Low-Cost Speaker and Language Recognition Systems Running on a Raspberry Pi

L. F. D'Haro, *Member, IEEE*, R. de Córdoba, *Member, IEEE*, J. I. Rojo, J. Díez, D. Avendaño, J.M. Bermudo

Abstract— **This paper describes two state-of-the-art and portable voice-based authentication and language recognition systems. While the authentication system allows secure access to a media center at home, the language recognition system can be used as a previous step to automatically transcribe and translate the recognized text from its original language into another one. The most important advantage of the developed systems is that they can run on a low cost embedded device, such as a Raspberry Pi (RPi), and using only open-source projects, which makes it feasible to replicate or include in other systems, but also allows its implementation as part of educational projects in electronics. The developed systems have been tested on real data with very good results. Regarding the authentication system, the validation process is done in 3.3 seconds in average with an Equal Error Rate (EER) of 19% on test files with 20 seconds, and tested with up to 87 different speakers. On the other hand, the language recognition system is able to recognize up to six languages. For this system, important efforts were done in order to reduce the processing time and memory requirements while keeping high the recognition rate. The final system uses 64 Gaussians and 200 i-vectors, obtaining an average cost error rate (C_{avg}) of 8.6% for the six languages.**

Keywords— **Speaker recognition, Language recognition, i-vectors, embedded devices, open-source tools.**

I. Introducción

Este artículo describe la implementación de dos tipos de sistemas de reconocimiento basados en el uso de la voz. El primero de ellos es un sistema de reconocimiento de locutor que permite el acceso a contenido multimedia en casa (tales como películas, música o fotos) mediante diferentes clases de dispositivos móviles (tales como tabletas, Smartphone ó controles remotos). Hoy por hoy, la mayoría de los sistemas de reconocimiento de locutor son utilizados como mecanismo de seguridad en aplicaciones de autenticación remota [1]. Sin embargo, nuestro sistema toma otro enfoque ya que está motivado por el creciente aumento de dispositivos capaces de albergar, visualizar o registrar todo tipo de producciones multimedia, así como el esfuerzo de diversas compañías por proveer aplicaciones y mecanismos de interacción multimodal que permitan acceder a distintos contenidos. Ejemplos de este tipo de aplicaciones son el Windows Media Center comercializado por Microsoft o el U-Verse Easy Remote de AT&T. Pese a la gran variedad de prestaciones incluidas, en general no cuentan con ningún tipo de sistema de autenticación que prevenga el acceso al contenido privado de un usuario por parte de otros usuarios.

Por otra parte, el segundo sistema permite la identificación automática del idioma hablado por una persona. Este tipo de sistemas son de bastante aplicabilidad hoy en día como paso previo para tareas automatizadas más complejas. Así por ejemplo, un sistema de reconocimiento de voz, necesario para transcribir las palabras dichas por una persona, requiere conocer previamente el idioma en que esa persona va a hablar con el fin de cargar adecuadamente los modelos acústicos y de lenguaje necesarios para realizar el reconocimiento; otro ejemplo de aplicación son los denominados call-centers o centros de atención automatizada de clientes. En este caso, si una persona llama hablando en inglés, el sistema debe reconocer dicho idioma y transferir la llamada a un operador que hable inglés. También se pueden mencionar otros usos como son los quioscos de información automatizada [2] o los sistemas de traducción automática de voz-a-voz [3].

Finalmente, conviene mencionar que los dos sistemas desarrollados se han diseñado y programado de tal forma que puedan ejecutarse en un sistema embebido de bajo coste, utilizando para ello una Raspberry Pi. Con el fin de lograr este objetivo, se hizo un esfuerzo importante por mantener un tiempo de procesamiento y de utilización de memoria muy bajo con el fin de permitir su utilización en sistemas reales. Así mismo, se buscaba también garantizar una alta tasa de reconocimiento por lo que se han investigado e implementado los algoritmos de reconocimiento más robustos y que constituyen los métodos más avanzados en este tipo de sistemas. Por otra parte, es necesario comentar que la implementación de ambos sistemas se realizó utilizando código de libre distribución en la web con el objetivo de conseguir una solución económicamente viable, con capacidad

Paper submitted in October 15th, 2013. This work has been supported by project TIMPANO (TIN2011-28169-C05-03) and the Academic and Educational Innovation Observatory of the Technical University of Madrid project No. 563.

L. F. D'Haro, Dpto. de Ing. Electrónica, ETSI de Telecomunicación, Universidad Politécnica de Madrid, Madrid, Spain, lfdharo@die.upm.es

R. Cordoba, Dpto. de Ing. Electrónica, ETSI de Telecomunicación, Universidad Politécnica de Madrid, Madrid, Spain, cordoba@die.upm.es).

J. I. Rojo, Universidad Politécnica de Madrid, Madrid, Spain, ji.rojo@alumnos.upm.es

J. Díez, Universidad Politécnica de Madrid, Madrid, Spain, jorge.diez.delafuente@alumnos.upm.es

D. Avendaño, Universidad Politécnica de Madrid, Madrid, Spain, diego.apeces@alumnos.upm.es

J. M. Bermudo, Universidad Politécnica de Madrid, Madrid, Spain, jose.bmera@alumnos.upm.es

de mejora permanente, y que permita que el sistema sea replicable. Por último, y con el objetivo de animar a las instituciones de educación superior de los diversos países iberoamericanos, es importante indicar que ambos proyectos se desarrollaron como parte de una iniciativa que busca integrar las actividades de investigación y la innovación educativa dentro de las asignaturas de grado, en este caso de los alumnos de 3^{r} año de la asignatura de Sistemas Digitales II para la titulación de Ingeniero graduado en Tecnologías y Servicios de Telecomunicación que se imparte en la Universidad Politécnica de Madrid. Para referencias adicionales y videos demostrativos ver: http://sdii.die.upm.es y http://lsed.die.upm.es.

Este artículo se organiza de la siguiente manera: en las secciones II y III se describe en detalle cada uno de los sistemas propuestos, empezando primero por el sistema de identificación de locutor y pasando luego al de idioma, especificando en cada caso los algoritmos, aplicaciones software y configuración hardware utilizados. Luego, en la sección IV se presentan los resultados y medidas realizadas tanto para el sistema de reconocimiento de locutor como los resultados para el sistema de reconocimiento de idioma. Y finalmente, en la sección V, se presentan las conclusiones generales y trabajos futuros para cada uno de los subsistemas.

II. SID: Descripción del Sistema

La Fig. 1 muestra los diferentes módulos que componen la arquitectura del sistema de autenticación por voz y acceso a los contenidos multimedia. Como se puede observar, la arquitectura es distribuida, lo cual permite a los usuarios acceder al servicio mediante un único (la Raspberry Pi). El servidor a su vez puede conectarse con varios *media centers* almacenando cada uno diferentes contenidos y estando físicamente ubicados en lugares diferentes. Así mismo, el servidor permite al usuario acceder al sistema mediante diversas clases de dispositivos clientes como por ejemplo, tabletas, PC o Smartphones. Una ventaja adicional de esta arquitectura es que cada módulo es independiente de los demás, por lo que esto permite al desarrollador remplazarlos por otros sin afectar al correcto funcionamiento del sistema. Finalmente, todos los módulos fueron implementados a partir de aplicaciones de libre distribución lo que permite que sus funcionalidades se puedan ir ampliando de forma constante.

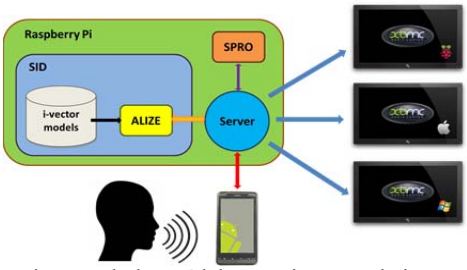

Figura 1. Arquitectura de los módulos que integran el sistema de acceso a contenidos multimedia.

A continuación, se explica en detalle cada módulo haciendo énfasis en el sistema de identificación de locutor.

A. Raspberry Pi (RPi)

La Raspberry Pi (http://www.raspberrypi.org/) es un ordenador de bajo coste desarrollado por la Fundación Raspberry Pi y que se comercializa con mucho éxito desde principios del año 2012. Las razones de su éxito están ligadas en gran parte a su pequeño tamaño, bajo consumo de potencia, disponibilidad de interfaces con una gran variedad de periféricos y al hecho de que puede ejecutar Linux como sistema operativo, lo cual la hace perfecta para el desarrollo de aplicaciones embebidas. Para este sistema, se utilizó una Raspberry Pi modelo B con 512 MB de memoria RAM, la cual incluye un procesador ARM1176JZF-S con un reloj a 700 MHz pero que en este caso se reconfiguró con la opción "modesta" lo cual permite su funcionamiento a 800 MHz. El objetivo de esta configuración fue permitir una ejecución más agradable para el usuario del software del *media center*. Como sistema operativo se utilizó la distribución Raspbmc (http://www.raspbmc.com/)(aunque también se hicieron pruebas con la distribución Raspbian sin ningún problema). Con el objetivo de ejecutar el sistema operativo, los programas y almacenar el contenido multimedia, se utilizó una tarjeta SD de 8 GB clase 10. Respecto a la memoria RAM utilizada por cada módulo, se realizaron pruebas y se determinó que el *media center* utilizaba hasta 70 MB mientras se reproducía una película (http://www.bigbuckbunny.org/) con formato de 1080p H.264 y con decodificación hardware, mientras que el sistema de autenticación con todos sus módulos y matrices llegaba a utilizar hasta 6 MB, y el servidor usaba 1 MB. De esta forma se confirmó que los 512 MB disponibles eran suficientes, con lo que para el sistema final se particionó la memoria de tal forma que la GPU pudiera utilizar 256 MB y dejar el resto para la CPU.

B. Media center

XBMC (http://xbmc.org/) es una aplicación de descodificación de vídeo multiplataforma de libre distribución desarrollada por la fundación XBMC. Esta aplicación permite a los usuarios reproducir y visualizar vídeos, música, podcast y ficheros multimedia desde un servidor local o remoto. Adicionalmente, incluye un gestor de perfiles y un servidor de eventos que permite la recepción de peticiones de carga de perfiles mediante el uso de paquetes UDP y que es aprovechado por el sistema de identificación para solicitar cargar el perfil del usuario autenticado y permitir el acceso a las listas de contenido multimedia personales. Cuando la aplicación se ejecuta por primera vez se carga automáticamente un perfil público que se usa hasta que se realiza la autenticación.

C. Cliente para Android

Este módulo permite a los usuarios controlar el media center mediante una interfaz gráfica de usuario (GUI). Para la arquitectura propuesta esta interfaz se basa en un cliente Android incluido como parte del proyecto XBMC que permite la ejecución de los comandos básicos de un control remoto (e.g. reproducir, parar, grabar, navegación por listas, entre otras). Además, su interfaz permite al usuario especificar los

valores de conexión con el servidor (i.e. dirección IP, y puerto). La interfaz original de este cliente se modificó ligeramente con el fin de incluir una serie de botones y cuadros de texto que permiten al usuario grabar su voz y enviar el audio al servidor (RPi) con el objetivo de realizar la autenticación.

Por otra parte, y con el fin de reducir la transmisión de datos y proveer una interfaz tipo manos libres, se incorporó un mecanismo de detección de voz basado en la energía de la trama. Esta detección se realiza usando reglas heurísticas que permiten garantizar un mínimo de habla y evitan el envío de grabaciones con ruidos espurios al servidor.

Finalmente, tras enviar los paquetes de datos con voz al servidor, el cliente entra en un modo de espera hasta que llegue el resultado de la autenticación. En caso de recibir un resultado positivo, el cliente muestra un mensaje de bienvenida mientras la interfaz del XBMC carga el perfil de usuario identificado y permite el acceso al contenido privado. En caso de que el resultado sea negativo, por estar la puntuación por debajo de un umbral configurable predefinido, el cliente muestra un mensaje de rechazo a la vez que la interfaz del XBMC mantiene cargado el perfil de usuario por defecto del sistema. De esta manera, el usuario tiene acceso al contenido público de su perfil a la vez que puede intentar hacer la autenticación nuevamente. En caso de que la autenticación no se produzca tras tres intentos, se deshabilita la interfaz de autenticación por voz y se pide al usuario que introduzca una contraseña escrita. Si la contraseña no es la correcta, el sistema solicita al usuario que mejore su modelo de habla mediante la incorporación de nuevas grabaciones, y bloquea el perfil durante los siguientes 30 minutos.

D. Servidor de autenticación

Este módulo es responsable de recibir los paquetes de voz enviados desde el cliente Android, almacenar el audio recibido ordenadamente en una carpeta predeterminada y ejecutar el proceso de identificación de locutor. Una vez se realiza la identificación, el servidor devuelve el resultado al cliente Android y queda a la espera de que, en caso de ser una identificación positiva, el cliente le solicite la carga de un perfil determinado que luego se redirige al cliente XBMC.

E. Sistema de identificación de locutor

Este módulo es el que realiza la tarea de identificación de locutor independiente de texto. Aunque es posible realizar una identificación dependiente de texto, lo cual haría que los resultados de identificación fueran mayores, se decidió mantener la independencia del texto con el fin de evitar el uso de contraseñas y reducir la posibilidad de engañar al sistema. Los pasos para realizar la identificación son los siguientes:

Parametrización: Tras recibir las muestras de audio capturadas por el cliente Android, este módulo extrae la información acústica relevante utilizando la herramienta SPro (http://spro.gforge.inria.fr/). En este caso, se extrajeron 12 coeficientes MFCC (Mel-Frequency Cepstral Coefficients), el logaritmo de la energía, así como los parámetros dinámicos (delta y delta-delta) [4], los cuales se calculan usando una ventana de Hamming de 25 ms y un desplazamiento de 10 ms. Estos parámetros fueron elegidos debido a que son los más utilizados actualmente para este tipo de aplicaciones. Sin embargo, en futuros desarrollos se pretende incorporar también coeficientes prosódicos por su aportación de información característica del usuario [5].

Detección de voz/no voz: Este módulo es el encargado de detectar únicamente las tramas de voz enviadas por el cliente Android. Este proceso es complementario al de detección de voz implementado en el cliente Android ya que permite hacer una detección más precisa al usar un algoritmo más preciso. El objetivo es mejorar las tasas de reconocimiento al emplear únicamente información de voz y reducir el tiempo de procesamiento. Este módulo es el implementado por la herramienta ALIZE/LIA_RAL (http://alize.univ-avignon.fr/) [6]. El algoritmo que se implementa consiste en el uso de un modelo GMM de dos/tres Gaussianas que se entrenan con los valores de la log-energía en cada trama y que tras un proceso iterativo de entrenamiento consigue que se asignen las tramas de más alta energía a una de las Gaussianas; durante la fase de evaluación cualquier trama que tenga una alta verosimilitud con dicha Gaussiana es etiquetada entonces como voz. Posteriormente se aplican algunas reglas de suavizado con el fin de evitar detectar como voz señales ruidosas de corta duración. Dada la importancia de este proceso, se plantea como línea futura integrar otros reconocedores más robustos al ruido como el propuesto en [7] que está disponible como código abierto (http://cs.uef.fi/pages/tkinnu/VQVAD/VQVAD.zip).

Extracción de i-vectores: Descritos en [8], los i-vectores son una representación en una baja dimensionalidad de las características acústicas de la voz, que permiten modelar de forma conjunta las variabilidades que pueda tener la voz del locutor a lo largo del tiempo, así como las variaciones del canal y de la voz dentro de una misma sesión. Gracias a su robustez, a los buenos resultados presentados en competiciones internacionales como las organizadas por NIST (http://www.itl.nist.gov/iad/mig/tests/lang/) y su independencia al contenido de la grabación los i-vectores constituyen el estado de la cuestión tanto en reconocimiento de locutor como de idioma; razón por la que se utilizaron en este sistema. La implementación se realizó mediante el software ALIZE/LIA_RAL que permite entrenar, extraer y puntuar los i-vectores.

Uno de los pasos necesarios para extraer los i-vectores es entrenar un modelo base independiente de locutor (UBM, Universal Background Model), así como la matriz de extracción de los i-vectores (matriz T en la terminología de los i-vectores); para ello se utilizó un subconjunto de ficheros de audio de la base de datos Callfriend Non-Caribean Spanish [9], tomando un total de 20 ficheros de hombres y 10 ficheros con voz de mujeres con una duración promedio de 28 y 23 minutos respectivamente. Adicionalmente, se agregaron 7 ficheros (de 5 minutos de duración promedio) con grabaciones de vídeos extraídos de internet con el fin de mejorar la robustez de los modelos. El modelo UBM final consta de 32 Gaussianas independientes de género y la matriz de extracción

de i-vectores es de dimensión 39*32 x 100 (39 parámetros acústicos: 12 MFCC + Energía + deltas + delta-deltas, 32 Guassianas y dimensión de proyección de 100 i-vectores). Estos valores permiten reducir el coste computacional y la memoria RAM utilizada para realizar los cálculos en tiempo de ejecución.

Clasificación: En esta etapa, el i-vector generado para el fichero de test se compara con los i-vectores entrenados del locutor que intenta acceder al sistema. El proceso de obtención de las puntuaciones (scoring) se hizo utilizando la distancia coseno [10] dado que es rápida de calcular y es altamente compatible con la proyección que se realiza en el espacio multidimensional de los i-vectores. La clasificación se realiza empleando la distancia promedio entre los i-vectores entrenados y el i-vector de la frase a evaluar. En función de que la distancia sea superior a un umbral pre-establecido se accede o no al sistema. Cabe mencionar que en el sistema actual no se aplican las técnicas de normalización de i-vectores más comunes tales como WCCN o LDA [11] aunque se plantea incluirlas en breve ya que estas técnicas se han incluido en la última versión de ALIZE/LIA_RAL.

III. LID: Descripción del Sistema

Por su parte, el sistema de reconocimiento de idioma está compuesto por tres módulos principales que se pueden ver en la Fig. 2. El primer módulo es la Raspberry Pi que provee la interfaz de usuario que permite acceder a todos los servicios, recoger las muestras de voz y ejecutar los algoritmos de reconocimiento de idioma. El segundo módulo es el sistema de identificación de idioma que se encarga de parametrizar los ficheros de audio, ejecutar la detección de segmentos de voz, extraer los i-vectores, fusionar las diferentes puntuaciones obtenidas por cada tipo de sistema independiente y realizar la clasificación. El tercer módulo es un servidor e interfaz web que permite a los usuarios capturar su voz o subir los ficheros de audio, llamar al sistema de reconocimiento de idioma, y luego al servicio online de transcripción y traducción. A continuación se describe cada módulo en detalle.

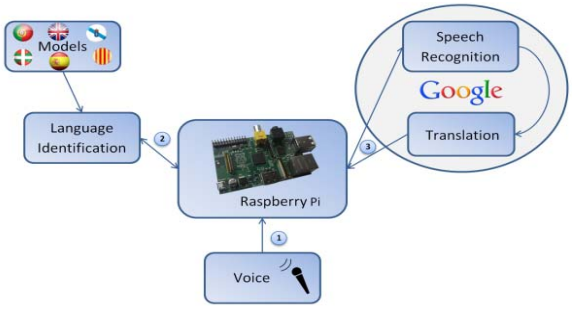

Figura 2. Módulos del sistema de reconocimiento de idioma.

A. Raspberry Pi (RPi)

Para este sistema se utilizó la misma Raspberry Pi del sistema de verificación de locutor. La única diferencia fue que se reconfiguró el reloj del sistema para funcionar a 1.0 GHz con el fin de garantizar que el sistema se ejecutara más rápido. Por otra parte, todos los programas se optimizaron con el fin de funcionar en la RPi y ser ejecutados usando la suite matemática Octave en su distribución para Raspbian. Teniendo en cuenta las restricciones de la RPi, los ficheros de audio se grabaron usando un micrófono USB y utilizando una frecuencia de muestreo de 8 KHz@16 bits.

B. Sistema de identificación de idioma

Este módulo está configurado de forma similar al sistema que se empleó en la competición internacional de reconocimiento de idioma Albayzin 2012 [12]. En esta evaluación, el sistema obtuvo muy buenos resultados gracias a la fusión de 3 sub-sistemas distintos [13]: 1) Sistema acústico basado en el uso de parámetros MFCC-SDC + filtrado RASTA + CMNV + i-vectores, 2) Sistema acústico basado en el uso de características RPLP-SDC + filtrado RASTA + CMNV + i-vectores, y 3) Sistema fonotáctio basado en el uso de cuentas de posteriogramas de trigramas + i-vectores. Al igual que la mayoría de los sistemas más avanzados actualmente, todos los subsistemas hacen uso de proyecciones sub-espaciales mediante i-vectores [8] que luego son calibrados y fusionados utilizando regresión logística multiclase. Una de las principales ventajas de este sistema fue el uso de las características RPLP que aportan robustez ante ruido [14] [15] y la incorporación del sistema fonotáctico [16] que utiliza un vector de características no disperso a partir de los valores de posteriogramas obtenidos usando el reconocedor de fonemas de libre distribución de la Universidad de Brno (http://speech.fit.vutbr.cz/software). A pesar de los buenos resultados obtenidos utilizando estos tres sub-sistemas, en el contexto de la aplicación para la RPi, se incluyeron únicamente los dos sub-sistemas acústicos con el fin de reducir la carga computacional y sin que por ello se redujera considerablemente la tasa de reconocimiento.

Dado que la configuración del sistema es la misma que para la evaluación Albayzin, este módulo es capaz de identificar 6 idiomas distintos, algunos de ellos bastante parecidos entre sí, que son: catalán, español, inglés, gallego, portugués y vasco. La identificación se realiza en los siguientes pasos:

Parametrización: Al igual que para el sistema de identificación por voz, este paso se realiza para extraer la información acústica más relevante. Los parámetros se extraen usando también SPro extrayendo los 7 primeros MFCCs y calculando los parámetros SDC [17]. Luego se aplica una normalización CMNV y un filtro RASTA [18]. Los parámetros RPLP se extrajeron usando el programa Ctucopy (http://noel.feld.cvut.cz/speechlab) con una configuración igual a la de los MFCCs.

Detección de Voz: Para este paso se utilizaron nuevamente las herramientas de ALIZE/LIA_RAL de la misma manera que para el sistema de verificación biométrica.

Extracción de i-Vectores: Para este sistema se recurrió a la implementación proporcionada por la Univ. Tecnológica de Brno (http://speech.fit.vutbr.cz/software/joint-factor-analysis-matlab-demo) con algunos cambios a fin de ejecutarse en Octave.

Fusión y clasificación: El objetivo de este paso es tomar los i-vectores de test previamente generados a partir de los

diferentes tipos de parámetros acústicos (MFCC y RPLP) y generar una puntuación que mida la similitud entre los i-vectores de prueba con el modelo de i-vectores entrenados offline para todos los ficheros que pertenecen al mismo idioma. Esta comparación se realiza utilizando un clasificador de regresión logística multi-clase. Posteriormente, las puntuaciones generadas se calibran y se fusionan las salidas de los dos sub-sistemas acústicos. En el caso del fusionador se utilizó un Back-end Gaussiano seguido por un discriminador de regresión logística multiclase. Para este sistema, los i-vectores se normalizaron previamente en longitud y se re-proyectaron utilizando una matriz de covarianza intra-clase (WCCN [19]), lo cual permite reducir las variaciones producidas por las diferencias en longitud de los ficheros de audio y de las variaciones de canal y locutor.

C. *Servidor web e integración con los servicios de Google*

Con el objetivo de permitir el acceso a los usuarios al sistema de identificación de idioma y posteriormente a los servicios online, se desarrolló una interfaz web (codificada usando Javascript y HTML5) alojada mediante un servidor web (en este caso Node (http://nodejs.org/)). La interfaz gráfica permite al usuario grabar su voz o subir un fichero de audio a fin de que se identifique el idioma del mismo. Tras la identificación, el sistema envía una petición POST al servicio de transcripción online de Google pasándole como información el fichero de audio y el idioma que ha reconocido. Una vez recibido el texto transcrito, se realiza una nueva solicitud POST esta vez al sistema de traducción online de Google, enviando el texto transcrito y el idioma al que se debe traducir (el cual es seleccionado por el usuario a través de la interfaz web). Dado que el sistema de identificación de idioma es capaz de reconocer hasta 6 idiomas distintos, la interfaz permite traducir los textos transcritos entre estos idiomas. Sin embargo, en futuras versiones se espera poder ampliar dicho sistema a más idiomas y con ello la interfaz con el fin de aceptar y traducir a otros idiomas soportados por Google.

IV. Resultados y estadísticas de ejecución

A. *Sistema de Identificación Biométrico*

La Fig. 3 muestra el tiempo que tarda en ejecutarse cada uno de los pasos del sistema de verificación en función de los recursos disponibles de la RPi (i.e. si el XBMC está detenido o se encuentra reproduciendo una película full HD). Estos resultados se obtuvieron usando un fichero de voz con una duración de 20 segundos. Tal como se puede observar, el tiempo que tarda el sistema en responder es de 2,5 segundos cuando el XBMC está detenido y de 5,0 segundos si está reproduciendo la película. Aunque este tiempo no es muy elevado, es evidente que se requieren esfuerzos adicionales para reducirlo. Para ello, se propone seguir algunas de las ideas planteadas en [20], en el que se describe un sistema de verificación por voz en tiempo real implementado en una FPGA. Aunque este sistema es muy rápido, conviene mencionar que el algoritmo de identificación implementado es menos robusto y que el dispositivo empleado no permite todas las funcionalidades ofrecidas por la RPi; pese a ello es posible realizar algunas optimizaciones similares a las reportadas.

La Fig. 4 muestra la curva DET (Detection Error Tradeoff) del sistema. Propuesta en [21], esta curva es una versión linealizada de la curva ROC (Receiver Operation Characteristic) en la que los ejes están en escala logarítmica. La curva muestra los errores de falsa alarma (i.e. cuando el sistema deja pasar a un impostor) contra los errores de falso rechazo (i.e. cuando el sistema rechaza al usuario correcto). La curva permite conocer además la tasa de igualdad de error (EER, Equal Error Rate) que es el valor en que valen lo mismo tanto los errores de falso rechazo como de falsa alarma; así mismo, dado que la curva muestra los resultados para todos los posibles valores de umbrales es posible determinar cuál es el valor óptimo del sistema para una determinada condición de compromiso entre ambos tipos de error. Para obtener los resultados de la Fig. 4, se utilizó un banco de pruebas con 87 usuarios (65 hombres y 22 mujeres), lo cual resulta en un total de 429 pruebas positivas (en las que el locutor debe poder ser identificado positivamente contra su modelo) y 36.894 pruebas falsas (en el que el sistema debe detectar que es un usuario falso). Para hacer más realistas las pruebas, cada modelo de locutor se entrenó usando sólo un minuto de audio; para las pruebas se usó una media de 5 ficheros de 20 s cada uno por locutor. El contenido de todas las locuciones era distinto por lo que el sistema es independiente de texto.

En la figura se puede observar que la tasa de EER es aprox. de un 19%, lo cual es un resultado bastante bueno teniendo en cuenta la longitud de los ficheros de entrenamiento y prueba, así como la independencia de género y texto. A modo de comparativa se pueden consultar los resultados de las últimas evaluaciones internacionales de reconocimiento de locutor [22] organizadas por la agencia norte-americana NIST (http://nist.gov/itl/iad/mig/spkr-lang.cfm) y en la que participan los mejores investigadores del mundo. Así por ejemplo, para la evaluación de 2008, el mejor sistema presentado obtuvo un EER de 9,7% utilizando ficheros de entrenamiento de entre 3-5 minutos y realizando las pruebas en ficheros de 10 segundos (condición short2-10sec). Mientras que en la evaluación de 2010, utilizando ficheros de entre 5 a 8 minutos con variaciones de canal (micrófono o conversaciones por teléfono) y comparando contra ficheros de 10 segundos (condición core-10sec), el mejor de todos los sistemas obtuvo un 6,2% de ERR. Como se puede observar, la tasa de error de nuestro sistema en la RPi es más alta pero hay que tener en cuenta que el tiempo de entrenamiento y las duraciones de los ficheros de prueba condicionan enormemente la calidad de los resultados, además que para hacer que el sistema funcionara más rápidamente hubo necesidad de reducir mucho el tamaño de los i-vectores y el número de Gaussianas. En los sistemas presentados en las competiciones mencionadas antes es común usar alrededor de 600 i-vectores, 2048 Gaussianas y modelos dependientes de género [23], dado que no se busca que los sistemas funcionen en tiempo real, sino que consigan la mejor tasa de error posible, independientemente de los recursos empleados.

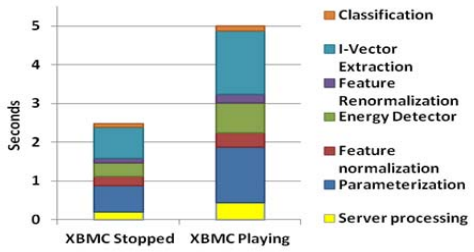

Figura 3. Tiempo de procesamiento para cada paso del proceso de reconocimiento considerando si el sistema de vídeo del XBMC está reproduciendo una película o no.

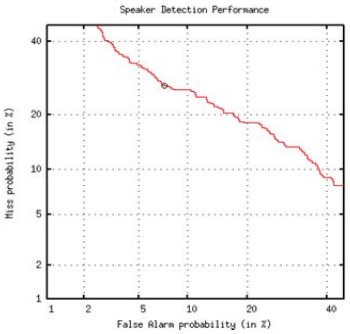

Figura 4. Curva DET con las tasas de falso rechazo y falsa aceptación del sistema sobre grabaciones de 20 segundos.

B. Sistema de Identificación de Idioma

Tal como se ha comentado previamente, el sistema de identificación de idioma (LID) se basa en la fusión de las puntuaciones de dos sub-sistemas acústicos, uno basado en usar MFCCs y otro con RPLPs utilizando ambos la misma configuración. En la Fig. 5 se muestran la tasa de error promedio del sistema (C_{avg}) utilizando los 941 ficheros de prueba empleados en la evaluación de Albayzin en función del número de Gaussianas y la dimensionalidad de los i-vectores; El tiempo de CPU se calcula al ejecutar el sistema de identificación en la RPi sobre un fichero de audio de 1 minuto de longitud aproximadamente. Tal como se puede ver, cuando el número de Gaussianas e i-vectores crece, el tiempo de procesamiento y requerimientos de memoria se incrementan a la vez que la tasa de error se reduce. Se puede apreciar que el menor error se consigue usando la misma configuración empleada durante la evaluación oficial, en el que se entrenó un modelo UBM de 512 Gaussianas e i-vectores de dimensionalidad 400. Sin embargo, el tiempo de procesamiento y de memoria es muy elevado, por lo que el sistema final usa 64 Gaussianas y 200 i-vectores. La razón es que aunque la tasa de error C_{avg} es 2,86% peor en valor absoluto, este resultado en cuanto a significancia estadística es equivalente al más complejo pero se ejecuta 5 veces más rápido y utiliza sólo la décima parte de memoria.

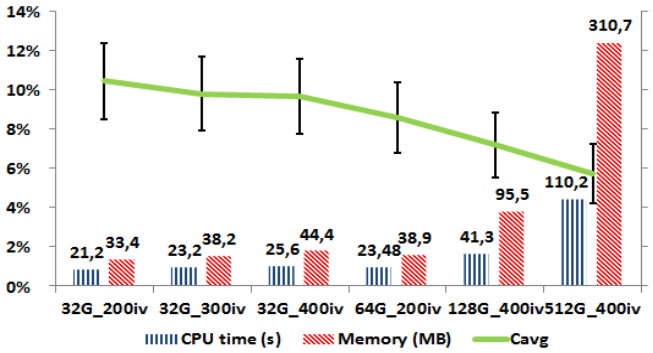

Figura 5. Comparación del rendimiento del sistema de identificación de idioma ejecutado en la RPi en función del número de Gaussianas e i-vectores.

Con la configuración seleccionada, el proceso completo de reconocimiento de un fichero de duración media de 30 segundos se realiza en aprox. 23,5 segundos. Este valor, aunque algo alto se debe principalmente a dos factores: a) el tiempo requerido para iniciar Octave (aprox. 5 segundos), y b) a la complejidad computacional para calcular los i-vectores que es de $O(K^3+K^2C+KCD)$, siendo K el No. de dimensiones del i-vector, C el No. de Gaussianas y D el tamaño de los vectores de características (56 dimensiones: 7 MFCC/RPLP + 49 SDCs). En el sistema implementado este proceso de cálculo se realiza dos veces (uno por cada sub-sistema acústico), con un total de 17,2 s. En la siguiente sección compararemos este resultado con otras implementaciones HW. En comparación, para las mismas condiciones, el sistema basado en PC tarda en promedio 1,9 segundos.

En cuanto a la tasa de error promedio (C_{avg}), el valor del sistema es de un 8,3% que es muy similar al resultado de 5,6% obtenido por el sistema que se ejecuta en el PC y que se presentó en la competición Albayzin LRE 2012 [24] [13]. Como comparativa entre el uso de sistemas basados en i-vectores y otros algoritmos (e.g. UBM-GMM o SVMs) se pueden consultar los resultados de las evaluaciones NIST LRE [25]. Por ejemplo, durante la competición del 2011, uno de los mejores sistemas presentados estaba basado en i-vectores y era capaz de reconocer hasta un total de 24 idiomas con una tasa de error de un 3% para ficheros de 30 segundos, de 7% para 10 segundos, y 18% para 3 segundos [26]. Por otra parte, los sistemas basados en SVMs o en modelos UBM-GMM eran bastante comunes varios años atrás en los que para la evaluación de 2003 obtenían un 6,1% y 4,8% respectivamente para la tarea de 30 s y sobre un total de 12 idiomas.

C. Comparación con otros algoritmos e implementaciones

Tal como ya hemos indicado anteriormente, los algoritmos incorporados en los dos sistemas desarrollados presentan resultados bastante buenos en comparación con otros sistemas más complejos. En esta sección queremos compararlos con sistemas implementados en otro tipo de plataformas hardware y con otros algoritmos de reconocimiento. La TABLA I presenta un resumen de los diferentes sistemas estudiados. Como se puede observar, existen un mayor número de sistemas de identificación de locutor que de idioma. Creemos

que la principal razón es que suele ser más necesario un sistema portable de reconocimiento de locutor que pueda ser instalado fácilmente en cualquier lugar con el fin de restringir el acceso a un lugar o datos, que un sistema de reconocimiento de idioma que en general se puede requerir más en aplicaciones offline (e.g. traducción automática, subtitulado de vídeos, etc.) y por tanto funcionan desde un PC.

TABLA I. COMPARATIVA DE TASA/VELOCIDAD PARA SISTEMAS CON OTRAS IMPLEMENTACIONES HW.

Ref.	HW	Características /Algoritmo	Datos	Tasa de acierto	Tiempo
[27]	• Cyclone II2C35 FPGA • Audio códec	• Vector Quantization • Clasificador basado en distancia Euclidea	• Audio 8KHz + 12 MFCC • Sistema dependiente de texto • 301 locutores • Test de 3 s	52,8 %	~15 ms
[28]Erro! Fonte de referência não encontrada.	• Xilinx V5 FPGA (fixed point)	• MFCC+SDC (offline y fuera de la FPGA) • Multi-Class SVM	• 3 Idiomas: Chino, Inglés y Japonés • Test de 30s	89,7 %	0,74 ms
[29]	• Xilinx Virtex-II XC2V6000 FGPA • 256 MB DDRRAM	• MFCC + deltas (offline y Tx a la FPGA) • 32 GMM	• 20 locutores • Test de 5 s	~67%	Tx: 16,8 ms Clasific.: 0,8 ms x clase
[30]	• Xilinx Spartan 3E FPGA board • 512 MB SDRAM	• 13 MFCC (FPGA) • GMM-UBM (Matlab)	• 31 locutores • Test de 5 s • Train 20 s	40%	>0,1ms
[31]	• dsPIC (Microchip) • 16KB RAM	• 13 MFCC + F0 + jitter + shimmer + CMN • GMM-UBM	• 168 locutores	90%	3,6 x Duración en (s) fichero audio
[32]Erro! Fonte de referência não encontrada.	• FPGA Spartan III	• Dependiente de texto • Sin VAD • SVM	• 52 locutores	N.D.	4,65 ms x cada trama de 25 ms

En general, en todas las implementaciones encontradas, el HW utilizado ha sido una FPGA ya que, como se describe en [33], tiene la gran ventaja de permitir el procesamiento paralelo de datos (algo común al comparar un modelo de locutor concreto frente a muchos otros simultáneamente), a la vez que disponen de una gran variedad de módulos y librerías matemáticas optimizadas para la arquitectura (e.g. FFT, filtrado, tablas de logaritmos para cálculo de scores, etc). Por otra parte, en buena parte de estos sistemas, encontramos que el HW se utilizaba únicamente para ejecutar una parte del reconocimiento (típicamente el clasificador), en tanto que tareas menos paralelizables como la parametrización o la ejecución de aplicaciones de usuario se realizaban en un servidor externo. En cualquier caso, el hecho de comparar estos sistemas entre sí es una tarea difícil dada la gran variedad en los datos usados, en el número y variedad de locutores/idiomas, algoritmos implementados, etc. Aun así, se puede comprobar en la tabla que los sistemas basados en FPGA son muchísimo más rápidos que en los sistemas propuestos pero las tasas de acierto y el número de modelos son menores. La única excepción es la referencia [28] pero únicamente identifica 3 idiomas y con 30 segundos de habla. Finalmente, creemos que todavía hay mucho margen de mejora para nuestro sistema en cuanto a su implementación, optimizando algunos de los pasos del algoritmo, re-implementando los algoritmos SW de forma más eficiente, reaprovechando mejor los recursos HW de la arquitectura utilizada, y realizando la parametrización de cada trama a medida que el usuario habla, de modo que el sistema funcionaría en tiempo real.

V. CONCLUSIONES Y TRABAJOS FUTUROS

En este artículo se han descrito dos sistemas distintos que se ejecutan en una Raspberry Pi. El primero es un sistema de verificación de locutor que permite el acceso a contenido multimedia privado. El sistema está basado en una arquitectura distribuida lo cual permite la incorporación de diversos módulos implementados mediante software de libre distribución, así como el control y acceso al sistema a través de diferentes dispositivos móviles. El sistema de verificación hace uso de la técnica de i-vectores con los cuales se han obtenido tasas buenas de verificación.

El segundo sistema permite el reconocimiento de idioma, así como la transcripción y traducción automática mediante el uso de una Raspberry Pi, algoritmos de reconocimiento basados en i-vectores acústicos, y la integración con servicios online. El sistema de reconocimiento empleado presenta una configuración similar a la utilizada en una competición internacional en la que obtuvo la mejor puntuación. En este caso, la diferencia principal estriba en el número de Gaussianas, tamaño de los i-vectores y en la fusión únicamente de sub-sistemas acústicos con el fin de reducir el tiempo de procesamiento y requerimientos de memoria.

Como trabajos futuros se plantean los siguientes en función del sistema. Para el sistema de identificación biométrica: la incorporación de nuevos algoritmos de normalización de los i-vectores y de las puntuaciones tales como Eigen Factor Radial [18] y PLDA. Adicionalmente, se trabajará en que el proceso de reconocimiento empiece tan pronto como el usuario habla con el fin de tener un sistema que se ejecute en tiempo real.

En cuanto al sistema de reconocimiento de idioma, se propone: Primero, utilizar un entorno matemático que sea más

rápido que Octave, como por ejemplo scikit-learn (http://scikit-learn.org) o Julia (http://julialang.org/); y en segundo lugar, reducir la complejidad del proceso de extracción de los i-vectores usando aproximaciones matemáticas similares a las que se proponen en [34].

AGRADECIMIENTOS

Este trabajo ha sido financiado por el proyecto TIMPANO (TIN2011-28169-C05-03) y el proyecto No. 563 del observatorio de Innovación Académica y Educacional de la UPM. Agradecemos también a Laura Alcocer Pérez-Regadera por sus contribuciones en la redacción de este artículo.

REFERENCIAS

[1] F. L. Alegre, "Application of ANN and HMM to Automatic Speaker Verification", IEEE LATIN AMERICA TRANSACTIONS, Vol. 5, No. 5, pp. 329-337, Sept. 2007.

[2] Karpov, A. A., Ronzhin, A. L. 2009. "Information enquiry kiosk with multimodal user interface". Pattern Recognition and Image Analysis, September 2009, Volume 19, Issue 3, pp 546-558.

[3] Eck, M.; Lane, I.; Zhang, Y., Waibel, A. 2010. "Jib-bigo: speech-to-speech translation on mobile de-vices" IEEE Spoken Language Technology Work-shop (SLT), pp. 165 – 166.

[4] Jurafsky, D., Martin, J. Speech and Language Processing. Pearson Education Limited, 2nd ed, 944 pp. ISBN-10: 1292025433

[5] E. Shriberg, L. Ferrer, S. Kajarekar, A. Venkataraman, A. Stolcke. 2005. Modeling prosodic feature sequences for speaker recognition. Speech Communication 46, 2005, pp. 455–472.

[6] Bonastre J. F., Scheeffer N., Matrouf D., et al. 2008. "ALIZE/SpkDet: a state-of-the-art open source software for speaker recognition", Speaker Odyssey.

[7] T. Kinnunen, P. Rajan, 2013. "A practical, self-adaptive voice activity detector for speaker verification with noisy telephone and microphone data", Proc. Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2013), pp. 7229--7233, Vancouver, Canada, May 2013.

[8] Dehak, N., Kenny, P. Dehak, R. Dumouchel, P., Ouellet, P. 2011. "Front-End Factor Analysis For Speaker Verification", IEEE Trans. on Audio, Speech and Language Processing, Vol. 19 (4).

[9] CallFriend Corpus, Linguistic Data Consortium, 1996, http://catalog.ldc.upenn.edu/LDC96S58

[10] Dehak, N., Dehak, R., Glass, J., Reynolds, D. Kenny, P. 2010. "Cosine similarity scoring without score normalization techniques", Speaker Odyssey.

[11] Garcia-Romero, D., and Espy-Wilson, C. Y. 2011. Analysis of i-vector length normalization in speaker recognition systems, in Int. Conf. on Speech Communication and Technology, 2011, pp. 249-252.

[12] Rodriguez-Fuentes, L. J., Brümmer, N., Penagarikano, M., Varona, A., Bordel, G., Diez, M. 2013."The Albayzin 2012 Language Recognition Evaluation". Interspeech.

[13] D'Haro, L. F., Cordoba, R., Caraballo, M. A., Pardo, J. M. 2013. "Low-resource language recognition using a fusion of phoneme posteriorgram counts, acoustic and glottal.based i-vectors". ICASSP.

[14] Hönig, F., Stemmer, G., Hacker, C., Brugnara, F. "Revising Perceptual Linear Prediction (PLP)". In Eurospeech 2005, p. 2997-3000.

[15] Rajnoha, J., and Pollák, P. 2011. "ASR systems in Noisy Environment: Analysis and Solutions for Increasing Noise Robustness". Radionengineering, Vol. 20, No. 1, April 2011, pp. 74-84.

[16] D'Haro, L. F., Glembek, O., Plchot, O., Matejka, P., Soufifar, M., Cordoba R., Cernocky, J. 2012. "Phonotactic language recognition using i-Vectors and phoneme posteriorgram counts", Interspeech.

[17] B. Bielefeld. 1994. "Language identification using shifted delta cepstrum", Fourteenth Annual Speech Research Symposium, 1994.

[18] Bousquet, P.M., Matrouf, D., and Bonastre, J. F. 2011. "Intersession compensation and scoring methods in the i-vectors space for speaker recognition", Interspeech, pp. 485-488.

[19] Hatch, A.O.; Stolcke, A. 2006. "Generalized Linear Kernels for One-Versus-All Classification: Application to Speaker Recognition," Acoustics, Speech and Signal Processing, 2006. ICASSP 2006.

[20] Ramos-Lara, R., López-García, M., Cantó-Navarro, E., and Puente-Rodriguez, L. 2013. "Real-time speaker verification system implemented on reconfigurable hardware". Journal of Signal Processing Systems, 71(2), 89-103.

[21] Martin, A. F., G. Doddington, T. Kamm, M. Ordowski, M. Przybocki. 1997. "The DET Curve in Assessment of Detection Task Performance", Proc. Eurospeech '97, Rhodes, Greece, September 1997, Vol. 4, pp. 1899-1903.

[22] NIST SRE evaluations. Web: http://www.nist.gov/itl/iad/mig/sre.cfm. Página consultada en enero de 2014.

[23] Saeidi, R., et al. 2013. "I4U submission to NIST SRE 2012:A large-scale collaborative effort for noise-robust speaker verification", Interspeech 2013, pp. 1986-1990.

[24] D'Haro, L.F., and Cordoba, R. 2012. "The GTH-LID System for the Albayzin LRE12 Evaluation", Iberspeech 2012, pp. 528-539.

[25] NIST LRE evaluations. Web: http://www.nist.gov/itl/iad/mig/lre.cfm. Página consultada en enero de 2014.

[26] E. Singer, P. Torres-Carrasquillo, D. Reynolds, A. McCree, F. Richardson, N. Dehak, and D. Sturim, 2012. "The MITLL NIST LRE 2011 Language Recognition System," Proc. Odyssey, pp. 209-215, Singapore, June 2012.

[27] J. Li, D.An, L. Lang, and D. Yang. 2012. "Embedded Speaker Recognition System Design and Implementation Based on FPGA", Procedia Engineering 29, pp. 2633 – 2637.

[28] Z. Nie, X. Zhang, and Z. Yang. 2012. "An FPGA Implementation of Multi-Class Support Vector Machine Classifier Based on Posterior Probability", Int. Proc. of Computer Science and Information Technology, vol 53(2), pp. 296 - 302, October 2012.

[29] P. EhKan, T. Allen, and S. F. Quigley. 2011. "FPGA Implementation for GMM-Based Speaker Identification", International Journal of Reconfigurable Computing.

[30] A. S. Poudel, D. Lekhak, K. Bashyal, and S. Shrestha. 2013. "Text-Independent Speaker Recognition System Based on FPGA", Final year project, Department of Electronics and Computer Engineering, Tribhuvan University.

[31] M. Lizondo, P. D. Agüero, A. J. Uriz, J. C. Tulli and E. L. Gonzalez. 2012. "Embedded speaker verification in low cost microcontroller", Congreso Argentino de Sistemas Embebidos 2012. Buenos Aires, Argentina. 15-17 Agosto, 2012.

[32] R. Ramos-Lara, M. López-García, E. Cantó-Navarro, and L. Puente-Rodriguez. 2009. "SVM Speaker Verification System Based on a low-cost FPGA", International Conference on Field Programmable Logic and Applications, pp. 582 – 586.

[33] A. Naufal, E. Phaklen, R. B. Ahmad, and S. Naseer. 2013. "Speaker Recognition System: Vulnerable and Challenges", International Journal of Engineering & Technology (0975-4024);Aug/Sep2013, Vol. 5 Issue 4, p 3191-3195.

[34] Li, M., Tsiartas, A., Segbroeck, M.V., Narayanan,S. 2013. "Speaker verification using simplified and supervised i-vector modeling", ICASSP 2013, Vancouver, Canadá.

Luis Fernando D'Haro (M'00), obtuvo el título de ingeniero electrónico en el año 2000 por la Universidad Autónoma de Occidente (Cali, Colombia), y su Ph.D. en el año 2009 por la Universidad Politécnica de Madrid. Desde el año 2007 es profesor ayudante doctor en la UPM.

Ricardo Córdoba (M'00), obtuvo su grado como Ingeniero en Telecomunicación y Doctorado por la Universidad Politécnica de Madrid (UPM) en 1991 y 1995 respectivamente. Enseña como profesor desde 1993 y es Profesor Titular desde 2003.

José Ignacio Rojo, es estudiante de cuarto curso del Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación en la Universidad Politécnica de Madrid.

Jorge Diez, fue estudiante del Liceo Francés de Madrid, especialidad Ciencias-Física. Actualmente es estudiante de cuarto curso del Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación en la UPM e imparte la asignatura de libre elección "Introducción a Linux".

Diego Avendaño, es estudiante de cuarto año del grado en Ingeniería de Tecnologías y Servicios de Telecomunicación de la Universidad Politécnica de Madrid.

José María Bermudo, es estudiante de cuarto año del grado en Ingeniería de Tecnologías y Servicios de Telecomunicación en la Universidad Politécnica de Madrid.