



INAOE

**Arquitectura Sistólica para el Filtrado
Espacial de Imágenes en Múltiples
Orientaciones**

por

Sayde Alcántara Santiago

Tesis sometida como requisito parcial para
obtener el grado de

**Maestro en Ciencias en la especialidad de
Ciencias Computacionales**

en el

**Instituto Nacional de Astrofísica, Óptica y
Electrónica**

Septiembre 2007

Tonantzintla, Puebla

Supervisada por:

Dr. César Torres Huitzil

©INAOE 2007

El autor otorga al INAOE el permiso de reproducir y distribuir copias
en su totalidad o en partes de esta tesis



Resumen

La construcción de sistemas de visión computacional artificial es un proceso complejo que implica diferentes etapas de procesamiento para la extracción de información útil a partir de las imágenes. Más aún, el diseñar e implementar sistemas de visión computacional no es una tarea fácil debido a la complejidad inherente y las limitaciones de tecnología de sistemas de procesamiento convencional para satisfacer la demanda computacional. No existe procedimiento unificado en la construcción de sistemas de visión, y en este sentido, los sistemas neuromórficos son una alternativa en la construcción de sistemas cuyos principios de procesamiento estén basados en los mecanismos de procesamiento neuronal de la corteza visual primaria. Para contar con dispositivos bio-inspirados, es necesario conocer y construir las etapas que comprende la percepción natural. Una de las etapas fundamentales en el proceso de percepción natural es la selectividad a la orientación de estímulos visuales que se realiza en la corteza visual primaria de los mamíferos [1]. Esta etapa involucra el procesamiento masivo por varias neuronas del estímulo visual, o elementos de procesamiento, y se realiza en un tiempo relativamente corto, en el orden de milisegundos. Por lo anterior, los sistemas de procesamiento convencionales no son convenientes para ser utilizados por su naturaleza secuencial, dado que este tipo de procesamiento presentan un acceso masivo y paralelo de datos (accesos a memoria), y procesos repetitivos.

En el presente trabajo se propone una arquitectura basada en un arreglo sistólico 3-D para la detección de bordes en múltiples orientaciones que imita, de forma aproximada, la selectividad a la orientación de las neuronas de la corteza visual primaria. La selectividad de las neuronas es implementada a través de un modelo simplificado mediante un filtrado Gabor-2D, y la organización de la arquitectura refleja la funcionalidad de una muy pequeña porción de la corteza visual primaria. La forma en que opera el filtro es por medio

de la utilización de máscaras de convolución 7×7 . El filtro Gabor-2D resalta los bordes de la imagen hallados en una orientación específica, teniendo como orientaciones a considerar: 0° , 45° , 90° , 135° , 180° , 225° , 270° , 315° . La arquitectura es diseñada bajo esquemas de procesamiento paralelo, enfocado a FPGA, bajo restricciones de operación en tiempo real. La arquitectura propuesta opera sobre imágenes en escala de grises de 512×512 , la cual fue sintetizada en un XCV3200E-6 FPGA VirtexE, logrando procesar 66.4 imágenes por segundo, ocupando un 97 % de los recursos disponibles, obteniendo ocho imágenes de salida.

Este trabajo provee una alternativa en el diseño de visión computacional, y no proporciona de ninguna manera alguna aportación biológica, por el contrario, hace uso de experimentos y conclusiones de investigadores en la materia para mostrar la plausibilidad de implementación de sistemas bio-inspirados en dispositivos digitales y muestra que estos principios pueden extenderse a la construcción de dispositivos más complejos.

Abstract

The construction of artificial vision systems is a complex process that implies different processing stages for the extraction of useful information from the image from a scene. Moreover, designing and implementing vision systems is not a easy task due to the inherent complexity and the limitations of technology in conventional processing systems to fulfill the requirements of the computational load. There is not unified procedure in the design and construction of vision systems, and in this sense, the neuromorphic systems are an alternative in the construction of a system whose architecture is based on neuronal system of Primary Visual Cortex. In order to count on bio-inspired devices, it is necessary to know and to construct the stages found in natural perception. One of the fundamental stages in the process of natural perception is the Orientation Selectivity of visual stimuli that is carried out the Primary Visual Cortex of mammals [1]. This stage involves massive processing by several cells of the visual stimulus, or processing element structure, and it is carried out in short time, in the order of milliseconds. Therefore, conventional computational systems of processing are not suitable to be used by its sequential nature, since this type of processing presents a massive and parallel of access of data (accesses to memory), and repetitive processing.

In the present work an architecture based on a 3D systolic array for detecting of edges in multiple directions, that it imitates, approximately, the orientation selectivity of the primary visual cortex cells. The selectivity of the cells is implemented through a simplified model of Gabor-2D filters, and the architecture organization reflects the functionality of a very small portion of primary visual cortex. The form in which the filter operates is by the use the convolution of 7×7 window masks. The Gabor-2D filter enhances the edges of the image found in a specific orientation, with orientations: 0° , 45° , 90° , 135° , 180° , 225° , 270° , 315° . The architecture is designed as a parallel processing computer implemented,

on an FPGA, under restrictions of real time operation. The proposed architecture operates on gray scale images of 512x512 size, and was synthesized in a XCV3200E-6 VirtexE FPGA, it reaches to process 66.4 images per second, and occupy 97 % of the resources available, obtaining eight output images.

This work provides an alternative in the design on computational vision processors, and it does not look for a biological contribution but, it is inspired by experiments and conclusions of researchers the biological area to show the possibility to be implemented in digital device and later is extended to the construction of more complex vision processors.

Agradecimientos

Al Consejo Nacional de Ciencia y Tecnología por la beca otorgada número 201574 para llevar a cabo los estudios de maestría.

Al Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE) por la formación académica recibida y el apoyo con sus instalaciones y servicios para una mejor estancia.

A mi asesor Dr. César Torres Huitzil por haberme brindado su apoyo, sus consejos, guía y tiempo invertido.

A mis revisores, Dra. Claudia Feregrino Uribe, Dr. Leopoldo Altamirano Robles y al Dr. Miguel O. Arias Estrada por el tiempo invertido en la revisión de este trabajo, así como su invaluable apoyo con sus sugerencias y disposición en el desarrollo y culminación de este trabajo.

A mi familia, por su apoyo incondicional. A mis padres, de quienes he recibido las enseñanzas y las herramientas más valiosas de la vida. A mis hermanas, Ary, Ita, Huichi, y a mi Ponchito por su gran cariño.

A Armando Pérez Leija, quien hace que la carga sea más ligera, brindándome su cariño, soporte y motivación.

A todos mis amigos, por su desinteresado apoyo. A quienes tomaron como suyo este proyecto, dándome palabras de aliento, sugerencias y motivación.

A todos aquellos quienes de una u otra forma han cultivado enseñanzas de la vida en mí, por todos los buenos y malos momentos de aprendizaje que he vivido con cada uno de ustedes, porque todo ha sido un constante aprendizaje.

Dedicatoria

a Dios
por brindarme tu ayuda

Índice general

1. Introducción	1
1.1. Motivación y Objetivo	2
1.1.1. Objetivo	3
1.2. Trabajo Relacionado	4
1.3. Organización de la Tesis	5
2. Marco Teórico	6
2.1. Introducción	6
2.2. Inspiración Biológica: Selectividad a la Orientación	8
2.3. Filtrado de Gabor	11
2.4. Aspectos generales de implementación	15
3. Procesamiento y Arquitecturas Paralelas para la Convolución Múltiple	18
3.1. Convolución de una imagen	18
3.1.1. Complejidad del operador de convolución	20
3.2. Esquema de direccionamiento y transferencia de datos	21
3.3. Convolución de una imagen con máscaras múltiples	25
3.4. Técnicas y Arquitecturas de Procesamiento Paralelo	25
3.4.1. Arquitecturas SIMD	27
3.4.2. Arquitectura Sistólica	27
3.4.3. Pipeline	29
3.5. Métodos y Herramientas	30

4. Arquitectura Sistólica 3-D para Filtrado en Múltiples Orientaciones	31
4.1. Analogía entre el Sistema Biológico y la Arquitectura Sistólica 3-D	31
4.2. Unidad Elemental de Procesamiento	36
4.3. Organización en Arreglos Sistólicos	38
4.3.1. Arreglo Sistólico 1-D	38
4.3.2. Arreglo Sistólico 2-D	41
4.3.3. Arreglo Sistólico 3-D	43
4.4. Descripción Funcional	45
4.5. Análisis de Desempeño	50
5. Modelado VHDL de la Arquitectura Sistólica 3-D	54
5.1. Aspectos Generales	54
5.2. Ejemplo de Aplicación del Filtro	55
5.3. Unidad de Procesamiento Elemental	58
5.4. Arquitectura Sistólica 2-D	64
5.5. Arquitectura Sistólica 3-D	72
6. Resultados Experimentales	76
6.1. Resultados de Simulación	77
6.1.1. Resultados obtenidos de los ocho filtros	80
6.1.2. Análisis de Precisión	83
6.2. Resultados de Síntesis	84
6.3. Discusión de resultados	85
7. Conclusiones y Trabajo Futuro	91
7.1. Conclusiones	91
7.2. Trabajo Futuro	93
Apéndices	97
A. RTL de la Arquitectura Sistólica 3-D	97
B. Imágenes de Prueba	99

C. Resultados de Máscaras 7x7	100
D. Resultados de 3 tamaños de máscaras	105
E. Resultados de Imágenes Procesadas	107

Índice de figuras

2.1. Vista conceptual de los campos receptivos de las neuronas simples en la corteza visual primaria.	9
2.2. Modelo conceptual de la organización de la corteza visual primaria, denominada <i>Organización Hipercolumnar</i> [6].	10
2.3. Función Gabor-2D, componente real y componente imaginaria, respectivamente.	13
2.4. Patrones del filtro Gabor-2D.	14
2.5. Dos vistas de la gráfica 3D correspondiente a la función Gabor-2D en su parte real, cuya orientación es 0°	16
2.6. Imagen de entrada 220x220.	16
2.7. Imágenes de salida después de aplicar filtros Gabor-2D con orientación en 0° y 45°	17
3.1. Diagrama conceptual de la aplicación del operador basado en ventana sobre una imagen de entrada.	19
3.2. Paralelismo a nivel de datos en operaciones de imágenes basadas en ventanas. . .	22
3.3. Tres ventanas se procesan en paralelo en la dirección vertical, lado izquierdo, y en la dirección horizontal, lado derecho de la figura.	23
3.4. Lectura de los píxeles de la imagen.	23
3.5. Cuatro ventanas se procesan en paralelo, tres ventanas inician su proceso de acuerdo a los datos leídos; una vez leído un dato localizado en otra columna, inicia el procesamiento de ventanas en dirección horizontal.	24
3.6. Esquema de interconexiones típicas de arreglos sistólico [18].	28

4.1.	Analogía correspondiente al modelo biológico de la corteza visual primaria, visto como la agrupación de neuronas de la misma orientación y un diagrama conceptual simplificado de la arquitectura con dicha organización.	32
4.2.	Diagrama a bloques de la arquitectura sistólica 3-D que aplica varios filtros en orientaciones específicas sobre una imagen de entrada.	34
4.3.	Analogía de la acción de una neurona sobre un área específica de la imagen de entrada y el funcionamiento del procesador etiquetado como <i>GWP</i> en la arquitectura.	37
4.4.	Analogía del modelo biológico de la corteza visual primaria actuando tres neuronas alineadas, y la arquitectura de un arreglo sistólico 1-D. La parte izquierda ilustra tres neuronas organizadas en forma lineal con orientación a 0° aplicadas sobre un área específica de la imagen de entrada. La parte derecha de la figura muestra un arreglo sistólico 1-D generalizado compuesto por <i>GWPs</i>	39
4.5.	Actividad de los <i>GWPs</i> de un arreglo sistólico 1-D.	41
4.6.	Analogía entre el modelo biológico de la corteza visual primaria en el que actúan nueve neuronas ubicadas en diferentes columnas y renglones con orientación a 0° aplicada sobre una imagen de entrada, colocando el resultado en el área que les corresponde formando la imagen de salida; teniendo su contraparte en un arreglo sistólico 2-D, el cual corresponden a un arreglo bidimensional de <i>GWPs</i> , entregando el resultado al colector de datos 2-D.	42
4.7.	Analogía entre el modelo biológico de la corteza visual primaria agrupando las neuronas en arreglos bidimensionales siendo su contraparte la extensión de la arquitectura sistólica 2-D a 3-D.	44
4.8.	Imagen con dimensiones 7×5 a la que se le aplicará dos filtros cuyas máscaras son de dimensiones 3×3	45
4.9.	Diagrama general del funcionamiento de un arreglo sistólico 1-D compuesto por tres procesadores, cuyas ventanas a procesar se encuentran al lado de cada procesador, y una vez procesadas se almacena el dato resultante en la imagen de salida.	46
4.10.	Diagrama general del funcionamiento del arreglo sistólico 2-D compuesto por tres arreglos sistólicos 1-D, cuyas ventanas a procesar se encuentran a la derecha de cada procesador.	49

4.11. Diagrama general del funcionamiento de un arreglo sistólico 3-D compuesto por dos arreglos sistólicos 2-D (mostrando sólo un arreglo sistólico 1-D de cada arreglo).	51
4.12. Desarrollo del crecimiento y decrecimiento progresivo de la actividad de siete procesadores que componen un arreglo sistólico 1-D.	52
5.1. Imagen de entrada 220x220.	56
5.2. Diagrama de tiempo de la lectura de 13 píxeles de la imagen, 7 de ellos corresponden al tamaño de la máscara necesarios para procesar una ventana de la imagen, y 6 píxeles más para procesar 7 ventanas en paralelo.	60
5.3. Diagrama de tiempo de la lectura de la tercera columna, destacando la señal <i>EP</i> la cual activa la actividad del GWP al que pertenece.	61
5.4. Diagrama de tiempo que muestra la actividad de siete procesadores, los cuales corresponden a la primera columna de la ventana de la imagen de entrada a procesar.	63
5.5. Diagrama de tiempo en la lectura de la tercera columna, y el procesamiento en el que se encuentran cuatro de los siete <i>GWPs</i>	65
5.6. Diagrama de tiempo en el que los GWP entregan su resultado al <i>Colector de Datos 1-D</i> , etiquetado como <i>px_1D</i>	66
5.7. Conexión entre las unidades de control y un arreglo sistólico 2-D.	67
5.8. Diagrama de tiempo de dos arreglos sistólicos 1-D, cuando el segundo de ellos inicia su actividad.	69
5.9. Diagrama de tiempo de un arreglo sistólico 2-D, almacenando los datos recolectados de dos colectores de datos 1-D en el <i>Colector de Datos 1-D</i>	71
5.10. Esquema RTL de un arreglo sistólico 3-D compuesto por tres arreglos sistólicos 2-D conectado con las unidades de control.	74
5.11. Diagrama de tiempo en el que se recolectan los datos de ocho arreglos sistólicos 2-D.	75
6.1. Diagrama a bloques del flujo de diseño de la Arquitectura Sistólica 3-D para validar los resultados.	76
6.2. Imágenes de salida generadas por software y por simulación para detección de bordes orientados a 45°.	79

6.3. Relación de error con respecto al número de bits que componen los coeficientes de la máscara.	84
6.4. Rendimiento de la arquitectura con diferentes grados de paralelismo.	86
6.5. Requerimientos de recursos para el arreglo sistólico 3-D en diferentes grados de paralelismo.	87
6.6. Tiempo de procesamiento de la arquitectura sistólica 3-D para diferentes tamaños de máscaras.	87
6.7. Requerimientos de recursos para el arreglo sistólico 3-D en diferentes grados de paralelismo.	88
A.1. Esquema RTL de la Arquitectura Sistólica 3-D.	98
B.1. Imágenes utilizadas para realizar pruebas con la arquitectura.	99
D.1. Imágenes de resultado después de aplicar máscaras de diferente tamaño.	106
E.1. Resultados de la imagen 192x192.	108
E.2. Resultados de la imagen 256x256.	109
E.3. Resultados de la imagen 800x600.	110

Índice de tablas

3.1. Complejidad computacional de la convolución de una imagen 2D expresada en términos de operaciones aritméticas elementales para un tamaño de imagen $M \times N$ y una máscara $w \times w$ [14].	20
3.2. Tasa de transferencia de datos (DTR) de una imagen $M \times N$ y una máscara $w \times w$ [14].	21
4.1. Ecuaciones del análisis matemático hecho a un arreglo sistólico 2-D [14].	53
5.1. Interfaz del elemento de procesamiento con una breve descripción de la funcionalidad de sus señales.	59
5.2. Interfaz Simplificada de la Arquitectura Sística 3-D.	73
6.1. Imágenes obtenidas tras aplicar el filtro Gabor-2D en software y en simulación, en las 8 orientaciones con la máscara ajustada a 10 bits.	82
6.2. Resultado de la Síntesis de la Arquitectura Sística 3-D.	85
C.1. Imágenes obtenidas tras aplicar el filtro Gabor-2D en software y en simulación, en las 8 orientaciones con la máscara 7×7 ajustada a 10 bits.	102
C.2. Máscaras 7×7 de las diferentes orientaciones.	104

Capítulo 1

Introducción

Los sentidos con los que cuenta el ser humano tienen características extraordinarias, difícilmente imitables. Específicamente, la percepción visual es una tarea compleja con la que se extrae información de utilidad del medio que lo rodea y, con base a esta información, se toma algún tipo de decisión. Sin embargo, el diseñar e implementar sistemas de visión computacional no es una tarea fácil debido a la complejidad inherente de la percepción y las limitaciones de tecnología de sistemas de procesamiento convencional.

Se han propuesto diferentes modelos en la construcción de sistemas de visión, sin embargo, ninguno puede ser considerado de propósito general, ya que son construidos para fines específicos y adecuados al problema. Es por ello que se han tomado caminos alternativos, y uno de ellos es la construcción de sistemas neuromórficos. Los sistemas neuromórficos construidos se basan en los principios usados por el sistema neuronal, esto es, tomar inspiración de la estructura y mecanismos de procesamiento inherentes de los sistemas biológicos en tarea de percepción visual [2].

El entendimiento de la percepción visual natural ha avanzado a través de diferentes estudios, y dada su naturaleza de procesamiento, el diseño y desarrollo de dispositivos de procesamiento de visión dedicados con capacidades de procesamiento espacio-tiempo son una alternativa para superar las limitaciones de tecnologías de visión convencional (aquellas basadas en una computadora de propósito general y un sensor visual).

Para poder construir un dispositivo de procesamiento de visión es necesario conocer las etapas y estructuras que rigen el funcionamiento de la percepción visual natural. Estudios realizados a la percepción visual en mamíferos ha brindado información sobre las

posibles etapas, organización jerárquica y funcionamiento del que se rigen para alcanzar su objetivo [3]. La primera de estas etapas se ubica en la corteza visual primaria, también llamada V1. Esta etapa es fundamental en el proceso de visión natural, ya que la información que se obtiene de ella es utilizada en las etapas subsecuentes [1] [4].

Mediante estudios experimentales se ha encontrado que dentro de la corteza visual primaria se hallan neuronas las cuales se clasifican en simples y complejas. Las neuronas simples responden preferentemente a una orientación específica a los estímulos visuales y se organizan en diferentes grupos a fin de destacar las características del estímulo visual, por ejemplo bordes, en ciertas orientaciones. Este grupo de neuronas actúan sobre un área específica el estímulo visual, y que en conjunto procesan la imagen captada. Lo anterior se ha comprobado experimentalmente mediante el registro de la actividad neuronal de seres vivos al ser estimuladas mediante patrones de iluminación con alguna orientación específica (pudiendo ser una línea o una barra). Este tipo de procesamiento involucra un gran número de elementos de procesamiento, requerimientos de interconectividad entre las neuronas, y se encuentra sujeto a restricciones en la velocidad de respuesta [1].

Una arquitectura bio-inspirada cuyo proceso sea imitar la selectividad a la orientación que realizan las neuronas simples en la corteza visual primaria ¹ no se puede realizar bajo un paradigma de modelos convencionales ya que el sistema neuronal presenta características como acceso masivo y paralelo de datos (accesos a memoria), interconectividad densa, procesos estructurales no lineales, procesos repetitivos, tiempo de procesamiento reducido.

1.1. Motivación y Objetivo

Avances en tecnologías de computadoras ha hecho que la construcción de sistemas de visión sea un problema manejable, resultando en un extenso uso de estos sistemas en aplicaciones tales como robótica, multimedia, realidad virtual, inspección industrial, ingeniería médica, y navegación autónoma.

En base a la tecnología disponible, se busca realizar un modelo aproximado de la primera etapa en la percepción visual bajo un enfoque digital tomando en cuenta el alto

¹El concepto en inglés es Orientation Selectivity of Primary Visual Cortical Cells.

poder de procesamiento que implica. La corteza visual primaria involucra un procesamiento masivo sobre una imagen captada, donde se necesita calcular, para cada neurona, un operador de ventana (convolución) y se debe realizar en varias orientaciones sobre la misma imagen involucrando tiempo de cómputo elevado. El problema radica en cómo imitar el paralelismo a través de una arquitectura adecuada; cómo resolver los problemas de acceso a datos dado que los modelos convencionales trabajan sobre un paradigma de acceso a memoria secuencial; cómo estructurar la organización 3D y de ahí la interconexión existente entre las neuronas; realizando este modelo bajo un enfoque digital.

Por lo anterior, la arquitectura debe reflejar la estructura y comportamiento de las neuronas en la corteza visual primaria, dando como resultado imágenes procesadas con significado en tiempo real; además de contar con una arquitectura digital que provea este medio como viable en la construcción de sistemas neuronales más complejos (por ejemplo, sistemas de percepción de movimiento y/o segmentación de imágenes).

Se ha elegido la tecnología FPGA (Field Programmable Gate Array) como medio para explorar y validar la arquitectura dado que provee estructuras para la implementación de modelos paralelos, alta densidad para lógica-aritmética, y se pueden manejar operaciones a nivel de bit para construir rutas de datos especializadas.

El hardware reconfigurable es una buena opción para el prototipado e implementación de sistemas neuromórficos, ya que con aquel se pueden definir arquitecturas de gran complejidad e implementar sistemas que requieran mecanismos de adaptación, plasticidad y aprendizaje a medio-largo plazo; es decir, cubren muchas de las necesidades de los sistemas neuromórficos. La aplicación del hardware reconfigurable al campo de la visión artificial está cada vez más extendida, para situaciones en las que el procesamiento en tiempo real es necesario [9].

1.1.1. Objetivo

El objetivo general del presente trabajo es diseñar e implementar una arquitectura basada en FPGA para el filtrado de imágenes en múltiples orientaciones imitando, de manera aproximada, el paralelismo del sistema biológico que determina la selectividad a la orientación que realizan las neuronas simples en la corteza visual primaria de mamíferos.

La arquitectura operará sobre imágenes de 512x512 en escala de grises, aplicando

8 filtros Gabor-2D mediante máscaras de convolución de tamaño 7×7 , cumpliendo con requisitos de tiempo real, esto es, 30 imágenes por segundo.

1.2. Trabajo Relacionado

Es necesario integrar desarrollos e investigaciones precedentes para guiar el diseño de sistemas con características de procesamiento paralelo y requerimientos de tiempo real, además de usar avances tecnológicos que hagan posible alcanzar el objetivo planteado. En el presente trabajo no se intenta realizar un nuevo modelo conceptual del sistema visual. Se tiene como objetivo la realización concreta de una arquitectura biológicamente inspirada para imitar, de manera aproximada, la selectividad a la orientación de la corteza visual primaria y con ello, probar la viabilidad del hardware en la construcción de sistemas más complejos con un impacto potencial para construir sistemas neuromórficos digitales.

En el trabajo de Shimonomura [1] se diseñó y fabricó múltiples chips VLSI analógico emulador de la selectividad a la orientación de neuronas en la corteza visual primaria. Cada chip emulaba el comportamiento de neuronas simples las cuales realizaban la selección de acuerdo a una orientación específica; 0° , 60° y 120° . El número de neuronas emuladas era igual al número de píxeles existentes en la imagen a procesar, es decir, cada neurona se dedica exclusivamente al procesamiento del área que le corresponde procesar, actuando todas en forma paralela. Este dispositivo fue realizado bajo diseño analógico, y muestra que para una implementación en hardware se requiere un alto poder computacional. Este sistema cumple con restricción en tiempo real. Con este trabajo se ha probado la factibilidad de realizar dicho procesamiento en sistemas analógicos, y expresa que aparentemente no sea posible llevar estos procesos a sistemas digitales. Con base a la revisión [15], se han detectado posibilidades que permitan definir una arquitectura adecuada para el problema de la selectividad a la orientación realizada en la corteza visual primaria bajo un dominio digital donde se tendría ventajas de flexibilidad y tiempo de diseño que lleve a desarrollos de sistemas mas complejos.

En [5] se presenta una red neuronal de alto desempeño basada en FPGA la cual implementa el filtro de Gabor, con el cual se intenta modelar el comportamiento de las neuronas de la corteza visual primaria. El filtro es implementado mediante una arquitectura CNN

(Cellular Neural Network), y opera a 120 MHz. Para el objetivo del trabajo, el filtro de Gabor está dado mediante máscaras de convolución las cuales se aplican en toda la imagen dando imágenes de salida filtradas.

Como una aproximación a lo que se busca alcanzar como objetivo de la tesis, en [14] se muestra una arquitectura sistólica 2D para el procesamiento de imágenes basado en ventanas. Se propone un esquema de direccionamiento y transferencia de datos apropiados con lo que permite explotar el paralelismo inherente del procesamiento de imágenes basado en ventanas. La arquitectura fue implementada en un FPGA y es comparable a otras arquitecturas en términos de rendimiento y uso de hardware. Esta arquitectura es de utilidad en lo que se pretende resolver, ya que tiene características semejantes; el hecho de involucrar el filtrado de una imagen a través de una ventana.

Los filtros que operan en el sistema son filtros Gabor ya que son éstas las que mejor se ajustan al tipo de transformación que producen las neuronas simples de la corteza visual primaria, además de ser el modelo más comúnmente aceptado. Un filtro Gabor-2D es ampliamente aplicado a procesamiento de imágenes, tales como detección de bordes, discriminación de textura y cálculo de la disparidad estéreo [1].

1.3. Organización de la Tesis

El presente documento se encuentra organizado de la siguiente forma. En el capítulo 2 se presenta el modelo conceptual del funcionamiento y organización de las neuronas simples en la corteza visual primaria. En el capítulo 3 se presentan arquitecturas paralelas adecuadas para modelar el sistema, presentando las razones por las cuales se eligen algunas de ellas para diseñar nuestra arquitectura. En el capítulo 4 se da una descripción funcional y organizacional de la arquitectura propuesta, analizando su desempeño. El capítulo 5 presenta los componentes principales modelados en VHDL con algunos detalles de interconexión entre ellos. En el capítulo 6 se exponen los resultados que se alcanzaron en las diferentes etapas de desarrollo, presentando algunos compromisos y limitaciones de la arquitectura propuesta. En el capítulo 7 se dan las conclusiones y algunas propuestas para un trabajo futuro que surgen a partir de la tesis. Adicionalmente, se añadieron apéndices de algunas imágenes procesadas.

Capítulo 2

Marco Teórico

2.1. Introducción

El sistema visual humano es uno de los sentidos que más se usan en el reconocimiento de características espaciales del ambiente. Aunque son muchos los avances de las últimas décadas, los sistemas computacionales inteligentes aún no son capaces de brindar dispositivos que se aproximen a la eficiencia del sistema visual humano en la extracción de información a partir de la escena. La conversión de una imagen a información útil es un proceso complejo y requiere algoritmos de alto nivel hasta llegar a la etapa de interpretación de la escena. No obstante, estos algoritmos se basan en características de bajo nivel que son computacionalmente muy costosas (por ejemplo, detección de bordes de una imagen).

Existen métodos de análisis y procesamiento de imágenes en computadora los cuales se basan en métodos de visión humana, de igual manera, existen otros tantos que no tienen contraparte en dicho sistema de visión. Con el afán de contar con dispositivos que brinden resultados satisfactorios en la percepción visual del ambiente, se ha tomado la alternativa en la construcción de sistemas neuromórficos.

Un sistema neuromórfico es un sistema basado en los principios usados por el sistema neuronal [2], esto es, tomar inspiración de la estructura y mecanismos de procesamiento inherentes de los sistemas biológicos. Propiamente, la Ingeniería Neuromórfica es un campo de investigación que trata del diseño de sistemas artificiales de computación que

utilizan propiedades físicas, estructuras o representaciones de la información basadas en el sistema nervioso biológico [9].

La construcción de sistemas neuromórficos es una tarea difícil, ya que los sistemas neuronales tienen características inherentes difícilmente modelables. Entre estas características se encuentra el paralelismo, proceso colectivo, velocidad de respuesta, entre otras. Por lo anterior, los sistemas convencionales no son una alternativa en la construcción de dichos sistemas, ya que tienen características que limitan la eficiencia de éstos. En el capítulo 3 se presentan alternativas en la construcción de arquitecturas adecuadas en la construcción de estos sistemas.

Para contar con un dispositivo de procesamiento de visión inspirado en sistemas neuronales, se debe partir de etapas tempranas de procesamiento en la visión natural. La primera etapa de procesamiento inicia en la corteza visual primaria del sistema visual humano. Enseguida, se presenta los fundamentos biológicos que sustentan la funcionalidad de la corteza visual primaria, del cual se propone un modelo biológico sobre la funcionalidad y organización de las neuronas en esta área del cual es interés de este trabajo.

El procesamiento visual inicia en el ojo, donde los lentes enfocan luz dentro de la retina. La retina no sólo traduce la luz a una señal neuronal, también ejecuta un proceso inicial de la entrada visual. Las neuronas del ganglio retinal conduce la salida de la retina hacia al *Núcleo Geniculado Lateral* (LGN- *Lateral Geniculate Nucleus*), el cual está localizado en un área llamada el Tálamo dentro del cerebro. El tálamo conduce esta salida a la corteza visual en la parte posterior del cerebro, donde el procesamiento visual toma lugar [28].

La corteza visual está dividida en áreas funcionales: V1, V2, V4, MT, MST, etc. Estas áreas están organizadas de manera jerárquica, donde las áreas más bajas como V1 alimentan a niveles superiores como V2. Además, hay retroalimentación de áreas superiores hacia áreas más bajas.

Concretamente, la corteza visual primaria posee 2 tipos básicos de neuronas, que de acuerdo a su campo receptivo se les han llamado simples y complejas. Las neuronas simples poseen un campo receptivo con una dirección y grado de inclinación específicos y lineales (rectos), por tanto responden mejor a estímulos que correspondan con su campo específico del cual sean responsables de procesar. Por lo anterior, se dice que las neuronas son selectivas a la orientación. Por otro lado, las células complejas poseen un campo

receptivo más flexible en este sentido y responden mejor al movimiento. Concretamente, basado en el interés de este trabajo, se concentrará en el funcionamiento y organización de las neuronas simples de la corteza visual primaria.

Además, la selectividad a lo largo de otras dimensiones de estímulos comúnmente asociada con V1, tales como dirección del movimiento y disparidad binocular, pueden ser obtenidas mediante la combinación de las salidas de las neuronas ajustadas a una orientación. Las áreas subsecuentes son selectivas a combinaciones de orden más alto de dimensiones previas [6].

Las neuronas simples de la corteza visual primaria cumplen con una organización para responder eficientemente a estímulos incidentes. Esta organización fue denominada *Organización Hipercolumnar* y la respuesta a los estímulos que inciden en ella corresponden a un filtro Gabor, el cual es la función que refleja, de una manera aproximada, el comportamiento de estas neuronas. En las siguientes secciones se expone las características de la Organización Hipercolumnas y el Filtro Gabor, para después ser modelado bajo una arquitectura hardware.

2.2. Inspiración Biológica: Selectividad a la Orientación

Hubel y Wiesel, ambos médicos, estudiaron las propiedades fisiológicas de las neuronas simples en la corteza visual primaria. Se sabe que estas neuronas reciben información proveniente de pequeñas áreas de la retina, o campos receptivos, los cuales se encuentran dispuestos de modo análogo a un mosaico. De igual manera, estas neuronas están asociadas a campos receptivos retinianos. Los campos receptivos de las neuronas contienen sub-regiones que ejercen una influencia excitatoria, y sub-regiones que ejercen una influencia inhibitoria a un estímulo aplicado. Verificaron que, mediante el registro de la actividad bioeléctrica de las neuronas, responden de modo máximo cuando una línea de una determinada anchura y orientación incide sobre cierta posición en su campo receptivo asociado.

El estímulo más efectivo para este campo receptivo particular es en el que una línea se encuentra en la región excitatoria. Esta línea debe de tener la orientación, posición y tamaño correctos para que la neurona logre ser excitada. Los estímulos que no son

adecuados en términos de posición, orientación o tamaño son menos efectivos. En la Fig 2.1 se encuentra ilustrada una neurona que reconoce estímulos en la orientación de 90° , cuyos campos inhibitorios se encuentran en color gris oscuro, y su campo excitatorio se encuentra en color gris claro. La figura 2.1(a) hace referencia a un estímulo adecuado, ya que se encuentra en tamaño, orientación y posición correctas, por lo tanto excita a la neurona. La figura 2.1(b) la neurona no se excita ya que el estímulo no cumple en encontrarse en la posición correcta para que la neurona sea excitada. las figuras 2.1(c) y 2.1(d), de igual manera, la neurona no se excita, ya que el estímulo no tiene la orientación correcta ni tamaño correcto, respectivamente.

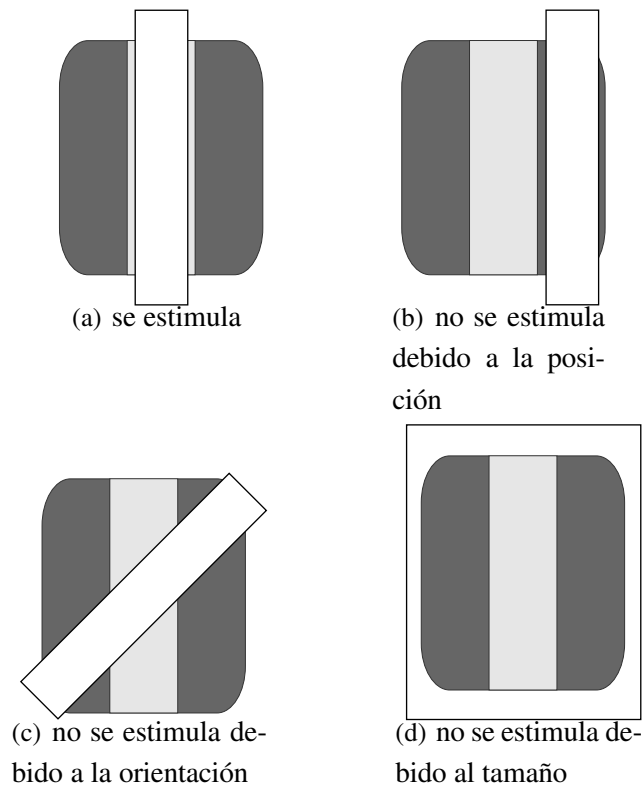


Figura 2.1: Vista conceptual de los campos receptivos de las neuronas simples en la corteza visual primaria.

La funcionalidad de estas neuronas en la corteza visual primaria se le denomina selectiva a la orientación, dado que responden a estímulos que se encuentran en determinada orientación asociadas a un campo retiniano específico. Este procesamiento que realizan

las neuronas simples proporciona características de bordes y textura de una imagen dada.

Hubel y Wiesel sugieren que puede ser pensada como una hoja de 2-D de tejido neuronal, con una extensión limitada en profundidad. Esta forma de organización la denominaron *Organización Hipercolumnar* [6].

Una *hipercolumna* es la unidad básica en que se organizan las neuronas en la corteza visual primaria en la recepción de la información la cual comprende un conjunto de *columnas*. Una *columna* es un conjunto de neuronas simples las cuales analizan líneas en una orientación específica, en una región particular. Esta organización se ilustra en la Fig. 2.2, donde se muestra la organización de una muy pequeña parte de la corteza visual primaria.

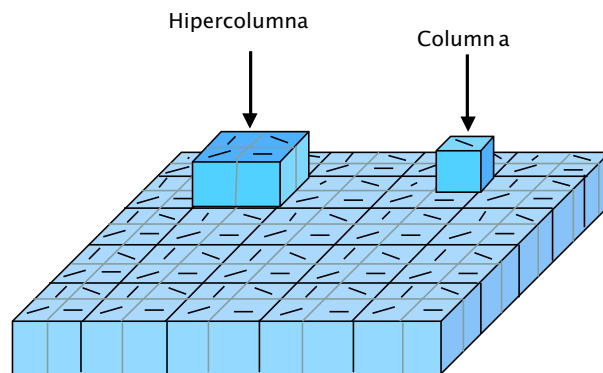


Figura 2.2: Modelo conceptual de la organización de la corteza visual primaria, denominada *Organización Hipercolumnar* [6].

En el modelo biológico de la corteza visual primaria, específicamente de las neuronas simples, las *columnas* son ilustradas como cubos. Estas *columnas* también son conocidas como *columnas de orientación* las cuales son sensibles sólo a un estímulo con cierta orientación espacial específica. El conjunto de las distintas *columnas de orientación* constituyen una *hipercolumna*. Estas *hipercolumnas* se van repitiendo, al igual que los elementos que la conforman, regularmente por toda la superficie de la corteza visual primaria, donde cada *hipercolumna* corresponde al elemento neuronal necesario para analizar un punto de la retina.

Las *hipercolumnas* se van interconectando con otras semejantes constituyendo un entramado complejo que ayuda a recibir y ordenar toda la información visual que llega a la

corteza visual primaria, lugar desde el cual será destinada hacia otros centros superiores encargados de otras funciones. Esta área constituye tan sólo el primer paso de procesamiento de la información dentro del cerebro.

Este modelo ha sido aceptado por investigadores en el área. Los trabajos de Wiesel y Hubel les valieron compartir el premio Nobel de Medicina en 1981 con el norteamericano Robert Sperry.

Las funciones de Gabor son las que mejor se ajustan al tipo de transformación que producen las neuronas simples de la corteza visual. Estas neuronas parecen ser el soporte fisiológico de los detectores de frecuencias espaciales orientadas. En la siguiente sección se presenta dicha función.

2.3. Filtrado de Gabor

Maffei y Fiorentini [7] sugirieron que las células simples eran el soporte fisiológico de los detectores de la frecuencia espacial, por lo que la corteza visual primaria operaba como un analizador de frecuencias espaciales.

Este problema tiene un gran interés en el ámbito del procesamiento visual, dado que el cómputo realizado por el sistema visual de primates y humanos parece utilizar representaciones de la imagen espacial y espectral, local y global. En síntesis, estos sistemas visuales han hallado una solución que parece aplicar un análisis bidimensional en el dominio frecuencial a cada campo receptivo retiniano, codificando la imagen en función de las respuestas a una población de neuronas, susceptibles de modelizarse mediante un banco de filtros de Gabor-2D.

Marcelja tras observar los perfiles de los campos receptivos y las frecuencias de sintonía de las células simples corticales, llegó a la conclusión de que la representación cortical de una imagen debe contener tanto parámetros espaciales como frecuenciales. También mostró que las funciones que mejor ajustan (minimizan la suma de los errores cuadráticos) los perfiles de las neuronas simples, registrados por Movshon, Thompson y Tolhurst son las funciones de Gabor. Daugman, 1980, se ocupó del estudio de perfiles bidimensionales [7].

Pollen y Ronner estudiaron pares de células simples adyacentes en la corteza visual del

gato y observaron que, generalmente, estaban sintonizadas a la misma frecuencia espacial y a la misma orientación, pero sus respuestas diferían, en cuanto a fase, en 90° , es decir, se hallan en cuadratura fásica. Esto les sugirió que las neuronas simples, sintonizadas a la misma frecuencia espacial y orientación, podían modelizarse mediante filtros Gabor simétricos y asimétricos [7].

Jones y Palmer señalaron que en el sistema visual humano, cada neurona simple tiene asociados dos campos receptivos retinianos. Por ser las funciones de Gabor funciones complejas, constan de parte real (even filter o fase coseno, $\phi = 0^\circ$) y parte imaginaria (odd filter fase seno, $\phi = 90^\circ$), que pueden descomponerse en dos partes, la parte que tiene un perfil simétrico emulará a un detector de barras y la que tiene un perfil asimétrico a un detector de líneas [7].

Un filtro de Gabor-2D es una sinusoidal compleja orientada modulada por una función Gaussiana de 2D, el cual está dado por la siguiente ecuación [7] [10]:

$$G_{a,f_0,\theta}(x,y) = g_a(x,y)e^{i2\pi f_0 x} \quad (2.1)$$

donde

$$g_a(x,y) = e^{-\pi a^2(x^2 + T^2 y^2)} \quad e \quad i = \sqrt{-1} \quad (2.2)$$

La frecuencia y orientación están dadas por f_0 y θ , respectivamente; $g_a(x,y)$ es la función Gaussiana con parámetro de escala a . Al aplicar una rotación de un ángulo θ , por la ecuación de Euler se tiene

$$\begin{aligned} x &= x \cos \theta + y \sin \theta \\ y &= -x \sin \theta + y \cos \theta \end{aligned} \quad (2.3)$$

sustituyendo la ecuación 2.3 en la ecuación 2.1 se tiene

$$G_{a,f_0,\theta}(x,y) = e^{-\pi a^2[(x \cos \theta + y \sin \theta)^2 + T^2(-x \sin \theta + y \cos \theta)^2]} e^{i2\pi f_0(x \cos \theta + y \sin \theta)} \quad (2.4)$$

Descomponiendo $G_{a,f_0,\theta}(x,y)$ en sus partes real e imaginaria, se tiene

$$G_{a,f_0,\theta}(x,y) = R_{a,f_0,\theta}(x,y) + iI_{a,f_0,\theta}(x,y) \quad (2.5)$$

donde

$$R_{a,f_0,\theta}(x,y) = g_a(x,y) \cos 2\pi f_0(x \cos \theta + y \sin \theta) \quad (2.6)$$

$$I_{a,f_0,\theta}(x,y) = g_a(x,y) \sin 2\pi f_0(x \cos \theta + y \sin \theta) \quad (2.7)$$

Las gráficas de la ecuación 2.6 y la ecuación 2.7 se aprecian en la Fig. 2.3.

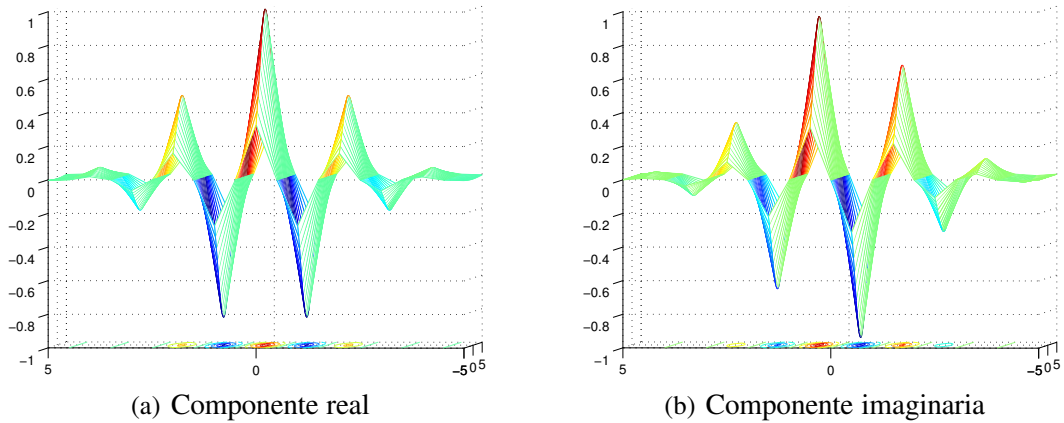


Figura 2.3: Función Gabor-2D, componente real y componente imaginaria, respectivamente.

Los parámetros que caracterizan un filtro Gabor-2D están, por lo tanto, dados por la frecuencia f_0 , la orientación θ y la escala a . Para interés de este trabajo, sólo se asumirá la parte real, dado que la parte imaginaria proporciona otro tipo de características como textura [7].

La frecuencia f_0 es la frecuencia radial de sintonía, ésta determina la anchura de la barra a detectar. La orientación θ determina el ángulo de inclinación que tiene la barra en la imagen, y a es un escalar que determina la anchura de banda angular.

Una vez que se determinan los valores de los parámetros de la función Gabor-2D, se obtienen los siguientes patrones para orientaciones: 0° y 45° . En la Fig. 2.4 se aprecian estos patrones.

En el procesamiento digital de imágenes, al aplicar un filtro se acentúa o disminuyen ciertos aspectos de la imagen en cuestión, en este caso, tomando sólo la parte real del filtro Gabor 2-D, resaltan los bordes de la imagen hallados en una orientación específica.

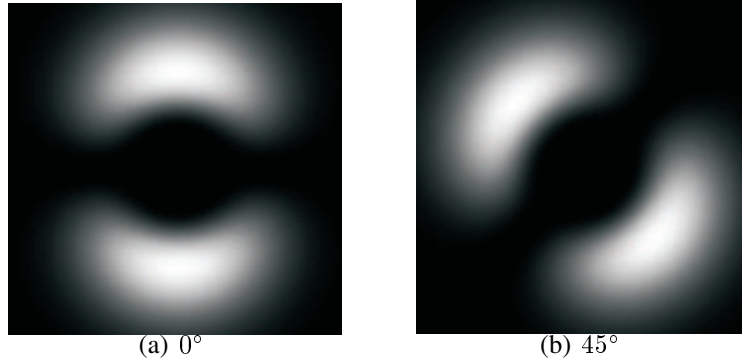


Figura 2.4: Patrones del filtro Gabor-2D.

El resultado de aplicar el filtro de Gabor-2D a una imagen en escala de grises $f(x, y)$ se obtiene mediante la convolución de la imagen con el filtro Gabor $G_{\sigma, f_0, \theta}(u, v)$, es decir

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x+u, y+v) G_{a, f_0, \theta}(u, v) du dv \quad (2.8)$$

En aplicaciones, la implementación de la ecuación 2.8 es costoso desde el punto de vista numérico y computacional. Por lo tanto, es necesario aproximar la función Gabor-2D con sumatorias, al dominio discreto, con ello se logra un mejor medio para aplicar el filtro en el procesamiento de una imagen, donde el filtro Gabor en el dominio espacial está dada por la ecuación 2.9:

$$H(u, v) = e^{-2\pi^2 \sigma_g^2 ((u-f \cos \theta)^2 + (v-f \sin \theta)^2)} \quad (2.9)$$

donde σ es la desviación estándar de la curva Gaussiana la cual determina la extensión del filtro en el dominio espacial.

Dada una vecindad de ventana de tamaño $w \times w$ con $w = 2k+1$, la convolución discreta de $f(x, y)$, imagen en niveles de gris, en su componente real de $G_{a, f_0, \theta}(x, y)$ es

$$G_R(x, y|a, f_0, \theta) = \sum_{l=-k}^k \sum_{m=-k}^k f(x+l, y+m) H(l, m) \quad (2.10)$$

Dado que se transforma la convolución de una integral infinita a una sumatoria finita, es de esperarse la pérdida de detalle en el resultado de la convolución, dado que se toman

menos muestras (determinado por el tamaño de la máscara) que se aproxima a la función real.

La forma en que operará el filtro será por medio de la utilización de máscaras que recorren toda la imagen centrando las operaciones sobre los píxeles que se encuadran en la región de la imagen original que coincide con la máscara y el resultado se obtiene mediante la sumatoria del producto de los píxeles originales y los diferentes coeficientes de las máscaras.

Para contar con una máscara 7×7 se especifica $k = 3$, obteniendo la siguiente máscara del filtro Gabor-2D con $\theta = 0^\circ$, $a = 0.239718425$ y $f_0 = 1/2$.

$$\begin{bmatrix} -0.132 & -0.16486 & -0.18839 & -0.19695 & -0.18839 & -0.16486 & -0.132 \\ 0.32553 & 0.40658 & 0.46459 & 0.48572 & 0.46459 & 0.40658 & 0.32553 \\ -0.5595 & -0.6988 & -0.79852 & -0.83483 & -0.79852 & -0.6988 & -0.5595 \\ 0.6702 & 0.83706 & 0.95651 & 1 & 0.95651 & 0.83706 & 0.6702 \\ -0.5595 & -0.6988 & -0.79852 & -0.83483 & -0.79852 & -0.6988 & -0.5595 \\ 0.32553 & 0.40658 & 0.46459 & 0.48572 & 0.46459 & 0.40658 & 0.32553 \\ -0.132 & -0.16486 & -0.18839 & -0.19695 & -0.18839 & -0.16486 & -0.132 \end{bmatrix} \quad (2.11)$$

Para las máscaras generadas del filtro Gabor-2D, a y f_0 se mantienen constantes, no así el valor de θ el cual se irá variando de acuerdo a las orientaciones que se desean aplicar.

La Fig. 2.5 corresponde a la gráfica generada por el filtro Gabor-2D discreto.

Para ilustrar el resultado de aplicar filtros Gabor-2D, se realizó la siguiente prueba en MATLAB. Contando con una imagen de entrada, Fig. 2.6, se le aplican dos filtros, máscaras, sintonizadas a 0° y 45° , obteniendo como resultado las imágenes de la Fig. 2.7.

En la siguiente sección se plantea un diseño para implementar una arquitectura basado en la organización y funcionalidad de las neuronas simples de la corteza visual primaria, mostrando la complejidad que conlleva dicha implementación.

2.4. Aspectos generales de implementación

Al buscar llevar la *organización hipercolumnar* al área de procesamiento digital de imágenes, se debe de tomar varias condiciones para lograr, aunque de manera aproximada, una organización de tal naturaleza. Se puede modelar una *columna* diseñando un

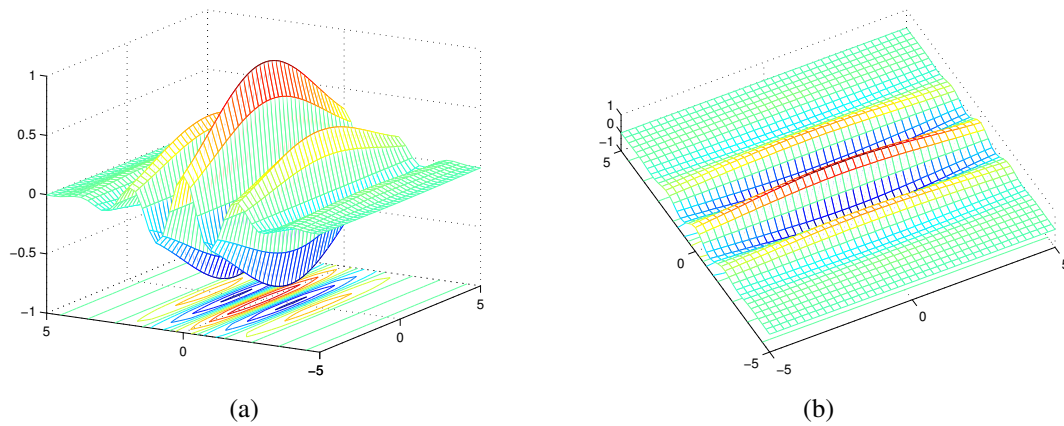


Figura 2.5: Dos vistas de la gráfica 3D correspondiente a la función Gabor-2D en su parte real, cuya orientación es 0° .

procesador que cumpla con la función que ésta desempeña. Además, este procesador estaría dedicado al procesamiento sobre un área específica de la imagen a analizar, es decir, es necesario contar con el mismo número de procesadores cómo píxeles se hallan en la imagen. Con lo anterior se logra contar con un procesamiento de las neuronas cuya detección sea sobre una orientación específica. Si ahora se modela una *hipercolumna*, entonces se habla de contar con cierto número de *columnas* dedicadas al análisis del mismo espacio de la imagen, y a su vez esta estructura replicada tanto número de veces como píxeles en la imagen. Aunado a esto, hay que contar con un procesamiento paralelo ya que las

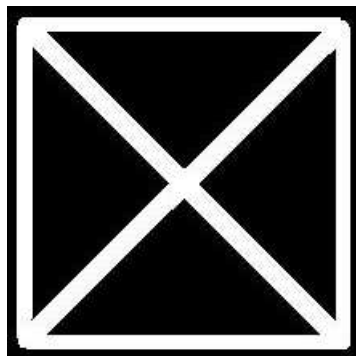


Figura 2.6: Imagen de entrada 220x220.

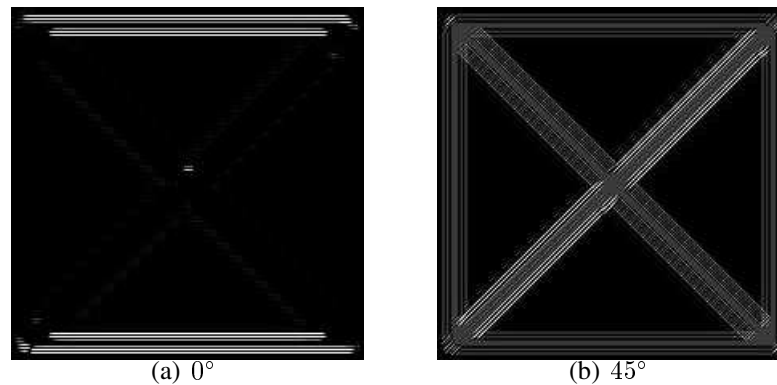


Figura 2.7: Imágenes de salida después de aplicar filtros Gabor-2D con orientación en 0° y 45° .

columns actúan al mismo tiempo al ser estimuladas. Bajo este diseño se debe contar con procesamiento paralelo, es decir, cada procesador debe actuar sobre su espacio asignado en forma paralela junto con otros procesadores, ya que la naturaleza de la *organización hipercolumnar* es una actuación simultánea. Una vez obtenida la respuesta, se debe ser capaz de manejar y organizar dicha salida para su interpretación posterior. Por lo anterior, se requiere un alto costo computacional para lograr igualar un proceso tan denso en paralelismo y acceso de datos, cumpliendo con requerimientos de tiempo y espacio.

La limitante de implementar el diseño anterior radica principalmente en la disposición de los datos (píxeles de la imagen) que se requieren para que todos los procesadores actúen en forma paralela. Por tanto, es necesario contar con una organización adecuada que brinde procesamiento paralelo y por otro lado no se vea comprometido el objetivo de aproximarse a la organización y funcionalidad de las neuronas simples de la corteza visual primaria.

Para evitar confusión, la *columna* la cual es parte fundamental en la *organización hipercolumnar* se nombrará como neurona en el resto del documento.

Capítulo 3

Procesamiento y Arquitecturas

Paralelas para la Convolución Múltiple

El filtrado por regiones, en el dominio espacial, se realiza mediante la operación de convolución de la imagen de entrada con una máscara o filtro de menor dimensión que la imagen. En este capítulo se presentan aspectos básicos sobre la convolución, además de algunas técnicas en el procesamiento paralelo convenientes para el tratamiento de este procesamiento.

3.1. Convolución de una imagen

La forma más simple de aplicar la convolución es teniendo una función de operación basado en ventana la cual es una suma de la multiplicación de cada uno de los píxeles comprendidos en la ventana de la imagen de entrada por los coeficientes de la máscara que le corresponden de acuerdo a la posición que ocupa en la máscara. La ecuación de convolución a aplicar a las imágenes viene dada por la ecuación 2.10.

Como se puede ver en dicha ecuación, se encuentra restringida a una vecindad de datos de la imagen centrada sobre un píxel de referencia. Una operación basada en ventana es ejecutada cuando una ventana, cuya área es $w \times w$ píxeles, es extraída de la imagen de entrada y es transformada de acuerdo a una máscara para producir un resultado de salida. El tamaño de la máscara es el mismo que de la ventana de la imagen, y sus valores

dependen del tipo de características que se deseen extraer de la imagen los cuales se mantienen constantes a través del procesamiento de la imagen. Esto se puede apreciar en la Fig. 3.1 [14].

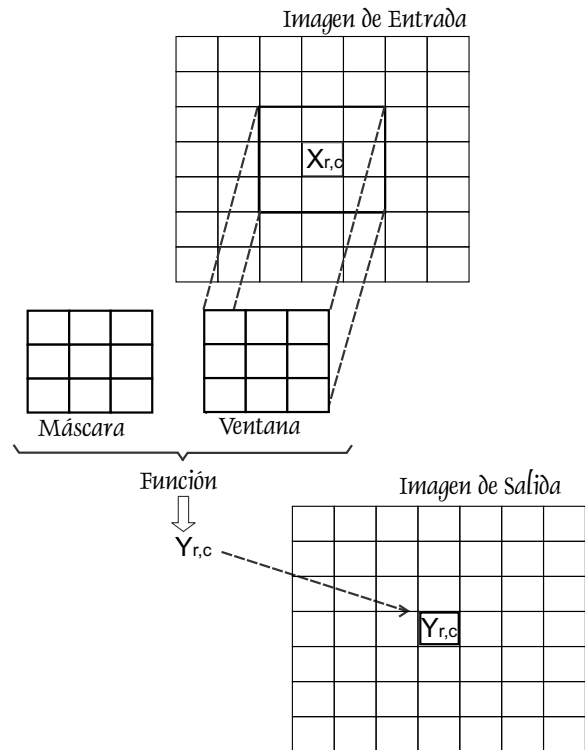


Figura 3.1: Diagrama conceptual de la aplicación del operador basado en ventana sobre una imagen de entrada.

La ventana extraída de la imagen de entrada se encuentra centrada en el píxel $X_{r,c}$ (un valor de píxel cuya ubicación es el renglón r y columna c), y una vez que obtiene el resultado de aplicar el operador basado en ventana, el resultado se ubica en la misma coordenada que del píxel central. Para obtener la imagen de salida se necesita aplicarle a todas las ventanas de tamaño $w \times w$ existentes en la imagen de entrada el operador de ventana. Esto es, si se tiene una imagen de entrada $M \times N$ se obtiene la imagen de salida $M \times N$ tras la aplicación de la máscara sobre $M \times N$ ventanas extraídas de la imagen de entrada. Estas ventanas de la imagen son extraídas al desplazar una ventana de referencia de tamaño $w \times w$ en la dirección horizontal y vertical a través de renglones y columnas en la imagen de entrada.

3.1.1. Complejidad del operador de convolución

Aplicar la operación de convolución sobre una imagen de entrada involucra una cantidad finita de operaciones repetitivas sobre los datos de la imagen. En visión computacional, este proceso es costoso.

Lo pesado que puede resultar el procesamiento de algoritmos de convolución basados en máscaras depende de varios factores, siendo de gran relevancia el tamaño de la imagen y de la máscara [14].

La complejidad de la operación de convolución puede ser expresada en términos de las operaciones aritméticas requeridas para procesar una imagen. El total de operaciones requeridas para ejecutar esta operación, excluyendo la generación de direcciones y otras operaciones de control, se encuentra resumido en la Tabla 3.1 teniendo una imagen de entrada $M \times N$ y una máscara $w \times w$ [14].

Operaciones Elementales	Número de ejecuciones
Multiplicación	$w^2 \times M \times N$
Suma	$(w^2 - 1) \times M \times N$
Cargar/Almacenar	$(2 \times w^2 + 1) \times M \times N$

Tabla 3.1: Complejidad computacional de la convolución de una imagen 2D expresada en términos de operaciones aritméticas elementales para un tamaño de imagen $M \times N$ y una máscara $w \times w$ [14].

Para ilustrar el número de operaciones elementales requeridas en un caso particular, se aplicará a una imagen de entrada 512×512 , con una ventana de máscara 7×7 , dando como resultado arriba de los 50 millones de operaciones. Por lo tanto, se requiere un poder computacional de procesamiento en giga-operaciones por segundo (GOPs) para lograr desempeño en tiempo real (por ejemplo 30 cuadros por segundo). La carga computacional crece con imágenes y máscaras de mayor tamaño [14].

En términos generales, un operador basado en ventana tiene una complejidad computacional de $O(w^2 \times M \times N)$ para una imagen $M \times N$ con una ventana de máscara $w \times w$.

Este procesamiento involucra una alta tasa de transferencia de datos (DTR), resumida en la Tabla 3.2.

Datos	Tasa de transferencia de datos (DTR)
Entrada	$DTR_I = b \times (2 \times w \times w) \times M \times N \times f_F$
Salida	$DTR_O = b \times M \times N \times f_F$

Tabla 3.2: Tasa de transferencia de datos (DTR) de una imagen $M \times N$ y una máscara $w \times w$ [14].

Para el cálculo de la transferencia de datos, se parte del caso generalizado de una imagen $M \times N$, aplicando una máscara $w \times w$, con b bits para la representación de los valores; y con una tasa de procesamiento de la imagen f_F .

3.2. Esquema de direccionamiento y transferencia de datos

La memoria se encuentra organizada en forma lineal, esto es, los píxeles vecinos en la imagen no son necesariamente elementos vecinos en la memoria lineal. Esto complica aún más los accesos a la memoria. Aunado a esto, el número de veces que se accede a la memoria por cada ventana a procesar corresponde al tamaño de la máscara, es decir, se deben acceder $w \times w$ veces a la memoria para contar con todos los datos necesarios para el procesamiento de una sola ventana, bajo un esquema estrictamente secuencial.

Dado que la convolución en el procesamiento de imágenes implica procesos repetitivos, y que un dato es requerido en más de un proceso, se infiere el reuso de datos. Esto se puede apreciar en la Fig. 3.2. Se tiene una imagen de entrada, en la que se distinguen tres ventanas de tamaño 7×7 las cuales son ventanas vecinas entre sí, en la que cada una de ellas representa una ventana a ser procesada. Se puede apreciar que las tres ventanas comparten datos. Las tres máscaras sólo se encuentran traslapadas en un renglón o en una columna con respecto a la tercera, esto con el fin de destacar los datos que se tienen en común. Esto implica que algunos píxeles podrían ser usados para el cálculo de las tres ventanas en el mismo ciclo de lectura de la memoria de entrada disminuyendo el número de accesos a la memoria por un factor de tres sobre el área de datos en común.

Partiendo de la Fig. 3.2, se aprecian tres ventanas que pueden ser procesadas en paralelo, esto es, conforme se van recibiendo cada uno de los datos que se encuentran en

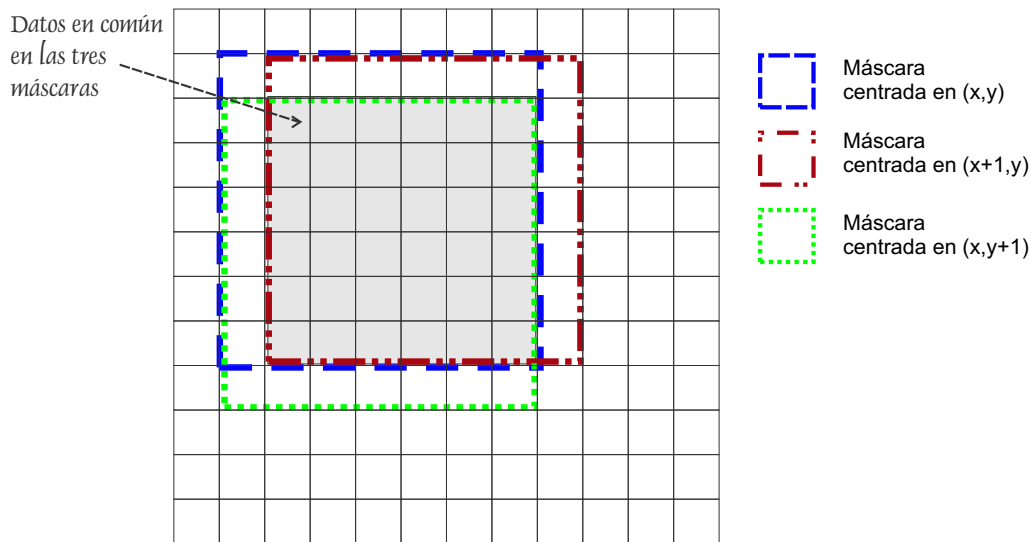


Figura 3.2: Paralelismo a nivel de datos en operaciones de imágenes basadas en ventanas.

las tres ventanas se van procesando, esto implica que, dependiendo en el orden en que se leen los datos, las ventanas acaban su proceso en diferente tiempo. La ventaja de esta manera de procesar las tres ventanas es que no hay necesidad de volver a leer los datos para procesar alguna de estas ventanas, aunque para otras ventanas adyacente sí. Así que hay que contar con un esquema que organice los procesamientos de ventanas que permita explotar el reuso de los datos.

En este caso como ejemplo, se proponen tres máscaras, pero se puede extender a un número mayor de máscaras a procesar de tal manera que los datos que tengan en común sean reutilizados. Por todo lo anterior, se requiere disponer de un diseño conveniente en el que se puedan reusar los datos leídos para el procesamiento de varias ventanas a la vez.

En [14] se propuso un esquema de direccionamiento y transferencia de datos apropiado y, junto con éste, un diseño del procesamiento bajo arquitecturas paralelas para explotar el paralelismo inherente, sin sacrificar eficiencia computacional en operaciones basadas en ventanas.

El procesamiento de las ventanas en paralelo se presenta en dos sentidos: vertical y horizontal. El procesamiento en dirección vertical consiste en procesar ventanas centradas en la misma columna pero en diferente renglón; y el procesamiento en dirección horizontal consiste en procesar ventanas centradas en el mismo renglón pero en diferente

columna. En la Fig. 3.3 se aprecian tres ventanas que se procesan en paralelo tanto en dirección vertical como horizontal.

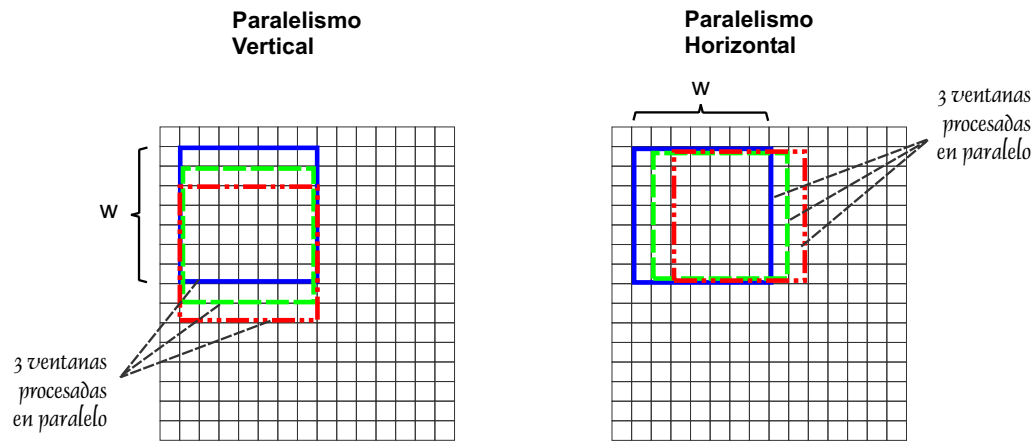


Figura 3.3: Tres ventanas se procesan en paralelo en la dirección vertical, lado izquierdo, y en la dirección horizontal, lado derecho de la figura.

Para aplicar estos dos direccionamientos en el procesamiento completo se debe de partir de la forma en que se acceden a los datos de la imagen de entrada, la cual se ilustra en la Fig. 3.4.

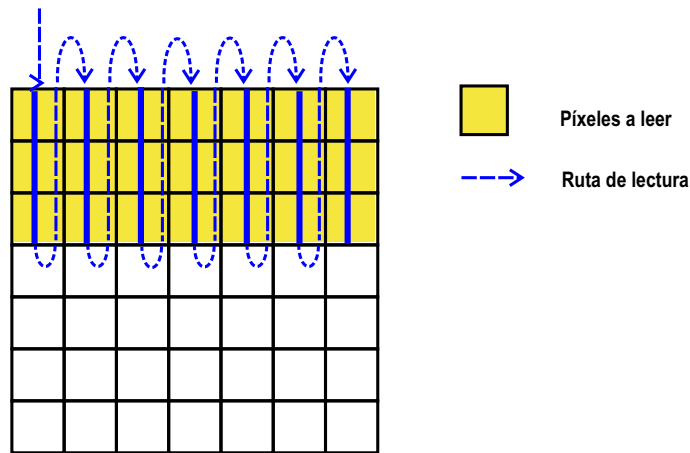


Figura 3.4: Lectura de los píxeles de la imagen.

El paralelismo que se presenta primero es el paralelo en dirección vertical, esto es porque los primeros datos leídos pertenecientes a la primera columna de la imagen son

también los datos que forman parte de las ventanas que se procesarán en paralelo. Una vez que se termine de leer cierto número de datos de la primera columna, inicia la lectura de la segunda columna, activándose nuevas ventanas las cuales se procesan en dirección horizontal. Es así como se presentan ambas direcciones de paralelismo. Esto se aprecia en la Fig. 3.5.

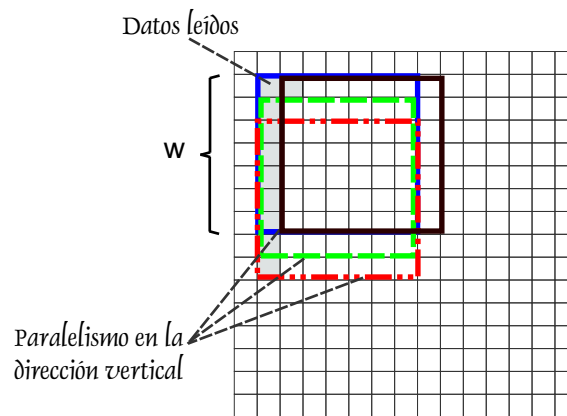


Figura 3.5: Cuatro ventanas se procesan en paralelo, tres ventanas inician su proceso de acuerdo a los datos leídos; una vez leído un dato localizado en otra columna, inicia el procesamiento de ventanas en dirección horizontal.

El número de datos leídos por columna depende del tamaño de la ventana (w) y el número de renglones que se desean procesar en paralelo (NR), es decir, además de leer el número de datos que corresponde al tamaño de la ventana de la máscara, lo cual corresponde sólo a una ventana de la imagen a procesar, es necesario leer adicionalmente más datos que se necesitan para ventanas centradas en otros renglones. Por ejemplo, si se desean procesar dos ventanas en paralelo, es necesario leer el número de datos de acuerdo al tamaño de la máscara. Ya que el primer dato leído de la imagen no es parte de la segunda ventana a procesar en paralelo, entonces, es necesaria la lectura adicional de un dato para así cubrir los datos requeridos por esta segunda ventana. Se debe establecer un compromiso entre los datos traslapados y el número de operaciones de ventanas ejecutadas en paralelo para lograr un buen desempeño.

Este esquema introduce un alto grado de paralelismo incrementando los requerimientos de almacenamiento local interno y rutinas de comunicación. Así que el compromiso entre desempeño y área debe ser elegida de acuerdo a la aplicación.

Hasta este punto, se ha mostrado la convolución sobre una imagen a la que se le aplica una máscara. Ahora se presentará las implicaciones para múltiples máscaras.

3.3. Convolución de una imagen con máscaras múltiples

Se puede pensar que para aplicar varias máscaras sobre una imagen, se puede dedicar tiempo en el procesamiento de la primera máscara, y una vez terminado, iniciar con la siguiente máscara, esto es, un procesamiento secuencial a nivel de las máscaras. Esto implicaría que el tiempo de procesamiento se incrementa k veces, donde k es el número de máscaras a aplicar. Aunque la cantidad en recursos hardware no se alteraría de manera importante, no es un camino conveniente ya que el sistema presentaría un bajo desempeño en el procesamiento completo.

Contrario al planteamiento anterior, se propone un tercer nivel de paralelismo. Esto es, se ha venido hablando de paralelismo en dos direcciones: horizontal y vertical. Ahora se identifica un tercer nivel, paralelismo entre las máscaras.

Aunque los requerimientos en recursos se incrementan, el desempeño se beneficia con los tres niveles de paralelismo. El acceso a los datos y la manera de procesarlos se mantiene igual, pero ahora las señales de control deben distribuirse a todos los arreglos sistólicos 2-D.

En la siguiente sección, se presentan técnicas y arquitecturas adecuadas para modelar el sistema de acuerdo a los requerimientos.

3.4. Técnicas y Arquitecturas de Procesamiento Paralelo

Para el procesamiento de imágenes resulta imperante el contar con un diseño adecuado que aporte al procesamiento un alto rendimiento. El procesamiento de imágenes involucra operaciones que aunque no sean muy complejas, generalmente son muy repetitivas, las cuales llegan a consumir gran parte de los recursos de una computadora convencional.

Para el desarrollo de tales sistemas, se requiere un análisis y diseño de algoritmos cuidadoso para hacer un buen uso del paralelismo inherente que se encuentra inmerso en el procesamiento de imágenes. Por tanto, es necesario comprender y elegir la arquitectura

que mejor se adecúe al paralelismo inherente del sistema además de tomar en cuenta un manejo de datos adecuado. Si se cumplen con estos aspectos importantes se logrará una mejora en el rendimiento de manera significativa, y tal hecho resulta conveniente para varias aplicaciones. Para agilizar la ejecución de dichas tareas, se hace uso del procesamiento paralelo, el cual se basa en utilizar múltiples procesadores para llevar a cabo tales operaciones.

El procesamiento paralelo es un término que se usa para denotar un grupo de técnicas significativas que se usan para realizar tareas simultáneas de procesamiento de datos con el fin de aumentar la velocidad computacional de un sistema de cómputo [18]. En lugar de procesar cada instrucción en forma secuencial como sucede en una computadora convencional, un sistema de procesamiento paralelo puede ejecutar procesamiento concurrente de datos para conseguir un menor tiempo de ejecución. El propósito del procesamiento paralelo es acelerar las posibilidades de procesamiento de la computadora y aumentar su eficiencia, esto es, la capacidad de procesamiento que puede lograrse durante un cierto intervalo de tiempo.

Para lograr arquitecturas paralelas es necesario hacer uso de conceptos como: pipeline, arreglo de procesadores y multiprocesadores [18]. Para el presente trabajo es de interés el diseño de un arreglo de procesadores, con uso de la técnica pipeline. Enseguida se presenta una breve descripción de estos conceptos.

Un arreglo de procesadores consiste de múltiples elementos de procesamiento (*PE*) cuya ejecución es en paralelo y síncrono, bajo la supervisión de una Unidad de Control. La Unidad de Control recupera y decodifica instrucciones de la memoria, y distribuye señales de control a todos los procesadores en el arreglo los cuales ejecutan la misma operación al mismo tiempo [18].

Dependiendo del modo de operación de los elementos de procesamiento y las señales de control recibidas, el arreglo de procesadores puede ser clasificada en: SIMD (Single Instruction Multiple Data - Flujo de instrucción único, Flujo de datos Múltiple), sistólico. En las siguientes secciones se da una breve descripción de estos dos tipos de arreglo de procesadores.

3.4.1. Arquitecturas SIMD

Una arquitectura SIMD consiste de un arreglo de elementos de procesamiento, elementos de memoria (M), una Unidad de Control (CU, por sus siglas en inglés), y una red de interconexión. En una arquitectura SIMD el paralelismo se logra por múltiples unidades de proceso (PEs), cada una de las cuales es capaz de ejecutar una operación especializada autónomamente.

El procesamiento en una arquitectura bajo SIMD consiste en operar la misma secuencia de instrucciones simultáneamente a un conjunto de datos discretos [14]. Las arquitecturas SIMD son usadas en aplicaciones que exhiben cantidades masivas de paralelismo de datos sin un flujo de control complicado o cantidades excesivas de comunicación interprocesador. Los arreglos SIMD son convenientes para procesamiento de imágenes ya que en estos procesos se presenta la misma operación ejecutada sobre el conjunto completo de datos [20].

3.4.2. Arquitectura Sistólica

El concepto de arquitectura sistólica fue propuesto por H. T. Kung y C. E. Leiserson, 1978 [13]. El concepto sistólico denota una clase simple de procesadores concurrentes, en los cuales los datos se mueven de forma regular y periódica similar a un bombeo sistólico del corazón.

Un sistema sistólico consiste de un conjunto de elementos procesadores interconectados, cada uno capaz de ejecutar algunas operaciones simples. La idea básica de un arreglo sistólico es que una vez que el dato esté disponible, éste es usado efectivamente dentro de varios elementos de procesamiento para producir una tasa de desempeño alta [13]. De esta manera, un arreglo sistólico puede explotar el paralelismo inherente de algunas aplicaciones. Dada que la arquitectura sistólica es simple, y tiene una comunicación regular y estructuras de control, presenta ventajas substanciales sobre arquitecturas complicadas en diseño e implementación. Los elementos procesadores en un sistema sistólico son típicamente interconectados de forma lineal, en arreglos de dos o tres dimensiones. En la figura Fig. 3.6 se aprecian algunos arreglos sistólicos típicos, en donde cada cuadrado corresponde a un procesador y cada hilo a una conexión entre los procesadores. Un arreglo sistólico combina la idea de usar numerosos elementos de procesamiento con pipeline,

técnica que se explica en la sección 3.4.3.

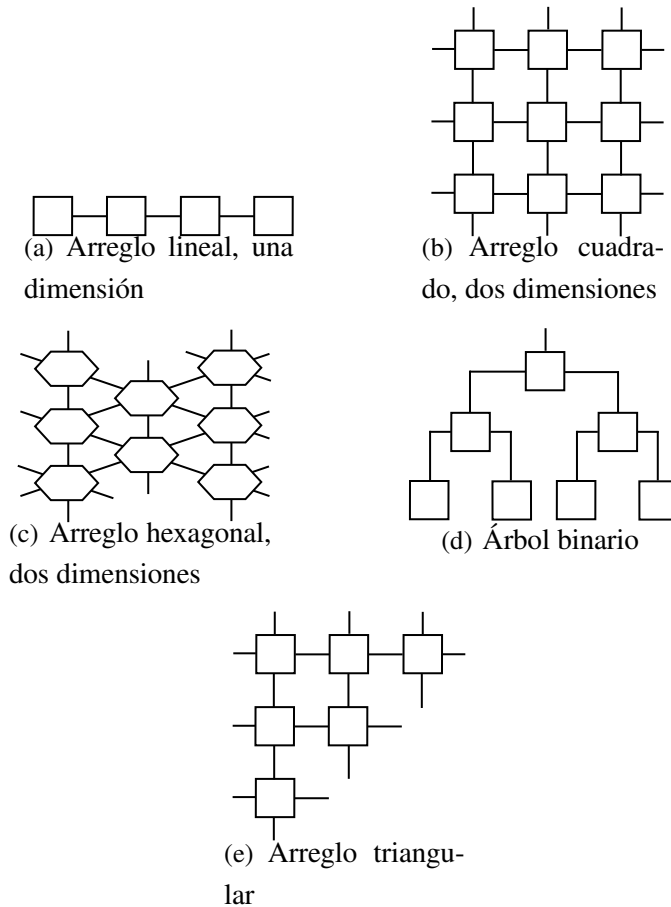


Figura 3.6: Esquema de interconexiones típicas de arreglos sistólico [18].

Bajo una aproximación sistólica, todas las operaciones son ejecutadas de manera síncrona independiente del dato procesado. Sólo el dato de control que es difundido a los elementos procesadores es la señal de reloj, y la comunicación del elemento procesador es local, sólo con sus vecinos más cercanos.

Las ventajas de los sistemas sistólicos son:

- Reuso de datos en un tiempo dado en diferentes elementos procesadores.
- Alto desempeño computacional con un ancho de banda de memoria modesto.
- Uso extenso de concurrencia.

- Algunos tiempos de elementos procesadores simples existentes.
- Flujo de datos y control son simples y regulares.
- Los diseños pueden ser modulares y escalables.

3.4.3. Pipeline

La técnica pipeline ha sido vista como una técnica de arquitectura para incrementar el desempeño de una aplicación. Esta técnica consiste en descomponer un proceso secuencial en suboperaciones, y cada subproceso se ejecuta en un segmento dedicado especial que opera en forma concurrente con los otros segmentos [18]. Cada segmento ejecuta un procesamiento parcial, dictado por la manera en que se divide el proceso completo. El resultado obtenido del cálculo en cada segmento se transfiere al siguiente segmento que corresponde para completar el proceso. El resultado final se obtiene después de que los datos han recorrido todos los segmentos.

Para un sistema pipeline, la salida actual es detenida algunos ciclos de reloj dependiendo de los estados (segmentos) introducidos dentro del sistema pipeline. En general, asumiendo que cada estado tiene el mismo retardo, el rendimiento de un sistema con pipeline con n estados es n veces más eficiente que el sistema sin pipeline, mientras su salida de latencia se incrementa mediante el retardo computacional asociado con los n estados almacenados.

Existen varias razones por las que una arquitectura paralela no puede operar a su máxima velocidad teórica. Los diferentes segmentos pueden requerir tiempos diferentes para completar su suboperación. Debe elegirse el ciclo de reloj para que iguale el tiempo de retraso del segmento con el máximo tiempo de propagación. Esto provoca que los otros segmentos desperdicien tiempo mientras esperan el siguiente ciclo de reloj. La técnica pipeline proporciona una operación más rápida que una secuencia puramente serial, aunque nunca se logra por completo la máxima velocidad teórica.

Varias operaciones de procesamiento de imágenes, particularmente las basadas en la aplicación de filtros o máscaras, son convenientes implementarlas bajo un esquema pipeline, ya que las funciones deben ser ejecutadas a lo largo de toda la cadena de datos; además el resultado, el cual es calculado en cada punto, es una función de datos de un

vecino limitado de píxeles.

La arquitectura propuesta es una arquitectura sistólica con uso de la técnica pipeline, la cual se explicará con mayor detalle en el capítulo 4.

3.5. Métodos y Herramientas

Los FPGAs (Field Programmable Gate Arrays) son dispositivos configurables que están constituidos por celdas distribuidas en su superficie; pueden ser programados o configurados por medio de las miles de compuertas que generalmente lo componen. Las tecnologías configurables tienen la ventaja de contar con un ciclo de diseño muy rápido y flexible, además de poder configurarse muchas veces.

Los avances en el diseño de los FPGAs han permitido mantener las ventajas del diseño de un ASIC (Application Specific Integrated Circuit), evitando los altos costos de desarrollo, la incapacidad de efectuar modificaciones posteriores al diseño, ofreciendo un bajo riesgo y desarrollo de prototipos rápidos.

Sin embargo, el uso de los FPGAs tiene sus desventajas, en los casos que se requieran producciones a gran escala de un diseño, dado que pueden resultar más costosos, además de requerir de un mayor consumo de potencia respecto a un ASIC. No obstante, dichas desventajas tienen que ser sopesadas con todas las ventajas que ofrece el uso de los FPGAs, sobre todo en la etapa de diseño.

Algunas de las características que hacen atractivo el diseño de algoritmos en los FPGAs, es que cuentan con estructuras que permiten paralelizar operaciones e introducir etapas de pipeline. De esta forma, una arquitectura desarrollada en un FPGA trabajará para un algoritmo en específico. Además, dado que los FPGAs permiten diseñar múltiples elementos procesadores a un bajo costo y de una forma más rápida, éstos son una opción muy atractiva en la implementación de arquitecturas de diseño paralelo planteadas anteriormente.

Dado que el procesamiento a tratar presenta una gran densidad computacional, identificándose procesos paralelos masivos y repetitivos, resulta conveniente la construcción de un diseño bajo tecnología FPGA.

Capítulo 4

Arquitectura Sistólica 3-D para Filtrado en Múltiples Orientaciones

En esta sección se muestra la arquitectura propuesta para el filtrado de imágenes en múltiples orientaciones que imita, de manera aproximada, la estructura y funcionamiento de las neuronas en la corteza visual primaria. En el presente capítulo se expone la organización conceptual del modelo biológico en analogía con la arquitectura sistólica 3-D, éste último consiste de un arreglo sistólico de procesadores con uso de la técnica pipeline, para aplicar múltiples filtros los cuales realizan una detección de bordes de acuerdo a una orientación. Cada uno de los módulos que componen la arquitectura se explican en secciones subsecuentes.

4.1. Analogía entre el Sistema Biológico y la Arquitectura Sistólica 3-D

La propuesta general de la arquitectura se basa en agrupar en un arreglo bidimensional aquellas neuronas (conjunto de neuronas) que realizan el cálculo en la misma orientación. El diseño se basa en lo expuesto en el trabajo de Shimonomura [1] en el que se agrupan las neuronas de la misma orientación en un solo chip. Con esto se logra que cada arreglo de neuronas se dedique exclusivamente al cálculo del filtro en una orientación específica sobre una imagen dada, teniendo tantos arreglos como número de orientaciones deseadas.

En la Fig. 4.1 se ilustra la analogía entre el modelo biológico de la corteza visual primaria agrupando los neuronas de la misma orientación en arreglos bidimensionales, y el diagrama general de la arquitectura con dicho comportamiento. En la parte izquierda de la figura, la cual corresponde al modelo biológico, se observa un conjunto de arreglos bidimensionales de neuronas columnas las cuales son filtros aplicados sobre una imagen de entrada teniendo un conjunto de imágenes de salida. Esto es, el arreglo bidimensional de columnas cuya orientación es en 0° aplica el filtro Gabor-2D, teniendo $\theta = 0^\circ$, dando como resultado una imagen de salida la cual resalta los bordes hallados a 0° en la imagen de entrada. Por tanto, se tiene el mismo número de imágenes de salida como filtros Gabor-2D aplicados.

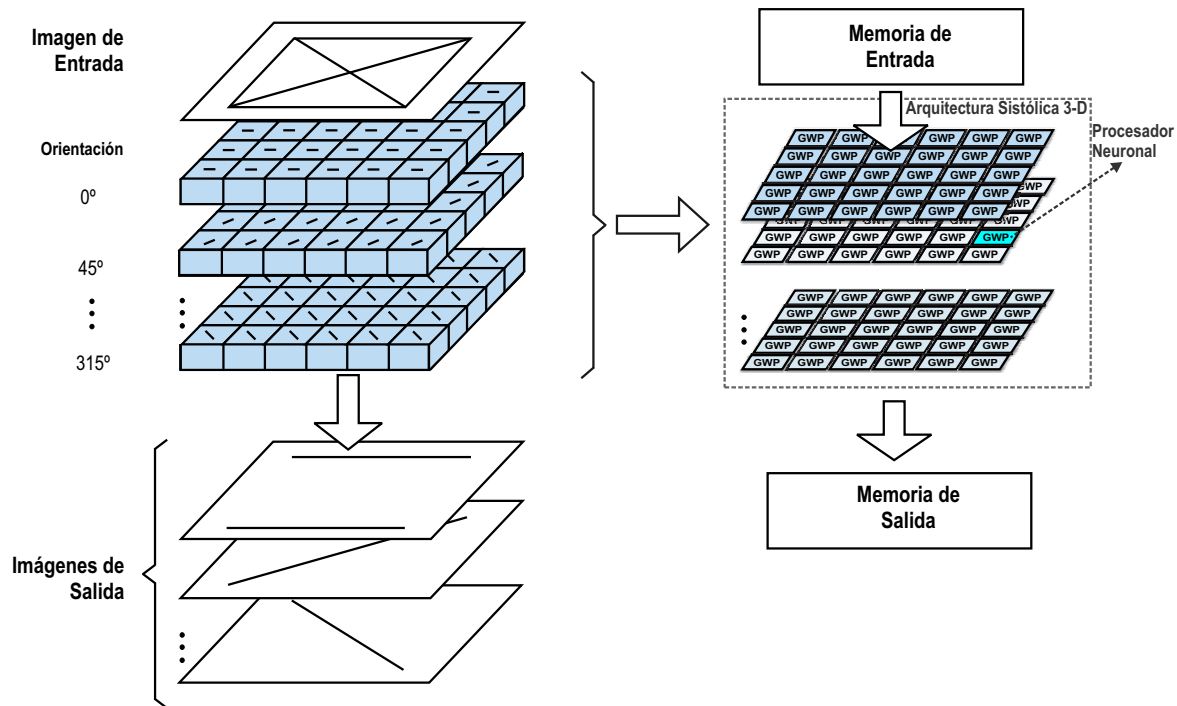


Figura 4.1: Analogía correspondiente al modelo biológico de la corteza visual primaria, visto como la agrupación de neuronas de la misma orientación y un diagrama conceptual simplificado de la arquitectura con dicha organización.

La Figura 4.1 ilustra, de manera simplificada, la aplicación de 8 filtros cuyas orientaciones van ascendiendo en 45° sobre una imagen de entrada. Estas orientaciones correspon-

den a 0° , 45° , 90° , 135° , 180° , 225° , 270° y 315° . En la parte derecha de la imagen, se ilustra de manera simplificada, un diagrama a bloques de la arquitectura hardware que se propone. Este modelo corresponde en agrupar elementos de procesamiento (etiquetados como GWP - Gabor Window Processor) en arreglos bidimensionales. Cada arreglo bidimensional corresponde a calcular el filtro Gabor-2D en una orientación específica sobre la imagen de entrada. De acuerdo a las 8 orientaciones deseadas, se cuenta con ocho arreglos bidimensionales sintonizados a una orientación específica. Los datos de entrada necesarios para realizar este procesamiento se encuentran en bancos de memoria de entrada, y las imágenes resultantes de aplicar el filtro se almacena en la memoria de salida.

En la Fig.4.2 se muestra un diagrama a bloques de los componentes que forman parte de la arquitectura sistólica 3-D. En la parte central de la figura, se encuentra el arreglo sistólico 3-D. Este arreglo se encuentra compuesto por varios arreglos sistólicos 2-D en el que cada uno de ellos corresponde al cálculo de un filtro Gabor-2D en una orientación específica. a continuación se da una breve presentación de los módulos que componen la arquitectura, y en secciones subsecuentes se presentan los detalles de la formación y funcionamiento de la arquitectura sistólica 3-D.

La arquitectura cuenta con cuatro unidades funcionales:

- Unidad de Control
- Unidad Generador de Direcciones
- Arreglo Sistólico 3-D
- Colector de Datos

El propósito principal de la *Unidad de Control* es manejar del flujo de datos y sincronizar las diferentes operaciones realizadas en la arquitectura. Esta unidad sincroniza las operaciones de los módulos y el intercambio de datos entre los procesadores en el arreglo. Además, inicia las tareas dentro del arreglo de procesadores, coordina operaciones y maneja el flujo de datos bidireccional entre la arquitectura y los bancos de memoria. Cada una de las señales producidas por esta unidad es conectada a cada uno de los arreglos sistólicos 2-D.

El propósito principal de la *Unidad Generador de Direcciones* es generar las direcciones para obtener los datos de la imagen en la memoria de entrada así como los datos

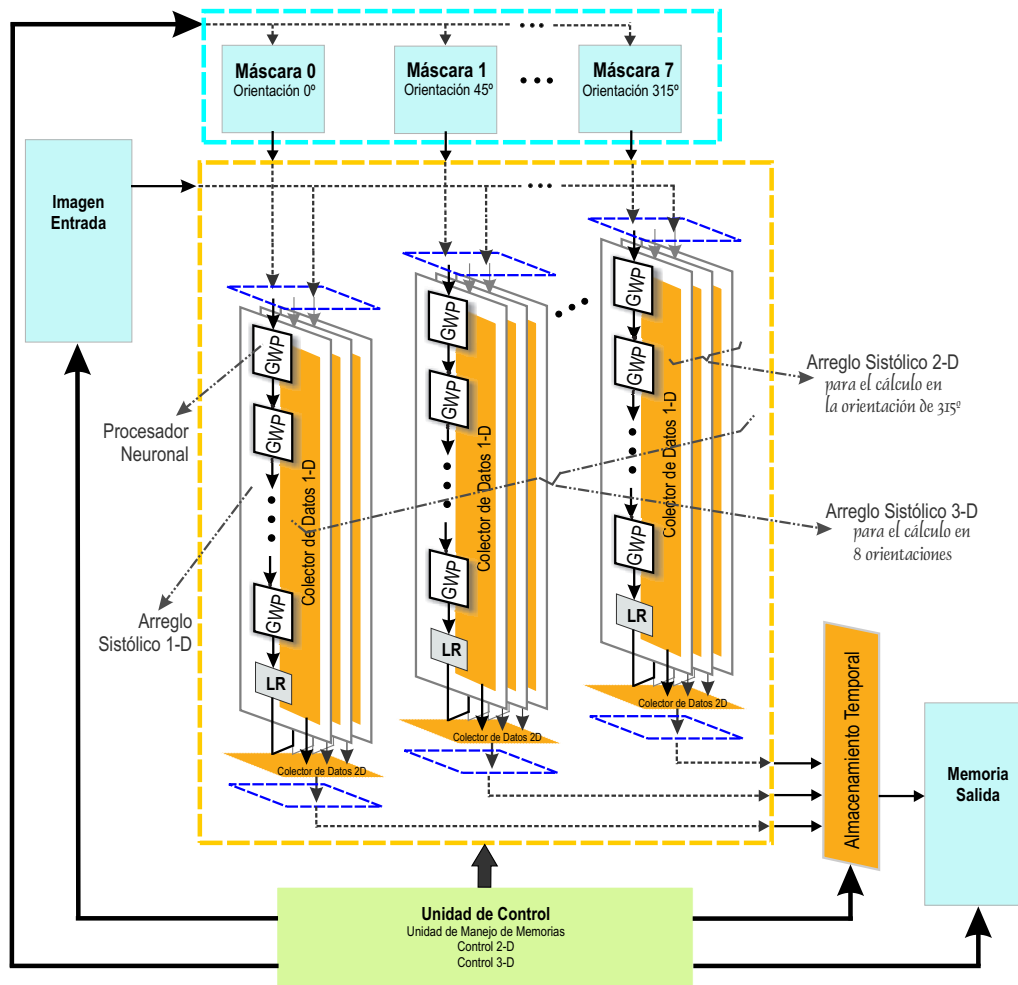


Figura 4.2: Diagrama a bloques de la arquitectura sistólica 3-D que aplica varios filtros en orientaciones específicas sobre una imagen de entrada.

de las máscaras, de acuerdo a un esquema de lectura de memoria predefinida, expuesta en la sección 3.2. Tanto la memoria de la imagen como la de los coeficientes de las máscaras son leídas bajo una lectura basada en columnas como se propone en [15]. Así también, genera las direcciones de la memoria de salida para almacenar los resultados producidos por la arquitectura.

El *Arreglo Sistólico 3-D* se encuentra compuesto por arreglos sistólicos 2-D, los cuales son arreglos bidimensionales de procesadores. Cada arreglo sistólico 2-D realiza el cálculo del filtro Gabor-2D en una orientación específica sobre una imagen de entrada. Cada procesador es responsable de realizar el cálculo del filtro sobre una ventana específica de la imagen. Para obtener la formación del arreglo sistólico 3-D, se describe una organización de procesadores en tres niveles. El primer nivel es un arreglo sistólico 1-D de procesadores que realiza el cálculo, en forma paralela, de varias ventanas centradas en diferentes renglones de la misma columna, paralelismo en dirección vertical. Partiendo del arreglo sistólico 1-D, se extiende el paralelismo en la dirección horizontal, es decir, realizar el procesamiento de varias ventanas centradas en diferentes renglones y columnas. Con ello se obtiene un arreglo sistólico 2-D. El tercer nivel de organización se encuentra extendiendo el arreglo sistólico 2-D a un arreglo sistólico 3-D, es decir, se tienen tantos arreglos sistólicos 2-D como número de orientaciones a calcular. Con ellos se busca calcular, en forma paralela, varias ventanas centradas en diferentes renglones y columnas siendo calculadas para diferentes orientaciones.

El módulo *Colector de Datos* es un arreglo de registros que colecta los resultados producidos en los diferentes niveles de organización de los procesadores. Esto es, el *Colector de Datos 1-D* colecta los datos producidos por cada uno de los procesadores que forman el arreglo sistólico 1-D; el *Colector de Datos 2-D* colecta los datos producidos por cada uno de los arreglos sistólicos 1-D los cuales forman el arreglo sistólico 2-D. El número de Colectores de Datos 2-D depende del número de arreglos sistólicos 2-D existentes. Ya que se obtienen datos de salida al mismo tiempo por parte de los arreglos sistólicos 2-D, se cuenta con un arreglo de registros, *Almacenamiento Temporal*, cuya función es la gestión de dichos datos para que sean almacenados en la memoria de salida.

El flujo de datos inicia con la lectura de los bancos de memoria donde se encuentran almacenados los píxeles de la imagen y coeficientes de la máscara. Los píxeles de la imagen son leídos en columnas y son transmitidos a cada uno de los arreglos sistólicos

2-D, que a su vez los transmiten a cada procesador los cuales son los elementos básicos de la arquitectura. El tamaño de la columna leída depende del número de ventanas a procesar en paralelo, de esta manera se usará el píxel leído tanto como sea posible, esto es, el píxel leído será usado por los procesadores que lo requieran para procesar la ventana de la imagen de la que son responsables. En un arreglo sistólico 2D, el resultado que se obtiene de cada procesador es capturado incrementalmente por un colector de datos. Una vez que se habilitan las señales de control para almacenar el resultado, éste se toma del colector de datos para ser almacenado en un banco de memoria de salida. El proceso anterior se aplica para cada uno de los arreglos sistólicos 2-D, es decir, para cada orientación deseada. Por tanto, por cada ciclo de reloj en el periodo de recolección de resultados, se reciben ocho datos de salida que son el resultado del cálculo del filtro Gabor-2D en ocho orientaciones aplicadas a la misma ventana de la imagen. Por lo anterior, se hace uso de un módulo adicional, *Almacenamiento Temporal*, cuyo objetivo es la recolección de datos de salida que provienen de los *Colectores de Datos 2-D*, además de gestionar el almacenamiento de los mismos en los bancos de memoria sin dar lugar a la pérdida de dichos datos. Con este diseño se logra que cada píxel de la imagen, además de ser usado en paralelo dentro de cada arreglo sistólico 2-D, sea usado para cada uno de los arreglos 2-D, lo cual reduce los accesos a memoria y fortalece la implementación de paralelismo en el diseño.

La forma detallada en cómo se van formando los arreglos sistólicos se explica en las siguientes secciones, mostrando en cada paso la analogía con el modelo biológico. Se parte de la unidad elemental de procesamiento hasta obtener un arreglo sistólico 3-D.

4.2. Unidad Elemental de Procesamiento

Como se ha mencionado, los elementos básicos de la arquitectura son los elementos de procesamiento, cuya tarea es la de aplicar el filtro Gabor-2D en una orientación específica mediante el procesamiento basado en ventanas. En la Fig.4.3 se presenta la analogía de la unidad elemental de procesamiento en el modelo biológico de la corteza visual primaria organizada en neuronas y en la arquitectura organizada en *GWPs*. En la parte izquierda de la figura se resalta la actuación de un conjunto de neuronas las cuales componen una neurona sobre una imagen de entrada. Esta neurona actúa sobre un área específica de la

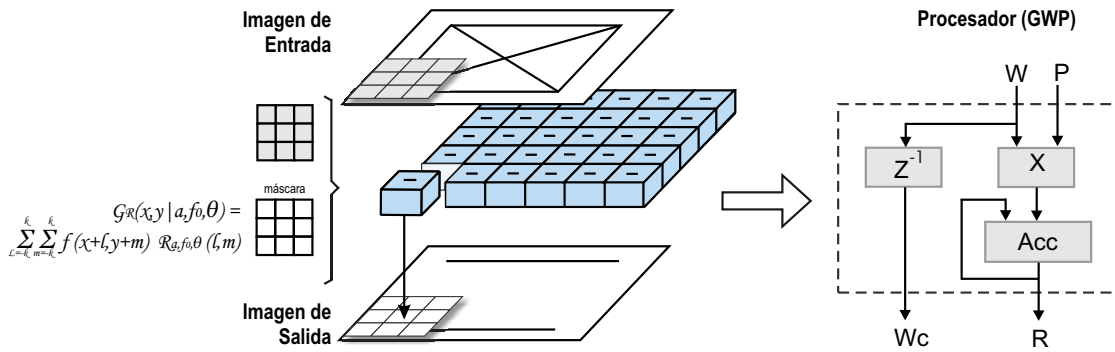


Figura 4.3: Analogía de la acción de una neurona sobre un área específica de la imagen de entrada y el funcionamiento del procesador etiquetado como GWP en la arquitectura.

imagen, aplicando el filtro de Gabor-2D cuya orientación es de 0° . El resultado de este cálculo forma parte de la imagen que se tiene de salida.

El elemento de procesamiento, etiquetado en la arquitectura como GWP , es responsable de calcular el filtro Gabor-2D en una orientación específica sobre una ventana de la imagen de entrada. Una vez que se obtiene el dato de resultado, éste se ubica en las mismas coordenadas del punto en el que está centrada la ventana de la imagen procesada. Al final de aplicar el filtro sobre toda la imagen de entrada se obtiene una imagen como salida.

En la parte derecha de la Fig.4.3 se muestra un diagrama simplificado de un procesador, el cual está diseñado para calcular la operación de convolución sobre la imagen de entrada. El GWP está compuesto de un registro (Z^{-1}), un multiplicador (X), un acumulador (Acc), además de recibir dos datos de entrada, el píxel de la imagen (P) y el coeficiente de la máscara (W). El multiplicador realiza la multiplicación de los datos de entrada, P y W , el cual se acumula al valor parcial almacenado, Acc , dando como salida dicho resultado en R . El registro Z^{-1} es utilizado para almacenar temporalmente el valor W para después enviarlo a su GWP vecino o a la línea de retardo, mostrado en la figura como Wc . La línea de retardo es requerida para poder estar en sincronía con el arreglo sistólico 1-D vecino.

4.3. Organización en Arreglos Sistólicos

Cuando se recibe un estímulo en la corteza visual primaria, todas las neuronas simples actúan en forma paralela sobre la imagen de acuerdo a su ubicación. Resulta complejo llevar este comportamiento a un diseño arquitectural, ya que se tienen limitantes para procesar toda la imagen por los elementos de procesamiento. Ya que no es posible contar con el mismo número de procesadores, como píxeles de la imagen de entrada, aplicando el filtro en forma paralela, es necesario formar arreglos que estén organizados de tal manera que permita contar con procesos en paralelo para acelerar el procesamiento de toda la imagen.

4.3.1. Arreglo Sistólico 1-D

De acuerdo al modelo neuronal en el que se basa la arquitectura hardware propuesta, en la parte izquierda de la Fig.4.4 se presenta una vista sobre el funcionamiento de tres neuronas de la corteza visual primaria aplicadas sobre un área específica de la imagen, teniendo como resultado tres datos los cuales forman parte de la imagen de salida. La contraparte a este modelo, es un arreglo sistólico 1-D, el cual se explica con mayor detalle en esta sección.

Un arreglo sistólico 1-D consiste de un arreglo lineal de $GWPs$, en la Figura 4.4 dispuestos de manera vertical, de tal forma que se realice el cálculo paralelo de varias ventanas centradas en diferentes renglones de la misma columna.

El arreglo sistólico 1-D lo componen los siguientes módulos:

- Un conjunto de $GWPs$
- Una línea de retardo (LR en la figura)
- Un *Colector de Datos 1-D*

Cada arreglo $GWPs$ procesa una ventana de la imagen de entrada. Los $GWPs$ que forman parte del arreglo lineal son los procesadores que operan sobre ventanas de la imagen que se encuentran ubicadas sobre la misma columna pero centradas en diferentes renglones. El número de estos procesadores depende del número de ventanas que se procesan en paralelo. Con ello se logra un primer nivel de paralelismo, en dirección vertical.

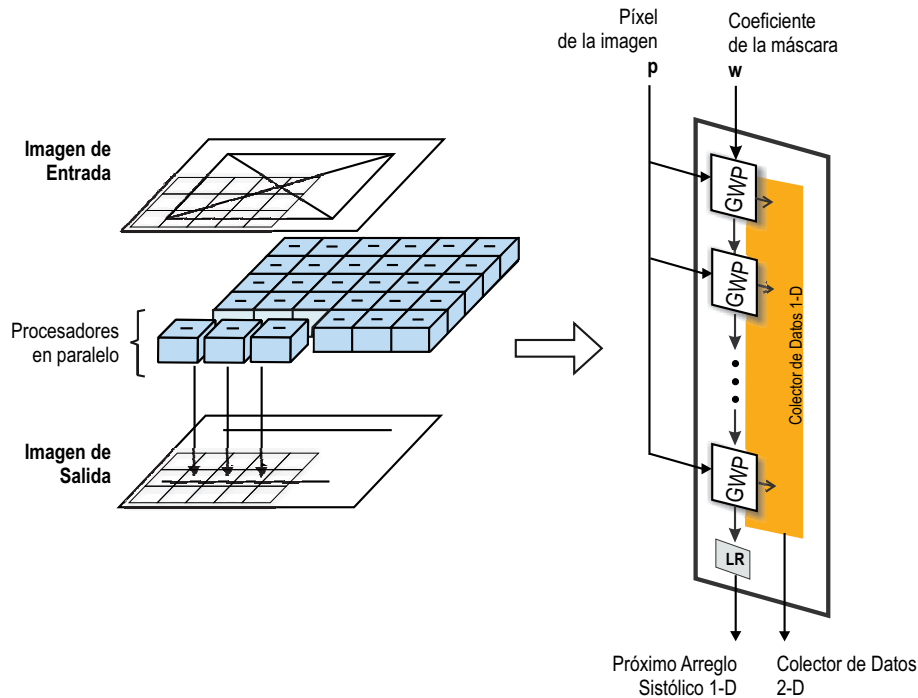


Figura 4.4: Analogía del modelo biológico de la corteza visual primaria actuando tres neuronas alineadas, y la arquitectura de un arreglo sistólico 1-D. La parte izquierda ilustra tres neuronas organizadas en forma lineal con orientación a 0° aplicadas sobre un área específica de la imagen de entrada. La parte derecha de la figura muestra un arreglo sistólico 1-D generalizado compuesto por GWP s.

Cada GWP recibe como entrada el píxel de la imagen (P), y el coeficiente de la máscara (W); ambos en cada ciclo de reloj. Para todos los GWP s se recibe el mismo píxel de la imagen, no así el mismo coeficiente de la máscara. Los GWP s en el arreglo sistólico 1-D se encuentran en diferentes etapas de procesamiento el cual viene dado por el dato de coeficiente de la máscara que se recibe. La habilitación de los GWP s es en forma gradual, activándose de la siguiente manera. El primer GWP del arreglo es el primero en iniciar su actividad, ya que el píxel de la imagen de entrada que se recibe forma parte de la ventana que le corresponde procesar. En el siguiente ciclo de reloj, el segundo GWP inicia su actividad. Para ese momento, dos GWP s están en actividad, el primero de ellos se encuentra procesando su segundo par de datos, y el segundo procesa su primer par de datos. Como se mencionó, los procesadores no reciben el mismo coeficiente de la máscara-

ra, y es precisamente porque no se encuentran en la misma etapa de procesamiento, esto es, el primero *GWP* necesita el segundo coeficiente de la máscara, mientras el segundo *GWP* necesita el primero. Es por esto que existe comunicación entre los procesadores siendo ésta en una dirección. Todo *GWP* retiene por un ciclo de reloj el coeficiente de la máscara que ha recibido, el cual es entregado a su *GWP* vecino. El primer *GWP* en iniciar su actividad recibe el coeficiente de la máscara directamente del banco de memoria, mientras que su vecino, que inicia su actividad en el siguiente ciclo de reloj, recibe los coeficientes de la máscara del primer *GWP*. Al iniciar actividad en el arreglo sistólico 1-D, los *GWP* se van activando y desactivando progresivamente, siendo gestionado por la unidad de control.

La línea de retardo (*LR*) es requerida en los límites del arreglo para retener por algunos ciclos de reloj, número de renglones procesados en paralelo menos 1 ($NR - 1$), los valores de los coeficientes de la máscara que va entregando el último *GWP*. Al iniciar la actividad de otro arreglo sistólico 1-D, éste último recibe los valores de la máscara del arreglo sistólico 1-D inmediato anterior. Para retener los valores de los coeficientes de la máscara es necesario contar con un arreglo de registros; el tamaño de éste depende del número de renglones que se están procesando en paralelo menos 1 ($NR - 1$). La línea de retardo está diseñada para fines de sincronía.

El Colector de Datos 1-D colecta datos del arreglo de *GWP*s. El primer dato en coleccionar es del *GWP* que se encuentra más arriba del arreglo sistólico 1-D, ya que fue el primer *GWP* en iniciar actividades y el primero en entregar un resultado. El segundo dato a coleccionar es del segundo más arriba *GWP* del arreglo, es decir, el segundo en iniciar actividades en el arreglo; y así sucesivamente se van coleccionando los resultados. En otras palabras, una vez que el periodo de latencia ha transcurrido, los *GWP*s entregan su resultado progresivamente, y así mismo el colector de datos recoge el resultado, siendo gestionado por la unidad de control de la arquitectura. El colector de datos entrega el dato capturado al *Colector de Datos 2-D* con el fin último de ser almacenado en el banco de memoria de salida. El progreso de colección de datos se aprecia en la Fig.4.5.

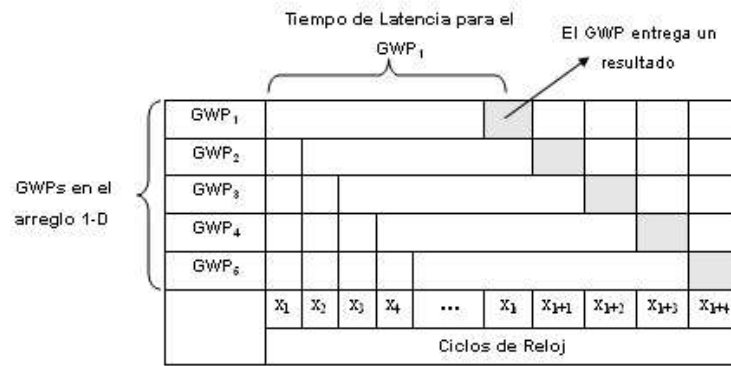


Figura 4.5: Actividad de los GWP de un arreglo sistólico 1-D.

4.3.2. Arreglo Sistólico 2-D

Como se ha mencionado, la corteza visual primaria está formada por grupo de neuronas las cuales detectan ciertas características de acuerdo a una orientación. De tal manera que la organización de las neuronas las visualizamos como un arreglo de neuronas ubicadas en diferente área para la detección de la misma orientación en un espacio de la imagen en específico.

Como se observa en la Fig.4.6 en su parte izquierda, nueve neuronas están actuando en paralelo en un espacio específico de la imagen de entrada. Esto es análogo a decir que, de acuerdo a la arquitectura hardware propuesta, un arreglo de tres arreglos sistólicos 1-D actúan sobre la imagen de entrada, teniendo cada uno de ellos tres procesadores que lo componen. En la parte izquierda de la figura se ilustra esta analogía expresada en forma generalizada, siendo un arreglo de arreglos sistólicos 1-D, formando un arreglo sistólico 2-D.

Para construir el arreglo sistólico 2-D, se parte de un arreglo sistólico 1-D, haciendo una extensión de éste último. Esta extensión permite el procesamiento horizontal, es decir, realizar el procesamiento de varias ventanas centradas en diferentes renglones y, ahora, diferentes columnas. El procesamiento a nivel vertical está dado por el arreglo sistólico 1-D.

El arreglo sistólico 2-D está basado en la interconexión de varios arreglos sistólicos 1-D. Entre estos arreglos existe una conexión en cascada dado que el primer arreglo sistólico

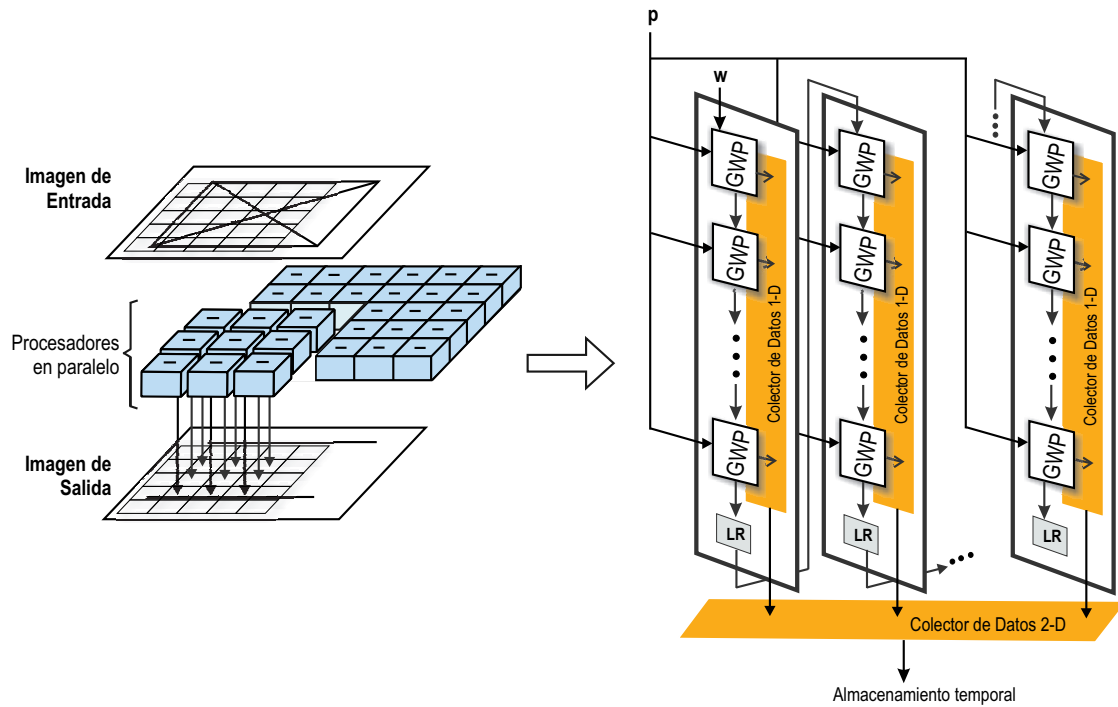


Figura 4.6: Analogía entre el modelo biológico de la corteza visual primaria en el que actúan nueve neuronas ubicadas en diferentes columnas y renglones con orientación a 0° aplicada sobre una imagen de entrada, colocando el resultado en el área que les corresponde formando la imagen de salida; teniendo su contraparte en un arreglo sistólico 2-D, el cual corresponden a un arreglo bidimensional de GWP_s , entregando el resultado al colector de datos 2-D.

1-D alimenta de datos (coeficientes de la máscara) al segundo arreglo sistólico 1-D, y así sucesivamente. Mientras que el píxel de la imagen de entrada (P) es el mismo para todos los arreglos sistólicos 1-D, y a su vez, para todos los GWP_s .

Cuando se inicia la lectura de los píxeles de la imagen, los datos leídos de la primera columna forman parte de las ventanas a procesar por parte del primer arreglo sistólico 1-D habilitado. Una vez que se inicia la lectura de la segunda columna de píxeles, éstos datos forman parte de las ventanas a procesar tanto del arreglo sistólico 1-D iniciado con anterioridad como por un nuevo arreglo sistólico 1-D. El número de arreglos sistólicos 1-D depende del número de columnas que se deseen procesar en paralelo. Se ha determinado que el número más conveniente de columnas a ejecutar en paralelo sea el que corresponda al tamaño de la máscara, de acuerdo a los resultados obtenidos en [14]. Con ello se logra

que, cuando el último arreglo sistólico 1-D se ha habilitado, el primer arreglo 1-D empieza a entregar resultados, teniendo todos los arreglos sistólicos 1-D habilitados y sin periodos muy largos de inactividad. Una vez que un *GWP* entrega su resultado, éste inicia sus valores en cero para estar listo en procesar una nueva ventana centrada en la siguiente columna a leer de la imagen de entrada.

El flujo de datos en el arreglo sistólico 2-D es de izquierda a derecha, y del arreglo sistólico 1-D funciona incrementalmente sobre un esquema sistólico pipeline.

El *Colector de Datos 2-D* colecta los resultados de los arreglos sistólicos 1-D. Dado que el número de arreglos sistólicos 1-D empleados en la arquitectura es igual al tamaño de la máscara y que la entrega de los resultados se presenta en forma incremental, *el Colector de Datos 2-D* es implementado como un multiplexor dedicando cierto tiempo a la captura de los resultados de cada uno de los arreglos sistólicos 1-D, siendo este tiempo dependiente del número de *GWPs* que forman un arreglo sistólico 1-D.

4.3.3. Arreglo Sistólico 3-D

Hasta este momento, se ha propuesto una arquitectura hardware con un comportamiento aproximado al de las neuronas simples selectivas en una sola orientación. La velocidad de respuesta de estas neuronas se aproxima diseñando el sistema bajo una arquitectura sistólica con técnica pipeline. Una vez construido el arreglo sistólico 2-D, se extiende a un arreglo sistólico 3-D, y con ello lograr el filtrado en múltiples orientaciones.

Para la arquitectura sistólica propuesta, el arreglo sistólico 2-D cumple el rol de un chip en analogía con el trabajo presentado por [1], en el que se detectan características de la imagen en una orientación específica, así su extensión a 3-D proporciona la detección en varias orientaciones deseadas. En la Fig.4.7 se presenta la analogía entre el modelo biológico de la corteza visual primaria teniendo su contraparte en el arreglo sistólico 3-D.

Para modelar la organización y comportamiento del modelo neuronal de V1 se han agrupado las neuronas de una orientación específica en un arreglo bidimensional. Para obtener el comportamiento de una *hipercolumna*, es necesario contar con varios arreglos bidimensionales de neuronas, donde cada uno de estos arreglos realice la selectividad en una orientación específica. Para este trabajo concretamente, son ocho arreglos bidimensionales de neuronas para contar con la detección de bordes en ocho orientaciones. En la

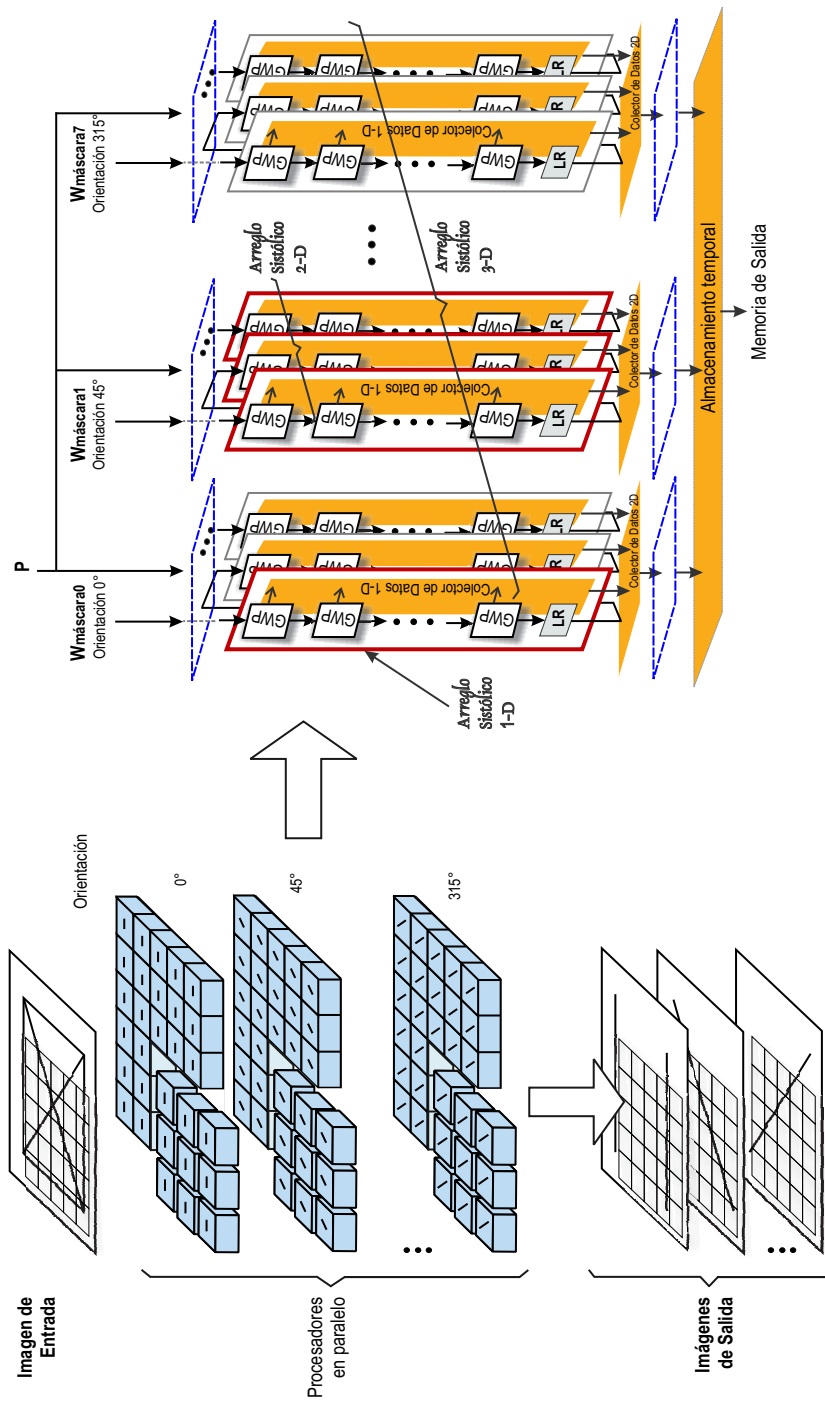


Figura 4.7: Analogía entre el modelo biológico de la corteza visual primaria agrupando las neuronas en arreglos bidimensionales siendo su contraparte la extensión de la arquitectura sistólica 2-D a 3-D.

parte izquierda de la Fig.4.7 se muestra esta propuesta de organización para la aplicación del filtro Gabor-2D en 8 orientaciones. La analogía a este comportamiento se ilustra en la parte derecha de la figura, siendo la extensión del arreglo sistólico 2-D a un arreglo sistólico 3-D de *GWPs*. Los arreglos sistólicos 2-D reciben diferente máscara de Gabor-2D ya que cada arreglo realizará la detección de bordes en una orientación específica.

4.4. Descripción Funcional

En esta sección se presenta una descripción funcional de la arquitectura propuesta partiendo del caso de tener una imagen de entrada a la que se le aplicarán dos filtros, como se ilustra en la Fig.4.8. Esta explicación está principalmente enfocada en el flujo de datos y cálculo realizado por los procesadores.

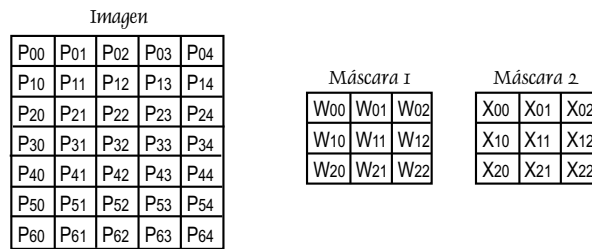


Figura 4.8: Imagen con dimensiones 7×5 a la que se le aplicará dos filtros cuyas máscaras son de dimensiones 3×3 .

Los datos que recibe cada procesador son de dos fuentes, el píxel de la imagen de entrada y el coeficiente de la máscara de Gabor-2D en una orientación específica. Esta lectura de datos implica varios accesos a los bancos de memoria. En la arquitectura propuesta se hace uso de un dato, para varias ventanas procesadas en paralelo, almacenado en registros, y con ello reducir los accesos a dichos datos.

En la Fig.4.9 se muestra un arreglo sistólico 1-D el cual se encuentra compuesto por tres procesadores, cuya máscara a aplicar es *Máscara 1* mostrada en la Fig.4.8. El número de procesadores corresponde al número de ventanas que se van a procesar en paralelo en la dirección vertical. La ventana de la imagen de entrada que se encuentra al lado de cada procesador es la ventana de dimensiones 3×3 que le corresponde procesar, ésta se ha

adicionado para mostrar la etapa de procesamiento en la que se encuentra cada procesador.

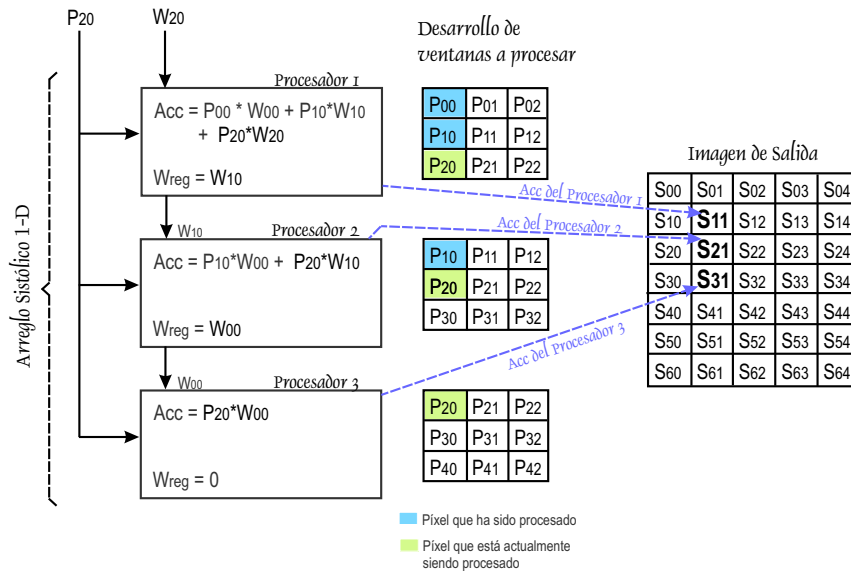


Figura 4.9: Diagrama general del funcionamiento de un arreglo sistólico 1-D compuesto por tres procesadores, cuyas ventanas a procesar se encuentran al lado de cada procesador, y una vez procesadas se almacena el dato resultante en la imagen de salida.

El flujo de datos inicia desde la primera lectura del píxel de la imagen, P_{00} , y el primer coeficiente de las máscaras, W_{00} . Enseguida estos datos son distribuidos a los elementos de procesamiento. En el caso del primer píxel de la imagen, éste es usado por sólo un elemento de procesamiento el cual inicia su actividad. En la siguiente lectura de los bancos de memoria de entrada se inicia actividad de un segundo procesador. El dato correspondiente al píxel de la imagen es usado en ambos procesadores, pero el primer procesador realiza el segundo cálculo parcial y el segundo procesador inicia su actividad con su primer cálculo, y ésto se logra mediante el paso de los coeficientes la máscara. La actividad de los procesadores se inicia de manera gradual, como se ha descrito. Para el caso ilustrado en la Fig.4.9, el pipeline se encuentra compuesto por nueve etapas correspondientes a los nueve píxeles que procesa por ventana; el *Procesador 1* se encuentra en la tercera etapa, el *Procesador 2* en su segunda etapa, y el *Procesador 3* ha iniciado su actividad.

El píxel P_{20} de la imagen de entrada es usado para los tres procesadores, dado que es un valor de píxel que les corresponde procesar. Con esto se logra que ese dato sea usado

en más de un procesador en un mismo ciclo de reloj, y así evitar su lectura posterior a menos que corresponda a un píxel que se necesite para el cálculo de otra(s) ventana(s) que en ese momento no se está procesando.

En cada procesador se encuentra un registro, W_{reg} , en el que se retiene, por un ciclo de reloj, el coeficiente de la máscara recibido. Una vez que el procesador realiza la operación que le corresponde, guarda el valor del coeficiente de la máscara para que en el siguiente ciclo de reloj sea enviado a su procesador vecino. Por lo tanto, el procesador que está por pasar el coeficiente recibido a su vecino se encuentra en una etapa adelante con respecto a su vecino. Por tanto, en el orden en el que inicia la actividad en cada procesador, en ese orden entregan su resultado (Acc) de procesar la ventana. Este paso de datos y procesamiento en diferentes etapas está basado en la técnica pipeline, y se ha implementado de manera circular ya que los procesadores que van entregando sus resultados reinician el cálculo de una nueva ventana empezando con el ciclo nuevamente. Cuando cada procesador inicia el cálculo de una ventana, los valores de sus registros internos se encuentran con valor cero.

Una vez que cada procesador ha terminado su cálculo, el valor acumulado en Acc es el valor que forma parte de la imagen de salida, ubicando ese valor en el lugar en el que se encuentra centrada la ventana de la imagen. En la Fig.4.9 se señala la ubicación de los valores de salida de cada procesador una vez que hayan acabado con su procesamiento.

Los procesadores en el arreglo sistólico 1-D procesan su ventana en forma paralela, siendo ésta en la dirección vertical, ya que el arreglo sistólico 1-D procesa ventanas de la imagen centradas en la misma columna pero en diferentes renglones. El procesador ubicado más arriba inicia y termina primero su cálculo. Los otros procesadores trabajan progresivamente de acuerdo a su posición en el arreglo. El *Colector de Datos 1-D* toma los resultados de los procesadores de la columna iniciando de arriba hacia abajo en el arreglo, teniendo un dato a la vez como resultado. Cuando el elemento de procesamiento entrega el resultado, se encuentra listo para iniciar un nuevo cálculo de ventana e iniciar con el ciclo descrito para obtener un nuevo dato de salida.

Extendiendo el paralelismo y procesamiento de ventanas en la dirección horizontal en la imagen de entrada, varios arreglos sistólicos 1-D son interconectados, formando un arreglo sistólico 2-D. Dado que la lectura de la imagen es por columna, el primer arreglo sistólico 1-D del arreglo sistólico 2-D es el primero en iniciar actividad. Una vez que se ha iniciado la lectura de la segunda columna de la imagen de entrada, los procesadores

que comprenden el segundo arreglo sistólico 1-D inician su actividad gradualmente.

Continuando con el ejemplo planteado, en la Fig.4.10 se tiene un arreglo sistólico 2-D el cual se encuentra formado por tres arreglos sistólicos 1-D, que a su vez están compuestos por tres procesadores, dando un total de nueve procesadores. La lectura de la imagen de entrada se ubica en la tercera columna. El arreglo sistólico 2-D ha recibido el píxel P_{22} , y el coeficiente de la máscara W_{22} . El *Procesador 1* se encuentra en la última etapa de su procesamiento mientras que para el *Procesador 9* está en el inicio de su actividad. En la figura se observa en la etapa en la que se encuentra cada procesador en el cálculo de su ventana. Se observa que la actividad de los procesadores inicia de manera progresiva, uno a la vez. Además, el paso del coeficiente de la máscara entre los arreglos sistólicos 1-D está conectada por la *Línea de Retardo*, la cual es útil para retener el coeficiente de la máscara el tiempo necesario para estar en sincronía con la etapa en la que se encuentra el siguiente arreglo sistólico 1-D.

El *Colector de Datos 2-D* recibe, del *Colector de Datos 1-D*, los resultados obtenidos por los procesadores. El *Colector de Datos 2-D* le dedica cierto tiempo a cada *Colector de Datos 1-D* para recuperar los datos de cada procesador en el arreglo sistólico 1-D. Una vez que el procesador entrega el dato se prepara, dando a sus variables el valor de cero, para cuando se inicie nuevamente su actividad con el procesamiento de otra ventana (esto ocurrirá cuando inicie la lectura de la cuarta columna de la imagen de entrada).

Hasta este punto, se ha explicado el flujo de datos en un arreglo sistólico 2-D aplicando un filtro sobre la imagen de entrada. Como se cuenta con más de un filtro para aplicar sobre la imagen de entrada, se extiende el arreglo de procesadores a un nivel más, creando un arreglo sistólico 3-D. Este consiste en crear arreglos sistólicos 2-D tantos como filtros se deseen aplicar sobre la imagen de entrada, esto es, ya que cada arreglo sistólico 2-D será responsable de aplicar un filtro sobre la imagen de entrada es necesario contar con el mismo número de arreglos sistólicos 2-D como filtros a aplicar. Todos los arreglos sistólicos 2-D comparten el mismo valor de píxel de la imagen, pero diferente coeficiente de la máscara, ya que esta última es la que determina el filtro aplicado sobre la imagen. Además, comparten las señales de control, pero no hay interconexión entre ellas entre los arreglos. Una vez que cada *Colector de Datos 2-D* tiene datos de salida, éstos son colectados por un registro temporal para después ser almacenados en los bancos de memoria de salida. El número de imágenes que se tendrán de salida depende del número de filtros

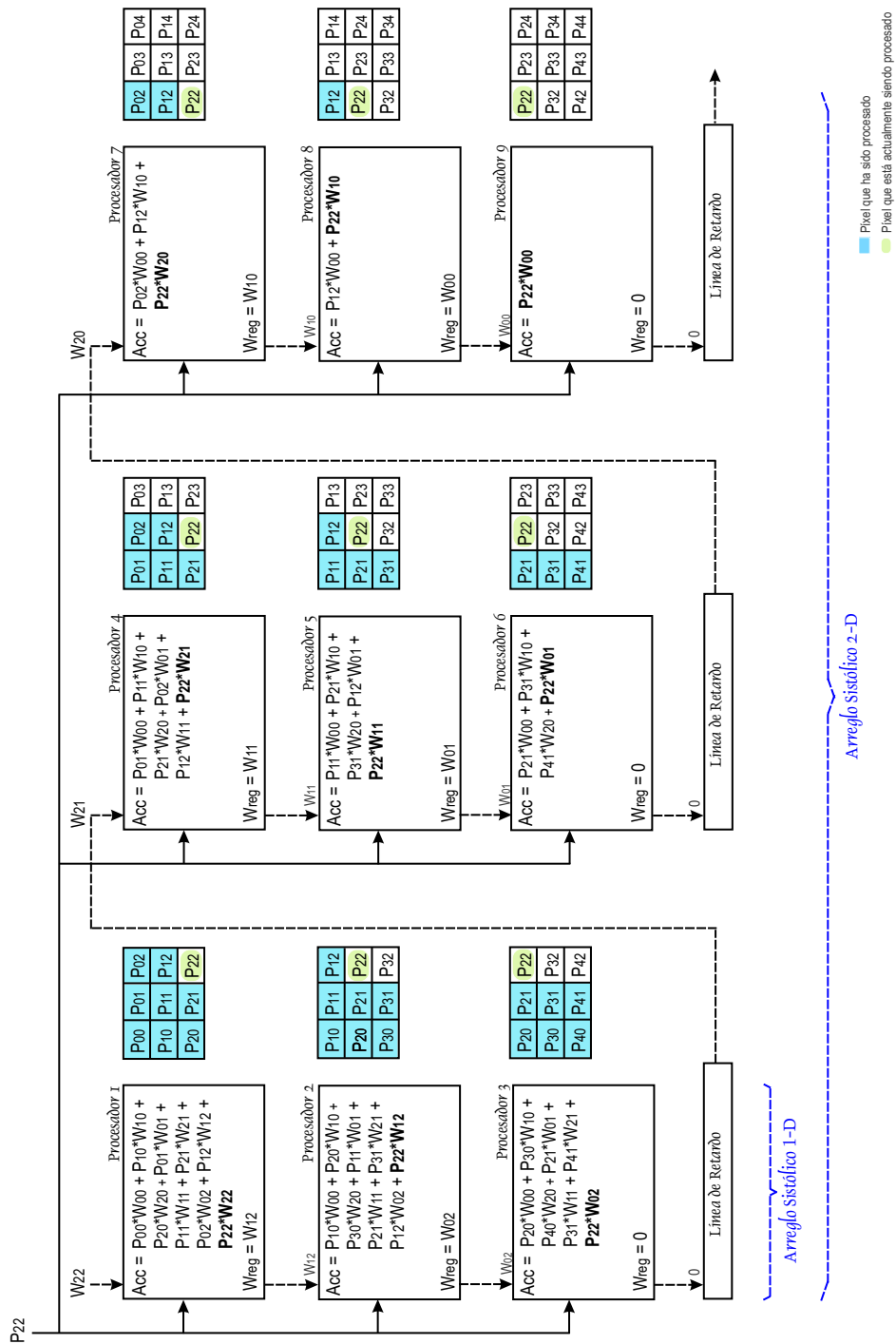


Figura 4.10: Diagrama general del funcionamiento del arreglo sistólico 2-D compuesto por tres arreglos sistólicos 1-D, cuyas ventanas a procesar se encuentran a la derecha de cada procesador.

aplicados a la imagen de entrada.

Para ejemplificar lo anterior, se tiene un arreglo sistólico 3-D el cual se encuentra formado por dos arreglos sistólicos 2-D donde cada uno de ellos aplica un filtro a la imagen de entrada. Cada arreglo sistólico 2-D cuenta con tres arreglos sistólicos 1-D, como se ha venido mostrando en el ejemplo. En la Fig.4.11 se tienen dos arreglos sistólicos 1-D los cuales forman parte de arreglos sistólicos 2-D diferentes. Los procesadores etiquetados como *Procesador 1* pertenecen a arreglos sistólicos 2-D distintos, pero que son responsables de procesar la misma ventana la cual se encuentra a su derecha. La diferencia entre estos procesadores radica en que cada procesador calcula una máscara diferente, W y X . Los procesadores etiquetados como *Procesador 1* han terminado de procesar su ventana y están listos para entregar su resultado almacenado en *Acc*. Los procesadores *Procesador 2* se encuentran en su última etapa, mientras que los procesadores *Procesador 3* se encuentran en su penúltima etapa.

Ya que se obtienen dos datos de salida a la vez, el colector de datos en este nivel es responsable de gestionar y organizar dichos datos de tal manera que se guarden en los bancos de memoria de salida sin tener pérdidas para que posteriormente se recuperen e interpreten de manera correcta, como imágenes.

Para el presente trabajo, se crean ocho arreglos sistólicos 2-D para realizar el cálculo de ocho filtros Gabor-2D en diferentes orientaciones. Teniendo que gestionar la salida de ocho datos a la vez para que sean almacenados en los bancos de memoria de salida. Esta gestión está a cargo del *Almacenamiento Temporal* el cual recibe los ocho datos de salida, y los agrupa de tal manera que formen dos datos. Con ello se pretende que sean el mínimo de accesos a memoria para almacenar datos. Una vez que se recuperen los datos almacenados en las memorias de salida, se realiza la correcta distribución de los datos formando las ocho imágenes.

4.5. Análisis de Desempeño

En esta sección se presenta un análisis teórico de desempeño de la arquitectura sistólica 3-D, iniciando este análisis con el análisis de desempeño de un arreglo sistólico 1-D, cuyos resultados se pueden escalar al arreglo sistólico 3-D.

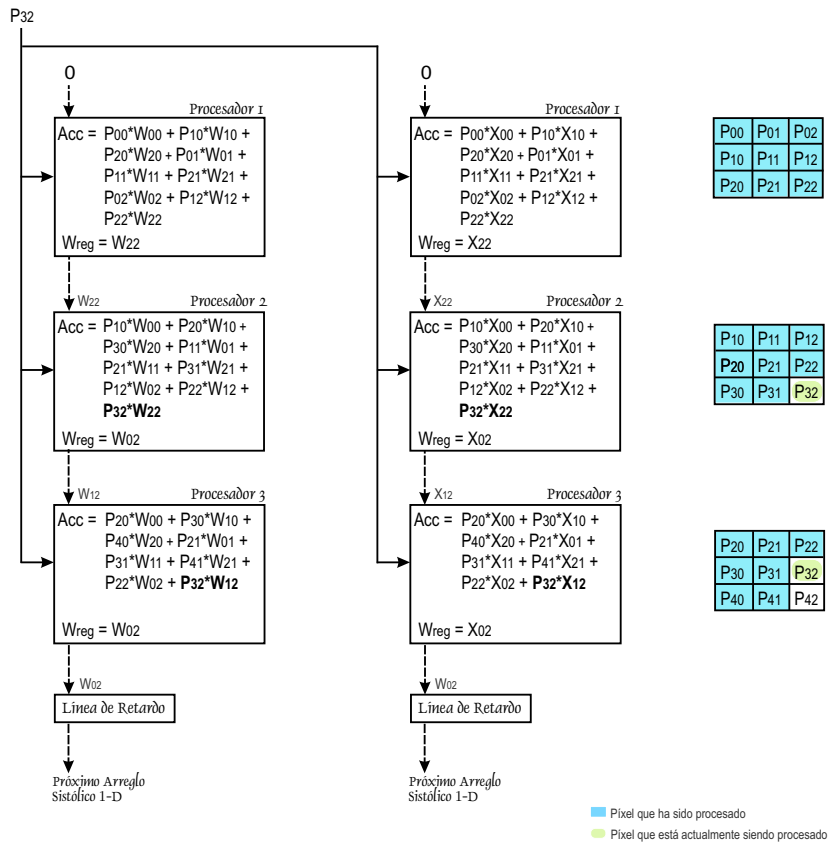


Figura 4.11: Diagrama general del funcionamiento de un arreglo sistólico 3-D compuesto por dos arreglos sistólicos 2-D (mostrando sólo un arreglo sistólico 1-D de cada arreglo).

Cuando la lectura de los píxeles de la imagen inicia, el píxel actual es distribuido al conjunto completo de los procesadores que comprender el arreglo sistólico 2-D. Sin embargo, algunos $GWPs$ no pueden iniciar inmediatamente su procesamiento ya que el píxel actual no es parte de los píxeles que le corresponde procesar. Lo anterior se ilustra en la Fig.4.12 donde se cuenta con siete procesadores, número que corresponde al tamaño de la ventana de la máscara (w). Los procesadores inician su actividad de manera progresiva. En esta figura se aprecia que los procesadores se mantienen activos w ciclos de reloj lo que corresponde al número de píxeles que procesan por columna, y después se desactivan mientras los píxeles que conforman otras ventanas están siendo leídos. Así como se van activando gradualmente los procesadores, una vez que alcanzan el número máximo de procesadores activos, empiezan a decrecer. Este proceso se repite en la lectura de la

siguiente columna hasta el final del procesamiento de la imagen de entrada.

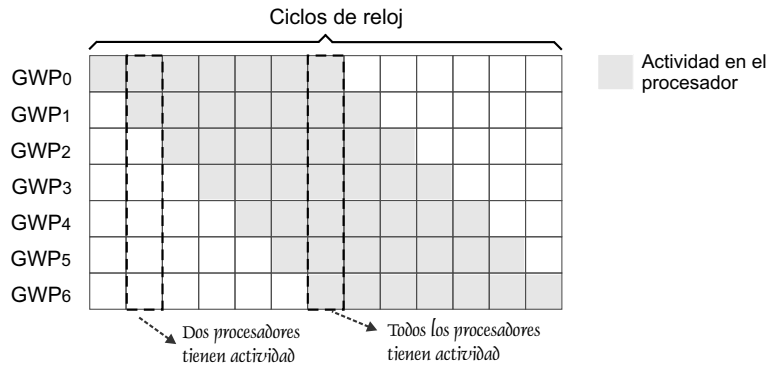


Figura 4.12: Desarrollo del crecimiento y decrecimiento progresivo de la actividad de siete procesadores que componen un arreglo sistólico 1-D.

Aunque la inactividad de una porción de procesadores del arreglo sistólico puede ser considerada como una desventaja en términos de eficiencia, éste se puede reconsiderar por ventajas como: todos procesador en el arreglo es extremadamente regular; las conexiones de los procesadores son locales, reduciendo problemas de interconexión; no es necesaria la distribución de líneas de control sobre todo el arreglo; no es requerida una unidad de control compleja [14].

En la Tabla 4.1 se resumen las ecuaciones utilizadas para el análisis de un arreglo sistólico 2-D del trabajo [14].

Dada una máscara de tamaño $TamMasc$; una imagen $M \times N$, donde M corresponde al número de filas y N al número de columnas de la imagen; se considera realizar $N_{GWP}(R)$ ventanas en la dirección vertical (ventanas centradas en diferentes renglones), y $N_{GWP}(C)$ ventanas en la dirección horizontal (ventanas centradas en diferentes columnas). F es la frecuencia de reloj y NIP_{GWP} es el promedio de número de operaciones elementales calculadas por GWP en cada ciclo de reloj.

Para un arreglo sistólico 3-D se tienen N_{Masc} número de máscaras. Partiendo de las ecuaciones de la Tabla 4.5, se tiene que el número de procesadores GWP aumenta por un factor de N_{Masc} .

Estas ecuaciones aplican para el arreglo sistólico 3-D, ya que no se ve alterado los tiempos dado que se está procesando los arreglos sistólicos 2-D en paralelo. Así, el tiempo

Descripción	Fórmula
Número de GWP s arreglo sistólico 2-D	$N_{GWP2D} = N_{GWP}(R) \times N_{GWP}(C)$
Tiempo de Latencia	$\tau_l = [(TamMasc - 1) \times (N_{GWP}(R) + TamMasc - 1)] \times \frac{1}{F} \times \frac{M}{N_{GWP}(R)}$
Tiempo de Procesamiento paralelo	$\tau_p = M \times N \times \left[\frac{N_{GWP}(R) + TamMasc - 1}{N_{GWP}(R)} \right] \times \frac{1}{F}$
Tiempo total	$T = \tau_l + \tau_p$
Eficiencia / Throughput	$Throughput = \frac{1}{T}$ $Throughput = F \times (N_{GWP}(R) \times N_{GWP}(C) \times N_{Masc}) \times NIP_{GWP}$

Tabla 4.1: Ecuaciones del análisis matemático hecho a un arreglo sistólico 2-D [14].

que se invierte en el procesamiento de un arreglo sistólico 2-D, es el mismo para otro.

De acuerdo a [14], la estimación simplificada en recursos hardware que consumiría un arreglo sistólico 2-D está dada por: registros internos = $N_{GWP}(R) \times N_{GWP}(C)$, número de multiplicadores = $N_{GWP}(R) \times N_{GWP}(C)$, número total de registros para la línea de retardo = $(N_{GWP}(R) - 1) \times (N_{GWP}(C) - 1)$. Por lo tanto, para un arreglo sistólico 3-D aumenta en un factor de N_{Masc} . Para apearse a los recursos reales de consumo se necesita considerar recursos de las unidades de control, memoria de entrada y salida, etc.

Partiendo del caso particular de aplicar el filtro a una imagen 512x512 mediante una máscara de convolución 7x7, procesando 7 ventanas en la dirección horizontal, y 7 ventanas en la dirección vertical; se tiene que se necesitan 492,544 ciclos de reloj para procesar la imagen. Si la arquitectura operara con una frecuencia de reloj mínima de 15 MHz se cumpliría con requerimientos de tiempo real. Para aplicar ocho máscaras de convolución sobre la misma imagen, dado que los filtros se ejecutan de forma paralela, el número de ciclos de reloj requeridos se mantiene fijo.

En el siguiente capítulo se realiza la validación completa, a nivel funcional y timing, de la arquitectura. Se muestran detalles del modelado y simulaciones de diferentes componentes.

Capítulo 5

Modelado VHDL de la Arquitectura Sistólica 3-D

La Arquitectura Sistólica 3-D comprende un arreglo sistólico 3-D (descrito en el capítulo anterior), así como las unidades de control. En este sistema se identifica procesamiento paralelo dentro del arreglo sistólico 3-D como en la gestión de las señales de control. El comportamiento del sistema ha sido modelado en VHDL bajo la herramienta Project Navigator de Xilinx ISE 8.2i. En el presente capítulo se muestra la validación a nivel funcional y timing de la arquitectura propuesta. La simulación fue realizada en ModelSim SE 6.0.

5.1. Aspectos Generales

La arquitectura fue diseñada para ser flexible, esto es, los módulos VHDL hacen uso de tipos genéricos que parametrizan variables del sistema.

Entre las consideraciones definidas para obtener una arquitectura flexible se encuentra lo siguiente:

- Los coeficientes de ventanas pueden cambiar de acuerdo a la resolución numérica requerida, partiendo de máscaras cuadradas con ancho impar, mayor o igual a tres
- La resolución de la imagen puede variar para proporcionar una gama de pruebas posibles, partiendo de imágenes en escala de grises

- Número de renglones a procesar en paralelo, para contar con pruebas que demuestren el número adecuado a ser considerado
- El número de orientaciones a calcular, esto es, se puede definir un número mayor/menor de orientaciones a calcular tomando en cuenta las restricciones en recursos hardware

El modelado HDL comprende los siguientes módulos:

- GWP, procesador elemental
- *Unidad de Direccionamiento* a los Bancos de Memoria
- Unidad de Control del arreglo sistólico 2-D
- Arreglo sistólico 1-D
- Arreglo sistólico 2-D

Para el diseño se consideraron los valores en representación de complemento a 2. Dado que los valores de los coeficientes de la máscara son valores decimales en el rango $[-1,1]$, se implementó el uso de aritmética de punto fijo con 8 bits para los valores de los píxeles de la imagen, y 10 bits para los valores de los coeficientes de la máscara, de los cuales 9 bits corresponden a la parte fraccionaria y un bit al signo.

Enseguida se muestra el funcionamiento de la arquitectura con diagramas de tiempo, específicamente del procesamiento del que es responsable el arreglo sistólico 3-D partiendo desde el procesador elemental. Para obtener mayor claridad de las señales de control y de los datos involucrados se dará seguimiento a la aplicación de un filtro sobre una imagen como ejemplo. La nomenclatura utilizada en este capítulo es, para el ancho de la máscara como w , el número de renglones procesados en paralelo como NR .

5.2. Ejemplo de Aplicación del Filtro

Partiendo de la imagen de la Fig. 5.1, se toma una porción de ella de tamaño 13x13. Esta imagen es una imagen sintética que se ha construido con líneas en las ocho orientaciones y así poder visualizar mejor la intención del filtro. La subimagen de tamaño 13x13

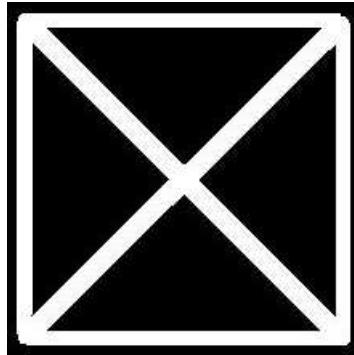


Figura 5.1: Imagen de entrada 220x220.

será nombrada por el resto del capítulo como imagen 13x13, ya que es preciso distinguirla porque sobre ella se concentra el desarrollo del comportamiento de los procesadores y señales de control. La imagen 13x13 tiene los siguientes datos:

$$\begin{bmatrix}
 255 & 255 & 14 & 3 & 0 & 247 & 254 & 255 & 253 & 255 & 255 & 255 & 255 \\
 255 & 255 & 0 & 5 & 10 & 2 & 255 & 252 & 248 & 254 & 255 & 255 & 255 \\
 255 & 255 & 1 & 8 & 1 & 0 & 14 & 255 & 245 & 255 & 255 & 255 & 255 \\
 255 & 255 & 5 & 4 & 0 & 25 & 0 & 0 & 255 & 252 & 255 & 255 & 255 \\
 255 & 255 & 9 & 0 & 15 & 0 & 5 & 0 & 14 & 242 & 255 & 245 & 255 \\
 255 & 255 & 0 & 0 & 16 & 0 & 0 & 4 & 0 & 14 & 242 & 255 & 242 \\
 255 & 255 & 7 & 1 & 0 & 0 & 0 & 0 & 12 & 0 & 7 & 251 & 254 \\
 255 & 255 & 4 & 0 & 0 & 20 & 4 & 0 & 10 & 6 & 0 & 0 & 255 \\
 255 & 255 & 0 & 3 & 3 & 0 & 2 & 4 & 0 & 0 & 14 & 3 & 0 \\
 255 & 255 & 0 & 4 & 5 & 0 & 0 & 13 & 0 & 15 & 0 & 5 & 10 \\
 255 & 255 & 0 & 0 & 6 & 9 & 5 & 0 & 0 & 2 & 1 & 8 & 1 \\
 255 & 255 & 11 & 0 & 0 & 4 & 0 & 0 & 12 & 0 & 5 & 4 & 0 \\
 255 & 255 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9 & 0 & 15
 \end{bmatrix} \tag{5.1}$$

Como parte de los datos requeridos, se necesita conocer la dirección en que se ubica cada dato en la memoria. Estas direcciones se muestran en la siguiente matriz.

$$\begin{bmatrix}
 6174 & 6175 & 6176 & 6177 & 6178 & 6179 & 6180 & 6181 & 6182 & 6183 & 6184 & 6185 & 6186 \\
 6394 & 6395 & 6396 & 6397 & 6398 & 6399 & 6400 & 6401 & 6402 & 6403 & 6404 & 6405 & 6406 \\
 6614 & 6615 & 6616 & 6617 & 6618 & 6619 & 6620 & 6621 & 6622 & 6623 & 6624 & 6625 & 6626 \\
 6834 & 6835 & 6836 & 6837 & 6838 & 6839 & 6840 & 6841 & 6842 & 6843 & 6844 & 6845 & 6846 \\
 7054 & 7055 & 7056 & 7057 & 7058 & 7059 & 7060 & 7061 & 7062 & 7063 & 7064 & 7065 & 7066 \\
 7274 & 7275 & 7276 & 7277 & 7278 & 7279 & 7280 & 7281 & 7282 & 7283 & 7284 & 7285 & 7286 \\
 7494 & 7495 & 7496 & 7497 & 7498 & 7499 & 7500 & 7501 & 7502 & 7503 & 7504 & 7505 & 7506 \\
 7714 & 7715 & 7716 & 7717 & 7718 & 7719 & 7720 & 7721 & 7722 & 7723 & 7724 & 7725 & 7726 \\
 7934 & 7935 & 7936 & 7937 & 7938 & 7939 & 7940 & 7941 & 7942 & 7943 & 7944 & 7945 & 7946 \\
 8154 & 8155 & 8156 & 8157 & 8158 & 8159 & 8160 & 8161 & 8162 & 8163 & 8164 & 8165 & 8166 \\
 8374 & 8375 & 8376 & 8377 & 8378 & 8379 & 8380 & 8381 & 8382 & 8383 & 8384 & 8385 & 8386 \\
 8594 & 8595 & 8596 & 8597 & 8598 & 8599 & 8600 & 8601 & 8602 & 8603 & 8604 & 8605 & 8606 \\
 8814 & 8815 & 8816 & 8817 & 8818 & 8819 & 8820 & 8821 & 8822 & 8823 & 8824 & 8825 & 8826
 \end{bmatrix}
 \tag{5.2}$$

Los valores de los coeficientes son valores con parte fraccionaria. Para esto, se hace una transformación de los valores de los coeficientes de la máscara a un valor binario con representación en punto fijo para ser estos datos en binario los valores de entrada. Durante el procesamiento del filtro aplicado sobre una imagen, se realiza el cálculo normal de datos binarios. Una vez que se obtiene el resultado, éste se interpreta de acuerdo a la representación realizada en punto fijo.

En los diagramas de tiempo que se ilustran en este capítulo, los valores de la máscara y del resultado se muestran como números enteros con signo. En el capítulo 6 se presentará los valores de las máscaras Gabor-2D, su representación en punto fijo, así como la interpretación de los valores de salida.

Por lo anterior, los valores de los coeficientes de la máscara que se presentan a continuación son valores enteros con signo, los cuales vienen dados por la representación de dichos valores en binario con punto fijo. La máscara a aplicar es una ventana de tamaño 7x7, la cual es la siguiente:

$$\begin{bmatrix} 7 & 3 & -47 & 86 & -33 & -81 & 115 \\ 3 & -52 & 103 & -43 & -116 & 179 & -81 \\ -47 & 103 & -48 & -139 & 234 & -116 & -33 \\ 86 & -43 & -139 & 255 & -139 & -43 & 86 \\ -33 & -116 & 234 & -139 & -48 & 103 & -47 \\ -81 & 179 & -116 & -43 & 103 & -52 & 3 \\ 115 & -81 & -33 & 86 & -47 & 3 & 7 \end{bmatrix} \quad (5.3)$$

El número de renglones que se van a procesar en paralelo, NR , son 7. Por lo tanto, el número de píxeles a leer por columna son 13. El tamaño de la imagen extraída para ejemplificar la simulación cubre los datos necesarios a leer de la memoria para procesar las ventanas que estén implicadas.

5.3. Unidad de Procesamiento Elemental

En la siguiente Tabla 5.3 se muestra las señales que contiene la interfaz de un procesador. Los puertos P y W son señales de entrada los cuales contienen los datos de la imagen y el coeficiente de la máscara, respectivamente. En dicho módulo se encuentra un multiplicador, el cual opera sobre dichos datos, y el valor resultante se acumula. El procesador opera si la señal de RST se encuentra en '0' y EP se encuentra activado, '1'. El valor de la señal W es retenida por un ciclo de reloj en un registro, y al siguiente ciclo de reloj es colocada como salida Wd .

Como parte de la arquitectura, existen un módulo de direccionamiento y control del flujo de datos. Este modulo es la *Unidad de Direccionamiento*, la cual controla los accesos a memoria para leer y guardar datos en los bancos de memoria, así como genera y gestiona señales de control.

En la Fig. 5.2 se observa la lectura de la primera columna de la imagen 13x13. Las líneas de control de dirección de la memoria de entrada corresponde a $addrMemImg$, y la señal de dirección de la memoria de la máscara es $addrMemMsk$. Estas señales son generadas por la *Unidad de Direccionamiento*. Como se puede observar, el primer píxel leído de la columna corresponde al valor 255 el cual se encuentra en la dirección 6174,

Puerto	Modo	Tamaño	Descripción
CLK	In	1	Señal del reloj
RST	In	1	Señal de reset del procesador
P	In	8	Píxel de la imagen de entrada
W	In	10	Coficiente de la máscara
EP	In	1	Señal de estado: activo/no activo
Wd	Out	10	Coficiente de la máscara anterior
Pd	Out	8	Resultado del cálculo

Tabla 5.1: Interfaz del elemento de procesamiento con una breve descripción de la funcionalidad de sus señales.

el segundo valor es 255 en la dirección 6394, y así sucesivamente. El número de píxeles a leer son 13, es decir, hasta la dirección 8814; es entonces cuando inicia la lectura de la siguiente columna en su dirección 6175. La señal que controla el número de píxeles a leer por columna es *cntActGWP* la cual es generada por la *Unidad de Direccionamiento*. Esta señal es un contador del número de píxeles a leer por columna, el cual va de 0 a 12. Con esta figura se muestra la lectura de los datos de entrada, píxel de la imagen y coeficiente de la máscara, los cuales son valores de entrada a los *GWPs*.

Para iniciar con la presentación de las señales y comportamiento de la arquitectura, se parte de un arreglo sistólico 1-D formando por 7 procesadores.

En la Fig. 5.3 se muestra, en un diagrama de tiempo, la activación gradual de los siete procesadores. La señal *EP* es la señal de habilitación del procesador. Cuando *EP* tiene el valor 1, habilita las actividades del GWP, en otro caso el GWP se encuentra inactivo. En la figura se aprecian las señales *EP* de 7 *GWPs*, etiquetados como: *ep_gwp0*, *ep_gwp1*, ..., *ep_gwp6*. El tiempo de actividad de los GWP depende del tamaño de la máscara, es decir, para este caso de aplicar una máscara 7x7 la habilitación de los GWP dura siete ciclos de reloj. Este número también corresponde a los píxeles de la ventana de la imagen 13x13 que le corresponde procesar a cada GWP. Los GWP se van activando en cascada, y así mismo se van deshabilitando, siendo el primer GWP quien inicia y termina primero su actividad.

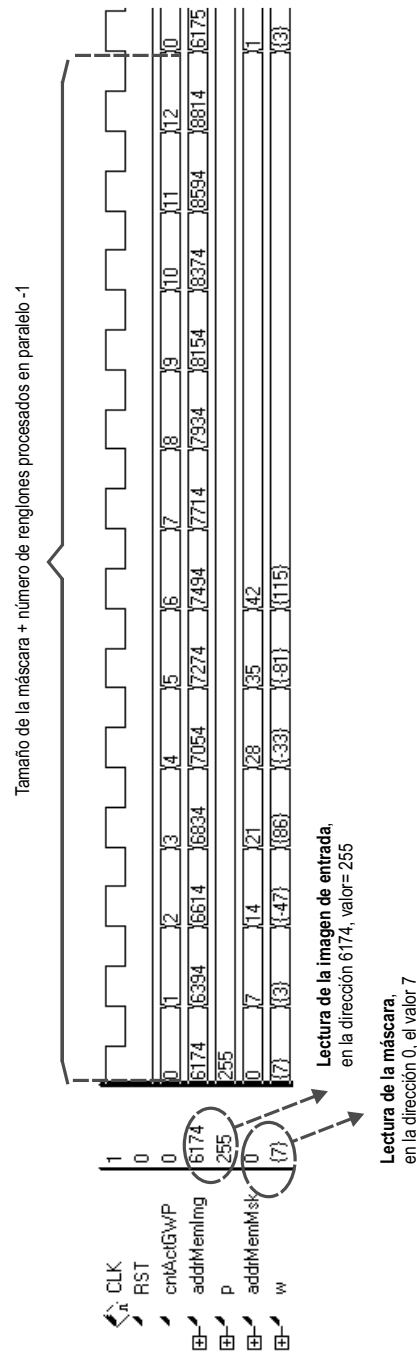


Figura 5.2: Diagrama de tiempo de la lectura de 13 píxeles de la imagen, 7 de ellos corresponden al tamaño de la máscara necesarios para procesar una ventana de la imagen, y 6 píxeles más para procesar 7 ventanas en paralelo.

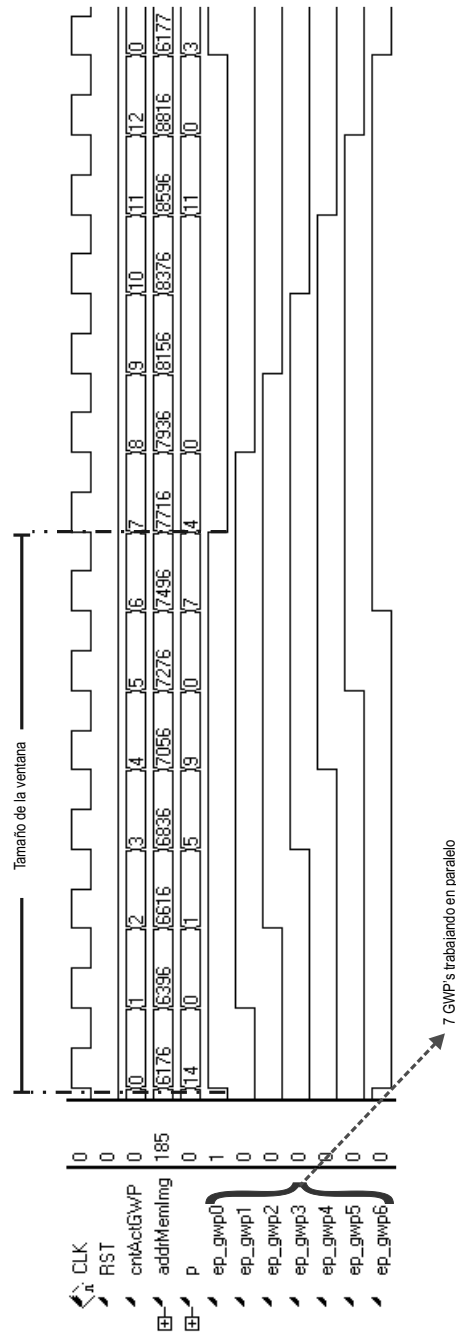


Figura 5.3: Diagrama de tiempo de la lectura de la tercera columna, destacando la señal *EP* la cual activa la actividad del GWP al que pertenece.

En la Fig. 5.4 se muestra la actividad de estos *GWPs*. Como se observa, el primer dato ubicado en la dirección 6154 sólo lo utiliza el primer procesador; cuando se lee el segundo dato (dirección 6392) inicia actividad el segundo procesador, habiendo en ese momento dos procesadores activos. Además, se puede apreciar el paso del coeficiente de la máscara entre los *GWPs*. En este segundo ciclo, el primer GWP se encuentra en su segundo procesamiento de datos, mientras que el segundo GWP se encuentra en el primero, así que el último dato leído de la máscara es utilizado por el primer procesador, pasando éste último el dato del coeficiente de la máscara al siguiente procesador. El resultado de multiplicar los valores de entrada p y w se almacena en la señal *result_gwp*, y éste a su vez es adicionado al valor acumulador, *acc_gwp*. Cuando se lee el tercer dato de la memoria de entrada, el valor 255 de la dirección 6614, inicia la actividad del tercer GWP, recibiendo el dato del coeficiente utilizado por el segundo GWP. Y así sucesivamente se van activando los siete procesadores. Cuando todos los procesadores se encuentran habilitados, el primer GWP obtiene el último dato necesario de esa columna para procesar su ventana y el último GWP ha iniciado su actividad. Es en el siguiente ciclo cuando inician los GWPs a deshabilitarse en el orden en que fueron habilitados.

Cuando se termina de leer la primera columna de datos de la imagen 13x13, los 7 *GWPs* han procesado siete pares de datos. El procesamiento es en paralelo en la dirección vertical, procesando ventanas de la imagen ubicadas en diferente renglón sobre la misma columna. Las ventanas de la imagen 13x13 que les corresponde procesar a cada GWP son las siguientes: el primer GWP es responsable del procesamiento de la ventana que inicia en la dirección 6174 y termina en la dirección 7500 formando un cuadrado 7x7. De igual manera, el segundo GWP inicia su actividad en la lectura del píxel en la dirección 6394 y termina en la dirección 7720, y así, sucesivamente, formando ventanas a procesar centradas en diferente renglón sobre la misma columna.

Una vez que se hayan leído los 13 píxeles necesarios para procesar las siete ventanas, se inicia la lectura de la siguiente columna de la imagen 13x13 en la dirección 6175. Los siete *GWPs* toman los píxeles que necesitan para continuar con su procesamiento, activándose de manera progresiva como se ha descrito.

En la Fig. 5.5 se muestra el diagrama de tiempo en la lectura de la tercera columna de la imagen 13x13. Con este diagrama se puede apreciar los valores de resultado de multiplicar los valores de entrada p y w , almacenado en *result_gwpx*, además del valor

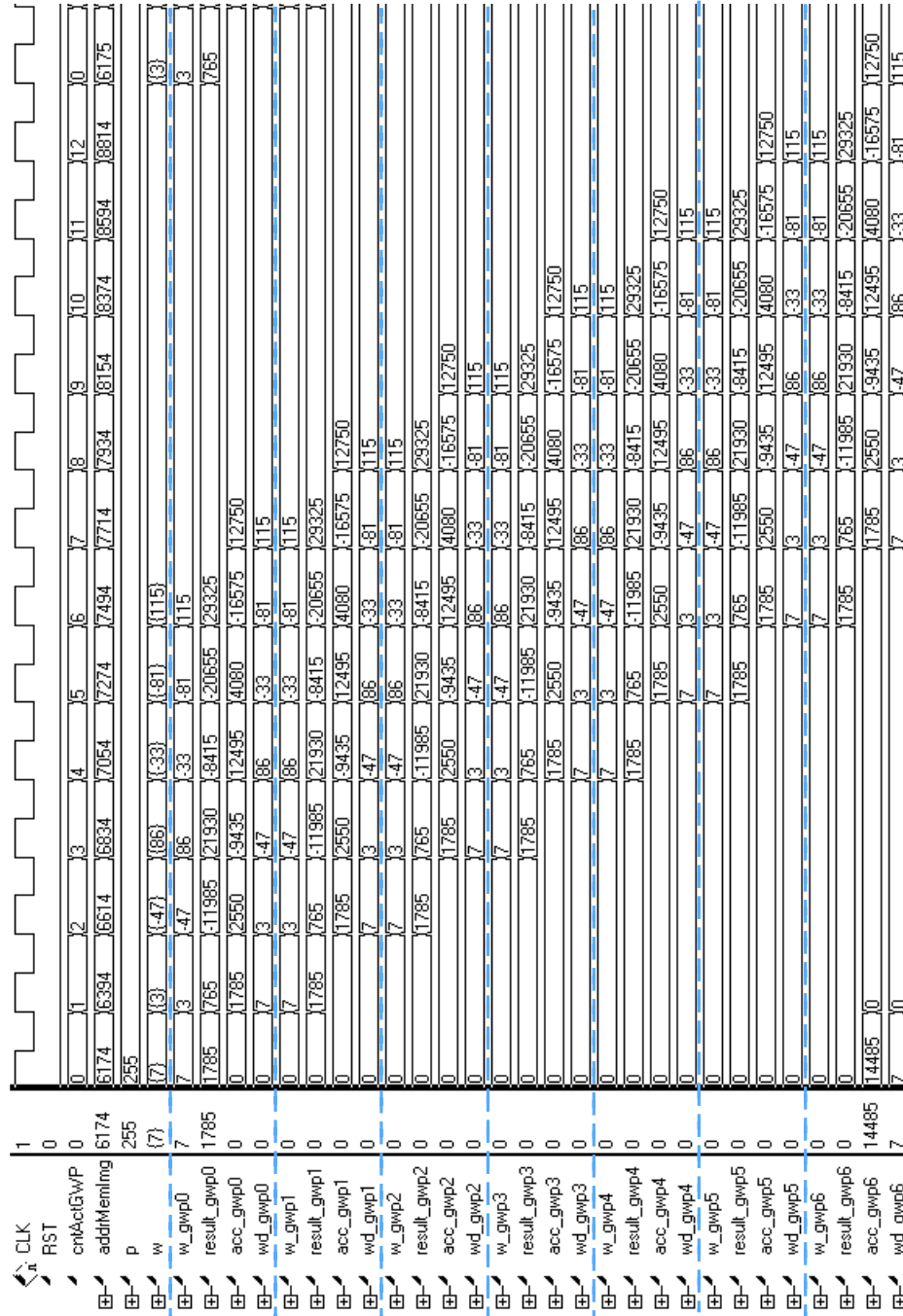


Figura 5.4: Diagrama de tiempo que muestra la actividad de siete procesadores, los cuales corresponden a la primera columna de la ventana de la imagen de entrada a procesar.

acumulado que hasta ese momento llevan almacenado los GWP_s , en la señal acc_gwp_x . Como la máscara es de dimensiones 7×7 , entonces el número de datos que se procesarán por ventana serán 49. En la figura se observa que el GWP_0 se encuentra un proceso adelante que su vecino GWP_1 , a su vez, GWP_1 se encuentra adelante en su procesamiento que GWP_2 , y así sucesivamente. Además, con este diagrama se resalta el paso de los coeficientes de la máscara a sus procesadores vecinos.

Cuando se llega a la lectura de la séptima columna de la imagen 13×13 , los GWP se encuentran en la espera de sus últimos siete par de datos a procesar. En la dirección 7500 es cuando el primer GWP procesa su último valor, y en el siguiente ciclo se almacena dicho valor en el *Colector de Datos 1-D*, éste último etiquetado como px_1D . El *Colector de Datos 1-D* es un arreglo cuyo tamaño depende del número de renglones a procesar, es decir, el número de GWP_s que comprende el arreglo sistólico 1-D. Cuando el primer GWP entrega su valor al colector de datos, éste lo almacena en la primera posición del arreglo. Cuando el segundo GWP termina con el procesamiento de su ventana, en el siguiente ciclo de reloj, entrega su resultado siendo colocado en la segunda posición del colector de datos. Este proceso de almacenamiento de datos en el *Colector de Datos 1-D* se realiza de manera repetitiva hasta la entrega del último resultado por parte del séptimo GWP del arreglo sistólico 1-D. En la Fig. 5.6 se puede apreciar este proceso. En la misma figura se observa la línea de retardo, compuesta por un arreglo de seis registros, en la cual se van reteniendo los valores de la máscara los cuales serán utilizados por un arreglo sistólico 1-D vecino. En la siguiente sección se presenta los módulos que comprende una arquitectura sistólica 2-D y diagramas de tiempo que ilustran el comportamiento de las señales principales.

5.4. Arquitectura Sistólica 2-D

Para construir la Arquitectura Sistólica 2-D se debe contar con las unidades de control así como un arreglo sistólico 2-D la cual comprende arreglos sistólicos 1-D. Las unidades de control son: la Unidad de Direccionamiento y *Unidad de Control 2-D*. Las interfaces de estas unidades no se detallan ya que se encuentran basados en las unidades diseñadas en el trabajo [14], y en éste último se encuentran los detalles de su funcionamiento.

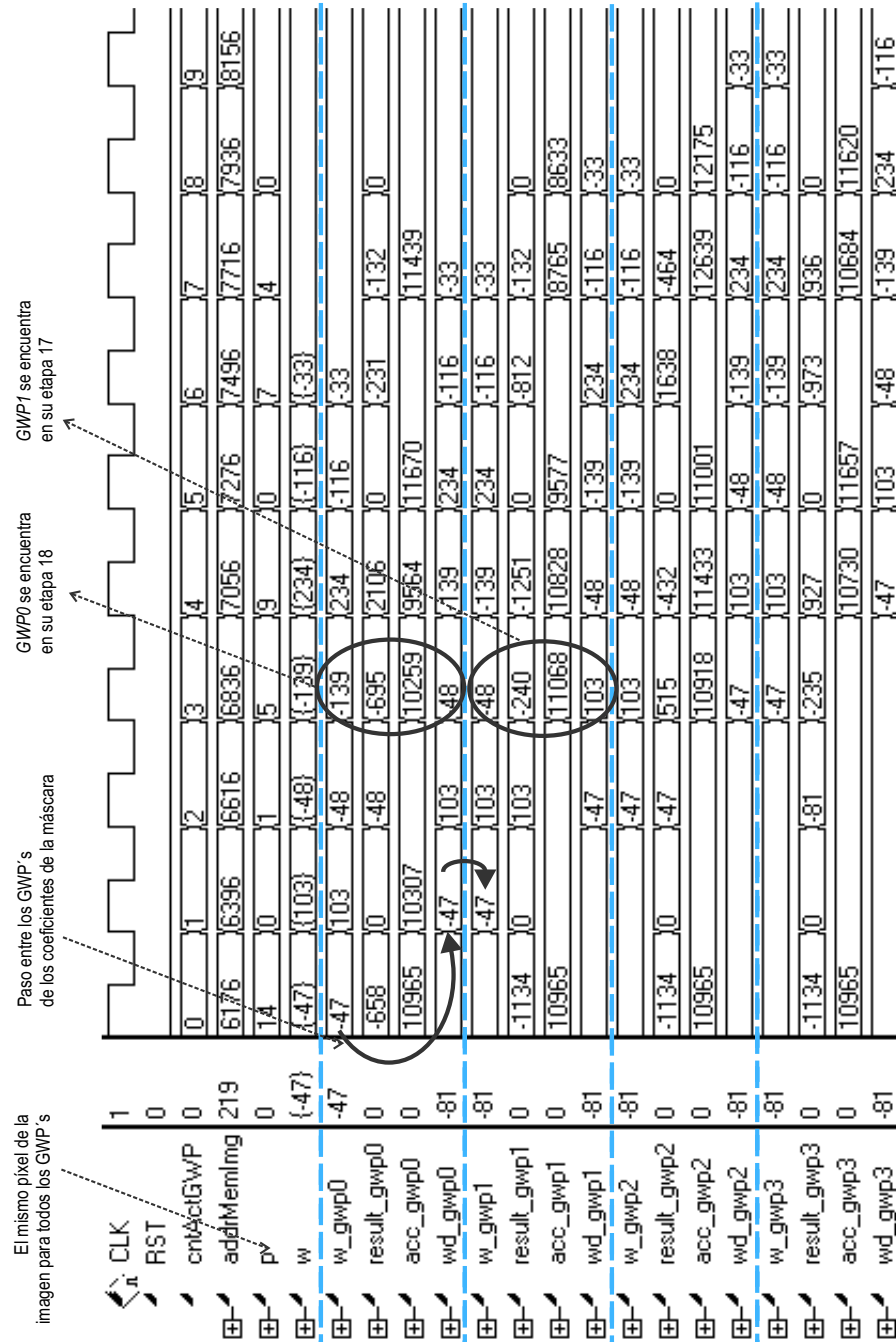


Figura 5.5: Diagrama de tiempo en la lectura de la tercera columna, y el procesamiento en el que se encuentran cuatro de los siete *GWP*s.

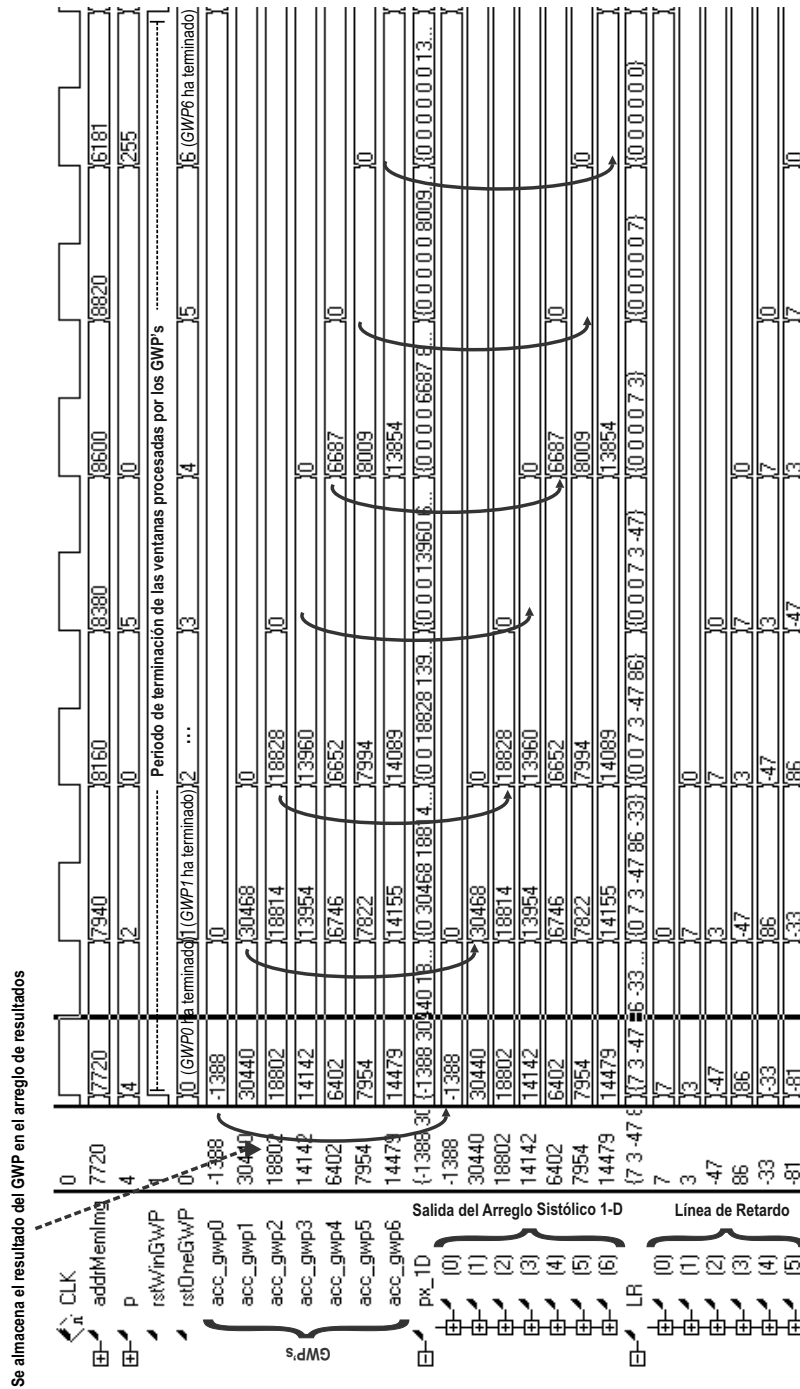


Figura 5.6: Diagrama de tiempo en el que los GWP entregan su resultado al *Colector de Datos 1-D*, etiquetado como *px_1D*.

En la Fig. 5.7 se ilustra las conexiones entre la *Unidad de Direccionamiento*, la *Unidad de Control 2-D* y el módulo de un arreglo sistólico 2-D. Sólo se ha ilustrado un arreglo sistólico 2-D con el objeto de destacar las señales de conexión entre las unidades de control y dicho arreglo.

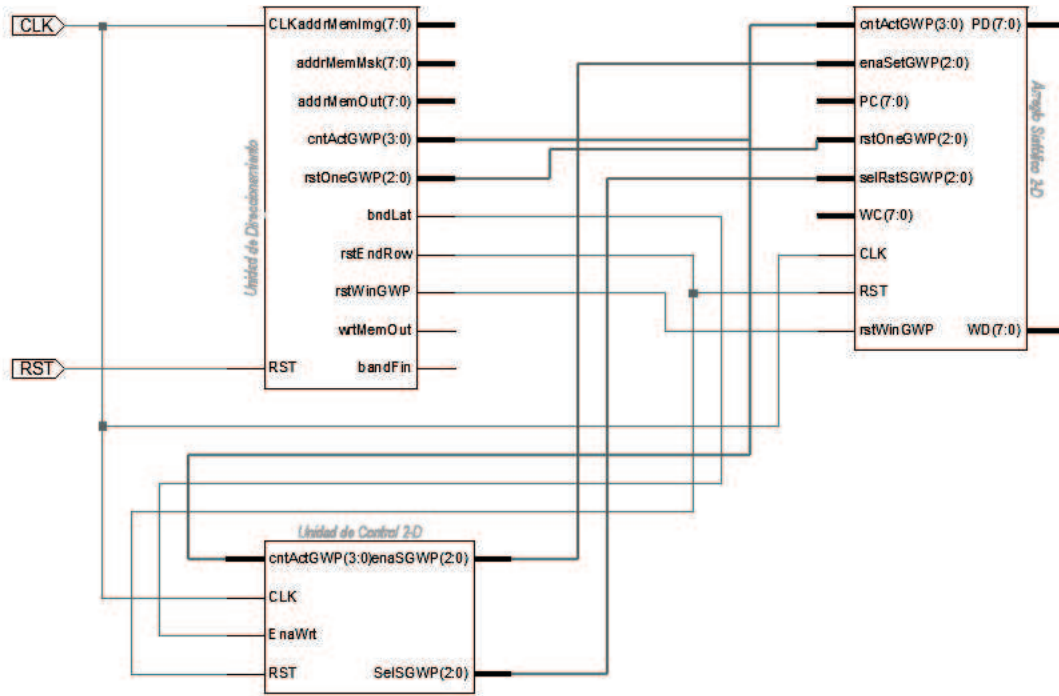


Figura 5.7: Conexión entre las unidades de control y un arreglo sistólico 2-D.

La *Unidad de Direccionamiento* genera señales de control tanto para la *Unidad de Control 2-D* como para el arreglo sistólico 2-D. A su vez, el arreglo sistólico 2-D recibe señales de control de ambas unidades de control.

Las señales de control que genera la *Unidad de Direccionamiento* para ambas unidades son: $cntActGWP$, $rstEndRow$. La señal $cntActGWP$ es un contador del número de píxeles a leer por columna de la imagen de entrada, es decir, $w + NR - 1$. La señal $rstEndRow$ indica la terminación de lectura por renglón de la imagen de entrada, es decir, el ancho de la imagen por $w + NR - 1$. Las señales de control dirigidas al arreglo sistólico 2-D son: $rstWinGWP$, $rstOneGWP$. La señal $rstWinGWP$ indica la ter-

minación de procesar una ventana de la imagen en una orientación, y junto con la señal *rstOneGWP* indica qué arreglo sistólicos 1-D se debe resetear para que se encuentre listo para procesar la siguiente ventana. La señal de control dirigida a la *Unidad de Control 2-D* es *bndLat* la cual indica que se ha cubierto el periodo de latencia.

La señal *enaSGWP* de la *Unidad de Control 2-D* habilita progresivamente cada uno de los arreglos sistólicos 1-D. Y la señal *SelSGWP* selecciona un arreglo sistólico 1-D para obtener el dato de salida.

Las señales de salida del arreglo sistólico 2-D son tanto el dato de resultado de haber procesado una ventana de la imagen de entrada (*PD*) como el dato de la máscara que ha terminado su uso (*WD*).

Para ver el comportamiento de estas señales con mayor detalle, se retoma el ejemplo planteado al inicio del capítulo, mostrando diagramas de tiempo. Con los diagramas de tiempo, hasta este momento, sólo se ha dado seguimiento al procesamiento completo de un arreglo sistólico 1-D hasta entregar sus valores de resultado al *Colector de Datos 1-D*. Este primer arreglo sistólico 1-D cubre los datos de la primera a la séptima columna de la imagen 13x13. Cuando se lee la segunda columna de la imagen 13x13, se inician actividades de un segundo arreglo sistólico 1-D. Esto se ilustra en la Fig. 5.8. Este segundo arreglo sistólico 1-D involucra los datos de la segunda columna a la octava de la imagen 13x13. De igual manera se van creando los demás arreglos sistólicos 1-D, siendo siete en total.

En la Fig. 5.8 se observa que en la lectura de la segunda columna de la imagen de entrada, el segundo arreglo sistólico 1-D, etiquetado como $1D_1$, inicia sus actividades, tomando los valores de los coeficientes de la máscara de la línea de retardo del arreglo sistólico anterior. Mientras que el primer arreglo sistólico 1-D, etiquetado como $1D_0$, continúa su proceso tomando los datos de la máscara directamente de la memoria de entrada.

Lo anterior se puede observar con los primeros procesadores de cada arreglo sistólico 1-D, etiquetados como $1D_0/x_gwp0$ y $1D_1/x_gwp0$, del primero y segundo arreglo sistólico 1-D respectivamente. En la lectura del valor del píxel 255 ubicado en la dirección 6175; el procesador $1D_0/$ toma dicho dato para ser multiplicado por el coeficiente de la máscara, cuyo valor es 3; el resultado es acumulado en $1D_0/acc_gwp0$ dando 13515 como el total acumulado. Mientras que el procesador del segundo arreglo sistólico 1-D

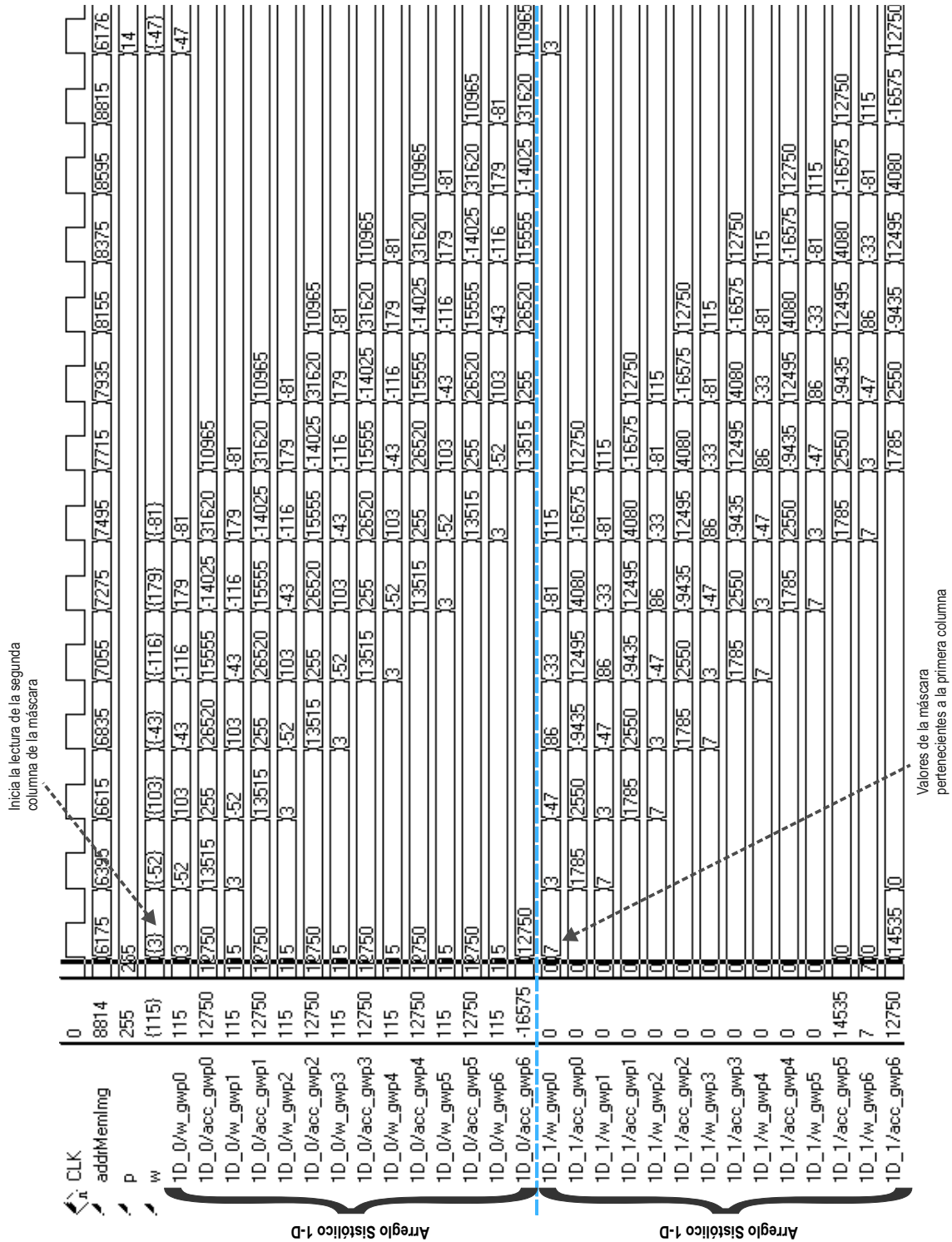


Figura 5.8: Diagrama de tiempo de dos arreglos sistólicos 1-D, cuando el segundo de ellos inicia su actividad.

recibe, de la línea de retardo del primer arreglo sistólico 1-D, el valor 7 como coeficiente de la máscara, y junto con el valor 255 correspondiente al píxel de la imagen, será el primer par de datos a calcular por parte de dicho procesador.

En la Fig. 5.9 se presentan dos colectores de datos 1-D; cuyos datos se almacenan en el *Colector de Datos 2-D*, etiquetado como $2D_0/px_D$. El tamaño del arreglo $2D_0/px_D$ es igual al número de arreglos sistólicos 1-D que comprendan el arreglo sistólico 2-D. En este caso, el arreglo sistólico 2-D contiene 7 arreglos sistólicos 1-D. La señal $rstWinGWP$ ha detectado el final del procesamiento de varias ventanas. Dado que los $GWPs$ del primer arreglo sistólico 1-D terminan de procesar la ventana que les fue asignada, el *Colector de Datos 2-D* toma los resultados que contiene el *Colector de Datos 1-D* del arreglo sistólico 1-D, ubicándolo en la primera posición de $2D_0/px_D$. Se puede ver que la señal $2D_0/SelSGWP$ determina qué arreglo sistólico 1-D está listo para entregar datos, y unido a la señal $rstOneGWP$, indica qué GWP se necesita iniciar sus valores para quedar listo cuando se inicia su actividad en el procesamiento de otra ventana. Estas dos señales son generadas por la *Unidad de Control 2-D* y la *Unidad de Direccionamiento*, respectivamente, como se ha mencionado anteriormente. Una vez que el segundo arreglo sistólico 1-D se encuentra con los resultados de sus procesadores, estos valores son colocados en la segunda posición del arreglo $2D_0/px_D$, y esto se repite para el resto de los arreglos sistólicos 1-D. En la figura se destaca dicho comportamiento mediante flechas punteadas. La señal $addrMemOut$ indica la dirección en donde se ubicará el resultado de salida cuyos valores se encuentran en el *Colector de Datos 2-D*. Este proceso se repite por toda la imagen de entrada, obteniendo una imagen de salida.

Finalmente, los valores obtenidos del procesamiento de un filtro sobre la imagen 13x13 son los siguientes:

$$\begin{bmatrix} -1388 & 19742 & 13493 & 13289 & -8844 & 22594 & -263 \\ 30468 & 11083 & 16627 & -6402 & 20188 & 770 & 10332 \\ 18828 & 42775 & -19789 & 19643 & 7537 & 7113 & 4936 \\ 13960 & 7546 & 27902 & -4180 & 10075 & 5541 & 12102 \\ 6687 & 9970 & 8764 & 22327 & -11356 & 17357 & 5683 \\ 8009 & 18519 & -2267 & -1430 & 30073 & -8055 & 8224 \\ 13854 & 13454 & -3061 & 2198 & 3657 & 20578 & -5747 \end{bmatrix} \quad (5.4)$$

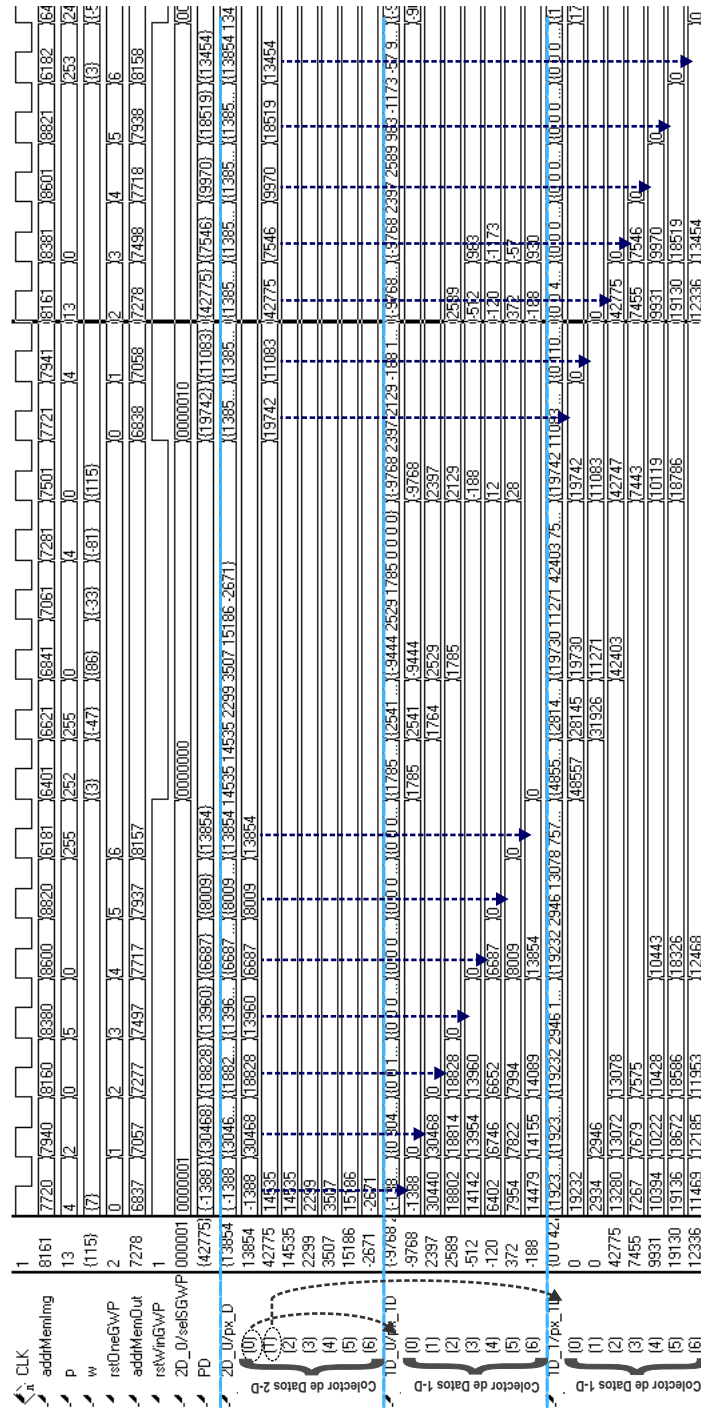


Figura 5.9: Diagrama de tiempo de un arreglo sistólico 2-D, almacenando los datos recolectados de dos colectores de datos 1-D en el *Colector de Datos 1-D*.

Las posiciones de almacenamiento en la imagen de salida corresponden a las siguientes direcciones.

$$\begin{bmatrix} 6837 & 6838 & 6839 & 6840 & 6841 & 6842 & 6843 \\ 7057 & 7058 & 7059 & 7060 & 7061 & 7062 & 7063 \\ 7277 & 7278 & 7279 & 7280 & 7281 & 7282 & 7283 \\ 7497 & 7498 & 7499 & 7500 & 7501 & 7502 & 7503 \\ 7717 & 7718 & 7719 & 7720 & 7721 & 7722 & 7723 \\ 7937 & 7938 & 7939 & 7940 & 7941 & 7942 & 7943 \\ 8157 & 8158 & 8159 & 8160 & 8161 & 8162 & 8163 \end{bmatrix} \quad (5.5)$$

En la siguiente sección se presenta la interfaz simplificada de la Arquitectura Sistólica 3-D y su funcionamiento con diagramas de tiempo.

5.5. Arquitectura Sistólica 3-D

A partir de la Arquitectura Sistólica 2-D se hace la extensión a 3-D creando instancias de los arreglos sistólicos 2-D y haciendo las conexiones de control necesarias para manejar este nuevo arreglo.

En la Tabla 5.2 se muestra la interfaz de la Arquitectura Sistólica 3-D, partiendo del caso de procesar ocho máscaras Gabor-2D 7×7 , con siete procesos paralelos en la dirección vertical, y siete en la dirección horizontal.

Básicamente las señales que comprende la interfaz de la arquitectura son las direcciones a los bancos de memoria, el píxel de la imagen, los ocho coeficientes correspondientes a las ocho filtros a aplicar, y los ocho datos de resultado por cada ventana procesada; además de las señales de reloj, reset y fin del procesamiento completo.

Para mostrar las conexiones existentes entre las unidades de control y un arreglo sistólico 3-D se muestra la Fig. 5.10. En esta figura se muestra un arreglo sistólico 3-D compuesto por tres arreglos sistólicos 2-D.

El objetivo de la Figura 5.10 es mostrar un acercamiento de las conexiones de las señales de control, generadas por la *Unidad de Control 2-D* y la *Unidad de Direccionalidad*, con los arreglos sistólicos 2-D. Como se observa, comparten las mismas señales

Puerto	Modo	Tamaño	Descripción
CLK	In	1	Señal del reloj
RST	In	1	Señal de reset para iniciar los valores internos de las unidades
memImg_P	In	8	Píxel de la imagen de entrada
Msk_0	In	10	Coficiente de la máscara 0, orientación 0°
Msk_1	In	10	Coficiente de la máscara 1, orientación 45°
Msk_2	In	10	Coficiente de la máscara 2, orientación 90°
Msk_3	In	10	Coficiente de la máscara 3, orientación 135°
Msk_4	In	10	Coficiente de la máscara 4, orientación 180°
Msk_5	In	10	Coficiente de la máscara 5, orientación 225°
Msk_6	In	10	Coficiente de la máscara 6, orientación 270°
Msk_7	In	10	Coficiente de la máscara 7, orientación 315°
EP	In	1	Señal de estado: activo/no activo
Wd	Out	10	Coficiente de la máscara anterior
Pd	Out	8	Resultado del cálculo

Tabla 5.2: Interfaz Simplificada de la Arquitectura Sistólica 3-D.

que controlan el funcionamiento interno del arreglo sistólico 2-D y estos arreglos a su vez generan los datos de salida de acuerdo a la orientación a calcular. En el Apéndice A se muestra el esquema RTL de la Arquitectura Sistólica 3-D en la que se muestra una vista general de la organización de los módulos que comprende la arquitectura.

Finalmente, se presenta un diagrama de tiempo, Fig. 5.11, en el que se recolectan los resultados generados por los ocho arreglos sistólicos 2-D que componen el arreglo sistólico 3-D. La señal $3D/pd$ es la que recolecta temporalmente los datos de los colectores de datos 2-D. El tamaño de esta señal depende del número de filtros a aplicar, esto es, el índice del dato en el arreglo corresponde al número de arreglo sistólico 2-D de donde fue generado. El filtro que se planteó inicialmente en el capítulo corresponde al segundo filtro aplicado a la imagen de entrada. Los ocho datos contenidos en el arreglo $3D/pd$ son obtenidos al mismo tiempo, ya que los filtros son ejecutados en forma paralela. Estos

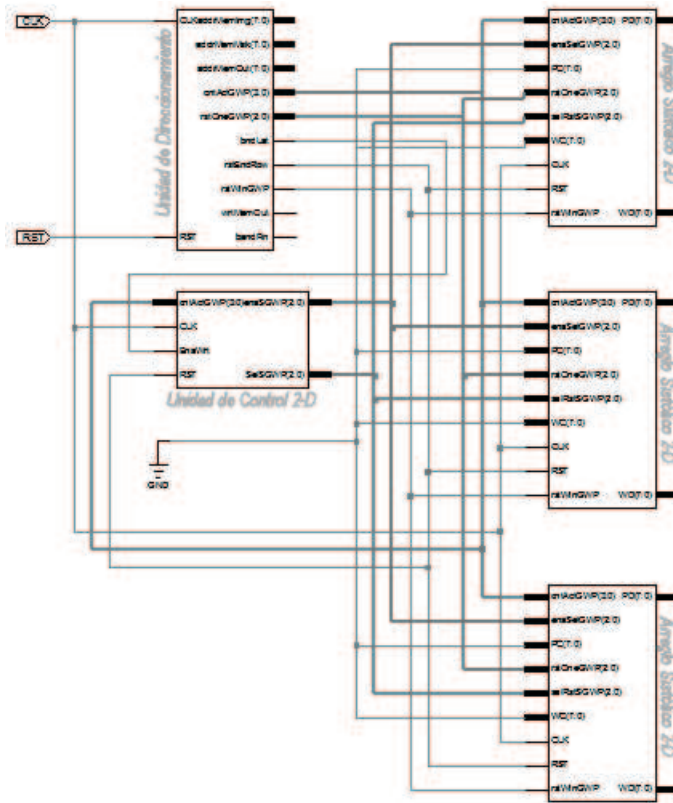


Figura 5.10: Esquema RTL de un arreglo sistólico 3-D compuesto por tres arreglos sistólicos 2-D conectado con las unidades de control.

resultados son almacenados en la memoria de salida, los cuales corresponden a la misma ubicación en la imagen de salida, determinada por *addrMemOut*. Tras aplicar los ocho filtros, se obtiene como resultado ocho imágenes de salida.

En el siguiente capítulo se presentan los resultados experimentales obtenidos de procesar imágenes $M \times N$ en escala de grises, aplicando 8 filtros Gabor-2D.

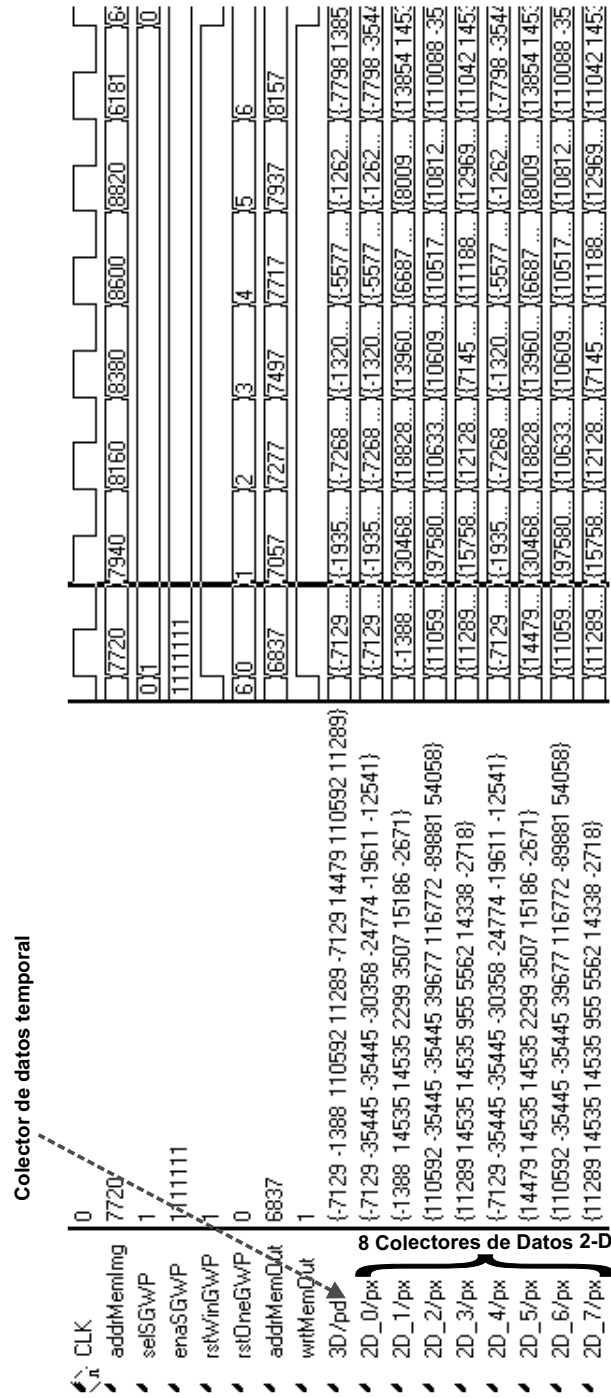


Figura 5.11: Diagrama de tiempo en el que se recolectan los datos de ocho arreglos sistólicos 2-D.

Capítulo 6

Resultados Experimentales

Para desarrollar y validar los resultados alcanzados con la Arquitectura Sistólica 3-D fue necesario realizar un flujo de diseño, el cual se muestra en la Figura 6.1.

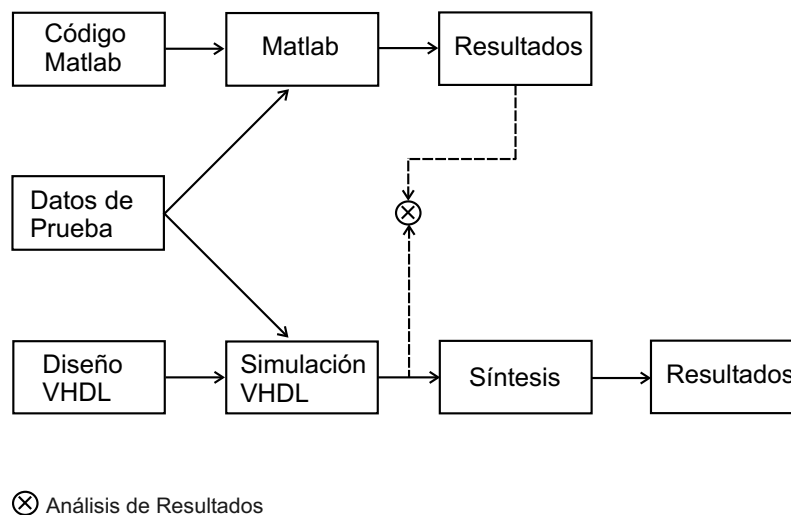


Figura 6.1: Diagrama a bloques del flujo de diseño de la Arquitectura Sistólica 3-D para validar los resultados.

El banco de pruebas del que se hace uso para validar los resultados obtenidos de la arquitectura es un programa realizado en MATLAB 7.0 el cual cuenta con varias tareas, entre las que destacan:

- Transformar las máscaras de Gabor, las cuales se encuentran en archivos de textos, en cadenas de bits con la implementación del punto fijo

- Transformar la imagen de entrada en un archivo de texto
- Transformar los archivos de textos que genera la simulación en archivos de imágenes bajo formato jpg
- Verificar los valores de salida con los valores esperados

Algunas imágenes utilizadas para realizar las pruebas se muestran en el Apéndice B.

6.1. Resultados de Simulación

Para probar y validar el modelo VHDL, se hicieron pruebas a nivel simulación. Para ello, fue necesario contar con archivos de texto como entradas (la imagen de entrada y las máscaras de Gabor) y el proceso genera archivos de texto (el número de ellos es de acuerdo al número de filtros aplicados).

La imagen de entrada se transforma a un archivo de texto, y las máscaras de Gabor se transforman a archivos de texto cuyos datos se encuentran en binario. Los archivos generados por la simulación son archivos de texto cuyos valores resultantes se encuentran en binario.

Para mostrar los resultados obtenidos, primero se presenta el desarrollo de aplicar un filtro sobre una imagen, y enseguida se exponen los resultados de los ocho filtros deseados mostrando los resultados obtenidos por software y por simulación en hardware. Para ejemplificar este proceso, se parte de la imagen mostrada en la Fig. 5.1 a la que se le aplicará un filtro mediante una máscara 7×7 . Los coeficientes de la máscara son representados en números binarios en punto fijo de 9 bits de los cuales ocho corresponden a la parte fraccionaria. Los valores obtenidos después de aplicar el filtro se almacenan en representación binaria de ocho bits.

La máscara 7×7 a aplicar es un filtro Gabor-2D con orientación en 45° . Los coeficientes de la máscara se muestran en la matriz 6.1.

$$\begin{bmatrix}
 0.028056 & 0.011407 & -0.18524 & 0.33725 & -0.13002 & -0.31745 & 0.44917 \\
 0.011407 & -0.20247 & 0.40289 & -0.16977 & -0.45306 & 0.70067 & -0.31745 \\
 -0.18524 & 0.40289 & -0.18556 & -0.54125 & 0.91491 & -0.45306 & -0.13002 \\
 0.33725 & -0.16977 & -0.54125 & 1 & -0.54125 & -0.16977 & 0.33725 \\
 -0.13002 & -0.45306 & 0.91491 & -0.54125 & -0.18556 & 0.40289 & -0.18524 \\
 -0.31745 & 0.70067 & -0.45306 & -0.16977 & 0.40289 & -0.20247 & 0.011407 \\
 0.44917 & -0.31745 & -0.13002 & 0.33725 & -0.18524 & 0.011407 & 0.028056
 \end{bmatrix} \quad (6.1)$$

La máscara es transformada a su representación binaria en punto fijo, quedando como se muestra en la matriz 6.2:

$$\begin{bmatrix}
 0000001110 & 0000000110 & 1110100001 & 0010101101 & 1110111101 & 1101011101 & 0011100110 \\
 0000000110 & 1110011000 & 0011001110 & 1110101001 & 1100011000 & 0101100111 & 1101011101 \\
 1110100001 & 0011001110 & 1110100001 & 1011101011 & 0111010100 & 1100011000 & 1110111101 \\
 0010101101 & 1110101001 & 1011101011 & 0111111111 & 1011101011 & 1110101001 & 0010101101 \\
 1110111101 & 1100011000 & 0111010100 & 1011101011 & 1110100001 & 0011001110 & 1110100001 \\
 1101011101 & 0101100111 & 1100011000 & 1110101001 & 0011001110 & 1110011000 & 0000000110 \\
 0011100110 & 1101011101 & 1110111101 & 0010101101 & 1110100001 & 0000000110 & 0000001110
 \end{bmatrix} \quad (6.2)$$

Cuando se realiza esta transformación, la nueva máscara Gabor adquiere pérdidas ya que la cantidad de bits dedicados a la parte fraccionaria no es suficiente para representar el número real con toda precisión, pero sí logra ser suficiente para la representación de un número aproximado al original. En la sección 6.1.2 se presenta una gráfica sobre el número de bits dedicados a la parte fraccionaria.

Una vez que el proceso de simulación ha generado el archivo de salida, éste se transforma en un archivo con formato jpg. En la Fig. 6.2 se muestran tres imágenes como resultado del proceso en software, y resultado del proceso en simulación. En la figura, la imagen 6.2(a) es la salida obtenida por el procesamiento en software con los valores originales de la máscara; la figura 6.2(b) corresponde a la imagen de salida obtenida por el procesamiento en software con los valores de la máscara en su representación binaria de 10 bits en punto fijo; la figura 6.2(c) corresponde a la imagen generada por simulación

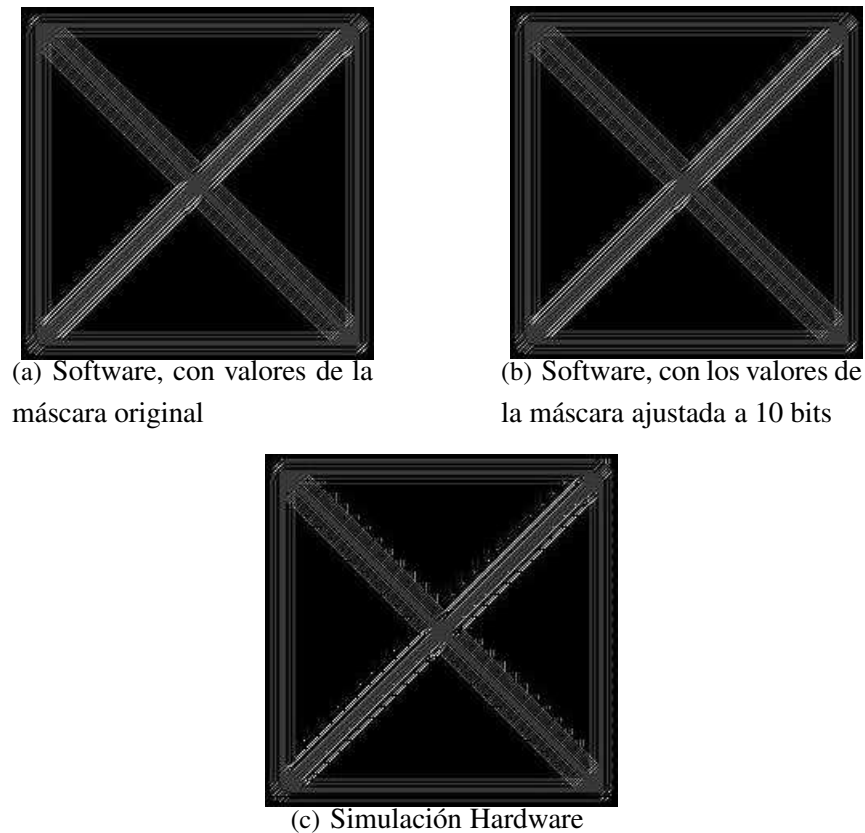


Figura 6.2: Imágenes de salida generadas por software y por simulación para detección de bordes orientados a 45° .

del modelado VHDL. El procesamiento por simulación genera un archivo de texto con los valores binarios de ocho bits.

Para verificar los valores obtenidos con los deseados, se procesa la imagen por software y por simulación. Con ello se comparan los resultados deseados (software), con los valores obtenidos de la simulación en VHDL. Para ejemplificar este procedimiento, se va a tomar una sub-imagen de la imagen de entrada. Para ser consistentes con el ejemplo que se ha venido trabajando desde el capítulo 5, la sub-imagen extraída es la imagen 13x13. Una vez que se ha aplicado el filtro a la imagen, tanto en software como en simulación, se obtiene el siguiente resultado.

En 6.3 se presentan dos matrices. La matriz izquierda corresponde a los valores obtenidos por software (con los valores de la máscara ajustados a 10 bits). La matriz derecha

corresponde a los valores obtenidos por simulación.

$$\begin{bmatrix} 0 & 77 & 53 & 52 & 0 & 88 & 0 \\ 119 & 43 & 65 & 0 & 79 & 3 & 40 \\ 74 & 167 & 0 & 77 & 29 & 28 & 19 \\ 55 & 29 & 109 & 0 & 39 & 22 & 47 \\ 26 & 39 & 34 & 87 & 0 & 68 & 22 \\ 31 & 72 & 0 & 0 & 117 & 0 & 32 \\ 54 & 53 & 0 & 9 & 14 & 80 & 0 \end{bmatrix} \quad \begin{bmatrix} 2 & 79 & 52 & 54 & 0 & 88 & 0 \\ 118 & 42 & 64 & 0 & 78 & 2 & 40 \\ 72 & 166 & 0 & 76 & 28 & 26 & 18 \\ 54 & 28 & 108 & 0 & 38 & 20 & 46 \\ 26 & 38 & 34 & 86 & 0 & 66 & 22 \\ 30 & 72 & 0 & 0 & 116 & 0 & 32 \\ 54 & 52 & 0 & 8 & 14 & 80 & 0 \end{bmatrix} \quad (6.3)$$

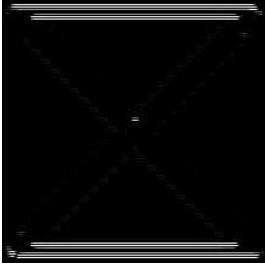
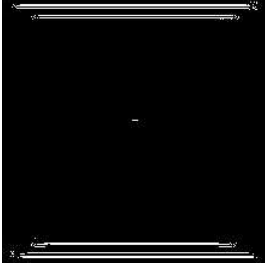
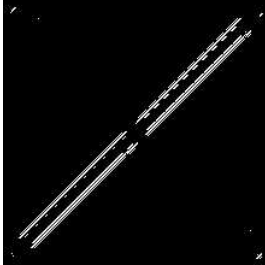
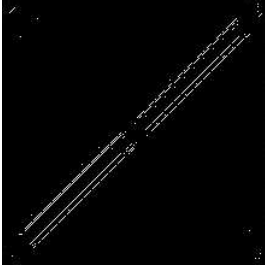
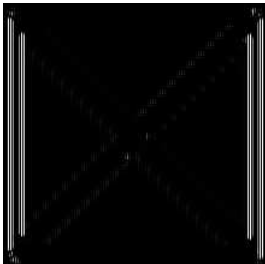
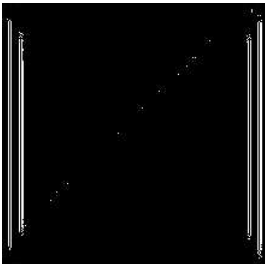
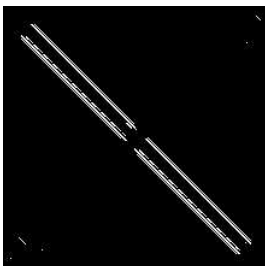
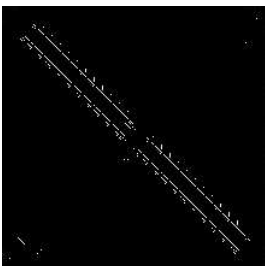
En la matriz 6.4 se muestra las diferencias entre los valores generados por software con la máscara con punto fijo menos los valores generados por simulación.

$$\begin{bmatrix} 2 & 2 & 1 & 2 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 2 & 1 & 0 & 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 0 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 & 0 & 2 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (6.4)$$

6.1.1. Resultados obtenidos de los ocho filtros

Partiendo de la imagen 220x220 de la Fig. 5.1, se le aplica 8 filtros Gabor-2D en las 8 orientaciones deseadas, cuyas máscaras son de 7x7. Se hace uso de esta imagen ya que resulta ilustrativa para el propósito en la detección de bordes en ciertas orientaciones. Este proceso se realiza en software y por simulación, obteniendo como resultado las imágenes mostradas en la Tabla 6.1.

En el Apéndice C se muestra el resultado de la imagen procesada por software (con la máscara original y ajustada a los 10 bits) y simulación; además de los valores de las máscaras utilizadas. En el Apéndice D y E, se muestran resultados realizados a diferentes imágenes.

Software	Simulación	Orientación
		0°
		45°
		90°
		135°

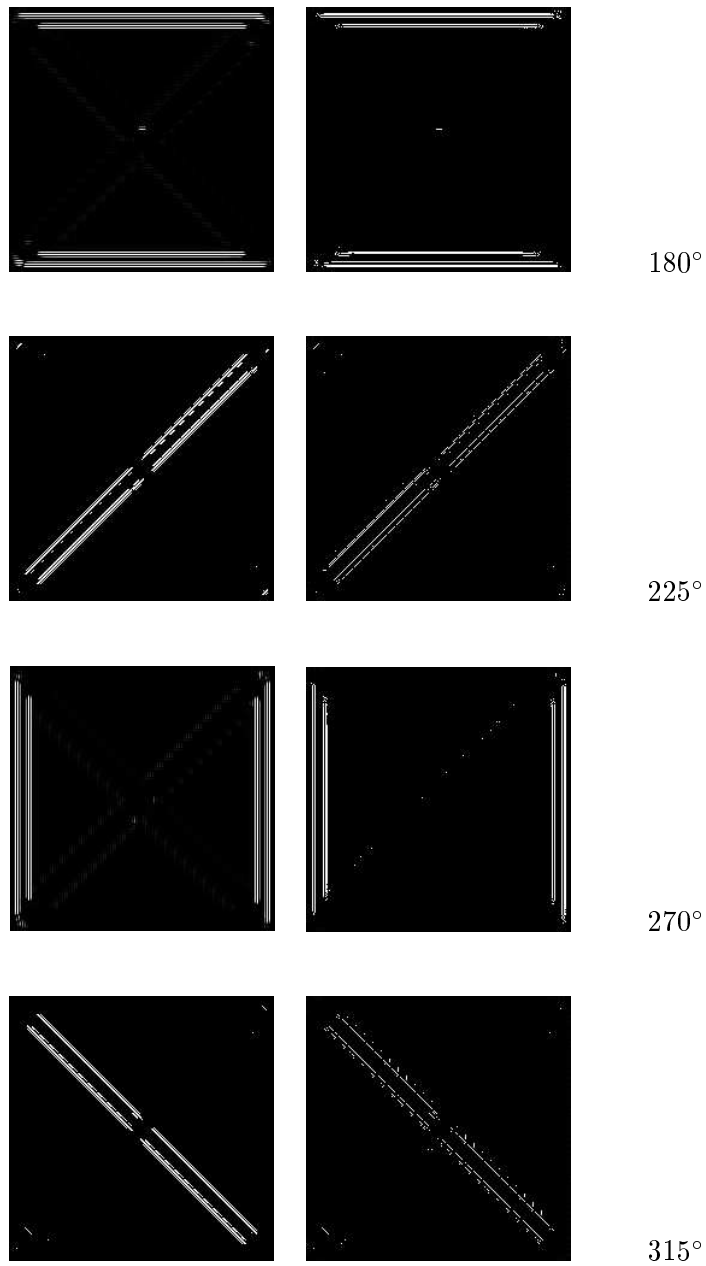


Tabla 6.1: Imágenes obtenidas tras aplicar el filtro Gabor-2D en software y en simulación, en las 8 orientaciones con la máscara ajustada a 10 bits.

6.1.2. Análisis de Precisión

Para presentar las imágenes de salida como el resultado del procesamiento, se realizaron pruebas hasta considerar las imágenes presentadas como el resultado del sistema. Haciendo un balance en recursos y de acuerdo a los resultados obtenidos, se tomaron decisiones sobre la precisión de algunos datos, tal es el caso de los coeficientes de la máscara.

Dado que los valores de las máscaras se encuentran entre el rango $[-1,1]$, se consideró que para representarlo en binario sólo se requería dedicar un bit a la parte entera, siendo éste utilizado para la representación del signo, y 9 bits a la parte fraccionaria. En la presente sección se presentarán las diferencias halladas al variar el número de bits que se dedican a la representación de la parte fraccionaria de los valores de las máscaras.

Partiendo de los valores de la máscara mostrados en la matriz 6.1, éstos se transforman a binario de 10 bits, 9 bits dedicados para la parte fraccionaria. En la matriz 6.5 se muestra los valores de los coeficientes de la máscara a los que se aproximan en la representación en binario.

$$\begin{bmatrix} 0.0273 & 0.0117 & -0.1855 & 0.3379 & -0.1309 & -0.3184 & 0.4492 \\ 0.0117 & -0.2031 & 0.4023 & -0.1699 & -0.4531 & 0.7012 & -0.3184 \\ -0.1855 & 0.4023 & -0.1855 & -0.5410 & 0.9141 & -0.4531 & -0.1309 \\ 0.3379 & -0.1699 & -0.5410 & 0.9980 & -0.5410 & -0.1699 & 0.3379 \\ -0.1309 & -0.4531 & 0.9141 & -0.5410 & -0.1855 & 0.4023 & -0.1855 \\ -0.3184 & 0.7012 & -0.4531 & -0.1699 & 0.4023 & -0.2031 & 0.0117 \\ 0.4492 & -0.3184 & -0.1309 & 0.3379 & -0.1855 & 0.0117 & 0.0273 \end{bmatrix} \quad (6.5)$$

Como se puede notar, existen diferencias entre cada valor de la máscara de la matriz ajustada a 10 bits, matriz 6.5, con la máscara original, matriz 6.1. A medida que se dedican más bits a la parte fraccionaria, así también los valores de los coeficientes son más precisos, y viceversa.

Tomando los valores obtenidos con sus diferencias, se muestra en la Fig. 6.3 una razón de error hallada que dependen del número de bits que se ocupan para formar un valor de la máscara.

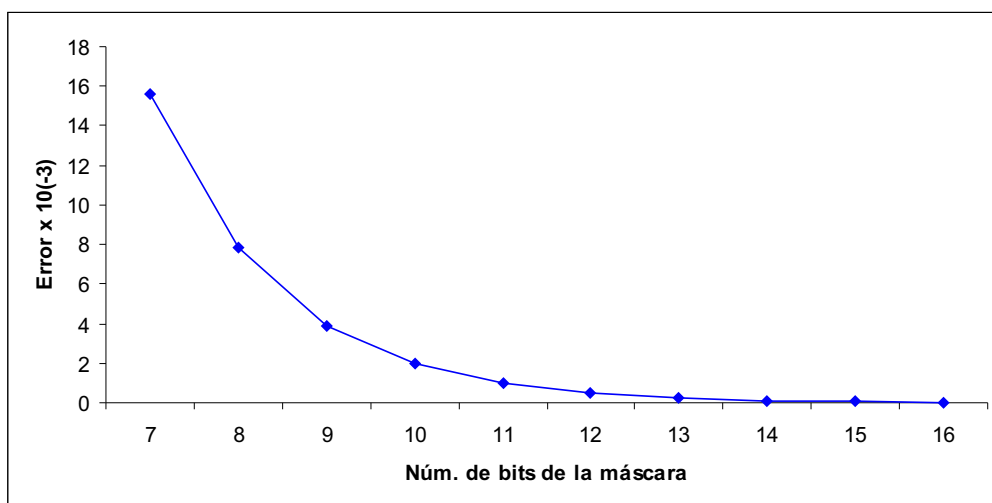


Figura 6.3: Relación de error con respecto al número de bits que componen los coeficientes de la máscara.

Como se puede observar, 16 bits para los valores de las máscaras sería el valor óptimo a utilizar ya que no habría error en la representación binaria de los valores de los coeficientes, el problema que conlleva esta elección son los recursos en un dispositivo con los que se debería contar para que se implemente esta elección. Se eligió 10 bits para la máscara, ya que se invierte lo suficiente en recursos para obtener resultados satisfactorios.

Para el caso de los datos de salida de los procesadores, cuenta con 19 bits. Se podría obtener como resultado el valor completo (parte entera y real) que se ha ido acumulando en los procesadores. Pero como un valor de píxel en escala de grises se encuentra en los valores enteros positivos del rango de 0 a 255, sólo basta obtener 8 bits del valor de salida, y esto corresponde a la parte entera del valor acumulado en los procesadores.

6.2. Resultados de Síntesis

En la Tabla 6.2 se observa el resultado de la síntesis de la arquitectura completa para aplicar 8 filtros Gabor-2D de máscaras 7×7 , cuyos datos comprenden 10 bits con 9 de ellos dedicados a la parte fraccionaria en representación en punto fijo. El dispositivo es un XCV3200E-6 FPGA y la síntesis se realizó sin jerarquía.

Se ha elegido un dispositivo con 3 millones 200 mil compuertas equivalentes por las

Módulo	Núm. de Slices	LUT's de 4 entradas	Núm. de Flip-Flops	Frecuencia (MHz)
GWP	65	118	19	190.367
Arreglo Sistólico 1-D	563	881	298	61.046
Arreglo Sistólico 2-D	3951	6192	2139	61.046
Arreglo Sistólico 3-D	31308	49471	16584	61.046
Unidad de Direccionamiento	221	381	169	32.703
Unidad de Control (2-D)	22	32	30	127.013
Arquitectura Completa	31563	49883	16811	32.703

Tabla 6.2: Resultado de la Síntesis de la Arquitectura Sistólica 3-D.

dimensiones en recursos que necesita la arquitectura. En el XCV3200E-6 ocupa 97 % de los slices que el dispositivo tiene disponibles (32448 slices).

6.3. Discusión de resultados

En la presente sección se exponen los resultados logrados en la construcción de una arquitectura que aplica 8 filtros espaciales a una imagen, realizando la detección de bordes en diferentes orientaciones. Las gráficas que se presentan fueron generadas con base a las ecuaciones de la sección 4.5.

Para realizar el cálculo de las ecuaciones, se parte del caso de contar con una imagen 512x512 a la que se le aplican 8 filtros Gabor-2D por medio de máscaras de dimensiones 7x7, con 7 ventanas procesadas en paralelo en la dirección vertical, y 7 ventanas procesadas en la dirección horizontal. El dato de la frecuencia a la que opera la arquitectura se toma de los resultados de síntesis obtenidos, el cual es de 32.703 MHz.

En la gráfica de la Figura 6.4 se muestra el rendimiento que presenta la arquitectura sistólica 3-D con diferente número de renglones procesados en paralelo. Se puede apreciar que entre mayor sea el número de estos renglones procesados en paralelo, menor es el tiempo de procesamiento empleado.

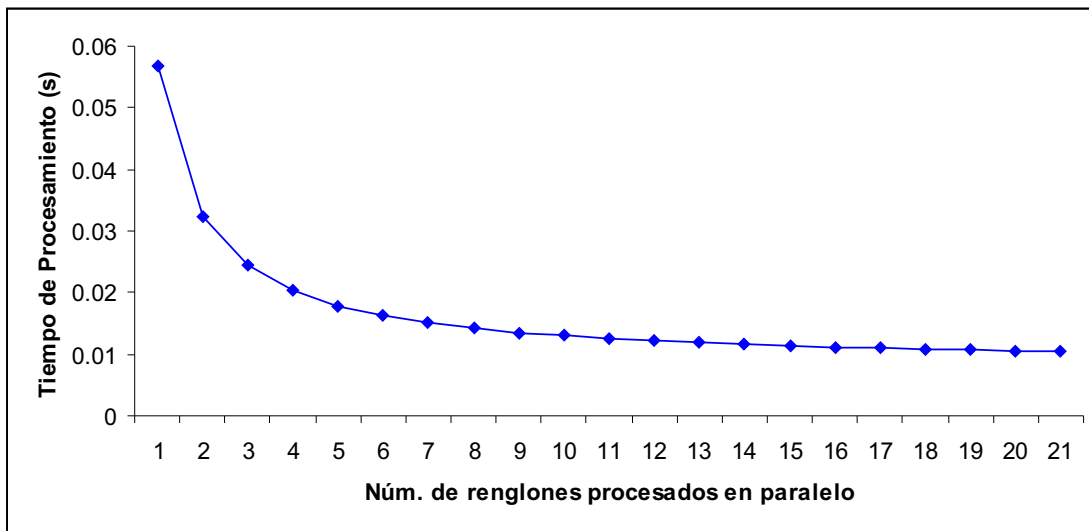


Figura 6.4: Rendimiento de la arquitectura con diferentes grados de paralelismo.

En la gráfica de la Figura 6.5 se muestran los requerimientos de recursos conforme se emplean mayor número de renglones procesados en paralelo. En este cálculo no se ha involucrando los recursos que requeriría la arquitectura completa, sólo al arreglo sistólico 3-D. La gráfica fue generada partiendo de la estimación simplificada en recursos hardware visto en la sección 4.5. Como se muestra en la gráfica, entre mayor sea el número de renglones procesados en paralelo, mayor son los recursos requeridos. Es por ello que se debe encontrar un balance entre el tiempo de procesamiento y recursos requeridos por la arquitectura.

Continuando con el caso particular del procesamiento de una imagen 512x512, en las siguientes figuras se presentan gráficas en las que se muestra el desempeño de la arquitectura al variar las dimensiones de la máscara. En la Figura 6.6 se presenta una gráfica del rendimiento de la arquitectura sistólica 3-D para cinco diferentes tamaños de máscaras. Se aprecia que entre mayor sea el tamaño de la máscara, mayor será el tiempo invertido en el procesamiento de la imagen; y que entre mayor sea el número de renglones procesados en paralelo menor será el tiempo empleado para procesar la imagen. En relación a esto último, en la gráfica se puede apreciar una tendencia que al emplear un mayor número de renglones procesados en paralelo no varía significativamente el tiempo

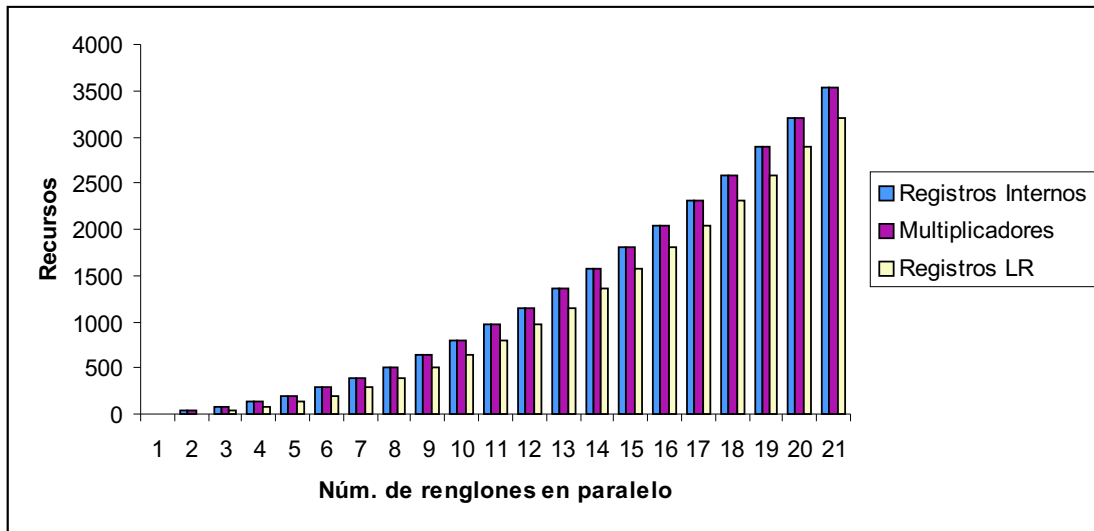


Figura 6.5: Requerimientos de recursos para el arreglo sistólico 3-D en diferentes grados de paralelismo.

de procesamiento.

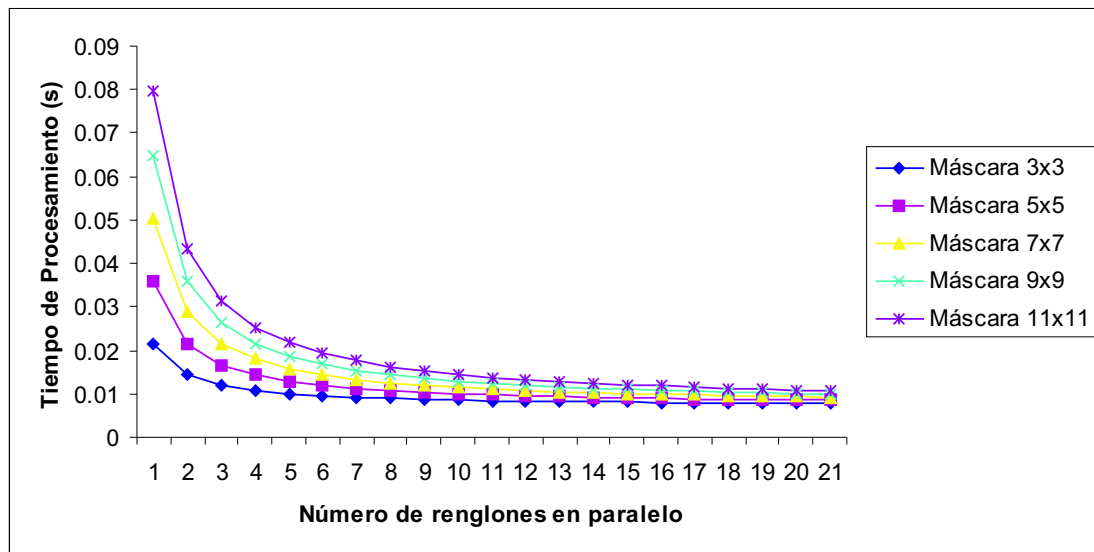


Figura 6.6: Tiempo de procesamiento de la arquitectura sistólica 3-D para diferentes tamaños de máscaras.

Continuando con el análisis del desempeño de la arquitectura para diferentes tamaños de máscaras, en la Figura 6.7 se muestra el desempeño de la arquitectura con base al número de imágenes procesadas por segundo. Como se puede notar, entre más pequeña sean las dimensiones de la máscara, mayor número de imágenes procesa por segundo, y éste número crece conforme se aumentan mayores niveles de procesamiento.

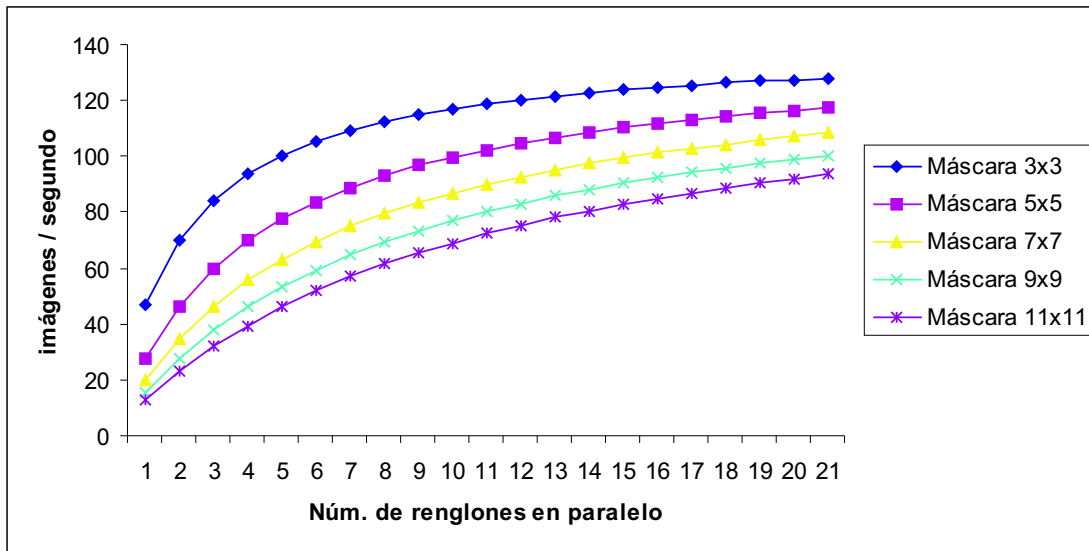


Figura 6.7: Requerimientos de recursos para el arreglo sistólico 3-D en diferentes grados de paralelismo.

Por lo anterior, se logra un desempeño de tiempo real, 30 imágenes por segundo, con sólo dedicar 4 renglones procesados en paralelo, en cualquier caso. Para el caso específico de una máscara 7x7 con 7 número de renglones procesados en paralelo, se procesa una imagen en 15 milisegundos, esto es 66.4 imágenes por segundo. Lo anterior corresponde al número de ciclos de reloj necesarios, visto en la sección 4.5, para procesar una imagen los cuales son 492,544 ciclos de reloj. Por lo anterior, la arquitectura propuesta logra cumplir con requerimientos de tiempo real. Si se requiere superar el tiempo de procesamiento con la arquitectura que se propone, se tiene la opción de elegir un dispositivo FPGA con mayores recursos que opere a mayor frecuencia, y con ello se puede lograr tiempo de procesamiento reducido. Además, la arquitectura es escalable, se puede procesar mayores filtros si el problema así lo requiere teniendo en cuenta el compromiso existente entre

recursos y desempeño.

La arquitectura cuenta con 392 procesadores, basado en las ecuaciones de la sección 4.5. El número de etapas que comprende el pipeline son 49. Con el esquema de direccionamiento y reuso de datos, se obtiene una arquitectura que muestra un buen desempeño logrando que un píxel de la imagen de entrada se lea a lo más dos veces.

Partiendo del caso de una máscara 7×7 , si se determina una máscara de menores dimensiones perdería la precisión del objetivo del filtro dado que cuenta con pocos elementos que reflejen las características de éste. De acuerdo a los resultados, una máscara 7×7 contiene mayores características del filtro que lo hacen actuar de una manera más aproximada al resultado esperado, la detección de bordes en ciertas orientaciones. Máscaras de tamaño 9×9 , 11×11 resaltan aún más las líneas en cierta orientación. La ventaja de utilizar máscaras grandes es que los errores producidos por efectos del ruido pueden minimizarse, o en el caso de contar con imágenes sin mucho contraste, pueden presentar resultados más efectivos; la contraparte a esto es que entre mayor sea las dimensiones de la máscara mayor es el costo computacional involucrado, como se ha visto. En el procesamiento de imágenes en general, la obtención de buenos resultados, al aplicar un filtro a una imagen, depende de las características de ésta última.

En el trabajo de [14] se diseña e implementa una arquitectura FPGA, la cual está basada en un arreglo sistólico 2D de procesadores de ventanas configurable 7×7 . Una imagen es procesada por máscaras de dimensiones 7×7 , pudiendo ser de mayores dimensiones si la aplicación así lo requiere. Los resultados que logra es de una frecuencia de 60 MHz, procesando una imagen 512×512 en niveles de gris en 8.35 milisegundos, esto es 119.7 imágenes por segundo; y con ello logrando tiempo real. La arquitectura fue implementada en un dispositivo FPGA XCV2000E-6 VirtexE, en la que ocupa 6118 slices, 10734 LUTs de cuatro entradas, y 1604 Flip-Flops, ocupando aproximadamente el 30 % de los recursos disponibles.

Los resultados presentados en las gráficas son consistentes con los resultados mostrados en [14], tomando en cuenta que se está aplicando un filtro a una imagen, que en nuestro caso son 8 filtros. Los resultados del presente trabajo se encuentran por debajo de los alcanzados por [14], siendo una de las principales razones la programación VHDL de los módulos que comprenden la arquitectura. En la Tabla 6.2 se puede observar que el módulo *Unidad de Direccionamiento* es el que determina la frecuencia de reloj a la que

trabajarla arquitectura, siendo éste uno de los módulos a mejorar para obtener un mejor desempeño por parte de la arquitectura.

Bajo las condiciones presentadas en el trabajo [16], una imagen 128x128 a la que se le aplican 8 filtros mediante máscaras 7x7, con la frecuencia de reloj reportado en la Tabla 6.2, se tiene un tiempo de procesamiento de 0.97 milisegundos.

En el trabajo de múltiples chips analógicos de Shimonomura [1] se logran requerimientos de tiempo real, 60 imágenes por segundo, dado que el número de neuronas (procesadores) existentes corresponde al mismo número de píxeles que contiene la imagen de entrada. De esta forma, cada procesador se dedica exclusivamente a un área específica de la imagen de entrada, y esto lo realiza en forma paralela. En este trabajo se realiza el cálculo en tres orientaciones: 0° , 60° y 120° . La respuesta a los filtros aplicados en las diferentes orientaciones no se obtiene de manera simultánea. Por otra parte, se tienen desventajas en el trabajo analógico cuando se habla de flexibilidad y escalamiento, ya que por la naturaleza de estos sistemas requiere más tiempo de construcción, y en consecuencia en el diseño de sistemas aún más complejos partiendo de este modelo, además de tener una dependencia a los componentes.

Capítulo 7

Conclusiones y Trabajo Futuro

7.1. Conclusiones

En el presente trabajo se ha desarrollado una arquitectura FPGA para el filtrado de imágenes en múltiples orientaciones mediante la aplicación de filtros Gabor-2D, cumpliendo con requerimientos de tiempo real.

La arquitectura logra el objetivo planteado mediante la extensión de una arquitectura sistólica 2-D a 3-D, mediante la adecuación necesaria para aplicar múltiples filtros Gabor-2D a una imagen, siendo los filtros Gabor-2D a aplicar mediante máscaras de convolución. La técnica utilizada es una arquitectura sistólica la cual permite un arreglo de procesadores actuando en paralelo, teniendo comunicación entre los procesadores mediante el paso de los coeficientes de la máscara de manera regular y síncrona, además de hacer uso de la técnica pipeline la cual cuenta con el número de etapas de acuerdo al número total de coeficientes de la máscara a procesar.

La arquitectura sistólica 3-D es una arquitectura flexible y escalable. Flexible para adecuar los parámetros del sistema a casos específicos, además de poder sintetizar la arquitectura a otro dispositivo FPGA. La arquitectura es escalable, ya que se puede dimensionar tanto en tamaños de máscara, número de filtros a aplicar, tamaño de imagen a procesar; tomando en cuenta compromisos entre el desempeño que se quiere del sistema en balance con los recursos disponibles del FPGA.

La arquitectura diseñada utiliza un adecuado direccionamiento de los datos, permitien-

do de esta forma utilizar el paralelismo inherente de los datos y disminuyendo el número de accesos a memoria. Con este diseño es posible la lectura de un píxel de la imagen de entrada a lo más dos veces.

El modelo conceptual del comportamiento y organización de las neuronas simples de la corteza visual primaria fue construido bajo un enfoque digital. Los sistemas desarrollados bajo un enfoque digital presenta ventajas en la construcción de sistemas más complejos, ya que sus tiempos de diseño son más cortos, y se puede ajustar fácilmente a diferente hardware para cumplir con los requerimientos de recursos y desempeño.

Teniendo un buen conocimiento en la correspondencia entre el modelado VHDL y los requerimientos del algoritmo se puede realizar un modelo adecuado del algoritmo ganando en eficiencia/desempeño. Además, con el desarrollo de nuevos dispositivos FPGA con características más enriquecidas la elección del dispositivo de acuerdo a la aplicación y la satisfacción de los requerimientos se pueden ver beneficiados sobremanera. En este sentido, entre las mejoras que se puede hacer a la arquitectura es la re-programación del módulo *Módulo de Direccionamiento* que refleje un mejor uso de los recursos hardware y así mejorar el desempeño completo de la arquitectura.

Además, se puede mejorar los resultados de las imágenes mediante la elaboración de un pre-procesamiento aplicado a las imágenes de entrada, para destacar las características de interés, y así la aplicación del filtro sea más eficiente, por ejemplo aplicar un filtro de mediana 3x3 con el fin de reducir el ruido preservando los bordes, o aplicar mayor contraste.

En el trabajo se ha propuesto una arquitectura basada en FPGA como alternativa para el procesamiento de imágenes al que se le aplican múltiples filtros Gabor-2D cumpliendo con requerimientos de tiempo real. Con este funcionamiento se aproxima al comportamiento y organización de las neuronas simples de la corteza visual primaria, presentando una alternativa en la construcción de sistemas de visión computacional. Por lo anterior, la arquitectura que se propone muestra la factibilidad de un modelo digital que permite resolver problemas de percepción visual bajo un enfoque biológicamente inspirado.

Como se ha visto en el desarrollo del presente trabajo, el hardware reconfigurable es una buena opción para el prototipado e implementación de sistemas neuromórficos. En específico, la arquitectura desarrollada que refleja, de forma aproximada, la estructura y comportamiento de las neuronas en la corteza visual primaria, dando como resultado

imágenes procesadas con significado en tiempo real.

7.2. Trabajo Futuro

Un borde es una característica local que indica un cambio brusco en el nivel de gris (alto contraste local) que puede ser una frontera entre un objeto y su entorno, e indica los límites entre objetos superpuestos. Por tanto, son características importantes que nos muestran la estructura de una escena. Lo que significa que si podemos detectar los bordes de una escena con exactitud, los objetos que hay en ella se podrían segmentar. De hecho el sistema visual humano funciona basándose fundamentalmente en bordes. Es por ello que resulta de importancia algoritmos que proporcionen la detección de bordes ya que al ser la primera etapa sobre la que se sustentan niveles superiores de procesamiento, los resultados finales dependen de lo exacta y eficiente que sea. Partiendo de la arquitectura sistólica 3-D se pueden realizar diseños de arquitecturas más complejas para la adquisición de información más relevante, por ejemplo un sistema de detección de movimiento, o un sistema de detección de objetos (segmentación).

Ya que las imágenes digitales tienen características propias que hacen que un procesamiento pueda o no ser adecuado de aplicar, es necesario implementar un pre-procesamiento a la imagen de entrada para eliminar ruido existente en la imagen, u otorgarle alto contraste para que una vez procesada se obtenga mayor información de ella.

Siguiendo la línea de arquitecturas bio-inspiradas, se puede extender este sistema a otras etapas halladas en la corteza visual. Como en la sección 2.1 se mencionó, la corteza visual primaria cuenta con dos tipos de neuronas de acuerdo a su funcionamiento: neuronas simples y neuronas complejas. Como trabajo futuro, se puede implementar el funcionamiento de las neuronas complejas, las cuales detectan textura. Con lo anterior, se pueden contar con imágenes en las que se discrimine la textura y se identifiquen los objetos existentes en ella.

Bibliografía

- [1] Shimonomura K., Yagi T. *A Multichip a VLSI System Emulating Orientation Selectivity of Primary Visual Cortical Cells*, IEEE Transactions on Neural Networks, Volume: 16, No. 4, 2005.
- [2] Mead C, *Neuromorphic Electronic Systems*, IEEE, Volume: 78, No. 10, October 1990.
- [3] Hubel D.H., Wiesel T.N., *Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex*, J. Physiol., vol. 160, pp. 106-154, 1962.
- [4] Lindblad, Thomas; Kinser, Jason M.; *Image Processing using Pulse-Coupled Neural Networks*, Ed. Springer-Verlag London; Great Britain, 1998.
- [5] Ocean Y. H. Cheung, Philip H. W. Leong, Eric K. C. Tsang, and Bertram E. Shi; *Implementation of Gabor-type filters on field programmable gate arrays*, In Proc. International Conference on Field Programmable Technology (FTP), pp. 327-328, 2005.
- [6] Yu W., Merilla P.A., Arthur J.V., Boahen K. A., Shi B E., *Neuromorphic Implementation of Orientation Hypercolumns*, IEEE Transactions on Circuits and Systems I, Volume: 52, No. 6, June 2005.
- [7] Aznar-Casanova, *Análisis multiescala y multiorientación de imágenes mediante un banco de filtros de Gabor-2D*, Revista Cognitiva; Vol. 12, No. 2, pp. 223-246, 2002.
- [8] Shi B, *A low-power orientation-selective vision sensor*, IEEE Transactions on Circuits and systems II, Volume: 47, No. 2, February 2000.

- [9] Ros E., F. J. Pelayo, A. Prieto, *Ingeniería Neuromórfica: El papel del hardware reconfigurable*, Actas de las II Jornadas sobre Computación Reconfigurable y Aplicaciones (JCRA'2002), pp. 89-92, (ISBN: 84-699-9448-4), Almuñecar (Granada), 18-20 Septiembre 2002.
- [10] Tsai D-M, Lin C-P, Huang K-T, *Defect detection in coloured texture surfaces using Gabor Filters*, The Imaging Science Journal, Vol 53, pp. 27-37, 2005.
- [11] Ji Yiming, Chang Kai H., Hung Chi-Cheng, *Efficient Edge Detection and Object Segmentation Using Gabor Filters*, ACMSE Huntsville, Alabama, USA, pp. 454-459, April 2004.
- [12] Recio-Recio Jorge, Ruiz-Fernández Luis, Fernández-Sarriá Alfonso, *Use of Gabor filter for texture classification of digital images*, Departamento de Ingeniería Cartográfica, Geodesia y Fotogrametría, Universidad Politécnica de Valencia, 2005.
- [13] H. T. Kung, *Why Systolic Architectures?*, IEEE Computer, pp. 37-46, January 1982.
- [14] Torres-Huitzil César, *Reconfigurable Computer Vision System for Real-time Applications*, Tesis de Doctorado en Ciencias Computacionales, Departamento de Ciencias Computacionales, Instituto Nacional de Astrofísica, Óptica y Electrónica, Agosto 2003.
- [15] Torres-Huitzil César, Arias M, *FPGA-Based Configurable Systolic Architecture for Window-Based Image Processing*, EURASIP Journal on Applied Signal Processing, Hindawi Publishing Corporation, Vol. 2005, No. 7, pp. 1024-1034, 2005.
- [16] Torres-Huitzil César, Girau Bernard, Castellanos-Sánchez Claudio, *On-chip visual perception of motion: A bio-inspired connectionist model of FPGA*, Elsevier, Neural Networks 18, No. 5-6, pp. 557-565, 2005.
- [17] Tejeda Yépez Juan Carlos, *Arquitectura FPGA para Filtrado de Imágenes en Tiempo Real*, Tesis de Maestría en Ciencias en la especialidad de Electrónica, Instituto Nacional de Astrofísica, Óptica y Electrónica, 2001.
- [18] Moldovan, Dan I, *Parallel Processing: From Applications to Systems*, Ed. Morgan Kaufmann Publishers, 1993.

- [19] Selim G. Aki, *The Design and Analysis of Parallel Algorithms*, Ed. Prentice-Hall, 1989.
- [20] Sek Meng Chai, *Real Time Image Processing on Parallel Arrays for Gigascale Integration*, Tesis Doctoral, Department of Computer Science, University of Massachusetts, Septiembre 1994.
- [21] Keith Diefendorff, Pradeep K. Dubey, *How Multimedia Workloads Will Change Processor Design*, IEEE Computer, Vol. 30, Issue 9, pp. 43-45, Septiembre 1997.
- [22] Martin C. Herbordt, TomVanCourt, Yongfeng Gu, Bharat Sukhwani, Al Conti, Josh Model, Doug DiSabello, *Achieving High Performance with PFGA-Based Computing*, IEEE Computer Society, 2007.
- [23] Duncan Buell, Tarek El-Ghazawi, Kris Gaj, Volodymyr Kindratenko, *High-Performance Reconfigurable Computing*, IEEE Computer Society, 2007.
- [24] Chang K. C., *Digital Systems Design with VHDL and Synthesis: An Integrated Approach*, Ed. IEEE Computer Society, 1999.
- [25] Antonio Martínez, Samuel Romero, Eduardo Ros, Alberto Prieto, Francisco J. Pelayo, *Implementación en hardware reconfigurable de un modelo de retina*, proyecto CORTIVIS, Facultad de Ciencias de Granada.
- [26] Gonzalez Rafael C., Woods Richard E., *Digital Image Processing*, Ed. Prentice Hall, 2ª edición. 2002.
- [27] Pratt William K., *Digital Image Processing*, 2ª edición, 1991.
- [28] Jain Ramesh, Kasturi Rangachar, Schunk Brian G., *Machine Vision*, Ed. Mc Graw Hill, 1995.
- [29] Ballard Dana H., Brown Christopher M., *Computer Vision*, Ed. Prentice Hall, 1982.

Apéndice A

RTL de la Arquitectura Sistólica 3-D

La Figura A.1 es el esquema RTL de la Arquitectura Sistólica 3-D, la cual comprende ocho filtros aplicados a una imagen de entrada, cuyas dimensiones de la máscara son 7x7. Se muestra una vista general de la organización de los módulos que comprende la arquitectura. Se encuentran alineados los ocho arreglos sistólicos 2-D que componen el arreglo sistólico 3-D. Además, se tienen los módulos de control, la *Unidad de Direccionamiento* y *Unidad de Control 2-D*, conectados a los ocho arreglos sistólicos 2-D.

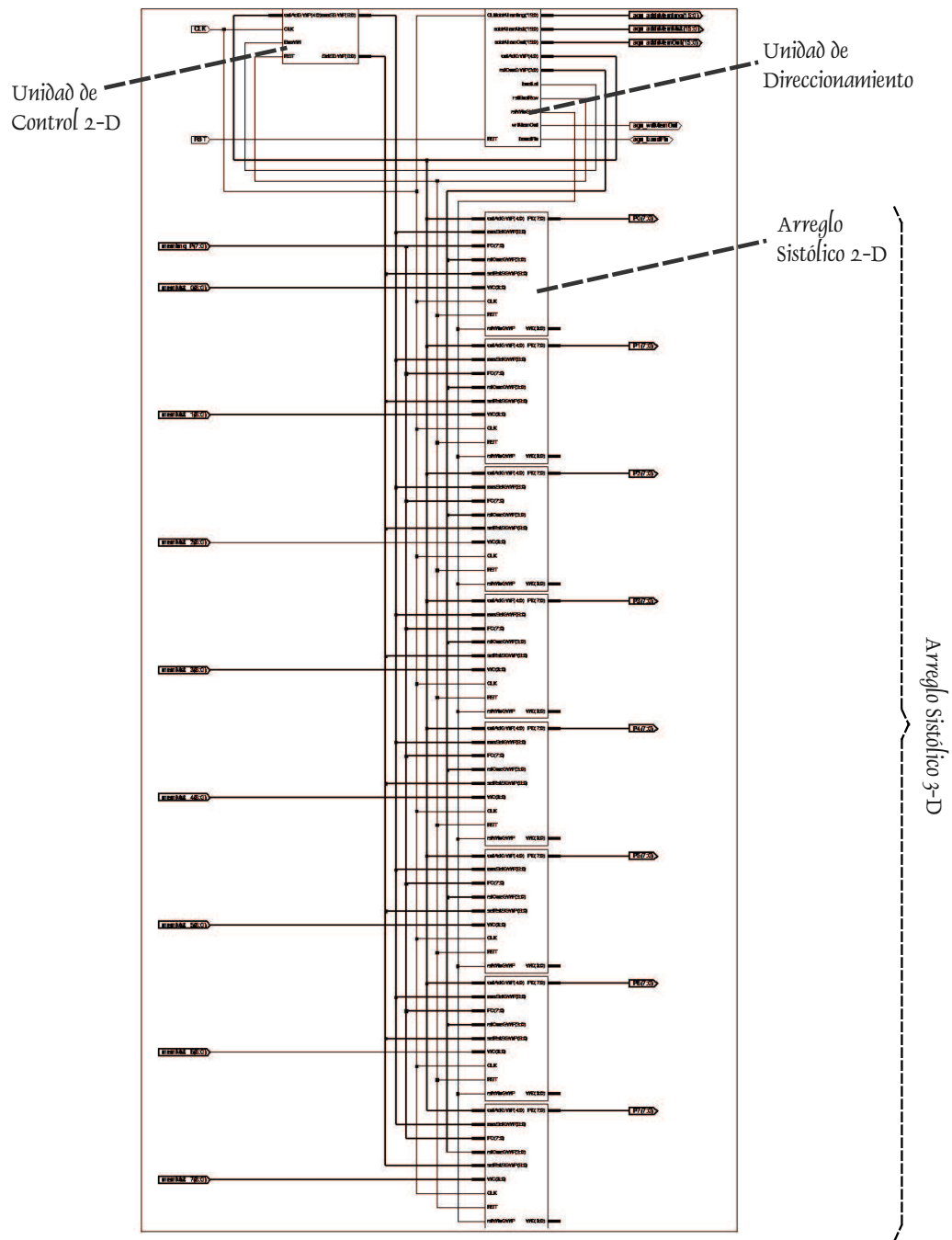
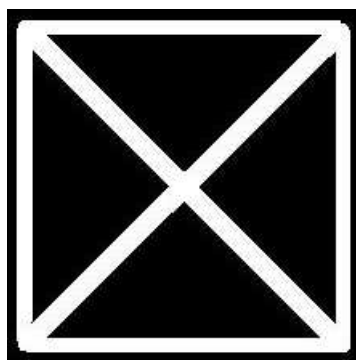


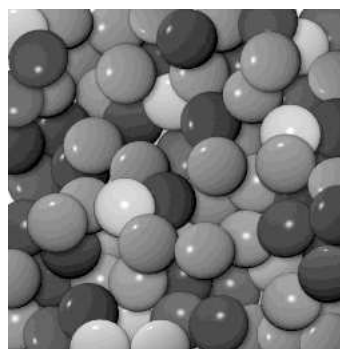
Figura A.1: Esquema RTL de la Arquitectura Sistólica 3-D.

Apéndice B

Imágenes de Prueba



(a) 220×220



(b) 256×256



(c) 192×192



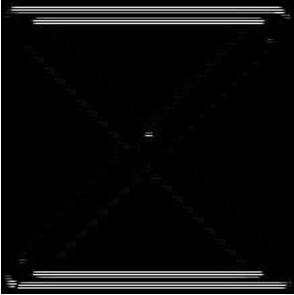
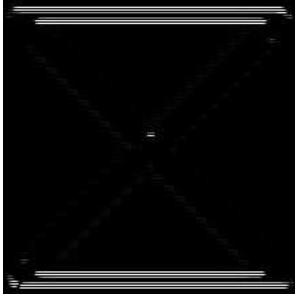
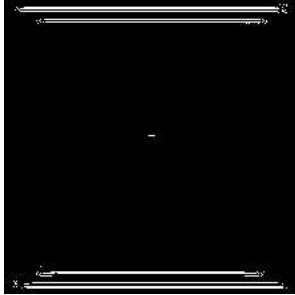
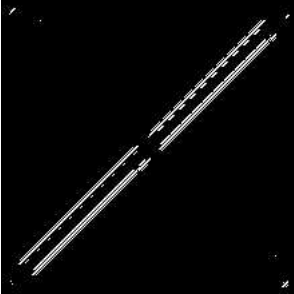
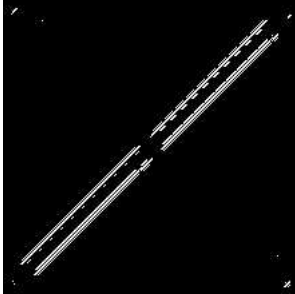
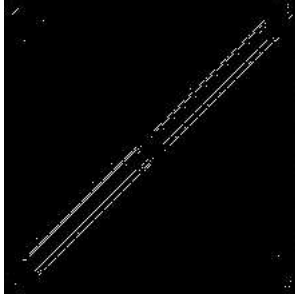
(d) 800×600

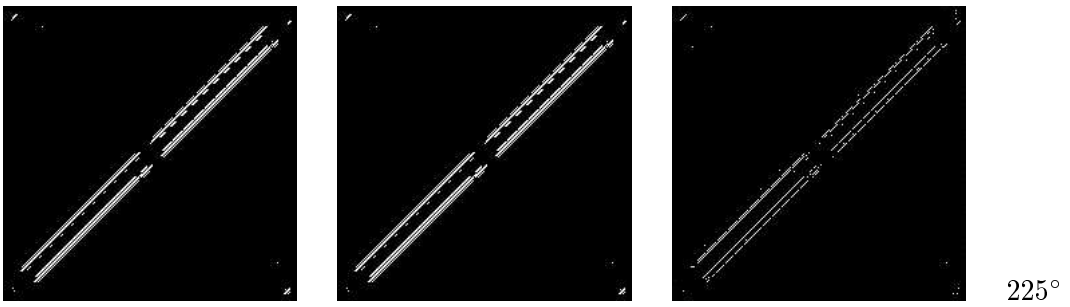
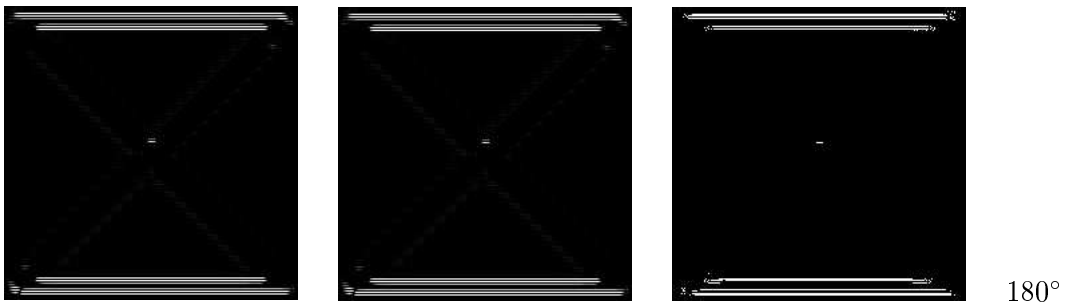
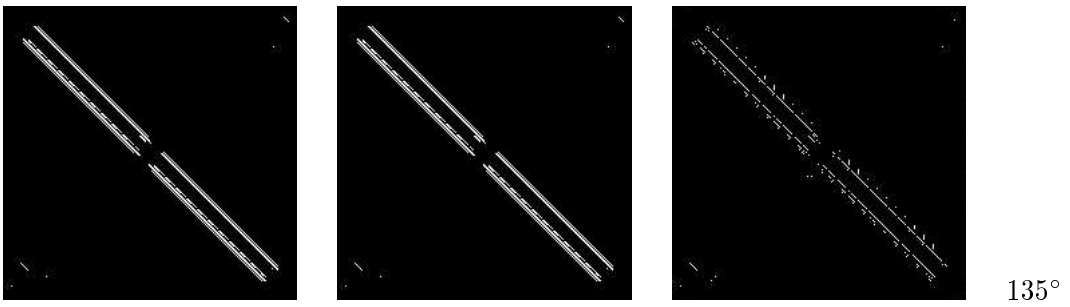
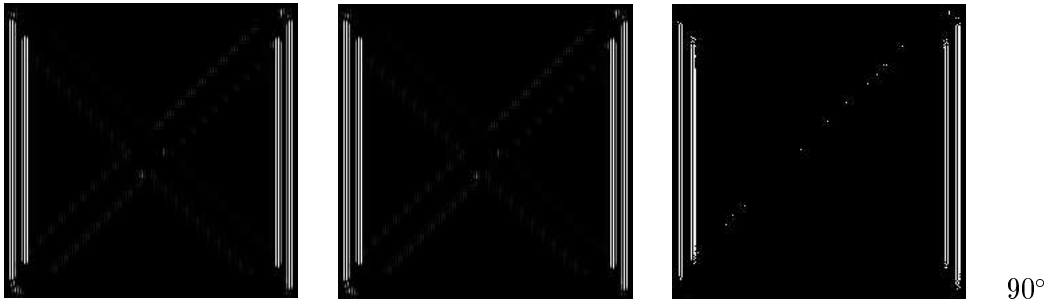
Figura B.1: Imágenes utilizadas para realizar pruebas con la arquitectura.

Apéndice C

Resultados de Máscaras 7x7

Las imágenes mostradas a continuación corresponden a las imágenes resultado obtenidas por software (con máscara 7x7 original y ajustada a 10 bits) y por simulación en las diferentes orientaciones.

Software (máscara original)	Software (máscara ajustada 10 bits)	Simulación HW (máscara ajustada 10 bits)	Orientación θ
			0°
			45°



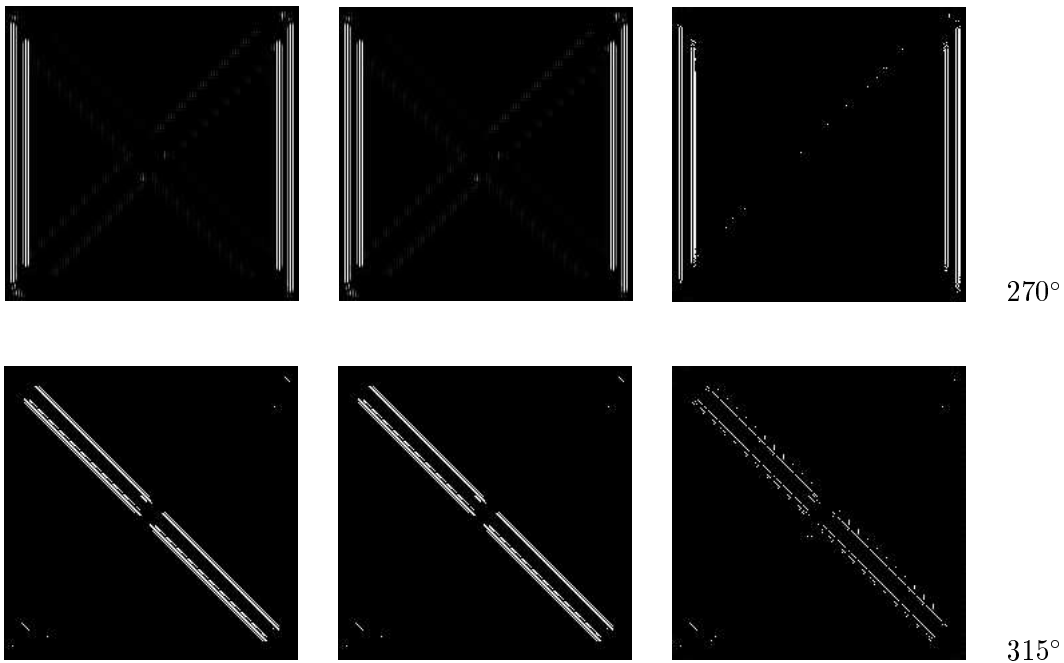


Tabla C.1: Imágenes obtenidas tras aplicar el filtro Gabor-2D en software y en simulación, en las 8 orientaciones con la máscara 7x7 ajustada a 10 bits.

La siguiente tabla muestra las máscaras 7x7 correspondientes al filtro Gabor-2D en las ocho orientaciones.

Máscara 7x7	θ
$\begin{bmatrix} -0.132 & -0.16486 & -0.18839 & -0.19695 & -0.18839 & -0.16486 & -0.132 \\ 0.3255 & 0.40658 & 0.46459 & 0.48572 & 0.46459 & 0.40658 & 0.32553 \\ -0.5595 & -0.6988 & -0.79852 & -0.83483 & -0.79852 & -0.6988 & -0.5595 \\ 0.6702 & 0.83706 & 0.95651 & 1 & 0.95651 & 0.83706 & 0.6702 \\ -0.5595 & -0.6988 & -0.79852 & -0.83483 & -0.79852 & -0.6988 & -0.5595 \\ 0.3255 & 0.40658 & 0.46459 & 0.48572 & 0.46459 & 0.40658 & 0.32553 \\ -0.132 & -0.16486 & -0.18839 & -0.19695 & -0.18839 & -0.16486 & -0.132 \end{bmatrix}$	0°

$$\begin{bmatrix}
 0.028056 & 0.011407 & -0.18524 & 0.33725 & -0.13002 & -0.31745 & 0.44917 \\
 0.011407 & -0.20247 & 0.40289 & -0.16977 & -0.45306 & 0.70067 & -0.31745 \\
 -0.18524 & 0.40289 & -0.18556 & -0.54125 & 0.91491 & -0.45306 & -0.13002 \\
 0.33725 & -0.16977 & -0.54125 & 1 & -0.54125 & -0.16977 & 0.33725 \\
 -0.13002 & -0.45306 & 0.91491 & -0.54125 & -0.18556 & 0.40289 & -0.18524 \\
 -0.31745 & 0.70067 & -0.45306 & -0.16977 & 0.40289 & -0.20247 & 0.011407 \\
 0.44917 & -0.31745 & -0.13002 & 0.33725 & -0.18524 & 0.011407 & 0.028056
 \end{bmatrix}$$

45°

$$\begin{bmatrix}
 -0.132 & 0.32553 & -0.5595 & 0.6702 & -0.5595 & 0.32553 & -0.132 \\
 -0.16486 & 0.40658 & -0.6988 & 0.83706 & -0.6988 & 0.40658 & -0.16486 \\
 -0.18839 & 0.46459 & -0.79852 & 0.95651 & -0.79852 & 0.46459 & -0.18839 \\
 -0.19695 & 0.48572 & -0.83483 & 1 & -0.83483 & 0.48572 & -0.19695 \\
 -0.18839 & 0.46459 & -0.79852 & 0.95651 & -0.79852 & 0.46459 & -0.18839 \\
 -0.16486 & 0.40658 & -0.6988 & 0.83706 & -0.6988 & 0.40658 & -0.16486 \\
 -0.132 & 0.32553 & -0.5595 & 0.6702 & -0.5595 & 0.32553 & -0.132
 \end{bmatrix}$$

90°

$$\begin{bmatrix}
 0.44917 & -0.31745 & -0.13002 & 0.33725 & -0.18524 & 0.011407 & 0.028056 \\
 -0.31745 & 0.70067 & -0.45306 & -0.16977 & 0.40289 & -0.20247 & 0.011407 \\
 -0.13002 & -0.45306 & 0.91491 & -0.54125 & -0.18556 & 0.40289 & -0.18524 \\
 0.33725 & -0.16977 & -0.54125 & 1 & -0.54125 & -0.16977 & 0.33725 \\
 -0.18524 & 0.40289 & -0.18556 & -0.54125 & 0.91491 & -0.45306 & -0.13002 \\
 0.011407 & -0.20247 & 0.40289 & -0.16977 & -0.45306 & 0.70067 & -0.31745 \\
 0.028056 & 0.011407 & -0.18524 & 0.33725 & -0.13002 & -0.31745 & 0.44917
 \end{bmatrix}$$

135°

$$\begin{bmatrix}
 -0.132 & -0.16486 & -0.18839 & -0.19695 & -0.18839 & -0.16486 & -0.132 \\
 0.3255 & 0.40658 & 0.46459 & 0.48572 & 0.46459 & 0.40658 & 0.32553 \\
 -0.5595 & -0.6988 & -0.79852 & -0.83483 & -0.79852 & -0.6988 & -0.5595 \\
 0.6702 & 0.83706 & 0.95651 & 1 & 0.95651 & 0.83706 & 0.6702 \\
 -0.5595 & -0.6988 & -0.79852 & -0.83483 & -0.79852 & -0.6988 & -0.5595 \\
 0.3255 & 0.40658 & 0.46459 & 0.48572 & 0.46459 & 0.40658 & 0.32553 \\
 -0.132 & -0.16486 & -0.18839 & -0.19695 & -0.18839 & -0.16486 & -0.132
 \end{bmatrix}$$

180°

0.028056	0.011407	-0.18524	0.33725	-0.13002	-0.31745	0.44917	225°
0.011407	-0.20247	0.40289	-0.16977	-0.45306	0.70067	-0.31745	
-0.18524	0.40289	-0.18556	-0.54125	0.91491	-0.45306	-0.13002	
0.33725	-0.16977	-0.54125	1	-0.54125	-0.16977	0.33725	
-0.13002	-0.45306	0.91491	-0.54125	-0.18556	0.40289	-0.18524	
-0.31745	0.70067	-0.45306	-0.16977	0.40289	-0.20247	0.011407	
0.44917	-0.31745	-0.13002	0.33725	-0.18524	0.011407	0.028056	

-0.132	0.32553	-0.5595	0.6702	-0.5595	0.32553	-0.132	270°
-0.16486	0.40658	-0.6988	0.83706	-0.6988	0.40658	-0.16486	
-0.18839	0.46459	-0.79852	0.95651	-0.79852	0.46459	-0.18839	
-0.19695	0.48572	-0.83483	1	-0.83483	0.48572	-0.19695	
-0.18839	0.46459	-0.79852	0.95651	-0.79852	0.46459	-0.18839	
-0.16486	0.40658	-0.6988	0.83706	-0.6988	0.40658	-0.16486	
-0.132	0.32553	-0.5595	0.6702	-0.5595	0.32553	-0.132	

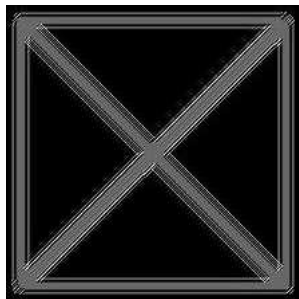
0.44917	-0.31745	-0.13002	0.33725	-0.18524	0.011407	0.028056	315°
-0.31745	0.70067	-0.45306	-0.16977	0.40289	-0.20247	0.011407	
-0.13002	-0.45306	0.91491	-0.54125	-0.18556	0.40289	-0.18524	
0.33725	-0.16977	-0.54125	1	-0.54125	-0.16977	0.33725	
-0.18524	0.40289	-0.18556	-0.54125	0.91491	-0.45306	-0.13002	
0.011407	-0.20247	0.40289	-0.16977	-0.45306	0.70067	-0.31745	
0.028056	0.011407	-0.18524	0.33725	-0.13002	-0.31745	0.44917	

Tabla C.2: Máscaras 7x7 de las diferentes orientaciones.

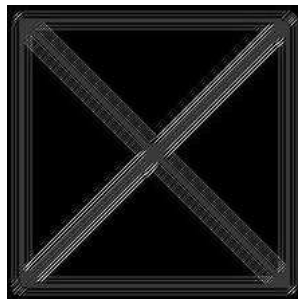
Apéndice D

Resultados de 3 tamaños de máscaras

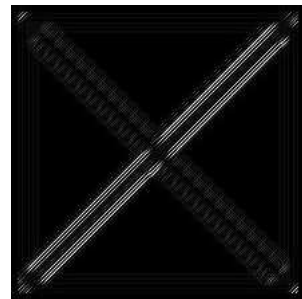
Se crean máscaras de convolución de tres tamaños (5x5, 7x7 y 11x11) partiendo de la función Gabor-2D cuya orientación es de 45° . Para obtener las imágenes de salida, los valores de resultado después de aplicar la máscara en la imagen, se normalizan. Las primeras tres imágenes se obtienen después de normalizar, y las siguientes tres imágenes se obtienen después de normalizar y aplicar un umbral de 180.



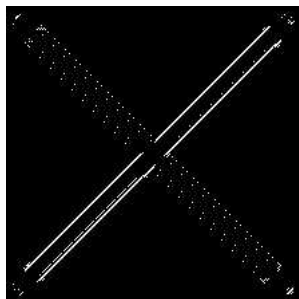
(a) Máscara 5x5, sin umbral



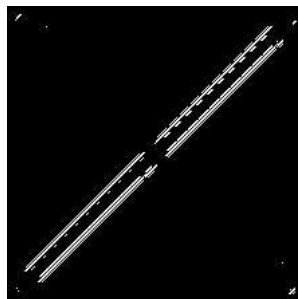
(b) Máscara 7x7, sin umbral



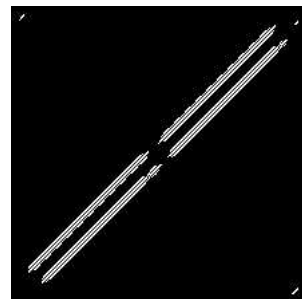
(c) Máscara 11x11, sin umbral



(d) Máscara 5x5, con umbral



(e) Máscara 7x7, con umbral



(f) Máscara 11x11, con umbral

Figura D.1: Imágenes de resultado después de aplicar máscaras de diferente tamaño.

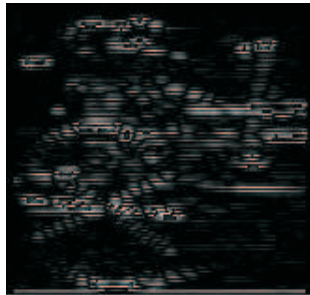
Apéndice E

Resultados de Imágenes Procesadas

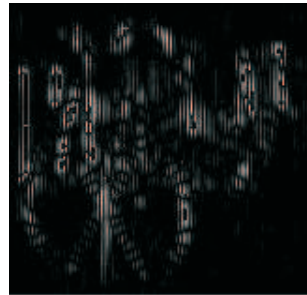
A continuación se presentan resultados de simulación sobre imágenes en escala de grises, aplicando máscaras 11x11.



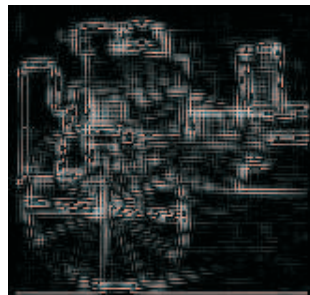
(a) Imagen original



(b) $\theta = 0^\circ$

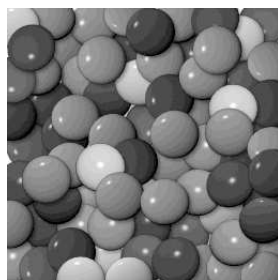


(c) $\theta = 90^\circ$

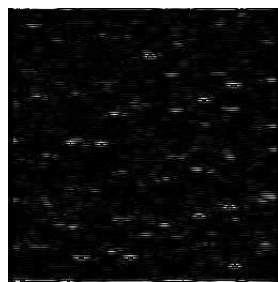


(d) Suma de ambas orientaciones

Figura E.1: Resultados de la imagen 192x192.



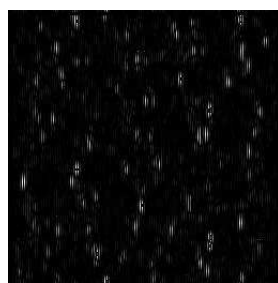
(a) Imagen original



(b) $\theta = 0^\circ$



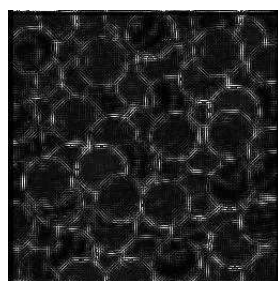
(c) $\theta = 45^\circ$



(d) $\theta = 90^\circ$

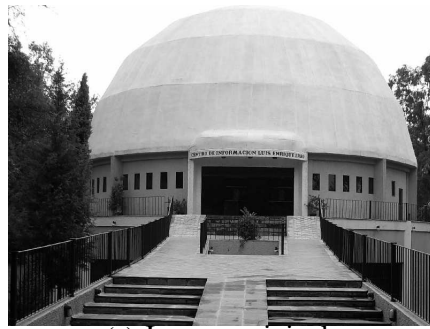


(e) $\theta = 135^\circ$

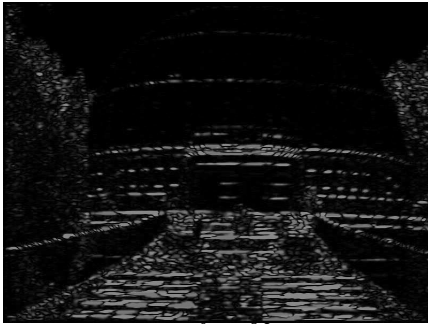


(f) Combinación de las
cuatro orientaciones

Figura E.2: Resultados de la imagen 256x256.



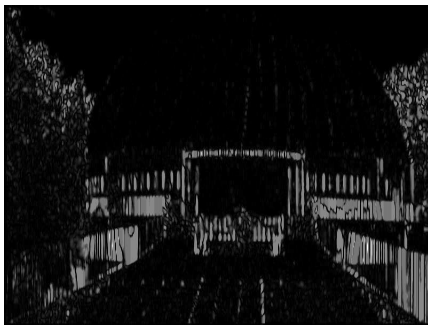
(a) Imagen original



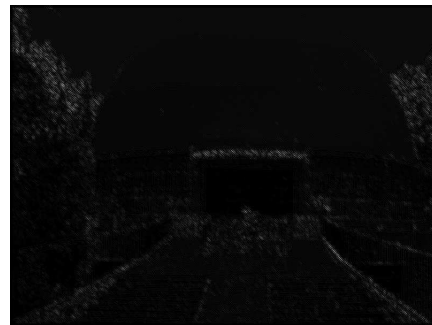
(b) $\theta = 0^\circ$



(c) $\theta = 45^\circ$



(d) $\theta = 90^\circ$



(e) $\theta = 135^\circ$



(f) Combinación de las cuatro orientaciones

Figura E.3: Resultados de la imagen 800x600.