FACULTAD DE Matemática-Física-Computación

DEPARTAMENTO DE COMPUTACION

TRABAJO DE DIPLOMA

"Algoritmos y programas para el cálculo de parámetros relacionados con la prosodia"

Autor: Lester Núñez Pérez

Tutora: Dra. María Esperanza Hernández-Díaz Huici

Consultante: Dr. Carlos Ariel Ferrer Riesgo

Santa Clara

2007

"Año 49 de la Revolución"

Universidad Central “Marta Abreu” de Las Villas

Facultad de Matemática-Física-Computación

TRABAJO DE DIPLOMA

“Algoritmos y programas para el cálculo de parámetros relacionados con la prosodia”

Autor: Lester Núñez Pérez

lnp@uclv.edu.cu

Tutora: Dra. María Esperanza Hernández-Díaz Huici

Profesora Auxiliar. CEETI.

Facultad de Ingeniería Eléctrica. UCLV.

E-mail: Mariae@uclv.edu.cu

Consultante: Dr. Carlos Ariel Ferrer Riesgo

Profesor Auxiliar. CEETI

Facultad de Ingeniería Eléctrica. UCLV.

E-mail: cferrer@fie.uclv.edu.cu

Santa Clara

2007

“Año 49 de la Revolución "

Hago constar que el presente trabajo fue realizado en la Universidad Central Marta Abreu de Las Villas como parte de la culminación de los estudios de la especialidad de Ciencias de la Computación, autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

_____

Firma del autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del Tutor

Firma del Jefe del Seminario de Programación

"La ciencia nace y crece en la medida
en que el hombre comience a cuestionarse
el mundo en que vive".

Aristóteles

A mi madre, mi abuelita, que hicieron posible la realización de mi carrera,
Y por ser ejemplos de madres.
A mi hermanita Claudia a quien quiero mucho, al igual que a mis tías,
Especialmente a Zonita que tanto me cuidó y aconsejó.
A mi tío por su ejemplo de persona, de padre, de profesional.
A mi papá, primos y demás familiares.
Y a mi novia Yilian por mantenerse a mi lado todo este tiempo,
Por confiar en mí y darme fuerzas en todo momento.
A todos va dedicada esta tesis.

A nuestro padre celestial, a mi tutora, cotutor que sin su ayuda no hubiese podido hacer este trabajo.

A todos los profesores que han influido grandemente en mi formación Profesional, aYoel, Roberto, Dennis y a todos

Los que hicieron posible la realización de esta investigación

Les lleguen a todos

Mi más grato agradecimiento

De todo corazón.

Resumen

En esta investigación se ha realizado la implementación de una serie de algoritmos que determinen de forma automática la intensidad, la entonación y el ritmo de la señal de voz, que con todos ellos unidos se puede conformar un software que analice dichos parámetros de la prosodia. También se ha implementado otro algoritmo que segmenta la señal en sílabas. Los algoritmos anteriores se desarrollaron en el lenguaje de programación Matlab por las facilidades que le brinda al programador. La aplicación tiene como propósito que los especialistas en voces puedan estudiar los parámetros que caracterizan el habla y utilizarlos para detectar determinadas patologías en la misma. El Capítulo 1 describe el marco teórico relativo al proceso de producción del habla humana con todos los sistemas y órganos que intervienen en él. En el Capítulo 2 se realiza un análisis de la arquitectura del sistema implementado. Finalmente se muestra una descripción detallada a nivel de usuario de cómo funciona el software.

Abstract

In this investigation was carried out the implementation of a series of algorithms that determine automatically the intensity, the intonation and the rhythm of the signal voice. All of them integrate a software application that analyzes these parameters of prosody. The software implements another algorithm that segments the speech signal in syllables. The previous algorithms were implemented in the programming language *Matlab* for the facilities that it offers to the programmer. The application has as purpose to allow the specialists study the parameters that characterize the speech and to use them to detect certain pathologies in sample. Chapter 1 presents the theoretical frame of the research, the process of production of the human speech with all the systems and organs involved in it. In Chapter 2 is carried out an analysis of the architecture of the implemented system. Finally a detailed description is shown at user's level of how the software works.

Índice

Introducción

Contribuir a realizar un avance científico amplía la visión personal de quien se dedica a esa tarea, emprendiendo así un itinerario personal de búsqueda para el pensamiento y la acción a través de un método (camino en griego), quedando los logros a disposición de cualquier persona que desee consultarlos.

Los avances científicos tienen como objetivo mejorar el conocimiento colectivo de la realidad en cada campo específico.

Actualmente en la sociedad mundial se han presentado diversos problemas en la comunicación oral de personas debido a discapacidades en el habla, por ello se ha manifestado una tendencia mundial a analizar las diferentes causas de tal problema y plantear las soluciones que los especialistas orienten en dependencia de la discapacidad.

Desde la década del ’80 se reportan en la literatura científica parámetros que miden alteraciones en los sonidos vocálicos y en cierto modo se correlacionan con las características que los especialistas evalúan de forma subjetiva. El reto consiste en lograr parámetros que se identifiquen unívocamente con características de la voz con valor para el diagnóstico y que a la vez se correlacionen con las percepciones de los especialistas, todo ello con un costo computacional adecuado y una argumentación sólida sobre la física de la producción de estos sonidos.

En este sentido el camino emprendido al iniciar esta investigación ha conducido a reflexionar sobre la voz como un sistema muy complejo, que implica aspectos biológicos, psicológicos, ambientales, culturales y sociales, y que es pilar de la comunicación humana.

Es por esto que la voz ofrece grandes y complejas vías de comprensión pudiendo ser evaluada subjetiva y objetivamente. La herramienta que frecuentemente se usa para analizarla subjetivamente es su audición la cual no puede ser cuantificada, siendo necesario valerse para esto de un laboratorio de voz, y especialistas para realizar la evaluación objetiva de la misma [1, 2].

Para el análisis de la inteligibilidad y el buen entendimiento de la voz ha surgido una rama de la lingüística llamada *prosodia* que estudia la intensidad, la entonación y el ritmo de la misma.

Si bien se registra un gran número de parámetros asociados a la vocalización, no son muchas las referencias que traten sobre el proceso de articulación o la prosodia y son estos los que más influyen en la claridad de la comprensión del mensaje hablado.

En la búsqueda de un índice de la inteligibilidad del habla, se nota la falta de un parámetro que represente la contribución de la prosodia, el cual se ha reportado como de gran relevancia en la comprensión del mensaje verbalmente transmitido.

¿Cómo contribuir a la obtención de un índice de la prosodia?

Para contribuir a la solución del problema científico antes plateado, se formuló la *hipótesis general de investigación* siguiente:

Con la aplicación de técnicas para el análisis de la voz, en especial, los aspectos físicos relacionados con la prosodia tales como la energía y la frecuencia fundamental, es posible llegar a determinar la presencia de patologías en pacientes discapacitados en el habla.

En conformidad con la hipótesis de investigación identificada, el *objetivo general* de la investigación consiste en implementar un conjunto de algoritmos que determinen de forma automática la intensidad, la entonación, el ritmo y las sílabas de una señal de voz en el lenguaje de programación Matlab con fines médicos y lingüísticos.

Este objetivo general fue desglosado en los *objetivos específicos* siguientes:

1. Realizar un estudio de los métodos de estimación prosódicos que más se están usando en la actualidad.
2. Implementar un algoritmo que determine de forma automática la intensidad de una señal de voz grabada.
3. Implementar un algoritmo que determine de forma automática la entonación de una señal de voz grabada.
4. Implementar un algoritmo que determine, de forma automática, la ritmicidad de una señal de voz grabada.
5. Implementar un algoritmo que segmente de forma automática la señal de voz en sílabas.
6. Analizar los resultados de los algoritmos implementados.

Las *Preguntas de investigación* que se han propuesto son las siguientes:

1. ¿Qué métodos de estimación de parámetros prosódicos son los más usados en la actualidad?
2. ¿Qué parámetro físico de la señal de voz influyen en la entonación que percibimos?
3. ¿Qué parámetro físico de la señal de voz determinan la intensidad de dicha emisión?
4. ¿Qué parámetro o parámetros físicos de la señal de voz brinda la información suficiente para segmentar la señal en sílabas con una buena precisión?

Con este proyecto se pretende realizar una herramienta para el análisis de voz que ayude a los especialistas a determinar posibles patologías en las mismas y ofrecer una respuesta a la demanda de software de análisis de voz del departamento de *Procesamiento de Voz* del *Centro de Estudios de Electrónica y Tecnologías de la Información* (*CEETI*) con soluciones económicamente factibles.

La implementación de estos algoritmos permitirá a los especialistas e investigadores realizar estudios y análisis comparativos, que contribuirán a una mejor comprensión del mecanismo de producción y percepción del habla.

El *impacto posible* que se pretende con este estudio es que con el desarrollo de estos algoritmos se contribuya a mejorar el trabajo de los especialistas en voz y por ende la atención diferenciada que nuestro país ofrece a las personas de la tercera edad y a los discapacitados.

Los resultados de la investigación poseen una *aplicación práctica* y *teórica* de gran trascendencia para todos los especialistas de sistemas de rehabilitación.

El proyecto es *viable* dado que las herramientas, literatura y otros recursos necesarios están disponibles en el *CEETI*.

La tarea planteada se enmarca en las investigaciones que lleva a cabo el Laboratorio de Procesamiento de Voz del CEETI con la finalidad de encontrar medidas que apoyen el diagnóstico y evaluación de los pacientes con dificultades en la comunicación oral. Una vez que la población cubana va camino al envejecimiento y

la atención primaria está bien establecida, los retos se dirigen hacia lograr mejor calidad de vida para las personas con algunas limitaciones que impiden su exitoso desempeño como seres sociales. Las dificultades en el habla y la vocalización requieren de un arduo trabajo de rehabilitación pero hasta ahora las acciones no pueden ser reportadas y cuantificadas por ser, mayormente dependiente de las habilidades de los especialistas y sus opiniones.

A nivel mundial están apareciendo muchos trabajos orientados a documentar estas actividades de forma científica.

Para la presentación de esta investigación, este *Trabajo de Diploma* se estructuró de la siguiente forma. Un Capítulo 1 que contiene el marco teórico-referencial acerca de los parámetros relacionados con la prosodia, y los algoritmos más usados para determinar dichos parámetros. Un Capítulo 2 en el que se describe la arquitectura del sistema a través de las fases del *Proceso de Desarrollo de Software*. Un Capítulo 3 donde se describe a nivel de usuario el software que soporta los algoritmos implementados y realiza el análisis de los resultados de los mismos.

CAPITULO 1. La prosodia y sus parámetros. Algoritmos que determinan dichos parámetros

El análisis de la prosodia es uno de los elementos del análisis de voz más importante pues de ella principalmente es de quien depende el buen entendimiento e inteligibilidad de la producción del habla. Es una de las materias en las que más se está investigando en la actualidad debido a los problemas que están presentando las personas con la voz, de lo cual no se tienen muchos conocimientos por parte de los especialistas, y existen muchos problemas de exactitud y precisión acorde a esta temática.

Por eso en este capítulo se ha propuesto realizar un estudio acerca de la producción del habla humana, de los aspectos teóricos relacionados con la prosodia y de los algoritmos más usados para determinar la intensidad, la entonación y el segmentado de la señal de voz en sílabas en la actualidad.

1.1- Producción de la voz humana

La voz es una onda de presión acústica. El proceso de producción natural de la voz pasa por varias etapas que incluyen: la concepción del mensaje en el cerebro, la transmisión de los impulsos nerviosos necesarios a cada uno de los órganos y músculos involucrados, y la ejecución en los mismos de los movimientos correspondientes. En esta última, radica el basamento físico de generación de la voz.

1.1.1- *Fisiología de la producción del habla*

Tradicionalmente necesitamos de tres grupos o sistemas de órganos para que, coordinadamente podamos producir lenguaje hablado. Estos grupos son [3]:

- *Sistema Respiratorio*: Comprende los pulmones y los músculos por los cuales se comprimen o dilatan los tubos bronquiales y la tráquea. Su función principal es la respiración, asegurando el abastecimiento de oxígeno a la sangre.
- *Sistema Fonatorio*: Está compuesto por la laringe, órgano responsable de la producción básica del sonido, que actúa como válvula de cierre de la vía que conecta hacia los pulmones, evitando que pasen los alimentos a través de la tráquea.

- *Sistema Articulatorio*: Está formado por las estructuras oro-faciales las que se subdividen en nasales y orales, las cuales permiten el ingreso y egreso del aire hacia y desde las cavidades pulmonares(respiración), y en el caso de las estructuras orales, la ingesta de alimentos.

Mediante estos sistemas se puede definir un modelo acústico de generación de voz humana, el cual está constituido por las etapas que se explican en el *Anexo 1*.
El habla como señal acústica se produce a partir de las ondas de presión que salen de la boca y las fosas nasales de una persona. El proceso comienza con la generación de la energía suficiente (flujo de aire) en los pulmones, la modificación de ese flujo de aire en las cuerdas vocales, y su posterior perturbación por algunas constricciones y configuraciones de los órganos superiores. Así, en el proceso fonador intervienen distintos órganos a lo largo del tracto vocal como se muestra en [4].

El conjunto de órganos que intervienen en la fonación (ver figura 1.1.1 en Anexo 2) pueden dividirse en tres grupos bastante bien delimitados [5-7]:

- Cavidades infraglóticas (sistema sub-glotal) u órgano respiratorio.
- Cavidad laríngea u órgano fonador.
- Cavidades supraglóticas.

1.1.1.1- Cavidades infraglóticas

Las cavidades infraglóticas constan de los órganos propios de la respiración (pulmones, bronquios y tráquea), que son la fuente de energía para todo el proceso de producción de voz [4].
En el proceso de inspiración, los pulmones toman aire, bajando el diafragma y agrandando la cavidad torácica. En el momento de la fonación, la espiración, provocada por la contracción de los músculos intercostales y del diafragma, aporta la energía necesaria para generar la onda de presión acústica que atravesará los órganos fonadores superiores [4, 7].

1.1.1.2- Cavidad laríngea

La cavidad laríngea es la responsable de modificar el flujo de aire generado por los pulmones y convertirlo (o no) , en una señal susceptible de excitar adecuadamente las posibles configuraciones de las cavidades supraglóticas [4].

El último cartílago de la tráquea, el cricoides, forma la base de la laringe, cuyo principal órgano son las cuerdas vocales que son dos pares de repliegues compuestos de ligamentos y músculos. El par inferior son las llamadas cuerdas vocales verdaderas, que pueden juntarse o separarse mediante la acción de los músculos crico-aritenoides lateral y posterior, y que están protegidas en su parte anterior por el cartílago tiroides, el más importante de la laringe, abierto por su parte posterior. Finalmente, la parte superior de la laringe está unida al hueso hioides [4].

En la figura 1.1.1.2 (ver Anexo 2) se muestra una vista longitudinal simplificada de la zona en la que se encuentran las cuerdas vocales, en sus posiciones extremas: abiertas y cerradas. A la apertura que queda entre las cuerdas vocales se les denomina glotis. La cavidad laríngea está terminada por la epiglotis, un cartílago en forma de cuchara, que permite cerrar la apertura de la laringe en el acto de la deglución[1] [4].

1.1.1.3- Cavidades supraglóticas

Las cavidades supraglóticas están constituidas por la faringe, la cavidad bucal y la cavidad nasal. Su misión fundamental de cara a la fonación es perturbar adecuadamente el flujo de aire procedente de la laringe, para dar lugar finalmente a la señal acústica generada a la salida de la nariz y la boca. La faringe es una cavidad en forma tubular que une la laringe con las cavidades bucal y nasal, y que suele dividirse en tres partes: faringe laríngea, faringe bucal (boca) y faringe nasal, las dos últimas separadas por el velo del paladar. El volumen de la faringe laríngea puede ser modificado por los movimientos de la laringe, la lengua y la epiglotis mientras que el volumen de la faringe bucal se modifica por el movimiento de la lengua [4, 8].

7

[1] Deglución: Es el proceso que consiste en determinar la posición de la laringe en el proceso de producción de habla.

La faringe nasal y las restantes cavidades nasales forman, desde el punto de vista de su acción sobre el flujo de aire procedente de la faringe, un resonador que puede o no conectarse al resonador bucal mediante la acción del velo del paladar. Según el resonador nasal esté o no conectado, el sonido será nasal u oral, respectivamente. Si hacemos una descripción de la cavidad bucal, podemos señalar las siguientes partes [1, 4]:

- Los labios en el extremo.
- Los dientes.
- La zona alveolar, entre los dientes y el paladar duro.
- El paladar, en el que a su vez, y de forma simplificada, podemos distinguir el paladar duro y el paladar blando o velo.

La raíz de la lengua forma la pared frontal de la faringe laríngea, y sus movimientos le permiten modificar la sección de la cavidad bucal (movimiento vertical), adelantar o retrasar su posición frente a la de reposo (movimiento horizontal), así como poner en contacto su ápice o la parte trasera con alguna zona del paladar [3].

El movimiento de los labios también interviene en la articulación, pudiendo ser de apertura o cierre y de protuberancia, alargando en este último caso la cavidad bucal [8].

1.1.2- Fonemas: vocales y consonantes.

El ***fonema*** es la mínima unidad lingüística capaz de producir cambios de significados. Estos fonemas pueden ser agrupados en dos grandes entidades: *Fonemas Vocálicos* y *Fonemas Consonánticos*. Ambos tipos de fonemas pueden clasificarse según el estado vibratorio de las cuerdas vocales, en fonemas fonados (o sonoros) y áfonos (o sordos)[9].

Desde el punto de vista mecano acústico, y además considerando que la voz es fisiológicamente normal, las **vocales** son los sonidos emitidos por la sola vibración de las cuerdas vocales sin ningún obstáculo entre la laringe y las aberturas oral y nasal. Dicha vibración se genera por el principio del oscilador de relajación, donde interviene una fuente de energía constante en la forma de un flujo de aire

proveniente de los pulmones. Son siempre sonidos de carácter sonoro y por consiguiente de espectro discreto [4].

Las vocales se clasifican según la apertura del tracto vocal (vocales abiertas y cerradas) y según el grado de elevación del dorso de la lengua (vocales anteriores, centrales y posteriores), figura 1.1.2 (ver Anexo 3)[10].

Los Fonemas Vocálicos son siempre fonados, a diferencia de los consonánticos que pueden ser o no fonados. Sin embargo, la principal diferencia entre los fonemas vocálicos y consonánticos es el tamaño de las constricciones asociadas a las articulaciones en el tracto, siendo en las consonantes estas obstrucciones mucho más estrechas [9].

Los *fonemas fonados* se producen por la acción de las cuerdas vocales, que obturan el paso del flujo de aire durante breves instantes de tiempos. Éste tren de pulsos es modificado por los órganos de articulación que vienen posteriormente añadiendo una cierta codificación. Esta codificación consiste en el resalte de determinadas frecuencias contenidas en el pulso glotal original o en su eliminación. Por lo tanto el sonido así emitido tiene una forma relativamente periódica [4].

Los *fonemas áfonos* se producen si la acción de las cuerdas vocales se ve alterada (posición de respiración). El flujo de aire procedente de los pulmones se vuelve más rápido produciendo fricciones y turbulencias. Esto se traduce en múltiples vibraciones en aquellos puntos en que por producirse un estrechamiento, la velocidad del flujo de aire alcanza sus máximos. Ahora bien, si el punto de estrechamiento o articulación se encuentra próximo al exterior (labios, dientes), el resto de la estructura influye escasamente en el sonido emitido. En caso contrario, zona velar o medial, la posición de los restantes órganos puede modular bastante el resultado [8]. Su forma de onda presenta un escaso carácter repetitivo.

Por otro lado las *consonantes* se clasifican según el *punto de articulación* (lugar de contacto de los articuladores que participan en la producción de un fonema específico), *modo de articulación* (forma como se interrumpe el flujo aéreo espiratorio), *función de las cuerdas vocales* (presencia o ausencia de estas en el momento de producir un fonema), y *posición del velo del paladar* (capacidad de controlar el paso de aire hacia la cavidad nasal u oral)[10].

1.1.2.1- Punto de articulación

Son los puntos de estrechamiento en las cavidades supraglóticas, antes mencionadas, para los sonidos sordos y sonoros, además son controlados por los órganos móviles.

Una lista de posibles puntos de articulación puede ser [11]:

- ***Bilabiales:*** oposición de ambos labios.
- ***Labiodentales:*** oposición de los dientes superiores con el labio inferior.
- ***Linguodentales:*** oposición de la punta de la lengua con los dientes superiores.
- ***Alveolares:*** oposición de la punta de la lengua con la región alveolar.
- ***Palatales:*** oposición de la lengua con el paladar duro.
- ***Velares:*** oposición de la parte posterior de la lengua con el paladar blando.
- ***Glotales:*** articulación en la propia glotis.

1.1.2.2- Modo de articulación

El modo de articulación define la forma concreta en que la articulación puede tener lugar. Existen dos posibilidades:

- Modo espirado.
- Modo no espirado.

El **modo espirado** se produce bajo un flujo de aire proveniente de los pulmones y es el modo habitual en la mayoría de los lenguajes [4, 12].

El **modo no espirado** se produce por compresión del aire en la cavidad bucal, que sufre dos cierres, uno de los cuales cede bruscamente con entrada o salida de flujo aéreo [13].

Dentro del modo espirado y dependiendo del tipo de aproximación [4] realizada por los órganos articulatorios podemos tener dos submodalidades:

- Consonantes.
- Sonantes.

I. **Consonantes**

Corresponden al máximo cierre del tracto vocal en algún punto concreto. Pueden ser oclusivas y fricativas [11, 14].

- ***Oclusivas:*** Se produce una obstrucción total al paso del flujo de aire, pero solamente durante breves instantes, dando lugar a una explosión posterior a la apertura del tracto vocal. Su duración por lo general es de entre 10 y 30 **ms**.
- ***Fricativas:*** Se produce una obstrucción incompleta del flujo, que busca las pequeñas hendeduras residuales que puedan existir, aumentando enormemente la velocidad del fluido en la zona, incluso, puede tener una duración superior a los 100 **ms**.

II. **Sonantes**

Son aquellas articulaciones en las cuales no aparece fricción debido a algún tipo de estrechamiento del tracto bocal, y solo se realiza una modificación por cierre parcial o derivación. Pueden ser: Nasales, Laterales, Vibrantes y Aproximantes [4, 8].

- ***Nasales:*** La cavidad nasal se abre al paso del flujo de aire, produciendo un cierre total o casi total en algún punto de articulación de la cavidad oral.
- ***Laterales:*** Se basan en una obstrucción incompleta alrededor del tercio anterior de la lengua, bloqueando de forma parcial o total al flujo glotal, pero los bordes laterales de la lengua permiten el paso por el espacio que queda a ambos lados de la lengua.
- ***Africados:*** Se define como una oclusión que viene seguida por fricación.
- ***Vibrantes:*** Variante de las líquidas, en las que el ápice realiza bruscas aperturas y cierres de su contacto con la zona

alveolar o prepalatal, junto con la oclusión total o parcial del flujo lateral por medio de las zonas laterales de la lengua.

- ***Aproximantes:*** Son sonidos fuertemente relacionados con la evolución de un sonido a otro en la articulación de un diptongo. En tal caso, uno de los sonidos de tipo vocálico se consonantiza por cierre excesivo. Para su articulación requieren siempre de la presencia de una vocal.

En la figura 1.1.2.2 (ver Anexo 3) se indican las consonantes clasificadas según el lugar y el modo de articulación, la sonoridad y la oro-nasalidad. En algunos casos una misma consonante aparece en dos categorías diferentes, correspondiente a las diferencias observadas [8, 15].

1.1.2.3- Función de las cuerdas vocales

Se asocia a la idea de si hay presencia o ausencia de las cuerdas vocales al momento de producir un fonema [16].

1.1.2.4- Posición del velo del paladar

Está relacionado con la capacidad de controlar el paso de aire hacia la cavidad nasal u oral.

Un ejemplo de un fonema fricativo /s/, donde el flujo de aire a través de la constricción es dirigida en contra de los incisivos bajos, y la fuente principal del ruido de la turbulencia está localizada alrededor de los dientes [16].

1.1.3- Prosodia como elemento de la gramática del lenguaje

La Gramática es el estudio de los elementos de la lengua y sus combinaciones que aplicada a la oralidad, estudia el sonido que pretende ser lingüístico: la organización del hilo fónico a través de un conjunto de mecanismos de naturaleza prosódica o suprasegmental. Sin estos mecanismos sería muy difícil hacer de ese continuo fónico algo inteligible. Cuando se hacen análisis gramaticales se parte, generalmente, de la escritura y de todo aquello que podemos ver, dejándose de lado la manera como se

organiza el sonido, para que sea perceptible la estructura profunda de la lengua. Debido a esta razón se pretende dedicarle algún espacio a los elementos prosódicos básicos, que pudiéramos considerar como los principios elementales de la organización de la lengua hablada [7].

La **prosodia** es una rama de la lingüística que analiza y representa formalmente aquellos elementos suprasegmentales de la expresión oral, tales como la **entonación**, la **intensidad** y el **ritmo** que son los que organizan el hilo del sonido que percibimos. Si el sonido no estuviera agrupado de manera significativa, no habría comunicación lingüística; se oirían gritos o murmullos, a lo sumo.

El oyente recibe, cuando se comunica lingüísticamente, segmentos sonoros relativos a las unidades de información que le envía el hablante. Con el sonido, va una serie de informaciones de naturaleza prosódica, no solamente con información referencial sino también con información dialectal, sociolingüística e inclusive, emotiva.

Ahora bien, ¿cuál sería la sustancia de la prosodia? Aquellos parámetros físicos que la conforman: la **frecuencia fundamental (F0)** o **entonación**, la **duración**, la **intensidad** y el **ritmo**. Todos ellos contribuyen a la percepción de la prosodia de modo que aún cuando la F0 descienda levemente, una elevación en la intensidad se percibe como un tono levemente ascendente. Pertenecientes a la sustancia de la prosodia serían también otros elementos derivados de la vibración de las pulsiones laríngeas: la pausa, el acento y la sílaba [5].

La **sílaba** es la unidad rítmica pulsional alrededor de un núcleo silábico [5].

El **acento** es la prominencia acústica de una sílaba [5].

Las **pausas**, por su parte, son las interrupciones o detenciones de la información que hacemos cuando hablamos o leemos en voz alta. Delimitan los **grupos fónicos**, o sea aquellas porciones del discurso comprendidas entre dos pausas. Los **grupos fónicos** no coinciden necesariamente con los **grupos prosódicos**, que, además de estar delimitados por pausas, pueden estarlo por las inflexiones de la F0. Estos **grupos prosódicos** se denominan también **unidades entonativas** o **unidades de entonación**.

La **entonación** es básicamente la evolución de la F0, mientras que el ritmo incluye tanto las duraciones de cada uno de los signos de síntesis como la localización y

duración de las pausas. Representa expresión a nivel de oraciones, siendo además mucho más clara la variación en la F0 [3]. Integra las palabras en unidades gramaticales. Es el recurso más común y elemental del enunciado de ahí que *Quilis* dijera: "*puede haber enunciados sin forma gramatical, pero sin entonación, no*". Entonces en dependencia de como se emplee la entonación, esta juega una función u otra, entre las que se encuentran:

- ***Sociolingüística y psicológica*** que refleja el origen geográfico del hablante, el grupo socioeconómico al que pertenece e incluso, características psicológicas [6].
- ***Expresiva*** permite conocer el estado anímico y emocional del hablante, en una especie de entonación secundaria.

Los patrones entonativos tienen carácter suprasegmental, es decir, que afectan globalmente a todo un grupo entonativo en lugar de hacerlo particularmente sobre cada unidad [3].

La prosodia evalúa las variaciones del **tono** de los fonemas sonoros en base al seguimiento de su frecuencia fundamental [17].

La **acentuación** es un rasgo que permite poner en relieve un fonema para diferenciarlo de otras unidades del mismo nivel dentro de un nivel morfológico (palabra).

Hasta ahora sólo hemos visto algunos elementos que constituyen la sustancia de la prosodia pero no hemos hablado de cómo todo ello contribuye a darle sentido al habla, es decir, no hemos abordado acerca de la forma de la prosodia.

La *forma de la prosodia* es todo aquello que otorga al sonido valor lingüístico, contrastivo y por ende confiere un sentido que el hablante percibe [6], lo cual evidencia que la prosodia tiene varias funcionalidades que explicaremos a continuación.

1.1.3.1 Funciones de la prosodia

Las funciones de la prosodia son las siguientes:

- ***Cohesiva o integradora:*** ya que divide el hilo fónico en parcelas de modo que el oyente pueda percibirlo como un oleaje ya que se presenta en olas sucesivas en unidades empaquetadas prosódicamente [6].
- ***Delimitadora o demarcativa:*** que consiste en la segmentación del discurso en unidades menores relacionadas con su estructura profunda que opera generalmente en sintagmas, es decir en los grupos gramaticales y la llevan a cabo las pausas y los cambios de la F0 [6].
- ***Pragmática del lenguaje:*** en el nivel del enunciado ya que a través de la prominencia tonal el hablante decide lo que va a tratar como información nueva y lo que va a considerar como información dada [6].
- ***Progresión de la información:*** debido a que las unidades de entonación se corresponden a grosso modo con las unidades de información pues esta no se ofrece de una sola vez sino en oleajes sucesivos. Cada unidad de información se basa en la unidad anterior y añade un nuevo elemento [6].

Otro elemento característico del habla es el **ritmo**, íntimamente ligado con el tiempo [1].

El ritmo vital está íntimamente ligado a la producción del lenguaje, el aparato fonador tiene una doble función: garantizarnos el oxígeno y la comunicación. *Cicerón* lo concibe como: “un medio lingüístico para conseguir un buen estilo”. El ritmo realza tanto lo conceptual, como la feliz elección del léxico y la formación del texto. El ritmo está constituido básicamente por la repetición de un patrón y es esencial para la percepción de los acontecimientos que ocurren en el tiempo.

Según [11] cada lengua tiene tendencias rítmicas propias, que definen su fisonomía particular. La tendencia de la lengua castellana es de construir unidades de cinco a diez sílabas, y entre ellas, las más frecuentes serían las de siete a ocho.

De este modo, la prosodia conlleva una gran cantidad de información no sólo lingüística sino también paralingüística. Es esta heterogeneidad de la prosodia la que, precisamente, permite matizar y enriquecer pragmáticamente el discurso. En este sentido, ***Quilis*** considera que la entonación es: "*el vehículo lingüístico ideal para*

transmitir las más diversas informaciones, que, aunque en el proceso de la comunicación vayan tremendamente mezcladas, el oyente descodifica automáticamente, y sabe si su interlocutor pregunta o afirma, es de Chile o de España, está enfadado o contento, pertenece a un estrato social o a otro, etc".

O bien, como expresa Navarro Tomás:

"Las inflexiones melódicas de la palabra sirven juntamente a la declaración del pensamiento, a la expresión de los movimientos del ánimo y a la manifestación de hábitos y maneras de decir de carácter local. Se puede considerar la entonación desde el punto de vista ideológico, emocional o idiomático, si bien no es nada fácil delimitar, sobre el fondo común en que estos hechos se producen, la proporción y medida que a cada uno de dichos aspectos corresponde".

La prosodia constituye, entonces, la infraestructura rítmica de la lengua hablada, su organización en el tiempo, y contribuye a facilitarle al hablante la retención de ciertos segmentos en la memoria [10]. Podemos ver, que la prosodia está formada por una serie de parámetros que el hablante no percibe discriminadamente, sino como un todo. Ese todo le confiere también una totalidad de significados, que se perciben como un conjunto, pero que podemos discriminar en el análisis.

1.2- Métodos de estimación del Pitch o de Frecuencia Fundamental (F0)

La información prosódica, es decir, la velocidad de entonación, se encuentra fuertemente influenciada por la frecuencia fundamental (f0) de vibración de las cuerdas vocales, cuyo inverso a su vez se conoce como período fundamental T_0. De forma general, la definición de periodicidad está dada en intervalos de análisis infinitos, sin embargo, en la práctica, su estimación se realiza sobre intervalos finitos de manera que permitan cubrir varios períodos del Pitch, o instantáneamente a partir de la diferencia entre dos momentos consecutivos de cierre glótico.

Los métodos de cálculo del Pitch, a partir de las técnicas estadísticas de estimación utilizadas, según [18] pueden clasificarse en:

- Estimación por promedios de ensamble, cuando en su cálculo no se considera el desarrollo de la señal en el tiempo.

- Estimación por promedios de tiempo, cuando se considera el desarrollo temporal para la señal analizada.

De acuerdo a las técnicas de estimación del Pitch en tiempo real, éstas se dividen en [18]:

- Análisis estacionario (en intervalos cortos de tiempo).
- Análisis no estacionario (generalmente recurriendo a las transformadas conjuntas tiempo-frecuencia).

A modo de resumen, en la mayoría de las implementaciones de los algoritmos de cálculo del Pitch, la metodología utilizada consta de tres etapas principales [18]:

I. El preprocesamiento que además de adecuar las características de la señal a la entrada del sistema, busca también reducir la información redundante en la misma.
II. La extracción de parámetros de estimación del Pitch.
III. El postprocesamiento que corrige los posibles errores del sistema de detección.

1.2.1- Estimación paramétrica

Uno de los tópicos más investigados dentro del campo del procesamiento digital de señales, corresponde al análisis e identificación para series de tiempo, principalmente en cuanto a propiedades de parámetros descritos por datos constantes o de lenta variación temporal [19]. Muchos problemas prácticos como *Control de calidad*, *Procesamiento orientado hacia reconocimiento, y de monitoreo o detección de fallos en plantas industriales* pueden describirse con ayuda de modelos paramétricos sujetos a cambios abruptos en instantes de tiempo desconocidos [18].

1.2.1.1- Detección de Cambios Abruptos

Como cambio abrupto, se entenderán aquellos ocurridos rápidamente con respecto al período de muestreo de las medidas [19, 20].

Puesto que la mayor cantidad de información contenida en un proceso de medida, recae sobre las no-estacionariedades y dado que la mayoría de los algoritmos adaptativos de estimación, básicamente pueden seguir solo cambios lentos, la

detección de cambios abruptos se convierte en un problema de interés dentro de diversas áreas de aplicación [18].

En el caso del procesamiento digital de señales de voz, esta filosofía se aplica principalmente en tareas de segmentación, para las cuales se realiza la descomposición automática en segmentos estacionarios (o débilmente no estacionarios), con longitud adaptada a las propiedades locales de la señal, posibilitando la detección de cambios en características, y estimación de los lugares, y tiempos (o espacios) para ocurrencia de los mismos [18, 20]. En este sentido, considerando el instante de cierre glótico como un cambio abrupto, podría igualmente pensarse en aplicaciones para estimar el período fundamental F0.

En [18, 20], se presenta la derivación formal de algoritmos de detección para cambios abruptos, que se realiza dentro de la clase de métodos estadísticos de análisis en procesos aleatorios estacionarios, considerando dos modelos:

- Procesos aleatorios completamente independientes.
- Modelos regresivos.

1.2.2- Estimación no paramétrica

Otro de los tópicos en los que se investiga dentro del campo del procesamiento digital de señales, es el análisis e identificación en el dominio del tiempo, de propiedades de parámetros variables [19].

2.2.1-Función de Autocorrelación

Un proceso estocástico se define como *estacionario* si todas sus características de aleatoriedad (distribución y momentos, esencialmente) presentan invarianza temporal. Adicionalmente, se asume la *ergodicidad* del proceso, cuando los valores promedio de tiempo y de ensamble son idénticos en el intervalo de análisis T_a *[18].* Luego, puede definirse la función de autocorrelación (***FAC***) $R_x(\tau)$, para un proceso que posea las particularidades anteriormente descritas, según [21] a partir de la siguiente función:

$$R_x(\tau) = \lim_{T_a \to \infty} {}^{1}\!/\!_{T_a} \int_{T_a} x(t)x^*(t+\tau)dt = \langle x(t), x(t+\tau) \rangle \qquad \textbf{(1.1)}$$

Por cuanto, la ***FAC*** tiene un papel básico en la descripción de los procesos *ergódicos [20, 22]*, es importante tener en cuenta las siguientes propiedades:

- *Paridad.*

$$R_x(\tau) = R_x(-\tau) = R_x(t, t+\tau) = E[x(t), x(t+\tau)] \forall \tau \in T_a$$

(1.2)

- *Máximo valor.*

$$|R_x(\tau)| \leq R_x(0)$$

(1.3)

Por cierto, asumiendo que *E*[x(t)]=0,

$$R_x(0) = E[x(t), x(t)] = E^2[x(t)] = \sigma_x^2$$

(1.4)

- *Aditividad*. Sean dos procesos ergódicos independientes $x_1(t)$ y $x_2(t)$, se asume que:

$$R_x T(\tau) = R_{x1}(\tau) + R_{x2}(\tau)$$

(1.5)

Donde, $R_x T(\tau)$ corresponde a la función de autocorrelación resultante de la interacción entre dichos procesos.

- *Convergencia*. Si el proceso aleatorio *x(t)* es no periódico, esto es, $x(t) \neq x(t+T_0)$, $T_0 \in T_a$, además de considerar que es centralizado, esto es, *E*[x(t)]= 0, entonces se cumple que [18, 20]:

$$\lim_{|\tau| \to \infty} R_x(\tau) = 0$$

(1.6)

- *Periodicidad*. Si el proceso aleatorio x(t) es periódico para un valor T_0, esto es, $x(t) = x(t+T_0), T_0 \in T_a$, entonces la respectiva ***FAC***, también será periódica:

$$R_x(\tau) = R_x(\tau + kT_0), \forall \tau = T_0, k \in Z$$

(1.7)

Por tanto, la existencia de valores iguales al máximo global, $R_x(kT_0) = R_x(0), k \in Z$, muestra que el proceso es periódico con frecuencia fundamental $F_0 = 1/T_0$. Sin embargo, aunque exista un solo máximo global en $\tau = 0$, pueden a veces presentarse otros máximos locales, cuyo mayor valor $R_x(\tau \max)$ sea suficientemente grande, como para asumir que el proceso tiene una parte periódica en $T_0 = \tau \max$.

- *Restricción de forma.* Si el proceso estocástico *x(t)* es físicamente realizable, entonces, cualquier forma que tome la Transformada de Fourier de su respectiva ***FAC*** (Transformada de Wiener-Jinchin [23]) está restringida a la siguiente condición:

$$S_x(w) = F\{R_x(\tau)\} \geq 0, \forall w \qquad \textbf{(1.8)}$$

La función $S_x(w)$ es denominada *Densidad Espectral de Potencia* del proceso *x(t)* o simplemente espectro de potencia, y nos brinda información acerca de la señal aleatoria en el dominio de la frecuencia [18]. Adicionalmente, sea la señal:

$$x(t) = \sum_n \chi_n \exp(i \lambda_n t) \qquad \textbf{(1.9)}$$

Entonces, puede definirse $R_x(\tau)$ a partir de:

$$\lim_{T_a \to \infty} \frac{1}{T_a} \int_{-T_a} x(t+\tau) x^*(t) dt = \sum_n |\chi_n|^2 \exp(i \lambda_n \tau) \qquad \textbf{(1.10)}$$

Donde el lado derecho de la expresión representa el espectro de potencia [18].

1.3- Técnicas de estimación del Pitch en tiempo real

En este epígrafe se van a explicar solo dos de las técnicas de estimación del Pitch en tiempo real las cuales fueron implementadas en el presente trabajo. Si se quiere profundizar más en las otras técnicas remítase a [18].

1.3.1- Análisis estacionario sobre intervalos cortos de tiempo

En los métodos de análisis sobre intervalos cortos de tiempo, los segmentos de voz con longitud ***T*** (denominada apertura), se procesan como si cada uno de ellos tuviese propiedades estadísticas independientes [24]:

$$y[n] = \sum_{m=-\infty}^{\infty} \Gamma\{x[m]\} w[n-m] \qquad \textbf{(1.11)}$$

Donde x[m] corresponde a la señal digitalizada de voz, w[n-m] es la función ventana centrada respecto a la apertura ***T***, escogida de manera que facilite la extracción de características acústicas en la señal de voz, mientras que $\Gamma\{.\}$ constituye la transformación a la cual se somete la secuencia aleatoria original, y que en el cálculo del Pitch, emplea comúnmente la función de correlación (*Short Time Cross Correlation Function STCCF*), o bien la transformada de Fourier (*Short Time Fourier Transform STFT*), en calidad de operadores asociados [25].

1.3.1.1- Estimación en el dominio del tiempo

En este caso se aplica el análisis en intervalos cortos (habitualmente de forma síncrona [21]), tomando en consideración las siguientes propiedades [26] :

- La existencia de un armónico fundamental, o de una estructura repetitiva, a una velocidad semejante con el período del Pitch.
- Tendencia para altas amplitudes al principio del período fundamental, que irán decayendo hacia el epílogo del mismo.
- Presencia de una serie de discontinuidades en la señal de voz, enfatizadas principalmente en términos de su segunda derivada.

Por tanto, la mayoría de los métodos de estimación del **Pitch** que operan en el dominio del tiempo, realizan su extracción a partir de la forma de onda de la señal [26]. Para ello, habitualmente, la etapa de preprocesamiento suele consistir en un

filtro de paso bajo, que busca eliminar o disminuir la influencia para altas frecuencias, dejando pasar los mínimos armónicos posibles del fundamental. Por tanto, a su salida se aplicarán métodos complementarios como detectores de umbral, o de cruces por cero.

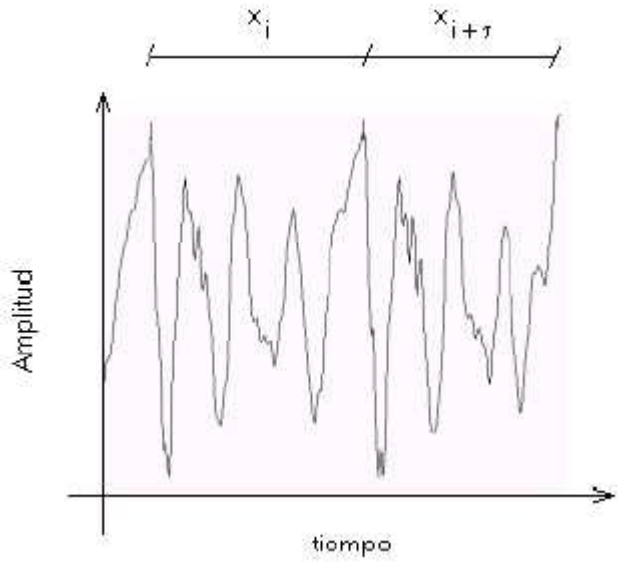


Figura 1.3.1.1: Función de correlación en intervalos cortos de tiempo.

1.3.1.1.1- Algoritmo Short Time Cross Correlation Function (STCCF)

Un primer algoritmo conocido como ***STCCF***, toma en (**1.11)** la función rectangular de ventaneo, w[m]=1, como se ilustra en la figura 1.3.1.1 permitiendo detectar la **FAC** de manera vectorial, según [21] a partir de:

$$R_x(\tau) = \frac{x_i x_{i+\tau}}{[x_i][x_{i+\tau}]} \qquad \textbf{(1.12)}$$

Donde la estimación T_0 del período fundamental se define como el argumento para el cual la ***STCCF*** toma su máximo valor, así en [21]:

$$T_0 = \arg\{\max(\tau)\}, \forall T_{0\min} \le \tau \le T_{0\max} \qquad \textbf{(1.13)}$$

La estimación del Pitch al realizarse sobre argumentos cortos de voz, supone previamente la toma de decisiones en cada intervalo de análisis, a partir de las

siguientes hipótesis: la presencia de sonidos (λ=1) o su ausencia (λ=0). Por tanto, como umbral de detección γ, puede ser empleado uno de los coeficientes de reflexión derivado de **(1.12)**:

$$T_{1x} = \frac{\overrightarrow{x_i} + \overrightarrow{x_{i+1}}}{\left|\overrightarrow{x_i}\right|\left|\overrightarrow{x_i}\right|} \leq \gamma, \lambda = 0$$
$$T_{1x} = \frac{\overrightarrow{x_i} + \overrightarrow{x_{i+1}}}{\left|\overrightarrow{x_i}\right|\left|\overrightarrow{x_i}\right|} > \gamma, \lambda = 1 \qquad \textbf{(1.14)}$$

u otro umbral obtenido a través de la práctica en estos tipos de estimaciones con el objetivo de ganar en precisión.

En orden a obtener la estimación no sesgada de la función de correlación en **(1.12)**, la media del proceso discretizado sobre el intervalo de análisis | μ_x | debe ser restada de la señal original:

$$x_0(n) = \left|x[n] - \mu_x\right| w[n], -\frac{N}{2} \leq n \leq \frac{N}{2} \qquad \textbf{(1.15)}$$

1.3.1.1.2- Algoritmo Magnitude Difference Function (AMDF)

De otro lado, en [27], se propone un algoritmo para el cálculo del Pitch con mayor estabilidad, basada en la función diferencia media de amplitud [21] (*Amplitude Mean Difference Function* ***AMDF***), que se fundamenta en la utilización de muestras sucesivas de señal, para construcción del siguiente modelo:

$$\Delta x(k,m) = \frac{1}{N}\sum_{n=m-(N-1)}^{m} \left|\chi_0(n) - \chi_0(n-k)\right| w(m-n) \qquad \textbf{(1.16)}$$

Donde "m" equivale al número de muestras empleadas en la estimación, al tiempo que "k" corresponde con el retraso para el cual se calcula el ***AMDF***.

En general la estimación del Pitch basada en (**1.12**) y (**1.16**) está afectada por la ***FAC***, en la forma de ventana empleada [21]:

$$R_x(\tau) \approx \frac{R_a(\tau)}{R_w(\tau)} \qquad \textbf{(1.17)}$$

Siendo $R_x(\tau)$ el equivalente de autocorrelación normalizada para la señal enventanada, en la misma medida que $R_w(\tau)$ representa su contraparte sobre la función de ventana. Así, considerando un tren de señales delta con período T_0, definidas como:

$$x(k) = \sum_{n=-\infty}^{\infty} \delta(k - k_0 - nT_0), 0 \le k_0 \le T_0 \qquad \textbf{(1.18)}$$

Para k_0 denotando la fase de los pulsos con la ventana, se tiene que (**1.17**), será:

$$R_x(T_0) = \frac{\sum_n w(k_0 + nT_0) w(k_0 + (n+1)T_0)}{R_w(T_0) \sum_n w^2(k_0 + nT_0)} \qquad \textbf{(1.19)}$$

Luego, la exactitud del algoritmo depende directamente del tipo de ventana utilizado. Por tanto, se asume que para el caso de aplicaciones en voz, la ventana *Hamming* es preferible a otras formas típicas [10], debido a su menor sensibilidad respecto a cambios rápidos de señal [21].

Finalmente, cabe anotar que en los algoritmos analizados de ***STCCF***, a efectos de disminuir tiempos de cómputo, es preferible realizar el cálculo numérico de la ***FAC*** empleando técnicas que apliquen *Transformada Rápida de Fourier (**FFT**)* [21]**.**

1.3.1.2- Estimación en el dominio de la frecuencia

Esta estimación se basa en el empleo de la FFT sobre características de tiempo contenidas en segmentos ventaneados de la señal de entrada, buscando enfatizar la información de sus armónicos.

El primer caso de análisis corresponde al *Cepstral*, como ejemplo de transformación homomórfica [28], utilizada para descomponer tanto la envolvente, como la estructura fina del espectro en una señal de voz, por medio de la Transformada Inversa de Fourier, aplicada sobre el logaritmo espectral de los segmentos analizados [26]. Este método de estimación se obtiene a partir de la acción conjunta de los siguientes bloques principales [18]:

- Un tren de impulsos periódicos *P(z)*.

- La función característica discreta de la Glotis *G(z)*, y
- La función de transferencia pulso del tracto vocal, incluyendo el efecto de radiación *M(z)*.

Luego, la señal de salida será por producto: *Y(z)= P(z) * G(z) * M(z)*, que para el dominio temporal, quedará en forma de convoluciones [26].

De otro lado, existen métodos basados en histogramas [29], en los cuales se realiza un análisis espectral sobre segmentos de voz utilizando los intervalos de distancia presentes entre diferentes armónicos localizados. Por tanto aquel valor que más veces se produzca, corresponderá con el período del Pitch. De esta manera no será necesario conocer cuál es el primer armónico, sino, que por comparación sucesiva, puede determinarse la frecuencia fundamental para el segmento analizado. Dentro de esta clase de algoritmos para el cálculo del Pitch, está el de producto de armónicos espectrales *(Harmonic Product Spectrum - HPS)*, que asume como frecuencia fundamental para una señal periódica, al máximo común divisor de sus frecuencias armónicas [20].

Otros tipos de sistemas que operan en el dominio de la frecuencia se explican en [26].

Una vez que se tenga la envolvente de la frecuencia fundamental se calcula el contorno del Pitch.

1.4- Contorno del Pitch

La construcción del contorno del Pitch se realiza con el objetivo de encontrar una curva que describa la entonación del hablante al decir la frase que se está analizando para luego el especialista pueda determinar si el hablante realizó la entonación de forma correcta o no, comparándola con la entonación de una voz grabada en condiciones de laboratorio de la misma frase dicha por un lingüista o un foníatra.

Como los métodos de detección de la F0 (o del Pitch) cometen errores que dificultan las interpretaciones de las curvas melódicas se realiza un proceso de estilización de la misma con el objetivo de eliminar dichos errores.

1.5- Estilización de la frecuencia fundamental o del Pitch

Se realiza una estilización de la curva melódica de la frecuencia fundamental con el objetivo de eliminar los errores anteriormente dichos quedando las curvas reducidas a una serie discreta de puntos, que unidos mediante líneas, proporcionan una representación de sus movimientos más relevantes basados en los puntos de inflexión [30]. De esta forma se pueden eliminar las variaciones irrelevantes de la F0.

1.6- Métodos de segmentación de la señal de voz en sílabas

La segmentación de voz consiste en dividir una emisión en diferentes trozos de acuerdo

con algún criterio. Es común que se realice segmentación de voz para separar en fonemas y también suele ser de interés según sílabas o unidades de nivel superior, como la palabra.

En el presente trabajo se pretende realizar la segmentación de la señal de voz en sílabas con el fin de que en trabajos futuros se haga un algoritmo para detectar la tonicidad de las sílabas, es decir, determinar cuando una sílaba es tónica o no.

La segmentación silábica constituye uno de los aspectos más difíciles de lograr de forma automática debido a que la señal de voz es muy dependiente del hablante y no existen umbrales fijos confiables para poder determinar, con exactitud de un 100 %, los extremos silábicos.

En la bibliografía consultada se han implementado varios métodos de segmentación de señales de voz en sílabas y en fonemas, aquí citaremos algunos de ellos:

- Segmentación Manual **[1]**
- Modelos Ocultos de Harkov [31, 32].
- Segmentado a través de un análisis de las transiciones espectrales [33, 34].

*1.6.*1- Segmentación Manual

Es en la que generalmente un experto lingüista genera la segmentación en base a análisis de los espectrogramas, curvas de energía, entonación y otros estudios utilizados para el análisis de voz.

Esta técnica posee la ventaja de que la experiencia del lingüista asegura un muy buen resultado en la segmentación. Sin embargo los costos en tiempo y recursos que lleva este proceso manual son altísimos, lo que lo hace solo aplicable a estudios muy especializados y en casos excepcionales.

1.6.2- Modelos Ocultos de Markov

La segunda técnica aplicable a la segmentación viene de la mano de los Sistemas de Reconocimiento Automático del Habla (RAH) basados en HMM.

Un Modelo Oculto de Markov (HMM) es un modelo paramétrico que permite describir hechos acústicos del habla, y que está caracterizado por una serie de variables estadísticas tales como:

- Conjunto de estados caracterizados cada uno por una determinada probabilidad tanto de emisión como de transición entre estados.
- Transición entre estados cada un cierto tiempo *T* que depende del estado del cual parte.
- Observación de salida que se produce siguiendo una función de densidad de probabilidad después de cada transición, dependiendo del estado en que se produce y no de como se ha llegado hasta él.

Este modelo está formado por dos procesos estocásticos, uno conocido que es la producción de los símbolos o salidas, y otro oculto, que es el paso de unos estados a otros.

Primero se entrena un sistema de RAH convencional y mediante el algoritmo de Viterbi se puede obtener la secuencia más probable de estados que determina la segmentación para lo cual se necesita contar con una transcripción correcta de la emisión de voz. [32].

Los *HMM* han sido utilizados frecuentemente desde hace varios años en la segmentación de la señal de voz en fonemas [31], ya que permiten obtener muy buenos resultados de forma automática pero requiere de mucho tiempo ya que se necesita entrenarlo con un conjunto de grabaciones y luego una etapa de prueba con otras grabaciones para verificar si no se requiere que se siga entrenando en función

de con que exactitud segmenta las grabaciones; y en el departamento de Procesamiento de Voz del CEETI no se cuenta con dichas grabaciones.

1.6.3- Segmentado a través de las transiciones espectrales

El segmentado de la señal de voz es una tarea ardua y difícil de realizar de forma automática pues en nuestro idioma existen muchas reglas ortográficas que influyen en la determinación de las sílabas.

Trabajos previos en el segmentado han demostrado que las posiciones máximas de las transiciones espectrales están en proximidad cercana (dentro de aproximadamente 10 ms) a los puntos críticos que se corresponden con la percepción de una consonante o con el inicio o fin de una sílaba [35].

En este método se realiza un análisis cuantitativo comparando las posiciones del límite de las sílabas obtenidas basado en el índice del cambio espectral máximo y las posiciones de los límites manualmente definidos. Para determinar los extremos silábicos se utilizaron las características espectrales de la señal de voz.

Las características espectrales usadas en este estudio son los *Coeficientes* de la *Mel-Frecuency Cepstrum (MFCC)*. Ellas se utilizan extensivamente en ASR (sistemas de reconocimiento automático) y los detalles en su cómputo pueden ser encontrados en [36]. Las señales de cada grabación primero se transforman en marcos espectrales (computados sobre ventanas Hamming[2] de 32 ms) y luego se transforman en un conjunto de 10 coeficientes de **MFCC** (excepto el coeficiente de orden cero que representa la energía total). Todos los coeficientes de la energía no fueron utilizados aquí porque este análisis se centra principalmente en las características espectrales. La razón del marco empleada en este estudio fue de 10 ms. Los coeficientes dinámicos de **MFCC** no fueron utilizados directamente en este estudio puesto que el índice del cambio espectral representa una medida dinámica por sí mismo.

El criterio usado en este estudio para la segmentación silábica fue basado en una medida del índice del cambio espectral en el tiempo. Puesto que el índice del cambio

28

[2] Es una función que se utiliza en Matlab para suavizar la envolvente de una determinada función.

espectral exhibe generalmente picos en la transición entre las sílabas, tal medida se puede utilizar para detectar límites entre las sílabas. Debe ser observado que no todos los picos en el índice del cambio espectral corresponden a un límite válido. Por ejemplo, un diptongo exhibiría un pico de la medida de la transición espectral situada aproximadamente en su centro; sin embargo esto no representa un límite válido de la sílaba.

La medida de la transición espectral empleada en este estudio fue la misma que la propuesta en [35] la cual puede ser interpretada como la magnitud del índice de cambio espectral. Esta medida de la transición espectral (STM), en el marco ***m***, se puede computar como un valor cuadrático medio:

$$STM(m) = \frac{\left(\sum_{n=-I}^{D} a_i^2(m)\right)}{D} \qquad \textbf{(1)}$$

donde **D** es la dimensión del vector espectral característico (10 en este caso) y $\mathbf{a_i(m)}$ es el coeficiente de regresión o el índice del cambio de la característica espectral **MFCC** definida como:

$$a_i(m) = \frac{\left(\sum_{n=-I}^{I} MFCC_i(n+m)*n\right)}{\left(\sum_{n=-I}^{I} n^2\right)} \qquad \textbf{(2)}$$

donde *n* representa el índice del marco y la *I* representa el número de marcos (en cada lado del marco actual) usados para computar estos coeficientes de regresión. Utilizamos I=2 para un paso del marco de 10 ms que corresponde a un intervalo de 40 ms centrado en el marco actual en el cual se computa el valor de STM. En un intervalo muy grande podría resultar que faltaran algunos límites silábicos mientras que un intervalo más corto podría dar lugar a la detección de demasiados límites falsos.

Este algoritmo está basado en la búsqueda de los máximos espectrales que se deben corresponder con los límites silábicos, lo cual, en la práctica no se cumple al ciento por ciento por lo que entre dichos máximos que se consideran extremos hay algunos que son falsos, para eliminarlos se tomaron varias medidas.

Una medida que se tomó para eliminar los límites falsos de las sílabas en este algoritmo es el filtrado de la señal de voz con un filtro de paso bajo para eliminar los sonidos de altas frecuencias. Otros criterios también fueron aplicados para detectar límites falsos de sílabas.

Los otros criterios usados fueron:

- Los límites silábicos tienen que tener una amplitud mayor que un umbral determinado. Esto elimina todos los máximos detectados como límites de sílabas en las zonas de silencio., provocados por ruido.
- Distancia entre dos posibles extremos silábicos contiguos tiene que ser un tiempo prudencial.
- Entre dos límites de diferentes sílabas debe haber un mínimo considerable.
- La sílaba debe tener una duración determinada. La separación entre el mínimo más pronunciado anterior y el posterior a la sílaba debe ser suficiente para que esta sea percibida como tal.

Todos estos criterios traen consigo obtener los verdaderos extremos silábicos con cierta precisión, lo cual hace más robusto el algoritmo.

1.7- Energía por tramas

La energía es una de los parámetros suprasegmentales de la señal de voz que estima la fluctuación de la intensidad de la misma en el transcurso del tiempo.

Este parámetro puede obtenerse calculando el valor cuadrático promedio de los valores instantáneos de la señal en segmentos de corta duración.

La fórmula matemática que determina la energía es la siguiente:

$$E_j = \frac{\sum_{i=j*N}^{(j+1)*N-1} x_i^2}{N} \quad , \qquad j = 0..E\left[\frac{M}{N}\right]-1$$

donde *M* es el número total de muestras de voz a procesar, *N* es el tamaño de las subtramas dado en número de muestras, *x* es el vector de los valores instantáneos de la señal de voz y *E* es el vector resultante de energía.

Otra vía es calculándola a partir de una fórmula en función del logaritmo de la norma de la señal lo que hace un poco más suave la curva que describe a la energía.

Dicha fórmula es:

$$E_j = 10 * \log_{10}\left(norm(x)\right)$$

Donde *"x"* es la señal, ***log₁₀*** es la función logaritmo base 10 y E_j es el valor de la energía que resulta del cálculo.

1.8- Matlab como entorno de programación

Matlab es un entorno de computación y desarrollo de aplicaciones, en lenguaje *Matlab*, totalmente integrado orientado para llevar a cabo proyectos en donde se encuentren implicados elevados cálculos matemáticos y la visualización gráfica de los mismos. Su nombre proviene de la contracción de los términos *"**MAT**rix **LAB**oratory"* **[37]**. Integra análisis numérico, cálculo matricial y procesado de señales, entre otras aplicaciones. Es un lenguaje de programación interpretado.

1.9- Conclusiones Parciales

En este capítulo se realizó un estudio acerca del mecanismo de producción del habla humana explicando detalladamente el funcionamiento del aparato fonador así como los órganos que intervienen en el mismo. Se precisaron los aspectos teóricos relacionados con la lingüística tales como las características fonéticas de las vocales y consonantes del idioma español, y los componentes suprasegmentales de la prosodia que son la intensidad, la entonación y el ritmo. Se describieron de forma detallada los algoritmos más utilizados en la actualidad para determinar dichos componentes en la señal de voz según la bibliografía consultada.

Capítulo 2: Arquitectura del sistema

En este capítulo se tratarán los aspectos relacionados con la arquitectura del sistema que aquí se implementó para lo cual presentamos los diagramas resultantes de analizar el sistema en función del flujo de trabajo e información en el transcurso de las diferentes fases del proceso de desarrollo de software.

Dicho capítulo está dividido en varios epígrafes, el primero está dedicado a realizar una descripción de los aspectos referentes al proceso de desarrollo de software usando como herramienta principal para ello el UML (*Unified Modeling Language*), el segundo está dedicado al modelo de casos de uso del sistema, el tercero al diagrama de transición de estados de los casos de uso más importantes, en el cuarto se muestra el modelo de despliegue, el quinto se dedica al modelo componentes y el último está dedicado a la vista de implementación que en su conjunto conforman una descripción de la arquitectura del sistema resaltando en cada caso los aspectos esenciales del mismo.

2.1- ¿Qué es UML?

UML (*Unified Modeling Language*) [38] es el lenguaje que se ha difundido entre la comunidad de desarrolladores de software que tiene como objetivos principales la especificación, visualización, construcción y documentación de los productos de un sistema intensivo de software.

Inicialmente este lenguaje surgió como iniciativa de los Doctores James Rumbaugh, Ivar Jacobson y Grady Booch a mediados de la década de los años 90 del siglo pasado, que atrajo la atención de disímiles empresas tales como: Microsoft, Oracle y Hewllet Packard, para dar origen a una secuencia de versiones que condijeron a la conformación de la versión 1.1 de UML, aprobada por la OMG (Object Management Group) en noviembre de 1997. Actualmente este lenguaje es usado por el RUP (Rational Unified Process) [39] como lenguaje de modelado para lo cual se basa en todos sus tipos de diagramas, que constituyen diferentes vistas del modelo del producto.

La siguiente figura 2.1 ilustra los diferentes diagramas que componen la estructura de un producto escrito por el lenguaje UML:

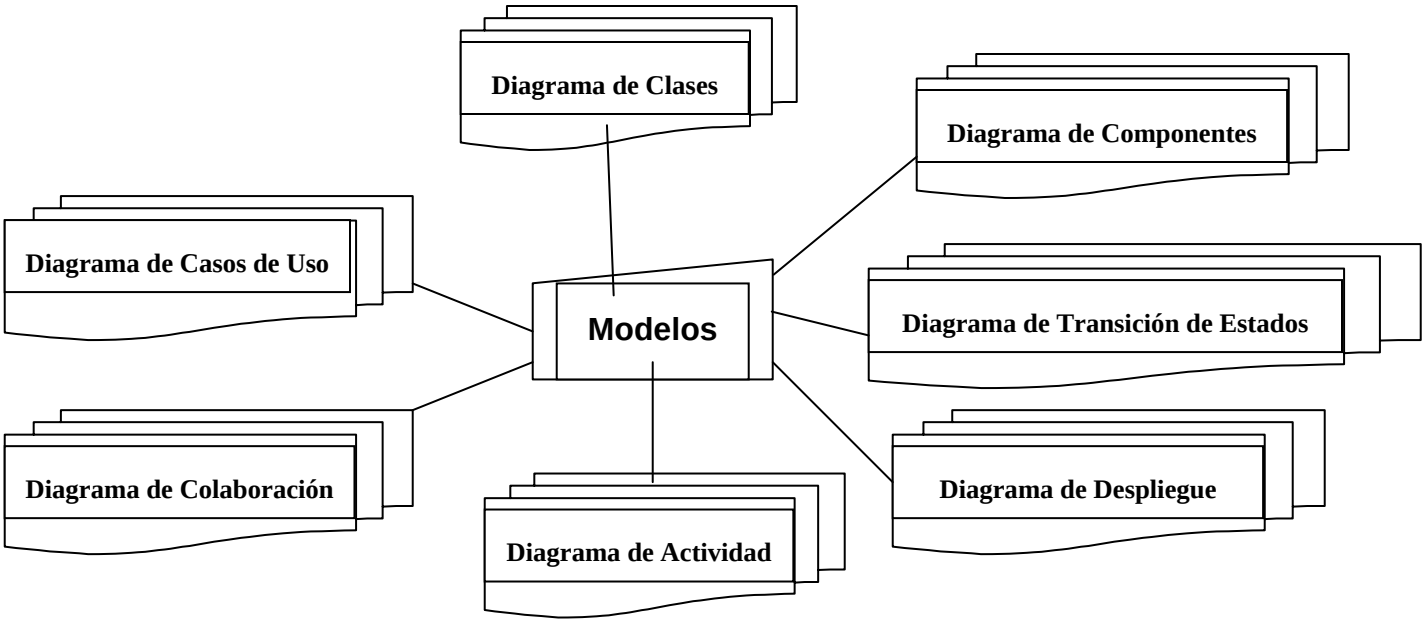


Figura 2.1: Diagramas de UML.

De los diagramas de UML que se mostraron en la figura anterior, en este sistema se emplearon solamente: *Diagrama de Casos de Uso*, *Diagrama de Transición de Estados*, *Diagrama de Despliegue* y *Diagrama de Componentes*.

2.1- Modelo de Casos de Uso

En este epígrafe se realiza el modelo de casos de uso ya que proporcionan un medio sistemático e intuitivo de capturar requisitos funcionales del sistema [39] basándose en los requerimientos de los usuarios. Ellos dirigen todo el proceso de desarrollo de software debido a que son el punto de partida para llevar a cabo la mayoría de las actividades como el análisis, diseño y prueba del software [39].

Este modelo se realiza identificando cada actor del sistema como los posibles usuarios para los cuales está realizado el sistema.

2.1.1- Actores y casos de uso del sistema

El sistema tiene un solo actor que es el especialista en foniatría o lingüista capaz de interpretar de forma correcta los resultados que brinda el software. Al especialista se le ha nombrado en el diagrama de la figura 2.1.1ª) como usuario.

Figura 2.1.1ª)

Los casos de uso más generales del actor usuario del sistema se muestran en la figura 2.1.1b):

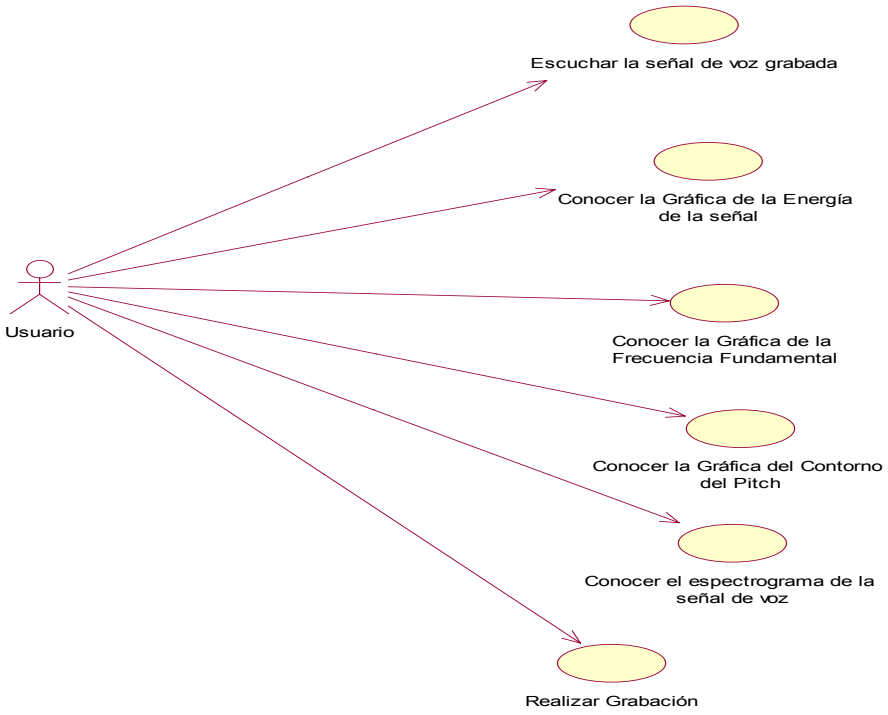
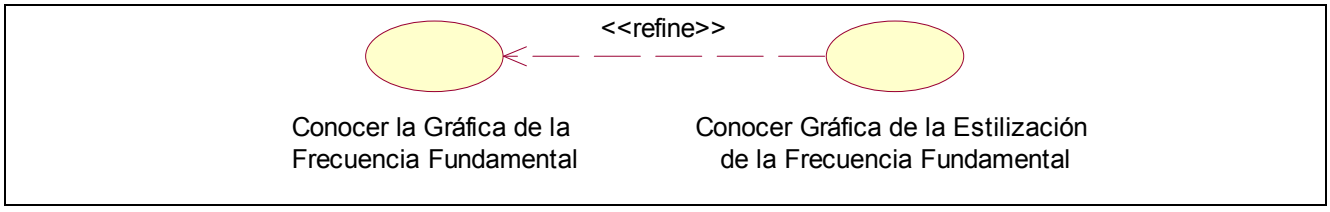


Figura 2.1.1b): Diagrama de casos de usos más importantes del sistema.

Si se quiere profundizar en los demás casos de uso del sistema ver el Anexo 4.

Seguidamente se realizó un refinamiento de algunos de los casos de uso generales del actor “Usuario” y se obtuvo el siguiente diagrama:



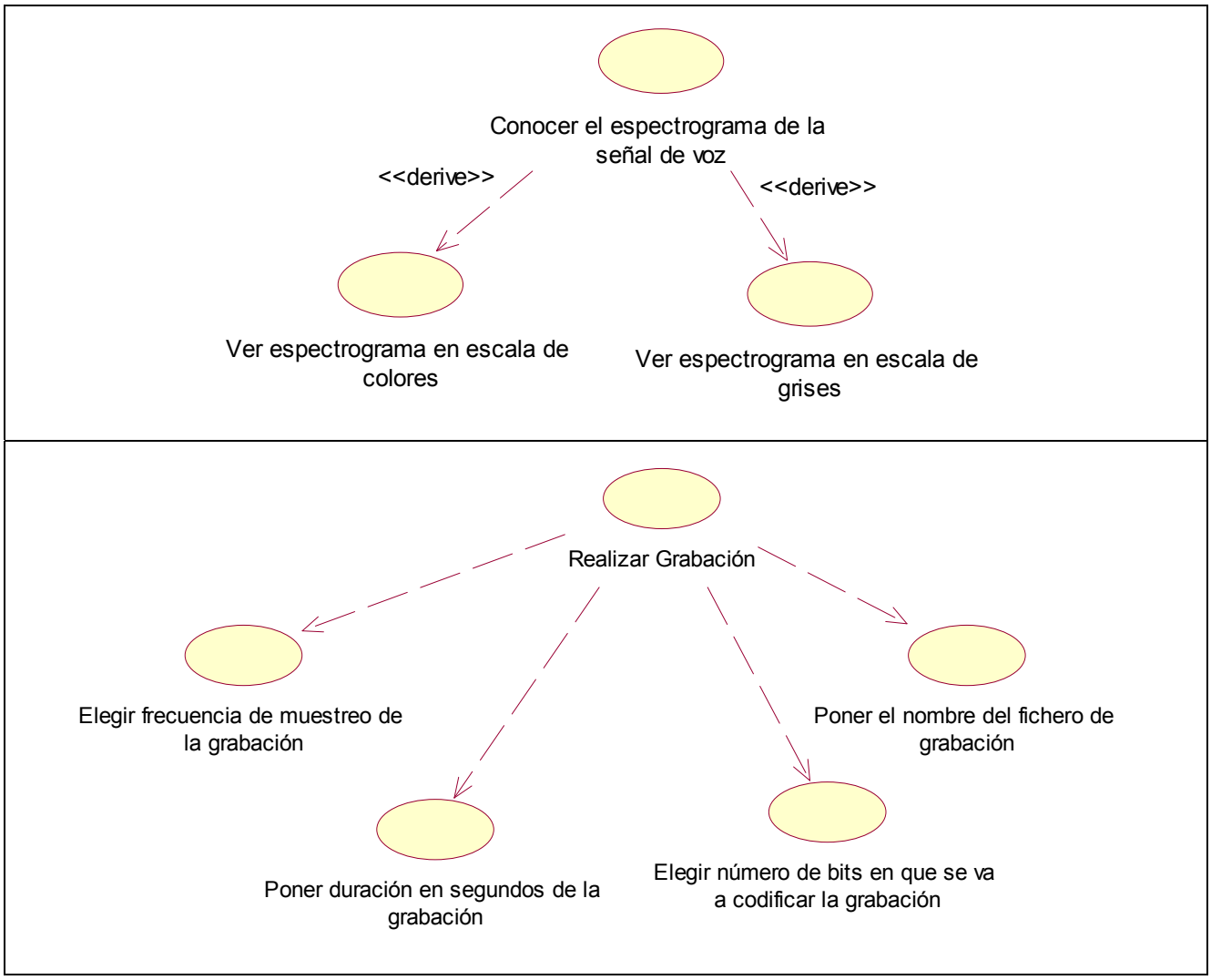


Figura 2.1.1c): Refinamiento de un caso de uso del actor "Usuario"

2.1.2- Descripción de los casos de uso del actor "Usuario"

En la tabla 2.1.2 que aparece a continuación se describen cada uno de los casos de uso del único actor del sistema.

Caso de uso	**Descripción**
Escuchar la señal de voz grabada	El actor utiliza este caso de uso para escuchar la grabación de voz que se tiene graficada para procesar, con el fin de saber lo que dice.
Conocer la Gráfica Energía de la señal	El actor utiliza este caso de uso para conocer gráficamente la envolvente de la energía de la señal de voz que es el parámetro físico que describe la intensidad de la señal que percibimos.

Conocer la Gráfica de la Frecuencia Fundamental (F0) de la señal	El actor utiliza este caso de uso para conocer gráficamente la envolvente de la Frecuencia Fundamental (F0) de la señal de voz que es el parámetro físico que describe la entonación de la señal que percibimos. Luego se corrigen los valores de la F0 estilizando la misma.
Conocer la Gráfica del Contorno del Pitch de la Señal	El actor utiliza este caso de uso para conocer el contorno del Pitch (tono) de la señal de voz que es el parámetro físico que caracteriza al Ritmo que percibimos de la señal.
Conocer espectrograma de la señal de voz	El actor utiliza este caso de uso para conocer la gráfica que describe el espectrograma de la señal. Este espectrograma puede verlo en escala de colores o de grises con el objetivo de conocer los formantes de la grabación.
Realizar Grabación	El actor utiliza este caso de uso para realizar la grabación de una señal de voz en un fichero .WAV que luego se procesará en el software. Para ello el actor debe configurarle al sistema algunos parámetros necesarios para realizar la grabación.
Buscar la ayuda del sistema	El usuario utiliza este caso de uso con el objetivo de buscar información acerca de alguna funcionalidad del software que desconoce.

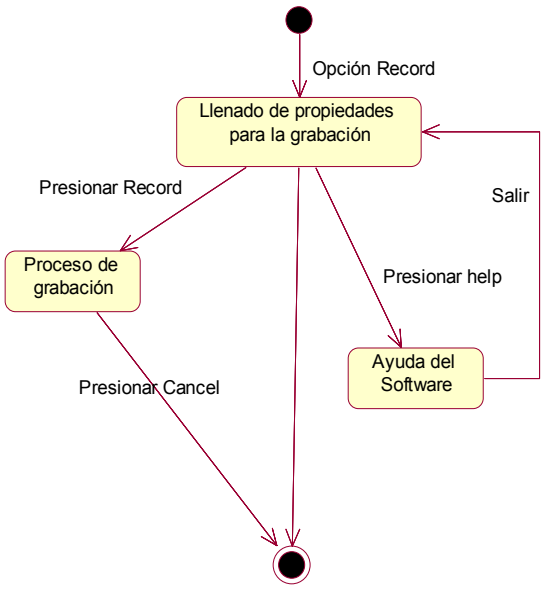
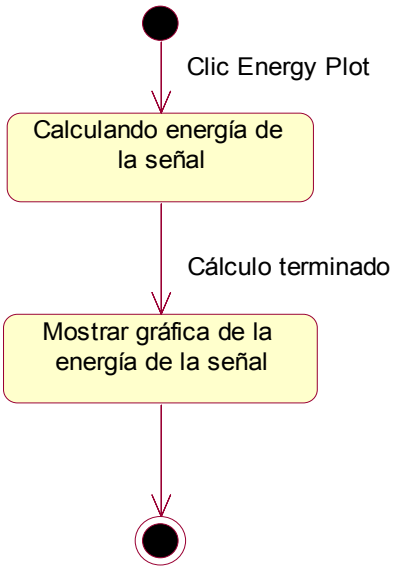
Tabla 2.1.2: Explicación de cada uno de los casos de uso

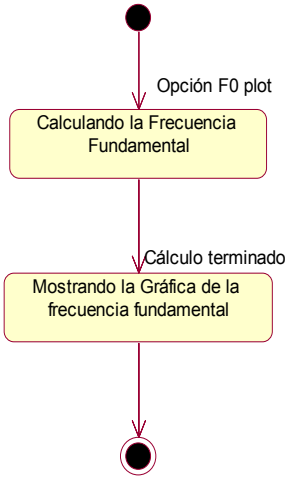
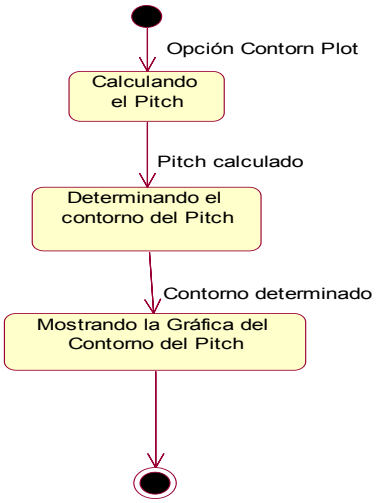
A continuación se mostrarán los diagramas de transición de estados de algunos de los casos de uso más importantes del sistema con el objetivo de detallar los estados por los que pasa el sistema durante la realización de dichos casos de uso.

2.2- Diagramas de Transición de Estados

El *diagrama de estados* es un diagrama que describe el conjunto de estados por los que pasa un objeto a lo largo de su vida en respuesta a eventos junto con su respuesta a esos eventos. Dicho objeto puede ser una *clase*, un *caso de uso* o un *sistema completo*.

En esta sección se va a mostrar el diagrama de estados de los casos de uso más importantes de este sistema.

<table>
<tr>
<td>Caso de uso: Realizar Grabación.
 
</td>
<td>Caso de uso: Conocer la Gráfica de la Energía de la señal.
 
</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</table>

Caso de uso: *Conocer Gráfica de la Frecuencia Fundamental (F0)*.	**Caso de uso:** *Conocer Gráfica del Contorno del Pitch*.
	

Los demás diagramas de transición de estados de los otros casos de uso más importantes se muestran en el Anexo 5.

En el siguiente epígrafe se definirán los aspectos físicos que caracterizan al sistema en cuestión con el objetivo de definir de forma detallada los componentes de hardware que necesita el sistema, como parte del modelo de diseño del software.

2.3- Modelo de Despliegue

El modelo de despliegue es un modelo de objetos que describe la estructura física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo interconectados [39]. Dichos nodos son elementos de hardware en los cuales pueden ejecutarse los elementos de software. Se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una notable influencia en su diseño [39].

En este modelo es necesario definir sus partes componentes:

- Conjunto de nodos: Cada uno representa un recurso de cómputo tales como un procesador o cualquier otro dispositivo similar.

- Relaciones entre nodos: Relaciones que representan los medios de comunicación entre los diferentes nodos tales como Intranet, bus, Internet, entre otros.
- Funcionalidad de un nodo: Se define por los componentes que se distribuyen sobre ese nodo.

El modelo de despliegue constituye en sí una correspondencia entre las arquitecturas software y hardware.

Después de analizado todos los aspectos de este modelo se determinó de que en el presente sistema sólo se necesita, como elemento de hardware, una computadora donde ejecutar la aplicación, que conste de una memoria RAM superior a los 128 MB para que la misma se pueda ejecutar sin mucha demora ya que los procedimientos implementados la necesitan; y una tarjeta de sonido en buen estado.

En la siguiente figura se muestra el diagrama de despliegue del sistema que solo consta de la computadora local del usuario ya que es el único dispositivo de hardware necesario para ejecutarla.

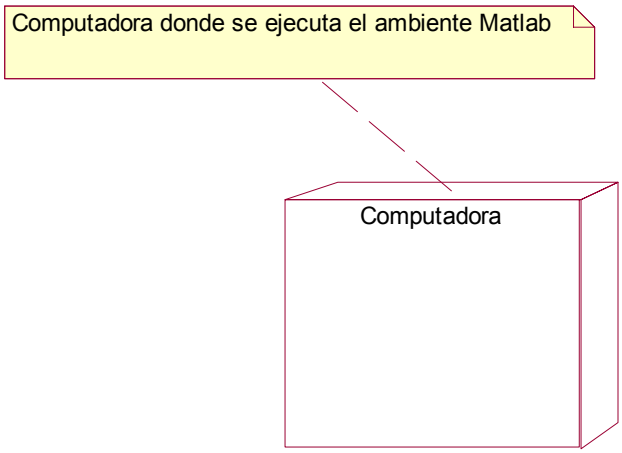


Figura 2.2: Diagrama de despliegue del sistema APA.

El usuario, para poder ejecutar la aplicación tiene que tener instalado en su computadora personal el ambiente de programación *Matlab* en su versión 6.5 u otra más reciente, y tener disponibles los ficheros componentes de la aplicación ya que el lenguaje de programación del mencionado ambiente es interpretado, y por supuesto, requiere de los ficheros fuentes para poder ejecutarla.

Seguidamente se analizará otro elemento de diseño del sistema, que en este caso es el modelo de componentes que muestra los componentes que conforman la aplicación.

2.4- Modelo de Componentes

Los componentes son un bloque de construcción importante ya que modelan los aspectos físicos de un sistema.

Los *componentes* se utilizan para modelar los elementos físicos de software que pueden hallarse en un nodo, tales como ejecutables, bibliotecas, tablas, archivos y documentos.

El diagrama de componentes del sistema se puede ver en el ver Anexo 6.

Seguidamente se hará un análisis detallado del modelo de implementación del software.

2.5- Modelo de Implementación

En este epígrafe se abordará la implementación del software. Todas las tecnologías utilizadas y una breve descripción de los algoritmos implementados que fueron ya explicados en el capítulo 1.

Para la implementación de esta aplicación se escogió el lenguaje de programación *Matlab*.

2.5.1- ¿Por qué Matlab?

Se escogió este lenguaje de programación por las ventajas y facilidades que nos brinda en el procesado de señales digitales y específicamente en el procesamiento de señales de voz. Es una herramienta que está siendo muy utilizada por los técnicos, ingenieros, informáticos, y en las universidades más prestigiosas a nivel mundial, por su aplicabilidad en la rama de las ciencias informáticas, las telecomunicaciones, electrónica, la eléctrica, entre otros campos.

En el departamento de *Procesamiento de Voz* del *CEETI* se cuenta con varios algoritmos y software ya implementados en *Matlab* lo que posibilita una compatibilidad de esta aplicación con dichos software y así poder conformar un

ambiente robusto que haga análisis de voces en el mismo ambiente de programación.

El sistema se puede ver como una caja negra la cual reciba como entrada una señal de voz grabada, haga un procesamiento de la misma y luego emita los resultados esperados que son las gráficas que describan:

- La entonación que es la estimación de la frecuencia fundamental de la señal de voz.
- Los cambios más significativos de la señal, los cuales se obtienen después de realizar una estilización de la frecuencia fundamental de la señal de voz.
- El contorno del Pitch (entonación o tono) de la señal de voz.
- La intensidad que es el comportamiento de la energía de la señal de voz en el tiempo.
- Las sílabas de la señal de voz.

En los próximos epígrafes se va a realizar una descripción detallada de los algoritmos implementados a través de los cuales se obtienen las gráficas anteriormente enunciadas.

2.5.1.1- Lo novedoso de Matlab

Lo novedoso del lenguaje *Matlab* es que basa su arquitectura en conceptos matemáticos; entre ellos está uno muy importante: la función. Mientras los lenguajes clásicos se basan en subrutinas u objetos *Matlab* dispone de una biblioteca formada exclusivamente por funciones. Este diseño tan simple es lo que llevado *Matlab* a su éxito, es un acercamiento matemático a la programación orientada a matemáticas.

El paradigma de programación que utiliza el lenguaje *Matlab* es la programación estructurada pues su unidad básica son las funciones.

Es un lenguaje script que tiene intérpretes en todos los sistemas operativos así que es multiplataforma siempre que se cuente con el intérprete adecuado con dicha plataforma.

2.5.1.2- Características del entorno Matlab

El entorno *Matlab* posee muchas características propias que favorecen y ayudan a realizar aplicaciones en el campo del procesamiento de señales las cuales son:

- Cálculos intensivos con gran precisión desde el punto de vista numérico.
- Gráficos y visualización avanzada.
- Lenguaje de alto nivel basado en vectores, arreglos y matrices.
- Colección muy útil de funciones de aplicación.

Cuenta con capacidades de cálculo técnico muy grandes y amplias, aunque limita un poco el tamaño de las matrices.
Posee *toolboxes* muy completos especializados en determinadas materias tales como el de señales que cuenta con un conjunto de herramientas para el procesamiento de las mismas con un alto grado de robustez y aplicabilidad. Otros toolboxes son: el de análisis de sistemas de cuadro y uno de matemática simbólica basado en Maple (Toolbox Symbolyc Math).

2.5.2- Algoritmo de estimación de la Frecuencia Fundamental (F0)

De los métodos de estimación de la *Frecuencia Fundamental (F0)* o *Pitch* estudiados en el capítulo anterior, en esta aplicación se implementó uno de ellos que es del tipo de los métodos no paramétricos, que está basado en una función AMDF en el dominio del tiempo. Se escogió este algoritmo para implementarlo porque según [40] tiene un mejor comportamiento en el dominio del tiempo que los demás estudiados ya que comete menos errores en la estimación del Pitch. Este algoritmo fue realizado a partir de uno que ya estaba implementado en el CEETI, que tomaba otras medidas para determinar sonoridad en segmentos silábicos, por lo que el presente es una modificación del otro; y produce una cierta mejora con respecto al que ya existía ya que realiza una mejor estimación del Pitch.
Las modificaciones que se le realizaron fueron las siguientes:

- En la determinación de la sonoridad de segmentos de señal se le modificaron los umbrales de energía mínima y de cruces por cero.
- Valores de Pitch mínimo y máximo a detectar.
- Tamaño de las muestras para el análisis de la señal.

Se implementó esta variante también porque reporta mejores resultados que los anteriormente estudiados debido a su gran estabilidad y precisión en los resultados según[40] .

2.5.2.1- Algoritmo de estimación de F0 basado en una función AMDF

Este algoritmo de estimación de la F0 que se implementó está basado, como se dijo anteriormente, en una función AMDF. Se comporta de una forma más estable en comparación con el basado en una función de autocorrelación y requiere de un menor tiempo de procesamiento que este.

La función que calcula la F0 por este método necesita que se le introduzca los siguientes parámetros:

- Los datos de la señal de voz grabada, es decir, se deben leer los datos de la grabación del fichero .wav y pasárselos como parámetro a la función. Se ha denotado este parámetro con la variable “xe”.
- La frecuencia de muestreo de la señal de voz grabada que indica con que frecuencia se hizo la grabación (fs).
- Las frecuencias mínima y máxima de búsqueda de la frecuencia fundamental (fmin, fmax).
- Tamaño de la ventana en segundos (WL).
- Solapamiento entre ventanas contiguas (S).

Una vez definidos los valores de estos parámetros y pasados a la función GrossF0Detector (que es la que implementa en algoritmo) comienza a ejecutarse la misma a través de una secuencia de pasos.

Seguidamente se mostrarán los pasos del algoritmo.

Pasos del algoritmo:

1) Primeramente se divide la señal en ventanas y se calcula el tamaño de dichas ventanas en muestras para analizarla mejor.
2) Calcular la longitud del paso entre ventanas contiguas.
3) Se calcula el número de segmentos en que se divide la señal teniendo en cuenta las longitudes de la señal y de cada ventana, y el paso entre ventanas.

4) Calcular los valores de Pitch mínimo y máximo en muestras a detectar.
5) Para cada uno de los segmentos en que se divide la señal para analizarla hacer:
   a) Tomar en un vector los valores de F0 del segmento a analizar.
   b) Determinar si el segmento es sonoro o no según cruces por cero o autocorrelación. Si es sonoro se le aplica la función AMDF y se obtiene un vector AMDF y se va al **paso c**. Si el segmento no es sonoro se le pone como F0 el valor cero y se va al **paso g**.
   c) Se suprime el valor medio, se normaliza y se invierte el vector AMDF (esto último con el objetivo de buscar máximos y no mínimos que es lo que devuelve el AMDF en los puntos correspondientes al período fundamental).
   d) Se buscan en dicho vector AMDF las posiciones de los tres máximos valores que están en el rango de los valores de Pitch mínimo y máximo.
   e) Se efectúa una ponderación de dichos máximos del vector AMDF, asignándole un peso inicial que es el valor que resultó de la normalización del vector, al cual se le adicionan otros pesos atendiendo a:
      - La coincidencia con puntos anteriores que se corresponden con el período fundamental. Aumenta el valor del peso en 0.10 unidades.
      - La existencia de múltiplos posteriores que se corresponden con puntos de período fundamental.
   f) Se calcula el valor de la F0 correspondiente al elemento, de los tres máximos del vector AMDF ponderados, en función del valor mayor resultante de la ponderación anterior.
   g) Se toma el próximo segmento a analizar.
6) Se devuelve el vector de valores de F0 de toda la señal.
7) Fin del Algoritmo.

En el anexo 7 se muestra un diagrama de bloques de este algoritmo.

2.5.3- Algoritmo de estilización de la Frecuencia Fundamental (F0)

Este algoritmo de estilización de la F0 se realiza con el objetivo de obtener los movimientos principales de la entonación en el nivel de la frase y de cierto modo

corregir los valores de la F0 obtenidos a partir del algoritmo anterior basado en una función AMDF. Este algoritmo se realiza también para representar de una manera simple los movimientos y las tendencias del tono (Pitch), excepto las micro-oscilaciones presentes en el patrón del tono.

Seguidamente se hará una descripción detallada de los pasos del algoritmo:

1) Se calcula de F0 a través del algoritmo de estimación de la F0 usando una función AMDF explicado en el epígrafe anterior.
2) En los valores de F0 obtenidos se buscan los rangos de valores donde la F0 es diferente de cero que esto se corresponde con los segmentos sonoros detectados en la estimación de la F0. Estos valores se ponen en un vector.
3) Para cada uno de estos vectores de valores diferentes de cero determinados en el paso anterior se hace lo siguiente
    a. Se calculan los mínimos locales del vector que es donde se describen cambios bruscos en la monotonía de la curva F0.
    b. Se calculan los máximos locales del vector que es donde se describen cambios bruscos en la monotonía de la curva F0.
    c. Se ordenan dichos mínimos y máximos locales por su posición en el vector, obteniéndose un nuevo vector de extremos locales de la curva F0.
    d. Se toman los tres primeros puntos del vector de extremos y se construye la recta de interpolación que mejor los ajuste por el método de los mínimos cuadrados.
    e. Se evalúan los tres puntos en dicha recta para obtener los nuevos tres puntos en que se convirtieron los anteriores al ser interpolados que van a ser los valores de la F0 estilizada final.
    f. Se toma el próximo valor del vector y se calcula la distancia vertical del punto a la recta de interpolación.
    g. Se calcula la desviación estándar que provocaría el nuevo punto a la recta si ésta pasara por él o muy próximo.

h. Si la distancia del punto que se está analizando a la recta es menor que el doble de la desviación estándar del punto se dice que el punto pertenece a la recta, y entonces se une el punto actual a los otros que pertenecen a la recta de interpolación actual y se construye una nueva recta de interpolación entre todos los puntos y se va al **paso i**. Si la distancia del punto no es menor que el doble de la desviación estándar del punto se dice que el punto no pertenece a la recta y se pasa al **paso j**.

i. Una vez obtenida la nueva recta de interpolación se pasa al **paso f** si no se ha terminado de analizar todos los valores del vector de extremos. Si ya se terminaron de analizar todos los valores del vector extremos se pasa al **paso k**.

j. Se toma el último punto que pertenecía a la última recta, el punto actual y el próximo al actual (si es que tiene próximo) y construyo entre ellos una nueva recta de interpolación lineal y voy al **paso e**. En caso de que el punto que se estaba analizando era el último del intervalo entonces se toman el anterior que pertenecía a la recta y a él y se hace lo mismo.

k. Se pasa a analizar el próximo vector de rango de valores diferentes de cero de la F0 y se va al **paso a**. Si ya se acabaron los vectores se concluye el algoritmo y se va al **paso 4**.

4) Se construye la curva resultante de la estilización de la F0.
5) Fin del Algoritmo.

En el anexo 8 se muestra un diagrama de bloques de este algoritmo.

2.5.4- Algoritmo de construcción del contorno del Pitch

En este sub-epígrafe trataremos la implementación del algoritmo de construcción del contorno del Pitch con el objetivo de visualizar las variaciones del Pitch que describen la ritmicidad de la señal de voz.

Este algoritmo consiste en ir quitando de la curva de F0 los segmentos de valores que son ceros y realizar un enlace del último valor diferente de cero al próximo valor de F0 diferente de cero construyendo entre ellos una curva suave que mantenga el mismo comportamiento de la F0 original.

A continuación se describirán, de forma detallada, los pasos de dicho algoritmo en su implementación.

Pasos del algoritmo:

1) Primeramente se calcula la F0 a través del algoritmo de estimación de la F0, basado en una función AMDF, explicado anteriormente.
2) Se buscan las posiciones donde la F0 calculada sea cero y se colocan en un vector de posiciones.

   Nota: Los únicos ceros que no se tienen en cuenta para construir el enlace son los ceros del inicio de la curva F0 original ni los del final de la misma.
3) Los intervalos de valores diferentes de cero se mantienen con sus valores pero los que son ceros se someten a la siguiente secuencia de pasos.
4) Para cada intervalo de ceros de F0 encontrado hacer:
   a) Se toma el primer elemento cero del intervalo actual.
   b) Se toma el valor del último elemento diferente de cero que está en alguna posición anterior a la posición del elemento del intervalo actual de ceros que se está analizando.
   c) Se toma el valor del primer elemento diferente de cero que está en alguna posición posterior a la posición del elemento del intervalo actual de ceros que se está analizando.
   d) Se calcula la diferencia de dichos valores diferentes de cero en que se encuentre el elemento cero que se está analizando, y la diferencia de las posiciones de los dichos valores. Se haya el cociente entre estas diferencias.
   e) El valor que se pone en la posición del cero actual es la suma del último elemento diferente de cero que está en alguna posición anterior al cero actual y el cociente de diferencias hallado en el paso anterior.
   f) Se pasa a analizar el próximo cero del intervalo actual
5) Se grafica el contorno de la F0 hallado.

6) Fin del Algoritmo.

Se puede ver un diagrama de bloques que describe dicho algoritmo en el anexo 9.

2.5.5- Algoritmo de estimación de la intensidad de la señal de voz

El algoritmo de estimación de la intensidad de la señal de voz se basa en determinar la envolvente espectral de la energía, que es el parámetro físico que caracteriza a la intensidad.

El fundamento teórico en que se basa este algoritmo es el que se describió brevemente en el epígrafe *"Energía por tramas"* del capítulo anterior.

Los pasos de dicho algoritmo son los siguientes:

1) Se leen los valores de la señal.
2) La señal se debe dividir en tramas para analizarla en segmentos de este tipo.
3) Se calcula el número de elementos que se van a analizar en cada trama.
4) Se calcula el número de tramas en que se va a dividir la señal.
5) Para cada trama hacer:
    a) Tomar los elementos de la señal que pertenecen a la trama actual.
    b) Se calcula el contorno de la energía a través de la fórmula logarítmica descrita en el capítulo anterior.
    c) Se analizan los cruces por cero para respetarlos y no alterarlos.
    d) Se pasa a calcular la energía en la próxima trama.
6) Se grafica la envolvente de la energía de la señal calculada.
7) Fin del Algoritmo.

En el anexo 10 se muestra un diagrama de bloques del algoritmo para cumplimentar la explicación anterior.

Con este algoritmo se concluye el cálculo de los parámetros suprasegmentales que caracterizan la prosodia de la voz.

En el próximo epígrafe se explica un algoritmo de segmentación de la señal de voz en sílabas.

2.5.6- Algoritmo de segmentación de la señal de voz en sílabas

Este algoritmo se implementó en esta aplicación para que sea usado en trabajos futuros para determinar la tonicidad de las sílabas de la señal de voz, que es otro aspecto que requiere de un trabajo arduo, al igual que la segmentación.

Los aspectos teóricos que fundamentan los pasos para la realización de este algoritmo se resumieron en un epígrafe del capítulo pasado.

Este algoritmo, como se planteó anteriormente se basa en la búsqueda de los máximos espectrales de los cuales hay algunos que no se corresponden con límites silábicos por lo que hay que tratar de eliminarlos por otros criterios.

Los pasos del algoritmo son:

1) Se leen los datos del fichero *.wav que es donde está la grabación.
2) Se hallan los valores y la frecuencia de muestreo de la señal grabada.
3) Se calcula la longitud de las ventanas en que se va a dividir la señal para su análisis. En este caso es de 30 milisegundos.
4) La distancia entre ventanas contiguas es de 5 milisegundos.
5) Se realiza un enventanado, con ventanas Hamming, de la señal para realizar el análisis espectral. Se obtiene una matriz espectral cuyas filas se corresponden con los valores de la señal después del enventanado.
6) Se realiza un enventanado rectangular de la señal para determinar la energía de la misma.
7) Por cada fila de la matriz espectral se hace lo siguiente:
   a) Calcular matriz de *coeficientes cepstrales*[3].
   b) Coger 10 coeficientes cepstrales de la matriz cepstral.
   c) Se almacenan dichos valores en otra matriz.
   d) Tomar los valores de la ventana a procesar.
   e) Se le calcula la energía a los valores de dicha ventana. La energía se calcula como el promedio de las amplitudes al cuadrado.
   f) Pasar a la próxima fila de la matriz espectral.
8) Se calculan 10 coeficientes cepstrales por cada transición espectral.

[3] Coeficientes cepstrales son números que se utilizan para describir la envolvente espectral de la señal de voz.

9) Se filtran, la matriz de los coeficientes cepstrales y la energía calculadas, con un filtro paso bajo a 120 Hz con el objetivo de suavizar la envolvente de la energía y la de los coeficientes cepstrales para facilitar el trabajo de detección de máximos locales que se corresponden con los límites silábicos.
10) Se calculan los máximos y mínimos locales de la envolvente de la energía.
11) Dichos valores máximos y mínimos locales son considerados en un inicio posibles extremos silábicos.
12) Para determinar si estos valores extremos son verdaderamente extremos silábicos se someten los mismos a una serie de criterios.
13) Para cada uno de los posibles extremos hacer:
    - Si la distancia hasta los extremos contiguos es muy pequeña, menor que 1/100 ms, se elimina, entre él y los contiguos, el extremo de menor energía. La distancia mínima entre sílabas es de 1/100 ms como criterio de este trabajo respaldado en resultados prácticos del laboratorio de Procesamiento de Voz del CEETI.
    - Se determina el mínimo descenso de energía para determinar transiciones espectrales, el cual es de 88% del mayor valor de energía de la ventana que se está analizando. Los extremos que en su entorno haya un descenso de energía menor que el umbral anterior se desecha como de ser extremo.
    - Se pasa a analizar el próximo posible extremo.
14) Se grafica la envolvente de energía anteriormente calculada.
15) Después de analizar cada uno de los posibles extremos y determinado los verdaderos extremos silábicos se pasa a señalarlos, a través de una línea vertical de color rojo en la gráfica de la envolvente de la energía anteriormente calculada y graficada.
16) Para corroborar si se segmentó la señal en sílaba se puso un botón en la ventana que muestra el resultado del algoritmo, el cual, cada vez que se pulse, se reproduce una de las sílabas detectadas en orden de aparición, y mientras esto ocurre, las anteriores líneas verticales de color rojo se pintan de color azul para señalar la sílaba que se ha reprodujo.
17) Fin del algoritmo.

Para cumplimentar el entendimiento del algoritmo se realizó un diagrama de bloques que se muestra en el anexo 11.

2.6- Conclusiones del capítulo

En este capítulo se realizó un análisis detallado de la arquitectura del sistema implementado, en función del flujo de trabajo e información a lo largo de las fases del proceso de desarrollo de software a través de los diferentes diagramas descritos en UML. Se realizaron los diagramas siguientes:

- Diagramas de Casos de Uso del sistema y se refinaron los más importantes. Este diagrama se hizo con el objetivo de describir cada una de las funcionalidades que el sistema le brinda al usuario.
- Diagrama de Transición de Estados para cada uno de los casos de uso del sistema que describe los estados por los que pasa cada uno de ellos.
- Diagrama de Despliegue que brinda la información de qué dispositivo(s) de hardware necesita el sistema para su funcionamiento. En este caso el sistema solo necesita de una computadora que tenga instalado el entorno *Matlab*, y el sistema multimedia en buen estado.
- Diagrama de Componentes donde se describen los componentes de software que conforman el sistema y como se interrelacionan entre sí.

Por último se realizó un modelo de implementación del sistema donde se enfatizó en la estructura detallada de cada uno de los algoritmos tales como:

- ✓ Algoritmo de estimación de la F0.
- ✓ Algoritmo de estilización de la F0.
- ✓ Algoritmo de construcción del contorno del Pitch.
- ✓ Algoritmo de estimación de la energía.
- ✓ Algoritmo de segmentación de la señal de voz en sílabas.

Se detalló en los pasos y se realizó un diagrama de bloques para cada uno de los algoritmos anteriormente citados.

Los cuales fueron implementados en el lenguaje *Matlab* porque brinda muchas herramientas para el procesamiento de señales digitales de cualquier tipo, incluidas las señales de voz.

Capítulo 3: Manual de usuario. Análisis de los resultados

Este capítulo tiene como objetivo realizar una explicación detallada de como trabajar con la aplicación en Matlab, *APA*, que implementa los algoritmos anteriormente explicados, para así facilitarle el trabajo al usuario del software.

*3.1- Diálogo de apertura del fichero *.wav*

Primeramente se abre el ambiente de programación *Matlab*. Luego se ejecuta el fichero "apa.m" por la línea de comandos del ambiente. Una vez ejecutado dicho fichero se muestra una ventana de diálogo para seleccionar el fichero "*.wav" a analizar:

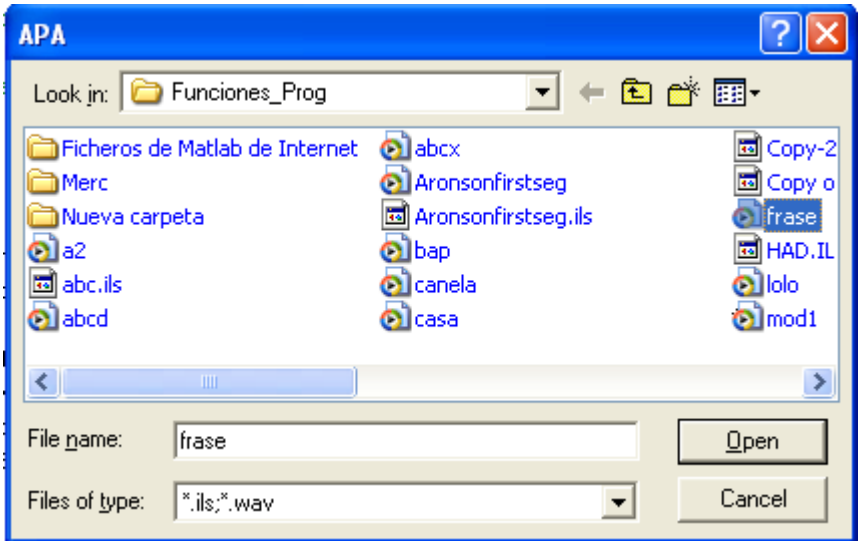


Figura 3.1: Diálogo de apertura de ficheros *.wav.

Una vez elegido el fichero *.wav, se presiona el botón *Open* para abrirlo, mostrándose la interfaz principal de la aplicación.

3.2- Interfaz principal de la aplicación

La interfaz principal de la aplicación es la que se muestra a continuación.

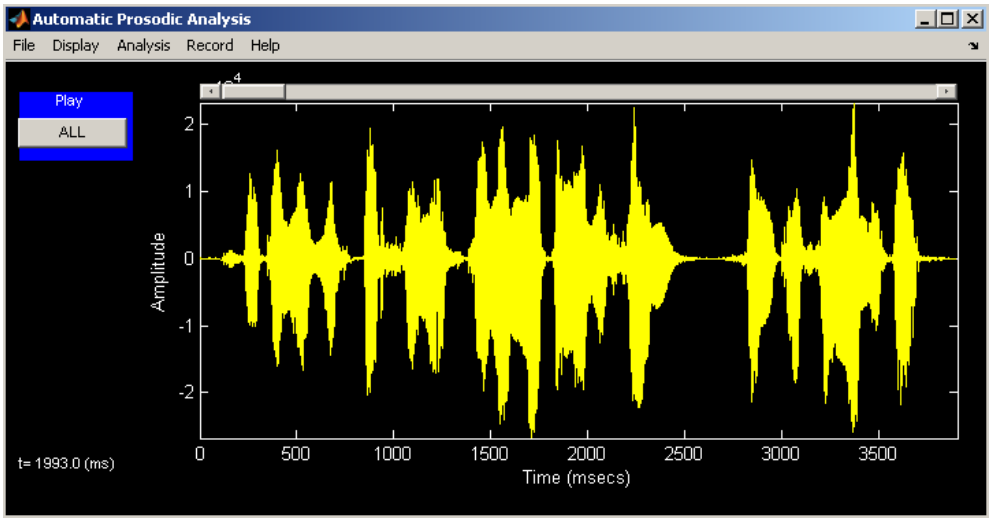


Figura 3.2: Interfaz principal de la aplicación.

En dicha interfaz aparece 1 botón, un menú de opciones y una gráfica que muestra la señal de voz que se está procesando actualmente. Dicha gráfica es de amplitud de la señal en el dominio del tiempo.

3.3- Gráfica de la señal grabada

En la gráfica de la señal grabada se muestra la señal graficada en el dominio del tiempo: amplitud (*amplitude*) contra tiempo (*time*) donde la escala de la amplitud es de $1*10^4$, y la del tiempo es de $5*10^2 msecs$ (milisegundos).

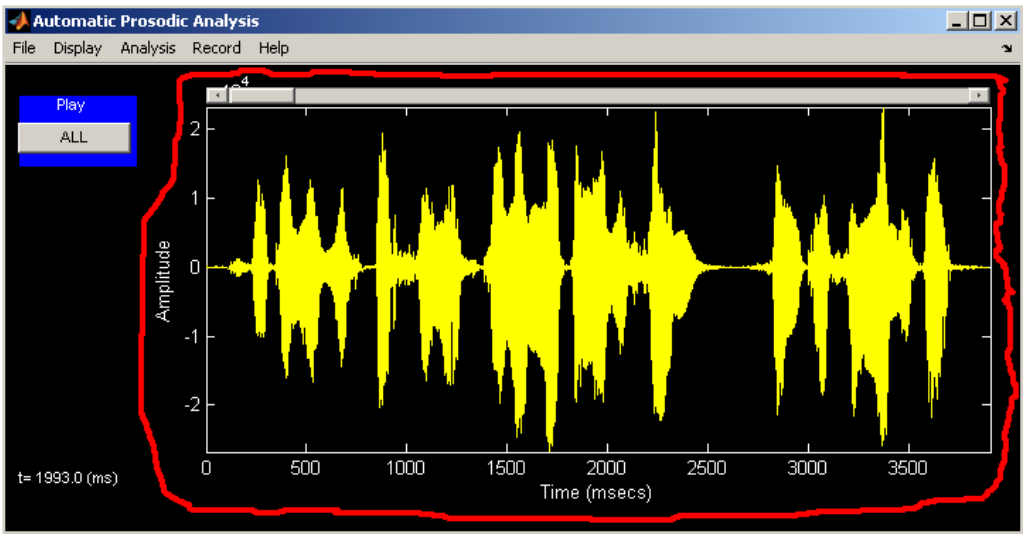


Figure 3.3: Gráfica de la señal de voz grabada.

3.4- Botón de la interfaz principal

Si se presiona el botón ***ALL*** se hace una reproducción de toda la señal de voz grabada.

En la esquina inferior izquierda de la aplicación se muestra el tiempo en milisegundos (ms) de la señal de voz que se corresponde con la posición del *mouse* en la gráfica de la señal, en el eje del tiempo.

3.5- Opciones del menú

En el menú de la aplicación existen varias opciones con nombres que identifican, con breves palabras, la funcionalidad de las mismas.

Las opciones son:

- File
- Display
- Analysis
- Record
- Help

3.5.1- Menú File

En la opción *"File"* del menú se despliegan las opciones que se muestran a continuación

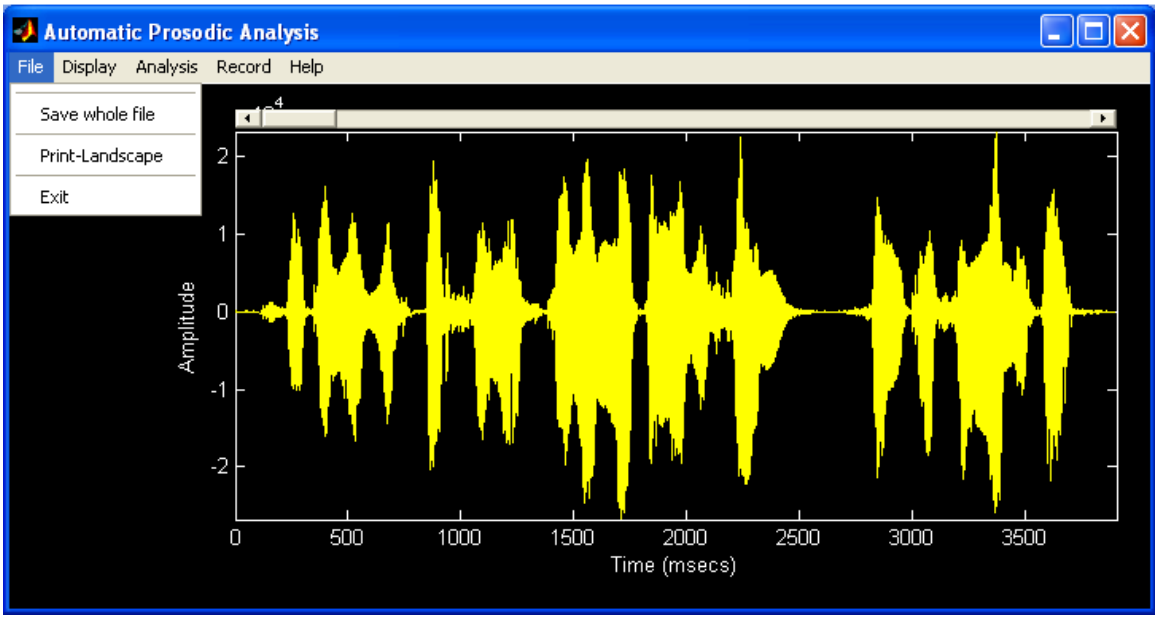


Figura 3.5.1: Opciones del menú *"File"*

3.5.1.1- Opción "Save whole file"

Se utiliza para salvar el fichero que se está procesando, en su conjunto, como un *"*.wav"*.

Cuando se da clic sobre esta opción del menú "File" se muestra la ventana de diálogo de la figura 3.5.1.1 que se muestra a continuación.

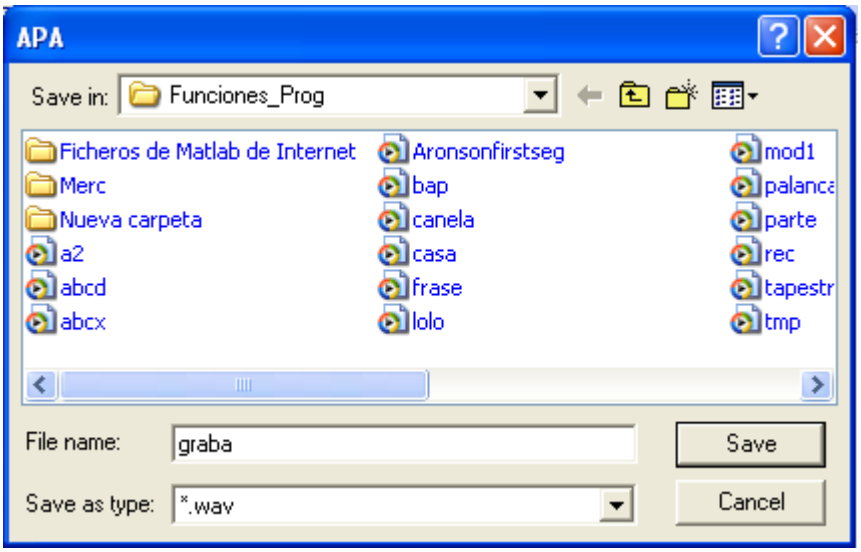


Figura 3.5.1.1: Diálogo para salvar fichero

3.5.1.2- Opción "Print-Landscape"

Con esta opción se envía para la impresora local un fichero con la imagen de la señal de voz grabada que se está procesando.

Primeramente se crea un fichero .mdi (*Microsoft Document Imaging*) con un nombre y camino que se les entran en un diálogo, como indica la figura 3.5.1.3, y se pinta dentro de dicho fichero la gráfica de la señal.

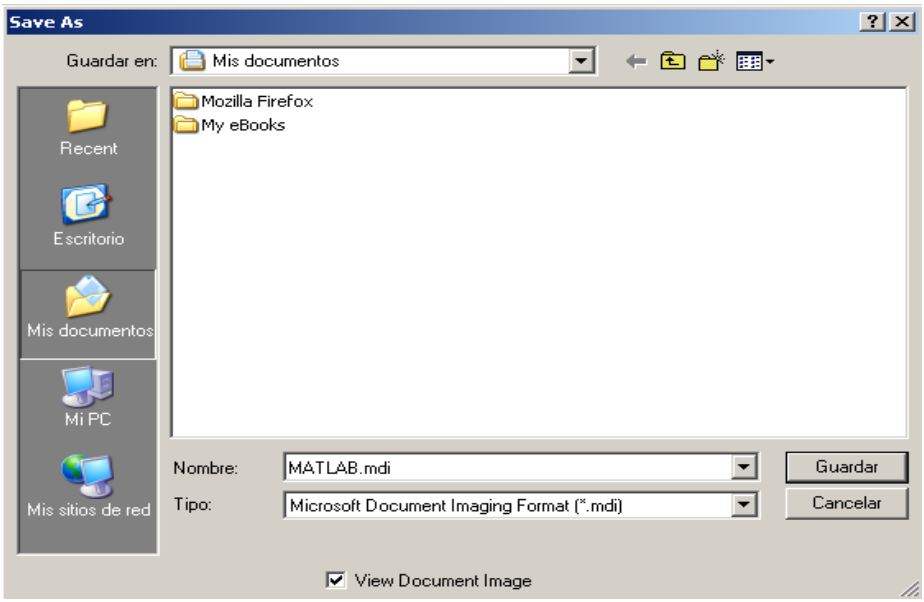


Figura 3.5.1.3: Ventana de diálogo para salvar el fichero .mdi que se va a imprimir.

Cuando se salva el fichero se muestra automáticamente el fichero con la imagen dentro. Entonces desde este fichero se envía para la impresora para imprimirlo.

3.5.1.4- Opción "Exit"

Nos brinda la posibilidad de abortar la ejecución de la aplicación, independientemente del estado en que se encuentre la misma.

3.5.2- Menú Display

En el menú *"Display"* se brindan las siguientes opciones:

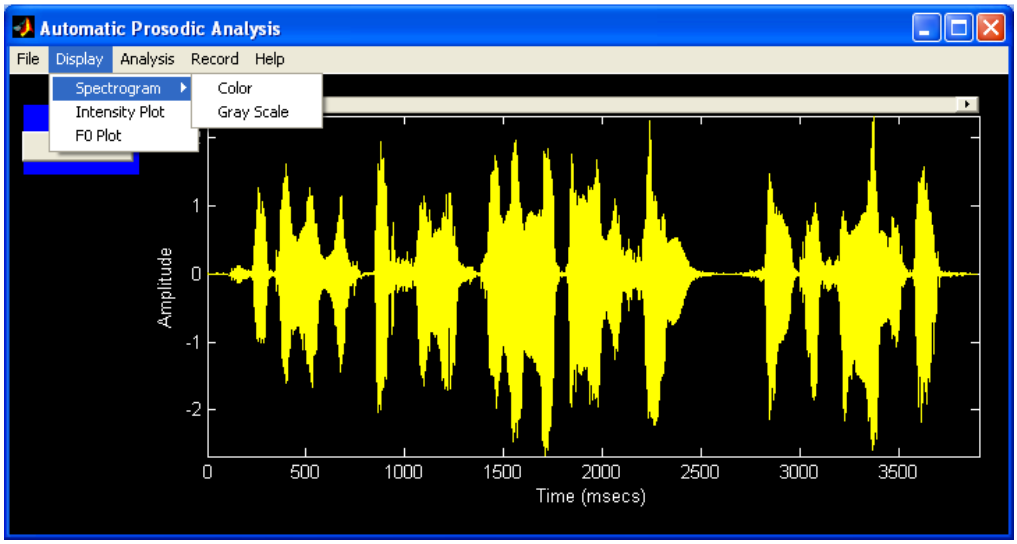


Figura 3.5.2: Opciones del menú *"Display"*.

Aquí se brindan las opciones mostradas en la figura 3.5.2.

3.5.2.1- Opción "Spectrogram"

El algoritmo que calcula la señal se tomó del software internacional COLEA, el cual se hizo con el objetivo de hacer análisis de voces analizando los mismos parámetros que los que se analizan en esta aplicación.

Sirve para construir el espectrograma de la señal de voz en escala de colores con la opción *"Color"*, o en escala de grises con la opción *"Gray Scale"*.

3.5.2.1.1- "Spectrogram" Color

Como se dijo anteriormente con esta opción se construye el espectrograma con una escala de colores como se muestra a continuación:

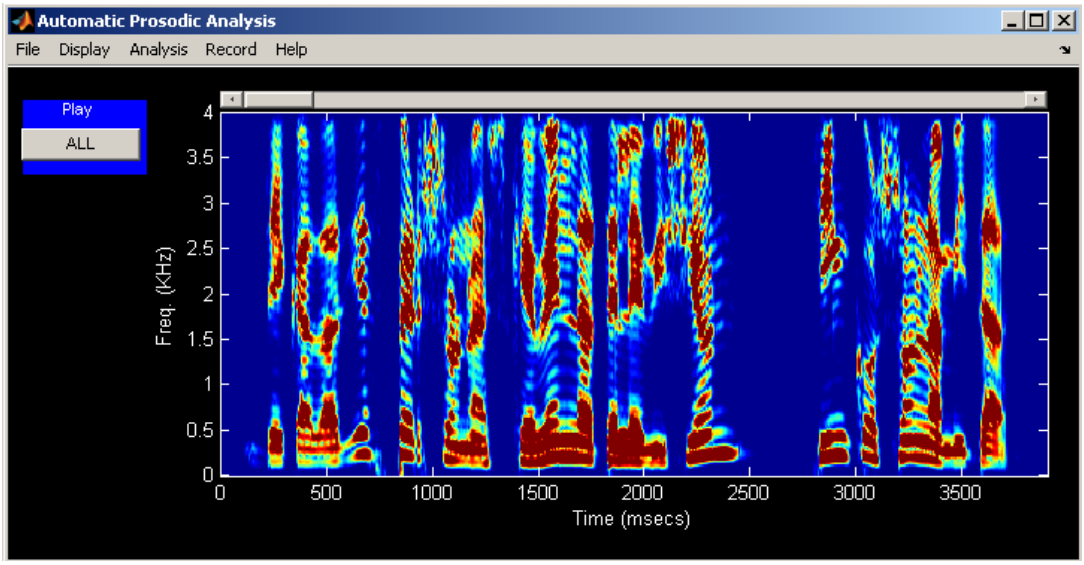


Figura 3.5.2.1.1: Espectrograma en escala de colores de una señal de voz.

3.5.2.1.2- "Spectrogram" Gray Scale

Se muestra la construcción de un espectrograma en escala de grises.

A continuación se presenta un ejemplo de un espectrograma construido en escala de grises.

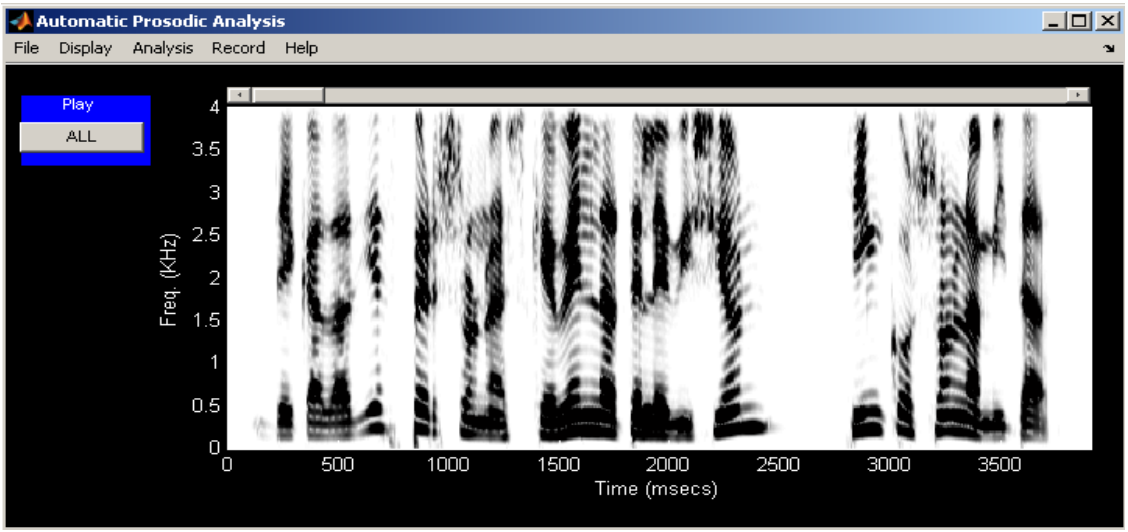


Figure 3.5.2.1.2: Espectrograma en escala de grises de una señal de voz.

3.5.2.2- Opción Intensity Plot

Se realiza una estimación de la energía por el método descrito en capítulos anteriores, obteniéndose una medida de la entonación, y se plotean los resultados dando lugar a la gráfica que se muestra a continuación.

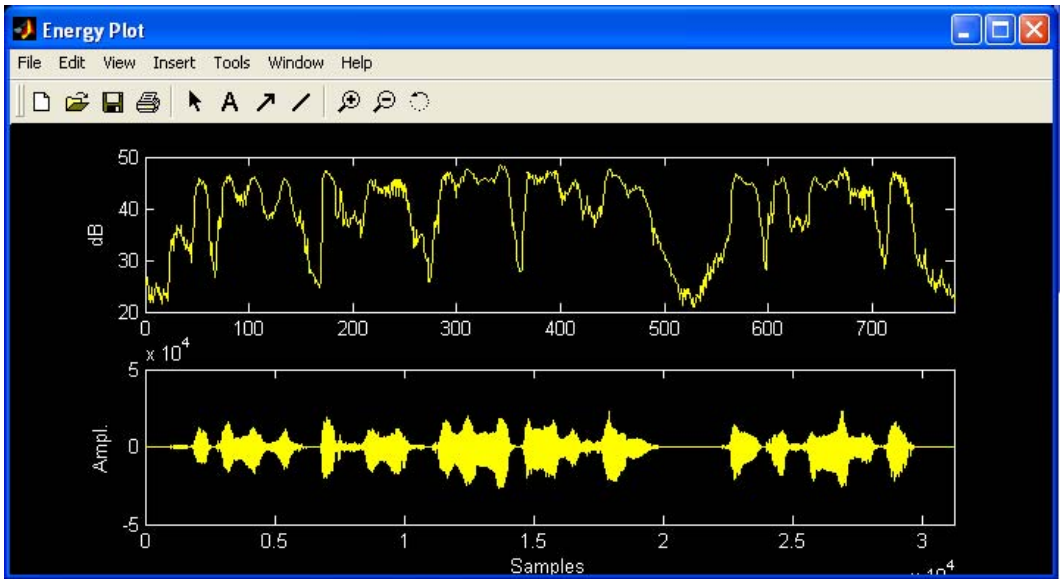


Figura 3.5.2.2: Energía de la señal de voz.

3.5.2.3- Opción F0 Plot

Aquí se realiza una estimación de la frecuencia fundamental (F0) o Pitch de la señal de voz, de forma automática, por el método de una función *AMDF* a partir de la cual se calcula la media de las diferencias de amplitudes de vectores de señal del mismo tamaño.

El gráfico de la figura 3.5.2.4, que muestra la F0 nos brinda una medida de la entonación de la señal de voz.

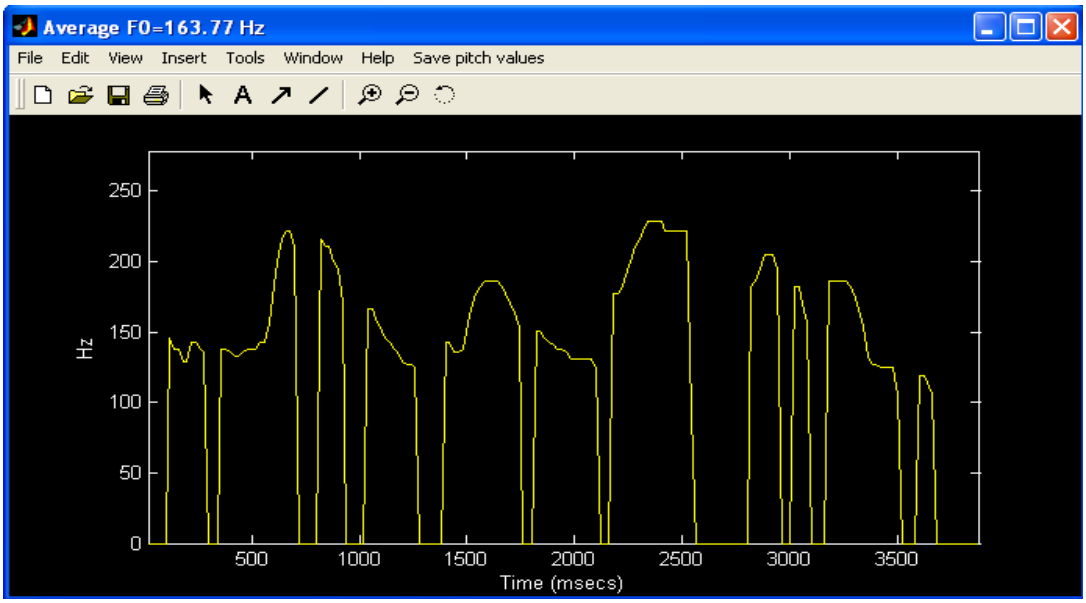


Figura 3.5.2.3: Gráfica de la frecuencia fundamental de la señal de voz

3.5.3- Menú Analysis

En este menú se realiza un análisis de los parámetros suprasegmentales de la señal de voz: entonación y ritmo. También se realiza una segmentación en sílabas basado en el algoritmo de las transiciones espectrales que en los capítulos anteriores se ha descrito rigurosamente.

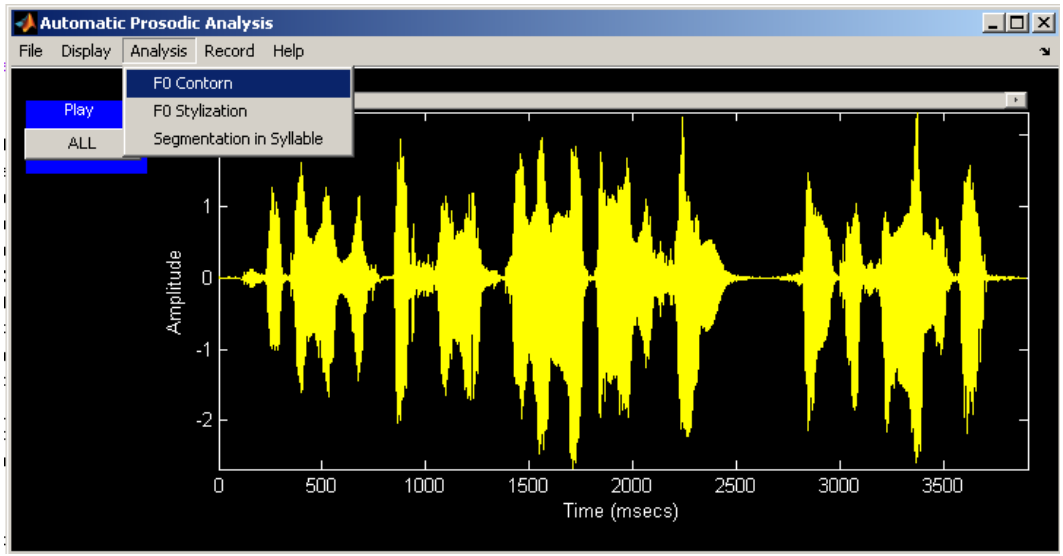


Figura 3.5.3: Menú *Analysis* con todas sus opciones.

Dichas opciones se van a explicar detalladamente en los siguientes subepígrafes.

3.5.3.1- Opción F0 Contorn

En esta opción se determina el contorno del Pitch o F0 a partir de la *F0 Plot* anteriormente explicada con el objetivo de obtener una gráfica de la entonación en toda la señal.

Un ejemplo se muestra en la figura 3.5.3.1.

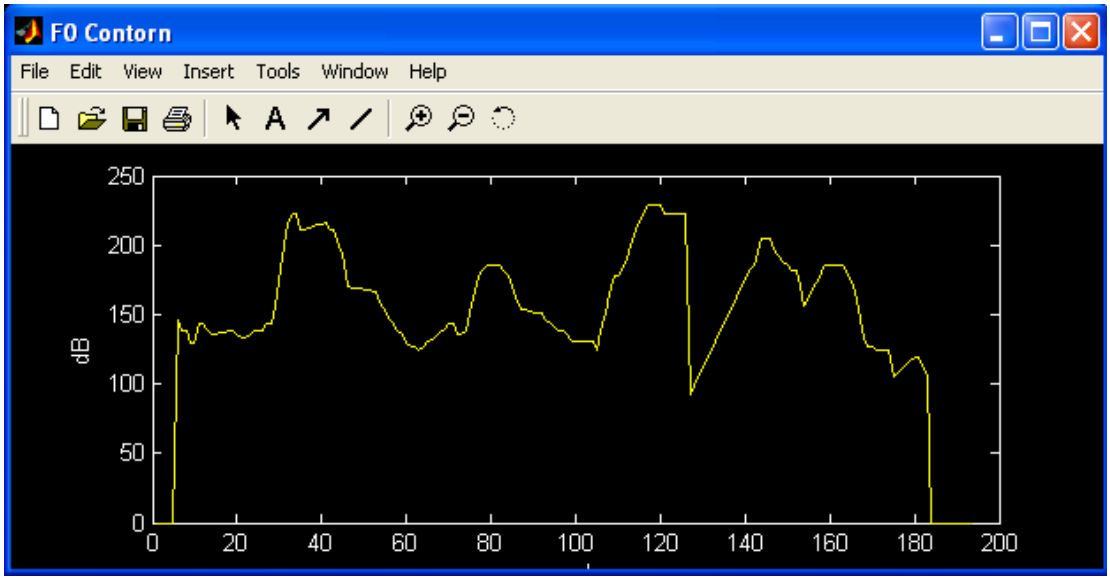


Figura 3.5.3.1: Contorno de la F0 como una medida de la entonación.

3.5.3.2- Opción F0 Stylization

Aquí se realiza una estilización de la frecuencia fundamental con el objetivo de obtener los movimientos más relevantes de la entonación en el nivel de la frase.

A continuación se muestra un ejemplo del resultado de este algoritmo.

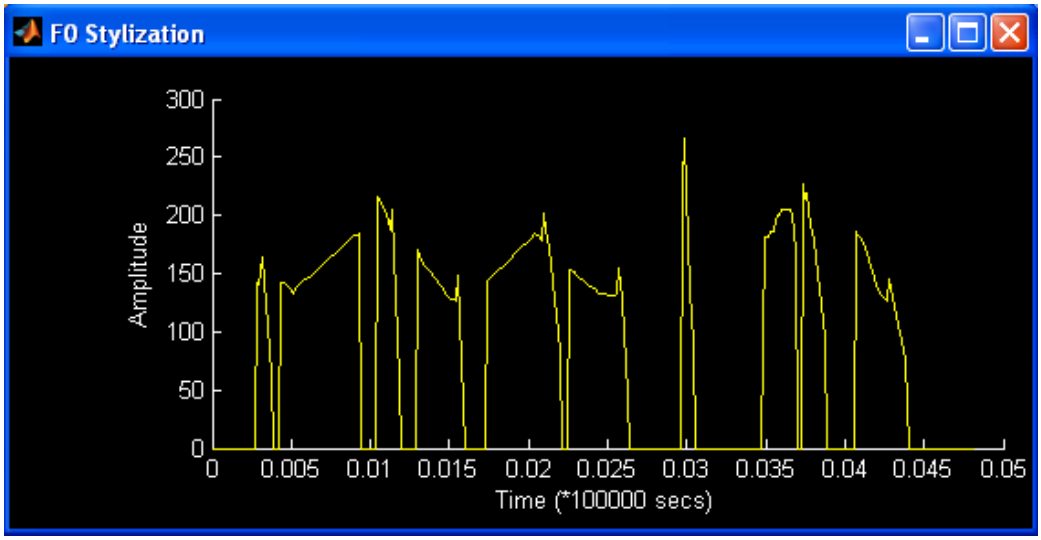


Figura 3.5.3.2: Ejemplo de estilización de la F0 en una frase

3.5.3.3- Opción Segmentation in Syllable

Se realiza la segmentación en sílabas de la señal de voz que se está analizando a través del algoritmo explicado en los capítulos anteriores.

Un ejemplo de como queda la segmentación en sílabas de la señal.

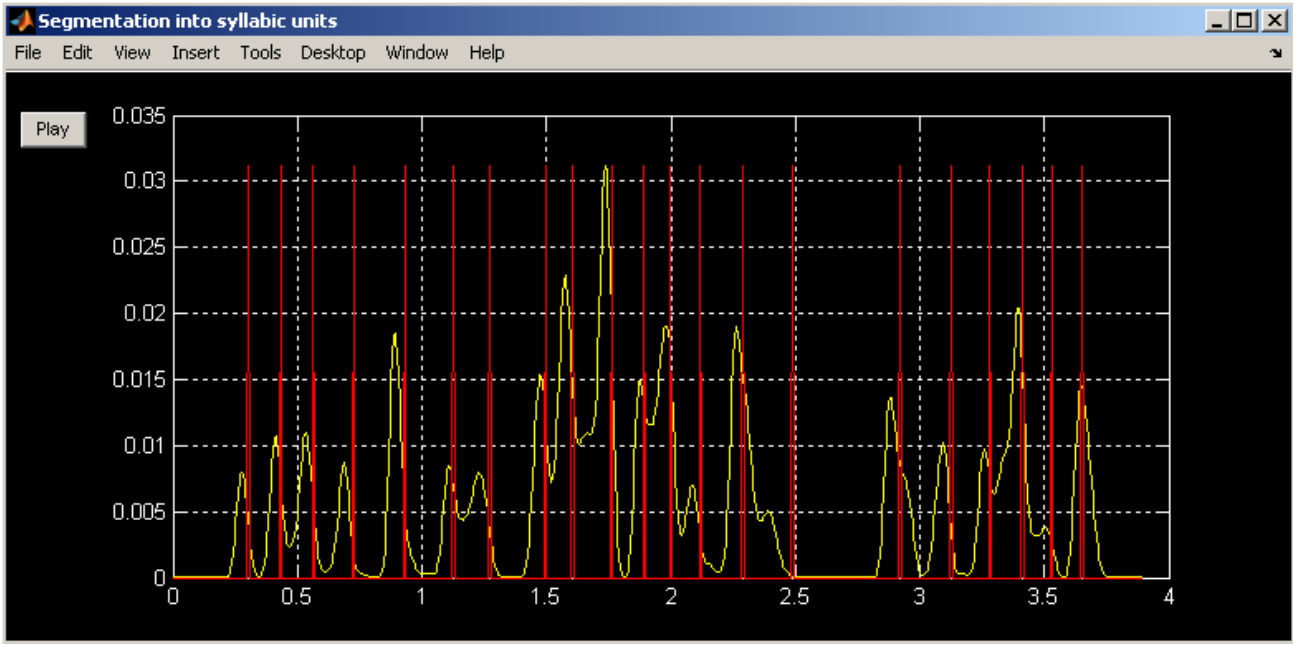


Figura 3.5.3.3 a): Ejemplo del segmentado en sílaba de la señal de voz

Donde la gráfica de color amarillo representa las transiciones espectrales de la señal de voz.

Los segmentos en rojo representan los posibles extremos silábicos calculados a través del algoritmo de segmentación anteriormente explicado.

El botón que aparece en la esquina superior izquierda que tiene como etiqueta *"Play"* es para reproducir los segmentos de sílabas determinados por el algoritmo lo que sirve para chequear si el algoritmo realizó una adecuada segmentación.

Cuando se presiona el botón *"Play"*, mientras se vaya reproduciendo la sílaba se selecciona la misma, y las anteriores, con un segmento de color azul. Un ejemplo de ello se muestra a continuación.

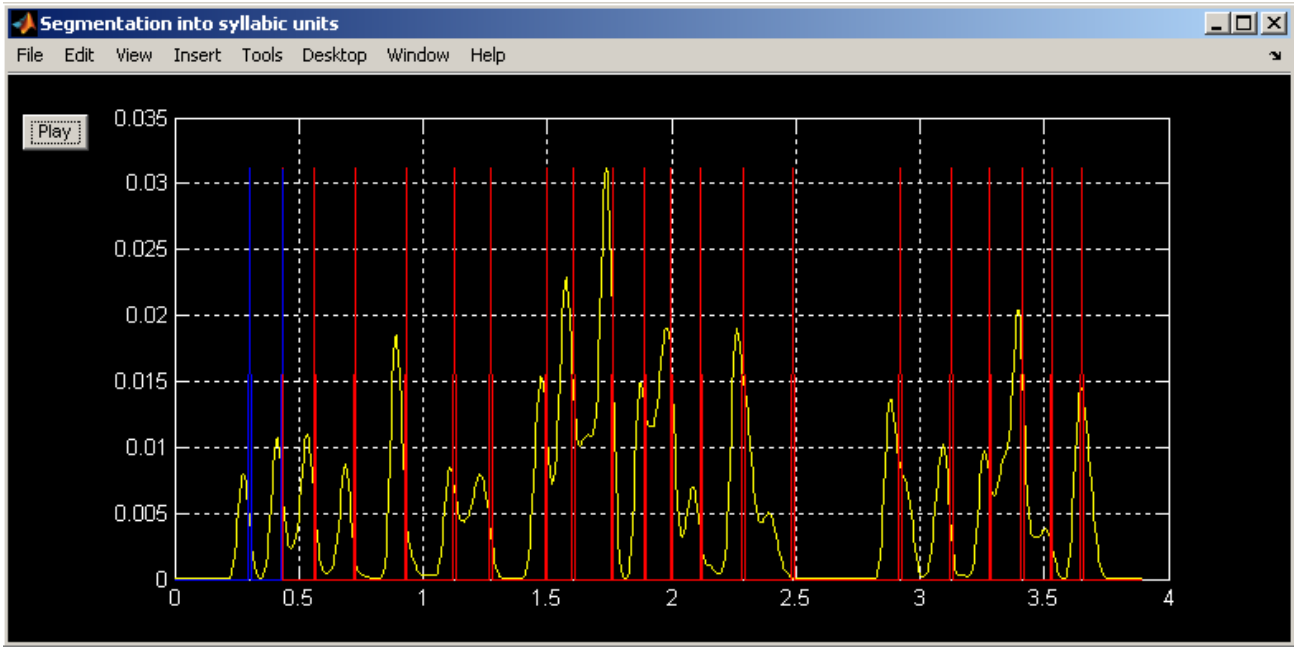


Figura 3.5.3.3 b): Ejemplo de reproducción de las sílabas obtenidas por el segmentado.

3.5.4- Menú Record

En este menú sirve para realizar una grabación en tiempo real a través de un micrófono conectado al ordenador para que el usuario hable a través de él y se

grabe en el sistema con unos parámetros configurables por el mismo usuario, necesario para poder reproducir la grabación.

Las opciones configurables por el usuario se muestran en una ventana que aparece cuando se selecciona la opción *"Record"* del menú.

La ventana se muestra a continuación:

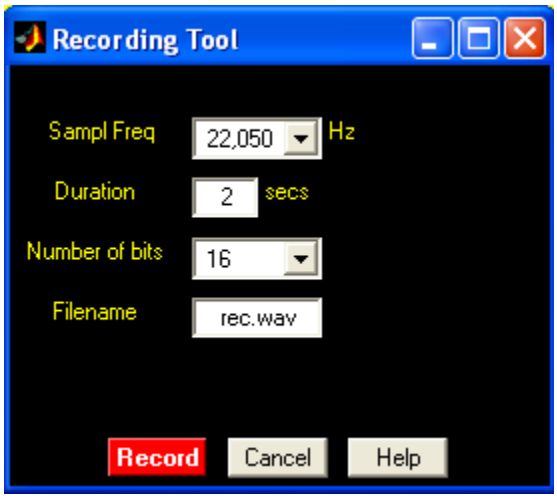


Figura 3.5.4: Opciones de configuración para realizar una grabación.

Las opciones son las siguientes:

- *Sampl Freq*: Frecuencia de muestreo en *Hz* con que se va a realizar la grabación: 8000, 11025, 16000, 22050 y 44100.
- *Duration*: Tiempo en segundos que va a durar la grabación.
- *Number of bits*: Número de bits de resolución de la grabación.
- *Filename*: Nombre del fichero donde se va a salvar la grabación en el directorio de trabajo.

Cuando ya se tenga lista la configuración de las opciones, se presiona el botón:

- *Record*: Para comenzar el proceso de grabación.
- *Cancel*: Para cancelar la grabación

- *Help*: Para recurrir a la ayuda.

3.5.5- Menú Help

Muestra una ventana de ayuda como la de la figura 3.5.5 donde se brinda una descripción sencilla y detallada de cada una de las opciones de *APA* y la ayuda de *Matlab* también la brinda. Todas en idioma inglés.

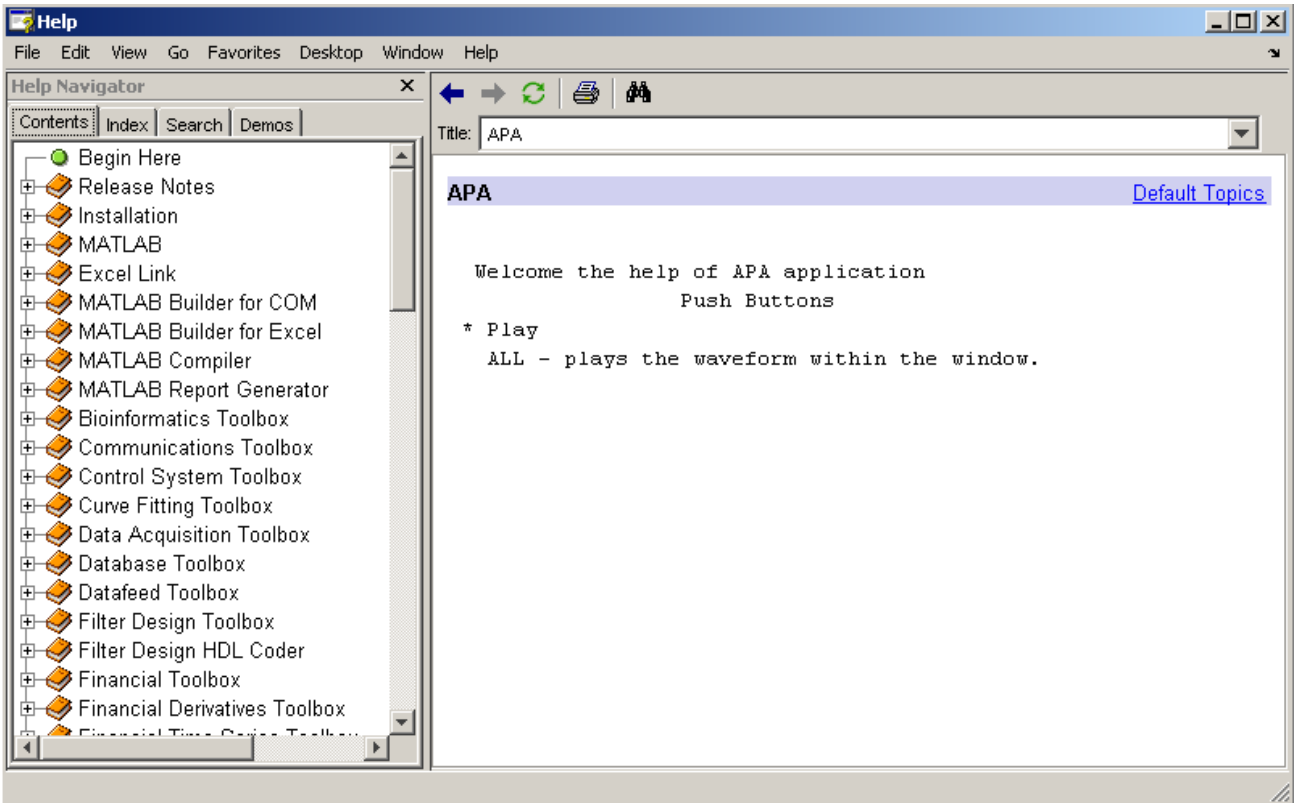


Figura 3.5.5: Ventana de ayuda de *APA*.

3.6- Análisis de los resultados de los algoritmos implementados

La validación de la calidad de los resultados de los algoritmos implementados no es una tarea sencilla, pues en primer lugar surgen las interrogantes: ¿cómo se mide la calidad?, ¿cuándo la calidad es buena?, ¿cuál es la frontera entre la buena calidad y la mala calidad, entre lo aceptable y lo no aceptable?, ¿cómo se toma esta decisión?, ¿quién toma esta decisión?

Como el análisis de los componentes de la voz es algo que no es exacto debido a la naturaleza misma de la voz que es dependiente del hablante, éste se define como un fenómeno en gran medida subjetivo. La decisión final de la calidad de la estimación de cada uno de dichos componentes está sujeta a la experiencia, la visión del tema que tengan el analista y el probador del software.

Por todo lo anteriormente planteado, en este trabajo, se emplearon diferentes vías de validación de los resultados tanto desde puntos de vistas objetivos como subjetivos.

La principal técnica de validación utilizada fue la comparación con los resultados de otros softwares equivalentes que estuvieran validados y que se utilizaran mucho en el mundo entero por la calidad de sus resultados.

3.6.1- Análisis de los resultados del algoritmo de estimación de la F0

Los resultados del algoritmo de estimación de la F0 basado en una función AMDF implementado en esta investigación condujo a que se concluyera de que son de calidad aceptable pues se analizaron en comparación con el software COLEA, realizado por Philipos C. Loizou en el año 1998 en Estados Unidos. Philipos implementó en dicho software un algoritmo que realiza una estimación de la F0 por el método de autocorrelación.

Se escogió este software porque está implementado en el lenguaje Matlab, al igual que la aplicación presente en esta investigación, lo que facilita el proceso de comparación; es muy usado internacionalmente por sus resultados de gran precisión.

Una de las principales diferencias objetivas que se evidenciaron entre los dos algoritmos de extracción de F0, el implementado en el COLEA y el del presente

trabajo, es que el primero no brinda la posibilidad al usuario de configurarle parámetros de entrada que se consideran no definidos de forma exacta, sino que lo calcula internamente por aproximaciones de cálculos matemáticos, y el que aquí se implementa brinda la posibilidad de configurarlos a estimación del usuario lo que amplía grandemente su aplicabilidad en diferentes contextos de grabaciones; tales parámetros son: la cantidad de tramas en que se va a dividir la señal para realizar un mejor análisis de la F0, los umbrales mínimos y máximos de detección de frecuencias válidas a tener en cuenta para estimar los valores del Pitch, el tamaño de las ventanas de análisis y el solapamiento entre ventanas contiguas.

Se realizaron diferentes pruebas de análisis de grabaciones en los dos algoritmos que mostraron los siguientes resultados:

Frase	**Algoritmos a comparar**	**Error cuadrático medio**	**# de Picos**
a e i o u	ESTF0 del COLEA	5.5421	4
	GrossF0Detector de APA		3
De cada diez personas que ven la televisión cinco son la mitad.	ESTF0 del COLEA	8.3710	3
	GrossF0Detector de APA		2
Canela	ESTF0 del COLEA	2.4583	2
	GrossF0Detector de APA		3

¿Hay helado en el coopelia?	ESTF0 del COLEA	3.0934	2
	GrossF0Detector de APA		4

Tabla 3.6.1: Errores y fallos de los algoritmos ESTF0 y GrossF0Detector en la estimación de la F0 en algunas grabaciones.

El número de picos indica que el algoritmo tuvo un fallo a la hora de estimar el Pitch en ese punto por lo que se considera una deficiencia del mismo. En las siguientes figuras se muestran los resultados de aplicar los algoritmos anteriores a la segunda grabación de la tabla anterior.

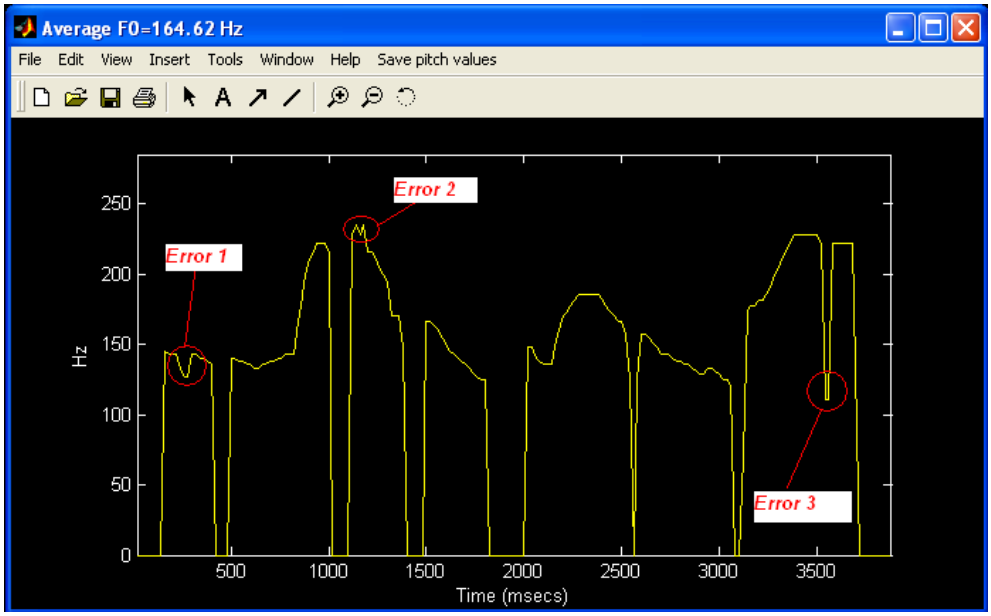


Figura 3.6.1ª): Resultado de aplicar el algoritmo ESTF0 del software COLEA a la segunda grabación de la tabla anterior.

En dicha figura se muestran los tres errores que se pueden ver a simple vista en que incurre el algoritmo ya que en la detección de la F0 no pueden existir picos ni huecos abruptos, según la bibliografía consultada.

A continuación se muestra el comportamiento del otro algoritmo (el implementado en esta investigación) en la detección de F0 en la misma grabación.

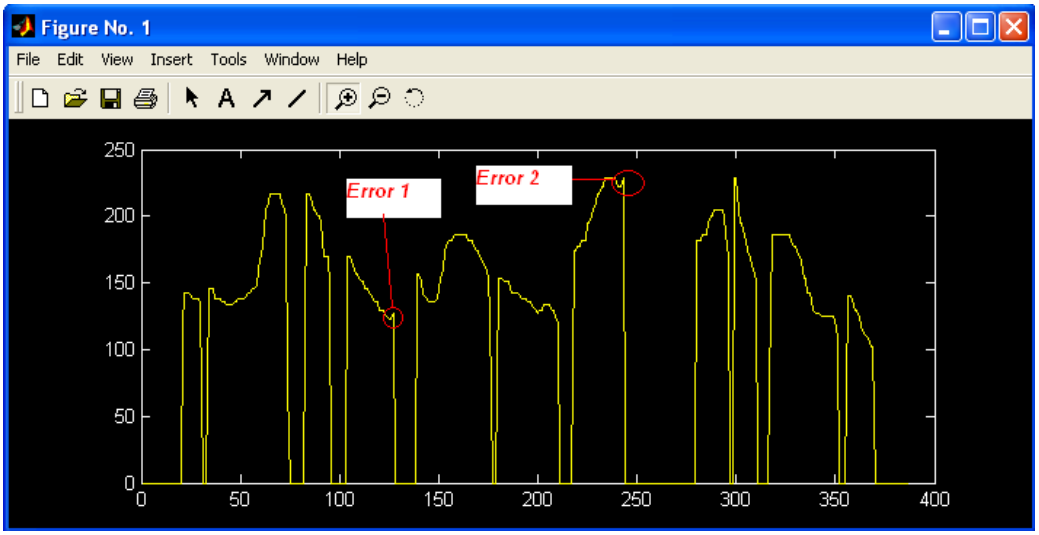


Figura 3.6.1[b]): Resultado de aplicar el algoritmo GrossF0Detector a la segunda grabación de la tabla anterior.

De la tabla 3.6.1, donde se comparan los dos algoritmos antes mencionados, se concluye de que el algoritmo GrossF0Detector es tan bueno y eficiente como el ESTF0 del COLEA, y en ocasiones se comporta de forma más eficiente debido a que el GrossF0Detector realiza el análisis de forma más precisa teniendo en cuenta más información de la señal que el ESTF0. Hay solamente diferencias sutiles en la estimación de la F0 entre los dos algoritmos, lo cual demuestra que los dos obtienen soluciones suficientemente buenas.

3.6.2- Análisis de los resultados del algoritmo de estimación del contorno del Pitch

Para realizar el análisis de los resultados de este algoritmo se le aplicaron pruebas al algoritmo implementado y se compararon sus resultados con los del algoritmo que el COLEA implementa con este fin. Dichas pruebas enfatizaron que el implementado en esta investigación aborda mejores resultados ya que realiza un chequeo más profundo acerca de los movimientos más relevantes de la F0. Cuando existe una parte donde la F0 es nula se realiza una interpolación con el siguiente contorno de F0 no nula a través de una curva que describe una trayectoria lineal que suaviza la envolvente del Pitch y no descarta ni modifica los valores no nulos.

Como este algoritmo está basado en la extracción de la F0, entonces, con el algoritmo ESTF0 del COLEA se incurren en los mismos errores que se mostraron en la tabla 3.6.1 frente al GrossF0Detector de APA, por lo que no es necesario volverlos a mostrar. Adicionalmente a estos errores, se comenten otros a la hora de realizar la interpolación para unir segmentos sonoros contiguos, los cuales son insignificantes.

3.6.3- Análisis de los resultados del algoritmo de estilización de la F0

El algoritmo de estilización de la frecuencia fundamental de la señal de voz está basado en el cálculo de la F0 y luego se interpolan dichos valores con el objetivo de descartar los menos significativos, los que no aportan información relevante y dejando solamente los imprescindibles de tal forma que se mantenga invariante la forma de la envolvente espectral de la misma. La estilización se realizó como se explicó en el capítulo 2 y los resultados que se obtuvieron al ponerlo en práctica con varias grabaciones de voz se consideran aceptables.

Para el análisis de dichos resultados se realizó una comparación con el software *Praat*, validado internacionalmente y muy usado para estos fines, con el objetivo de verificar cuan buenos son.

Los resultados de dicha comparación se muestran en la siguiente tabla.

Frase	Algoritmos	Error cuadrático medio	% de aciertos
a e i o u	praat	2.0439	97.9661
	F0Estiliza	4.6702	92.0221
De cada diez personas que ven la televisión cinco son la mitad	praat	3.1820	96.8180
	F0Estiliza	10.5689	80.3166
Canela	praat	1.0230	98.6731
	F0Estiliza	5.0209	88.5000
¿Hay helado en el coopelia?	praat	3.0934	96.7726
	F0Estiliza	7.9032	85.0972

Tabla 3.6.3: Errores y por ciento de aceptación en la estilización de la F0 en grabaciones.

Los resultados anteriores demuestran que el algoritmo que implementa el *Praat* para realizar una estilización de la frecuencia fundamental obtiene estimados de mayor calidad que el implementado en esta aplicación, lo cual no lo desecha sino que se propone a que se mejore la calidad de sus resultados. Los errores se deben a la forma de estilización usada en este algoritmo, se debe realizar tomando una nueva estimación para poder decir que un punto es significativo o no.

3.6.4- Análisis de los resultados del algoritmo de segmentación

El algoritmo de segmentación de la señal de voz en sílabas se validó desde el punto de vista de analizar si realiza la segmentación automática de forma correcta en frases grabadas cuya segmentación se conoce de antemano y que no tienen una larga duración, por lo que se realiza una segmentación manual y se cuentan los extremos detectados. Y para corroborar si dichos extremos fueron detectados de forma correcta se escucha la grabación en la misma aplicación a través del cliqueo de un botón que las reproduce una por una de extremo a extremo según el segmentado.

Seguidamente se muestra, en la tabla 3.6ª), el análisis de los resultados en grabaciones que son frases dichas por interlocutores que no son expertos y que se eligieron de forma aleatoria, en diversas condiciones de laboratorio lo cual influye negativamente en el desempeño de dicho algoritmo pues en la grabación se introduce ruido que tiende a confundirlo. Solo la primera grabación fue dicha por un experto.

No se utilizaron voces de expertos porque no se pudo encontrar una base de datos disponible que porte grabaciones de este tipo.

Frase	**Sílabas detectadas**	**Sílabas de forma automática**		**% Aciertos**
	Manual	**Bien**	**Mal**	
De cada diez personas que ven la televisión cinco son la mitad.	20	19	1	**95%**
¡Ay mi madre, se me fue la guagua!	10	7	3	**70%**
La historia de Haití es muy interesante	14	12	2	**85.7%**
¿Te gusta el helado de chocolate?	12	12	0	**100%**
Fui a comprar plátanos	7	6	1	**85.7%**
La historia de Haití es muy complicada.	13	12	1	**92.3%**
Procesamiento de señales.	9	8	1	**88.8%**
Dame una palanca y moveré el mundo.	13	10	3	**76.9%**
Promedio de aciertos				**86.8%**

Tabla 3.6ª): Resultado del segmentado automático de frases dichas por interlocutores comunes.

A continuación, en la tabla 3.6[b]), se muestran los resultados del segmentado de palabras aisladas dichas por interlocutores que tampoco son expertos y que se eligieron de forma aleatoria, en condiciones de laboratorio no idóneas.

Frase	**Sílabas detectadas**	**Sílabas de forma automática**		**% Aciertos**
	Manual	**Bien**	**Mal**	
Canela	3	3	0	**100%**
Casa	2	1	1	**50%**
Citación	3	3	0	**100%**
Palanca	3	2	1	**50%**
Lolo	2	2	0	**100%**
Mesa	2	1	1	**50%**
Promedio de aciertos				**75%**

Tabla 3.6[b]): Resultado del segmentado automático de palabras dichas por interlocutores comunes.

En las tablas anteriores se evidencia de que ante frases el algoritmo funciona mejor, con un por ciento de aciertos de un 86.8%, que ante palabras aisladas con un 75% de aciertos.

Se detectaron problemas en el desempeño del algoritmo en el segmentado ante cualquiera de los dos tipos de grabaciones los cuales se citan a continuación de acuerdo a las sílabas mal detectadas.

Esencialmente presenta problemas en la detección de las sílabas que contienen los siguientes fonemas los cuales los detecta como sílabas aisladas o en otras:

- Fonemas fricativos tales como: s, ch, z.
- Fonemas nasales: n, m.

3.7- Conclusiones parciales

En este capítulo se realizó una explicación detallada de las posibilidades que brinda el software *"APA"*. Se elaboró un manual de usuario con todas las opciones bien explicadas para que los usuarios no les cueste trabajo operar con él. También se analizaron los resultados de los algoritmos implementados en la aplicación en la fase de prueba del software con el objetivo de determinar cuan buenos son los resultados de los mismos haciendo un análisis detallado basado en grabaciones almacenadas en el departamento de Procesamiento de Voz del CEETI.

Los algoritmos implementados, de forma general, obtienen buenos resultados en la estimación de los diferentes parámetros de la voz. El de estimación de la F0 obtiene mejores resultados que el implementado en el software COLEA ya que este a veces realiza estimaciones erróneas debido a que analiza la voz un poco más superficialmente que el implementado en este trabajo. Sin embargo el algoritmo de estilización de la F0 no obtiene muy buenos resultados comparados con el praat[4] que realiza una interpolación más precisa, por lo que se recomienda que se mejore su implementación para trabajos futuros.

El algoritmo de segmentación tiene muy buena calidad en sus resultados pero presentó problemas en la detección de las sílabas que contenían, principalmente, los fonemas n, m, s, ch y z.

[4] Praat: Software internacional que realiza estimaciones de parámetros de la voz y que es muy usado en el mundo actual con mucha frecuencia.

Conclusiones

En esta investigación se hizo un estudio que abarcó, de forma concisa, todo el proceso de producción del habla humana y los algoritmos de estimación de los parámetros suprasegmentales de la prosodia como son la intensidad, la entonación y el ritmo, que más comúnmente se están usando en la actualidad.

De los algoritmos estudiados para determinar dichos parámetros suprasegmentales de la voz se implementaron algunos de ellos, los cuales fueron escogidos en dependencia de los recursos disponibles en el departamento de Procesamiento de Voz del CEETI.

- Se implementó un algoritmo de estimación de la entonación de la señal de voz que percibimos.
- Se implementó un algoritmo que determina la entonación.
- Se implementó un algoritmo que determina el comportamiento de la ritmicidad de la señal de voz.
- Se implementó un algoritmo que segmenta la señal de voz en sílabas.

Se confeccionó un manual de usuario donde se explica de forma detallada las facilidades que le brinda el sistema al usuario y como operar con él.

Con la implementación de dichos algoritmos y la realización del manual de usuario se cumplieron los objetivos trazados al comienzo de esta investigación.

Se llegó a la conclusión de que estos parámetros de la prosodia sirven para que los especialistas puedan apreciar anomalías en voces de pacientes y así poder emprender un trabajo investigativo con dichos pacientes para determinar las patologías que presentan, cumpliéndose de esta forma la hipótesis de investigación planteada y respondiéndose con todo lo anterior las preguntas de investigación.

Recomendaciones

Según los resultados alcanzados por esta investigación se recomienda, para trabajos futuros, los siguientes aspectos:

- Combinar el algoritmo de estilización y el de segmentado en sílabas con el objetivo de aumentar el rendimiento de este último.
- Refinar el algoritmo de segmentación con el objetivo de aumentar su precisión en la determinación de los extremos silábicos principalmente ante los fonemas *n, m, s, ch, z.*
- Realizar un sistema que a partir de los resultados que se obtienen con *APA* se pueda realizar un chequeo de las posibles patologías qu6e pueda presentar el paciente cuya voz se ha analizado con APA.

Bibliografía

1. Rediries, E., *Capitulo 26*. 2005.
2. Gurlekian, M.C.P.y.J., *Evaluación objetiva de la voz. Valores de referencia para Rosario y alrededores*. 2007.
3. K. S. R. B. Yegnanarayana, S.P.K., *Source and system Features for speaker recognition using AANN models.* 2001.
4. Valgadamara, J.J., *Creación de una base de datos fonética de voces chilenas.* 2004.
5. Rediries, E., *Capitulo 22*. 2005.
6. Rediries, E., *Capitulo 23*. 2005.
7. Rediries, E., *Capitulo 21*. 2005.
8. Miyara, F., *Acústica del tracto vocal y voz humana.* 2004.
9. Llisterri, J., *Concatenación de unidades almacenadas mediante síntesis paramétrica.* 2003.
10. Anguera, X., *Robust Speaker Segmantation for Meeting: The ICSI-SRI Spring 2005 Diarization System.* 2005.
11. J. Gil, J.L., *Fonética y Fonología del español en España*. 2003, Universidad Autónoma de Barcelona: Barcelona, España.
12. J. Bernal, J.B., P Gómez, *Reconocimiento de voz y fonética acústica.* México: Alfa Omega, 2000.
13. Laboratories, P., *International Phonetics Association*. 2006.
14. Menaldi, J., *La voz Patológica.* Ed. Panamericana, Buenos Aires, Argentina, 2002.
15. Kent, R., *Acoustic Analysis of Speech.* Delmar, Canadá, 2002.
16. Pardo, J.M., *Sistema de producción del habla. Apuntes de Ingeniería Neusensorial*. 2002, Universidad Politécnica de Madrid: Madrid, España.
17. Almiñana, J.M.G., *Desarrollo de un módulo de asignación de parámetros prosódicos para la versión en español del sistema de conversión texto-habla ACTOR.* 2004.
18. Castaño, R.A., *Estimación de contornos del Pitch en línea sobre DSP*. 2005.
19. Nikiforov Igor V., B.M., *Detection of abrupt changes: Theory and application*, ed. P.H. Inc. 1993, Rennes, Francia. 3, 10, 401-405.
20. Holger Quast, O.S., Manfred R. Schroeder, *Robust pitch tracking in the car environment*, in *Drittes Physykaliches Institute and Technology AT&T Bell Labs*. 2004, Universitat Gottingen and Development Daimlerchrysler Research.
21. Castellanos Cesar Germán, O.D.C.G. *Comparación de algoritmos de estimación del pitch en el análisis acústico de la voz normal y patológica*. in *VII Simposio de Tratamiento de Señales, Imágenes y Visión Artificial*. 2002. Bucaramanga Colombia.

22. autores, C.d., *Teoria sobre la Prosodia.* 2006.

23. Gadiyar Gopalkrishna H., P.R., *Ramanujan-fourier series, the wiener-khintchine formula and the distribution of prime pairs*, in *Institute of Advanced Study in Mathematics*. 1999, University of Madras: India.

24. Gómez, O.D.C., *Identificación de la voz normal y disfuncional, a partir de su análisis acústico objetivo*, in *Departamento de Ingeniería Electrónica*. 2002, Universidad Politécnica de Valencia: Valencia.

25. Bermúdez J. B., S.J.B., *Reconocimiento de voz y fonética acústica*. 2000, Universidad Politécnica de Madrid: Madrid.

26. Janer, G.L., *Transformada wavelet aplicada a la extracción de información en señales de voz*. 1998, Universidad Politécnica de Cataluña: Barcelona.

27. Toledano, D.T., *Segmentación y etiquetado fonéticos automáticos*. 2001, Universidad Politécnica de Madrid: Madrid.

28. Alan V. Oppenheim, W.R.S., *Digital signal processing*, in *Digital signal processing*, P. Hall, Editor. 1975: New Jersey, USA. p. page 480-531.

29. Seltzer, M.L., *Automatic detection of corrupt spectrographic features for robust speech recognition*, in *Department of Electrical Computer Engineering*. 2000, Carnegie Mellon University: Pennsylvania.

30. Garrido, J.M., *Modelización de patrones melódicos para la síntesis y el reconocimiento del habla*, in *Bellaterra, Department de Filología Espanyola*. 1991, Uuniversidad Autónoma de Barcelona: Barcelona.

31. P. A. Taylor, S.D.I., *Automatic phone segmentation*. Septiembre 1991: Genova, Italia. p. 709-711.

32. Matthew J. Makashay, C.W.W., Ann K. Syrdal, Aliasir Conkie, *Preceptual evaluation of automatic segmentation in Text-to-Speech synthesis.* Octubre, 2000.

33. Huici, M.E.H.D., *Exploring articulation in dysarthric speech by means of spectral transitions.* Centro de Estudios de Electrónica y Tecnologías de la Información (CEETI), Abril 2007.

34. Sorin Dusan, L.R., *On the relation between maximun spectral transition positions and phone boundaries*, in *Center for Advanced Information Processing Rutgers University*. 2005, Rutgers University: New Jersey, USA.

35. Furui, S., *On the Role of Spectral Transition for Speech Perception.* Journal Acoustics, Soc. Amer, 1986. **80(4)**: p. 1016-1025.

36. S. Davis, P.M., *Comparison of parametric representations for monosyllabic word recognition.* IEEE Trans. on Acoustics, Speech and Signal Process., 1980. **28 (4)**: p. 357-366.

37. Javier García de Jalón, J.I.R., Alfonso Brazaléz, *Aprenda Matlab 6.5 como si estuviera en primero*. Agosto 2002.

38. Rumbaugh, J.J., I. y G. Booch, *El Lenguaje Unificado de Modelado. Manual de Referencia*. 2000.

39. Ivar Jacobson, G.B., James Rumbaugh, *El Proceso Unificado de Desarrollo de Software*. 2000.

40. Lawrence R. Rabiner, M.J.C., Aaron E. Rosenberg and Carol A. McGonegal, *A Comparative Performance Study of Several Pitch Detection Algorithms*, in *Transactions on Acoustics, Speech and Signal Processing*. 1976. p. 399-417.

41. Epstein, M.A., *A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy in Linguistics.* 2002.

Anexos

Anexo 1

En este anexo se definen las etapas por las que pasa un modelo acústico de voz humana.

Etapas del modelo acústico de voz humana:

- **Generación del sonido madre:** Se produce mediante vibración de las cuerdas vocales (se distinguen dentro de esta categoría al menos tres patrones vibratorios diferentes), mediante flujo de aire y mediante golpes de presión acumulada [41].
- **Modulación del tracto vocal:** Se puede considerar al tracto vocal como un tubo de sección variable controlado según la articulación determinada por los músculos de la faringe, el cuerpo y punta de la lengua, el velo del paladar y los labios. Al igual que en un tubo simple, el fenómeno de resonancias en el tracto aumentan la amplitud de un grupo de frecuencias alrededor de una determinada banda de frecuencia. A cada resonancia se le denomina *formante* [41].
- **Radiación de salida:** La salida del sonido tiene asociada una impedancia de radiación que influye sobre el sonido variando la composición espectral del mismo, así como los niveles de presión sonora con respecto al ángulo de salida [41].

Anexo 2

En la figura 1.1.1 se muestra el aparato fonador humano y en la figura 1.1.1.2 una vista longitudinal de las cuerdas vocales cerrada y abierta.

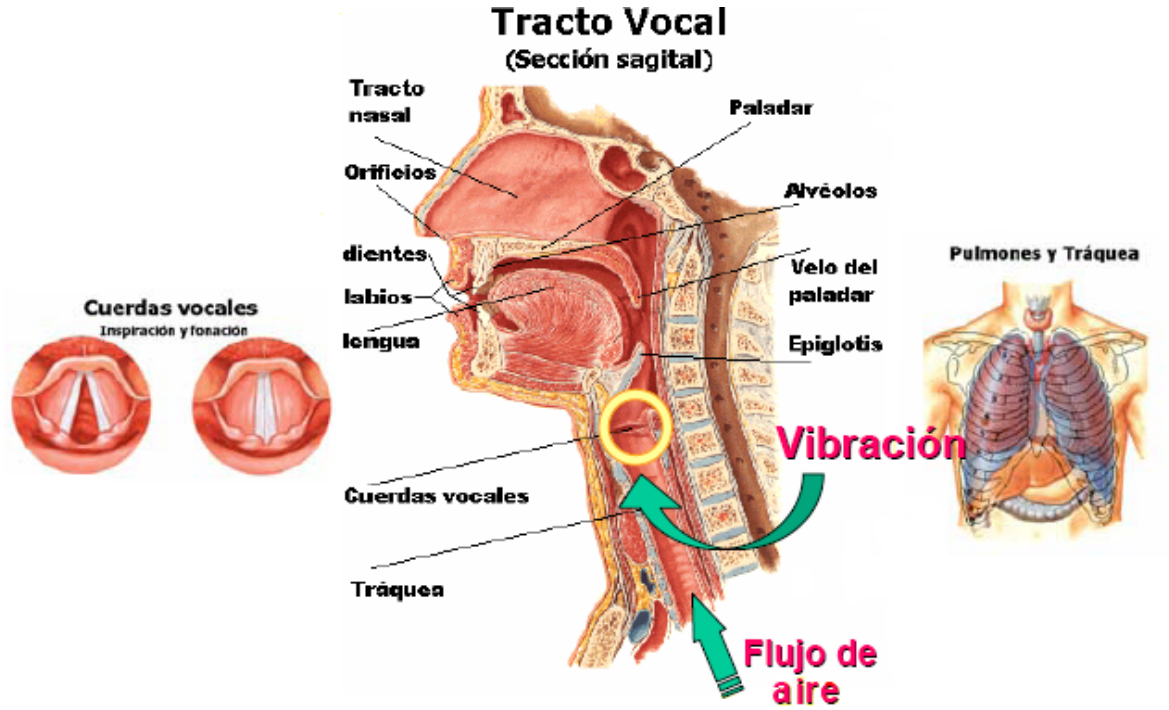


Figura 1.1.1. El aparato fonador humano.

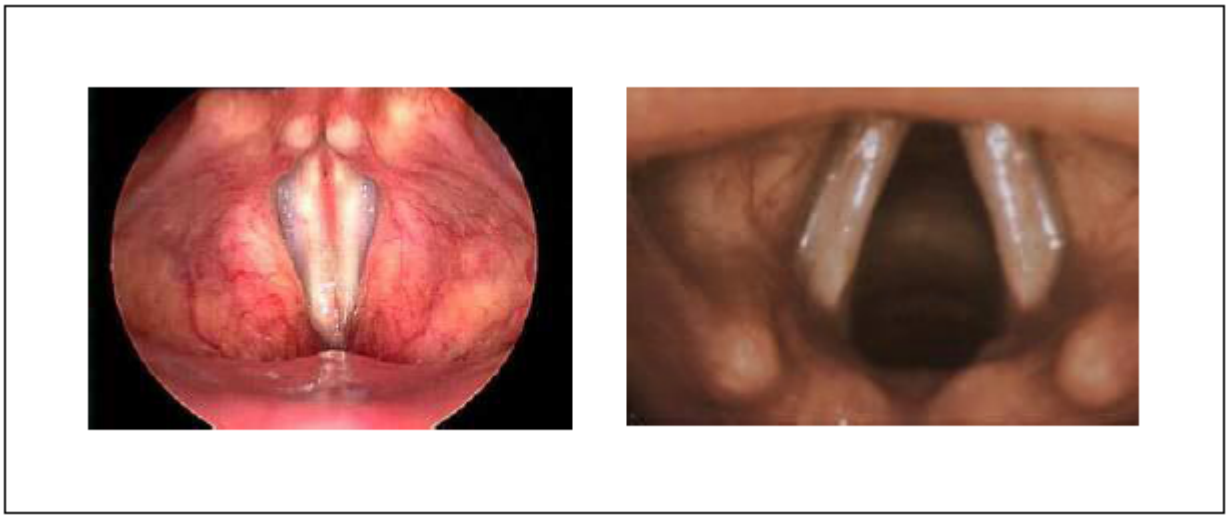


Figura 1.1.1.2: Vista longitudinal de las cuerdas vocales cerrada y abierta [30].

Anexo 3

En la figura 1.1.2 se muestra la clasificación de los fonemas vocálicos según su articulación, y en la figura 1.1.2.2 se presenta la clasificación de los fonemas consonánticos según su lugar y modo de articulación.

Posición Vertical	**Tipo de vocal**	**Posición horizontal (avance)**		
		Anterior	**Central**	**Posterior**
Alta	Cerrada	I		u
Media	Media	E		o
Baja	Abierta		a	

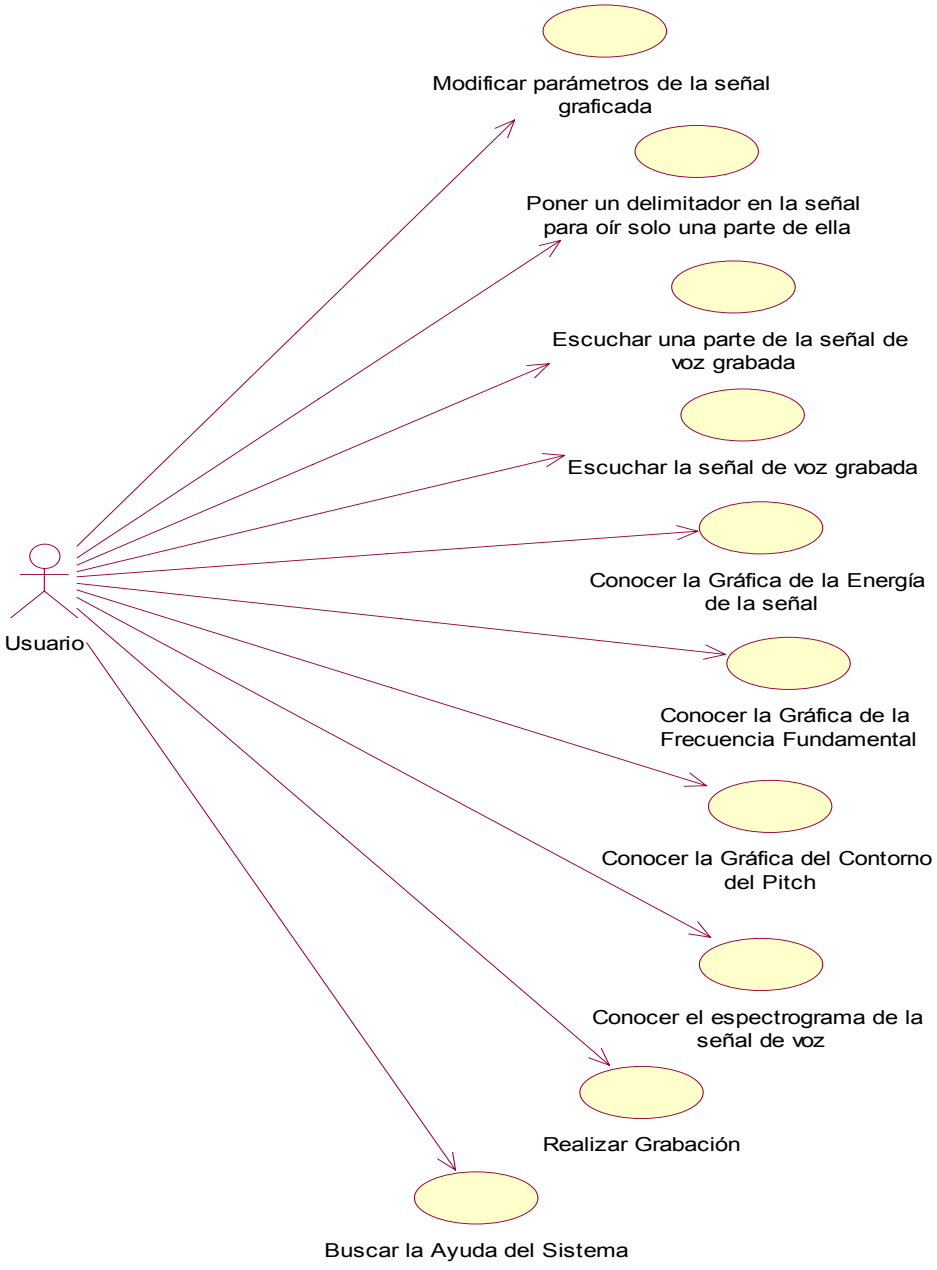
Figura 1.1.2. Clasificación de los fonemas vocálicos considerando su articulación [15].

Lugar de Articulación	Modo de Articulación								
	Oral								Nasal
	Oclusiva		Fricativa		Africada	Lateral	Vibrante	Aproximante	
	Sorda	Sonora	Sorda	Sonora	Sorda	Sonora	Sorda	Sonora	Sonora
Bilabial	p	b, v		b, v				w	m
Labiodental			f						
Linguodental			z						
Alveolar	t	d	s	y	ch	l	r, rr		n
Palatal				(y)	(ch)	ll		i	ñ
Velar	k	g	j						
Glotal			h						

Figura 1.1.2.2: Clasificación de los fonemas consonánticos según su lugar y modo de articulación [8].

Anexo 4

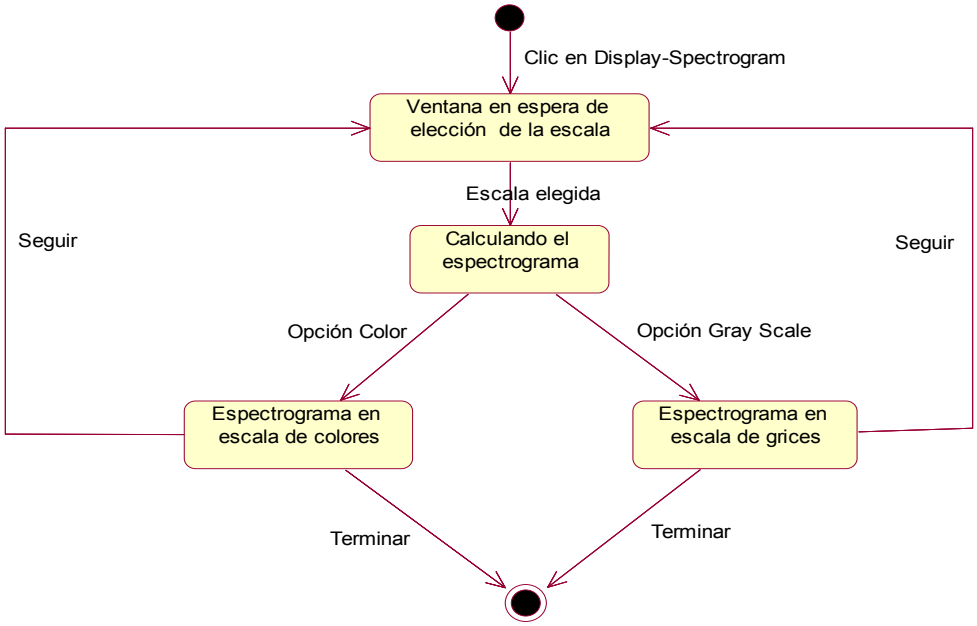
Diagrama de casos de uso ampliado con todas las opciones que brinda el software al usuario.



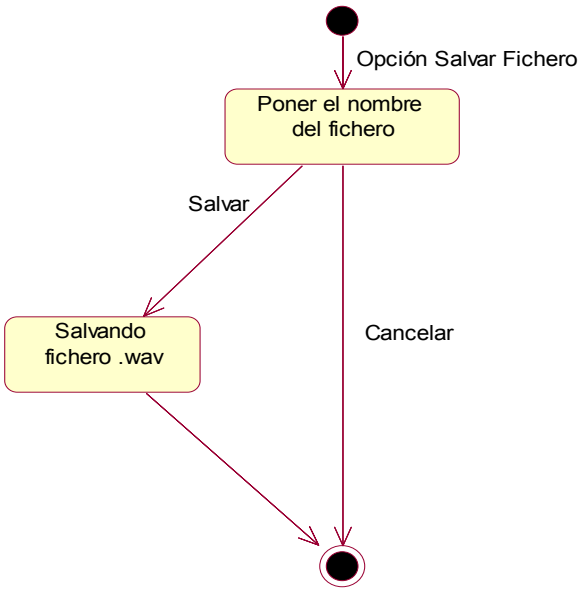
Anexo 5

Diagramas de transición de estados de otros casos de uso.

Caso de uso: *Conocer espectrograma de la señal de voz*.

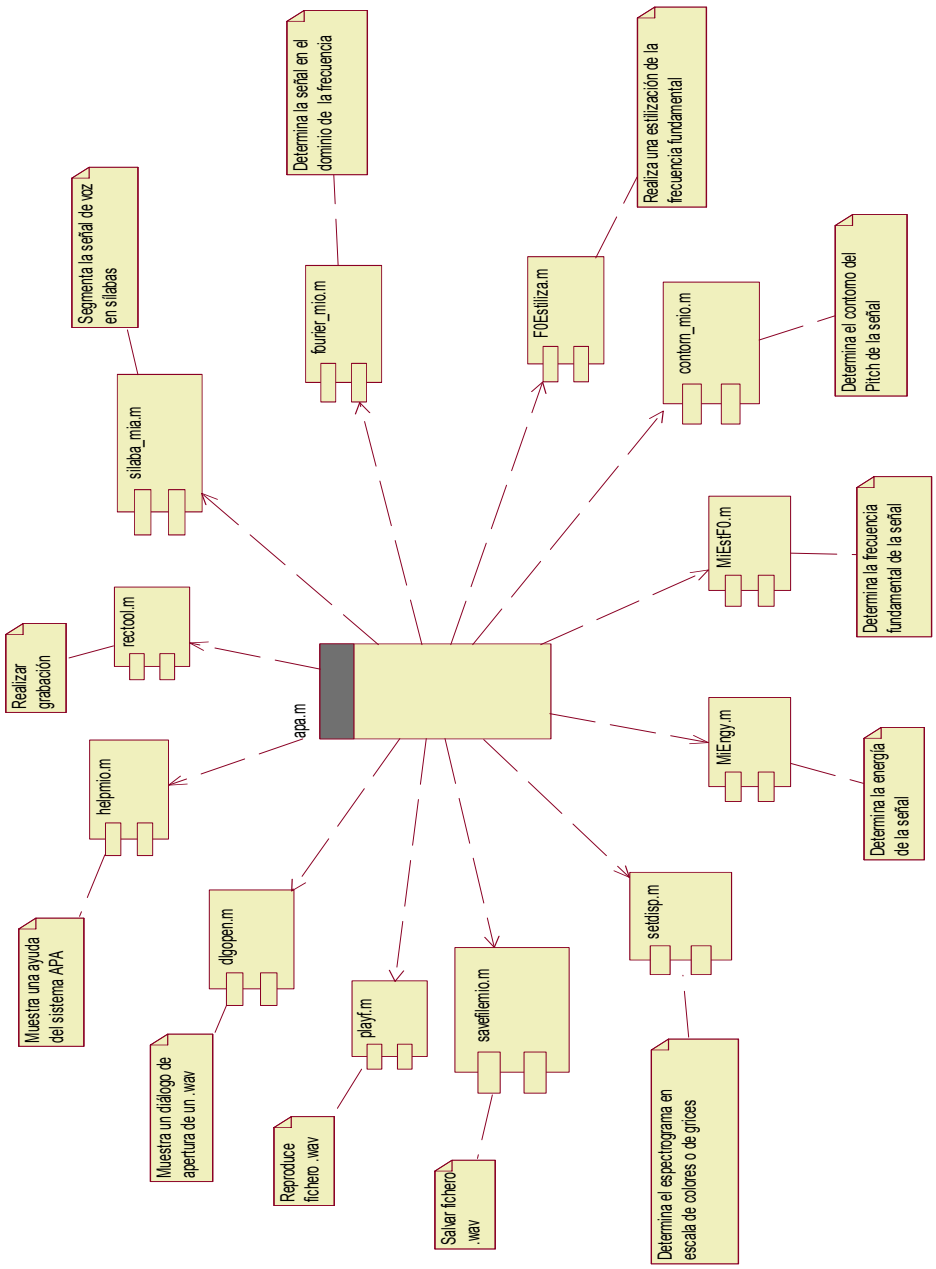


Caso de uso: *Salvar Fichero .wav.*



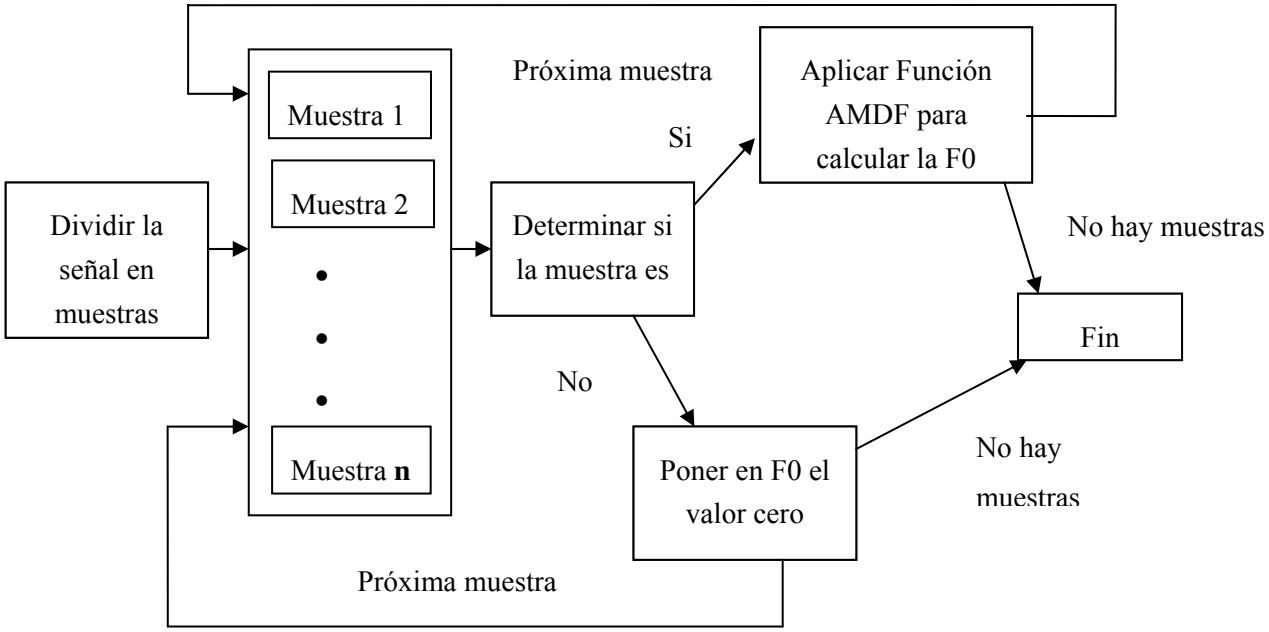
Anexo 6

Modelo de componentes del sistema.



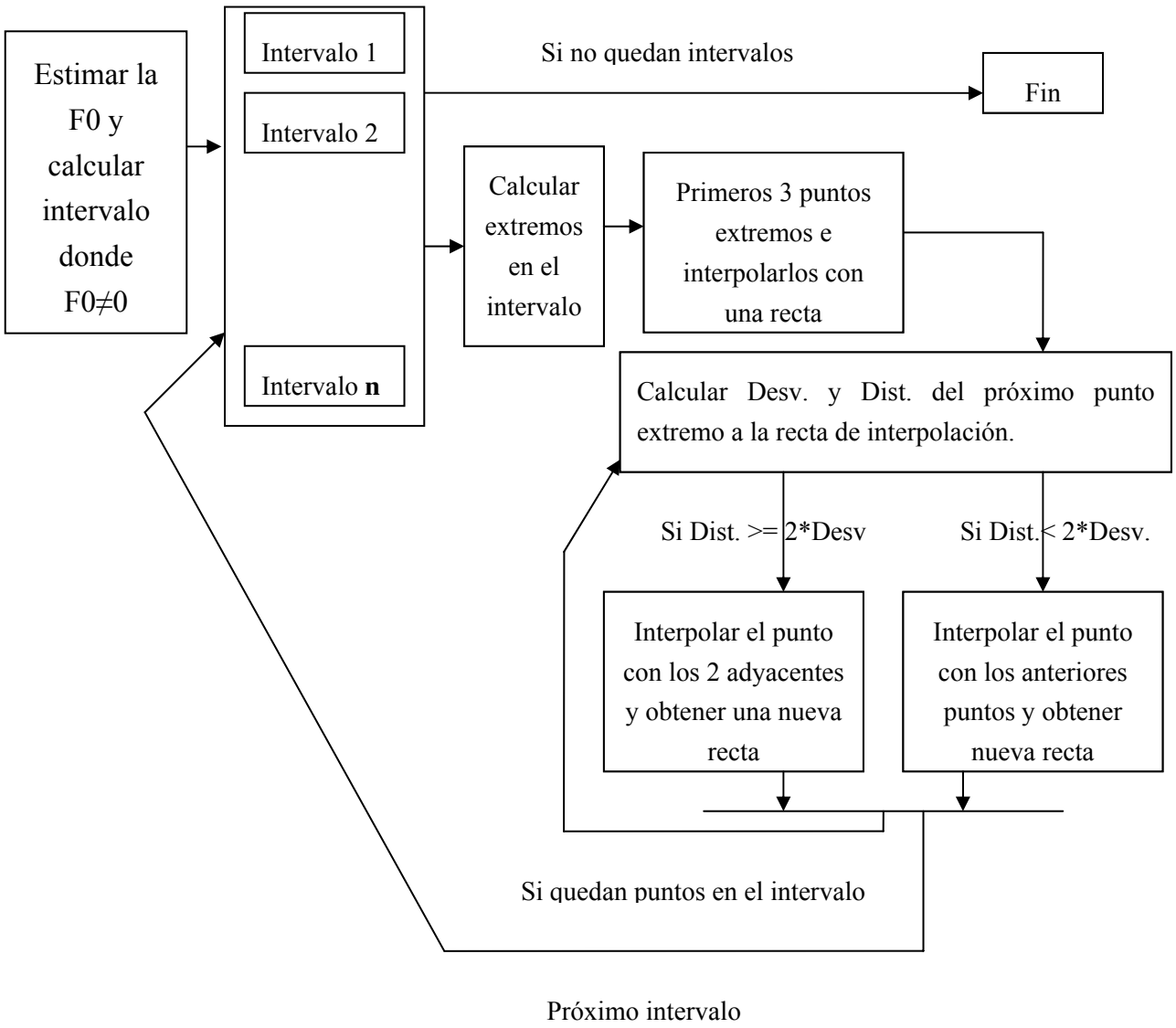
Anexo 7

Diagrama de bloques del algoritmo de estimación de la frecuencia fundamental.



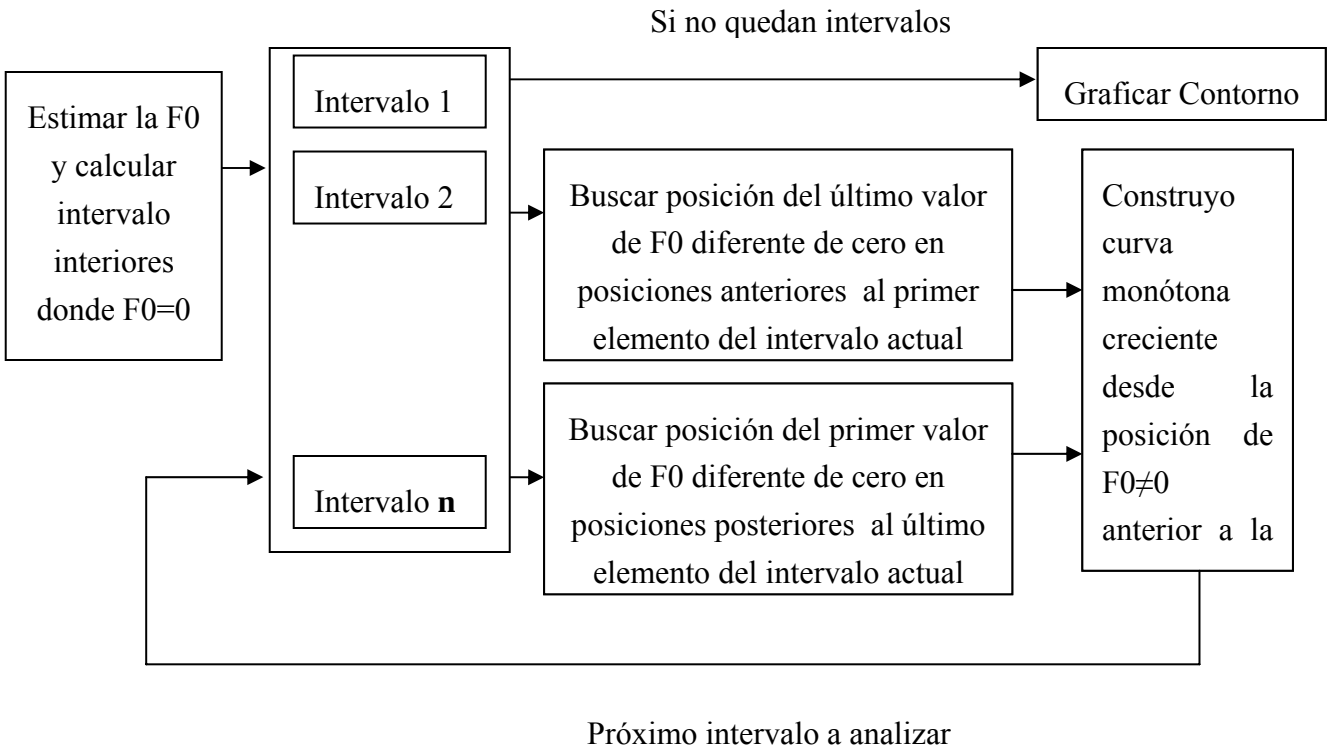
Anexo 8

Diagrama de bloques del algoritmo de estilización de la frecuencia fundamental.



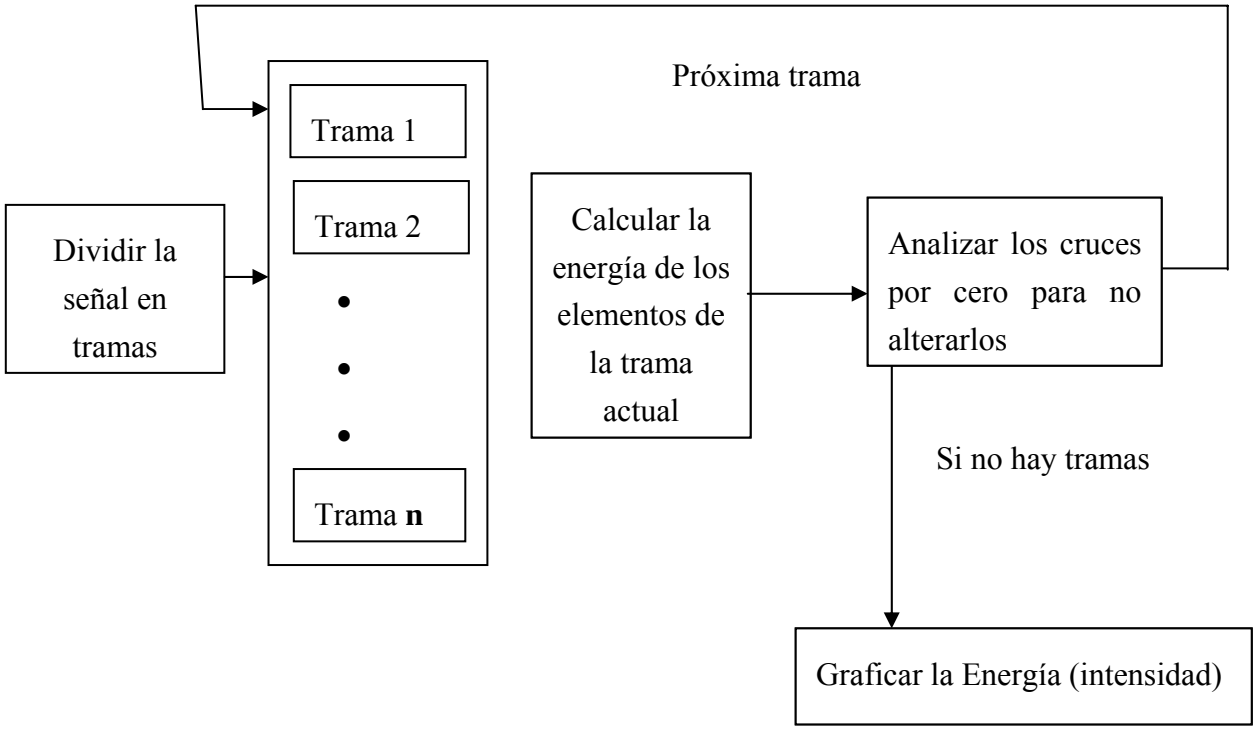
Anexo 9

Diagrama de bloques del algoritmo de construcción del contorno del Pitch.



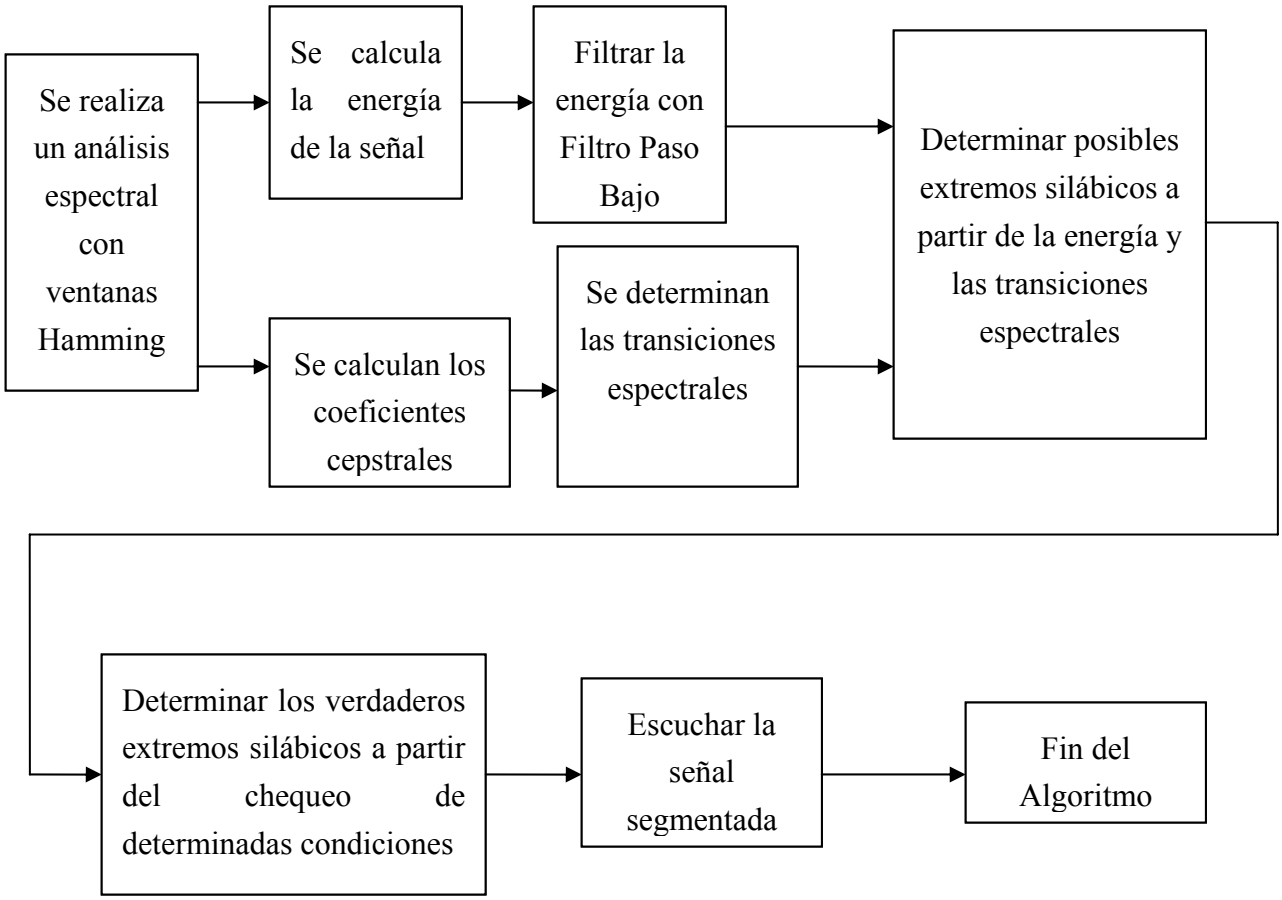
Anexo 10

Diagrama de bloques del algoritmo de estimación de la intensidad de la señal, que se basa en el cálculo de la energía de la señal que es el parámetro físico que la caracteriza.



Anexo 11

Diagrama de bloques del algoritmo de segmentación de la señal de voz en sílabas.



Anexo 12

En este anexo se muestran las principales funciones implementadas en el lenguaje de programación *Matlab*.

Lista de funciones más importantes usadas del colea:

- MiEngy.m
- Spectrogram.m
- Rectool.m
- Helpmio.m

Seguidamente se pondrá la lista de funciones implementadas en esta investigación:

- APA.m
- GrossF0DetectorMio.m (ver anexo 13)
- contorn_mio.m (ver anexo 14)
- F0Estiliza.m (ver anexo 15)
- silaba_mia.m (ver anexo 16)

Anexo 13

Código fuente de la función APA.m.

```
function APA(infile)
% Realiza el analisis automatico de la prosodia
colordef none  % comment this line in MATLAB 4.x
global hl hr doit xs fsm
global fed shW
global filename
global fno Srate n_Secs filename AXISLOC  nLPC HDRSIZE
global nChannels fftSize filterWeights UpperFreq LowFreq ChanpUp
global center LPCSpec SpecUp Dur DurUp filterA filterB  S1 S0
global HDRSIZE cAxes En Be TIME centSMSP Asmsp Bsmsp OVRL WAV1 CLR
global TWOFILES wav agcsc MAX_AM upFreq upFreq2 FIX_SCALE
global SHOW_CRS SHOW_CHN ftype ftype2 bpsa bpsa2 PREEMPH LPC_ONLY
global lpcParam fft_par NAR_BAND SPEC_EXP FILT_TYPE VOL_MAX
global VOL_NORM
global sli n_samples FFT_SET XFFT
global tpc boc
global smp frq TOP
global preUp  defUp w64Up w128Up w256Up w512Up narUp
global crsUp chnUp lpcUp chnlpUp fbrd fnar
global ovlpfilt SET_X_AXIS LD_LABELS
if (nargin < 1)
 [pth,fname] = dlgopen('open','*.ils;*.wav','APA');
 if ((~isstr(fname)) | ~min(size(fname))),
    return;
 end
 filename=[pth,fname];
else
  filename=infile;
end
pos = get(0, 'screensize'); % get the screensize
sWi = pos(3);
sHe = pos(4);
WIDTH   =round(0.7*sWi);
HEIGHT  =round(0.43*sHe) ;
LEFT    =round(0.025*sWi);
BOTTOM  =round(sHe-HEIGHT-40);
LPCSpec = 1;            % if 0 display FFT, else LPC spectrum
TIME=1;                 % if 0 display spectrogram, else time waveform
WAV1=0;                 % If 1, then it is a .wav file with 8 bits/sample
CLR=1;                  % If 1 display spectrogram in color, else in gray scale
TWOFILES=0;             % If 1, then display two files
TOP=0;
fp = fopen(filename,'r');
if fp <=0
        disp('ERROR! File not found..')
        return;
```

```
end
ftype='short';
bpsa=2;    % bytes per sample
ftype2='short';
bpsa2=1;
ind1=find(filename == '.');
if length(ind1)>1,
   ind=ind1(length(ind1));
else,
   ind=ind1;
end;
ext = lower(filename(ind+1:length(filename )));
[HDRSIZE, xSrate, bpsa, ftype] = gethdr(fp,ext);
if xSrate==0,
   return;
else
   Srate=xSrate;
end;
if strcmp(ftype,'ascii')
 x=fscanf(fp,'%f',inf);
else
 x  = fread(fp,inf,ftype);
end
fclose(fp);
if (Srate<6000 | Srate>45000) & (nargin<2)
h=warndlg('Sampling rate not in the range: 10,000 < F < 45,000 . Setting it to  10,000 Hz.','WARNING!');
  disp('Warning! Sampling rate not in the range: 6,000 < F < 45,000');
  disp('...Setting it to the default value of 10,000 Hz.');
  Srate=10000;
end
x= x - mean(x);  %----------remove the DC bias----
if (nargin==2)
 Srate  = str2num(Srate1);
 if Srate<10000 | Srate>45000
        error('Invalid sampling frequency specified: 10,000<F<45,000');
 end
end
MAX_AM=2048; % This allows 12-bit resolution
mx=max(x);
agcsc=MAX_AM/mx;
n_samples = length(x);
n_Secs    = n_samples/Srate;
S1=n_samples;
S0=0;
Be=S0;
En=S1;
OVRL=1;  % if 1 then hold off plots, else hold on plots in 'pllpc'
fprintf('Samp.Freq: %d Hz, num.samples: %d (%4.2f secs)\n',Srate,n_samples,n_Secs);
fno =  figure('Units', 'Pixels', 'Position', [LEFT BOTTOM WIDTH HEIGHT],...
        'Resize','on','Name','Automatic Prosodic Analysis','NumberTitle','Off',...
        'Menubar','None','KeyPressFcn','getkb','Color','k');
%------------ deterime the dimensions of the axis ------------
```

```
le=round(0.2*WIDTH);
bo=round(0.174*HEIGHT);
wi=round(0.773*WIDTH);
he=round(0.739*HEIGHT);
AXISLOC = [le bo wi he];
cAxes = axes('Units','Pixels','Position',AXISLOC);
axes(cAxes);
        Et=1000*n_samples/Srate;
        xax=0:1000/Srate:(Et-1000/Srate);
        plot(xax,x,'y')
        xlabel('Time (msecs)');
        ylabel('Amplitude');
        set(gca,'Color','k');
   set(gca,'Xcolor','w');
   set(gca,'YColor','w');
        if min(x)<-1000 | mx >1000
          axis([0 Et min(x)-200 max(x)+200]);
        else
          axis([0 Et min(x) max(x)]);
        end
       set(gca,'XColor','w');
  set(gca,'YColor','w');
xywh = get(fno, 'Position');
axi=AXISLOC;
%----------Buttons----------
left = 10;
wide = 80;
top  = xywh(4) - 10;
high = 22;
if 9*(22+8) > xywh(4),
    high=17;
end;
inc  = high + 8;
%---------- Display the slider and the push-buttons----------
sli = uicontrol('Style','slider','min',0,'max',1000','Callback',...
        'getslide','Position',[axi(1) axi(2)+axi(4)+2 axi(3) 12]);
top = top - inc;
uicontrol('Style','Frame','Position',[left top-high-10 wide+5 high+30],...
        'BackgroundColor','b');

uicontrol('Style','text','Position',[left+wide/3 top 40 high-3],'BackGroundColor','b',...
        'HorizontalAlignment','left','ForeGroundColor','w','String','Play');
plUp = uicontrol('Style', 'PushButton', 'Callback', 'playf("all")', ...
         'HorizontalAlign','center', 'String', 'ALL',...
        'Position', [left top-high wide high]);%[left top-high wide/2 high]
%
%----------MENUS----------
%
ff=uimenu('Label','File');
  uimenu(ff,'Label','&Save whole file','Callback','savefilemio("whole","APA")','Separator','on');
  uimenu(ff,'Label','Print-Landscape','Callback','cprint("landscape","printer")','Separator','on');
```

```
uimenu(ff,'Label','Exit','CallBack','quit all','Separator','on');
[xs,fsm]=wavread(filename);
fd=uimenu('Label','Display');
    fd0= uimenu(fd,'Label','Spectrogram');
        uimenu(fd0,'Callback','setdisp(''spec'',''clr'')',...
            'Label','Color');

uimenu(fd0,'Callback','setdisp(''spec'',''noclr'')',...
            'Label','Gray Scale');
        uimenu(fd,'Label','Intensity Plot','Callback','MiEngy');
    fdf0=uimenu(fd,'Label','F0 Plot','Callback','GrossF0DetectorMio(70,300,0.03,0.01)');
  fv=uimenu('Label','Analysis');
    uimenu(fv,'Label','F0 Contorn','CallBack','contorn_mio');
    uimenu(fv,'Label','F0 Stylization','CallBack','F0Estiliza');
    uimenu(fv,'Label','Segmentation in Syllable','CallBack','silaba_mia');
fv1=uimenu('Label','Record','CallBack','rectool');
uimenu('Label','Help','Callback','helpmio(''apa'')');
doit=0;
```

Anexo 14

Código fuente de la función GrossF0DetectorMio.

function [F0]=GrossF0Detector(fmin,fmax, WL,S)

```
%function [F0]=GrossF0Detector(x,fs,fmin,fmax,WL,S)
%Detección de Frecuencia Fundamental por AMDF,
%x:señal de voz, fs: frecuencia de muestreo (Hz),
%fmin y fmax:límite inferior y superior de F0 (Hz),
%WL: tamaño de ventana (Seg), S: solapamiento (Seg),
%sugerencia:WL=0.03 y S=0.015
global xs fsm
x=xs;
fs=fsm;
LargeJumpFlag=0;
Winlength=floor(WL*fs);%llevarla a muestras
  Step=floor(S*fs);
  NSeg=floor((length(x)-Winlength)/Step);%# de Segmentos
posi=fix(fs/fmax);%pitch mínimo en muestras
posf=fix(fs/fmin);%pitch máximo en muestras
SD=0.003;%xcorr(x-mean(x))/length(x);
F0=[];
f=x;
if NSeg > 0,%si hay al menos un segmento, buscar F0
  for i=1:NSeg,
    %i
    Seg=f(((i-1)*Step+1):i*Step+Winlength);
    Seg=Seg-Mean(Seg);%eliminar DC
    MaxPos=[];ProbPos=[];ProbVal=[];
    if Voiced(Seg,fs,posi,posf,SD((length(SD)+1)/2)),
      if i>1 & ~isempty(max(find(F0>0))),%F0(i-1)>0,
PrevPos=fs/F0(max(find(F0>0)));%F0(i-1);
      else
        PrevPos=0;
      end;
      if i==23,
        a=5;
      end;
      AMDFVect=AMDF(Seg,posi-2,posf+2);
      AMDFVect=max(AMDFVect)-AMDFVect;
MaxPos=FindNMaxInRange(AMDFVect,3,posi,posf);
[ProbPos,ProbVal]=FindProb(AMDFVect,MaxPos,...
       ProbPos,ProbVal,posi,posf,PrevPos);
                  if ~isempty(find(ProbVal==max(ProbVal))),
```

```
Pos=find(ProbVal==max(ProbVal));% si hay dos iguales
          end;
        if ~isempty(ProbPos) & ...
            ProbPos(Pos(1))~=0,%por si viniera en cero

F0(i)=fs/ProbPos(Pos(1));%coger el primero
        else
          F0(i)=0;
        end;


     else
       F0(i)=0;
     end;
     if i>2,%Chequeo de saltos de pitch
        if LargeJumpFlag,%salto previo
          F0(i-1)=median(F0((i-2):i));
          LargeJumpFlag=0;
        end;
        Bigger=max(F0((i-1):i));
        if Bigger~=0 & (abs(F0(i)-F0(i-1))/Bigger > 0.2),
          LargeJumpFlag=1;
        end;
     end;
   end;


end;


%------------ deterime the dimensions of the axis ------------

pos = get(0, 'screensize'); % get the screensize
sWi = pos(3);
sHe = pos(4);

% WIDTH   =round(0.9375*sWi);
WIDTH   =round(0.7*sWi);
HEIGHT  =round(0.43*sHe) ;
LEFT    =round(0.025*sWi);
BOTTOM  =round(sHe-HEIGHT-40);


xt=1:NSeg;
xt=20*xt;


eFt =  figure('Units', 'Pixels', 'Position', [LEFT BOTTOM WIDTH HEIGHT],...
          'Resize','on','Name','F0 Plot','NumberTitle','Off',...
          'Menubar','None','KeyPressFcn','getkb','Color','k');
plot(F0);
xlabel('Time (*10 msecs)');
ylabel('Amplitude');


%_____________________________________
_____________________________
function ZC = ZeroCrossings(x)
%Cantidad de cruces por cero del vector x
ZC=0;
for i=2:length(x),
   if ((x(i)>=0) & (x(i-1))<0) | ((x(i)<0) & (x(i-1))>=0),
      ZC=ZC+1;
```

```
  end;
end;
%_____
function y = AMDF(x,posi,posf);
%La función AMDF del vector x, de posi a posf,
%(el índice 1 es dezplazamiento cero)
temp=[x' zeros(1,length(x))];
y=zeros(1,length(x));
j=1:length(x);
for i=posi:posf,
  %for j=1:length(x),
    y(i)=sum(abs( x(j)'-temp(i+j-1) ));
  %end;
end;
%_____
function Pos = FindAllMaxPos(x);
%halla las posiciones de todos los máximos de x
der=diff(x);Pos=[];
for i=2:length(der),
  if der(i)<0 & der(i-1)>=0,
    Pos=[Pos i];
  end;
end;
%_____
function Pos = FindNMaxInRange(x,N,posi,posf);
%posiciones de los tres mayores máx. de x en el rango
Pos=[];
AllPos = FindAllMaxPos(x);
Maxs=AllPos(find(AllPos>=posi & AllPos<=posf));
for i=1:N,
  [v,p]=max(x(Maxs));%se obtiene máximo y posición
  Pos=[Pos Maxs(p)];%se anade posición a salida
  if ~isempty(p),
    Maxs(find(Maxs==Maxs(p)))=[];%se elimina máximo
  end;
end;
%_____
function [ProbPos,ProbVal] = FindProb(x,MaxPos,ProbPos,...
        ProbVal,posi,posf,PrevPos)
%Asigna probabilidades a los maximos segun varios criterios
MToMMMR=1.6;%umbral de relación entre máximos
Found=zeros(1,length(MaxPos));
TempProb=[];
if ~isempty(MaxPos), %si hay al menos un máximo
  if x(MaxPos(1))~=0,
TempProb=x(MaxPos)/x(MaxPos(1));%normalizar
Temp=find(TempProb(1)>MToMMMR*TempProb);
    TempProb(Temp)=[];
    MaxPos(Temp)=[];
```

```
    Found(Temp)=[];
  end;%y eliminar pequeños
end;
if ~isempty(TempProb),%si se halló probabilidad
  if isempty(ProbPos),%si no había previa, almacenar hallados
    ProbPos=MaxPos;
    ProbVal=TempProb;
  else%si había, buscar coincidencia con previos
    for i=1:length(ProbPos),
      for j=1:length(MaxPos),%añadir prob. en coinc.
        if ( MaxPos(j)>ProbPos(i)-2 )...%y...
            & ( MaxPos(j)<ProbPos(i)+2),

ProbVal(i)=ProbVal(i)+TempProb(j);
          Found(j)=1;%señalar que se halló
        end;
      end;
    end;
    for i=1:length(Found),%añadir no encontrados
      if ~Found(i),
        ProbVal=[ProbVal TempProb(i)];%Valor
        ProbPos=[ProbPos MaxPos(i)];%Probabilidad
      end;
    end;
  end;
  %Buscar múltiplos y submultiplos
  for i=1:length(ProbPos),
    for j=1:length(MaxPos),%añadir prob. en coinc.
      k=2;l=2;
      while MaxPos(j)/k > posi,
        if ( MaxPos(j)/k > ProbPos(i)-2 )...
            &( MaxPos(j)/k < ProbPos(i)+2 ),
          ProbVal(i)=ProbVal(i)+0.05;
        end;
        k=k+1;
      end;
    end;
  end;
  %10% a los cercanos al valor en segmento anterior
  for i=1:length(ProbPos),
    if PrevPos>ProbPos(i)*0.75 & PrevPos < ProbPos(i)*1.25,
      ProbVal(i)=ProbVal(i)+0.1;
    end;
  end;
end;
%_____
_____
function V = Voiced(x,fs,posi,posf,SD);
%detecta sonoridad según cruces por cero y autocorr.
%fs(Hz),winlength(muestras)
Noise8Bits=3;
MinZc=240;MaxZc=4300;
V=1;
ACrr=xcorr(x);
if ZeroCrossings(x)*fs/length(x)<MinZc...
```

```
    |
ZeroCrossings(x)*fs/length(x)>MaxZc
,
  V=0;
end;
if max(ACrr(posi:posf)) < 0.3*ACrr(1),
  V=0;
end;%si el máx < 0.3 no hay
periodicidad
if
ACrr((length(ACrr)+1)/2)<length(x)*(
max([SD/5.62 0.025]))^2,
  V=0;
end;%si tiene poca energía es ruido
también
```

Anexo 15

Código fuente de la función contorn

```
function contorno
% Esta funcion calcula el contorno de la frecuencia fundamental de una
% sennal de voz
global nChannels filename Srate eFig
global HDRSIZE S0 S1 ratePps En Be  n_Secs En2 Be2
global filename2 TWOFILES n_Secs2 Srate2 TOP HDRSIZE2
global bpsa bpsa2 ftype ftype2 bf0 af0 SrateChange
if isempty(filename),
   [pth,fname] = dlgopen('open','*.ils;*.wav');
   if ((~isstr(fname)) | ~min(size(fname))),
      return;
   end
   filename=[pth,fname];
end
[f01]=EstF0Mio(filename);
cant= length(f01);
i=1;
inc=0;
[P]= BuscarPosCeros(f01);
k=1;
b=length(P);
%No me fijo en los ceros que hayan al inicio
while ((k<b)&(f01(k)==0)),
   k=k+1;
end
if k>1,
   k=3; % Para pasar al par siguiente
end
 while (k<b),
   pos=P(k);
   pos1=P(k+1);
   % Para no quitar los ceros al final de la F0
   if pos1~=cant,
      elem2= f01(pos1+1);
      if pos >=2,
        elem1= f01(pos-1);
      else
        elem1=30;
      end
      elem= elem2-elem1;
      dif=pos1-pos+2;
      inc= elem/dif;
      p=pos;
      while p<=pos1,
         if p==1,
            incr=30;
         else
            incr= f01(p-1)+inc;
         end
         f01(p)= incr;
         p=p+1;
      end
   end
   k=k+2;
 end
 %PLoteo el contorno
```

```
 pos = get(0, 'screensize'); % get the screensize
sWi = pos(3);
sHe = pos(4);
 WIDTH   =round(0.9375*sWi);
HEIGHT  =round(0.43*sHe) ;
LEFT    =round(0.025*sWi);
BOTTOM  =round(sHe-HEIGHT-80);
nmF='F0 Contorn';
eF =  figure('Units', 'Pixels', 'Position', [LEFT BOTTOM WIDTH HEIGHT],...
          'Pointer','crosshair',...
          'Resize','on','Name',nmF,'NumberTitle','Off');
% %Button
% xywh = get(eF, 'Position');
% top  = xywh(4) - 10;
% high = 22;
% wide = 80;
% plUp = uicontrol('Style', 'PushButton',  'Callback', 'playf(''all'')', ...
%          'HorizontalAlign','center', 'String', 'Play',...
plot(f01);
xlabel('samples');
ylabel('dB');
% Esta funcion busca las posiciones inicial y final de una secuencia de
% ceros en un arreglo
function [Posix]=BuscarPosCeros(arr)
Posix=[];
cant=length(arr);
i=1; j=2;
while (i<cant),
   %Buscar el primer cero a partir de la posición en que estoy en el
   %arreglo de la F0
   if arr(i)==0,
      j=i;
      while ((j<cant)&(arr(j)==0))
         j=j+1;
      end
          if (j<cant),
         % Encontré la posición del último cero de la secuencia
          k=j-1;
          %Pongo en Posix la posición inicial y final de cada secuencia
          %de ceros del arreglo
          [Posix]= [Posix i k];
          i=j-1;
      else
          [Posix]= [Posix i j];
      end
   end
 i=i+1;
end
```

Anexo 15

Código fuente de la función F0Estiliza.

```
function [est]= F0Estiliza
global filename Srate n_samples
%Estilizar la frecuencia fundamental
if isempty(filename),
   [pth,fname] = dlgopen('open','*.ils;*.wav');
   infile=[pth,fname];
else
   infile=filename;
end

%Calculo la frecuencia fundamental f0
[xe,fs]=wavread(infile);
[f01]=GrossF0Detector(xe,fs,70,300, 0.03,0.01);

% Extremos=[1:length(f01)];
%Determino los rangos de ceros que hay en f01
[rango]=RangoPosCeros(f01);

%Determinar los rangos de valores de f01 donde no hay ceros
n=length(f01);
[noceros]=NoCeros(rango,n);

%Determino la cantidad de rangos diferente de cero que hay en la f01
cr=length(noceros)/2;

%Llenar el arreglo coeficientes de ceros
coeficientes=zeros(length(f01),1);
i=1;
while (i<=cr),
   %Coger el primer indice del rango de ceros "i" que es donde empiezan los primeros ceros
   j=2*i;
   rangi=noceros(j-1);
   range= noceros(j);

   %Calcular los minimos locales en esta parte de f01

[MinLocal]=FindAllMin(f01(rangi:range));

   %Calcular los maximos locales en esta parte de f01
   [MaxLocal]= FindAllMax(f01(rangi:range));

   %Ordenar los extremos locales
   Extremos=sort([MinLocal MaxLocal]);

   for x=1:length(Extremos),
      Extremos(x)=rangi+Extremos(x)-1;
   end

   k=1;
   if length(Extremos)>=3,
   if (Extremos(k)>rangi),
      while ((k+2)<=length(Extremos))&(Extremos(k+2)<=range),
         if (Extremos(end)-Extremos(k)+1)>=3,

            %Obtengo los tres puntos siguientes
            x= [Extremos(k) Extremos(k+1) Extremos(k+2)];
            y=f01(x);
            ultimo=x(end);
            [P,S] = POLYFIT(x,y,1);
coeficientes(x(1):x(end))=polyval(P,x(1):x(end));
            if ((k+3)<=length(Extremos))&(Extremos(k+3) >=rangi),
               if (Extremos(k+3)<=range),
k=k+3;
                  pto(1)=Extremos(k);
                  pto(2)=f01(pto(1));


dist=abs(polyval(P,pto(1)));
```

```
%dist=DistPtoRecta(pto,P(1),P(2));

                    %Calculo la
desviacion estandar del punto
                    pol=
polyval(P,x(1):pto(1));
                    desv=
std(f01(x(1):pto(1))-pol);
while
(dist<=2*desv)&(Extremos(k)<=range
)
x=[x Extremos(k)];
y=f01(x);
ultimo= x(end);
[P,S] = POLYFIT(x,y,1);

coeficientes(x(1):x(end))=polyval(P,x(
1):x(end));
k=k+1;
if      (k<=length(Extremos))      &
(Extremos(k)<=range),
pto(1)=Extremos(k);

pto(2)=f01(pto(1));
dist=abs(polyval(P,pto(1)));
desv=std(f01(ultimo:pto(1))-
(coeficientes(ultimo:pto(1)))');
 end
 end
 if (Extremos(k)<=range),
                      k=k-2;
                    end
                  end
                end
          else
 x=(Extremos(k):Extremos(k+1));
            y=f01(x(1):x(end));
            [P,S] = POLYFIT(x,y,1);

coeficientes(x(1):x(end))=polyval(P,x(
1):x(end));
          end
          k=k+1;
       end
       if ((k+2)<=length(Extremos)),
            if Extremos(k+1)<=range,
                x=(Extremos(k):range);
                y=f01(x(1):x(end));
                [P,S] = POLYFIT(x,y,1);

coeficientes(x(1):x(end))=polyval(P,x(1):x
(end));
             else
coeficientes(ultimo:Extremos(k))=polyval(
P,ultimo:Extremos(k));
             end
       else
          if ((k+1)<=length(Extremos)),
            pos=Extremos(k+1);
            x=(Extremos(k):range);
            y=f01(x(1):x(end));
            %de la forma P=[m,n]
            [P,S] = POLYFIT(x,y,1);

coeficientes(x(1):x(end))=polyval(P,x(1):x
(end));
          end
       end
    else
       if Extremos(k)<=range,
          x=(k:range);
          y=f01(k);
          %Construyo la recta que une a
todos los puntos de f0 que pertenecen a
la recta de
          %interpolacion actual. Esta
funcion devuelve los coeficientes de la
recta y=m*x+n
          %de la forma P=[m,n]
          [P,S] = POLYFIT(x,y,1);

coeficientes(x(1):x(end))=f01(x(1):x(end))
;
       end
    end
    else
       x=(rangi:range);
       y=f01(x(1):x(end));


       %de la forma P=[m,n]
```

```
      [P,S]      =      POLYFIT(x,y,1);
coeficientes(x(1):x(end))=polyval(P,x(
1):x(end));
   end
   if      (length(Extremos)==1      ||
length(Extremos)==2)&((Extremos(k)
>=rangi) & (Extremos(k)<=range)),
      x=(rangi+k-1:range);
      y=f01(x(1):x(end));
      ultimo=x(end);
      [P,S] = POLYFIT(x,y,1);
coeficientes(x(1):x(end))=f01(x(1):x(e
nd));
   end
   i=i+1;
end
coeficientes(Extremos(end):length(f0
1))=0;
pos = get(0, 'screensize'); % get the
screensize
sWi = pos(3);
sHe = pos(4);

WIDTH   =round(0.7*sWi);
HEIGHT  =round(0.43*sHe) ;
LEFT    =round(0.025*sWi);
BOTTOM  =round(sHe-HEIGHT-40);
eF =  figure('Units', 'Pixels', 'Position',
[LEFT BOTTOM WIDTH HEIGHT],...
        'Resize','on','Name','F0
Stylization','NumberTitle','Off',...
        'Menubar','None','KeyPressFc
n','getkb','Color','k');

le=round(0.2*WIDTH);
bo=round(0.174*HEIGHT);
wi=round(0.773*WIDTH);
he=round(0.739*HEIGHT);

AXISLOC = [le bo wi he];
cAxes                                  =
axes('Units','Pixels','Position',AXISLO
C);
axes(cAxes);
        Et=1000*length(coeficientes)/i;
%
Et=1000*n_samples/length(coeficientes);
        xax=0:1000/i:(Et-1000/i);

x=0:length(coeficientes)/length(f01)/fs:(le
ngth(coeficientes)-1)/fs;
% set(gca,'Name','F0 Stylizated');
% plot(x,f01,'r');
hold on
grid
zoom
%                    eF1=figure('Name','F0
origin','NumberTitle','Off');
plot(x,coeficientes,'y');
xlabel('Time (*100000 secs)');
ylabel('Amplitude');
grid
hold

% Funcion que determina los puntos
minimos locales de F0
function Pos = FindAllMin(x)
%halla las posiciones de los minimos
locales del vector x
der=diff(x);    % Halla la diferencia entre
cada elemento y su antecesor en x
Pos=[];
n=length(der);
for i=2:n,
  if der(i)>=0 & der(i-1)<0,
    Pos=[Pos i];
  end;
end;

% Funcion que determina los puntos
maximos locales de F0
function P= FindAllMax(x)
%halla las posiciones de los máximos
locales del vector x
der=diff(x);P=[];
for i=2:length(der),
  if der(i)<0 & der(i-1)>=0,
    P=[P i];
  end;
end;
```

```
% Funcion que determina los rangos de posiciones de un arreglo donde hay
% ceros
function [cer]=RangoPosCeros(arr)
i=1;
cer=[];
cont=length(arr);
while(i<=cont),
   if arr(i)==0,
      j=i;
      while (j<=cont) & arr(j)==0,
         j=j+1;
      end
      if (j<=cont),
         cer=[cer i j-1];
      end
      i=j-1;
   end
   i=i+1;
end

%Formar los rangos de valores que no contienen ceros de la f01
function [noceros]=NoCeros(arr,n)
cont=(length(arr)/2)+1;
noceros=[];
if arr(1)~=1,
   noceros=[noceros 1 arr(1)];
   i=2;
   j=2*i-1;
   while j<=length(arr),
      if j~=length(arr)-1,
         noceros=[noceros      arr(j-1) arr(j)];
      else
         noceros=[noceros      arr(j-1) arr(j)];
         noceros=[noceros arr(j+1) n];
      end
      j=j+2;
   end
else
   i=2;
   j=2*i-1;
   while j<=length(arr),
      if j~=length(arr),
         noceros=[noceros arr(j-1) arr(j)];
      else
         noceros=[noceros arr(j) n];
      end
      j=j+2;
   end
end
```

Anexo 16

Código fuente de la función Silaba_mia.

```
function [E]=silaba_mia(infile)
global filename n_samples C0 C1 pos PosMax au y
if isempty(filename),
if (nargin < 1)
[filename,pathname,filterindex]=uigetfile('*.wav','Data Selection');
   filename=[pathname,filename];
else
   filename=infile;
end
end
[x,fs,wmode,fidx]=readwav(filename,'s',-1,-1);
w=round(0.03*fs);%30 ms windows
s=round(0.005*fs);%every 5 ms
envent=enframe(x,hamming(w),s);%Hamming for spectral analysis
envent2=enframe(x,w,s);%Rectangular for energy
mycc=[];
sz=size(envent);% number of rows
for i=1:sz(1)
   %cepstral coefficients matrix
   cep=rceps(envent(i,:));
   cep=cep(1:10);%includes CC0
   mycc=[mycc ; cep];
   %Energy
   ventana = envent2(i,:);
   E(i)= sum(ventana.^2)/length(ventana);
end
for t=6:(sz(1)-5)
   %Transition measure
     for i=1:10
      a(i)=0;
      nsum(i)=0;
      for n=t-5:t+5
         a(i)=[a(i)+(mycc(n,i)*n)];
         nsum(i)=[nsum(i)+n^2];
      end
      D(t)=mean((a(i).^2)/nsum(i));
   end
end
%Time scale in ms
ms=[30:5:(sz(1)*5+30)-1];
D=[0 0 0 0 0 D];
B = [0.0032 0.0069 0.0169 0.0294 0.0397 0.0437 0.0397 0.0294 0.0169 0.0069 0.0032];
D = filter(B,1,D);
E= filter(B,1,E);

EnDeriv=[0 diff(E)];
```

```
i=2:length(EnDeriv);
PosMax=find(EnDeriv(i)<=0 & EnDeriv(i-1)>0);%Todos los maximos
PosMin=find(EnDeriv(i)>0 & EnDeriv(i-1)<=0);%Todos los minimos
%vector de las x
y=0:length(x)/length(E)/fs:(length(x)-1)/fs;
a=zeros(1,length(E));
a(PosMax)=max(E);
pos = get(0, 'screensize'); % get the screensize
sWi = pos(3);
sHe = pos(4);
 WIDTH   =round(0.9375*sWi);
HEIGHT  =round(0.43*sHe) ;
LEFT    =round(0.025*sWi);
BOTTOM  =round(sHe-HEIGHT-80);
nmF='Segmentation into syllabic units';
sF =  figure('Units', 'Pixels', 'Position', [LEFT BOTTOM WIDTH HEIGHT],...
          'Pointer','crosshair',...
          'Resize','on','Name',nmF,'NumberTitle','Off');
plot(y,E);
hold on
grid
zoom
MinSep=round(0.1*fs);%Minima separación entre sílabas>1/100mS(ó 10/seg)(en muestras)
MinMaxAmp=0.05*mean(E);%Minima Amplitud de un maximo para ser tenido en cuenta=3*EnMin
WinLength=w;
PosAllMax=PosMax;%Dejar una copia de todos
i=1;
% MinEn=min(E);
% amplitud=MinMaxAmp*MinEn;
amplitud=MinMaxAmp;
while i<length(PosMax),%Con este while quedan dos en las esquinas
   if i==1,
      val2=PosMax(i);
      coc=MinSep/round(WinLength/4);
      val4=PosMax(i+1);
      eposmax=E(val2);
      eposmax3=E(val4);
      if (val4-val2 < coc & eposmax < eposmax3)...
          | (eposmax < amplitud),
            PosMax(i)=[];%Eliminar los que estan muy cercanos (el de menor En), o En<K*EnMin
            i=i-1;
      end
   else
```

```
    val2=PosMax(i);
    val3=PosMax(i-1);
    coc=MinSep/round(WinLength/4);
    val4=PosMax(i+1);
    eposmax=E(PosMax(i));
    eposmax2=E(PosMax(i-1));
    eposmax3=E(PosMax(i+1));
    if ( val2-val3 < coc & eposmax < eposmax2)...
        | (val4-val2 < coc & eposmax < eposmax3)...
        | (eposmax < amplitud),
          PosMax(i)=[];%Eliminar los que estan muy cercanos (el de menor En), o En<K*EnMin
          if i>3,%corregir el count(hay menos elem.) pa'q' chequee al previo en las new cond.
             i=i-2;
          else
             i=i-1;

end;%MinTrans*Energy(PosMax(i))>max([min(Energy(PosMax(i-1):PosMax(i))) min(Energy(PosMax(i):PosMax(i+1)))]),
    end;
  end
  i=i+1;
end
a=zeros(1,length(E));
a(PosMax)=max(E);
% plot(y,a,'m')%En miyan los grandes bien separados
%Ahora analizar los que quedan para determinar silabas (tercera condicion)
PosMaxSG=PosMax;%Una copia de los separados y de gran amplitud
MinTrans=0.88;%Minimo descenso de Energy para separar dos silabas=25%
i=1;
while i<length(PosMax),%
  if i==1,
minim2=min(E(PosMax(i):PosMax(i+1)));
    eminim=E(PosMax(i));
    if ( minim2> MinTrans*eminim),
      PosMax(i)=[];%Eliminar donde no hubo transicion,(cruce por En<K*EnMin)
      i=i-1;
    end
  else
   %if min(Energy(PosMax(i-1):PosMax(i))) > MinTrans*(min([Energy(PosMax(i-1)) Energy(PosMax(i))])...
    % );%+ abs(Energy(PosMax(i-1)) - Energy(PosMax(i)))/10 ),%la diferencia influye 1/10
    minim= min(E(PosMax(i-1):PosMax(i)));
```

```
    eminim=E(PosMax(i));
minim2=min(E(PosMax(i):PosMax(i+1))
);
    if ( minim > MinTrans*eminim) |...
          ( minim2> MinTrans*eminim),
          PosMax(i)=[];%Eliminar donde no hubo transicion,(cruce por En<K*EnMin)
          i=i-1;
    end;
  end
  i=i+1;
end
if i==length(PosMax),
  minim= min(E(PosMax(i-1):PosMax(i)));
  eminim=E(PosMax(i));
  trans=MinTrans*eminim;
  if ( minim > trans),
    PosMax(i)=[];%Eliminar donde no hubo transicion,(cruce por En<K*EnMin)
  end
end
for i=1:length(PosMax)-1,
  val=PosMax(i);
  val2=PosMax(i+1);
  dif=val2-val;
    PosMax(i)=PosMax(i)+floor(dif/5);
   PosMax(i)=PosMax(i)+floor(dif/3);
end
a=zeros(1,length(E));
a(PosMax)=max(E);
plot(y,a,'r')%En rojo los considerados como silabas
au=a;
i=2;
PosOnset=[];
while i<length(PosMax),
  StartPos=find( E == min(E(PosMax(i-1):PosMax(i))) );%comienzo un seg. en el min.
  SegLength=0;
  while StartPos < PosMax(i),

j=StartPos+1:PosAllMax(min(find(PosAllMax>StartPos)));%un segmento de pendiente positiva
    if SegLength < sum( sqrt(  (E(j)-E(j-1)).^2) ),
      SegLength=sum( sqrt(  (E(j)-E(j-1)).^2) );
      TempPosOnset=find(EnDeriv == max(EnDeriv(j)));
    end;
StartPos=PosMin(min(find(PosMin>max(j))));
  end;
```

```
  PosOnset=[PosOnset TempPosOnset];
  i=i+1;
end;
% Buttons.
pos = get(0, 'screensize'); % get the screensize
sWi = pos(3);
sHe = pos(4);
% WIDTH   =round(0.9375*sWi);
WIDTH   =round(0.7*sWi);
HEIGHT  =round(0.43*sHe) ;
LEFT    =round(0.025*sWi);
BOTTOM  =round(sHe-HEIGHT-40);
le=round(0.2*WIDTH);
bo=round(0.174*HEIGHT);
wi=round(0.773*WIDTH);
he=round(0.739*HEIGHT);
AXISLOC = [le bo wi he];
xywh = get(sF, 'Position');
axi=AXISLOC;
left = 10;
wide = 80;
top  = xywh(4) - 10;
high = 22;
high=22;
if 9*(22+8) > xywh(4),
   high=17;
end;
inc  = high + 8;
po=length(x);
p2=length(a);
C0=1;
C1=floor(PosMax(1)*(po/p2));
for (i=2:length(PosMax)),
   C0= [C0 C1(i-1)];
   C1= [C1 floor(PosMax(i)*(po/p2))];
end
pos=[];
pos=1;
plUp = uicontrol('Style', 'PushButton', 'Callback', 'playfmio("sel")', ...
        'HorizontalAlign','center', 'String', 'Play',...
        'Position', [left top-high wide/2 high]);
```