este cálculo tomando un intervalo de la señal y el siguiente, con un ancho determinado por el valor del período entero, obtenido por el método de la autocorrelación (Sección 2.1). Con dichos vectores, a los que llamaremos X e Y respectivamente, se obtiene el valor de la proyección ortogonal β determinado por la Ec. 2.

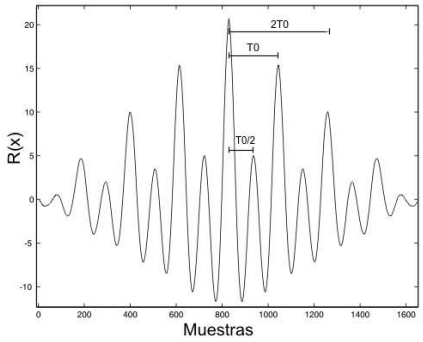


Figura 2: Autocorrelación

$$\beta = \frac{A - B}{C + D} \quad (2)$$

con
$A = (X(i_0), Y(i_0 + 1))|Y(i_0)|^2$
$B = (Y(i_0), Y(i_0))(X(i_0), Y(i_0 + 1))$
$C = (X(i_0), Y(i_0 + 1))[|Y(i_0)|^2 (Y(i_0), Y(i_0 + 1))]$
$D = (X(i_0), Y(i_0))[|Y(i_0 + 1)|^2 (Y(i_0), Y(i_0 + 1))]$
y donde $(X(i_0), Y(i_0))$ es el producto interno entre $X(i_0)$ e $Y(i_0)$.

β es el factor de corrección necesario para ajustar el valor de T_0 entero al valor racional apropiado. Si el valor de β obtenido fuera mayor que uno o menor que cero, se debe recalcular hasta llegar a cumplir la condición $0 \leq \beta \leq 1$ reajustando la cantidad de muestras de los vectores X e Y.

2.3. Producto Armónico

La utilización de técnicas espectrales permite determinar directamente, sobre el eje de frecuencias, los valores máximos correspondientes a las armónicas principales de la señal. En este caso, se pretende determinar la frecuencia fundamental de la señal en determinados sectores [1] . Para ello se extraen de la señal original vectores de datos $s(k)$, a los cuales se les realiza una FFT.

La señal resultante $s(k)$ esta dada por la Ec. 3, siendo $m = 512$ muestras y f la señal original.

$$s(k) = f(k...k + m) \quad (3)$$

Una vez realizada la FFT, la resolución en este dominio para el cálculo de la frecuencia fundamental es menor que la disponible en el dominio temporal para valores bajos de pitch. Por ejemplo, para un valor de pitch de $f_0 = 80$Hz si el algoritmo opera en el dominio temporal existe una resolución de 0,4Hz, mientras que ésta se reduce a 31,25Hz para cualquier valor de

frecuencia fundamental si se trabaja en el dominio frecuencial. El peor caso de resolución de pitch para algoritmos que operan en el dominio del tiempo ocurre para altos valores de frecuencia fundamental, resultando de 5,6Hz para $f_0 = 300$Hz.

Para resolver este inconveniente se agregan ceros a la señal hasta lograr un vector de 40000 muestras, lo que proporciona una resolución teórica de 0,4Hz para todos los valores de frecuencia, lo cual es un error aceptable para la frecuencia mas baja (80Hz).

Una vez calculada la FFT, se generan seis vectores a partir del vector original, tal como lo indica la Ec. 4, con i=1..6 y V(k)=FFT(s(k)). Obteniéndose así los seis espectros mostrados en la Fig. 3.

$$V_i(k) = V(k.i) \tag{4}$$

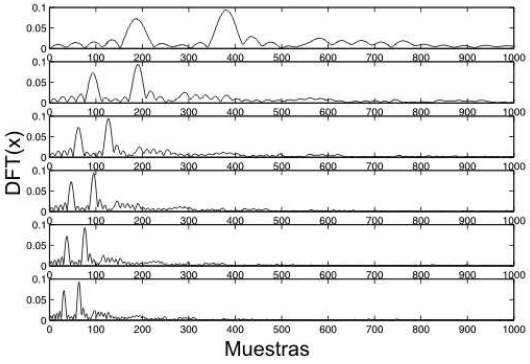


Figura 3: Espectros de los diferentes armónicos

Luego, con estos seis espectros, se realiza la operación descrita por la Ec. 5, obteniéndose como resultado la señal de la Fig. 4.

$$P(k) = \prod_{i=1}^{6} V_i(k) \tag{5}$$

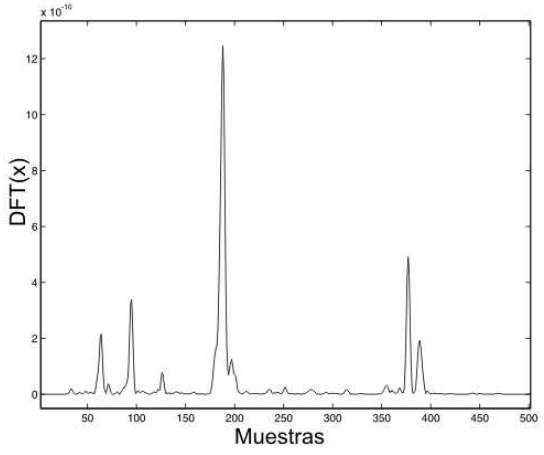


Figura 4: P(k)

De esta última gráfica se extrae el valor máximo M_i determinado en número de muestras. Por último, se realiza la operación de la Ec. 6 y se obtiene el valor de pitch f_0, donde $N_{muestras} = 40000$ y F_s es la frecuencia de muestreo.

$$f_0 = \frac{M_i.F_s}{N_{muestras}} \tag{6}$$

2.4. Cepstrum

Este método de extracción de pitch se basa en la utilización de la transformada cepstrum [11]. Para poder realizar este proceso se asume que la señal de voz $f(t)$ es el resultado de la convolución de la respuesta al impulso del tracto vocal $h(t)$ con la señal emitida por la glotis $s(t)$.

$$f(t) = h(t) * s(t) \tag{7}$$

El objetivo de este método es deconvolucionar la señal $f(t)$ y así obtener $s(t)$. Para lograrlo se trabaja en el dominio frecuencial, siendo $F(w)$ la transformada de $f(t)$.

$$F(w) = H(w).S(w) \tag{8}$$

Para realizar esta separación, se precisa calcular la FFT del logaritmo de $F(w)$

$$FFT(log|F(w)|) = FFT(log|H(w).S(w)|) = \tag{9}$$

$$= FFT(log|H(w)| + log|S(w)|) =$$

$$= FFT(log|H(w)|) + FFT(log|S(w)|)$$

En tanto podemos decir entonces que la transformada cepstrum es:

$$C = FFT(log|F(w)|) \tag{10}$$

Aplicando esta transformada a 512 muestras de una señal de audio se obtiene como resultado la señal mostrada en la Fig. 5.

Para determinar el valor del pitch se extrae el índice en donde se encuentra el valor máximo q, que esta dado en $quefrency$. Mediante la Ec. 11 se lo convierte en el valor de frecuencia fundamental, donde F_s es la frecuencia de muestreo.

$$f_0 = \frac{F_s}{q-1} \tag{11}$$

2.5. Error Absoluto envolvente

Este es otro método propuesto por Yohav Medan [6] que plantea el cálculo del error cuadrático medio que existe entre una cierta cantidad de muestras de la señal y las siguientes.

Para realizar esta operación se extrae un vector $X(t)$ de 512 muestras de la señal, y otro $Y(t)$ consecutivo

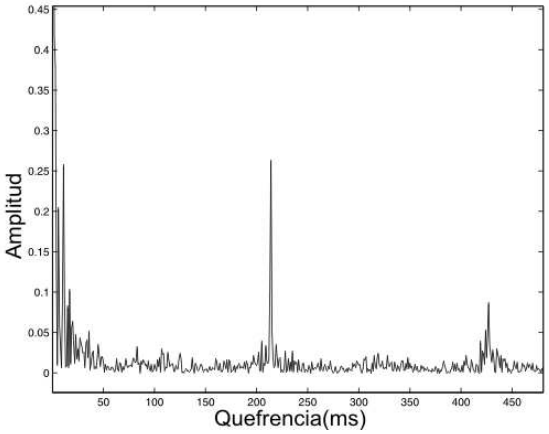


Figura 5: Cepstrum

al anterior de la misma longitud. Variando la longitud del primer vector, y en consecuencia también la del segundo, se intenta encontrar el valor de longitud que indique que el vector $Y(t)$ es el siguiente período de $X(t)$.

Para maximizar la similitud entre un período y el siguiente se hace una corrección del valor de la envolvente, minimizando el siguiente error e:

$$e =< (X_i - a.Y_i)^2 > \tag{12}$$

de donde resulta el valor óptimo de a:

$$a = \frac{(X_i, Y_i)}{|Y_i|^2} \tag{13}$$

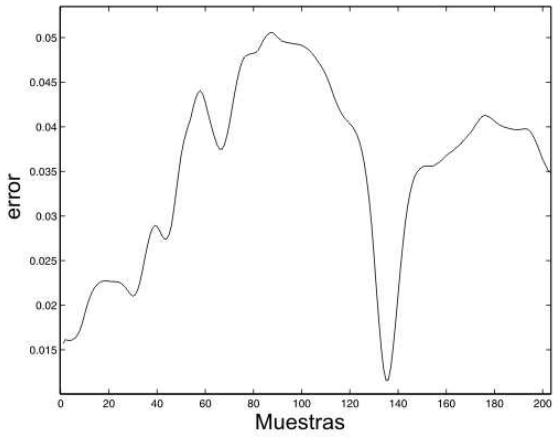


Figura 6: Error Absoluto envolvente

Una vez obtenido el valor de longitud de $X(t)$ para el cual el error es mínimo, se obtiene el valor de pitch tal como se indicó en el algoritmo de la Sección 2.1. Un ejemplo de la utilización de este algoritmo se observa en la Fig. 6.

2.6. Viterbi

Como se mencionó al comienzo de la Sección 2, uno de los problemas principales de los algoritmos de extracción de pitch es la detección de valores doble o mitad con respecto al valor correcto.

Para solucionar este inconveniente se ha decidido utilizar un algoritmo de programación dinámica (Viterbi) para encontrar la secuencia de valores de pitch adecuados siguiendo una función de optimización [8].

Para cada valor de pitch encontrado se supondrán dos valores de pitch adicionales candidatos, que corresponden a la frecuencia fundamental de la octava siguiente y la anterior.

La función de optimización utilizada se compone de dos factores. El primero de ellos es el error obtenido en el algoritmo de detección, el cual se pretende minimizar. En aquellos algoritmos de detección que proporcionan valores máximos para el pitch correcto (por ejemplo, el método de autocorrelación), se ha realizado un ajuste para convertir este máximo en un mínimo ($e_{ac} = 1 - \frac{R(T_0)}{R(0)}$).

El segundo factor que debe ser tenido en cuenta es la derivada de la f_0 en cada instante de tiempo evaluado. Dicho valor también debe ser minimizado para evitar grandes saltos de frecuencia fundamental que no son posibles de realizar por limitaciones fisiológicas.

En consecuencia, la función a optimizar es la descripta en la Ec. 14, siendo $e_i^{j_i}$ el valor de error retornado por el algoritmo de detección para el instante i-ésimo y el candidato de pitch j-ésimo de dicho instante. $|\frac{f_0{}_i^{j_i} - f_0{}_i^{j_{i-1}}}{\Delta t}|$ es la derivada de la frecuencia fundamental con respecto al tiempo, de acuerdo a los valores de pitch candidatos j_i y j_{i-1} seleccionados en los instantes i e $i-1$.

$$argmin_j \sum_i |e_i^{j_i}| + |\frac{f_0{}_i^{j_i} - f_0{}_i^{j_{i-1}}}{\Delta t}| \tag{14}$$

A través del siguiente ejemplo se puede observar como el algoritmo elegiría el camino óptimo $j = \{2, 1, 2\}$, cuyo error mínimo acumulado final es: $e_{min} = e_1^2 + e_2^1 + |\frac{f_0{}_1^2 - f_0{}_2^1}{\Delta t}| + e_3^2 + |\frac{f_0{}_2^1 - f_0{}_3^2}{\Delta t}|$

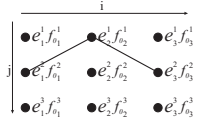


Figura 7: Ejemplo de la secuencia óptima

3. IMPLEMENTACION

La necesidad del análisis, de la robustez de los algoritmos de extracción de pitch a la compresión del audio, surge durante el desarrollo de un software de entrenamiento para personas hipoacúsicas [3], el cual necesita enviar los archivos de sonido de cada sesión a un servidor remoto. Estos archivos podrán ser analizados tanto por los desarrolladores del proyecto como por los profesionales a cargo, permitiéndoles hacer un seguimiento de la evolución del usuario.

El algoritmo de compresión utilizado en este trabajo es el MP3 [7], el cual es un algoritmo de compresión con pérdidas que aprovecha las limitaciones del oído humano medio y elimina toda aquella información que no es capaz de percibir este.

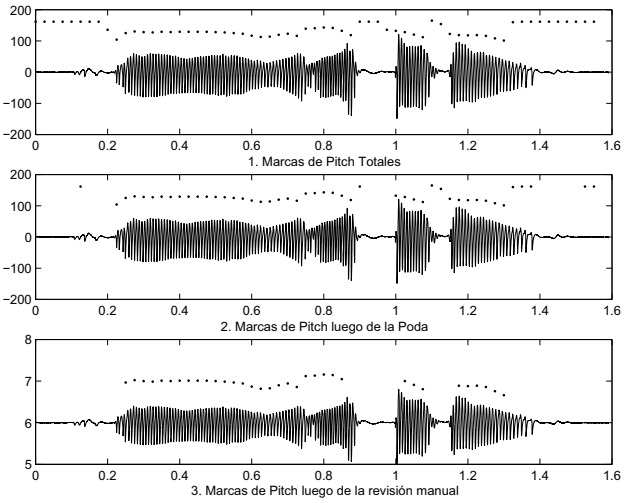


Figura 8: Proceso de depuración de los valores de pitch de la base de datos usados como referencia

3.1. Base de datos

Para realizar los experimentos se utilizaron archivos de la base de datos CMU ARCTIC[5] (disponibles en forma gratuita), generados por el Instituto de Tecnologías del Lenguaje de la Universidad de Carnegie Mellon (USA). Esta compuesta por archivos de audio de un solo locutor en idioma inglés, y consiste en 1150 oraciones balanceadas fonéticamente. Estas fueron grabadas usando una frecuencia de muestreo de 32KHz en estudios con muy buenas condiciones acústicas.

Los archivos están en formato estéreo: un canal con la señal de voz y otro con la señal de un electroglotógrafo. Este último canal es muy útil ya que puede ser utilizado para encontrar valores de pitch de referencia, los cuales se utilizarán para comparar con los valores de pitch detectados en la señal de voz y para establecer el error de estimación.

Cada señal del electroglotógrafo se analizó a través del método de detección de pitch que se basa en la autocorrelación a intervalos prefijados de tiempo. Luego, se eliminaron los instantes de tiempo con valores de energía bajos correspondientes a silencios.

Por último, se verificó cada archivo manualmente y se eliminaron los instantes de tiempo con valores de pitch espúreos, que en general corresponden a instantes de coarticulación o transiciones sonoro/sordo.

En la Fig. 8 se puede observar la evolución de las marcas de pitch en cada uno de los pasos descritos anteriormente. En el gráfico inferior se encuentran los valores de pitch finales que serán usados como referencia en los experimentos.

3.2. Descripción del Experimento

Para la realización del experimento se utilizaron 50 archivos de la base de datos mencionada en la Sección 3.1, los cuales fueron comprimidos usando MP3 con distintos *bitrates*: 8, 16, 32, 64, 128 y 160 Kbps. Luego fueron descomprimidas nuevamente a formato *WAV*, con un *bitrate* de 256Kbps, la misma codificación que el archivo original. Este proceso simula la compresión/descompresión utilizada para la transmisión via internet.

Como estas señales tienen una buena relación señal a ruido, se les agregó ruido blanco hasta lograr $(S/N)_{dB}$ de 20 y 10 dB, para poder así simular una situación de análisis real y poder determinar la robustez de los algoritmos al ruido.

Luego, las señales se analizaron con cada uno de los métodos de extracción de pitch detallados en la sección 2 y se calculó el porcentaje de errores de pitch doble y mitad. Sobre los resultados correctos se calculó el error cuadrático medio (RMSE) para cada método y *bitrate*, obteniendo como resultados los detallados en la Sección 3.3. La frecuencia fundamental se evaluó usando la escala logarítmica en base dos para trabajar con octavas.

3.3. Resultados y Discusiones

Los resultados experimentales de la Fig. 9 muestran el RMSE para cada uno de los algoritmos estudiados a diferentes *bitrates*. Los algoritmos que usan la autocorrelación ofrecen los mejores resultados.

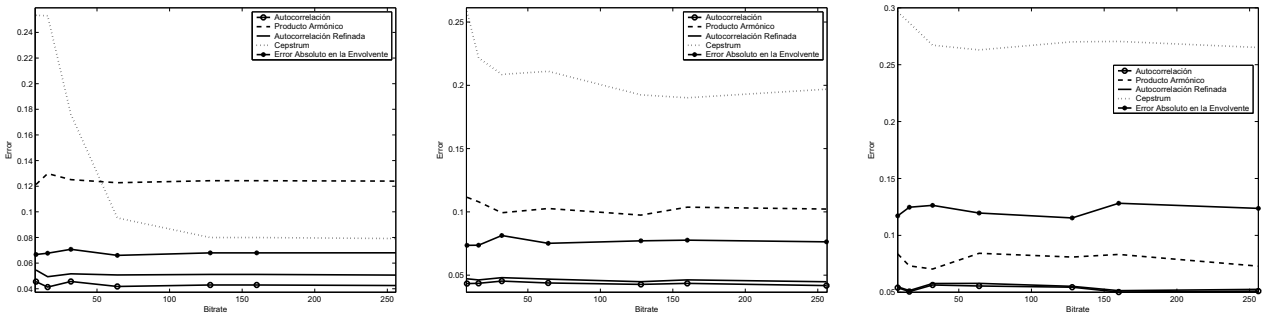
No existe degradación relevante en el RMSE para los diferentes *bitrates* en casi todos los algoritmos estudiados, incluso cuando se reduce la SNR a $20dB$ o $10dB$. El mismo comportamiento se observa en los porcentajes de pitch doble o mitad, donde no se detecta un aumento de los mismos debido a una mayor compresión.

El algoritmo que utiliza el producto armónico en el dominio de la frecuencia presenta una degradación importante en el error de estimación para los *bitrates* mas bajos. Además, la inclusión de más muestras para aumentar la resolución espectral, tal como se indicó en la Sección 2.3, no se refleja en un menor error de detección.

Por otra parte, el algoritmo que utiliza cepstrum, que opera también en el dominio de la frecuencia, presenta una degradación aún mas importante en el RMSE que el algoritmo que utiliza el producto armónico. En consecuencia, los resultados demuestran la poca robustez de la utilización del dominio frecuencial para la estimación del *pitch*.

Otro resultado destacable es el mayor error en la estimación del pitch utilizando valores no enteros. La falta de una referencia con valores de pitch mas precisos y el hecho de utilizar una interpolación lineal en la propuesta de Medan, para estimar el pitch no entero, han repercutido en los resultados de este algoritmo.

La mayor parte de los errores de *pitch doubling* o



SNR 60dB SNR 20dB SNR 10dB

Figura 9: RMSE para distintos bitrates y valores de relación señal ruido

pitch halving ocurrieron en los algoritmos que utilizan el dominio frecuencial. Por el contrario, los algoritmos de autocorrelación tienen errores de *pitch doubling* o *pitch halving* inferiores al uno porciento. Por lo tanto, resulta deseable utilizar los algoritmos en el dominio del tiempo en la estimación del pitch en las sesiones de práctica de oralización, para evitar errores de diagnóstico que pueden llegar a indicar valores espúreos de frecuencia fundamental al usuario hipoacúsico.

4. CONCLUSIONES

En este artículo se presentó un estudio de la influencia de la compresión de audio en el rendimiento de los algoritmos de extracción de pitch. Cinco algoritmos diferentes fueron implementados y evaluados: autocorrelación, autocorrelación optimizada, producto armónico, cepstrum y error absoluto envolvente.

Los resultados experimentales muestran que el aumento del error en RMSE debido a la compresión no es relevante. El mismo comportamiento se revela para el porcentaje de casos de pitch doble y mitad. La disminución de la relación señal ruido degrada el RMSE de todos los algoritmos, indicando la poca robustez de los mismos a la variación de la SNR.

Luego de los resultados obtenidos se ha decidido incluir en el sistema de práctica de la oralización la compresión de audio, usando MP3 para disminuir la cantidad de información transmitida via internet.

Referencias

[1] Paul Chiristopher Bagshaw. *Automatic prosodic analysis for computer aided pronunciation teaching*. PhD thesis, 1994.

[2] S. Euler and J. Zinke. The influence of speech coding algorithms on automatic speech recognition. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 621–624, 1994.

[3] Juan Manuel Garin, Pablo Daniel Agüero, and Juan Carlos Tulli. Nuevos aportes al entrenamiento de personas hipoacúsicas. *XIII Taller Internacional de Software Educativo*, Diciembre 2008.

[4] Xuedong Huang, Alex Acero, and Hsiao-Wuen Hon. *Spoken language processing*. Prentice Hall PTR, 2001.

[5] John Kominek and Alan W Black. Cmu arctic databases for speech synthesis. Technical Report CMU-LTI-03-177, Language Technologies Institute, School off Computer Science, Carnegie Mellon University, 5000 Forbes Ave.,Pittsburgh, PA15213, USA, 2003.

[6] Yoav Medan, Eyal Yair, and Dan Chazan. Super resolution pitch determination of speech signals. *IEEE Transactions on Signal Processing*, 39(1):40–48, 1991.

[7] Ted Painter and Andreas Spanias. Perceptual coding of digital audio. *Proceedings of IEEE*, 88(4):451–515, 2000.

[8] Holger Quast, Olaf Schreiner, and Manfred Schroeder. Robust pitch tracking in the car environment. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1:353–356, 2002.

[9] Cristobal Rivero and Prabhat Mishra. Lossles audio compression: A case study. Technical Report 08-415, Department of computer and information Science and Engineering, University of Florida, Gainesville, FL32611, USA, August 2008.

[10] Salina Abdul Samad, Aini Hussain, and Low Kok Fah. Pitch detection of speech signals using the cross-correlation technique. *Proceedings of TENCON 2000*, 1:283–286, 2000.

[11] Fangming Wang and P. Yip. Cepstrum analysis using discrete trigonometric transforms. *IEEE Transactions on Signal Processing*, 39(2):538–541, 1991.