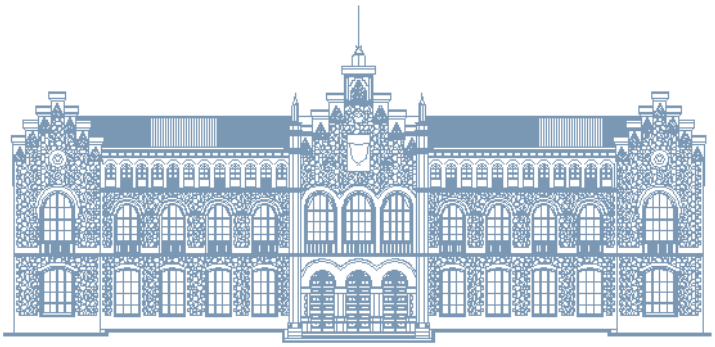
UNIVERSITAT POLITÉCNICA DE CATALUNYA

Escuela de Ingeniería de Terrassa

Ingeniería Técnica de Telecomunicaciones, especialidad Imagen y Sonido



Trabajo Fin de Carrera

Síntesis Audiovisual de la voz

Autor: Bárbara Godayol Roca

Tutor: Ignasi Esquerra Llucià

Mayo 2010

ÍNDICE

ANEXOS:

AGRADECIMIENTOS

Al tutor Ignasi Esquerra por el continúo seguimiento que ha tenido sobre mi trabajo.

A mis padres por su disponibilidad en todo lo que he necesitado.

A mi hermano por su ayuda y sus ideas.

A José Mª por darme esos buenos momentos de tiempo libre que tanto se necesitan.

CAPITULO 1. Introducción

Este proyecto de fin de carrera titulado **síntesis audio-visual del habla** ha sido realizado en el laboratorio de Teoría *del senyal i comunicacions (TSC)* de la Escuela de Ingeniería de Terrassa (EET) durante el periodo de Febrero de 2010 a Mayo de 2010. La especialidad de los estudios han sido de Ingeniería Técnica de Telecomunicaciones, especialidad en Imagen y Sonido y ha sido supervisado por el profesor Ignasi Esquerra.

El proyecto tiene como principal objetivo el desarrollo de una aplicación de *"talking head", e*s decir, mostrar la imagen de una persona hablando con una voz sintética. Cuando se habla de "talking head" lo primero en que se piensa es en modelos 3D realizados a través de ordenador, que emulan las figuras humanas. Este tipo de "talking head" es el más conocido dado que son los más extendidos en la red. Pero esto no significa que no existan trabajos que la parte visual este compuesta por una persona hablando, como por ejemplo los trabajos de Eric Cosatto y Hans Peter Graf[1] o Kyoung-Ho Choi[2]. En este proyecto, se ha querido dar un toque más personal, por eso se ha escogido estudiar un "talking head" con imágenes de personas.

1 Photo-Realistic Talking-Heads from Image Samples

2 Automatic Creation of a Talking Head From a Video Sequence

La síntesis de voz es un concepto que está en expansión y en continuo estudio dado que aporta una gran ayuda a personas discapacitadas y además es un nuevo sistema de interacción entre persona-ordenador.

Para llevar a cabo el objetivo planteado, tanto audio como video serán tratados por separado para luego unirlos y crear finalmente la síntesis audiovisual del habla.

Se ha creído conveniente dividir el proyecto en tres fases importantes, la 1ª será obtener la **base de datos**, dónde se obtendrán los archivos de video y audio, se almacenarán e indexarán de forma que la búsqueda para su posterior uso sea fácil y rápida. Como 2ª fase se hablará de **programación**, dónde se creará el programa principal para la realización del objetivo deseado. En está fase se manipularán tanto los archivos de video como los archivos de audio para poder unirlos y crear las frases finales, además en esta fase se creará la voz sintética con el programa *Festival*[3] . Y finalmente, la 3ª fase será la llamada **interfaz de usuario** que permitirá al usuario un fácil uso de la aplicación.

Como ya se ha comentado, audio y video serán tratados por separado. En audio se desea hacer una síntesis por fonemas (unidad básica de la lengua).

3 Festival Speech Synthesis System

Festival se encargará de dividir las palabras en fonemas y posteriormente unirá los fonemas necesarios para la creación final de la palabra.

Por ejemplo, si se quiere analizar la palabra *casa*, lo primero que hará *Festival,* es separar la palabra en fonemas:

/k/ /a/ /s/ /a/

Una vez obtenidos los 4 fonemas, se encargará de buscar en su base de datos que archivos contienen esos fonemas, los unirá, y creará la voz sintética que pronunciará dicha palabra.

En la síntesis de video se busca conseguir el mismo resultado. En este caso, la partición se realizará mediante visemas (imagen visual del articulema (postura de los órganos articulatorios durante la emisión del fonema)).

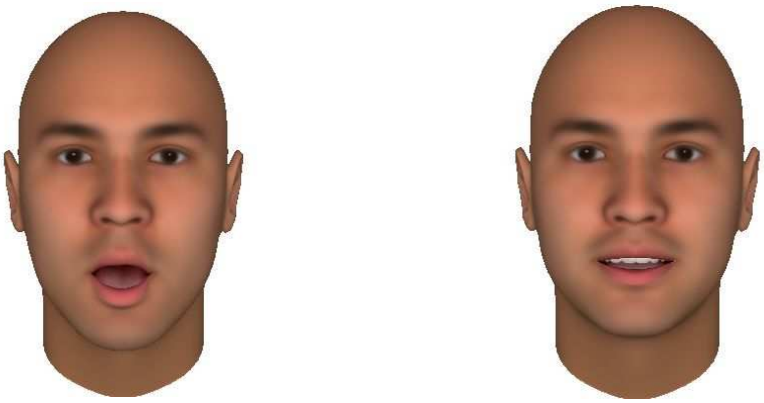


Fig. 1 Visemas para las vocales /a/ /i/

En el caso del proyecto, las imágenes escogidas no serán creadas por ordenador si no que serán extraídas de archivos de video.

Al contrario que en la síntesis del audio, ahora no se cuenta con un programa como *Festival*, que nos facilite el trabajo. Por lo tanto se habrá de hacer manualmente. Por lo tanto, se ha de crear un programa que realice tanto la búsqueda como la división de los archivos y finalmente la unión de estos.

Como se ha comentado anteriormente los visemas son la imagen visual de los fonemas, por lo tanto para realizar la búsqueda de los visemas se utilizará la división realizada en los archivos por fonemas.

Así pues el esquema que describe el funcionamiento del proyecto es el siguiente:

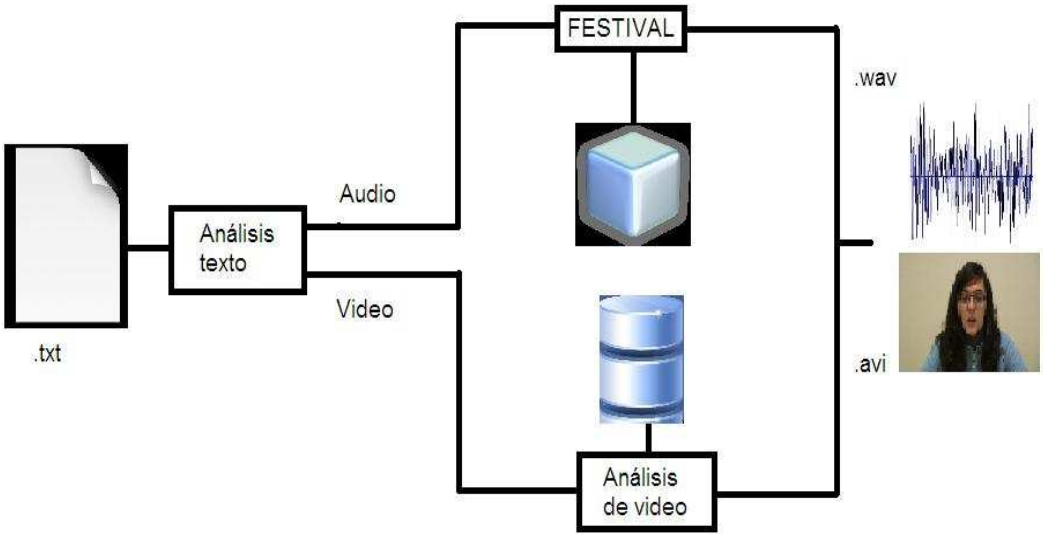


Fig. 2 Esquema principal del Proyecto

El proyecto está pensado para un uso exclusivo en *Linux* debido a la facilidad que este software libre da a los usuarios para descargar, instalar e utilizar programas sin necesidad de obtener

ningún tipo de licencia ni pagar por ella.

Antes de seguir, hay que comentar que el proyecto ideal, el cual ha sido explicado anteriormente, debido a problemas y limitaciones en el periodo de trabajo, no ha podido ser llevado a cabo en su totalidad. La síntesis de video no ha llegado a ser por visemas, sino que se ha tenido que adaptar a una síntesis por palabras.

Debido a esta adaptación, el programa se limita a buscar palabras almacenadas en la base de datos, es decir, la síntesis se verá reducida al número de palabras almacenadas.

Aun así, con estas deficiencias se ha logrado crear un programa que cumple la mayoría de requisitos, como son el análisis correcto del audio, la unión de los archivos, o una interfaz de usuario sencilla.

CAPITULO 2. Estado del arte

En este capitulo se hablará de las diferentes herramientas que existen en el mundo tecnológico para poder llevar a cabo el proyecto.

2.1 Codificación de video y audio

El primer paso en la realización de cualquier aplicación basada en archivos multimedia es obtener los archivos que serán necesarios. En este proyecto, se han obtenido a través de una cámara digital, esto ha permitido que el paso de cámara a ordenador sea más fácil.

Estos archivos almacenan tanto señal de video como de audio. Normalmente los archivos al ser pasados al ordenador son comprimidos para que ocupen el menor espacio posible, la compresión puede presentar pérdidas o no.

En el caso que la compresión sea con pérdidas se obtienen archivos finales con menor calidad que los iniciales, pero esto no supone un problema, porque las pérdidas suelen ser pequeñas y los archivos finales ocupan menos. Si la compresión es sin pérdidas, la calidad no se verá afectada pero no se conseguirá que los archivos ocupen poco. Como se puede ver el proceso de compresión es muy importante, ya que puede variar el espacio de memoria requerido por la aplicación. Este proceso se encuentra en la codificación del archivo, y dependerá del formato que se utilice

en cada caso.

Por lo tanto antes de cualquier manipulación de archivos se debe tener en cuenta cual es la codificación del material que será utilizado.

2.1.1 Formatos

Los diferentes archivos adquiridos pueden encontrarse en diferentes formatos, tanto de video como de audio.

• Vídeo

MPEG (Moving Picture Expert Group)

Este grupo de trabajo se encarga de desarrollar estándars de codificación de audio y video y de la creación de normas auxiliares, como estándars de meta datos, entre otros. Estas son las partes para estándars de video:

- **MPEG-1 (part 2):** códec de vídeo para señales no entrelazadas (progresivas).

- **MPEG-4 (part 10), AVC o H.264:** es una norma que define un códec de vídeo de alta compresión. La intención del proyecto fue la de crear un estándar capaz de proporcionar buena calidad de imagen con tamaños de memoria notablemente inferiores a estándars previos.

DV VIDEO

El formato DV (*Digital Video*) es un estándar de vídeo de gama doméstica, industrial y broadcast. Fue desarrollado como formato digital de vídeo para un entorno industrial, pero su excelente relación calidad-precio provocó que se haya convertido en el formato predominante en el vídeo doméstico, como *Mini DV*, y que hayan surgido versiones profesionales, *DVCAM* y *DVCPRO*.

WMV (Windows Media Video)

Es el nombre genérico que se le da a un conjunto de algoritmos de compresión ubicados en el grupo de tecnologías de video desarrolladas por *Microsoft*. Este formato se combina casi siempre con el sonido de formato *Windows Media Audio.*

THEORA

Es un códec de video libre, desarrollado por la fundación *Xiph.org*[4] como una parte de su proyecto *Ogg.* Basado en la codificación de video con perdidas. Generalmente se encuentra dentro del contenedor *Ogg.*

En la actualidad la codificación en **alta definición** está ganando terreno y se puede encontrar en cualquier archivo multimedia.

BLU-RAY DISC

También conocido como Blu-ray o BD, es un formato de disco óptico de nueva generación para el vídeo de alta definición y almacenamiento de datos de alta densidad.

4 http://www.xiph.org/

ARCHIVOS .TOD

Los archivos con extensión .TOD son archivos de video en alta definición. Estos archivos se crean utilizando el codec MPEG2 HD.

• Audio

MPEG

Hay diferentes posibilidades para escoger códigos de audio:

- **MP3:** MP3 es el acrónimo de MPEG-1 layer 3. Es un formato de audio digital comprimido con perdidas. Fue el primer formato de compresión de audio popularizado gracias a Internet.

- **MPEG-2 Audio**: introduce algunas mejoras respecto MPEG-1, como por ejemplo poder codificar más de dos canales. Además permite descodificadores compatibles con MPEG-1.

VORBIS

Codificador de audio libre, de compresión con perdidas. Forma parte del proyecto *Ogg*. Se encuentra al mismo nivel que MPEG-2.

WMA (Windows Media Audio)

Es el formato de audio que *Microsoft* ha diseñado. Inferior técnicamente a formatos como MP3 o Vorbis.

Waveform Audio Format

Es un formato de audio digital normalmente sin compresión de

datos. Desarrollado y propiedad de *Microsoft* y de *IBM* que se utiliza para almacenar sonidos en el *PC,* su extensión es *.wav.* es el formato principal usado por *Windows*.

2.1.2 Contenedores multimedia

Los archivos de audio y video se suelen encontrar almacenados juntos en otros archivos llamados contenedores multimedia.
Un contenedor multimedia es un tipo de archivo informático que almacena información de video, audio, meta datos... siguiendo un formato preestablecido en su especificación.
Algunos ejemplos de archivos contenedores que podemos encontrar actualmente son:

MPEG

- **MPEG-4 (part 14):** el nombre oficial que se utiliza para llamarlo es mp4. Fu diseñado para transportar audio y video, aunque también soporta otros corrientes de datos como subtítulos o imágenes estáticas.

OGG

Es un formato desarrollado por la fundación *Xiph.org.* Diseñado para dar un alto grado de eficiencia en el *streaming* y en la compresión de archivos.

AVI (Audio Video Interleave)

Fue definido por Microsoft. Posteriormente fue mejorado mediante

las extensiones de forma del grupo OpenDML de la compañía Matrox. El formato avi permite almacenar simultáneamente un flujo de datos de video y varios flujos de audio.

2.2 Programas utilizados

El siguiente proceso realizado en el proyecto es la programación del programa. Para llevarlo a cabo será necesario conocer los diferentes lenguajes y programas disponibles en el entorno *Linux.*

2.2.1 Scripts

Un *script* o archivo de órdenes es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano. Los script son casi siempre interpretados, pero no todo programa interpretado es considerado un script.

El uso habitual de los scripts es realizar diversas tareas como combinar componentes, interactuar con el sistema operativo o con el usuario. Por este uso es frecuente que los shells (Linea de comandos) sean a la vez intérpretes de este tipo de programas. Los archivos *script* suelen ser identificados por el sistema a través del encabezamiento en el contenido del archivo.

BASH

Está basado en la shell de Unix. Fue escrito para el *proyecto GNU* y es el intérprete de comandos por defecto en la mayoría de las distribuciones de Linux. Su nombre es un acrónimo de *Bourne-Again Shell* .

La linea de encabezamiento para convertir el lenguaje bash en un script es:

#!/bin/bash

AWK

Es un lenguaje de programación diseñado para procesar datos basados en texto, ya sean ficheros o flujos de datos. AWK fue una de las primeras herramientas en aparecer en Unix.

PERL

Es un lenguaje de programación diseñado por Larry Wall en 1987. Perl toma características del lenguaje C, del lenguaje interpretado shell (sh), AWK, sed, Lisp y, en un grado inferior, de muchos otros lenguajes de programación.

La linea de encabezamiento para convertir el lenguaje Perl en un script es:

#!/usr/bin/perl

2.2.2 Procesado de video

Un paso importante en la síntesis audiovisual del habla es el procesado de vídeo. Con este procesado obtendremos los archivos de video finales que serán vistos por el usuario.

Ffmpeg

Es una colección de software libre que puede grabar, convertir y hacer streaming de audio y vídeo. Incluye *libavcodec*, una biblioteca de códecs. Está desarrollado en *GNU/Linux*, pero puede ser compilado en la mayoría de los sistemas operativos, incluyendo Windows.

Mplayer

Es un reproductor multimedia multiplataforma liberado bajo la licencia *GPL*. Reproduce la mayoría de los archivos. Además junto al paquete de descarga de MPlayer, se puede encontrar la aplicación *MEncoder*, una herramienta esencial para el proceso de codificación de vídeo o audio. El reproductor puede funcionar en la mayoría de las plataformas.

VLC

Es un reproductor multimedia y framework del proyecto VideoLAn; es software libre distribuido bajo la licencia GPL. Soporta muchos códecs de audio y video, así como diferentes tipos de archivos, además de DVD, VCD y varios protocolos streaming. También puede ser utilizado como servidor en unicast o multicast. Con versiones para Linux, Windows, Mac, entre otros.

2.2.3 Procesado de audio

El siguiente paso importante para nuestra síntesis será el procesado de los archivos de audio, que consiste en la unión o segmentación de dichos archivos, entre otras funciones.

Festival

Es un sistema de Síntesis de voz de propósito general para múltiples lenguajes, desarrollado originalmente por el Centro de Investigación de Tecnologías del Lenguaje de la Universidad de Edimburgo. Se distribuye como software libre.

El proyecto incluye la documentación completa para desarrollar sistemas de síntesis de voz con varios APIs, siendo un entorno ideal para el desarrollo e investigación de las técnicas de síntesis de voz.

El proyecto está escrito en lenguaje C++ y está implementado como un interprete de comandos, el cual puede conectarse con diversos

módulos y aplicaciones. El proyecto festival es multilingüe (actualmente soporta inglés (británico y americano), y castellano) aunque el inglés es el más avanzado. Además algunos grupos han desarrollado herramientas que permiten utilizar otros idiomas con el proyecto.

Las herramientas y la documentación completas para la utilización de nuevas voces en el sistema están disponibles en el proyecto FestVox de la *Carnegie Mellon University*. El proyecto Festvox pretende hacer de la construcción de voces sintéticas nuevas un proceso más sistemático y mejor documentado, haciendo lo posible para que cualquiera pueda construir nuevas voces.

PRAAT

Praat es una aplicación que da especial importancia al estudio de síntesis vocal mediante todo tipo de análisis: espectral, pitch, formants y intensidad entre otros. Así pues, el objetivo de Praat se centra en el análisis y la síntesis del habla, de modo que se logre manipular grabaciones de voz y crear gráficos de gran calidad.

2.2.4 Manipulación archivos multimedia

Para la creación de la base de datos es necesario manipular los archivos multimedia obtenidos a través de la cámara de video. Dicha manipulación se realiza a través de programas de edición de video y audio.

SONY VEGAS

Es un sistema profesional de edición no lineal, originalmente publicado por *Sonic Foundry*, ahora es propiedad y está gestionado por *Sony Creative Software*. Aúna una potente edición de audio y vídeo en una única y completa plataforma de creación.

MEDIACODER

Es un transcodificador de audio y video, publicado bajo la licencia libre *GNU/GPL* y compatible con los sistemas operativos *Windows* y *Wine*. Es desarrollado desde 2005 por Stanley Huang, un creador independiente. Puede convertir una gran variedad de formatos de vídeo y audio mediante códecs y herramientas de código abierto. Entre sus usos se pueden citar: compresión, conversión de formatos, extracción de audio de archivos de video y streaming.

AVS VIDEO EDITOR

Es un software producido por *Online Media Technologies Ltd.* Es un editor de video para realizar todo el proceso, la importación de los vídeos, sonido e imágenes, crear la película, exportarla a un archivo, incluso hasta grabarla a CD o DVD.

WAVESURFER

WaveSurfer es una editor de audio, de código abierto, utilizado en estudios de fonética acústica. Es un programa simple pero potente para la visualización interactiva de formas de onda de presión acústica, espectrogramas y transcripciones. Puede leer y escribir una serie de formatos de archivo de transcripción utilizados en la investigación del lenguaje industrial. Se ejecuta en la mayoría de plataformas, incluyendo Microsoft Windows, Mac OS X, Linux. Está escrito a base de scripts y permite la instalación de plugins.

CAPITULO 3. Fases proyecto

Como ya se ha explicado en el apartado Introducción, se ha decidido dividir el proyecto en 3 fases importantes. En este apartado se pretende explicar paso a paso la realización del proyecto, además de los problemas ocasionados y las soluciones adoptadas.

3.1 Creación base de datos e indexación

Creación de la base de datos e indexación ha sido la fase clave para realizar las fases posteriores. Gracias a esta fase se han obtenido los archivos de video y de audio, necesarios para el proyecto.

En un principio se contaba con 150 archivos de video y 150 de audio, que almacenarían las frases necesarias para obtener una buena síntesis. Finalmente, como ya se explicará a continuación, se acabaron grabando únicamente 20 frases, por lo tanto 20 archivos de audio y 20 de video.

Una base de datos es un conjunto de datos que pertenecen al mismo contexto, guardados sistemáticamente para su posterior uso. Para llevar a cabo este proyecto ha sido necesaria su implementación debido a la gran magnitud de archivos de audio y de video utilizados.

Ha sido conveniente crear una base de datos donde tener los archivos almacenados e indexados.

Mediante la indexación se ha conseguido mayor agilidad en las búsquedas, lo cual se traduce en mayor rapidez a la hora de mostrar resultados.

Para crear la base de datos lo primero de todo fue obtener el corpus de frases, que consiste en una lista de frases que contienen todos los posibles fonemas, sonidos y reglas de la lengua española. Es decir, frases que ayudarán a tener la mayoría de fonemas posibles.

En este primer paso empezaron a aparecer los primeros problemas. Según explican los creadores de Festival, el número mínimo de frases es de unas 460 frases, y el proyecto empezó contando con 150 frases (corpus de frases)[5] y se acabó grabando 20. Se grabaron 20 con la intención de que estas servirían para hacer las primeras pruebas y después cuando ya se tuviera todo el programa grabar las frases que faltaban, pero por falta de tiempo no se pudo realizar el paso de aumentar la base de datos. Así pues, la base de datos va a ser muy limitada en cuanto a posibilidades, pero esto no provocará que el proyecto no cumpla con las necesidades que en un principio se dictaron.

El siguiente problema apareció cuando se tuvo que grabar a la persona hablando. Podía existir un desfase entre voz e imagen debido a la diferencia de velocidades (luz y sonido).

5 ANEXO 1: CORPUS DE Frases

La solución que se adoptó fue utilizar un micrófono externo a la cámara, aunque conectándolo a ella, y situarlo lo más próximo a la persona que en ese momento estaba hablando. Además se pensó que si se iba a tratar por separado audio y vídeo, esto no iba a suponer un problema como se había creído en un principio.

Las imágenes fueron grabadas con una cámara JVC EVERIO[6], que graba en alta definición, los archivos resultantes estaban almacenados en archivos multimedia `.TOD`.

Estos archivos complicaban el trabajo, debido a su gran calidad ocupaban mucha memoria. Además los archivos `.TOD` no son muy conocidos y la mayoría de programas no saben como manejarlos. Así que antes de cualquier tipo de manipulación se tuvo que hacer una transcodificación con el programa *Sony Vegas*. Ahora los nuevos archivos se encuentran en `.avi`, este nuevo tipo de archivo facilitará su posterior uso.

La grabación fue continua, es decir, se grabó una frase tras otra. Por lo tanto, antes de separar audio de video se tuvo que proceder a separar las frases. Cuando las frases estuvieron divididas solo fue necesaria la ayuda de MediaCoder para poder separar el audio del video.

6 JVC Everio GZ-HD3 Compatc HD Camcorder

Una vez obtenidos los archivos necesarios había que pensar como almacenarlos e indexarlos para que la búsqueda y el posterior uso fuera sencillo y rápido.

Sabiendo que para crear voces en *Festival* se necesita seguir una forma predeterminada, se optó por utilizar esta forma de indexación y almacenamiento para todo el proyecto.

Según indica *Festival*, los archivos almacenados deben ser llamados por las iniciales del locutor, seguidas del número de frase que almacenan. Además, necesita unos archivos extra (`.lab`), estos archivos deben tener el mismo nombre que el archivo al que pertenecen y *Festival* los utiliza para buscar los fonemas en los diferentes archivos de audio.

Así pues, cada frase debe tener un archivo `.lab` dónde se indiquen los fonemas que incluye y la duración de estos.

Los archivos de audio se manipularon utilizando el programa W*avesurfer* que permitió una segmentación más sencilla, gracias a que permite ver la forma de onda de las señales y así visualmente detectar cambios de frecuencia, periodicidad, silencios…

El proceso de división en fonemas y el almacenaje de los tiempos se fue haciendo frase a frase.

Para los archivos de video se crearon también archivos `.lab`, pero en este caso la división fue por palabras y se introdujeron los tiempos de inicio y final de palabra. Para la partición de palabras se han utilizado editores de video que han ayudado a poder leer los labios mientras se escuchaba la palabra y así conseguir una partición más exacta.

Estos archivos se crearon cuando en la 2ª fase, se tuvo una idea de como realizar el programa para la síntesis de video. Además, para que la búsqueda fuera más fácil, una vez creados los 20 archivos `.lab` se creó un catálogo donde se almacenaron todos juntos.

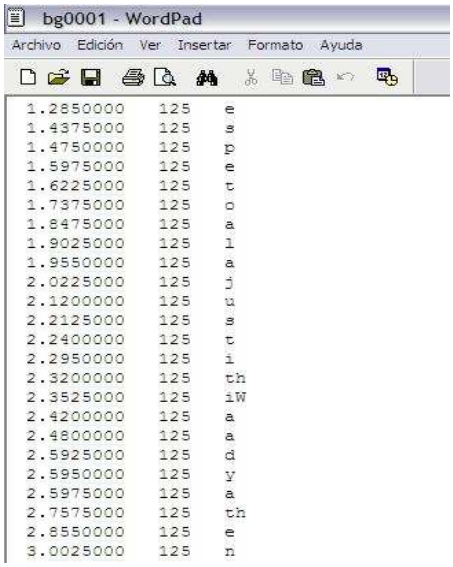


Fig. 3 Archivos `.lab` para Festival

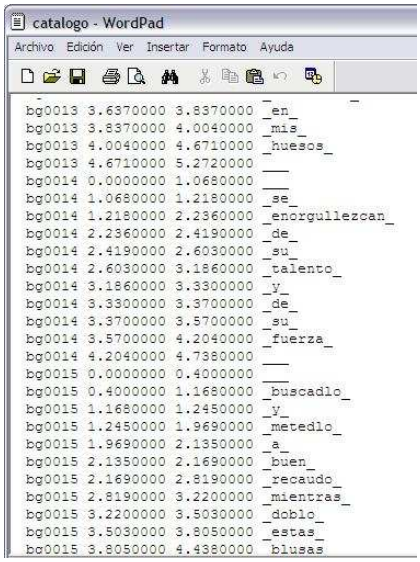


Fig. 4 Catálogo de palabras almacenadas

Otro de los problemas que apareció fue la precisión en la que debían ser divididos tanto los fonemas como las palabras. Una mala partición podría llevar a que el resultado final fuera nefasto. Este problema fue el más largo en ser solucionado, ya que es difícil

controlar si la partición esta bien hecha, o si por el contrario ocasiona problemas… Los archivos fueron revisados una y otra vez buscando imperfecciones.

Además fue un problema que aún siendo dada por acabada la fase de *creación de base de datos* se tuvo que retomar, cambiando de nuevo alguna partición debido a que ocasionaba problemas cuando se realizaba la 2ª fase.

Aun así, una vez finalizado el proyecto se puede decir que la partición final tanto de audio como de video es correcta.

3.2 Programación

Está fase ha sido la que más tiempo ha durado debido a su complejidad y a la falta de experiencia en el mundo de la programación en lenguajes como *bash* o *Perl* o la creación de scripts para programas como *Praat o Festival.*

Hasta conseguir el objetivo final esta fase ha sufrido diferentes cambios y diferentes etapas, que serán comentadas a continuación.

3.2.1 Creación de la voz Sintética

La primera etapa de la fase de programación consiste en el estudio y aprendizaje de crear voces sintéticas con *Festival* para obtener

finalmente una voz sintética con los archivos anteriormente almacenados.

Gracias a la base de datos de archivos de audio que se había creado y al manual *Building Synthetic Voices*[7] (apartado *unit selection databases*), la creación de la voz sintética fue más sencilla.

Las instrucciones necesarias para crear la voz fueron las siguientes. Todas ellas llevadas a cabo mediante la linea de comandos (shell) de Linux.

1. Crear un directorio donde se van a almacenar los archivos necesarios para dicha voz. Este directorio debe llamarse de la siguiente forma: nombreasociacion_idioma_inicialeslocutor

```
$ mkdir cmu_es_bg3
```

2. Una vez creado el directorio (carpeta), hay que cambiar el directorio actual por el nuevo.

```
$ cd cmu_es_bg3
```

3. La siguiente instrucción indica a Festvox que introduzca en el directorio todas las carpetas y archivos necesarios para la creación de la voz.

```
$FESTVOXDIR/src/unitsel/setup_clunits cmu es bg3
```

FESTVOXDIR: variable que indica el directorio donde se encuentra festvox.

7 http://festvox.org/bsv/

4. Ahora hay que llenar las carpetas que se han obtenido. Para ello, se han de copiar los archivos .wav y .lab, creados en la base de datos, en las carpetas /wav/ y /lab/ respectivamente.

5. A continuación, hay que añadir en la carpeta /etc/ un archivo con extensión `.data`. Este archivo debe incluir TODAS las frases almacenadas siguiendo el siguiente esquema:

   `( nombre_archivo "Frase que incluye" )`

En este proyecto se ha creado un archivo txt.done.data[8] que incluye las 20 frases almacenadas.

6. Las siguientes instrucciones deben ser llamadas tal y como indica el manual para que festival se encargue de crear finalmente la voz.

```
$ Festival -b festvox/build_clunits.scm '(build_prompts "etc/txt.done.data")'

$ Festival -b festvox/build_clunits.scm '(build_utts "etc/txt.done.data")'

$ ./bin/make_pm_wave wav/*wav
$ ./bin/make_mcep wav/*wav

$ Festival -b festvox/build_clunits.scm '(build_clunits "etc/txt.done.data")'
```

Al seguir las instrucciones anteriormente indicadas, la voz sintética ya está creada.

Como resultado se ha obtenido una carpeta contenedora, con el nombre cmu_es_bg3 (nomenclatura indicada por Festival).

8 ANEXO 2: txt.done.data

· ***cmu*** : asociación que ha creado la voz (debería ser UPC, pero como es una voz en pruebas, se ha utilizado un nombre al azar)

· ***_es_*** : idioma, en este caso español

· ***bg3*** : iniciales de la persona que ha grabado los archivos de audio.

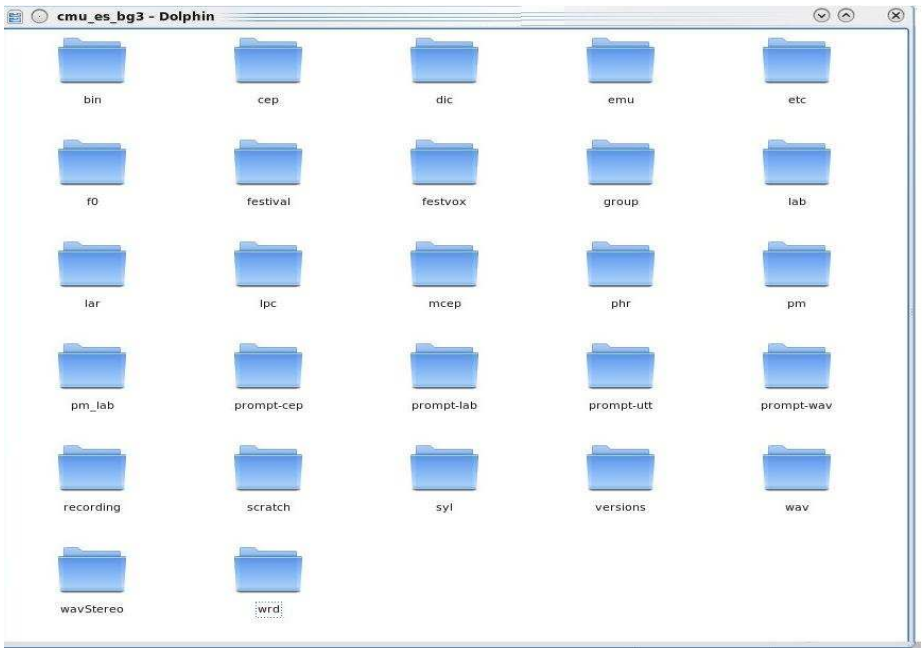


Fig. 5 Carpeta ***cmu_es_bg3***

3.2.2 Vídeo por palabras

Una vez obtenida la voz sintética se creyó conveniente pensar en un prototipo del proyecto. Este prototipo pretende crear el programa que va a realizar la síntesis de video, es decir, crear un programa que realice funciones como *Festival* pero para la síntesis de video.

En el prototipo fue dónde se empezó a hablar de síntesis de palabras y no de fonemas/visemas. La partición era más fácil y rápida. Los resultados obtenidos tras las segmentaciones y las uniones se analizarían mejor.

El prototipo debía seguir los siguientes pasos:

1. Leer texto, separado por palabras
2. Buscar palabras en el catálogo
3. Segmentar los videos según la duración de cada palabra
4. Unir los archivos para obtener la frase final
5. Reproducir

Para llevar a cabo el prototipo se utilizaron los archivos iniciales, dónde audio y video aún estaban unidos. Además se utilizaron programas como *Mencoder* para la segmentación y unión de archivos, y *Mplayer* para reproducir el resultado final.

Lo primero que se hizo fue buscar las funciones que *Mencoder* ofrecía para realizar los pasos 3 y 4. Se encontraron funciones que permitían además de cortar videos y unirlos, seguir manteniendo los formatos de codificación de los archivos. Con esto se consigue que las duraciones halladas en la base de datos sean más exactas, ya que si se cambia el formato de codificación podían variar propiedades como los frames por segundo y esto llevaría a un cambio en la duración, entonces la segmentación no sería correcta, es decir, no correspondería con la palabra escogida.

La función utilizada para el Paso 3:

```
mencoder ./avi/bg0004.avi -ss 00.00 -endpos 00.250 -ovc copy
-oac copy -o ./programa_sintesis/silencio$contador.avi
```

Esta función permite seleccionar el inicio y la duración de cada corte. Ademas con -oac y -ovc se puede escoger el formato de codificación tanto de audio como de video respectivamente. En el ejemplo, copy indica que se quiere mantener el formato del archivo inicial.

La función utilizada para el Paso 4:

```
mencoder archivo1.avi archivo2.avi -ovc copy -oac copy -o
archivo_salida.avi
```

Para utilizar está función únicamente es necesario indicar los archivos que se van a unir, el formato de codificación que se va a utilizar y finalmente indicar cual va a ser el archivo de salida.

Gracias a estas funciones, las duraciones del archivo catálogo coincidían con las palabras seleccionadas, y los segmentos creados con *Mencoder* eran correctos, por lo tanto el resultado final sería el esperado.

Debido a que audio y video iban unidos, solo con la partición de un archivo `.avi` se conseguía la síntesis audiovisual.

A partir de aquí se puede decir que se ha conseguido un programa

que permite la lectura de ficheros de texto, separarlos por palabras y obtener una síntesis adecuada.

Este prototipo simplemente consiste en dos scripts escritos bajo el lenguaje *Perl*, un script principal que procesa y busca el texto que el usuario introduce, y un script secundario que segmenta y une los videos para cada palabra.

3.2.3 Vídeo por palabras + Festival

Una vez creada la voz sintética y el prototipo, había que conseguir unirlos en una sola aplicación.

La clave para conseguir el objetivo fue crear un diagrama de flujos donde se viera el proceso que se tenía que seguir para una buena síntesis.

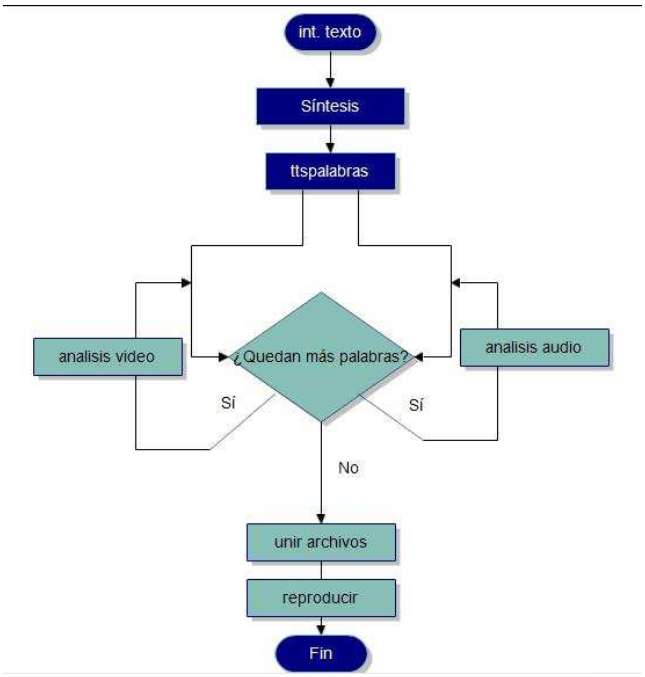


Fig. 6 Diagrama de flujos

Cuando finalmente se obtuvo un buen esquema que cumplía todos los requisitos, era la hora de llevar a cabo la programación.

La aplicación consistirá en un script principal (`Síntesis`[9]) que se encargará de llamar a los demás scripts y a la interfaz gráfica.

Sintesis es un script escrito en lenguaje bash, este script se encarga de leer el texto, de modificarlo poniendo en cada línea una palabra diferente, de pasarlo todo a minúsculas para aumentar las formas posibles de poner una palabra, y finalmente de buscar en el catálogo las diferentes palabras.

A continuación, cuando `Síntesis` ya tiene el texto preparado lo pasa al script ***ttspalabras***[10] que será el encargado de hacer el análisis. Este nuevo script leerá palabra por palabra y realizará la síntesis.

En este script se trata audio y video por separado como se ha comentado ya anteriormente, aunque finalmente se acaban uniendo dichos archivos para una síntesis final. Es decir, debido a que ahora video y audio se tratan por separado, los archivos utilizados para la síntesis de video ya no deben ser los iniciales como en el caso del prototipo, ahora la base de datos para la síntesis de video debe contener los archivos que únicamente tienen video.

9 ANEXO 3: script Sintesis
10 ANEXO 4: script ttspalabras

La síntesis de audio se realizará modificando archivos `.scm` que incluyen las instrucciones para *Festival (*fstvl.scm[11]*)*.

Después, se llamará al programa *Festival*, el cual creará un archivo `.wav` por cada palabra analizada.

La síntesis de vídeo, se realizará utilizando el programa creado en el prototipo.

Así pues el proceso que debe seguir el script ttspalabras es el siguiente:

1. Leer palabra
2. Crear archivo de video, cortando video necesario.
3. Crear archivo de audio con Festival
4. Unir archivo de video con audio
5. Unir archivos para la creación de la frase final

El script `ttspalabras` se ha creado sobre los scripts utilizados en el prototipo, por lo tanto solo habrá que añadir las funciones 3 y 4, que son las que añaden el audio al proyecto.

Para el paso 3 se ha llamado al programa *festival*, y se le ha pasado como parámetro de entrada las instrucciones de *festival*.

11 ANEXO 7: script fstvl.scm

Como ya se ha comentado anteriormente, tras esta operación obtendremos un archivo .wav con la palabra seleccionada.

```
festival -b instrucciones_festival.scm
```

Para llevar a cabo el paso 4, se ha utilizado la siguiente función que ofrece *Mencoder.*

```
mencoder archivo_video.avi -o archivo_salida.avi -ovc copy -oac copy
-audiofile archivo.wav
```

Cuando se realizaron las primeras pruebas con los archivos de video y *Mencoder* apareció un problema, los archivos de video estaban almacenados en `.mp4` y Mencoder no trabaja bien con este tipo de archivos. Así pues, se tuvo que volver a la fase creación de base de datos para cambiar todos los archivos `.mp4` por `.avi`.

Otro problema que se encontró, fue que los archivos de audio y vídeo habían sido creados por diferentes métodos, entonces las duraciones eran diferentes, y por lo tanto el resultado final no era el adecuado. Cuando se presentó este problema lo primero que se pensó fue en modificar uno de los archivos, video o audio, para que los dos tuvieran la misma duración y no presentarán problemas al unirlos con *Mencoder*.

Se escogió la opción de modificar el audio, ya que existe el programa *Praat* que mediante scripts `.praat` permite modificar la duración de los archivos. En las primeras pruebas no se tuvo en cuenta que *Festival* cuando crea los `.wav` añade un silencio tanto

antes como después de la palabra. Así pues, al realizar la modificación de los archivos de audio, el resultado obtenido era desastroso, las voces sonaban muy aceleradas y no quedaban bien cuadradas con lo que en ese momento se veía en la imagen.

Una vez se observó que estos silencios modificaban tanto el resultado final se optó por eliminarlos utilizando una función que *Festival* ofrece.

```
ch_wave -start inicio -end fin audio.wav > corto_audio.wav
```

Esta función se encuentra dentro del paquete speech_tools. Esta función nos permite escoger el inicio y el final del trozo de audio que queremos recortar. Como parámetro de entrada pasamos el archivo a modificar.

Ahora si, con los archivos de audio más exactos la modificación con *Praat* dio un resultado más correcto.

Para solucionar los problemas comentados anteriormente se han creado dos scripts más, que son `cambia_duracion`[12] y `quita_silencios`[13].

---

12 ANEXO 5: script cambia_duracion

13 ANEXO 6: script quita_silencios

cambia_duracion es un script escrito en Perl que recibe como parámetro de entrada un archivo de texto donde se encuentran las duraciones del audio y del video, además del nombre del archivo que se está modificando.

En este script se leen las duraciones y se procede a calcular el factor de escalabilidad por el cual será modificado el archivo de audio.

Factor = dur_video/ dur_audio

En este mismo script se modifica el script `instrucciones.praat`[14] que contiene las funciones necesarias para que el programa *Praat* realice el cambio de duración en los archivos de audio.

quita_silencios es un script escrito bajo el lenguaje *Perl*. En este script se eliminan los silencios que Festival introduce antes y después de cada palabra.

Gracias a una función que incorpora *Festival*, se pueden almacenar archivos `.lab` para cada palabra. Es decir, se obtendrán archivos `.lab` que indicarán los tiempos de cada fonema. Estos archivos una vez obtenidos en `ttspalabras`, serán los que se pasarán al script `quita_silencios`.

14 ANEXO 8: scripts instrucciones.praat

En este script se leerán los tiempos de inicio y fin de cada palabra para poder llamar a la función mencionada anteriormente, y cortar los archivos de audio, eliminando así los silencios.

Cuando los problemas ocasionados por la diferencia en las duraciones estuvo solucionado, ya se podía dar por finalizada la parte de Síntesis. Ahora solo era necesario reproducir, pero aquí apareció otro problema.

Al reproducir los archivos no se veían correctamente debido a que mientras se estaba cargando el programa (*Mplayer*), ya empezaba la reproducción. Es decir, los primeros milisegundos del archivo no eran vistos correctamente.

La solución que se adoptó en este caso fue crear unos videos sin información antes y después de la frase para que este problema no afectará al resultado de la Síntesis.

Después de resolver todos los problemas que iban apareciendo y de seguir el diagrama de flujos que se había planteado en un principio, se puede decir que la parte de programación se ha resuelto correctamente y que se obtienen unos resultados buenos, aunque la síntesis de video final no sea exactamente como se pretendía en un principio.

3.3 Interfaz de usuario

La 3ª fase ha sido la parte más sencilla y más rápida. La incorporación de la interfaz se ha realizado en el script `Sintesis`.

En este script se han creado funciones necesarias para que funcionará correctamente, además se han añadido frases en la interfaz como las instrucciones de uso, o indicadores de los pasos que está realizando la aplicación en cada momento.

Las funciones que se han creado en el script `Sintesis` para la interfaz de usuario son `Start_display()`, que permite abrir un display dónde recaerán las frases que se vayan escribiendo, otra de las funciones ha sido `check_quit()`, que a parte de permitir salir de la aplicación, cierra también la interfaz.

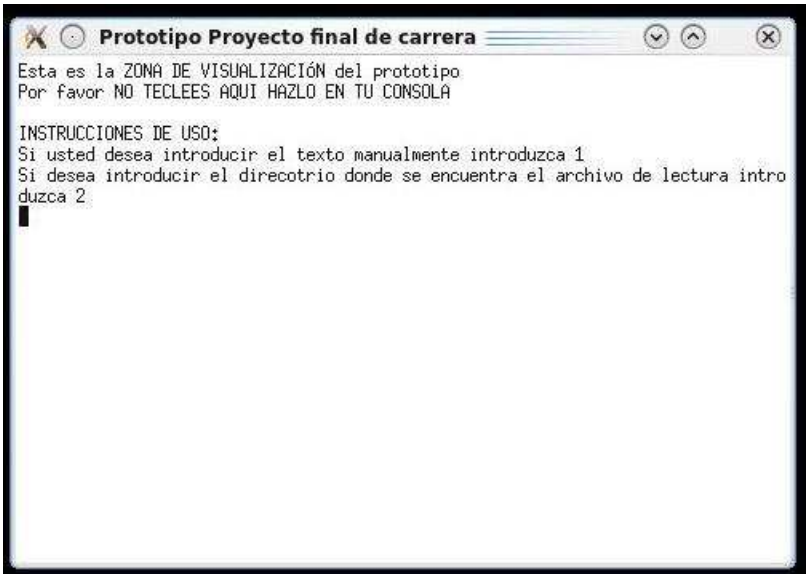


Fig. 7 Interfaz de usuario

CAPITULO 4. Instalación y ejecución

En este apartado se hablará de los programas que deben ser instalados antes de poder utilizar la aplicación: *Síntesis audiovisual del habla*. Además se dará una breve explicación de como puede hacerse. Finalmente se explicará como el usuario puede añadir está aplicación en su ordenador y utilizarla.

4.1 Instalaciones previas

Linux ofrece lenguajes de programación sin necesidad de instalar librerías externas, ya que vienen por defecto al instalar *Linux.* Estos lenguajes han sido los utilizados para realizar la aplicación, es decir, no será necesario instalarlos. En cambio, los programas utilizados como F*estival*, *Mplayer*, entre otros no vienen por defecto con Linux.

Así pues para una perfecto uso de la aplicación, habrá que instalar los siguientes programas (el orden que se marca no es necesario seguirlo, pero si lo es el cumplimiento de todos ellos).

1. Festival, que incluye el paquete festvox. Pero habrá que instalar también el paquete speech_tools para que funcione todo correctamente.

2. Mencoder y Mplayer

3. Praat

4. Wavesurfer

A continuación una breve descripción de como instalar programas en Linux.

1. Se debe escribir lo siguiente en la línea de comandos:

```
$ sudo apt-get install Praat Wavesurfer MEncoder MPlayer
```

A continuación te pedirá la contraseña de super usuario y ya estarán instalados los programas.

4.1.1 Festival

Aunque Festival viene con la mayoría de distribuciones Linux y se puede cargar con el *aptitude*, solamente viene con una voz en español que no es muy buena y manca de algunas herramientas extras que necesitamos. Por lo tanto, si la instalación de *Festival* se realiza como se describe en el apartado 1, uno de los paquetes extra necesarios para el buen funcionamiento de la aplicación no se instalará y la aplicación no funcionará correctamente.

Antes de instalar festival hay que asegurarse que se tienen instaladas las librerías: *Lcurses, gcc y g++*. La librería *Lcurses* se encuentra dentro del paquete *ncurses*.

Así pues para instalar las librerías se debe escribir:

```
$ sudo apt-get install libncurses5-dev gcc g++
```

A continuación se han de descargar los archivos indicados en la tabla y descomprimirlos en una carpeta creada, en el disco local, previamente con el nombre festival-1.96 por ejemplo.

Nombre	URL	Descripción
festival-1.96-beta.tar.gz	http://www.cstr.ed.ac.uk/downloads/festival/	Distribución del código fuente de Festival
festvox-2.0-release.tar.gz	http://festvox.org/download.html	Distribución del código fuente de Festvox
speech_tools-1.2.96-beta.tar.gz	http://www.cstr.ed.ac.uk/downloads/festival/	Herramientas para Festival (Edimburgo)
multisyn_build-1.8.tgz	http://www.cstr.ed.ac.uk/downloads/festival/	Entorno para usar el engine MultiSyn
festvox_ellpc11k.tar.gz	http://www.cstr.ed.ac.uk/downloads/festival/	Voz española *el_diphone*[15]

A partir de ahora, el directorio creado pasará a ser el directorio base, donde se instalaran las herramientas necesarias y Festival.

15 el_diphone es necesario para el desarrollo de nuevas voces sintéticas en español.

La instalación de las herramientas como del propio Festival, se ha de realizar en este orden, ya que es importante el orden en que se instalen los paquetes:

1. Instalar speech_tools (herramientas extras de festival)

```
#festival-1.96/speech_tools> ./configure
#festival-1.96/speech_tools> make
```

2. Instalar festvox (herramienta de festival para crear voces)

```
#festival-1.96/festvox> ./configure
#festival-1.96/festvox> make
```

3. Instalar festival

```
#festival-1.96/Festival> ./configure
#festival-1.96/Festival> make
```

4. Instalar voces (necesaria para crear voces en español) mediante la descompresión del paquete "**festvox_ellpc11k.tar.gz**" sobre el directorio base /festival-1.96.

5. Para comprobar que todo ha sido instalado correctamente se pueden ejecutar los siguientes comandos:

```
#cd festival-1.96/Festival/bin
#festival-1.96/Festival/bin> ./festival
#festival$ (voice_el_diphone)
#festival$ (SayText "hola mundo")
```

En este ejemplo, en primer lugar (línea 2), se ha puesto en funcionamiento el motor de síntesis Festival (en su versión 1.96). Después, en la línea 3, se ha cargado la voz en español *el_diphone*. Finalmente, en la línea 4, mediante (SayText “”), se ha escrito un texto que será sintetizado por la voz *el_diphone*.

4.2 Instalación de la aplicación

Este apartado esta creado especialmente para explicar como ejecutarlo, ya que no necesita ningún tipo de instalación.

La ejecución es muy sencilla, el usuario simplemente habrá de escoger el espacio donde añadir la carpeta contenedora (Síntesis_del_habla), que contiene todos los scripts y carpetas para que la aplicación funcione correctamente.

El usuario deberá recordar el directorio donde se encuentra dicha carpeta para su posterior uso, además de comprobar que todos los archivos mostrados en la **Fig. 8** están incluidos en su carpeta.

Si los programas externos a la aplicación, anteriormente indicados, han sido instalados correctamente el proceso de ejecución de la aplicación no presentará ningún problema.

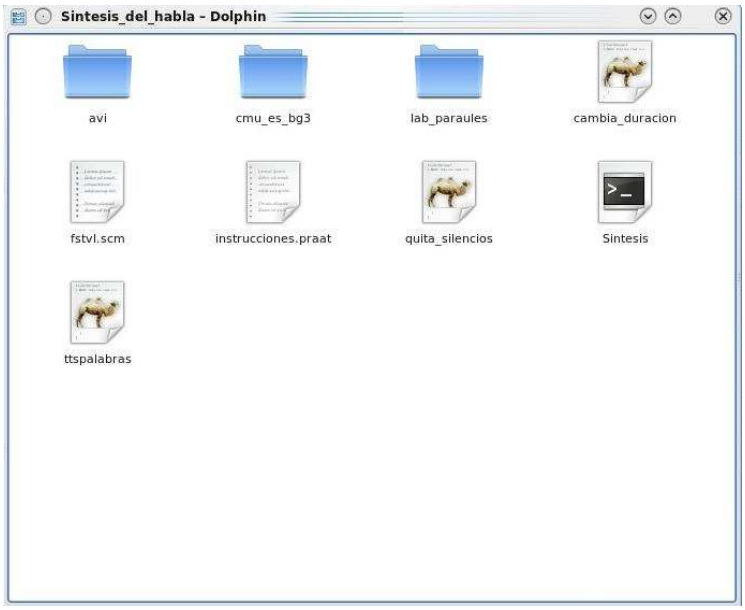


Fig. 8 Carpeta contenedora de la aplicación

Para llevar a cabo la ejecución de la aplicación, en primer lugar el usuario deberá abrir una terminal, en la cual cambiará el directorio actual por el de la carpeta contenedora de la aplicación.

Ejemplo: $*cd Escritorio/scripts/Sintesis_del_habla*

Una vez dentro de la carpeta, sólo será necesario ejecutar el programa escribiendo lo siguiente en la línea de comandos:

./Sintesis

Ahora si, el programa ya está listo para usarse, la interfaz de usuario se abrirá y mostrará como moverse dentro del programa.

Hay que tener en cuenta que cada vez que se ejecute la aplicación se creará una carpeta temporal llamada *programa_sintesis* y se

eliminará la anterior si esta existiera. Dentro de esta carpeta se crearán los archivos de salida, es decir, los archivos `.avi` que contienen el resultado final de la síntesis. Por lo tanto, si el usuario desea guardar los archivos creados, deberá copiarlos en otra carpeta para que estos no sean eliminados.

CAPITULO 5. Conclusiones

En un principio, la inexperiencia en el uso del sistema Linux y en los lenguajes de programación que este ofrecía, provocó un gran esfuerzo a la hora de aprender su funcionamiento y sus posibilidades. Aun así, hay que agradecer todo este esfuerzo que ha permitido obtener un mayor nivel de experiencia tanto del sistema operativo, como de aplicaciones y programas (*Mencoder, Mplayer, Praat, Bash, Perl, openOffice*...).

Al final del proyecto se han llevado a cabo la mayoría de los objetivos que se plantearon en un principio. Aunque como ya se ha comentado, la síntesis de video por visemas ha quedado reducida a una síntesis por palabras. Aun así, se ha desarrollado completamente una aplicación que ofrece la posibilidad de realizar una Síntesis audiovisual del habla.

La parte del proyecto dedicada al audio se ha conseguido tal y como se esperaba. *Festival* ha sido el principal programa en toda esta fase. Conseguir una voz sintética con *Festival* no ha sido una tarea fácil, puesto que encontrar un buen manual donde se explique como crear la voz en castellano con tus propios archivos ha sido complicado.

Una parte clave para la creación de voces sintéticas es la base de datos. Como ya se ha comentado, la base de datos es muy escasa y por lo tanto la voz sintética creada no es lo que en un principio se

esperaba. Se ha podido observar que los acentos crean uno de los mayores problemas, además debido a que se han tenido que modificar los archivos de audio para reducir su tamaño, se puede apreciar un leve ruido de fondo. Aun con todo esto, lo que realmente se buscaba en el proyecto era conseguir aprender y crear una voz sintética, y esto si que se ha llevado a cabo correctamente.

En el caso de la parte de vídeo, la aplicación que se ha creado es totalmente correcta, funciona de una forma sencilla y rápida y da al usuario un resultado satisfactorio. Por lo tanto, que la partición sea por palabras y no por visemas no es más que una limitación del proyecto, pero que no ha influido para obtener un buen resultado.

En la base de datos del video ocurre lo mismo que en la del audio. Pero en este caso las deficiencias son más visibles. A parte de las pocas frases almacenadas, estas no han sido grabadas en las mejores condiciones. La imagen no es totalmente fija, hay cambios de posición del locutor, cambios en la mirada e incluso en el color de fondo. Estas deficiencias en el audio no se notan, pero a la hora de visualizar el resultado no se ve una imagen correcta, es decir, dependiendo de como y cuando estuvo grabada la frase, el locutor sale mirando a la derecha o a la izquierda, o con la cabeza movida.

Cuando las palabras escogidas para la síntesis se encuentran en diferentes archivos donde el locutor cambia constantemente, al unir los videos la imagen del locutor no para, es decir, aparece en

un movimiento continuo que no es muy agradable a la vista. Estos efectos además son más visibles en palabras cortas como /a/ /la/ /con/, debido a que su duración es mínima los cambios son más bruscos.

Aún con estas deficiencias el objetivo del proyecto era obtener una aplicación que dividiera los videos en las palabras que se indicaban y que video y audio quedaran sincronizados.

Como se ha podido observar en los videos realizados este objetivo se ha conseguido satisfactoriamente. La selección de la palabra se hace correctamente, y tras la unión video con audio, éstos quedan sincronizados.

Así pues, se puede considerar que el resultado final del proyecto es correcto. Además se puede asegurar que, simplemente modificando el apartado de análisis de video y cambiando la base de datos por una más detallada, se puede llegar a obtener el objetivo marcado en un principio (Sintesis por visemas) y con mayor calidad.

5.1 Futuros proyectos

Durante la realización del proyecto, han ido apareciendo diversas ideas que pueden convertirse en futuros proyectos, por eso se ha creído conveniente crear un apartado que las recoja.

Todas las ideas están basadas en el proyecto Síntesis audiovisual

del habla. Es decir, las ideas tendrán como objetivo mostrar una síntesis audiovisual pero cada nueva idea ofrecerá o una mejora, o una nueva forma de conseguir el objetivo.

• Proyecto inicial

La primera idea que se planteó fue conseguir el proyecto idílico. Es decir, conseguir una partición por fonemas en audio y visemas en video.

Como ya se ha comentado, la partición de audio se ha conseguido en *Síntesis audiovisual del habla*, mediante el programa Festival. Así pues esta idea se basa en conseguir la síntesis de video por visemas.

Para llevar a cabo esta idea se necesitará modificar la base de datos de la que parte el análisis de video, los archivos `.lab` ya no deberán estar divididos por palabras sino por fonemas.

Como se ha comentado anteriormente la base de datos tanto en audio como en video es escasa, además el video presenta un movimiento continuo en el locutor, que enmascara cualquier buen resultado. Así pues en esta nueva idea, también se ha pensado aumentar la base de datos y mejorarla. Es decir, mejorar la grabación de archivos, asegurar que los cambios entre imágenes es mínima y aumentar el número de frases almacenadas. Todos estos cambios conllevarían a crear de nuevo una voz sintética con festival, además de volver a realizar las particiones oportunas.

• Síntesis a dos voces

Cuando se realizó el paso de grabar a los locutores, se grabaron a dos personas para poder elegir que grabación era mejor a la hora de la Síntesis. Aquí apareció la segunda idea, se pensó que se podría modificar la aplicación principal permitiendo al usuario elegir la voz y la imagen que quería utilizar para su síntesis. En un principio se ha pensado crear una voz e imagen femenina y una masculina.

Así pues, el usuario podría hacer cualquier elección, tanto voz femenina con imagen femenina, como voz femenina con voz masculina y viceversa.

Para llevar a cabo esta nueva idea, la base de datos debería ser modificada tanto para el audio como para el video. Además de crear nuevas voces sintéticas, se debería modificar el script principal `Sintesis`, añadiendo la opción de escoger voz e imagen. Debido a la complejidad de los pasos mencionados anteriormente, en este caso la síntesis se realizaría por palabras como en la aplicación base, aunque esto no implica que las nuevas bases de datos puedan contener más información.

• Síntesis modificando video

Cuando se observó que las duraciones de los archivos no coincidían y que por lo tanto Mencoder presentaba problemas, apareció la siguiente idea.

La 3ª idea que se planteó fue modificar la duración de los archivos de video y no los de audio.

En principio se pensó realizar modificaciones en los archivos de video parecidas a las que realiza *Praat* en los archivos de audio.

Praat para modificar la duración de los archivos utiliza un factor de escalabilidad. Con este factor *Praat* realiza interpolaciones según el factor indicado. En el caso que el factor se encuentre entre 0.5 y 2 los resultados son correctos, pero cuando se superan estos limites *Praat* empieza a presentar problemas.

Por lo tanto, en esta idea lo que se busca es crear un programa que nos permita introducir un factor como en el caso de *Praat* y modificar el archivo de video introducido, añadiendo o eliminando frames.

Para realizar esta idea habría que estudiar más profundamente como funcionan los vídeos, y además habría que encontrar o crear una aplicación que permita copiar o eliminar *frames* para conseguir la duración deseada.

Este proceso es bastante largo y duro. Debido a la gran carga que puede suponer este nuevo cambio, se ha pensado que la síntesis sería la misma que en un principio, es decir, por palabras. Aunque si sería necesario modificar la base de datos para mejorarla.

• Síntesis observando los videos

La 4ª idea se basa en los visemas. Esta idea surgió durante la indexación, es decir, cuando se estaban creando los archivos `.lab`.

En esta idea se pretende que la búsqueda de archivos no se realice mediante catálogos o archivos externos a los videos si no que se realice mediante comparaciones entre los archivos de video y unas imágenes base.

Para llevar a cabo esta idea, principalmente se deberán conseguir las imágenes que mostrarán la posición de los labios para cada fonema, estas imágenes pueden ser creadas por ordenador, como se ha mostrado en la **Figura 1** del apartado introducción.

A continuación se necesitará un programa que analice uno a uno los archivos, y que vaya comparando la posición de los labios en cada momento, con la imagen base que corresponde al fonema que se está buscando.

Hasta aquí consiste la idea principal, pero también se ha pensado que el programa creado podría añadir una función que comparará los videos escogidos tras la búsqueda, con los que ya han sido sintetizados. Con esta nueva función se conseguiría disminuir el continuo movimiento del locutor.

CAPITULO 6. Referencias

[1] http://www.linusakesson.net/programming/tty/index.php

[2] http://www.wikipedia.org/

[3] http://logopediaperu.blogspot.com/2009/05/visemas-y-kinemasmaterial-de-apoyo-para.html

[4] http://tecnologiaeducativa.portafolioseducativos.org.ve/te/?p=1280

[5]http://www.fon.hum.uva.nl/praat/manual/Intro_8_2__Manipulation_of_duration.html

[6] http://www.mplayerhq.hu/DOCS/HTML-single/es/MPlayer.html

[7] http://guimi.net/blogs/hiparco/uso-de-mencoder/

[8] http://festvox.org/

ANEXO I: Corpus de frases

0001: Con estoico respeto a la justicia adyacente guardó sus flechas.
0002: Fue inyectado en el abdomen y en una pierna.
0003: La tensión volvió a aumentar el domingo.
0004: Cuando todavía eran baratos el vodka y el caviar.
0005: Sino el país mental de un patriota enloquecido.
0006: Abría sus puertas a estos flacos alumnos afroamericanos.
0007: Palestina seguía en estado paupérrimo después de su duunvirato.
0008: El presidente de la Federación Portuguesa de Fútbol.
0009: Bajo los efectos de la hipnosis tocaba la guzla perfectamente.
0010: El oftalmólogo dijo que le hizo entrar en un proceso subfebril.
0011: Como ya se comprobó recientemente en Sudán y Afganistán.
0012: En el posfranquismo se apuntó al partido de Areilza.
0013: Bautizaba mi piel y se inmiscuía en mis huesos.
0014: Se enorgullezcan de su talento y de su fuerza.
0015: Buscadlo y metedlo a buen recaudo mientras doblo estas blusas.
0016: Esta población causa cierta preocupación a las familias.
0017: Pero la coincidencia de los flecos no puede obviarse.
0018: La cultura subbética no es anterior a la anteislámica.
0019: Tiene un nombre polaco con connotaciones judías.
0020: Ha pasado la noche adyuntándolo todo y clamando al cielo.
0021: Se trataba de un tramoyista huidizo y cuellierguido.
0022: Los aztecas no usaban el callialto por ser muy abrupto.
0023: Comió en un restaurante en compañía de su esposa.
0024: Debemos desyemarlo antes de comerlo.
0025: Después de reiteradas amenazas de su exmarido infructuosas.
0026: Pañuelo y castañuelas no forman hiato.
0027: Poco antes de la inauguración del curso en Cataluña.
0028: Fue abyecto, vil, obcecado, etcétera.
0029: Y al salir de la subprefectura se fueron a comer unas jaibas.
0030: Los campeones y subcampeones de las ligas de España.
0031: Arzálluz se reunió con él hace una semana y media.
0032: Pero muy peligrosa cuando reúne fuerza financiera.
0033: Eduardo Peña Abizanda y Francisco Serrano no son druidas.
0034: La figura del alcalde era constantemente eclipsada.
0035: Olaf es del barrio de Santa Eulalia pero juega al golf.
0036: Con un buen solenoide se consigue un gran ding dong.
0037: En sus ojos la luz del Ibaizábal flotaba como una golondrina.
0038: Todo ello se hunda desnortado, depauperado, etcétera.
0039: Al oír el pitido puede decirte que va bien.
0040: Iré después de que el mediador estadounidense vaya al rugby.
0041: Ni que siempre actúen como polluelos desjuiciados.
0042: El subgobernador tiene una plantación de caucho.
0043: Una pequeña rotura fibrilar del músculo abdominal.
0044: La foto de los ñus fue la primera que puso en el álbum.
0045: No se estaba prejuzgando el resultado del proceso.
0046: No fue al club del Garraf por un problema de amígdalas.
0047: Conseguir que despegue con objetivos en las urnas.
0048: El Real Madrid ampliará el aforo del Bernabéu.
0049: Su reingreso es debido a una indigestión por exceso de flúor.
0050: Baraja los naipes mientras evalúo el resultado.
0051: También le transmitieron que es preferible un buen pacto.
0052: Si es tan obvio el interés, cogednos y metedlo en vuestro club.
0053: Con un hórrido fascista de zarzuela.

0054: La agresión de Azpeitia sería una excepción.
0055: En el comunicado emitido ayer por la Alianza Atlántica.
0056: Os adjunto los archivos correspondientes a la nueva adjudicación.
0057: Cataluña es una nación y España no, y tiene equipo de rugby.
0058: Baobab deificaba a Job tras su postdoctorado.
0059: Algunos prefieren el pavo guienés y otros el gazpacho.
0060: Luzbel pensaba que la vida era jauja y no daba aguinaldos.
0061: Y comieron habichuelas al compás de las castañuelas.
0062: Fui huidizo como Melchor Miralles y Amando de Miguel.
0063: Lucía Urigoitia falleció en un tiroteo.
0064: Borges se inventó algo para Buenos Aires.
0065: Ciudad autónoma de Melilla y Junta de Andalucía.
0066: Víctimas del engreimiento o la voluptuosidad, su ego les puede.
0067: Aproximadamente un millón de españoles sufren glaucoma.
0068: Tanto palestinos como israelíes no se pueden esconder.
0069: Se pintó la cara con achiote y guardó el arma en el tahalí.
0070: Aullar y maullar son cosas parecidas.
0071: Se había adoptado en Manlleu hacía tiempo y estaba compungido.
0072: Utilizaron cauchotina para impermeabilizarlo, resguardarlo, etcétera.
0073: Formaban un buen tándem trabajando el caucho.
0074: Con ese carácter abnegado ahuyentas a todo el mundo.
0075: La jusbarba o brusco es una planta esmilácea.
0076: Pero haremos bien tomando precauciones si otra vez se acerca.
0077: Hemos plantado un concepto positivo en nuestro huerto.
0078: Que ayer se reincorporó a los entrenamientos.
0079: En Léijar tuvieron que desratizar tres viviendas.
0080: Lo argumentó el coordinador general de la materia.
0081: No obyecto nada sobre el tema del subfusil.
0082: Pero cuando llueve riesgo equivale a caída, quizá es mejor no ir rápido.
0083: Almunia le dejó claro a Aznar todo el asunto, fue una pasada.
0084: Con animarle a esforzarse para obtener una satisfacción.
0085: Aznar se entrevistará el miércoles con Julio Anguita.
0086: Continúan siendo los países con mayor protección social.
0087: Era una niña preciosa con dos hoyuelos enormes.
0088: Se saltó el stop y salió haciendo zigzag.
0089: No soportaba el tictac del reloj, decían en el Kremlin.
0090: Hijo del también matador Paquirri y de Carmen Ordóñez.
0091: Seguían obcecadamente las palabras del subprefecto.
0092: Les debían subministrar armas para el Cáucaso.
0093: Seguía manteniendo una actitud coadyutoria.
0094: Devuelvo la mirada y duermes todavía junto a la albahaca.
0095: El acuerdo no incluye ningún tipo de intercambio económico.
0096: Ya ubicamos el arroyuelo situado al sudsudeste del río Éufrates.
0097: Franco narra sus hazañas bélicas en Marruecos.
0098: Eran palabras extrañas como facsímil, Kafka, Kremlin, gremlins, etcétera.
0099: Barrionuevo obsequió a Pedro Jota Ramírez.
0100: Y para que la emoción no decaiga el atleta lo intenta de nuevo.
0101: ¿Entraña riesgos la genética sintética?
0102: ¿De cuántos estamos hablando exactamente?
0103: ¿Y dices que tienes la colección completa?
0104: ¿Has comido todo lo que te han puesto en el plato?
0105: ¿Desde cuándo dices que come así?
0106: ¿Cómo se supone que vamos a hacerlo?
0107: ¿De qué cantidad exacta estamos hablando?
0108: ¿Cómo se puede enviar a la guerra a un país?
0109: ¿Cómo se va a aceptar que la mujer tome la iniciativa?
0110: ¿Tienes idea de lo bien que sabe?
0111: ¿Han pasado la tarde en esa playa desierta?
0112: ¿Has corrido durante tres horas?

0113: ¿Qué no obtendré siendo realmente santo?
0114: ¿Has estado con Alberto y María?
0115: ¿Hasta cuándo va a estar entre nosotros?
0116: ¿Cuántas veces has estado en su jardín?
0117: ¿Tienes todos los discos de ese grupo?
0118: ¿Quiere alguien explicarme de qué se trata?
0119: ¿Dónde estabas tú en aquel verano del cincuenta y seis?
0120: Has traído todo lo necesario, ¿no?
0121: ¿Hasta dónde dices que eres capaz de llegar?
0122: ¿Desde dónde dices que ha venido?
0123: ¿Quiénes serán los que realmente den la cara?
0124: ¿A partir de qué hora dices que vienen?
0125: ¿Todos estamos esperando para entrar en su casa?
0126: ¿Alguien sabe cómo se puede abrir esto?
0127: ¿A partir de dónde se puede empezar a cortar?
0128: ¿Todos cantan el mismo tipo de canciones?
0129: ¿De veras vas a ir vestido así?
0130: ¡Qué historia la de Juan Y Sonia!
0131: ¡Todos van vestidos de la misma manera!
0132: ¡Quién poseyese otra clase de estropajo!
0133: Nunca me ocurrió nada, ¡gracias a Dios!
0134: ¡Tienes las manos encima de ella!
0135: Para no despertar a Mascardi, abrió la puerta con la mayor suavidad, pero la precaución fue inútil, porque los goznes crujieron. Tomando las cosas en broma, pensó que para la noche convendría comprar una lata de aceite y echar unas cuantas gotas en varias puertas de la casa.
0136: Comprendí entonces, de golpe, por qué todo resultaba, en casa, tan espantosamente complicado; por qué todo rostro que no fuera el de mamá o papá me parecía monstruoso; por qué la sombra de la vecina, al pasar fugazmente por la ventana que daba a la galería, me sumergía en una oscuridad aterradora.
0137: El jinete no vio la soga por ir a lo loco. A duras penas entendía lo sucedido. Abrió los ojos y tras pellizcarse varias veces se acercó al caballo a cuatro patas, se montó y salió de allí lleno de magulladuras.
0138: A lo largo y ancho de nuestra vida jamás hemos sido testigos de semejante cohesión. En consecuencia dejemos a un lado el costumbrismo y nombremos a Juan de Prusia rey del culebrón.
0139: No pudo menos que advertir la diferencia entre una pensión y otra. Depende de la decisión de un puñado de señores, de traje negro, sentados alrededor de una mesa redonda.
0140: Durante días me sumergí en el río como para purificarme de los horrores del pasado, para limpiar mi piel de los estigmas, de aquel olor rancio, el sudor de la muerte, una transpiración hecha de fiebres malignas, ulceraciones y patatas cocidas.
0141: Y, aunque los referéndums le gusten tan poco como a González, difícilmente podrá librarse de convocar uno que le autorice a poner la realidad británica a la hora del resto del continente en asuntos tales como la moneda, los derechos sociales, la Administración local o el sistema electoral.
0142: Los proyectos de doña Esmerenciana, la alcaldesa prihísta, la canalización, la reforestación de la plaza, mejoras en el hospital para lograr la erradicación de la tuberculosis, enfermedad que aqueja a nuestra bella población, la nivelación de los valles, una alcantarilla y una capa de asfalto, etcétera.
0143: Seguí el primer impulso, aunque era arriesgado, pero no ocurrió nada. Se trasladó a la iglesia conmigo algo compungido, pero no sufrió más desgaste y pudimos guardarlo y aislarlo dentro del ataúd.
0144: Entré en una cafetería y tomé un nuevo café. Mientras encendía un cigarrillo me sorprendí a mí misma pensando en cómo puede llegar cierta gente a ciertos puestos de responsabilidad poseyendo una mentalidad a nivel de subcultura. Luego sonreí en mi interior ante el pensamiento de que aquí todos estamos con un timón en las manos que ninguno sabemos manejar.
0145: Garantizan una mayor estabilidad y neutralidad de la política cultural. Quienes critican estos sistemas señalan su tendencia al conservadurismo y al elitismo. Frente a eso está, en su favor, una menor vulnerabilidad ante las críticas, a veces oportunistas, de los medios de comunicación. En todo caso es un mercado dotado ya de una amplia tradición y muy consolidado.
0146: Ron, mulatas, un minúsculo aeropuerto, traficantes de cocaína, caucheros, indios que fingían los dolores del parto mientras daba a luz la mujer, calles sin pavimentar, humedad y un mercadillo lleno de chucherías de contrabando.
0147: Día cuarenta de la marcha: una tarde en Bélgica, en un pequeño nudo ferroviario, los alemanes nos entregaron una vaca y un caballo que vagaban perdidos a orillas de la carretera.
0148: La palabra romanticismo sigue surtiendo su efecto entre algunas mujeres, se lo aseguro, y fueron

muchas las que me llamaron con la intención de devolverme la milagrosa cartulina y recabar de paso alguna información sobre aquel extraño club, información que yo, es obvio, no les regateé en ningún sentido.

0149: José Miguel Arroyo ha visto cómo se frustraba su desinteresada disposición a lidiar seis toros en Las Ventas el doce de mayo a beneficio de los niños y ancianos de Mostar, previa invitación de los cascos azules de la Peña Taurina Medjugorje.

0150: En la carta de su ahijado se pudo leer la siguiente postdata adyacente: Fue un viaje alucinante, vimos de todo: ñus, guepardos, leones cazando a dúo. Todo fue maravilloso.

ANEXO II: txt.done.data

(bg0001 "Con estoico respeto a la justicia adyacente guardó sus flechas.")

(bg0002 "Fue inyectado en el abdomen y en una pierna.")

(bg0003 "La tensión volvió a aumentar el domingo.")

(bg0004 "Cuando todavía eran baratos el vodka y el caviar.")

(bg0005 "Sino el país mental de un patriota enloquecido.")

(bg0006 "Abría sus puertas a estos flacos alumnos afroamericanos.")

(bg0007 "Palestina seguía en estado paupérrimo después de su duunvirato.")

(bg0008 "El presidente de la Federación Portuguesa de Fútbol.")

(bg0009 "Bajo los efectos de la hipnosis tocaba la guzla perfectamente.")

(bg0010 "El oftalmólogo dijo que le hizo entrar en un proceso subfebril.")

(bg0011 "Como ya se comprobó recientemente en Sudán y Afganistán.")

(bg0013 "Bautizaba mi piel y se inmiscuía en mis huesos.")

(bg0014 "Se enorgullezcan de su talento y de su fuerza.")

(bg0015 "Buscadlo y metedlo a buen recaudo mientras doblo estas blusas.")

(bg0016 "Esta población causa cierta preocupación a las familias.")

(bg0017 "Pero la coincidencia de los flecos no puede obviarse.")

(bg0018 "La cultura subbética no es anterior a la anteislámica.")

(bg0019 "Tiene un nombre polaco con connotación judías.")

(bg0020 "Ha pasado la noche adyuntándolo todo y clamando al cielo.")

(bg0150 "En la carta de su ahijado se pudo leer la siguiente postdata adyacente: Fue un viaje alucinante, vimos de todo: ñus, guepardos, leones cazando a dúo. Todo fue maravilloso.")

ANEXO III: Script Sintesis

```
#!/bin/bash

##########################################################################
###
# By Barbara Godayol                                    #
# Programa principal para construir el prototipo                #
# del proyecto sintesis audiovisual de voz                      #
# (Sintesis por palabras)                               #
#                                                 #
##########################################################################
###

##########################################################################
###
# inicios y procesos                                    #
##########################################################################
###

init_programa(){ # creo el directorio donde se almacenaran los archivos creados

# En la carpeta ./programa_sintesis se guardaran todos los ficheros
# de la ejecucion del script
# Se borra la carpeta para eliminar ficheros de ejecuciones anteriores
rm -rf  ./programa_sintesis

# Se crean la carpeta y los ficheros vacios para la escritura del texto

mkdir  ./programa_sintesis
touch  ./programa_sintesis/texto_inicio.txt
touch  ./programa_sintesis/palabras_seleccionadas.txt
rm -rf  ./programa_sintesis/texto_inicio.txt
}
##########################################################################
########
start_display() {

# Arrancamos el display que es la zona de visualizacion de lo que escribe
# el usuario. Nuestro display sera un terminal tipo "xterm"
# (disponible en todos los sitemas unix)
# La ejecucion del xterm guardara el tty del xterm en un fichero (tty.log).
# A continuacion se ejecuta en el xterm un cat que redirige la entrada estandar
# a la salida estandar.

xterm  -T "Prototipo Proyecto final de carrera" -e "tty>./programa_sintesis/tty.log ; cat -"
&
```

```
# Se espera un segundo (comando sleep) a que se cree el terminal anterior
sleep 1

# Se lee en la variable TTY el tty que se guardo en el fichero tty.log
read TTY < ./programa_sintesis/tty.log

# Se escriben las frases de inicio en el display
echo "Esta es la ZONA DE VISUALIZACIÓN del prototipo"  > $TTY
echo "Por favor NO TECLEES AQUI HAZLO EN TU CONSOLA" > $TTY
echo "  " > $TTY
echo "INSTRUCCIONES DE USO:" > $TTY
echo "Si usted desea introducir el texto manualmente introduzca 1" > $TTY
echo "Si desea introducir el direcotrio donde se encuentra el archivo de lectura
introduzca 2" > $TTY
}
##########################################################################
######

introducir_texto () {

# Se guarda en la variable texto.
read funcion

if [ "$funcion" = 1 ]
then
        echo "  " > $TTY
        echo "Introduzca el texto que desee analizar" > $TTY
        read texto
        #guardamos el texto introducido por el usuario en un archivo .txt
        echo "$texto" > ./programa_sintesis/texto_inicio.txt
fi

if [ "$funcion" = 2 ]
then
      echo "  " > $TTY
      echo "Introduzca el directorio del archivo siguiendo este esquema:" > $TTY
      echo "/home/nom_usuari/carpeta_del_archivo/nombre_archivo.txt" > $TTY
        read texto
        #guardamos el texto introducido por el usuario en un archivo .txt
        cp $texto ./programa_sintesis/texto_inicio.txt

fi

echo "  " > $TTY
echo "El texto introducido es: $texto" > $TTY
```

```
# Aplicar funcion para separar el texto en palabras.

sed -e 's/ /\
/g' ./programa_sintesis/texto_inicio.txt > ./programa_sintesis/texto_inicio2.txt

# pasaremos a minusculas todo el texto introducido por el usuario para mejorar la
búsqueda
cat ./programa_sintesis/texto_inicio2.txt | tr [:upper:] [:lower:] >
./programa_sintesis/texto_inicio.txt

#eliminamos texto_inicio2 ya que solo es un documento temporal

rm ./programa_sintesis/texto_inicio2.txt

#Buscamos las palabras del texto en el catalogo y creamos una lista de palabras
necesarias
busqueda_palabras
#Entramos en el corazon del programa
cat ./programa_sintesis/palabras_seleccionadas.txt  |  ./ttspalabras

}

##########################################################################
#############
check_quit () {

# Si el usuario introduce quit se sale del programa.
# Para ello se deben eliminar todos los procesos hijos lanzados, es decir,
# todos los procesos lanzados en background con &. Para mirar los hijos se usa
# el comando pgrep -P.


if [ "$1" = salir ]
        then
        # Se aÃ±anden los PIDs de los hijos del script al fichero pids.log
        pgrep -P $$ > ./programa_sintesis/pids.log

        # Se eliminan todos los PIDS
        kill -9 $(cat ./programa_sintesis/pids.log)

        # Se sale del programa correctamente (valor 0)
        exit 0
fi
}

##########################################################################

procesar_tx (){
echo "  " > $TTY
```

```
echo "EL TEXTO HA SIDO ANALIZADO CORRECTAMENTE" > $TTY

# Bucle infinito que procesa las lineas a transmitir

while true
        do

        # Se lee una linea de la entrada estandar del script
        echo "1.Si desea reproducir el fichero escriba PLAY"
      echo "2.Si desea introducir una nueva frase escriba NUEVA"
      echo "3.Si quiere cerrar la aplicacion introduzca SALIR"

        read mensaje

        # Se mira si el usuario quiere salir
        check_quit $mensaje

        echo "Usted escribio: $mensaje" > $TTY

#Miramos las opciones que el usuario puede elegir
if [ "$mensaje" = play ]
then
  mplayer ./programa_sintesis/salida.avi;
fi

if [ "$mensaje" = nueva ]
then
        pgrep -P $$ > ./programa_sintesis/tty.log
        kill -9 $(cat ./programa_sintesis/tty.log)
        programa_principal
fi
        done
}

##########################################################################
###
##########################################################################
###
# Sintesis                                                #
##########################################################################
###
busqueda_palabras(){
echo "  " > $TTY
echo "EL TEXTO ESTA SIENDO ANALIZADO..." > $TTY

#buscamos en catalago las palabras introducidas por el usuario

for i in `cat ./programa_sintesis/texto_inicio.txt ` ; do ( grep "_${i}_"
lab_paraules/catalogo | head -1 >> ./programa_sintesis/palabras_seleccionadas.txt) ;
done
```

```
# eliminamos los guiones bajos utilizadas para mejorar la búsqueda

sed -e 's/_/ /g' ./programa_sintesis/palabras_seleccionadas.txt >
./programa_sintesis/palabras_seleccionadas1.txt

mv ./programa_sintesis/palabras_seleccionadas1.txt
./programa_sintesis/palabras_seleccionadas.txt
}
##########################################################################
programa_principal(){
init_programa
start_display
introducir_texto
procesar_tx
}
##########################################################################
###
# PARTE principal                                                      #
##########################################################################
###

programa_principal
```

ANEXO IV: Script ttspalabras

```
#! /usr/bin/perl

#open(F,$ARGV[0]);

#variable contador que utilizaremos posteriormente para dar nombre a nuestros archivos

#creamos silencio antes de empezar el video
$contador = 0;

system ("mencoder ./avi/bg0004.avi -ss 00.00 -endpos 00.250 -ovc copy -oac copy -o
./programa_sintesis/silencio$contador.avi");

system("ch_wave -start 0.00 -end 00.2 ./cmu_es_bg3/wav/bg0004.wav >
./programa_sintesis/corto_audio.wav");

system("mencoder ./programa_sintesis/silencio$contador.avi -o
./programa_sintesis/salida.avi -ovc copy -oac copy -audiofile
./programa_sintesis/corto_audio.wav");

system("rm -rf ./programa_sintesis/silencio$contador.avi");


while ($linea = <STDIN>) {
chomp($linea);

#Leemos cada linea del documento de texto. Ordenamos a perl que haga las divisiones a
partir de los espacios. Y almacenamos las variables que nos interesan
@linea = split(/ /,$linea);
$fichero = $linea[0];
$inicio  = $linea[1];
$final   = $linea[2];
$palabra = $linea[4];

$duracion = $final - $inicio;
$fichero = "./avi/".$fichero.".avi";

# Condicion para cambiar los valores
if ($inicio >= 10.000){
        $inicio = "00:00:".$inicio;}
else {
        $inicio = "00:00:0".$inicio;}



$duracion = "00:00:0".$duracion;
```

```
$segmento = "segmento".$contador.".avi";
$paso = "paso".$contador.".avi";


# PRIMER PASO: Obtencion segmento de video

system ("mencoder $fichero -ss $inicio -endpos $duracion -ovc copy -oac copy -o
./programa_sintesis/$paso");

# SEGUNDO PASO: Obtencion segmento de audio

#Para cada palabra modificamos las instrucciones para Festival
system("sed -e s/PALABRA/$palabra/g fstvl.scm > nou_fstvl.scm");

system("sed -e s/LABS/labs$contador/g nou_fstvl.scm > nou_fstvl1.scm");

system("mv nou_fstvl1.scm nou_fstvl.scm");

system("rm -rf nou_fstvl1.scm");


#llamamos al programa festival y le pasamos las instrucciones creadas anteriormente
system("festival -b nou_fstvl.scm");

system("cd ..");

#Quitamos silencios de los archivos creados por Festival
system("cat ./programa_sintesis/labs$contador.lab  |  ./quita_silencios");


# TERCER PASO: Union video+audio

#Antes de unir video y audio la duracion de ambos archivos debe ser la misma

system("ch_wave -info ./programa_sintesis/corto_audio.wav | grep Duration >>
./programa_sintesis/audio$contador.txt");

system("echo corto_audio $duracion >> ./programa_sintesis/audio$contador.txt ");

system("sed -e 's/00:00:0//g' ./programa_sintesis/audio$contador.txt >
./programa_sintesis/proceso.txt");

system("sed -e 's/Duration: //g' ./programa_sintesis/proceso.txt >
./programa_sintesis/audio$contador.txt");

system("rm -rf ./programa_sintesis/proceso.txt");

#cambiamos duracion
system("cat ./programa_sintesis/audio$contador.txt  |  ./cambia_duracion");
```

```
#Llamamos a praat para que nos cree los archivos
system("praat nova_duracio.praat");


#Unimos archivos de video y audio.

system("mencoder ./programa_sintesis/$paso -o ./programa_sintesis/$segmento -ovc
copy -oac copy -audiofile ./programa_sintesis/mod_corto_audio.wav");

#eliminamos archivos creados temporalmente para que no se vayan acumulando en
nuestra carpeta
system("rm -rf nou_fstvl.scm");

system("rm -rf nova_duracio.praat");

system("rm -rf ./programa_sintesis/audio$contador.txt");

system("rm -rf ./programa_sintesis/$paso");

system("rm -rf ./programa_sintesis/labs$contador.lab");

system("rm -rf ./programa_sintesis/audio.wav");

system("rm -rf ./programa_sintesis/mod_corto_audio.wav");

system("rm -rf ./programa_sintesis/corto_audio.wav");

# CUARTO PASO: Concatenacion segmentos

system ("mencoder ./programa_sintesis/salida.avi ./programa_sintesis/$segmento -ovc
copy -oac copy -o ./programa_sintesis/salida2.avi");

system("mv ./programa_sintesis/salida2.avi ./programa_sintesis/salida.avi");


$contador = $contador + 1;

}

#creamos un silencio al final
$segmento = "segmento".$contador.".avi";

$paso = "paso".$contador.".avi";

system ("mencoder ./avi/bg0004.avi -ss 00.00 -endpos 00.200 -ovc copy -oac copy -o
./programa_sintesis/$paso");

system("ch_wave -start 0.00 -end 00.2 ./cmu_es_bg3/wav/bg0004.wav >
./programa_sintesis/corto_audio.wav");
```

```
system("mencoder ./programa_sintesis/$paso -o ./programa_sintesis/$segmento -ovc
copy -oac copy -audiofile ./programa_sintesis/corto_audio.wav");

system ("mencoder ./programa_sintesis/salida.avi ./programa_sintesis/$segmento -ovc
copy -oac copy -o ./programa_sintesis/salida2.avi");

system("mv ./programa_sintesis/salida2.avi ./programa_sintesis/salida.avi");

system("rm -rf ./programa_sintesis/$paso");

system("rm -rf ./programa_sintesis/corto_audio.wav");
```

ANEXO V: Script cambia_duracion

```
#! /usr/bin/perl

$linea = <STDIN>;
chomp($linea);


#obtenemos el valor de la duracion del audio
$dur_audio = $linea;

while ($linea = <STDIN>) {
chomp($linea);
@linea = split(/ /,$linea);

#obtenemos el valor de la duracion del video y el nombre del archivo a analizar
$Nom = $linea [0];
$dur_video = $linea [1];

#Obtenemos el factor de escalabilidad
$factor = $dur_video / $dur_audio;

#Modificamos el archivo de instrucciones de praat para que funcione correctamente para
cada archivo de audio

system("sed -e s/NOM/$Nom/g instrucciones.praat > nova_duracio.praat");
system("sed -e s/FACTOR/$factor/g nova_duracio.praat > nova_duracio1.praat");
system("sed -e s/DURACION/$dur_audio/g nova_duracio1.praat >
nova_duracio.praat");
system("rm -rf nova_duracio1.praat");
}
```

ANEXO VI: Script quita_silencios

```
#! /usr/bin/perl

# la primera linia es "nom_arxiu"
$linea = <STDIN>;
$nom = $linea;

$ultimo = 0;
$fin = 0;

$linea = <STDIN>;
chomp($linea);
@linea = split(/ /,$linea);
$inicio = $linea[0];

while ($linea = <STDIN>) {
$fin = $ultimo;
chomp($linea);
@linea = split(/ /,$linea);
$ultimo = $linea[0];

}
```

ANEXO VII: Script fstvl.scm

```
(cd "cmu_es_bg3")
(load "festvox/cmu_es_bg3_clunits.scm")
(voice_cmu_es_bg3_clunits)
(set! utt (Utterance Text "PALABRA"))
(utt.synth utt)
(utt.save.wave utt "../programa_sintesis/audio.wav")
(utt.save.segs utt "../programa_sintesis/LABS.lab")
```

ANEXO VIII: Script instrucciones.praat

```
Read from file... ./programa_sintesis/NOM.wav
To Manipulation... 0.01 75 600
Create DurationTier... shorten 0 DURACION
Add point... 0.01 FACTOR
select Manipulation NOM
plus DurationTier shorten
Replace duration tier
select Manipulation NOM
Get resynthesis (overlap-add)
Write to WAV file... ./programa_sintesis/mod_NOM.wav
```