IMPLEMENTACIÓN DE UN RECONOCEDOR DISTRIBUIDO DE VOZ EN TIEMPO REAL SOBRE IP

Juan A. Morales Cordovilla[1], *Timo Bauman*[2], José L. Pérez Córdoba[1],
Antonio M. Peinado Herreros[1], Ángel M. Gómez García[1]

[1]Dpto. de Teoría de la Señal, Telemática y Comunicaciones. Universidad de Granada. España
[2] Hamburg University (Elsnet Summerschooll), Alemania

RESUMEN

En este trabajo se presenta una implementación de un sistema de reconocimiento distribuido del habla en tiempo real para su aplicación en un entorno de Internet. Desarrollado como una aplicación cliente-servidor, el cliente hace uso del *front-end* estándar definido por la ETSI. Incluye un detector de voz para sólo enviar información cuando el locutor habla y la transmisión de la información se hace de acuerdo al RFC 3557. El servidor realiza el reconocimiento utilizando programación dinámica, incluye a su vez un detector de voz y utiliza la técnica de mitigación de pérdida de paquetes propuesta en el estándar de la ETSI. La modularidad del diseño permite la utilización de cualquier otro reconocedor sin que por ello se vean afectados los clientes. Las pruebas reales de funcionamiento para la aplicación particular desarrollada han demostrado una alta fiabilidad en varias condiciones de transmisión.

1. INTRODUCCIÓN

El desarrollo de dispositivos portátiles con conexión a Internet abre un campo amplio a las aplicaciones del reconocimiento de voz. Estos dispositivos no suelen disponer de una gran potencia de cálculo y/o memoria por lo que es preferible implementar un sistema de reconocimiento de voz remoto. La Figura 1 muestra un diagrama de dicho sistema. Como se observa en la citada figura, se hace uso del paradigma cliente-servidor seguido en el desarrollo de aplicaciones en Internet. En este caso, el cliente se encarga de recoger la señal de voz y enviarla codificada al servidor (reconocimiento de voz basado en red, NSR, siglas inglesas para *Network based Speech Recognition*) o bien de obtener un conjunto de parámetros que representan a la voz (reconocimiento de voz distribuido, DSR, *Distributed Speech Recognition*). Esta filosofía tiene la ventaja adicional de que el usuario no debe preocuparse por el sistema de reconocimiento y permite diseñar de forma separada el cliente y el servidor así como realizar el mantenimiento y actualización de forma separada tanto del cliente como del servidor.

Este trabajo se ha realizado con la financiación del MEC/FEDER dentro del proyecto TEC2004-03829/TCM..

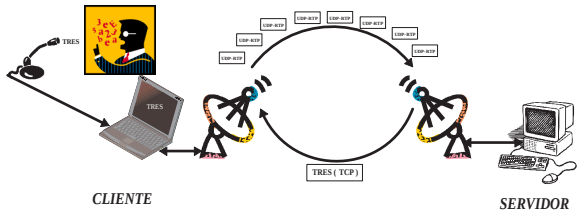

Figura 1. Esquema de un sistema de reconocomiento a través de IP.

Cada uno de los dos procedimientos de abordar el reconocimiento tiene sus ventajas e inconvenientes. Por ejemplo, DSR requiere la definición de un *front-end* estándar a seguir por todos los clientes. El Instituto Europeo de Estándares de Telecomunicación, ETSI (*European Telecommunications Standard Institute*) creó un grupo de trabajo, *Aurora DSR Working Group*, que ha publicado un conjunto de estándares sobre DSR. Sin embargo, es necesario que los fabricantes incluyan software y hardware que soporten dichos estándares.

Por otro lado, el uso de NSR no requiere la utilización de dichos estándares ya que, por ejemplo, en un teléfono móvil sólo hay que utilizar los codificadores de voz ya instalados. Sin embargo, estos codificadores de voz introducen una degradación en el reconocimiento además de requerir una tasa de bits superior.

Un ámbito de aplicación directa del DSR lo constituyen los ordenadores portátiles y PDA's. El abaratamiento de los precios, la extensión de su uso y la facilidad de conexión a Internet crean un entorno de desarrollo de aplicaciones de DSR. La implementación de los estándares ETSI para los *front-end* de reconocimiento en un portátil no ofrece ningún problema tanto del punto de vista de recursos hardware como software. Estos dispositivos vienen equipados con el hardware necesario y la potencia de cálculo que ofrecen excede la necesidades mínimas exigibles para la implementación de un *front-end*. La implementación del servidor no ofrece ningún problema ya que se puede situar en cualquier ordenador potente con conexión directa a Internet en cualquier parte del mundo.

Este trabajo presenta una implementación de un reconocedor distribuido del habla sobre IP que trabaja en

tiempo real. El sistema DSR se implementa como una aplicación cliente-servidor y hace uso de la interfaz de programación de aplicaciones API (*Application Programming Interface*) BSD socket para la conexión a través de la pila de protocolos de Internet. El cliente se puede instalar en cualquier computador personal que posea un equipo de audio que permita grabar la voz de un locutor. En el cliente se instala el *front-end* estándar de la ETSI, ES 201 108 [1] y la transmisión de las características de la señal de voz se transmiten de acuerdo al RFC 3557 [2].

El servidor lleva a cabo la tarea de reconocimiento del habla, tarea que se puede basar en el uso de modelos ocultos de Markov o en cualquier otra técnica. La aplicación implementada para demostrar el buen funcionamiento del sistema DSR a través de Internet es el reconocimiento de los primeros 20 dígitos más la palabra ”PARA” utilizada para indicar el final de la locución al reconocedor. Para ello se ha utilizado programación dinámica mediante el algoritmo DTW (*Dynamic Time Warping*) [3]. Pero hay que dejar claro que la aplicación desarrollada permite utilizar cualquier otra técnica de reconocimiento así como realizar tareas de reconocimiento más complejas y todo ello sin que se vea afectado el cliente ni el procedimiento de transmisión.

Este artículo se organiza como se indica a continuación. En la Sección 2 se describe de forma general las características del sistema de reconocimiento. Las secciones 3 y 4 describen los detalles de la implementación del cliente y el servidor respectivamente. Los resultados de rendimiento (tasa de reconocimiento) se presentan en la Sección 5. Finalmente, la Sección 6 contiene las conclusiones de este artículo.

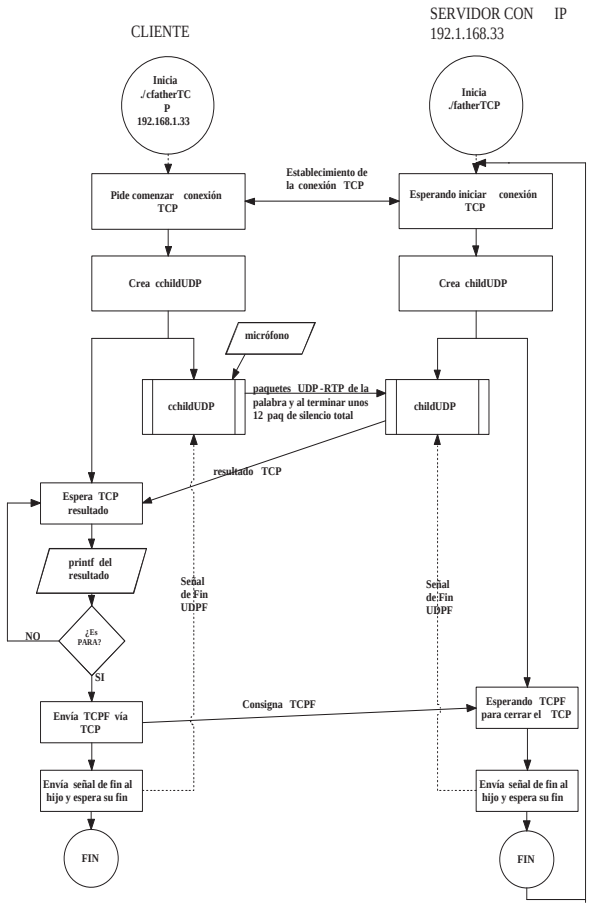

Figura 2. Esquema general del sistema cliente/servidor DSR.

2. DESCRIPCIÓN DEL SISTEMA DSR

En este apartado se van a describir las características globales del sistema de reconocimiento distribuido desarrollado. El sistema desarrollado como un sistema DSR se implementa como una aplicación cliente-servidor y hace uso de la interfaz de programación de aplicaciones API (*Application Programming Interface*) BSD socket para la conexión a través de la pila de protocolos de Internet.

El diseño de la aplicación se ha realizado con el objetivo de cumplir tres objetivos básicos:

- Escalabilidad y versatilidad (posibilidad de modificar o ampliar partes del sistema sin que se vean afectadas las partes que no se cambian).
- Eficiencia (simplificar las tareas con el objetivo de lograr la mayor rapidez posible y el mayor aprovechamiento de memoria y CPU).
- Claridad en la programación (hacer una programación que dé facilidad comprensiva).

Además, el sistema es capaz de trabajar tanto en el sistema operativo (SO) Windows XP como en Linux. Es posible tener el servidor trabajando en Linux y el cliente en Windows (o viceversa). Los programas han sido hechos específicamente para trabajar en Linux. La portabilidad a Windows se consigue a través de la plataforma Cygwin [4]. La captura de muestras de voz se hace a través de la librería de código abierto PortAudio [5] disponible también para varios SO (multiplataforma).

La comunicación entre el cliente y el servidor es a través de TCP y UDP. El esquema general de funcionamiento del cliente y del servidor se muestra en la figura 2. La conexión TCP sólo se utiliza para el control previo a la comunicación y el envío con seguridad de los resultados del reconocimiento del servidor al cliente. La conexión UDP, que no es segura, se utiliza para enviar/recibir las características de la señal de voz sin pausa y en tiempo real con ayuda del protocolo RTP.

Se ha utilizado multiprogramación (padre e hijo). Los padres sólo se encargan de comunicar y representar los resultados del reconocimiento vía TCP mientras que los hijos se encargan básicamente de la comunicación UDP y funcionan al ritmo del micrófono.

A continuación se indican otras características a destacar de este sistema DSR. Utilización de un algoritmo de detección de actividad de voz, VAD (*Voice Activity Detection*), que permite reducir el número de paquetes enviados al servidor ya que sólo se envían éstos cuando el locutor

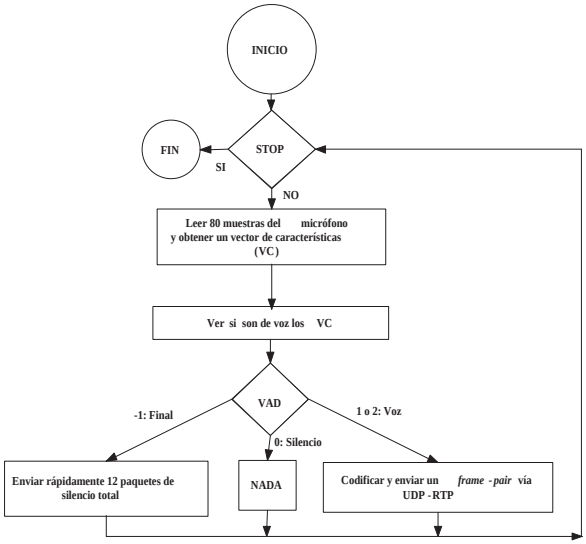

Figura 3. Organigrama del programa hijo del cliente UDP.

está hablando. Utilización de un algoritmo de mitigación de errores que trata de mejorar la tasa de reconocimiento cuando se producen pérdidas de paquetes. Para evitar el problema de la llegada desordenada de los paquetes éstos se numeran haciendo uso de la cabecera del protocolo RTP. Por último, indicar que al hacer una programación tan modularizada se permite escalar y hacer cambios con mucha facilidad, como por ejemplo, cambiar fácilmente el reconocedor DTW por uno basado en Modelos ocultos de Markov.

3. CLIENTE DSR

Esta es la parte que se encarga de captar las muestras de voz del micrófono, extraer los vectores de coeficientes, detectar si son de voz o no y en caso de que lo sean codificarlos y enviarlos. En la figura 3 se muestra el organigrama del cliente UDP así como todas las partes que funcionan conjuntamente con éste.

Como se ha comentado antes, para facilitar la programación se utiliza la librería PortAudio que es un conjunto de APIs de sonido en código C abierto que pueden ser llamadas desde casi cualquier SO. PortAudio proporciona una serie de funciones que permiten interactuar con la tarjeta de sonido para configurar la lectura de muestras y la propia lectura.

3.1. Extracción de características

Este apartado resume el estándar de la ETSI ES 201 108 [1], que especifica el *front-end* utilizado para la extracción de las características utilizadas en el reconocedor.

Según el estándar de la ETSI, para una frecuencia de muestreo de 8000 Hz. (empleada en este trabajo) cada vector de características se forma con las 200 últimas muestras (trama de 25 ms) y se forma uno nuevo cada 80 muestras (10 ms), generando 13 coeficientes Mel-cepstrales, MFCC (*Mel Frequency Cepstral Coefficients*), incluido el de orden 0, además del logaritmo de la energía de dicha trama. Con el fin de cuantificar eficientemente dichos parámetros se utiliza una cuantificación del tipo SVQ (*Split Vector Quantization*), aplicando diccionarios de 64 centros para los MFCC del 1 al 12, mientras que el coeficiente cepstral de orden 0 y la energía emplean diccionarios de 256 centros. La tasa final de este estándar es de 4.4 kbps.

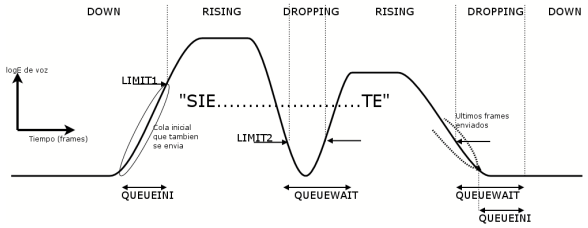

Figura 4. VAD del cliente.

3.2. Detector de actividad de voz, VAD

Mediante el detector de actividad de voz del cliente se realiza una estimación de si hay señal de voz en cuyo caso se prepara una transmisión de vectores de características al servidor. En caso contrario no realiza ninguna transmisión. El funcionamiento del VAD se explica por medio de tres posibles estados (DOWN, RISING y DROPPING) en los cuales se entra según sea el valor del logaritmo de la energía (LogE) de la trama actual y circundantes. En la figura 4 se representa un ejemplo en el que se muestra el LogE de cada trama respecto al tiempo. La curva corresponde aproximadamente a la palabra SIETE.

Normalmente el estado del VAD es DOWN, es decir, que no hay voz y sí ruido. La función VAD() va revisando continuamente el valor actual LogE de la señal de voz. Cuando el frame actual supera un valor (LIMIT1) bastante alto con respecto al ruido de fondo se considera que hay voz y se pasa al estado RISING (que hay voz elevada). En el estado RISING se van transmitiendo tramas a medida que van llegando. En este estado estará hasta que la curva decrezca y se pase el límite LIMIT2. Entonces se pasa al estado DROPPING (que está desapareciendo la señal de voz). Al entrar en este estado se espera varias tramas (QUEUEWAIT=11) antes de terminar y devolver -1 indicando que la palabra ha finalizado. La utilidad de la espera final es prevenir que el VAD piense erróneamente que ha terminado la palabra por el simple hecho de que se desvanezca la señal durante unos milisegundos como por ejemplo ocurre en la ”t” de la palabra siete.

Finalmente una vez que se ha detectado el final de una locución se envían consecutivamente unas 12 tramas de silencio para indicar al VAD del servidor que ha terminado la palabra.

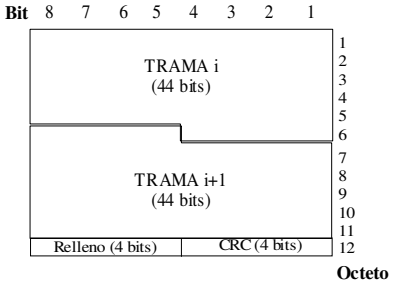

Figura 5. Formato de un *frame-pair*.

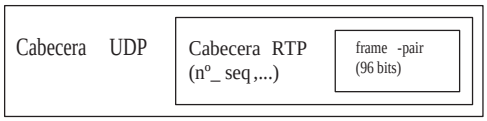

Figura 6. Encapsulamiento de un *frame-pair* vía RTP-UDP.

3.3. Paquetes RTP

Los vectores de coeficientes se envían de dos en dos tal como lo recomienda el estándar de la ETSI (RFC 3557). La disposición de la codificación de las dos tramas (un *frame-pair*) para su envío se muestra en la figura 5.

Como se observa, a continuación de las dos tramas codificadas se utilizan 4 bits para la comprobación de errores en el decodificador más 4 bits de relleno (tasa de bits final de 4,6 kbps). Si se detecta que los bits han sufrido alguna alteración por el camino se aplica el algoritmo de mitigación de errores. Esta comprobación se implementa a través de un código polinómico o CRC cuyo polinomio generador es el polinomio $g(X) = 1 + X + X^4$.

El *frame-pair* se transmite haciendo uso del protocolo RTP [2]. De los campos de la cabecera del protocolo RTP sólo se va a usar el campo de 'número de secuencia' para asegurar que los paquetes llegan ordenados. Posteriormente, el paquete RTP se transmite en el campo de datos de un datagrama UDP como se muestra en la figura 6.

4. SERVIDOR DSR

Dentro de un sistema DSR, el servidor es la parte encargada de llevar a cabo el reconocimiento propiamente dicho. Sin embargo, no es esta la única tarea a realizar. El programa hijo del servidor está encargado de recibir paquetes UDP, aplicarles el algoritmo de ordenación RTP, decodificarlos, aplicarles el algoritmo de mitigación de errores, realizar el reconocimiento si son paquetes de voz y enviar el resultado correspondiente. En la figura 7 se muestra el organigrama del servidor UDP así como todas las partes que funcionan conjuntamente con éste.

Según la figura, tras la inicialización se entra en el bucle principal en el que lo primero que se hace es esperar recibir un paquete UDP en modo bloqueante. Recibido el

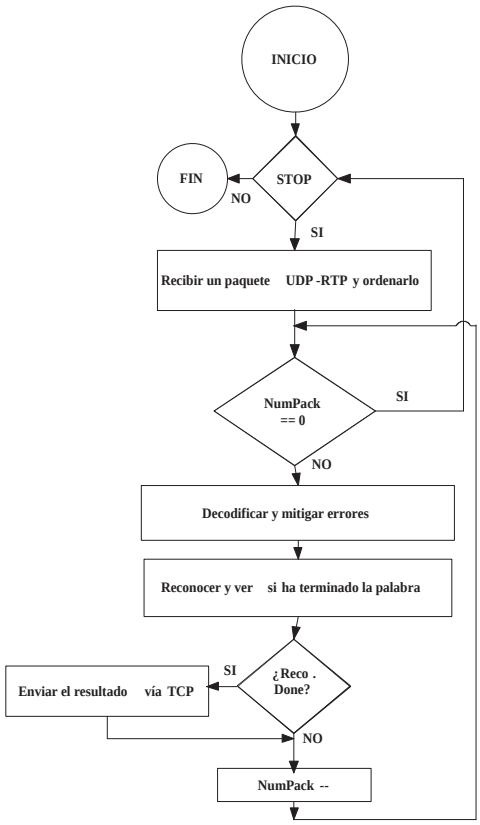

Figura 7. Organigrama del programa hijo del servidor UDP.

paquete, se le aplica el algoritmo de ordenación RTP. Este algoritmo suele devolver como salida un solo paquete RTP para ser procesado pero algunas veces puede que no devuelva ningún paquete (si ha recibido un paquete retrasado) o un número determinado de paquetes.

El paquete o paquetes devueltos se decodifican aplicándoles el algoritmo de mitigación de errores. El resultado suele ser dos tramas aunque algunas veces puede que no se devuelva nada, como es el caso en el que el *frame-pair* tenga el CRC erróneo, o un número determinado de tramas (si ha llegado un *frame-pair* con CRC bueno tras otros de CRC erróneo). El resultado pasa al detector de voz del servidor que se encarga de ir pasando tramas al reconocedor cuando detecta que son de voz. Cuando termina el reconocimiento de la palabra se envía el resultado vía TCP al cliente.

4.1. Algoritmo de mitigación de pérdidas

El problema de la transmisión a través de IP es la pérdida de paquetes. Como se ha visto, el *frame-pair* incluye un campo para la detección de errores en la transmisión. Para seguir la compatibilidad con el estándar, los paquetes perdidos se van a considerar que han llegado con errores de forma que se marcan como que ha fallado la comprobación de su CRC.

Los *frame-pair* erróneos no se descartan inmediata-

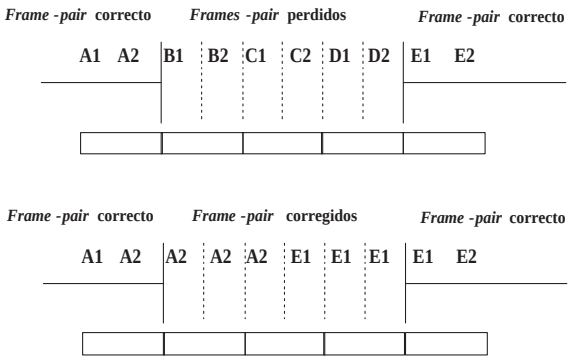

Figura 8. Ejemplo de corrección de *frames-pairs* erróneos.

mente si no que se van almacenando en un buffer a la espera de que llegue uno correcto con la esperanza de corregirlos. Al llegar uno correcto se ejecuta la función encargada de hacer la corrección y de devolver las tramas corregidas que habían quedado en espera en el buffer. La corrección aplicada se muestra en la figura 8. Se observa que la corrección consiste en sustituir la primera mitad de las tramas erróneas por la última trama correcta (A2) y la segunda mitad por la trama actual correcta (E1).

4.2. El reconocedor y el VAD asociado

La idea de utilizar un VAD en el reconocedor es tratar de disminuir el consumo de CPU por parte de la tarea de reconocimiento de forma que sólo se ejecuta esa tarea cuando el VAD asociado detecta *frame-pairs* que contienen voz. El funcionamiento de este VAD es más simple que el VAD del cliente. Sólo dos estados DOWN (no hay voz) y RISING (hay voz). En la figura 9 se muestra un ejemplo de funcionamiento del VAD del servidor. Si la energía de las tramas que se reciben no supera el limite TOTALSILENT se clasifican como tramas de silencio total y no se hace nada con ellas.

Como se ha comentado anteriormente, el reconocedor implementado es un reconocedor que hace uso del algoritmo DTW. Sin embargo, dada la alta escalabilidad de la programación de esta aplicación, podríamos utilizar otro reconocedor como, por ejemplo, uno basado en modelos ocultos de Markov.

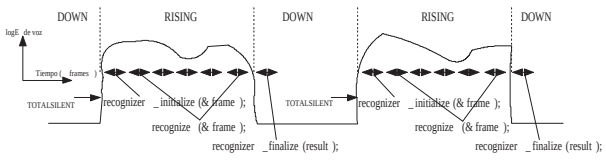

Figura 9. VAD del servidor.

5. RESULTADOS DE RECONOCIMIENTO

En este apartado se muestran los resultados de reconocimiento obtenidos en la ejecución de la aplicación DSR. Se comprueba tanto la efectividad del reconocedor como la del algoritmo de mitigación de errores que trabaja conjuntamente con el algoritmo de ordenación RTP. Se han hecho dos tipos de medidas. En un principio se han obtenido resultados de reconocimiento simulando un canal IP para varias probabilidades de pérdidas de paquete y diferentes longitudes de ráfagas.

El segundo tipo de medida de resultados se ha hecho en un entorno real. El cliente se ha ejecutado en un ordenador portátil conectado a Internet a través de una red inalámbrica WiFi. El servidor se ha instalado en un ordenador personal conectado permanentemente a Internet. Se han realizado varias pruebas de reconocimiento bajo diferentes niveles de potencia de la señal recibida, tratando situaciones en las que calidad de la señal recibida era muy buena (0 % de paquetes perdidos) a situaciones en que la calidad era muy mala (con un tanto por ciento de paquetes perdidos apreciable).

La tarea de reconocimiento para la que se han realizado las pruebas consiste en el reconocimiento monolocutor de 22 palabras (los números del CERO al VEINTE y la palabra PARA). Para obtener los datos de reconocimiento en la simulación del canal se han tomado cuatro condiciones de canal:

- Canal con una tasa de perdidas de paquetes UDP-RTP del 20 % y una la longitud media de las ráfagas perdidas de 1 paquete (condición marcada como 'Canal 20-1').
- Canal con un 30 % de perdidas y una longitud media de 2 (condición 'Canal 30-2').
- Canal con un 40 % de perdidas y una longitud media de 2 (condición 'Canal 40-2').
- Canal con un 50 % de perdidas y una longitud media de 4 (condición 'Canal 50-4').

Los resultados de reconocimiento se muestran en la figura 10. En dicha gráfica se muestran resultados de reconocimiento cuando no se utiliza algoritmo de mitigación ni tampoco el de ordenación RTP (ya que si no se introducirían paquetes falsos que empeoraría aun más el reconocimiento). Se observa un pobre resultado en el reconocimiento ya que la tasa de reconocimiento disminuye considerablemente a medida que el canal sufre un incremento en la tasa pérdidas, llegando a una tasa de reconocimiento del orden del 52,3 % (casi la mitad de las palabras fallan al ser reconocidas) bajo la ultima condición de canal ('Canal 50-4').

Cuando se utiliza el algoritmo de mitigación y el de ordenación RTP se comprueba una mejora considerable en la tasa de reconocimiento. Además, el sistema se muestra más robusto a un aumento a la tasa de pérdida de paquetes y/o la longitud de la ráfaga. Es de destacar que para

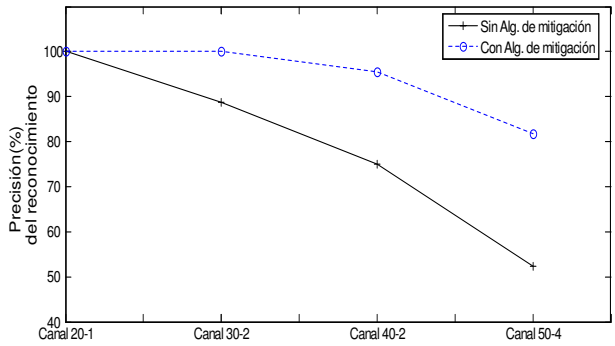

Figura 10. Resultados de reconocimiento. Canal X-Y hace referencia a un canal con una tasa de pérdidas del X % y una longitud de ráfaga de Y paquetes

la ultima condición de canal 'Canal 50-4' la tasa de reconocimiento aumenta desde el 52,3 % del caso anterior a un 81,8 %.

Las medidas de resultados en un entorno real han contado con la dificultad de tener el control sobre los errores ya que los errores dependen de la intensidad de la potencia de la señal. Estos son apreciables para una potencia baja o muy baja. Los errores que se cometen son principalmente ráfagas de paquetes perdidos más que errores puros de CRC. Los resultados también demostraron que el algoritmo de mitigación ayudaba a mejorar el reconocimiento, sobre todo cuando las ráfagas de error por palabra no eran mayores de 8 paquetes.

6. RESUMEN

En este trabajo se ha presentado una implementación de un sistema de reconocimiento distribuido del habla en tiempo real para su aplicación en un entorno de Internet. Desarrollado como una aplicación cliente-servidor, el cliente hace uso del *front-end* estándar definido por la ETSI. Incluye un detector de voz para sólo enviar información cuando el locutor habla y la transmisión de la información se hace de acuerdo al RFC 3557. El servidor realiza el reconocimiento utilizando programación dinámica, incluye a su vez un detector de voz y utiliza la técnica de mitigación de pérdida de paquetes propuesta en el estándar de la ETSI. La modularidad del diseño permitela utilización de cualquier otro reconocedor sin que por ello se vean afectados los clientes. Las pruebas reales de funcionamiento para la aplicación particular desarrollada han demostrado una alta fiabilidad en varias condiciones de transmisión.

7. BIBLIOGRAFÍA

[1] ETSI ES 201 108, "Speech processing, transmisión and quality aspects (stq); distributed speech recognition; front-end feature extraction algorithm; compression algorithms," 2000.

[2] IETF. RFC 3557, "Rtp payload format for etsi european standard es 201 108 distributed speech recognition encoding," 2003.

[3] C. S. Myers y L. R. Rabiner, "A comparative study of several dynamic time-warping algorithms for connected word recognition," *The Bell System Technical Journal*, vol. 60(7), pp. 1389–1409, September 1981.

[4] "http://www.cygwin.com/," .

[5] "http://www.portaudio.com/," .