UNIVERSIDAD DE SEVILLA

FACULTAD DE CIENCIAS FÍSICAS

TESIS DOCTORAL

UNA APORTACIÓN AL DISEÑO DE FILTROS DE BLOQUES BIDIMENSIONALES

Autor: CARLOS ROMERO PÉREZ

Director: JOSÉ IGNACIO ACHA CATALINA

1997

UNIVERSIDAD DE SEVILLA

FACULTAD DE CIENCIAS FÍSICAS

Dpto. Electrónica y Electromagnetismo

TESIS DOCTORAL

UNA APORTACIÓN AL DISEÑO DE FILTROS DE BLOQUES BIDIMENSIONALES

Autor: CARLOS ROMERO PÉREZ

Director: JOSÉ IGNACIO ACHA CATALINA

1997

Tesis Doctoral: UNA APORTACIÓN AL DISEÑO DE FILTROS DE BLOQUES BIDIMENSIONALES

Autor: CARLOS ROMERO PÉREZ

Director: JOSÉ IGNACIO ACHA CATALINA

Tutor: ANTÓN CIVIT BREU

El tribunal nombrado para juzgar la tesis doctoral arriba citada, compuesto por los señores

Presidente:

Vocales:

Secretario:

Acuerda otorgarle la calificación de

Sevilla, a de de 1997

A mis padres, Estanislao y Clara

AGRADECIMIENTOS

Quisiera agradecer al Dr. José Ignacio Acha Catalina la ayuda que me ha brindado desde el momento en que accedió a dirigir esta tesis, así como la paciencia demostrada al intentar hacer de mi un investigador y su preocupación por mi futuro profesional.

Los últimos años de elaboración de esta tesis los he efectuado en el Grupo de Teoría de la Señal y Comunicaciones de la Escuela de Ingenieros donde el Dr. Justo Calvo, Dr. Javier Payán, Dr. Carlos Crespo, J. I. Falgueras, y Álvaro me han tratado como a un compañero más; y en especial a Sergio Cruces tengo que agradecerle que hasta el último momento me ha facilitado la terminación de la tesis poniendo a mi disposición su mesa de trabajo y ordenador.

En la Facultad de Físicas, agradezco al Dr. Antón Civit el haberse ofrecido como Tutor de mi Tesis.

La elaboración de una Tesis requiere la máxima dedicación, por lo que compatibilizar la misma con el trabajo desarrollado en SAINCO no hubiera sido posible sin la libertad de horario y movimientos que me han permitido todos mis jefes (Dr. José Antonio Moreno, José Pinilla) y compañeros (Manolo Reina, Arturo Martín, Enrique Ramos y Alfonso Domínguez).

En los viajes por Congresos que he tenido que realizar a Madrid no me ha faltado nunca el cobijo de mis amigos, Cristina (Señora de Sevilla) y Juanjo

(Señor de Pérez), y de los Asturianines Beatriz y Carlos (Dr. Carlos) Señores de Bousoño.

Finalmente, las personas que más han soportado todos los desvelos, agobios, contrariedades y demás penurias implícitas a una larga elaboración de Tesis, y sin cuyo apoyo no habría podido llegar hasta este punto, son mis padres, Estanislao y Clara, abuelo, Dr. José Escobar (tío Pepe, para entendernos) , hermana, Clara (Dr. Clara Romero, redactora de algún que otro documento), y por supuesto mi Anita (llamada en otras Tesis Anita y Olé, que me ha ayudado en la redacción del trabajo y a superar días conflictivos).

RESUMEN

En esta tesis se presenta una nueva aproximación para la obtención de la matriz de bloques de filtros digitales bidimensionales. A partir del concepto de descomposición polifásica bidimensional, se deriva la expresión general para la matriz de bloques de sistemas LTI bidimensionales, y se propone un algoritmo general de cálculo para las componentes polifásicas, independientemente del tipo de matriz de submuestreo utilizada y si la función escalar es o no de variables separables.

Se generaliza el uso de algoritmos rápidos de convolución lineal para la implementación en bloques de filtros digitales bidimensionales. La particularidad de la matriz de bloques de ser Toeplitz en 2 niveles permite obtener estructuras computacionales con reducido costo hardware, aplicando de forma iterada algoritmos de convolución monodimensionales.

Se propone una nueva estructura para la implementación de filtros digitales monodimensionales y bidimensionales con respuesta impulsiva finita, basadas en la teoría de filtros FIR interpolados. La estructura obtenida, denominada Estructura Mixta, consigue reducir el tiempo de procesamiento de la implementación con un costo en el número de procesadores significativamente menor que la implementación en bloques equivalente. Se

deducen expresiones para el orden de interpolación que minimiza el tiempo de procesamiento de las diferentes implementaciones. Se generaliza este tipo de implementación para el caso de filtros con respuesta impulsiva infinita.

Por último, se demuestra la estabilidad de las implementaciones en bloques al probar que si el filtro escalar es estable siempre existe, al menos, una implementación en bloques estable, para cualquier matriz de submuestreo empleada.

ÍNDICE GENERAL

PARTE I: INTRODUCCIÓN, REVISIÓN Y MOTIVACIONES

PARTE II: PROPUESTAS, EJEMPLOS, APLICACIONES, CONCLUSIONES Y LÍNEAS FUTURAS DE TRABAJO

PARTE I

INTRODUCCIÓN, REVISIÓN Y MOTIVACIONES

CAPÍTULO 1

INTRODUCCIÓN

1.1 INTRODUCCIÓN

El procesado digital de señales está presente en multitud de campos de la ciencia y la tecnología, dentro de los cuales existe una gran diversidad de aplicaciones en las que es imprescindible el procesamiento en tiempo real. Una de las técnicas dirigidas a aumentar la velocidad de procesamiento en la implementación de filtros digitales, logra su objetivo incrementando el denominado *throughput rate,* utilizando procesamiento en paralelo. Estructuras típicas englobadas dentro de esta técnica son la implementación multicaminos (*L-paths*) e implementación en bloques.

La historia del procesamiento en paralelo de los filtros digitales es relativamente reciente. Aun así, se pueden distinguir cuatro etapas cronológicas que sintetizan la evolución de dicha técnica.

1.1.1 Estructuras Recursivas en Variables de Estado

En sus orígenes, los primeros estudios sobre el tema fueron efectuados por [Gold y Jordan 68], y a partir de ellos otros autores propusieron una serie de estructuras, implementadas en variables de estado, para filtros monodimensionales con respuesta impulsiva infinita, que permitían procesar en paralelo un número determinado de muestras. No existía una técnica estandarizada para efectuar la implementación en bloques del filtro escalar. La elección de una u otra estructura se basaba en criterios de complejidad computacional, sensibilidad o efecto de la cuantización de los coeficientes. A pesar de todo, ya se conocía una propiedad importante de la implementación en bloques, demostrada por [Mitra y Gnanasekaran 78], consistente en que si los polos del sistema escalar están dentro de la circunferencia unidad, los polos de la estructura en bloques no solo se garantiza que estén dentro de la circunferencia unidad, sino que se aproximan hacia el origen, asegurando la estabilidad de la implementación en bloques.

1.1.2 Sistemas Invariantes al Desplazamiento por Bloques

La etapa que vamos a describir es, bajo nuestro punto de vista, clave en la evolución del concepto de implementación en bloques de filtros digitales. En los trabajos efectuados por [Barnes y Shinnaka 80], basándose en sistemas invariantes al desplazamiento por bloques, muestran de forma explícita una

expresión para la matriz de bloques en términos de la función de transferencia escalar. Se demuestra que la matriz de bloques es Toeplitz, y relacionan las componentes de la matriz de bloques con la función de transferencia escalar a través de la fórmula integral de Cauchy. Vuelven a demostrar, pero de forma general, la propiedad de que si λ es un polo de la función de transferencia escalar, entonces λ^L es un polo de la función de transferencia en bloques de longitud L. Otra propiedad interesante que demuestran es que la función de transferencia escalar puede obtenerse de forma única a partir de cualquier fila o columna de la matriz de bloques, propiedad que años más tarde se utilizará conjuntamente con el concepto de descomposición polifásica del filtro escalar para obtener un método de cálculo de las componentes de la matriz de bloques.

A pesar de los importantes logros teóricos obtenidos, en especial la relación entre las componentes de la matriz de bloques y la función de transferencia escalar, la implementación en bloques de los filtros digitales se efectúan en variables de estado.

1.1.3 Relación entre Matriz de Bloques y Descomposición Polifásica

El siguiente paso en este desarrollo histórico es la implementación en bloques a partir de los coeficientes de la matriz de bloques. De esta forma, para efectuar la implementación en bloques de un filtro digital no se implementa en variables de estado, sino que se utilizan los coeficientes de la matriz de bloques.

En [Hayashi et al, 86] se proponen dos estructuras para implementación en multicaminos, que formalmente es una técnica de procesamiento en paralelo análoga a la implementación en bloques, de las cuales la denominada aproximación por multicaminos retrasados identifica los filtros empleados en la implementación con las componentes polifásicas del filtro escalar.

Por otro lado, [Chwen y Alexander 87] efectúan un estudio paralelo al realizado por [Barnes y Shinnaka 80] para la implementación en bloques de filtros multidimensionales. Generalizan los resultados obtenidos por sus predecesores al caso N-dimensional, e identifican los coeficientes de la matriz de bloques con las componentes de un tipo de descomposición polifásica del filtro escalar.

1.1.4 Aplicación de la Teoría de Algoritmos Rápidos de Convolución

Reducido el problema de la implementación en bloques al cálculo de las componentes polifásicas del filtro escalar, se plantea el problema de la reducción del costo hardware que conlleva. La solución se obtuvo aplicando la teoría de algoritmos rápidos de convolución [Blahut 85]. Concretamente, [Acha 89] consigue reducir drásticamente el número de procesadores utilizados para la implementación en bloques de filtros digitales monodimensionales, aplicando los algoritmos de Winograd para convolución lineal.

1.2 ORGANIZACIÓN DE LA TESIS

Hemos estructurado esta tesis en 5 capítulos, de los cuales el primero de ellos lo hemos dedicado a mostrar, desde el punto de vista del autor, una breve historia sobre el tema central de la misma. El resto de los capítulos están organizados como indicamos en las siguientes secciones.

1.2.1 Capítulo 2, Revisión Teórica

Durante el desarrollo de la tesis se utilizan una serie de resultados teóricos obtenidos por otros autores. Aunque están debidamente referenciados, hay un conjunto de ellos que, por su especial importancia, hemos creído conveniente transcribir. Los resultados que mencionamos se refieren a propiedades de la matriz de bloques, estabilidad y aplicación de algoritmos rápidos de convolución para su implementación.

1.2.2 Capítulo 3, Motivaciones

Este capítulo pone de manifiesto los principales inconvenientes que plantea la implementación en bloques de filtros digitales. Se marcan una serie de objetivos para solucionar, ó aliviar, los problemas planteados, los cuales se abordarán en el capítulo posterior.

1.2.3 Capítulo 4, Propuestas

Es el capítulo central de esta tesis, y en el que tratamos de alcanzar los objetivos marcados en el capítulo 3. Está dividido en 7 apartados, de los cuales, el primero de ellos se dedica a realizar una pequeña introducción. A continuación se profundiza en el concepto de matriz de bloques de filtros bidimensionales. Una vez generalizado este concepto, se aplica la teoría de algoritmos rápidos de convolución para su implementación. En el siguiente apartado se propone un nuevo tipo de implementación, que aumenta la velocidad de procesamiento de sistemas monodimensionales y bidimensionales, con un costo significativamente menor en el número de procesadores empleados que la implementación en bloques equivalentes. En el penúltimo apartado de este capítulo se muestran de forma comparativa diversos resultados de diseño. Finalmente, se añade un anexo destinado a demostrar la estabilidad de la implementación en bloques.

1.2.4 Capítulo 5, Aplicaciones, Conclusiones y Líneas Futuras de Trabajo

En este capítulo se muestran algunas aplicaciones en las que tendrían cabida las propuestas efectuadas. Así mismo, se presentan unas conclusiones, y para finalizar, se proponen una serie de líneas futuras de trabajo originadas en la elaboración de esta tesis.

CAPÍTULO 2

REVISIÓN

2.1 INTRODUCCIÓN

En el desarrollo de la tesis se utilizan una serie de resultados obtenidos por otros autores. En este capítulo mostraremos de forma pormenorizada algunos de estos resultados que, si bien pueden encontrarse en los artículos referenciados, por la importancia de su significado creemos interesante resaltar. En particular, haremos hincapié en las propiedades de la matriz de bloques, mostraremos unos ejemplos de algoritmos Winograd para convolución lineal, y recordaremos el principio de transformación y su aplicación en la implementación de la matriz de bloques de filtros digitales monodimensionales.

2.2 PROPIEDADES DE LA MATRIZ DE BLOQUES

La descripción cronológica efectuada en el primer capítulo señala el trabajo efectuado por [Barnes y Shinnaka 80] como fundamental en el desarrollo teórico sobre implementación en bloques de filtros digitales

monodimensionales. Años más tarde [Chwen y Alexander 87] generalizan los resultados obtenidos por sus antecesores para el caso de filtros N-dimensionales. En el desarrollo teórico efectuado en ambos trabajos se parte de la hipótesis de que el sistema digital, monodimensional o multidimensional, cumple la propiedad de invariancia al desplazamiento por bloques. Parte de las propiedades que vamos a señalar son debidas a esta característica del sistema, que es determinante para la reducción del número de procesadores requeridos en la implementación en bloques del filtro digital.

2.2.1 Invariancia al Desplazamiento por Bloques

Un sistema N-dimensional multientradas y multisalidas (MIMO) puede representarse en función de sus entradas y salidas como

$$\bar{y}(\vec{n}) = \sum_{\vec{m}=0}^{\infty} \bar{h}(\vec{n},\vec{m}) \quad \bar{x}(\vec{m}) \tag{2.2.1}$$

donde $\bar{x}(\vec{m}) \in R^{\vec{W}_i^{\&}}$, $\bar{y}(\vec{n}) \in R^{\vec{W}_o^{\&}}$ y $\bar{h}(\vec{n},\vec{m}) \in R^{\vec{W}_0^{\&},\vec{W}_i^{\&}}$, siendo $\vec{W}_k = (W_{k1}, W_{k2}, \cdots, W_{kN})$, $\vec{W}_k^{\&} = \prod_{l=1}^{N} W_{kl}$, y $R^{i,j}$ el conjunto de matrices reales de dimensiones $i \times j$. En el caso de que $\vec{W}_o = (W_{o1}, W_{o2}, \cdots, W_{oN}) \neq (W_{i1}, W_{i2}, \cdots, W_{iN}) = \vec{W}_i$, el filtro de bloques se denomina multitasa, mientras que si $\vec{W}_o = \vec{W}_i = \vec{W}$ el sistema se denomina de razón constante. Un sistema (MIMO) de razón constante, con $\vec{W}^{\&}$ entradas y

$\vec{W}^{\&}$ salidas, se dice que es invariante al desplazamiento por bloques si los elementos de la matriz $\bar{h}(\vec{n},\vec{m})$ cumplen la propiedad

$$h_{\vec{r}}^{\vec{s}}(\vec{n},\vec{m}) = h\left((\vec{n}-\vec{m})\otimes\vec{W}+\vec{s}-\vec{r}\right) \tag{2.2.2}$$

donde $\otimes$ es el operador $\vec{m}\otimes\vec{n} = (m_1n_1, m_2n_2, \cdots, m_Nn_N)$, siendo $\vec{W} = (W_1, W_2, \cdots, W_N)$, y donde $h_{\vec{r}}^{\vec{s}}(\vec{n},\vec{m})$ es elemento situado en la fila $s_1 + W_1s_2 + \cdots + W_1W_2\cdots W_{N-1}s_N$, y columna $r_1 + W_1r_2 + \cdots + W_1W_2\cdots W_{N-1}r_N$.

Si $\bar{h}(\vec{n},\vec{m})$ cumple la condición (2.2.2), entonces se puede identificar el término

$$\bar{h}(\vec{n},\vec{m}) = \bar{h}(\vec{n}-\vec{m}) \tag{2.2.3}$$

expresión equivalente a la invariancia al desplazamiento en sistema escalares.

De la definición de invariancia al desplazamiento por bloques se deduce que la matriz de bloques de un sistema invariante al desplazamiento por bloques es una matriz Toeplitz en N niveles. En el capítulo 4 demostramos esta propiedad para filtros de bloques bidimensionales, sin necesidad de recurrir al concepto de invariancia al desplazamiento por bloques.

2.2.2 Propiedades

La transformada Z multidimensional de los elementos de la matriz de bloques vendrán dados por la expresión

$$H_{\vec{r}}^{\vec{s}}(\vec{Z}) = \sum_{\vec{m}=0}^{\infty} h_{\vec{r}}^{\vec{s}}(\vec{m})\vec{Z}^{-\vec{m}} = \sum_{\vec{m}=0}^{\infty} h(\vec{m}\otimes\vec{W}+\vec{s}-\vec{r})\,\vec{Z}^{-\vec{m}} \qquad (2.2.4)$$

que no es más que la representación de los coeficientes de un tipo de descomposición polifásica, a la que denominamos descomposición polifásica rectangular. Aplicando la definición de transformada inversa al término $h(\vec{m}\otimes\vec{W}+\vec{s}-\vec{r})$ obtenemos la expresión

$$H_{\vec{r}}^{\vec{s}}(\vec{Z}) = \left(\frac{1}{2\pi i}\right)^{N} \sum_{\vec{m}=0}^{\infty} \oint_{\vec{C}} H(\vec{\xi}) \prod_{j=1}^{N} \xi_j^{m_j W_j + s_j - r_j - 1} d\xi_j \; \vec{Z}^{-\vec{m}} \qquad (2.2.5)$$

Intercambiando el orden del sumatorio, y efectuándolo para cada valor de $\vec{m}$, obtenemos la siguiente relación entre los coeficientes de la matriz de bloques y la función de transferencia escalar

$$H_{\vec{r}}^{\vec{s}}(\vec{Z}) = \left(\frac{1}{2\pi i}\right)^{N} \oint_{\vec{C}} H(\vec{\xi}) \prod_{j=1}^{N} \frac{d\xi_j}{1 - z_j^{-1}\xi_j^{W_j}} \vec{\xi}^{\,s_j - r_j - 1} \qquad (2.2.6)$$

donde $\vec{\xi} = (\xi_1, \xi_2, \cdots, \xi_N)$ son variables complejas que recorren el contorno cerrado $\vec{C}$ en la región de convergencia que engloba el origen de coordenadas.

El valor de $H_r^{\vec{s}}(\vec{Z})$ (2.2.6) vendrá dado por el sumatorio de residuos de la función dentro del recinto cerrado $\vec{C}$, es decir

$$H_r^{\vec{s}}(\vec{Z}) = \sum_P \mathrm{Res}\left(H(\vec{\xi}) \prod_{j=1}^{N} \frac{\vec{\xi}^{s_j - r_j - 1}}{1 - z_j^{-1} \xi_j^{W_j}} \right) \tag{2.2.7}$$

donde P es el conjunto de polos de la función de transferencia escalar $H(\vec{Z})$ dentro del contorno cerrado $\vec{C}$. El residuo de esta función en $(\lambda_1, \lambda_2, \cdots, \lambda_N)$ provoca la aparición de N denominadores con ceros situados en $z_j = \lambda_j^{W_j}$, lo que demuestra que $\left(\lambda_1^{W_1}, \lambda_2^{W_2}, \cdots, \lambda_N^{W_N}\right)$ es polo de $H_r^{\vec{s}}(\vec{Z})$.

Esta propiedad tiene una importante repercusión. Nos asegura que si el filtro escalar es estable entonces la implementación en bloques también lo es. Además, el acercamiento de los polos al origen hace que la implementación en bloques tenga mejor comportamiento en lo referente a ciclos límite, así como menor sensibilidad en los coeficientes y al ruido por redondeo.

Es interesante destacar como la expresión (2.2.5) demuestra que dada una longitud de bloque $\vec{W}$, existe una única función de transferencia N-

dimensional en bloques que pueda ser generada a partir de la función de transferencia escalar $H(\vec{Z})$.

Por último, la ecuación (2.2.4) puede ser expresada como

$$H_{\vec{r}}^{\vec{s}}(\vec{Z}) = \vec{Z}^{\vec{s}-\vec{r}} \sum_{\vec{m}=0}^{\infty} h(\vec{m} \otimes \vec{W} + \vec{s} - \vec{r}) \vec{Z}^{-(\vec{m} \otimes \vec{W} + \vec{s} - \vec{r})} = \vec{Z}^{\vec{s}-\vec{r}} \sum_{\vec{k}=0}^{\infty} h(\vec{k}) \vec{Z}^{-\vec{k}} \sum_{\vec{m}=0}^{\infty} \vec{\delta}(\vec{k} - \vec{m} \otimes \vec{W} - \vec{s} + \vec{r}) \tag{2.2.8}$$

donde δ es la delta de Kronecker N-dimensional. Teniendo en cuenta que

$$\sum_{\vec{r}=0}^{\vec{L}-\vec{I}} \sum_{\vec{m}=0}^{\infty} \vec{\delta}(\vec{k} - \vec{m} \otimes \vec{W} - \vec{s} + \vec{r}) = 1 \tag{2.2.9}$$

donde $\vec{I}$ es el vector unidad N-dimensional, multiplicando ambos términos de la expresión (2.2.8) por $\vec{Z}^{\vec{r}-\vec{s}}$, y sumando sobre la variable $\vec{r}$, relacionamos la función de transferencia escalar con los elementos de la matriz de bloque como indica la expresión

$$H(\vec{Z}) = \sum_{\vec{r}=0}^{\vec{L}-\vec{I}} \vec{Z}^{\vec{r}-\vec{s}} H_{\vec{r}}^{\vec{s}}(\vec{Z}^{\vec{L}}) \tag{2.2.10}$$

2.3 ALGORITMOS DE WINOGRAD PARA CONVOLUCIÓN LINEAL

La relación entre secuencia de salida y secuencia de entrada en un sistema LTI causal monodimensional viene dado por la siguiente expresión

$$y(n) = \sum_{m=0}^{\infty} h(n-m)\, x(m) \tag{2.3.1}$$

relación denominada convolucional. En el caso de que ambas secuencias, h(n) y x(n), sean de longitud finita, el producto de convolución es equivalente a realizar el producto polinomial de los polinomios $X(z^{-1})$ y $H(z^{-1})$

$$\begin{aligned} X(z^{-1}) &= x(0) + z^{-1}x(1) + \cdots + z^{-N_x}x(N_x - 1) \\ H(z^{-1}) &= h(0) + z^{-1}h(1) + \cdots + h^{-N_h}h(N_h - 1) \end{aligned} \tag{2.3.2}$$

Aprovechando esta equivalencia, diversos autores han desarrollado algoritmos para convolución lineal. Entre estos algoritmos destacamos los algoritmos de Winograd [Blahut 87] por ser algoritmos óptimos en lo referente al número de multiplicaciones empleados. En las dos próximas secciones vamos a mostrar dos algoritmos que utilizaremos extensamente a lo largo del desarrollo de la tesis.

2.3.1 Algoritmo de Convolución Lineal 2 x 2

El algoritmo que vamos a mostrar efectúa la convolución de los polinomios

$$d = d(0) + d(1)z^{-1}$$
$$g = g(0) + g(1)z^{-1} \qquad (2.3.3)$$

La secuencia de salida $s = d * g$, donde $*$ es el operador convolución, se calcularía según la definición de convolución con 4 productos y 1 adición. El algoritmo de Winograd es el siguiente

$$\begin{bmatrix} s(0) \\ s(1) \\ s(2) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} g(0) & & \\ & g(0)+g(1) & \\ & & g(1) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} d(0) \\ d(1) \end{bmatrix} \qquad (2.3.4)$$

donde los términos de la diagonal principal se conocen de antemano, no teniendo que efectuar la adición. El número de operaciones aritméticas que se efectúan con este algoritmo es de 3 productos y 3 adiciones. El número de operaciones ha aumentado respecto al cálculo directo, pero el número de productos ha disminuido, propiedad que se utilizará en el próximo apartado.

2.3.2 Algoritmo de Convolución Lineal 3 x 3

El algoritmo que mostramos a continuación efectúa la convolución de dos polinomios de orden 2

$$d = d(0) + d(1)z^{-1} + d(2)z^{-2}$$
$$g = g(0) + g(1)z^{-1} + g(2)z^{-2} \tag{2.3.5}$$

que de forma directa necesitaría un total de 9 productos y 4 adiciones. El algoritmo Winograd es el siguiente

$$\begin{bmatrix} s(0) \\ s(1) \\ s(2) \\ s(3) \\ s(4) \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ -1 & 2 & -2 & -1 & 2 \\ -2 & 1 & 3 & 0 & -1 \\ 1 & -1 & -1 & 1 & -2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2}g(0) & & & & \\ & \frac{1}{2}(g(0)+g(1)+g(2)) & & & \\ & & \frac{1}{6}(g(0)-g(1)+g(2)) & & \\ & & & \frac{1}{6}(g(0)+2g(1)+4g(2)) & \\ & & & & g(0) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 2 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d(0) \\ d(1) \\ d(2) \end{bmatrix} \tag{2.3.6}$$

con un total de 5 multiplicaciones y 20 adiciones distribuidas como

Preadiciones

$t_1 = d(1) + d(2)$
$t_2 = d(2) - d(1)$
$D_0 = d(0)$
$D_1 = d(0) + t_1$
$D_2 = d(0) + t_2$
$D_3 = t_1 + t_1 + t_2 + D_1$
$D_4 = d(2)$

Postadiciones

$s(0) = S_0 + S_0$
$T_1 = S_1 + S_1$
$T_2 = S_2 + S_2$
$T_3 = S_4 + S_4$
$T_4 = T_3 - S_0 - S_3$
$T_5 = S_1 + S_2$
$s(1) = T_1 - T_2 + T_4$
$s(2) = -s(0) + T_2 + T_5 - S_4$
$s(3) = -T_4 - T_5$
$s(4) = S_4$

2.4 IMPLEMENTACIÓN EN BLOQUES CON ALGORITMOS DE CONVOLUCIÓN LINEAL

La implementación en bloques de filtros digitales permite aumentar la velocidad de procesamiento de un sistema, procesando en paralelo un número determinado de muestras. Concretamente, en el caso monodimensional la expresión que relaciona los vectores de entrada y salida es

$$\begin{bmatrix} Y_0(z) \\ Y_1(z) \\ \vdots \\ Y_{L-1}(z) \end{bmatrix} = \begin{bmatrix} H_0(z) & z^{-1}H_{L-1}(z) & \cdots & z^{-1}H_1(z) \\ H_1(z) & H_0(z) & & z^{-1}H_2(z) \\ \vdots & & & \vdots \\ H_{L-1}(z) & H_{L-2}(z) & \cdots & H_0(z) \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \\ \vdots \\ X_{L-1}(z) \end{bmatrix} \qquad (2.4.1)$$

donde los elementos de los vectores de entrada y salida, y los coeficientes de la matriz de bloques son las componentes polifásicas de la secuencia de entrada, salida y sistema, respectivamente.

$$X(z) = \sum_{k=0}^{L-1} X_k(z^L)\, z^{-k}$$
$$Y(z) = \sum_{k=0}^{L-1} Y_k(z^L)\, z^{-k} \qquad (2.4.2)$$
$$H(z) = \sum_{k=0}^{L-1} H_k(z^L)\, z^{-k}$$

La expresión (2.4.1) sugiere que para efectuar la implementación en bloques de longitud L son necesarios L^2 procesadores. Análogamente, en el caso multidimensional una implementación en bloques de un sistema MIMO, de razón constante $\vec{L} = (L_1, L_2, \cdots, L_N)$, requiere $\left(\prod_{j=1}^{N} L_j \right)^2$ procesadores para su implementación.

La solución, en el caso monodimensional, para reducir el número de procesadores fue propuesta por [Acha 89]. En este trabajo se utilizan los algoritmos de convolución lineal para implementar la matriz de bloques de filtros digitales monodimensionales, aprovechando que los algoritmos de Winograd no utilizan la operación de división ni la propiedad conmutativa de la multiplicación.

2.4.1 Principio de Transformación

Al aplicar los algoritmos de convolución lineal en la implementación de la matriz de bloques se hace uso del principio de transformación. Este principio establece que [Acha 89] dado un algoritmo

$$s = T\,d = C\,G\,A\,d \tag{2.4.3}$$

donde G es una matriz diagonal, y A y C son matrices de coeficientes constantes, entonces

$$e = A^t\,G\,C^t\,f \tag{2.4.4}$$

es un algoritmo para calcular $e = T^t f$, donde el superíndice "t" indica que es la matriz traspuesta. Si el algoritmo (2.4.3) es óptimo, respecto al número de multiplicaciones, entonces (2.4.4) también es óptimo.

2.4.2 Reducción del Número de Procesadores

Como mencionamos en el apartado 2.3, los algoritmos de Winograd para convolución lineal requieren mayor número de operaciones aritméticas que el cálculo directo de la expresión (2.3.1). Sin embargo, tienen la propiedad de ser algoritmos óptimos en lo referente al número de multiplicaciones efectuadas.

multiplicaciones también es mínimo. A continuación transcribimos la obtención de una de las estructuras derivadas en [Acha 89], aplicando un algoritmo de convolución Winograd 2 x 2 a la implementación en bloques de longitud L=2 de un filtro monodimensional.

Partiendo del algoritmo de convolución lineal 2 x 2 (2.3.4), y aplicando el principio de transformación obtenemos el siguiente algoritmo traspuesto

$$\begin{bmatrix} e(0) \\ e(1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} g(0) & & \\ & g(0)+g(1) & \\ & & g(1) \end{bmatrix} \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \end{bmatrix} \qquad (2.4.5)$$

que puede ser expresado como

$$\begin{bmatrix} e(0) \\ e(1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} f(0)-f(1) & & \\ & f(1) & \\ & & f(2)-f(1) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} g(0) \\ g(1) \end{bmatrix} \qquad (2.4.6)$$

que de forma compacta puede expresarse como

$$\begin{bmatrix} e(0) \\ e(1) \end{bmatrix} = \begin{bmatrix} f(1) & f(0) \\ f(2) & f(1) \end{bmatrix} \begin{bmatrix} g(1) \\ g(0) \end{bmatrix} \qquad (2.4.7)$$

Teniendo en cuenta la expresión de la implementación en bloques de longitud L=2

Teniendo en cuenta la expresión de la implementación en bloques de longitud L=2

$$\begin{bmatrix} Y_0(z) \\ Y_1(z) \end{bmatrix} = \begin{bmatrix} H_0(z) & z^{-1}H_1(z) \\ H_1(z) & H_0(z) \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \end{bmatrix} \tag{2.4.8}$$

identificando miembros con (2.4.7)

$$\begin{aligned} &e(0) = Y_0(z) \qquad g(1) = X_0(z) \qquad e(1) = Y_1(z) \qquad g(0) = X_1(z) \\ &f(1) = H_0(z) \qquad f(0) = z^{-1}H_1(z) \qquad f(2) = H_1(z) \end{aligned} \tag{2.4.9}$$

obtenemos la siguiente expresión para la implementación en bloques de longitud L=2

$$\begin{bmatrix} Y_0(z) \\ Y_1(z) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} z^{-1}H_1(z) - H_0(z) & & \\ & H_0(z) & \\ & & H_1(z) - H_0(z) \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \end{bmatrix} \tag{2.4.10}$$

que puede esquematizarse como se indica en la figura 2.1.

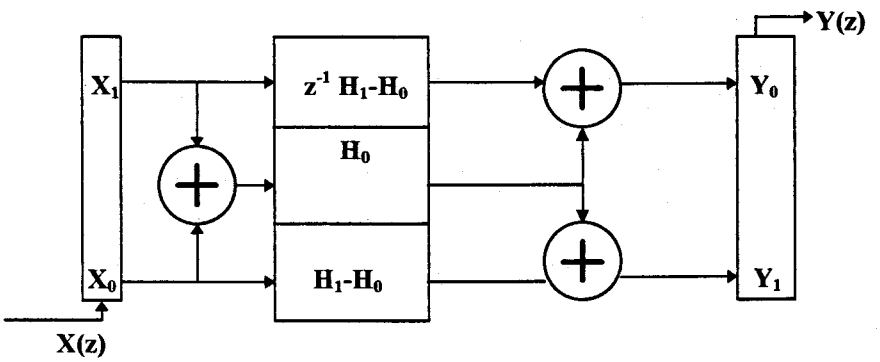

Figura 2.1

2.4.3 Estructuras Iteradas

Aunque es posible utilizar algoritmos de convolución de órdenes superiores para obtener estructuras con mayor longitud de bloque, el elevado número de adiciones hace conveniente el uso de algoritmos de convolución no óptimos. En [Acha 89] se propone la aplicación de algoritmos de Winograd mediante iteración. Por ejemplo, para implementar en bloques de longitud L=4 un filtro digital monodimensional

$$\begin{bmatrix} Y_0(z) \\ Y_1(z) \\ Y_2(z) \\ Y_3(z) \end{bmatrix} = \begin{bmatrix} H_0(z) & z^{-1}H_3(z) & z^{-1}H_2(z) & z^{-1}H_1(z) \\ H_1(z) & H_0(z) & z^{-1}H_3(z) & z^{-1}H_2(z) \\ H_2(z) & H_1(z) & H_0(z) & z^{-1}H_3(z) \\ H_3(z) & H_2(z) & H_1(z) & H_0(z) \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \\ X_2(z) \\ X_3(z) \end{bmatrix} \qquad (2.4.11)$$

puede utilizarse un algoritmo óptimo de Winograd, con 7 multiplicaciones, que se traducen en 7 procesadores, y 41 adiciones [Blahut 87]. Sin embargo, observando la expresión (2.4.11) vemos que es posible reescribirla como

$$\begin{bmatrix} \Psi_0 \\ \Psi_1 \end{bmatrix} = \begin{bmatrix} H_0 & H_2 \\ H_1 & H_0 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} \tag{2.4.12}$$

siendo

$$\Psi_0 = \begin{bmatrix} Y_0(z) \\ Y_1(z) \end{bmatrix} \qquad \Psi_1 = \begin{bmatrix} Y_2(z) \\ Y_3(z) \end{bmatrix} \qquad X_0 = \begin{bmatrix} X_0(z) \\ X_1(z) \end{bmatrix} \qquad X_1 = \begin{bmatrix} X_2(z) \\ X_3(z) \end{bmatrix}$$

$$H_0 = \begin{bmatrix} H_0(z) & z^{-1}H_3(z) \\ H_1(z) & H_0(z) \end{bmatrix} \quad H_1 = \begin{bmatrix} H_2(z) & H_1(z) \\ H_3(z) & H_2(z) \end{bmatrix} \quad H_2 = \begin{bmatrix} z^{-1}H_2(z) & z^{-1}H_1(z) \\ z^{-1}H_3(z) & z^{-1}H_2(z) \end{bmatrix}$$

(2.4.13)

La matriz de la expresión (2.4.12) es una matriz Toeplitz, similar a la expresión (2.4.7), por lo que podemos emplear el mismo algoritmo para su implementación. En este caso, las adiciones del algoritmo se convierten en adiciones matriciales, de dimensiones 2x1, mientras que los productos son a su vez productos matriciales de la forma (2.4.7), pudiendo de nuevo utilizar el mismo algoritmo para su implementación (Figura 2.2)

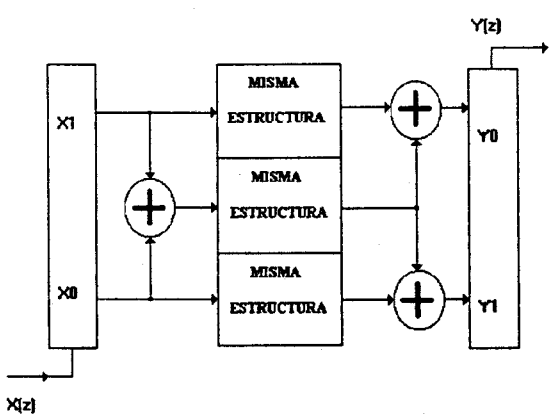

Figura 2.2

El número de procesadores es superior al requerido utilizando un algoritmo de Winograd, 9 procesadores frente a 7, sin embargo el número de adiciones se decrementa significativamente, 15 adiciones frente a 41.

2.5 RESUMEN

En este capítulo se ha recordado el concepto de invariancia al desplazamiento por bloques. Como consecuencia de este concepto se muestran cuatro propiedades, deducidas por otros autores, de gran interés para el desarrollo de esta tesis. En primer lugar, se menciona la particularidad de la matriz de bloques de ser Toeplitz en N niveles. En segundo lugar, se demuestra la estabilidad de las componentes de la matriz de bloques, así como mejor comportamiento frente a ciclos límites y menor sensibilidad a la cuantización de los coeficientes. Se demuestra que, escogida una longitud de bloque, la matriz de bloques que se obtiene es única. Por último, se deduce una expresión que relaciona la función de transferencia escalar con los elementos de la matriz de bloques.

Implementar físicamente la matriz de bloques requeriría un elevado costo hardware. En el caso monodimensional este problema se ha resuelto, en gran medida, derivando estructuras con menor complejidad hardware a partir de algoritmos de convolución lineal. Se muestran dos algoritmos de Winograd

para convolución lineal 2 x 2 y 3 x 3 que, al ser óptimos respecto al número de multiplicaciones, generan estructuras óptimas respecto al número de procesadores. Se describe el principio de transformación y su aplicación para obtener estructuras en bloques de filtros digitales.

Finalmente, se describen las estructuras iteradas. Estas estructuras no son óptimas respecto al número de procesadores empleados. Sin embargo, permiten obtener implementaciones en bloques de elevada longitud de bloque, que requerirían algoritmos de convolución de difícil diseño y elevado costo en el número de adiciones, a partir de estructuras de pequeña longitud.

CAPÍTULO 3

MOTIVACIONES

3.1 INTRODUCCIÓN

El objetivo de este capítulo consiste en plantear una serie de problemáticas que surgen tras el análisis del actual estado de conocimiento sobre la teoría y aplicación de implementaciones en bloques de filtros digitales. Hemos dividido el capítulo en tres apartados, correspondientes a las líneas de investigación que se han desarrollado en la elaboración de esta tesis.

3.2 GENERALIZACIÓN DEL CONCEPTO DE IMPLEMENTACIÓN EN BLOQUES

La construcción de la teoría de filtros de bloques se ha basado en la propiedad de invariancia al desplazamiento por bloques. A partir de ella se demuestra que la matriz de bloques es una matriz Toeplitz, pudiendo aplicar, en el caso monodimensional, los algoritmos de convolución lineal para su implementación y reducir así el número de procesadores requeridos. La exigencia de esta propiedad restringe, en el caso multidimensional, las

estructuras obtenidas a un caso particular de implementaciones en bloques, denominadas rectangulares.

3.2.1 Nueva Aproximación a la Teoría de Filtros de Bloques

Con la idea de generalizar la implementación en bloques de filtros digitales multidimensionales, y en particular bidimensionales, con matrices de submuestreo arbitrarias, es necesario obtener una nueva expresión para la matriz de bloques que no se derive de la propiedad de invariancia al desplazamiento por bloques. La matriz obtenida debe ser equivalente, en el caso de submuestreos rectangulares, a la matriz derivada por [Chwen y Alexander 87], conservando todas las propiedades deducidas anteriormente.

3.2.2 Algoritmo de Cálculo de las Componentes Polifásicas

Los elementos de la matriz de bloques se identifican con las componentes de la descomposición polifásica rectangular del filtro digital. De esta forma, el problema de encontrar los elementos de la matriz de bloques se reduce a encontrar las componentes polifásicas. En el caso de filtros con respuesta impulsiva finita no ofrece ninguna dificultad, pero en el caso de filtros con respuesta impulsiva infinita la dificultad aumenta a medida que aumenta la longitud del bloque. Este problema se multiplica para el caso de

descomposiciones polifásicas multidimensionales de filtros IIR con denominador de variables no separables.

De las relaciones existentes entre las componentes polifásicas rectangulares y las componentes de modulación es posible encontrar un método de cálculo de las componentes polifásicas. Sin embargo, basándonos en estas equivalencias, y generalizándolas para descomposiciones polifásicas no rectangulares, vamos a proponer un algoritmo de cálculo que requiere menor costo computacional. Como se muestra en el apéndice final del capítulo 4, el algoritmo de cálculo que hemos desarrollado permite demostrar la estabilidad de los elementos de la matriz de bloques, independientemente de la matriz de submuestreo utilizada.

3.3 REDUCCIÓN DEL NÚMERO DE PROCESADORES

Deducida la matriz de bloques general para filtros bidimensionales, un objetivo primordial consiste en la reducción del número de procesadores requeridos para la implementación en bloques del filtro. En el caso monodimensional la adaptación de algoritmos óptimos de Winograd para convolución lineal consigue reducir significativamente el número de procesadores utilizados. Sin embargo, en el caso bidimensional no tenemos constancia de haberse obtenido ninguna solución equivalente.

3.3.1 Aplicación de Algoritmos de Convolución Lineal

Por este motivo, basándonos en los resultados obtenidos para implementación en bloques de filtros monodimensionales, generalizamos la aplicación de los algoritmos de convolución lineal a la implementación en bloques, con matriz de submuestreo arbitraria, de filtros bidimensionales.

3.3.2 Reducción del Número de Operaciones Aritméticas

A diferencia del caso monodimensional, la longitud del bloque, es decir el número de muestras procesadas en paralelo, no determina unívocamente la estructura, sino que existen infinitas soluciones que permiten alcanzar el throughput rate requerido. Pero, incluso escogida una matriz de submuestreo y los algoritmos de convolución, el orden de aplicación de los algoritmos conducen a diferentes estructuras. Aunque el número de procesadores es el mismo, el número de adiciones en el pre y post procesado varía según el orden utilizado. Esta circunstancia será demostrada en el capítulo 4, permitiéndonos optimizar las estructuras obtenidas mediante aplicación de algoritmos de convolución lineal.

3.4 ESTRUCTURAS HÍBRIDAS

La aplicación de algoritmos de convolución para la implementación en bloques de filtros digitales resuelve, en gran medida, el principal inconveniente objetable a la utilización de este tipo de procesamiento, al disminuir el costo hardware de la implementación. No obstante, el número de procesadores sigue siendo elevado, sobretodo en el caso bidimensional. Desgraciadamente, la teoría de algoritmos de convolución impone un número mínimo de productos al efectuar la convolución de dos secuencias, lo que se traduce en un número mínimo de procesadores al utilizar esta técnica.

3.4.1 Nuevas Estructuras para Filtros FIR

El incremento de velocidad de procesamiento de un sistema no solo se consigue mediante técnicas de procesamiento en paralelo. En el proceso de diseños de filtros existen gran variedad de procedimientos para obtener filtros con reducida complejidad aritmética, factor determinante a la hora de conseguir implementaciones más veloces. Dentro de las técnicas destinadas al diseño de filtros con respuesta impulsiva finita destacamos los filtros obtenidos mediante interpolación (IFIR) [Neuvo et al, 84] [Neuvo et al, 87], que consiguen reducir drásticamente el número de operaciones aritméticas de la implementación escalar.

La especial construcción de este tipo de filtros permite realizar una estructura híbrida, formada por la cascada de un filtro de bloques y un filtro escalar, con velocidad de procesamiento similar a las obtenidas con estructuras en bloques convencionales y con menor costo hardware, a la que denominaremos implementación mixta.

3.4.2 Nuevas Estructuras para Filtros IIR

Conceptualmente es posible diseñar filtros con respuesta impulsiva infinita mediante interpolación, aunque los resultados obtenidos difieren considerablemente respecto a los filtros IFIR. Sin embargo, las estructuras obtenidas al implementar el filtro IIR interpolado con implementación mixta permiten, sobretodo en el caso monodimensional, aumentar la velocidad de procesamiento del sistema reduciendo el número de procesadores respecto a la implementación en bloques convencional.

3.5 RESUMEN

En este capítulo se definen las líneas de investigación seguidas en el desarrollo de esta tesis, pudiendo dividir la temática de las mismas en tres grupos. La primera línea de investigación consiste en la generalización del concepto de implementación en bloques de filtros digitales multidimensionales,

y en particular bidimensionales, al caso de implementaciones en bloques no rectangulares. Para ello, se efectuará una nueva aproximación a la teoría de filtros de bloques no basada en la propiedad de invariancia al desplazamiento por bloques. Los elementos de la matriz de bloques, obtenida con la nueva aproximación, se identifican con las componentes de descomposiciones polifásicas no rectangulares, tal como se demostrará en el capítulo 4. Por este motivo, se desarrollará un algoritmo de cálculo de componentes polifásicas que sea independiente de la matriz de submuestreo escogida, y de que el filtro escalar sea o no de variables separables.

La segunda línea de investigación propone la utilización de algoritmos de convolución lineal, tal como se efectúa en el caso monodimensional, para la obtención de estructuras bidimensionales con reducido número de procesadores. Las estructuras que se estudiarán serán las denominadas iteradas.

Las estructuras obtenidas al utilizar algoritmos de convolución lineal requieren un mínimo de procesadores. Esta particularidad suscita la tercera línea de investigación, consistente en la derivación de nuevas estructuras para filtros con respuesta impulsiva finita, basadas en la teoría de filtros IFIR, que permitan obtener implementaciones con tiempos de procesamiento análogos a los conseguidos con implementación en bloques, pero con un costo hardware inferior. Esta línea de investigación no solo trataría la implementación de los

filtros digitales, sino que englobaría también la etapa de diseño. Finalmente, se propone la aplicación de este método de diseño e implementación para el caso de filtros con respuesta impulsiva infinita.

PARTE II

PROPUESTAS, EJEMPLOS, APLICACIONES, CONCLUSIONES

Y LÍNEAS FUTURAS DE TRABAJO

CAPÍTULO 4

PROPUESTAS

4.1 INTRODUCCIÓN

El elevado costo computacional que requiere el procesado lineal de secuencias bidimensionales justifica el uso de técnicas de procesamiento en paralelo para aumentar la velocidad de procesamiento del sistema. Si bien el tratamiento conceptual fue establecido hace ya una década [Chwen y Alexander 87], hasta la fecha no se han efectuado estudios dirigidos a la reducción del costo hardware que conlleva este tipo de implementación.

Siguiendo la línea marcada en el capítulo dedicado a las motivaciones de la tesis, hemos dividido este capítulo en 8 secciones. La sección 4.2 generaliza el concepto de matriz de bloques bidimensional, permitiendo procesar secuencias bidimensionales empleando muestreos arbitrarios. En la sección 4.3 se aplican algoritmos de convolución para obtener estructuras con reducido costo hardware. En las secciones 4.4 y 4.5 se plantea un nuevo tipo de arquitectura que reduce la velocidad de procesamiento del sistema así como el número de procesadores requeridos. En la sección 4.6 se muestran algunos resultados de diseño que permiten comparar las diferentes implementaciones. La sección 4.7

se incluye como anexo al resto del capítulo, demostrando la estabilidad de las implementaciones en bloques. Finalmente, en la sección 4.8 se resume el contenido del capítulo.

4.2 MATRIZ DE BLOQUES BIDIMENSIONAL

En el capítulo 2 revisamos la construcción de sistemas en bloques multidimensionales a partir del concepto de invariancia al desplazamiento por bloques [Chwen y Alexander 87]. Se mostró cómo la implementación en bloques de un sistema escalar estable es también estable, con menor sensibilidad en los coeficientes y al ruido por redondeo. Teniendo en cuenta que la definición de invariancia al desplazamiento por bloques implica que la matriz de bloques de un sistema N-Dimensional es una matriz Toeplitz en N niveles, junto con la relación entre el filtro escalar y las componentes de la matriz de bloques a través de la fórmula integral de Cauchy, se deriva un método para la obtención de las componentes de la matriz de bloques a partir del filtro escalar. Como mencionamos en el capítulo 2, la demostración efectuada por [Barnes y Shinnaka 80] para el caso monodimensional, y posteriormente [Chwen y Alexander 87] para el caso multidimensional, en la que la función de transferencia escalar puede ser derivada a partir de cualquier fila o columna de la matriz de bloques, establece una relación entre las componentes de la matriz de bloques y las componentes de un tipo de descomposición polifásica, llamada descomposición rectangular. Con esta

relación, la obtención de la matriz de bloques se limita al cálculo de la descomposición polifásica rectangular.

Partiendo de la idea de procesamiento en bloques bidimensional, vamos a derivar una expresión general de la matriz de bloques, así como la relación entre las componentes de la matriz de bloques y las componentes polifásicas. Mostraremos como, a diferencia del caso monodimensional, el sistema escalar puede tener múltiples descomposiciones polifásicas equivalentes, existiendo una relación entre las distintas componentes polifase. Finalmente, propondremos un algoritmo general para el cálculo de las componentes de la matriz de bloques, independientemente de que el sistema escalar sea o no de variables separables.

4.2.1 Submuestreo Bidimensional

El concepto de submuestreo juega un papel muy importante en muchas aplicaciones del procesamiento digital, englobadas bajo el nombre de sistemas multitasa [Vaidyanathan 90], tales como bancos de filtros, codificación en sub-bandas, modificación de frecuencia de muestreo, donde es básico el uso de las operaciones denominadas *diezmado* e *interpolación*. También es de gran importancia dentro del procesamiento en paralelo, multicaminos y procesamiento en bloques, donde la conversión entre sistema escalar/vectorial y viceversa se efectúa a través de un sistema de diezmadores e interpoladores respectivamente. No obstante, la importancia de este concepto no se limita a su

empleo como operador, sino que sirve como representación de sistemas y señales digitales. Es precisamente este motivo por el que incluimos un apartado referido al concepto de submuestreo bidimensional.

Una secuencia bidimensional, cuya representación en el dominio transformado viene dado por la expresión

$$X(z_1,z_2)=\sum_{n_1=-\infty}^{\infty}\sum_{n_2=-\infty}^{\infty}x(n_1,n_2)\,z_1^{-n_1}\,z_2^{-n_2} \qquad (4.2.1)$$

puede ser interpretada de forma abstracta como la unión de un número finito o infinito de muestras, $x(n_1,n_2)$. El conjunto de puntos que definen esta secuencia forma un reticulado en el espacio de los índices n_1 x n_2. Un submuestreo de esta secuencia bidimensional puede ser visto como una aplicación lineal desde el reticulado origen al reticulado de submuestreo [Vetterli y Karlsson 90], donde la aplicación viene definida por la matriz de submuestreo de coeficientes enteros

$$M=\begin{bmatrix} m_{00} & m_{01} \\ m_{10} & m_{11} \end{bmatrix} \qquad (4.2.2)$$

de tal forma que las muestras pertenecientes al reticulado de muestreo, a las que denominamos $x^p(u_1, u_2)$, están relacionadas con las muestras del reticulado origen como indica la relación entre índices

$$\begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = M \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{4.2.3}$$

representando el par ordenado (n_1,n_2) una posición en el reticulado origen y el par (u_1,u_2) en el reticulado de muestreo.

Atendiendo al valor de los coeficientes de la matriz de submuestreo podemos distinguir tres casos. El caso más general es cuando todos los coeficientes son distintos de cero, al que llamamos submuestreo *hexagonal*. Cuando $m_{01} = m_{10} = 0$ la operación que se efectúa es un escalado, denominándolo submuestreo *rectangular*. Finalmente, el caso intermedio se produce cuando uno de los coeficientes, bien m_{01} o m_{10}, es cero. En este último caso uno de los ejes del retículo de submuestreo es colineal con uno de los ejes del retículo origen, por lo que lo denominamos submuestreo *colineal*.

El conjunto de puntos que forman la secuencia de submuestreo genera un sub-retículo dentro del retículo origen. Este enrejado no tiene una descripción única, sino que puede ser descrito por distintas matrices de submuestreo, todas ellas relacionadas entre sí mediante matrices de enteros con determinante unidad [Vetterli y Karlsson 90]. Otra propiedad interesante de la matriz de submuestreo es el significado del determinante

$$|M| = m_{00}m_{11} - m_{01}m_{10} \tag{4.2.4}$$

que indica el número de muestras de la secuencia origen incluidas dentro de una celda imaginaria del retículo de submuestreo. Todas las matrices de submuestreo relacionadas por matrices de enteros con determinante unidad tienen el mismo determinante, pero el hecho de que dos matrices de submuestreo tengan el mismo determinante no es condición suficiente para definir el mismo reticulado.

4.2.2 Descomposición Polifásica Bidimensional

Como mencionamos anteriormente, una secuencia bidimensional puede ser vista como la unión de un número finito o infinito de muestras, descripción que puede ser expresada en el dominio transformado (4.2.1). Sin embargo, cualquier descripción es válida siempre que englobe el conjunto de muestras que conforman la secuencia en su totalidad. De esta forma, podemos expresar la secuencia como unión de un número finito de sub-secuencias disjuntas, donde cada sub-secuencia es una secuencia submuestra de la secuencia original. Concretamente, en el caso general de submuestreos hexagonales (4.2.2), se puede expresar la secuencia original en función de las secuencias submuestras, en el dominio transformado, como

$$X(z_1, z_2) = \sum_{(k,l)\in M(k,l)} X_{k,l}(z_1, z_2) \tag{4.2.5}$$

siendo $M(k,l)$ el conjunto de pares ordenados, (k,l), incluidos dentro de la celda imaginaria formada por los pares ordenados $(0,0)$, (m_{00},m_{10}), (m_{01},m_{11}), $(m_{00}+m_{01}\ ,\ m_{10}+m_{11})$, y donde los términos $X_{k,l}$ (z_1,z_2) representan las transformadas de las secuencias submuestra. El número total de sub-secuencias viene dado por el determinante de la matriz de submuestreo, $|M|$.

Teniendo en cuenta la relación entre índices (4.2.3), se pueden expresar las transformadas de las secuencias submuestra como

$$X_{k,l}(z_1,z_2)=\sum_{u_1=-\infty}^{\infty}\sum_{u_2=-\infty}^{\infty}x_{k,l}^{p}(u_1,u_2)z_1^{-(u_1m_{00}+u_2m_{01})}z_2^{-(u_1m_{10}+u_2m_{11})}z_1^{-k}z_2^{-l} \qquad (4.2.6)$$

donde

$$x_{k,l}^{p}(u_1,u_2)=x(k+u_1m_{00}+u_2m_{01},l+u_1m_{10}+u_2m_{11}) \qquad (4.2.7)$$

y reordenando la expresión (4.2.6)

$$X_{k,l}(z_1,z_2)=z_1^{-k}z_2^{-l}\sum_{u_1=-\infty}^{\infty}\sum_{u_2=-\infty}^{\infty}x_{k,l}^{p}(u_1,u_2)\left(z_1^{m_{00}}z_2^{m_{10}}\right)^{-u_1}\left(z_1^{m_{01}}z_2^{m_{11}}\right)^{-u_2} \qquad (4.2.8)$$

relacionamos las transformadas Z de las secuencias submuestra con las muestras que las componen. El término formado por el doble sumatorio no es

más que la definición de transformada Z de las secuencias $x_{k,l}^{p}(u_1,u_2)$ a las que denominamos *secuencias polifase*, es decir

$$X_{k,l}^{p}(\xi_1,\xi_2)=\sum_{u_1=-\infty}^{\infty}\sum_{u_2=-\infty}^{\infty}x_{k,l}^{p}(u_1,u_2)\,\xi_1^{-u_1}\,\xi_2^{-u_2} \tag{4.2.9}$$

donde las variables complejas ξ_1 , ξ_2 , están asociadas a la razón de muestreo representada por los índices u_1, u_2. De esta forma, identificando miembros en (4.2.8), vemos que

$$X_{k,l}(z_1,z_2)=z_1^{-k}\,z_2^{-l}\,X_{k,l}^{p}(z_1^{m_{00}}z_2^{m_{10}},z_1^{m_{01}}z_2^{m_{11}}) \tag{4.2.10}$$

y sustituyendo en (4.2.5)

$$X(z_1,z_2)=\sum_{(k,l)\in M(k,l)}z_1^{-k}\,z_2^{-l}\,X_{k,l}^{p}(z_1^{m_{00}}z_2^{m_{10}},z_1^{m_{01}}z_2^{m_{11}}) \tag{4.2.11}$$

conseguimos una nueva representación de la secuencia origen, en el dominio transformado, en función de las secuencias polifase. La representación de la secuencia mostrada en (4.2.11) la denominamos *descomposición polifásica asociada a la matriz de submuestreo M.*

4.2.3 Descomposiciones Polifásicas Equivalentes

Aunque la descomposición polifásica asociada a una matriz M es única (4.2.11), el retículo de submuestreo definido por la matriz M, y por lo tanto la composición de las secuencias polifase, puede venir descrito por cualquier matriz de enteros P relacionada con M mediante una matriz C

$$\begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} \\ m_{10} & m_{11} \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} \tag{4.2.12}$$

cuyo determinante valga

$$c_{00}c_{11} - c_{10}c_{01} = \pm 1 \tag{4.2.13}$$

En ambas representaciones hay el mismo número de secuencias polifase, dado por el determinante de las matrices de submuestreo, pero además, al generar ambas matrices el mismo retículo de submuestreo, las muestras que componen las distintas secuencias polifase son idénticas. Conocida la expresión de las componentes polifásicas asociadas a una matriz de submuestreo, M, conocemos la expresión para cualquier descomposición asociada a otra matriz, P, siempre que se cumplan las relaciones (4.2.12) y (4.2.13). Efectivamente, supongamos que una muestra del espacio origen, $x(n_1, n_2)$, pertenece a la

secuencia polifásica $x^{P}_{k,l}(u_1,u_2)$ asociada a la matriz M, y a la secuencia polifásica $x^{P}_{q,r}(v_1,v_2)$ asociada a la matriz P, entonces de la relación entre índices

$$\begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} \\ m_{10} & m_{11} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} k \\ l \end{bmatrix}$$
$$\begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = \begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} + \begin{bmatrix} q \\ r \end{bmatrix} \qquad (4.2.14)$$

y (4.2.12) deducimos

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix}^{-1} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix}^{-1} \begin{bmatrix} k-q \\ l-r \end{bmatrix} \qquad (4.2.15)$$

Por lo tanto, conocida la expresión de las secuencias polifase asociadas a una matriz M, (4.2.9), la expresión de las mismas para cualquier matriz Pque defina el mismo retículo de submuestreo es

$$X^{P}_{q,r}(\xi_1,\xi_2) = \sum_{u_1=-\infty}^{\infty} \sum_{u_2=-\infty}^{\infty} x^{P}_{k,l}(u_1,u_2)\, \xi_1^{-v_1}\, \xi_2^{-v_2} \qquad (4.2.16)$$

donde

$$v_1 = \frac{c_{11}u_1 - c_{01}u_2}{|C|} + \frac{1}{|M|}\left(p_{11}(k-q) - p_{01}(l-r)\right)$$
$$v_2 = \frac{-c_{10}u_1 + c_{00}u_2}{|C|} + \frac{1}{|M|}\left(-p_{10}(k-q) + p_{00}(l-r)\right) \tag{4.2.17}$$

que puede expresarse como

$$X^P_{q,r}(\xi_1,\xi_2) = X^P_{k,l}(\xi_1^{\frac{c_{11}}{C}}\xi_2^{-\frac{c_{10}}{C}}, \xi_1^{-\frac{c_{01}}{C}}\xi_2^{\frac{c_{00}}{C}})\xi_1^{-\frac{(p_{11}(k-q)-p_{01}(l-r))}{M}}\xi_2^{-\frac{(-p_{10}(k-q)+p_{00}(l-r))}{M}} \tag{4.2.18}$$

A continuación vamos a demostrar una propiedad de las matrices de submuestreo de la forma (4.2.2), que nos permitirá reducir el estudio al caso de matrices de submuestreo colineales, sin quitar generalidad a los resultados que obtengamos. Para ello demostraremos que dada una matriz de submuestreo arbitraria *P*, siempre es posible encontrar una matriz de submuestreo colineal, *M*, de la forma

$$M = \begin{bmatrix} m_{00} & 0 \\ m_{10} & m_{11} \end{bmatrix} \tag{4.2.19}$$

que define el mismo retículo de submuestreo. La relación de partida

$$\begin{bmatrix} m_{00} & 0 \\ m_{10} & m_{11} \end{bmatrix} = \begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} \tag{4.2.20}$$

junto con la condición (4.2.13) implica que

$$(c_{00}\, p_{00} + c_{10}\, p_{01})(c_{01}\, p_{10} + c_{11}\, p_{11}) = |M| \tag{4.2.21}$$

$$p_{00}\, c_{01} + p_{01}\, c_{11} = 0 \tag{4.2.22}$$

La ecuación (4.2.22) permite expresar c_{01} en función de c_{11}, y sustituyendo en (4.2.21)

$$(c_{00}p_{00} + c_{10}p_{01})(-c_{11}\frac{p_{10}p_{01}}{p_{00}} + c_{11}p_{11}) = |M| \tag{4.2.23}$$

es decir

$$(c_{00}p_{00} + c_{10}p_{01}) = \pm\frac{p_{00}}{c_{11}} \tag{4.2.24}$$

El valor de los coeficientes de la matriz de enteros de determinante unidad depende de la relación entre p_{00} y p_{01}. En el caso de que p_{01} sea múltiplo de p_{00}, es decir $p_{01} = K\,p_{00}$, el valor de los coeficientes de la matriz C es

$$C = \begin{bmatrix} \pm 1 - K & -K \\ 1 & 1 \end{bmatrix} \tag{4.2.25}$$

si, por el contrario, es p_{00} múltiplo de p_{01} entonces

$$C = \begin{bmatrix} 1 & -1 \\ \pm 1 - K & K \end{bmatrix} \tag{4.2.26}$$

Finalmente, en el caso de que p_{00} y p_{01} no sean múltiplos, llamando K al máximo común divisor de ambos términos

$$MCD(s,t) = a\,s + b\,t$$

4.2.4 Procesamiento Bidimensional en Bloques

El propósito de este apartado es deducir la expresión de la matriz de bloques a partir de la definición de procesamiento en bloques, partiendo de la suposición de que el sistema escalar, $H(z_1, z_2)$, es un sistema LTI.

El objetivo principal del procesamiento en bloques es aumentar la velocidad de procesamiento de un sistema, $H(z_1,z_2)$, procesando en paralelo un número determinado de muestras, es decir, incrementando el denominado *throughput rate*. La descomposición polifásica de una secuencia discreta (4.2.11) describe la secuencia como suma, o superposición, de un número finito de secuencias, llamadas componentes polifásicas. El número de componentes polifásicas viene dado por el determinante de la matriz de submuestreo asociada, $|M|$. Como vimos en el apartado anterior, siempre es posible encontrar una matriz de submuestreo colineal de la forma (4.2.19), que descomponga la secuencia en función de $|M|$ secuencias polifase. Teniendo en cuenta (4.2.11) y la relación general para sistemas LTI

$$Y(z_1,z_2) = H(z_1,z_2)X(z_1,z_2) \tag{4.2.30}$$

se pueden representar las secuencias $Y(z_1,z_2)$, $H(z_1,z_2)$ y $X(z_1,z_2)$ como

$$Y(z_1,z_2)=\sum_{k=0}^{m_{00}-1}\sum_{l=\left\lceil \frac{m_{10}}{m_{00}}k\right\rceil}^{\left\lceil \frac{m_{10}}{m_{00}}k+m_{11}\right\rceil-1}Y_{k,l}^{p}(z_1^{m_{00}}z_2^{m_{10}},z_2^{m_{11}})z_1^{-k}z_2^{-l} \tag{4.2.31}$$

$$H(z_1,z_2)=\sum_{k=0}^{m_{00}-1}\sum_{l=\left\lceil \frac{m_{10}}{m_{00}}k\right\rceil}^{\left\lceil \frac{m_{10}}{m_{00}}k+m_{11}\right\rceil-1}H_{k,l}^{p}(z_1^{m_{00}}z_2^{m_{10}},z_2^{m_{11}})z_1^{-k}z_2^{-l} \tag{4.2.32}$$

$$X(z_1,z_2)=\sum_{k=0}^{m_{00}-1}\sum_{l=\left\lceil \frac{m_{10}}{m_{00}}k\right\rceil}^{\left\lceil \frac{m_{10}}{m_{00}}k+m_{11}\right\rceil-1}X_{k,l}^{p}(z_1^{m_{00}}z_2^{m_{10}},z_2^{m_{11}})z_1^{-k}z_2^{-l} \tag{4.2.33}$$

donde el operador $\lceil * \rceil$ indica el mayor entero más próximo. Para facilitar la notación reescribimos (4.2.31), (4.2.32) y (4.2.33) como

$$Y(z_1,z_2)=\sum_{k=0}^{m_{00}-1}Y_k z_1^{-k} \tag{4.2.34}$$

$$H(z_1,z_2)=\sum_{k=0}^{m_{00}-1}H_k z_1^{-k} \tag{4.2.35}$$

$$X(z_1,z_2)=\sum_{k=0}^{m_{00}-1}X_k z_1^{-k} \tag{4.2.36}$$

donde

$$Y_k = \sum_{l=\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil}^{\left\lceil \frac{m_{10}}{m_{00}}k+m_{11} \right\rceil -1} Y_{k,l}^{p}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}) z_2^{-l} \tag{4.2.37}$$

$$H_k = \sum_{l=\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil}^{\left\lceil \frac{m_{10}}{m_{00}}k+m_{11} \right\rceil -1} H_{k,l}^{p}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}) z_2^{-l} \tag{4.2.38}$$

$$X_k = \sum_{l=\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil}^{\left\lceil \frac{m_{10}}{m_{00}}k+m_{11} \right\rceil -1} X_{k,l}^{p}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}) z_2^{-l} \tag{4.2.39}$$

Sustituyendo (4.2.34), (4.2.35) y (4.2.36) en (4.2.30), tenemos

$$\sum_{k=0}^{m_{00}-1} Y_k z_1^{-k} = \sum_{k=0}^{m_{00}-1} H_k z_1^{-k} \sum_{k=0}^{m_{00}-1} X_k z_1^{-k} \tag{4.2.40}$$

que puede ponerse como

$$\sum_{k=0}^{m_{00}-1} Y_k z_1^{-k} = \sum_{k=0}^{m_{00}-1} \left(\sum_{k_1=0}^{k} H_{k-k_1} X_{k_1} + z_1^{-m_{00}} \sum_{k_1=k+1}^{m_{00}-1} H_{m_{00}-k_1+k} X_{k_1} \right) z_1^{-k} \tag{4.2.41}$$

y en forma matricial

$$\begin{bmatrix} Y_0 \\ Y_1 \\ \vdots \\ \vdots \\ Y_{m_{00}-1} \end{bmatrix} = \begin{bmatrix} H_0 & z_1^{-m_{00}} H_{m_{00}-1} & z_1^{-m_{00}} H_{m_{00}-2} & \cdots & z_1^{-m_{00}} H_1 \\ H_1 & H_0 & z_1^{-m_{00}} H_{m_{00}-1} & \cdots & z_1^{-m_{00}} H_2 \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ H_{m_{00}-1} & H_{m_{00}-2} & \cdots & \cdots & H_0 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ \vdots \\ X_{m_{00}-1} \end{bmatrix} \tag{4.2.42}$$

Como puede apreciarse, la matriz (4.2.42) es una matriz Toeplitz, donde los elementos que la forman son vectores (4.2.38). Cada uno de los productos de vectores $H_{k2}\, X_{k1}$ pueden a su vez escribirse como

$$H_{k_2} X_{k_1} = z_2^{-\left(\left\lceil \frac{m_{10}}{m_{00}} k_1 \right\rceil + \left\lceil \frac{m_{10}}{m_{00}} k_2 \right\rceil\right)} \sum_{l=0}^{m_{11}-1} \left(\sum_{l_1=0}^{l} H^{p}_{k_2, \left\lceil \frac{m_{10}}{m_{00}} k_2 \right\rceil + (l-l_1)} X^{p}_{k_1, \left\lceil \frac{m_{10}}{m_{00}} k_1 \right\rceil + l_1} \right) z_2^{-l} +$$

$$+ z_2^{-\left(\left\lceil \frac{m_{10}}{m_{00}} k_1 \right\rceil + \left\lceil \frac{m_{10}}{m_{00}} k_2 \right\rceil\right)} \sum_{l=0}^{m_{11}-1} z_2^{-(l-m_{11})} \left(\sum_{l_1=l+1}^{m_{11}-1} H^{p}_{k_2, \left\lceil \frac{m_{10}}{m_{00}} k_2 \right\rceil + (l-l_1)+m_{11}} X^{p}_{k_1, \left\lceil \frac{m_{10}}{m_{00}} k_1 \right\rceil + l_1} \right)$$

(4.2.43)

Para obtener la expresión de la matriz de bloques es útil reescribir la definición de descomposición polifásica (4.2.33) como

$$Y(z_1, z_2) = \sum_{k=0}^{m_{00}-1} z_1^{-k} z_2^{-\left\lceil \frac{m_{10}}{m_{00}} k \right\rceil} \sum_{l=0}^{m_{11}-1} Y^{p}_{k, \left\lceil \frac{m_{10}}{m_{00}} k \right\rceil + l} (z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}) z_2^{-l} \tag{4.2.44}$$

de esta forma se pone de manifiesto que la polifase $Y^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l}$ es la componente l del vector Y_k (4.2.36). Teniendo en cuenta (4.2.41),(4.2.43) y (4.2.44), obtenemos la siguiente expresión para $Y^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l}$

$$
\begin{aligned}
Y^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l} = & \sum_{k_1=0}^{k}\left(\sum_{l_1=0}^{l} H^{p}_{k-k_1,\left\lceil \frac{m_{10}}{m_{00}}(k-k_1) \right\rceil+(l-l_1)} X^{p}_{k_1,\left\lceil \frac{m_{10}}{m_{00}}k_1 \right\rceil+l_1}\right) + \\
& + \sum_{k_1=0}^{k}\left(z_2^{-m_{11}} \sum_{l_1=l+1}^{m_{11}-1} H^{p}_{k-k_1,\left\lceil \frac{m_{10}}{m_{00}}(k-k_1) \right\rceil-m_{11}+(l-l_1)} X^{p}_{k_1,\left\lceil \frac{m_{10}}{m_{00}}k_1 \right\rceil+l_1}\right) + \\
& + \sum_{k_1=k+1}^{m_{00}-1}\left(z_1^{-m_{00}} z_2^{-m_{10}} \sum_{l_1=0}^{l} H^{p}_{m_{00}+k-k_1,\left\lceil \frac{m_{10}}{m_{00}}(m_{00}+k-k_1) \right\rceil+(l-l_1)} X^{p}_{k_1,\left\lceil \frac{m_{10}}{m_{00}}k_1 \right\rceil+l_1}\right) + \\
& + \sum_{k_1=k+1}^{m_{00}-1}\left(z_1^{-m_{00}} z_2^{-m_{10}} z_2^{-m_{11}} \sum_{l_1=l+1}^{m_{11}-1} H^{p}_{m_{00}-k-k_1,\left\lceil \frac{m_{10}}{m_{00}}(m_{00}+k-k_1) \right\rceil+m_{11}+(l-l_1)} X^{p}_{k_1,\left\lceil \frac{m_{10}}{m_{00}}k_1 \right\rceil+l_1}\right)
\end{aligned}
\tag{4.2.45}
$$

expresión que relaciona las componentes polifásicas de la secuencia de salida, asociadas a la matriz de submuestreo (4.2.19), con las componentes polifásicas de la secuencia de entrada y del sistema escalar. La ecuación (4.2.45) demuestra que es posible calcular en paralelo todas las secuencias polifásicas que forman la secuencia de salida, aumentando la velocidad de procesamiento del sistema. La expresión (4.2.45) puede escribirse de forma matricial, como

$$\begin{bmatrix} Y_0 \\ Y_1 \\ \vdots \\ \vdots \\ Y_{m_{00}-1} \end{bmatrix} = \begin{bmatrix} H_0 & z_1^{-m_{00}} z_2^{-m_{10}} H_{m_{00}-1} & z_1^{-m_{00}} z_2^{-m_{10}} H_{m_{00}-2} & \cdots & z_1^{-m_{00}} z_2^{-m_{10}} H_1 \\ H_1 & H_0 & z_1^{-m_{00}} z_2^{-m_{10}} H_{m_{00}-1} & \cdots & z_1^{-m_{00}} z_2^{-m_{10}} H_2 \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ H_{m_{00}-1} & H_{m_{00}-2} & \cdots & \cdots & H_0 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ \vdots \\ X_{m_{00}-1} \end{bmatrix}$$

(4.2.46)

siendo

$$Y_k = \begin{bmatrix} Y^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor} \\ \vdots \\ Y^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor + m_{11} - 1} \end{bmatrix} \quad X_k = \begin{bmatrix} X^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor} \\ \vdots \\ X^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor + m_{11} - 1} \end{bmatrix}$$

$$H_k = \begin{bmatrix} H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor} & z_2^{-m_{11}} H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor + m_{11} - 1} & \cdots & z_2^{-m_{11}} H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor + 1} \\ \vdots & \vdots & & \vdots \\ H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor + m_{11} - 1} & H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor + m_{11} - 2} & \cdots & H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}} k \right\rfloor} \end{bmatrix}$$

donde los coeficientes que aparecen en los vectores Y_k , X_k y en las matrices H_k son las componentes polifásicas de la secuencia de salida, entrada y sistema escalar, respectivamente, es decir

$$Y^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l} = Y^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}})$$

$$X^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l} = X^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}) \qquad (4.2.47)$$

$$H^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l} = H^{p}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}})$$

La ecuación matricial (4.2.46) es una expresión general para la implementación en bloques de un sistema escalar bidimensional.

4.2.5 Cálculo de las Componentes Polifásicas

Con el desarrollo teórico efectuado en la sección 4.2.4, hemos reducido el problema del cálculo de la matriz de bloques de un sistema bidimensional LTI, al cálculo de las componentes polifásicas asociadas a la matriz de submuestreo requerida para aumentar la velocidad de procesamiento del sistema. En el caso de que el sistema bidimensional sea un sistema con respuesta impulsiva finita, el cálculo de las componentes polifásicas es inmediato una vez escogida la matriz de submuestreo, ya que de (4.2.7) y (4.2.9) llegamos a

$$H^{p}_{k,l}(\xi_1,\xi_2) = \sum_{u_1=-\infty}^{\infty} \sum_{u_2=-\infty}^{\infty} h(m_{00}u_1 + m_{01}u_2, m_{10}u_1 + m_{11}u_2)\, \xi_1^{-u_1}\, \xi_2^{-u_2} \qquad (4.2.48)$$

donde solo un número finito de sumandos es distintos de cero.

En el caso de sistemas IIR la expresión (4.2.48) es también válida, pero al ser infinito el número de sumandos no es posible obtener una expresión cerrada. De nuevo vamos a considerar el caso de implementaciones en bloques con matrices de submuestreo colineales, de la forma (4.2.19).

Diversos autores han mostrado la relación entre las componentes polifásicas de un sistema monodimensional y las componentes de modulación [Vaidyanathan 90] [Fliege 94]. Vamos a generalizar esta relación al caso de descomposiciones polifásicas con matriz de submuestreo (4.2.19).

Teniendo en cuenta que la descomposición polifásica viene descrita por la matriz de submuestreo M (4.2.19), definimos la secuencia de submuestreo asociada a la matriz M como [Vetterli y Karlsson 90]

$$w_M(n_1,n_2)=\frac{1}{|M|}\sum_{k=0}^{m_{00}-1}\sum_{l=0}^{m_{11}-1}W_{|M|}^{(m_{11}k-m_{10}l)n_1+m_{00}ln_2} \tag{4.2.49}$$

donde $W_N^n=e^{-\frac{j2\pi n}{N}}$. Puede verificarse fácilmente que

$$w_M(n_1,n_2)=\begin{cases}1 & para\begin{cases}n_1=m_{00}u_1\\ n_2=m_{10}u_1+m_{11}u_2\end{cases}\\ 0 & Caso\,contrario\end{cases} \tag{4.2.50}$$

por lo que podemos representar la secuencia $h(n_1,n_2)$ en el dominio *temporal* como

$$h(n_1,n_2)=\sum_{k=0}^{m_{00}-1}\sum_{l=0}^{m_{11}-1}h(n_1,n_2)w_M\left(n_1-k,n_2-\left\lceil\frac{m_{10}}{m_{00}}k\right\rceil-l\right) \qquad (4.2.51)$$

Teniendo en cuenta (4.2.44), la expresión de $h(n_1,n_2)$ en el dominio transformado en función de las componentes polifásicas asociadas a la matriz de submuestreo M es

$$H(z_1,z_2)=\sum_{k=0}^{m_{00}-1}\sum_{l=0}^{m_{11}-1}z_1^{-k}z_2^{-\left\lceil\frac{m_{10}}{m_{00}}k\right\rceil}z_2^{-l}H^p_{k,\left\lceil\frac{m_{10}}{m_{00}}k\right\rceil+l}(z_1^{m_{00}}z_2^{m_{10}},z_2^{m_{11}}) \qquad (4.2.52)$$

Por otro lado, el cálculo de la transformada Z de la expresión (4.2.51), y la comparación con (4.2.52), nos permite afirmar

$$Z\left(h(n_1,n_2)w_M\left(n_1-k,n_2-\left\lceil\frac{m_{10}}{m_{00}}k\right\rceil-l\right)\right)=z_1^{-k}z_2^{-\left\lceil\frac{m_{10}}{m_{00}}k\right\rceil}z_2^{-l}H^p_{k,\left\lceil\frac{m_{10}}{m_{00}}k\right\rceil+l}(z_1^{m_{00}}z_2^{m_{10}},z_2^{m_{11}}) \qquad (4.2.53)$$

Aplicando la definición de transformada Z bidimensional llegamos a la conclusión de que

$$z_1^{-k} z_2^{-\left\lceil \frac{m_{10}}{m_{00}} k \right\rceil} z_2^{-l} H^{P}_{k,\left\lceil \frac{m_{10}}{m_{00}} k \right\rceil + l}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}) = \frac{1}{|M|} \sum_{k_2=0}^{m_{00}-1} \sum_{l_2=0}^{m_{11}-1} H(z_1 W_{|M|}^{-(m_{11}k_2 - m_{10}l_2)}, z_2 W_{|M|}^{-m_{00}l_2}) W_{m_{00}}^{-k_2 k} W_{m_{11}}^{-l_2 l} \tag{4.2.54}$$

expresión que relaciona las componentes polifásicas, asociadas a la matriz de submuestreo colineal (4.2.19), con las componentes de modulación.

La relación (4.2.54) expone un método práctico para calcular las componentes polifásicas, independientemente de que el sistema sea o no de variables separables. Sin embargo, como mencionamos anteriormente, el cálculo de las componentes polifásicas de sistemas con respuesta impulsiva finita se efectúa por inspección de la expresión de la transformada Z del sistema. En el caso de sistemas con respuesta impulsiva infinita vamos a proponer un algoritmo de cálculo de las componentes polifásicas, basado en la igualdad (4.2.52), pero de menor complejidad computacional , y fácil de implementar con un programa de cálculo numérico.

4.2.6 Algoritmo de Cálculo de Componentes Polifásicas de Sistemas IIR

La idea principal del algoritmo es encontrar un polinomio bidimensional $P(z_1,z_2)$ tal que, si el sistema viene descrito como

$$H(z_1,z_2) = \frac{N(z_1,z_2)}{D(z_1,z_2)} \tag{4.2.55}$$

el producto polinomial $D(z_1,z_2)$ $P(z_1,z_2)$ sea un polinomio bidimensional función de $z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}$, es decir

$$B(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}) = D(z_1,z_2)P(z_1,z_2) \tag{4.2.56}$$

de esta forma, el cálculo de las componentes polifásicas se reduce al cálculo de las componentes polifásicas del sistema con respuesta impulsiva finita

$$A(z_1,z_2) = N(z_1,z_2)P(z_1,z_2) \tag{4.2.57}$$

siendo

$$H^P_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil + l}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}}) = \frac{A^P_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil + l}(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}})}{B(z_1^{m_{00}} z_2^{m_{10}}, z_2^{m_{11}})} \tag{4.2.58}$$

Vamos a calcular el valor del polinomio $P(z_1,z_2)$. Teniendo en cuenta la expresión de la respuesta impulsiva del sistema (4.2.55), se deduce que las componentes de modulación en (4.2.55) valdrán

$$H(z_1 W_{|M|}^{-(m_{11}k_2 - m_{10}l_2)}, z_2 W_{|M|}^{-m_{00}l_2}) = \frac{N(z_1 W_{|M|}^{-(m_{11}k_2 - m_{10}l_2)}, z_2 W_{|M|}^{-m_{00}l_2})}{D(z_1 W_{|M|}^{-(m_{11}k_2 - m_{10}l_2)}, z_2 W_{|M|}^{-m_{00}l_2})} \tag{4.2.59}$$

Sustituyendo (4.2.59) en (4.2.54), y efectuando el doble sumatorio vemos que puede expresarse

$$\sum_{k_2=0}^{m_{00}-1}\sum_{l_2=0}^{m_{11}-1}\frac{N(z_1W_{|M|}^{-(m_{11}k_2-m_{10}l_2)},z_2W_{|M|}^{-m_{00}l_2})}{D(z_1W_{|M|}^{-(m_{11}k_2-m_{10}l_2)},z_2W_{|M|}^{-m_{00}l_2})}W_{m_{00}}^{-k_2k}W_{m_{11}}^{-l_2l}=\frac{Q(z_1,z_2)}{\prod_{k_2=0}^{m_{00}-1}\prod_{l_2=0}^{m_{11}-1}D(z_1W_{|M|}^{-(m_{11}k_2-m_{10}l_2)},z_2W_{|M|}^{-m_{00}l_2})} \tag{4.2.60}$$

pero el denominador de (4.2.60) es el denominador de (4.2.54), es decir, utilizando la nomenclatura (4.2.58) tenemos que

$$B(z_1^{m_{00}}z_2^{m_{10}},z_2^{m_{11}})=\prod_{k_2=0}^{m_{00}-1}\prod_{l_2=0}^{m_{11}-1}D(z_1W_M^{-(m_{11}k_2-m_{10}l_2)},z_2W_M^{-m_{00}l_2}) \tag{4.2.61}$$

que puede expresarse como

$$B(z_1^{m_{00}}z_2^{m_{10}},z_2^{m_{11}})=D(z_1,z_2)\times\prod_{\substack{k_2=0\\k_2+l_2\neq 0}}^{m_{00}-1}\prod_{l_2=0}^{m_{11}-1}D(z_1W_{|M|}^{-(m_{11}k_2-m_{10}l_2)},z_2W_{|M|}^{-m_{00}l_2}) \tag{4.2.62}$$

con lo que el polinomio bidimensional buscado vale

$$P(z_1,z_2)=\prod_{\substack{k_2=0\\k_2+l_2\neq 0}}^{m_{00}-1}\prod_{l_2=0}^{m_{11}-1}D(z_1W_{|M|}^{-(m_{11}k_2-m_{10}l_2)},z_2W_{|M|}^{-m_{00}l_2}) \tag{4.2.63}$$

El algoritmo propuesto calcula los $|M|-1$ productos polinomiales (4.2.63), para obtener la expresión del polinomio $P(z_1,z_2)$. Conocido éste, se calcula el denominador de todas las componentes polifásicas (4.2.62), y se genera el polinomio $A(z_1,z_2)$ según (4.2.57). Finalmente, por inspección es posible calcular sus componentes polifase, y aplicando (4.2.58) obtenemos la expresión de las componentes polifásicas del sistema.

4.2.7 Ejemplo

Para clarificar las ideas expresadas en el apartado 4.2 vamos a mostrar un ejemplo de obtención de la matriz de bloques del sistema bidimensional

$$H(z_1,z_2)=\frac{0.45\,z_1^{-1}z_2^{-1}-z_1^{-1}-0.5\,z_2^{-1}}{1-0.6\,z_1^{-1}-0.6\,z_2^{-1}+0.27\,z_1^{-1}z_2^{-1}}$$

empleando la matriz de submuestreo

$$P=\begin{bmatrix}2 & 2\\ -1 & 1\end{bmatrix}$$

Para obtener una matriz colineal que defina el mismo retículo de submuestreo, buscamos una matriz de enteros con determinante unidad. Como puede observarse, los coeficientes p_{00} y p_{01} son iguales. Aplicando cualquiera

de las expresiones (4.2.25) o (4.2.26) obtendremos dos matrices con determinante unidad. Por ejemplo, de (4.2.26) se obtiene

$$C = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$

con lo que la matriz de submuestreo colineal resultante queda de la forma

$$M = \begin{bmatrix} 2 & 0 \\ -1 & 2 \end{bmatrix}$$

Para calcular las componentes polifásicas asociadas a la matriz M utilizamos el algoritmo descrito en el apartado 4.2.6. Para facilitar la notación vamos a emplear notación matricial, donde los polinomios resultantes se obtienen del siguiente modo

$$T = \begin{bmatrix} t_{00} & \cdots & t_{0L_1} \\ \vdots & \vdots & \vdots \\ t_{L_2 0} & \cdots & t_{L_2 L_1} \end{bmatrix}$$

indica que

$$T(z_1, z_2) = \sum_{k=0}^{L_1} \sum_{l=0}^{L_2} t_{lk} z_1^{-k} z_2^{-l}$$

Efectuando (4.2.63) llegamos a la siguiente expresión para el polinomio $P(z_1,z_2)$.

$$P=\begin{bmatrix} 1.0000 & 0.6000 & 0.3600 & 0.2160 \\ 0.6000 & 0.4500 & 0.1080 & 0.0972 \\ -0.3600 & -0.1080 & -0.1215 & -0.0437 \\ -0.2160 & -0.0972 & -0.0437 & -0.0197 \end{bmatrix}$$

El denominador de las componentes polifásicas se calcula a través de (4.2.62), mientras que el numerador de las distintas polifases se corresponden con las componentes polifásicas del polinomio $A(z_1,z_2)$, generado según (4.2.57). El resultado obtenido es

$$B=\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0000 & -0.1296 \\ 0 & 0.0000 & -0.2160 & 0.0000 & 0.0000 \\ -0.7200 & 0.0000 & 0.0000 & 0.0000 & 0.0525 \\ 0.0000 & 0.0000 & 0.0583 & 0.0000 & 0.0000 \\ 0.1296 & 0.0000 & 0.0000 & 0.0000 & -0.0053 \end{bmatrix}$$

$$A=\begin{bmatrix} 0 & -1.0000 & -0.6000 & -0.3600 & -0.2160 \\ -0.5000 & -0.4500 & -0.3600 & -0.0540 & 0.0000 \\ -0.3000 & 0.4050 & 0.2565 & 0.1215 & 0.0875 \\ 0.1800 & 0.1080 & 0.1093 & 0.0109 & 0.0000 \\ 0.1080 & -0.0486 & -0.0219 & -0.0098 & -0.0089 \end{bmatrix}$$

Obtenidas las polifases de *A*, empleando la expresión (4.2.58) calculamos las componentes polifásicas del sistema. Las expresiones de las polifases de *A* y del denominador *B*, una vez diezmadas, son

$$B=\begin{bmatrix} 1.0000 & 0 & 0 \\ -0.7200 & -0.2160 & -0.1296 \\ 0.1296 & 0.0583 & 0.0525 \\ 0 & 0 & -0.0053 \end{bmatrix}$$

$$A_{00}=\begin{bmatrix} 0 & 0 & 0 \\ -0.3000 & -0.3600 & -0.2160 \\ 0.1080 & 0.1093 & 0.0875 \\ 0 & 0 & -0.0089 \end{bmatrix}$$

$$A_{01}=\begin{bmatrix} -1.0000 & 0 \\ 0.4050 & -0.0540 \\ -0.0486 & 0.0109 \end{bmatrix}$$

$$A_{10}=\begin{bmatrix} -0.5000 & -0.6000 & 0 \\ 0.1800 & 0.2565 & 0.0000 \\ 0 & -0.0219 & 0.0000 \end{bmatrix}$$

$$A_{11}=\begin{bmatrix} -0.4500 & -0.3600 \\ 0.1080 & 0.1215 \\ 0 & -0.0098 \end{bmatrix}$$

La expresión de la matriz de bloques, empleando la matriz de submuestreo M, queda de la forma (4.2.46)

$$\begin{bmatrix} Y^p_{0,0} \\ Y^p_{0,1} \\ Y^p_{1,0} \\ Y^p_{1,1} \end{bmatrix} = \begin{bmatrix} H^p_{0,0} & z_2^{-2}H^p_{0,1} & z_1^{-2}z_2H^p_{1,0} & z_1^{-2}z_2^{-1}H^p_{1,1} \\ H^p_{0,1} & H^p_{0,0} & z_1^{-2}z_2H^p_{1,1} & z_1^{-2}z_2H^p_{1,0} \\ H^p_{1,0} & z_2^{-2}H^p_{1,1} & H^p_{0,0} & z_2^{-2}H^p_{0,1} \\ H^p_{1,1} & H^p_{1,0} & H^p_{0,1} & H^p_{0,0} \end{bmatrix} \begin{bmatrix} X^p_{0,0} \\ X^p_{0,1} \\ X^p_{1,0} \\ X^p_{1,1} \end{bmatrix}$$

donde

$$H^p_{k,l}(z_1^2z_2^{-1}, z_2^2) = \frac{A^p_{k,l}(z_1^2z_2^{-1}, z_2^2)}{B(z_1^2z_2^{-1}, z_2^2)}$$

Para conseguir la expresión de la matriz de bloques asociada a la matriz de submuestreo hexagonal P, nos valemos de la relación deducida en la sección 4.2.3 entre componentes polifásicas equivalentes (4.2.18). Teniendo en cuenta

$$P = \begin{bmatrix} 2 & 2 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} = MC^{-1}$$

la equivalencia entre componentes polifásicas

$$X^p_{k,l} \leftrightarrow X^p_{q,r}$$
$$H^p_{k,l} \leftrightarrow H^p_{q,r}$$
$$Y^p_{k,l} \leftrightarrow Y^p_{q,r}$$

puede representarse con la siguiente tabla

(k,l)	(q,r)	(k-q)	(l-r)
(0,0)	(0,0)	0	0
(0,1)	(2,0)	-2	1
(1,0)	(1,0)	0	0
(1,1)	(3,0)	-2	1

Sustituyendo en la ecuación (4.2.18) el valor de los coeficientes involucrados, llegamos a la siguiente relación para las componentes polifásicas de la secuencia de salida $Y(z_1, z_2)$

$$Y_{0,0}^{p}(z_1^2 z_2^{-1}, z_1^2 z_2) = Y_{0,0}^{p}(z_1^2 z_2^{-1}, z_2^2)$$
$$Y_{2,0}^{p}(z_1^2 z_2^{-1}, z_1^2 z_2) = Y_{0,1}^{p}(z_1^2 z_2^{-1}, z_2^2) z_1^2 z_2^{-1}$$
$$Y_{1,0}^{p}(z_1^2 z_2^{-1}, z_1^2 z_2) = Y_{1,0}^{p}(z_1^2 z_2^{-1}, z_2^2)$$
$$Y_{3,0}^{p}(z_1^2 z_2^{-1}, z_1^2 z_2) = Y_{1,1}^{p}(z_1^2 z_2^{-1}, z_2^2) z_1^2 z_2^{-1}$$

Sustituyendo el valor de las componentes $Y_{k,l}^{p}(z_1^2 z_2^{-1}, z_2^2)$, $X_{k,l}^{p}(z_1^2 z_2^{-1}, z_2^2)$ y $H_{k,l}^{p}(z_1^2 z_2^{-1}, z_2^2)$, por las componentes equivalentes, llegamos a la siguiente expresión para la matriz de bloques asociada a la matriz de submuestreo hexagonal P

$$\begin{bmatrix} Y_{0,0}^{p} \\ Y_{1,0}^{p} \\ Y_{2,0}^{p} \\ Y_{3,0}^{p} \end{bmatrix} = \begin{bmatrix} H_{0,0}^{p} & z_1^{-2} z_2 H_{1,0}^{p} & z_1^{-4} H_{2,0}^{p} & z_1^{-6} z_2 H_{3,0}^{p} \\ H_{1,0}^{p} & H_{0,0}^{p} & z_1^{-4} H_{3,0}^{p} & z_1^{-4} H_{2,0}^{p} \\ H_{2,0}^{p} & z_1^{-2} z_2 H_{3,0}^{p} & H_{0,0}^{p} & z_1^{-2} z_2 H_{1,0}^{p} \\ H_{3,0}^{p} & H_{2,0}^{p} & H_{1,0}^{p} & H_{0,0}^{p} \end{bmatrix} \begin{bmatrix} X_{0,0}^{p} \\ X_{1,0}^{p} \\ X_{2,0}^{p} \\ X_{3,0}^{p} \end{bmatrix}$$

donde todos los coeficientes son función de $\left(z_1^2 z_2, z_1^2 z_2^{-1}\right)$. La expresión de los elementos de la matriz de bloques asociada a la matriz de submuestreo hexagonal P, en su versión diezmada, queda de la forma

$$B = z_1^2 \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0.72 & -0.2160 & -0.1296 \\ 0.1296 & 0.0583 & 0.0525 & 0 \\ 0 & -0.0053 & 0 & 0 \end{bmatrix}$$

$$A_{0,0} = z_1^2 z_2^{-1} \begin{bmatrix} 0 & -0.3 & -0.36 & -0.216 \\ 0.1080 & 0.1093 & 0.0875 & 0 \\ 0 & -0.0089 & 0 & 0 \end{bmatrix}$$

$$A_{2,0} = z_1^2 \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0.4050 & -0.0540 \\ -0.0486 & 0.0109 & 0 \end{bmatrix}$$

$$A_{1,0} = z_1 \begin{bmatrix} 0 & -0.5 & -0.6 \\ 0.18 & 0.2565 & 0 \\ -0.0219 & 0 & 0 \end{bmatrix}$$

$$A_{3,0} = z_1 \begin{bmatrix} 0 & -0.45 & -0.36 \\ 0.1080 & 0.1215 & 0 \\ -0.0098 & 0 & 0 \end{bmatrix}$$

donde

$$H_{q,r}^{p}(z_1, z_2) = \frac{A_{q,r}^{p}(z_1, z_2)}{B(z_1, z_2)}$$

4.3 ESTRUCTURAS 2D BASADAS EN LA TEORÍA DE ALGORITMOS RÁPIDOS DE CONVOLUCIÓN

La implementación en bloques del sistema digital permite aumentar la velocidad de procesamiento del sistema, procesando en paralelo un número determinado de muestras. Sin embargo, como se desprende de (4.2.46), la implementación real de un sistema en bloques presenta una gran complejidad hardware, que aumenta a medida que se incrementa el número de muestras procesadas en paralelo. En el caso monodimensional, la utilización de estructuras basadas en algoritmos rápidos de convolución [Acha 89][Mou y Duhamel 91] disminuye considerablemente el problema.

4.3.1 Matriz Toeplitz en 2 Niveles

La implementación de la matriz de bloques bidimensional, definida en (4.2.46), tiene un costo elevado en lo referente al número de procesadores utilizados, $|M|^2$. Para reducirlo volveremos a utilizar la teoría de algoritmos de convolución lineal monodimensionales. La aplicación directa no es posible, ya que la matriz (4.2.46) no es Toeplitz, y por lo tanto no es posible aplicar el principio de transformación. Sin embargo, vamos a demostrar que la matriz de bloques de un sistema LTI bidimensional es Toeplitz en 2 niveles.

Como vimos en el apartado 4.2.4, la matriz de bloques de un sistema LTI bidimensional, asociada a la matriz de submuestreo M de la forma (4.2.19), es de la forma

$$H=\begin{bmatrix} H_0 & z_1^{-m_{00}} z_2^{-m_{10}} H_{m_{00}-1} & z_1^{-m_{00}} z_2^{-m_{10}} H_{m_{00}-2} & \cdots & z_1^{-m_{00}} z_2^{-m_{10}} H_1 \\ H_1 & H_0 & z_1^{-m_{00}} z_2^{-m_{10}} H_{m_{00}-1} & \cdots & z_1^{-m_{00}} z_2^{-m_{10}} H_2 \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ H_{m_{00}-1} & H_{m_{00}-2} & \cdots & \cdots & H_0 \end{bmatrix} \tag{4.3.1}$$

donde los coeficientes de la matriz valen

$$H_k=\begin{bmatrix} H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}}k \right\rfloor} & \cdots & z_2^{-m_{11}} H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}}k \right\rfloor+1} \\ \vdots & \ddots & \vdots \\ H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}}k \right\rfloor+m_{11}-1} & \cdots & H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}}k \right\rfloor} \end{bmatrix} \tag{4.3.2}$$

Observando la expresión (4.3.2) vemos que la componente de la matriz situada en la fila f_0 y columna c_0 será

$$H_k(f_0,c_0)=\begin{cases} H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}}k \right\rfloor+f_0-c_0} & si\ c_0 \le f_0 \\ z_2^{-m_{11}} H^p_{k,\left\lfloor \frac{m_{10}}{m_{00}}k \right\rfloor+m_{11}-c_0+f_0} & si\ c_0 > f_0 \end{cases} \tag{4.3.3}$$

Cualquier elemento situado en (f_1,c_1) que cumpla la propiedad

$$f_1 - f_0 = c_1 - c_0 \tag{4.3.4}$$

tendrá el mismo valor que (4.3.3), por lo que las matrices (4.3.2) son Toeplitz. De igual forma, en la matriz de bloques bidimensional (4.3.2) el elemento situado en la fila f_0 y columna c_0 vendrá dado por la expresión

$$H(f_0,c_0) = \begin{cases} H_{f_0-c_0} & si\ c_0 \le f_0 \\ z_1^{-m_{00}} z_2^{-m_{10}} H_{m_{00}+f_0-c_0} & si\ c_0 > f_0 \end{cases} \tag{4.3.5}$$

por lo que cualquier elemento situado en (f_1,c_1) que cumpla (4.3.4) tendrá el mismo valor, y por lo tanto es Toeplitz. Por este motivo decimos que la matriz de bloques de un sistema LTI bidimensional es Toeplitz en 2 niveles.

4.3.2. Estructuras Iteradas

La propiedad demostrada anteriormente, en la que la matriz de bloques bidimensional es Toeplitz en 2 niveles, nos permite aplicar la teoría de algoritmos de convolución lineal de forma análoga al caso monodimensional. Esto es posible siempre que el algoritmo de convolución lineal que empleemos no utilice la propiedad conmutativa de la multiplicación, ni use la operación de división [Blahut 85].

La expresión matricial (4.2.46) es análoga a la expresión del procesamiento en bloques de un sistema monodimensional de longitud m_{00} x m_{00}, donde los coeficientes de la matriz de bloques son a su vez matrices Toeplitz de dimensiones m_{11} x m_{11}, y donde los coeficientes de los vectores de entrada y salida, de dimensión m_{00} x 1, son vectores de dimensión m_{11} x 1. Por lo tanto, si el algoritmo de convolución empleado cumple los requisitos anteriormente mencionados, las adiciones y multiplicaciones del algoritmo se traducen en adiciones y multiplicaciones de matrices. De igual forma, los productos matriciales generados no son más que la implementación en bloques de longitud m_{11} x m_{11} de los sistemas (4.3.2), pudiendo aplicar la teoría de algoritmos de convolución lineal para su implementación. A las estructuras así obtenidas las denominaremos estructuras iteradas, por su analogía con el caso monodimensional [Blahut 85] [Acha 89].

Como ejemplo, obtendremos una estructura iterada para la matriz de bloques asociada a la matriz de submuestreo colineal del ejemplo mostrado en la sección 4.2.7

$$\begin{bmatrix} Y^p_{0,0} \\ Y^p_{0,1} \\ Y^p_{1,0} \\ Y^p_{1,1} \end{bmatrix} = \begin{bmatrix} H^p_{0,0} & z_2^{-2}H^p_{0,1} & z_1^{-2}z_2H^p_{1,0} & z_1^{-2}z_2^{-1}H^p_{1,1} \\ H^p_{0,1} & H^p_{0,0} & z_1^{-2}z_2H^p_{1,1} & z_1^{-2}z_2H^p_{1,0} \\ H^p_{1,0} & z_2^{-2}H^p_{1,1} & H^p_{0,0} & z_2^{-2}H^p_{0,1} \\ H^p_{1,1} & H^p_{1,0} & H^p_{0,1} & H^p_{0,0} \end{bmatrix} \begin{bmatrix} X^p_{0,0} \\ X^p_{0,1} \\ X^p_{1,0} \\ X^p_{1,1} \end{bmatrix} \qquad (4.3.6)$$

Reescribiendo (4.3.6) con la notación (4.2.46) queda

$$\begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix} = \begin{bmatrix} H_0 & z_1^{-2} z_2 H_1 \\ H_1 & H_0 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} \tag{4.3.7}$$

que es una matriz Toeplitz 2 x 2, donde los elementos de la matriz de bloques son matrices Toeplitz 2 x 2 de la forma

$$H_0 = \begin{bmatrix} H_{0,0}^p & z_2^{-2} H_{0,1}^p \\ H_{0,1}^p & H_{0,0}^p \end{bmatrix} \qquad H_1 = \begin{bmatrix} H_{1,0}^p & z_2^{-2} H_{1,1}^p \\ H_{1,1}^p & H_{1,0}^p \end{bmatrix} \tag{4.3.8}$$

y donde las componentes de los vectores de entrada y salida son

$$Y_0 = \begin{bmatrix} Y_{0,0}^p \\ Y_{0,1}^p \end{bmatrix} \quad Y_1 = \begin{bmatrix} Y_{1,0}^p \\ Y_{1,1}^p \end{bmatrix} \quad X_0 = \begin{bmatrix} X_{0,0}^p \\ X_{0,1}^p \end{bmatrix} \quad X_1 = \begin{bmatrix} X_{1,0}^p \\ X_{1,1}^p \end{bmatrix} \tag{4.3.9}$$

En la revisión efectuada en el capítulo 2 mostramos un algoritmo óptimo para la implementación en bloques de longitud 2 , aplicando el principio de transposición a un algoritmo de Winograd para convolución lineal 2 x 2. El algoritmo [Acha 89]

$$\begin{bmatrix} e_0 \\ e_1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} f & -f_1 & \\ & f_1 & \\ & & f_2 - f_1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \end{bmatrix} \tag{4.3.10}$$

es equivalente a efectuar

$$\begin{bmatrix} e_0 \\ e_1 \end{bmatrix} = \begin{bmatrix} f_1 & f_0 \\ f_2 & f_1 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \end{bmatrix} \tag{4.3.11}$$

por lo que identificando miembros (4.3.7) (4.3.11), obtenemos el siguiente algoritmo para la expresión (4.3.7)

$$\begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix} = \begin{bmatrix} I & I & 0 \\ 0 & I & I \end{bmatrix} \begin{bmatrix} z_1^{-2} z_2 H_1 - H_0 & & \\ & H_0 & \\ & & H_1 - H_0 \end{bmatrix} \begin{bmatrix} 0 & I \\ I & I \\ I & 0 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} \tag{4.3.12}$$

siendo I la matriz identidad 2 x 2. El algoritmo (4.3.12) puede ser implementado como se muestra en la figura 4.1

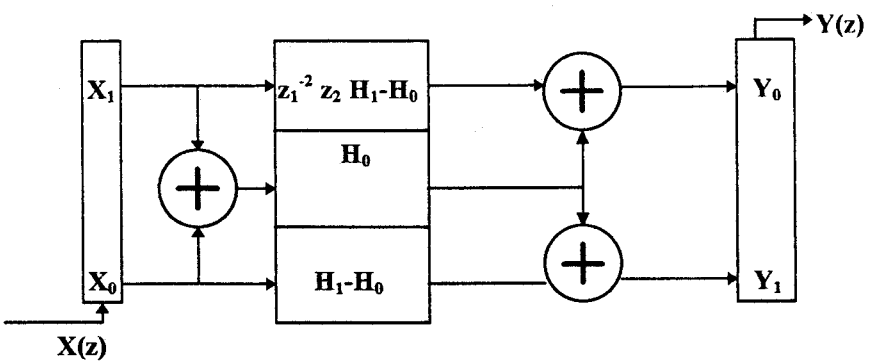

Figura 4.1

La implementación en bloques resultante requiere un total de 3 adiciones matriciales de dimensión 2 x 1 y 3 productos matriciales. Por otra parte, las tres componentes de la matriz diagonal son matrices Toeplitz, pudiendo aplicar de

nuevo el algoritmo (4.3.10) para efectuar los productos matriciales, requiriendo cada uno 3 adiciones y 3 procesadores . El costo hardware de la estructura (Figura 4.1) es de 9 procesadores y 15 adiciones escalares entre el pre- y post-procesado.

En general, al emplear una estructura iterada para implementar en bloques un sistema bidimensional, escogida la matriz de submuestreo (4.2.19) si el algoritmo utilizado para efectuar (4.3.1) requiere A_1 adiciones y M_1 multiplicaciones, y el algoritmo para (4.3.2) necesita A_2 sumas y M_2 productos, el costo de la estructura es

$$\begin{aligned} A &= A_1 m_{11} + M_1 A_2 \quad \textit{adiciones} \\ M &= M_1 M_2 \qquad\qquad\quad \textit{procesadores} \end{aligned} \tag{4.3.13}$$

4.3.3 Optimización de la Estructura

Los algoritmos de Winograd para convolución, tanto cíclica como lineal, son óptimos en el sentido de emplear el mínimo número de multiplicaciones, lo que se traduce en implementaciones en bloques con el mínimo número de procesadores. Sin embargo, para utilizar la teoría de algoritmos rápidos de convolución monodimensionales hemos tenido que emplear algoritmos iterados, donde el número de multiplicaciones, y por lo tanto de procesadores, requeridos no es el óptimo. A pesar de este inconveniente, los algoritmos

iterados poseen la ventaja de mantener el número de adiciones considerablemente más bajo que los algoritmos Winograd equivalentes.

Aunque la utilización de algoritmos iterados es una buena aproximación para la implementación en bloques bidimensionales, podemos optimizar el número de adiciones efectuadas en el pre y post procesado. La matriz de bloques (4.2.46) se obtuvo a partir de la expresión (4.2.45) reordenando términos y utilizando notación matricial, pero no es la única posibilidad. Otra posible representación matricial de (4.2.45) es

$$\begin{bmatrix} Y_0 \\ Y_1 \\ \vdots \\ Y_{m_{11}-1} \end{bmatrix} = \begin{bmatrix} H_0 & z_2^{-m_{11}} H_{m_{11}-1} & z_2^{-m_{11}} H_{m_{11}-2} & \cdots & z_2^{-m_{11}} H_1 \\ H_1 & H_0 & z_2^{-m_{11}} H_{m_{11}-1} & \cdots & z_2^{-m_{11}} H_2 \\ H_2 & H_1 & H_0 & \cdots & z_2^{-m_{11}} H_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ H_{m_{11}-1} & H_{m_{11}-2} & H_{m_{11}-3} & \cdots & H_0 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_{m_{11}-1} \end{bmatrix} \qquad (4.3.14)$$

donde

$$Y_l = \begin{bmatrix} Y^p_{0,l} \\ Y^p_{1,\left\lfloor \frac{m_{10}}{m_{00}} \right\rfloor + l} \\ \vdots \\ Y^p_{m_{00}-1,\left\lfloor \frac{m_{10}}{m_{00}}(m_{00}-1) \right\rfloor + l} \end{bmatrix} \qquad X_l = \begin{bmatrix} X^p_{0,l} \\ X^p_{1,\left\lfloor \frac{m_{10}}{m_{00}} \right\rfloor + l} \\ \vdots \\ X^p_{m_{00}-1,\left\lfloor \frac{m_{10}}{m_{00}}(m_{00}-1) \right\rfloor + l} \end{bmatrix}$$

$$H_l = \begin{bmatrix} H^p_{0,l} & z_1^{-m_{00}} z_2^{-m_{10}} H^p_{m_{00}-1,\left\lfloor \frac{m_{10}}{m_{00}}(m_{00}-1)\right\rfloor+l} & \cdots & z_1^{-m_{00}} z_2^{-m_{10}} H^p_{1,\left\lfloor \frac{m_{10}}{m_{00}}\right\rfloor+l} \\ H^p_{1,\left\lfloor \frac{m_{10}}{m_{00}}\right\rfloor+l} & H^p_{0,l} & \cdots & z_1^{-m_{00}} z_2^{-m_{10}} H^p_{2,\left\lfloor \frac{m_{10}}{m_{00}}2\right\rfloor+l} \\ \vdots & \vdots & \vdots & \vdots \\ H^p_{m_{00}-1,\left\lfloor \frac{m_{10}}{m_{00}}(m_{00}-1)\right\rfloor+l} & H^p_{m_{00}-2,\left\lfloor \frac{m_{10}}{m_{00}}(m_{00}-2)\right\rfloor+l} & \cdots & H^p_{0,l} \end{bmatrix}$$

y puede comprobarse fácilmente que es Toeplitz en 2 niveles.

Los algoritmos empleados para implementar ambas expresiones de la matriz de bloques (4.2.46) (4.3.14) son los mismos, variando únicamente el orden de ejecución. El número de procesadores que se requieren con ambas implementaciones coincide, pero el número de adiciones difiere en el caso de que los coeficientes de la diagonal principal de la matriz de submuestreo (4.2.19) sean distintos. En el caso de utilizar la expresión (4.3.14) el costo de la implementación, utilizando los mismos algoritmos que en (4.2.46), es

$$\begin{aligned} A &= A_2 m_{00} + M_2 A_1 \quad \textit{adiciones} \\ M &= M_1 M_2 \qquad\quad \textit{procesadores} \end{aligned} \tag{4.3.15}$$

Teniendo en cuenta (4.3.13) y (4.3.15), la implementación óptima, desde el punto de vista del número de adiciones, será aquella con menor número de adiciones en el pre- y post- procesado.

4.3.4 Ejemplo

Para finalizar la sección 4.3 mostraremos un ejemplo de optimización de la estructura de un sistema bidimensional en bloques, minimizando el número de adiciones efectuadas en el pre- y post- procesado, eligiendo el orden de aplicación del algoritmo iterado. La matriz de submuestreo que gobierna la descomposición polifásica es

$$M = \begin{bmatrix} 2 & 0 \\ -1 & 3 \end{bmatrix}$$

Aplicando (4.2.46), llegamos a la siguiente expresión para la implementación en bloques asociada a la matriz M

$$\begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix} = \begin{bmatrix} H_0 & z_1^{-2} z_2 H_1 \\ H_1 & H_0 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \end{bmatrix}$$

donde

$$Y_0 = \begin{bmatrix} Y_{0,0}^p \\ Y_{0,1}^p \\ Y_{0,2}^p \end{bmatrix} \quad Y_1 = \begin{bmatrix} Y_{1,0}^p \\ Y_{1,1}^p \\ Y_{1,2}^p \end{bmatrix} \quad X_0 = \begin{bmatrix} X_{0,0}^p \\ X_{0,1}^p \\ X_{0,2}^p \end{bmatrix} \quad X_1 = \begin{bmatrix} X_{1,0}^p \\ X_{1,1}^p \\ X_{1,2}^p \end{bmatrix}$$

$$H_0 = \begin{bmatrix} H^p_{0,0} & z_2^{-3}H^p_{0,2} & z_2^{-3}H^p_{0,1} \\ H^p_{0,1} & H^p_{0,0} & z_2^{-3}H^p_{0,2} \\ H^p_{0,2} & H^p_{0,1} & H^p_{0,0} \end{bmatrix} \quad H_1 = \begin{bmatrix} H^p_{1,0} & z_2^{-3}H^p_{1,2} & z_2^{-3}H^p_{1,1} \\ H^p_{1,1} & H^p_{1,0} & z_2^{-3}H^p_{1,2} \\ H^p_{1,2} & H^p_{1,1} & H^p_{1,0} \end{bmatrix}$$

siendo todas las componentes polifásicas función de $(z_1^2 z_2^{-1}, z_2^3)$. Para implementar el sistema en bloques vamos a utilizar un algoritmo Winograd para convolución lineal 2 x 2 (4.3.10) con 1 preadición, 2 postadiciones y 3 multiplicaciones, y para convolución lineal 3 x 3 [Acha 89]

$$\begin{bmatrix} e_0 \\ e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & -1 & 2 & 0 \\ 0 & 1 & 1 & 4 & 1 \end{bmatrix} \begin{bmatrix} (2f_0 - f_1 - 2f_2 + f_3)/2 & & & & \\ & (2f_1 + f_2 - f_3)/2 & & & \\ & & (-2f_1 + 3f_2 - f_3)/6 & & \\ & & & (-f_1 + f_3)/6 & \\ & & & & 2f_1 - f_2 - 2f_3 + f_4 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & -1 & 1 \\ 4 & 2 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix}$$

con 5 multiplicaciones y 17 adiciones distribuidas como

<u>Preadición</u>

$t_1 = g_1 + g_2$

$t_2 = g_0 - g_1$

$D_0 = g_2$

$D_1 = g_2 + t_1$

$D_2 = g_2 + t_2$

$D_3 = t_1 + t_1 + t_2 + D_1$

$D_4 = g_0$

<u>Postadición</u>

$T_1 = S_1 + S_3$

$e_0 = S_0 + S_2 + T_1$

$e_1 = T_1 - S_2 + S_3$

$e_2 = T_1 + S_2 + S_3 + S_3 + S_3 + S_4$

siendo [Blahut 85]

$$\begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ D_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & -1 & 1 \\ 4 & 2 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix} \qquad \begin{bmatrix} e_0 \\ e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & -1 & 2 & 0 \\ 0 & 1 & 1 & 4 & 1 \end{bmatrix} \begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix}$$

La iteración de ambos algoritmos, según (4.3.13), lleva a una implementación con

Número de Procesadores: 15

Número de Adiciones: 60

tal como hemos esquematizado en la figura 4.2

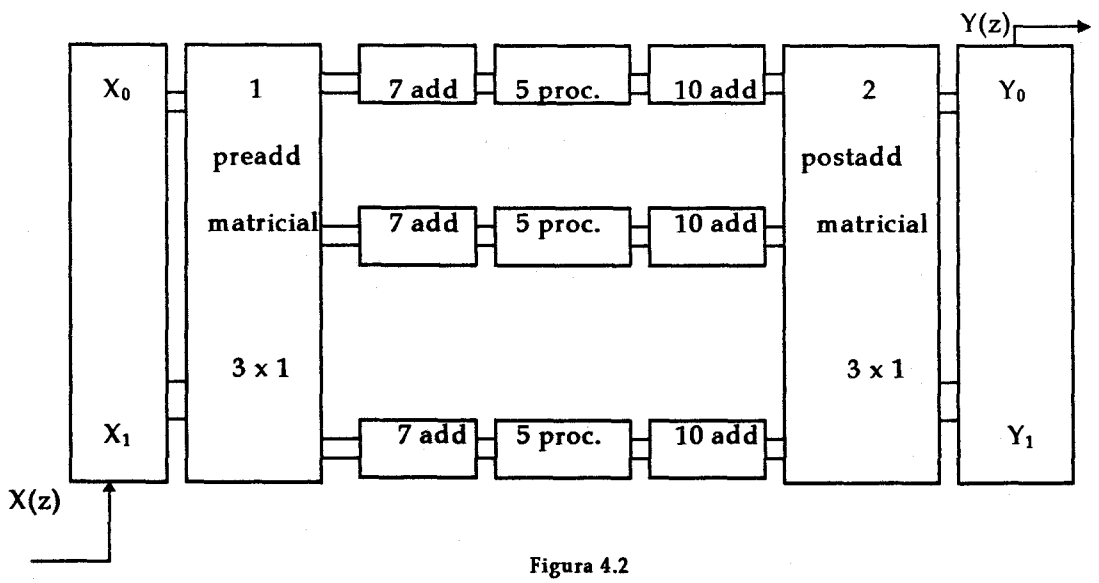

Figura 4.2

Si en vez de efectuar (4.2.46) empleamos (4.3.14), llegamos a la siguiente expresión para la implementación en bloques asociada a la matriz M

$$\begin{bmatrix} Y_0 \\ Y_1 \\ Y_2 \end{bmatrix} = \begin{bmatrix} H_0 & z_1^{-2}z_2H_2 & z_1^{-2}z_2H_1 \\ H_1 & H_0 & z_1^{-2}z_2H_2 \\ H_2 & H_1 & H_0 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ X_2 \end{bmatrix}$$

siendo

$$Y_0 = \begin{bmatrix} Y_{0,0}^p \\ Y_{1,0}^p \end{bmatrix} \quad Y_1 = \begin{bmatrix} Y_{0,1}^p \\ Y_{1,1}^p \end{bmatrix} \quad Y_2 = \begin{bmatrix} Y_{0,2}^p \\ Y_{1,2}^p \end{bmatrix} \quad X_0 = \begin{bmatrix} X_{0,0}^p \\ X_{1,0}^p \end{bmatrix} \quad X_1 = \begin{bmatrix} X_{0,1}^p \\ X_{1,1}^p \end{bmatrix} \quad X_2 = \begin{bmatrix} X_{0,2}^p \\ X_{1,2}^p \end{bmatrix}$$

$$H_0 = \begin{bmatrix} H_{0,0}^p & z_2^{-3}H_{1,0}^p \\ H_{1,0}^p & H_{0,0}^p \end{bmatrix} \quad H_1 = \begin{bmatrix} H_{0,1}^p & z_2^{-3}H_{1,1}^p \\ H_{1,1}^p & H_{0,1}^p \end{bmatrix} \quad H_2 = \begin{bmatrix} H_{0,2}^p & z_2^{-3}H_{1,2}^p \\ H_{1,2}^p & H_{0,2}^p \end{bmatrix}$$

Aplicando los mismos algoritmos, la estructura que obtenemos requiere

Número de Procesadores:	15
Número de Adiciones:	49

reduciendo el número de adiciones, y por lo tanto incrementando la velocidad de procesamiento (fig. 4.3)

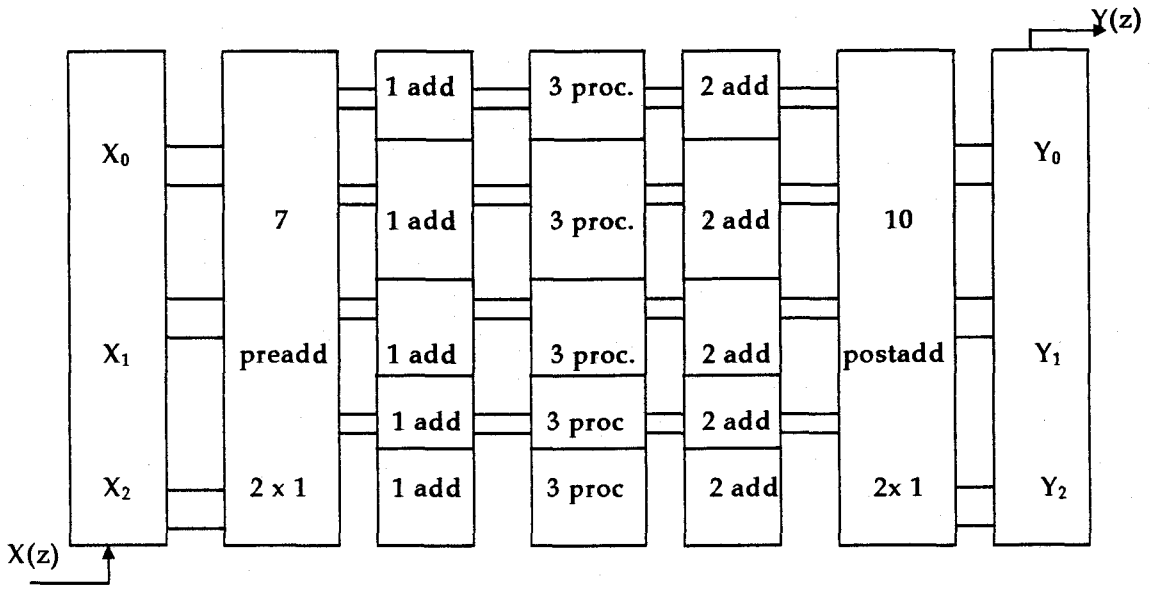

Figura 4.3

4.4 ESTRUCTURAS MONODIMENSIONALES BASADAS EN LA TEORÍA DE FILTROS IFIR

La aplicación de la teoría de algoritmos de convolución para la obtención de estructuras en bloques ha permitido disminuir en gran medida el elevado costo hardware que requiere el procesamiento en paralelo. No obstante, incluso utilizando algoritmos óptimos, el número mínimo de procesadores necesarios para implementar en bloques bidimensionales un sistema LTI, con matriz de submuestreo (4.2.19), es

$$\textit{Número mínimo de procesadores: } 4|M| - 2(m_{00} + m_{11}) + 1 \qquad (4.4.1)$$

La expresión (4.4.1) presenta un serio inconveniente al uso de estructuras en bloques para filtros bidimensionales. El número de procesadores es muy alto, incluso para pequeños valores de $|M|$.

En este apartado vamos a emplear una nueva aproximación al problema del incremento de velocidad de procesamiento en sistemas monodimensionales. Las implementaciones que obtendremos serán estructuras híbridas, al estar formadas por la cascada de dos sistemas, uno escalar y otro en bloques. A este tipo de implementaciones les denominaremos implementaciones mixtas. La gran ventaja de este tipo de implementaciones es el reducido costo hardware que presentan, así como la facilidad de diseño. El principal inconveniente es la existencia de un límite en la velocidad máxima de procesamiento del sistema implementado, lo que convierte a esta técnica en una alternativa a la implementación en bloques siempre que la velocidad de procesamiento requerida esté por debajo, o sea igual, al límite existente.

4.4.1 Filtros FIR Interpolados

La teoría de filtros IFIR surgió como alternativa al diseño convencional de filtros FIR con ancho de banda de transición estrechos [Neuvo et al, 84] [Neuvo et al, 87], en la medida en que se obtienen buenas características con menor número de coeficientes que los filtros convencionales equivalentes, presentando éstos menor sensibilidad al ruido por redondeo. Otra ventaja

añadida es la facilidad del método de diseño, pudiendo utilizar cualquiera de las técnicas de diseño de filtros conocidas.

Los filtros IFIR están formados por la cascada de dos sistemas (Fig. 4.4), el filtro de forma interpolado, $G(z^L)$, y el filtro interpolador, $I(z)$. El filtro de forma interpolado tiene un espectro consistente en la repetición periódica del espectro del filtro de forma, G(z), contraído por un factor 1/L, con centro de simetría en $\omega_c = 2\pi k / L$, siendo $k \in [0,L-1]$.

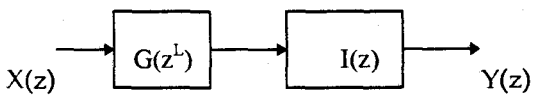

Figura 4.4

El filtro de forma se diseña de tal manera que, al efectuar la interpolación, una de las imágenes generadas esté centrada en el intervalo de frecuencias correspondiente a la banda de paso requerida. El filtro interpolador se diseña de forma que el sistema global cumpla las especificaciones en la banda de rechazo, eliminando las imágenes no deseadas, así como en la banda de paso.

La ventaja de este método de diseño estriba en el hecho de que las especificaciones del filtro de forma y del filtro interpolador son más suaves que las originales, obteniendo implementaciones con menor número de coeficientes.

Aunque originalmente el método de diseño de filtros IFIR se propuso para filtros paso de bajas FIR de banda estrecha, puede ser empleado para filtros paso de altas, paso de banda y rechazo de banda, tanto de banda estrecha como banda ancha [Saramäki et al, 88].

Los inconvenientes son las limitaciones de diseño, especialmente en los casos de filtros paso de banda o rechazo de banda con frecuencia central arbitraria, y en general en filtros de banda ancha. Otro inconveniente es el coste en el número de retrasos del sistema global, superior al de filtros convencionales, que hace que el retraso de grupo del sistema sea muy elevado.

4.4.2 Elección del Orden de Interpolación

El parámetro fundamental en este tipo de diseño es el orden de interpolación del filtro de forma interpolado, $G(z^L)$. El orden del filtro de forma, G(z), y del filtro interpolador, I(z), dependen de su valor. Concretamente, a medida que aumenta L, disminuye el orden de G(z), mientras que aumenta el de I(z). De esta forma, para obtener el sistema escalar con menor tiempo de procesamiento es necesario escoger L de tal forma que minimice el número de coeficientes del sistema global. A continuación deduciremos una expresión aproximada para el orden de interpolación que minimiza el número de coeficientes de la implementación IFIR, suponiendo

que los filtros de forma e interpolador se diseñan como filtros FIR lineales equiripple [Oppenheim y Schafer 75].

Existen diversas aproximaciones para conocer el orden mínimo necesario para que un filtro con repuesta impulsiva finita cumpla las especificaciones de rizado en las bandas. Una buena aproximación nos la proporciona la expresión

$$O_H = \frac{-20\log_{10}\sqrt{\delta_p \delta_s} - 13}{14.6\,\Delta f_H} \tag{4.4.2}$$

atribuida a Kaiser [Fliege 94], donde O_H es el orden del filtro FIR, δ_p y δ_s son el rizado en la banda de paso y rechazo respectivamente, y el término Δf_H representa el ancho de banda de transición del filtro. Para deducir las especificaciones de diseño, en lo referente a rizado en las bandas, de los filtros de forma e interpolador, hay que tener en cuenta que la cascada del filtro $G(z^L)$ con I(z) debe cumplir las especificaciones del sistema global. Suponiendo que δ_p es suficientemente pequeño [Fliege 94], se llega a la conclusión de que si el valor del rizado de los filtros de forma e interpolador cumple

$$\begin{aligned} \delta_{p,G} + \delta_{p,I} &\cong \delta_p \\ \delta_{s,G} = \delta_{s,I} &\cong \delta_s \end{aligned} \tag{4.4.3}$$

entonces el sistema global cumplirá las especificaciones de rizado. Independientemente del valor que elijamos para $\delta_{p,G}$ y $\delta_{p,I}$, siempre será posible expresar

$$\delta_{p,I} = \alpha \delta_{p,G} \tag{4.4.4}$$

permitiéndonos expresar el rizado en las bandas de los filtros G(z) e I(z) como (4.4.3)

$$\delta_{p,G} = \frac{\delta_p}{(1+\alpha)} \quad ; \quad \delta_{p,I} = \frac{\alpha\,\delta_p}{(1+\alpha)} \quad ; \quad \delta_{s,G} = \delta_{s,I} = \delta_s \tag{4.4.5}$$

Por otro lado, el valor del ancho de banda de transición del filtro de forma depende del orden de interpolación utilizado, como indica la expresión

$$\Delta f_G = L\,\Delta f_H \tag{4.4.6}$$

mientras que el ancho de banda de transición del filtro interpolador es

$$\Delta f_I = \frac{1}{L} - \left(\Delta f_H + 2 f_p\right) \tag{4.4.7}$$

para el caso de filtros paso de bajas, paso de altas, paso de banda y rechazo de banda centrados en frecuencias múltiplos de $\frac{\pi}{L}$, siendo Δf_H el ancho de banda de transición del filtro global, y f_p la anchura de la banda de paso en el caso de filtros LP y HP o la mitad de la banda de paso en el caso BP o BS (Band Stop). Para el caso de filtros centrados en frecuencias no múltiplos de $\frac{\pi}{L}$, la expresión (4.4.7) no es válida, por lo que los resultados que obtengamos no podrán ser utilizados.

La velocidad de procesamiento del filtro IFIR depende del número de operaciones aritméticas que se efectúan. Minimizar el tiempo de procesamiento del sistema se traduce en encontrar el orden de interpolación que minimice el número de operaciones aritméticas del sistema global. Antes de continuar con el desarrollo teórico, vamos a definir los conceptos de *tiempo de procesamiento por muestra* (TPM) y *velocidad de procesamiento por muestra* (VPM), que nos permitirán comparar resultados entre distintas implementaciones. Definimos el tiempo de procesamiento por muestra de un sistema como el número de operaciones aritméticas efectuadas en un ciclo de computo (NOP) por número de muestras computadas

$$TPM = \frac{N° \; Operaciones \; Aritmeticas}{N° \; Muestras \; Computadas} \qquad (4.4.8)$$

En el caso de sistemas escalares el número de muestras computadas en un ciclo de cálculo es 1, por lo que el TPM coincide con el número de operaciones aritméticas. En sistemas en paralelo, como la implementación en bloques, el parámetro TPM es un indicativo del tiempo de procesamiento empleado por el sistema para computar una muestra. La velocidad de procesamiento por muestra la definimos como

$$VPM = (TPM)^{-1} \tag{4.4.9}$$

concepto que empleamos para definir otro parámetro de interés al que llamamos *eficiencia de la implementación* (η_H), definida como

$$\eta_H = \frac{VPM}{N° \ procesadores} \tag{4.4.10}$$

indicativo del costo hardware de la implementación.

El TPM del sistema de la figura 4.4 es la suma de los TPM de los filtros de forma interpolado e interpolador. A su vez, el TPM del filtro de forma interpolado es análogo al del filtro de forma, G(z), ya que, aunque el orden de ambos filtros es distinto, el número de coeficientes es el mismo. Por lo tanto, el TPM del filtro IFIR es

$$TPM_H = TPM_G + TPM_I \tag{4.4.11}$$

Encontrar el valor del orden de interpolación que minimice el TPM_H se reduce a calcular

$$\frac{d\,TPM_H}{d\,L} = \frac{d\,(TPM_G + TPM_I)}{d\,L} = 0$$
$$\frac{d\,TPM_G}{dL} = -\frac{d\,TMP_I}{dL} \tag{4.4.12}$$

En el caso de filtros FIR escalares el TPM coincide con el número de operaciones aritméticas (NOP), por lo que su valor será

$$TPM_H = \begin{cases} \textit{fase lineal} & \begin{cases} O_H \ \textit{par} & \rightarrow \dfrac{3\,O_H}{2} + 1 \\ O_H \ \textit{impar} & \rightarrow \dfrac{3\,O_H + 1}{2} \end{cases} \\ \textit{fase no lineal} & \rightarrow 2\,O_H + 1 \end{cases} \tag{4.4.13}$$

Teniendo en cuenta (4.4.13) y (4.4.12) llegamos a la conclusión de que la condición de mínimo se reduce a

$$\frac{d\,O_G}{d\,L} = -\frac{d\,O_I}{d\,L} \tag{4.4.14}$$

El valor aproximado de lo órdenes de los filtros de forma e interpolador viene dado por (4.4.2)

$$O_G = \frac{-20\log\sqrt{\frac{\delta_p\delta_s}{1+\alpha}} - 13}{14.6\,L\,\Delta f_H} \tag{4.4.15}$$

$$O_I = \frac{-20\log\sqrt{\frac{\delta_p\delta_s\,\alpha}{1+\alpha}} - 13}{14.6\left(\frac{1}{L} - (\Delta f_H + 2f_p)\right)} \tag{4.4.16}$$

por lo que, efectuando (4.4.14) llegamos a la siguiente expresión para el orden de interpolación que minimiza el TPM del sistema escalar

$$L = \frac{1}{\Delta f_H + 2f_p + \sqrt{\Delta f_H\left(1 + \frac{20\log\sqrt{\alpha}}{20\log\sqrt{\frac{\delta_p\delta_s}{1+\alpha}} + 13}\right)}} \tag{4.4.17}$$

que en el caso más usual, con $\alpha = 1$, deriva en

$$L = \frac{1}{\Delta f_H + 2f_p + \sqrt{\Delta f_H}} \tag{4.4.18}$$

4.4.3 Implementación Mixta Monodimensional

La implementación IFIR consigue reducir el número de coeficientes del filtro respecto a métodos convencionales, aumentando la velocidad de procesamiento del sistema escalar. El sistema global puede ser implementado usando técnicas de procesamiento en paralelo, como la implementación en bloques, disminuyendo el TPM del filtro IFIR. Sin embargo, observando la estructura de la figura 4.4, notamos que la implementación en bloques de longitud L del filtro de forma interpolado viene descrita por la expresión

$$\begin{bmatrix} Y_0^p \\ Y_1^p \\ \vdots \\ Y_{L-1}^p \end{bmatrix} = \begin{bmatrix} G(z^L) & 0 & \cdots & 0 \\ 0 & G(z^L) & \cdots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & G(z^L) \end{bmatrix} \begin{bmatrix} X_0^p \\ X_1^p \\ \vdots \\ X_{L-1}^p \end{bmatrix} \tag{4.4.19}$$

por lo que la implementación en bloques de longitud L del filtro de forma interpolado solo requiere L procesadores, es decir prácticamente la mitad del costo requerido usando algoritmos óptimos de convolución.

La estructura que proponemos en esta sección aprovecha esta particularidad de los filtros interpolados, implementando el filtro $G(z^L)$ en bloques de longitud L, y eliminando las imágenes no deseadas con el filtro interpolador (Figura 4.5)

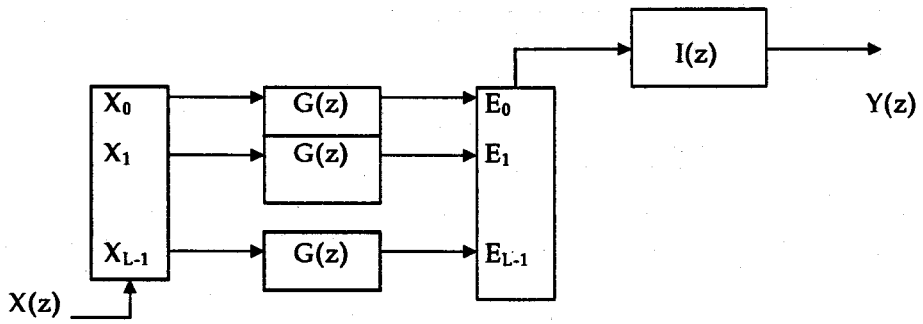

Figura 4.5

obteniendo una implementación L+1 procesadores.

4.4.4 Elección del Orden de Interpolación

La implementación de la figura 4.5 tiene una estructura multitasa. En el apartado 4.4.2 encontramos una expresión para el orden de interpolación que conducía al sistema escalar con menor número de coeficientes, y por lo tanto con mayor velocidad de procesamiento. Sin embargo, la velocidad de procesamiento del sistema de la figura 4.5 viene dado por la velocidad de procesamiento del bloque más lento, por lo que la elección del orden de interpolación dado por (4.4.17) no corresponderá, en general, al sistema con mayor velocidad de procesamiento.

Para encontrar el valor óptimo vamos a hacer ciertas consideraciones cualitativas. Por un lado, el orden del filtro de forma disminuye a medida que aumenta el orden de interpolación, mientras que se incrementa el orden del filtro interpolador, por lo que valores elevados del orden de interpolación harán que el bloque más lento sea el filtro interpolador, tanto más lento cuanto mayor sea el orden de interpolación. Por otro lado, si el orden de interpolación

es lo suficientemente bajo para que el bloque más lento sea el formado por el filtro de forma interpolado actuando en paralelo, entonces el sistema con mayor velocidad de procesamiento será aquel con mayor orden de interpolación. Confrontando ambas tendencias llegamos a la conclusión de que el valor óptimo para el orden de interpolación será el valor que obtengamos de

$$TPM_G = TPM_I \tag{4.4.20}$$

que teniendo en cuenta la definición (4.4.8) se traduce en

$$\frac{NOP_G}{L} = NOP_I \tag{4.4.21}$$

El número de operaciones aritméticas (NOP) depende del tipo de filtro FIR. En el caso de filtros FIR lineales el número de operaciones aritméticas puede expresarse como

$$NOP_H = \frac{3O_H}{2} + \phi_H \tag{4.4.22}$$

donde el coeficiente ϕ_H es

$$\phi_H = \begin{cases} 1 & si\ O_H\ es\ par \\ 1/2 & si\ O_H\ es\ impar \end{cases} \tag{4.4.23}$$

mientras que en el caso de filtros no lineales, el número de operaciones aritméticas viene dado por

$$NOP_H = 2\,O_H + 1 \tag{4.4.24}$$

Fijándonos en primer lugar en el caso de filtros lineales, la identidad (4.4.21) se traduce en

$$O_G - O_I\ L + \frac{2}{3}(\phi_G - L\phi_I) = 0 \tag{4.4.25}$$

que al sustituir (4.4.15) y (4.4.16) en (4.4.25) queda

$$\frac{-20\log\sqrt{\frac{\delta_p\delta_s}{1+\alpha}} - 13}{14.6\ L\ \Delta f_H} - \frac{\left(-20\log\sqrt{\frac{\delta_p\delta_s\ \alpha}{1+\alpha}} - 13\right)L^2}{14.6\left(1 - L\left(\Delta f_H + 2\ f_p\right)\right)} + \frac{2}{3}\left(\phi_G - L\ \phi_I\right) = 0 \tag{4.4.26}$$

Reagrupando términos llegamos a la siguiente condición

$$L^3 + a_1\ L^2 + a_2\ L + a_3 = 0 \tag{4.4.27}$$

donde

$$a_1 = -\frac{\frac{29.2}{3}\left(\phi_I + \phi_G\left(\Delta f_H + 2\ f_p\right)\right)}{\left(20\log\sqrt{\frac{\delta_p\delta_s\ \alpha}{1+\alpha}} + 13\right) + \frac{29.2}{3}\left(\Delta f_H + 2\,f_p\right)\phi_I} \tag{4.4.28}$$

$$a_2 = \frac{\left(\Delta f_H + 2f_p\right)\left(20\log\sqrt{\frac{\delta_p\delta_s}{1+\alpha}} + 13\right) + \frac{29.2}{3}\Delta f_H\ \phi_G}{\left(\left(20\log\sqrt{\frac{\delta_p\delta_s\,\alpha}{1+\alpha}} + 13\right) + \frac{29.2}{3}\left(\Delta f_H + 2f_p\right)\phi_I\right)\Delta f_H} \qquad (4.4.29)$$

$$a_3 = -\frac{20\log\sqrt{\frac{\delta_p\delta_s}{1+\alpha}} + 13}{\left(\left(20\log\sqrt{\frac{\delta_p\delta_s\,\alpha}{1+\alpha}} + 13\right) + \frac{29.2}{3}\left(\Delta f_H + 2f_p\right)\phi_I\right)\Delta f_H} \qquad (4.4.30)$$

La solución de la ecuación de tercer grado (4.4.27) la resolvemos mediante la fórmula de Tartaglia [Rey y Castro 67], proporcionándonos el valor del orden de interpolación que minimiza el TPM del sistema

$$L = \sqrt[3]{-\frac{q}{2} + \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}} + \sqrt[3]{-\frac{q}{2} - \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}} \qquad (4.4.31)$$

donde

$$p = a_2 - \frac{a_1^2}{3}$$
$$q = \frac{2\,a_1^3}{27} - \frac{a_1\,a_2}{3} + a_3 \qquad (4.4.32)$$

En el caso de filtros FIR no lineales, partiendo de (4.4.24) llegamos a la expresión (4.4.27), donde los coeficientes son

$$a_1 = -\frac{7.3\left(1+\left(\Delta f_H + 2f_p\right)\right)}{\left(20\log\sqrt{\frac{\delta_p \delta_s \alpha}{1+\alpha}} + 13\right) + 7.3\left(\Delta f_H + 2f_p\right)} \tag{4.4.33}$$

$$a_2 = \frac{\left(\Delta f_H + 2f_p\right)\left(20\log\sqrt{\frac{\delta_p \delta_s}{1+\alpha}} + 13\right)}{\left(\left(20\log\sqrt{\frac{\delta_p \delta_s \alpha}{1+\alpha}} + 13\right) + 7.3\left(\Delta f_H + 2f_p\right)\right)\Delta f_H} \tag{4.4.34}$$

$$a_3 = -\frac{20\log\sqrt{\frac{\delta_p \delta_s}{1+\alpha}} + 13}{\left(\left(20\log\sqrt{\frac{\delta_p \delta_s \alpha}{1+\alpha}} + 13\right) + 7.3\left(\Delta f_H + 2f_p\right)\right)\Delta f_H} \tag{4.4.35}$$

y donde el valor del orden de interpolación óptimo viene dado por (4.4.31).

4.4.5 Limitaciones de Diseño

Con la estructura propuesta es posible diseñar filtros paso de bajas (LP), paso de altas (HP), paso de banda (BP) y rechazo de banda (BS), siempre que los anchos de banda de los filtros no sobrepasen unos límites. En el caso de

filtros LP, HP, el ancho de banda máximo de los filtros viene dado por la siguiente expresión

$$B < \frac{\pi}{L} \tag{4.4.36}$$

donde L es el orden de interpolación utilizado. En el caso de filtros BP y BS, si la frecuencia central del filtro es múltiplo de $\frac{\pi}{L}$, el ancho de banda máximo que es posible diseñar con esta técnica viene limitado por la expresión

$$B < \frac{2\pi}{L} \tag{4.4.37}$$

Finalmente, en el caso de filtros BP y BS, cuya frecuencia central no es múltiplo de $\frac{\pi}{L}$, la elección del orden de interpolación se realiza por un proceso de eliminación. En primer lugar se debe calcular los posibles valores del orden de interpolación que permiten diseñar filtros con el ancho de banda especificado. Estos valores del orden de interpolación vienen dados por la siguiente expresión

$$L < \frac{\pi}{B} \tag{4.4.38}$$

A continuación, se evalúa para cada valor permitido del orden de interpolación el coeficiente

$$\omega_0 = \left| \omega_c - \frac{2\pi k}{L} \right| \tag{4.4.39}$$

siendo k el entero que más aproxima $\frac{2\pi k}{L}$ al valor de la frecuencia central del filtro ω_c. El proceso de discriminación termina eliminando los órdenes de interpolación que no cumplen la siguiente expresión

$$\frac{B}{2} < \omega_0 < \frac{\pi}{L} - \frac{B}{2} \tag{4.4.40}$$

Para obtener el orden de interpolación óptimo de la implementación, es necesario diseñar cada uno de los casos válidos, eligiendo posteriormente el de mayor velocidad de procesamiento.

Como ejemplo, vamos a mostrar el proceso de elección del orden de interpolación óptimo para el diseño de un filtro BP con frecuencia central $\omega_c = 0.7\pi$, y ancho de banda $B = 0.2\pi$. De la expresión (4.4.38) se deduce que los valores posibles para el orden de interpolación son L<5. Calculando (4.4.39) para cada valor de L, obtenemos los siguientes resultados para el coeficiente ω_0

L	2	3	4
ω_0	0.3π	0.033π	0.2π

Finalmente, de los valores posibles solo L=2 cumple la expresión (4.4.40), convirtiéndose en el orden de interpolación óptimo con implementación IFIR o mixta.

4.5 ESTRUCTURAS BIDIMENSIONALES BASADAS EN LA TEORÍA DE FILTROS IFIR

Dentro del procesamiento bidimensional, el uso de filtros con respuesta impulsiva finita está ampliamente difundido por ofrecer ciertas ventajas sobre los filtros recursivos, entre las que se encuentran la facilidad de diseño, estabilidad y posibilidad de diseñar filtros de fase cero. Desgraciadamente el número de coeficientes que requieren es muy elevado, problema de especial importancia debido al gran volumen de datos que implica el procesamiento digital bidimensional.

En esta sección vamos a generalizar la idea de la implementación IFIR al diseño de filtros FIR bidimensionales. Deduciremos una expresión para el orden de interpolación que minimiza el número de coeficientes de la

implementación escalar, suponiendo que los filtros FIR bidimensionales de forma, $G(z_1,z_2)$, e interpolador, $I(z_1,z_2)$, se diseñan mediante transformación [Dudgeon y Mersereau 84] de filtros FIR monodimensionales equiripple.

Finalmente, propondremos la implementación mixta bidimensional como alternativa a la de bloques, que permitirá reducir drásticamente el TPM del sistema, manteniendo el costo hardware significativamente bajo.

4.5.1 Filtros IFIR Bidimensionales

Un filtro FIR interpolado bidimensional está formado por la cascada de dos filtros (Figura 4.6), a los que denominaremos filtro de forma interpolado, $G(z_1^L,z_2^L)$, y filtro interpolador, $I(z_1,z_2)$, diseñados de tal forma que una de las imágenes generadas en el proceso de interpolación del filtro de forma cumpla las especificaciones de diseño en lo referente al ancho de banda y ancho de banda de transición, mientras que el filtro interpolador elimina las imágenes no deseadas, cumpliendo el filtro global las especificaciones exigidas.

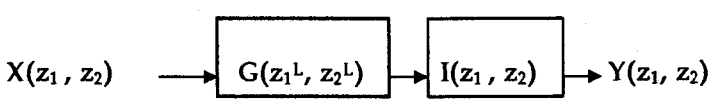

Figura 4.6

Como hemos mencionado, vamos a centrar nuestro estudio en filtros bidimensionales diseñados mediante transformación de filtros

monodimensionales equiripple. Hemos elegido la técnica de transformación por la facilidad de diseño, ya que nos permite utilizar métodos clásicos de diseño de filtros monodimensionales como el algoritmo de Parks y McClellan [Oppenheim y Schafer 75], y aún más importante, por la posibilidad de implementar los filtros resultantes empleando la estructura óptima propuesta por Mecklenbräuker y Mersereau [Dudgeon y Mersereau 84] (Figura 4.7). Hacemos notar que, aunque el desarrollo teórico está basado en filtros de fase cero, la generalización a filtros de fase lineal es inmediata [Soo-Chang y Jong-Jy 96].

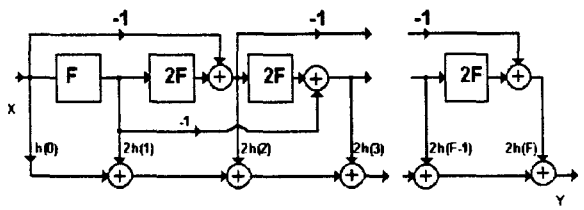

Figura 4.7

Un filtro FIR lineal de fase cero puede representarse en el dominio transformado como

$$H(z_1,z_2) = h(0,0) + \sum_{n_1=1}^{N_1} h(n_1,0)\left(z_1^{-n_1} + z_1^{n_1}\right) + \sum_{n_2=1}^{N_2} h(0,n_2)\left(z_2^{-n_2} + z_2^{n_2}\right) +$$
$$+ \sum_{n_1=1}^{N_1}\sum_{n_2=1}^{N_2} h(n_1,n_2)\left(z_1^{n_1}z_2^{n_2} + z_1^{-n_1}z_2^{-n_2}\right) + \sum_{n_1=1}^{N_1}\sum_{n_2=1}^{N_2} h(n_1,n_2)\left(z_1^{n_1}z_2^{-n_2} + z_1^{-n_1}z_2^{n_2}\right)$$

(4.5.1)

El número de operaciones aritméticas que se efectúan al implementar (4.5.1) con estructura directa es

$$NOP = \begin{cases} N^{\circ}\ de\ productos = 2\,N_1N_2 + N_1 + N_2 + 1 \\ N^{\circ}\ de\ adiciones = 2\left(2\,N_1N_2 + N_1 + N_2\right) \\ Total = 6\,N_1\,N_2 + 3\left(N_1 + N_2\right) + 1 \end{cases} \qquad (4.5.2)$$

De esta forma, si un filtro $H(z_1,z_2)$ ha sido diseñado por transformación de un filtro monodimensional de orden 2N, siendo $F(z_1,z_2)$ el polinomio transformador

$$F(z_1,z_2) = \sum_{n_1=-F_1}^{F_1} \sum_{n_2=-F_2}^{F_2} f(n_1,n_2)\, z_1^{-n_1} z_2^{-n_2} \qquad (4.5.3)$$

el número de operaciones aritméticas que se efectúan es

$$NOP_H = \begin{cases} N^{\circ}\ \text{Pr}oductos &= 2\,N\,F_1\,F_2 + N\left(F_1 + F_2\right) + 2\,N + 1 \\ N^{\circ}\ Adiciones &= N\left(4\,F_1\,F_2 + 2\left(F_1 + F_2\right) + 1\right) + N - 1 \\ Total &= 6\,N\,F_1\,F_2 + 3\,N\left(F_1 + F_2\right) + 4\,N \end{cases} \qquad (4.5.4)$$

utilizando la estructura óptima [Dudgeon y Mersereau 84], e implementando el filtro transformador con estructura directa (4.5.2).

4.5.2 Orden de Interpolación Óptimo

Hemos visto como el orden de interpolación es un parámetro fundamental para determinar la velocidad de procesamiento de los filtros IFIR monodimensionales. De forma análoga, la velocidad de procesamiento de los filtros IFIR 2D vendrá impuesta por el orden de interpolación elegido. En este apartado deduciremos una expresión aproximada para el orden de interpolación que minimiza el número de operaciones aritméticas del sistema global, suponiendo que los filtros bidimensionales se obtienen mediante transformación de filtros monodimensionales equiripple, de tal forma que, si el orden del filtro generador monodimensional es 2N, el orden del filtro bidimensional es N x N. Así mismo, supondremos que los órdenes de los filtros transformadores del filtro de forma y filtro interpolador son F_G x F_G y F_I x F_I respectivamente.

De acuerdo con las suposiciones efectuadas, el orden del filtro generador del filtro de forma, $G(z_1,z_2)$, vale aproximadamente

$$2\,N_G = \frac{-20\log\sqrt{\frac{\delta_p\delta_s}{1+\alpha}} - 13}{14.6\,L\,\Delta R_H} \qquad (4.5.5)$$

donde ΔR_H es la especificación para el radio de banda de transición del filtro global, y donde el rizado en las bandas del filtro de forma y filtro interpolador

cumplen las relaciones (4.4.5). De igual forma, el orden del filtro generador del filtro interpolador es

$$2\,N_I = \frac{-20\log\sqrt{\frac{\delta_p \delta_s\,\alpha}{1+\alpha}} - 13}{14.6\left(\frac{1}{L} - \left(\Delta R_H + 2\,R_p\right)\right)} \tag{4.5.6}$$

donde R_p es el radio de paso del filtro global, en el caso de filtros LP y HP, o la mitad del radio de la banda en el caso de filtros BP y BS.

El número de operaciones aritméticas del sistema global es la suma de las operaciones efectuadas en el filtro de forma interpolado, que coincide con el número de operaciones del filtro de forma, y filtro interpolador

$$NOP_H = NOP_G + NOP_I \tag{4.5.7}$$

La condición de mínimo se traduce en

$$\frac{d\,NOP_G}{d\,L} = -\frac{d\,NOP_I}{d\,L} \tag{4.5.8}$$

que teniendo en cuenta (4.5.4), conduce a la relación

$$\frac{d\,N_G}{d\,L} = -\frac{\left(3\,F_I^2 + 3\,F_I + 2\right)}{\left(3\,F_G^2 + 3\,F_G + 2\right)}\;\frac{d\,N_I}{d\,L} \tag{4.5.7}$$

En el caso de que $F_G = F_I$, la expresión (4.5.7) es idéntica a (4.4.13), por lo que el valor del orden de interpolación óptimo viene dado por (4.4.16). En el caso más general, el valor del orden de interpolación viene dado por

$$L = \frac{1}{\Delta R_H + 2R_p + \sqrt{\dfrac{\Delta f_H\left(1 + \dfrac{20\log\sqrt{\alpha}}{20\log\sqrt{\dfrac{\delta_p\delta_s}{1+\alpha}} + 13}\right)\left(3\,F_I^2 + 3\,F_I + 2\right)}{\left(3\,F_G^2 + 3\,F_G + 2\right)}}} \tag{4.5.8}$$

4.5.3 Implementación Mixta Bidimensional

Aunque con la expresión deducida en el apartado anterior se consiguen filtros FIR 2D con reducido número de operaciones aritméticas, el elevado número de datos que caracteriza el procesado bidimensional puede convertir en insuficiente el aumento en la velocidad de procesamiento conseguido con esta técnica de diseño.

Como indicamos en el apartado 4.2.4, implementando el sistema bidimensional en bloques, empleando una matriz de submuestreo adecuada, se puede conseguir la velocidad de procesamiento necesaria para la aplicación en

la cual se emplea el filtro digital. Desafortunadamente, el número de procesadores requeridos es muy elevado, incluso para un número pequeño de muestras procesadas en paralelo.

Observando el esquema en bloques de la implementación IFIR bidimensional (Figura 4.6), nos damos cuenta de que si el filtro de forma interpolado lo implementamos en bloques bidimensionales con matriz de submuestreo

$$M = \begin{bmatrix} L & 0 \\ 0 & L \end{bmatrix} \tag{4.5.9}$$

la matriz de bloques asociada a la matriz de submuestreo (4.5.9) es de la forma (4.2.44)

$$G_{Bloques}(z_1^L, z_2^L) = \begin{bmatrix} G(z_1^L, z_2^L) & 0 & \cdots & 0 \\ 0 & G(z_1^L, z_2^L) & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & G(z_1^L, z_2^L) \end{bmatrix} \tag{4.5.10}$$

matriz diagonal de dimensiones $L^2 \times L^2$, lo que significa que la implementación en bloques solo requiere L^2 procesadores, frente a los $4L^2 - 4L + 1$ que requeriría una implementación en bloques rectangular de un filtro bidimensional convencional, con idéntica longitud de bloque. La generalización de este método de diseño para matrices de submuestreo no rectangulares es un

objetivo que nos hemos marcado como línea futura de investigación, como explicaremos en el capítulo dedicado a este fin.

Limitándonos al caso de implementaciones IFIR 2D con orden de interpolación L, proponemos la siguiente implementación como alternativa a la implementación en bloques bidimensionales de longitud $L \times L$ (Figura 4.8).

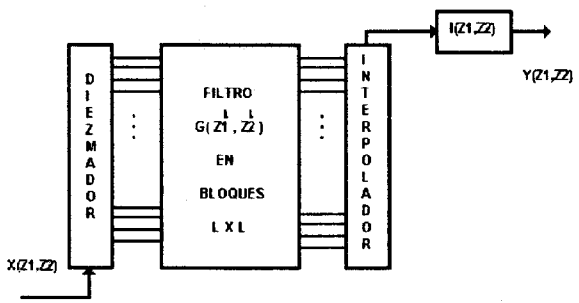

Figura 4.8

La estructura de la figura 4.8, a la que denominamos estructura mixta bidimensional, está compuesta por L^2+1 procesadores, y la velocidad de procesamiento viene determinado por el bloque más lento. Igual que en la implementación mixta monodimensional, esta técnica de diseño puede ser utilizada siempre que el ancho de banda del filtro no supere ciertos límites, que coinciden con los mostrados en el apartado 4.4.5, donde en el caso bidimensional la limitación sobre el parámetro B se refiere al radio del ancho de banda.

4.5.4 Elección del Orden de Interpolación

La velocidad de procesamiento de la implementación mixta bidimensional depende del bloque más lento. Análogamente al caso monodimensional, la elección del orden de interpolación calculado mediante (4.5.8) no conducirá, en general, a la implementación mixta con menor tiempo de procesamiento por muestra (TPM). Utilizando las mismas consideraciones cualitativas que en la implementación mixta monodimensional, llegamos a la conclusión que la estructura mixta 2D con menor tiempo de procesamiento por muestra será aquella que iguale los TPM de los filtros de forma interpolado, implementado en paralelo, y del filtro interpolador (4.4.20), que sustituyendo la definición de TPM implica

$$\frac{NOP_G}{L^2} = NOP_I \tag{4.5.11}$$

Sustituyendo el valor del número de operaciones aritméticas que se efectúan en el filtro de forma y filtro interpolador (4.5.4) llegamos a la siguiente igualdad

$$\frac{N_G}{L^2} = \frac{\left(6F_{1I}F_{2I} + 3(F_{1I} + F_{2I}) + 4\right)}{\left(6F_{1G}F_{2G} + 3(F_{1G} + F_{2G}) + 4\right)} N_I \tag{4.5.12}$$

que teniendo en cuenta (4.5.5) y (4.5.6) se transforma en la siguiente expresión

$$L^4 \frac{\left(-20\log\sqrt{\frac{\delta_p\delta_s\alpha}{1+\alpha}}-13\right)}{\left(-20\log\sqrt{\frac{\delta_p\delta_s}{1+\alpha}}-13\right)} \frac{\left(6F_{1I}F_{2I}+3(F_{1I}+F_{2I})+4\right)}{\left(6F_{1G}F_{2G}+3(F_{1G}+F_{2G})+4\right)} \Delta R_H + L\left(\Delta R_H + 2R_p\right) - 1 = 0 \tag{4.5.13}$$

que, una vez evaluada, muestra el valor del orden de interpolación que minimiza el TPM de la implementación mixta bidimensional. En el caso particular de que el rizado en la banda de paso de los filtros de forma e interpolador sean idénticos, y los filtros transformadores tengan el mismo orden, la expresión (4.5.13) queda de la forma

$$L^4 \Delta R_H + L\left(\Delta R_H + 2R_p\right) - 1 = 0 \tag{4.5.14}$$

4.5.5 Implementación Mixta de Filtros con Respuesta Impulsiva Infinita

Con la implementación mixta es posible aumentar la velocidad de procesamiento de filtros FIR diseñados mediante interpolación (IFIR), manteniendo el costo hardware razonablemente bajo. Aunque el desarrollo teórico efectuado solo ha contemplado el diseño de filtros FIR, la generalización para filtros IIR es conceptualmente posible.

Como en el caso de filtros FIR, un filtro IIR interpolado está formado por la cascada del filtro de forma interpolado y el filtro interpolador, y donde uno de ellos, o ambos, se diseña como filtro con respuesta impulsiva infinita. Los límites de diseño, en lo referente al ancho de banda del filtro resultante, coinciden con los límites mostrados para el diseño de filtros FIR (4.4.36-40). Sin embargo, los resultados obtenidos para el orden de interpolación óptimo, según el tipo de implementación que se utilice, no son válidos para el diseño de filtros IIR. La obtención de expresiones para el orden de interpolación de filtros IIR, empleando la estructura mixta, es un objetivo para futuras investigaciones.

A pesar de no disponer de resultados teóricos que avalen esta línea de investigación , es posible extraer una serie de conclusiones de los resultados de diseño obtenidos para un conjunto de ejemplos que mostraremos en la siguiente sección. El resultado principal es que, si bien la implementación IIR escalar no es un método apropiado para el diseño de filtros IIR, el diseño con estructura mixta es una alternativa a la implementación en bloques de filtros IIR, consiguiendo velocidades de procesamiento similares con menor costo hardware.

4.5.6 Diseño de Filtros IIR Bidimensionales

Antes de mostrar los resultados obtenidos en diversos diseños de filtros digitales, vamos a explicar la técnica que hemos utilizado para diseñar los filtros IIR bidimensionales escalares. El método, conceptualmente, no es original, sin embargo conviene resaltar una importante característica que utilizaremos en el diseño de filtros IIR 2D y su implementación mixta.

Fijadas unas especificaciones de diseño para un filtro IIR bidimensional, si el filtro resultante se obtiene a partir de un filtro IIR monodimensional como

$$H_{2D}(z_1, z_2) = H_{1D}(z_1) H_{1D}(z_2) \tag{4.5.15}$$

si el filtro IIR monodimensional es de la forma

$$H_{1D}(z) = \frac{\sum_{n=0}^{N} a_n z^{-n}}{1 + \sum_{m=1}^{M} b_m z^{-m}} \tag{4.5.16}$$

entonces el filtro bidimensional es de la forma

$$H_{2D}(z_1,z_2)=\frac{\sum_{n_1=0}^{N}\sum_{n_2=0}^{N}A_{n_1,n_2}z_1^{-n_1}z_2^{-n_2}}{1+\sum_{\substack{m_1=0\\ m_1+m_2\neq 0}}^{M}\sum_{m_2=0}^{M}B_{m_1,m_2}z_1^{-m_1}z_2^{-m_2}} \tag{4.5.17}$$

y tiene la siguiente simetría en los coeficientes

$$\begin{aligned} A_{n_1,n_2} &= A_{n_2,n_1} \\ B_{m_1,m_2} &= B_{m_2,m_1} \end{aligned} \tag{4.5.18}$$

por lo que el número de operaciones aritméticas que se efectúan por muestra computada es

$$NOP_{IIR2D}=\frac{3}{2}\left(N^2+M^2\right)+\frac{7}{2}\left(M+N\right)+1 \tag{4.5.19}$$

que en el caso en que los órdenes del numerador y denominador coincidan, el número de operaciones aritméticas efectuadas se expresa como

$$NOP_{IIR2D}=3N^2+7N+1 \tag{4.5.20}$$

En el caso de que el numerador del filtro monodimensional sea además un filtro FIR de fase lineal, aún es posible reducir el número de operaciones aritméticas aprovechando las siguientes simetrías en los coeficientes

$$A_{n_1,n_2} = A_{N-n_1,n_2} = A_{n_1,N-n_2} = A_{n_2,n_1} = A_{N-n_2,n_1} = A_{n_2,N-n_1} = A_{N-n_1,N-n_2} = A_{N-n_2,N-n_1} \tag{4.5.21}$$

que reduce el número de operaciones aritméticas a

$$NOP_{IIR2D} = \begin{cases} \frac{9}{8}N^2 + \frac{3}{2}M^2 + \frac{11}{4}N + \frac{7}{2}M + 2 & N\ par \\ \frac{9}{8}N^2 + \frac{3}{2}M^2 + \frac{5}{2}N + \frac{7}{2}M + \frac{11}{8} & N\ impar \end{cases} \tag{4.5.22}$$

Para esclarecer los resultados obtenidos vamos a comprobar el número de operaciones aritméticas que se efectuarían en un filtro IIR bidimensional de orden N x N, suponiendo que no exista simetría en los coeficientes, aprovechando la simetría (4.5.18), y con simetría (4.5.21)

N	*1*	*2*	*3*	*4*	*5*
No Simetr.	13	33	61	97	141
Sim. (4.5.18)	11	27	49	77	111
Sim. (4.5.21)	9	24	42	68	95

Número de Operaciones Aritméticas IIR

4.6 COMPARACIÓN DE RESULTADOS

En esta sección vamos a mostrar algunos resultados obtenidos para el diseño de filtros digitales monodimensionales y bidimensionales. Los diferentes casos se han diseñado tanto como filtros FIR como IIR, empleando métodos clásicos de diseño escalar y en bloques, así como con técnicas de interpolación (IFIR, IIIR e implementaciones mixtas). Tanto en el caso monodimensional como bidimensional, los diseños de los filtros IIR se han efectuado como filtros elípticos, por considerar que son los casos más desfavorables al aplicar estructuras basadas en técnicas de interpolación.

Los parámetros que vamos a comparar entre las distintas implementaciones son el número de retrasos requeridos para la implementación (solo en implementaciones escalares), orden de interpolación o longitud de bloque empleado, tiempo de procesamiento por muestra (TPM) (4.4.8), cuya reducción es el principal objetivo de esta tesis, número de procesadores empleados, y eficiencia de la implementación (4.4.10).

En las comparaciones entre la implementación en bloques e implementación mixta, la longitud del bloque elegido coincide con el orden de interpolación óptimo de la implementación mixta.

Finalmente, la técnica utilizada para filtros BP bidimensionales conduce a filtros IIR con distinto comportamiento frecuencial a los filtros FIR obtenidos por transformación. Es posible, aplicando la misma técnica de diseño, obtener filtros FIR con respuesta frecuencial análoga a los filtros IIR diseñados.

4.6.1 Comparación de Resultados Monodimensionales

Los tres casos de diseño que se van a contemplar vienen esquematizados a continuación:

1) Filtro LP con frecuencia de paso $\omega_p = 0.05\ \pi$, frecuencia de corte $\omega_s = 0.1\ \pi$, rizado en la banda de paso $\delta_p = 0.01$ y rizado en la banda de rechazo $\delta_s = 0.001$.

2) Filtro HP con frecuencia de paso $\omega_p = 0.919\ \pi$, frecuencia de corte $\omega_s = 0.888\ \pi$, rizado en la banda de paso y de rechazo $\delta_p = \delta_s = 0.023$.

3) Filtro BP con frecuencia central $\omega_c = 0.7\ \pi$, anchura en la banda de paso $B_p = 0.08\ \pi$, ancho de banda $B = 0.2\ \pi$, rizado en la banda de paso $\delta_p = 0.01$ y rizado en la banda de rechazo $\delta_s = 0.001$.

	Filtro LP				
	Z	L	TPM	Nº Procesadores	η
FIR_{conv}	108	1	163	1	0.0061
IFIR	137	4	72	1	0.014
$Bloq_{Fir}$	-	3	30	5	0.0067
Mix_{Fir}	-	3	20.33	4	0.012
IIR	12	1	21	1	0.048
IIIR	28	2	29	1	0.034
$Bloq_{Iir}$	-		14	3	0.024
Mix_{Iir}	-	2	10.5	3	0.032

	Filtro HP				
	Z	L	TPM	Nº Procesadores	η
FIR_{conv}	102	1	154	1	0.0065
IFIR	125	4	63	1	0.016
$Bloq_{Fir}$	-	3	28.6667	5	0.007
Mix_{Fir}	-	3	19.333	4	0.013
IIR	10	1	17	1	0.056
IIIR	24	2	24	1	0.042
$Bloq_{Iir}$	-	2	10.5	3	0.032
Mix_{Iir}	-	2	8.5	3	0.04

	Filtro BP				
	Z	L	TPM	Nº Procesadores	η
FIR_{conv}	90	1	136	1	0.007
IFIR	128	2	116	1	0.0086
$Bloq_{Fir}$	-	2	47	3	0.007
Mix_{Fir}	-	2	26	3	0.013
IIR	20	1	35	1	0.03
IIIR	50	2	52	1	0.02
$Bloq_{Iir}$	-	2	21.5	3	0.0155
Mix_{Iir}	-	2	17.5	3	0.019

4.6.2 Comparación de Resultados Bidimensionales

El diseño de los filtros bidimensionales los hemos efectuado por transformación [Dudgeon y Mersereau 84], en el caso de filtros con respuesta impulsiva finita, mientras que los filtros IIR se han diseñado como cascada de dos sistemas de variables separables, correspondientes a dos filtros elípticos monodimensionales. Las especificaciones de diseño, referidas a un eje coordenado de frecuencias son los siguientes:

1. Filtro LP con frecuencia de paso ω_p=0.05 π, frecuencia de corte ω_s=0.2 π, rizado en la banda de paso δ_p=0.01, y rizado en la banda de rechazo δ_s=0.001.
2. Filtro BP con frecuencia central ω_c=0.2 π, frecuencias de paso ω_p=(0.2±0.05) π, frecuencias de corte ω_s=(0.2±0.1) π, rizado en la banda de paso δ_p=0.01, y rizado en la banda de rechazo δ_s=0.001.

En este último caso los filtros FIR diseñados por transformación generarán estructuras con un comportamiento espectral con simetría circular (figura 4.9)

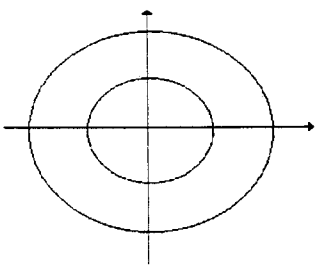

Figura 4.9

mientras que los filtros IIR presentan una característica de la forma (figura 4.10)

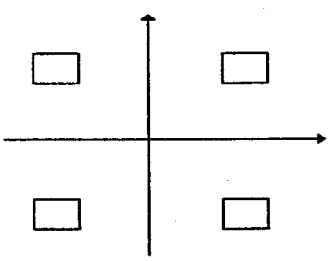

Figura 4.10

	Filtro LP				
	Z	L	TPM	Nº Procesadores	η
FIR_{conv}	1296	1	576	1	0.0017
IFIR	3969	3	480	1	0.0021
$Bloq_{Fir}$	-	2	184	9	0.0006
Mix_{Fir}	-	2	160	5	0.00125
IIR	50	1	68	1	0.015
IIIR	288	2	109	1	0.0092
$Bloq_{Iir}$	-	2	23.25	9	0.0048
Mix_{Iir}	-	2	42	5	0.0047

	Filtro BP				
	Z	L	TPM	Nº Procesadores	η
FIR_{conv}	12321	1	1760	1	0.00057
IFIR	17689	2	1152	1	0.00087
$Bloq_{Fir}$	-	2	1571.5	9	0.000071
Mix_{Fir}	-	2	240	5	0.00083
IIR	338	1	454	1	0.0022
IIIR	1800	2	550	1	0.0018
$Bloq_{Iir}$	-	2	136.75	9	0.00081
Mix_{Iir}	-	2	113.5	5	0.0018

4.7 APÉNDICE: ESTABILIDAD EN IMPLEMENTACIONES EN BLOQUES

Al implementar un filtro digital es fundamental que la estructura utilizada sea estable. En los trabajos efectuados por [Barnes y Shinnaka 80] y [Chwen y Alexander 87] se demuestra la estabilidad de la implementación en bloques de filtros monodimensionales y bidimensionales con la propiedad de invariancia al desplazamiento por bloques.

Basándonos en algunos resultados obtenidos en este capítulo, vamos a demostrar en este apéndice que dada una matriz de submuestreo arbitraria, siempre es posible encontrar una implementación en bloques estable asociada a una matriz de submuestreo equivalente a la matriz de submuestreo original. La demostración se efectúa para matrices de submuestreo bidimensionales, aunque es fácilmente extrapolable al caso monodimensional y multidimensional.

Para demostrar la estabilidad de la implementación en bloques de un sistema bidimensional partimos de la hipótesis de que el sistema

$$H(z_1,z_2)=\frac{N(z_1,z_2)}{D(z_1,z_2)} \tag{4.7.1}$$

es estable. Esto implica, de acuerdo con el teorema de Shanks [Dudgeon y Mersereau 84], que los ceros del polinomio denominador están dentro del bicírculo unidad, es decir $|z_1| < 1, |z_2| < 1$.

4.7.1 Matriz de Submuestreo Colineal de Enteros Positivos

Por otro lado, si la matriz que define la razón de muestreo es de la forma

$$P = \begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix} \tag{4.7.2}$$

hemos demostrado en este capítulo que siempre es posible encontrar una matriz de enteros con determinante unidad que verifique

$$\begin{bmatrix} m_{00} & 0 \\ m_{10} & m_{11} \end{bmatrix} = \begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} \tag{4.7.3}$$

$$c_{00}c_{11} - c_{01}c_{10} = \pm 1 \tag{4.7.4}$$

lo que convierte a la implementación en bloques asociada a la matriz de submuestreo M, y a la implementación asociada a P en equivalentes. La propiedad (4.7.3) puede ser más restrictiva, exigiendo que los elementos de la matriz de submuestreo colineal sean enteros positivos. Para demostrar esta

condición basta comprobar que siempre es posible encontrar un entero positivo k tal que,

$$\begin{bmatrix} n_{00} & 0 \\ n_{10} & n_{11} \end{bmatrix} = \begin{bmatrix} m_{00} & 0 \\ -m_{10} & m_{11} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ k & 1 \end{bmatrix} \tag{4.7.5}$$

si los elementos m_{ij} son enteros positivos, los elementos de la matriz N son enteros positivos. Esta propiedad se cumple para cualquier entero $k \geq \left\lceil \frac{m_{10}}{m_{11}} \right\rceil$.

4.7.2 Estabilidad de las Componentes Polifásicas

Una vez demostrado que dada una matriz de submuestreo arbitraria P (4.7.2), siempre es posible encontrar una matriz de submuestreo colineal equivalente de coeficientes positivos, pasamos a demostrar la estabilidad de la implementación en bloques asociada.

En el apartado 4.2.4 se demostró la equivalencia entre las componentes de la matriz de bloques y las componentes polifásicas asociadas a una matriz de submuestreo colineal (4.2.46). Se ha mostrado un algoritmo de cálculo de las componentes polifásicas basado en la relación entre las componentes de modulación y las componentes polifásicas. Como se demuestra en la elaboración del algoritmo, las componentes polifásicas del filtro escalar asociadas a la matriz de submuestreo vienen dadas por la expresión

$$H^{P}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l}(z_1^{m_{00}}z_2^{m_{10}}, z_2^{m_{11}}) = \frac{A^{P}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l}(z_1^{m_{00}}z_2^{m_{10}}, z_2^{m_{11}})}{B(z_1^{m_{00}}z_2^{m_{10}}, z_2^{m_{11}})} \tag{4.7.6}$$

donde

$$B(z_1^{m_{00}}z_2^{m_{10}}, z_2^{m_{11}}) = D(z_1, z_2) \times \prod_{\substack{k_2=0 \\ k_2+l_2\neq 0}}^{m_{00}-1} \prod_{l_2=0}^{m_{11}-1} D(z_1 W_{|M|}^{-(m_{11}k_2 - m_{10}l_2)}, z_2 W_{M|}^{-m_{00}l_2}) \tag{4.7.7}$$

es el denominador de todas las componentes polifásicas, y donde los numeradores de las distintas componentes polifásicas , $A^{P}_{k,\left\lceil \frac{m_{10}}{m_{00}}k \right\rceil+l}(z_1^{m_{00}}z_2^{m_{10}}, z_2^{m_{11}})$, corresponden a las componentes polifásicas del polinomio

$$A(z_1, z_2) = N(z_1, z_2)P(z_1, z_2) \tag{4.7.9}$$

siendo

$$P(z_1, z_2) = \prod_{\substack{k_2=0 \\ k_2+l_2\neq 0}}^{m_{00}-1} \prod_{l_2=0}^{m_{11}-1} D(z_1 W_{|M|}^{-(m_{11}k_2 - m_{10}l_2)}, z_2 W_{|M|}^{-m_{00}l_2}) \tag{4.7.10}$$

Por otro lado, si (z_{10}, z_{20}) es un cero del polinomio bidimensional $D(z_1, z_2)$, entonces $(z_{10}W_{|M|}^{(m_{11}k_2 - m_{10}l_2)}, z_{20}W_{|M|}^{m_{00}l_2})$ es un cero del polinomio

bidimensional $D(z_1 W_{|M|}^{-(m_{11}k_2 - m_{10}l_2)}, z_2 W_{|M|}^{-m_{00}l_2})$. Por lo tanto, si (z_{10}, z_{20}) es un polo de la función de transferencia escalar (4.7.1), entonces, como se deduce de (4.7.7), $(z_{10} W_{|M|}^{(m_{11}k_2 - m_{10}l_2)}, z_{20} W_{|M|}^{m_{00}l_2})$ con $k_2 = 0,1,\cdots,m_{00} - 1$, y $l_2 = 0,1,\cdots,m_{11} - 1$, serán polos de las componentes polifásicas. Sustituyendo $(z_{10} W_{|M|}^{(m_{11}k_2 - m_{10}l_2)}, z_{20} W_{|M|}^{m_{00}l_2})$ en (4.7.7) obtenemos que

$$B\left(z_{10}^{m_{00}} z_{20}^{m_{10}} W_{|M|}^{(m_{11}k_2 - m_{10}l_2)m_{00}} W_{|M|}^{m_{10}m_{00}l_2}, z_{20}^{m_{11}} W_{|M|}^{m_{00}m_{11}l_2}\right) = B\left(z_{10}^{m_{00}} z_{20}^{m_{10}}, z_{20}^{m_{11}}\right) = 0 \qquad (4.7.11)$$

que demuestra que si (z_{10}, z_{20}) es un polo de $H(z_1, z_2)$, entonces $\left(z_{10}^{m_{00}} z_{20}^{m_{10}}, z_{20}^{m_{11}}\right)$ es un polo de las componentes polifásicas asociadas a la matriz de submuestreo colineal M

$$M = \begin{bmatrix} m_{00} & 0 \\ m_{10} & m_{11} \end{bmatrix} \qquad (4.7.12)$$

Trabajando con la versión diezmada de las componentes polifásicas, llamando ξ_1, ξ_2 a las variables complejas asociadas a la razón de muestreo indicada por (4.7.12)

$$\begin{aligned} \xi_1 &= z_1^{m_{00}} z_2^{m_{10}} \\ \xi_2 &= z_2^{m_{11}} \end{aligned} \qquad (4.7.13)$$

si (z_{10}, z_{20}) es un polo de $H(z_1, z_2)$, entonces (ξ_{10}, ξ_{20}) es un polo de todas las componentes polifásicas $H^p_{k,\left\lceil \frac{m_{10}}{m_{00}} k \right\rceil + l}(\xi_1, \xi_2)$, donde

$$\begin{aligned} \xi_{10} &= z_{10}^{m_{00}} z_{20}^{m_{10}} \\ \xi_{20} &= z_{20}^{m_{11}} \end{aligned} \tag{4.7.14}$$

además, como la función de transferencia escalar es estable, entonces $|z_{10}| < 1, |z_{20}| < 1$. Teniendo en cuenta que siempre es posible encontrar una matriz de submuestreo colineal de enteros positivos equivalente a la matriz de submuestreo original, la expresión (4.7.14) demuestra que la implementación en bloques asociada a la matriz de submuestreo colineal de enteros positivos es estable si la función de transferencia escalar lo es.

4.8 RESUMEN

En este capítulo se han abordado las líneas de investigación marcadas en el capítulo anterior. De esta forma, en la sección 4.2 se aplica una nueva aproximación para la obtención de la matriz de bloques de filtros digitales bidimensionales. Se comenta el concepto de submuestreo y de descomposición polifásica asociada al submuestreo. Se muestra una serie de propiedades de la matriz de submuestreo, así como el concepto de descomposición polifásica equivalente, deduciéndose una expresión que relaciona las componentes

polifásicas entre descomposiciones polifásicas equivalentes. Se deriva la expresión general para la matriz de bloques de sistemas LTI bidimensionales, y se propone un algoritmo general de cálculo para las componentes polifásicas, independiente del tipo de matriz de submuestreo utilizada.

En la sección 4.3 se generaliza el uso de algoritmos rápidos de convolución lineal para la implementación en bloques de filtros digitales bidimensionales. La particularidad de la matriz de bloques de ser Toeplitz en 2 niveles permite obtener estructuras computacionales con reducido costo hardware, aplicando de forma iterada algoritmos de convolución monodimensionales.

En las secciones 4.4 y 4.5 se propone una nueva aproximación para la implementación de filtros digitales monodimensionales y bidimensionales, basadas en la teoría de filtros FIR interpolados. La estructura obtenida, denominada Estructura Mixta, consigue reducir el tiempo de procesamiento de la implementación con un costo en el número de procesadores significativamente menor que la implementación en bloques equivalente. Se deducen expresiones para el orden de interpolación que minimiza el tiempo de procesamiento de las diferentes implementaciones. Finalmente, se generaliza este tipo de implementación para el caso de filtros con respuesta impulsiva infinita.

En la sección 4.6 se comparan los resultados de diseño obtenidos para los distintos tipos de implementaciones.

Por último, en la sección 4.7 se incluye un apéndice donde se estudia la estabilidad de las implementaciones en bloques, demostrándose que si el filtro escalar es estable siempre existe, al menos, una implementación en bloques estable.

CAPÍTULO 5

EJEMPLOS, APLICACIONES, CONCLUSIONES Y LÍNEAS FUTURAS DE TRABAJO

5.1 INTRODUCCIÓN

En el desarrollo de esta tesis se han obtenido una serie de resultados teóricos y estructuras computacionales para la implementación de filtros digitales monodimensionales y bidimensionales. En este capítulo mostraremos de forma comparativa las diferentes estructuras mediante dos ejemplos de implementación de filtros digitales. Este capítulo incluye también un apartado donde se recogen algunas aplicaciones en las que la utilización de técnicas de procesamiento en paralelo, y por lo tanto las estructuras obtenidas en esta tesis, están justificadas. Para finalizar, expondremos una serie de conclusiones que resumen el trabajo desarrollado, así como un conjunto de líneas futuras de trabajo que han surgido en la elaboración de esta tesis.

5.2 EJEMPLOS

El propósito de este apartado es mostrar de forma comparativa distintas implementaciones de un mismo filtro digital, en particular el tiempo de procesamiento de las diferentes estructuras. Las estructuras que compararemos serán la implementación escalar convencional, implementación IFIR (en el caso de filtros FIR), implementación IIIR (en el caso de filtros con respuesta impulsiva infinita), implementación en bloques e implementación mixta. Con objeto de realzar los resultados temporales, hemos supuesto que las implementaciones de las estructuras se efectúan en procesadores de señal de alta velocidad, donde las operaciones aritméticas se realizan en un ciclo de instrucción de 20 ns.

5.2.1 Ejemplo Monodimensional

En este primer ejemplo abordaremos el diseño e implementación de un filtro digital para sintonizar la banda de 62.5 Khz, empleando una frecuencia de muestreo de 500 Khz. El canal de transmisión utilizado es la línea de distribución de electricidad de alta tensión, y la señal de información es una señal QAM de 32 niveles, donde cada símbolo está formado por 256 muestras, proporcionando un canal de comunicación de 9765 bps. El filtro se diseña como filtro paso de banda, con frecuencia central en 62.5 Khz, ancho de banda 9.6

Khz, rizado en la banda de paso menor que 0.1, y atenuación en la banda de rechazo mayor que 0.001, siendo la frecuencia de muestreo de 500 Khz.

Suponiendo que se emplea un procesador de señal cuyo ciclo de instrucción es de 20 ns, el número de instrucciones (de un ciclo de duración) que es posible efectuar entre dos muestras consecutivas de señal es de 100.

El primer diseño que efectuamos es como filtro FIR lineal. El orden mínimo encontrado para cumplir las especificaciones de diseño es N_{FIR}=223. En la figura 5.1.a se muestra la respuesta impulsiva del filtro escalar diseñado, y de forma más detallada en las figuras 5.1.b, 5.1.c y 5.1.d.

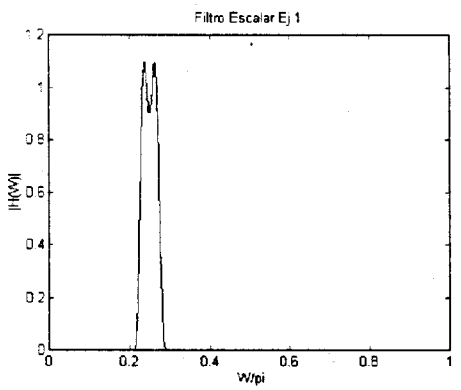

Figura 5.1.a

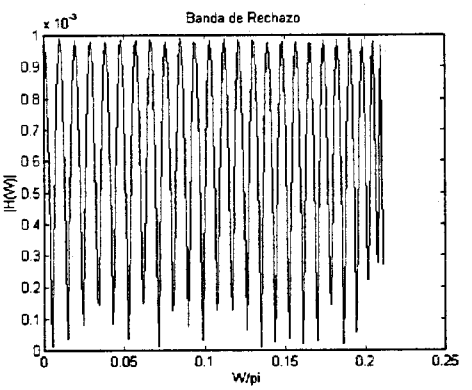

Figura 5.1.b

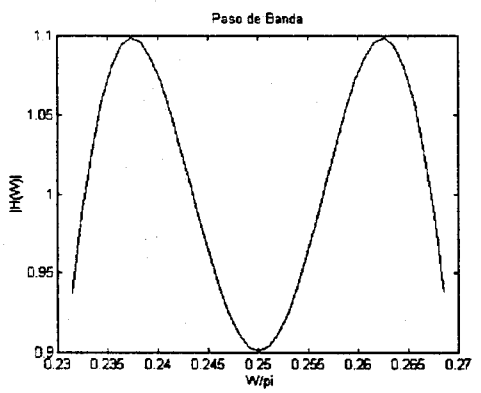

Figura 5.1.c

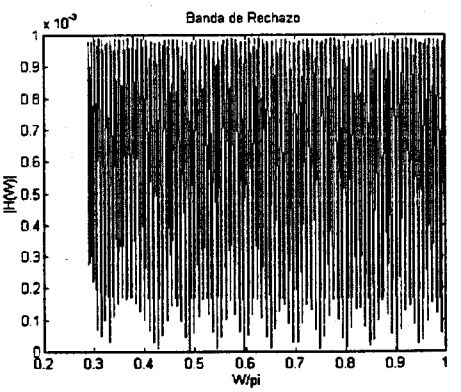

Figura 5.1.d

Para aprovechar el set de instrucciones del procesador, donde existen operaciones de multiplicación, acumulación y desplazamientos de memoria (MACD) en un solo ciclo de instrucción, efectuamos la implementación con estructura directa, sin aprovechar la característica de fase lineal, por lo que el número de operaciones aritméticas es NOP = 2 x O_H + 1 = 447. Para utilizar esta implementación sería necesario el uso de un buffer de entrada para no perder las muestras, ya que se reciben 4 muestras por cada una procesada.

Con objeto de procesar la señal de información en tiempo real, procedemos a efectuar una implementación en bloques de longitud L = 4, utilizando una estructura iterada 2 x 2, que requiere 9 procesadores y 15 adiciones entre el pre- y post-procesado. La figura 5.2 muestra el error, en el dominio temporal, entre la implementación escalar y la de bloques.

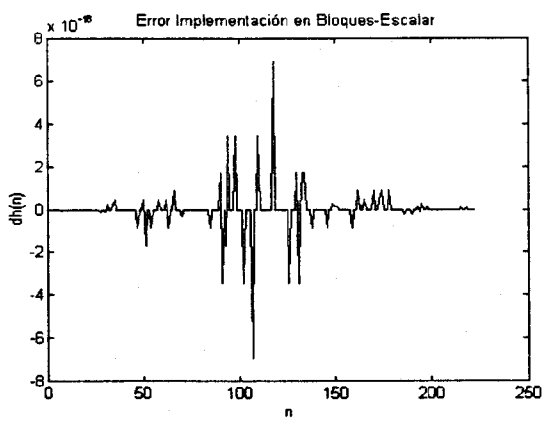

Figura 5.2

El tiempo de procesamiento por muestra de la implementación en bloques es de $TPM_{Bloques} = 32$, que permite procesar las muestras en tiempo real, aumentando el costo hardware de la estructura (9 procesadores).

Como mencionamos en el capítulo 4, es posible obtener implementaciones escalares con reducido número de coeficientes utilizando métodos alternativos de diseño, como por ejemplo la implementación IFIR. El filtro que deseamos diseñar es paso de banda (BP), centrado en la frecuencia normalizada $\omega_c = 0.25$. Si el filtro de forma interpolado se genera a partir de un filtro LP o HP, el ancho de banda del filtro limita el orden de interpolación a L<52, , mientras que si se genera a partir de un filtro de forma BP el límite es L< 26. De forma cualitativa sabemos que las implementaciones óptimas, con menor número de coeficientes, se consiguen diseñando el filtro de forma como paso de bajas o paso de altas. Escogiendo esta opción, el orden de interpolación óptimo de interpolación (4.4.18) es L= 7.88 . Teniendo en cuenta este resultado vemos que L = 8 es un valor próximo al valor óptimo y que al diseñar el filtro de forma como filtro LP generará una imagen en la zona de interés.

Exigiendo que el rizado en la banda de paso de los filtros de forma e interpolador valgan $\delta_{p,G} = \delta_{p,I} = 0.05$ y $\delta_{p,G} = \delta_{s,I} = 0.001$, el orden mínimo encontrado para el filtro de forma, que cumple las especificaciones es $N_G = 29$. En las figuras 5.3.a y 5.3.b se muestran las respuestas impulsivas de los filtros de forma, G(z), y filtro de forma interpolado, $G(z^8)$.

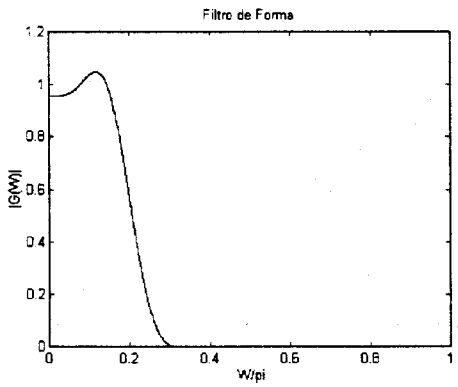

Figura 5.3.a

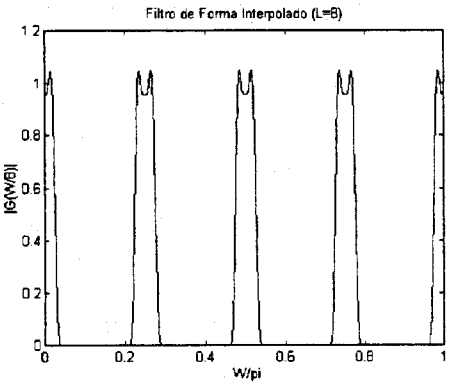

Figura 5.3.b

Como se deduce de la figura 5.3.b, el filtro interpolador se diseña como filtro paso de banda. El orden mínimo encontrado que hace cumplir las especificaciones de diseño al filtro IFIR es N_I = 20. La repuesta impulsiva del filtro interpolador la mostramos en la figura 5.4

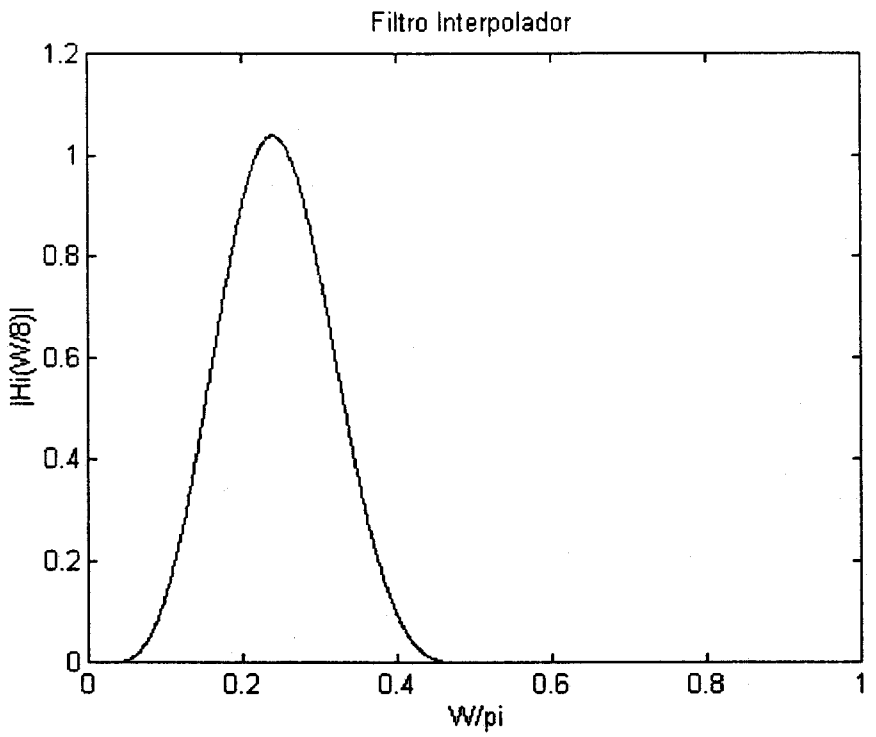

Figura 5.4

El filtro IFIR diseñado cumple las especificaciones de diseño, tal como se muestra en las figuras 5.5.a, 5.5.b, 5.5.c y 5.5.d, presentando un tiempo de procesamiento por muestra de $TPM_{IFIR} = 59 + 41 = 100$.

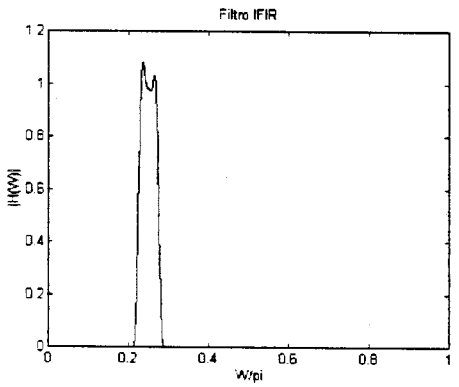

Figura 5.5.a

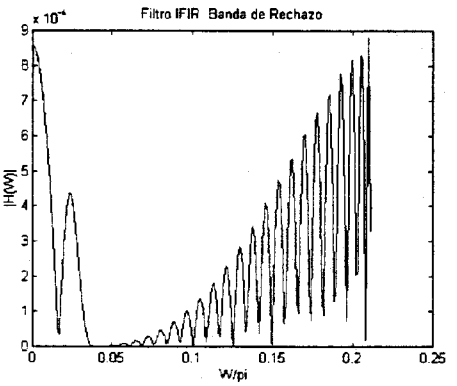

Figura 5.5.b

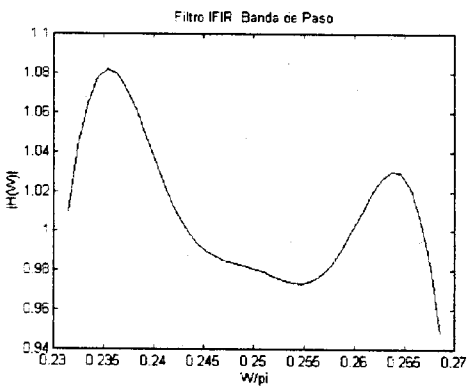

Figura 5.5.c

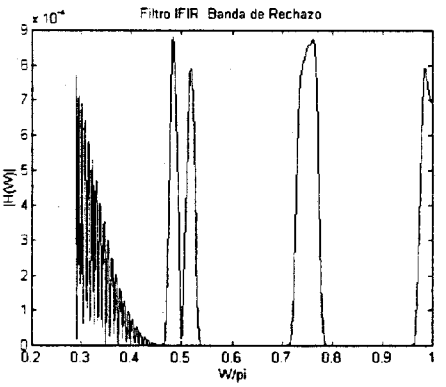

Figura 5.5.d

Con la estructura diseñada sería posible procesar las muestras en tiempo real sin necesidad de emplear un buffer de entrada. Si embargo, el procesador gastaría todo el tiempo de procesamiento que dispone en el filtrado de una muestra. Análogamente al caso anterior, recurrimos al procesamiento en paralelo para aumentar la velocidad de procesamiento del sistema. La alternativa a la implementación en bloques de la estructura IFIR es la

implementación mixta. Como hemos mencionado, el diseño de los filtros se realiza como filtros FIR de fase lineal, aunque en la implementación no se aprovecha la simetría de los coeficientes para optimizar el uso del set de instrucciones del procesador. Por este motivo, al calcular el orden de interpolación óptimo con estructura mixta lo efectuamos como si fuese un filtro de fase no lineal, sin simetría en los coeficientes. El orden óptimo encontrado es L = 4.2436 (4.4.32 - 4.4.35). Teniendo en cuenta este valor, escogiendo L_{mixta}= 4 y diseñando el filtro de forma como filtro paso de altas, se generará una imagen centrada en la zona de interés. El orden mínimo encontrado para el filtro de forma cumpliendo unas especificaciones de rizado $\delta_{p,G} = \delta_{p,I} = 0.05$ y $\delta_{p,G} = \delta_{s,I} = 0.001$, es N_G = 60. En las figuras 5.6.a y 5.6.b se muestran los comportamientos frecuenciales de los filtros de forma, G(z), y filtro de forma interpolado, $G(z^4)$.

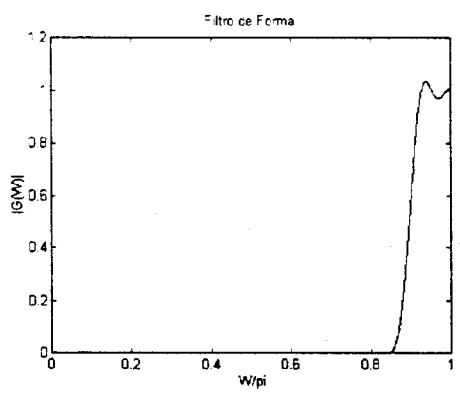

Figura 5.6.a

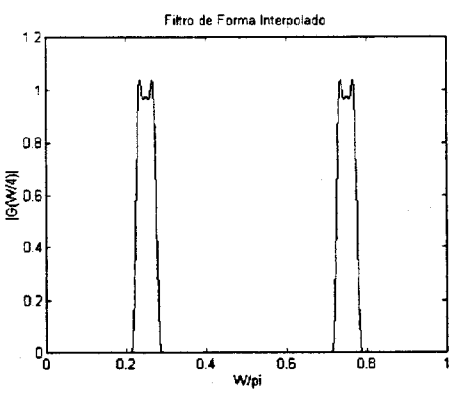

Figura 5.6.b

La figura 5.6.b sugiere que el filtro interpolador se diseñe como filtro paso de bajas. El orden mínimo encontrado que hace cumplir las

especificaciones de diseño del filtro IFIR es $N_I = 9$. El comportamiento frecuencial del filtro interpolador se muestra en la figura 5.7, mientras que en las figuras 5.8.a, 5.8.b, 5.8.c y 5.8.d se muestra el comportamiento del filtro IFIR resultante.

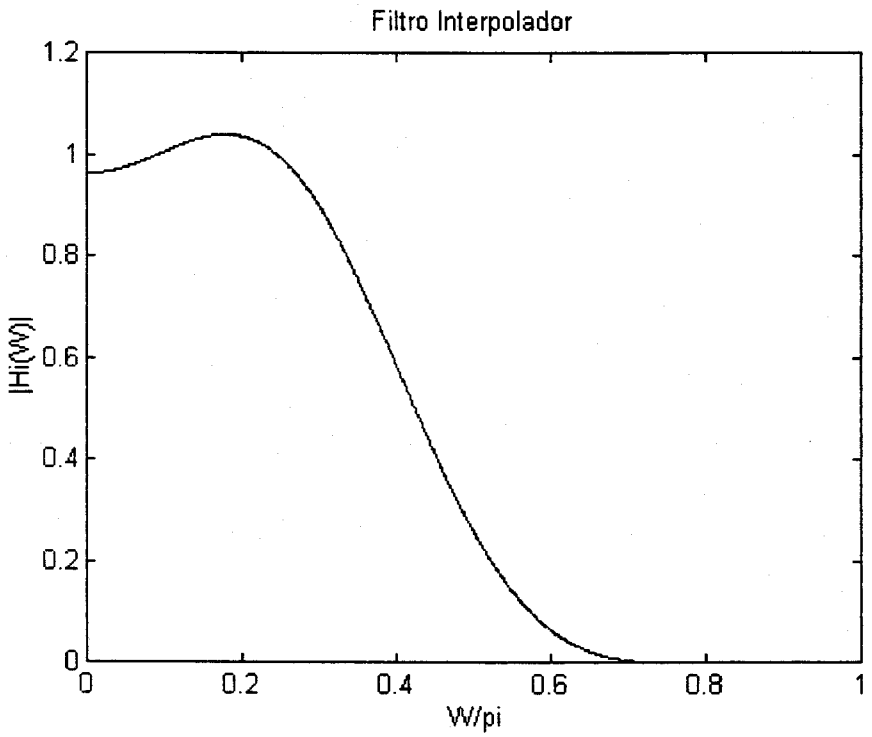

Figura 5.7

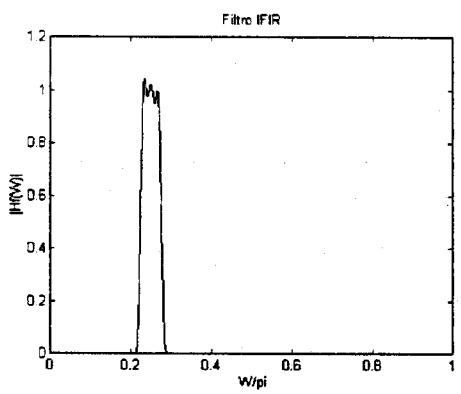

Figura 5.8.a

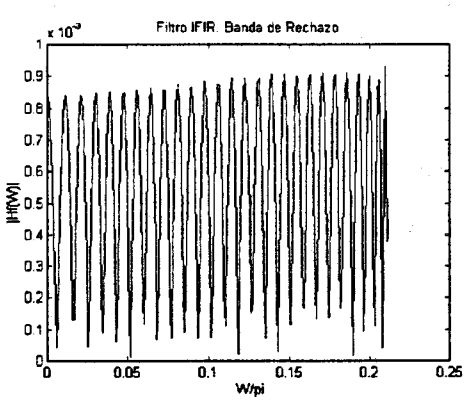

Figura 5.8.b

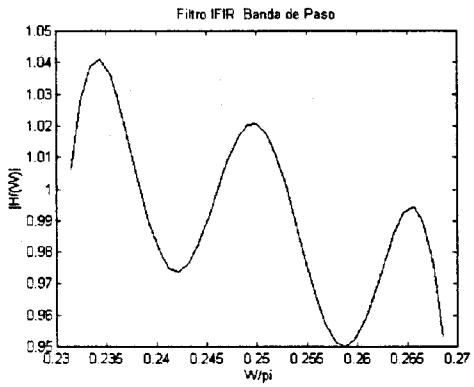

Figura 5.8.c

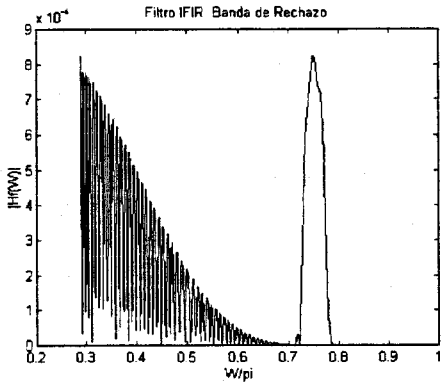

Figura 5.8.d

El tiempo de procesamiento por muestra del filtro de forma interpolado es $TPM_G = 121$, mientras que el del filtro interpolador es $TMP_I = 19$. Aunque la estructura diseñada no es óptima respecto al número de operaciones aritméticas efectuadas con implementación escalar , al efectuar la implementación mixta del filtro IFIR el tiempo de procesamiento por muestra del sistema lo determina el filtro de forma interpolado implementado en bloques de longitud 4, con un $TPM_{Mixta} = 30.25$, siendo el costo hardware de la estructura de 5 procesadores.

Como es sabido, la implementación de filtros con respuesta impulsiva finita ofrece una serie de ventajas sobre los filtros con respuesta impulsiva infinita. Entre otras podemos nombrar la estabilidad de la estructura, de particular importancia a la hora de implementar el filtro en procesadores de punto fijo. Aunque hemos conseguido, con la estructura mixta, reducir considerablemente el tiempo de procesamiento de la implementación, si este incremento de velocidad fuese aún insuficiente, por requerimientos del

procesado posterior, no podríamos diseñar estructuras más veloces con estructura mixta. Si es posible, por contra, incrementar la velocidad de procesamiento aumentando la longitud del bloque, si usamos esta técnica. Por supuesto, el costo hardware de la estructura aumentaría considerablemente. En el caso de necesitar menores tiempos de procesamiento, manteniendo el costo hardware reducido, sería imprescindible el diseño del sistema como filtro IIR.

Un diseño IIR escalar convencional como filtro elíptico, cumpliendo las especificaciones de diseño, conduce a un filtro de orden $N_{IIR} = 10$, con un tiempo de procesamiento por muestra de $TPM_{IIR} = 40$. Las figuras 5.9.a, 5.9.b, 5.9.c y 5.9.d, muestran el comportamiento frecuencial del filtro elíptico

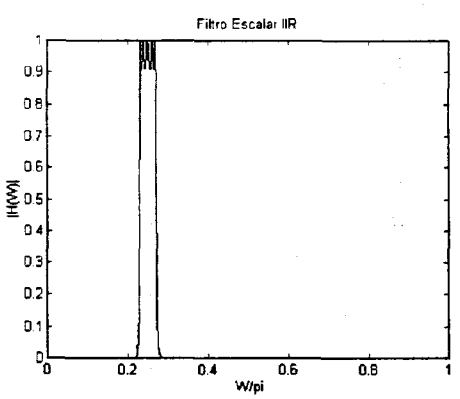

Figura 5.9.a

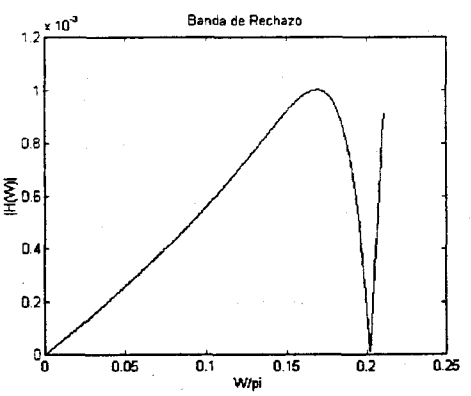

Figura 5.9.b

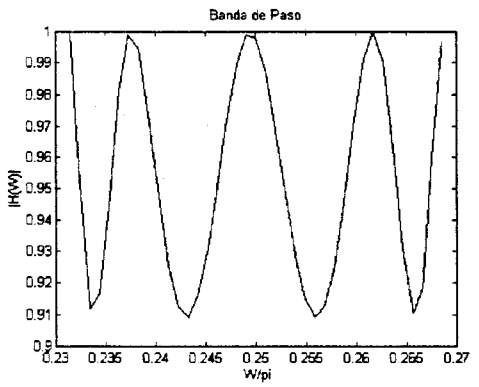

Figura 5.9.c

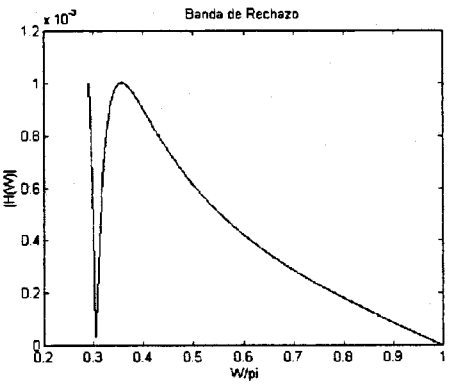

Figura 5.9.d

Para conseguir disminuir el tiempo de procesamiento del filtro IIR diseñado, efectuamos una implementación en bloques de longitud L = 2. En la figura 5.10 se muestra el error en la respuesta impulsiva cometido al implementar el filtro en bloques.

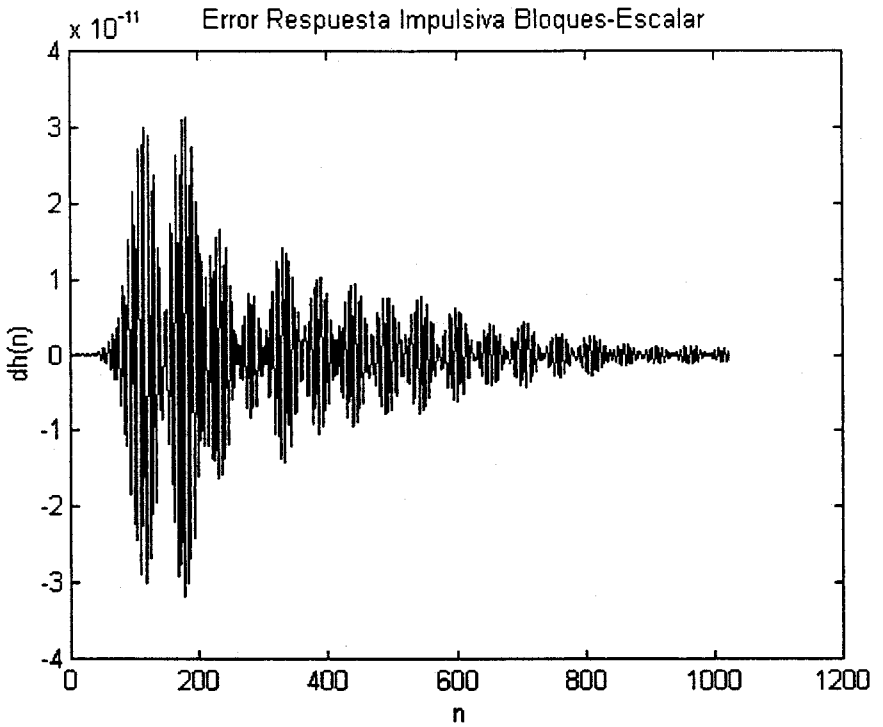

Figura 5.10

El tiempo de procesamiento por muestra de esta implementación se ha reducido a $TPM_{Bloques} = 22$, inferior a los tiempos de procesamientos obtenidos para el caso de filtros con respuesta impulsiva finita, mientras que el número de procesadores empleados se ha incrementado a 3.

Como en el diseño de filtros FIR, nos preguntamos si es posible obtener implementaciones escalares con reducido número de operaciones aritméticas. Vamos a diseñar el filtro como cascada de dos filtros IIR, a los que, por analogía con los filtros con respuesta impulsiva finita, denominaremos filtros IIIR.

Aprovechando que la frecuencia central del filtro es $\omega_c = 0.25$, diseñamos el filtro de forma como filtro paso de altas, que al interpolar con L = 4 generará una imagen centrada en la zona de interés. Exigiendo un rizado en la banda de paso de $\delta_{p,G} = 0.05$ y un rizado en la banda de rechazo de $\delta_{s,G} = 0.001$, encontramos que el orden mínimo del filtro de forma que cumple las especificaciones de diseño es $N_G = 5$. La figura 5.11.a muestra la respuesta impulsiva del filtro de forma, G(z), mientras que la figura 5.11.b muestra la respuesta del filtro de forma interpolado, $G(z^4)$.

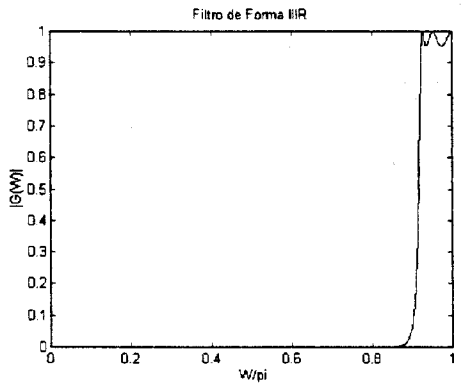

Figura 5.11.a

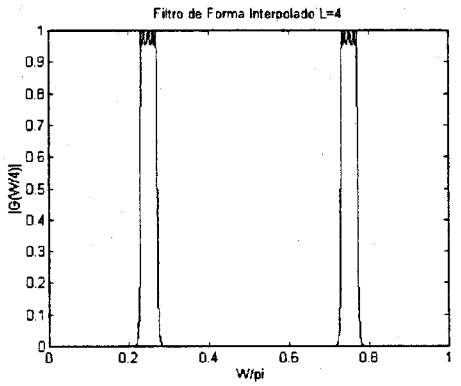

Figura 5.11.b

Para eliminar la imagen no deseada (Fig. 5.11.b) diseñamos el filtro interpolador como filtro paso de bajas, con las mismas especificaciones de rizado que el filtro de forma. El orden mínimo encontrado es $N_I = 4$, y la respuesta impulsiva viene representada en la figura 5.12.

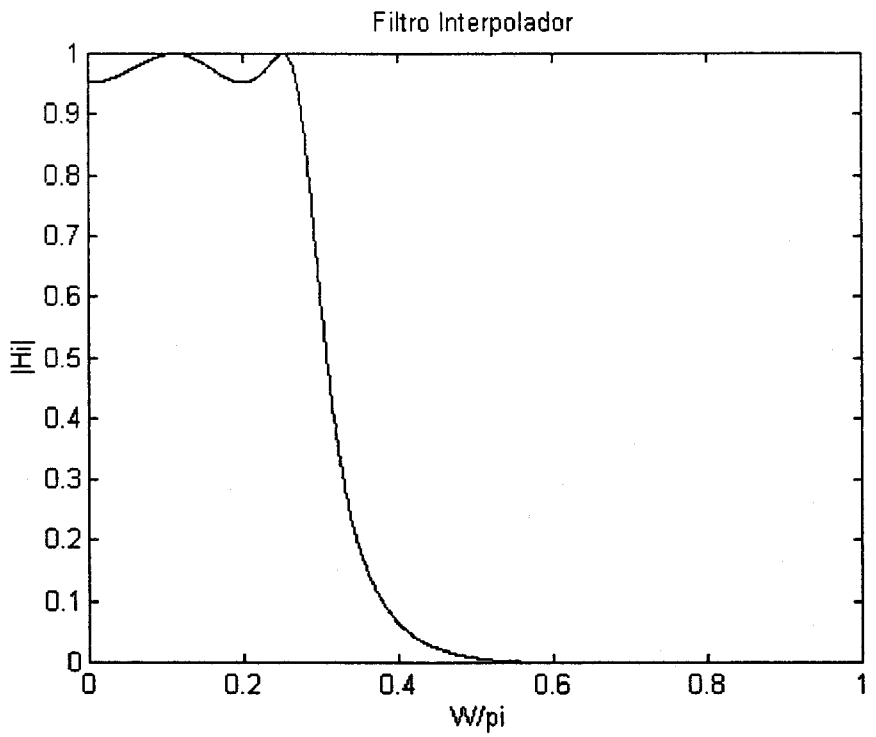

Figura 5.12

La cascada del filtro de forma interpolado y el filtro interpolador cumple las especificaciones de diseño, como se muestra en las figuras 5.13.a, 5.13.b, 5.13.c y 5.13.d. El tiempo de procesamiento de la estructura es $TPM_{IIR} = 33$, mejorando el obtenido en el caso convencional.

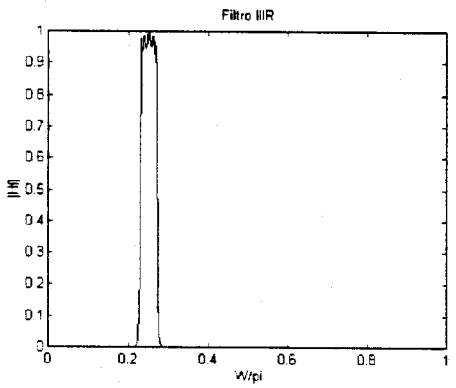

Figura 5.13.a

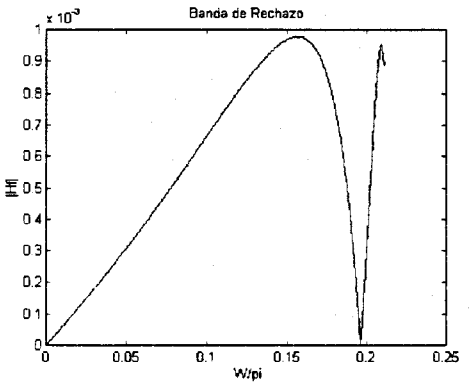

Figura 5.13.b

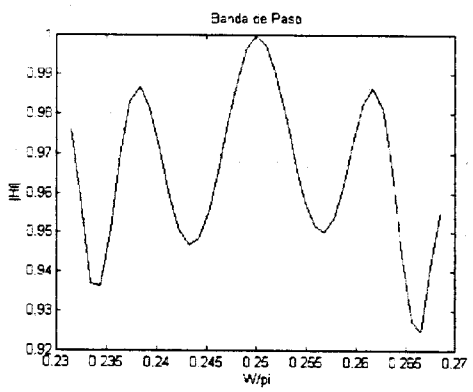

Figura 5.13.c

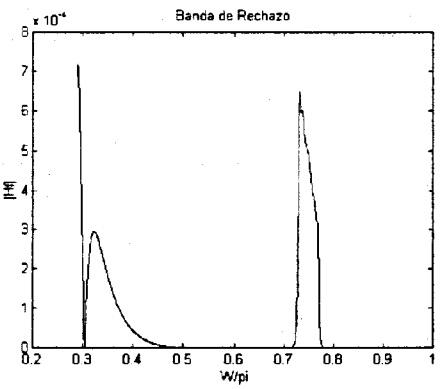

Figura 5.13.d

Aún es posible dism uir el tiempo de procesamiento del filtro IIR diseñado implementándolo con estructura mixta. El tiempo de procesamiento con estructura mixta lo determina el bloque más lento, es decir, el filtro interpolador, con $TPM_{Mixta} = 15$, requiriendo 3 procesadores.

En la siguiente tabla mostramos, de forma comparativa, los resultados obtenidos en este primer ejemplo.

	Nº Procesadores	TPM	η
FIR Escalar	1	447	0.0022
FIR Bloques	9	32	0.0035
IFIR	1	100	0.01
Mixta	5	30.25	0.0066
IIR Escalar	1	40	0.025
IIR Bloques	3	22	0.015
IIIR	1	33	0.03
Mixta	3	15	0.022

Tabla de Resultados Ejemplo 1º

5.2.2 Ejemplo Bidimensional

En este segundo ejemplo vamos a abordar el diseño e implementación de un filtro digital bidimensional que forma parte de un sistema de posicionamiento de aeronaves en un espacio aéreo de dimensiones 203 x 203 Km^2. El sistema está constituido por un emisor de pulsos periódicos y un conjunto de receptores que, midiendo el desfase temporal entre el pulso enviado y el recibido, envían a un procesador información sobre la distancia

que les separa de la aeronave. Los sensores forman un reticulado cuya celda unidad mide 1 Km^2, y donde los sensores se sitúan en los vértices de las celdas. La información transmitida por cada sensor es la inversa de la distancia que les separa de la aeronave (Fig. 5.14), y es enviada una vez por segundo.

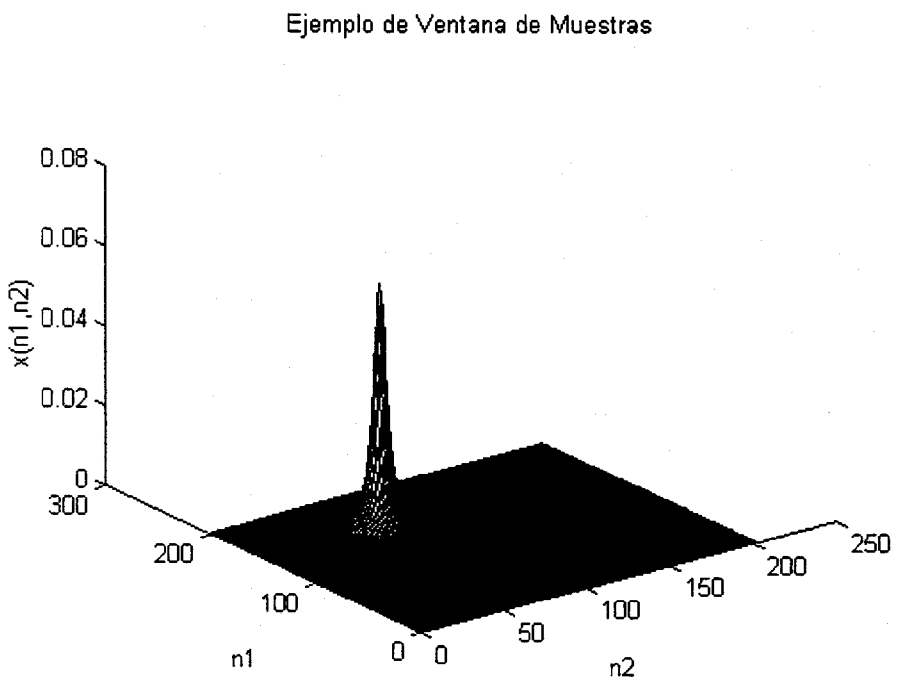

Figura 5.14

Un análisis espectral de la señal de información muestra que ésta es paso de bajas, y que un filtro LP bidimensional con radio de paso $R_p = 0.2\ \pi$ permite recuperar la información original. El radio de rechazo es $R_s = 0.25\ \pi$, mientras que el rizado en la banda de paso y en la banda de rechazo son iguales y de valor $\delta_p = \delta_s = 0.01$. El ruido en el canal es un ruido blanco aditivo Gaussiano de media cero y varianza 0.001 (Figura 5.15).

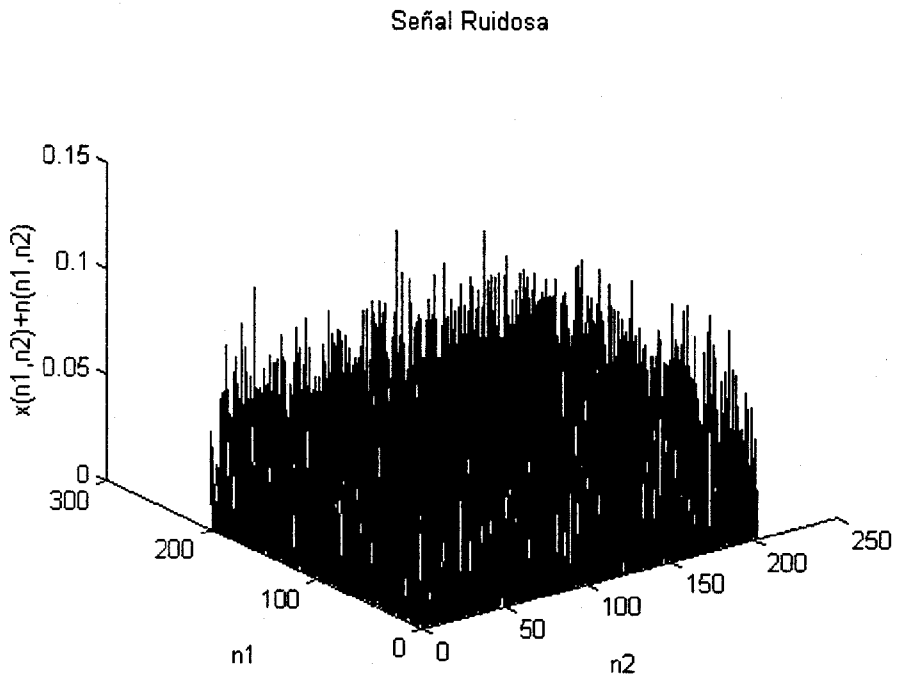

Figura 5.15

El diseño como filtro FIR bidimensional de fase cero, empleando la técnica por transformación descrita en el capítulo 4, conduce a una implementación de orden N_{FIR} = 41x 41, con un tiempo de procesamiento por muestra de TPM_{FIR} = 656. En las figuras 5.16.a, 5.16.b y 5.16.c se muestra la respuesta impulsiva del filtro diseñado, y en la figura 5.17 se aprecia el efecto del filtrado en la señal ruidosa recibida.

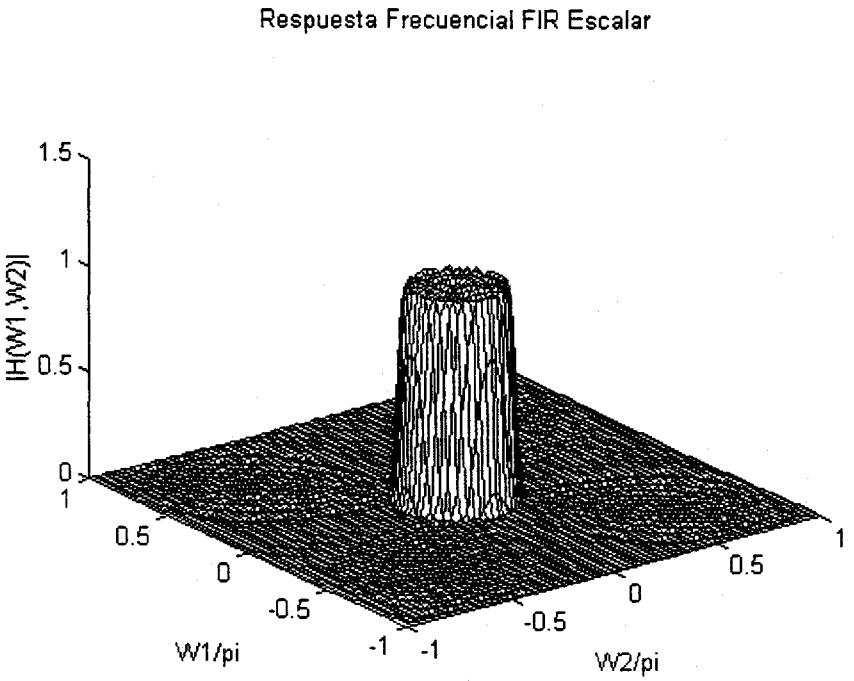

Figura 5.16.a

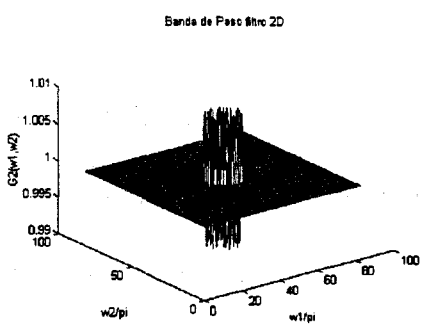

Figura 5.16.b

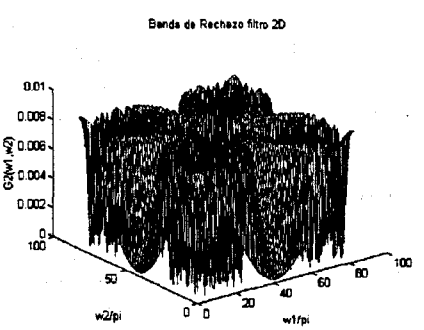

Figura 5.16.c

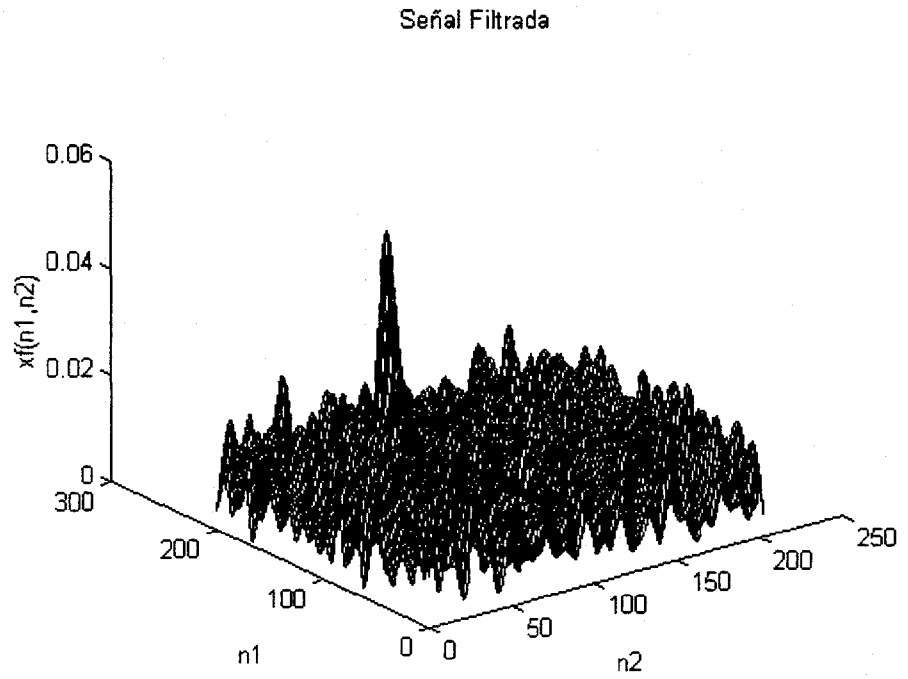

Figura 5.17

La ventana de procesamiento está formada por 41209 muestras (Fig.5.15), requiriendo el procesado un total de 27033104 operaciones aritméticas. Para trabajar con este elevado volumen de datos, implementamos el filtro digital con un procesador de alta velocidad, con ciclo de instrucción de 20 ns, y donde las operaciones aritméticas, adiciones y multiplicaciones, se efectúan en un solo ciclo de instrucción. Para filtrar la ventana de 203 x 203 muestras se requieren 0.54 s, que consume más de la mitad del tiempo de procesamiento disponible. Para aumentar la velocidad de procesamiento del filtro digital se efectúa una implementación en bloques.

Como se mencionó en el capítulo 4, los filtros de bloques no se obtienen por transformación. Por este motivo, puede darse el caso de que el tiempo de procesamiento de una determinada implementación tenga mayor tiempo de procesamiento que la implementación escalar convencional. Esto ocurre al implementar el filtro LP diseñado en bloques bidimensionales de longitud 2x2, con un TPM= 840.25, y con un costo hardware de 9 procesadores. Para reducir el tiempo de procesamiento empleamos una implementación en bloques gobernada por la matriz de submuestreo

$$M = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \tag{5.2.1}$$

Al implementar el filtro digital en bloques, aplicando dos algoritmos de convolución lineal Winograd de forma iterada 3x3 y 2x2, la estructura resultante está formada por 15 procesadores y efectúa 55 adiciones entre el pre- y post-procesado. En la figura 5.18 se muestra el error, en el dominio frecuencial, entre la implementación escalar y la de bloques. El tiempo de procesamiento por muestra de la estructura se ha reducido a $TPM_{Bloques} = 378$, por lo que el tiempo consumido en filtrar la ventana de muestras es de 0.31 s.

Aunque se ha conseguido disminuir el tiempo de filtrado, el costo hardware ha aumentado considerablemente. Con objeto de conseguir una estructura con menor tiempo de procesamiento que la obtenida con un diseño

convencional, diseñamos un nuevo filtro digital empleando la teoría de filtros IFIR.

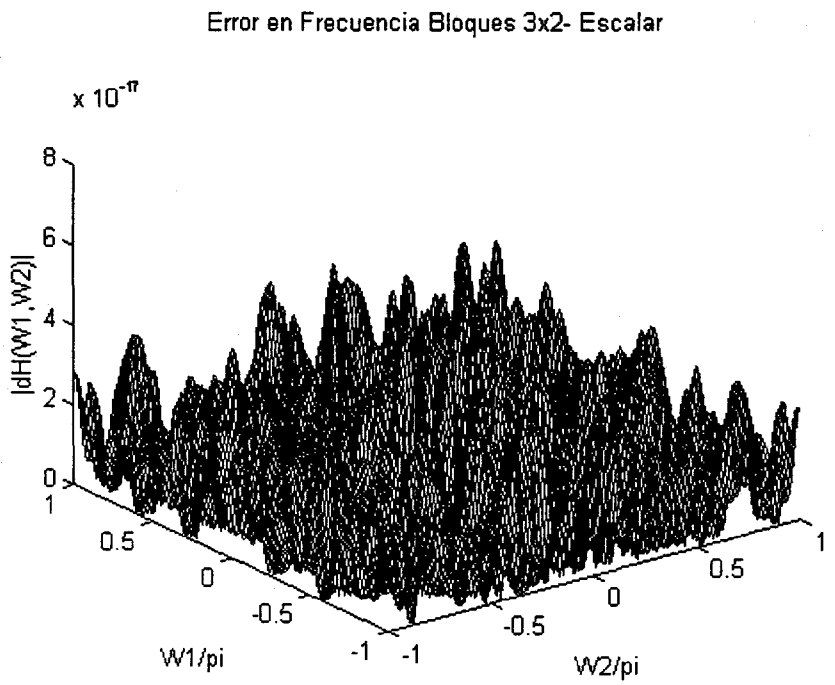

Figura 5.18

La limitación impuesta por el ancho de banda del filtro permite emplear la técnica de diseño IFIR para órdenes de interpolación L= 2, 3. Haciendo que el rizado en la banda de paso de los filtros de forma e interpolador sean $\delta_{p,G} = \delta_{p,I} = 0.005$ y $\delta_{s,G} = \delta_{s,I} = 0.01$ respectivamente, el orden de interpolación óptimo encontrado es L = 2.61, que está dentro del límite permitido. Efectuando la implementación para ambos órdenes de interpolación llegamos a la conclusión de que la estructura óptima se obtiene con un orden de interpolación de L = 2. Con estas especificaciones encontramos que el orden

mínimo del filtro de forma es N_G = 23 x 23. En las figuras 5.19.a y 5.19.b se muestra la respuesta frecuencial del filtro de forma y del filtro de forma interpolado.

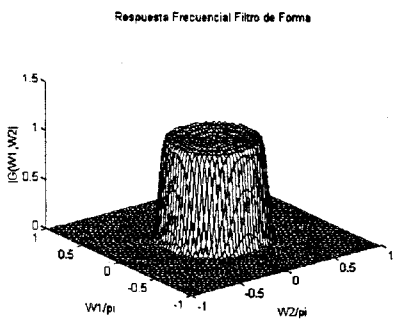

Figura 5.19.a

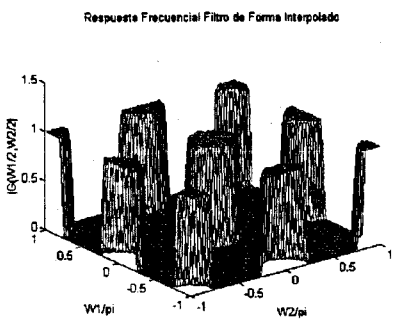

Figura 5.19.b

El filtro interpolador lo diseñamos como filtro paso de bajas. El orden mínimo encontrado es N_I = 5 x 5 (Fig. 5.20). La cascada del filtro de forma interpolado, $G(z^2)$, y el filtro interpolador, $H_I(z)$, cumple las especificaciones de diseño, tal como se muestra en las figuras 5.21.a, 5.21.b y 5.21.c. El tiempo de procesamiento por muestra de la estructura es TPM_{IFIR} = 448, invirtiendo 0.37 s en filtrar la ventana de muestras.

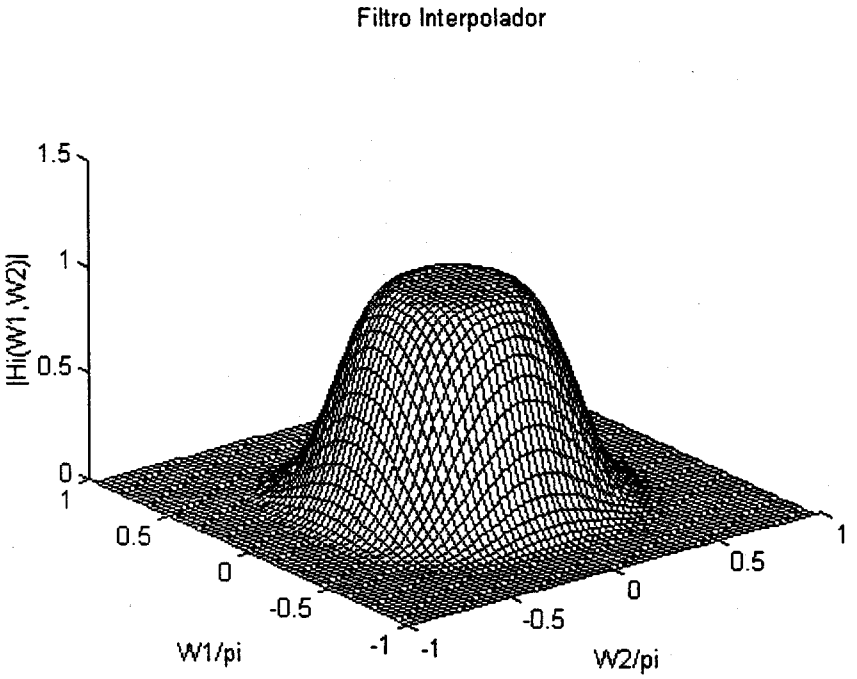

Figura 5.20

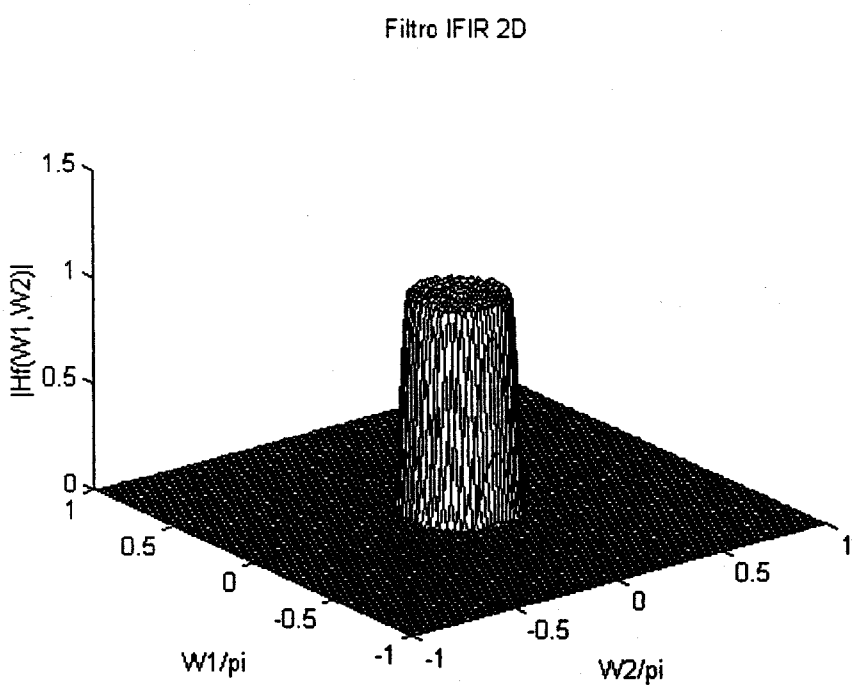

Figura 5.21.a

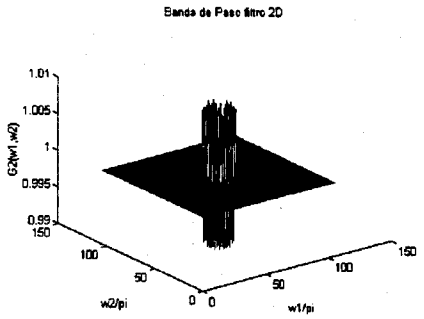

Figura 5.21.b

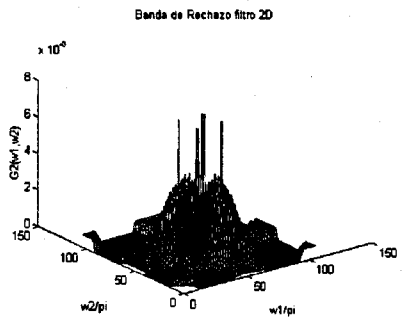

Figura 5.21.c

Diseñando el filtro como filtro IFIR hemos conseguido un sistema escalar de igual velocidad de procesamiento que la implementación en bloques del filtro FIR convencional, empleando un solo procesador.

Para aumentar la velocidad de procesamiento vamos a efectuar una implementación mixta. En primer lugar calculamos el orden de interpolación óptimo, de valor $L_{Mixta} = 2$, que coincide con el orden de interpolación óptimo del filtro IFIR descrito anteriormente. El sistema que determina el tiempo de procesamiento de la implementación mixta es el filtro de forma interpolado implementado en bloques de longitud 2 x 2, con un $TPM_{Mixta} = 92$. El costo hardware de la estructura es de 5 procesadores, requiriendo 0.066 s para filtrar la ventana de muestras.

En el caso de imponer restricciones más estrictas, como por ejemplo incrementar la frecuencia de adquisición de las ventanas de muestras o aumentar el área de detección, las implementaciones anteriormente descritas pueden no cumplir las especificaciones exigidas. Supongamos que se amplía la

ventana de detección a un cuadrado de área 407 Km^2. En estas condiciones el filtro FIR escalar descrito anteriormente emplearía 2.17 s en filtrar la ventana, mientras que la implementación en bloques de longitud 3 x 2 gastaría 1.25 s. Análogamente, la implementación IFIR resulta insuficiente para esta aplicación, con 1.48 s. Naturalmente, aumentando la longitud de la implementación en bloques es posible alcanzar la velocidad de procesamiento requerida. Sin embargo, el número de procesadores involucrados aumenta considerablemente. En cambio, la implementación mixta consigue procesar la nueva ventana de muestras en tiempo real, invirtiendo 0.3 s en el filtrado.

Aunque con la implementación mixta se logra procesar la ventana de muestras, es conveniente disminuir aún más el tiempo de procesamiento. Por este motivo pasamos a implementar el filtro digital como filtro con respuesta impulsiva infinita.

Debido a que los filtro diseñados no son de fase cero, se introduce un desplazamiento en la señal filtrada que da lugar a un error en la localización espacial de la aeronave. Sin embargo, este error coincide con el retraso de grupo del filtro, que al conocerlo posibilita la recuperación de la información.

El primer diseño que efectuamos es como filtro IIR escalar elíptico. El orden mínimo encontrado para cumplir las especificaciones de diseño es

N_{IIR}=5x 5, con un tiempo de procesamiento por muestra de TPM_{IIR} = 97. En la figura 5.22 se representa la respuesta frecuencial del filtro diseñado.

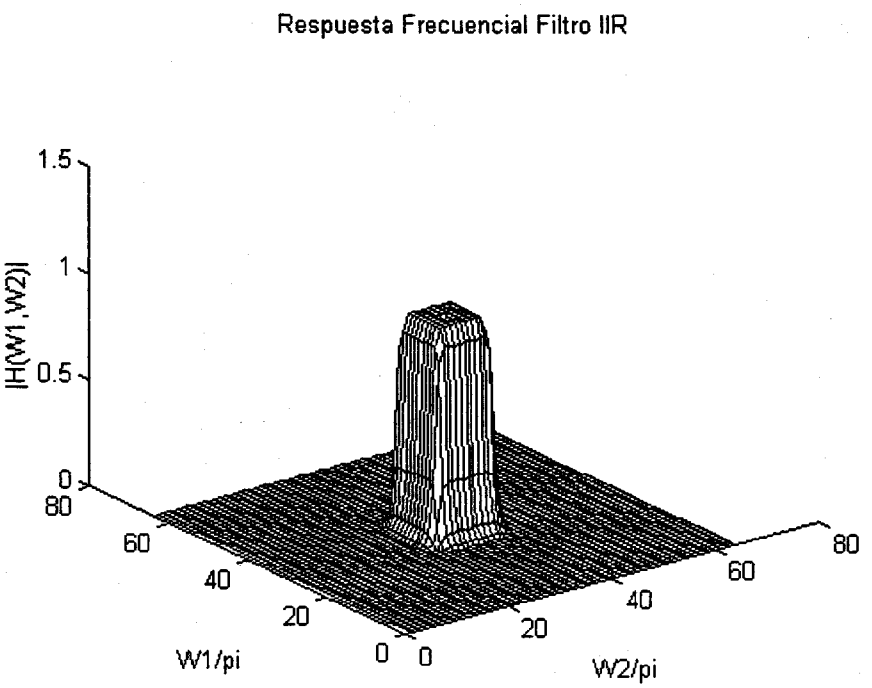

Figura 5.22

El filtro diseñado emplea 0.321 s en filtrar la ventana de muestras, igual que la implementación mixta mencionada anteriormente. La implementación en bloques de longitud 2 x 2, con $TPM_{Bloques}$ = 28, disminuye el tiempo de procesamiento de la ventana a 0.09 s, con un incremento en el costo hardware de 9 procesadores.

Para reducir el tiempo de procesamiento con menor complejidad hardware efectuamos el diseño del filtro digital como IIR bidimensional. El

rizado en la banda de paso de los filtros de forma e interpolador es $\delta_{p,G} = \delta_{p,I} = 0.005$, y el rizado en la banda de rechazo es $\delta_{s,G} = \delta_{s,I} = 0.01$. El orden mínimo encontrado para el filtro de forma es N_G=4x4. En las figuras 5.23.a y 5.23.b se muestra la respuesta frecuencial del filtro de forma y del filtro de forma interpolado respectivamente.

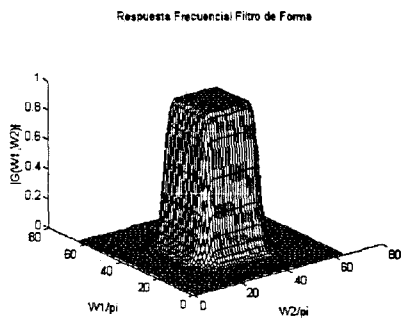

Figura 5.23.a

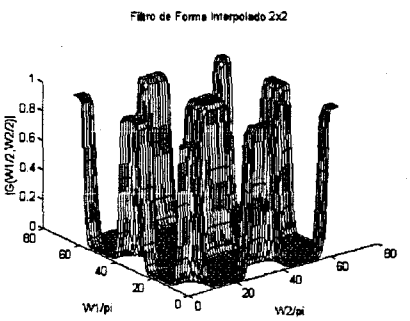

Figura 5.23.b

El filtro interpolador se diseña como filtro paso de bajas. El orden mínimo encontrado es N_I = 3 x 3 (Fig. 5.24), resultando una estructura con un tiempo de procesamiento por muestra de TPM_{IIIR} = 112.

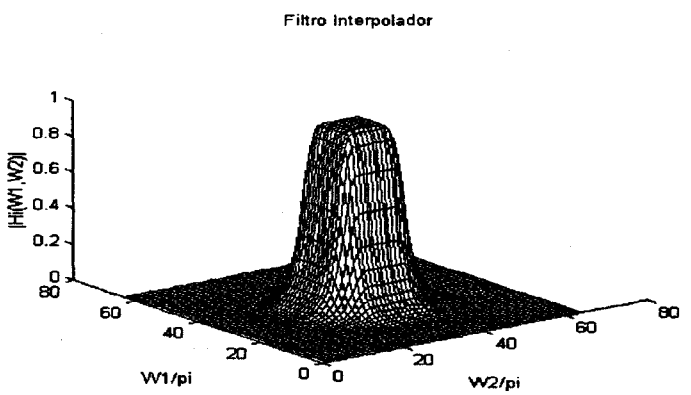

Figura 5.24

La ventaja de esta implementación se obtiene al efectuar la implementación mixta del filtro IIIR. Implementándolo de esta forma el tiempo de procesamiento por muestra se reduce hasta TPM_{Mixta} = 43, empleando 0.14 s en procesar la ventana de muestras. El costo hardware de la implementación es de 3 procesadores. La respuesta impulsiva del filtro IIIR se muestra en las figuras 5.25.a, 5.25.b y 5.25.c .

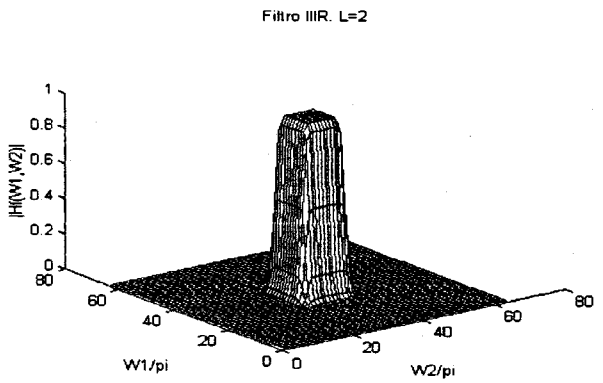

Figura 5.25.a

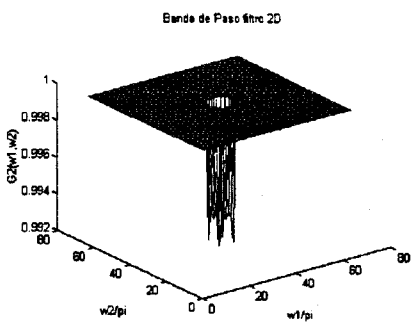

Figura 5.25.b

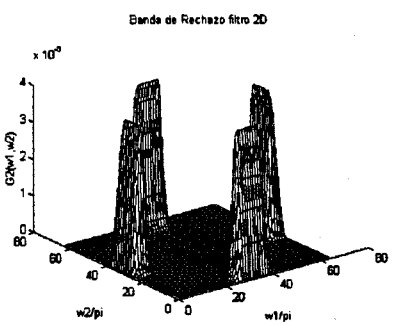

Figura 5.25.c

Finalmente, en la siguiente tabla se muestra de forma comparativa los resultados obtenidos para las distintas implementaciones del filtro IIIR bidimensional del ejemplo desarrollado.

	Nº Procesadores	TPM	η
FIR Escalar	1	656	0.0015
FIR Bloques	15	378	0.0002
IFIR	1	448	0.0022
Mixta	5	92	0.0022
IIR Escalar	1	97	0.0103
IIR Bloques	9	28	0.004
IIIR	1	112	0.0089
Mixta	5	43	0.0046

Tabla de Resultados Ejemplo 2º

5.3 APLICACIONES

El objetivo buscado al aplicar técnicas de procesamiento en paralelo, como las estudiadas en esta tesis, es desarrollar métodos de implementación y diseño de filtros digitales que permitan el procesado en tiempo real de señales, a la vez que se mantiene la complejidad hardware razonablemente baja. La

necesidad de utilizar más de un procesador viene impuesta por la tecnología disponible, así por ejemplo aplicaciones que hace una década requerían un entorno multiprocesador, con tiempos de ciclo de instrucción de 150 ns [Hayashi et al, 1986] , en la actualidad podrían ser implementados con un solo procesador de 20 ns por ciclo de instrucción.

Hoy en día existen gran diversidad de campos de aplicación para las estructuras computacionales estudiadas. La necesidad de su utilización se deriva, principalmente, del requerimiento de ejecución en tiempo real que, debido a la frecuencia de trabajo o debido al gran volumen de datos, no es posible alcanzar con implementaciones escalares convencionales.

Una aplicación la encontramos por ejemplo en radar, para el diseño e implementación de filtros transversales para la detección de objetivos móviles en presencia de ecos generados por objetos fijos. Filtros sintonizados para comunicación de señales vía satélite, o para transmisión de voz /datos a través de líneas de distribución de electricidad son otros posibles campos de aplicación.

Las estructuras descritas en el capítulo 4 pueden formar parte de otras estructuras, como ocurre al implementar en bloques, o con implementación mixta, los filtros de análisis y síntesis de una estructura multitasa para codificación/decodificación de señales de audio, voz o vídeo. También es

posible utilizar estas técnicas en aplicaciones no lineales, como en el filtrado de señales homomórficas.

En el campo del procesamiento de imágenes la utilización de las técnicas de diseños de filtros estudiadas, así como las estructuras de procesamiento en paralelo, permiten procesar el gran volumen de datos que caracteriza este tipo de aplicaciones. Dentro de este grupo de aplicaciones estarían, entre otras, el realce de imágenes, extracción de contornos o eliminación de ruido.

Finalmente, una generalización de las técnicas estudiadas al caso multidimensional permitiría la aplicación de las estructuras a campos tales como filtrado de ondas portadoras (*Beamforming*), tronco común de aplicaciones como radar, sonar o procesamiento de señales geofísicas.

5.4 CONCLUSIONES

En el desarrollo de esta tesis se ha profundizado en tres ideas fundamentales. Por un lado, se ha proporcionado una nueva aproximación a la implementación en bloques de filtros digitales lineales e invariantes en el tiempo (LTI). Se han generalizado al caso bidimensional las estructuras óptimas, basadas en algoritmos rápidos de convolución, obtenidas por otros autores [Acha 89] para la implementación en bloques de filtros digitales

monodimensionales. Por último, utilizando la teoría de filtros IFIR monodimensionales, se propone una nueva línea para el diseño e implementación de filtros digitales que permite obtener estructuras con tiempos de procesamientos equivalentes a las obtenidas con la implementación en bloques, pero con un costo hardware significativamente menor.

5.4.1 Nueva Aproximación a la Teoría de la Implementación en Bloques

En la literatura existente, la derivación de la matriz de bloques siempre se ha efectuado a partir del concepto de invariancia al desplazamiento por bloques. En esta tesis se ofrece otra perspectiva al problema.

Se deriva la expresión de la matriz de bloques de un sistema LTI bidimensional a partir del concepto de descomposición polifásica del mismo.

El tratamiento realizado permite la implementación en bloques de filtros bidimensionales con matriz de submuestreo arbitraria, no limitándose al caso de implementaciones en bloques rectangulares.

Se deduce una expresión que relaciona las componentes polifásicas, y por lo tanto las matrices de bloques, de descomposiciones polifásicas equivalentes. De igual modo, se proporciona un algoritmo de cálculo para las componentes polifásicas de filtros con respuesta impulsiva infinita,

independientemente de cual sea la matriz de submuestreo elegida y si es o no de variables separables. Además, el algoritmo propuesto es fácilmente extrapolable al caso multidimensional.

5.4.2 Estructuras Computacionales para Implementación en Bloques 2D

Obtenida la matriz de bloques de un filtro bidimensional, implementar físicamente el sistema supondría un elevado costo hardware. Aprovechando los trabajos efectuados por [Acha 89] para el caso monodimensional, se deducen una serie de estructuras computacionales para la implementación de la matriz de bloques bidimensional. Estas estructuras se obtienen al aplicar de forma iterada algoritmos de convolución lineal monodimensionales. Las estructuras obtenidas no son óptimas, respecto al número de procesadores empleados, pero reducen su número drásticamente.

Se demuestra, además, la existencia de un orden de aplicación de la iteración que optimiza el número de adiciones que se efectúan en la pre- y post-adición.

5.4.3 Diseño e Implementación con Estructura Mixta

Una condición indispensable para obtener estructuras con alta velocidad de procesamiento es diseñar filtros con reducido número de coeficientes.

Existen diversas técnicas que logran este objetivo. Dentro de éstas, el diseño de filtros FIR mediante interpolación (IFIR) ofrece una importante característica.

Partiendo de la estructura escalar de los filtros FIR interpolados, se deriva una nueva estructura, a la que se denomina implementación mixta, que permite obtener estructuras computacionales con tiempos de procesamiento similares a las implementaciones en bloques equivalentes, pero con un costo en el número de procesadores significativamente menor.

Se derivan expresiones para calcular el orden de interpolación que minimiza el tiempo de procesamiento de la estructura escalar (IFIR) y de la implementación mixta, tanto del caso monodimensional como bidimensional.

La existencia de un valor óptimo del orden de interpolación implica que cualquier otra implementación, escalar o mixta, con distinto orden de interpolación tendrá menor velocidad de procesamiento. Esta propiedad establece un límite de validez para la estructura mixta como alternativa a la de bloques. Además de la limitación anteriormente mencionada, el ancho de banda del filtro limita el conjunto de valores posibles para el orden de interpolación .

A pesar de las limitaciones existentes, la implementación mixta presenta un alto grado de aplicación. En particular, en el caso de implementación de filtros FIR bidimensionales diseñados por transformación, las

implementaciones en bloques de pequeña longitud presentan mayores tiempos de procesamiento que la implementación escalar, convirtiendo a la implementación mixta en la única posibilidad para disminuir el tiempo de procesamiento del sistema, siempre que la implementación mixta óptima cumpla las especificaciones temporales exigidas en la aplicación.

Se generaliza el diseño de filtros generados por interpolación al caso de filtros con respuesta impulsiva infinita, a los que se denominan IIIR por analogía con los filtros IFIR.

En el caso monodimensional se observa que la implementación óptima con estructura mixta presenta menor tiempo de procesamiento que la implementación en bloques equivalente. Sin embargo, el valor del orden de interpolación óptimo es bajo. Esta particularidad se traduce en un limitado rango de validez para la implementación mixta como alternativa a la de bloques.

En el caso bidimensional, la implementación en bloques presenta menor tiempo de procesamiento que la implementación mixta para cualquier orden de interpolación. La utilización de la estructura mixta, con menor velocidad de procesamiento que la de bloques, tendría aplicación únicamente para reducir el costo hardware.

5.5 LÍNEAS FUTURAS DE TRABAJO

Con la elaboración de esta tesis hemos querido aportar algunas soluciones al problema del diseño e implementación de filtros digitales, monodimensionales y bidimensionales, con reducido tiempo de procesamiento. Existen, no obstante, problemas no resueltos o generados en el transcurso de la misma. En las siguientes secciones mencionamos algunas de las líneas de investigación que ha suscitado este trabajo.

5.5.1 Estructuras Óptimas Bidimensionales

Las estructuras bidimensionales para la implementación en bloques de filtros digitales bidimensionales, obtenidas en el capítulo 4, fueron generadas aplicando de forma iterada dos algoritmos óptimos de convolución lineal monodimensional. Las estructuras iteradas requieren menor número de adiciones en el pre-procesado y post-procesado que la estructura óptima equivalente, con la desventaja de emplear mayor número de procesadores.

Con objeto de optimizar las estructuras obtenidas para la implementación en bloques de filtros digitales bidimensionales, es necesario el diseño de nuevos algoritmos óptimos de convolución lineal bidimensional que, al aplicarlos en la implementación en bloques, minimicen el número de procesadores.

5.5.2 Implementación Mixta con Matriz de Submuestreo Arbitraria

La implementación mixta propuesta en esta tesis se limita, en el caso bidimensional, al caso de implementaciones en bloques del filtro de forma con matriz de submuestreo rectangular. Con objeto de generalizar la aplicación de este tipo de implementación, es imprescindible ampliar el estudio al caso de matrices de submuestreo arbitrarias.

Entre otras ventajas destacamos la posibilidad de disminuir el orden del filtro de forma y del filtro interpolador interpolador, consiguiendo reducir el tiempo de procesamiento de la implementación escalar y mixta. El primer problema que nos hemos encontrado es la necesidad de diseñar el filtro de forma distorsionado, para compensar el efecto de la interpolación no rectangular.

5.5.3 Implementación Mixta de Filtros IIR

Los resultados conseguidos referentes al diseño e implementación con estructura mixta de filtros IIR bidimensionales no han sido concluyentes. El estudio efectuado solo contemplaba el diseño de filtros IIR bidimensionales obtenidos por la cascada de dos filtros monodimensionales elípticos. La generalización de este estudio engloba varios frentes. Algunas de estas líneas

de trabajo serían, estudiar las mismas estructuras con otros tipos de filtros monodimensionales, como por ejemplo Butterworth, Chebyshev. Obtener expresiones para los órdenes de interpolación óptimos con estructura escalar y mixta. Diseño de filtros IIR bidimensionales con variables no separables.

BIBLIOGRAFÍA

[Rey y Castro 67] J. Rey Pastor y A. de Castro Brzezicki, "Elementos de Matemáticas", Ed. SAETA, 1967.

[Gold y Jordan 68] B. Gold y K. L. Jordan, " A note on digital filter synthesis", Proc. IEEE (Lett), vol. 56, Oct. 1968.

[Oppenheim y Schafer 75] Alan V. Oppenheim y Ronald W. Schafer, "Digital Signal Processing", Ed. Prentice-Hall, 1975.

[Mitra y Gnanasekaran] Sanjit K. Mitra y R. Gnanasekaran, " Block Implementation of Recursive Digital Filters-New Structures and Properties", IEEE Trans. on Circuits and Systems, vol. CAS-25, Nº 4, Abril 1978.

[Barnes y Shinnaka 80] Casper W. Barnes y Shinji Shinnaka, " Block-Shift Invariance and Block Implementation of Discrete-Time Filters", IEEE Trans. on Circuit and Systems, vol. CAS-27, Nº 8, Agosto 1980.

[Neuvo et al, 84] Yrjö Neuvo y Dong Cheng-Yu y Sanjit K. Mitra, " Interpolated Finite Impulse Response Filters", IEEE Trans. on Acoustics, Speech, and Signal Procesing, vol. ASSP-32, Nº 3, Junio 1984.

[Dudgeon y Mersereau 84] Dan E. Dudgeon y Russell M. Mersereau, " Multidimensional Digital Signal Procesing", Ed. Prentice-Hall 1984.

[Blahut 85] Richard E. Blahut, "Fast Algorithms for Digital Signal Processing", Ed. Addison Wesley 1985.

[Hayashi et al, 86] Katsuhiko Hayashi, Kaushal K. Dhar, Kazunori Sugahara y Kotaro Hirano, " Design of High-Speed Digital Filters Suitable for Multi-DSP Implementation", IEEE Trans. on Circuits and Systems, vol. CAS-33, Nº 2, Febrero 1986.

[Neuvo et al, 87] Yrjö Neuvo, Ganesh Rajan y Sanjit K. Mitra, " Desing of Narrow-Band FIR Bandpass Digital Filters with Reduced Arithmetic Complexity", IEEE Trans. on Circuits and Systems, vol. CAS-34, Nº 4, Abril 1987.

[Chewn y Alexander 87] Chewn-Jye Ju y Winser E. Alexander, " Block Realization of Multidimensional IIR Digital Filters and Its Finite Word Effects", IEEE Trans. on Circuit and Systems, vol. CAS-34, Nº 9, Septiembre 1987.

[Saramäki et al, 88] Tapio Saramäki, Yrjö Neuvo y Sanjit K. Mitra, " Design of Computationally Efficient Interpolate FIR Filters", IEEE Trans. on Circuits and Systems, vol. 35, Nº 1, Enero 1988.

[Acha 89] J. I. Acha, "Computational Structures for Fast Implementation of L-Path and L-Block Digital Filters", IEEE Trans. on Circuits and Systems, vol. 36, Nº 6, Junio 1989.

[Vaidyanathan 90] P. P. Vaidyanathan, " Multirate Digital Filters, Filter Banks, Polyphase Networks, and Applications: A Tutorial", Proc. IEEE, vol. 78, Nº 1, Enero 1990.

[Vetterli y Karlsson 90] Martin Vetterli y Gunnar Karlsson, " Theory of Two-Dimensional Multirate Filter Banks", IEEE Trans. Acoustics, Speech, and Signal Processing, vol. 38, Nº 6, Junio 1990.

[Mou y Duhamel 91] Zhi-Jian Mou y Pierre Duhamel, " Short-Length FIR Filters and Their Use in Fast Nonrecursive Filtering", IEEE Trans. on Signal Processing, vol. 39, Nº 6, Junio 1991.

[Fliege 94] N. J. Fliege, " Multirate Digital Signal Processing", Ed. John Wiley & Sons, 1994.

[Soo-Chang y Jong-Jy 96] Soo-Chang Pei y Jong-Jy Shyu, " General Form for Designing Two-Dimensional Quadrantally Symmetric Linear-Phase FIR Digital Filters by Analytical Least-Squares Method", Signal Processing 48, Pg. 165-174, 1996.

CARLOS ROMERO PEREZ

UNA APORTACION AL DISEÑO DE FILTROS DE BLOQUES BIDIMENSIONALES

Apto (Cum Laude)

25 Abril 97.

J.A. MICHELL

F. Javier Payán Somet.

David Peñalta

ALFONSO GAGO BOUZA

Vicente Calvo Aguilera.

Carlos Romero Pérez