Búsqueda de música por tarareo

Ernesto López
Martín Rocamora
Gonzalo Sosa

Instituto de Ingeniería Eléctrica

Facultad de Ingeniería

Universidad de la República Oriental del Uruguay

Julio 2004

Tutores:

Alvaro Pardo
Juan Pechiar
Federico Lecumberry

Colaboradores:

Luis Jure
Diego Azar
Daniel Maggiolo

Proyecto de fin de carrera.
Plan 1997.

Índice general

1

Introducción

1.1. Escenario

Anochece. Alfredo llega a su apartamento. Está agobiado, pasó diez horas en la oficina. Apoya pesadamente su mano sobre la puerta, que lo reconoce y se abre. La portátil del estar se enciende. Se sirve una copa de vino y se recuesta en el sofá. Tararea un par de compases de "A hard day's night". Desde los parlantes de su sistema de audio comienza a sonar la canción. Alfredo se siente un poco mejor.

La situación ilustrada presenta de forma intuitiva el problema de búsqueda de música por melodía. Mas concretamente, este problema consiste en construir una máquina capaz de reconocer una pieza musical a partir de un fragmento de melodía.

El interés por un sistema de búsqueda de música por contenido surge originalmente en la investigación musicológica, para encontrar repeticiones de cierto patrón dentro de una pieza musical o entre piezas distintas. Otra importante área de aplicación es la psicoacústica, en donde una de las inquietudes es entender la forma en que las personas identifican la melodía de una pieza polifónica. Asimismo, es útil para el estudio de la memoria musical, es decir cómo se retiene una melodía en la memoria y cual es la capacidad de reproducirla.

Actualmente, el constante crecimiento de la capacidad de almacenamiento y procesamiento de los sistemas informáticos permite guardar enormes cantidades de datos. La música es uno de los tipos de información almacenada. Acompañando a la gran capacidad de almacenar información deben existir formas apropiadas de recuperarla. La forma existente de buscar música se basa en consultas textuales, esto quiere decir que se deben suministrar datos tales como título, artista, álbum, etc. Sin embargo, es más natural recordar la música a través de características musicales. Por ejemplo, la melodía es uno de los rasgos más representativos en la música occidental. Desafortunadamente, no existen formas de recuperar música utilizando características musicales como la melodía. Es necesario incluir información de contenido en los datos almacenados que posibiliten su acceso de manera más natural. Esto se hace evidente en las características del nuevo estándar MPEG-7, que proporciona un conjunto de descriptores de información audiovisual para acceder al contenido multimedia. La búsqueda por contenido tiene ya cierto desarrollo en imágenes[1], pero es reciente su estudio en el dominio del audio.

En el caso de búsqueda de música por melodía, la voz cantada resulta una manera sencilla y natural de formular la consulta. Sin embargo, simular de forma automática la habilidad de los humanos de reconocer melodías (así como otros procesos cognitivos), es una tarea desafiante. Las computadoras son sordas, por lo que solo pueden lograr una comprensión aproximada del contenido melódico de la voz cantada a través del análisis de las distintas representaciones de la señal.

Las técnicas utilizadas en la búsqueda de música por contenido provienen fundamentalmente de dos áreas: procesamiento de señales y reconocimiento de patrones. La primera brinda herramientas para extraer características musicales de las señales de audio. La segunda es usada para establecer criterios de similitud a partir de estas características.

[1] *Google* por ejemplo, proporciona una herramienta de búsqueda de imágenes por contenido.

1.2. Objetivo

Este trabajo pretende abordar los aspectos teóricos y prácticos del problema de búsqueda de música por melodía. Para ello se estudian técnicas de tratamiento de señales de audio y de reconocimiento de patrones. Luego se aplican estas técnicas en la implementación un sistema que utiliza la voz cantada para realizar búsquedas de música por melodía.

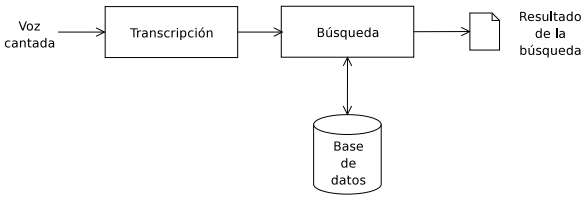


Figura 1.1: Diagrama de bloques de la arquitectura del sistema de búsqueda de música por melodía implementado.

1.3. Descripción de este trabajo

Una melodía se describe a través de una secuencia de notas. Resulta natural entonces que los sistemas de búsqueda de música utilicen notas para comparar melodías.

Por razones prácticas, el dominio de búsqueda consiste en melodías codificadas en alguna notación simbólica. Además, no existen métodos para extraer la melodía principal de una pieza musical a partir de una grabación[1][2]. Como consecuencia de lo anterior, la mayoría de los sistemas de búsqueda utilizan MIDI en lugar de audio comprimido (OGG, MP3).

La entrada del sistema es una señal acústica de voz capturada por un micrófono y luego digitalizada. Esta señal de audio digital se codifica PCM en algún formato de audio como WAV o AIFF. Adoptar el enfoque basado en notas para la búsqueda, hace necesario extraer de la consulta la secuencia de notas que componen la melodía cantada.

En base a las consideraciones anteriores, puede establecerse la arquitectura de un sistema de búsqueda (ver figura 1.1). La primer etapa consiste en la transcripción de la consulta a una secuencia de notas. En la siguiente se compara esta secuencia con las melodías almacenadas en una base de datos. Finalmente, el sistema devuelve una lista de piezas musicales ordenadas según su similitud con la consulta.

A continuación se describe cómo se implementa cada una de las etapas en este trabajo. La etapa de transcripción involucra las siguientes tareas:

> Estimar el contorno de frecuencia fundamental de la voz para determinar la altura de las notas.
>
> Segmentar la señal de audio para establecer el tiempo de comienzo y fin de cada nota.
>
> Realizar un análisis melódico para ajustar la altura de las notas a la escala temperada.

En la figura 1.2 se muestra un diagrama de bloques del proceso de transcripción.

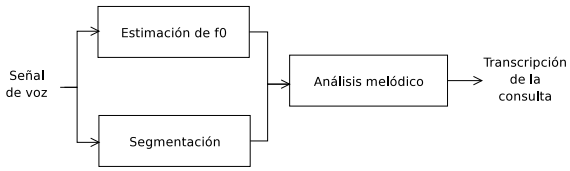


Figura 1.2: Esquema de las tareas involucradas en la etapa de transcripción.

Las tareas que constituyen la etapa de búsqueda son:

> Codificación de la secuencia de notas para independizar la comparación de melodías del tempo y altura.
>
> Establecer criterios de similitud flexibles para contemplar adornos y errores en la consulta, así como errores en la transcripción automática.

El diagrama de bloques de esta etapa se presenta en la figura 1.3.

Este trabajo hace hincapié en la transcripción de la voz cantada ya que es un problema para el cual no existe aún una solución completamente satisfactoria. La voz cantada es uno de los instrumentos musicales más difíciles de tratar. Las grandes variaciones tímbricas, los recursos expresivos, la

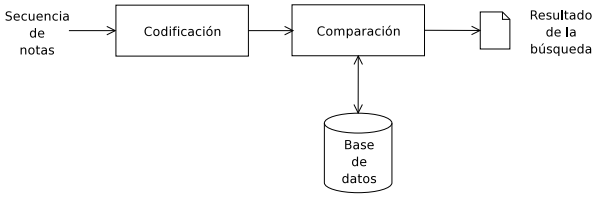


Figura 1.3: Diagrama de bloques de la etapa de búsqueda del sistema.

entonación inexacta y errores de altura y duración son algunas de la características que dificultan el análisis. Lograr una buena transcripción es esencial para el correcto funcionamiento de un sistema de búsqueda de música por melodía. En la práctica, los principales problemas que deben resolverse son: identificar los eventos musicales en la señal de audio y establecer el valor de altura de las notas dentro del conjunto discreto de valores de un sistema de afinación.

Por otra parte, en la etapa de búsqueda la principal dificultad es encontrar criterios de similitud adecuados que permitan manejar el compromiso existente entre tolerancia a errores de la consulta y capacidad de discriminación.

1.4. Organización

A continuación se detalla la organización de este documento.

En el Capítulo 2, *Búsqueda de música por contenido*, se trata más detalladamente el problema de búsqueda de música por melodía y se presentan brevemente algunas de las soluciones propuestas.

En el Capítulo 3, *Procesamiento de la voz*, se describe el mecanismo de producción de la voz y se señalan las características de la señal de voz digitalizada.

En el Capítulo 4, *Frecuencia fundamental de la voz*, se estudian técnicas para estimar la frecuencia fundamental de la voz y se describe la implementación de dos algoritmos, uno de los cuales es el utilizado en el sistema.

En el Capítulo 5, *Segmentación de audio en notas*, se exploran técnicas de detección de eventos en señales de audio. Se detalla la implementación de dos algoritmos basados en distintas técnicas y se selecciona el más adecuado para la aplicación.

En el Capítulo 6, *Detección de tempo*, se analizan técnicas de detección automática de tempo y tiempo de pulso. Esto se hace para estudiar la posibilidad de transcribir la consulta a notación musical.

En el Capítulo 7, *Transcripción*, se describe cómo se combina la información extraída para obtener la representación simbólica de la melodía. Se evalúan métodos para el ajuste de la afinación a la escala temperada.

En el Capítulo 8, *Técnicas de búsqueda de música por melodía*, se señalan los problemas involucrados al comparar melodías, se plantean soluciones y se describe el sistema de búsqueda implementado.

En el Capítulo 9, *Implementación*, se detallan los aspectos mas importantes de la implementación del sistema.

En el Capítulo 10, *Evaluación*, se presentan los experimentos realizados para la evaluación del sistema y sus resultados.

En el Capítulo 11, *Conclusiones y trabajo futuro*, se analiza el trabajo realizado desde una perspectiva crítica. Se sugieren futuras direcciones de investigación.

2

Búsqueda de música por contenido

2.1. Resumen

En este capítulo se presentan distintas direcciones de investigación sobre búsqueda de música por contenido. Se analizan algunos aspectos de la búsqueda de música por melodía y se mencionan los principales trabajos realizados en esta área.

2.2. Clasificación

Las distintas líneas de trabajo en búsqueda de música por contenido pueden dividirse según la siguiente clasificación: basadas en texto, de alto nivel y de bajo nivel[2].

La búsqueda basada en consultas de texto fue la primera en desarrollarse y requiere que el usuario conozca alguno de los atributos textuales de la pieza que busca, tales como título y compositor. Estos descriptores son muy concretos y si el usuario es capaz de proporcionarlos, permiten acceder rápidamente y sin confusión a la pieza musical buscada. Existen sistemas capaces de realizar búsquedas por contenido por texto, utilizando información como orquestación o características perceptuales (alegre, oscuro, brillante, etc). Esto presenta la dificultad de que el usuario debe recordar suficientes características para definir una pieza en particular y manejar conceptos subjetivos.

En cuanto a búsqueda a partir de información de alto nivel, existe una serie de trabajos que pueden dividirse en dos categorías. Una de ellas comprende la distinción en clases de sonidos, como por ejemplo la discriminación entre música, voz hablada, ruido, silencio. La otra categoría es la clasificación de música en géneros (rock, jazz, etc).

La búsqueda de bajo nivel consiste en identificar patrones rítmicos o melódicos específicos. Dado que la melodía es una de las características más representativas en la música occidental, la gran mayoría de los trabajos sobre búsqueda de bajo nivel se concentran en ella. Otros enfoques son la búsqueda de música a partir únicamente de patrones rítmicos o a través de un ejemplo, es decir usando como consulta un fragmento del audio original.

2.3. Búsqueda de música por melodía

Un sistema general de búsqueda de música debería ser apropiado para un amplio espectro de usuarios, desde músicos expertos hasta personas sin entrenamiento musical. Esto tiene la implicancia directa de que la consulta debe poder ingresarse usando la voz. Sin embargo, las características de la voz cantada hacen que su transcripción sea difícil. Una alternativa para evitar estas dificultades es ingresar la consulta usando un instrumento musical (por ejemplo, un teclado MIDI), pero se limita el número de usuarios. Muchos sistemas simplifican la transcripción imponiendo que se cante cada nota usando la sílaba "ta" de forma de identificar fácilmente los inicios de notas[3][4].

Transcribir música polifónica es una tarea difícil para un músico experto y prácticamente imposible para una máquina. Esto tiene una consecuencia fuerte sobre el contenido de la base de datos de un sistema de búsqueda por melodía. Al no existir mecanismos automáticos de transcripción, se debe almacenar música representada en forma simbólica (por ejemplo, MIDI) en lugar de archivos de au-

dio crudo o comprimido. Frecuentemente se argumenta que existen suficientes archivos MIDI en Internet como para que un sistema de este tipo sea viable[5], pero la necesidad de almacenar la música en forma simbólica no deja de ser una limitante.

El cuerpo de búsqueda de la mayoría de los sistemas desarrollados consiste en melodías representadas por archivos MIDI monofónicos. Esto se debe fundamentalmente a que existen técnicas ampliamente desarrolladas que permiten comparar secuencias de notas, como por ejemplo *Programación Dinámica*. Otro problema entonces es extraer la melodía de una pieza polifónica. Aún cuando se cuenta con las distintas voces por separado (como en MIDI polifónico) es difícil establecer cual de ellas se percibe como la melodía principal[5]; podría ocurrir incluso que se perciba una línea melódica no presente físicamente.

Al comparar melodías es necesario ignorar los cambios que no afectan el reconocimiento de una línea melódica. Concretamente, estos cambios son la transposición de altura y la modificación del tempo. Sin embargo, existen alteraciones que modifican la línea melódica pero permiten que aún sea reconocible. Ejemplos de esto son los errores aislados en la duración o altura y los rasgos expresivos. A medida que se acumulan alteraciones la melodía probablemente se torne menos reconocible[2]. Se requiere entonces que la etapa de búsqueda utilice criterios de comparación flexibles de forma de tolerar algunas alteraciones.

A partir de las consideraciones anteriores, el modelo adoptado por los sistemas de búsqueda de música por melodía es el siguiente[1],

1. Ingreso de la consulta cantada, tarareada o silbada.

2. Transcripción de la consulta a alguna notación simbólica.

3. Comparación entre la traducción de la consulta y los elementos de la base de datos codificados en notación simbólica.

La etapa de transcripción es la más crítica de este proceso. Recientemente se explora la posibilidad de utilizar el contorno continuo de frecuencia fundamental de la voz (microentonación) en lugar de la secuencia de notas para comparar melodías.

2.4. Antecedentes

El problema de búsqueda de música por tarareo fue introducido por primera vez en [4], donde se presenta el modelo basado en notas descripto anteriormente. Si bien la representación simbólica de la melodía y la etapa de búsqueda son muy simples, es el primer sistema completo desarrollado.

Uno de los sistemas mas conocidos es el denominado *MELDEX*, propuesto en [3]. Proporciona distintas opciones para la representación de la melodía y el criterio de búsqueda. Incorpora además la información rítmica.

Muchos sistemas de búsqueda utilizan para comparar melodías los criterios musicales propuestos en [6]. Estos criterios permiten asignar distinta relevancia a las alteraciones frecuentes en la interpretación de una melodía.

Una recopilación completa de los trabajos más importantes en el área de búsqueda de música por contenido puede encontrarse en [2].

3

Procesamiento de la voz

3.1. Resumen

Siendo la voz el objeto de estudio, es necesario conocer detalladamente sus características a fin de modelarla y tratarla adecuadamente.

El tratamiento de la voz se hace sobre señales digitales derivadas a partir de la onda sonora, dadas las ventajas del procesamiento digital. La señal digitalizada debe representar adecuadamente a la señal original para lo cual es necesario determinar los parámetros involucrados en la digitalización y el procesamiento, como frecuencia de muestreo, rango dinámico, enventanado, etc.

3.2. Mecanismo de producción

En el procesamiento de voz es útil una buena comprensión del mecanismo de producción de la voz, para determinar cómo puede ser modelado.

En la figura 3.1 podemos ver los órganos que intervienen en la producción de la voz[7] [1]. Los pulmones proveen la fuente de energía en la forma de un flujo de aire, que pasa por la laringe donde se encuentran las cuerdas vocales. La onda de presión es modulada de distintas formas para producir componentes de potencia acústica en el espectro audible. Las cuerdas vocales son dos membranas dentro de la laringe orientadas de adelante hacia atrás (ver figura 3.2). Por delante se unen en el cartílago tiroides (palpable sobre el cuello, inmediatamente por debajo de la unión con la cabeza; en los varones suele apreciarse como una protuberancia conocida como nuez de Adán). Por detrás, cada una está sujeta a uno de los dos cartílagos aritenoides, los cuales pueden separarse voluntariamente por medio de músculos. La abertura entre ambas cuerdas se denomina glotis. Cuando las cuerdas vocales se encuentran separadas, la glotis adopta una forma triangular. El aire pasa libre-

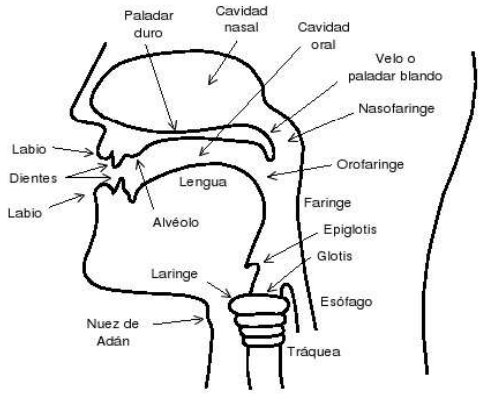


Figura 3.1: Esquema de órganos que intervienen en la producción de la voz.

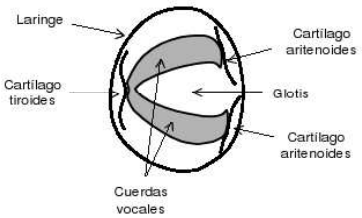


Figura 3.2: Corte transversal de la laringe.

[1] Fragmentos extraídos de la referencia.

mente y prácticamente no se produce sonido. Es el caso de la respiración. Cuando la glotis comienza a cerrarse, el aire que la atraviesa proveniente de los pulmones experimenta una turbulencia, emitiéndose un ruido de origen aerodinámico conocido como aspiración. Al cerrarse mas, las cuerdas vocales comienzan a vibrar a modo de lengüetas, produciéndose un sonido tonal, es decir periódico. La frecuencia de este sonido depende de varios factores, entre otros del tamaño y la masa de las cuerdas vocales, de la tensión que se les aplique y de la velocidad del flujo del aire proveniente de los pulmones. A mayor tamaño, menor frecuencia de vibración, lo cual explica por qué en los varones, cuya glotis es en promedio mayor que la de las mujeres, la voz es en general mas grave. A mayor tensión la frecuencia aumenta, siendo los sonidos mas agudos. Así, para lograr emitir sonidos en el registro extremo de la voz es necesario un mayor esfuerzo vocal. También aumenta la frecuencia al crecer la velocidad del flujo de aire, razón por la cual al aumentar la intensidad de emisión se tiende a elevar espontáneamente el tono de voz. Este fenómeno puede apreciarse al cantar, cuando al elevar el volumen de la voz, nos resulta mas sencillo dar determinadas notas agudas.

La frecuencia fundamental de la señal de voz está típicamente entre los 80Hz (voz masculina) y hasta alrededor de los 600Hz (voz de niño o voz aguda de mujer).

3.3. Clasificación de los sonidos

Existen muchas formas de clasificar los sonidos de la voz, pero introduciremos un tipo de clasificación que resulta útil en el análisis.

Los sonidos de característica tonal, se denominan sonoros [2] (voiced), y constituyen la mayoría de los sonidos en el lenguaje. Las vocales forman parte de esta categoría. Otro conjunto de sonidos sonoros son los sonidos nasales (“m”, “n”, “ñ”), los cuales se producen de la forma descrita, excepto que la boca permanece cerrada en algún punto, haciendo que el aire pase mayoritariamente por la cavidad nasal. La forma de onda de estos sonidos tiene la apariencia de la salida de un filtro excitado por una sucesión de pulsos (ver figura 3.3).

[2]Otras formas de referirse a estos sonidos son: armónicos, tonales, vocalizados.

Un segundo grupo, conocido como sonidos sordos (unvoiced), consiste en los sonidos producidos al expeler el aire a través de una restricción en la parte alta del aparato fonador (como en “sh”, “f”, “s”). Estos sonidos tienen generalmente menos energía que los sonidos sonoros, no hay periodicidad y la forma de onda parece aleatoria (con alguna limitada modulación espectral).

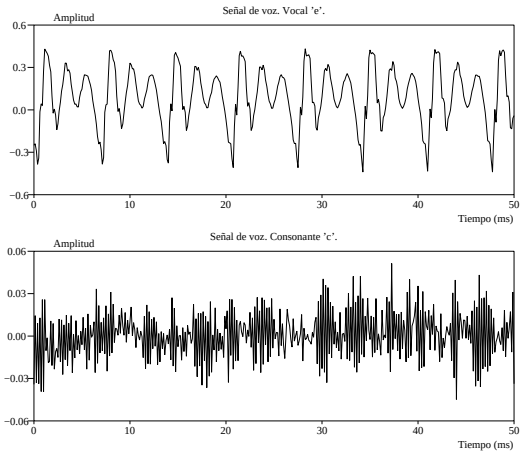


Figura 3.3: Forma de onda de sonido sonoro y sonido sordo. Fonema correspondiente a la vocal “e”, en la palabra “predicción”. Fonema correspondiente a la consonante “c” antes de la “i”, en la palabra “predicción”.

3.4. Voz hablada y voz cantada

Existen diferencias claras entre voz hablada y voz cantada respecto a su frecuencia fundamental[8]. En primer lugar el rango de f0 para voz cantada es más amplio que para voz hablada. La frecuencia fundamental de la voz cantada usualmente varía entre 60 y 1000 Hz. La altura de la voz cantada es estable durante tramos más largos que para la voz hablada, ya que al cantar se mantiene la altura de una nota por cierto tiempo. En la voz hablada esto no ocurre por lo que el tono de la voz puede variar en tramos más cortos. El vibrato, que consiste en una modulación de la frecuencia fundamental (con frecuencia de modulación entre 4 y 8 Hz), está usualmente presente en la voz cantada y raras veces en la voz hablada.

De las características mencionadas, la principal diferencia en la implementación de un algoritmo de detección de f0 para voz cantada y para voz hablada es el rango de frecuencias a detectar.

3.5. Modelo de producción

El mecanismo de producción de la voz puede ser modelado como una fuente de energía periódica o aleatoria excitando un filtro fuertemente no lineal[9]. Sin embargo, a pesar de que se puede hacer otro tipo de análisis, normalmente se adopta una aproximación lineal de este modelo.

El modelo lineal corresponde a una fuente de excitación, que consiste en un tren de pulsos periódico (para sonidos sonoros), o ruido (para sonidos sordos), que se aplica a un filtro lineal (el cual en su forma mas general es un conjunto de ceros y polos). La forma de onda de la voz se considera entonces como la convolución entre la excitación y la respuesta al impulso del filtro lineal. Este modelo forma parte de la base del análisis de voz (ver figura 3.4).

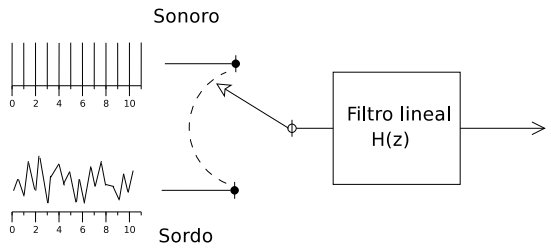


Figura 3.4: Modelo lineal de producción de la voz.

3.6. Propiedades de la voz digitalizada

3.6.1. Consideraciones generales

La señal digitalizada trasmite por si misma poca información y el propósito del procesamiento de voz es aplicar transformaciones apropiadas a este material crudo para derivar rasgos representativos de la señal bajo análisis[7].

Estos rasgos varían mas lento respecto a las variaciones rápidas de amplitud de la señal original, y se relacionan de alguna manera con la secuencia de fonemas que contiene la información lingüística. Otros rasgos pueden relacionarse con características mas cualitativas como la expresión, la altura de la voz, la identidad del hablante, etc. La tasa de cambio de estos rasgos es de sólo 4 a 5 por segundo, si bien esto no es constante y pueden ocurrir cambios rápidos entre un estado estacionario y otro.

Al no tener conocimiento previo de la estadística de la señal de voz, las muestras se consideran como provenientes de una fuente aleatoria de distribución desconocida. El objetivo de los algoritmos de procesamiento es hacer estimaciones de los parámetros estadísticos locales. Una estimación precisa puede hacerse únicamente en el límite de un número infinito de muestras de una distribución estacionaria. Pero para que la señal de voz contenga información, la estadística de la misma debe cambiar frecuentemente, siendo la distribución de muestras de la señal de voz fuertemente no estacionaria. Sin embargo, se puede asumir que en tramos cortos de tiempo, pequeñas secuencias de muestras pueden ser localmente estacionarias. De esta forma, los algoritmos pueden trabajar sobre pequeños bloques de muestras consecutivos. Desafortunadamente, esto implica que la precisión de las estimaciones será limitada.

3.6.2. Ancho de banda y frecuencia de muestreo

En la figura 3.5 se muestra en un espectrograma la distribución de energía de una señal de voz. A efectos ilustrativos se trazó una línea a 4kHz. Se puede apreciar que la mayor parte de la energía se concentra por debajo de este ancho de banda, a pesar de que hay componentes de frecuencia más altos (sonidos sordos). Si bien depende de la aplicación, en general una frecuencia de muestreo (fs) de 8 kHz es adecuada para el procesamiento de voz. En aplicaciones en las que se pretenda tratar la señal de voz con más precisión (por ejemplo en música) serán necesarias fs más altas, tales como 22050 Hz o 44100 Hz.

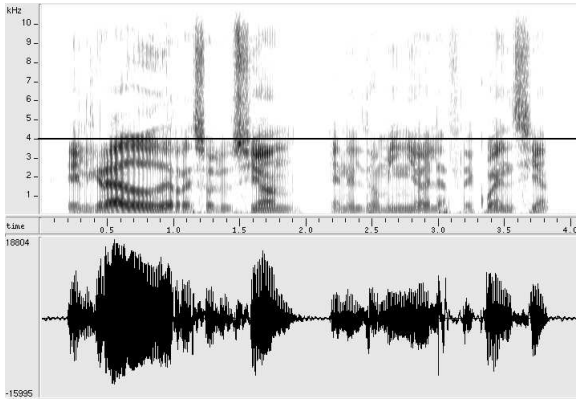


Figura 3.5: Espectrograma de señal de voz.

3.6.3. Rango dinámico

Otra consideración importante es la resolución usada en el proceso de cuantización, que está determinada por el rango dinámico de la señal de voz. Este es de alrededor de 40dB entre la voz normal mas alta y el umbral de audición. Asimismo, el ruido de fondo (cuantización, ruido eléctrico, etc) se torna molesto para el procesamiento cuando la SNR es peor que 30dB. Por lo tanto el rango dinámico total para calidad razonable debe ser 70dB, lo que equivale a 12 bits de resolución (6dB por bit). En base a esto normalmente se adopta una resolución de 16 bits.

3.6.4. Largo del bloque de análisis

La elección del largo del bloque de análisis es un tema crucial. Si es muy corto, el algoritmo puede no tener suficientes muestras para funcionar adecuadamente, y si es muy largo los parámetros estadísticos cambian dentro del bloque dando lugar a resultados que son un promedio mas a largo plazo. En la práctica bloques de 20-30ms (160 a 240 muestras a fs = 8kHz), son adecuados para la mayoría de las aplicaciones.

3.6.5. Enventanado

Seleccionar un bloque de muestras es enventanar la señal, y este proceso es de mucha relevancia. Una ventana rectangular tiene una forma de *sinc* en frecuencia con lóbulos secundarios amplios. La convolución en frecuencia de este espectro con el de la señal produce distorsión considerable. Si el algoritmo de estimación involucra análisis en el dominio de la frecuencia (como DFT), deberán ser usadas ventanas mas suaves en el dominio del tiempo (como Hamming o Hanning).

Si se pretenden detectar cambios rápidos en la señal de voz, es común procesar la señal en bloques que se solapan en el tiempo. Por ejemplo, para análisis espectral preciso, cada DFT puede ser hecha sobre bloques de 32ms (256 muestras a 8kHz), pero desplazados solo 5 ms. Los rasgos espectrales de bloques adyacentes serán similares durante tramos estacionarios de la señal, mientras que podrán detectarse cambios temporales de hasta 5 ms.

3.6.6. Resolución espectral

La resolución que se obtiene en el dominio de la frecuencia está dada por el largo del bloque de análisis N, es decir, la cantidad de puntos de la señal usados para calcular la DFT, siendo esta de fs/N. Esto implica el compromiso existente entre resolución temporal y resolución espectral. Para fs = 8kHz y 200 muestras, la resolución es de 40Hz. Esto indica que rasgos espectrales separados menos de 40Hz no se pueden resolver. Para la señal de voz, el detalle espectral más fino corresponde al tono de una voz grave masculina, y este es raras veces menor a 60Hz. En general transformadas de 200 a 250 muestras son adecuadas para análisis de voz (a fs = 8kHz).

3.6.7. Densidad espectral de potencia

Recordando el modelo de producción de la voz, veamos las características de la densidad espectral de potencia. La densidad espectral de potencia debería consistir, de acuerdo al modelo, en el producto del espectro de la excitación con la respuesta en frecuencia del sistema vocal. La transformada de un tren de pulsos consiste en armónicos igualmente espaciados. Estos armónicos estarían modulados en amplitud por la transferencia del sistema. Este es el caso de los sonidos sonoros y se puede apreciar claramente en la figura 3.6, de la densidad espectral de potencia de un sonido sonoro.

El efecto es menos claro para el caso de sonidos sordos. La transformada de una excitación gaus-

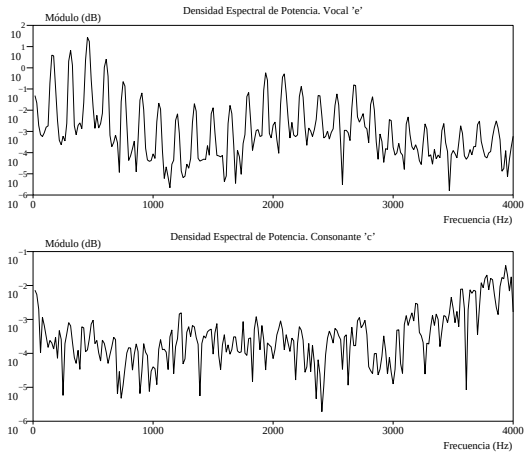


Figura 3.6: Densidad espectral de potencia de sonido sonoro y sonido sordo. Fonema correspondiente a la vocal "e", en la palabra "predicción". Fonema correspondiente a la consonante "c" antes de la "i", en la palabra "predicción".

siana debería ser un espectro plano, y uno esperaría ver en la gráfica de la densidad espectral únicamente la respuesta del aparato vocal. Sin embargo, ya que la transformada se realiza sobre un número finito de muestras, se ve reducida la varianza de la señal, y una secuencia aleatoria en el dominio del tiempo se transforma en una secuencia aleatoria en el dominio de la frecuencia, lo cual oscurece los resultados (ver figura 3.6). Hay que tener presente que si la fs es baja (8kHz), la representación de los sonidos sordos es pobre.

Retomando la gráfica de la densidad espectral de potencia para sonidos sonoros se pueden apreciar algunos rasgos importantes. Además de los armónicos igualmente espaciados, se puede ver que la envolvente de la densidad espectral decae gradualmente al aumentar la frecuencia. Esta envolvente no es uniforme y consiste en un conjunto de picos y valles, siendo los picos particularmente notorios. Estos corresponden con las resonancias del aparato vocal y equivalen a los polos del modelo del filtro lineal. La cantidad de picos y la ubicación relativa de los mismos, son características bastante específicas para cada tipo de sonido. Al parecer estos polos (normalmente referidos como formantes) contienen mucha de la información lingüística del habla, y gran parte del análisis de voz implica el reconocimiento de los mismos. Las vocales presentan estructuras de formantes bien definidas, y si bien su ubicación exacta varía con el hablante y con la realización del habla, se pueden establecer patrones de ubicaciones relativas y mapas de formantes. Hay generalmente 4 o 5 formantes en la región hasta los 4kHz para sonidos sonoros, aunque la 4ta y 5ta tienden a estar muy cerca y pueden no apreciarse separadas si la resolución espectral es baja.

Para los sonidos sordos la envolvente espectral no decae ya que estos sonidos tienen predominancia de altas frecuencias. El número de formantes es menor (2 o 3) y son menos pronunciadas respecto a los sonidos sonoros.

4

Frecuencia fundamental de la voz

4.1. Resumen

En este capítulo se analizan algoritmos de detección de la frecuencia fundamental de señales de voz. Debido a que el objetivo es desarrollar un sistema de transcripción de una melodía cantada a notación musical, el hincapié se hará en señales monofónicas de voz cantada. Se estudian técnicas usadas a estos efectos, y se implementan algunas de las mismas. Finalmente se desarrollan y evalúan dos algoritmos, uno de los cuales se basa en cepstrum y otro en una variante del método de autocorrelación.

4.2. Introducción

En el proceso de transcripción de una melodía monofónica a partir de una señal de audio cruda a notación musical (ej. MIDI) es necesario extraer las alturas y duraciones de las notas.

La altura de un sonido es un concepto subjetivo determinado a partir de la percepción, si bien se relaciona en general con la frecuencia fundamental de la onda sonora. Esto tiene excepciones, como en el caso de señales no periódicas que producen una sensación de altura (ruido de banda limitada, sonidos percutivos, etc.), o señales periódicas de frecuencia fundamental fuera del rango de existencia de altura. A partir de lo anterior, para realizar la transcripción se debe diseñar un algoritmo que estime la evolución de la frecuencia fundamental (f0) de la señal de audio.

La frecuencia fundamental de una señal periódica es el inverso del período. Este puede definirse como el mínimo desplazamiento temporal que mantiene a la señal invariante. Esta definición se aplica a señales perfectamente periódicas, pero las señales de interés (voz, señales musicales, etc.) distan bastante de esta idealización. A su vez las señales acústicas son en general no estacionarias, lo que dificulta aún más su análisis. Estas características hacen que el problema de estimación de altura sea una tarea desafiante, la cual es objeto de estudio desde hace décadas. El interés mayoritario en resolver este problema en forma precisa y eficiente se debe a su aplicación en codificación de voz. Por esta razón existen múltiples técnicas y algoritmos desarrollados, si bien no es un tema agotado.

4.3. Objetivo

El objetivo de este trabajo es estudiar las técnicas usadas en la extracción del contorno de f0 para señales de voz. Asimismo se pretende implementar algunas de estas técnicas, para utilizarlas en la transcripción de la voz cantada. Se evalúan los algoritmos implementados sobre bases de datos de voz disponibles para este propósito. Finalmente se elige aquel de mejor desempeño (considerando diferentes aspectos) y más adecuado a las necesidades de la aplicación.

4.4. Resumen de técnicas utilizadas

Existen diversas técnicas para la detección de la frecuencia fundamental de señales de audio. Una de las principales motivaciones del estudio de este problema es la codificación de señales a baja tasa de bits (empleada fundamentalmente en tele-

fonía), donde usualmente es necesario la estimación de f0.

Comenzamos describiendo brevemente algunas de las técnicas mas básicas, clasificándolas según el dominio donde se realiza la detección. En algún caso esto puede no ser tan claro, como en *Cepstrum*. Un estudio exhaustivo de todas las técnicas usadas se debe consultar en la bibliografía específica[10] y no es el objetivo de este trabajo. Sin embargo, para ampliar el panorama se mencionan finalmente algunos métodos relevantes.

4.4.1. Técnicas en el dominio del tiempo

Tasa de cruces por cero (ZCR)

Es una de las técnicas mas simples y consiste en contar la cantidad de veces que la señal cruza el nivel de cero para estimar su período.

La principal ventaja es el bajo costo computacional. Sin embargo, suele ser poco preciso cuando se trabaja con señales ruidosas o señales donde alguno de los armónicos es mas potente que la fundamental.

Autocorrelación (ACF)

Este es uno de los métodos mas usados y está basado en la función de autocorrelación, la cual es una medida de la similitud de la señal con ella misma desplazada en el tiempo. Esta se calcula de la siguiente forma:

$$r(\tau) = \frac{1}{N} \sum_{k=0}^{N-1} x(k)x(k+\tau). \qquad (4.1)$$

El cálculo de la similitud en función del desplazamiento temporal permite determinar la periodicidad de la señal, ya que la coincidencia es alta para desplazamientos múltiplos del período. Para estos valores de desplazamiento la función presenta picos (ver figura 4.1), el mayor de ellos ubicado en cero ya que hay coincidencia total.

El procedimiento consiste en filtrar la señal a un ancho de banda de unos 800 Hz, calcular su función de autocorrelación, y detectar el primer pico, que en teoría debe ubicarse en un retardo τ igual al período de la señal.

Es interesante notar que la información de formantes continúa presente (limitada al ancho de banda filtrado), en la forma de picos menores entre cero y el pico de periodicidad. Esto puede llevar a confusiones en algoritmos de detección automática de tono, y es práctica común eliminar la información de formantes antes de intentar medir el tono de la señal de voz.

Una variante de esta técnica consiste en tomar la diferencia en vez del producto en la ecuación 4.1,

$$d(\tau) = \frac{1}{N} \sum_{k=0}^{N-1} |x(k) - x(k+\tau)|. \qquad (4.2)$$

Esta función recibe el nombre de función diferencia (*AMDF*, Average Magnitude Difference Function) y presenta mínimos donde la autocorrelación presenta máximos. Sin embargo, estas dos funciones son estadísticamente independientes y se pueden combinar para lograr una estimación mas robusta, como en *WACF* (Weighted Autocorrelation Function)[11].

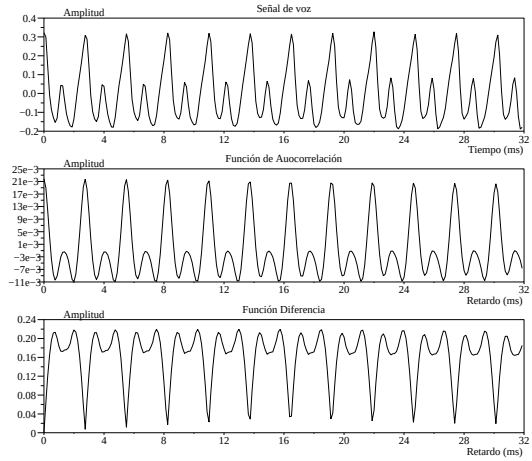


Figura 4.1: Función de Autocorrelación y Función Diferencia de un tramo de una señal de voz. La distancia entre los picos coincide con el período de la señal.

4.4.2. Técnicas en el dominio de la frecuencia

Autocorrelación del espectro

Se podría pensar que una forma directa de determinar la frecuencia fundamental de una señal

es a través del primer armónico en el espectro. Sin embargo, en la densidad espectral de potencia de una señal de voz los armónicos aparecen modulados por la transferencia del aparato vocal. El resultado es que algunos armónicos se ven amplificados por los polos de la transferencia (formantes) o atenuados por los ceros. Esto hace que encontrar el primer armónico no sea sencillo.

Existen métodos que plantean maneras mas robustas de obtener el primer armónico del espectro. Uno de ellos aprovecha el hecho de que el espectro de una señal periódica presenta cierta periodicidad. El primer armónico se puede obtener a través de la función de autocorrelación del espectro.

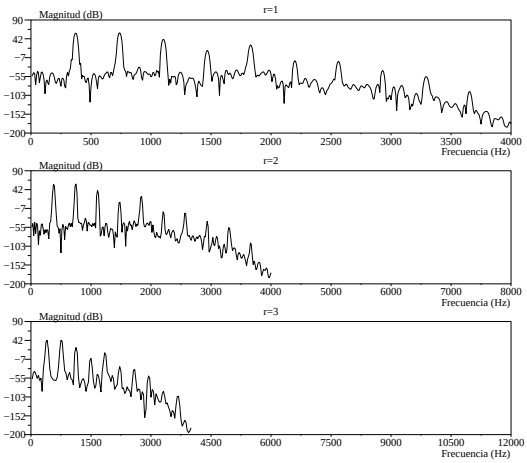


Figura 4.2: Potencia Espectral comprimida en frecuencia distintos factores r . El armónico r coincide con el primer armónico en cada caso.

Producto Armónico Espectral

Se trata de otra manera de detectar el primer armónico del espectro. El *Producto Armónico Espectral* (*HPS*) consiste en el producto de varias réplicas de la magnitud del espectro comprimidas en frecuencia,

$$Y(\omega) = \prod_{r=1}^{R} |X(\omega r)| \tag{4.3}$$

donde $X(\omega)$ es el espectro de la señal. La motivación de esta técnica surge del hecho de que para señales periódicas, la compresión del eje de frecuencia por un factor entero k causa que el armónico k coincida con la frecuencia fundamental (ver figura 4.2). Debido a que el espectro entre armónicos es casi nulo, al hacer el producto, el resultado es el realce del primer armónico y la cancelación del resto. Para obtener el valor de f0 solo se necesita detectar el mayor pico del HPS (ver figura 4.3).

El problema de esta técnica es que si alguno de los primeros armónicos no está presente, ya sea porque no existe en la señal o porque queda oculto por la transferencia del aparato vocal, ocurre la cancelación del primer armónico y por lo tanto el mayor pico del HPS puede no coincidir con la frecuencia fundamental.

Cepstrum

El Cepstrum es la antitransformada de Fourier del logaritmo de la potencia espectral de la señal.

$$C(\tau) = \mathcal{F}^{-1}\left(log\left(\left|\mathcal{F}\left(x(n)\right)\right|^2\right)\right). \tag{4.4}$$

Por lo tanto, el Cepstrum puede ser interpretado como un análisis en frecuencia de la función densidad espectral de potencia (*DEP*) en dB (por el logaritmo). El dominio del Cepstrum se corresponde con el dominio temporal y sus unidades se denominan *quefrency*. Los valores del Cepstrum son una medida de la velocidad de variación de los componentes espectrales. Esto significa que los componentes correspondientes a variaciones len-

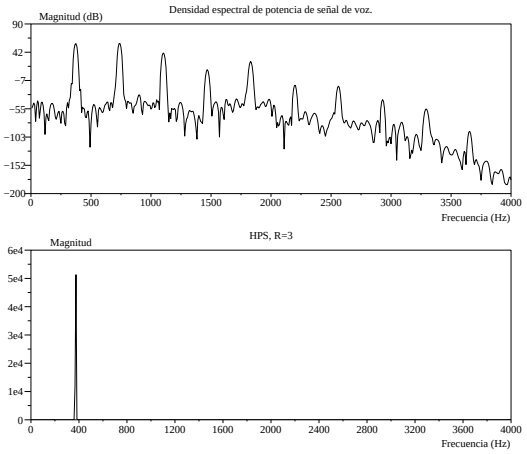


Figura 4.3: Potencia Espectral y HPS. En el HPS se resalta el primer armónico y se cancelan los otros.

tas de la DEP se encuentran en la zona baja del eje de la quefrency, mientras que las variaciones rápidas se encuentran en la zona alta de la quefrency.

La DEP de una señal de voz consiste en el producto del espectro de la excitación con la respuesta en frecuencia del aparato fonador. La excitación se transforma en un conjunto de armónicos equiespaciados en el dominio de la frecuencia para sonidos sonoros. En la DEP, estos armónicos aparecen modulados por la respuesta en frecuencia del aparato fonador. Las variaciones rápidas de la DEP, correspondientes a la excitación, se ubican en la parte alta del Cepstrum y aparecen como picos agudos situados en múltiplos del período del tono de voz. La amplitud de estos picos cae con la quefrency. La envolvente espectral, que corresponde a las variaciones lentas de la DEP se ubica en la parte de las bajas quefrencys (ver figura 4.4).

La frecuencia fundamental se determina a través de la ubicación del primer pico del Cepstrum.

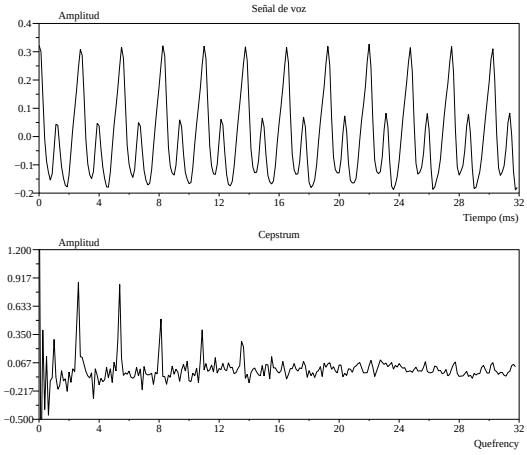


Figura 4.4: Señal de voz y Cepstrum. Los picos agudos del Cepstrum distan entre si el período de la señal.

4.4.3. Otras técnicas

Existen otras formas de abordar el problema de la detección de f0. Una de ellas, utilizada en codificación de voz, es obtener una estimación de la excitación glotal a través de *predicción lineal* (*LPC*). Como esta señal no tiene información de las formantes resulta más directa la estimación de f0 (ver figura 4.5).

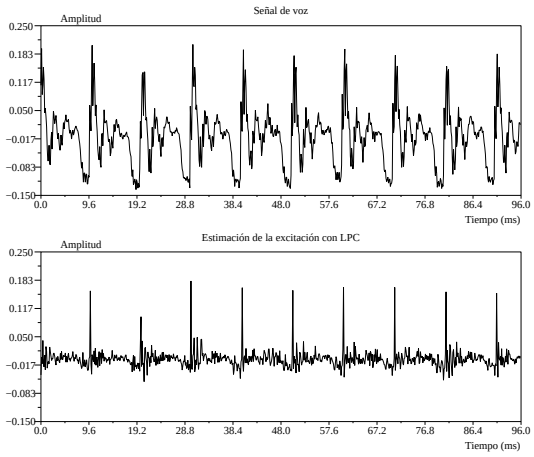


Figura 4.5: Señal de voz y excitación. La excitación no presenta la información de los resonadores del aparato vocal.

Otras técnicas utilizan la *transformada de Wavelet*[12], que permite un análisis multi-resolución y se usa ampliamente en tratamiento de imágenes y procesamiento de música.

4.5. Algoritmos implementados

Luego de estudiar las técnicas utilizadas en la detección de f0, se simularon algunas de ellas. Estas simulaciones se hicieron para lograr un primer acercamiento y verificar a grandes rasgos su funcionamiento. Las técnicas probadas fueron Autocorrelación, Cepstrum, función diferencia, HPS y LPC. En base al resultado de las simulaciones y reportes de performance de la bibliografía consultada[13][14], se decidió trabajar con Cepstrum y Función Diferencia y se implementaron algoritmos basados en estas técnicas.

4.5.1. Parámetros para el procesamiento de la señal de voz

A continuación se describe la forma en que se procesa la señal de voz en los algoritmos implementados.

Frecuencia de muestreo

Como se mencionó anteriormente, la energía de la voz para sonidos tonales, se concentra por debajo de los 4 kHz. Es para este tipo de sonidos que tiene sentido estimar la frecuencia fundamental, por lo que resulta suficiente trabajar con una frecuencia de muestreo de 8 kHz. El ancho de banda adicional que proporciona una frecuencia de muestreo mayor, no agrega información de interés para los sonidos sonoros y presenta la desventaja de aumentar el costo computacional. Por este motivo se trabaja con señales de voz muestreadas a una frecuencia de 8 kHz.

Largo del bloque de análisis

Remitiéndonos a la sección 3.6.1, se puede considerar que la voz es localmente estacionaria en tramos relativamente cortos (de 20 a 30 ms). Esto impone un límite superior al tamaño del bloque de análisis. Si se trabaja con bloques demasiado largos aumenta la probabilidad de que las propiedades estadísticas de la señal cambien dentro del bloque. No tiene sentido estimar un valor de f0 para un tramo de señal cuyas propiedades estadísticas cambien. Por otro lado no es posible trabajar con bloques de tamaño menor al mayor período que se pretende detectar, lo que impone un límite inferior para el largo del bloque. En base a lo anterior, se escogió el largo de 26.5 ms[1] como un buen compromiso entre ambas limitaciones.

Una consideración a tener en cuenta, es que la relación entre el rango de frecuencias a detectar y el tamaño del bloque no es directa, sino que depende del algoritmo. El rango exacto se aclara en la descripción de cada algoritmo y es consistente con los límites teóricos del tono de la voz.

Solapamiento de bloques

La resolución temporal está dada por el solapamiento entre bloques. A su vez, la tasa de estimaciones de f0 debe ser suficiente para acompañar las variaciones del tono de la voz. Una tasa de estimaciones de 100 valores de f0 por segundo es más que apropiada, por lo que el valor de salto entre tramas se estableció en 10 ms.

[1] Este valor es usado frecuentemente y facilita la evaluación, como se verá en la sección 4.6.1.

4.5.2. Posprocesamiento

El contorno de f0 obtenido como salida de un algoritmo de detección puede ser ruidoso y presentar errores aislados. Por esta razón es común procesar esta señal con técnicas de suavizado.

Una forma usual de suavizar una señal es mediante un filtrado pasabajos. Si bien esto reduce el ruido presente en la estimación de f0, afecta las discontinuidades (pasajes sonoro-sordo o cambios de altura) haciendo que las transiciones sean más suaves. Los errores gruesos aislados[2], especialmente nocivos en la transcripción de una melodía, son atenuados pero pueden no eliminarse usando esta técnica. Dicha clase de errores puede corregirse sin alterar los bordes del contorno de f0, utilizando un filtro de mediana. El filtrado de mediana, comúnmente usado en tratamiento de imágenes, consiste en una ventana de un número impar de puntos que se desliza sobre la señal, asignando a la muestra del centro el valor de la mediana. Esto permite reducir ruido impulsivo preservando los bordes.

En ambos algoritmos implementados se utiliza esta técnica de posprocesamiento con el objetivo de eliminar los errores gruesos. El filtro de mediana actúa sobre el contorno de f0 únicamente si se detecta un error grueso. Estos puntos se determinan filtrando pasabajos el contorno de f0, tomando la diferencia con el contorno original y detectando los puntos que superan determinado umbral[15]. Una vez realizado esto, se aplica un filtro de mediana en los puntos seleccionados y en sus vecinos inmediatos. El largo de la ventana se estableció en 7 muestras lo que permite suprimir hasta 3 muestras que difieran notoriamente del valor de las muestras vecinas. En la figura 4.6 se muestra el proceso paso a paso.

Es importante señalar que eliminar los errores gruesos de esta forma trae aparejado que el contorno de f0 se modifique en algunos puntos perdiéndose precisión. De todas formas, en relación a la aplicación del algoritmo esto es preferible frente a la presencia de errores impulsivos.

[2] En los algoritmos de estimación de f0 son frecuentes los errores de armónico o de subarmónico.

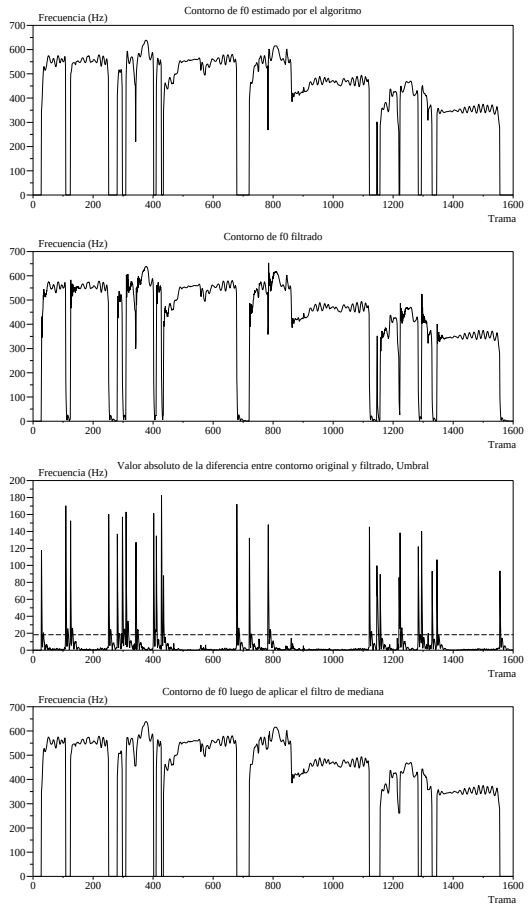


Figura 4.6: Proceso de posprocesamiento paso a paso.

4.5.3. Algoritmo basado en la función diferencia

Uno de los algoritmos diseñados utiliza la función diferencia y se basa en el algoritmo descrito en [16], con un conjunto de modificaciones para adaptarlo al sistema de transcripción de voz cantada.

Función diferencia

Para una señal perfectamente periódica de período T se cumple,

$$x(t) - x(t+T) = 0, \quad \forall t. \tag{4.5}$$

En base a lo anterior se puede construir una función que consista en la diferencia entre la señal y una versión retardada de la misma, cuya variable sea el retardo,

$$d(\tau) = \sum_{j=1}^{W} (x_j - x_{j+\tau})^2 \tag{4.6}$$

donde W corresponde al largo del bloque. Esta función se anula para valores del retardo múltiplos del período.

Para señales de voz, las cuales no son perfectamente periódicas, la función diferencia no se anula en múltiplos del período, pero presenta mínimos cercanos a cero. El valor del retardo correspondiente al primer mínimo coincide con el período de la señal.

La idea básica del algoritmo es calcular la función diferencia para cada bloque de señal, y estimar el valor de f0 detectando el primer mínimo. Vale la pena mencionar que se debe utilizar una ventana rectangular dado que otro tipo de ventana altera la periodicidad de la señal. El retardo máximo a considerar debe ser al menos el período máximo que se pretende detectar. Dado que la menor frecuencia a detectar es 60 Hz el retardo máximo corresponde a 17 ms, lo que equivale a $\tau_{max} = 136$ muestras a 8 kHz. Cada valor de la función diferencia se calcula como la suma de W términos, que consisten en la resta de dos muestras que distan τ. Para completar el cálculo para todos los valores de τ se deben considerar bloques de $W + \tau_{max}$ muestras. Por otra parte no se impone un límite superior para la frecuencia a detectar, más allá del dado por la resolución del muestreo.

Función diferencia normalizada

La función diferencia de un tramo de voz sonoro se acerca a cero en múltiplos del período. Por otro lado, los mínimos de la función diferencia de una señal no periódica (tramo de voz sordo), no están cercanos a cero. Es posible discriminar entre tramos sonoros y tramos sordos a partir del valor del mínimo de la función diferencia. Esto puede hacerse estableciendo un umbral de decisión, para lo cual es necesario normalizar la función diferencia. La normalización adoptada es la propuesta en [16], que consiste en dividir los valores de la fun-

ción entre la media acumulada,

$$d_{norm}(\tau) = \begin{cases} 1, & \text{si } \tau = 0 \\ \frac{d(\tau)}{\frac{1}{\tau}\sum_{j=1}^{\tau} d(j)}, & \text{en otro caso} \end{cases} \quad (4.7)$$

La función diferencia se anula para el retardo nulo, por lo que se debe imponer un retardo mínimo a partir del cual identificar el primer mínimo. Dada la normalización propuesta, el valor en $\tau = 0$ de la función diferencia normalizada es 1 y no es necesario restringir la búsqueda del primer mínimo (ver figura 4.7).

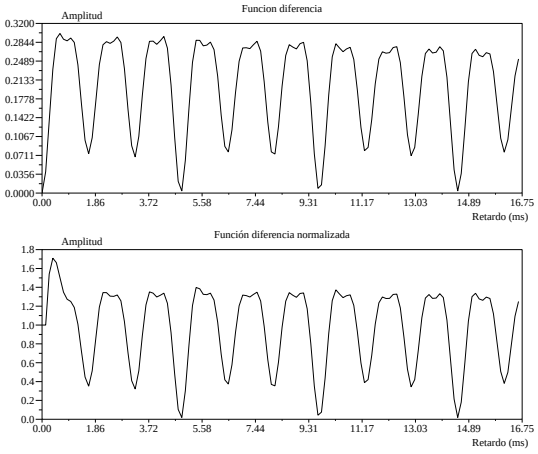


Figura 4.7: Función diferencia y función diferencia normalizada. La función diferencia normalizada no se anula en cero.

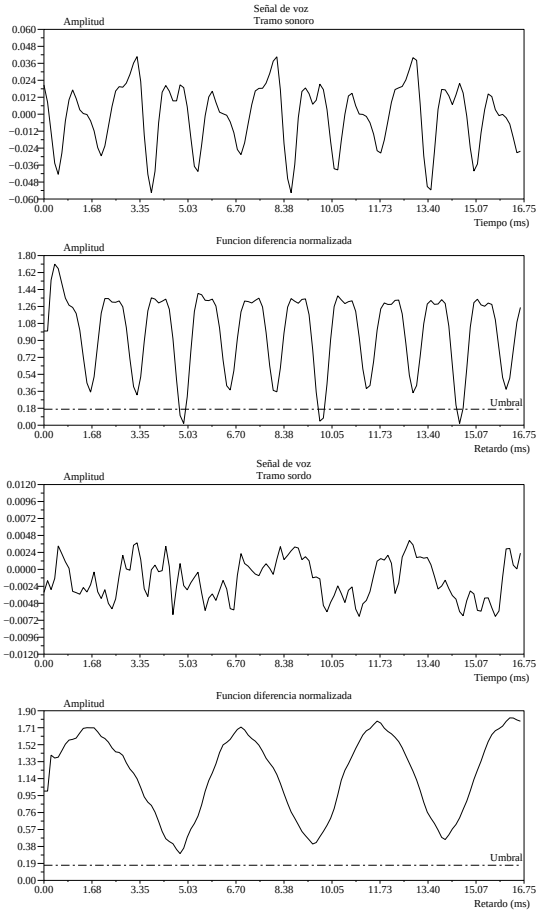


Figura 4.8: Función diferencia. Los mínimos superan el umbral para el tramo sonoro y no lo superan para el tramo sordo.

Umbral de decisión sonoro-sordo

Como se mencionó en la parte anterior, es necesario discriminar si tiene sentido la estimación de una frecuencia fundamental en el tramo de trabajo. Esto se hace a través de un umbral de decisión sobre la función diferencia, no estimando un valor de f0 si sus mínimos se ubican por encima de este umbral. El umbral puede interpretarse como una medida de la aperiodicidad tolerable en la señal, a los efectos de estimar su frecuencia fundamental.

Para los sonidos sordos es deseable que todos los mínimos de la función diferencia estén por encima del umbral (como en la gráfica inferior de la figura 4.8) y no se obtenga un valor de f0.

El umbral de decisión, por otra parte, es útil para seleccionar correctamente el mínimo correspondiente al período en la función diferencia. En sonidos sonoros, es frecuente que existan mínimos locales de menor retardo y menos profundos que el mínimo del período (ver figura 4.8), que provienen de las formantes del aparato vocal. A través del umbral, es posible descartar estos mínimos locales, estimando el período de la señal a través del mínimo de menor retardo que cae por debajo del umbral. Sin embargo, en presencia de armónicos fuertes, los mínimos locales pueden ser del orden del mínimo del período, y no se descartarían a través

del umbral[3]. Por esta razón, estimar el período mediante el mínimo de menor retardo puede conducir a errores de armónico, por lo que se adopta el criterio de considerar el menor mínimo por debajo del umbral como estimación del período. Desafortunadamente, suele ocurrir que mínimos de mayor orden sean más profundos que el mínimo del período, dando lugar a un error de subarmónico. Los siguientes pasos del algoritmo procuran evitar este último tipo de errores.

Aproximación por parábolas

Luego de tomar todos los mínimos por debajo del umbral, la estimación de f0 se hace considerando el retardo del menor de ellos.

Ya que conocemos la función diferencia sólo en algunos puntos (debido al muestreo) tomar el mínimo puede dar lugar a errores como se puede apreciar en la figura 4.9.

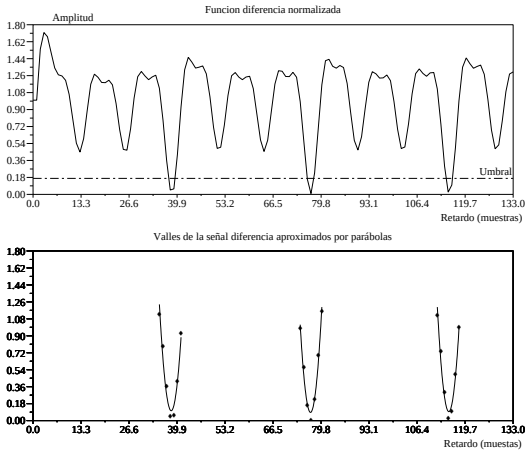


Figura 4.9: Aproximación por parábolas para aumentar la resolución y evitar errores al buscar el mínimo.

Cada valle donde se presenta un mínimo puede aproximarse por una parábola. Para evitar los errores mencionados, se aproxima de esta forma la función diferencia para cada mínimo y se estima el valor de f0 seleccionando el menor mínimo de las parábolas.

[3] Algunos armónicos se resaltan al coincidir con las formantes del aparato vocal.

La resolución de estimar el valor de f0 a través de la ubicación del mínimo de la función diferencia es de una muestra. Por esta razón el error absoluto de la estimación es constante, lo que implica que el error relativo para señales de período pequeño es mayor que para señales de período grande. El error relativo crece con la frecuencia de la señal, siendo el peor caso el límite superior del rango de f0 que se pretende detectar. Para una señal de frecuencia fundamental de 1 kHz el error relativo es 12.5 %. Para la aplicación final de algoritmo es necesario distinguir alturas que disten un semitono (distancia mínima entre notas), lo que impone una resolución mayor a la mitad de este intervalo. La distancia entre semitonos no es lineal y está dada por un factor de $\sqrt[12]{2}$, por lo que la resolución debe ser mayor a 3 %. La aproximación por parábolas incrementa la resolución en la estimación del período a menos del tiempo entre muestras.

Pendiente

El algoritmo de detección debe seleccionar el menor mínimo de la función diferencia. Sin embargo, es posible que el menor mínimo no se corresponda con el período de la señal. Si el mínimo elegido se encuentra en el intervalo entre cero y el retardo correcto, la frecuencia estimada es mayor que la real dando lugar a un error de armónico. Si por el contrario, el mínimo seleccionado se ubica más allá del correcto, la frecuencia estimada es menor que la real (error de subarmónico). Para las señales de interés se pudo observar que los errores más frecuentes corresponden a la estimación de un subarmónico. Como forma de atenuar estos errores se suma una recta de pendiente positiva a la función diferencia favoreciendo la detección de un mínimo de menor orden (ver figura 4.10). La elección de la pendiente implica un compromiso entre la reducción de los errores de subarmónico y el incremento de los errores de armónico.

Filtrado de la señal de voz

La señal de voz puede estar afectada de ruido aditivo en la forma de componentes de muy baja frecuencia. Esto puede deberse por ejemplo, a la respiración del cantante cuando se encuentra muy cerca del micrófono, o a interferencia de la

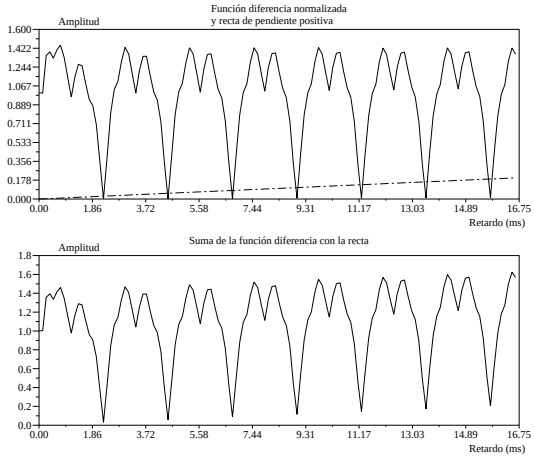


Figura 4.10: La suma de una recta de pendiente positiva reduce la probabilidad de errores de subarmónico.

red eléctrica. A los efectos de detectar la frecuencia fundamental este fenómeno introduce aperiodicidad indeseable en la señal bajo análisis. Como forma de atenuarlo, se filtra pasaaltos la señal de voz para suprimir los componentes de baja frecuencia. Teniendo en cuenta el límite inferior del rango de f0, se decidió establecer la frecuencia de corte en 50 Hz (ver figura 4.11).

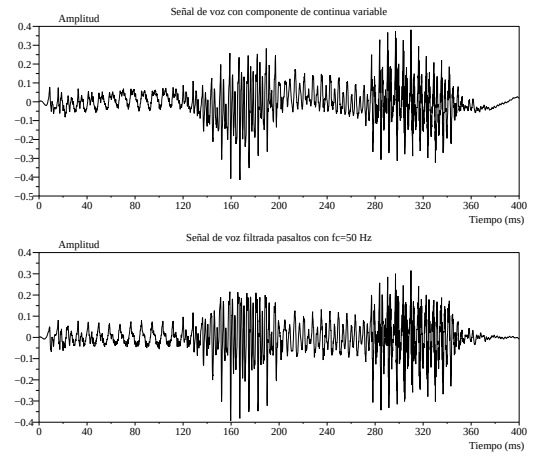


Figura 4.11: Señal de voz afectada por componente de baja frecuencia y señal filtrada pasaaltos.

En la figura 4.12 se observa como afecta el ruido de baja frecuencia a la función diferencia.

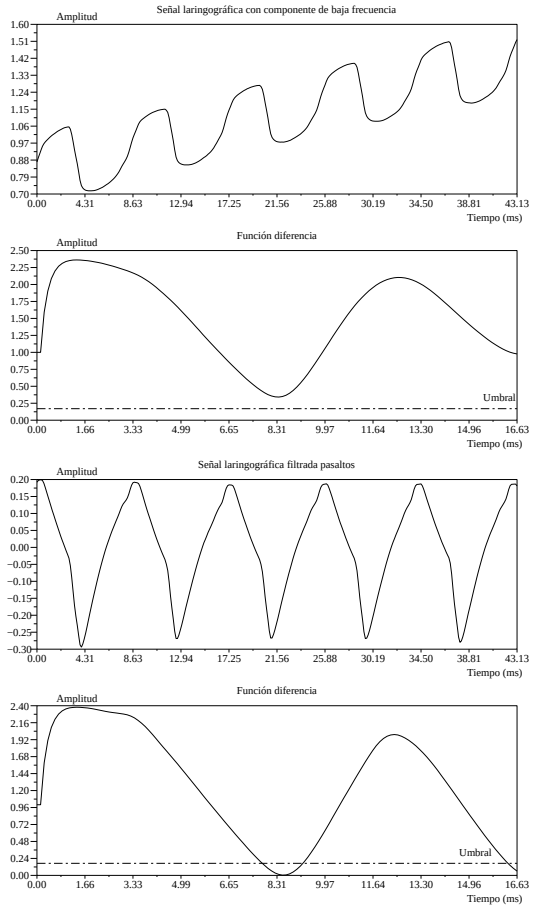


Figura 4.12: Señal laringográfica afectada por componente de baja frecuencia y función diferencia, antes y luego del filtrado pasaaltos.

Un punto esencial en el desempeño del sistema de detección de f0 es el filtrado pasabajos de la señal. La elección de una frecuencia de corte apropiada puede hacer milagros[17].

La señal diferencia contiene información de las resonancias del aparato vocal, que se manifiestan a través de valles, en general menos profundos que los de los múltiplos del período de la señal. Ocasionalmente, estos mínimos pueden ser del mismo orden y provocar estimaciones erróneas del algoritmo de detección de f0. Esto se debe a que las formantes del aparato vocal amplifican componentes de alta frecuencia, por lo que el fenómeno puede

ser eliminado o al menos atenuado mediante un filtrado pasabajos (ver figura 4.13).

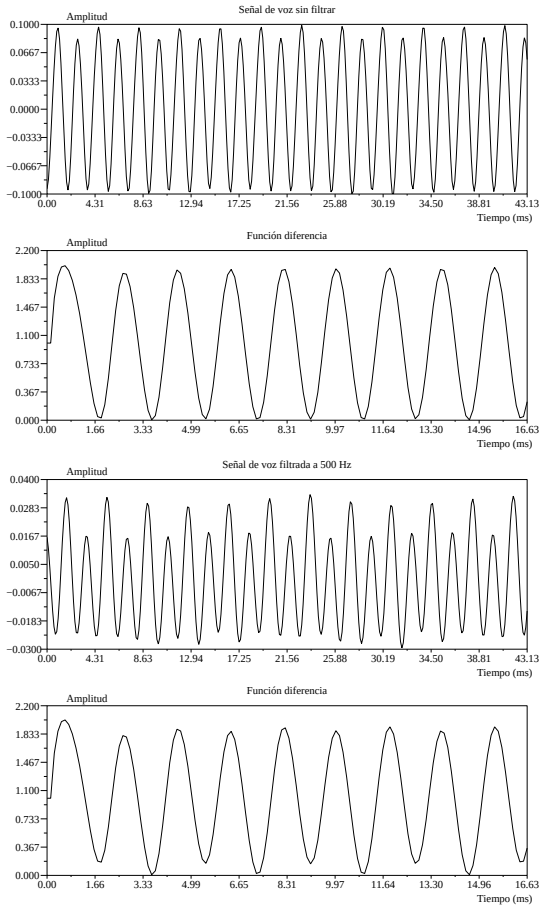


Figura 4.13: Señal de voz sin filtrado y luego de filtrar y sus respectivas funciones diferencia.

Otro motivo por el cual el filtrado pasabajos mejora el desempeño del algoritmo es por el hecho de aumentar la periodicidad de la señal. Comúnmente, las imperfecciones en la periodicidad de la señal de voz aparecen como componentes de alta frecuencia. El suavizado de la señal debido al filtrado, aumenta su periodicidad haciendo que los mínimos de la señal diferencia se acerquen mas a cero, facilitando su detección.

A partir de las consideraciones anteriores se filtra la señal de voz con un filtro pasabanda de Butterworth de orden 4, de frecuencias de corte 50 y 500 Hz. Debido a que la zona de transición de este filtro entre la región pasabanda y suprimebanda superior es amplia (ver figura 4.14), las frecuencias del orden de 1 kHz son atenuadas pero no totalmente eliminadas (la ganancia del filtro a 1 kHz es de aproximadamente -32 dB). Se comprobó experimentalmente que el algoritmo es capaz de detectar frecuencias fundamentales superiores a 1 kHz.

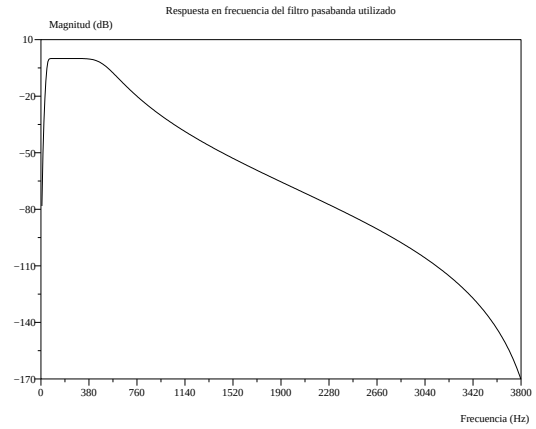


Figura 4.14: Respuesta del filtro pasabanda utilizado. Filtro de Butterworth de orden 4 con frecuencias de corte 50 y 500 Hz.

Consideraciones del cálculo de la señal diferencia

El cálculo de la señal diferencia de la ecuación 4.6 para bloques solapados es computacionalmente costoso. Sin embargo, se puede reducir el número de operaciones haciendo los cálculos en forma recursiva. El cálculo de la función diferencia se realiza tomando bloques de largo W, los cuales se solapan un número de muestras dado por el salto S (10 ms o 80 muestras). Para el primer bloque la función diferencia se calcula usando la siguiente ecuación:

$$d_1(\tau) = \sum_{j=1}^{W} (x_j - x_{j+\tau})^2 \tag{4.8}$$

El siguiente bloque comienza S muestras a partir del comienzo del anterior. Si calculamos la función diferencia para este bloque repetimos el cálculo para los puntos que van desde el comienzo del bloque actual hasta el fin del bloque anterior ($W - S$

puntos). Utilizando los cálculos que se repiten podemos expresar la función diferencia para el bloque actual en función de la calculada para el bloque anterior.

$$d_2(\tau) = d_1(\tau) - \sum_{j=1}^{S}(x_j - x_{j+\tau})^2 + \sum_{j=W+1}^{W+S}(x_j - x_{j+\tau})^2. \quad (4.9)$$

De esta forma la función diferencia correspondiente a un bloque puede ser calculada a partir del cálculo del bloque anterior, sumando y restando un bloque de largo S. Más específicamente, para cada nuevo bloque se computa únicamente el tercer sumando de la ecuación 4.9, reduciendo el número de operaciones en un factor de aproximadamente $\frac{W}{S}$ (para los parámetros elegidos esto corresponde a una reducción del orden de 2.6).

Otra forma de calcular la función diferencia se basa en expresarla en función de la autocorrelación de la siguiente forma:

$$d(\tau) = \sum_{j=1}^{W}(x_j^2 + x_{j+\tau}^2 - 2x_j x_{j+\tau}) \quad (4.10)$$

$$d(\tau) = r(0) + r_\tau(0) - 2r(\tau) \quad (4.11)$$

Imponiendo que la señal es cero fuera del bloque de análisis, se puede ver que la autocorrelación puede estimarse como[18],

$$r(\tau) = \mathcal{F}^{-1}\left(\frac{1}{W}\left|X(e^{-jw})\right|^2\right) \quad (4.12)$$

Esta forma de calcular la función diferencia permite mayor eficiencia computacional a través del uso de la FFT, pero presenta la desventaja de ser menos precisa. Esto último se debe a que la autocorrelación calculada a partir de la ecuación 4.12, supone que la señal es nula fuera del intervalo de análisis. Los valores de la autocorrelación se derivan de una suma de menos términos para valores de τ mayores, lo que introduce un sesgo no deseado en la estimación. Es posible corregir el sesgo dividiendo cada valor de la autocorrelación entre $W - \tau$, si bien esto no mejora la precisión.

El número de operaciones para calcular la función diferencia directamente es del orden de $2W\tau_{max}$. Aprovechando el solapamiento entre bloques se puede reducir a $2S\tau_{max}$ (que corresponde al tercer sumando de 4.9). Si se computa a partir de la FFT, el orden de operaciones es de $2Nlog_2N$, donde N es el número de puntos de la FFT que corresponde a la potencia de 2 inmediata superior a $2W$. Para los parámetros elegidos, el número de operaciones para un bloque en cada caso es 57.000, 22.000 y 11.000 respectivamente. La reducción de operaciones que ofrece la FFT frente a la precisión de la función diferencia obtenida no justifica su uso.

4.5.4. Algoritmo basado en cepstrum

El otro algoritmo implementado se basa en el análisis Cepstral de la señal de voz. Un estudio de este método se puede encontrar en [9]. La elección del mismo está basada en los resultados obtenidos en las simulaciones y en que es una técnica ampliamente utilizada dada su precisión y bajo costo computacional.

Cepstrum

El cepstrum de una señal se define como la antitransformada de Fourier del logaritmo de la densidad espectral de potencia, como se mencionó en 4.4.2. El cepstrum presenta picos agudos situados en múltiplos del período de la señal, los cuales reciben el nombre de rahmonics. El algoritmo estima el valor de f0 a partir de la ubicación del primer rahmonic del cepstrum para cada bloque de señal de voz.

Veremos como se relaciona el largo de la ventana de datos con el rango de frecuencias a detectar. Como la señal de voz es una señal real su cepstrum es una función real y par. Dadas estas características, solo la mitad de sus valores contienen información no redundante. Esta cantidad de valores debe ser suficiente para contener el primer rahmonic, ya que el período de la señal corresponde a su valor de quefrency. El caso límite lo impone el período máximo a detectar, por lo que el tamaño mínimo de la ventana debe ser el doble del período máximo. Para el tamaño de ventana de datos elegido, el período máximo que se puede detectar es de 26.5/2 ms lo que corresponde a una frecuencia de aproximadamente 76 Hz. Dado que la frecuencia mínima a detectar se impuso en 60 Hz como un límite seguro, es posible aumentar el rango de detección para el mismo largo de ventana usando relleno de ceros. Una ventana de datos de 26.5 ms corresponde a 212 muestras a fs

= 8kHz. Si mediante el relleno de ceros la cantidad de puntos de la FFT se aumenta a 256 el período máximo alcanzado en el cepstrum es 16 ms lo que equivale a 62.5 Hz.

Al realizar simulaciones con señales de baja frecuencia se pudo observar que el algoritmo no consigue detectar f0. La razón es el decaimiento dado por la envolvente del cepstrum $(1/N)$[19]. Esto hace que rahmonics alejados del origen tengan una amplitud reducida (ver figura 4.15). Es posible compensar este decaimiento multiplicando por una función inversa a la envolvente. Sin embargo, esto ocasiona un aumento en la probabilidad de ocurrencia de errores de subarmónico.

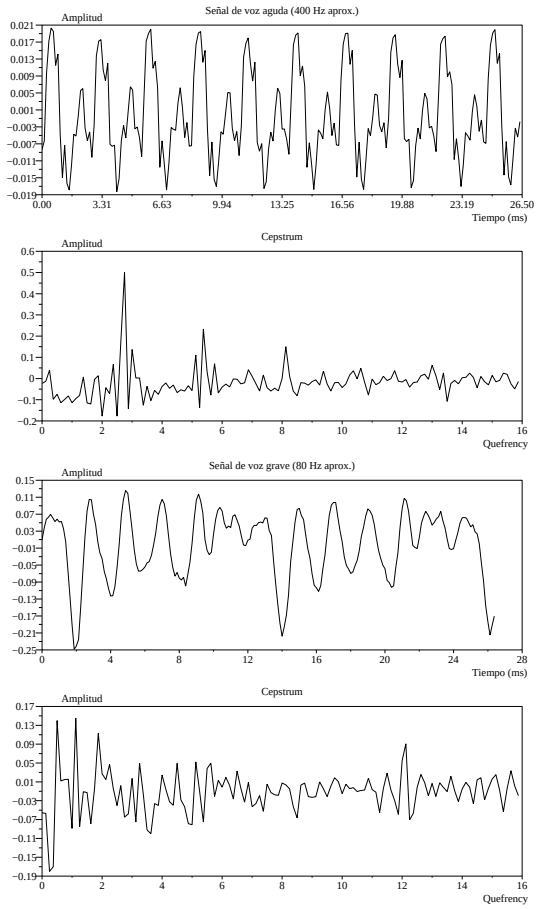


Figura 4.15: Cepstrum para voz aguda y grave. En el segundo caso, el rahmonic es más difícil de distinguir.

En el algoritmo implementado se optó por aumentar la cantidad de puntos de la FFT de 212 a 256 mediante relleno de ceros, pero no compensar el decaimiento del cepstrum. A pesar de que en la práctica, esto no aumenta el rango de detección considerablemente (debido al decaimiento del cepstrum), es más eficiente ya que permite usar algoritmos de FFT de potencia de 2.

La obtención del cepstrum implica el cálculo de la densidad espectral de potencia (DEP) de un bloque de señal de voz, por lo que es necesario enventanar los datos con una ventana suave en el tiempo, como se mencionó en 3.6.5. La ventana usada es la ventana de *Hanning*. Por otro lado, como la DEP es en este caso una función par, al pasar al dominio de la quefrency mediante un análisis espectral, no se debe enventanar ya que esto produciría serias distorsiones[9].

Filtrado pasaaltos de la señal de voz

Como se señaló anteriormente, la señal de voz puede estar afectada de componentes de muy baja frecuencia perjudiciales en la detección de f0. En el algoritmo implementado se filtra pasaaltos con un filtro de Butterworth de orden 4 de frecuencia de corte 50 Hz.

Liftrado del Cepstrum

El cepstrum de una señal de voz contiene la información de la envolvente espectral en las bajas quefrencys y la estructura de armónicos en la forma de picos agudos. La parte correspondiente a la envolvente espectral es de amplitud comparable a la del primer rahmonic. Esto puede conducir a errores si se detecta el primer rahmonic a partir de la amplitud máxima del cepstrum. Para evitar este tipo de errores, es preciso eliminar del cepstrum la información de la envolvente espectral. Esta se extiende hasta la mitad de la quefrency correspondiente al primer rahmonic, ya que la respuesta del aparato vocal está muestreada en la DEP por la estructura de armónicos. Filtrando pasaaltos la densidad espectral de potencia, lo que se conoce como liftrado, es posible eliminar esta información. Esto es equivalente a multiplicar el cepstrum por una ventana rectangular (respuesta de un filtro pasaaltos ideal). La quefrency de corte, la impone la ubicación del primer rahmonic, que depende del

tono de voz. La elección de una quefrency de corte genérica para el liftrado conduce al compromiso entre no eliminar completamente la respuesta del aparato fonador (en voces de tono grave) o suprimir el primer rahmonic (en voces de tono agudo). Mediante un liftrado más suave que el de un filtro ideal (ver figura 4.16), es posible atenuar la parte remanente de la envolvente espectral para voces graves mientras que no se suprime completamente el primer rahmonic para voces agudas. Al atenuar la amplitud del primer rahmonic se puede producir un error de subarmónico si esta queda por debajo de la amplitud de un rahmonic de mayor orden.

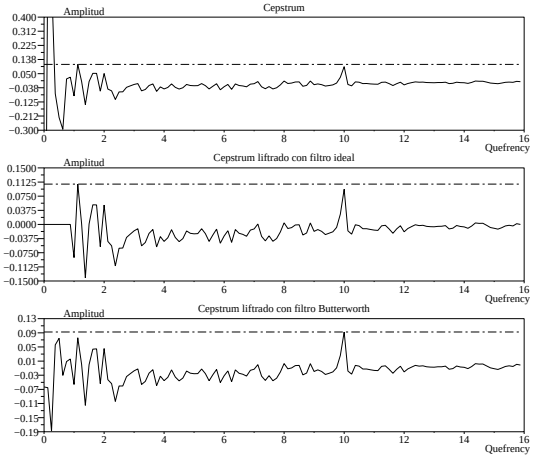


Figura 4.16: Cepstrum sin liftrar y cepstrum liftrado con distintos tipos de filtro. Solo en el último el máximo del cesptrum corresponde al rahmonic.

Decisión sonoro-sordo

Únicamente para sonidos sonoros tiene sentido la estimación de una frecuencia fundamental. Por esta razón es común que los algoritmos de estimación de f0 realicen una discriminación entre sonidos sonoros y sonidos sordos.

Una de las características que diferencia ambas clases de sonidos es la distribución de energía. Como se mencionó en 3.6.7, en los sonidos sordos predominan los componentes de alta frecuencia, mientras que los sonidos sonoros presentan una disminución de potencia en alta frecuencia. En el algoritmo diseñado se aprovecha el cálculo de la densidad espectral de potencia que se realiza para obtener el cepstrum, para discriminar entre sonidos sordos y sonoros. Se utiliza la potencia normalizada en baja frecuencia (hasta 1 kHz) y se establece un umbral sobre esta potencia. Aquellos sonidos que tienen potencia en baja frecuencia por encima del umbral son considerados sonoros. En otro caso se consideran sordos y se impone f0 = 0 (ver figura 4.17).

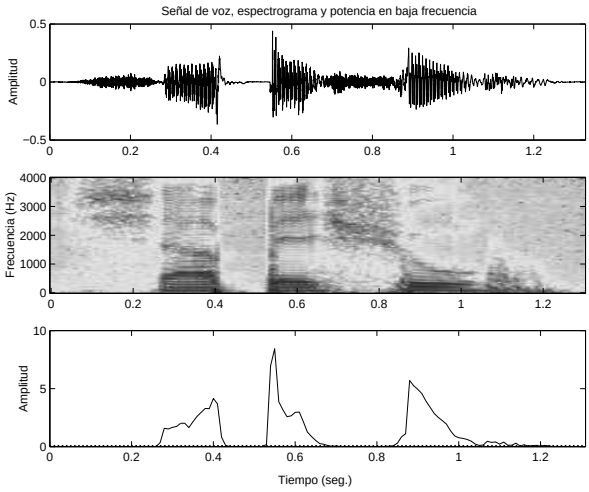


Figura 4.17: Espectrograma de señal de voz y potencia en baja frecuencia para cada trama. Sólo si la potencia supera el umbral se estima f0.

Aproximación por parábolas

Como ya se mencionó, la estimación del período de la señal corresponde a la ubicación del primer rahmonic, y este se obtiene como la muestra donde se da el máximo del cepstrum liftrado. De esta manera, la resolución de la estimación no supera el tiempo entre muestras. Esto implica que el error absoluto es constante, por lo que el error relativo crece con la frecuencia de la señal. Para señales de alta frecuencia este error, inherente al muestreo, no es despreciable. Por ejemplo, el período de una señal de 800 Hz muestreada a 8 kHz es de 10 muestras. Al ser la resolución una muestra, el error en la estimación puede llegar al 10 %. Para distinguir notas que disten un semitono, es necesario una resolución de 3 %.

Una forma de atenuar este problema es aproximar el rahmonic por una parábola y estimar el período como la ubicación del máximo de la parábola. Si bien se pueden sugerir otro tipo de aproximaciones dada la forma del rahmonic, por ejemplo dos rectas que formen un ángulo agudo, experimentalmente se obtuvieron buenos resultados usando esta aproximación.

4.6. Evaluación

El principal objetivo de la evaluación es cuantificar el desempeño de los algoritmos desarrollados para decidir cual utilizar en la aplicación de transcripción de voz cantada. Asimismo se compara su comportamiento frente a algoritmos de estimación de f0 disponibles en software de procesamiento de voz.

Un conjunto de algoritmos comúnmente utilizados como referencia en la evaluación de la estimación de f0[13][14], son los que provee el software *Praat*[20], para el estudio de fonética por computadora. Los algoritmos usados en la evaluación son *ACF* (Autocorrelation) y *FCC* (Forward Cross correlation). El software de procesamiento de audio *Wavesurfer*[21] también cuenta con una herramienta de extracción de f0. Es posible elegir entre los métodos *AMDF* y *ESPS*, ambos usados en la evaluación.

Los algoritmos se evaluaron sobre bases de datos de señales de voz, grabadas junto con señales laringográficas, que son una referencia para la estimación de la frecuencia fundamental. Las señales laringográficas se obtienen midiendo la resistencia entre dos electrodos que son colocados a ambos lados de la laringe. Estas señales son formas de onda mas simples que las señales de voz, por lo cual se puede obtener una estimación confiable de f0 (ver figura 4.18).

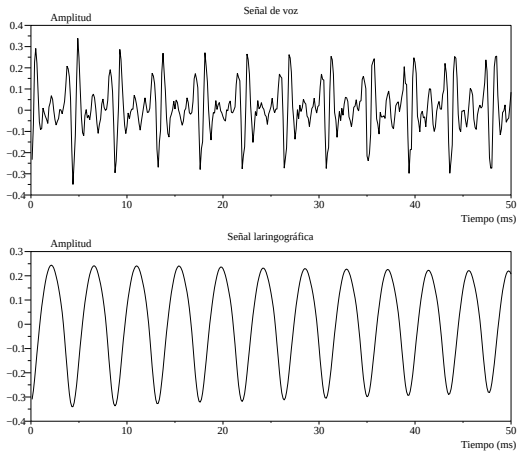


Figura 4.18: Señal de voz y señal laringográfica. A partir de la segunda se puede obtener una estimación confiable de f0.

4.6.1. Base de Datos

Para la evaluación se utilizaron 3 bases de datos construidas para este propósito que contienen los archivos de voz con sus respectivas señales laringográficas.

BD1 Keele Pitch Database. Esta base de datos de dominio público pretende ser, según sus autores[22], una referencia común para la evaluación de algoritmos de estimación de f0. Contiene las señales de voz y laringográficas de 10 personas (5 mujeres y 5 hombres) leyendo un texto fonéticamente balanceado, 'The North Wind Story'. Las señales de audio están muestreadas a una tasa de 20 kHz. Además contiene archivos de texto con estimaciones precisas de f0 dadas por los autores, utilizando una ventana de 26.5 ms y salto de 10 ms.

BD2 Esta base de datos[23] consta de 6 archivos de voz y sus respectivas señales laringográficas, generados por un contra tenor para la frase en francés: 'vos mépris chaque jour'. Las señales fueron muestreadas a 44.1 kHz, para distintas realizaciones de la frase: hablando, susurrando, gritando, cantando, cantando piano y cantando forte.

BD3 Un extracto de la canción *Memory* del musical *Cats* fue grabado por 3 cantantes profesionales (2 mujeres y 1 hombre)[24]. Se dispone además de las señales laringográficas. Las señales están muestreadas a 24 kHz.

Para la evaluación los archivos de audio fueron submuestreados a una frecuencia de 8 kHz por lo mencionado en 4.5.1.

A los efectos de comparar la performance de los distintos algoritmos es necesario que la tasa de estimaciones sea la misma. Habiendo elegido una tasa de 100 estimaciones por segundo, equivalente a un salto de 10 ms, se configuraron todos los algoritmos evaluados para que funcionen de esta forma. Este valor es frecuentemente utilizado y corresponde al valor por defecto en la mayoría de los algoritmos. La lista de algoritmos evaluados es la siguiente:

1. **AMDF** Provisto por *Wavesurfer*.
2. **ESPS** Provisto por *Wavesurfer*.
3. **ACF** Provisto por *Praat*.
4. **FCC** Provisto por *Praat*.
5. **CEPS** Implementación basada en Cepstrum.
6. **DIF** Implementación basada en la señal diferencia.

Para los algoritmos del software *Praat* se utilizaron los parámetros por defecto ya que la documentación señala que están optimizados para señales de voz y son los adecuados en la mayoría de los casos. El único parámetro alterado es el rango de frecuencias a detectar, estableciéndose el límite superior en 1 kHz. La frecuencia mínima está relacionada con el salto (tasa de estimaciones), de la forma $0{,}75/f_{min}$ para el ACF y $0{,}25/f_{min}$ para el FCC. Para imponer la tasa de estimaciones deseada se cambió el límite inferior de 75 Hz a 25 Hz en el FCC y para el ACF se utilizó el valor por defecto de 75 Hz. En el software *Wavesurfer* se configuró el rango de frecuencias a detectar de 60 Hz a 1 kHz y se usó el salto por defecto de 10 ms, para ambos métodos.

La forma de evaluación consiste en comparar la estimación de f0 dada por los algoritmos con alguna fuente de referencia. Para la base de datos BD1 se comparan las estimaciones de f0 obtenidas por los algoritmos con las proporcionadas como referencia en la base de datos. Para las bases de datos BD2 y BD3 se aplica cada algoritmo a la señal laringográfica y a la señal de voz y se comparan ambas estimaciones. Esto es práctica común en la evaluación de algoritmos de detección de f0.

Teniendo en cuenta la aplicación final del algoritmo se definen 3 clases excluyentes de errores,

1. *Error grueso (EGR)*: la frecuencia detectada difiere mas de un 20 % de la referencia.
2. *Error de f0 cero (Ef_00)*: la estimación o la referencia (pero no ambas) es cero.
3. *Error global (EGL)*: diferencia en Hz entre la estimación y la referencia.

En la transcripción de una melodía los errores gruesos son los más críticos ya que modifican la melodía cantada, por lo que es deseable que sean poco frecuentes. Un indicador importante en la evaluación es la cantidad de errores de este tipo que comete el algoritmo. Los algoritmos diseñados hacen una discriminación entre sonidos sordos y sonoros para decidir si se debe estimar una frecuencia fundamental. En el caso de un sonido sordo (o silencio), el valor de f0 asignado es cero. Esta discriminación puede no coincidir para la señal de voz y la señal laringográfica, por lo que durante una transición la frecuencia detectada puede valer cero solo para una de ellas. No parece apropiado considerar este tipo de errores como errores gruesos y se los agrupa en otra clase. Los errores globales permiten cuantificar la precisión de la estimación dada por el algoritmo. Para calcular el error global total se promedia el error global de las estimaciones que no pertenezcan a las restantes clases de error, es decir que no correspondan a un error grueso o error de f0 cero.

4.6.2. Resultados

A continuación se presentan los resultados de la evaluación agrupados de dos formas. En la figura 4.19 se aprecia para cada algoritmo los distintos tipos de error discriminado según la base de datos usada y un valor de error total que corresponde al promedio ponderado de los anteriores. Este resumen toma en cuenta todo el material disponible para la evaluación por lo que agrupa voz hablada y voz cantada. Considerando sólo los archivos de voz cantada, lo cual tiene más relevancia dada la aplicación final del algoritmo, se obtienen los resultados de la figura 4.20.

Algoritmo	BD	EGR %	Ef0 %	EGL Hz
AMDF	BD1	0.53	33.32	1.69
	BD2	1.52	7.08	1.62
	BD3	0	8.98	1.94
Total		**0.88**	**11.82**	**1.73**
ESPS	BD1	2.98	1.62	0.36
	BD2	0.96	6.49	1.73
	BD3	0.27	6.12	1.05
Total		**1.06**	**5.61**	**1.30**
ACF	BD1	0.31	4.7	0.29
	BD2	1.19	15.91	2.71
	BD3	0.65	8.96	0.95
Total		**0.88**	**11.94**	**1.78**
FCC	BD1	0.33	6.02	0.48
	BD2	1.48	18.06	2.58
	BD3	1.99	14.07	1.36
Total		**1.46**	**14.9**	**1.86**
CEPS	BD1	0.71	10.71	0.37
	BD2	5.80	12.19	3.53
	BD3	2.99	11.92	1.53
Total		**4.11**	**11.87**	**2.40**
DIF	BD1	0.19	1.74	0.45
	BD2	0.39	12.25	1.90
	BD3	0.51	7.91	1.02
Total		**0.39**	**9.21**	**1.39**

Figura 4.19: Resultados para voz hablada y cantada

Algoritmo	EGR %	Ef0 %	EGL Hz
AMDF	0.26	21.51	1.38
ESPS	1.53	2.73	0.51
ACF	0.35	4.29	0.47
FCC	0.68	6.65	0.66
CEPS	0.35	9.55	0.53
DIF	0.13	3.18	0.52

Figura 4.20: Resultados para voz cantada

4.7. Conclusiones

En el presente trabajo se estudiaron algunas técnicas usadas en la estimación de f0, se desarrollaron dos algoritmos basados en distintas técnicas (cepstrum y señal diferencia) y se evaluó su desempeño.

Es importante aclarar que la evaluación realizada es limitada dado el volumen de la base de datos, pero permite un acercamiento formal al desempeño de los algoritmos desarrollados. Del mismo modo la comparación con el resto de los algoritmos está sujeta a las condiciones en las que se realiza la evaluación (como frecuencia de muestreo utilizada, tipo de errores cuantificados, etc.). Los algoritmos diseñados fueron optimizados para estas condiciones las cuales son adecuadas para la aplicación final.

El interés principal es que el algoritmo a utilizar funcione correctamente para señales de voz cantada con la menor cantidad posible de errores gruesos (críticos para la transcripción).

Hechas las anteriores puntualizaciones podemos señalar que los algoritmos diseñados presentan, bajo las condiciones de evaluación, un desempeño comparable a los restantes algoritmos evaluados. El algoritmo desarrollado basado en la señal diferencia es el utilizado en la aplicación final dado su buen desempeño (presenta el porcentaje más bajo de errores gruesos), si bien es computacionalmente más costoso que el algoritmo basado en cepstrum.

5

Segmentación de audio en notas

5.1. Resumen

La transcripción de una melodía cantada a notación simbólica (ej. MIDI) implica determinar el comienzo, duración y altura de las notas. En este capítulo se estudian técnicas de detección de eventos en señales de audio, para establecer el tiempo de inicio de las notas. Se implementan dos algoritmos basados en distintas técnicas y se selecciona el más adecuado para la aplicación final.

5.2. Introducción

El término evento en un contexto musical puede no ser sencillo de definir. En forma simplificada se puede establecer que un evento es un fenómeno que presenta continuidad, por al menos, la mínima duración perceptible[25]. Bajo esta perspectiva, los cambios tímbricos, los rasgos expresivos y las notas, pueden denominarse eventos musicales. Nos interesa determinar para una melodía cantada aquellos eventos que corresponden a la ocurrencia de notas.

La segmentación automática de audio en notas, más allá de su uso en la transcripción, es importante en otras aplicaciones como edición de audio (cortar y pegar, time stretching), sincronismo de audio y video, entre otras.

Lamentablemente esta tarea no es trivial. Los límites entre notas son a veces difusos. En particular, las señales de voz cantada contienen un conjunto de rasgos (consonantes percusivas, sibilantes) que hacen que los límites entre notas sean difíciles de resolver. Es importante entonces poder discriminar comienzos genuinos de cambios graduales y modulaciones que tienen lugar durante el transcurso de una nota.

En los últimos años se han propuesto soluciones al problema de segmentación automática, si bien aún se mantiene sin resolver para un gran conjunto de aplicaciones. En música monofónica con instrumentos de ataque marcado (percutivos, de cuerda pulsada, etc) el problema de detección automática de inicio de notas no presenta gran dificultad. El comienzo de notas suave (característico de instrumentos de cuerda frotada) y recursos tales como el glissando, el legato o el fade-in, son problemáticos para cualquier método de segmentación automática. Existen propuestas que abordan el caso de música polifónica con mayor o menor éxito dependiendo del material analizado. Los trabajos van desde sistemas que resuelven adecuadamente hasta 4 voces de piano, a otros que analizan música con voz con desempeño más limitado[25][26][27][28][29].

Es posible distinguir distintos tipos de comienzo de notas en una señal de voz. Para el caso de notas que comienzan con fonemas oclusivos como “ta”, “pa”, “da”, la repentina liberación de energía posterior a la restricción del pasaje de aire, produce comienzos marcados. El inicio de estas notas resulta sencillo de determinar dado el gran cambio de energía presente en la señal. Cuando la nota comienza con un aumento gradual de energía (por ejemplo, una consonante nasal), el inicio es más suave y por lo tanto más difícil de establecer. En señales de voz cantada existen inicios suaves e inicios marcados por lo que un algoritmo de detección automática debe manejar adecuadamente ambos casos.

5.3. Objetivo

El objetivo de este trabajo es estudiar técnicas de detección de inicio de notas en señales de audio y desarrollar un algoritmo de detección automática apropiado para señales de voz monofónicas. La segmentación en notas dada por este algoritmo servirá como complemento del contorno de frecuencia fundamental a los efectos de la transcripción de una melodía cantada. En algunas ocasiones la información de altura no es suficiente para la segmentación (como por ejemplo en el caso de notas sucesivas de igual altura, ver figura 5.1), por lo que el inicio de las notas debe establecerse mediante otro procedimiento. El desempeño del algoritmo de segmentación automática deberá ser suficiente para aportar esa información faltante. No se pretende una discriminación exacta de todas las notas presentes en la señal, ya que para señales de voz este objetivo es difícil de alcanzar.

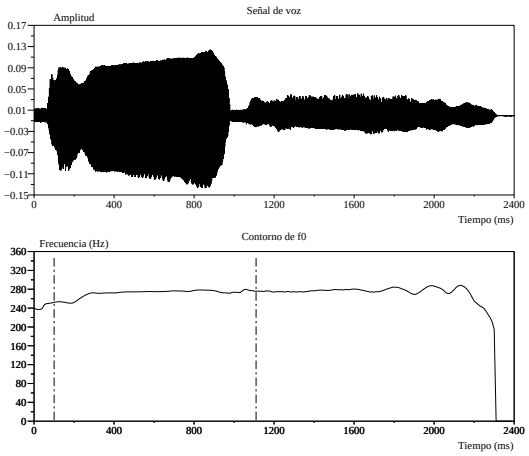


Figura 5.1: Señal de voz y contorno de f0 para notas consecutivas de igual altura. El inicio de las notas se indica sobre el contorno de altura mediante líneas verticales.

5.4. Resumen de técnicas utilizadas

En general los algoritmos de detección automática de notas pueden dividirse en dos etapas. La primera de ellas es la construcción de una función de detección a partir de la señal de audio en donde se evidencien más claramente los eventos. Una función de detección robusta presentará picos agudos indicando el inicio de las notas y se mantendrá dentro de niveles más acotados durante los tramos estacionarios de la señal. La siguiente parte consiste en localizar los picos de la función de detección (peak picking) que correspondan a inicios genuinos evitando los picos espurios que puedan aparecer. El procesamiento de la señal puede hacerse con técnicas en el dominio del tiempo, en el de la frecuencia o combinándolas.

5.4.1. Técnicas en el dominio del tiempo

Los algoritmos en el dominio del tiempo utilizan la amplitud de la señal y calculan para ventanas solapadas parámetros tales como valor RMS, valor de pico, media del valor absoluto, pendiente de la envolvente, etc. Estos métodos son en general simples y rápidos, por lo que se utilizan ampliamente.

Dixon[27] calcula la envolvente a partir de la media del valor absoluto de la señal en ventanas y utiliza su derivada como función de detección. Su aplicación consiste en el seguimiento del tempo de una señal de piano, por lo que no requiere una detección muy robusta. Esta técnica también es usada por Schloss[30] para transcripción de señales de percusión en las que las notas tienen fuerte ataque y decaimiento.

Klapuri[28] usa un banco de filtros para dividir la señal en bandas. Calcula la derivada de la envolvente de cada banda y combina la información usando conceptos psicoacústicos de sonoridad. Obtiene un buen desempeño para música polifónica.

5.4.2. Técnicas en el dominio de la frecuencia

Las notas de ataque fuerte presentan un aumento considerable de energía. Un método tradicional consiste en calcular la potencia espectral de la señal y detectar los cambios bruscos. Sin embargo, no se obtienen buenos resultados para notas de inicio suave. También se propone observar los in-

crementos de energía en alta frecuencia, con cierto éxito para un amplio rango de señales (por ejemplo, percutivas). Otra técnica similar es tomar la diferencia del módulo de la *STFT* (Short Time Fourier Transform) índice a índice entre tramas sucesivas de la siguiente forma,

$$dE(m,k) = |S(m,k)| - |S(m-1,k)| \tag{5.1}$$

donde m es el número de trama y k el índice de la STFT. El vector $dE(m)$ es una medida de la variación de energía entre tramas sucesivas. En general, las componentes de este vector serán pequeñas en tramos estacionarios mientras que en el inicio de notas podrán tomar valores significativos. A partir de dE se proponen diversas funciones de detección, como la suma o la media del valor absoluto de sus componentes, la desviación estándar de su distribución, etc.

Mas recientemente, Bello y Sandler[25] han propuesto considerar la información de fase para discriminar entre estados estacionarios y transitorios en una señal de audio. Mediante la STFT , una señal $s(n)$ se descompone en sinusoides de frecuencia $\Omega_k = 2\pi k/N$ cuya amplitud y fase corresponden al módulo y la fase de cada índice de la transformada. De acuerdo a la teoría del *Phase Vocoder*[31], para una sinusoide en condiciones ideales, la fase actual del k-ésimo índice de la STFT puede predecirse como:

$$\tilde{\varphi}_p(m,k) = \tilde{\varphi}(m-1,k) + \Omega_k T \tag{5.2}$$

donde Ω_k es la frecuencia angular correspondiente al k-ésimo índice y T es el salto de la STFT. El tilde indica que la fase está desdoblada. Los sonidos reales, sin embargo, no son perfectamente estacionarios por lo que la predicción difiere de la fase real en cierta desviación de fase. Expresando la fase de tramas sucesivas en función de la fase de la trama anterior se tiene,

$$\begin{aligned}\tilde{\varphi}(m-1,k) &= \tilde{\varphi}(m-2,k) + \Omega_k T + \tilde{\varphi}_d(m-1,k)\\ \tilde{\varphi}(m,k) &= \tilde{\varphi}(m-1,k) + \Omega_k T + \tilde{\varphi}_d(m,k)\end{aligned}$$

donde $\tilde{\varphi}_d$ es la desviación de fase de la predicción debido a la no estacionariedad. Esta desviación surge de variaciones en la frecuencia instantánea de la k-ésima sinusoide. Tomando la diferencia de la desviación de fase de las ecuaciones anteriores ($\tilde{\varphi}_d(m,k) - \tilde{\varphi}_d(m-1,k)$), obtenemos el error de predicción de fase de la trama m en función de las tramas $m-1$ y $m-2$,

$$\begin{aligned}d\varphi(m,k) = \quad & \mathrm{Arg}\big[\tilde{\varphi}(m,k) - 2\tilde{\varphi}(m-1,k) + \\ & \tilde{\varphi}(m-2,k)\big]\end{aligned} \tag{5.3}$$

siendo *Arg* el argumento principal. Este valor es una medida de la variación de la frecuencia instantánea de la sinusoide k-ésima y por lo tanto un indicador de su estabilidad. Para una sinusoide estable la frecuencia instantánea es prácticamente constante por lo que $d\varphi$ toma valores pequeños. Al ocurrir un evento impredecible, como el comienzo de una nota, $d\varphi$ se aleja de cero.

Es posible construir una función de detección que indique los momentos donde ocurren transitorios a partir de la distribución de $d\varphi$ según k. Sin embargo, la frecuencia instantánea de las componentes sinusoidales de un sonido en estado estacionario no es perfectamente estable, lo que constituye una limitante de la técnica. Es posible descomponer un sonido en una parte determinística y otra aleatoria[32][33]. La parte determinística es la que puede representarse mediante sinusoides (parciales). Por el contrario, los componentes ruidosos no pueden ser modelados como sinusoides ocasionando variaciones de fase inesperadas en los componentes de la FFT. Los índices de la FFT que representan parciales de la señal son de fase estable. La fase de los restantes componentes (de pequeña amplitud) es sumamente inestable. En la figura 5.2 se muestra este fenómeno.

La representación de los sonidos sordos de la voz consiste fundamentalmente en la parte aleatoria. Estos sonidos son muy frecuentes y en una melodía cantada su ubicación puede no coincidir con el inicio de una nota (por ejemplo la consonante “s” al final de una palabra). Usando la técnica descripta puede ser difícil resolver adecuadamente estos casos.

Se han propuesto diversas formas de combinar la información de fase y de energía. Las más sencillas consisten en obtener una función de detección más definida mediante el producto de las funciones de detección derivadas de la energía y la fase[26]. En [34] se propone tomar la distribución de la distancia Euclidiana entre los complejos de la STFT trama a trama.

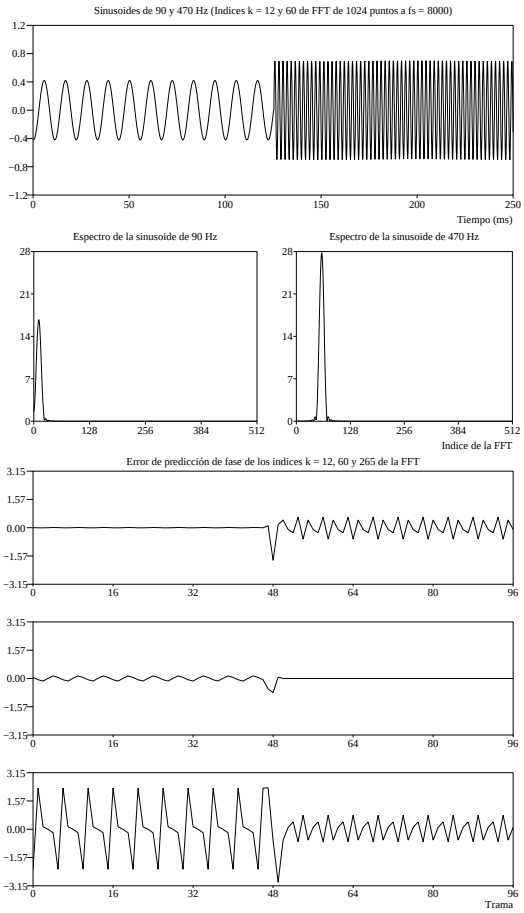


Figura 5.2: Error en la predicción de fase de tres componentes de una secuencia de sinusoides. Cuando la frecuencia de la componente está presente en la señal (magnitud del espectro apreciable), la predicción de fase es buena. En caso contrario, la fase se comporta de forma inesperada.

5.4.3. Otras técnicas

Existen trabajos que abordan el problema de segmentación de notas usando técnicas menos tradicionales.

Jensen y Murphy[35] derivan un conjunto de 12 rasgos a partir de la señal de audio (frecuencia fundamental, brillo, contenido de alta frecuencia, amplitud, etc) que sirven como entrada a una red neuronal. La función de detección se obtiene directamente como la salida de la red. Los autores no reportan resultados satisfactorios (20 % de error).

Abdallah y Plumbley[36] proponen el uso de *ICA* (análisis de componentes independiente) como entrada a un modelo oculto de Markov del cual se deriva la función de detección.

5.5. Algoritmos implementados

Habiendo explorado las técnicas utilizadas en la detección de eventos en señales de audio, se seleccionaron dos de ellas en base a su desarrollo, amplia utilización y resultados reportados. Las técnicas usadas son las propuestas por Klapuri[28] basada en la derivada de las envolventes de distintas bandas, y Bello y Sandler[34] que combina la información de potencia y fase en el dominio complejo.

5.5.1. Algoritmo en el dominio complejo

Es posible construir funciones de detección a partir de la información de energía y de fase que se obtiene al hacer un análisis espectral de la señal de audio. Se han propuesto distintas formas de combinar esta información para generar una función de detección robusta. En esta sección se analizan las características de estas funciones de detección, las diversas formas de combinarlas y se describe el algoritmo implementado.

Funciones de detección

Función basada en la energía. Los algoritmos basados en la energía consisten en asociar los inicios de nota a cambios bruscos en la energía de la señal. Suelen ser rápidos y fáciles de implementar. Sin embargo su efectividad es reducida cuando los transitorios no son pronunciados. Hay varias formas de construir una función de detección a partir de la energía. La más sencilla es tomar la diferencia de potencia entre tramas de la señal. En la sección 5.4.2 se mencionaron varios ejemplos de funciones de detección derivadas a partir de la diferencia de potencia de la ecuación 5.1. En la figura 5.3 se observa la media del valor absoluto de dE. Esta función de detección es eficaz solo para tipos particulares de señales.

Función basada en la fase. Se señaló anteriormente que es posible construir funciones de de-

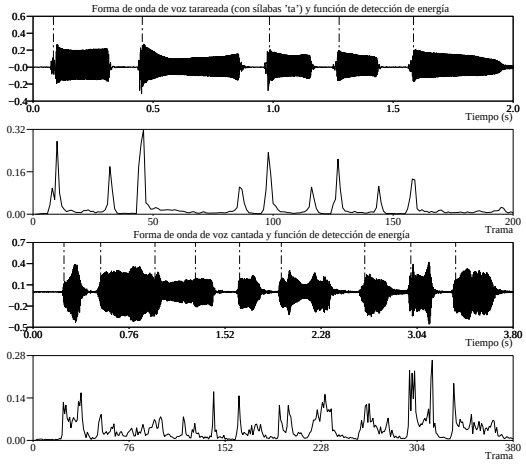


Figura 5.3: Función de detección basada en la energía para voz tarareada y voz cantada. La función no es robusta en el caso de voz cantada ya que los comienzos de nota pueden no ser pronunciados.

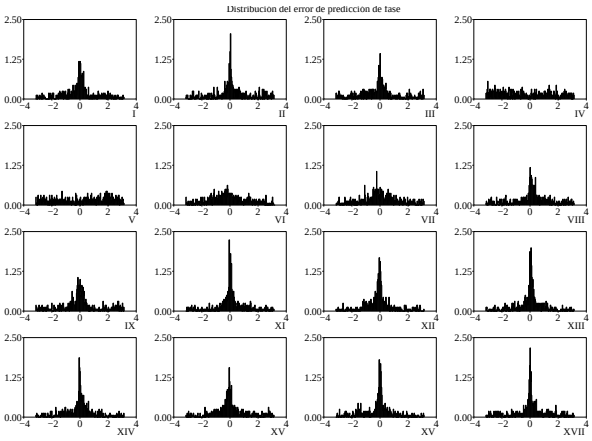


Figura 5.4: Histogramas del error de predicción de fase para notas de guitarra. Las figuras IV a VI marcan claramente el transitorio entre notas. En el resto de las figuras se aprecia la concentración de valores en cero.

tección en base a la distribución de los valores del error de predicción de fase $d\varphi$ (ecuación 5.3). Durante tramos estacionarios de la señal la predicción de fase es buena por lo que el valor de los componentes de $d\varphi$ es pequeño. En los transitorios la desviación de fase se incrementa y los componentes de $d\varphi$ toman valores mayores.

Considerando el conjunto de componentes de $d\varphi$ como un proceso aleatorio X donde cada componente $x \in X$ toma valores en el rango $[-\pi, \pi]$ (argumento principal), es posible observar su función densidad de probabilidad, por ejemplo, a través de un histograma. Esta función se asemeja, durante los estados estacionarios, a una distribución normal ya que presenta forma de campana, es simétrica respecto a su media y unimodal. En la figura 5.4 se observa una secuencia de histogramas que contienen el pasaje por un transitorio entre estados estacionarios. Cuando se presenta el transitorio, la desviación de fase aumenta por lo que los componentes de $d\varphi$ toman valores mayores y la distribución se vuelve más dispersa (crece la desviación estándar). Más tarde, al retornar a un estado estacionario, el error de predicción de fase decrece y aumenta la concentración de valores en cero, tomando la distribución una forma aguda con un pico más pronunciado. Cuantificando este comportamiento se pueden construir funciones de detección que permitan discriminar entre transitorios y estados estacionarios para indicar los comienzos de nota. Se han propuesto varias formas de observar la distribución de los valores de $d\varphi$, como la media del valor absoluto o la desviación estándar. En la figura 5.5 se puede observar la función de detección obtenida usando la desviación estándar. Los comienzos de notas corresponden a los picos marcados seguidos de valles, si bien su comportamiento es bastante ruidoso. Como una alternativa más robusta Bello y Sandler proponen el uso del rango interquartil (*IQR*, Interquartil Range). Este consiste en dividir el conjunto de datos en dos grupos, uno a cada lado de la media, y calcular la mediana de cada uno de ellos (m_1 y m_2). Luego el rango interquartil es la diferencia entre la mediana del grupo superior y el grupo inferior,

$$IQR = m_2 - m_1. \tag{5.4}$$

El rango interquartil es menos sensible que la desviación estándar a pequeñas variaciones en la dispersión de los componentes de $d\varphi$. En la figura 5.5 se observa su comportamiento.

En la mayoría de las señales reales la variación de frecuencia instantánea de las sinusoides que componen una nota puede ser grande incluso durante estados estacionarios. Esto da lugar a una

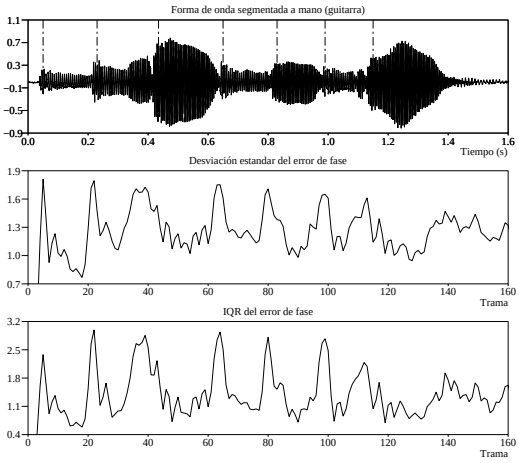


Figura 5.5: Funciones de detección basadas en la fase. Estas funciones son robustas en el caso de señales muy estacionarias.

distribución dispersa de $d\varphi$ y a una función de detección poco robusta. El análisis de dispersión puede ser complementado con un análisis de la forma de la distribución[25]. La Kurtosis es una medida de la agudeza o chatura de la forma de la distribución, y consiste en el cuarto momento central normalizado,

$$\gamma = \frac{\mu_4(\mu)}{\sigma^4}. \tag{5.5}$$

De acuerdo a su definición la Kurtosis decrece para distribuciones chatas y se incrementa para distribuciones agudas. De esta forma los picos presentes en la Kurtosis indican el inicio de estados estacionarios. Bello y Sandler proponen ubicar el comienzo del estado estacionario correspondiente a una nota mediante la Kurtosis y luego determinar el instante de inicio de la misma buscando el primer pico precedente en la función de rango interquartil.

Métodos de combinación. Como se discutió previamente, los eventos musicales no siempre van acompañados de un incremento pronunciado de energía. Un ejemplo de esto podría ser las notas producidas por los instrumentos de cuerda frotada. Por otro lado, el problema de estabilidad de la frecuencia instantánea hace que el método de la fase sea ineficaz, principalmente para voz cantada. Estas consideraciones conducen a la conclusión de que ambos métodos no pueden ser utilizados separadamente para la detección robusta de eventos.

La forma mas directa de combinar la información de amplitud y fase es la propuesta en [26]. Se obtienen las funciones de detección $\eta_e(m)$ y $\eta_f(m)$ basadas en la distribución de los vectores diferencia de energía $dE(m)$ y error de predicción de fase $d\varphi(m)$ respectivamente, construidas como,

$$\eta_e(m) = \text{media}(f_e^m(|x|)) \tag{5.6}$$

$$\eta_f(m) = \text{media}(f_f^m(|x|)) \tag{5.7}$$

donde $f^m(x)$ es la función densidad de probabilidad en la trama m. La función de detección combinada es simplemente,

$$\eta_T(m) = \eta_e(m) \times \eta_f(m) \tag{5.8}$$

y es motivada por el comportamiento similar de ambas funciones de detección. Como las dos funciones $\eta_e(m)$ y $\eta_f(m)$ se obtienen de manera independiente, el resultado es una función de detección mas robusta. Esto se puede observar en la figura 5.6.

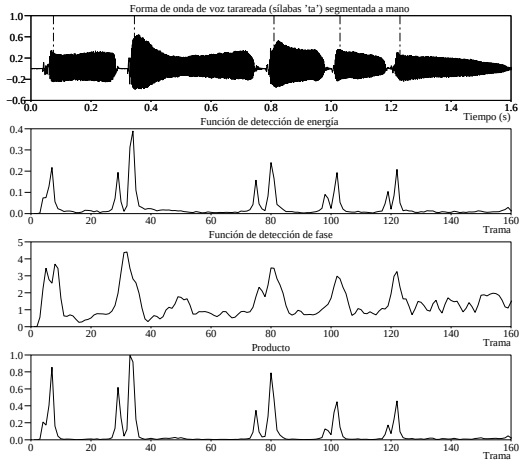


Figura 5.6: Combinación de la información. Producto de la función de detección de fase y la de energía.

Es posible considerar simultáneamente la información de fase y de energía combinándolas en el dominio complejo como propone Duxbury[34]. Durante estados estacionarios la amplitud y frecuencia instantánea de las componentes de la

señal se mantienen relativamente constantes y es posible predecir su valor de amplitud y fase. La amplitud y fase de cada componente está dada por los complejos de la FFT. Se puede obtener una función de detección midiendo la distancia Euclidiana en el dominio complejo entre la predicción y el valor real de cada índice. La predicción del k-ésimo índice de la FFT en su notación polar está dada por,

$$\hat{S}_k(m) = \hat{R}_k(m)e^{j\hat{\phi}_k(m)} \tag{5.9}$$

donde la predicción de amplitud $\hat{R}_k(m)$ corresponde a la amplitud de la trama anterior $R_k(m-1)$ y la predicción de fase $\hat{\phi}_k(m)$ se puede calcular a partir de la ecuación 5.3 como la suma de la fase previa y la diferencia de fase entre tramas sucesivas,

$$\hat{\phi}_k(m) = \mathrm{Arg}\big[2\tilde{\varphi}_k(m-1) - \tilde{\varphi}_k(m-2)\big]. \tag{5.10}$$

Considerando el complejo correspondiente a el k-ésimo índice de la FFT de la trama actual,

$$S_k(m) = R_k(m)e^{j\phi_k(m)} \tag{5.11}$$

podemos tomar la distancia a la predicción en el dominio complejo,

$$\begin{aligned}\Gamma_k(m) = \quad & \Big\{\big[\Re(\hat{S}_k(m)) - \Re(S_k(m))\big]^2 + \\ & \big[\Im(\hat{S}_k(m)) - \Im(S_k(m))\big]^2\Big\}^{\frac{1}{2}}.\end{aligned} \tag{5.12}$$

Este procedimiento se explica gráficamente en la figura 5.7. Sumando las distancias para todo k se obtiene la función de detección,

$$\eta(m) = \sum_{k=1}^{N} \Gamma_k(m). \tag{5.13}$$

En la figura 5.8 se comparan las funciones de detección propuestas.

Descripción del algoritmo

El algoritmo implementado utiliza la información de energía y fase a partir de la STFT combinadas en el dominio complejo. El módulo y la fase de cada índice de la FFT de la trama actual se comparan con la predicción obtenida de las dos tramas anteriores y se construye la función de detección $\eta(m)$ de la ecuación 5.13. En la figura 5.9 se aprecian las etapas del proceso. Se calcula la STFT para tramas de 20 ms con salto de 10 ms utilizando enventanado Hanning. Esto corresponde a tramas de largo L = 160 muestras a frecuencia de muestreo f_s = 8 kHz. Es importante contar con una buena resolución en frecuencia para desdoblar la fase ade-

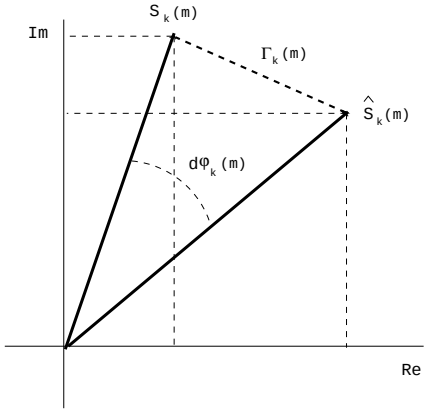


Figura 5.7: Diagrama fasorial en el dominio complejo donde se aprecia la distancia entre el vector real y su predicción. Se indica el error de predicción de fase $d\varphi_k(m)$.

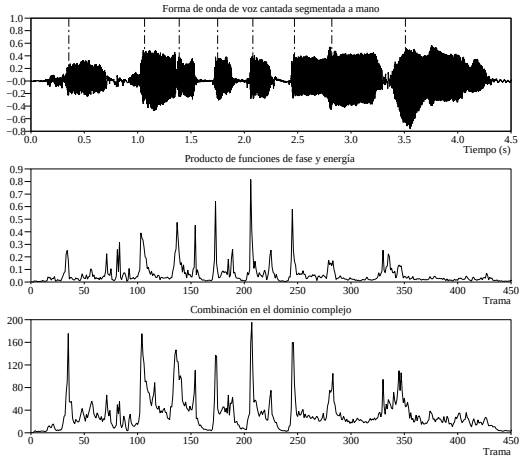


Figura 5.8: Comparación de las funciones de detección descriptas. La combinación en el dominio complejo es la forma mas adecuada de utilizar la información de amplitud y fase. Observar que también aparecen picos en los finales de notas abruptos.

cuadamente por lo que se utilizan transformadas de $N = 1024$ puntos.

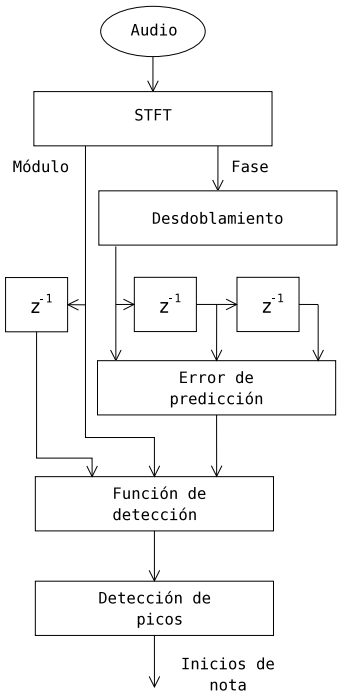


Figura 5.9: Diagrama de flujos del algoritmo en el dominio complejo.

Desdoblamiento de fase

El desdoblamiento de fase es necesario en el cálculo del error de predicción de fase de la ecuación 5.3. Al calcular la fase de un complejo se obtiene su argumento principal, por lo que la respuesta en fase dada por la FFT presenta discontinuidades[37]. Para poder comparar la fase de distintas tramas es necesario que ésta sea una función continua. En la figura 5.10 se observa la respuesta en fase de una trama con y sin desdoblamiento.

Cálculo de la función de detección

El cálculo de $\Gamma_k(m)$ puede simplificarse mapeando $\hat{S}_k(m)$ en el eje real. Esto implica rotar los fasores de la figura 5.7, de forma tal que el argumento de $S_k(m)$ sea la desviación de fase $d\varphi_k(m)$,

$$S_k(m) = R_k(m)e^{jd\varphi_k(m)}. \tag{5.14}$$

Con esta transformación, $\hat{S}_k(m) = \hat{R}_k(m)$ es un número real y la ecuación 5.12 queda,

$$\begin{aligned}\Gamma_k(m) = \quad & \Big\{\big[\hat{R}_k(m) - \Re(S_k(m))\big]^2 + \\ & \Im(S_k(m))^2\Big\}^{\frac{1}{2}}\end{aligned} \tag{5.15}$$

que equivale a,

$$\begin{aligned}\Gamma_k(m) = \quad & \Big\{\big[\hat{R}_k(m) - \\ & R_k(m)\cos(d\varphi_k(m))\big]^2 + \\ & \big[R_k(m)\,\mathrm{sen}(d\varphi_k(m))\big]^2\Big\}^{\frac{1}{2}}.\end{aligned} \tag{5.16}$$

Desarrollando y simplificando términos,

$$\begin{aligned}\Gamma_k(m) = \quad & \big\{\hat{R}_k(m)^2 + R_k(m)^2 - \\ & 2\hat{R}_k(m)R_k(m)\cos(d\varphi_k(m))\big\}^{\frac{1}{2}}.\end{aligned} \tag{5.17}$$

A partir de la FFT de las últimas tres tramas se evalúa la ecuación 5.3 para obtener $d\varphi_k(m)$ para todo k. La predicción del módulo del índice k-ésimo de la trama m se obtiene como $\hat{R}_k(m) = R_k(m-1)$. Con estos elementos se calcula $\Gamma_k(m)$ para todo k y se obtiene la función de detección evaluando la ecuación 5.13.

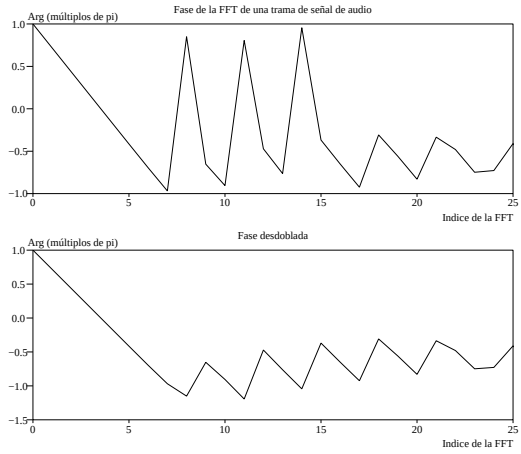


Figura 5.10: Desdoblemiento de fase. Es necesario para comparar la fase de tramas sucesivas.

Detección de picos

Como muestra la figura 5.8, establecer un umbral para la detección de los picos de la función de detección puede no ser sencillo. Esta función tiende a ser ruidosa y su amplitud varía considerablemente entre distintas señales.

Una alternativa para facilitar la detección es suavizar la función mediante un filtrado pasabajos. Esto sin embargo reduce la resolución temporal y puede eliminar los transitorios más débiles.

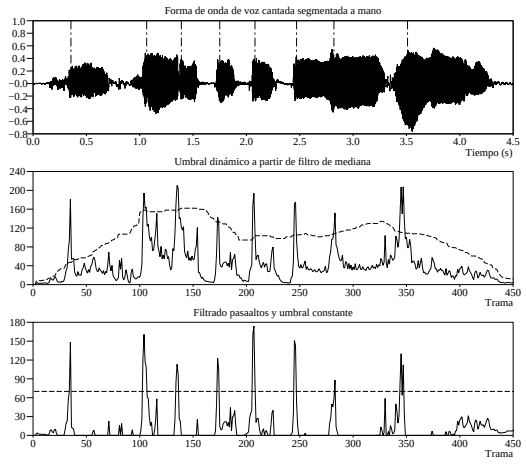


Figura 5.11: Propuestas del umbral para la detección de picos. El objetivo del umbral es la eliminación de picos espurios.

Bello y Sandler[25] proponen el uso de un umbral dinámico calculado a partir de la mediana de los valores de la función de detección en un intervalo de largo h,

$$U(m) = C\ \text{mediana}(\eta(n)), \quad n \in [m - \frac{h}{2}, m + \frac{h}{2}]$$

donde C es una constante de ajuste.

Otra alternativa es eliminar la componente de baja frecuencia que modula la amplitud de la función de detección mediante un filtrado pasaaltos y luego utilizar un umbral fijo. Se obtuvo mejores resultados de esta forma por lo que fue la opción adoptada en el algoritmo implementado. En la figura 5.11 se observa el umbral en cada caso.

5.5.2. Algoritmo basado en la derivada de la envolvente

El incremento de energía en la señal de audio cuando se produce un nuevo evento se manifiesta en un aumento de amplitud de la envolvente de la forma de onda. Tomando la derivada de la envolvente es posible construir una función de detección que presenta picos donde hay cambios bruscos de amplitud.

Los sistemas que utilizan la envolvente de amplitud obtienen buenos resultados en aplicaciones particulares como sonidos percutivos o detección de los eventos más salientes. Klapuri[28] propone extraer envolventes de amplitud para distintas bandas de frecuencia de la señal de audio logrando un sistema de detección más robusto. Scheirer[38] señaló la importancia de procesar la señal en bandas de frecuencia de forma similar al sistema auditivo y combinar finalmente la información resultante. Su trabajo apunta a que se pueden realizar ciertas simplificaciones sobre la señal de audio preservando la información rítmica. La experiencia que propone consiste en dividir una señal en bandas de frecuencia, extraer la envolvente de amplitud de cada banda y modular bandas de ruido con las envolventes de amplitud. En la señal que se obtiene al combinar las bandas de ruido modulado pueden reconocerse las mismas características rítmicas de la señal original. Esto no se cumple al trabajar con una única envolvente.

Separar la señal en bandas de frecuencia se asemeja en cierta forma al proceso de percepción del sistema auditivo, por lo que detectar los eventos en distintas bandas y luego combinarlos parece un camino razonable. Por ejemplo, cuando un nuevo evento está asociado con cambios de altura o cambios tímbricos puede producir la aparición repentina de información en alguna banda de frecuencia (ver figura 5.12). En el caso de música polifónica dividir la señal en bandas de frecuencia es a grandes rasgos separar la información de los distintos instrumentos.

En el algoritmo implementado la señal de audio es normalizada en amplitud y dividida en 6 bandas de frecuencia. Se extrae la envolvente de cada banda y se obtiene su derivada. Esta señal es utilizada para determinar candidatos de comienzo de nota, a los cuales se les asigna un valor de intensidad. Finalmente se combina la información de

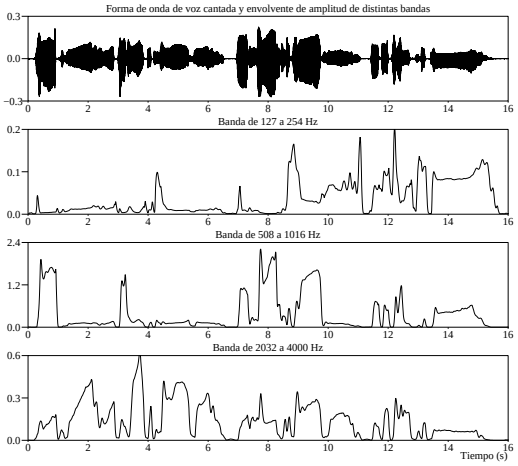


Figura 5.12: Envolventes de amplitud de distintas bandas de frecuencia. Los eventos aparecen mas claramente en alguna banda que en la envolvente de la señal lo que facilita su detección.

las distintas bandas y se seleccionan los candidatos más prominentes. En la figura 5.13 se observa un diagrama del proceso.

Filtrado en bandas

Dado que para la selección y combinación de candidatos de inicio de nota se utilizan conceptos psicoacústicos, el filtrado en bandas debe guardar relación con la discriminación en frecuencia del sistema auditivo. Por esta razón se utiliza un filtrado en bandas de octava, como se propone en [38][39].

La señal de audio muestreada a 8 kHz se separa en 6 bandas de frecuencia usando filtros Butterworth de orden 6. La primer banda está dada por un filtro pasabajos de frecuencia de corte 127 Hz, las siguientes 4 bandas corresponden a filtros pasabanda de frecuencias de corte 127 y 254 Hz, 254 y 508 Hz, 508 y 1016 Hz, y 1016 y 2032 Hz, y la última banda se obtiene mediante un filtro pasaaltos de frecuencia de corte de 2032 Hz. Se pudo observar, así como señala Scheirer, que el algoritmo no es sensible a la elección particular de las frecuencias de corte (manteniendo el criterio de octavas), ni a la implementación de los filtros.

Extracción de envolventes

Las señales correspondientes a cada banda se rectifican onda completa y se deciman a una frecuencia de muestreo de 100 Hz (factor de decimación de 80) para reducir el costo computacional de las siguientes etapas. Luego se obtienen las envolventes convolucionando cada señal con una ventana mitad Hanning (coseno elevado) de 100 ms. Esta ventana tiene una respuesta pasabajos con frecuencia de corte de aproximadamente 10 Hz y produce una integración de potencia similar a la que realiza el sistema auditivo, manteniendo cambios repentinos pero enmascarando modulaciones rápidas[38].

Cálculo de la derivada

Tradicionalmente los algoritmos que trabajan con envolventes de amplitud calculan su derivada de primer orden y asocian los puntos de máxima pendiente al comienzo de notas.

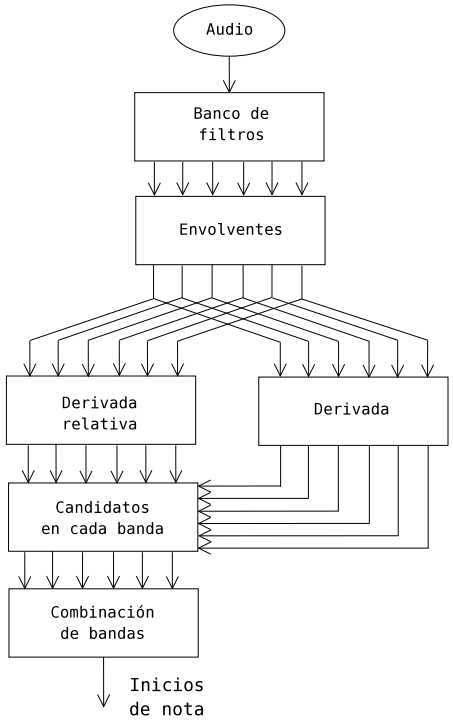


Figura 5.13: Diagrama de flujos del algoritmo basado en la derivada de la envolvente.

Klapuri[28] señala que la derivada de primer orden es apropiada para reflejar la sonoridad (intensidad) de las notas pero sus máximos no indican adecuadamente el instante de su inicio. Esto se debe en primer lugar a que la envolvente correspondiente al inicio de un sonido no es monótonamente creciente en la mayoría de los casos, dando lugar a varios máximos locales en la derivada de primer orden. Asimismo, muchos sonidos (en particular los graves) tardan cierto tiempo en alcanzar el punto donde su amplitud crece con pendiente máxima, y éste es posterior al inicio físico del sonido. Por lo anterior Klapuri propone utilizar la derivada relativa, dividiendo la derivada de primer orden entre la envolvente de amplitud,

$$D_r(t) = \frac{\frac{d}{dt}A(t)}{A(t)} \tag{5.18}$$

donde A(t) indica la función envolvente de amplitud. Esto tiene una relevancia importante desde el punto de vista psicoacústico. El incremento de amplitud percibido por el sistema auditivo está relacionado con la intensidad del sonido, siendo un mismo incremento más relevante para sonidos de menor amplitud. Del mismo modo, el mínimo incremento de intensidad perceptible (ΔI) es aproximadamente proporcional a la intensidad del sonido. Esto quiere decir que la relación $\Delta I/I$ (fracción de Weber) es constante. Para una sinusoide de frecuencia f, $I(t) = A(t) * f$ representa la intensidad percibida[1]. La función derivada de amplitud relativa $D_r(t)$ coincide entonces con la fracción de Weber. La función propuesta resuelve los problemas mencionados asociados a la derivada de la envolvente. Las oscilaciones de amplitud que ocurran durante el inicio de una nota serán poco relevantes al considerar la derivada relativa. Por otra parte el máximo de la derivada relativa se ubica antes que el máximo de la derivada, coincidiendo con el inicio físico de la nota (ver figura 5.14). Tomar la derivada relativa como en la ecuación 5.18 es equivalente a considerar la derivada del logaritmo de la envolvente de amplitud. Una forma numéricamente flexible de tomar el logaritmo es a través de la ley μ de compresión[40],

$$y(t) = \frac{\ln[1 + \gamma A(t)]}{\ln[1 + \gamma]}. \tag{5.19}$$

[1] Esto es una aproximación muy gruesa de las curvas de Fletcher y Munson.

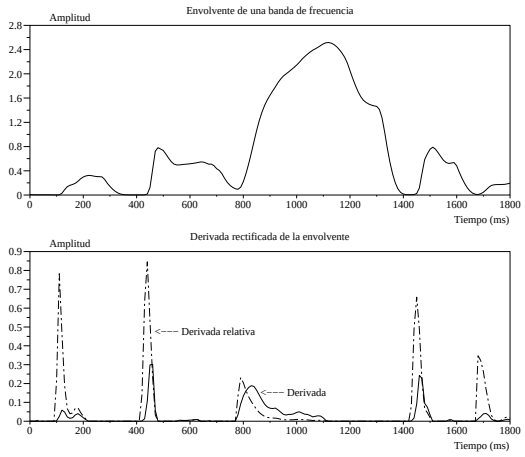


Figura 5.14: Envolvente de amplitud, derivada y derivada relativa. El máximo de la derivada relativa coincide con el inicio físico de las notas.

La constante γ permite ajustar la transformación haciéndola más lineal ($\gamma < 0{,}1$) o más logarítmica ($\gamma > 1000$). En el algoritmo implementado se calcula la derivada de la ley μ de las envolventes de amplitud usando un valor de γ = 100, el cual se estableció en base a simulaciones.

Intensidad de componentes

En la función derivada relativa se identifican los picos y se les asocia un valor de intensidad según el cual serán considerados correspondientes a eventos. Como ya se mencionó, Klapuri señala que la derivada de la envolvente refleja adecuadamente la intensidad (o sonoridad) del evento. Por esta razón el valor de intensidad se establece como el primer máximo de la derivada de amplitud a partir del instante del pico de la derivada relativa.

Selección de candidatos

La función derivada relativa puede presentar picos espurios que no corresponden a inicios genuinos de nota, sino a variaciones en la pendiente durante el ataque. Modulaciones de amplitud debidas a vibrato, respiración, etc, originan también la aparición de picos indeseables. Como la mayoría de estos casos ocurren en puntos donde

la amplitud de la envolvente es considerable, los picos de la derivada relativa son pequeños. Ambos fenómenos pueden apreciarse en la figura 5.15. Como forma de evitar este problema se adopta-

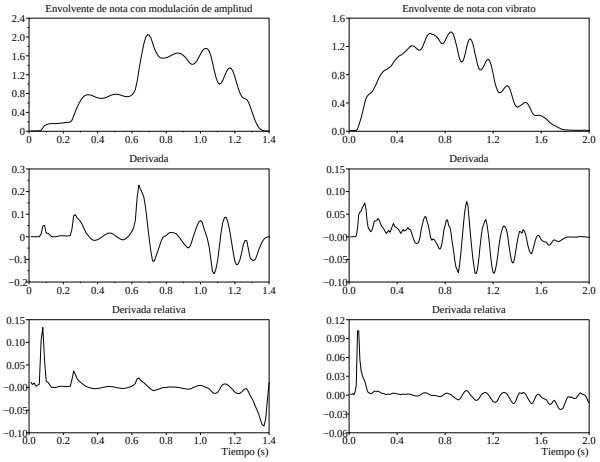


Figura 5.15: La derivada tiene picos espurios debido a modulaciones de amplitud en la envolvente. Este problema es atenuado mediante la derivada relativa pero no completamente eliminado, lo que hace necesario el uso de umbrales. Observar que la derivada relativa indica claramente el comienzo de las notas.

ron dos criterios de selección de candidatos en las distintas bandas. El primero consiste en establecer un umbral sobre la amplitud de la derivada relativa, eliminando los candidatos que no lo superen (ver figura 5.16). El segundo criterio es descartar los candidatos que se encuentran a menos de 50 ms de otro de mayor intensidad.

Los candidatos que aún sobreviven se validan si su intensidad supera un umbral dinámico que se extiende por 200 ms desde la intensidad del candidato previo y decae hasta la mitad de su valor (ver figura 5.17). Este criterio propuesto por Uhle y Herre[41] es motivado por el hecho de que no es posible un cambio instantáneo de fortissimo a pianissimo. Esto soluciona por ejemplo, el problema de la detección de múltiples notas en el caso de señales con fuerte reverberación.

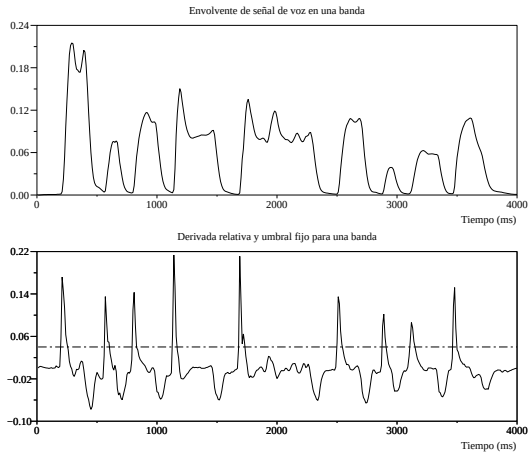


Figura 5.16: Umbral estático sobre la derivada relativa. Los picos que lo superan son candidatos a comienzos de nota.

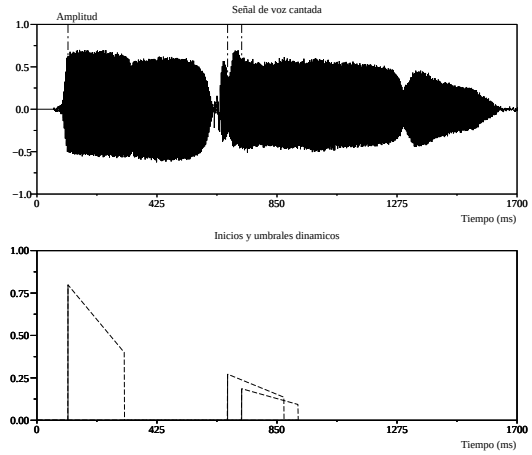


Figura 5.17: Umbral dinámico sobre la intensidad de los candidatos en cada banda. El tercer candidato, que no corresponde a un evento genuino, es eliminado ya que su intensidad es inferior al umbral.

Combinación de las bandas

La elección definitiva de los inicios de nota se realiza combinando la información de candidatos de las distintas bandas.

Klapuri originalmente utiliza el modelo psicoacústico de sonoridad de Moore para combinar

la información de las distintas bandas[28]. Posteriormente simplifica el modelo argumentando que muchos de los detalles psicoacústicos resultaron poco críticos en el desempeño del algoritmo[40]. Un modelo simplificado similar a éste es el que se adopta en el presente trabajo.

Se asume que candidatos cercanos menos de 50 ms corresponden al mismo evento, dando origen a un único candidato ubicado en la mediana sus posiciones. Su valor de intensidad se obtiene como el producto entre el máximo de intensidad del conjunto y el número de candidatos. Los siguientes pasos de selección son similares a los de elección de candidatos en una banda, es decir, un umbral de amplitud fijo y otro dinámico. El umbral dinámico en este caso se construye como una función de decaimiento que se extiende 200 ms desde el candidato previo y 100 ms desde el candidato posterior, como se aprecia en la figura 5.18.

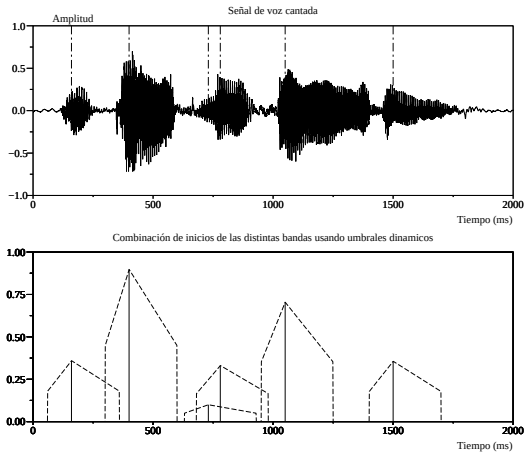


Figura 5.18: Umbral dinámico sobre la intensidad de los candidatos en la combinación de las bandas. El tercer candidato es eliminado mediante este criterio. Los eventos que superan el umbral se consideran los comienzos de nota de la señal de audio.

5.6. Evaluación

La evaluación formal de un sistema de detección de comienzos de nota es una tarea muy delicada que debe ser realizada por un experto (por ejemplo, un músico). El modo usual de proceder es marcar a mano los eventos de una base de datos de audio de prueba y luego comparar los resultados con los obtenidos por el sistema. La mayor dificultad radica en encontrar el momento preciso de ocurrencia de los eventos en el proceso de marcado, sobre todo en señales de voz cantada. Por estos motivos, en la presente sección no se pretende hacer una evaluación formal de los algoritmos implementados, solamente se intenta reflejar a grandes rasgos el comportamiento del sistema de detección de eventos elegido para la aplicación final.

Luego de algunas pruebas básicas con señales de voz cantada, resultó evidente que el desempeño del algoritmo basado en la derivada es superior al algoritmo en el dominio complejo. Las razones de ello se explican en la sección 5.7. Por lo tanto, éste es el algoritmo evaluado a continuación.

5.6.1. Método de evaluación

Los experimentos fueron realizados sobre una amplia variedad de señales monofónicas de diversa complejidad. El primer grupo de señales está compuesto por catorce fragmentos extraídos principalmente de grabaciones comerciales de instrumentos musicales. El segundo grupo está constituido por melodías de voz cantada divididas en tres bases de datos que se detallan a continuación:

BD1 Un extracto de la canción *Memory* grabado por 3 cantantes profesionales[23].

BD2 Siete fragmentos de canciones grabadas por cantantes no profesionales.

BD3 Cinco melodías tarareadas grabadas por cantantes no profesionales.

Las bases de datos *BD2* y *BD3* son las de mayor relevancia ya que pueden considerarse consultas a un sistema de búsqueda de audio por contenido.

La evaluación consiste en analizar la cantidad de aciertos y errores cometidos por el algoritmo al procesar una señal de audio. Es importante aclarar que no se tiene en cuenta el error en la ubicación exacta del evento, solamente se considera si el evento está presente o no en la señal. Para cuantificar el comportamiento del sistema se definen los siguientes valores,

1. *ET*: Número total de eventos en la señal.

2. *ED*: Número total de eventos detectados correctamente.

3. *EA*: Número total de eventos adicionales detectados, es decir, eventos detectados que no aparecen en la señal.

4. *Error(%)*: Porcentaje total de errores, definido como,

$$\text{Error}(\%) = 100 \times \frac{(ET - ED) + EA}{ET}. \quad (5.20)$$

5.6.2. Resultados

En las figuras 5.19 y 5.20 se muestran los resultados para los dos grupos de señales evaluados.

Instrumento	ET	ED	EA	Error(%)
Guitarra	110	86	0	21.8
Violín	43	24	2	48.8
Bajo	29	29	0	0
Saxofón	16	13	0	18.8
Piano	43	28	0	34.8
Total	**241**	**180**	**2**	**26.1**

Figura 5.19: Resultados para distintos instrumentos musicales.

BD	ET	ED	EA	Error(%)
BD1	66	59	3	15.2
BD2	103	88	5	18.5
BD3	45	45	0	0.0
Total	**214**	**192**	**8**	**13.6**

Figura 5.20: Resultados para voz cantada.

Respecto a las señales del primer grupo, el desempeño mas pobre es para las grabaciones de violín. En general, los ataques producidos por instrumentos de cuerda frotada son débiles, lo que dificulta su detección a través del método de la derivada. Los malos resultados para las señales de piano radican en la complejidad de las melodías escogidas para la evaluación.

Para señales de voz cantada, en términos generales, el comportamiento es bueno. Es importante destacar que el algoritmo prácticamente no comete errores en el caso de melodías tarareadas (*DB3*).

Un aspecto crítico del algoritmo es el ajuste de los parámetros. Establecer los umbrales implica un compromiso entre la cantidad de aciertos (*ED*) y errores cometidos (*EA*), ya que al reducir su valor disminuyen las restricciones para validar eventos. Como se explicó en la sección 5.3, el objetivo del algoritmo de segmentación es complementar el contorno de frecuencia fundamental para la transcripción de una melodía cantada. Al combinar esta información, no es posible descartar los eventos adicionales del algoritmo de segmentación a partir de la frecuencia fundamental. Por ejemplo, no es correcto utilizar como criterio para descartar eventos adicionales el hecho de que la altura no cambie (ver figura 5.1). Por el contrario, la detección de eventos faltantes mediante la frecuencia fundamental es en general posible. La excepción es cuando el evento faltante no va acompañado de un cambio de altura. Estas consideraciones condujeron a elegir los parámetros del algoritmo de segmentación de forma de evitar la detección de eventos adicionales.

5.7. Conclusiones

Se estudiaron técnicas de detección de eventos en señales de audio y se desarrollaron dos algoritmos para la detección automática de los inicios de nota en una melodía cantada.

Las técnicas propuestas para la detección de eventos musicales funcionan adecuadamente en aplicaciones particulares en donde los inicios de nota son marcados. Los sistemas evaluados con música que contiene voz cantada reportan resultados pobres para ese caso[25][28].

La voz cantada tiene características que hacen que sea menos estacionaria que otro tipo de instrumentos. La utilización de técnicas que discriminan entre transitorios y estados estacionarios para la detección de los inicios de nota es menos efectiva para señales de voz. Por otro lado, el método tradicional de la derivada de la envolvente produce buenos resultados si bien presenta dificultades para la detección de transitorios débiles. Se concluye entonces que para trabajar con señales de voz este método es la opción más adecuada.

6

Detección de tempo

6.1. Resumen

En este capítulo se analizan técnicas de detección automática de tempo y tiempo de pulso. Esta información junto con la altura e inicio de las notas es importante para la completa representación de una melodía. Se trabaja sobre señales de voz cantada sin conocimiento previo de sus características rítmicas.

6.2. Introducción

La tarea de un algoritmo de detección automática de tempo es la de realizar lo que intuitivamente hacemos golpeando los pies o manos para seguir el ritmo de la música. El beat o pulsación se refiere a el conjunto de impulsos igualmente espaciados que se perciben al escuchar un tramo de música y están relacionados con la tasa de ocurrencia y acentuación de las notas. El tempo es la frecuencia de estos impulsos, expresada en unidades de duración de notas por unidad de tiempo (a pesar de que usualmente se habla de pulsos por minuto, bpm). Ambos conceptos son parte del fenómeno mas complejo del ritmo, pero este trabajo se concentra solo en la estimación del tempo y el tiempo de pulso.

La secuencia de pulsos de beat puede describirse en términos de frecuencia y fase, como cualquier señal periódica. La frecuencia de los pulsos corresponde al tempo, mientras que la fase indica el tiempo de ocurrencia. Un algoritmo de detección automática debe ser capaz de determinar tanto la frecuencia como la fase de los pulsos.

Una de las aplicaciones mas importantes consiste en la transcripción automática de música. Conocer el pulso permite cuantificar la duración de las notas y ubicarlas en relación al tiempo de pulso. Imponiendo cierta estructura métrica, el comienzo del compás y la figura a la que corresponde la duración de los pulsos (negras, corcheas) es posible la transcripción a una partitura. Por otro lado, al comparar melodías, la ubicación relativa de las notas respecto al compás permite diferenciar aquellas que desde el punto de vista de alturas y duraciones son equivalentes. Esto puede ser de utilidad en la búsqueda de música en bases de datos usando una melodía como consulta. Asimismo utilizar el tempo en la codificación para la búsqueda de una melodía cantada permite reducir los rasgos expresivos ajustando la duración y ubicación de las notas, así como independizarse del tempo de la interpretación. Otras aplicaciones pueden ser edición de audio, sincronismo de imágenes y audio, acompañamiento automático.

Se han realizado gran cantidad de trabajos en relación a este tema utilizando distintas técnicas. Debido a la complejidad del problema se desarrollaron algoritmos que funcionan adecuadamente para música polifónica en un rango acotado de estilos musicales, la mayoría de los cuales tienen un ritmo marcado. La tarea específica de estimar el tempo de un tramo corto de melodía cantada no ha sido abordada aún[42].

Determinar el tempo de un fragmento de una melodía en ausencia de referencias (como la presencia de otros instrumentos) puede ser un problema mal condicionado. En algunos casos la ubicación y acentuación de las notas puede no aportar suficiente información incluso para un experto.

6.3. Objetivo

El objetivo en este trabajo, es estudiar técnicas de extracción automática de tempo para señales de audio, implementar algoritmos y evaluar la posibilidad de usar esta información en la transcripción de tramos cortos de una melodía cantada.

6.4. Resumen de técnicas utilizadas

Scheirer[38] propone un método de extracción de tempo utilizando una simplificación de la señal de audio que preserva la información rítmica. Esto se basa en la experiencia psicoacústica de separar la señal en distintas bandas de frecuencia, obtener la envolvente de amplitud de la señal de cada banda y modular bandas de ruido con dichas envolventes. En la señal que se obtiene al sumar las distintas bandas de ruido modulado, se pueden reconocer las mismas características rítmicas de la señal original si el número de bandas es mayor a 4 (ver figura 6.1). Dado que lo único que se conserva en la transformación son las envolventes de amplitud, se concluye que para extraer el tempo es suficiente trabajar con dicha información [1].

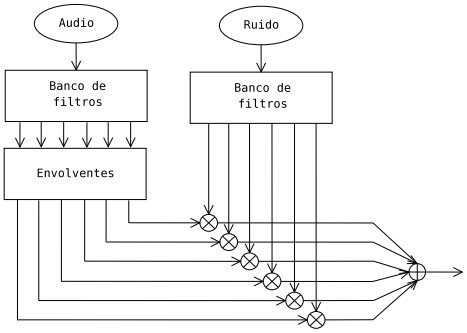


Figura 6.1: La señal de salida contiene la misma información rítmica que la entrada .

El algoritmo propuesto separa la señal original en 6 bandas de frecuencia, extrae las envolventes de amplitud de cada banda, las deriva y rectifica media onda. Estas señales pasan por un banco de 150 filtros resonantes (filtros peines). Cada filtro resuena a un período determinado, de forma tal que el banco de filtros cubre el rango (discreto) de tempo que se quiere estimar. Luego se halla la energía a la salida de cada filtro y se suman los valores de un mismo filtro para cada banda. El período del filtro que presente energía máxima representa el tempo buscado. Un diagrama de flujos del sistema se puede ver en la figura 6.2. Una vez hallado el tempo de un tramo de señal, la fase de los pulsos se obtiene como el tiempo donde se ubica el máximo de la suma de las salidas del filtro correspondiente al tempo estimado.

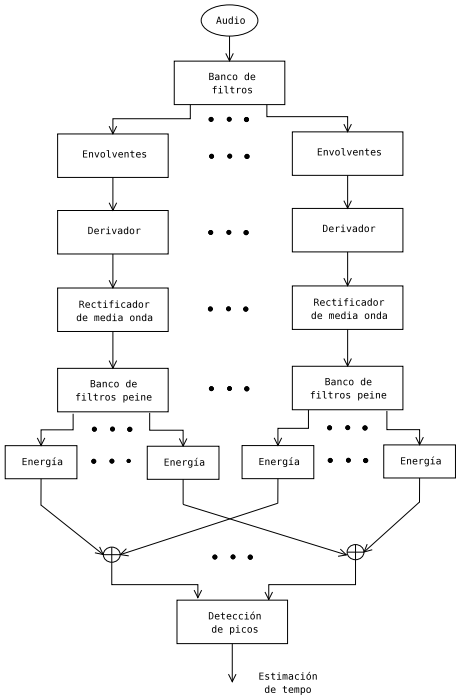


Figura 6.2: Diagrama del método de Scheirer.

El sistema fue evaluado cualitativamente por el autor para extractos cortos de música de varios estilos, hallando el tempo correctamente en 41 de 60 ejemplos.

Goto y Muroaka[43] han presentado dos métodos de detección automática de tempo para música popular. Uno de ellos fue desarrollado para música que contiene instrumentos de percusión. El otro trabajo realiza un análisis en el dominio de la frecuencia para detectar cambios de acordes,

[1] Pueden encontrarse ejemplos en la página del autor http://sound.media.mit.edu/eds/beat/

asumiendo que estos ocurren en posiciones métricas particulares. Este sistema utiliza información de alto nivel específica del estilo de música analizado lo cual constituye una limitante.

Klapuri[40] propone un método basado en la técnica de Scheirer, en el que introduce otra forma de calcular la derivada de las envolventes y normalizaciones sobre la salida de los filtros peine. En su trabajo reporta una notoria mejora en comparación con el trabajo de Scheirer.

Dixon[44] se basa en el supuesto de que el ritmo es una propiedad de bajo nivel y que la información que requiere un algoritmo de detección automática de tempo es el tiempo de inicio de los eventos musicales. La señal de audio se procesa detectando los eventos rítmicos mas prominentes. Los instantes en que estos se producen son analizados para generar hipótesis de tempo y beat. Estas hipótesis se comparan determinando cual de ellas se ajusta mejor al tramo de música.

6.5. Algoritmos implementados

Se seleccionaron aquellas técnicas aplicables a la detección de tempo de una melodía cantada y se implementaron dos algoritmos. Uno de ellos se basa en la técnica propuesta por Scheirer con modificaciones, algunas de las cuales sugiere Klapuri. También se implementó el método de Dixon sustituyendo su técnica de detección de eventos por la desarrollada en este trabajo (ver sección 5.5.2).

Ambas técnicas asumen la ocurrencia de eventos pronunciados en los tiempos de pulso. Esto sin embargo no es lo más frecuente en algunos estilos musicales. Asimismo, en el caso de una melodía despojada de los instrumentos que la acompañan, puede ocurrir que las notas en tiempos de pulso sean esporádicas. De lo anterior se deduce que la aplicación de las técnicas desarrolladas es limitada, particularmente para melodías monofónicas cantadas.

6.5.1. Detección utilizando filtros peine

Como se mencionó anteriormente, Scheirer argumenta que las envolventes de amplitud en bandas de frecuencia son suficientes para estimar el tempo de una señal de audio. Una vez obtenidas las envolventes se examina su periodicidad a través de filtros resonantes y se combina la información de las distintas bandas para estimar el tempo y la ubicación de los pulsos. A continuación se describe la implementación de cada una de sus etapas.

Filtrado en bandas y extracción de envolventes

La señal se divide en 6 bandas usando filtros elípticos de orden 6, con 3 dB de ripple en la banda pasante y 40 dB de rechazo en la región suprimebanda. En la banda inferior se utiliza un filtro pasabajos con frecuencia de corte de 200 Hz, en las siguientes 4 bandas se implementan filtros pasabanda con frecuencias de corte de 200 y 400 Hz, 400 y 800 Hz, 800 y 1600 Hz, 1600 y 3200 Hz; en la última banda se usa un filtro pasaaltos de frecuencia de corte 3200 Hz. La figura 6.3 muestra la respuesta en magnitud de estos filtros.

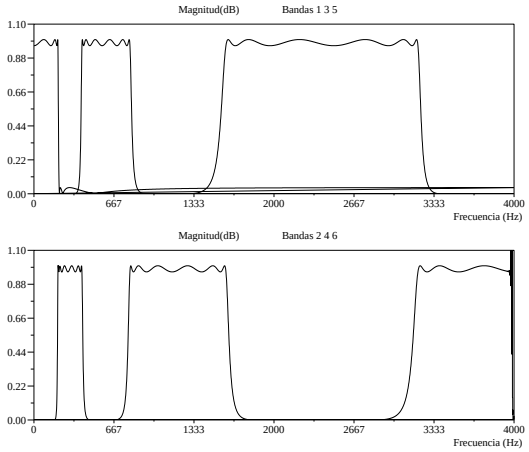


Figura 6.3: Respuesta en magnitud del banco de filtros elípticos.

Para hallar las envolventes en cada banda se deciman las señales a una frecuencia de muestreo de 200 Hz (para reducir el costo computacional), se rectifican onda completa y se convolucionan con una ventana mitad Hanning de 200 ms (ver figura 6.4). Esta ventana tiene una característica pasabajos con frecuencia de corte de aproximadamente 10 Hz. La convolución con la ventana produce una integración de energía similar a la del sistema auditivo del ser humano, preservando los cambios repentinos pero enmascarando modulaciones

rápidas. De esta forma las envolventes obtenidas son representativas desde un punto de vista psicoacústico.

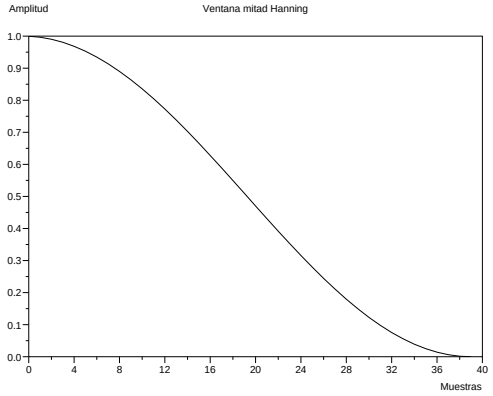


Figura 6.4: Ventana mitad Hanning.

Cada envolvente se comprime según la ley μ (ver sección 5.5.2) y se obtiene la derivada de primer orden. Este proceso es una forma flexible de calcular la derivada relativa de una señal, la cual indica el inicio de los eventos musicales mejor que la derivada. Esta señal rectificada media onda es la que se examina para hallar su periodicidad.

Banco de filtros resonantes

Luego de extraer y procesar cada envolvente se utiliza un banco de filtros peine para hallar el tempo de la señal de audio. Cabe mencionar que para la estimación de tempo se podría utilizar cualquier método que detecte la periodicidad de una señal (por ejemplo el método de la autocorrelación). Sin embargo este método mantiene la información de fase, necesaria para determinar cuando ocurren los pulsos.

Un filtro peine con retardo T y ganancia α está dado por una ecuación recursiva en el tiempo que depende de la entrada $x(t)$ actual y de la salida $y(t)$ retardada un período,

$$y(t) = \alpha y(t-T) + (1-\alpha)x(t). \tag{6.1}$$

Si la entrada a este filtro es una señal periódica su salida converge a un valor máximo cuando el período coincide con el retardo del filtro. Al excitar un filtro peine de retardo T con un tren de pulsos de período τ y ganancia A su salida se refuerza si $T = \tau$,

$$\begin{aligned} y_0 &= (1-\alpha)A \\ y_\tau &= \alpha(1-\alpha)A + (1-\alpha)A = (1-\alpha)A(1+\alpha) \\ y_{2\tau} &= (1-\alpha)A(\alpha^2+\alpha+1) \\ &\vdots \\ y_{n\tau} &= (1-\alpha)A\sum_{i=0}^{n}\alpha^i \end{aligned}$$

Por lo tanto,

$$\lim_{n\to\infty} y_{n\tau} = \frac{(1-\alpha)A}{(1-\alpha)} = A. \tag{6.2}$$

Por otro lado si el período del tren de pulsos es distinto a T la convergencia será a un valor menor. Si λ es el mínimo común múltiplo entre T y τ, la salida se refuerza cada un tiempo λ. Aplicando una lógica similar a la anterior se obtiene,

$$\lim_{n\to\infty} y_{n\lambda} = \frac{(1-\alpha)A}{1-\alpha^{\lambda/T}}. \tag{6.3}$$

Para que el filtro sea estable se debe cumplir que $|\alpha| < 1$ y dado que $\lambda/T \geq 1$,

$$1-\alpha^{\lambda/T} \geq 1-\alpha. \tag{6.4}$$

De lo anterior se deduce que la salida para una entrada de período que coincide con el retardo del filtro peine, será de mayor energía que para una entrada de cualquier otro período.

Este ejemplo se puede generalizar para toda señal periódica haciendo un análisis en frecuencia. El módulo de la respuesta en frecuencia de un filtro peine corresponde a

$$|H(e^{j\omega})| = \left|\frac{1-\alpha}{1-\alpha^{-j\omega\tau}}\right| \tag{6.5}$$

y presenta τ polos en $\omega = 2\pi n/\tau$ con $0 \leq n < \tau$. La salida será mayor para señales de período $T = \tau$ ya que los picos de la respuesta en frecuencia del filtro se alinean con la distribución de energía de la señal.

Para cada una de las 6 bandas de frecuencia, se implementa un banco de 60 filtros peine, donde el retardo τ de los filtros varía cubriendo el rango de

tempo de 60 a 180 bpm. Esto último implica una resolución del algoritmo de 2 bpm.

La respuesta al impulso de cada filtro peine tiene un decaimiento exponencial y toma su valor de energía media para un tiempo t, tal que $\alpha^{t/\tau} = 0{,}5$. Scheirer afirma que el compartamiento de los filtros se asemeja a la percepción humana para un valor de tiempo de energía media entre 1500 y 2000 ms. Para la implementación se tomó un valor de $t = 1500$ ms, lo cual determina un valor de α para cada filtro.

Considerando las últimas τ salidas de los filtros se calcula su energía y para cada uno de ellos se suman los valores de las distintas bandas de frecuencia. Este resultado se divide entre τ y se examina determinando el filtro con salida máxima como se propone en [40]. El tempo de la señal se estima a partir de su retardo τ como $bpm = 60/\tau$. En las figuras 6.5 y 6.6 se grafica la energía de la salida de los filtros peine para un tren de pulsos y una señal de voz cantada respectivamente.

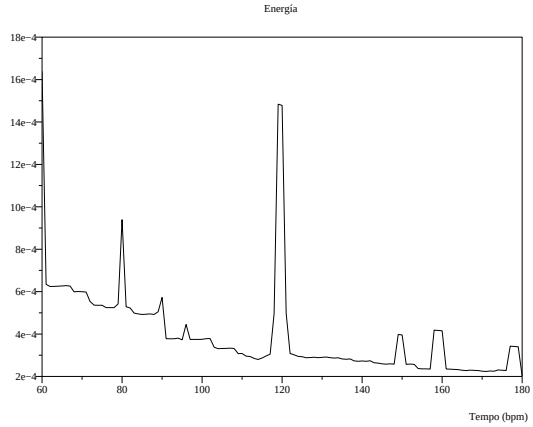


Figura 6.5: Energía a la salida de los filtros peines para un tren de pulsos de 120 bpm.

Determinación de la fase

Una vez determinado el tempo del tramo de señal analizado, es posible de forma directa estimar el tiempo en donde se perciben los pulsos. En el algoritmo implementado la fase del pulso se calcula como el valor temporal del máximo de la suma de las salidas del filtro correspondiente al tempo estimado (ver figuras 6.7 y 6.8).

Observaciones

El algoritmo se probó con distintos tipos de señales. Para el caso de un tren de pulsos periódico es capaz de estimar correctamente el tempo y el

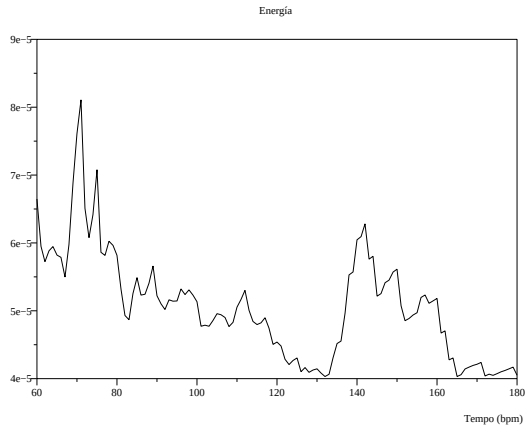


Figura 6.6: Energía a la salida de los filtros peines para una señal de voz cantada de 71 bpm.

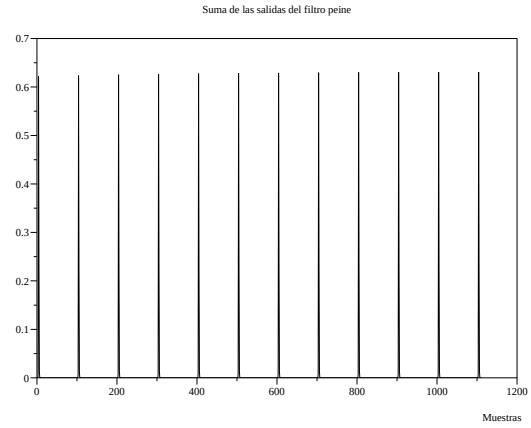


Figura 6.7: Salida del filtro peine para un tren de pulsos de 120 bpm. El período es de 100 muestras lo que a una frecuencia de muestreo de 200 Hz corresponde a 0.5 segundos.

tiempo de pulso. La salida de los filtros peine presenta picos bien marcados en este tipo de señales.

Se estudió cuan robusto es el método a variaciones en la periodicidad del tren de pulsos. Agregando ruido blanco a la ubicación de los pulsos se observó que si la potencia del ruido es mayor al 5 % del período del tren de pulsos la salida se ve fuertemente distorsionada.

En señales de audio polifónico de tempo marcado el algoritmo funciona correctamente a pesar de que la salida de los filtros peine es menos clara. Por otro lado, si el tempo no es marcado (como en algunos pasajes de música clásica), la salida de los filtros peine es ruidosa no pudiéndose estimar correctamente el tempo.

Las señales de voz cantada monofónicas son las mas relevantes en relación a la aplicación final del algoritmo. Si la melodía cantada presenta un tempo muy marcado, el algoritmo funciona correctamente. Sin embargo en la mayoría de los casos la salida de los filtros peine no tiene un pico bien definido provocando una estimación incorrecta.

Una característica importante del algoritmo es su elevado costo computacional. La etapa que involucra mayor número de operaciones es el banco de filtros peine.

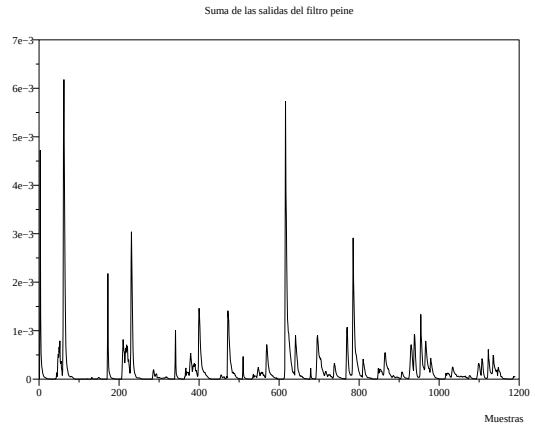


Figura 6.8: Salida del filtro peine para una señal de voz cantada de 71 bpm. El período de la señal es de aproximadamente 170 muestras lo que a una frecuencia de muestreo de 200 Hz corresponde a 0.85 segundos.

6.5.2. Detección a partir del inicio de notas

La técnica propuesta por Dixon[44] se basa en que la información rítmica está dada por la ubicación temporal de los eventos musicales (notas, acordes, sonidos de percusión). Cada evento es caracterizado por su prominencia y tiempo de inicio.

La primer etapa del algoritmo consiste en extraer los eventos musicales de la señal de audio. Mediante la distancia entre eventos se construye un conjunto ordenado de hipótesis de tempo. La ubicación de los primeros eventos se toma como hipótesis de fase. Esta información se combina para formar un grupo de agentes cada uno de los cuales representa una hipótesis de tempo y fase. Luego cada agente recorre el conjunto de eventos en forma secuencial. Por cada evento que confirme las hipótesis del agente, la probabilidad de que estas estimaciones de tempo y fase sean las correctas se incrementa. Finalmente se selecciona el agente con mayor probabilidad y sus hipótesis son las estimaciones del algoritmo.

A continuación se detallan cada una de las etapas.

Extracción de eventos

El algoritmo de estimación de tempo utiliza el tiempo de inicio y la sonoridad de los eventos. Para extraer esta información Dixon calcula la derivada de la envolvente de amplitud de la señal y detecta los picos determinando el tiempo de inicio. La sonoridad se calcula como una función lineal del logaritmo del valor de la envolvente de amplitud. Utilizar únicamente la envolvente de amplitud funciona adecuadamente para sonidos de inicio muy marcado, pero no es apropiado para señales de voz.

En el algoritmo implementado se utiliza el método basado en la derivada de la envolvente en bandas descripto en el capítulo de segmentación de audio en notas. Este método, tal como sugiere Klapuri[28], atribuye a la sonoridad del evento el valor de la derivada de primer orden de la envolvente.

Hipótesis de tempo

Una vez obtenidos los eventos se determinan todos los intervalos entre ellos (*IOI*, Interonset Interval) y se los agrupa según su similitud. Normalmente IOI se refiere a el intervalo entre dos eventos sucesivos, pero en este caso se consideran todos los pares de eventos posibles, lo que reduce la influencia de eventos que no se correspondan con el tiempo de pulso. Cada grupo se caracteriza por la media de los IOI que contiene, lo que constituye el intervalo del grupo. El proceso de agrupamiento consiste en comparar cada IOI con el intervalo de los grupos existentes. Si hay un grupo cuyo intervalo es similar al IOI éste se agrega al grupo y en caso contrario se crea un grupo nuevo. La similitud entre IOI e intervalo de grupo se establece como una diferencia menor a 50 ms. Esto es necesario debido a la imperfección de regularidad del tempo de una melodía cantada y la precisión limitada del algoritmo de segmentación. La construcción incremental de los grupos hace que su intervalo cambie a medida que se adicionan IOI. Una vez que todos los IOI fueron procesados, los grupos que difieren menos de 50 ms se unen en un único grupo. La figura 6.9 es un ejemplo de agrupamiento de intervalos.

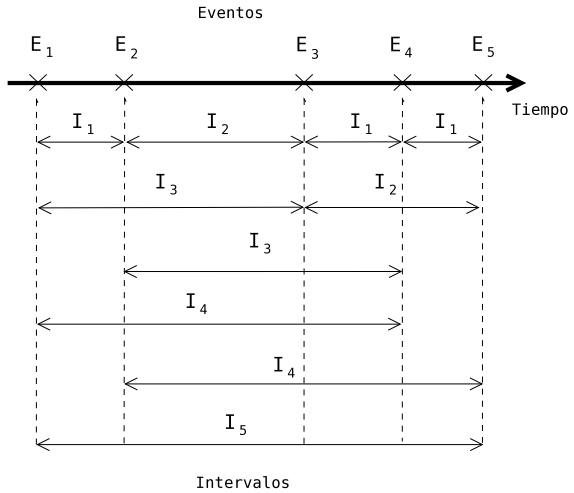


Figura 6.9: Agrupamiento de intervalos entre eventos.

Para establecer las hipótesis de tempo, a cada grupo se le asigna un puntaje según la cantidad de elementos en el grupo y en los grupos relacionados. Dos grupos están relacionados si sus intervalos son uno múltiplo del otro. Esto refleja el hecho de que existen intervalos que representan múltiplos o divisores del tempo. La relación de multiplicidad tiene una tolerancia de 50 ms. El puntaje se determina mediante el número de elementos del grupo multiplicado por el factor dado por la función

$$f(d) = \begin{cases} 6-d, & 1 \leq d \leq 4 \\ 1, & 5 \leq d \leq 8 \\ 0, & \text{en otro caso} \end{cases} \tag{6.6}$$

donde d es la relación de multiplicidad entre grupos. La elección de $f(d)$ otorga más relevancia a las relaciones cercanas entre grupos. Por ejemplo, dado un tempo, es más probable que existan IOI que representen la mitad que un décimo del tempo.

Las hipótesis de tempo consisten en el intervalo de los grupos de mayor puntaje. En el algoritmo implementado se consideran los 4 grupos de puntaje más alto.

Hipótesis de fase

Hasta el momento se construyeron hipótesis de tempo, es decir del período de los pulsos, pero no de su ubicación o fase. Asumiendo que al menos uno de los eventos del tramo inicial de la señal coincide con un pulso se pueden considerar los primeros eventos como hipótesis de fase. En una melodía cantada este supuesto es razonable, por lo que el algoritmo toma como hipótesis de fase los eventos presentes en el tramo inicial. La duración del tramo inicial se estableció en 2 segundos.

Seguimiento del pulso

Con las hipótesis obtenidas se realiza un proceso de seguimiento de los eventos para determinar cual de ellas se ajusta mejor a los datos. El sistema es capaz de seguir pequeñas variaciones en el tempo pero no puede resolver cambios significativos.

Inicialización. Se crea un agente por cada par de hipótesis de tempo y fase. A partir de sus hipótesis cada agente predice la ubicación del siguiente pulso. Durante la etapa de seguimiento las predicciones se comparan con la ubicación de los eventos a fin de testear las hipótesis.

Un agente se caracteriza por su historia y estado. El estado corresponde a las hipótesis actuales de fase y tempo. La historia consiste en el conjunto de eventos coincidentes con las predicciones del agente hasta el momento. Las hipótesis iniciales y la historia permiten luego construir la secuencia de pulsos de beat.

Seguimiento de eventos. Los eventos se procesan en orden y son considerados por los agentes como posibles tiempos de pulso. Las predicciones, que se construyen a partir de las hipótesis actuales sumando a la fase múltiplos enteros del intervalo de tempo, se comparan con la ubicación de los eventos. Se establecen dos niveles de ventanas de tolerancia alrededor de la predicción que representan la variación en la ubicación de los pulsos tolerable por el agente para aceptar el evento, como se aprecia en la figura 6.10. La ventana interna, establecida en 40 ms a ambos lados de la predicción, representa la tolerancia del agente a la ubicación de los eventos en el caso de un pulso estricto. Por otro lado, la ventana externa constituye la tolerancia del sistema a cambios de tempo y fase y se estableció en 10 y 20 % de la hipótesis de tempo del agente antes y después de la predicción respectivamente. La asimetría se debe al hecho de que es más común la reducción expresiva del tempo que su incremento.

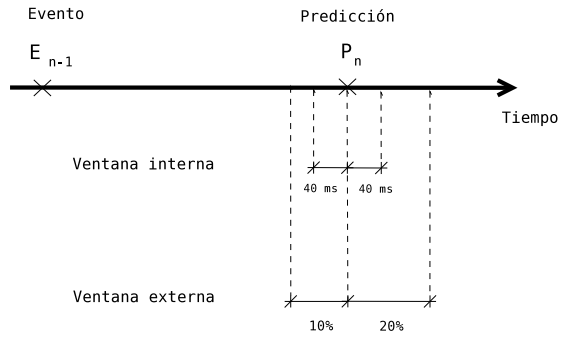


Figura 6.10: Ventanas de tolerancia alrededor de la predicción. El seguimiento toma diferentes caminos según la ubicación del siguiente evento respecto a las ventanas.

Cuando un agente procesa un evento existen tres posibilidades de ubicación del evento respecto a las ventanas. En el caso en que se encuentre fuera de la ventana externa simplemente es ignorado por el agente. Otra posibilidad es que se ubique dentro de la venta interna. El evento es aceptado como tiempo de pulso y se actualizan las hipótesis del agente. La nueva hipótesis de fase es el tiempo de ocurrencia del evento y la hipótesis de tempo se corrige en función del error de predicción de la siguiente forma,

$$IT(n) = IT(n-1) + \frac{E}{c} \tag{6.7}$$

donde $IT(n)$ es el intervalo de tempo del agente al aceptar n eventos, E es el error de predicción y c es un factor de ajuste. Se incrementa el puntaje del agente de acuerdo a la sonoridad del evento y al error de predicción. Si este evento no corresponde a la primera predicción desde el último evento aceptado, se interpolan los pulsos faltantes dividiendo el intervalo de tiempo en tramos iguales. El último caso es que el evento quede fuera de la ventana interna pero dentro de la externa. El evento es aceptado del mismo modo que en el caso anterior y además se crea un nuevo agente idéntico que no acepta el evento como tiempo de pulso. Esto permite tomar en cuenta ambas posibilidades y de acuerdo al puntaje final elegir la más adecuada.

Manejo de agentes. Dado que el comportamiento futuro de los agentes sólo depende de su estado actual, cuando dos agentes tienen las mismas hipótesis uno de ellos puede eliminarse para reducir el costo computacional.

La condición de igualdad de estados tiene cierta tolerancia, que se establece en 40 ms para el tempo y la fase. Debido a que el puntaje se determina sólo a partir de la relación entre predicciones y eventos, se elimina el agente de menor puntaje hasta ese momento.

Evaluación de agentes. El sistema construido de esta forma compara los agentes basado en tres factores: la equiespacialidad de los pulsos elegidos, el número de veces que las predicciones coinciden con eventos y la sonoridad de los eventos seleccionados.

Solo cuando un evento coincide con la predicción se incrementa el puntaje del agente. La equiespacialidad de los pulsos y la sonoridad de

los eventos se tienen en cuenta en el cálculo del puntaje,

$$P_j(n) = P_j(n-1) + (1 - \frac{e}{2})s(n); \qquad (6.8)$$

donde $P_j(n)$ es el puntaje del agente al aceptar n eventos, $s(n)$ es la sonoridad del evento y e es el error relativo en la predicción. Este último se calcula como la diferencia entre la predicción y el evento, dividido entre el ancho de la ventana externa posterior. De esta forma la sonoridad del evento queda multiplicada por un número entre 0.5 y 1 de acuerdo al error de predicción.

El algoritmo retorna la secuencia de pulsos de beat obtenida a partir del agente con mayor puntaje (ver figura 6.11).

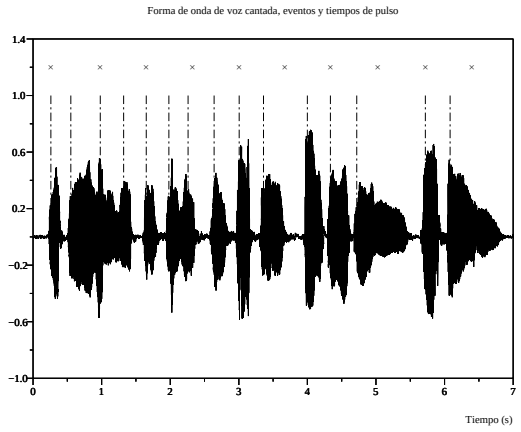


Figura 6.11: Estimación de los tiempos de pulso (cruces) a partir de los eventos (líneas punteadas) para una señal de voz cantada.

Observaciones

El proceso de agrupamiento de intervalos es exitoso en ubicar el tempo correcto dentro de los grupos de puntaje más alto. Por otro lado, en el tramo inicial de la señal de audio es altamente probable que alguno de los eventos coincida con uno de los tiempos de pulso. Esto se debe a que la duración del tramo es considerable respecto al rango de tempo típico de una melodía cantada. El peor caso es para melodías de tempo lento. Si el tempo fuera de 60 bpm, dado que la duración del tramo inicial son 2 segundos, existe la posibilidad de que hayan 2 eventos coincidentes con el pulso. Lo anterior implica que luego de la inicialización exista un agente con las hipótesis correctas de tempo y fase.

Debido a la construcción del algoritmo, luego del seguimiento, los agentes de mayor puntaje tendrán asociado un tempo que se relaciona con el tempo correcto, siendo usualmente un múltiplo. Esto último se debe a que la coincidencia de eventos es más probable bajo las hipótesis de un tempo más rápido. Por ejemplo, si en la melodía es frecuente la ocurrencia de eventos en la mitad del intervalo de beat, probablemente el agente del doble de tempo tenga mayor puntaje que el del tempo real. Esto es muy común en estructuras métricas como $\frac{2}{4}$ y $\frac{4}{4}$. Sin embargo, tales errores no son relevantes en la transcripción ya que implican la elección de otra unidad métrica básica.

En cuanto a la fase, los agentes cuyas hipótesis iniciales no tengan alguna relación directa con la fase correcta, generalmente no confirmarán sus predicciones de tiempos de pulso. Los agentes de mayor puntaje entonces, presentarán frecuentemente la fase correcta o un desfasaje de 180º (contrafase). En el caso del ejemplo anterior, el agente que comience a contrafase y con el tempo correcto, puede alcanzar uno de los puntajes mas altos.

Como característica importante del algoritmo cabe mencionar la influencia de los parámetros en su comportamiento. La elección del tamaño de las ventanas de tolerancia de predicción es un compromiso entre permitir variaciones mayores de tempo y aceptar eventos que no corresponden a tiempos de pulso. Al aceptar un nuevo evento la hipótesis de tempo se ajusta mediante un término de corrección proporcional al error de predicción. La constante de proporcionalidad c de la ecuación 6.7 regula la influencia del último intervalo de pulso en la hipótesis actual de tempo.

Para la estimación del tempo el algoritmo trabaja únicamente con un vector que contiene la ubicación de los inicios de notas. Este volumen de información es muy inferior al que se maneja al trabajar con la señal de audio, por lo tanto el costo computacional es bajo (despreciable frente a la segmentación de notas).

6.6. Evaluación

Se realizó un evaluación preliminar para comparar ambos algoritmos implementados, determinándose que el desempeño del algoritmo de estimación a partir del inicio de eventos es ampliamente superior para el caso de melodías cantadas. En esta sección se pretende reflejar el comportamiento de este algoritmo pero no una evaluación formal del mismo. Para una evaluación de ese tipo es necesario contar con la participación de un experto y se deben definir más precisamente los criterios de evaluación.

6.6.1. Método de evaluación

El algoritmo fue evaluado utilizando música polifónica y melodías de voz cantada. Este último caso es el de mayor relevancia para la aplicación final de búsqueda de música por melodía. Se utilizaron fragmentos de música polifónica de distintos estilos, con y sin referencias rítmicas (ver apéndice A). Las melodías de voz cantada utilizadas se agrupan en dos bases de datos:

BD1 Extracto de la canción *Memory* del musical *Cats* grabado por 3 cantantes profesionales[23].

BD2 Un conjunto de 12 melodías grabadas por cantantes no profesionales.

El método de evaluación para música polifónica consiste en tomar los dos candidatos de mayor puntaje y obtener como salida la señal de audio original acompañada de golpes de campana indicando el pulso estimado. Si la estimación es correcta tanto en tempo como en fase para alguno de los dos casos se considera exitoso el desempeño.

Para el caso de melodías cantadas, que como ya se mencionó es un problema más complejo, el criterio de evaluación es menos rígido y se considera exitoso el desempeño si la estimación correcta está dentro de los cuatro primeros candidatos.

6.6.2. Resultados

En las tablas 6.12 y 6.13 se presentan los resultados para música polifónica y voz cantada respectivamente. El error consiste en el porcentaje de experiencias no exitosas.

Estilo	Archivos	Bien	Mal	Error(%)
Rock	3	3	0	0
Pop	3	3	0	0
Tango	3	2	1	33
Chacarera	3	2	1	33
Clásico	3	2	1	33
Jazz	3	1	2	66
Candombe	3	1	2	66
Milonga	3	0	3	100
Bossa	3	0	3	100

Figura 6.12: Resultados para música polifónica de distintos estilos.

BD	Archivos	Bien	Mal	Error(%)
BD1	3	2	1	33
BD2	12	10	2	17
Total	15	12	3	20

Figura 6.13: Resultados para voz cantada.

Para música polifónica el algoritmo funciona correctamente si el pulso es marcado (aunque no existan instrumentos de percusión). Sin embargo si no hay referencias rítmicas fuertes como en los archivos de Milonga y Bossa utilizados (sólo voz y guitarra) las estimaciones son incorrectas. La poliritmia presente en algunos estilos provoca errores de estimación.

Respecto a la voz cantada se puede observar que el desempeño es relativamente bueno considerando la dificultad del problema.

6.7. Conclusiones

Se abordó el problema de la detección automática del tempo y tiempo de pulso en señales de audio. Los trabajos consultados, que parecen reflejar el estado del arte del tema, reportan buenos resultados para música polifónica de tempo marcado. El problema de la detección de tempo de una melodía monofónica no es tratado en forma específica y presenta mayor dificultad. Esto se debe a la ausencia de referencias rítmicas como instrumentos de percusión o cambios de acorde. La información con la que se cuenta es la ubicación temporal y la

sonoridad de las notas. Se eligieron técnicas que utilizan esta información y se implementaron dos algoritmos.

El algoritmo basado en la técnica de detección de periodicidad de envolventes usando filtros peine produce buenos resultados para señales de audio polifónicas de tempo marcado. Sin embargo su aplicación a melodías cantadas es limitada. La periodicidad de las envolventes no es perfecta debido a la posible ausencia de eventos en tiempo de pulso o a imperfecciones en el tempo de la interpretación. Esto produce una salida ruidosa de los filtros peine provocando errores en la estimación.

La técnica propuesta por Dixon busca el tempo y la fase que mejor se ajustan a la ubicación de los inicios de notas. Las imperfecciones en la regularidad del tempo son absorbidas mediante tolerancia en la ubicación de los eventos. En el manejo de múltiples hipótesis que hace el algoritmo la ausencia de un evento en tiempo de pulso no descarta hipótesis y puede conducir de todas formas a una estimación correcta. El sistema logra encontrar exitosamente el tempo (o un tempo relacionado) en la mayoría de los casos, pero con ocasionales errores de fase.

Se pudo constatar que es factible el uso de algoritmos de detección automática de tempo en la transcripción de una melodía, si bien puede ser necesaria cierta interacción con el usuario para evitar los errores mencionados. El algoritmo más adecuado para esta aplicación es el basado en la detección a partir del inicio de notas.

Un sistema de búsqueda de música por melodía debería ser de uso general y por lo tanto de fácil manejo. Incluir la detección automática de tempo requiere aumentar la interacción con el usuario. Por ejemplo, se podrían ofrecer varias opciones para que el usuario seleccione la correcta. Si bien sería deseable contar con la información de tempo para lograr una mejor transcripción, se decide no incluir la detección automática de tempo en el sistema desarrollado para no dificultar su uso.

7
Transcripción

7.1. Resumen

En este capítulo se describe la última etapa de un sistema de transcripción de una melodía cantada a notas musicales. Denominamos transcripción al proceso que permite obtener una representación simbólica de la melodía a partir de las muestras de la señal de audio. La representación elegida consiste en las alturas y tiempos de inicio y fin de las notas.

Esta última etapa de la transcripción parte de la información de inicio de notas y frecuencia fundamental dada por los algoritmos descriptos anteriormente.

7.2. Introducción

En un sistema de búsqueda de música por contenido el usuario graba su voz cantando la melodía buscada. La consulta consiste entonces en una señal digital de audio crudo. La base de datos está compuesta por melodías codificadas de forma adecuada para realizar la búsqueda. No es posible comparar la consulta directamente con la información almacenada por lo que se debe extraer de la señal de audio la información de alto nivel que representa la melodía.

El correcto funcionamiento de la etapa de transcripción es crucial en el desempeño del sistema de búsqueda. Los errores típicos en la transcripción, como adición, sustitución u omisión de notas, alteran la coincidencia con la melodía original, por lo que dificultan la búsqueda. A partir de la transcripción comúnmente se derivan códigos para la búsqueda (contornos o intervalos de altura y duración) que permiten eludir errores de interpretación, e independizar la búsqueda de la afinación y tempo de la consulta. Una transcripción correcta es un punto de partida necesario para obtener cualquier tipo de código de búsqueda.

La frecuencia fundamental de la voz cantada es continua y presenta variaciones considerables dentro de una misma nota (ver figura 7.1). El algoritmo de detección de f0 estima la frecuencia fundamental cada unos pocos milisegundos produciendo un contorno de altura muy detallado que representa la micro-entonación de la melodía cantada. Sin embargo en la transcripción es necesario obtener la macro-entonación de la interpretación, que consiste en la secuencia de notas de la melodía. Las pequeñas variaciones del contorno de f0 (rasgos expresivos, inestabilidad) deben eludirse en el proceso de transcripción. En la figura 7.1 se puede ver el contorno de f0 (micro-entonación) y las notas estimadas (macro-entonación). Por otro lado, en la música occidental la altura de las notas varía en un rango discreto de frecuencia dado por la escala temperada. La transcripción requiere entonces, por un lado, aproximar el contorno de frecuencia fundamental correspondiente a una nota a un único valor de frecuencia, y por otro asociar a este valor una nota musical (por ejemplo pasar de Hz a números MIDI).

Vale la pena aclarar que no se realiza una transcripción a notación musical sino a una notación apropiada para la búsqueda. La transcripción a notación musical requiere el conocimiento de información adicional como el tempo, comienzo de compás y métrica.

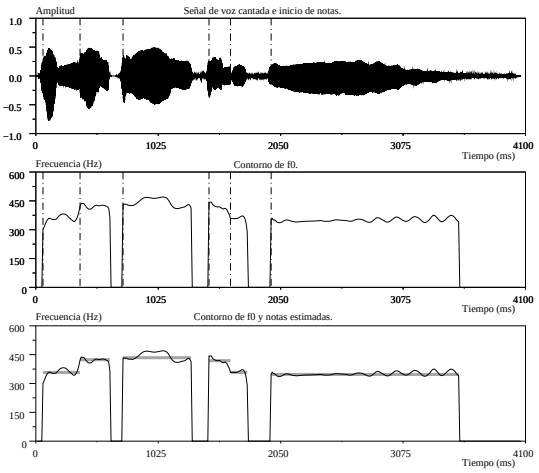


Figura 7.1: Señal de voz cantada, contorno de frecuencia fundamental y altura estimada de cada nota (macro-entonación). Se aprecian variaciones significativas del contorno dentro de una misma nota, lo que dificulta la asignación de un único valor de altura.

7.3. Objetivo

El objetivo del presente trabajo es obtener una representación simbólica de la melodía interpretada. Esto se logra combinando adecuadamente la información de inicio de notas y frecuencia fundamental. Asimismo se utiliza el contorno de f0 para mejorar la segmentación agregando notas faltantes. Por último se analizan diversas técnicas propuestas para la cuantización de alturas a la escala temperada y se selecciona la más apropiada.

7.4. Sistema de transcripción

El sistema consta de tres etapas: combinación de inicio de notas y f0, agregado de notas faltantes y cuantización de alturas, como se observa en el diagrama de la figura 7.2. La entrada al sistema es la salida de los algoritmos de segmentación de notas y de detección de f0, es decir los tiempos de inicio de notas y el contorno de f0. El algoritmo de segmentación devuelve dos conjuntos de tiempos de inicio, uno indicando los eventos más prominentes y otro que señala eventos débiles.

La primera etapa consiste en determinar en forma preliminar el comienzo y fin de cada nota. El comienzo está dado por los eventos fuertes obtenidos en la segmentación y el fin se establece cuando la nota se extingue (el contorno de f0 se anula) o al comenzar una nueva nota. El sistema de transcripción considera los eventos fuertes como confiables y asume que corresponden a inicios genuinos. En la etapa inicial pueden faltar las notas mas débiles por lo que se busca agregarlas usando otros criterios. En esto consiste el siguiente bloque del sistema. Por un lado se utilizan los eventos débiles como posibles inicios de nota, y por otro se buscan cambios de altura significativos. Finalmente se aproxima y cuantiza el contorno de f0 de cada nota a una altura en la escala temperada. A continuación se detallan cada una de las etapas del sistema.

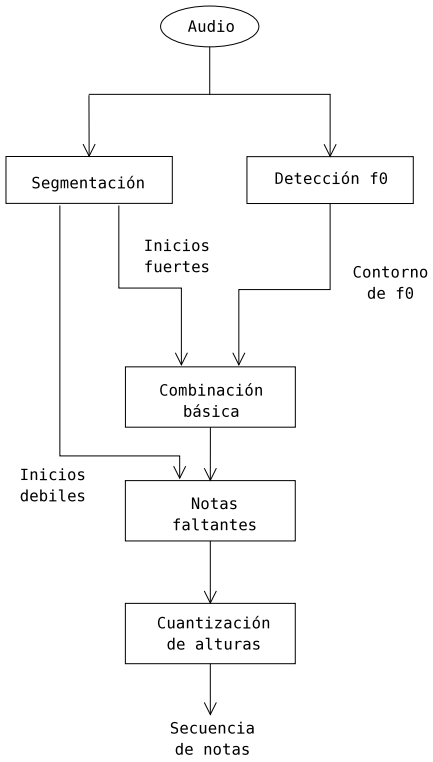


Figura 7.2: Diagrama de bloques del sistema de transcripción.

7.4.1. Combinación básica de inicio de notas y contorno de f0

Para facilitar la segmentación se discrimina entre tramos de sonido y de silencio a través de un umbral sobre la envolvente de amplitud de la señal. Esto se realiza por dos razones. La primera consiste en que al inicio de una nota la señal de audio si bien no es audible presenta cierta periodicidad por lo que tiene una frecuencia fundamental asociada. Mediante el umbral se evita establecer el inicio de una nota antes del punto donde la señal comienza a ser audible. Lo mismo ocurre al determinar el tiempo de fin cuando la señal se extingue. La segunda se debe a que tanto en el ataque como al desvanecerse una nota la frecuencia fundamental no es estable lo que podría dificultar la estimación de altura (ver figura 7.3).

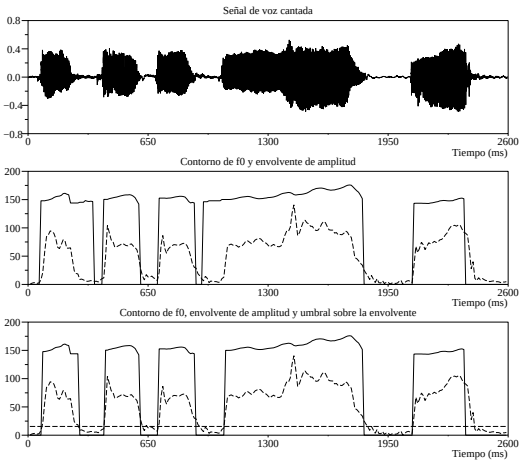


Figura 7.3: El uso de un umbral sobre la envolvente de amplitud de la señal de audio permite evitar la estimación de altura en tramos de amplitud muy reducida en donde la frecuencia no es estable y establecer el inicio y fin de las notas cuando la señal empieza o deja de ser audible respectivamente. La envolvente se indica con línea punteada.

El inicio y fin de cada nota se determina recorriendo el contorno de frecuencia fundamental considerando los inicios fuertes dados por la segmentación. Los inicios se establecen cuando el contorno de f0 pasa de cero a un valor no nulo y se mantiene durante cierto tiempo mínimo, y en los puntos indicados por los inicios fuertes de la segmentación. El tiempo de fin puede estar dado por dos situaciones diferentes:

> El comienzo de una nueva nota indicado por un inicio fuerte.
>
> El contorno de f0 se anula durante cierto tiempo mínimo.

Hasta el momento entonces, se cuenta con una segmentación básica en intervalos candidatos a notas que están limitados solo por los eventos mas prominentes. Al omitir los inicios débiles es posible que esta segmentación no distinga notas intermedias dentro de un intervalo.

7.4.2. Búsqueda de notas omitidas

Como ya se mencionó las notas faltantes se agregan usando dos criterios. Uno de ellos es considerar los eventos débiles dados por el algoritmo de segmentación y verificar si corresponden a inicios de notas genuinos. El otro consiste en buscar cambios de altura dentro de un intervalo que indiquen la existencia de más de una nota.

Eventos menos prominentes

El algoritmo de segmentación entrega un conjunto de tiempos de inicio de eventos débiles. Estos corresponden a los eventos detectados cuya intensidad supera un umbral menor que el de los eventos fuertes. La mayoría de ellos provienen de modulaciones de amplitud que tienen lugar durante el transcurso de una nota, mientras que otros se deben a transiciones débiles entre notas. Un evento débil se valida como inicio de nota si va acompañado de un cambio de altura significativo (ver figura 7.4).

Dado que el intervalo entre notas sucesivas de la escala temperada es un semitono, el mínimo cambio de altura considerado como significativo es el correspondiente a un semitono. Este intervalo equivale aproximadamente a un 6 % del valor de frecuencia. Para validar el evento débil se obtiene la mediana del contorno de frecuencia durante cierto intervalo de tiempo antes y después del mismo. Luego se comparan estos valores determinando si su diferencia supera un umbral establecido a partir del intervalo de semitono. El valor utilizado

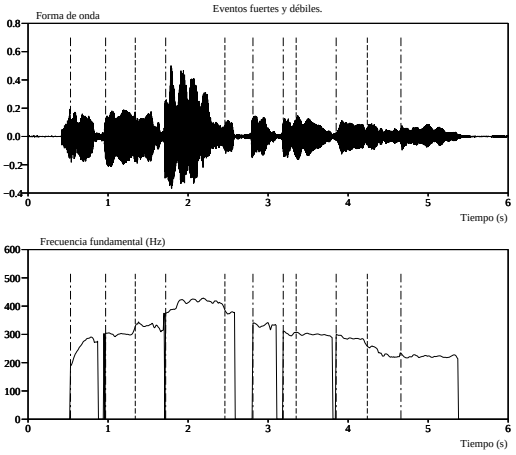


Figura 7.4: Se indican los eventos fuertes (raya y punto) y los eventos débiles (línea punteada) dados por el algoritmo de segmentación. Los eventos débiles que tienen asociado un cambio de altura se establecen como nuevos inicios de nota, como es el caso del primer, segundo y cuarto evento débil de la figura. El tercero que no está acompañado por un cambio de altura no se valida como inicio de nota.

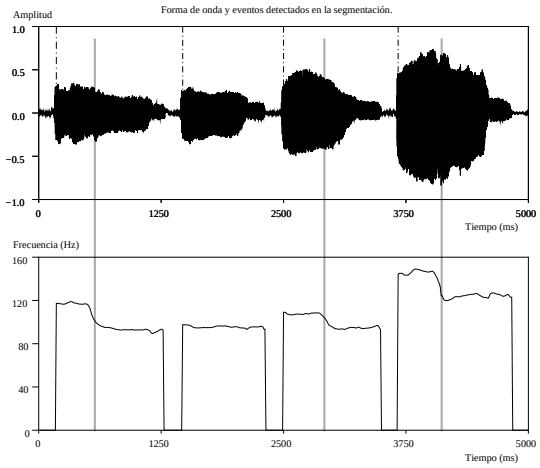


Figura 7.5: Notas ligadas. En la figura se muestra con línea punteada los eventos detectados por la segmentación y con línea gruesa la transición entre notas ligadas. Usualmente el algoritmo de segmentación no detecta las notas ligadas debido a que la transición no es fácil de deducir a partir de la envolvente de la forma de onda.

es menor al 6 % ya que el contorno de f0 presenta transiciones suaves entre notas sucesivas.

Cambios de altura

Puede ocurrir que algunas notas no estén indicadas por el algoritmo de segmentación. Un ejemplo de esto son las notas ligadas, en donde hay un cambio de altura pero sin ataque. La transición entre notas no va acompañada de un aumento de amplitud de la envolvente de la forma de onda, por lo que el algoritmo de segmentación en general no es capaz de detectarla (salvo cuando el cambio de altura es muy significativo, ver sección 5.5.2). En la figura 7.5 se observa un ejemplo de notas ligadas.

Otro ejemplo son las notas de ataque muy suave, en las que la intensidad del evento no es suficiente para ser considerado como evento débil por el algoritmo de segmentación. Esto puede ocurrir, por ejemplo, con fonemas sonoros nasales (“m”,“n”,“ñ”) como se puede apreciar en la figura 7.6.

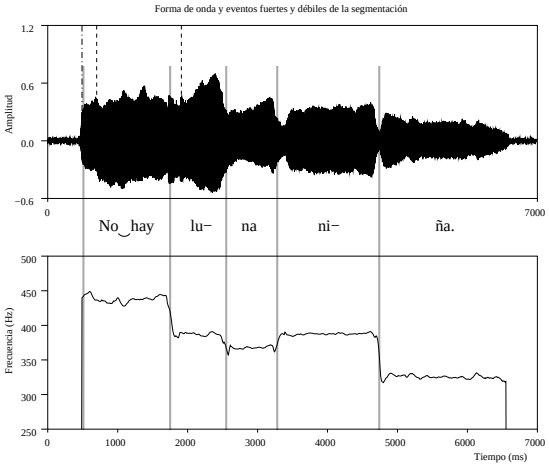


Figura 7.6: Los ataques débiles también son problemáticos para el algoritmo de segmentación. En la figura se muestran los eventos dados por la segmentación para un tramo de voz cantada en donde predominan los inicios de nota con fonemas sonoros nasales. Los comienzos de notas genuinos se indican con línea gruesa.

Las notas omitidas, en la mayoría de los casos, pueden ser agregadas observando los cambios de

altura en el contorno de f0. Esto sin embargo no es una tarea sencilla debido a que la expresividad en la interpretación y la falta de entrenamiento en cantantes inexpertos introducen un conjunto de rasgos en el contorno de f0[1] que pueden ser considerados por error como notas adicionales. Podemos clasificar estas características de la siguiente forma:

Transiciones suaves El cambio de altura en las transiciones entre notas es en general suave. En el inicio de una nota el contorno de f0 comúnmente crece durante cierto tiempo hasta alcanzar un valor luego del cual se mantiene relativamente estable. Este fenómeno es típico en un cantante buscando la afinación correcta. Los cantantes no entrenados necesitan escuchar su propia voz para estabilizar la altura de la nota que están cantando. La tensión de las cuerdas vocales necesaria para dar determinada nota es una habilidad que se adquiere con la práctica, por lo que en cantantes entrenados las transiciones son bien definidas, salvo en casos donde se busque cierta expresividad. Al finalizar una nota ocurre un fenómeno similar. Los cantantes no entrenados tienden a no preocuparse en mantener la altura de una nota hasta su fin, provocando que el contorno de f0 decaiga. También puede ocurrir que antes de finalizar una nota el cantante comience gradualmente a entonar la siguiente. Las transiciones suaves no son un rasgo exclusivo de los cantantes no entrenados sino que pueden deberse a recursos expresivos como el *glissando* [1].

Típicamente estas transiciones tienen una duración entre 70 a 120 ms. Sin embargo en notas cortas las transiciones son de menor duración.

Picos Son notas cortas que presentan un incremento gradual de altura seguido de un decremento gradual y surgen de adornos de interpretación. La altura de la nota corresponde al máximo de frecuencia.

Inestabilidad La variabilidad del contorno de altura durante el transcurso de una nota es una característica frecuente en cantantes no entrenados y en interpretaciones expresivas. Si el contorno es inestable se dificulta la asignación de una altura, pero peor aún, si las variaciones superan el intervalo de semitono pueden considerarse por error como notas adicionales de corta duración. Es común la existencia de picos espurios que no deben considerarse como nuevas notas.

Vibrato Es uno de los rasgos más característicos de la voz cantada y consiste en una modulación casi sinusoidal de la frecuencia fundamental de la voz, con frecuencia de modulación de entre 4 y 8 Hz y amplitud de uno a dos semitonos. La regularidad de las oscilaciones es un objetivo buscado por los cantantes profesionales al utilizar este recurso expresivo. En el caso de cantantes inexpertos, si bien el fenómeno tiende a seguir el modelo descripto, es menos predecible debido a inestabilidades locales. A los efectos de la transcripción el vibrato debe eludirse, evitando confundir los cambios de altura con notas adicionales y asignando a la nota la altura correspondiente al valor medio de las oscilaciones.

El vibrato puede apreciarse claramente en la última nota de la figura 7.1. Las restantes características mencionadas se muestran en la figura 7.7.

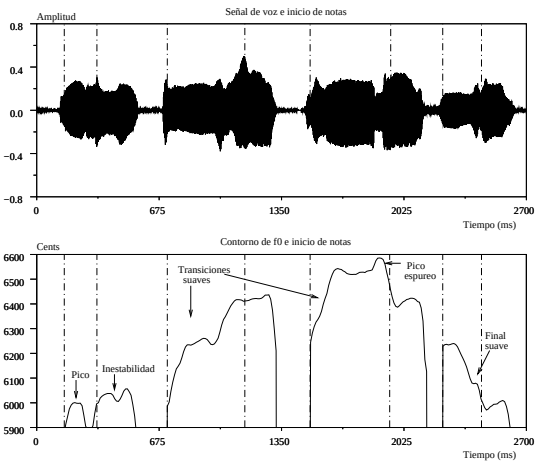


Figura 7.7: El contorno de f0 tiene normalmente un conjunto de rasgos que dificultan la tarea de agregar, a partir de los cambios de altura, notas omitidas en las etapas anteriores.

[1] Deslizamiento continuo de altura en el pasaje entre notas.

Teniendo en cuenta las características del contorno de f0 antes mencionadas se establecieron ciertas reglas para agregar las notas omitidas, a partir de las variaciones de altura.

En el algoritmo implementado se procesan uno a uno todos los intervalos candidatos a nota identificados hasta el momento. Cada intervalo se caracteriza por un valor de altura dado por la mediana del contorno de f0. Este valor refleja adecuadamente la altura de una nota como se explica en la sección 7.4.3. En primer lugar se determina si el intervalo puede dividirse en más de una nota en función de su duración y de la variabilidad del contorno de f0 (cuanto tiempo se desvía mas de un semitono de su altura). Si el intervalo no puede dividirse según estos criterios se pasa al siguiente, de lo contrario se recorre el contorno de f0 repitiendo sucesivamente los siguientes pasos:

> Se identifica un tramo del contorno de f0 que se desvía más de un semitono de la altura asignada y si supera cierta duración mínima (*durmin*) es considerado como candidato a nueva nota.
>
> Se asigna una altura al candidato y se observa su estabilidad determinando el tiempo que el contorno se mantiene dentro de una banda centrada en esta altura (ver figura 7.8). Si el contorno de altura del candidato es suficientemente estable (tiempo de permanencia dentro de la banda mayor a *testable*) se lo valida como una nueva nota.

Al agregar notas omitidas usando los cambios de altura, debido a las características mencionadas del contorno de f0, existe un compromiso entre la cantidad de omisiones corregidas y el número de notas erróneas agregadas. En este sentido el algoritmo implementado es conservador, ya que las notas omitidas en la etapa de segmentación son esporádicas (ver los resultados del algoritmo de segmentación de audio en notas, sección 5.6.2). Un criterio mas flexible puede empeorar la segmentación en notas. De esta forma, se estableció la duración mínima de una nueva nota en *durmin* = 150 ms y el criterio de estabilidad consiste en que el contorno de f0 permanezca durante un tiempo *testable* = 120 ms dentro de una banda del 1 % de la altura asignada.

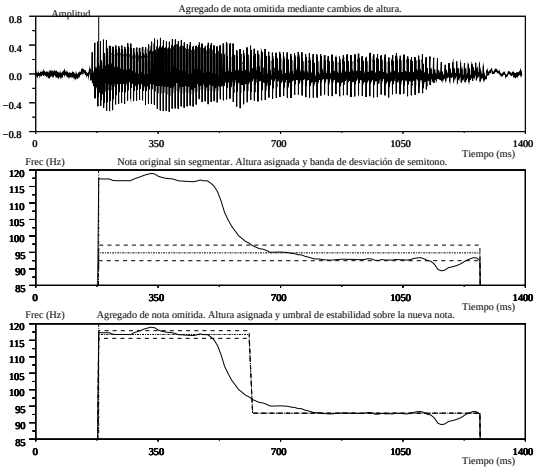


Figura 7.8: En la figura se ilustra el proceso de agregado de notas omitidas a partir de cambios de altura. El algoritmo de segmentación no es capaz de discriminar las notas de la figura. Inicialmente se asigna una única altura a todo el intervalo. Luego se identifica un tramo que se desvía más de un semitono de la altura. Se caracteriza este tramo por un valor de altura y se establece una banda de tolerancia a ambos lados para observar su estabilidad.

Es importante señalar que ninguno de los dos criterios usados para agregar notas omitidas en la etapa de segmentación es capaz de marcar la transición entre notas sucesivas de la misma altura.

7.4.3. Cuantización de alturas

Para asignar una altura a cada nota es necesario en primer lugar aproximar el contorno de f0 de la nota a un único valor de frecuencia y luego asociar este valor a una altura de un sistema de afinación (por ejemplo, escala temperada con A_4 = 440 Hz).

Aproximación del contorno de f0

Como ya resulta claro a luz de lo visto anteriormente el contorno de f0 de una nota puede ser muy variable y resulta difícil establecer un criterio para asignarle un único valor de frecuencia. El valor de frecuencia que mejor representa la altura de la nota es aquel que toma el contorno cuando alcanza la estabilidad. Esto implica ignorar las

transiciones suaves, inestabilidades, picos espurios. Una forma de hacer esto es analizar el contorno de f0 de cada nota, identificar distintas situaciones (picos, transiciones suaves, vibrato) y asignar una altura usando criterios diferentes en cada caso[1]. En este trabajo se optó por una forma sencilla pero efectiva: el valor de altura de la nota es la mediana del contorno. Típicamente la región de transición y los picos espurios son de corta duración respecto a la zona de estabilidad. Por lo tanto, la mediana, que toma un valor en torno a los mas frecuentes del contorno, no se ve afectada por estas características y se aproxima al valor de estabilidad. Esto no sucede si se caracteriza la altura de la nota con la media del contorno, como se muestra en la figura 7.9. Si el contorno de la nota es muy inestable resulta difícil asignar una altura incluso en forma manual y la mediana produce en general una buena estimación. La altura de una nota con vibrato es la media de las oscilaciones por lo que también en este caso la mediana es una aproximación correcta. El caso mas problemático de los definidos en la sección 7.4.2 son las notas clasificadas como *picos*. Su altura corresponde al valor máximo de frecuencia que alcanza el contorno, sin embargo la mediana es siempre inferior. De todas formas esto no constituye una desventaja crítica de la forma de aproximación de altura ya que los *picos* no son frecuentes y en muchos casos la altura asignada no difiere significativamente de la correcta.

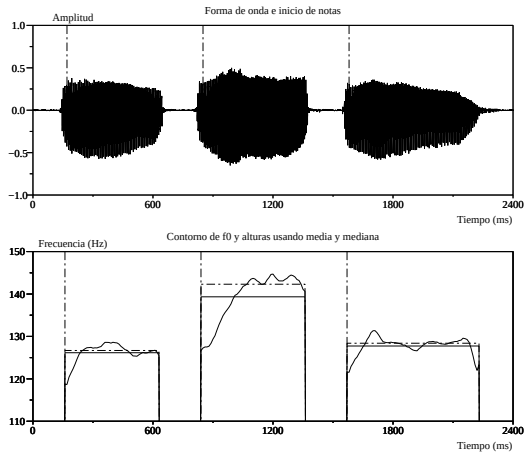


Figura 7.9: Aproximación del contorno de f0 a un valor de frecuencia usando los valores de media (linea continua) y mediana (linea discontinua). En las transiciones suaves, como en la segunda nota, la mediana resulta más apropiada.

Ajustando la afinación a la escala temperada

El proceso de transcripción hasta este punto transformó la información digital dada por las muestras de audio en una serie de eventos definidos por una altura en Hz y un tiempo de inicio y fin en milisegundos que representan las notas cantadas.

La música de diversas culturas utiliza distintos sistemas de afinación. Estos consisten en una organización de alturas basadas en el intervalo de octava[2], que toma diferentes formas según la cultura (por ejemplo, 12 intervalos en la música occidental y 22 intervalos en la música tradicional de la India). Una hipótesis necesaria para la transcripción de una melodía es el sistema de afinación a usar. Este trabajo se concentra en música occidental, por lo que la transcripción se hará a la escala temperada.

La afinación occidental ha evolucionado a través del tiempo desde la afinación justa o escala cromática a la escala temperada, que divide el intervalo de octava en doce intervalos llamados semitonos. Las frecuencias de notas consecutivas en la escala se relacionan de la siguiente forma:

$$f_{nota(i+1)} = f_{nota(i)} \cdot \sqrt[12]{2}. \tag{7.1}$$

Por convención, se estableció como referencia para la escala temperada el *LA* de la octava media del piano en $A_4 = 440$ Hz.

El semitono es la unidad básica de altura en la música occidental, si bien el ser humano es capaz de distinguir intervalos mas pequeños. Una unidad de medida de intervalo muy utilizada es el *cent*, que equivale a un centavo de semitono. Por lo tanto, un intervalo de octava corresponde a 1200 cents.

El estándar MIDI (Musical Instrument Digital Interface), desarrollado para el manejo y comunicación de instrumentos digitales, constituye una representación conveniente de la escala temperada. Cada nota se representa en el código MIDI con

[2]Distancia entre alturas cuando la frecuencia de una es el doble de la otra.

un número entero que se relaciona con su frecuencia a través de la siguiente ecuación:

$$n_{MIDI} = 69 + \text{round}\left[12\frac{log(\frac{f}{440})}{log(2)}\right]. \qquad (7.2)$$

La nota A_4 se representa con el número MIDI 69, y la siguiente en la escala, A_4# con el número 70. El A_4 se ubica 6900 cents por encima de la nota MIDI 0.

En la aplicación de búsqueda de audio por melodía, es necesario comparar la consulta con el contenido de la base de datos, que consiste en música afinada a la escala temperada. Los cantantes sin embargo no tienen la capacidad de ajustarse a un sistema de afinación sin escuchar una referencia, salvo raras excepciones (oído absoluto). La interpretación entonces no respeta exactamente los intervalos ni la referencia de la escala temperada. Se hace necesario corregir el desajuste natural entre la melodía cantada y el sistema de afinación. Se han propuesto diversas maneras de resolver este problema, las cuales se detallan a continuación.

A. Nota MIDI más cercana Consiste en aproximar la frecuencia estimada a la nota MIDI más cercana utilizando la ecuación 7.2.

B. Método de McNab[3] Este método intenta determinar y corregir el desajuste entre la interpretación del usuario y la escala temperada. Se aproximan las alturas cantadas a una escala temperada con referencia variable. Para cada nota se calcula la distancia a esta escala, se ajusta su referencia según la desviación y se aproxima a la nota más cercana. Por ejemplo, si se cantan tres notas de alturas 5990, 5770 y 5540 cents por encima de la nota MIDI 0, la primera se indica como C_4 (6000 cents) y se corre la referencia 10 cents hacia abajo. Luego la segunda nota corresponde a B_3b (5790 cents en lugar de 5800 cents). La referencia se establece 20 cents por debajo de la anterior, totalizando 30 cents de corrimiento. Por último la tercer nota es A_3b. Si bien en la escala temperada la nota más cercana a 5540 cents es G_3 (5500 cents), al cambiar la referencia, A_3b corresponde a 5570 cents. Este método se basa en que al cantar una nota se usa la anterior como referencia, por lo que se acumulan los errores que se comenten respecto a la escala temperada (se tiende a comprimir los intervalos grandes y a expandir los intervalos pequeños).

C. Método de Pollastri[45] A diferencia del método anterior la hipótesis utilizada por Pollastri es que cada cantante tiene un tono de referencia en mente y canta notas de una escala temperada referida a este tono. Esto implica que los errores no se propagan y tienden a ser constantes (si bien hay una pequeña dependencia del error con el intervalo). El método consiste en determinar la desviación más frecuente para estimar el tono de referencia y así ajustar las notas cantadas a la escala temperada absoluta. Para ello se divide el semitono en 10 intervalos iguales de 0.2 semitonos solapados 0.1 semitono, se calcula la desviación de cada nota como la parte fraccionaria de la ecuación 7.2 (sin redondear) y se asocia al intervalo correspondiente. La desviación más frecuente corresponde a la media de las desviaciones del intervalo con más notas. A cada nota se le resta esta desviación y se redondea a la nota MIDI más cercana. Un ejemplo de este método puede verse en la figura 7.10.

D. Método de Klapuri[46] Las hipótesis de este método son las mismas que las del anterior, salvo que no se considera que la desviación depende del intervalo. Esto se manifiesta en el hecho de que Klapuri calcula el tono de referencia a partir de todas las desviaciones y no sólo de las más frecuentes. El promedio de las desviaciones δ se calcula como:

$$\delta = \frac{1}{T}\sum_{t=1}^{T}\left[\text{mod}(n'_{MIDI}(t) + 0.5, 1) - 0.5\right] \qquad (7.3)$$

donde n'_{MIDI} es la nota MIDI de la ecuación 7.2 sin el redondeo y T es el número de notas. De esta forma $\delta \in$ [-0.5, 0.5). El ajuste se realiza de la siguiente forma:

$$n_{MIDI}(t) = n'_{MIDI}(t) + 0.5 - \text{mod}(\delta + 0.5, 1). \quad (7.4)$$

Comparación de los métodos Se implementó cada uno de los métodos y se realizó una comparación usando siete melodías con un total de 150 notas. Las primeras cinco de ellas fueron interpretadas por una cantante entrenada y las restantes por un cantante inexperto.

Int	Rango	Notas en el intervalo	Desv.
1	0.0-0.199	54.023; 59.054	0.039
2	0.1-0.299		0
3	0.2-0.399	53.321	0.321
4	0.3-0.499	53.321	0.321
5	0.4-0.599		0
6	0.5-0.699	55.685; 53.623; 53.652; 58.691	0.653
7	0.6-0.799	65.776; 55.685; 53.623; 57.795; 53.652; 58.691	0.697
8	0.7-0.899	53.776; 57.795; 53.816; 55.836	0.805
9	0.8-0.999	53.816; 55.836; 60.919	0.857
10	0.9-0.099	54.023; 59.045; 60.919	0.947

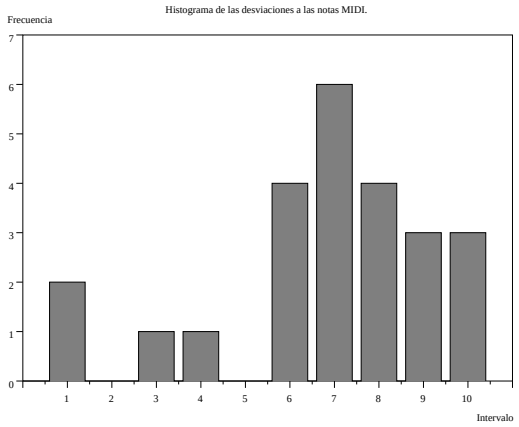


Figura 7.10: Ejemplo de ajuste utilizando el método de Pollastri para la melodía *Feliz Cumpleaños*. La tabla muestra los intervalos y la asignación de notas según el desajuste. El histograma presenta la distribución de notas en función del intervalo. Se observa un sesgo bien definido a cierto valor de desviación que representa la relación entre la afinación del cantante y la escala temperada con $A_4 = 440$ Hz.

Los resultados obtenidos (ver figura 7.11) son cualitativamente similares a los que reporta Pollastri[45](redondeo 25 %, McNab 56 % y Pollastri 13 %). La diferencia de los porcentajes de error entre ambas experiencias se debe a que el desempeño de los algoritmos depende de la habilidad de los cantantes. La experiencia de Pollastri fue realizada con 5 cantantes no experimentados.

melodía / notas	A	B	C	D
1 / 25	2	7	1	2
2 / 20	0	1	0	0
3 / 16	1	2	0	0
4 / 28	1	2	1	1
5 / 22	1	6	0	1
6 / 14	6	4	3	5
7 / 25	11	8	2	8
Total / 150	22	30	7	17
Error (%)	**15**	**20**	**5**	**11**

Figura 7.11: Comparación de los distintos métodos de ajuste de afinación. En la tabla se muestra para cada método el número de notas que difieren de la melodía original y un porcentaje de error global (A: redondeo, B: McNab, C: Pollastri, D: Klapuri).

Sobre los resultados se pueden puntualizar algunas observaciones. El método mas básico es el redondeo a la nota MIDI mas cercana. A diferencia de los otros métodos, no maneja hipótesis sobre la capacidad de afinación de los cantantes. Las notas pueden ajustarse en diferentes sentidos, distorsionando los intervalos cantados. Esto se refleja en el pobre desempeño obtenido. Llama la atención el alto porcentaje de error del método de McNab, mayor incluso que al redondear a la nota MIDI más cercana. Esto indica que las hipótesis de este método no son correctas, es decir los errores no se acumulan. El mejor desempeño es el del método de Pollastri, seguido por el método de Klapuri. Esto sugiere que es acertada la idea de que al cantar cada persona tiene un tono de referencia en mente. Pollastri lo deduce a partir de la desviación más frecuente a una escala absoluta, mientras que Klapuri promedia todas las desviaciones. Promediar las desviaciones tiene implícita la hipótesis de una desviación constante. Pollastri es más flexible en este sentido no considerando desviaciones poco frecuentes. La diferencia en el desempeño de ambos métodos fortalece la hipótesis hecha por Pollastri de que además de la desviación respecto a la escala absoluta puede existir una desviación adicional pequeña que depende del intervalo y se relaciona con la dificultad de cantarlo. En una melodía popular los intervalos difíciles para un cantante inexperto no son los más frecuentes, por lo que es factible que exista una desviación claramen-

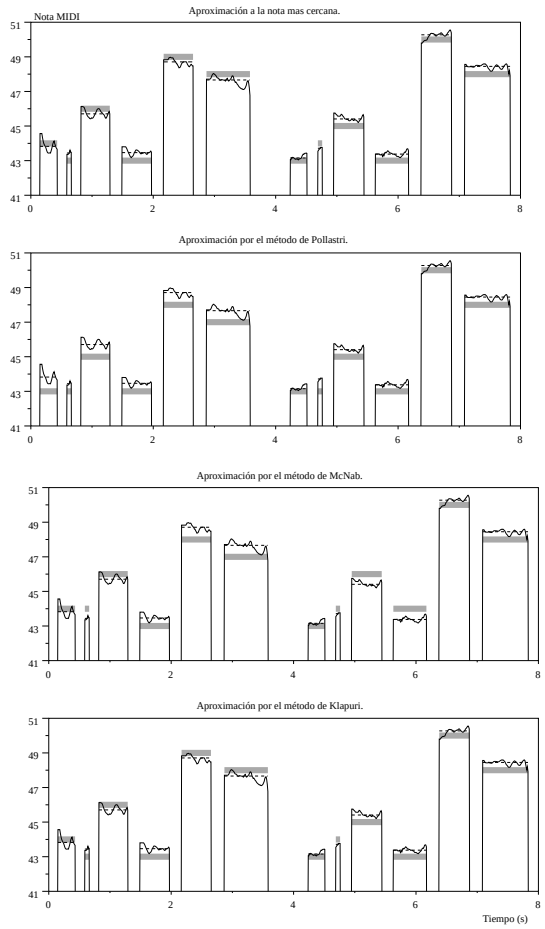


Figura 7.12: Ejemplo de ajuste de afinación usando los distintos métodos. En las gráficas se observa el contorno de f0, las notas MIDI sin redondeo que surgen de la mediana del contorno (línea punteada) y las notas MIDI dadas por cada método de ajuste.

te predominante (como en el histograma de la figura 7.10).

Por último se presenta un ejemplo del comportamiento de los distintos métodos para una interpretación de las primeras 12 notas de la melodía *Feliz Cumpleaños*. Se cantó usando la sílaba *ta* con el fin de evitar otro tipo de errores. En la figura 7.12 se compara el contorno de f0 con la altura ajustada según cada método. Las diferencias entre las distintas transcripciones pueden verse más claramente en los pentagramas de la figura 7.13.

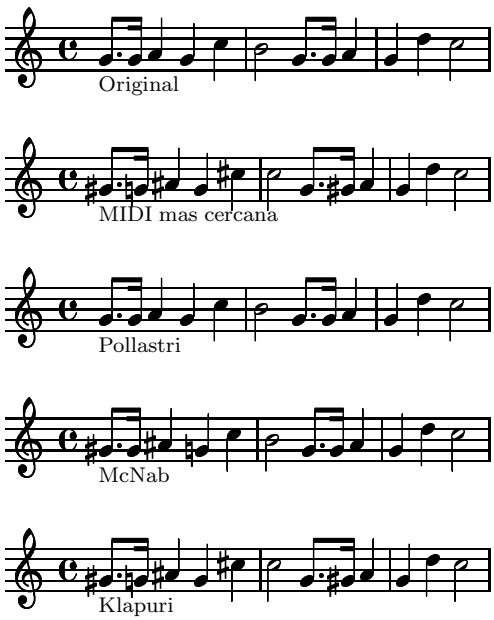


Figura 7.13: Transcripción usando los distintos métodos de ajuste para las primeras notas de la melodía *Feliz Cumpleaños*.

7.5. Evaluación

Se llevó a cabo una evaluación del sistema de transcripción desarrollado para dar una idea aproximada de su funcionamiento. Tomando como referencia la partitura se compara el desempeño del algoritmo desarrollado frente a otros dos programas de transcripción automática que reflejan el estado del arte en el tema[47]. Los programas utilizados son SoloExplorer[48] y MAMI[49].

7.5.1. Base de datos

Para la evaluación se construyó una base de datos que consiste en 22 melodías cantadas (con letra) interpretadas por dos cantantes entrenados (un hombre y una mujer). Las interpretaciones se basaron en las partituras de cada fragmento y fueron grabadas a 8 kHz, 16 bits, mono. La base de datos contiene un total de 390 notas musicales.

7.5.2. Método de evaluación

El procedimiento de evaluación consiste en comparar la salida MIDI de cada sistema con la partitura de las melodías. En la comparación se contabilizó la cantidad de aciertos (cuando las notas son iguales), la cantidad de notas unidas, la cantidad de notas insertadas y la cantidad de notas que difieren en altura. Cabe aclarar que cuando se produce una unión de notas, no se contabiliza ningún acierto ni diferencia de altura. Por esta razón, el total de notas de una melodía corresponde a la suma de los aciertos, la cantidad de notas unidas y las cantidad de notas que difieren en altura. Es importante señalar que no se considera la duración de las notas en esta evaluación.

7.5.3. Resultados

En la figura 7.14 se muestran los resultados obtenidos para los distintos sistemas evaluados.

Algoritmo	A	NU	I	NA
Solo Explorer	279	58	12	53
Total (%)	**71.54**	**14.87**	**3.08**	**13.59**
MAMI	252	30	15	108
Total (%)	**64.62**	**7.69**	**3.85**	**27.69**
Tararira	307	50	10	33
Total (%)	**78.72**	**12.82**	**2.56**	**8.46**

Figura 7.14: Resultados de la evaluación de los distintos sistemas de transcripción automática. Se indican los aciertos (A), cantidad de notas unidas (NU), cantidad de notas insertadas (I) y cantidad de notas con diferente altura (NA), sobre un total de 390 notas.

Estos resultados indican que el desempeño del sistema desarrollado es comparable al de los restantes sistemas evaluados, lo que confirma que las etapas críticas de la transcripción automática, segmentación y ajuste a la escala temperada, funcionan correctamente.

8

Técnicas de búsqueda de música por melodía

8.1. Resumen

La forma más natural de comparar melodías es a través de las notas que las componen. Sin embargo, en un sistema de búsqueda de música por melodía, la consulta inevitablemente está afectada de errores que hacen que comparar directamente las secuencias de notas no sea efectivo. En este capítulo se señalan los problemas involucrados al comparar melodías, se plantean soluciones y se describe el sistema de búsqueda implementado.

8.2. Introducción

Los principales desafíos que presenta la búsqueda de música a partir de la melodía son la codificación de la información y los criterios de similitud. La línea melódica es información de más alto nivel que la secuencia de notas específica que la compone. Una melodía puede identificarse a pesar de que se interprete en diferentes alturas, a distinto tempo (dentro de límites razonables) y con adornos o rasgos expresivos. Por lo tanto un sistema de búsqueda por melodías debe tomar en cuenta estas particularidades. La independencia respecto a la altura y al tempo puede lograrse en la codificación de las notas (codificación invariante a la transposición y al tempo). Mediante criterios de similitud flexibles durante la búsqueda es posible establecer tolerancia a las alteraciones provenientes de adornos. Esto también toma en cuenta los errores en la interpretación y en la transcripción automática.

Al trabajar con secuencias de notas, la búsqueda de melodías es básicamente un problema de búsqueda de coincidencia de cadena de caracteres (*String Matching*). Este tema tiene amplio desarrollo y existen técnicas bien conocidas que permiten detectar coincidencia exacta, es decir, identificar una secuencia que contiene cierta subsecuencia. Sin embargo en la búsqueda de música se necesitan criterios flexibles por lo que dichas técnicas no son apropiadas. Para tratar este problema se recurre a la búsqueda aproximada de coincidencia de cadena de caracteres. Una forma de medir similitud entre secuencias es a través de la *distancia de edición*, que consiste en la cantidad mínima de alteraciones necesarias para transformar una secuencia en la otra.

Las notas musicales se describen a través de dos características: altura y duración. Muchos de los sistemas de búsqueda desarrollados utilizan únicamente la información de altura[4][50][51]. Resulta evidente que la información de tiempo (duración) es útil para diferenciar melodías. En el presente trabajo se sugiere una forma de combinar ambas características para mejorar la efectividad de la búsqueda.

Existe un camino alternativo a la búsqueda basada en secuencias de notas (*Event Based Search*). Este consiste en utilizar el contorno de frecuencia fundamental y comparar series temporales (*Frame Based Search*). Para ello normalmente se usa la técnica de Deformación Temporal Dinámica (*DTW*, Dynamic Time Warping) que proviene del área de reconocimiento de voz. En el algoritmo implementado se hace uso de esta técnica como refinamiento de la búsqueda basada en eventos.

8.3. Objetivos de un sistema de búsqueda

8.3.1. Requisitos esenciales

Uno de los posibles usos de un sistema de este tipo es la búsqueda de música a través de Internet, por lo que es normal que el usuario no posea entrenamiento musical. Por otra parte, la etapa de transcripción no es perfecta. Para que el sistema sea efectivo entonces, debe ser lo suficientemente robusto frente a los errores típicos de la consulta y la transcripción.

Requisitos respecto a la consulta

Respecto a las imprecisiones de la consulta el sistema debe ser flexible a:

Altura absoluta. Diferentes personas cantan la misma línea melódica a distintas alturas. Necesariamente el sistema debe ser independiente de la altura específica a la que se interpreta la melodía.

Altura relativa. Es frecuente que cantantes no experimentados compriman o expandan los intervalos de altura (ver Capítulo 7).

Duración de notas. Los cantantes inexpertos no respetan estrictamente la duración de las notas. Esto es especialmente notorio en notas de larga duración.

Tempo. Una melodía es la misma aún si se interpreta a diferente tempo. El tempo de la interpretación es en general consistente, si bien puede variar desde la mitad al doble del tempo original[52][53].

Variaciones locales de tempo. El intervalo entre notas está afectado por imprecisiones, pero en menor medida que la duración. Esto puede asociarse a variaciones locales de tempo.

Requisitos respecto a la transcripción

El sistema de búsqueda debe ser robusto frente a los errores típicos de la transcripción automática:

Cuantización de altura. La frecuencia fundamental de una nota cantada puede variar considerablemente en el tiempo. El sistema de transcripción asigna un único valor de frecuencia para cada nota y luego cuantiza este valor a un sistema de afinación. Este proceso puede introducir errores en la altura de las notas.

Segmentación de notas. No existe una técnica de segmentación perfecta[52][53]. Los ligados y ataques suaves son problemáticos para cualquier algoritmo de segmentación. El sistema de trascripción puede entonces omitir notas presentes e introducir notas adicionales.

8.3.2. Eficacia

Es conveniente definir algún criterio para cuantificar la eficacia de un sistema de búsqueda. Para la evaluación de sistemas de búsqueda de documentos se utilizan generalmente dos conceptos que se describen a continuación[42].

Precisión. Es la proporción de documentos relevantes recuperados. Se calcula como la razón entre la cantidad de documentos relevantes recuperados para una consulta en particular sobre el total de documentos recuperados.

Recuperación. Es la proporción de documentos relevantes recuperados sobre la totalidad de documentos relevantes. La *Recuperación* (*Recall*) es difícil de medir ya que es necesario conocer los documentos relevantes que no fueron recuperados en la consulta.

Un sistema de búsqueda ideal recupera todos los documentos relevantes y únicamente éstos, lo que implica *Precisión* $= 1$ y *Recuperación* $= 1$. Muchas veces no es posible alcanzar este desempeño, como por ejemplo, en los sistemas de búsqueda aproximada. En estos casos se busca maximizar la cantidad de documentos relevantes recuperados (aumentar la *Recuperación*) y minimizar el número de documentos recuperados (aumentar la *Precisión*). Lo primero se relaciona con la codificación usada, lo que determina la capacidad de discriminación del sistema. Por otra parte, la cantidad de documentos recuperados puede controlarse en la búsqueda mediante un umbral de similitud. Existe en la práctica un compromiso entre *Precisión* y

Recuperación. Una codificación exacta permite buena discriminación lo que implica alta *Precisión*, pero ocasiona baja *Recuperación* en el caso de que la consulta contenga errores. Es posible aumentar la *Recuperación* retornando más documentos si se utiliza un umbral de similitud bajo. Sin embargo, esto degrada la *Precisión*.

En un sistema de búsqueda de música el objetivo es retornar únicamente el tema que el usuario desea y no aquellos que se le asemejan en algún sentido. Existe un único tema relevante en la colección, por lo que si es retornado se obtiene *Recuperación* $= 1$. Idealmente el tema relevante obtiene el mayor nivel de similitud en la búsqueda, y entonces, si se retorna un único tema se tendrá también *Precisión* $= 1$. Para que esto ocurra es necesario utilizar codificación estricta que permita buena discriminación, pero como contrapartida el sistema es poco robusto frente a errores en la consulta, con lo que se corre el riesgo de obtener *Recuperación* $= 0$ en muchos casos. Para tolerar errores se hace necesario el uso de codificación flexible a costa de disminuir la probabilidad de obtener *Precisión* $= 1$.

8.3.3. Eficiencia

En los sistemas de búsqueda aproximada no es posible utilizar los métodos de indexado tradicionales (ordenamiento o hashes) para agilizar la búsqueda. Es difícil establecer un criterio de indexado ya que existe incertidumbre en la consulta, inclusive en la primera nota. Por esta razón, el indexado puede afectar la efectividad del sistema. De todas formas se han propuesto algunas técnicas que reducen considerablemente el tiempo de búsqueda[54].

Al no utilizar indexado la búsqueda aproximada requiere recorrer la base de datos en forma exhaustiva, por lo que la eficiencia de los algoritmos de búsqueda aproximada es muy inferior a la de los algoritmos de búsqueda exacta. El tiempo de búsqueda en el primer caso crece linealmente con el tamaño de la base de datos, mientras que por ejemplo en el caso de búsqueda binaria el crecimiento es logarítmico[3].

Respecto a la representación de la melodía, el uso de series temporales requiere tiempos de búsqueda intolerables para la aplicación dado el gran volumen de datos que se procesa. Por ejemplo, una consulta de 10 segundos, tiene típicamente unas 15 notas, mientras que una tasa de 10 estimaciones de frecuencia fundamental por segundo produce una secuencia de largo 100.

8.4. Codificación

8.4.1. Información relevante

¿Cuáles son las características del sonido que nos permiten distinguir melodías? Una melodía mantiene su identidad bajo ciertas transformaciones en altura, tempo, timbre, sonoridad, localización espacial, ambiente reverberante y en algunas ocasiones bajo cambios rítmicos[55]. Por ejemplo, somos capaces de identificar una melodía interpretada en distintos instrumentos, a diferente altura, o distinto volumen. Para distinguir una melodía los elementos mas determinantes son los intervalos de altura y de duración de las notas que la componen. Al seleccionar una codificación de melodías adecuada para la búsqueda se deben tener presentes estas consideraciones.

8.4.2. Invarianza a la transposición y al tempo

Definamos *realización* de una melodía a una secuencia particular de notas que la representa. Si la secuencia se desplaza cierta cantidad de semitonos (transposición) o si se modifica su tempo, se obtiene otra realización de la misma melodía.

No es posible utilizar búsqueda de coincidencia de caracteres para comparar directamente las notas de dos realizaciones de una melodía y determinar si corresponden a la misma melodía. Es necesario derivar una codificación de la secuencia de notas que sea invariante tanto a la transposición como al tempo.

Una secuencia de notas se representa por una secuencia de alturas, una secuencia de duraciones y una secuencia de tiempos de inicio (como por ejemplo en el estándar MIDI). Esta es la información con la que se cuenta para construir la codificación adecuada para la búsqueda.

Invarianza a la transposición

La invarianza a la transposición se obtiene considerando los intervalos de la secuencia de alturas. Las alturas se pueden representar como números MIDI, con lo que la secuencia de alturas puede ser una secuencia de enteros. Dada la secuencia de alturas $A = (a_1, a_2, \ldots, a_n)$, la representación en intervalos es una secuencia:

$$\overline{A} = (a_2 - a_1, a_3 - a_2, \ldots, a_n - a_{n-1}).$$

Consideremos una secuencia A' transposición de A. Entonces, los elementos de las secuencias se relacionan por $a'_i = a_i + c$. Resulta evidente que ambas secuencias tienen la misma representación en intervalos, es decir,

$$\overline{A} = \overline{A'}.$$

Esta codificación es invariante a la transposición por lo que puede utilizarse para comparar las secuencias de alturas.

Invarianza al tempo

La secuencia de duraciones consiste típicamente en números reales que representan intervalos de tiempo, expresados por ejemplo en segundos. Esta representación, además de no ser invariante al tempo, no es apropiada para la búsqueda de coincidencia de caracteres ya que sus elementos no pertenecen a un alfabeto discreto. Es necesario codificar las duraciones de forma invariante al tempo y cuantizarlas a un alfabeto discreto. Una forma de lograr invarianza al tempo es normalizar las duraciones respecto a una duración de referencia d_{ref} invariante al tempo. Un valor apropiado para d_{ref} es el período de pulso, como en la notación musical. Lamentablemente, en el caso de una melodía cantada, no siempre es posible estimar el pulso (ver Capítulo 6), por lo que este criterio no es aplicable. Sin conocer el tempo, no existe una representación de las duraciones completamente aceptable. Una alternativa sencilla utilizada frecuentemente es considerar d_{ref} como la duración de la nota previa[56].

Dada la secuencia de duraciones $D = (d_1, d_2, \ldots, d_n)$, la representación en duraciones relativas es una secuencia:

$$\overline{D} = (\frac{d_2}{d_1}, \frac{d_3}{d_2}, \ldots, \frac{d_n}{d_{n-1}}).$$

Si consideramos la melodía que origina la secuencia D y alteramos su tempo, obtenemos otra secuencia de duraciones D'. Los elementos de dichas secuencias se relacionan por $d'_i = c\ d_i$. Es sencillo observar que ambas melodías tienen la misma representación en duraciones relativas, es decir,

$$\overline{D} = \overline{D'}.$$

La codificación propuesta es invariante al tempo siendo apropiada para comparar las secuencias de duraciones.

8.4.3. Cuantización de intervalos

Como ya se mencionó, un sistema de búsqueda por melodía debe ser robusto frente a errores en la consulta. La cuantización de los intervalos de altura y duración permite ajustar la tolerancia a estos errores. Asimismo, cuantizar los intervalos a un alfabeto discreto es necesario para utilizar técnicas de búsqueda de coincidencia de cadena de caracteres.

Intervalos de altura

En el caso ideal de una consulta sin errores resultaría efectivo utilizar los intervalos de altura sin cuantizar. La cadena de caracteres usada para la búsqueda sería directamente la secuencia de intervalos. Sin embargo, cuando la consulta contiene errores, pequeñas diferencias en los intervalos (por ejemplo, un semitono) generan secuencias distintas. De esta forma, secuencias musicalmente cercanas pueden ser muy diferentes para el algoritmo de búsqueda de coincidencia de caracteres. Es preferible entonces utilizar una cuantización más gruesa para incrementar la tolerancia a errores.

Una serie de experimentos sobre la memoria musical[57] indican que la dirección de los intervalos es un elemento importante en el reconocimiento de melodías. Esto motiva la cuantización en tres niveles, denominada comúnmente *DUR* (down, up, repeat). Muchos sistemas de búsqueda de música por melodía utilizan este tipo de contorno[4][58]. La cuantización DUR tiene la ventaja de encubrir los intervalos inexactos interpretados por el usuario. En contrapartida, el alfabeto es poco discriminativo, fundamentalmente para bases de datos extensas. Esto se debe a que, al ser un

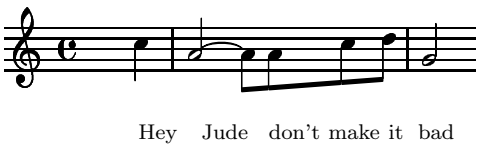
alfabeto muy reducido, un contorno dado coincide con una gran fracción de la base de datos. Asimismo, se desaprovecha la habilidad musical del usuario, ya que incluso cantantes no experimentados son capaces de resolver adecuadamente muchos intervalos de altura[59].

La cuantización plantea entonces un compromiso entre la tolerancia a errores y la capacidad de discriminación del sistema. Las codificaciones DUR y de intervalos exactos descriptas anteriormente, corresponden a los extremos de este compromiso. Dado que la tolerancia a errores y la capacidad de discriminación son requisitos esenciales de un sistema de búsqueda de música, es necesario una clasificación de intervalos intermedia respecto a dichos extremos.

En [42] se utiliza un vector Q para representar los niveles de cuantización, cuyos elementos indican el valor absoluto de las fronteras de cuantización en semitonos. Se estudian cuantizaciones en 5 y 7 niveles. Por ejemplo, $Q = [0\ 1\ 3]$ indica 5 niveles de cuantización, que corresponden a repetición, intervalo de semitono o tono ascendente, intervalo de semitono o tono descendente, intervalo mayor a un tono ascendente e intervalo mayor a un tono descendente. En [51] se propone una clasificación en clases de intervalos: intervalos pequeños (de 1 a 3 semitonos), medianos (de 3 a 7 semitonos) y grandes (al menos 6 semitonos), denominada *QPI* (Quantized Partially overlapping Intervals). Las clases se representan con los símbolos α, β y γ, respectivamente (ver figura 8.1). En la figura 8.2 se muestra un ejemplo de algunas de las formas de cuantización de intervalos de altura mencionadas.

Semitonos	Símbolos QPI
$8, \ldots$	γ
$6, 7$	$\mathcal{B}$
$4, 5$	β
3	$\mathcal{A}$
$1, 2$	α
0	o
$-2, -1$	$-\alpha$
-3	$-\mathcal{A}$
$-5, -4$	$-\beta$
$-7, -6$	$-\mathcal{B}$
$\ldots, -8$	$-\gamma$

Figura 8.1: Cuantización QPI. Los símbolos α, β y γ, corresponden a los intervalos pequeños, medianos y grandes respectivamente. Dado que las clases se solapan, se utilizan los símbolos $\mathcal{A}$ y $\mathcal{B}$ para intervalos que pertenecen a dos clases adyacentes.



MIDI	72	69	69	72	74	67
DUR	*	D	R	U	U	D
$Q = [0\ 1\ 3]$	*	-2	0	2	1	-2
QPI	*	$-\mathcal{A}$	o	$\mathcal{A}$	α	$-\mathcal{B}$

Figura 8.2: Ejemplo de distintas cuantizaciones de intervalos de altura.

Intervalos de duración

Como se mencionó en la sección 8.3.1, los cantantes no entrenados no respetan la duración de las notas. El error mas frecuente es finalizarlas prematuramente. Por otra parte, el tempo de una interpretación suele ser consistente. Debido a que los tiempos de inicio de nota están asociados al pulso, los intervalos entre inicios de nota son también consistentes. Por esta razón, en algunos sistemas de búsqueda[56], se considera como duración el intervalo entre inicios de notas sucesivas (*IOI*, Inter Onset Interval).

Para lograr invarianza al tempo se introdujo el concepto de duración relativa, que consiste en el cociente entre la duración de la nota actual y la duración de la nota previa. Al cantar despreocupadamente una melodía, en algunas ocasiones se cometen aproximaciones en la duración de las notas. Cuando la duración de notas sucesivas es similar, la relación entre duraciones es más bien coherente; sin embargo, cuando la duración de notas sucesivas es muy dispar, se suele perder coherencia. Por ejemplo, al interpretar una corchea seguida de una negra, la relación probablemente sea cercana a 2.

En cambio, la relación de duraciones de una blanca luego de una fusa, puede ser muy diferente a su valor justo de 16.

Al cuantizar los intervalos de duración es posible atenuar las aproximaciones del cantante antes mencionadas, suavizando la relación de duraciones por medio de una función logarítmica[1][56]. Para ello se sugiere en [1] utilizar el alfabeto de enteros dado por la siguiente ecuación,

$$r(i) = \text{round}\left(10 \log_{10}\left(\frac{d_i}{d_{i-1}}\right)\right). \qquad (8.1)$$

$r(i)$	0	1	2	3	4	5	6
$\frac{d_i}{d_{i-1}}$	$\frac{1}{1}$			$\frac{2}{1}$		$\frac{3}{1}$	$\frac{4}{1}$

Figura 8.3: Ejemplo de valores asignados por la cuantización propuesta en la ecuación 8.1.

8.5. Búsqueda aproximada de cadena de caracteres

La tarea de un sistema de búsqueda aproximada consiste en encontrar buenas ocurrencias de cierto patrón en una base de datos. Para ello es necesario definir una medida de distancia. En búsqueda aproximada de cadena de caracteres normalmente se utiliza la distancia denominada *distancia de edición*. La distancia de edición consiste en el número mínimo de alteraciones necesarias para transformar una cadena de caracteres en otra y se determina a través del algoritmo conocido como *Programación Dinámica* (*DP*, Dynamic Programming).

8.5.1. Distancia de edición

Para calcular la distancia de edición entre dos secuencias $A = (a_1, a_2, \ldots, a_m)$ y $B = (b_1, b_2, \ldots, b_n)$ típicamente se consideran las siguientes transformaciones locales,

- sustitución: $a_i \rightarrow b_j$
- inserción: $\lambda \rightarrow b_j$
- omisión: $a_i \rightarrow \lambda$,

siendo λ la secuencia vacía.

La distancia de edición entre las secuencias A y B, $D(A, B)$, es el número mínimo de transformaciones locales necesarias para transformar A en B.

Es posible generalizar la distancia de edición definiendo un conjunto de transformaciones locales T y una función de costo w que asigna un peso $w(t)$ a cada transformación $t \in T$. Dadas dos secuencias A y B existe más de una serie de transformaciones locales que convierten A en B[1]. El costo de una serie es la suma de los costos de cada una de sus transformaciones. La distancia entre las dos secuencias es el costo de la serie de transformaciones de menor costo.

En la distancia de edición, también llamada *distancia de Levenshtein*, los costos asignados son $w(a \rightarrow a) = 0$ (coincidencia), $w(a \rightarrow b) = 1$ (sustitución) y $w(a \rightarrow \lambda) = w(\lambda \rightarrow a) = 1$ (omisión e inserción). Una variante de la distancia de edición es la *distancia de Hamming*, en donde la única trasformación permitida es la sustitución y la asignación de costos es $w(a \rightarrow a) = 0$ (coincidencia) y $w(a \rightarrow b) = 1$ (sustitución).

8.5.2. Cálculo de la distancia de edición

La distancia entre dos secuencias $A = (a_1, a_2, \ldots, a_m)$ y $B = (b_1, b_2, \ldots, b_n)$ puede calcularse usando el algoritmo de Programación Dinámica. Este consiste en construir una matriz $D(m+1, n+1)$ en forma recursiva, en donde cada elemento d_{ij} es la distancia entre las secuencias prefijo $a_1 \ldots a_i$ y $b_1 \ldots b_j$. La ecuación de recurrencia para el cálculo de d_{ij} es,

$$d_{ij} = \min(d_{r-1,s-1} + w(a_r \ldots a_i \rightarrow b_s \ldots b_j)). \qquad (8.2)$$

Para calcular la recurrencia hay que establecer condiciones iniciales asignando valores a la primer fila y la primer columna de la matriz. El problema de búsqueda aproximada de cadenas de caracteres consiste en obtener la distancia entre las dos secuencias completas. Para ello, se inicializa la matriz con $D(i, 0) = i$ y $D(0, j) = j$ para $0 \leq i \leq m$ y $0 \leq j \leq n$. La distancia entre las secuencias es el elemento d_{mn} de la matriz.

[1]Por ejemplo, una sustitución puede verse como una omisión y una inserción

Para calcular distancia de Levenshtein la ecuación 8.2 puede expresarse como,

$$d_{ij} = \min \begin{cases} d_{i-1,j} + 1 \\ d_{i,j-1} + 1 \\ d_{i-1,j-1} + 0 & a_i = b_j \\ d_{i-1,j-1} + 1 & a_i \neq b_j \end{cases}$$

con $i \geq 1$ y $j \geq 1$.

La ecuación de recursión para evaluar la distancia de Hamming es,

$$d_{ij} = \min \begin{cases} d_{i-1,j-1} + 0 & a_i = b_j \\ d_{i-1,j-1} + 1 & a_i \neq b_j \end{cases}$$

con $i \geq 1$ y $j \geq 1$.

La cantidad de operaciones requeridas para calcular la distancia de edición usando Programación Dinámica es $O(mn)$.

8.5.3. Búsqueda aproximada de patrones

En el caso de búsqueda de melodías, se buscan ocurrencias aproximadas de la consulta dentro de la melodía. Esto corresponde al problema de *búsqueda aproximada de patrones*, el cual presenta algunas diferencias respecto al problema de búsqueda aproximada de cadena de caracteres. Para permitir que una de las secuencias esté contenida dentro de la otra, la matriz D debe inicializarse con $D(i,0) = i$ y $D(0,j) = 0$ para $0 \leq i \leq m$ y $0 \leq j \leq n$. La distancia entre las secuencias es el mínimo de la última fila de la matriz. De esta forma, el patrón puede comenzar en cualquier punto de la melodía, ya que no se penalizan las omisiones en la consulta antes del inicio o luego del fin de la ocurrencia del patrón.

La mejor ocurrencia del patrón puede identificarse a través del mínimo de la última fila de la matriz D, d_{mt}. El índice t indica la posición del fin de la ocurrencia del patrón en la melodía. Para identificar el comienzo se debe seguir el camino que va desde d_{mt} hasta la primera fila produciendo el menor costo (camino mínimo). En la figura 8.4 se muestra un ejemplo del cálculo de la distancia de Levenshtein entre dos secuencias (patrón y melodía), indicándose el camino mínimo y el alineamiento entre las secuencias.

	*	−2	0	3	2	−4	0	6
*	0	0	**0**	0	0	0	0	0
3	1	1	1	**0**	1	1	1	1
2	2	2	2	1	**0**	1	2	2
0	3	3	2	3	**1**	1	1	2
−4	4	4	4	3	2	**1**	2	2
0	5	5	4	4	3	2	**1**	2

melodía	*	−2	0	3	2	λ	−4	0	6
patrón			*	3	2	0	−4	0	

Figura 8.4: Ejemplo del cálculo de la distancia de edición usando Programación Dinámica. Se indica en la matriz el camino mínimo, es decir el camino de menor costo. En la tabla inferior se muestra el alineamiento entre las secuencias.

8.5.4. Consideraciones sobre búsqueda de melodías

La búsqueda de melodías presenta ciertas características que la diferencian del problema de búsqueda de coincidencia de cadena de caracteres. A continuación se señalan algunas de estas diferencias.

Operaciones de edición

Para comparar secuencias musicales, en el cálculo de la distancia de edición puede ser conveniente permitir algunas transformaciones locales adicionales. En [6] se sugiere considerar dos nuevas trasformaciones locales denominadas *fragmentación* y *consolidación*. La consolidación consiste en la sustitución de una secuencia de notas por una única nota cuya duración es la suma de las duraciones de las notas de la secuencia. La fragmentación es la sustitución de una nota por una serie de notas, donde la duración de la serie es la misma que la de la nota original. Sustituir una nota por dos de igual altura y mitad de duración no debería penalizarse muy costosamente ya que son musicalmente similares. En la figura 8.5 se ilustra

el conjunto de transformaciones locales mas comunes utilizadas en la comparación de secuencias musicales.

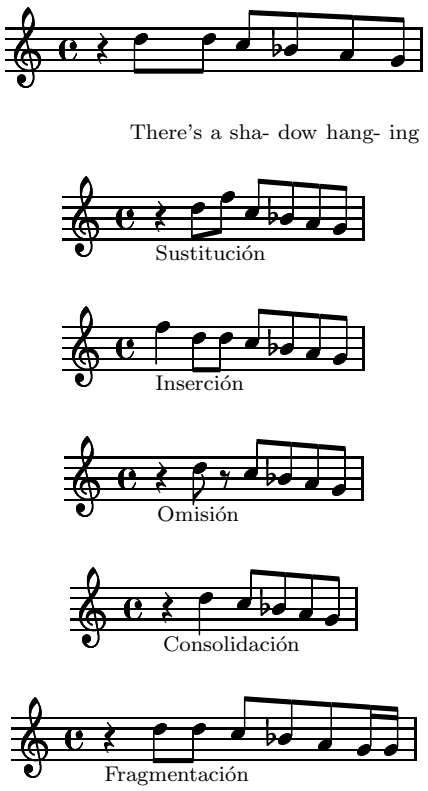


Figura 8.5: Transformaciones locales usadas en el cálculo de la distancia de edición para la búsqueda de melodías.

Funciones de costo

Al establecer el costo de las transformaciones locales es posible introducir consideraciones musicales. Por ejemplo, el costo de una sustitución puede establecerse en función del grado de disonancia entre las notas involucradas. En [6] se asigna un costo creciente con la disonancia del intervalo: quinta, tercera, sexta, cuarta, séptima y segunda.

Combinación de información de altura y duración

La comparación de secuencias de notas difiere de la comparación de cadenas de caracteres, ya que cada nota involucra dos dimensiones, altura y duración. Se dificulta entonces definir una distancia de edición entre secuencias de notas. Una alternativa es definir el costo de cada transformación local como una combinación lineal de las diferencias de altura y duración. Por ejemplo, en [6] se penaliza una sustitución ($a_i \rightarrow b_j$) con,

$$w(a_i \rightarrow b_j) = w_{altura}(a_i, b_j) + k\, w_{dur}(a_i, b_j)$$

donde w_{altura} depende de la diferencia de alturas entre a_i y b_j, w_{dur} depende de la diferencia de duraciones y k es un factor que representa la contribución relativa de las diferencias de altura y duración.

Distancia de edición al usar intervalos

Al usar una representación en intervalos una única transformación en la secuencia de notas afecta dos intervalos. Por ejemplo, la secuencia de notas MIDI $A = (60, 64, 67, 71)$ tiene la representación en intervalos $\overline{A} = (4, 3, 4)$. Una alteración de un semitono en la altura de la tercera nota, $A' = (60, 64, 68, 71)$, produce la representación en intervalos $\overline{A'} = (4, 4, 3)$. La distancia de Levenshtein entre las secuencias de alturas es $D(A, A') = 1$, mientras que entre las secuencias de intervalos es $D(\overline{A}, \overline{A'}) = 2$. La representación en intervalos entonces, presenta la desventaja de que las transformaciones locales definidas para comparar secuencias de notas se hacen más costosas.

8.6. Algoritmo implementado

El sistema de búsqueda implementado identifica la mejor ocurrencia de la secuencia de notas obtenida de la transcripción de la consulta, en una base de datos de melodías MIDI monofónicas. Para ello, se codifica la información de altura y duración de forma invariante a la transposición y al tempo respectivamente y se realiza una búsqueda aproximada de cadena de caracteres.

En el diseño del sistema se hizo hincapié en maximizar la precisión, es decir en retornar un úni-

co candidato en la búsqueda. A los efectos de aumentar la capacidad de discriminación del sistema se implementó una etapa de refinamiento de la búsqueda basada en series temporales utilizando el contorno de frecuencia fundamental de la consulta.

8.6.1. Codificación

La codificación de las notas obtenidas en la etapa de transcripción es esencial en el desempeño del sistema de búsqueda. Se analizaron y evaluaron distintas formas de codificación para determinar la mas apropiada en relación al objetivo de maximizar la precisión.

Altura

Como se mencionó en 8.4.3 la codificación de alturas debe seleccionarse de forma tal que el sistema de búsqueda provea tolerancia a errores y capacidad de discriminación. En base a esto se consideraron dos formas de codificación invariantes a la transposición. Una de ellas es la codificación QPI descripta en 8.4.3, que clasifica los intervalos de altura en pequeños, medianos y grandes. La otra consiste en utilizar los intervalos de altura exactos pero con tolerancia de un semitono, es decir, dos intervalos de altura que difieren en un semitono se consideran equivalentes. La motivación de esta codificación es que en la transcripción automática son frecuentes errores de un semitono que provienen fundamentalmente de ajustar las alturas a la escala temperada.

Ambas alternativas se compararon junto a DUR e intervalos exactos, que constituyen los extremos del compromiso entre discriminación y tolerancia a errores. Para ello se implementó un sistema de búsqueda que determina la distancia de edición solo a través de la altura utilizando cada una de estas codificaciones. La evaluación se realizó sobre una base de datos de 208 temas de *The Beatles* y las consultas se dividieron en dos clases según el número de notas, cortas (15 notas o menos) y largas (mas de 15 notas). En la figura 8.6 se muestra la clasificación de las consultas utilizadas en la evaluación.

Se controló la precisión de la búsqueda de forma de no retornar más de 20 candidatos. Por ejemplo, si se devuelven 30 candidatos con distancia de edición mínima, entre los cuales está el correcto, se considera una búsqueda fallida.

	cortas	largas	Total
canto	27	47	74
tarareo	15	8	23
Total	42	55	97

Figura 8.6: Clasificación de las consultas usadas en la evaluación.

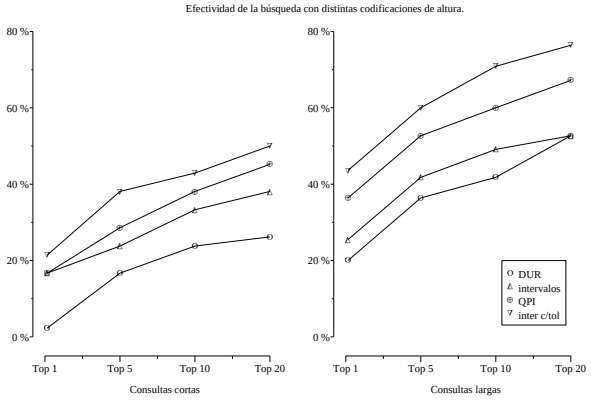


Figura 8.7: Comparación del comportamiento del sistema de búsqueda con las diversas codificaciones de altura propuestas. Se puede apreciar que la capacidad de discriminación se incrementa notoriamente al aumentar el largo de la consulta.

En la figura 8.7 se presenta la comparación del desempeño de las distintas codificaciones. Como era de esperar, el desempeño usando codificación DUR e intervalos exactos es inferior al obtenido con las otras codificaciones. También se aprecia que al incrementar el largo de la consulta mejora el comportamiento. Las codificaciones QPI e intervalos con tolerancia se diferencian en que la primera presenta mayor tolerancia a errores pero menor capacidad de discriminación dado que la cuantización de los intervalos es mas gruesa. Los resultados obtenidos indican que para la evaluación realizada, la codificación en intervalos con tolerancia

aprovecha mejor la capacidad del usuario para resolver los intervalos de altura manteniendo adecuada tolerancia a errores.

Para alcanzar el objetivo de maximizar la precisión, la evaluación señala que la codificación en intervalos con tolerancia es la opción mas adecuada.

Duración

Un error común de los cantantes inexpertos es finalizar prematuramente las notas. Para eludir los errores en la duración se sugiere considerar el intervalo entre notas IOI, que suele ser mas consistente que las duración. Con el objetivo de confirmar esta hipótesis, se modificó el sistema de búsqueda de forma de calcular la distancia de edición solo a partir de la información de duración de las notas. En la figura 8.8 se comparan los resultados al considerar las duraciones y los IOI. El algoritmo implementado utiliza como información de duración los IOI.

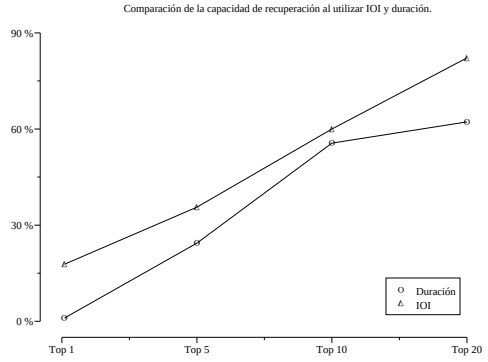


Figura 8.8: Capacidad de recuperación al utilizar los dos criterios de duración: intervalo entre el comienzo y el fin de una nota e intervalo entre los comienzos de notas sucesivas (IOI). Los resultados mostrados son comparativos, es decir, no indican el desempeño del sistema basándose solo en la información de duración.

La información de duración se codificó utilizando la ecuación 8.1 ya que de esta forma se atenúan las aproximaciones que realiza el cantante en notas sucesivas de duración dispar (ver sección 8.4.3).

8.6.2. Cálculo de la distancia de edición

El algoritmo de búsqueda calcula la distancia de edición combinando la información de duración y altura. Al realizar la combinación se prioriza la información de altura en base a las siguientes consideraciones,

- En la música occidental, la altura permite mayor discriminación que la duración, ya que existen mas combinaciones naturales de una secuencia de n alturas que de una secuencia de n duraciones[51].
- En una melodía cantada comúnmente se cometen grandes aproximaciones en la duración de las notas.
- Los errores de segmentación de notas en la transcripción automática tienen mayor repercusión en los intervalos de duración que en los intervalos de altura. Por ejemplo, si no se segmentan dos notas consecutivas de la misma altura (este es el error mas común en los sistemas de transcripción) se afecta un solo intervalo de altura y tres intervalos de duración.

Sean a_i^a y b_j^a los intervalos exactos de altura y a_i^d y b_j^d los intervalos de duración codificados según la ecuación 8.1 de las secuencias a comparar. La distancia de edición definida para el algoritmo implementado está dada por la siguiente ecuación de recurrencia,

$$d_{ij} = \min \begin{cases} d_{i-1,j} + i & \\ d_{i,j-1} + o & \\ d_{i-1,j-1} + s & \\ d_{i-1,j-1} + c & |a_i^a - b_j^a| < 2 \text{ y} \\ & |a_i^d - b_j^d| < 2 \\ d_{i-1,j-1} + s_d & |a_i^a - b_j^a| < 2 \end{cases}$$

donde i es el costo de una inserción, o el de una omisión, s el de una sustitución, c el de una coincidencia de duración y altura y s_d el de una sustitución de duración. El valor asignado al costo de cada transformación se especifica en la tabla de la figura 8.9. Resulta útil definir una medida de *similitud* a partir de la distancia de edición. Para ello se

	Costo
i	1
o	1
s	1
c	-1
s_d	0

Figura 8.9: Valor de costo asignado a cada transformación local en el algoritmo implementado.

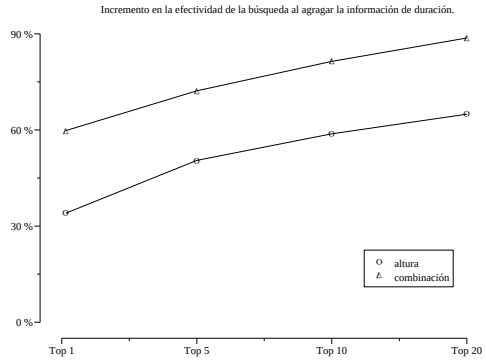


Figura 8.10: Efectividad del sistema de búsqueda utilizando solo la información de altura y combinando la altura y la duración. Los resultados confirman que la información temporal es fundamental en el reconocimiento de melodías.

normaliza la distancia de edición para que tome valores entre 0 y 100,

$$\text{similitud} = 100\,\frac{(m-1)-D}{2(m-1)}$$

con D la distancia de edición y m el número de notas de la consulta.

Cabe señalar que la tolerancia de un semitono en los intervalos de altura se implementa en el cálculo de la distancia de edición. Se introduce también tolerancia en los intervalos de duración.

Las transformaciones locales *consolidación* y *fragmentación* no se implementan. Si bien estas transformaciones permiten disimular durante la búsqueda los errores de segmentación del algoritmo de transcripción[52], pueden aumentar las falsas coincidencias si estos errores no son frecuentes.

En la figura 8.10 se compara el comportamiento del sistema al utilizar la información de altura y duración combinada de la forma propuesta, frente a utilizar únicamente la información de altura. Se aprecia claramente que incluir la información de duración aumenta la capacidad de discriminación del sistema de búsqueda. A pesar de las frecuentes aproximaciones de tiempo que se realizan al cantar, la información de duración que proporciona el usuario resulta útil para reconocer la melodía.

8.6.3. Refinamiento utilizando el contorno de f0

Debido a que no existe una técnica perfecta de segmentación de audio en notas, algunos sistemas de búsqueda eluden la etapa de transcripción utilizando el contorno de frecuencia fundamental (*f0*) para comparar melodías. Para ello se utiliza normalmente la técnica de Deformación Temporal Dinámica (DTW), la que desafortunadamente presenta una serie de limitantes para esta aplicación. Al trabajar con secuencias largas (mucho más que las secuencias de notas) el tiempo de procesamiento requerido se torna intolerable. Adicionalmente, es necesario imponer que el usuario cante un fragmento de la melodía previamente definido[52][53].

Al contar con la transcripción de la consulta es posible utilizar esta técnica liberándose de las limitantes mencionadas. Si se realiza una búsqueda basada en notas es posible identificar las mejores ocurrencias del patrón buscado en la base de datos. De esta forma no se imponen restricciones sobre la consulta y es posible aplicar DTW a un conjunto reducido de candidatos. Se implementó una segunda etapa del algoritmo de búsqueda usando esta técnica para refinar la selección de candidatos eludiendo errores de segmentación.

Deformación Temporal Dinámica Local

Intuitivamente la forma de comparar dos series temporales de distinto largo es, en primer lugar, ajustarlas al mismo largo y luego compararlas punto a punto permitiendo cierta deforma-

ción temporal. La técnica de Deformación Temporal Dinámica Local (*LDTW*, Local Dynamic Time Warping) permite comparar series temporales de esta forma.

Dadas dos series temporales x e y de largo m, para calcular la distancia k-ésima de LDTW se construye una matriz $\mathcal{D}(m, m)$ usando la siguiente ecuación de recurrencia,

$$d_{ij} = \begin{cases} |x_i - y_j|^2 + \min \begin{cases} d_{i-1,j-1} \\ d_{i,j-1} \\ d_{i-1,j} \end{cases} & |i-j| \leq k \\ \\ \infty & |i-j| > k \end{cases}$$

para lo cual la matriz debe inicializarse con $d_{1j} = |x_1 - y_j|^2$ con $j \in [1, k+1]$ y $d_{i1} = |x_i - y_1|^2$ con $i \in [1, k+1]$. El valor de distancia es la raíz cuadrada del mínimo de d_{mj} y d_{im} con $i, j \in [m-k, m]$.

El cálculo de la distancia k-ésima LDTW puede implementarse usando el algoritmo de Programación Dinámica restringido a una banda diagonal de ancho $2k+1$ (ver figura 8.11), lo que requiere un número de operaciones de $O(km)$.

La deformación temporal local máxima permitida de una secuencia respecto a la otra es k muestras. Es sencillo observar que la distancia Euclidiana corresponde a la distancia LDTW con $k = 0$.

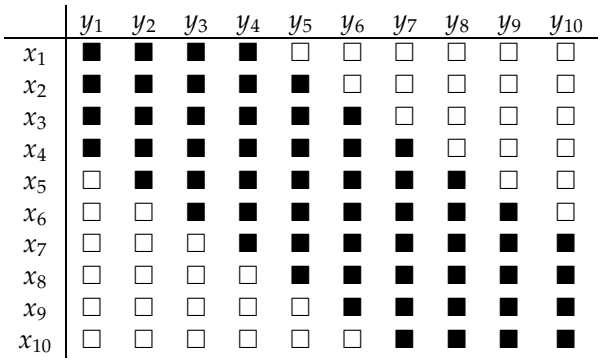


Figura 8.11: La distancia LDTW puede calcularse por medio del algoritmo de Programación Dinámica restringido a una banda de ancho $2k+1$. En la figura se representa la matriz de Programación Dinámica indicando la banda para el cálculo de la distancia LDTW con $k = 3$ entre dos series x e y.

Implementación

A continuación se describe como se utiliza la técnica de comparación de series temporales para depurar la búsqueda basada en notas. En primer lugar se construye una serie temporal a partir de la secuencia de notas de cada ocurrencia de la consulta en la base de datos. Luego se obtiene la distancia LDTW entre cada una de estas series y el contorno de frecuencia fundamental de la señal de voz. Por último se ordenan los candidatos según el valor de distancia. Los pasos involucrados en este proceso son los siguientes,

Ubicar el comienzo y fin de la ocurrencia. Esto se realiza en la primera etapa de búsqueda al calcular la distancia de edición. Identificar el comienzo y fin de la ocurrencia es encontrar el alineamiento entre las secuencias, lo que constituye un problema difícil propenso a errores. El alineamiento se determina a través del camino mínimo (ver sección 8.5.3). Sin embargo, el camino mínimo no es único. Mas aún, cuando se establecen tolerancias en el cálculo de la distancia de edición se incrementa el número de caminos mínimos. Por esta razón, para procurar un buen alineamiento entre las secuencias se elimina la tolerancia. El comienzo y el fin se establecen como los índices del primer y último elemento del camino mínimo cuyos intervalos de altura coinciden exactamente.

Normalizar el tiempo. La consulta y la ocurrencia tienen normalmente distinto tempo, por lo que sus duraciones serán diferentes. Para aplicar la técnica de LDTW se requiere que las series a comparar sean del mismo largo. Es necesario entonces normalizar el tiempo de las ocurrencias para igualar la duración de la consulta. En esta etapa se descartan los candidatos de tempo excesivamente distinto al de la consulta (mayor al doble y menor a la mitad) estableciendo límites al factor de normalización.

Construir el contorno de altura. A partir de la secuencia de notas de la ocurrencia normalizada en el tiempo se construye una serie temporal de valores de altura a la misma tasa de muestreo que el contorno de f0 de la consulta. Los tramos de silencio se completan con el valor de altura de la nota previa. Esto también se realiza para el contorno de f0 de la voz. La razón de lo anterior es la imprecisión en la duración de las notas en una melodía cantada.

Decimar. Para reducir el costo computacional se deciman las series temporales en un factor de 10 (de 100 Hz a 10 Hz).

Pasar a forma normal. Es necesario llevar las series a forma normal para poder compararlas, es decir transformarlas en series de varianza uno y media nula. Esto se logra restando a la serie su media y dividiendo entre su varianza[60]. Si los valores de altura son números MIDI, dos series que correspondan al mismo fragmento de melodía tendrán la misma varianza independientemente de su altura[2]. En este caso, para pasar a forma normal solo es necesario restar la media.

Aplicar LDTW. Se calcula la distancia LDTW entre las series correspondientes a la consulta y cada una de las ocurrencias. El valor de k utilizado es 10, lo que implica una deformación temporal máxima de un segundo. La deformación máxima permitida es relativamente grande para que aún cuando existan errores de alineamiento entre las series, el valor de distancia sea representativo.

En la figura 8.12 se muestra la transcripción de la consulta y la ocurrencia correspondiente a la melodía buscada, representadas en una pianola. Se incluye también la comparación entre las series temporales de altura. Si bien se aprecia cierta similitud entre las secuencias de notas, la correspondencia nota a nota no es muy definida. En estos casos la búsqueda basada en notas puede no ser suficiente para identificar la pieza buscada. Comparar las series de altura es una alternativa inmune a los errores de transcripción que permite depurar la búsqueda.

[2]Esto no sucede si los valores de la serie son alturas en Hz.

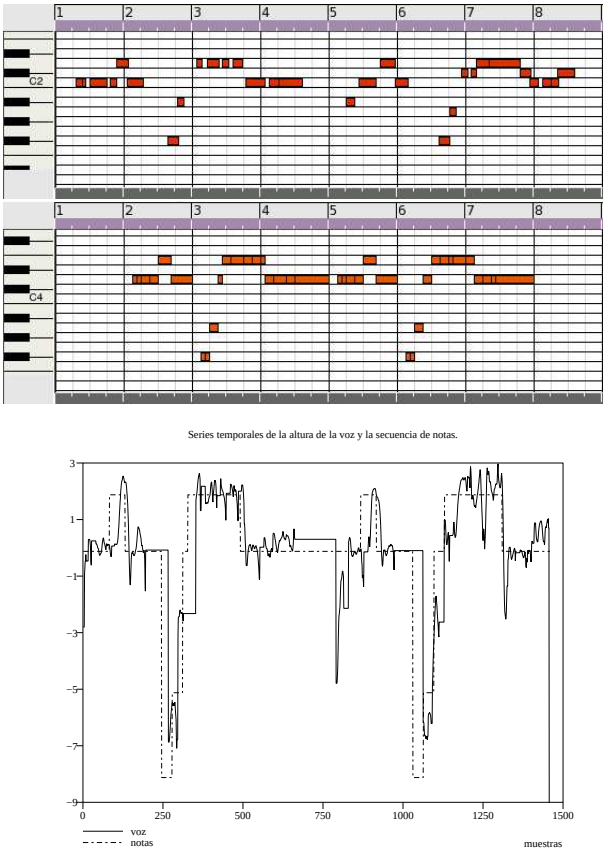


Figura 8.12: Secuencias de notas de la consulta (superior) y ocurrencia (inferior) de los primeros compases de la canción "We can work it out". En la parte inferior de la figura se presentan sus correspondientes series temporales luego de ser normalizadas en el tiempo, transformadas a forma normal y alineadas por el algoritmo implementado.

Sólo se recurre a esta segunda etapa de búsqueda si en la búsqueda basada en notas no hay un candidato cuyo valor de *similitud* esté lo suficientemente alejado del resto. El número de candidatos que pasan a la segunda etapa está limitado a no más de 20. Las evaluaciones realizadas indican que si el candidato buscado no se ubica dentro del grupo de los 20 primeros, los errores en la interpretación o la transcripción hacen que la consulta sea irreconocible.

Resultados

En la figura 8.13 se presenta el incremento en el desempeño del sistema al agregar la etapa de comparación de series temporales. La evaluación señala que la segunda etapa generalmente tiene éxito en determinar el resultado correcto a partir del conjunto de candidatos extraído de la búsqueda primaria.

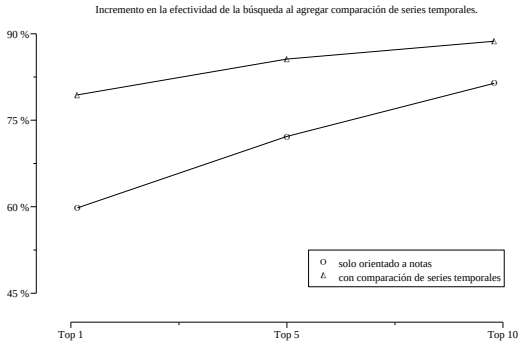


Figura 8.13: Incremento en la efectividad del algoritmo de búsqueda al agregar la etapa de comparación de los contornos de frecuencia. La gran dificultad de la búsqueda orientada a notas es que no existe un sistema de transcripción automática perfecto. Sin embargo, el contorno de f0 de la voz puede obtenerse de forma precisa.

9

Implementación

9.1. Resumen

Combinando los algoritmos desarrollados se implementó un sistema de transcripción y búsqueda de melodías. A continuación se describen las características de la implementación.

9.2. Generalidades

Una de las decisiones de diseño de este proyecto fue la de trabajar exclusivamente con herramientas de software libre. A grandes rasgos las características del software libre son: código abierto, libertad de distribución, libertad de uso y libertad de modificación. El software usado en el desarrollo del proyecto (sistemas operativos, compiladores, editores de audio, procesadores de texto, etc.) cumple con estas características. A su vez, la intención es que el software producido en este proyecto se libere bajo alguna licencia de software libre.

Estas decisiones se basan en distintos motivos. En primer lugar el software libre favorece la divulgación del conocimiento lo que contribuye al desarrollo tecnológico y a la formación profesional. Por otro lado, en el plano técnico, permite estudiar el código y corregirlo o adaptarlo a necesidades particulares. Asimismo, el fácil acceso al software posibilita su amplio uso, lo que contribuye a la depuración y al desarrollo, ya sea mediante reporte de fallas, sugerencias o aporte de código. Estas ventajas fueron confirmadas durante el transcurso del proyecto.

9.2.1. Implementación de prototipos

Cada algoritmo estudiado fue programado de modo de ajustar sus parámetros y evaluar su desempeño. De esta forma se compararon las distintas técnicas y se seleccionaron las más adecuadas. Luego se implementaron prototipos de los sistemas de transcripción y de búsqueda, lo que permitió calibrarlos y observar el funcionamiento global. Para programar los algoritmos y prototipos se utilizó Scilab[61].

9.2.2. Lenguaje de programación y plataforma

En base a los prototipos se desarrolló una aplicación usando el lenguaje de programación C++ sobre GNU/Linux y arquitectura Intel compatible. La elección de este lenguaje se debe fundamentalmente a dos razones. Una de ellas es que en aplicaciones de audio es importante la rapidez de procesamiento ya que se trabaja con gran cantidad de datos. Al ser C++ un lenguaje compilado permite obtener aplicaciones más rápidas que usando lenguajes interpretados como Java. La otra razón es la amplia utilización de este lenguaje por lo que hay una gran cantidad de bibliotecas disponibles.

En relación a la portabilidad, se intentó cumplir con el estándar ISO de C++ por lo que debería ser posible compilar la aplicación en otras plataformas, si bien no se hicieron esfuerzos por hacerlo.

9.3. Sistema de transcripción

El prototipo en Scilab del sistema de transcripción se programó en C++ utilizando las bibliotecas

sndlib[62] para el manejo de archivos y señales de audio, *Midiio*[63] para trabajar con archivos MIDI y *libDSP*[64] modificada, que implementa operaciones con vectores. La documentación del código fuente se incluye en el documento *Tararira Manual de Referencia*.

Una de las principales dificultades de transformar el prototipo a C++, fue la implementación de algunas herramientas matemáticas y de tratamiento de señales que incluye Scilab. Dentro de las más interesantes se encuentran una rutina para el diseño de un filtro Butterwoth paramétrico en orden y tipo, y la resolución de un problema de mínimos cuadrados, los cuales se tratan detalladamente en el apéndice B.

9.4. Sistema de búsqueda

El sistema de búsqueda se implementó en C++ a partir del prototipo en Scilab. En esta sección se describen algunos aspectos técnicos de la base de datos. Se utilizará el término base de datos en su sentido mas amplio.

9.4.1. Base de datos

Descripción

La base de datos consiste en un conjunto de archivos de texto almacenado en un directorio, que incluye toda la información necesaria para la búsqueda. Cada archivo de texto identifica una única pieza musical a través del nombre y contiene la información de la melodía codificada en forma similar al formato MIDI. Mas concretamente, dicha información se expresa en tres columnas que representan respectivamente el tiempo de comienzo, la duración y la altura de cada nota (ver figura 9.1).

Es necesario resaltar que la base de datos se implementó de la manera descripta debido a que el cuerpo de música que contiene es relativamente pequeño y entonces, el tiempo de búsqueda no es excesivo. Sin embargo, para un cuerpo de música mas extenso, una base de datos real resulta mas eficiente y fácil de mantener.

```
3        1.84375  60
4.84375  0.25     60
5.21875  0.25     62
5.59375  0.25     64
5.96875  0.3125   67
⋮        ⋮        ⋮
```

Figura 9.1: Aspecto de un elemento de la base de datos. Las dos primeras columnas indican el tiempo de comienzo y la duración en segundos de cada nota. La tercer columna contiene la altura en formato MIDI.

Construcción

Se recolectaron de Internet todas las canciones de *The Bealtes* en formato MIDI[65][66][67] y se emplearon para construir la base de datos. La motivación de esta decisión es que a partir de su vasta discografía es posible obtener una base de datos completa y de tamaño adecuado para la evaluación del sistema. Además, su música está ampliamente difundida por lo que casi cualquier persona es capaz de tararear alguna de sus canciones.

Debido a que el sistema de búsqueda se implementó para trabajar sobre audio monofónico, es necesario extraer la melodía de los archivos MIDI. Esto debe hacerse manualmente, ya que no existe ninguna forma automática satisfactoria. Surge entonces el complejo dilema de establecer cual es la melodía en un pieza musical polifónica. Dicho problema escapa al alcance de este trabajo por lo que se adoptó una solución práctica que normalmente proporciona buenos resultados: se asume que la melodía de la pieza es la melodía interpretada por el cantante.

A continuación se describe el proceso de construcción de la base de datos. En primer lugar, se extrae la melodía del archivo MIDI polifónico. Para ello se utilizó el excelente software de edición y composición *Rosegarden*[68]. Esta tarea es sencilla, ya que típicamente hay un canal dedicado exclusivamente a la melodía. Luego mediante el software *midi2text*, que utiliza la biblioteca *Midiio*, se convierte el archivo MIDI monofónico a texto con el formato indicado en la figura 9.1. Por último, se asocia este archivo de texto con los datos del tema

correspondiente (título y álbum) a través de un archivo que contiene la información de toda la base de datos. La descripción del proceso da la pauta de la sencillez que implica la ampliación de la base de datos.

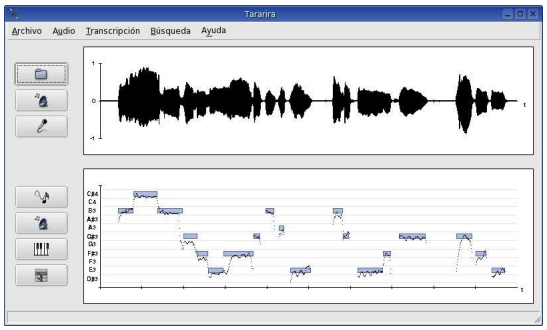


Figura 9.2: Interfaz gráfica de la aplicación implementada. Se observa una consulta y su transcripción. La transcripción se muestra en forma de pianola.

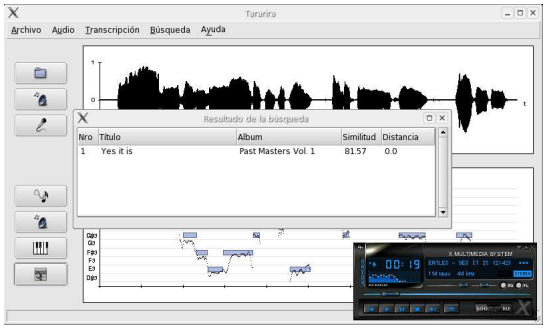


Figura 9.3: El resultado de la búsqueda se presenta como una lista ordenada de candidatos.

9.5. Interfaz gráfica

La aplicación desarrollada cuenta con una interfaz gráfica de usuario, implementada en GTK+2.0[69], que permite la interacción amigable con el sistema de transcripción y búsqueda.

A través de la interfaz gráfica es posible ingresar una consulta al sistema grabando una señal de audio desde el micrófono, transcribir dicha consulta y realizar la búsqueda. La aplicación permite reproducir la señal de audio grabada, así como los archivos que produce la transcripción (formatos wav y MIDI). Una vez finalizada la búsqueda se despliega una lista ordenada de candidatos. Es posible reproducir los archivos que devuelve la búsqueda seleccionándolos de la lista. En el caso de que la búsqueda devuelva un único candidato (ver Capítulo 8) este comienza a reproducirse automáticamente.

La figura 9.2 es una imagen de la aplicación en la que se aprecia una señal de audio en la parte superior y su correspondiente transcripción en la parte inferior. Luego de realizar la búsqueda se presenta una lista ordenada de resultados como se muestra en la figura 9.3.

10

Evaluación

10.1. Resumen

Se llevaron a cabo una serie de evaluaciones del sistema de búsqueda en condiciones reales para dar una idea aproximada de su funcionamiento y estudiar su alcance. En este capítulo se describen las evaluaciones realizadas y se presentan los resultados.

10.2. Metodología

Los datos recolectados para la evaluación pueden dividirse en dos conjuntos. El primero (que denominaremos *BD1*) se compone de archivos de audio adquiridos asegurando ciertas condiciones básicas: ambiente silencioso, instrucciones sobre la forma de realizar la grabación (por ejemplo, distancia al micrófono). El segundo grupo (*BD2*) fue recolectado en condiciones mucho más libres y comprende usuarios de habilidad musical muy heterogénea.

Las grabaciones fueron realizadas usando computadoras personales de uso doméstico, con tarjetas de audio y micrófonos de bajo costo. En ningún caso se establecieron restricciones sobre la forma de cantar o el tramo de melodía interpretado.

10.3. Resultados

Debido a que las principales características que determinan el desempeño del sistema son la forma de cantar y el largo de la consulta, se dividieron los datos en voz cantada y tarareada así como en consultas cortas y largas. Se establece que una consulta es corta si tiene un máximo de 15 notas (por simplicidad se contabilizan las notas detectadas, que pueden diferir de las interpretadas).

El hecho de intentar devolver exclusivamente el tema musical que se busca, tiene como contrapartida el inconveniente de que en algunos casos se retorna un único candidato equivocado. Debido a que estos errores son especialmente indeseables, también se contabilizan y presentan separadamente como *Errores graves*.

Voz cantada		
	Cortas	Largas
Cantidad	50	218
Notas promedio	12,08	26,20
Primero (%)	71,43	75,69
Primeros 10 (%)	78,00	82,75
Errores graves (%)	10,2	0,41
Tarareo		
	Cortas	Largas
Cantidad	28	68
Notas promedio	11,86	23,03
Primero (%)	75,00	82,35
Primeros 10 (%)	89,29	88,24
Errores graves (%)	3,57	4,41
Silbido		
Cantidad	14	
Notas promedio	22,36	
Primero (%)	78,57	
Primeros 10 (%)	100	
Errores graves (%)	0	

Figura 10.1: Resultados de la evaluación realizada para la base de datos *BD1*.

Voz cantada		
	Cortas	Largas
Cantidad	30	43
Notas promedio	10,7	25,09
Primero (%)	26,67	41,86
Primeros 10 (%)	46,67	51,16
Tarareo		
	Cortas	Largas
Cantidad	20	18
Notas promedio	12,05	21,56
Primero (%)	50,00	55,56
Primeros 10 (%)	55,00	66,67
Silbido		
Cantidad	24	
Notas promedio	21,58	
Primero (%)	41,67	
Primeros 10 (%)	45,83	

Figura 10.2: Desempeño del sistema para la base de datos *BD2*.

10.4. Conclusiones

En primer lugar, la evaluación confirma algunas hipótesis sobre el problema. El peor escenario es la voz cantada, como se refleja en los resultados obtenidos. Esto se debe a que las características que presenta la voz cantada con letra dificultan su transcripción. Sin embargo, la transcripción de la voz tarareada o el silbido es considerablemente más sencilla, ya que en muchas ocasiones la señal está naturalmente segmentada. Cabe señalar que bajo tarareo se incluyó todo lo que no es voz cantada, es decir que no se restringe a los casos en que se canta exclusivamente con sílabas que comienzan con fonemas oclusivos. Esto último hace que la segmentación pueda no ser evidente (ver Capítulo 5). Por otra parte, también se confirma que al aumentar el largo de la consulta el desempeño mejora, ya que la melodía se torna más identificable.

En cuanto a los *Errores graves*, su bajo porcentaje no significa un costo excesivo frente a la atractiva propiedad del sistema de retornar un único elemento en lugar de una larga lista de candidatos.

Los resultados muestran una enorme diferencia en el desempeño del sistema para las distintas bases de datos. Esto se debe a diversas razones. En primer lugar, los usuarios de la base de datos *BD1* generalmente interpretan una melodía que les es familiar, lo que es necesario para lograr el objetivo del sistema de retornar un único candidato. Que el usuario esté familiarizado con la melodía que interpreta colabora a que la consulta esté bien formulada[59]. Por otra parte, las grabaciones de la base de datos *BD2* presentan en general una serie de características que alteran el desempeño del sistema: baja relación señal a ruido, ambiente ruidoso y consultas mal formuladas. Al escuchar las grabaciones es posible apreciar risas, soplidos sobre el micrófono, conversaciones de fondo o varias personas cantando simultáneamente. Asimismo, algunas consultas demuestran muy poca habilidad musical del usuario o recuerdo vago de la melodía. Existen también situaciones difíciles de resolver como melodías interpretadas en *bocca chuisa*[1]. Es importante destacar que al desarrollar el sistema se utilizó como fuente de pruebas, consultas del tipo de las de la base de datos *BD1*, es decir bajo las condiciones básicas mencionadas.

La conclusión más importante que puede extraerse de las evaluaciones realizadas con *BD1* es que el sistema parece funcionar correctamente bajo las condiciones para las cuales fue diseñado. La experiencia realizada con *BD2* muestra el comportamiento espontáneo de los usuarios la primera vez que se enfrentan a un sistema de este tipo. Sin embargo, es probable que no represente fielmente la situación real en la que un usuario interesado en encontrar cierto tema musical pone todo su empeño.

[1] Del italiano *boca cerrada*.

11

Conclusiones y trabajo futuro

11.1. Resumen

En este capítulo final se analiza el trabajo realizado desde una perspectiva crítica, destacando sus virtudes y sus limitantes. Se sugieren líneas de trabajo futuro.

11.2. Conclusiones

En vista de la amplitud y complejidad del problema de búsqueda de música por melodía, el presente trabajo es apenas una aproximación al tema. De todas formas, se abordaron varios de los aspectos involucrados y se logró desarrollar un sistema completo. Esto último tiene cierta relevancia, ya que generalmente cada grupo de investigación trabaja en una parte específica del problema[2]. Otro elemento a señalar es el modo de trabajo adoptado. Para resolver cada etapa del sistema, se estudiaron las técnicas existentes, se implementaron prototipos de las más prometedoras y se seleccionaron las más apropiadas adaptándolas a la aplicación. Por esta razón, las técnicas utilizadas se acercan al estado del arte del tema, lo que se ve reflejado en las evaluaciones realizadas. La integración de los avances más recientes fue posible gracias a que la mayoría de los investigadores en el ámbito académico comparten su trabajo por Internet.

A la hora de implementar un sistema de búsqueda de música es necesario definir su alcance. Este sistema se enfoca a retornar únicamente el tema musical buscado, lo que impone ciertas restricciones respecto a los usuarios. Como se estableció en el Capítulo 8 existe un compromiso entre tolerancia a errores en la interpretación y la capacidad de discriminación entre melodías. Al intentar identificar un único tema, se prioriza la discriminación frente a la tolerancia a errores. El usuario entonces debe estar preferentemente familiarizado con la melodía que busca, de forma tal de elaborar una consulta razonable. Otra restricción para que la búsqueda sea exitosa es que la consulta debe ser lo suficientemente larga (unas 15 notas). En general la mayoría de los sistemas desarrollados son más flexibles en cuanto a la habilidad del usuario y el largo de la consulta, pero despliegan como resultado una larga lista de posibilidades.

En cuanto a las melodías que se almacenan en la base de datos surgen otras limitantes. Es necesario que exista un elemento de la base de datos que contenga explícitamente la melodía que canta el usuario. Esto probablemente no ocurra si se canta una combinación de instrumentos, como por ejemplo un tramo de voz seguido de un riff de guitarra. La solución generalmente adoptada es almacenar las líneas melódicas más factibles de ser cantadas, a pesar de que esto puede ser difícil de determinar. Una opción más apropiada es buscar sobre archivos MIDI polifónicos, pero esto aumenta enormemente la complejidad y tiende a incrementar las coincidencias falsas.

Una decisión delicada es establecer el grado de interacción con el usuario. Este sistema pretende minimizar la interacción de modo que la aplicación sea de uso sencillo. Es poco probable que una persona utilice un sistema que no comprende con facilidad o que impone demasiadas restricciones sobre su comportamiento. Por esta razón no se incluye la etapa de detección automática de tempo en el sistema, ya que es necesario elegir la opción correcta dentro de un conjunto de posibilida-

des. El sistema desarrollado entonces, no requiere el ajuste de parámetros ni impone restricciones a la forma de cantar. En contrapartida no puede adaptarse a la habilidad del usuario. Para permitir búsquedas personalizadas, por ejemplo, según el grado de confiabilidad de la consulta, se debe aumentar la interacción.

En este trabajo se hace especial énfasis en la etapa de transcripción. La correcta transcripción de la consulta es requisito esencial para el buen funcionamiento del sistema. Por otro lado, transcribir automáticamente la voz cantada tiene aplicaciones más allá de este problema. Se pudo confirmar en la práctica la recurrente afirmación de que tratar la voz cantada es una tarea sumamente compleja. El peor escenario lo constituye la voz cantada con letra. En este caso el contorno de frecuencia fundamental y la envolvente de la señal de voz presentan una infinidad de particularidades difíciles de caracterizar que entorpecen el análisis[1]. Esto es especialmente notorio en los cantantes no experimentados. Sin embargo, en el caso de tarareo o silbido, el desempeño de la etapa de transcripción es más que apropiado para esta aplicación.

La música se graba y distribuye en forma digital, por lo que existen grandes archivos de música en este formato. Sin embargo, la cantidad de música disponible en notación simbólica es significativamente menor. Ya que no existe una forma automática de pasar del formato digital a la notación simbólica (transcripción de música polifónica), para implementar un sistema real de búsqueda es necesario que los formatos de audio incluyan información de contenido. Esto está actualmente en proceso a través del estándar MPEG-7.

Cabe señalar algunas críticas respecto al enfoque de comparar melodías basado en notas. Identificar melodías comparándolas nota a nota es una simplificación enorme del proceso cognitivo de reconocimiento de melodías. Reducir el problema al nivel de notas permite desarrollar aplicaciones de juguete, pero ha sido imposible hasta el momento trasladarlas a situaciones reales[1]. Con esto no se pretende descalificar este tipo de trabajos, ya que permiten esclarecer un poco algunos aspectos del funcionamiento de estos mecanismos mentales. Mas allá de traducir la consulta a notas parecen ser necesarios otros niveles de descripción o comprensión de la misma.

Como consecuencia de las críticas al enfoque orientado a notas, recientemente se estudia la posibilidad de identificar melodías directamente a partir de características derivadas de la señal de voz, como el contorno de frecuencia fundamental. Lamentablemente, esto también presenta desventajas, por ejemplo, su alto costo computacional. Esta alternativa y la basada en notas son consideradas siempre antagónicas y no se conocen antecedentes de usarlas en forma conjunta. En este trabajo se combina de forma novedosa ambos enfoques aprovechando las ventajas de cada uno.

11.3. Trabajo futuro

Las evaluaciones realizadas en este trabajo permiten visualizar en forma aproximada el funcionamiento del sistema y de sus distintas etapas. Sin embargo, para tener una idea más precisa es indispensable llevar a cabo evaluaciones exhaustivas y con procedimientos acordados en conjunto con otros grupos de investigación para poder comparar resultados. El sistema debe probarse bajo condiciones mas reales, es decir utilizando bases de datos mas extensas y con una gama de usuarios mas amplia. Por otro lado, es posible extender los objetivos del sistema integrando búsquedas mas flexibles, por ejemplo, para identificar una melodía que se recuerda muy vagamente.

Cómo ya se mencionó, la etapa de transcripción automática es la mas crítica de un sistema de búsqueda por melodía. Por esta razón, merece especial atención y es necesario continuar su desarrollo, ya que hasta hoy no existe una solución completamente satisfactoria. Asimismo, existen diversas aplicaciones de la transcripción automática fuera de este problema. Por ejemplo, sería de gran utilidad para los músicos que no dominan la escritura musical. Existen algunas direcciones por donde continuar este trabajo. Una de ellas es estudiar la posibilidad de incluir la detección automática de tempo en la transcripción. También sería interesante extenderla a otros instrumentos musicales más allá de la voz cantada.

Respecto al problema de búsqueda de música por melodía existe mucho trabajo por delante, el cual involucra distintas áreas del conocimiento. Es necesario profundizar en el estudio las características de la voz cantada, con el objetivo de intentar entender la melodía más allá de simplemente

transcribirla. Del mismo modo, se requiere ahondar en los mecanismos mentales involucrados en el reconocimiento de melodías, de forma de definir criterios de similitud más cercanos a los de la percepción. Por otro lado, parece ser acertada la idea de encarar el problema en un dominio más cercano al audio, fundamentalmente complementando el enfoque basado en notas, como se realiza en este trabajo.

A

Música utilizada en detección de tempo

Tema	Estilo	Artista
Romanian Folk Dances	Clásico	B. Bartok
Danse Bohemienne	Clásico	C. Debussy
Rondo alla Turca	Clásico	W. A. Mozart
Paradise City	Rock	Guns n' Roses
Sweet child oh mine	Rock	Guns n' Roses
If	Rock	J. Satriani
Sunday	Jazz	S. Getz
Swingin'	Jazz	A. Sandoval
Bluebird	Jazz	C. Parker
Superstition	Pop	Stivie Wonder
Se...	Pop	Djavan
Epilogue	Pop	Sting
Chacarera de las Piedras	Chacarera	A. Yupanqui
Chacarera Santiagueña	Chacarera	A. Yupanqui
Para cantar he nacido	Chacarera	M. Sosa
Hermanos	Milonga	A. Zitarrosa
Negro milonguero	Milonga	A. Zitarrosa
El aromo	Milonga	A. Yupanqui
Yira, yira	Tango	C. Gardel
Caliente	Tango	A. Piazzola
Volvió una nocha	Tango	R. Goyeneche
Chega de saudade	Bossa	C. Veloso
Garota de Ipanema	Bossa	J. Gilberto
A felicidade	Bossa	T. Jobim
Montevideo	Candombe	R. Rada
Místico	Candombe	Lady Jones
Candombe de hoy	Candombe	H. Fatorruso

Figura A.1: Datos de música utilizada

B

Detalles de implementación

B.1. Mínimos Cuadrados

B.1.1. Planteo del problema

En la estimación de la frecuencia fundamental de una trama de señal se aproximan los mínimos de la función diferencia normalizada por parábolas. Esto permite aumentar la resolución en la estimación y evitar errores al buscar el menor mínimo (ver sección 4.5.3).

Para cada mínimo de la función diferencia se consideran cinco muestras centradas en el mínimo. Denominamos $t = [\, t_1 \quad t_2 \quad t_3 \quad t_4 \quad t_5 \,]^T$ y $b = [\, y_1 \quad y_2 \quad y_3 \quad y_4 \quad y_5 \,]^T$ a los vectores de índices y valores de la función diferencia respectivamente, a aproximar por una parábola. El problema de mínimos cuadrados en este caso consiste en determinar los coeficientes de la parábola $f(t) = \alpha + \beta t + \gamma t^2$ que minimicen el error cuadrático,

$$\sum_{i=1}^{5} \left[(\alpha + \beta t_i + \gamma t_i^2) - y_i \right]^2 .$$

La solución de este problema se obtiene resolviendo las ecuaciones normales,

$$A^T A x = A^T b$$

donde,

$$A = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \\ 1 & t_4 & t_4^2 \\ 1 & t_5 & t_5^2 \end{bmatrix}, \quad x = \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix}, \quad b = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix}.$$

El número de condición del problema de mínimos cuadrados es $\text{cond}(A^T A) \approx \text{cond}(A) \times \text{cond}(A)$. La matriz A en este caso es una matriz de Van der Monde la cual tiene típicamente un número de condición alto, por lo que el problema está mal condicionado.

B.1.2. Implementación

En una primera instancia de la implementación se resolvieron directamente las ecuaciones normales usando la descomposición LU y simple precisión. Se pudo comprobar que de esta forma los errores numéricos son intolerables para la aplicación. Como alternativa para reducir el número de condición del problema se utilizó la descomposición QR.

Una matriz $A(m, n)$ se puede factorizar en una matriz $Q(m, n)$ ortonormal y una matriz $R(n, n)$ triangular superior tal que $A = QR$. Usando esta descomposición las ecuaciones normales pasan a ser,

$$Rx = Q^T b.$$

El número de condición del problema es $\text{cond}(R) \approx \text{cond}(A)$, ya que

$$\text{cond}(A) \approx \text{cond}(Q) \times \text{cond}(R) \approx \text{cond}(R)$$

por ser Q una matriz ortonormal. Para la descomposición QR se utilizó el método de Gram-Schmidt modificado, que es numéricamente más estable que el método clásico de Gram-Schmidt.

De esta manera los errores numéricos se redujeron de forma que no afectan el funcionamiento del algoritmo, sin incrementar el orden del número de operaciones.

B.2. Filtro Butterworth

B.2.1. Introducción

El procesamiento de la señal de audio para su transcripción requiere el uso de filtros de selección de frecuencias. Se optó por la utilización de filtros IIR de la clase Butterworth. La técnica de diseño usada consiste en transformar un filtro analógico al dominio digital mediante una transformación bilineal[19][37][70].

Los filtros IIR se caracterizan por su mayor rapidez y peor desempeño (en cuanto a la selectividad) respecto a los filtros FIR. Sin embargo la aplicación no requiere una selección de frecuencias estricta, por lo que su desempeño es adecuado. Por esta misma razón, la elección de filtros de la clase Butterworth también es apropiada ya que, su lenta transición entre la zona pasabanda y suprimebanda (roll-off) no es problemática. Además poseen la ventaja de no presentar ripple en la respuesta en frecuencia y ser de diseño sencillo.

Se programó un conjunto de funciones que permiten diseñar filtros de esta clase, paramétricos en orden y tipo.

B.2.2. Filtro Butterworth

Existen distintas clases de filtros analógicos utilizados para selección de frecuencias. La clase de filtros *Butterworth* es la que presenta la respuesta en frecuencia más plana en la banda pasante. Sin embargo para un número dado de polos tiene el roll-off más lento. La magnitud de su respuesta en frecuencia tiene un comportamiento monótono. Por otro lado los filtros *Chebyshev* presentan roll-off más rápido a costa de permitir cierto grado de ripple en la banda pasante (Tipo I) o en la banda atenuante (Tipo II). La clase de filtros *elípticos* tienen ripple tanto en la banda pasante como en la atenuante pero su roll-off es el más rápido dado cierto número de polos.

El módulo al cuadrado de la respuesta en frecuencia de un filtro pasabajos Butterworth analógico de orden N es,

$$|H(j\Omega)|^2 = \frac{1}{1 + (j\Omega / j\Omega_c)^{2N}} \tag{B.1}$$

donde Ω es la frecuencia en rad/s y Ω_c la frecuencia de corte. A medida que crece el parámetro N (orden del filtro) el roll-off se hace más agudo, es decir, el filtro se acerca al filtro ideal.

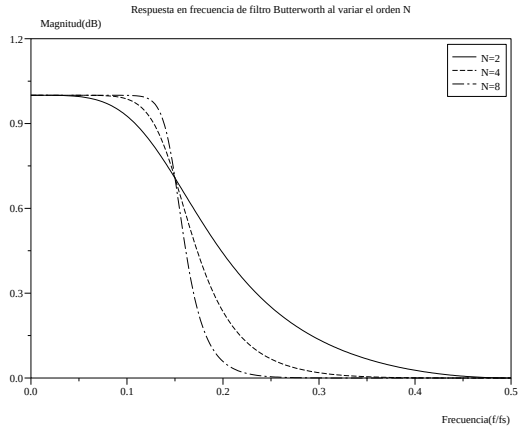


Figura B.1: Respuesta en frecuencia del filtro Butterworth en función del orden. Al aumentar el orden el filtro se acerca al pasabajos ideal.

Sustituyendo $s = j\Omega$ en la ecuación B.1 y calculando las raíces del denominador obtenemos los polos del módulo al cuadrado de la respuesta en frecuencia,

$$s_k = (-1)^{\frac{1}{2N}}(j\Omega_c) = \Omega_c e^{j\frac{\pi}{2N}(2k+N-1)} \tag{B.2}$$

$$\text{con} \quad k = 0, 1, \ldots, 2N-1.$$

Entonces, hay $2N$ polos ubicados de forma equiespaciada sobre la circunferencia de radio Ω_c en el plano s. Es importante señalar que su ubicación es simétrica respecto al eje imaginario.

Típicamente la respuesta en frecuencia de los filtros Butterworth se describe mediante el módulo al cuadrado como en la ecuación B.1, ya que permite una expresión cerrada. Para obtener la función de transferencia $H(s)$ del filtro partiendo de la ecuación B.1, observamos que $|H(j\Omega)|^2 = H(j\Omega)H^*(j\Omega)$ y que si la respuesta al impulso $h(t)$ es real, $H^*(j\Omega) = H(-j\Omega)$. Teniendo en cuenta estas consideraciones y sustituyendo $s = j\Omega$ se obtiene,

$$H(s)H(-s) = \frac{(j\Omega_c)^{2N}}{s^{2N} + (j\Omega_c)^{2N}}. \tag{B.3}$$

Recordando que los polos se presentan en pares simétricos respecto al eje imaginario, es posible ex-

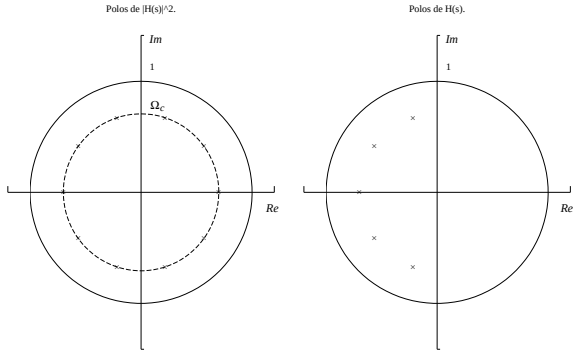


Figura B.2: Ubicación de los polos de $|H(s)|^2$ y $H(s)$ de un filtro Butterworth de orden 5. Los polos de $|H(s)|^2$ se encuentran equiespaciados sobre una circunferencia de radio Ω_c, la distancia angular de separación entre ellos es π/N radianes. La configuración es simétrica respecto al eje imaginario. Para que el filtro sea causal y estable se eligen los polos ubicados en el semiplano izquierdo.

presar el término de la derecha de la ecuación anterior como el producto de dos funciones, una con los polos de parte real positiva y otra con los polos de parte real negativa. Es sencillo observar que ambas funciones son la misma pero con argumento opuesto. Falta asociar $H(s)$ a una de estas funciones. De forma de que el filtro sea causal y estable, $H(s)$ se elige como la función cuyos polos se ubican en el semiplano izquierdo (ver figura B.2). La expresión que se obtiene para $H(s)$ es,

$$H(s) = \frac{(j\Omega_c)^N}{\prod_{k=0}^{N-1}(s - s_k)} \tag{B.4}$$

con

$$s_k = \Omega_c e^{j[\frac{\pi}{2} + \frac{\pi}{2N} + \frac{\pi}{N}k]}, \quad k = 0, 1, \ldots, N-1.$$

B.2.3. Diseño de filtros digitales

Introducción a los filtros digitales La forma más directa de implementar un filtro digital es mediante la convolución de su respuesta al impulso con la señal de entrada. Otra forma de construir filtros digitales es a través de una ecuación de recurrencia. Esto es más general que la convolución ya que involucra muestras de la señal de entrada y muestras previamente calculadas de la salida del filtro. El filtro está determinado, en este caso, por un conjunto de coeficientes de recursión que se obtiene a partir de la función de transferencia del filtro discreto $H(z)$.

En el primer caso se parte de la respuesta al impulso del filtro deseado. Para implementar la convolución es necesario truncar la respuesta al impulso ideal, por lo cual este tipo de filtros se denomina de respuesta al impulso finita (FIR). En el caso de los filtros de recurrencia, la respuesta al impulso está compuesta por sinusoides que decaen exponencialmente en amplitud, siendo infinita en teoría. Por esta razón los filtros de recurrencia reciben el nombre de filtros de respuesta infinita (IIR). La elección ente un tipo u otro implica un compromiso entre desempeño y rapidez, como veremos a continuación.

Un filtro FIR se implementa usando los valores de la respuesta al impulso. Cuanto mayor cantidad de puntos, mejor se aproxima la respuesta ideal. Para una aproximación razonable es necesario un orden de cientos o miles de puntos, dependiendo de la aplicación. Por otro lado, los filtros IIR se implementan utilizando directamente la expresión de la función de transferencia, ya que los coeficientes de recurrencia están dados por los coeficientes de los polinomios numerador y denominador. En el diseño de filtros IIR de selección de frecuencias normalmente son necesarios unos pocos coeficientes (del orden de la decena). La cantidad de puntos involucrados en cada caso hace que el número de operaciones sea mucho mayor para los filtros FIR, por lo que los filtros IIR son notoriamente más rápidos.

Un ejemplo servirá para aclarar estas ideas. La intención es comparar un filtro IIR y un filtro FIR pasabajos de respuesta similar. Como filtro IIR tomemos un Chebyshev de orden 6 y 5 % de ripple, de frecuencia de corte 0.2. Para diseñar el filtro FIR partimos de la respuesta al impulso de un filtro pasabajos ideal (*sinc*). Es necesario considerar 51 puntos de dicha respuesta para igualar el desempeño del filtro Chebyshev en cuanto a selectividad de frecuencia (mismo roll-off), como se aprecia en la figura B.3. Ya que el filtro IIR elegido tiene solo doce coeficientes, resultará más de cuatro veces más rápido.

Sin embargo, respecto al desempeño, los filtros

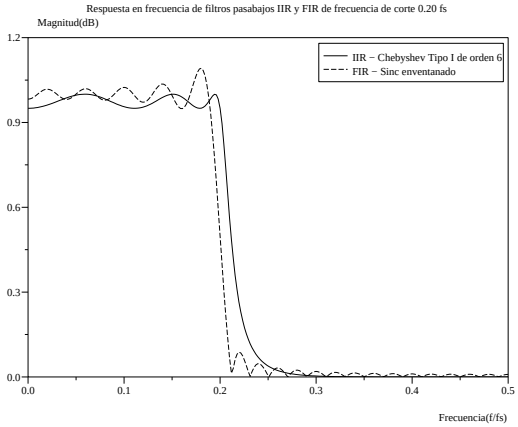


Figura B.3: Respuesta en frecuencia de la magnitud de un filtro IIR Chebyshev de orden 6 y un filtro FIR *sinc* enventanado de 51 puntos.

IIR son limitados: las técnicas de diseño están restringidas a filtros de selección de frecuencias. Es difícil modelar respuestas en magnitud muy particulares utilizando filtros de este tipo. Esto se debe fundamentalmente a dos razones. En muchos casos no es posible obtener una expresión de la respuesta en frecuencia deseada de la cual derivar la ecuación en recurrencia. Por ejemplo, al modelar sistemas reales, comúnmente se puede medir la respuesta al impulso pero no expresar analíticamente su función de transferencia. En estos casos es directo el uso de filtros FIR. En segundo lugar, en la práctica la cantidad de polos que puede utilizarse en un filtro IIR está limitada debido a errores de redondeo. Estos errores degradan el desempeño del filtro y provocan eventualmente que el sistema se torne inestable. Otra limitante de los filtros IIR es que las técnicas de diseño no permiten especificar la respuesta en fase. Si es necesario aproximar una repuesta en fase particular se debe utilizar un filtro FIR.

Diseño de filtros digitales a partir de filtros analógicos El diseño de filtros digitales IIR a partir de filtros analógicos se basa principalmente en la existencia de técnicas ampliamente desarrolladas de diseño de filtros analógicos. Asimismo, muchas de las técnicas de diseño de filtros analógicos tienen expresiones cerradas por lo que los métodos de diseño de filtros digitales a partir de dichas técnicas son de implementación sencilla. Por el contrario las técnicas de diseño de filtros digitales aplicadas directamente en tiempo discreto no conducen a expresiones cerradas.

El procedimiento de diseño consiste en primer lugar en transformar las especificaciones del filtro discreto a tiempo continuo. Luego se diseña el filtro analógico prototipo (Butterworth, Chebyshev, elíptico) que cumple estas especificaciones; esto significa determinar el orden y la frecuencia de corte. Finalmente se obtiene la transferencia del filtro discreto $H(z)$ aplicando una transformación que convierte la función del filtro analógico $H_c(s)$ a tiempo discreto.

La transformación que mapea el plano complejo s en el plano complejo z debe ser tal que las características esenciales de la respuesta en frecuencia del filtro prototipo se mantengan en la respuesta en frecuencia del filtro discreto. Esto requiere que el eje imaginario del plano s se mapee en la circunferencia unidad del plano z[1]. Otra condición importante que debe cumplir dicha transformación es que un sistema analógico estable conduzca a un sistema digital estable. Es necesario entonces que el semiplano izquierdo del plano s sea mapeado en el círculo unidad del plano z. Existen dos tipos de transformaciones que cumplen las condiciones anteriores: *Invarianza al impulso* y *Transformación Bilineal*.

Invarianza al Impulso. En la técnica de Invarianza al impulso, el sistema discreto queda definido por el muestreo de la respuesta al impulso del sistema en tiempo continuo. Si $h_c(t)$ es la respuesta al impulso del sistema analógico, la respuesta al impulso del sistema discreto está dada por $h[n] = T_d h_c(nT_d)$, donde T_d es el período de muestreo. Ya que el sistema discreto proviene del muestreo de una señal analógica, la respuesta en frecuencia del filtro digital puede estar distorsionada por solapamiento de espectro (aliasing), por lo que la técnica no es apropiada para el diseño de filtros de banda ilimitada (por ejemplo, pasaaltos)[2]. Si bien el plano s no se relaciona en forma sencilla

[1] Recordar que la respuesta en frecuencia de un sistema analógico es $H(\Omega) = H(s)|_{s=j\Omega}$, mientras que la respuesta en frecuencia de un sistema discreto es $H(\omega) = H(z)|_{z=e^{j\omega}}$.

[2] No es posible utilizar el parámetro T_d para controlar el aliasing.

con el plano z aplicando esta transformación, puede obtenerse la expresión de la transferencia del filtro discreto $H(z)$ para derivar los coeficientes de recursión.

Transformación bilineal. Esta técnica consiste en aplicar una transformación algebraica entre las variables s y z de la forma,

$$s = 2\left(\frac{1-z^{-1}}{1+z^{-1}}\right). \tag{B.5}$$

Debido a que el eje imaginario del plano s se mapea en una sola vuelta de la circunferencia unidad del plano z, se evita el fenómeno de aliasing presente en la técnica anterior. Como contrapartida, la transformación entre la frecuencia en tiempo continuo y tiempo discreto no es lineal ya que $-\infty \leq \Omega \leq \infty$ corresponde a $-\pi \leq \omega \leq \pi$. La relación entre las variables de frecuencia Ω y ω puede determinarse evaluando la ecuación B.5 en la circunferencia unidad $z = e^{j\omega}$,

$$\Omega = 2\tan(\omega/2). \tag{B.6}$$

Esta compresión no solo se manifiesta en el módulo (ver figura B.4), sino también en la fase de la repuesta en frecuencia del filtro discreto, tornándola no lineal. Por lo tanto, esta técnica puede aplicarse solo en casos donde dicha compresión de frecuencia es tolerable.

Al construir filtros discretos de selección de frecuencia a partir de un prototipo analógico, la transformación bilineal mantiene las características de la respuesta en frecuencia. Asimismo, las expresiones sencillas de estos filtros analógicos hacen que el procedimiento de diseño sea directo al usar este método.

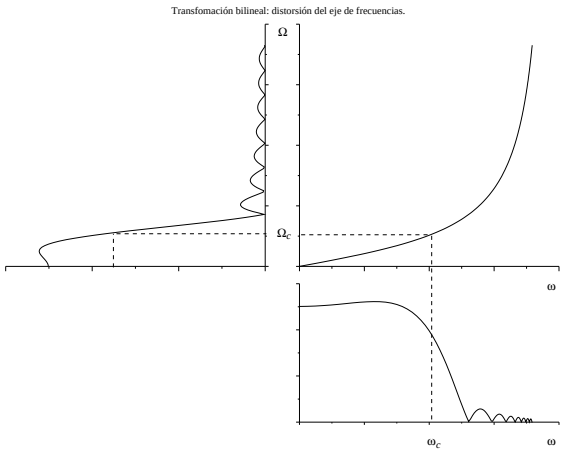


Figura B.4: Distorsión de frecuencia de la transformación bilineal. Se representa gráficamente la relación entre la frecuencia analógica Ω y la frecuencia discreta ω dada por la ecuación B.6. Se observa la respuesta en módulo de un filtro pasabajos.

B.2.4. Transformaciones en Frecuencia

Como ya se mencionó existen técnicas sencillas de diseño de filtros analógicos pasabajos (Butterworth, Chebyshev, elíptico). Afortunadamente, mediante la técnica denominada *Transformación en Frecuencia*, alcanza con diseñar un filtro analógico pasabajos para obtener un filtro IIR de cualquier tipo (pasabajos, pasabanda, etc.). La técnica consiste en obtener la función de transferencia del filtro deseado aplicando una transformación a un filtro pasabajos prototipo de frecuencia de corte arbitraria. La transformación en frecuencia puede llevarse a cabo tanto en el dominio analógico como en el dominio discreto. El diseño de un filtro digital a partir de un prototipo analógico pasabajos se representa en la figura B.5.

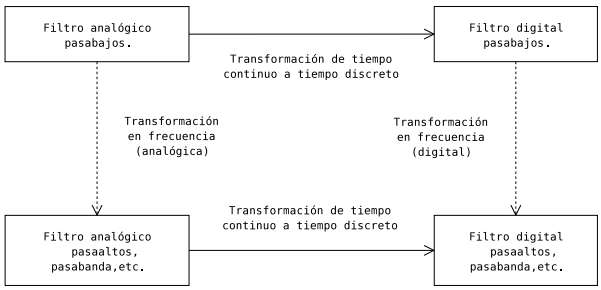


Figura B.5: Alternativas de diseño de un filtro digital a partir de un pasabajos analógico prototipo.

Al igual que la transformación bilineal, la transformación en frecuencia mantiene las características de la respuesta en frecuencia del filtro prototipo. Por ejemplo, para diseñar un filtro pasaaltos sin ripple en las bandas pasante y atenuante, el filtro prototipo pasabajos de partida debe ser de la clase Butterworth, ya que cumple con esas características.

Haremos hincapié en las transformaciones en frecuencia en el dominio discreto. Consideremos

que la transformación de frecuencia mapea el plano complejo Z en el plano complejo z. Para asegurar que se mantengan las propiedades de estabilidad y causalidad, se deben cumplir las siguientes condiciones:

> Transformar la circunferencia unidad del plano Z en la circunferencia unidad del plano z.
>
> Transformar el círculo unidad del plano Z en el círculo unidad del plano z.

Típicamente estas transformaciones consisten en el producto de factores del estilo pasatodo[3],

$$G(z^{-1}) = \pm \prod_{k=1}^{N} \left(\frac{z^{-1} - \alpha_k}{1 - \alpha_k z^{-1}} \right). \tag{B.7}$$

Si la transferencia del filtro prototipo es $H_{lp}(Z)$, el filtro deseado es:

$$H(z) = H_{lp}(Z)\big|_{Z^{-1}=G(z^{-1})}.$$

Tomemos por caso el diseño de un filtro pasabajos de frecuencia de corte ω_p a partir de un filtro digital pasabajos prediseñado, de frecuencia de corte arbitraria θ_p. La transformación en frecuencia que convierte un filtro pasabajos en otro filtro pasabajos alterando la frecuencia de corte es,

$$Z^{-1} = \frac{z^{-1} - \alpha}{1 - \alpha z^{-1}}, \tag{B.8}$$

donde α se elige de forma que θ_p se mapee en ω_p. Para determinar el valor de α se evalúa la ecuación B.8 en la circunferencia unidad ($Z^{-1} = e^{-j\theta}$ y $z^{-1} = e^{-j\omega}$). Despejando ω se obtiene,

$$\omega = \arctan\left(\frac{(1-\alpha^2)\,\text{sen}\,\theta}{2\alpha + (1+\alpha^2)\cos\theta} \right).$$

Para que la frecuencia de corte del filtro prediseñado θ_p se mapee en la frecuencia de corte deseada ω_p, α debe ser,

$$\alpha = \frac{\text{sen}\,\big((\theta_p - \omega_p)/2\big)}{\text{sen}\,\big((\theta_p + \omega_p)/2\big)}.$$

En la figura B.6, se detallan las transformaciones en frecuencia para obtener cada tipo de filtro.

[3] Esto se cumple para valores de α pequeños.

Tipo	Transformación
Pasabajos	$Z^{-1} = \frac{z^{-1}-\alpha}{1-\alpha z^{-1}}$ $\alpha = \frac{\text{sen}\,\big((\theta_p-\omega_p)/2\big)}{\text{sen}\,\big((\theta_p+\omega_p)/2\big)}$
Pasaaltos	$Z^{-1} = \frac{z^{-1}+\alpha}{1+\alpha z^{-1}}$ $\alpha = -\frac{\cos\,\big((\theta_p+\omega_p)/2\big)}{\cos\,\big((\theta_p-\omega_p)/2\big)}$
Pasabanda	$Z^{-1} = -\frac{(K+1)z^{-2}-2\alpha K z^{-1}+(K-1)}{(K-1)z^{-2}-2\alpha K z^{-1}+(K+1)}$ $\alpha = \frac{\cos\,\big((\omega_{p2}+\omega_{p1})/2\big)}{\cos\,\big((\omega_{p2}-\omega_{p1})/2\big)}$ $K = \cot\left(\frac{\omega_{p2}-\omega_{p1}}{2}\right)\tan\left(\frac{\theta_p}{2}\right)$
Suprimebanda	$Z^{-1} = \frac{(K+1)z^{-2}-2\alpha z^{-1}+(1-K)}{(1-K)z^{-2}-2\alpha z^{-1}+(K+1)}$ $\alpha = \frac{\cos\,\big((\omega_{p2}+\omega_{p1})/2\big)}{\cos\,\big((\omega_{p2}-\omega_{p1})/2\big)}$ $K = \tan\left(\frac{\omega_{p2}-\omega_{p1}}{2}\right)\tan\left(\frac{\theta_p}{2}\right)$

Figura B.6: Transformaciones en frecuencia para obtener cada tipo de filtro.

B.2.5. Implementación

Los filtros digitales utilizados en la aplicación son filtros IIR de la clase Butterworth, del tipo pasabajos, pasabanda y pasaaltos. El procedimiento de diseño consiste en partir de un filtro analógico pasabajos prototipo, transformarlo al dominio discreto mediante la transformación bilineal y obtener el filtro deseado usando una transformación en frecuencia.

A continuación se describen más detalladamente los pasos necesarios para la implementación de

una rutina de diseño de filtros Butterworth[4] de orden y de tipo genérico. Los parámetros de diseño son: el tipo de filtro, la(s) frecuencia(s) de corte, y el orden N.

1. *Diseño de filtro analógico pasabajos de la clase Butterworth.* Debido a que el filtro tiene solo polos, se deben determinar únicamente la ganancia G_c y los polos s_k en función de N. Esto puede hacerse directamente a partir de la ecuación B.4,

$$G_c(H(s)) = \Omega_c^N$$

$$s_k = \Omega_c e^{j[\frac{\pi}{2}+\frac{\pi}{2N}+\frac{\pi}{N}k]}, \quad k = 0, 1, \ldots, N-1.$$

   La frecuencia de corte Ω_c se determina imponiendo la frecuencia de corte ω_c en el dominio discreto. El valor de ω_c se elige arbitrariamente como $\pi/2$ rad, con lo que el filtro prototipo discreto tiene un ancho de banda de $f_s/4$ Hz. De esta forma $\Omega_c = 2$ rad/s (esto quedará claro en el siguiente punto).

2. *Pasaje al dominio discreto.* Al filtro analógico se le aplica la transformación bilineal (ecuación B.5). Con esta transformación la relación entre las frecuencias analógica y discreta es $\Omega = 2\tan(\omega/2)$. El filtro digital tiene N ceros en $Z = -1$ y mantiene el número de polos. Los polos se ubican en,

$$Z_k = \frac{2 + s_k}{2 - s_k}.$$

   La ganancia es,

$$G_d = \frac{G_c}{\prod_{k=0}^{N-1}(2 - s_k)}.$$

3. *Transformación en frecuencia.* Se transforma la función de transferencia del filtro pasabajos discreto mediante la transformación en frecuencia correspondiente según el tipo de filtro deseado (figura B.6). Es necesario determinar los ceros, los polos y la ganancia del filtro resultante.

[4]Para generalizar el procedimiento a otra clase de filtro, sólo es necesario modificar el primer paso partiendo del filtro analógico correspondiente.

Los polos y ceros se obtienen a partir de los polos y ceros del filtro pasabajos. Cada polo o cero z_k se determina como,

$$z_k \quad / \quad G(z_k^{-1}) = Z_k^{-1}$$

donde $G(z^{-1})$ es la transformación en frecuencia y Z_k es el polo o cero del filtro pasabajos. A continuación se detalla como se transforman los polos, los ceros y la ganancia para cada tipo de filtro.

Filtro Pasabajos.

Ceros: $z_k = -1, \quad k = 0, \ldots, N-1.$

Polos: $z_k = \frac{\alpha + Z_k}{1 + \alpha Z_k}, \quad k = 0, \ldots, N-1.$

Ganancia: $G_d' = G_d \frac{(1-\alpha)^N}{\prod_{k=0}^{N-1}(1+\alpha Z_k)}.$

Filtro Pasaaltos.

Ceros: $z_k = -1, \quad k = 0, \ldots, N-1.$

Polos: $z_k = \frac{Z_k - \alpha}{1 - \alpha Z_k}, \quad k = 0, \ldots, N-1.$

Ganancia: $G_d' = G_d \frac{(1+\alpha)^N}{\prod_{k=0}^{N-1}(1-\alpha Z_k)}.$

Filtro Pasabanda.

Ceros: $z_k = \pm 1, \quad k = 0, \ldots, N-1.$

Polos[5]: $z_k \,/\, G(z^{-1}) = Z_j^{-1}$

$$k = 0, \ldots, 2N-1$$

$$j = 0, \ldots, N-1.$$

Ganancia: $G_d' = G_d \frac{2^N}{\prod_{i=0}^{N-1}[(K+1)+(K-1)z_i]}.$

donde K es el parámetro de la transformación (ver figura B.6).

[5]La ecuación $G(z^{-1}) = Z_j^{-1}$ es en este caso un polinomio de segundo grado por lo que cada polo se mapea en un par de polos complejos conjugados. Por esta razón se deja expresado en forma genérica.

4. *Obtención de los coeficientes de recursión.* A partir de los ceros, polos y ganancia, se determinan los coeficientes de los polinomios numerador y denominador de la transferencia del filtro. Estos coeficientes son los usados en la recursión.

La rutina de diseño de filtros se implementó en C++ usando como referencia el código de Scilab.

B.2.6. Estabilidad

Como ya se mencionó, el número de polos de un filtro IIR está limitado en la práctica debido a errores numéricos. Esto se ilustra en el siguiente ejemplo. Consideremos un filtro Butterworth pasabajos de orden 6 y frecuencia de corte 0.02. Los coeficientes del numerador son del orden de 10^{-8} y los coeficientes del denominador son del orden de 10, como se aprecia en la figura B.7. Al trabajar con precisión simple, el error de redondeo es la diez millonésima parte del valor. Por lo tanto, el error en los coeficientes del denominador es 10^{-6}, cien veces mayor que el valor de los coeficientes del numerador. Este filtro no funciona ya que la contribución de la señal de entrada es despreciable frente a los valores de la salida previamente calculados. En estos casos, el comportamiento del filtro se degrada, pudiendo incluso hacerse inestable.

a_0	4.863988E-08	b_0	1.000000E+00
a_1	2.918393E-07	b_1	- 5.514535E+00
a_2	7.295981E-07	b_2	1.268911E+01
a_3	9.727975E-07	b_3	- 1.559364E+01
a_4	7.295981E-07	b_4	1.079330E+01
a_5	2.918393E-07	b_5	- 3.989359E+00
a_6	4.863988E-08	b_6	6.151231E-01

Figura B.7: Coeficientes de un filtro Butterworth de frecuencia de corte 0.02 y orden 6.

A medida que aumenta el número de polos del filtro, la diferencia en el orden de los coeficientes del numerador y del denominador se incrementa. Por lo tanto, un filtro IIR puede tener cierto número máximo de polos. Este número depende de la frecuencia de corte y del tipo de filtro, y se resume en la figura B.8.

Frecuencia de corte	Número máximo de polos
0.02	4
0.05	6
0.10	10
0.25	20
0.40	10
0.45	6
0.48	4

Figura B.8: Límite práctico aproximado del número de polos usando precisión simple.

Existen dos formas de aumentar el número de polos de un filtro IIR más allá de este valor máximo. Una de ellas es utilizar precisión doble, tanto en el cálculo de los coeficientes[6] como en el filtrado. La otra forma consiste en implementar el filtro en etapas de menor orden en cascada. Por ejemplo, un filtro de orden 6 puede llevarse a cabo en tres etapas de orden 2.

En la rutina de diseño de filtros implementada se optó por utilizar precisión doble como forma de aumentar el número máximo de polos alcanzable. Es importante señalar que el aumento de la precisión enlentece el proceso de filtrado.

[6] Incluyendo el valor de π.

Bibliografía

[1] E. Pollastri, *Processing Singing Voice for Music Retrieval*. PhD thesis, Università Degli Studi Di Milano, 2003.

[2] A. S. Durey, *Content-based access of musical databases*. PhD thesis, Georgia Institute of Technology, 2000.

[3] R. J. McNab, L. A. Smith, I. H. Witten, C. L. Henderson, S. J. Cunningham, "Towards the digital music library: Tune retrieval from acoustic input," *Proceedings of the first ACM Conference on Digital Libraries*, pp. 11–18, 1996.

[4] A. Ghias, J. Logan, D. Chamberlin, B. C. Smith, "Query by humming: Musical information retrieval in an audio database," *Proceedings of the first ACM Conference on Digital Libraries*, pp. 231–236, 1995.

[5] A. Uitdenbogered, J. Zobel, "Manipulation of music for melody matching," *Proceedings of ACM Multimedia '98*, pp. 235–240, 1998.

[6] M. Mongeau, D. Sankoff, *Comparison of musical sequences*, vol. 24. Kluwer Academic Publishers, 1990.

[7] F. Miyara, "La voz humana." FCEIA, Escuela de Ingeniería Electrónica, Universidad Nacional de Rosario, `http://www.eie.fceia.unr.edu.ar/~acustica/biblio/fonatori.pdf`.

[8] D. Gerhard, "Pitch-based acoustic feature analysis for the discrimination of speech and monofonic singing," *Journal of the Canadian Acoustical Assosiation*, pp. 152–153, 2002.

[9] T. Chilton, "Speech analisys," 1999. School of Electronic and Physical Sciences, University of Surrey.

[10] E. Gomez, *Melodic desciption of audio signals for music content processing*. PhD thesis, Universitat Pompeu Fabra, 2002.

[11] P. de la Cuadra, A. Master, C. Sapp, "Efficient pitch detection techniques for interactive music," *Proceedings of International Computer Music Conference*, 2001.

[12] C. Wendt, A. Petropulu, "Pitch determination and speech segmentation using the discrete wavelet transform." Electrical and Computer Engineering Departament, Drexel University.

[13] A. de Cheveignè, H. Kawahara, "Comparative evaluation of f0 estimation algorithms," *Eurospeech*, pp. 2451–2454, 2001.

[14] X. Sun, "A pitch determination algorithm based on subharmonic-to-harmonic ratio," *6th International Conference of Spoken Language Processing*, vol. 4, pp. 676–679, 2000.

[15] K. Chen, "Intelligent pitch detection," 2001. `http://www.isl.uiuc.edu/~kenchen/IVE/ipd/ipd.html`.

[16] A. de Cheveignè, H. Kawahara, "Yin, a fundamental frecuency estimator for speech and music," *Journal Acoustic Society of America*, vol. 111, pp. 1917–1930, 2002.

[17] S. Uppgard, "Implementation and analysis of pitch tracking algorithms," Master's thesis, Departament of Signals, Sensors and Systems, Kungliga Tekniska Högskolan, 2001.

[18] M. H. Hayes, *Statistical digital signal processing and modeling*. John Wiley & Sons, Inc., 1996.

[19] A. V. Oppenheim, R. W. Schafer, *Digital Signal Processing*. Prentice Hall, 1975.

[20] P. Boersma, D. Weenink, *Praat*. Versión 4.1.5, 2003. `http://www.praat.org/`.

[21] K. Sjölander, J. Beskow, *Wavesurfer*. Versión 1.5.3, 2003. `http://www.speech.kth.se/wavesurfer/`.

[22] F. Plante, G. F. Meyer, W. A. Ainsworth, *A pitch extraction reference database*. Keele University UK, 1995. `ftp://ftp.cs.keele.dc.uk/pub/pitch`.

[23] N. Henrich, *VOQUAL Singing Database 2*. KTH Stockholm, Sweden, 2002. `http://www.limsi.fr/WkG/VOQUAL/voicematerial.html`.

[24] N. Henrich, *VOQUAL Singing Database 1*. LIMSI CNTS/LAM, France, 2001. `http://www.limsi.fr/WkG/VOQUAL/voicematerial.html`.

[25] J. P. Bello, M. Sandler, "Phased based note onset detection for music signals," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2003.

[26] C. Duxbury, J. P. Bello, M.Davies, M. Sandler, "A combined phase and amplitude based approach to onset detection for audio segmentation," *Proceedings of the 4th European Workshop on Image Analysis for Multimedia Interactive Services*, 2003.

[27] S. Dixon, "Learning to detect onsets of acoustic piano tones," *Proceedings of the Workshop on Current Directions in Computer Music Research*, 2001.

[28] A. P. Klapuri, "Sound onset detection by applaing psichoacoustic knowldege," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1999.

[29] C. Duxbury, M. Snadler, M. Davies, "A hybrid approach to musical note onset detection," *Proceedings of the 5th International Conference on Digital Audio Effects*, 2002.

[30] W. A. Schloss, *On the Automatic Transcription of Percusive Music: From Acoustic Signal to High Level Analysis*. PhD thesis, Stanford University, CCRMA, 1985.

[31] J. L. Flanagan, R. M. Golden, "Phase vocoder," *The Bell System Technical Journal*, vol. 45, no. 8, pp. 1493–1509, 1966.

[32] X. Serra, "Musical sound modeling with sinusoids plus noise," in *Musical Signal Processing*, Swets & Zeitlinger, 1997.

[33] J. Pampin, "Ats: A system for sound analysis transformation and synthesis based on a sinusoidal plus critical-band noise model and psychoacoustics," *To be published in Proceedings of the 2004 Computer Music Conference*, 2004.

[34] C. Duxbury, J. P. Bello, M. Davies, M. Sandler, "Complex domain onset detection for musical signals," *Proceedings of the 6th Conference on Digital Audio Effects*, 2003.

[35] K. Jensen, D. Murphy, "Segmenting melodies into notes," *Proceedings of the Danisk Society for Automatic Recognition of Patterns*, 2001.

[36] S. A. Abdallah, M. D. Plumbley, "Probability as metadata: event detection in music using as a conditional density model," *4th International Symposium on Independent Component Analysis and Blind Signal Separation*, 2003.

[37] A. V. Oppenheim, R. W. Schafer, *Discrete-Time Signal Processing*. Prentice Hall, 1999.

[38] E. D. Scheirer, "Tempo and beat analysis of acoustic signals," *Journal of the Acoustic Society of America*, vol. 103, no. 1, pp. 588–601, 1998.

[39] A. P. Klapuri, "Automatic transcription of music," Master's thesis, Tampere Univerity of Technology, 1997.

[40] A. P. Klapuri, "Musical meter estimation and music transcription," *Cambridge Music Processing Colloquium*, 2003.

[41] C. Uhle, J. Herre, "Estimation of tempo, micro time and time signature from percussive music," *Proceedings of the 6th Conference on Digital Audio Effects*, 2003.

[42] W. Chai, "Melody retrieval on the web," Master's thesis, MIT, 2001.

[43] M. Goto, Y. Muroaka, "Real-time beat tracking for drumless audio signals: Chord change detection for musical decisions," *Proceedings of the International Joint Conference on Artificial Intelligence Workshop on Computational Auditory Scene Analysis*, pp. 135–144, 1997.

[44] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *Journal of New Music Research*, vol. 30, no. 1, 2001.

[45] E. Pollastri, G. Haus, "An audio front end for query-by-humming systems," *Proceedings of International Symposium of Music Information Retrieval*, 2001.

[46] T. Viitaniemi, A. Klapuri, A. Eronen, "A probabilistic model for the estimation of single-voice melodies," *Finnish Signal Processing Symposium, Tampere University of Technology*, 2003.

[47] L. P. Clarisse, J. P. Martens, M. Lesaffre, B. De Baets, H. De Meyer, M. Leman, "An auditory model based transcriber of singing sequences," *Proceedings of International Symposium of Music Information Retrieval*, pp. 116–123, 2002.

[48] G. Raskinis, *Solo Explorer*. Versión 1.007, 2002. `http://www.recognisoft.com/`.

[49] M. Lerman, J. P. Martens, B. De Baets, H. De Meyer, *MAMI*. Versión 1.1.5, 2002. `http://www.ipem.ugent.be/MAMI`.

[50] A. Uitdenborgerd, J. Zobel, "Melodic matching techniques for large music databases," *Proceedings of the ACM Multimedia*, pp. 57–66, 1999.

[51] K. Lemström, *String Matching Techinques for Music Retrieval*. PhD thesis, Department of Computer Science, University of Helsinki, 2000.

[52] R. B. Dannenberg, N. Hu, "A comparison of melodic database retrieval techniques using sung queries," *Joint Conference on Digital Libraries, New York: ACM Press*, pp. 301–307, 2002.

[53] D. Shasha, Y. Zhu, "Warping indexes with envelope transforms for query by humming," *Proceedings of the 2003 ACM SIGMOD Conference on Management of Data*, pp. 181–192, 2003.

[54] T. Sonoda, Y. Muraoka, "A www-based melody retrieval system - an indexing method for a large database," *Proceedings of the ICMC2000*, 2000.

[55] D. Levitin, *Memory for Musical Attributes*. Music, Cognition, and Computerized Sound, MIT Press, 1999.

[56] B. Pardo, W. Birmingham, "Encoding timing information for musical query matching," *Third International Conference on Music Information Retrieval*, pp. 267–268, 2002.

[57] W. J. Dowling, "Scale and countour: Two components of a theory of memory for melodies," *Psychological Review*, pp. 341–354, 1978.

[58] T. Bröndsted, S. Augustensen, B. Fisker, C. Hansen. J. Klitgaard, L. W. Nielsen, T Rasmussen, "A system for recongition of hummed tunes," *Proceedings of the COST G-6 Conference on Digital Audio Effects*, 2001.

[59] A. Lindsay, "Using countor as a mid-level representation of melody," Master's thesis, Massachusetts Institute of Technology, 1996.

[60] D. Q. Goldin, P. C. Kanellakis, "On similarity queries for time-series data: Constraint specification and implementation," *Proceedings of the 1997 ACM SIGMOD Conference on Management of Data*, pp. 13–25, 1997.

[61] INRIA-ENPC, *Scilab*. Versión 2.7, 2003. `http://scilabsoft.inria.fr`.

[62] B. Schottstaedt, *sndlib*. Versión 17.0, 2003. `http://ccrma.stanford.edu/software/snd/sndlib/`.

[63] C. S. Sapp, *Midiio*. Versión 1.1.1, 2003. `http://midiio.sapp.org`.

[64] J. Laako, *libDSP*. Versión 4.6.0, 2003. `http://libdsp.sourceforge.net`.

[65] Bealtes Midi Albums. `http://www.geocities.com/SunsetStrip/Studio/7779/`.

[66] Bealtes Midi And Video Heaven. `http://beatles.zde.cz/`.

[67] The Beatles Midi. `http://www.fortunecity.com/tinpan/morrissey/612/b-midi.htm`.

[68] C. Cannam, R. Brown, G. Laurent, *Rosegarden*. Versión 0.9.8, 2004. `http://www.rosegardenmusic.com/`.

[69] GIMP Toolkit, *GTK+*. Versión 2.0, 2003. `http://www.gtk.org`.

[70] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1999. `http://www.DSPguide.com`.