CODIFICADOR DE VOZ MULTIPULSO A 9.6 Kbps EN TIEMPO REAL SOBRE EL TMS320C30

Luis Miguel Valiño, Asunción Moreno, Francisco Vallverdú
Departamento de Teoría de Señal y Comunicaciones
E.T.S.I.Telecomunicación de Barcelona

ABSTRACT

In this paper we present a full duplex implementation of a multipulse speech coder at 9,6 Kbps. The coder works over a DSP: TMS 320C30 in real time. LPC parameters and excitation pulses have been quantized by VQ techniques. The SNR values obtained are in the range of 13 dB to 15 dB.

INTRODUCCION

El codificador de voz multipulso nace debido a la necesidad de transmitir voz a baja velocidad con alta calidad. Se enmarca en un tipo de codificadores denominados híbridos. El funcionamiento básico de este tipo de codificadores consta de dos etapas:

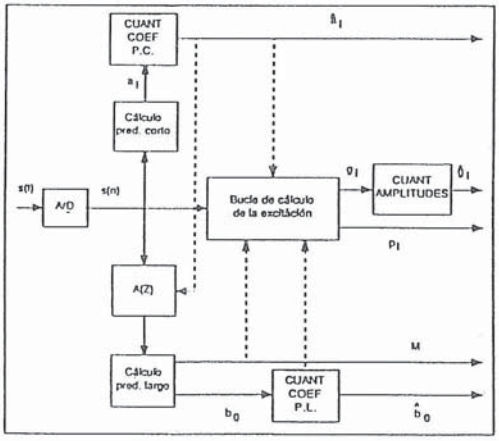
En una primera etapa se analiza la señal de voz para extraer una serie de parámetros mediante los cuales se construirá un filtro todo-polos. Mediante la excitación de este filtro con una señal adecuada se obtendrá la señal de voz sintetizada.

La segunda etapa consiste en hallar la excitación que produzca un menor error perceptual en la señal sintetizada, es decir, se realiza un análisis por síntesis.

El objetivo del presente artículo es describir el funcionamiento del codificador multipulso, la cuantificación utilizada y su implementación sobre un DSP (Digital Speech Procesor) hasta lograr la ejecución en tiempo real.

CODIFICADOR MULTIPULSO

El esquema del codificador de voz multipulso cuantificado sería el siguiente [1]:

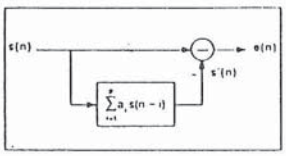


Este se puede dividir en tres etapas:

Cálculo del Predictor Corto A(z):

La señal de voz es muestreada a 8 KHz y dividida en tramas de 25 ms por una ventana de Hamming. A continuación se realiza una parametrización LPC de diez coeficientes. Los coeficientes LAR de dicha parametrización se cuantifican mediante un sistema de cuantificación vectorial de cuatro etapas con la primera de ellas adaptativa (AMSVQ) [2]. Esta cuantificación conlleva una dificultad añadida en cuanto a que aumenta en gran medida el tiempo de ejecución, debido al gran número de operaciones que es necesario realizar. La actualización de estos coeficientes (a_i) se realiza cada 20 ms.

El esquema del Predictor Corto es el siguiente:



$$e(n) = s(n) - \sum_{i=1}^{p} a_i\, s(n-i) \qquad \text{con } P=10$$

para calcular los coeficientes se minimiza la expresión $E[e^2(n)]$, cuyo desarrollo suponiendo estacionariedad y ergodicidad nos lleva al sistema de ecuaciones:

$$\sum_{i=1}^{p} a_i R_m(i-j) = R_m(j) \qquad j = 1, \ldots, p \qquad R_m(\tau) = \sum_{n=m}^{\bar{m}} s(n)\, s(n-\tau)$$

los límites del sumatorio se acotan mediante la ventana de Hamming.

Cálculo del Predictor Largo:

Para eliminar la redundancia de pitch existente en la señal de salida de la primera etapa se realiza una parametrización de un coeficiente. Esta se cuantifica mediante un cuantificador escalar. La actualización de este coeficiente se realiza cada 5 ms la amplitud (b_0) y cada 10 ms la posición (M).

El esquema del Predictor Largo en análisis es totalmente análogo al del Predictor Corto. Su ecuación de diseño es la siguiente:

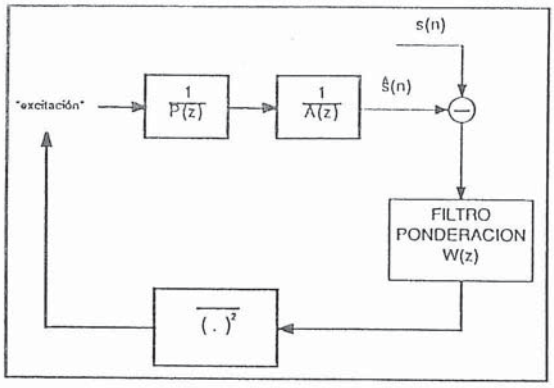
$d(n) = e(n) - b_0 * e(n-M)$ con M=pitch y e(n) el residuo de predicción corta

b_0 se calcula de forma que minimice la potencia del error de predicción y para el cálculo de M se utiliza el método de minimización del residuo. El cálculo de M conlleva un aumento considerable del tiempo de ejecución ya que se calcula entre 128 posibles valores.

Bucle de cálculo de la Excitación Multipulso

En esta etapa se calcula la señal de excitación en un bucle de análisis por síntesis donde se minimiza un error cuadrático medio perceptual. Esta excitación consta de 4 pulsos en 40 posibles posiciones. Para el cálculo de estos pulsos se utiliza un método iterativo en el cual en cada iteración se calcula la posición (p_i) y amplitud (g_i) de un nuevo pulso dejando fijos los pulsos anteriormente hallados. Al finalizar la búsqueda de la excitación se realiza una reoptimización final de amplitudes.

El esquema de cálculo de la excitación multipulso es el siguiente:



$$E = \sum_{n=0}^{N-1} [\, (s(n) - \hat{s}(n)) * w(n) \,]^2$$

$$excit(n) = \sum_{i=1}^{Q} g_i \, \delta(n - p_i)$$

$$W(z) = \frac{A(z)}{A(z/\gamma)} = \frac{1 - \sum_{i=1}^{p} a_i z^{-i}}{1 - \sum_{i=1}^{p} a_i \gamma^i z^{-i}}$$

W(z) se encarga de permitir más ruido en las zonas en las que existe más señal y menos ruido en las zonas en las que existe menos señal, de esta forma se mejora la sensación auditiva.

Para el cálculo de excit(n) se minimiza la expresión E. Esta señal de excitación se recalcula cada 5 ms. La cuantificación utilizada es del sistema ganancia-forma. En concreto para la forma se utiliza el cuantificador de normalización de signo y cuantificación vectorial de las amplitudes en valor absoluto. Para la cuantificación de la ganancia se utiliza un cuantificador de 5 bits de Ley A.

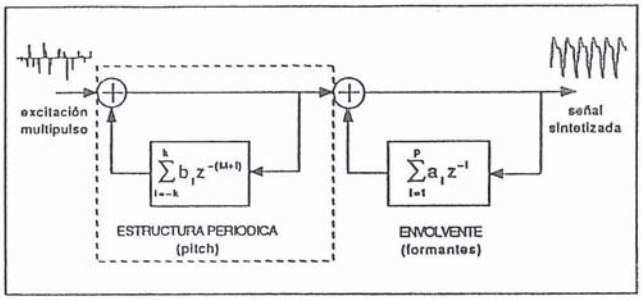
Por último, en este apartado dedicado al codificador multipulso, hallaremos la velocidad de transmisión obtenida a partir de los distintos métodos de cuantificación propuestos para cada etapa.

VALORES A ENVIAR	PERIODO ACT. (ms)	Nº DE BITS	VELOCIDAD (bps)
10 coef. predictor corto	20	27	1350
10 coef. predictor largo	5	3	600
retardo predictor largo	10	7	700
ganancia excitación	5	5	1000
amplitudes pulsos excit.	5	12	2400
posiciones pulsos excit.	5	17	3400

La velocidad de transmisión resultante es de 9450 bps.

DECODIFICADOR MULTIPULSO

El esquema del decodificador multipulso será:



MULTIPULSO EN SIMULACION

En una primera fase de implementación se realizó un sistema que trabajaba en simulación de forma que el algoritmo fuese idéntico al que se utilizaría posteriormente en la implementación en tiempo real. Se pudo comprobar el correcto funcionamiento del algoritmo y una SNR de 15 dB (aprox.) sin cuantificar con una pérdida de 1 dB cuantificando las tres etapas de análisis.

MULTIPULSO EN TIEMPO REAL

Para lograr la realización en tiempo real se seleccionó el DSP TI TMS320C30, debido a su demostrada eficacia en otro tipo de codificadores (Vocoder LPC-10, ...) en cuanto a su velocidad de cálculo.

Algunas de las características más importantes de este DSP son las siguientes [3,4]:

- ciclo de instrucción de 60 ns lo cual permite 33,3 MFLOPS de pico
- dos bloques de memoria RAM interna lo cual permite realizar dos accesos por ciclo de instrucción
- pipeline de 4 niveles, el procesador puede ejecutar 4 operaciones consecutivas en paralelo en 4 unidades de hardware diferentes. Para el correcto aprovechamiento del pipeline se tendrá que programar el DSP de forma adecuada
- realización de bucles con la instrucción RPTB y RPTS sin romper el pipeline, además al utilizar estas instrucciones el procesador no tiene que ir a buscar la instrucción contenida en el bucle a una memoria externa, sino que la tiene almacenada internamente, mejorando con ello la velocidad

Además la placa sobre la que se encuentra el DSP incorpora memoria de cero estados de espera.

El primer paso en la implementación en tiempo real fue la realización de un estudio minucioso del algoritmo con objeto de simplificarlo al máximo (realizando las mismas funciones) para ganar tiempo de ejecución. Esta fase se realizó siempre teniendo en cuenta las especiales características del DSP.

Una vez optimizado al máximo el algoritmo, mediante medidas de tiempo se averiguó cuales eran las partes críticas del programa en cuanto a tiempo de ejecución (bien por complejidad de cálculo o bien por ser partes que se realizan en multitud de ocasiones). De esta forma estas partes pueden ser optimizadas en assembler y además se puede distribuir la memoria disponible de forma más eficaz.

Los resultados obtenidos pusieron de manifiesto que gran parte del tiempo de ejecución era utilizado por rutinas que realizaban productos escalares, filtrados y comparaciones. Entre todas estas rutinas destacaba por su elevadísimo número de operaciones la cuantificación vectorial del predictor corto, esta se realiza en 4 etapas, la primera de ellas de 8 codewords y las tres restantes de 256 codewords de dimensión 10.

Para la optimización en assembler de esta rutina se utiliza la instrucción del procesador producto-acumulación en paralelo (programando en C el compilador no la aprovecha); los bucles son realizados mediante la instrucción RPTS; no se desplazan ni copian elementos de matrices sino que se trabaja con apuntadores; se utiliza una instrucción existente a la hora de realizar comparaciones-decisiones que permite aprovechar los tres ciclos de instrucción que programando en C se pierden (pipeline), estos se utilizan para comenzar a realizar el siguiente cálculo.

El tiempo de ejecución de la rutina se divide por 10.

Este método de optimización es extrapolable a otras rutinas críticas.

CONCLUSIONES

Se han obtenido dos claras conclusiones:

1ª) El codificador de voz MULTIPULSO se adapta perfectamente a un tipo de codificador de complejidad media, velocidad de transmisión baja y alta calidad. La SNR obtenida oscila entre 13 y 15 dB para distintas señales de voz. La velocidad de transmisión obtenida es de aproximadamente 9600 bps. La calidad subjetiva se puede calificar de muy buena.

2ª) El DSP TI TMS320C30 se adapta perfectamente a las necesidades requeridas a la hora de realizar una implementación de una aplicación de procesado de voz en tiempo real. Sirva de referencia que el programa multipulso codificador-decodificador en una primera versión en C tardaba en procesar una trama de 20 ms aproximadamente 205 ms. Con un estudio de las partes críticas del programa y su posterior optimización en assembler se ha logrado un tiempo de ejecución de 14 ms por trama, con lo cual quedan 6 ms por trama suficientes para realizar posibles empaquetamientos de datos para transmitir los distintos parámetros por distintas redes de comunicación.(Nota: en un PC-286 convencional el procesado de una señal de voz de 1 segundo puede durar 20 horas).

REFERENCIAS

[1] Honold Ripoll: "Diseño de un codificador de voz LPC de excitación multipulso a 9.6 Kbps", P.F.C. Universidad Politécnica de Cataluña, Diciembre 1989.
[2] Rodríguez Fonollosa "Cuantificación vectorial adaptativa aplicada a la codificación de voz" Tesis Doctoral. Universidad Politécnica de Cataluña, 1989.
[3] "TMS320C30 C Compiler Reference Guide", TI 1991.
[4] "Third Generation TMS320C3X User's Guide", TI 1989.