Tesis Fin de Máster

Máster en Tecnologías de la Información y Comunicaciones en Redes Móviles

Dispositivo de interfaz de usuario orientado a enseñanza del habla por ordenador: Caracterización acústica y diseño e implementación de aplicación interactiva

ESCUELA DE INGENIERÍA Y ARQUITECTURA

UNIVERSIDAD DE ZARAGOZA

FEBRERO 2013

AUTOR:

ÁLVARO ARRÚE LOBERA

DIRECTOR:

DR. ANTONIO MIGUEL ARTIAGA

RESUMEN

El aprendizaje del habla es un proceso considerado natural y sencillo en la especie humana. Sin embargo, es fruto de complejos procesos mentales y de maduración de vital importancia al tratarse de una de las primeras destrezas sociales adquiridas, base de futuros desarrollos cognitivos, intelectuales y emocionales. Problemas al aprender a hablar pueden suponer futuros retrasos en la educación, o dificultades en la integración social de los niños, especialmente en aquellos casos en los que se añade algún grado de discapacidad mental o física.

Esta Tesis de Fin de Máster se centra en la caracterización de un dispositivo de interfaz de usuario para su utilización como herramienta orientada al reconocimiento automático del habla en entornos pedagógicos a través de aplicaciones informáticas.

Para ello se han diseñado dos soluciones de software. La primera de ellas se centra en la creación de una plataforma base sobre la que desarrollar aplicaciones que hagan uso de toda la potencialidad de un dispositivo sensor orientado a la creación de interfaces naturales de usuario. Esta solución, además, ha servido de base para el desarrollo de herramientas y aplicaciones usadas a lo largo del mismo.

La segunda es un reconocedor de palabras aisladas con el que evaluar las capacidades de dicho dispositivo en cuanto a sus capacidades de obtención de señales de audio y el procesado interno que sobre éstas realiza y cómo afectan a la acción de dicho reconocedor. Para poder llevar a cabo esta caracterización se ha creado una base de datos de señales de voz con diferentes locutores, posiciones y opciones de procesado de audio que han servido de base para la obtención de resultados cuantificables.

A mi familia, a mis amigos, a Anat, por acompañarme durante este trayecto, en un año tan lleno de cambios y oportunidades, en el que sobre todo, he aprendido.

Contenido

Índice de figuras

Índice de tablas

1. INTRODUCCIÓN

1.1. Antecedentes y motivación

El aprendizaje del habla es un proceso considerado natural y sencillo en la especie humana. Sin embargo, es fruto de complejos procesos mentales y de maduración de vital importancia al tratarse de una de las primeras destrezas sociales adquiridas, base de futuros desarrollos cognitivos, intelectuales y emocionales. Problemas al aprender a hablar pueden suponer futuros retrasos en la educación, o dificultades en la integración social de los niños, especialmente en aquellos casos en los que se añade algún grado de discapacidad mental o física.

Estas circunstancias revelan la importancia de las técnicas de enseñanza del habla y la necesidad de profundizar en la investigación y desarrollo de nuevas tecnologías aplicadas a este fin. De esta forma, el Grupo de Tecnologías de las Comunicaciones (GTC) de la Universidad de Zaragoza, a través del laboratorio ViVoLab, lleva desarrollando a lo largo de los años una importante labor en este campo, diseñando diferentes herramientas pedagógicas y logopédicas de enseñanza infantil asistida por ordenador.

A su vez, las recientes apariciones de sistemas NUI (Natural User Interface) se presentan como dispositivos muy interesantes para el desarrollo de nuevos métodos pedagógicos con una fuerte componente de interactividad (unida a diferentes elementos multimedia).

Tradicionalmente, el desarrollo de NUI se debía realizar con soluciones propietarias con poca capacidad de modificación o adaptación a nuevas necesidades. Otra solución era la creación de una plataforma propia mediante la integración de hardware y programando un software específico con el consiguiente coste material y humano necesario para poder desarrollarlo y validarlo. En trabajos previos del GTC [1] se ponen de relieve el alto coste humano y material en la adquisición de bases de datos de muestras de voz y las dificultades encontradas al utilizar HW (Hardware) muy específico en su implementación..

Microsoft Kinect supone una revolución en este campo al introducir un dispositivo hardware de altas prestaciones audiovisuales a un precio muy competitivo y unos entornos de desarrollo versátiles y en continuo crecimiento que permiten la aparición de una nueva comunidad de desarrolladores técnico-científicos capaces de llevar este dispositivo al laboratorio pese a estar diseñado originalmente para el ocio. Esta comunidad se sustenta en los principios de código fuente abierto lo que permite avanzar de forma más rápida y eficiente en la consecución de objetivos.

1.2. Objetivos

En esta Tesis Fin de Máster (TFM) se pretende caracterizar el hardware de sonido de Mirosoft Kinect, respecto a su utilización junto a un reconocedor de voz. Microsoft Kinect incorpora HW de sonido capaz de implementar un sistema de supresión acústica de eco (AEC, Acoustic Echo Cancellation), un control automático de ganancia (AGC, Automatic Gain Control) y un sistema de generación y translación de haz de sonido entrante.

A través del SDK (Software Development Kit) proporcionado por Microsoft, se procederá al estudio y caracterización de estos procesos, y cómo afectan a un reconocedor de voz de

palabras aisladas. Este reconocedor será implementado en Matlab aprovechando la flexibilidad de este software para analizar y procesar las diferentes señales de voz capturadas.

Paralelamente, utilizando este aprendizaje y caracterización del dispositivo Kinect, se diseñará una plataforma de desarrollo de aplicaciones audiovisuales, con reconocimiento de voz y gestos, que facilite la creación de NUI. Esta plataforma puede ser de gran utilidad para la introducción de nuevas técnicas pedagógicas para niños y adolescentes en general, pero también para niños con necesidades especiales, al proporcionar una interactividad mucho más natural y sencilla que las aplicaciones tradicionales basadas en sistemas operativos comunes y que necesitan un aprendizaje previo. En el diseño de esta plataforma se seguirá una aproximación que prime la escalabilidad del sistema, garantizando la modularidad entre las diferentes capacidades del dispositivo Kinect y de su SDK, y maximizando la compatibilidad con pasados y futuros desarrollos.

1.3 Organización de la memoria

En el Capítulo 1 se ha desarrollado una breve introducción a la TFM y los objetivos perseguidos.

En el Capítulo 2, se abordan los principios teóricos de los reconocedores automáticos del habla. También se repasan las características técnicas del dispositivo Kinect.

En el Capítulo 3 se detallan los diferentes desarrollos realizados en el proyecto tanto en la plataforma de software como en el reconocedor de palabras aisladas. En este capítulo también se detallan los diferentes resultados obtenidos a nivel de tasa de aciertos de los diferentes modelos entrenados y se analizan desde un punto de vista cuantitativo, cualitativo y dimensional dichos resultados.

En el Capítulo 4 se exponen las conclusiones obtenidas sobre el resultado del proyecto, así como las posibles líneas futuras de investigación y desarrollo que podrían seguirse.

En el Capítulo 5 se puede encontrar el Diagrama de Gantt por tareas de la TFM.

Con respecto a los anexos:

En el ANEXO A se presenta una descripción más detallada y extensa de los reconocedores automáticos del habla y la teoría de fondo que los sustenta. En el ANEXO B se procede a detallar la historia, modos de trabajo y especificaciones técnicas del dispositivo Microsoft Kinect. En el ANEXO C se pueden encontrar las diferentes frases y palabras utilizadas para el entrenamiento y test en el reconocedor de palabras aisladas. En el ANEXO D se explica de forma detallada la plataforma de desarrollo de aplicaciones para Microsoft Kinect. En el ANEXO E se pueden encontrar los resultados del reconocedor de palabras aisladas. El ANEXO F describe los diferentes elementos de software utilizados en el desarrollo de esta TFM. En el ANEXO G se encuentran las diferentes funciones y scripts desarrollados en Matlab para el reconocimiento de palabras y el análisis de resultados.

2. ESTADO DEL ARTE

2.1. Reconocimiento automático del habla

2.1.1. Introducción

El RAH (Reconocimiento Automático del Habla) es un proceso de clasificación de secuencias de patrones, capaz de procesar la señal de voz emitida por el ser humano y reconocer la información contenida en ella, convirtiéndola en texto o emitiendo órdenes que actúen en consecuencia. Esta tecnología permite la comunicación oral hombre-máquina, abriendo un amplio abanico de aplicaciones prácticas en diferentes sectores:

- Sistemas de dictado.
- Sistemas de control oral para personas con discapacidad.
- Sistemas de asistencia e información para personas con problemas de audición (p.e. subtitulación automática de contenidos audiovisuales).
- Aplicaciones telefónicas (p.e. servicios de asistencia técnica telefónica).
- Aplicaciones biométricas (p.e. identificación del locutor).
- Autoaprendizaje de idiomas
- Interfaces multimodales de interacción hombre-máquina. Equipamiento electrónico o informático, como ordenadores, smartdevices (smartphones o tabletas), módulos hardware de control (p.e. dashboard de un coche) y entretenimiento (p.e. Microsoft Kinect) que incluyen el habla como un elemento más de interacción con el usuario y de forma bidireccional.

La investigación y desarrollo de RAH tiene como objetivo final el reconocedor ideal, es decir, aquel con una tasa de error nula; sin embargo, la complejidad del sistema obliga al desarrollo e investigación de diversas técnicas para cada uno de los diferentes elementos de la arquitectura del reconocedor (parametrización, entrenamiento, modelado del lenguaje, adaptación del locutor…).

2.1.1. Estado del arte

Diseñar una máquina capaz de imitar al ser humano y su comportamiento, y particularmente su capacidad para hablar de forma natural, ha sido objeto de estudio por parte de ingenieros y científicos a lo largo de los siglos. A lo largo del siglo XX, y especialmente desde su segunda mitad, se han realizado numerosos avances desde el reconocimiento simple de palabras aisladas hasta reconocedores de habla continua con grandes vocabularios y capacidades multilocutor. En el ANEXO A se puede encontrar una descripción más detallada de esta evolución y de los principales hitos en su camino.

2.1.2. Arquitectura

Funcionamiento

De acuerdo a la aproximación a partir del teorema de Bayes, podemos describir un reconocedor automático del habla como un sistema en el que se debe disponer de unos patrones asociados a las partes del habla que se van a reconocer (fonemas, palabras, frase…), de manera que procesando una secuencia de vectores de características u observaciones $\boldsymbol{O}$,

tengamos como respuesta la secuencia de patrones que representan a este vector con la probabilidad mayor. De la forma que indica la fórmula fundamental del reconocimiento automático del habla:

$$\hat{W} = \arg\max_{W} P(O \mid W) \cdot P(W) \qquad \text{Ecuación 2-1}$$

- $P(W)$ es la probabilidad de la secuencia de palabras W o modelo de lenguaje
- $P(O \mid W)$ es la probabilidad de observar la secuencia O cuando se pronuncia la secuencia W

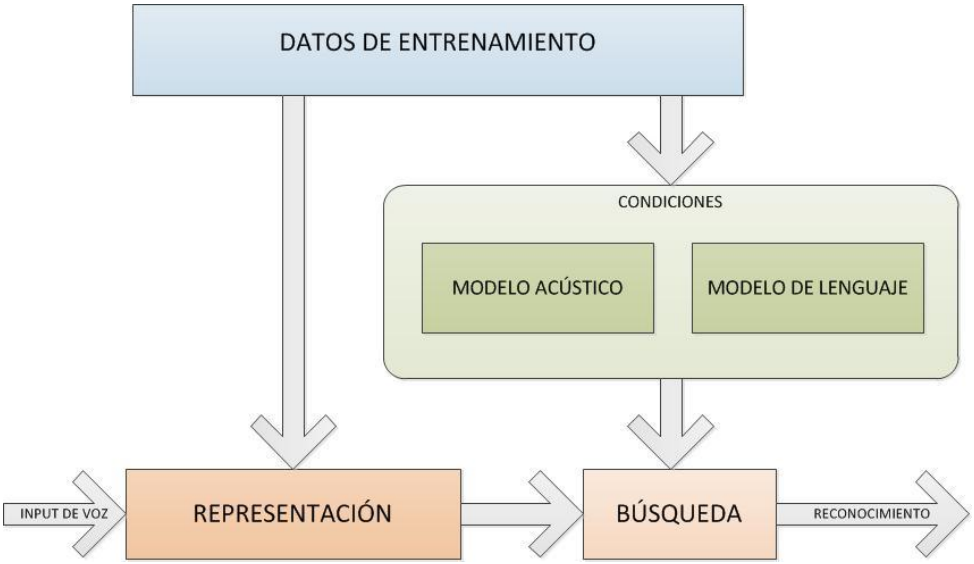

Figura 1 Esquema de la arquitectura de un RAH genérico

Base de datos

En la base de datos podemos encontrar las palabras que utilizaremos para entrenar al sistema. Estas palabras deben ser elegidas con cuidado atendiendo a factores como la inclusión de todos los fonemas del lenguaje a reconocer, y además deben estar balanceadas, es decir, que estos fonemas tengan una tasa de aparición similar a su presencia en dicho lenguaje. Estas palabras deben ser suficientes, ya que su número tiene incidencia directa en la calidad del reconocedor.

Extracción de características

El módulo de extracción de características, común a la fase de entrenamiento y de reconocimiento, tiene como objetivo procesar las señales de audio tomadas del locutor con el objetivo de resaltar la información relevante para el RAH y eliminar aquella que no tenga importancia fonética. Tradicionalmente dos son las técnicas utilizadas en la extracción de características, Linear Prediction Coefficients (LPC)[2] y Mel-Frequency Cepstral Coefficients (MFCC)[3]. Esta última técnica será la utilizada en esta TFM.

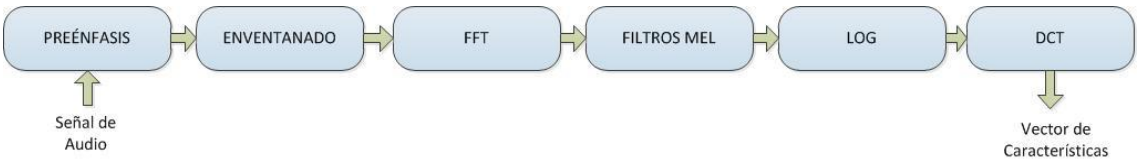

Figura 2 Cadena de tratamiento de señales de voz para extraer su vector de características

Modelo acústico

Existen diferentes métodos para obtener el modelado acústico, sin embargo, y como en el caso de esta TFM, los más utilizados son los relacionados con los modelos ocultos de Markov o HMM (Hidden Markov Models)[4]. Los modelos ocultos de Markov son una máquina de estados finita (Figura 3), cuyas observaciones son una función probabilística del estado, es decir, un HMM es un proceso doblemente estocástico formado por:

1) Un proceso estocástico oculto, no observable directamente, que corresponde a las transiciones entre los diferentes estados
2) Un proceso estocástico observable cuya salida es una secuencia de vectores de voz.

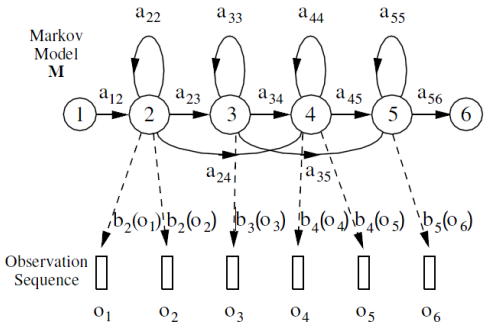

Figura 3: Estructura genérica de un HMM

En el ANEXO A se puede encontrar una definición más detallada de los HMM, su definición y características.

Adaptación supervisada

La variabilidad en el habla es uno de los factores más complejos que se encuentran en los RAH. Esta variabilidad se agrupa en dos grandes grupos (ver Tabla 5 en el ANEXO A), factores intrínsecos (variabilidad del contexto, del locutor, estilo…) y factores extrínsecos, como las condiciones ambientales en las que se produce la adquisición de las señales de voz. Uno de los objetivos de esta TFM es el estudio y la compensación de los efectos del canal haciendo uso del procesado de audio de Kinect.

Las técnicas de adaptación supervisada se encargan de adaptar un modelo genérico a un modelo particular al locutor, siendo dos de los algoritmos más utilizados el MLLR(Maximum Likelihood Linear Regression)[5] y el MAP (Maximum a Posteriori)[6]. En esta TFM utilizaremos la técnica MAP de adaptación supervisada. Esta adaptación utiliza un modelo inicial genérico de partida, o modelo independiente del locutor, cuya información es por tanto conocida a priori. Con la información nueva observada en el locutor particular, combinada con la

conocida, y ponderadas por un factor de adaptación fijado de antemano, se estima la nueva información.

Reconocimiento

En este módulo del reconocedor es donde finalmente se encuentra la secuencia de fonemas que maximiza la ecuación fundamental del reconocedor (Ecuación 2-1). La solución directa sería comparar cada posible secuencia con la estimación aproximada y la elección de la que diera un resultado mayor. Esta solución, aunque válida para vocabularios pequeños, sería computacionalmente demasiado exigente para un sistema de vocabulario estándar. En el caso de esta TFM, cuyo objetivo es la caracterización de Microsoft Kinect en el reconocimiento del habla, la resolución de la palabra estimada se calculará de forma directa. Sin embargo, es importante destacar las alternativas a utilizar en un escenario de habla continua y que se pueden encontrar en el ANEXO A.

2.2. Hardware/Software: Microsoft Kinect

2.2.2. Microsoft Kinect

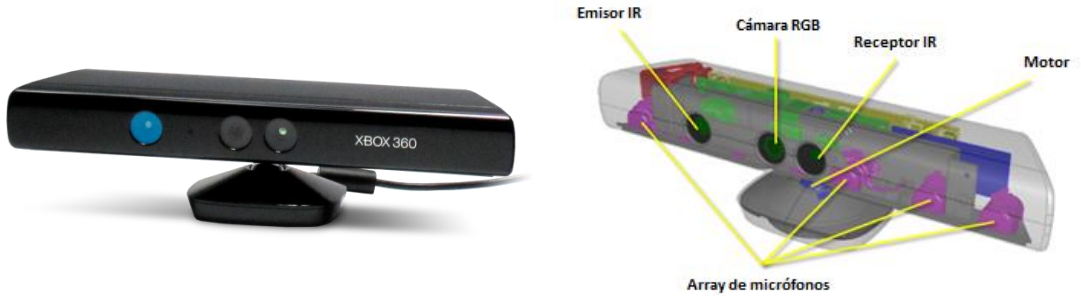

Figura 4 Kinect y sus componentes

El dispositivo Microsoft Kinect (Figura 4) aúna en un solo dispositivo diferentes métodos de entrada de información del usuario, como se puede apreciar en la Tabla 1, orientados a la implementación de interfaces naturales de usuario (NUI), tipo especial de interfaz multimodal que busca una interacción instintiva con el usuario [7].

Tabla 1 Características técnicas del sensor Microsoft Kinect

	Hardware
Video	• Cámara con señal de salida RGB o YUV de 640x480@8bits@30fps • Señal de profundidad de 640*480@11bits@30fps obtenida a partir de un emisor de matriz de infrarrojos y un receptor de infrarrojos CMOS, obteniendo la profundidad por cada pixel.
Procesado de Video	• Sensor de profundidad basado en la detección de la divergencia de la emisión de una matriz estructurada de infrarrojos. • Detección de 20 articulaciones por jugador activo • Detección de hasta 6 personas con 2 jugadores activos • En caso de articulaciones ocultas, sistema de inferencia y estimación de las mismas con parámetro de fiabilidad. • Estimación automática del plano de suelo
Audio	• Array de cuatro micrófonos • Frecuencia de muestreo de 16KHz y 24 bit de resolución
Procesado de	• Array de 4 micrófonos para beam detection: Resolución de 10° de

audio	aproximadamente -50° a 50° de detección. • Automatic Echo Cancellation • Noise suppresion • Automatic Gain Control
Motor	• Control de inclinación del módulo
Acelerómetro	• Utilizado para estimar inclinación de la cámara
Ángulo de visión	• 43° Vertical, 57° Horizontal
Rango	• Modo lejano: 1.2-3.9m • Modo cercano: 0.4-3m (solo Kinect for Windows)
Resolución	• Resolución espacial x/y: 3mm @2m • Resolución de profundidad z: 1cm @2m

Microsoft Kinect incluye sistemas de captura de audio y video con capacidad de procesado por hardware en la propia consola, siendo transmitido un flujo de audio y video hasta el PC.

2.2.3. Características acústicas de Kinect y procesado de señal de audio

Aunque visualmente es un dispositivo muy potente para su coste de mercado (gracias a la fabricación en escala), desde el punto de vista del sonido, Microsoft Kinect incluye importantes avances en tratamiento de audio [8].

El dispositivo Kinect dispone de cuatro micrófonos supercardiodes en array, 3 de ellos en un extremo de la carcasa y otro en el extremo contrario del frontal del dispositivo.

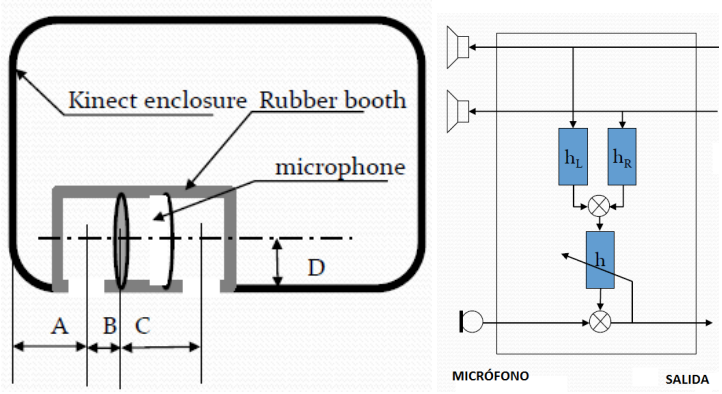

Figura 5 Caja acústica de los micrófonos y esquema AEC de Kinect

Gracias a esta arquitectura acústica, y al procesado digital que realiza a través del sistema de audio de Windows, Kinect es capaz de ofrecer diferentes mejoras en la captura de sonido, a saber:

Acoustic Echo Reduction

Kinect debe enfrentarse al ruido generado por los sonidos reproducidos por la aplicación que se está ejecutando y, para ello, utiliza algoritmos de cancelación activa de ruido y supresión activa de ruido de forma adaptativa.

Noise Suppression y Automatic Gain Control

Además de la AEC, Kinect introduce algoritmos de eliminación de ruido de forma estadística y un sistema variable de ganancia de los micrófonos para adaptar dinámicamente señales de voz de baja presión sonora.

Beamforming

Uno de los puntos más interesantes en el tratamiento de audio de Kinect es la capacidad de formar haces de captura (Beamforming) de sonido aumentando la directividad del array de 4 micrófonos. Para ello, Kinect crea un haz móvil (Beamsteering), que busca el máximo de intensidad de la señal de voz recibida, obteniendo una ganancia de directividad máxima y reduciendo el ruido de interferencia procedente de otras direcciones.

2.2.4 Entornos de desarrollo de Microsoft Kinect

La comunicación del dispositivo Microsoft Kinect se realiza a través de un puerto USB 2.0 y con diferentes librerías incluidas en los distintos kits de desarrollo.

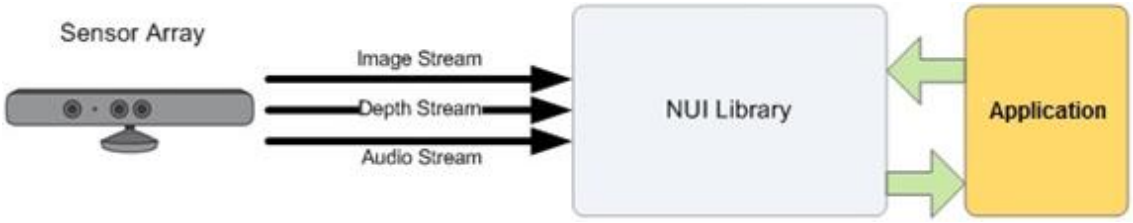

Figura 6 Interacciones Hardware/Software de Kinect[1]

El éxito del lanzamiento, su bajo coste y sus características avanzadas de captura de información, promovieron la aparición de diferentes drivers y SDK desarrollados de forma independiente y ajena a Microsoft. Finalmente, la compañía de Redmond, consciente de la importancia de alimentar a una gran comunidad de desarrolladores de software y servicios, así como del entorno académico y científico, hizo público un SDK[2] propietario pero de libre uso sin ánimo de lucro para el desarrollo de aplicaciones en PC utilizando Kinect.

En la realización de esta TFM se ha utilizado el dispositivo Kinect original para consola XBOX360, por su menor precio, facilidad de adquisición y con el objetivo final de caracterizar su respuesta acústica, usándolo con un reconocedor automático del habla con palabras aisladas con el objetivo de evaluar el efecto del procesado en la cadena de audio a efectos de reconocimiento. En el ANEXO B se puede encontrar más información acerca de las características del dispositivo Kinect.

[1] http://msdn.microsoft.com/en-us/library/jj131023.aspx
[2] http://www.microsoft.com/en-us/kinectforwindows/

3. DESARROLLO

3.1 Framework de software

Plataforma de desarrollo

Esta plataforma ha sido implementada utilizando Visual Studio 2010 con los SDK correspondientes a DirectX y Kinect for Windows actualizados a su última versión y utilizando C++ como lenguaje de programación. El uso de C++ como lenguaje de programación [9] frente a C# se decidió como una de los requisitos de la TFM con el fin de garantizar máxima compatibilidad con las aplicaciones previamente realizadas por el Departamento de Tecnologías de Voz y facilitar su adaptación al sensor Kinect.

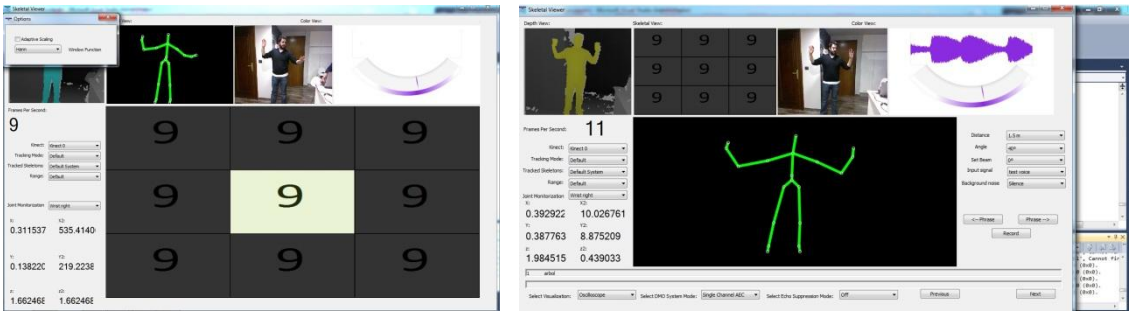

Figura 7 Imágenes de Total Explorer y Voice Sampler en funcionamiento

El software se puede dividir en diferentes módulos de acuerdo a las funcionalidades que en él se pueden encontrar y dependiendo de la naturaleza de la información capturada, así como de la configuración de los dispositivos:

1) Inicialización: Setup de Kinect y de las variables de trabajo, juego, audio…
2) Visual: Centrado en la presentación de los flujos de video y el esqueleto
3) Audio: Centrado en el tratamiento y visualización de las señales de audio capturadas.

Total Explorer

La plataforma se ha diseñado de una forma flexible para poder desarrollar videojuegos de tipo tablero, usados anteriormente por el departamento en actividades de educación especial. Total Explorer es la base para el desarrollo de este tipo de aplicaciones tablero, con la incorporación de las capacidades de detección de gestos y voz de Kinect.

Voice Sampler

Esta aplicación permite seleccionar los modos de funcionamiento de la arquitectura audio de Kinect y facilita la labor de grabación de las muestras de voz para su posterior uso en el reconocedor de palabras aisladas.

En el ANEXO D se puede encontrar más información sobre la implementación y funcionalidades de la plataforma de software.

3.2. Obtención de los datos de voz

3.2.1. Preparación de las medidas

Introducción y objetivos

La obtención de los datos se debe estudiar con detenimiento para poder obtener muestras significativas que permitan, por una parte, obtener frases de entrenamiento para poder adaptar los modelos base a cada locutor y, posteriormente, capturar palabras de test para cuantificar la efectividad del reconocedor.

En el caso particular de esta TFM se han realizado estas medidas en un entorno similar al del uso habitual de un dispositivo Kinect en un entorno doméstico y destinado a un uso lúdico. Para ello, se han presentado los elementos de la cadena de grabación en una sala de estar con una distribución asimétrica y tomando medidas en las áreas de trabajo visual del sensor.

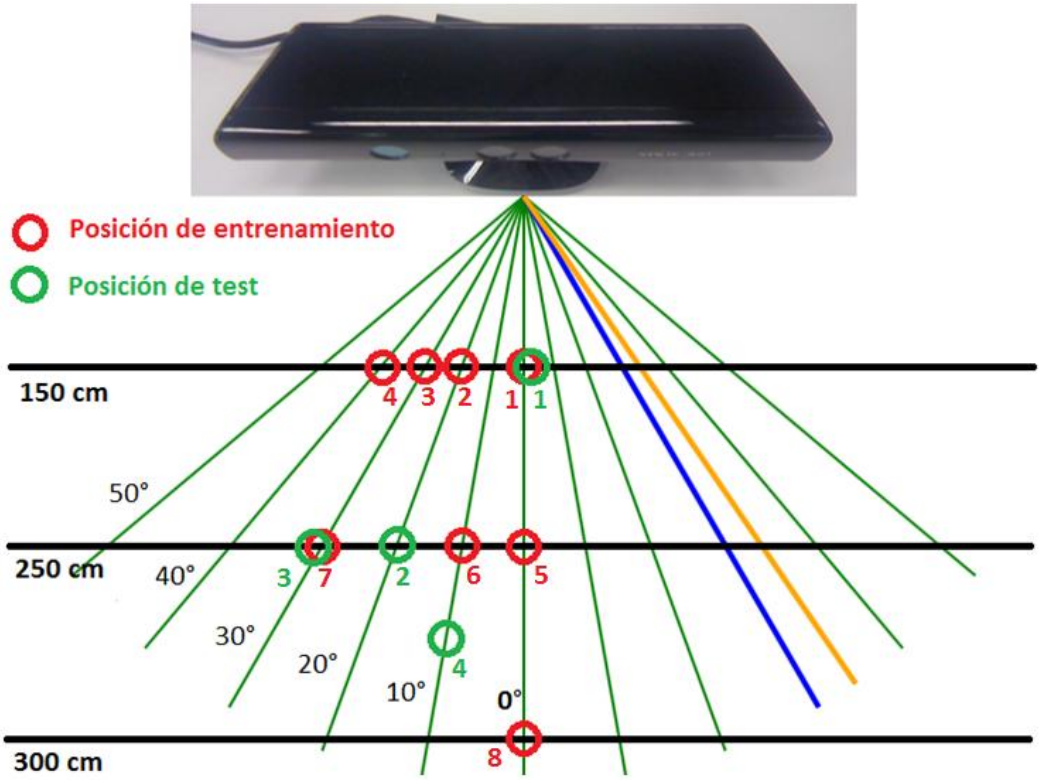

Figura 8 Posiciones de grabación de frases de entrenamiento y test

Disposición de los dispositivos y de los componentes

El dispositivo Microsoft Kinect se situó en un punto fijo para el resto de medidas, a 1.9 m del suelo y a 10 cm de la pared, con el eje visual inclinado para obtener un rayo directo a la boca de un locutor de altura 1.70m, de pie y a 2m del eje vertical del dispositivo, emulando las circunstancias habituales de uso. Ningún elemento extraño se sitúa entre el dispositivo y el locutor, manteniendo en todo momento la condición de LoS (Line of Sight) tanto para la parte visual como la acústica.

Asunción de simetría respecto al eje

De acuerdo a las especificaciones técnicas de Microsoft, el dispositivo Kinect tiene simetría funcional respecto al eje perpendicular al frontal de la carcasa, o eje de captura de

información. Estas consideraciones se han tomado como medida de precaución, ya que en todo momento las capturas de voz se han realizado manteniendo la condición de LoS con el dispositivo.

Posiciones de entrenamiento y test

En la Figura 8 se pueden observar las diferentes posiciones de referencia donde se tomaron tanto las muestras de entrenamiento como de test para cada uno de los locutores y cada una de las diferentes configuraciones de tratamiento de señal del dispositivo Kinect.

3.2.2. Locutores y palabras de entrenamiento y test

Locutores

Para la caracterización del Reconocedor Automático del Habla (RAH) se capturaron muestras de voz de entrenamiento y test de dos locutores distintos: un hombre y una mujer jóvenes con la voz ya madura, sin entrenamiento previo en la locución y sin enfermedades o trastornos aparentes en el habla.

En esta TFM, al tratarse de un estudio preliminar, se han elegido dos locutores cercanos a la franja demográfica de uso de Microsoft Kinect, sin embargo sería necesario profundizar en otros tipos de usuario. En cuanto a la aplicación práctica de la Kinect para personas con necesidades especiales, al ser un sector muy específico, no entraría en las categorías asociadas al estudio estadístico de la población, siendo ideal un entrenamiento específico de los modelos de acuerdo a las necesidades especiales de sus usuarios.

Frases de entrenamiento

Para esta TFM se obtuvo un corpus propio de acuerdo a las diferentes características de Kinect en captura y tratamiento de audio. Para el entrenamiento de los modelos se tomaron muestras de cada locutor leyendo una batería de frases de entrenamiento con los principales fonemas en español y con una mayor presencia de aquellos más comunes en lengua española [10]. Se puede consultar la lista de frases en el ANEXO C. Las frases de entrenamiento se grabaron en cada una de las posiciones fijas descritas en la Figura 8.

Tabla 2. Número de muestras de voz tomadas en la base de datos

	Locutores	Posiciones	Modos de Kinect	Nº de Frases	Total de frases
Entrenamiento	2	8	2	50	1.600
Test	2	4	2	54	864
				TOTAL	**2.464**

Palabras de test

Para las palabras de test utilizadas para determinar las tasas de éxito del RAH se grabaron 54 palabras para cada locutor, con diferentes características de grabación en la Kinect y en las posiciones marcadas en la Figura 8. Cabe destacar que las grabaciones de test se realizaron en puntos de especial significación entre los utilizados para el entrenamiento y también en puntos que no estaban referenciados originalmente en el entrenamiento. Se puede encontrar el listado completo de palabras de test en el ANEXO C.

3.2.3. Configuración de Microsoft Kinect

A través del SDK para Kinect oficial de Microsoft es posible configurar diferentes parámetros en la captura de audio de acuerdo a las opciones mostradas en la Tabla 3.

En verde se resaltan las opciones que se utilizaron para capturar las muestras de entrenamiento y test con el objetivo de caracterizar el dispositivo Kinect, sin ninguna mejora de sonido a partir de tratamiento y con todos los sistemas de corrección disponibles. Cada una de estas opciones de captura de sonido se realizó para cada una de las posiciones de test y entrenamiento y para cada uno de los locutores. Para poder organizar y ejecutar la captura de muestras de una forma rápida, eficaz y ordenada se desarrolló una aplicación basada en Kinect SDK, Voice Sampler (ver ANEXO D).

Tabla 3 Modos de Kinect capturados para entrenamiento y test

Configuración array	Acoustic Echo Suppression	
Single Channel	Off	On
Single Channel con AEC	Off	On
Optibeam Array	Off	On
Optibeam Array con AEC	Off	On

3.3. Procesado de los datos

Una vez obtenidos las capturas de datos de entrenamiento y test para cada locutor y con los diferentes modos de audio de Kinect, es necesario tratar los datos para obtener resultados acerca de la bondad del dispositivo Kinect de Microsoft a la hora de utilizarlo como elemento de captura dentro de un reconocedor automático del habla.

Para ello, es necesario aplicar diferentes cálculos y modificaciones a las muestras PCM a fin de poder cuantificar matemáticamente la tasa de aciertos.

En esta TFM se ha implementado un reconocedor de palabras aisladas en Matlab haciendo uso De modelos HMM entrenados mediante la herramienta de software libre HTK (Hidden Markov Model Toolkit).

3.3.1 Estructuras y archivos

*Ficheros *.list → File path list*

El script List_test_scripts se encarga de buscar en el árbol de directorios indicado todas las carpetas con archivos de test o train que se encuentren, generando un archivo de texto con extensión .list, en el cual se pueden encontrar los paths a cada archivo .mfc con los vectores de características de cada audio. El script revisa directorio a directorio buscando las carpetas test y analizando, carpeta a carpeta, el nombre del locutor, distancia, grados y procesado de los archivos. Al utilizar una nomenclatura estructurada a la hora de capturar las muestras (ver ANEXO D.6) es sencillo automatizar esta operación.

*Ficheros *.txt → Listas de palabras*

Existen diferentes archivos .txt con la lista de palabras de test grabadas y el diccionario de posibles palabras contra las que testear.

Estructuras

Una de las condiciones seguidas en la elaboración del procesado en Matlab de las señales de voz capturadas es el mantenimiento de una política de alta automatización de resultados, dada la gran cantidad de archivos, modelos, frases y tests utilizados. Para ello, se han creado dos estructuras de datos tipo struct en Matlab que facilitan la obtención de variables y datos de una forma ordenada y recurrente: st_tests y st_models.

- Structs: st_tests
  - speaker_name: nombre del locutor
  - distance: distancia en cm al micrófono
  - degrees: grados respecto al eje de grabación de la kinect
  - processing: procesado o no de la señal de test
  - m_LH: [54x57 double]: matriz con la log verosimilitud de los pares palabra dicha/palabra del diccionario
  - recogn_word: índice de la palabra reconocida (máxima log verosimilitud)
  - recogn_LH: log-verosimilitud de la palabra reconocida
  - file: nombre del File Path List utilizado para el test

- Structs: st_models
  - speaker_name: nombre del locutor
  - distance: distancia en cm al micrófono
  - degrees: grados respecto al eje de grabación de la kinect
  - processing: procesado o no de las señales de entrenamiento del modelo
  - model_db: modelo entrenado correspondiente tests: struct st_tests con los tests para ese modelo

3.3.2. Procesado

Extracción de los vectores de características

Haciendo uso de HTK se han obtenido los vectores de características de las diferentes muestras de voz tanto para entrenamiento como para test. Estos vectores de características mantienen la misma nomenclatura que las señales de audio capturadas pero tienen extensión .mfc. Cada uno de los vectores de características de cada frame de sonido procesado está compuesto por 12 MFCC más la información de energía (por triplicado), que junto a las derivadas y aceleraciones hacen un total de 39 elementos por frame.

Baseline: Modelo simple

Previamente a los procesos de adaptación y test, se ha generado un modelo simple que servirá de base para los diferentes pasos. Este modelo simple, baseline para los diferentes locutores, posiciones y características de grabación está formado por fonemas, cada uno de los cuales tiene 3 estados. Cada uno de estos estados a su vez está modelado por 8 gaussianas.

Dados los objetivos de la presente TFM, este modelo, aunque demasiado simple para RAHs que deban trabajar en entornos reales de funcionamiento, permite una mayor flexibilidad y

reducción de la complejidad a la hora de afrontar al análisis de las características del dispositivo Kinect.

Funciones en Matlab

Para la implementación del reconocedor de palabras aisladas en Matlab se han programado diferentes funciones y scripts en lenguaje M. Asimismo, se ha usado la librería vivo_reco[3] de HTK para Matlab con el fin de testear los diferentes modelos con las palabras de test correspondientes, para cada posición, locutor y modo de funcionamiento del audio de Kinect. Se pueden dividir en tres grandes grupos: entrenamiento, reconocimiento y obtención de resultados. En el ANEXO G se puede encontrar un listado de las diferentes funciones y scripts implementados.

3.4. Obtención de resultados

3.4.1. Análisis cuantitativo

Para realizar el análisis cuantitativo del reconocedor de palabras aisladas con los modelos entrenados y las posiciones de test, se deben tener en cuenta cómo se han realizado las medidas y en qué condiciones.

Para cada locutor se han diferenciado dos tipos de procesado:

- **Raw**: Modo de captura del micrófono de Kinect sin procesado de señal ni Beamforming.
- **Proc**: Modo de captura del micrófono de Kinect con procesado de señal y Beamforming.

A su vez, tenemos los datos para los dos locutores descritos en la toma de medidas:

- **Álvaro**: Hombre joven con la voz ya madura, sin entrenamiento previo en la locución y sin enfermedades o trastornos aparentes en el habla.
- **Anat**: Mujer joven con la voz ya madura, sin entrenamiento previo en la locución y sin enfermedades o trastornos aparentes en el habla.

Las posiciones de cada modelo son los indicados en la Figura 8: 8 posiciones de grabación y 4 de test.

Por otro lado, se han incluido los datos referentes al baseline de las medidas. Este baseline y sus tasas de aciertos se han calculado aplicando el reconocedor de palabras aisladas al modelo simple, sin entrenar, las palabras grabadas para cada posición de test. La obtención de los resultados para el baseline ayuda a identificar el funcionamiento del RAH con modelos adaptados por entrenamiento MAP.

Debe destacarse que en esta TFM se han utilizado modelos de partida extremadamente sencillos, con pocos estados y gaussianas.

[3] http://www.vivolab.es

Se puede encontrar una tabla resumen de valores (Tabla 16) y su representación gráfica (Figura 33) en el ANEXO E donde se puede observar de una manera global el comportamiento de cada modelo entrenado respecto a las distintas posiciones de test. En esta figura global de los datos se puede observar una clara diferencia entre la tasa de aciertos por locutor, siendo mucho mejor en términos globales, la tasa para el locutor Álvaro que para el locutor Anat. Esto puede deberse a diferencias en la dicción de ambos locutores, siendo la del locutor Anat mucho más rápida que la del locutor Álvaro. De una forma más clara se puede observar también en la Figura 9. Esta diferencia tiene incidencia directa en la tasa de reconocimiento global por locutor.

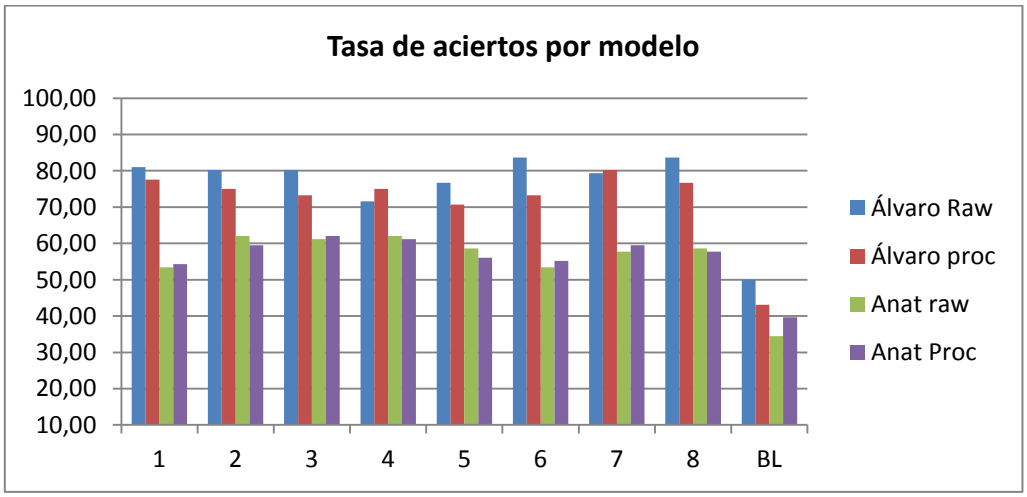

Figura 9 Tasa de aciertos por modelo comparado entre locutores

Respecto a la tasa de aciertos por test, podemos observar cómo en ambos locutores hay una relación directa entre la tasa de aciertos del baseline y la tasa de aciertos en los modelos adaptados: cuanto mejores han sido los resultados en el baseline, mejor se comportan los modelos adaptados de una forma lineal y fácilmente visualizable (Figura 10 y Figura 11).

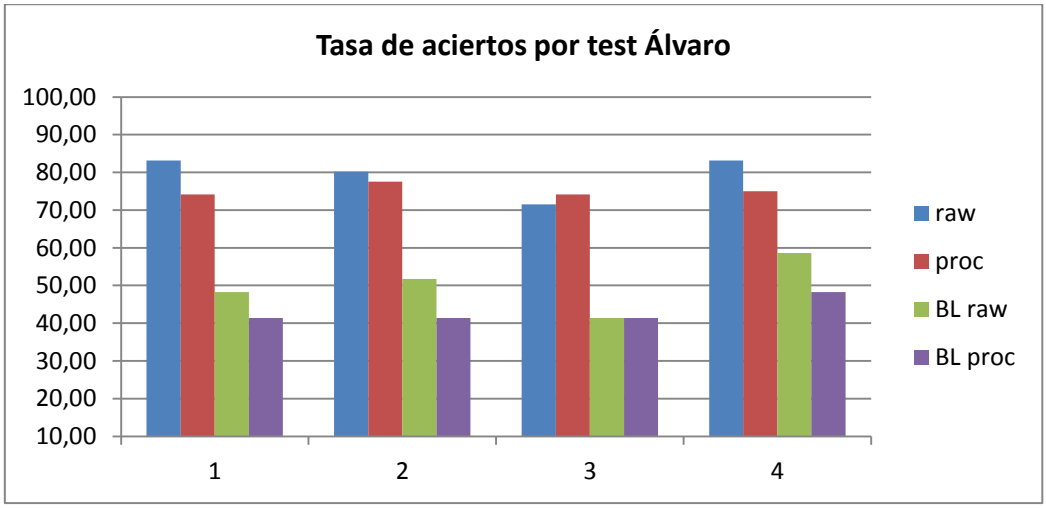

Figura 10 Tasa de aciertos por test para el locutor Álvaro

En ambos casos, además, el reconocimiento es mejor en los modelos y tests donde no se ha realizado procesado de audio, o con una diferencia estadísticamente despreciable, salvo en el

caso del Test 3. Este test se realizó en una posición angular alejada del eje de grabación (30°), y es en este caso particular donde se comportan mucho mejor los modelos con procesado de audio. En este caso domina el efecto del Beamsteering de Kinect permitiendo una tasa de reconocimiento mucho mayor al permitir dirigir el haz del micrófono al punto de interés.

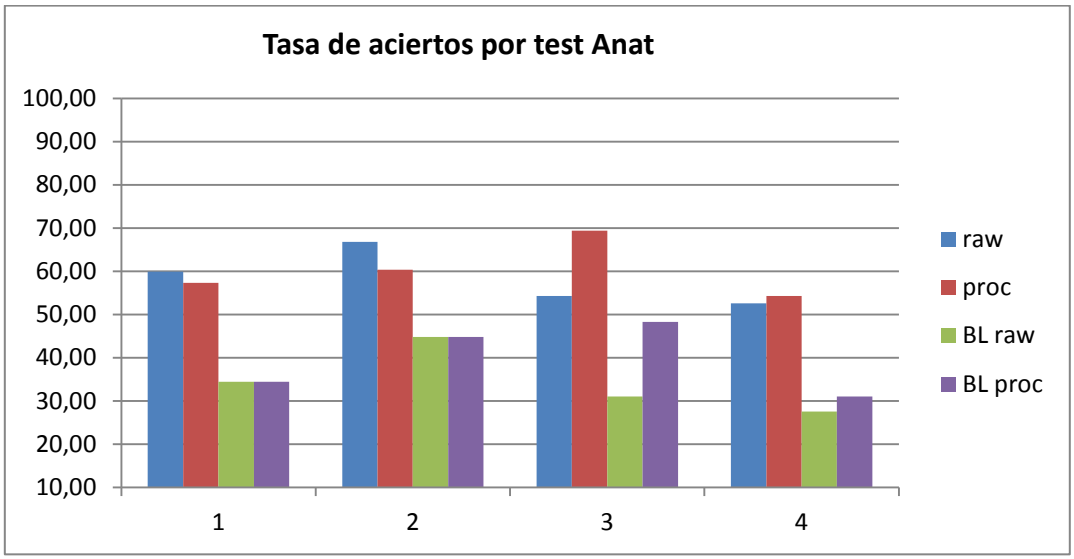

Figura 11 Tasa de aciertos por test para el locutor Anat

Este mismo efecto se puede observar en la posición 7 de la Figura 9. Esta posición es la más alejada en distancia y ángulo al eje de grabación de Kinect y es, de nuevo, en este caso donde se puede apreciar la mejora del Beamsteering sobre la tasa de reconocimiento en ambos locutores.

3.4.2. Análisis dimensional

Como método para intentar identificar parámetros relacionados con la diversidad geográfica de los modelos (en las diversas posiciones utilizadas para entrenamiento y test) se han aplicado los principios de reducción no lineal de dimensionalidad descritos por la Universidad de Stanford [11] y el toolbox de libre acceso. A diferencia de los métodos tradicionales, como PCA o MDS, con este análisis no lineal se utiliza la distancia geodésica frente a la euclídea. En líneas generales, este método calcula las distancias geodésicas entre los puntos (o datos de entrada) más cercanos entre sí, pero a su vez es capaz de mantener intacta la estructura subyacente de los datos, ya que dadas ciertas condiciones, la distancia euclídea puede no ser un camino válido.

En el caso del locutor Álvaro, que como hemos visto en el análisis cuantitativo posee unas mejores tasas de acierto, al analizar la variación del error residual conforme aumentamos la dimensionalidad, podemos observar en la Figura 12 que dicho error converge a dos niveles diferenciados a partir de la dimensión número 4, dependiendo de si el audio ha sido procesado o no. Aunque a priori se podría determinar que las grabaciones procesadas tienen un menor error residual, al haber analizado que la tasa de aciertos es menor en este caso, es posible discernir que el procesado AEC y de Beamforming está eliminando información en las grabaciones que es importante para el reconocedor de palabras aisladas, algo que ha sido corroborado en diversos estudios sobre robustez al ruido mediante sustracción espectral en RAH.

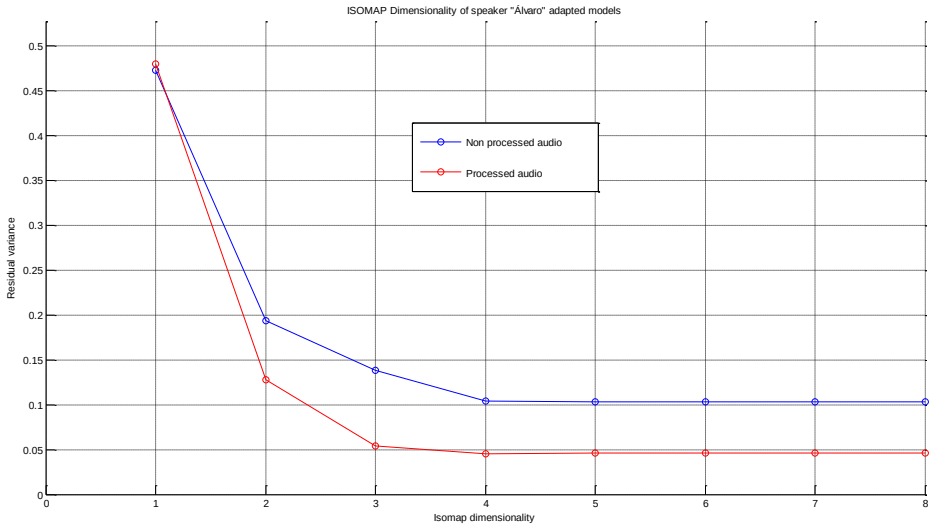

Figura 12 Análisis de la variación residual según la dimensionalidad de los modelos entrenados para el locutor Álvaro

Sin embargo, realizando este mismo análisis para el locutor Anat, cuyos resultados finales en cuanto a tasa de aciertos son mucho peores que para el locutor Álvaro, es difícil determinar esta misma situación. Sí que se mantiene la tendencia a converger al mismo valor y que en el caso de los modelos procesados el error residual también se mantiene por debajo de las grabaciones Raw, destacando que en el nº de dimensión 1 ya existe una importante atenuación del error residual.

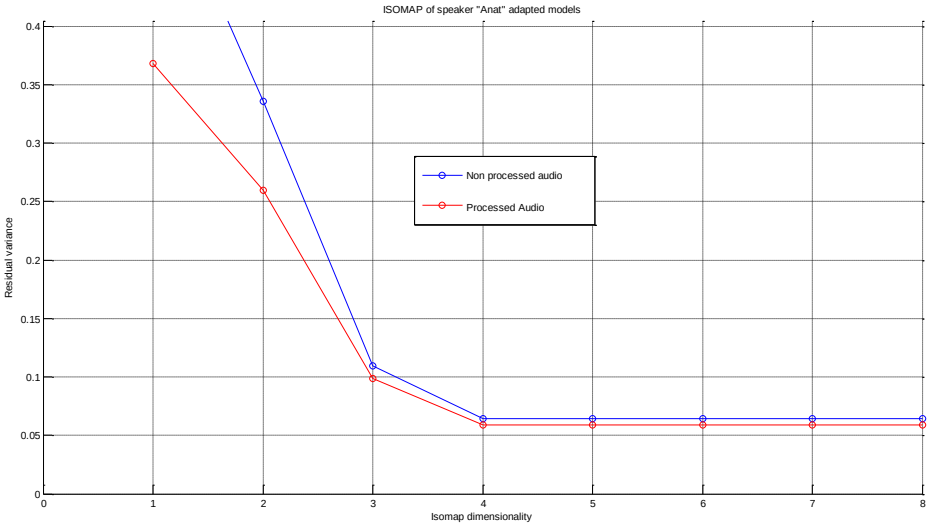

Figura 13 Análisis de la variación residual según la dimensionalidad de los modelos entrenados para el locutor Anat

El análisis no lineal ejecutado en la función isomap.m (ver ANEXO G) realiza también un análisis de "vecindad" de cada punto a estudiar, en este caso los modelos entrenados. En este análisis se define una constante de vecindad k (en nuestro caso k=3) y se determina por cada punto, cuáles son los otros k puntos más cercanos en términos de distancia geodésica.

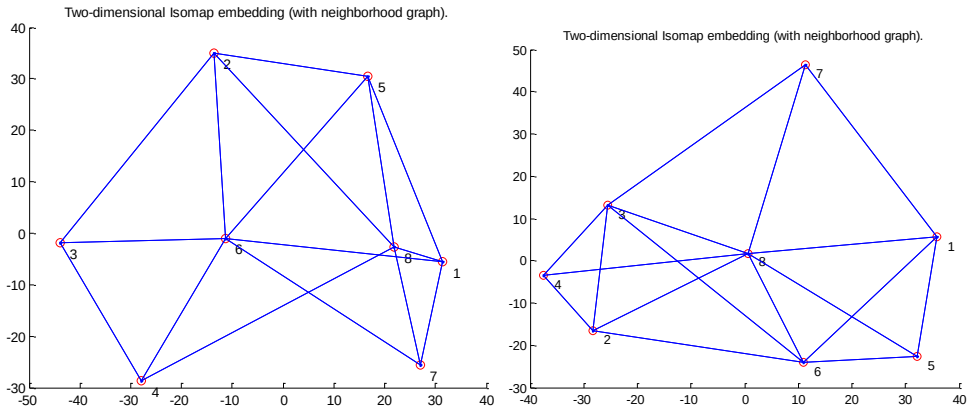

Figura 14 Mapa de vecindad de los modelos adaptados al locutor Álvaro (Raw izquierda, Proc. derecha)

De nuevo, en el caso del locutor Álvaro, se produce una interesante diferencia entre los modelos procesados y los que no lo están. En la Figura 14 podemos observar las diferentes relaciones de vecindad entre los modelos entrenados, que aparecen numerados del 1 al 8. En el caso de los modelos raw a la izquierda podemos observar una cercanía entre los modelos 1, 5 y 8. Estas posiciones corresponden a las medidas realizadas en el eje de grabación del dispositivo Kinect. Sin embargo en esta medida Raw, los puntos más alejados corresponden a las medidas con mayor diferencia de ángulo-distancia. Por otro lado, en el gráfico de modelos procesados, a la derecha, esta dependencia está más diluida, pero sin embargo es mucho más fuerte en los modelos 2, 3 y 4. Estos modelos, como se puede ver en la Figura 8, están posicionalmente cerca entre sí y con una angulación pronunciada respecto al eje. Esta relación más fuerte puede resultar de los efectos de Beamsteering de la Kinect en su captación, acercando los modelos que presentan diferentes ángulos.

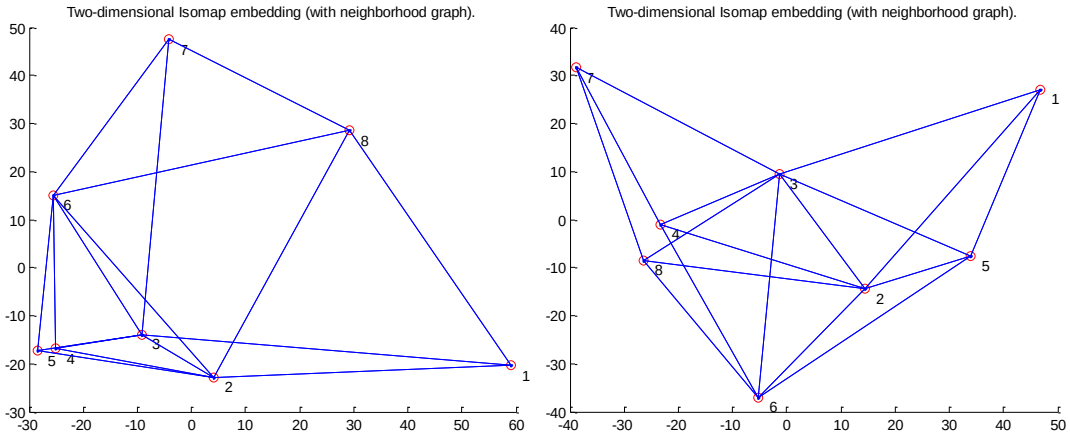

Figura 15 Mapa de vecindad de los modelos adaptados al locutor Anat (Raw izquierda, Proc Derecha)

En el caso del locutor Anat (Figura 15), los problemas derivados en la baja tasa de aciertos, probablemente debida a un peor modelado, también se manifiestan en las relaciones de vecindad, siendo difícil encontrar patrones de cercanía más allá de los de los puntos 2, 3 y 4, que de nuevo en este caso tienen relaciones muy fuertes, especialmente en el caso de procesado.

4. CONCLUSIONES Y LÍNEAS FUTURAS

En el presente proyecto se han implementado diferentes plataformas para la caracterización de Microsoft Kinect orientado al reconocimiento automático del habla y el desarrollo de aplicaciones específicas en este ámbito.

Microsoft Kinect supone una plataforma muy interesante para la creación de innovadores interfaces naturales de usuario, dentro de las diferentes alternativas existentes en el mercado, y proporciona, en un mismo dispositivo, una variada serie de sensores avanzados a un coste muy reducido. Unido a la presencia de un SDK desarrollado por Microsoft y puesto a disposición de la comunidad técnico-científica, convierte al dispositivo Kinect en una herramienta muy atractiva alimentada por una creciente comunidad de desarrolladores.

A través de la implementación de un reconocedor de palabras aisladas con adaptación MAP a los modelos y locutores y realizado en Matlab, ha sido posible evaluar las tasas de aciertos de Microsoft Kinect para dos locutores, en unas posiciones determinadas y diferentes a las de entrenamiento, sin utilizar todas las diferentes combinaciones de captura de audio de las que dispone. El objetivo inicial de esta TFM era determinar los efectos fundamentales de los diferentes procesados de audio que efectúa el conjunto Hardware-SDK sobre las muestras de audio tomadas, fundamentalmente cuando se comporta como un micrófono normal y cuando aplica conjuntamente técnicas de conformado de haz y supresión activa de eco y de ruido.

Analizando los resultados obtenidos, con las medidas disponibles, es posible discernir un comportamiento peor con el procesado de audio de reducción de ruido allí donde el ángulo de la fuente de sonido está cerca del eje de captura. Sin embargo, mejora su funcionamiento gracias al uso del conformado de haz para aquellos ángulos más alejados. Desde un punto de vista de análisis dimensional no lineal, también se ha demostrado que aparecen patrones de acercamiento de los modelos dependiendo de la acción de estos procesados. A este respecto, una posible línea de trabajo sería seguir profundizando en estas funciones de Beamsteering para lo cual sería interesante contar con una base de datos menos extensa en vocabulario pero más rica en posiciones geográficas respecto al eje de captura de Kinect. De esta forma sería posible seguir investigando en el análisis dimensional no lineal con unos resultados más precisos, al disponer de mayor cantidad de información.

Queda también pendiente para futuros proyectos la cuantificación de las tasas de acierto con un ruido conocido y reproducido de fondo, para así poder caracterizar de forma completa las opciones de procesado de señal de audio de Kinect. Esta opción ya queda recogida en la plataforma de software y puede ser reutilizada en futuras capturas de datos.

Respecto a la plataforma de software para Kinect, ésta ha sido implementada con la intención de crear una plantilla universal para el desarrollo de aplicaciones orientadas al habla, ya sea de forma experimental, lúdica o pedagógica, manteniendo unos principios de modularidad que permiten su adaptación tanto a aplicaciones antiguas como futuras. Durante el desarrollo de esta TFM, Microsoft ha ido mejorando progresivamente las funcionalidades del SDK de Windows, añadiendo nuevas funciones, optimizando otras y, en líneas generales, proporcionando nuevas herramientas de trabajo para el desarrollador, desde las primeras versiones beta hasta las diferentes versiones oficiales. Estas novedades fueron introducidas en la plataforma garantizando su compatibilidad en el futuro.

Las líneas de trabajo con el SDK de Microsoft y la plataforma de desarrollo implementada son muy variadas. Desde un punto de vista pedagógico, sería muy interesante desarrollar minijuegos orientados a la captura de datos de entrenamiento para niños con dificultades en el habla, que a su vez podría ser utilizado en el desarrollo de aplicaciones especiales o incluso personalizadas a estos usuarios. En cuanto al desarrollo de algoritmos de detección de gestos, se está convirtiendo por sí misma en una nueva disciplina dentro de las ciencias de la computación.

La utilización activa del Beamsteering abre nuevas posibilidades en el desarrollo de aplicaciones de diarización de locutores de forma puramente acústica o con retroalimentación de la detección de esqueletos. Un mayor conocimiento de los patrones de directividad de los micrófonos de Kinect cuando conforman haz y cómo varían según la dirección de éste en cuanto a lóbulos principales o secundarios, también sería una línea de investigación muy interesante.

5. DIAGRAMA DE GANTT

A continuación se presentan las diferentes tareas que se han ejecutado para la realización de la presente TFM y que se muestran en el diagrama de Gantt de la Figura 16:

- Documentación inicial: Teoría de RAH
- Aprendizaje Microsoft Kinect
    - Componentes Hardware
    - Búsqueda de SDK y Hardware más apropiado
    - Aprendizaje Kinect for Windows SDK
        - Documentación
        - Programación en C++
        - Análisis de las capacidades del SDK
        - Aprendizaje de la API de Kinect
        - Instalación, setup y verificación
- Programación framework para aplicaciones de Kinect
    - Programación framework base en C++
    - Programación versión base o esqueleto del minijuego en C++
    - Programación Voice-Sampler en C++
    - Actualización SDK
- Captura de señales de audio para entrenamiento y test
    - Grabación de las muestras de voz
    - Revisión de la calidad y la nomenclatura
- Programación reconocedor de palabras aisladas en Matlab
    - Programación de scripts y funciones del reconocedor
    - Experimentos y captura de datos
    - Análisis de resultados
- Redacción de la memoria

Esta TFM se ha realizado a tiempo parcial, dadas las condiciones laborales del autor durante el transcurso de su ejecución.

Código	Tareas
1	**Documentación inicial: Teoría de RAH**
2	**Aprendizaje Microsoft Kinect**
2.1	○ Componentes Hardware
2.2	○ Búsqueda de SDK y Hardware más apropiado
2.3	○ Aprendizaje Kinect for Windows SDK
2.3.1	• Documentación
2.3.2	• Programación en C++
2.3.3	• Análisis de las capacidades del SDK
2.3.4	• Aprendizaje de la API de Kinect
2.4.5	• Instalación, setup y verificación
3	**Programación framework para aplicaciones de Kinect**
3.1	○ Programación framework base en C++
3.2	○ Programación esqueleto minijuego en C++
3.3	○ Programación Voice-Sampler en C++
3.4	○ Actualización SDK
4	**Captura de señales de audio para entrenamiento y test**
4.1	o Grabación de las muestras de voz
4.2	○ Revisión de la calidad y la nomenclatura
5	**Programación reconocedor de palabras aisladas**
5.1	○ Programación de scripts y funciones
5.2	○ Experimentos y captura de datos
5.3	○ Análisis de resultados
6	**Redacción de la memoria**

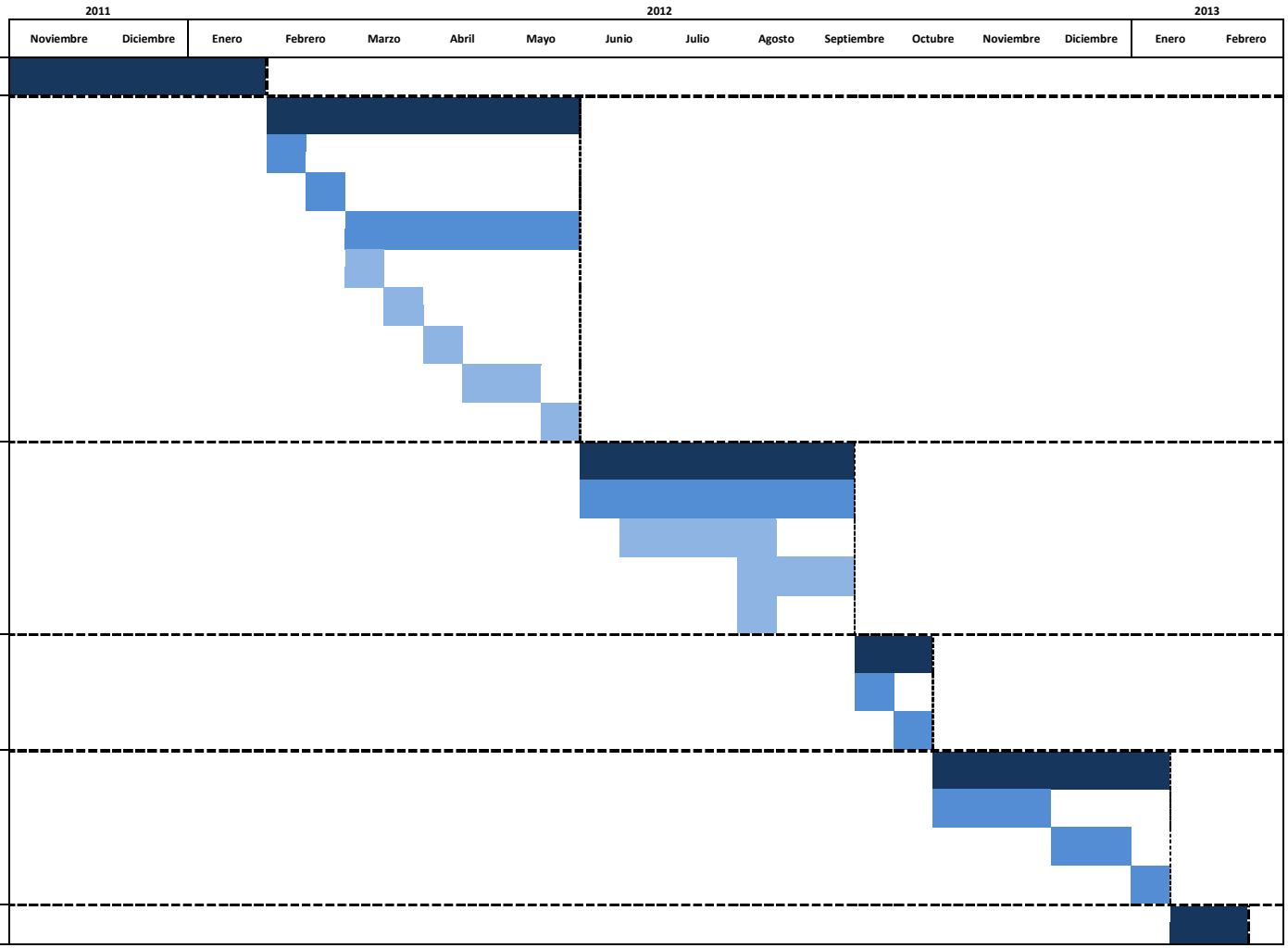

Figura 16 Diagrama de Gantt del proyecto y las tareas realizadas

6. BIBLIOGRAFÍA

[1] A. Ortega, F. Sukno, E. Lleida, A. Frangi, A. Miguel, L. Buera, E. Zacur. AV@CAR: A Spanish Multichannel Multimodal Corpus for In-Vehicle Automatic Audio-Visual Speech Recognition, Language Resources and Evaluation Conference (LREC): 763-766, 2004.

[2] F. Itakura, Minimum Prediction Residual Principle Applied to Speech Recognition, IEEE Trans. Acoustics, Speech and Signal Proc. 23 (1): 57-72, 1975.

[3] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences, IEEE Trans. Acoustics, Speech and Signal Proc. 28(4): 357-366, 1980.

[4] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition, Bell Syst. Tech. J. 62(4): 1035-1074, 1983.

[5] C. J. Leggetter and P.C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models, Computer Speech Language 9(2): 171-185, 1995.

[6] J. L. Gauvain and Chin-Hui Lee. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains, IEEE Transactions on Speech And Audio Processing 2(2): 291-298, 1994.

[7] Microsoft Corporation. HUMAN INTERFACE GUIDELINES Kinect for Windows v1.5.0.

[8] I. Tashev. Audio for Kinect: Pushing it to the limit, CREST Symposium on Human Harmonized Information Technology Kyoto University, April 2012.

[9] I. Horton. Beginning Visual C++ 2010, Wiley Publishing, Inc. 2010.

[10] A. Moreno, D. Poch, A. Bonafonte, E. Lleida, J. Llisterri, J.B. Mariño and C. Nadeu. ALBAYZÍN Speech Database: Design of the Phonetic Corpus, In: Eurospeach'93, Third European Conference on Speech Communication and Technology. Berlin, Germany 21-23 September 1993. Vol.1, pp. 175-178. ISSN: 1018-4074.

[11] J. B. Tenenbaum, V. de Silva and J.C. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science 290 (22 December): 2319-2323, 2000.

[12] K. H. Davis, R. Biddulph, and S. Balashek, Automatic Recognition of Spoken Digits, J. Acoust. Soc. Am. 24 (6): 627-642, 1952.

[13] J. Suzuki and K. Nakata, Recognition of Japanese Vowels—Preliminary to the Recognition of Speech, J. Radio Res. Lab. 37 (8): 193-212, 1961.

[14] T. B. Martin, A. L. Nelson, and H. J. Zadell, Speech Recognition by Feature Abstraction Techniques, Tech. Report AL-TDR-64-176, Air Force Avionics Lab, 1964.

[15] A. J. Viterbi, Error Bounds for Convolutional Codes and an Asymptotically Optimal Decoding Algorithm, IEEE Trans. Information Theory 13 (2): 260-269, 1967.

[16] F. Jelinek, Continuous Speech Recognition by Statistical Methods, Proc. IEEE 64(4): 532-556, 1976.

[17] Z. Ghahramani. An introduction to hidden Markov models and Bayesian networks, International Journal of Pattern Recognition and Artificial Intelligence 15 (1): 9-42, 2001.

[18] A. P. Dempster, N. M. Laird and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm, Journal of the Royal Statistical Society Series B (Methodological) 39(1): 1-38, 1977.

[19] R. Moore. Speech Processing, McGraw-Hill, 1990.

ANEXO A: RECONOCEDORES AUTOMÁTICOS DEL HABLA

A.1. Reconocimiento automático del habla

A.1.1. Introducción

El habla, en su definición formal, es el uso particular o individual por parte del ser humano de la lengua para comunicarse. Tradicionalmente, esta comunicación se realizaba de forma efectiva entre personas, actuando tanto como emisor y como receptor del mensaje, sin embargo, la aparición de la electrónica y la informática durante el siglo XX, introduce un nuevo elemento en la comunicación, siendo un paso natural el desarrollo de técnicas de comunicación oral hombre-máquina. Los reconocedores automáticos del habla (RAH) concentran su labor en este campo, siendo su objetivo final el perfeccionamiento de esta comunicación oral con múltiples beneficios sociales, académicos, industriales, militares y, finalmente, económicos.

El RAH es un proceso de clasificación de secuencias de patrones, capaz de procesar la señal de voz emitida por el ser humano y reconocer la información contenida en ella, convirtiéndola en texto o emitiendo órdenes que actúen en consecuencia. Esta tecnología permite la comunicación oral hombre-máquina, abriendo un amplio abanico de aplicaciones prácticas en diferentes sectores:

- Sistemas de dictado.
- Sistemas de control oral para personas con discapacidad.
- Sistemas de asistencia e información para personas con problemas de audición (p.e. subtitulación automática de contenidos audiovisuales).
- Aplicaciones telefónicas (p.e. servicios de asistencia técnica telefónica).
- Aplicaciones biométricas (p.e. identificación del locutor).
- Autoaprendizaje de idiomas
- Interfaces multimodales de interacción hombre-máquina. Equipamiento electrónico o informático, como ordenadores, smartdevices (smartphones o tabletas), módulos hardware de control (p.e. dashboard de un coche) y entretenimiento (p.e. Microsoft Kinect) que incluyen el habla como un elemento más de interacción con el usuario de forma bidireccional.

La investigación y desarrollo de RAH tiene como objetivo final el reconocedor ideal, es decir, aquel con una tasa de error nula. Sin embargo, la complejidad del sistema obliga al desarrollo e investigación de diversas técnicas para cada uno de los diferentes elementos de la arquitectura del reconocedor (parametrización, entrenamiento, modelado del lenguaje, adaptación del locutor…). A todo ello se unen los distintos avances en hardware y software procedentes de la electrónica y la informática, con un crecimiento exponencial de su capacidad de procesado y una disminución directa de su consumo y precio, lo que ha permitido que en los últimos años aparezcan, finalmente, las primeras aplicaciones basadas en RAH destinadas a un mercado global y a un precio razonable para el consumidor.

A.1.1. Estado del arte

Diseñar una máquina capaz de imitar al ser humano y su comportamiento, y particularmente su capacidad para hablar de forma natural, ha sido objeto de estudio por parte de ingenieros y

científicos a lo largo de los siglos. En los años 1930, los laboratorios Dell [12], propusieron un modelo para el análisis y síntesis del habla, dando inicio a una aproximación sistemática al problema del reconocimiento automático del habla que progresivamente ha llevado de un autómata capaz de responder ante un pequeño número de sonidos hasta los actuales sistemas, capaces de responder ante largas frases expresadas con fluidez y naturalidad, teniendo en cuenta la variabilidad estadística del idioma utilizado.

La primera mitad del siglo XX pone de manifiesto la importancia de la respuesta frecuencial como modo de identificación de la naturaleza fonética de la señal de voz, sentando la base de los sistemas modernos de RAH, y de los algoritmos utilizados, identificando el concepto de medida de la potencia espectral del habla (y sus variaciones temporales) o de sus variantes como el cepstrum. Continuando con esta línea, se determina la relevancia del uso de los formantes, modos de resonancia naturales del tracto vocal, como las zonas de mayor concentración de energía de los fonemas y su utilidad en el reconocimiento de palabras aisladas [13]. También se comienzan a utilizar técnicas de análisis estadístico mejorando la fiabilidad de estos primeros reconocedores.

Durante los años sesenta se desarrollaron diferentes tecnologías como el análisis por banco de filtros, métodos de normalización temporal [14] y los primeros pasos de las metodologías de programación dinámica [15], permitiendo el reconocimiento de pequeños vocabularios, en torno a 10-100 palabras aisladas.

La década de los setenta introduce los modelos de reconocimiento de patrones, los métodos LPC de representación espectral [2], programación dinámica para resolver la problemática de concatenación de palabras, etc... Aumentando los vocabularios en un orden de magnitud, entre 100-1000 palabras.

Los ochenta incluyen finalmente los modelos ocultos de Markov (HMM) y modelos estocásticos del lenguaje [4] [16]. Estas dos tecnologías clave tienen como consecuencia un avance sustancial de los vocabularios de los RAH, permitiendo grandes diccionarios de entre 1000 e infinitas palabras, siendo ya limitados por otras condiciones gracias al uso de métodos estadísticos y que, finalmente, permite afrontar problemas de reconocimiento del habla continua de una forma eficaz.

En los años 90 continúan las líneas maestras trazadas en la década anterior. Se construyen grandes vocabularios sin limitaciones en los modelos de lenguaje y avances en el reconocimiento y comprensión de habla continua. Se introducen métodos de comprensión estocástica del lenguaje, aprendizaje estadístico de modelos acústicos y del lenguaje, así como un entorno de máquina de estados finitos. Cabe destacar la aparición del HTK Toolkit[4] (Hidden Markov Model Toolkit) de la Universidad de Cambridge en 1999, herramienta versátil y abierta para el manejo y manipulación de HMMs y de uso muy extendido en el ámbito de la investigación de RAH.

En la pasada década, es importante destacar la integración de los RAH en diferentes aplicaciones de uso cotidiano. La extensión de dispositivos inteligentes (smartphones,

[4] http://htk.eng.cam.ac.uk/

tabletas), con conectividad a internet ha aumentado de forma exponencial la exposición de la gente a este tipo de interacción hombre máquina, dejando de ser una excepción para convertirse en un elemento más de comunicación con dispositivos electrónicos e informáticos. También se ha de destacar la presencia de interfaces multimodales que incluyen los RAH como un elemento más de interacción, dando lugar a HMIs avanzados con carácter militar, académico, y lúdico.

A.1.2. Arquitectura

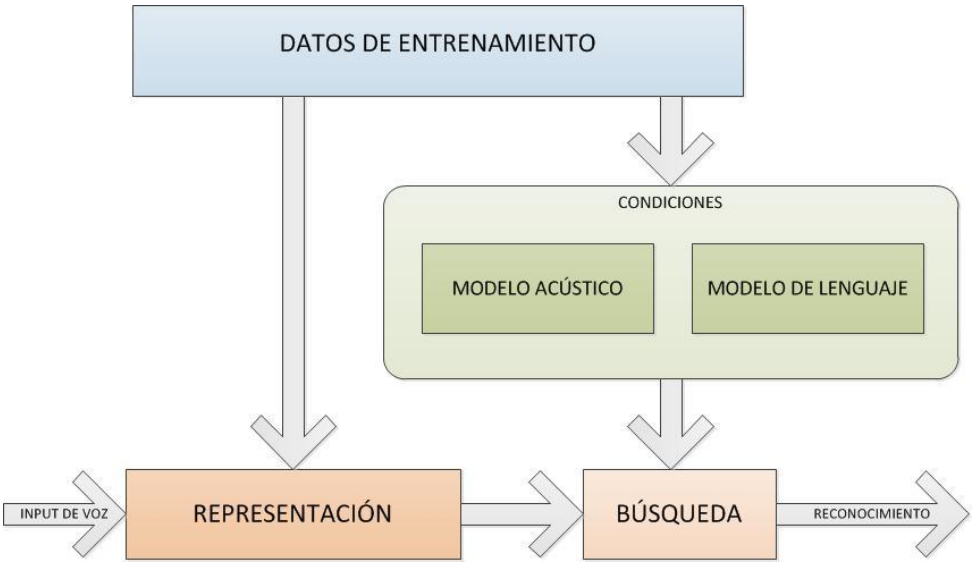

Figura 17 Esquema de la arquitectura de un RAH genérico

Funcionamiento

A partir de los diferentes desarrollos tecnológicos en el ámbito del RAH descritos en el apartado anterior, actualmente y de forma genérica, los RAH que ofrecen mejores resultados son aquellos basados en el teorema de Bayes, la teoría de la información y las técnicas de comparación de patrones y programación dinámica.

De acuerdo a la aproximación a partir del teorema de Bayes, podemos describir un reconocedor automático del habla como un sistema en el que se debe disponer de unos patrones asociados a las partes del habla que se van a reconocer (fonemas, palabras, frases…), de manera que procesando una secuencia de vectores de características u observaciones $\boldsymbol{O}$, tengamos como respuesta la secuencia de patrones que representan a este vector con la probabilidad mayor.

La secuencia de vectores $\boldsymbol{O}$ es extraída y procesada a partir de las señales de audio capturadas mediante un micrófono u otro tipo de transductor a partir de diferentes

métodos de extracción de características. Este conjunto de observaciones $\boldsymbol{O}$ se puede definir como:

$$\boldsymbol{O} = (\boldsymbol{o}_1,...,\boldsymbol{o}_t,...,\boldsymbol{o}_T)\ t \in [1,T] \qquad \text{Ecuación A-1}$$

siendo $\boldsymbol{o}_t$ el vector de características observado en el instante t, siendo T el número de observaciones.

La salida deseada del sistema será una secuencia de fonemas (o palabras si estas son la unidad mínima sintáctica) que deberían aproximarse a las pronunciadas por el locutor:

$$\boldsymbol{W} = (\boldsymbol{w}_1,...,\boldsymbol{w}_j,...,\boldsymbol{w}_N) \qquad \text{Ecuación A-2}$$

donde N es el número de palabras o fonemas de la secuencia.

El RAH deberá aproximarse a esta secuencia de fonemas deseada, que desde un punto de vista probabilístico será aquella secuencia con mayor probabilidad de acuerdo a las observaciones $\boldsymbol{O}$. Expresado de forma matemática:

$$\hat{\boldsymbol{W}} = \arg\max_{\mathrm{W}} P(\boldsymbol{W} \mid \boldsymbol{O}) \qquad \text{Ecuación A-3}$$

Dado que esta probabilidad no es calculable directamente, aplicamos el teorema de Bayes:

$$\hat{\boldsymbol{W}} = \arg\max_{\mathrm{W}} \frac{P(\boldsymbol{O} \mid \boldsymbol{W}) \cdot P(\boldsymbol{W})}{P(\boldsymbol{O})} \qquad \text{Ecuación A-4}$$

- $P(\boldsymbol{W})$ es la probabilidad de la secuencia de palabras $\boldsymbol{W}$, que se define como el modelo de lenguaje
- $P(\boldsymbol{O} \mid \boldsymbol{W})$ es la probabilidad de observar la secuencia $\boldsymbol{O}$ cuando se pronuncia la secuencia $\boldsymbol{W}$
- $P(\boldsymbol{O})$ se refiere a la probabilidad de las secuencias acústicas. Se puede eliminar al no afectar a la maximización.

Obtenemos de esta forma la fórmula fundamental del reconocimiento automático del habla:

$$\hat{\boldsymbol{W}} = \arg\max_{\mathrm{W}} P(\boldsymbol{O}\,|\,\boldsymbol{W}) \cdot P(\boldsymbol{W}) \qquad \text{Ecuación A-5}$$

En el esquema de la arquitectura de un reconocedor automático del habla podemos encontrar entonces dos procesos independientes:

- Fase de entrenamiento: Fase en la que el reconocedor aprende, a partir de las muestras de voz y texto, los modelos acústicos $P(\boldsymbol{O}\,|\,\boldsymbol{W})$ y los modelos de lenguaje $P(\boldsymbol{W})$.
- Fase de reconocimiento: En ella se produce el reconocimiento propiamente dicho obteniendo como salida la secuencia de palabras reconocidas.

Como se puede observar en el esquema del reconocedor genérico (Figura 17), existen diferentes módulos dentro de cada una de las fases de las que está compuesto:

Base de datos

En la base de datos podemos encontrar las palabras que utilizaremos para entrenar al sistema. Estas palabras deben ser elegidas con cuidado atendiendo a factores como la inclusión de todos los fonemas del lenguaje a reconocer y, además, deben estar balanceadas, es decir, que estos fonemas tengan una tasa de aparición similar a su presencia en dicho lenguaje. Estas palabras deben ser suficientes, ya que su número tiene incidencia directa en la calidad del reconocedor.

Extracción de características

El módulo de extracción de características, común a la fase de entrenamiento y de reconocimiento, tiene como objetivo procesar las señales de audio tomadas del locutor con el objetivo de resaltar la información relevante para el RAH y eliminar aquella que no tenga importancia fonética. Tradicionalmente, dos son las técnicas utilizadas en la extracción de características, Linear Prediction Coefficients (LPC)[2] y Mel-Frequency Cepstral Coefficients (MFCC)[3]. Esta última técnica será la utilizada en esta TFM.

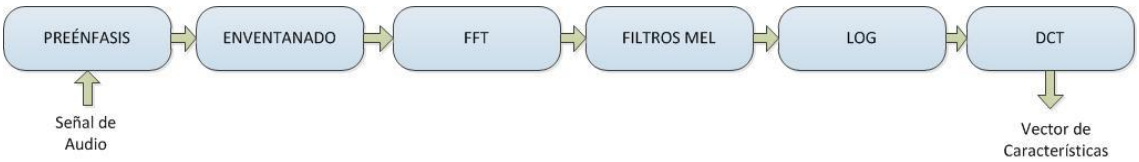

Figura 18 Cadena de tratamiento de señales de voz para extraer su vector de características

El proceso de extracción de características a partir de la técnica MFCC se puede observar en la Figura 18. Comienza con la aplicación de un preénfasis de las altas frecuencias para compensar el desplazamiento frecuencial de los datos sonoros debido a la forma de radiación de los labios en la fonación. Posteriormente, se realiza un proceso de enventanado, de 25ms de duración, con la posibilidad de añadir un solape de entorno a los 10 ms para poder obtener una mayor resolución temporal a la hora de realizar la transformada de Fourier de estos datos enventanados. A esta información espectral se le aplican los filtros de Mel (banco de filtros de frecuencia y amplitud variable que replican las ventanas perceptuales del oído humano),

introduciendo factores psicoacústicos en la extracción de datos y aumentando la resolución en las medias y bajas frecuencias. Finalmente, se aplica la Transformada Discreta del Coseno (DCT) para decorrelar los coeficientes del vector resultante. Los coeficientes obtenidos con esta técnica poseen una gran cantidad de información en el dominio espectral (de especial relevancia en los formantes). Sin embargo, el vector de características que finalmente será utilizado en el reconocedor incluye otros elementos de información en el dominio temporal (energía, frecuencia de los formantes, velocidad de la fonación) para poder reflejar la variabilidad temporal del habla.

Modelo acústico

Existen diferentes métodos para obtener el modelado acústico, siendo los más utilizados, como en el caso de la presente TFM, los relacionados con los modelos ocultos de Markov o HMM (Hidden Markov Models)[4]. Los modelos ocultos de Markov son una máquina de estados finita, cuyas observaciones son una función probabilística del estado. En otras palabras, un HMM es un proceso doblemente estocástico formado por:

1) Un proceso estocástico oculto, no observable directamente, que corresponde a las transiciones entre los diferentes estados.
3) Un proceso estocástico observable cuya salida es una secuencia de vectores de voz.

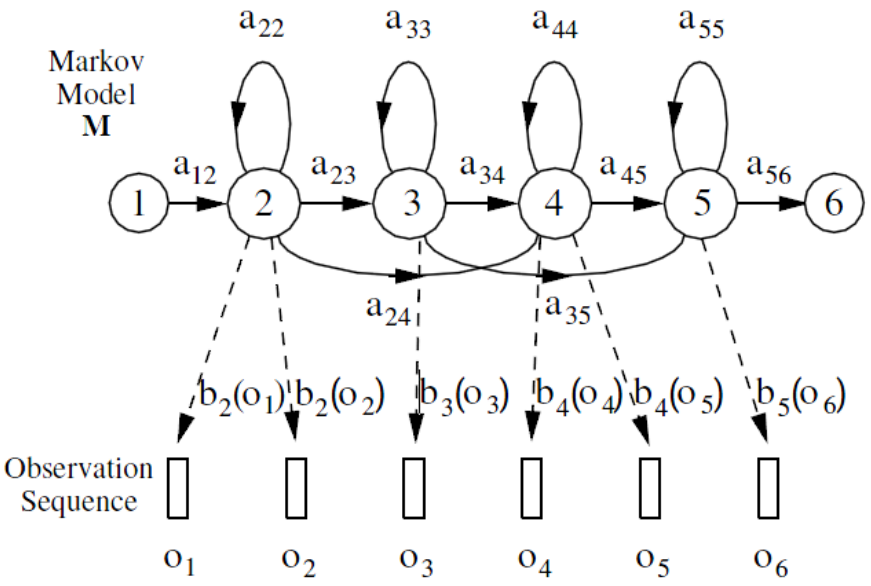

Figura 19: Estructura genérica de un HMM

A diferencia de un modelo de Markov, en el que son observables los propios estados, en un HMM, estos estados no son visibles, pero sí lo son las variables influenciadas por dicho estado. La unidad mínima de habla elegida (fonema, sílaba, palabra, frase…) queda representada como una sucesión de estados "ocultos", cada uno de los cuales está caracterizado por una función de densidad de probabilidad o pdf (probability density function) y una probabilidad de transición al resto de posibles estados. Existen diferentes niveles de HMMs, sin embargo, frente a opciones más complejas (matemática y computacionalmente hablando) son muy comunes el uso de HMM de orden 1[17], cuyas restricciones son: 1) el proceso estocástico de generación de las observaciones sólo depende en cada momento de un único estado

(independencia de la salida); 2) la transición entre estados sólo depende de los estados origen y destino; y 3) el proceso es de izquierda a derecha en la transición de estados (no es ergódico).

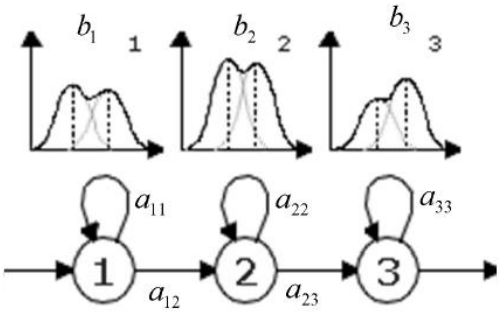

Figura 20 HMM de orden 1

Dadas estas restricciones, los parámetros fundamentales que definen un modelo oculto de Markov de orden 1 son:

1) $\boldsymbol{s}$: Es el conjunto de estados del modelo, siendo N el número de estados que lo forman.

$$\boldsymbol{S} = (\boldsymbol{s_1},...,\boldsymbol{s_j},...,\boldsymbol{s_N}) \quad \text{Ecuación A-6}$$

2) $b_j, 1 \le j \le N$: pdf asociada al estado j-ésimo.

$$\boldsymbol{B} = \{b_{jt}\} \quad \text{Ecuación A-7}$$

3) $a_{ij}, 1 \le i \le N, 1 \le j \le N$: Se corresponde con la probabilidad de transición del estado i al j

$$a_{ij} = P(\boldsymbol{s}_i \mid \boldsymbol{s}_j) \; \boldsymbol{A} = \{a_{ij}\} \quad \text{Ecuación A-8}$$

4) Una matriz de estados iniciales $\boldsymbol{\Pi} = \{\pi_i\}$ con $\pi_i = P(\boldsymbol{s}_i)$

Definido el número de estados N y las matrices $\boldsymbol{A}$, $\boldsymbol{B}$ y $\boldsymbol{\Pi}$, se puede describir un HMM como $\lambda\{\boldsymbol{A},\boldsymbol{B},\boldsymbol{\Pi}\}$.

Para completar el modelo, sería necesario definir la pdf de cada estado. En este proyecto se utilizarán Gaussian Mixture Models, que son mezclas de Gaussianas ponderadas con diferentes pesos, junto a la restricción número 2 de un HMM de orden 1, para la observación en el instante t en el estado j:

$$b_j(\boldsymbol{o}_t) = \sum_{c=1}^{C} w_{\boldsymbol{j,c}} \boldsymbol{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_{j,c}, \boldsymbol{\Sigma}_{j,c}) \quad \text{Ecuación A-9}$$

Estas GMM se obtienen mediante la utilización del algoritmo iterativo EM (Estimation-Maximization)[18], encargado de realizar una estimación de máxima verosimilitud de parámetros de problemas en los que existen ciertas variables ocultas, obteniendo los parámetros que las representan $\boldsymbol{\mu}_{j,c}$, $\boldsymbol{\Sigma}_{j,c}$, media y covarianza para cada una de las gaussianas ponderadas por $w_{j,c}$.

Para terminar de definir correctamente un HMM, además del número de estados N y $\lambda\{\boldsymbol{A},\boldsymbol{B},\boldsymbol{\Pi}\}$, es necesario el número de Gaussianas que se utilizarán en la mezcla, definido en la Ecuación A-9.

El entrenamiento de los HMM se realiza generalmente mediante la estimación de máxima verosimilitud ML (Maximum Likelihood) usando el algoritmo EM.

Adaptación supervisada

La variabilidad en el habla es uno de los factores más complejos que se encuentran en los RAH. Esta variabilidad se agrupa en dos grandes grupos (ver Tabla 5), factores intrínsecos (variabilidad del contexto, del locutor, estilo…) y factores extrínsecos, como las condiciones ambientales en las que se produce la adquisición de las señales de voz. Uno de los objetivos de esta TFM es el estudio y la compensación de los efectos del canal haciendo uso del procesado de audio de Kinect.

Dentro de los factores intrínsecos, la variabilidad en el habla de los diferentes locutores se postula como una línea de desarrollo de importancia capital. Cada individuo posee una forma única de hablar, determinada por características fisiológicas (forma del tracto vocal, longitud de las cuerdas vocales), modificadas por condiciones psicológicas y coyunturales (lengua materna o no, dificultades en la pronunciación, resfriados o congestiones…). Esta variabilidad por individuo o por circunstancias temporales, complica sobremanera el desarrollo de los reconocedores.

Los modelos independientes del modelo se aproximan a este problema entrenando dichos modelos con el mayor número posible de locutores, reforzando la diversidad de características y acentos. Sin embargo, estos modelos acaban siendo demasiado generales. Con el objetivo de mejorar estos modelos generalistas, se utilizan otros modelos especializados usando información de los locutores a estudio y obteniendo los denominados modelos adaptados. Frente a una aproximación de modelo monolocutor (en el que tenemos mucha información de un único locutor a estudio), generalmente en un sistema multilocutor, la información de los diferentes locutores es mucho más reducida o nula, por lo que se debe buscar la forma de adaptar nuestros modelos de forma eficiente para que en la práctica funcionen como sistemas monolocutor.

Las técnicas de adaptación supervisada se encargan de esta tarea, siendo dos de los algoritmos más utilizados el MLLR y el MAP. Se denominan supervisados porque para el correcto

entrenamiento de los nuevos modelos adaptados es necesaria una transcripción correcta del material de entrenamiento.

Maximum Likelihood Linear Regression (MLLR)

Se trata de una adaptación lineal con la que se realiza una serie de transformaciones para reducir las diferencias entre el modelo independiente y las locuciones de entrenamiento de cada locutor [5]. MLLR adapta cada una de las gaussianas al locutor aplicando una transformación lineal:

$$\hat{\boldsymbol{\mu}}_{\boldsymbol{jc}} = W_c \boldsymbol{\mu}_{\boldsymbol{jc}} \qquad \text{Ecuación A-10}$$

siendo $\boldsymbol{\mu}_{\boldsymbol{jc}}$ el vector de medias de la c-ésima Gaussiana en el estado j y W_c la matriz de transformación lineal de medias

$$\hat{\boldsymbol{\Sigma}}_{j,c} = \boldsymbol{H}\boldsymbol{\Sigma}_{j,c}\boldsymbol{H} \qquad \text{Ecuación A-11}$$

donde $\boldsymbol{\Sigma}_{j,c}$ es la matriz de covarianzas de la c-ésima gaussiana en el estado j y $\boldsymbol{H}$ la matriz de transformación.

Se puede utilizar MLLR de dos formas para proceder con la adaptación: 1) global (se adaptan las medias, las varianzas, o ambas, de todas las Gaussianas de cada estado de forma global) o 2) se dividen las Gaussianas en clases de regresión y se aplica la adaptación por separado.

El MLLR global es recomendable cuando no se tienen muchos datos de entrenamiento. En el segundo caso, dependiendo de la cantidad de datos, se determinará qué Gaussianas por separado pueden ser adaptadas o no. En el caso de tener una cantidad de datos suficiente, esta adaptación será más efectiva que en el caso global.

Maximum a Posteriori (MAP)

Esta adaptación utiliza un modelo inicial genérico de partida, o modelo independiente del locutor, cuya información es por tanto conocida a priori. Con la información nueva observada en el locutor particular, combinada con la conocida, y ponderadas por un factor de adaptación fijado de antemano, se estima la nueva información [6].

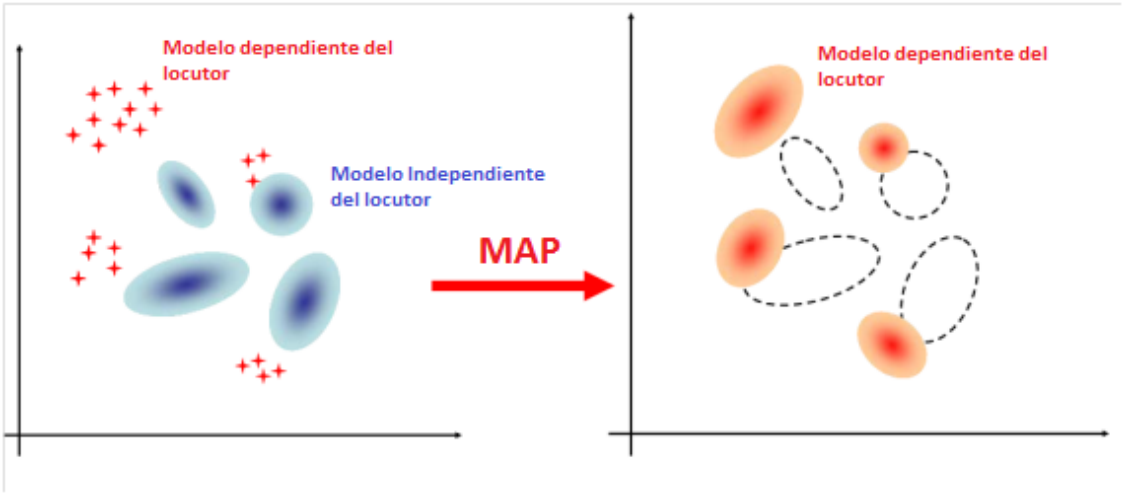

Figura 21 Proceso de adaptación supervisada de las gaussianas siguiendo un proceso MAP

Matemáticamente, la actualización de las medias se realiza a partir de:

$$\hat{\boldsymbol{\mu}}_{jc} = \frac{N_{jc}}{N_{jc}+\tau}\bar{\boldsymbol{\mu}}_{jc} + \frac{\tau}{N_{jc}+\tau}\boldsymbol{\mu}_{jc} \qquad \text{Ecuación A-12}$$

- $\boldsymbol{\mu}_{jc}$: media del modelo independiente en el estado j y gaussiana c
- $\bar{\boldsymbol{\mu}}_{jc}$: media de los datos de adaptación
- N_{jc} : probabilidad de ocupación de los datos de adaptación
- τ : peso de la información a priori

Para una N_{jc} pequeña, se puede apreciar que la nueva media se mantendrá cercana a los valores del modelo independiente. A su vez, el valor de τ también determinará la cercanía del valor de la media al valor del locutor o al del modelo independiente.

La técnica MAP será utilizada en este TFM a la hora de adaptar los modelos genéricos simples a cada locutor.

MAP necesita más datos del locutor para obtener buenos resultados que MLLR (al actuar sobre cada una de las Gaussianas) Sin embargo, cuando éste no es el caso suele dar un mejor resultado que MLLR.

Modelo de lenguaje

El modelo de lenguaje incorpora el conocimiento lingüístico necesario en el RAH, es decir, las restricciones propias que se imponen en el modo en el que se concatenan las palabras o los fonemas para una determinada tarea del reconocimiento. Matemáticamente responden a la probabilidad $P(\boldsymbol{W})$ de la fórmula del reconocedor (Ecuación A-5). Las secuencias de fonemas no pueden ser representadas probabilísticamente de un modo completo (por razones computacionales), por eso se tiende a acotar la dependencia entre vocablos próximos. En este aspecto se juega con la necesidad de crear dependencias de n fonemas (N-gramas) que guíen

la tarea del reconocedor pero sin llegar a un número elevado de N fonemas ya que restarían flexibilidad al mismo. En esta TFM, el modelo de lenguaje es secundario ya que se centrará en el reconocimiento de palabras aisladas de acuerdo a parámetros espaciales, algorítmicos y de hardware. Sin embargo, es importante destacar la existencia de este módulo dentro del reconocimiento automático del habla.

Reconocimiento

En este módulo es donde finalmente se encuentra la secuencia de fonemas que maximiza la ecuación fundamental del reconocedor (Ecuación A-5). La solución directa sería comparar cada posible secuencia con la estimación aproximada y la elección de la que diera un resultado mayor. Esta solución, aunque válida para vocabularios pequeños, sería computacionalmente demasiado exigente para un sistema de vocabulario estándar. En el caso de esta TFM, cuyo objetivo es la caracterización de Microsoft Kinect en el reconocimiento del habla, la resolución de la palabra estimada se calculará de forma directa. No obstante, es importante destacar las alternativas a utilizar en un escenario de habla continua.

Como solución a este problema se recurre al algoritmo de Viterbi [15] para disminuir los requerimientos computacionales. La idea fundamental del algoritmo de Viterbi para reconocimiento de palabras consiste en recorrer el diagrama de transiciones de estados a través del tiempo, almacenando en cada estado la máxima probabilidad acumulada hasta llegar a dicho estado y el estado anterior desde el que se llega con dicha probabilidad. La máxima probabilidad acumulada se obtiene a partir de la ecuación de recurrencia:

$$\mathrm{P}_t(j) = \max_i \left\{ \mathrm{P}_{t-1}(i) a_{ij} \right\} b_j(\boldsymbol{o}_t) \quad 2 \le t \le T; 1 \le j \le N \qquad \text{Ecuación A-13}$$

Esta ecuación indica que la probabilidad del mejor camino que termina en el instante t y en el estado j, se obtiene de elegir el camino de mayor probabilidad de entre todos los caminos que se inician desde los N estados posibles en t-1, y que acaban en el estado j en t, y multiplicar este valor por la probabilidad de emisión del símbolo $\boldsymbol{o}$ en el instante t.

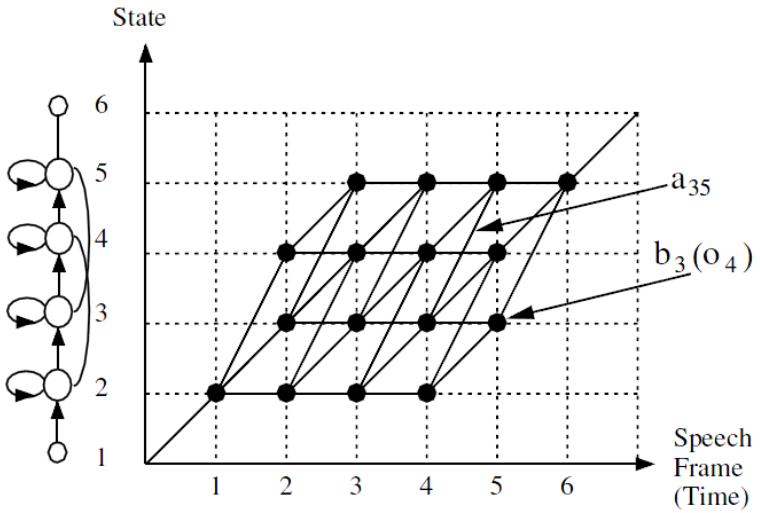

Figura 22 El algoritmo de Viterbi para reconocimiento de palabras aisladas

Visualmente, como se puede apreciar en la Figura 22, el algoritmo de Viterbi se puede interpretar como la búsqueda del mejor camino en una matriz cuyo eje vertical representa los estados posibles y el eje horizontal representa los frames de datos de entrada, y por ende, el tiempo. Cada punto representa la log-verosimilitud de observar ese frame en ese instante, y cada unión de puntos representa la log-verosimilitud de transición entre ellos. De esta forma, la probabilidad de cada camino se calcula como la suma de los logaritmos de las probabilidades de transición y las probabilidades de salida de los puntos recorridos.

A.1.3. Clasificación y precisión de RAH

De acuerdo a las características y capacidades de los RAH [19], estos pueden clasificarse de acuerdo a las variables descritas en la Tabla 4.

Tabla 4 Tipos de Reconocedores Automáticos del Habla

<table>
<tr><th>Locutores</th><th>Estilo del habla</th><th>Vocabulario</th><th>Estructura de dialogo</th><th>Condiciones de trabajo</th></tr>
<tr><td>Independientes</td><td>Palabras aisladas</td><td rowspan="2">Pequeño (<100 palabras)</td><td rowspan="2">Comandos simples</td><td rowspan="2">Condiciones de laboratorio</td></tr>
<tr><td>Dependientes</td><td>Habla continua</td></tr>
<tr><td>Adaptados</td><td>Habla leída</td><td rowspan="3">Gran vocabulario (> 1000 palabras)</td><td rowspan="3">Lenguaje natural</td><td rowspan="3">Condiciones de campo</td></tr>
<tr><td>Monolocutor</td><td rowspan="2">Habla espontánea</td></tr>
<tr><td>Multilocutor</td></tr>
</table>

A su vez, como se indica en el apartado relativo a adaptación, muchos son los factores que afectan al habla, otorgando una gran variabilidad a la voz, lo cual implica mayores dificultades a la hora de implementar un RAH. Estos factores pueden ser propios del reconocedor según su arquitectura e implementación, o intrínsecos, mientras que los que son ajenos a éste se denominan extrínsecos. La Tabla 5 presenta un resumen de los principales factores intrínsecos y extrínsecos que afectan a los RAH.

Tabla 5 Factores que afectan a la precisión de un RAH

<table>
<tr><th colspan="2">Factores que afectan a la precisión de un RAH</th></tr>
<tr><td>Número de palabras del vocabulario</td><td rowspan="8">Intrínsecos</td></tr>
<tr><td>Palabras del vocabulario pueden llevar a confusión fonética</td></tr>
<tr><td>Dependencia/independencia del sistema respecto al locutor</td></tr>
<tr><td>Palabras aisladas, habla discontinua o habla continua</td></tr>
<tr><td>Uso de reglas gramaticales (como ayuda para determinar la validez de lo reconocido)</td></tr>
<tr><td>Habla leída o habla espontánea</td></tr>
<tr><td>Condiciones de trabajo (ruido, efectos acústicos de sala, limitaciones de los micrófonos)</td></tr>
<tr><td>Aprovechamiento de las relaciones entre los elementos de la jerarquía utilizada en el RAH
(p.e. fonemas → palabras →frases: Hay combinaciones de palabras que no se pueden dar en una frase, detectándose un error en el reconocimiento)</td></tr>
</table>

Variabilidad del locutor (prosodia particular, dificultades en la fonación, edad, género…)	
Frecuencia de muestreo de la señal de voz (p.e. limitaciones del sonido telefónico)	**Extrínsecos**
Parámetros de ajuste de los diferentes algoritmos y aproximaciones utilizados	

ANEXO B: MICROSOFT KINECT

B.1. Introducción: Interfaces multimodales e Interfaces Naturales de Usuario

En los últimos tiempos y con el aumento de la interactividad hombre-máquina, han surgido los denominados interfaces multimodales. Dentro de esta definición podemos englobar a todos aquellos sistema encargados de la relación entre usuarios y tecnología entre los que se incluyen de forma integrada diferentes métodos de entrada, y también de salida, de información a través de diferentes interfaces. Frente a la clásica interacción a través de teclado y ratón o interfaces táctiles sobre diferentes opciones de menú, los interfaces multimodales añaden diferentes métodos como la captura de movimientos corporales o el uso de reconocedores automáticos del habla.

Dentro de estos interfaces multimodales cabe destacar las nuevas iniciativas de desarrollo de interfaces naturales de usuario o NUI (Natural User Interfaces). NUI es la designación aceptada para aquellos HMI (Human Machine Interfaces) que se refieren a una interfaz de usuario que 1) es invisible o se vuelve invisible tras una serie de interacciones y 2) está basada en la naturaleza o elementos naturales. Se aplica el adjetivo natural como contraposición a la utilización tradicional de elementos de interacción en GUIs (Graphical User Interface), en los que es necesario un proceso de aprendizaje previo a su utilización a través de diferentes elementos como ventanas, símbolos (p.e. maximizar, minimizar, cerrar) o rutinas aprendidas (clic, doble-click, click derecho). Los NUIs buscan que la relación hombre-máquina sea natural e instintiva y que necesite un periodo de aprendizaje mínimo o de autoaprendizaje sobre la marcha [7].

A nivel de hardware, diferentes dispositivos orientados al desarrollo de NUIs han sido introducidos en el mercado, destacando entre todos ellos el sensor Kinect de Microsoft.

B.2. Microsoft Kinect

Tras un tiempo de espera y entre diferentes rumores, el secreto Proyecto Natal de Microsoft sale a la luz el año 2010 con la denominación Microsoft Kinect[5]. Se trata de un controlador de videojuegos para su consola XBOX360 y la primera iniciativa comercial de NUI en el mercado global. Rápidamente se convierte en un éxito de ventas, siendo el dispositivo electrónico de más rápida venta hasta la fecha con 8 millones de unidades en 60 días.

Figura 23 Kinect y sus componentes

[5] http://www.xbox.com/es-ES/kinect

El controlador Kinect aúna en un solo dispositivo diferentes métodos de entrada de información del usuario, como se puede apreciar en la Tabla 6, orientados a la implementación de NUIs.

Tabla 6 Características técnicas del sensor Microsot Kinect

	Hardware
Video	• Cámara con señal de salida RGB o YUV de 640x480@8bits@30fps • Señal de profundidad de 640*480@11bits@30fps obtenida a partir de un emisor de matriz de infrarrojos y un receptor de infrarrojos CMOS, obteniendo la profundidad por cada pixel.
Procesado de Video	• Sensor de profundidad basado en la detección de la divergencia de la emisión de una matriz estructurada de infrarrojos. • Detección de 20 articulaciones por jugador activo • Detección de hasta 6 personas con 2 jugadores activos • En caso de articulaciones ocultas, sistema de inferencia y estimación de las mismas con parámetro de fiabilidad. • Estimación automática del plano de suelo
Audio	• Array de cuatro micrófonos • Frecuencia de muestreo de 16KHz y 24 bit de resolución
Procesado de audio	• Array de 4 micrófonos para beam detection: Resolución de 10° de aproximadamente -50° a 50° de detección. • Automatic Echo Cancellation • Noise suppresion • Automatic Gain Control
Motor	• Control de inclinación del módulo
Acelerómetro	• Utilizado para estimar inclinación de la cámara
Ángulo de visión	• 43° Vertical, 57° Horizontal
Rango	• Modo lejano: 1.2-3.9m • Modo cercano: 0.4-3m (solo Kinect for Windows)
Resolución	• Resolución espacial x/y: 3mm @2m • Resolución de profundidad z: 1cm @2m

Kinect fue desarrollado por dos compañías, PrimeSense, encargada de los sensores de profundidad, y Rare, subsidiaría de Microsoft. PrimeSense continuaría desarrollando sensores de profundidad de forma propietaria y para otras marcas.

En la primavera de 2012, Microsoft lanza al mercado Kinect for Windows, evolución de la Kinect original orientada a sistemas PC, incluyendo una serie de novedades, siendo la principal un modo de detección de profundidad en primer plano, pensado para distancias cortas en torno a 40 cm.

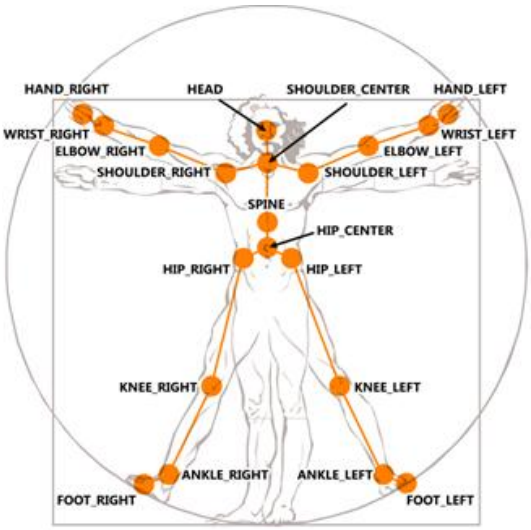

Figura 24 Articulaciones monitorizadas por el motor de esqueletos de Kinect[6]

Microsoft Kinect incluye sistemas de captura de audio y video con capacidad de procesado por hardware en la propia consola, siendo transmitido un flujo de audio y video hasta el PC.

A grandes rasgos, desde el punto de vista de imagen, Kinect[7] es capaz de capturar mapas de profundidad gracias a un emisor de luz estructurada infrarroja y un receptor CMOS. Este haz de luz estructurada de acuerdo a una malla de puntos pseudoaleatorios "choca" con los objetos y rebota hasta el receptor CMOS. A partir de la distorsión de puntos, Kinect procesa la distancia de cada uno de estos píxeles con una precisión de en torno a 1 cm a dos metros de distancia. Además Kinect incluye una cámara que permite recibir en el PC un flujo de video de hasta 1024x768 pixeles en formato RGB o YUV. Desde el punto de vista de tratamiento de señal de imagen, Kinect es capaz de detectar la presencia de usuarios y generar un esqueleto de jugador capturando hasta 20 articulaciones con una tasa de 30 fps y con un máximo de hasta 2 jugadores activos de pie, o 4 jugadores sentados (en la última actualización del SDK). A su vez, Kinect es capaz de discernir por estimación el plano de suelo.

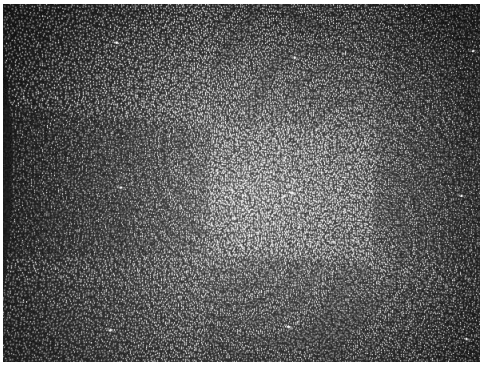

Figura 25 Matriz pseudoaleatoria emitida por Kinect

[6] http://msdn.microsoft.com/en-us/library/hh855347.aspx
[7] http://msdn.microsoft.com/en-us/library/hh855347.aspx

Kinect también incluye un motor en su base que permite modificar vía software la inclinación del sensor para poder apuntar mejor a la zona de interés. Se incluye un acelerómetro que permite estimar la inclinación y, junto al motor, equilibrar de forma automática el dispositivo.

B. 3. Características acústicas de Kinect y procesado de señal de audio

Aunque visualmente es un dispositivo muy potente para su coste de mercado (gracias a la fabricación en escala), desde el punto de vista del sonido, Microsoft Kinect incluye importantes avances en tratamiento de audio [8].

El dispositivo Kinect dispone de cuatro micrófonos supercardiodes en array, 3 de ellos en un extremo de la carcasa y otro en el extremo contrario del frontal del dispositivo. Tanto la disposición de los micrófonos como la caja acústica en la que se encuentran han sido optimizados para mejorar sus parámetros de directividad de acuerdo a las necesidades del dispositivo. Cada uno de estos micrófonos tiene asociado un conversor analógico-digital de 24bits para tener una resolución lo suficientemente elevada.

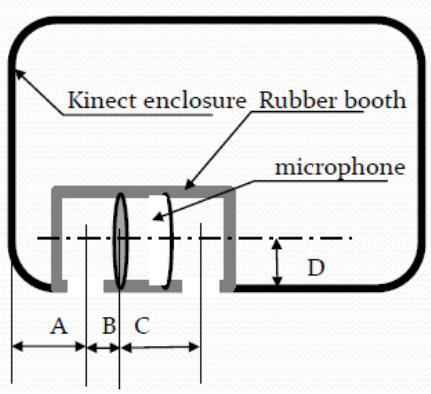

Figura 26 Caja acústica de los micrófonos de Kinect [8]

Gracias a esta arquitectura acústica y al procesado digital que realiza a través del sistema de audio de Windows, Kinect es capaz de ofrecer diferentes mejoras en la captura de sonido con el objetivo de mejorar la inteligibilidad de los locutores para el posterior reconocimiento automático del habla. Las principales innovaciones de Microsoft Kinect a la hora de implementar NUIs son las siguientes:

B.3.1. Acoustic Echo Reduction

Al ser un dispositivo básicamente diseñado para el ocio y el entretenimiento, Kinect debe enfrentarse al ruido generado por los sonidos reproducidos por la aplicación que se está ejecutando. Además, por la disposición típica de altavoces y Kinect (altavoces y sensor en el plano de la pantalla) se encuentra mucho más cerca de la fuente de ruido que del locutor. Esto hace obligatoria la utilización de supresión activa de ruido, englobándose en este concepto algoritmos de cancelación activa de ruido y supresión activa de ruido. Ésta se realiza de forma adaptativa, al conocer el sonido que se está enviando a los altavoces, como se puede ver en la Figura 27.

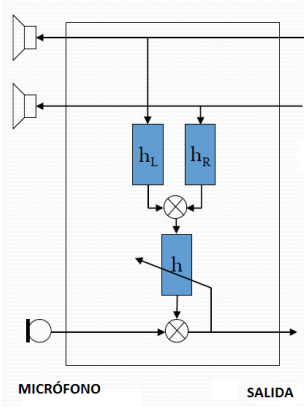

Figura 27 Esquema de AEC estéreo de Kinect [8]

A través de esta disposición, Kinect calibra los filtros de mezcla para los canales L y R y utiliza un filtro adaptativo conjunto para la cancelación del sonido reproducido. Kinect es así capaz de realizar reducción acústica de eco para sonido estéreo.

A través del SDK es posible activar o desactivar este procesado AEC.

B.3.2. Noise Suppression y Automatic Gain Control

Además de la AEC, Kinect introduce algoritmos de eliminación de ruido de forma estadística y un sistema variable de ganancia de los micrófonos para adaptar dinámicamente señales de voz de baja presión sonora.

B.3.3. Beamforming

Uno de los puntos más interesantes en el tratamiento de audio de Kinect, es la capacidad de formar haces de captura (Beamforming) de sonido aumentando la directividad del array de 4 micrófonos situado en el frontal del dispositivo. Para ello, Kinect crea un haz móvil (Beamsteering) que busca el máximo de intensidad de la señal de voz recibida, obteniendo una ganancia de directividad máxima y reduciendo el ruido de interferencia procedente de otras direcciones.

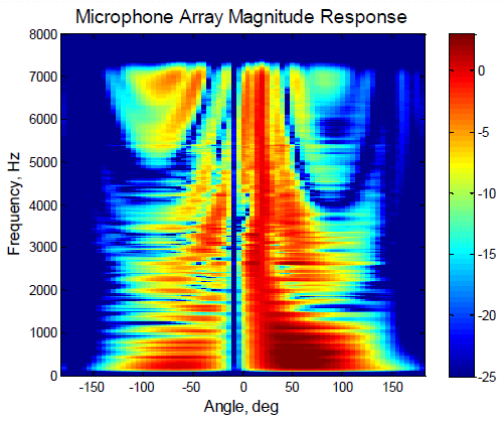

Figura 28 Respuesta en magnitud del array de micrófonos con Beamsteering [8]

Según las especificaciones de Microsoft, este Beamforming tiene un ángulo de cobertura de 100° (-50° - 50°) con una precisión de 10° para un total de 10 regiones de formación de haz.

A través del SDK es posible activar o desactivar el beamforming, funcionando así los micrófonos individualmente y ofreciendo la posibilidad de grabar las cuatro señales de audio de forma independiente.

B.4. Entornos de desarrollo de Microsoft Kinect (Kinect SDK y OPENNI)

La comunicación del dispositivo Microsoft Kinect se realiza a través de un puerto USB 2.0 y con diferentes librerías incluidas en los distintos kits de desarrollo.

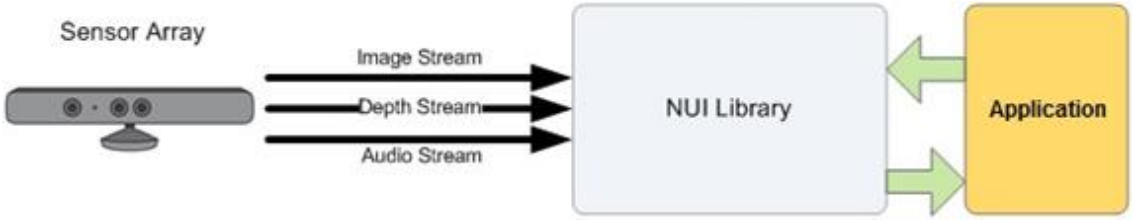

Figura 29 Interacciones HW/SW de Kinect[8]

El éxito del lanzamiento, su bajo coste y sus características avanzadas de captura de información promovieron la aparición de diferentes drivers y SDK desarrollados de forma independiente y ajena a Microsoft. Finalmente, la compañía de Redmond, consciente de la importancia de alimentar a una gran comunidad de desarrolladores de software y servicios, así como del entorno académico y científico, hizo público un SDK propietario pero de libre uso sin ánimo de lucro para el desarrollo de aplicaciones en PC utilizando Kinect.

Dos son los principales entornos disponibles en el mercado: Kinect for Windows SDK, desarrollado por Microsoft y OpenNI[9], software libre y mantenido por la comunidad de desarrolladores. Las principales características y diferencias entre ambos sistemas se pueden encontrar en la Tabla 7.

Tabla 7 Características de las diferentes SDK

	Kinect SDK	OpenNI
Número de articulaciones	20	20
Lag	Ms	ms
Detección de articulaciones	Más preciso y rápido	Más robusto a falsos positivos
Flujo de video	Hasta 1024x768	Hasta 800x600
Audio	Si	No
Dependencia de plataforma	Sólo Windows 7 o más reciente	Windows, Linux, OS X
Soporte de máquinas virtuales	Si	-
Modo de licencia	Licencia gratuita sin ánimo de lucro. De pago para aplicaciones comerciales.	Libre

[8] http://msdn.microsoft.com/en-us/library/jj131023.aspx
[9] http://www.openni.org/

Lenguajes (o wrappers)	C++, C#, Visual Basic	C++, C#, Python, C synchronous, Java, Javascript
Otros	Reconocimiento de dedos	Reconocimiento de dedos

La principal diferencia a efectos de esta TFM es que el SDK desarrollado por Microsoft incluye una API (Application Programming Interface) para toda la parte de audio, frente a OpenNI, que pese a tener otras ventajas comparativas, deja de lado esta importante parte del dispositivo para centrarse en la parte visual.

Tabla 8 Requisitos técnicos de la plataforma Kinect for Windows SDK

Requisitos técnicos de la plataforma con Windows SDK
• **Microsoft Windows 7, Microsoft Windows Embedded Standard 7 o Windows 8**
• **Procesador 32 bit (x86) or 64 bit (x64)**
• **Procesador Dual-core 2.66-GHz o superior**
• **Bus USB 2.0 dedicado (limitación de ancho de banda)**
• **2 GB RAM**
• **Visual Studio 2010**
• **.NET Framework 4.0**
• **Kinect for Windows SDK**
• **Opcionalmente:**
○ **DirectX SDK**

Desde el lanzamiento del primer SDK de Microsoft, diferentes versiones del mismo han ido apareciendo espaciados en el tiempo, incluyendo nuevos ejemplos de desarrollo y mejoras y ajustes.

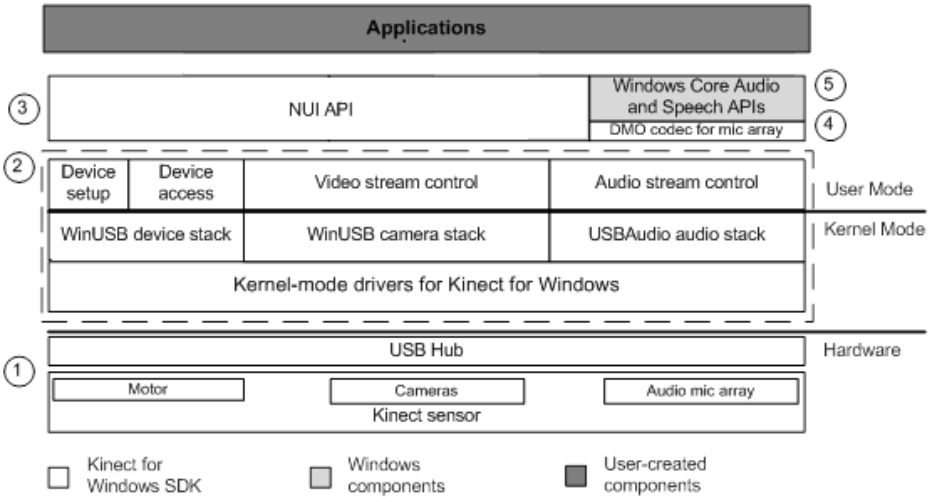

Figura 30 Arquitectura del SDK[10]

A su vez, nuevas opciones y posibilidades se han ido introduciendo paulatinamente, incluyendo detección de dedos, de rasgos faciales así como diferentes toolkits para testear las

[10] http://msdn.microsoft.com/en-us/library/jj131023.aspx

aplicaciones desarrolladas, sobre todo tras la aparición de Kinect for Windows, al incluir esta evolución del dispositivo el modo de trabajo en campo cercano.

En la realización de esta TFM se ha utilizado el dispositivo Kinect original para consola XBOX360, por su menor precio, facilidad de adquisición y con el objetivo final de caracterizar su respuesta acústica usándolo con un reconocedor automático del habla con palabras aisladas, con la finalidad de evaluar el efecto del procesado en la cadena de audio a efectos de reconocimiento.

ANEXO C: FRASES DE ENTRENAMIENTO Y TEST

Tanto las frases utilizadas para entrenar los diferentes modelos mediante técnica de adaptación MAP, como las palabras de test grabadas para hacer el análisis de resultados, han sido elegidas siguiendo las indicaciones dadas por la base de datos de la lengua castellana reflejado en el proyecto Albayzin [10], manteniendo los criterios de balanceo correcto de los tipos y apariciones de fonemas de acuerdo a su presencia estadística en el idioma español.

Tabla 9 Frases de entrenamiento

1	Francia, Suiza y Hungría ya hicieron causa común
2	mi primer profesor de lengua fue López García
3	Guillermo y Yolanda practicaban ciclismo con Jaime
4	el primero en Guipúzcoa y el segundo en Valladolid
5	fue aquel hombre tan gordo el que se acercó en globo
6	éramos un grupo de profesores de lengua y literatura
7	yo no recuerdo en mi pueblo ningún caso de triquinosis
8	éramos un grupo de gente bastante distinguida
9	después ya se hizo muy amiga nuestra
10	los achaques de Jesús remitieron sin causar disgustos
11	su viuda se casó con un profesor de inglés de allá
12	durante tus estudios has hecho algún viaje
13	dos años y medio en Vilanova ya fueron suficientes
14	pero en el fondo le tengo cariño
15	puede haber deficiencia de agua en Logroño
16	la sangre se revuelve con estas migas de pan blanco
17	los yernos de Ismael no engordaran los pollos con hierba
18	mi mujer es profesora de lengua y literatura
19	sabia un poquito de inglés pero muy poco
20	en julio hacíamos en Burgos otra vez la preparación
21	un niño muy rico que se llama Ignacio
22	no en lenguaje hablado pero si en lenguaje escrito
23	era muy gordo, muy gordo y con un tupé inmenso
24	ellos ya desde un principio se quedan con el dinero
25	después de la mili ya me vine a Cataluña
26	estuve en Guernica dando clase de lengua y literatura
27	firmaban como cántabros incluso en tumbas funerarias
28	aunque ellos ya engrasaron los ejes como yo les enseñé
29	habla un poco de su prepotencia y de su orgullo
30	trabajaba en Granollers y vivía en Barcelona
31	es uno de los recuerdos más bonitos que guardo
32	vimos que el ambiente había cambiado mucho
33	bajamos un día al mercadillo de Palma
34	por las noches organizaban bailes y fiestas
35	yo entré en filas a cumplir el servicio militar

36	uno llevaba el chorizo el otro llevaba el gazpacho
37	con Gema y con Blanca me veo por las noches en casa
38	subimos un rato al apartamento y luego ya nos fuimos
39	le voy a contar por qué he estudiado lengua
40	llevas quince años fuera de la montaña
41	existe un viento del norte que es un viento frio
42	nos dio el disgusto más grande de nuestra vida
43	se hacen chorizos y salchichones de dos clases
44	antes de primero de bachiller ya te traumatizaban
45	la profesora que tengo tampoco es muy agradable
46	yo tengo derecho a que a mí se me entienda
47	al segundo día empezaron a llegar colegios
48	en estos pueblos preciosos de Vizcaya y Guipúzcoa
49	me he tomado un café con leche en un bar
50	dentro de muy poco pues va a estar la mitad cubierto

Tabla 10 Palabras de test

1	árbol
2	bruja
3	campana
4	casa
5	cuchara
6	ducha
7	flan
8	fuma
9	globo
10	grifo
11	jarra
12	lápiz
13	llave
14	mariposa
15	niño
16	pala
17	pan
18	periódico
19	piano
20	piña
21	plátano
22	preso
23	puerta
24	semáforo

25	sol
26	taza
27	toalla
28	tortuga
29	zapato
30	dos ocho cuatro seis uno cero tres cinco siete nueve
31	cero cero uno uno dos siete
32	ocho cuatro ocho ocho seis dos ocho cinco uno cuatro uno uno tres cero siete siete
33	cero nueve cinco cero nueve seis
34	tres nueve cero siete cinco dos uno ocho seis cuatro
35	cero cero tres cero tres nueve
36	nueve cinco uno nueve nueve uno uno cinco cero ocho uno cero cero cero uno seis
37	uno seis uno seis dos uno
38	cuatro tres cero seis nueve ocho siete uno dos cinco
39	cero cero tres uno cuatro tres
40	nueve cinco ocho siete dos dos ocho dos ocho cuatro cero seis uno cero seis cinco
41	ocho nueve ocho ocho cero cuatro
42	cuatro dos cinco ocho nueve seis tres cero uno siete
43	cero cero cuatro cero cuatro dos
44	tres cuatro seis tres dos seis uno dos seis uno nueve cinco cero dos ocho tres
45	cero siete seis cero siete siete
46	seis uno cuatro nueve cinco dos cero siete tres ocho
47	cero cero cuatro uno cinco seis
48	cuatro cuatro ocho cuatro siete uno nueve seis uno dos uno uno nueve cero dos seis
49	tres cinco cinco tres cinco seis
50	nueve cinco dos ocho uno cuatro tres seis siete cero
51	cero cero siete cero siete uno
52	dos cinco cuatro dos ocho tres nueve siete seis seis tres cuatro cinco uno tres ocho
53	dos dos cuatro dos uno uno
54	cero uno dos seis ocho nueve siete cinco cuatro tres

ANEXO D: FRAMEWORK DE SOFTWARE

D.1. Introducción

El uso de C++ como lenguaje de programación [9] frente a C# se decidió como una de los requisitos de la TFM con el fin de garantizar máxima compatibilidad con las aplicaciones previamente realizadas por el Departamento de Tecnologías de Voz y facilitar su adaptación al sensor Kinect. Microsoft, dentro de una política de promoción y aceptación de C# ofrece una mayor cantidad de recursos (tutoriales, ejemplos, videos, soporte técnico, foros) para este lenguaje que para C++. De esta forma hace una clara diferenciación entre los desarrolladores no profesionales o casuales, y los desarrolladores científicos o core, a los que deja más libertad de trabajo a cambio de menor soporte. En esta TFM no se han utilizado Windows Forms, realizando todo el trabajo de programación al nivel más bajo posible con Kinect SDK.

El software se puede dividir en diferentes módulos de acuerdo a las funcionalidades que en él se pueden encontrar y dependiendo de la naturaleza de la información capturada, así como de la configuración de los dispositivos.

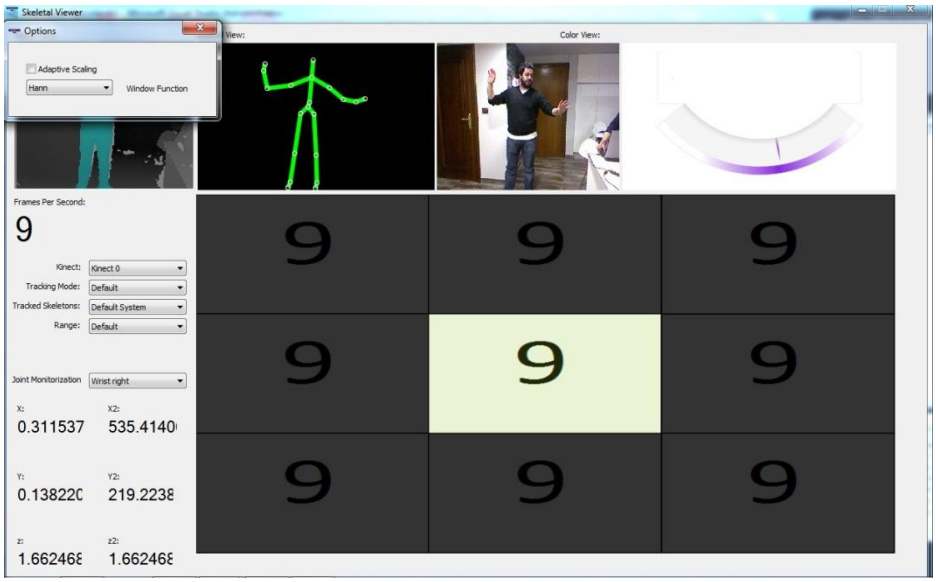

Figura 31 Imagen de Total Explorer en funcionamiento

D.2. Aplicaciones

Total Explorer

Para la aplicación pedagógica se ha diseñado la plataforma de un minijuego que hace un uso básico de las capacidades de captura de movimientos y audio de Kinect para la creación de interfaces NUI.

Se trata de la base de un minijuego basado en un tablero sobre el que se selecciona con gestos una casilla que presenta un pictograma. Una vez seleccionado se espera a que el usuario pronuncie la palabra correspondiente al pictograma.

Este juego básico se basa en un tablero con un número variable de celdas. Cada una de estas celdas posee un pictograma. Cuando el usuario de la aplicación selecciona una de estas celdas tiene un tiempo limitado para pronunciar la palabra asociada al pictograma. Dependiendo del éxito o fallo en la palabra se producirá una acción correctora o se premiará el éxito.

A partir de esta sencilla asociación de tablero y celdas se pueden desarrollar diferentes modos de juego que van más allá de la correcta identificación y pronunciación de la palabra, como pueden ser buscar parejas de pictogramas, elegir secuencias de palabras asociadas, escuchar la palabra y elegir la celda correcta etc…

Voice Sampler

Esta aplicación permite seleccionar los modos de funcionamiento de la arquitectura audio de Kinect y facilita la labor de grabación de las muestras de voz para su posterior uso en el reconocedor de palabras aisladas.

El diseño de esta aplicación se podría haber centrado únicamente en la parte de audio, eliminando la captura y tratamiento de los flujos de imagen. Sin embargo, al ser estos flujos los más exigentes desde un punto de vista de ancho de banda y de tratamiento, se optó por dejarlos funcionando a la vez que se capturaban las muestras para poder tener una estimación más realista de lo que serían las condiciones de funcionamiento en una aplicación de Kinect para PC.

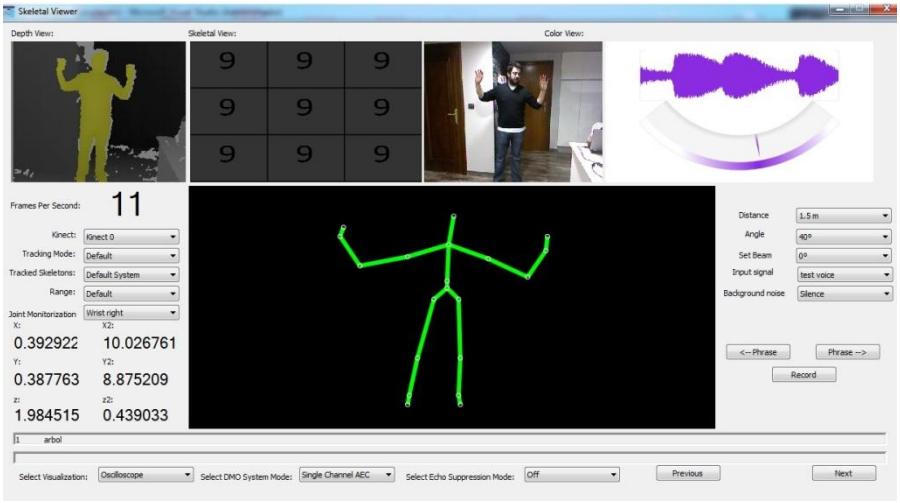

Figura 32 Imagen de Voice Sampler en funcionamiento

También por ello se tomó como PC de captura de datos un ordenador portátil ligeramente por debajo de las especificaciones originales de Microsoft. Después de realizar diferentes pruebas se pudo determinar que este PC portátil era sobradamente capaz de cumplir con las especificaciones de captura de datos y no sufrir pérdidas de framerate en la captura y tratamiento de esqueletos. Esto es debido a que desde el principio se optó por el desarrollo de la parte gráfica utilizando las librerías Direct2D incluidas en DirectX, que mejoran el rendimiento del tratamiento y representación de imágenes y videos en Windows al hacer uso de la GPU de la tarjeta gráfica, liberando recursos en la CPU. No fue hasta la versión 1.5 que Microsoft incluyo Direct2D de forma nativa en Kinect SDK.

D.3. Inicialización

En esta fase se crea la ventana de Windows en la que correrá la aplicación y se inicializan todas las variables de las diferentes instancias de las clases que se van a utilizar. Así, se crea y se inicializa un único Factory Direct2D para todos los elementos de visualización de la aplicación. Respecto al juego, se inicializa el tablero de juego junto con las diferentes casillas de acuerdo al modo de juego seleccionado.

En el caso del dispositivo Kinect, se inicializa detectando el número de Kinects conectadas, eligiendo la primera visible por el sistema operativo como predeterminada y asignando el setup inicial de audio y video, creando los correspondientes eventos y callbacks asociados. Una vez chequeado el correcto funcionamiento del dispositivo, se crean los flujos de audio y video, se devuelve el control al usuario y se comienza la captura de los distintos eventos relacionados con la Kinect y propios de la interfaz gráfica.

D.4. Eventos: Captura de esqueletos, RGB y profundidad

El SDK de Kinect permite la obtención de los distintos flujos de información por encuesta o por evento. Para el desarrollo de esta aplicación se ha elegido la opción de tratamiento de la información por evento, al ser más eficiente, consumir menos recursos y permitir una integración más fácil con posteriores desarrollos. Existen cuatro eventos relacionados con la información proveniente de Kinect:

1) Frame RGB capturado: Kinect avisa al sistema operativo de que un frame RGB se encuentra disponible para su tratamiento. La aplicación captura este frame, lo procesa si es necesario y lo muestra por pantalla renderizando la información en superficie target asignada en la GUI adaptando sus dimensiones si no son nativas.
2) Frame de profundidad capturado: Kinect avisa de que tiene un frame de profundidad disponible para ser capturado. A efectos prácticos, un frame de profundidad es como un frame RGB, solo que en lugar de tener la información de color por cada pixel poseemos la información de profundidad en el eje Z. Antes de su representación se hace un truncado de bits de 13 a 8 para representar con esa resolución, pese a capturarse con una más alta. Finalmente, el frame modificado es renderizado en la interfaz gráfica.
3) Captura de esqueleto: Kinect avisa de que tiene un objeto esqueleto capturado y listo para ser procesado. El objeto esqueleto posee la información de cada una de las 20 articulaciones (en el caso de jugadores de pie). Para cada articulación tenemos la información de su posición XYZ, así como si la posición ha sido medida o ha sido estimada por no tener visibilidad directa de la misma (por objetos u otras partes del cuerpo). A este esqueleto se le aplica un proceso de suavizado de la información para evitar jitter en la imagen y obtener secuencias de movimientos más suaves para el reconocimiento de gestos[11]. En esta fase ya tenemos toda la información necesaria del esqueleto para poder tratarla y, si fuera necesario, presentarla por pantalla renderizando de nuevo a través de Direct2D. Kinect SDK incluye opciones para obtener

[11] http://msdn.microsoft.com/en-us/library/jj131429.aspx

información de la orientación y rotación de los huesos entre articulaciones tomando como referencia la posición de la articulación origen y destino.

D.5. Captura de voz y creación del archivo de muestra

El cuarto evento implementado en la aplicación Voice Sampler es el relacionado con el audio. Kinect avisa de que posee un frame de audio en espera de ser capturado del buffer y procesado. Según el modo de inicialización del sistema de audio, a través del DMO de DirectX o a través de la API de sonido de Windows7 se tienen distintas posibilidades. Si se hace a través del API de audio de Windows 7, se pueden obtener por separado las capturas de audio de los 4 micrófonos sin procesar. Si se utiliza el DMO de DirectX, modo elegido en esta TFM, se tiene acceso a un único flujo de audio con el procesado de señal deseado (y previamente indicado en la Kinect). En este segundo caso, si se ha elegido la opción Optibeam que genera automáticamente beamsteering hacia el origen de la voz, el objeto de la clase Kinect instanciado poseerá información relativa a:

1) Ángulo del haz creado (-50° a 50°, en pasos de 10°)
2) Ángulo de origen del sonido: Ángulo en ° de origen del sonido
3) Nivel de confianza en la estimación de la posición de origen del audio (muy interesante para saber si es un dato de fiar o no).

Si la opción de guardar en archivo está activada, se genera un archivo .wav donde se van almacenando las muestras de audio ya procesadas. Este archivo se genera al comienzo de la grabación y se nombra de acuerdo a una nomenclatura fija.

D.6. Nomenclatura de las muestras

Los archivos wav creados para almacenar las muestras de voz capturadas se nombran de acuerdo a una nomenclatura fija que permita su manipulación posterior de una forma clara y fácilmente ordenable desde el punto de vista del sistema operativo. Este nombre de archivo refleja los diferentes parámetros de captura y tratamiento de señal así como el locutor, la posición y las condiciones ambientales en las que ésta se produjo:

NOMBRE-Tipodemedida_Phrase-xxx_Number-xxx_Total-xxx_Distance-x_Angle-x_MIC-x_AEC-x_BEAM-x_INPUT-x_BG-x_hh-mm-ss.wav

- NOMBRE: Nombre del locutor
- Tipodemedida: Descripción específica (Test/Train)
- Phrase: nº de frase/palabra de la lista de entrenamiento/test
- Total: nº de la medida en la sesión de grabación
- Distance: Distancia al micrófono según la perpendicular al eje del array

Código	0	1	2
Valor	1.5m	2.5m	3m

- Angle: Ángulo respecto de la perpendicular al eje del array

Código	0	1	2	3	4	5	6	7	8	9	10
Valor	-50°	-40°	-30°	-20°	-10°	0°	10°	20°	30°	40°	50°

- MIC: Configuración del array de micrófonos

Código	Valor
0	Single Microphone
1	Single Microphone + AEC
2	Optibeam
3	Optibeam + AEC

- AEC: Echo Suppression Mode (errata en la nomenclatura, en realidad es Noise Suppression)

Código	Valor
0	Off
1	On

- BEAM: Ángulo del BEAM formado por el array si lo hubiera

Código	0	1	2	3	4	5	6	7	8	9	10
Valor	-50°	-40°	-30°	-20°	-10°	0°	10°	20°	30°	40°	50°

- INPUT: Tipo de sonido

Código	0	1	2	3
Valor	Voice	AWGN	Tone	Chirp

- BG: Ruido de fondo

Código	Valor
0	Off
1	On

- hh-mm-ss: Hora de la creación del archivo

D.7. Representación visual: Direct2D

Para representar los diferentes elementos gráficos de la aplicación que no sean elementos UI de Windows (flujos de video RGB, profundidad, esqueletos y tablero de juego), se ha utilizado el sistema Direct2D[12] incluido en DirectX[13].

Direct2D permite dibujar elementos variables de una forma muy eficiente ya que hace uso de las capacidades de la tarjeta gráfica liberando ciclos de CPU que pueden ser utilizados para el procesado de los inputs o el control de diferentes parámetros del juego. De esta forma se

[12] http://msdn.microsoft.com/es-es/library/windows/desktop/dd370990(v=vs.85).aspx
[13] http://windows.microsoft.com/es-ES/windows7/products/features/directx-11

obtiene el máximo de un sistema como Kinect que es muy exigente computacionalmente. Direct2D se utiliza generando una Factory encargada de gestionar los targets donde se dibujarán las imágenes así como de las diferentes herramientas que se generen (texturas, pinceles, degradados de imagen, texto). Sólo se debe instanciar un único Factory para toda la aplicación, pasándolo por referencia a todas aquellas instancias de clases que hagan uso de ella. Al reservar recursos hardware de la tarjeta gráfica es imprescindible observar con atención la destrucción del recurso al hacer uso del destructor de las clases y de la aplicación, para ello, solamente la última instancia debe liberarlo durante su destrucción.

D.8. Módulo de juego – Tablero

El objetivo de este módulo es crear la base sobre la que desarrollar ésta y otras aplicaciones pedagógicas y puede ser útil tanto para reforzar el aprendizaje del usuario como para poder capturar diferentes palabras que posteriormente pueden ser utilizadas para entrenar modelos personalizados para el grado de discapacidad en el habla o mejorar sistemas de adaptación supervisada, en el caso particular de niños con problemas de aprendizaje o expresión oral.

En este caso, se han creado dos clases: Board y Square.

- Clase CBoard: En esta clase se define un tablero, así como las normas del juego, y se encarga la gestión de las celdas.
- Clase CSquare: En esta clase se encuentra el pictograma a representar, el temporizador para responder, la palabra a reconocer o reproducir, y las herramientas de dibujo Direct2D propias de la celda.

D.9. Opciones de trabajo, visualización y grabación

La aplicación posee diferentes menús desplegables para la selección de diferentes opciones de setup de la Kinect que pueden cambiarse online, así como la selección de la información capturada que se desea mostrar por pantalla.

D.9.1. Menús desplegables de modos de trabajo

Tabla 11 Modos de seguimiento de jugadores y Kinect

Menú desplegable	Opciones
Kinect	Selección de la Kinect activa
Tracking Mode	- Default (de pie) - Sentado
Tracked Skeletons	- Default - Jugador más cercano 1 - 2 jugadores más cercanos - Fijar el jugador más cercano - Fijar los dos jugadores más cercanos
Rango	- Default (lejano) - Cercano
Joint Monitorization	Selección de la articulación de la que se mostrarán las coordenadas X Y Z. Las articulaciones son las 20 que es capaz de mostrar de monitorizar Kinect de acuerdo a la Figura 24.

Tabla 12 Modos de trabajo de Kinect en audio

Menú desplegable	Opciones
Select DMO System Mode	- Single Channel - Single Channel AEC - Optibeam Array - Optibeam Array AEC
Select Echo suppression Mode	- Off - On

D.9.2. Opciones de visualización de la señal de audio

Permite seleccionar la visualización en tiempo o en frecuencia del audio capturado por la Kinect, así como seleccionar la ventana elegida para filtrar la respuesta a presentar con sus diferentes características de lóbulo principal y secundarios.

Tabla 13 Opciones de visualización de la señal de audio

Menú desplegable	Opciones
Select Visualization	• FFT • Oscilloscope
Window function	• Rectangular • Hann • Hamming • Nuttal • Blackman-Harris • Blackman-Nuttal

D.9.3. Botones de acción de la aplicación

En Voice Sampler existen botones de acción específicos relacionados con la captura de muestras de sonido para entrenamiento y test.

Tabla 14 Botones de acción para captura de sonido

Nombre	Acción
Record	Procede a grabar el audio capturado y tratado por la Kinect
← Phrase	Presenta la frase/palabra previa del script de frases/test
Phrase →	Presenta la frase/palabra siguiente del script de frases/test
Previous	Cambia a los parámetros definidos en la línea anterior del script de test
Next	Cambia a los parámetros definidos en la línea posterior del script de test

También existen menús desplegables presentando las diferentes opciones de la relación de tablas presentadas en el apartado de Nomenclatura de los archivos de audio.

D.10. Scripts

La aplicación Voice Sampler puede hacer uso de dos archivos txt que permiten automatizar la captura de datos para entrenamiento y test. Para su correcto funcionamiento deben encontrarse en la carpeta en la que se encuentra el ejecutable de la aplicación:

- ***Script.txt***: Cada línea de este archivo de texto contiene una serie de campos separados por espacios con los diferentes parámetros que se han de utilizar para capturar el sonido, así como los parámetros de grabación ambiental. Se puede pasar de una línea a otra utilizando los botones previous y next de la GUI.

Tabla 15 Campos de configuración del script de automatización de medidas

Locutor	Distancia	Ángulo	Modo de array	Modo de supresión de eco	Reservado	Input Signal	Background Noise
Nombre	Selección	grados	Selección	Selección	No usado	No usado	No usado

- ***test.txt:*** Cada línea de este archivo de texto tiene cada palabra/frase de test que se ha de decir. Se puede pasar de una línea a otra utilizando los botones ←Phrase y Phrase→ de la GUI siendo presentada cada frase por pantalla.

D.11. Software y Hardware utilizado en su desarrollo

- Ordenador sobremesa Intel i7 Quad Core 2.4Ghz 4Mb RAM (Desarrollo)
- Ordenador portátil Intel i5 Dual Core 2.4GHz 4Mb RAM (Captura)
- Windows 7 Professional 64 Bit
- Kinect for XBOX 360
- Visual Studio 2010
- DirectX SDK
- Windows SDK
- Kinect for Windows SDK v1.5

ANEXO E: RESULTADOS Y VALIDACIÓN

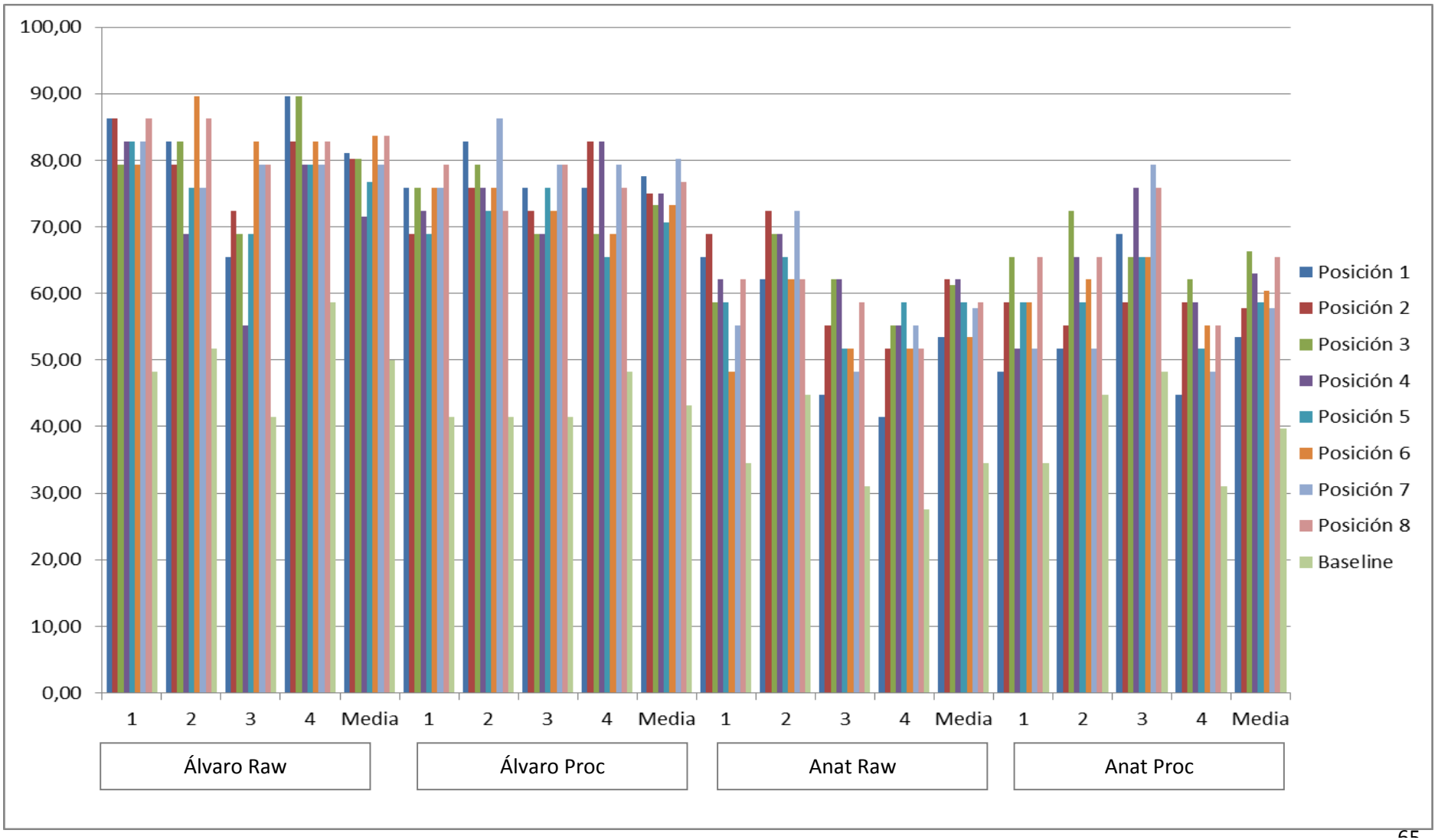

Figura 33 Tasa de aciertos de todos los modelos entrenados y testeados en las cuatro posiciones de referencia

Tabla 16 Tasa de aciertos para cada modelo y test

		BASELINE	1	2	3	4	5	6	7	8	Media	
Álvaro Raw	1	48,28	86,21	86,21	79,31	82,76	82,76	79,31	82,76	86,21	83,19	POSICIONES DE TEST
	2	51,72	82,76	79,31	82,76	68,97	75,86	89,66	75,86	86,21	80,17	
	3	41,38	65,52	72,41	68,97	55,17	68,97	82,76	79,31	79,31	71,55	
	4	58,62	89,66	82,76	89,66	79,31	79,31	82,76	79,31	82,76	83,19	
	Media	**50,00**	**81,03**	**80,17**	**80,17**	**71,55**	**76,72**	**83,62**	**79,31**	**83,62**	**79,53**	
Álvaro Proc	1	41,38	75,86	68,97	75,86	72,41	68,97	75,86	75,86	79,31	74,14	
	2	41,38	82,76	75,86	79,31	75,86	72,41	75,86	86,21	72,41	77,59	
	3	41,38	75,86	72,41	68,97	68,97	75,86	72,41	79,31	79,31	74,14	
	4	48,28	75,86	82,76	68,97	82,76	65,52	68,97	79,31	75,86	75,00	
	Media	**43,10**	**77,59**	**75,00**	**73,28**	**75,00**	**70,69**	**73,28**	**80,17**	**76,72**	**75,22**	
Anat Raw	1	34,48	65,52	68,97	58,62	62,07	58,62	48,28	55,17	62,07	59,91	
	2	44,83	62,07	72,41	68,97	68,97	65,52	62,07	72,41	62,07	66,81	
	3	31,03	44,83	55,17	62,07	62,07	51,72	51,72	48,28	58,62	54,31	
	4	27,59	41,38	51,72	55,17	55,17	58,62	51,72	55,17	51,72	52,59	
	Media	**34,48**	**53,45**	**62,07**	**61,21**	**62,07**	**58,62**	**53,45**	**57,76**	**58,62**	**58,41**	
Anat Proc	1	34,48	48,28	58,62	65,52	51,72	58,62	58,62	51,72	65,52	57,33	
	2	44,83	51,72	55,17	72,41	65,52	58,62	62,07	51,72	65,52	60,34	
	3	48,28	68,97	58,62	65,52	75,86	65,52	65,52	79,31	75,86	69,40	
	4	31,03	44,83	58,62	62,07	58,62	51,72	55,17	48,28	55,17	54,31	
	Media	**39,66**	**53,45**	**57,76**	**66,38**	**62,93**	**58,62**	**60,34**	**57,76**	**65,52**	**60,34**	
		POSICIONES DE CADA MODELO										

ANEXO F: SOFTWARE UTILIZADO PARA EL DESARROLLO DEL RAH

HTK Hidden Markov Model Toolkit

HTK[14] es un toolkit portable a diferentes plataformas, diseñado para construir y manipular modelos ocultos de Markov. Se trata de una plataforma muy versátil y potente, cuya utilización, aunque inicialmente pensada para investigación en reconocimiento del habla, ha transcendido a otras áreas como el reconocimiento de caracteres o la secuenciación de estructuras de ADN, al ser aplicables los modelos ocultos de Markov.

HTK contiene una serie de librerías y herramientas, con acceso al código fuente en C. Estas herramientas permiten la creación, entrenamiento, tratamiento y testeo de HMM y de las gaussianas relacionadas en cada estado. Además, HTK contiene una profusa documentación con la teoría que describe los modelos ocultos de Markov y una orientación hacía la aplicación práctica de ésta.

Desarrollado originalmente por la Universidad de Cambridge, sus derechos de explotación y distribución fueron transferidos a una empresa privada que posteriormente sería absorbida por Microsoft. Actualmente, aunque Microsoft mantiene los derechos sobre HTK, estos han sido licenciados de nuevo a la Universidad de Cambridge permitiendo un modelo colaborativo a la hora de introducir mejoras en el código fuente que pueden ser incluidos en las consiguientes revisiones del toolkit.

Mathworks MATLAB

Matlab[15] es un entorno de desarrollo matemático con su propio lenguaje de programación, el lenguaje M. Desarrollado por Mathworks, Matlab incluye las herramientas necesarias para poder trabajar de una forma intuitiva, flexible y potente en diversas disciplinas técnicas y científicas (tratamiento de señal y comunicaciones, imagen y video, sistemas de control, biología…) a través de diferentes funciones matemáticas ya compiladas que pueden ser usadas para crear nuevas funciones a partir de ellas y extender el alcance de los objetivos de cada proyecto.

Esta flexibilidad, recomienda su utilización allí donde otras soluciones comerciales (hojas de cálculo,…) o lenguajes de programación, exigen un esfuerzo de tiempo y recursos elevados al no estar específicamente diseñados para los requerimientos de una determinada problemática.

Matlab tiene una gran penetración tanto dentro del mundo académico como en el industrial, siendo referente en multitud de áreas técnicas. Se trata de un software propietario compuesto de una funcionalidad básica y diferentes toolbox que pueden ser adquiridos comercialmente. Sin embargo, la gran distribución de este software, ha generado una importante comunidad a su alrededor, con multitud de contribuciones libres en muy diversas disciplinas.

[14] http://htk.eng.cam.ac.uk/]
[15] http://www.mathworks.es/products/matlab/]

En la elaboración de esta TFM se ha utilizado el ISOMAP Toolkit de reducción no lineal de dimensiones de la universidad de Stanford[16]

También se pueden encontrar diferentes alternativas a Matlab en el mercado, como Mathematica, o de software libre (OCTAVE), pero no tienen la funcionalidad y la versatilidad del original o su facilidad de uso.

[16] http://isomap.stanford.edu/

ANEXO G: FUNCIONES IMPLEMENTADAS EN MATLAB

G.1. Funciones de entrenamiento

Esta fase se encarga de entrenar los modelos simples para realizar la adaptación MAP a cada uno de los locutores y las condiciones de grabación.

1) train_model.m

Este script se encarga de entrenar cada modelo simple de acuerdo a las frases de entrenamiento de cada posición, a través de una adaptación supervisada MAP dado el propio modelo simple, un string con cada frase utilizada en la adaptación y el path de los archivos .mfc correspondiente a dichos textos. Como salida tenemos el modelo adaptado para la posición, el locutor y el procesado de audio de Kinect correspondiente, que es salvado en un archivo .mat de acuerdo a la nomenclatura autoexplicativa correspondiente.

2) Get_Models_Script.m

Este script tiene dos objetivos fundamentales:

1) Obtener los archivos .list con los paths de los archivos .mfc que se van a utilizar en el entrenamiento MAP de los modelos simples. Para ello se le indica un path inicial en el que comenzar a buscar los diferentes directorios donde se encuentran los audios procesados de entrenamiento. Como resultado se obtiene un archivo .list para cada posición, locutor y procesado, y nombrado de acuerdo a estas características.
2) Preparar el entorno y las variables para ejecutar train_model.m y obtener el modelo entrenado.

Al finalizar este script obtenemos todos los modelos entrenados mediante MAP para los cuales tenemos datos de entrenamiento suficientes. Además, estos modelos se guardan en archivos .mat de Matlab con una nomenclatura correcta.

G.2. Funciones de reconocimiento

En esta fase se produce el reconocimiento aislado de palabras propiamente dicho y la preparación de diferentes structs con datos útiles y accesibles para la obtención de resultados estadísticos y visuales en la siguiente fase. De nuevo, se incide en una organización y programación orientada a poder automatizar grandes cantidades de datos, facilitando la futura escalabilidad a nuevos locutores o condiciones de grabación.

1) List_Tests_Script.m

Este script se encarga de crear la estructura st_tests y de introducir la información para cada una de las variables y para cada uno de los tests a realizar. Este proceso se realiza en 2 pasos:

a. Enumerar los diferentes tests grabados buscando en el árbol de directorios desde un path indicado. De esta forma, buscando por los directorios y haciendo uso de la nomenclatura de los archivos se pueden determinar por un lado los valores de locutor, distancia, grados y procesado. A su vez, genera un

archivo .list con los paths de los archivos .mfc a utilizar en el reconocedor de palabras.

b. Completar la información de st_tests para cada cada modelo y test, incluyendo asimismo el nombre del archivo .list a utilizar.

Tras ejecutar este test se obtiene un struct con todos posibles tests para cada locutor, posición, distancia, grado y procesado, así como los diferentes parámetros de cada una de ellas fácilmente accesibles, dimensionados e inicializados.

2) List_models.m

Este script realiza un trabajo similar a List_Tests_Script pero en lugar de tests, se encarga de recopilar todos los modelos entrenados que se encuentran en el directorio raíz y actualiza sus valores internos para cada elemento del struct st_models. Hay que destacar que en la variable model_db, se almacena el modelo entrenado para luego poder trabajar directamente con él. Además, a cada uno de los modelos del struct se le asocia una estructura st_tests con todos los posibles tests y donde se almacenarán los resultados del reconocedor.

3) run_test.m

Esta función es la encargada de evaluar la log-verosimilitud entre las palabras dichas y las palabras existentes en el diccionario del reconocedor.

- Parámetros de entrada: st_models, st_tests, i (índice del modelo a evaluar), j (índice del test a evaluar).
- Salida: modelo actualizado con los resultados de log-verosimilitud entre todas las palabras dichas en el test y todas las palabras del diccionario del reconocedor. Esta log-verosimilitud se almacena en una matriz guardada en la variable m_LH y que presenta los valores para cada pareja palabra dicha – palabra del diccionario.

4) Script_todo.m

Este script evalúa 1) List_Tests_Script.m, 2) List_models.m y, finalmente, llama a run_test.m para los conjuntos modelo-test de interés. Al tratarse de un proceso que puede llegar a ser muy intensivo computacionalmente guarda st_models en un archivo .mat de Matlab al finalizar el proceso.

G.2. Funciones de resultados y representaciones

1) GetFigure.m

Este script se encarga de generar una matriz de resultados de los diferentes modelos y tests de cada uno de ellos de acuerdo a una serie de parámetros de búsqueda. El objetivo es facilitar la obtención de resultados a la hora de evaluar el modelo y el reconocedor de palabras aisladas y preparar las variables para ejecutar otras rutinas más específicas de representación y comparación.

Los parámetros de entrada que se deben determinar para cada modelo y test se pueden encontrar en la Tabla 17:

Tabla 17 Variables de trabajo del script GetFigure.m a determinar para obtener una matriz de resultados

GetFigure.m Opciones		
Variables	**Opciones**	**Explicación**
m_speaker	0) Álvaro 1) Anat 2) Ambos	De entre los modelos, seleccionar los indicados respecto al locutor
t_speaker	0) Álvaro 1) Anat 2) Ambos	De entre los tests, seleccionar los indicados respecto al locutor
m_distance	0) 150cm 1) 250cm 2) 300cm 'Todo'	De entre los modelos, seleccionar los indicados respecto a la distancia al sensor. 'Todo' selecciona todas las opciones.
t_distance	0) 100cm 0) 150cm 1) 250cm 2) 275cm 3) 250+275cm 'Todo'	De entre los tests, seleccionar los indicados respecto a la distancia al sensor. 'Todo' selecciona todas las opciones.
m_degrees	0) 0° 1) 10° 2) 20° 3) 30° 4) 40° 'Todo'	De entre los modelos, seleccionar los indicados respecto al ángulo con el sensor. 'Todo' selecciona todas las opciones.
t_degrees	0) 0° 1) 10° 2) 20° 3) 30° 'Todo'	De entre los tests, seleccionar los indicados respecto al ángulo con el sensor. 'Todo' selecciona todas las opciones.
m_processing	0) Sin procesado 1) procesado 2) Ambos	De entre los modelos, seleccionar los indicados respecto al procesado de audio realizado.
t_processing	0) Sin procesado 1) procesado 2) Ambos	De entre los tests, seleccionar los indicados respecto al procesado de audio realizado.
SP_word	0 - 54 siendo 0 todas las palabras	De entre las palabras de test grabadas, seleccionar las indicadas.
CB_word	0 - 57 siendo 0 todas las palabras	De entre las palabras de test del diccionario, seleccionar las indicadas.

GetFigure genera dos matrices booleanas, M y W, donde se indica qué modelos, qué tests y qué palabras es necesario analizar, sirviendo como entrada para las funciones de evaluación.

2) *Comp_degrees.m*

Función que chequea si el modelo o test debe incluirse en la matriz M de GetFigure de acuerdo al ángulo respecto al dispositivo Kinect y según los valores de los parámetros de entrada.

3) *Comp_distance.m*

Función que chequea si el modelo o test debe incluirse en la matriz M de GetFigure de acuerdo a la distancia respecto al dispositivo Kinect y según los valores de los parámetros de entrada.

4) ***Comp_pos.m***

Función que chequea si el modelo o test debe incluirse en la matriz M de GetFigure de acuerdo a la posición dispositivo Kinect y según los valores de los parámetros de entrada.

5) ***Comp_proc.m***

Función que chequea si el modelo o test debe incluirse en la matriz M de GetFigure de acuerdo al procesado de audio del dispositivo Kinect y según los valores de los parámetros de entrada.

6) ***Funciones de evaluación***

Funciones generadas a partir de diferentes parámetros de GetFigure y que una vez obtenida la matriz de resultados permiten evaluar la tasa de aciertos correspondiente a los datos experimentales. Pueden variar dependiendo de las necesidades específicas de resultados o representación.

- ***tasa_de_aciertos.m*** → Tasa de aciertos para los modelos y tests seleccionados
- ***tasa_de_aciertos_word.m*** → Tasa de aciertos de una palabra particular en los modelos y tests seleccionados.

7) ***Funciones de creación de supervectores***

Un supervector es una matriz en la que en cada columna tenemos concatenadas todas las medias de las diferentes gaussianas utilizadas para modelar las salidas de los HMMs en cada modelo entrenado. Para cada locutor se genera uno de estos supervectores diferenciando entre los modelos con y sin procesado de audio. Se han creado cuatro funciones de este estilo que dan como salida el supervector utilizado en posteriores scripts:

- Svect_Alvaro_raw.m: Supervector con los modelos asociados al locutor Álvaro para cada posición y sin procesado de señal en la captura.
- Svect_Alvaro_proc.m: Supervector con los modelos asociados al locutor Álvaro para cada posición y con procesado de señal en la captura.
- Svect_Anat_raw.m: Supervector con los modelos asociados al locutor Anat para cada posición y sin procesado de señal en la captura.
- Svect_Anat_proc.m: Supervector con los modelos asociados al locutor Anat para cada posición y con procesado de señal en la captura.

8) ***Isomap.m***

En esta función se utilizan los supervectores para calcular el diagrama de vecindad de los diferentes modelos y la variación del error residual con el aumento de la dimensionalidad de los datos.