UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE CIENCIAS QUIMICAS E INGENIERIA

Diseño de Arquitectura en Dispositivos Reconfigurables FPGA para Procesamiento de Imágenes en Tiempo Real Mediante Sistemas Difusos Tipo 2 por Intervalos

T E S I S

Que presenta para obtener el grado de MAESTRO EN CIENCIAS

Ing. Emanuel Ontiveros Robles

DIRECTOR DE TESIS:
Dr. José Luis González Vázquez

CO-DIRECTOR DE TESIS:
Dr. Oscar Castillo López

Tijuana, B. C. **Julio de 2016**

Universidad Autónoma de Baja California

FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA

COORDINACIÓN DE POSGRADO E INVESTIGACIÓN

FOLIO No. 173

Tijuana, B. C., a 6 de junio de 2016

C. Emanuel Ontiveros Robles
Pasante de: Maestro en Ciencias
Presente

El tema de trabajo y/o tesis para su examen profesional, en la Opción TESIS

Es propuesto, por los C. Dres. José Luis González Vázquez y Oscar Castillo López

Quienes serán los responsables de la calidad de trabajo que usted presente, referido al tema Diseño de Arquitectura en Dispositivos Reconfigurables FPGA para Procesamiento de Imágenes en Tiempo Real Mediante Sistemas Difusos Tipo 2 por Intervalos.

el cual deberá usted desarrollar, de acuerdo con el siguiente orden:

I.- INTRODUCCION
II.- PROCESAMIENTO DIGITAL DE IMÁGENES Y TIEMPO REAL
III.- LÓGICA DIFUSA
IV.- HARDWARE RECONFIGURABLE Y ARQUITECTURAS BASADAS EN LÓGICA DIFUSA
V.- DISEÑO DE ARQUITECTURA DE LÓGICA DIFUSA
VI.- EXPERIMENTOS Y EVALUACIÓN DE RENDIMIENTO
VII.- CONCLUSIONES

Dr. José Luis González Vázquez
Director de Tesis

Dr. Juan Ramón Castro Rodríguez
Secretario

Dr. Oscar Castillo López
Co-Director de Tesis

Dr. Luis Enrique Palafox Maestre
Director

DEDICATORIA

Dedico esta tesis a mi bella esposa *Gabriela Ortega Torres* quien es mi más grande motivación*, a* mis padres *Fidel Ontiveros* e *Yrma Robles* y hermana *Lizette Ontiveros* por su gran apoyo al iniciar este proyecto.

AGRADECIMIENTOS

Agradezco Dios, a mi esposa *Gabriela Ortega* por el apoyo incondicional durante esta etapa de mi formación profesional y en todos mis proyectos. Agradezco al *Dr. José Luis González Vázquez*, mi asesor y director de tesis, por todo el conocimiento que me transmitió pero principalmente por la confianza y el tiempo dedicado. Agradezco al *Dr. Juan Ramón Castro Rodríguez*, por participar activamente en mi formación como maestro en ciencias. Al *Dr. Eduardo Álvarez Guzmán* por sus retroalimentaciones oportunas para mejorar la calidad de mis trabajos. Al *Dr. Oscar Castillo* por su participación en la mayoría de los trabajos de publicación derivados de mi investigación.

A la Facultad de Ciencias Químicas de la Universidad Autónoma de Baja California por brindarme las oportunidades tanto en unidades de aprendizaje como en instalaciones y equipo de experimentación necesarios para llevar a cabo mi investigación con éxito. A CONACYT por la financiación de mi investigación por medio de la beca de manutención.

RESUMEN de la Tesis de Emanuel Ontiveros Robles presentada como requisito parcial para la obtención del grado de MAESTRO EN CIENCIAS. Tijuana, Baja California, México. Julio de 2016.

Diseño de arquitectura en dispositivos reconfigurables FPGA para procesamiento de imágenes en tiempo real mediante sistemas difusos tipo 2 por intervalos

Resumen aprobado por:

Dr. Oscar Castillo López
Codirector de tesis

Dr. José Luis González Vázquez
Director de tesis

Resumen

El presente proyecto de tesis busca atacar una de las principales limitaciones de la lógica difusa, el alto costo computacional. Con el fin de poder lograr la implantación de un sistema difuso en el contexto de procesamiento de imágenes en tiempo real, el cual es un contexto ampliamente demandante de tasa de procesamiento.
Los sistemas difusos ofrecen una alternativa a los métodos tradicionales de procesamiento de imágenes y entre sus principales aportaciones se encuentra la cualidad de la tolerancia a perturbaciones, sin embargo el desarrollo de aplicaciones de PDI empleando lógica difusa se ve limitado en buena medida debido al coste computacional que estos implican, a menudo mucho mayor que los métodos tradicionales.
En el presente trabajo se propone una arquitectura en hardware reconfigurable FPGA orientada hacia el procesamiento de imágenes en tiempo real mediante sistemas tipo 2 por intervalos como herramienta para reducir el costo computacional de los sistemas difusos y así reducir tiempos de ejecución, contemplando estrategias como paralelismo, pipelining, memoria distribuida.
Con el fin de evaluar la arquitectura evalúan las siguientes métricas de rendimiento.

- Tasa de procesamiento en (Fuzzy Logic Inference per Second) FLIPS.
- Calidad de la detección de bordes Figure of Merit of Pratt (FOM de Pratt).
- Fidelidad al modelo teórico Error relativo porcentual.

La arquitectura propuesta además de resolver el caso de estudio planteado, podrá potencialmente ser empleado para aplicaciones de otra índole, en el sector salud como diagnóstico de problemas en los ojos, centros de manufactura en el control de calidad, e incluso como un coprocesador para la aceleración de sistemas inteligentes que podrían ser difusos, neuro-difusos he incluso sistemas difusos tipo 2 que en software comerciales representen una gran carga computacional y por consiguiente tiempos de ejecución largos.
Los resultados obtenidos son evidencia de que la arquitectura no solo permite la implementación de la lógica difusa como alternativa viable para la detección de bordes en tiempo real sino que además logrando tasas de procesamiento que superan las metas propuestas y lo habilitan para procesar imágenes de alta resolución.

Palabras clave:

FPGA, Logica Difusa, Tiempo real, Deteccion de bordes

ABSTRACT of the thesis, presented by Emanuel Ontiveros Robles in order to obtain the MASTER of CIENCIES. Tijuana, Baja California, México. July, 2016.

Architecture Design on Reconfigurable FPGA Devices for Real-Time Digital Image Processing Through Intervals Type 2 Fuzzy Inference Systems

Approved by:

Dr. Oscar Castillo López
Thesis Advisor

Dr. José Luis González Vázquez
Thesis Advisor

Abstract

This thesis project aims to address one of the main limitations of fuzzy logic, the high computational cost. In order to achieve the implementation of a fuzzy system in the context of image processing in real time, which is a widely demandant processing fee context. Fuzzy systems offer an alternative to traditional methods of image processing and its main contributions is the robustness to disturbance, however the development of digital image processing using fuzzy logic is limited largely due to the computational cost these often involve much larger than traditional methods.

In this paper an architecture proposed on reconfigurable hardware as a tool to reduce the computational cost of fuzzy systems and reduce runtimes on FPGA, the architecture is oriented to image processing in real time by intervals type 2 fuzzy inference system. Additionally using hardware strategies such as parallelism, pipelining, and distributed memory.

The architecture evaluation is proposed following the next performance metrics.

• Processing rate (Fuzzy Logic Inference per Second) FLIP.

• Quality of edge detection Figure of Merit of Pratt (Pratt FOM).

• Loyalty theoretical model relative error percentage.

The proposed architecture in addition to solving the case of proposed study may potentially be used for applications other in the health sector and diagnosis of eye problems, centers of manufacturing quality control, and even as a coprocessor for acceleration of intelligent systems that could be fuzzy, neuro-fuzzy fuzzy type 2 systems that represent a large computational cost and consequently long run times.

The results are evidence that architecture not only allows the implementation of fuzzy logic as a viable alternative for edge detection in real time but also achieving processing fees that exceed the proposed goals and enable it to process high-resolution images .

Keywords:

FPGA, Fuzzy Logic, Real-Time, Edge Detection

Contenido

Resumen VI
Abstract VIII
Índice de figuras XII
Índice de tablas XIV
Capítulo 1 Introducción 1
1.1. Objetivos 2
1.1.1. Objetivo general 2
1.1.2. Objetivos específicos 2
1.1.3. Organización del trabajo 3
Capítulo 2 Procesamiento digital de imágenes y tiempo real 4
2.1. Sensores y formatos de imágenes digitales 4
2.1.1. Espacio de colores RGB 4
2.1.2. Escala de grises 5
2.1.3. Resolución de imagen 5
2.2. Procesamiento Digital De Imágenes 5
2.2.1. Procesamiento de bajo nivel 5
2.2.2. Ruido y perturbaciones 6
2.2.3. Procesamiento de nivel medio 8
Técnicas tradicionales de detección de bordes 8
2.2.4. Procesamiento de alto nivel 11
2.2.5. Extracción de características en imágenes 11
2.2.6. Calidad de detección de bordes 13
Capítulo 3 Lógica difusa 14
3.1.1. Funciones de membrecía. 15
3.1.2. Operadores de lógica difusa. 16
Unión entre A y B 17
3.1.3. Sistema de inferencia difuso de Mamdani 17
Lógica difusa tipo 2 por intervalos 20
Capítulo 4 Hardware Reconfigurable y Arquitecturas Basadas en Lógica Difusa 23
4.1.1. Estrategias de diseño en dispositivos reconfigurables 24
Paralelismo 24

Segmentación ... 24
Memoria distribuida ... 24
Notación de punto fijo ... 24
Capítulo 5 Diseño de arquitectura de lógica difusa ... 26
5.1. Metodología propuesta ... 26
5.1.1. Entorno de aplicación ... 26
5.1.2. Diseño FLED-T1 ... 27
Fuzzificador-T1 ... 27
Inferencia y reglas ... 28
Defusificación ... 28
5.1.3. Detección de bordes empleando IT2FL ... 30
5.2. Arquitectura de Sistema Difuso T1 ... 32
5.2.1. Módulo TrapMF ... 32
5.2.2. Módulo Fusificación ... 33
5.2.3. Modulo Inferencia/Regla ... 34
5.2.4. Módulo Defusificación ... 36
5.2.5. Fuzzy Edge Detector Architecture ... 37
5.2.6. Arquitectura FLED-T2 ... 37
Capítulo 6 Experimentos y Evaluación de Rendimiento ... 39
6.1. Experimento 1 Tasa de procesamiento FPGA ... 40
6.1.1. TrapMF ... 40
6.1.2. Fusificador ... 41
6.1.3. Inferencia ... 42
6.1.4. Defusificación ... 42
6.1.5. FLED-T2 ... 43
6.1.6. Análisis de resultados experimento 1 ... 44
6.2. Experimento 2 Calidad de bordes detectados ... 45
6.2.1. Análisis de resultados experimento 2 ... 47
6.3. Experimento 3: Rendimiento ante perturbaciones ... 48
6.3.1. Análisis de resultados experimento 3 ... 50
6.4. Experimento 4: Fidelidad al modelo teórico ... 51
6.4.1. Análisis de resultados Experimento 4 ... 55
Capítulo 7 Conclusiones ... 56

Referencias 59

Índice de figuras

Figura **Página**

Figura 2.1 Espacio de colores RGB 4
Figura 2.2 Efecto del ruido Impulsivo 6
Figura 2.3 Efecto del ruido Gaussiano 7
Figura 2.4 Ejemplo grafico de kernel genérico 7
Figura 2.5 Ejemplo de “ruido sal y pimienta” reducido con filtro de la mediana 8
Figura 2.6 Ventanas de convolución para cálculo de gradiente 9
Figura 2.7 Pixel y vecinos más próximos 9
Figura 2.8 a) Imagen estándar, b) Bordes de Sobel..... 10
Figura 2.9 Proceso para la identificación de objetos 11
Figura 3.1 A) Conjuntos tradicionales bivaluados, B) Conjuntos difusos 14
Figura 3.2 Complemento..... 16
Figura 3.3 Intersección entre A y B 16
Figura 3.4 Unión entre A y B 17
Figura 3.5 Sistema difuso de Mamdani 17
Figura 3.6 Regla..... 18
Figura 3.7 Conocimiento parcial de la regla r 19
Figura 3.8 Conocimiento agregado 19
Figura 3.9 Sistema difuso tipo 2 por intervalos..... 21
Figura 3.10 Fusificador IT2 Fuzzy 21
Figura 3.11 Inferencia IT2 Fuzzy..... 22
Figura 3.12 Reducción de tipo IT2 Fuzzy 22
Figura 3.13: Esquema básico de FPGA..... 23
Figura 3.14 Explicación grafica del paralelismo 24
Figura 5.1 Distribución de conjuntos difusos de entrada..... 27
Figura 5.2 Distribución de conjunto difuso de salida. 29
Figura 5.3 Conocimiento agregado del sistema 30
Figura 5.4 Funciones de membrecía de entrada..... 31
Figura 5.5: Superficie Δ/h 32
Figura 5.6 Modulo TrapMF 33
Figura 5.7 Modulo Fusificación..... 34

Figura 5.8 FSM Fusificación 34
Figura 5.9 Modulo Inferencia 35
Figura 5.10 FSM Inferencia 35
Figura 5.11 Modulo Defusificación 36
Figura 5.12 FSM Defusificación. 36
Figura 5.13 Arquitectura FLED-T1 37
Figura 5.14 Arquitectura FLED-T2 38
Figura 6.1 Grafica de tiempo de TrapMF 40
Figura 6.2 Diagrama de tiempo Fusificación 41
Figura 6.3 Diagrama de tiempo Inferencia 42
Figura 6.4 Grafica de tiempo Defusificación 42
Figura 6.5 Grafica de tiempo de la arquitectura FLED-T2 43
Figura 6.6 Requerimientos en MFLIPS para tiempo real 44
Figura 6.7 Rendimiento del FLED-T2 ante perturbaciones gaussianas. 48
Figura 6.8 Rendimiento de Canny ante perturbaciones gaussianas. 50
Figura 6.9 Comportamiento del error. 52
Figura 6.10 Comportamiento del error para 1 bit fraccional. 53
Figura 6.11 Comportamiento del error para 3 bits fraccionales. 53
Figura 6.12 Máxima superficie de error 54
Figura 6.13 Superficie de error para sistema propuesto 54

Índice de tablas

Tabla **Página**

Tabla 2.1 Resoluciones estándar de imágenes digitales 5
Tabla 2.2 Orden en tamaños estándar de imagen 11
Tabla 5.1 Entorno de aplicación 26
Tabla 5.2 Sistema difuso FLED-T1 27
Tabla 5.3 Recursos TrapMF 33
Tabla 5.4 Recursos Fusificación 34
Tabla 5.5 Recursos Inferencia 35
Tabla 5.6 Recursos Defusificación 36
Tabla 5.7 Recursos Arquitectura FLED-T1 37
Tabla 5.8 Recursos Arquitectura FLED-T2 38
Tabla 6.1 Imágenes empleadas 39
Tabla 6.2 Rendimiento TrapMF 40
Tabla 6.3 Rendimiento Fusificación 41
Tabla 6.4 Rendimiento Inferencia 42
Tabla 6.5 Rendimiento Defusificación 43
Tabla 6.6 Rendimiento Arquitectura FLED-T2 43
Tabla 6.7 Calidad del FLED-T2 con respecto a detectores convencionales 45
Tabla 6.8 Resultados obtenidos por detectores de bordes 46
Tabla 6.9 Detección de bordes en diferentes contextos 47
Tabla 6.10 FOM de imágenes contaminadas 48
Tabla 6.11 Detección de bordes para imágenes contaminadas con ruido gaussiano 49
Tabla 6.12 FOM Imágenes contaminadas Canny 50
Tabla 6.13 Detección de bordes para distribución de máximo error 55

Capítulo 1

Introducción

En la actualidad, los algoritmos de procesamiento digital de imágenes (PDI) están cada vez más presentes en aspectos importantes de la vida cotidiana, tales como la salud, comunicaciones, seguridad civil, control de calidad en la industria, seguridad.

Dada la importancia de las aplicaciones antes mencionadas, es importante hacer esfuerzos por mejorar los algoritmos actuales de PDI, mismos que tienen ciertas limitantes, uno de los principales problemas de estos algoritmos es que suelen ser susceptibles a perturbaciones en el ambiente, esto cobra importancia debido a que limita el rendimiento de los algoritmos en ambientes no ideales, otro de los principales problemas es el alto costo computacional que demandan, lo cual suele ser una limitante en aplicaciones de tasa de procesamiento critica. ‘

Es necesario pues incrementar la tasa de procesamiento de los sistemas basados en PDI sin descuidar también la robustez de los mismos y la calidad de los resultados.

Como se abordará en el capítulo 2, los algoritmos de PDI empleados para las aplicaciones mencionadas, suelen ser de alto grado de complejidad y contienen dentro de sí algoritmos de menor complejidad empleados como etapas del mismo, la presente investigación se centra en uno de los más importantes algoritmos del procesamiento digital de imágenes, la detección de bordes.

La detección de bordes como tal no entra en la categoría de procesamiento de alta complejidad, sin embargo, está presente en la mayoría de los algoritmos empleados en las aplicaciones del PDI, como ya ha sido mencionado antes, contiene el principal problema de ser susceptible a perturbaciones, tales como niveles de contraste o bien ruidos del ambiente, y en menor medida tiene el problema del costo computacional, debido a que no es un procesamiento de alta complejidad.

Existe sin embargo una alternativa que mejora los rendimientos de la detección de bordes, se trata de la detección de bordes mediante Lógica Difusa, diversos autores han demostrado la eficacia sistemas difusos, por ejemplo, de los sistemas difusos tipo 1 [1]–[7] , de los sistemas difusos tipo 2 por intervalos [8] y de los sistemas difusos tipo 2 generalizados [9], [10] sin embargo, una limitante como tal de los sistemas difusos en aplicaciones de tasa de procesamiento critica es su alta complejidad comparado con las alternativas convencionales.

Si bien la detección de bordes mediante sistemas difusos ofrece una solución al rendimiento ante perturbaciones, vuelve critico el problema del costo computacional debido a que los sistemas difusos tienen el problema de ser algoritmos de mayor

complejidad que los convencionales, es debido a esto que los sistemas difusos normalmente no son una opción viable en aplicaciones reales.

Con base en lo antes mencionado, la presente investigación se enfoca en atender los problemas del costo computacional mediante el diseño de una arquitectura de hardware, de propósito específico, que permita no solo realizar detección de bordes mediante sistemas difusos sino además ofrezca una tasa de procesamiento que permita detección de bordes en tiempo real, es decir 24 cuadros por segundo en una resolución estándar VGA (640x480 px).

1.1. Objetivos

1.1.1. Objetivo general

El objetivo de la presente investigación es el de diseñar una arquitectura de hardware reconfigurable inspirados en un sistema de inferencia difuso tipo 2 por intervalos que realice la detección de bordes, basados en la tecnología FPGA (Field Programmable Gate Array) mediante lenguaje descriptor de hardware VHDL y que permita de esta manera lograr realizar una detección de bordes en tiempo real, con alta fidelidad al modelo matemático y tolerante a perturbaciones.

1.1.2. Objetivos específicos

A continuación se enlistan los objetivos específicos de la investigación.

- ✓ Formular un sistema difuso tipo 2 por intervalos que realice la detección de bordes (FLED-T2) y que tenga las siguientes características.
    - Parametrizado
    - Logre una buena calidad en la detección de bordes evaluados mediante métricas de calidad de bordes.
    - Demuestre robustez ante perturbaciones evaluada mediante métricas de calidad de bordes.
- ✓ Diseño de una arquitectura de hardware que sintetice el sistema difuso FLED-T2 con las siguientes metas de rendimiento.
    - Como mínimo alcanzar la tasa de procesamiento de 7.3 MFLIPS (Millones de inferencias lógicas por segundo) correspondiente a lograr procesamiento en tiempo real (24 FPS) para una imagen de resolución estándar VGA (640 x 480 pixeles).
    - Demuestre fidelidad al modelo teórico - matemático mediante un error máximo porcentual del 5%.

1.1.3. Organización del trabajo

En cuanto a la organización del trabajo, se comienza por los fundamentos del PDI y tiempo real, siguiendo con fundamentos de lógica difusa y finalmente los fundamentos del hardware reconfigurable.

Habiendo ahondado sobre los fundamentos se procede con el diseño del sistema difuso FLED-T2, y posteriormente con el diseño de la arquitectura FLED-T2, mismas que son evaluadas mediante métricas de calidad de bordes, tasa de procesamiento, robustez y fidelidad al modelo teórico, para concluir con los análisis de resultados y las conclusiones.

Capítulo 2

Procesamiento digital de imágenes y tiempo real

En este capítulo se abordarán los fundamentos sobre los cuales se construye el desarrollo, experimentación y análisis de resultados.

En primer lugar, los fundamentos y generalidades del procesamiento digital de imágenes, hablando de los diferentes niveles de procesamiento digital de imágenes y abordando la explicación de conceptos importantes para discusiones posteriores. Posteriormente se introduce a los conceptos relacionados con la lógica difusa, los principios y el estado del arte de los sistemas difusos tipo 2 por intervalos, y finalmente se explican los principios, características y conceptos importantes relativos a los sistemas reconfigurables y sus estrategias de diseño que serán empleadas en el diseño de la arquitectura propuesta.

2.1. Sensores y formatos de imágenes digitales

Una imagen digital es definida como una función bidimensional f(x, y), donde “x” y “y” representan ejes de un plano coordenado y el valor de f se le conoce como la intensidad del pixel en x, y, donde todas sus variables son discretas y finitas.

2.1.1. Espacio de colores RGB

El modelo de colores RGB se basa en una síntesis aditiva del Rojo, Verde y Azul, la gama de colores que pueden alcanzarse es asociada con la resolución en bits de cada color, en la Fig. 2.1 se muestra una representación de colores RGB para una resolución de 8 bits.

En este caso la cantidad de colores diferentes que se pueden alcanzar es un poco más de 16.5 millones.

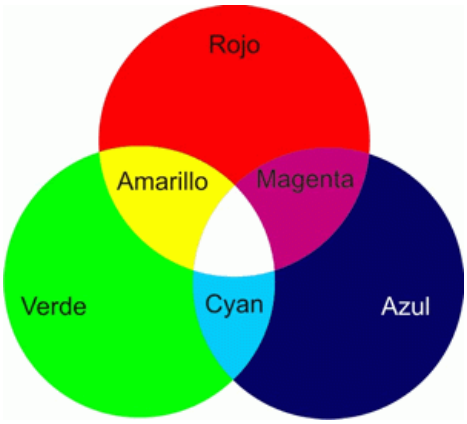


Figura 2.1 Espacio de colores RGB

2.1.2. Escala de grises

Como se puede observar debido a la cantidad de colores diferentes, el procesamiento de imágenes generalmente opta por evitar el procesamiento en RGB u otro espacio de colores debido a que pequeños cambios pueden producir resultados indeseados, esto debido a la susceptibilidad a ruidos que presentan muchos algoritmos de procesamiento de imágenes de nivel medio y alto.

Es por el problema antes mencionado que nace la necesidad de convertir el mapa de colores RGB a un solo dominio en escala de grises. Esto se logra mediante la ecuación (2.1):

$$Gris = \frac{R+G+B}{3} \tag{2.1}$$

Existen diferentes algoritmos de conversión, el presentado consiste únicamente en el promedio de las 3 mascaras de color.

2.1.3. Resolución de imagen

La resolución de una imagen denota el grado de detalle de la misma, existen diferentes estándares de resolución medidos en pixeles, a continuación la tabla muestra algunos de los estándares más empleados.

Tabla 2.1 Resoluciones estándar de imágenes digitales

Estándar	Ancho	Alto	Pixeles
VGA (Video Graphics Array)	640	480	307200
S-VGA (Super Video Graphics Array)	800	600	480000
XGA (eXtended Graphics Array)	1024	768	786432
S-XGA (Super eXtended Graphics Array)	1240	1024	1269760
U-XGA (Ultra eXtended Graphics Array)	1600	1200	1920000

2.2. Procesamiento Digital De Imágenes

El procesamiento de imágenes consiste en todo método, función, algoritmo que haga uso de la información contenida en una imagen digital para la obtención de un resultado esperado.

En [11] se clasifica el procesamiento de imágenes en 3 niveles:

2.2.1. Procesamiento de bajo nivel

Este nivel es caracterizado por el hecho de que tantos sus argumentos de entrada como de salida son imágenes, se refiere principalmente a labores de pre-procesamiento tales como, ajuste de brillo, corrección de histograma, mejorar contraste o reducción de ruido.

Los procesos de pre-procesamiento, son procesos que tienen como finalidad la de acondicionar la imagen para posteriores etapas de procesamiento, realizándose como una etapa previa a procesamientos de mayor nivel.

2.2.2. Ruido y perturbaciones

Uno de los principales pre-procesamientos está relacionado con las perturbaciones en las imágenes, a continuación se explican dos de los ruidos más comunes presentes en imágenes. Estos ruidos o perturbaciones son en muchas ocasiones causados por defectos en los dispositivos de capturas como cámaras defectuosas, efectos de la iluminación o partículas en el ambiente. En [12] se explican los ruidos y sus efectos en procesamiento digital de imágenes.

Ruido impulsivo o sal y pimienta

El ruido impulsivo es un ruido que afecta aleatoriamente un porcentaje de los pixeles de la imagen, dándole valores máximos o mínimos, lo cual da se observa con la aparición de puntos negros y blancos en la imagen, dichos puntos no tienen relación alguna con los pixeles vecinos. La Fig. 2.2 muestra un ejemplo de los efectos del ruido impulsivo.

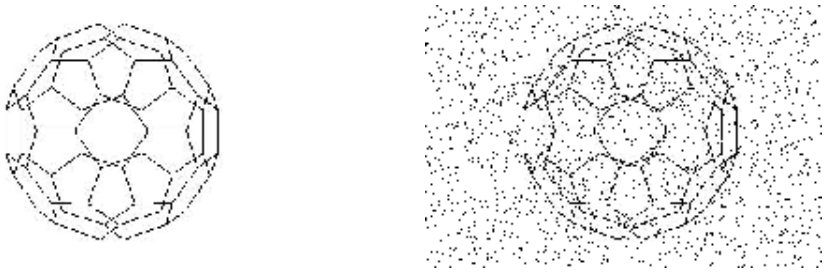


Figura 2.2 Efecto del ruido Impulsivo

Ruido Gaussiano

El ruido Gaussiano afecta a todos los pixeles de la imagen de acuerdo a una curva gaussiana, tal y como se muestra en el ecuación (2.2).

$$f(x) = \frac{1}{2\pi\sigma} e^{\left(\frac{x-m}{2\sigma}\right)^2} \tag{2.2}$$

Como se puede observar, el ruido gaussiano depende en mayor medida del valor de la varianza y esté afecta de manera que el histograma de la imagen se comporta cada vez más como una distribución gaussiana. La Fig. 2.3 muestra el efecto de este ruido en imágenes

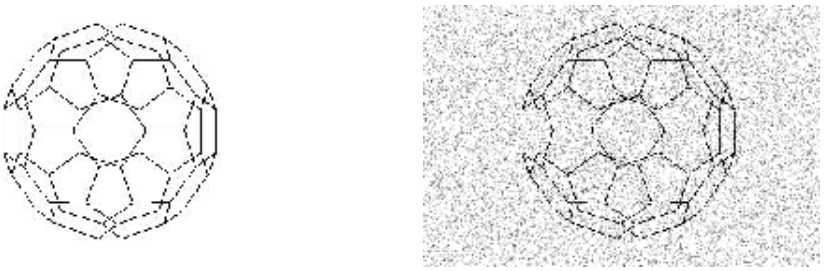


Figura 2.3 Efecto del ruido Gaussiano

Filtros espaciales

Con el fin de corregir el efecto de las perturbaciones se emplean comúnmente los filtros espaciales [11], los filtrados espaciales consisten en el paso de una matriz rectangular con pesos llamada ventana o kernel, evaluando el valor del pixel central de acuerdo a los pesos ponderados de sus vecinos, esta operación se conoce como convolución bidimensional. La Fig. 2.4 muestra los operandos de la convolución bidimensional, es decir, el kernel del filtro y el segmento de imagen.

h (Kernel)

1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3

f (Segmento de imagen)

x+1,y+1	x+1,y+2	x+1,y+3
x+2,y+1	x+2,y+2	x+2,y+3
x+3,y+1	x+3,y+2	x+3,y+3

Figura 2.4 Ejemplo grafico de kernel genérico

Donde se trabaja con segmentos de imágenes, en este caso de 3 x 3 pixeles llamadas ventanas, donde las ventanas se encuentran en función del pixel central. La ecuación X muestra el desarrollo de la operación de convolución para la imagen parcial contra el kernel del filtro de manera genérica.

$$g(x,y) = \sum\sum h(i,j)xf(x+i,y+j) \tag{2.3}$$

El valor del pixel central sería entonces la suma de las multiplicaciones de la ventana con el mismo y sus vecinos próximos según el tamaño de la ventana.

En el caso de una ventana de 3x3 el número de multiplicaciones es 9 y además la suma, ahora bien si consideramos una imagen estándar 480 x 640 pixeles el aplicar un filtro espacial de 3x3 representa 2,764, 800 multiplicaciones más sus respectivas

sumas, esto es solo un ejemplo de la carga computacional que representa un PDI de bajo orden.

Existen diferentes variantes de filtros espaciales y estos se diferencian en su kernel únicamente, es importante elegir uno en base al tipo de ruidos o perturbaciones que se esperan.

A continuación se muestra un ejemplo del ruido impulsivo reducido con el filtro de la mediana. El kernel del filtro de la mediana se muestra en la ecuación (2.4) mismo que suele ser especialmente efectivo contra el ruido impulsivo [13].

$$\begin{vmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{vmatrix} \tag{2.4}$$

La Fig. 2.5 muestra los efectos del ruido impulsivo empleando la convolución de cada vecindad de pixeles por el kernel correspondiente al filtro de la mediana.

Figura 2.5 Ejemplo de “ruido sal y pimienta” reducido con filtro de la mediana

2.2.3. Procesamiento de nivel medio

Este nivel se caracteriza por tener imágenes como argumentos de entrada y atributos de la misma como argumentos de salida, se refiere a labores como segmentación, detección de bordes, contornos he identificación de patrones.

Técnicas tradicionales de detección de bordes

Uno de los procesos más usados dentro del procesamiento de imágenes es la detección de bordes, a menudo como paso previo a la extracción de características tales como áreas o contornos. Existen gran variedad de detectores de bordes como se profundiza en [14], en el presente trabajo se abordan algunas de las técnicas

tradicionales de detección de bordes basadas en el gradiente, mismas que se obtiene mediante la convolución de 2 kernels como se observa en la Fig. 2.6.

Gx(x,y)

$\frac{1}{2+K}$

1	0	-1
K	0	-K
1	0	-1

Gy(x,y)

$\frac{1}{2+K}$

-1	-K	-1
0	0	0
1	K	1

Figura 2.6 Ventanas de convolución para cálculo de gradiente

Los valores de k definen el tipo de detector que es, por ejemplo, para k=1 se trata del detector de Sobel, para k=2 corresponde al detector de Prewitt y para k=0.5 corresponde al detector de Frei-Chen.

En la Fig. 2.6, se muestran dos matrices o ventanas que se conocen como ventanas de convolución, estas, como su nombre lo indica, funcionan al ser convolucionadas con los datos de la imagen original, es decir, un pixel y sus 8 vecinos, como se muestra en la Fig. 2.7.

I(x-1,y-1)	I(x,y-1)	I(x+1,y-1)
I(x-1,y)	I(x,y)	I(x+1,y)
I(x-1,y+1)	I(x,y+1)	I(x+1,y+1)

Figura 2.7 Pixel y vecinos más próximos

El resultado de la convolución da como resultados el gradiente renglón y el gradiente columna que bien son componentes del llamado vector gradiente, que es el vector que apunta hacia la máxima variación. Todo lo anterior se expresa en la ecuación (2.5).

$$\Delta f(x,y) = \begin{pmatrix} \frac{\Delta f(x,y)}{\Delta x} \\ \frac{\Delta f(x,y)}{\Delta y} \end{pmatrix} \begin{cases} \overline{\Delta f(x,y)} = \sqrt{\frac{\Delta f(x,y)}{\Delta x}^2 + \frac{\Delta f(x,y)}{\Delta y}^2} \\ \emptyset = \tan^{-1}\left(\frac{\frac{\Delta f(x,y)}{\Delta y}}{\frac{\Delta f(x,y)}{\Delta x}}\right) \end{cases} \quad (2.5)$$

Donde $\Delta f(x,y)$ es el vector gradiente y $\frac{\Delta f(x,y)}{\Delta x}$ y $\frac{\Delta f(x,y)}{\Delta y}$ son el gradiente renglón y el gradiente columna, respectivamente.

Analizando el orden de la convolución nos encontramos con la siguiente expresión, representada en la ecuación (2.6).

$$\frac{\partial f(x,y)}{\partial x} = \sum_{i=1}^{n}\sum_{j=1}^{m} f(x,y) * Gx(x,y)$$
$$\frac{\partial f(x,y)}{\partial y} = \sum_{i=1}^{n}\sum_{j=1}^{m} f(x,y) * Gy(x,y) \quad (2.6)$$

Se puede observar que para el cálculo de los gradientes renglón y columna se necesitan para el caso de una ventana de convolución de 3x3, una cantidad de 9 multiplicaciones y 9 sumas por cada gradiente, esto daría un total de 18 multiplicaciones y 18 sumas por pixel tan solo en el cálculo de los gradientes fila y columna, en la siguiente tabla se muestra una extrapolación en contexto de tamaños estándar de imágenes. La Fig. 2.8 muestra la evaluación de un detector de bordes de Sobel.

a) b)

Figura 2.8 a) Imagen estándar, b) Bordes de Sobel

La Tabla 2.2 da cuenta de lo demandante del proceso en términos de costo computacional, y aunque sea realizable la cantidad de operaciones en un tiempo razonable, es importante considerar que para aplicaciones en tiempo real es posible que para compensar la tasa de procesamiento se tenga que ceder en términos económicos para la adquisición de dispositivos de cálculo más complejos o mejores procesadores.

Tabla 2.2 Orden en tamaños estándar de imagen

Estándar	Multiplicaciones	Sumas	Millones de operaciones	Millones de operaciones
VGA	2.8	2.8	5.5	132.7
S-VGA	4.3	4.3	8.6	207.4
XGA	7.1	7.1	14.2	339.7
S-XGA	12.4	12.4	22.9	548.5
U-XGA	17.3	17.3	34.6	829.4

2.2.4. Procesamiento de alto nivel

Este nivel al igual de el de bajo nivel, tiene como argumentos de entrada y salida imágenes, sin embargo además de esto tiene información obtenida de la imagen, se refiere a trabajos de análisis en los cuales se pretenda obtener he interpretar información de la imagen.

2.2.5. Extracción de características en imágenes

La identificación de objetos de una imagen es un proceso de alta complejidad dentro de los algoritmos del procesamiento digital de imágenes [15], está constituido por etapas, como se muestra a continuación en Fig. 2.9.

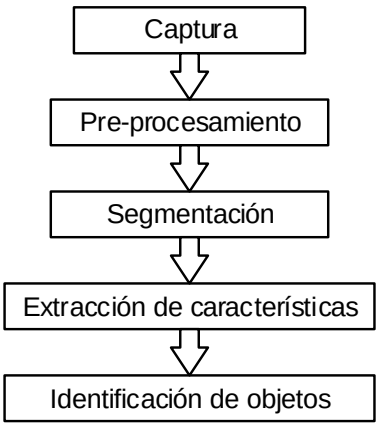


Figura 2.9 Proceso para la identificación de objetos

Donde los procesos tienen diferentes propósitos, explicados a continuación.

Captura

Se encarga de ser la interfaz entre el mundo real y el sistema de cómputo.

Pre-Procesamiento

Su objetivo es el de modificar la imagen preparándola para los posteriores procesamientos, reduciendo el ruido, modificando tamaños, acondicionando la información.

Segmentación

Se encarga de aislar el área de interés para así llevar a cabo los siguientes procesos de manera más eficiente.

Extracción de características

Se enfoca en obtener información a través de la imagen previamente procesada, por ejemplo, posición, áreas, perímetros, resaltar figuras geométricas, etc.

Identificación de objetos

Por medio de la información obtenida se encarga de la toma de decisiones, en biometría por ejemplo, si la huella corresponde o no a la persona que se desea identificar.

2.2.6. Calidad de detección de bordes

Como se puede observar, en el procesamiento de alto nivel, generalmente contiene procesamientos de nivel inferior, la mayoría de los procesamientos de alto nivel en imágenes contienen como etapa intermedia un detector de bordes, puesto que reduce la cantidad de información a evaluar y es usada para poder extraer características de la imagen, es por ello que la presente tesis hace énfasis en la detección de bordes, sin embargo, es importante contar con una métrica de calidad para la detección de bordes.

La FOM (Figure of Merit) de Pratt [16] es una métrica para evaluar la calidad de los bordes detectados, para la evaluación de esta métrica es necesario contar con una imagen ideal de los bordes esperados, y la imágenes que se desea evaluar, la ecuación para el cálculo se muestra en (2.7).

$$R = \frac{1}{I_N}\sum_{i=1}^{I_A}\frac{1}{1+ad^2} \tag{2.7}$$

Donde I_N es el valor máximo entre I_i e $I_A$que son la cantidad de bordes detectados para la imagen a evaluar y para la imagen ideal, respectivamente.

La variable a representa una constante de escalamiento que tomará el valor de 1/9 acorde al artículo original, la variable d representa la distancia de cada borde detectado al borde ideal más cercano.

Realizando la evaluación se obtiene un valor entre 0 y 1 donde el 0 representa una detección deficiente y el 1 representa una detección buena, esto con respecto a la imagen ideal o referencia.

Capítulo 3

Lógica difusa

Debido a la alta sensibilidad a perturbaciones, los detectores de bordes convencionales necesitan etapas de pre procesamiento con el fin de reducir los efectos de las perturbaciones, sin embargo, la lógica difusa ofrece la posibilidad de realizar la detección de bordes con mayor tolerancia a perturbaciones y sin necesidad de pre procesamientos.

La lógica difusa tiene como objetivo la aproximación del razonamiento humano en forma matemática, y fue propuesta en 1965 por Lofti Zadeh [17].

La lógica difusa surge como una expansión de la lógica clásica, particularmente el concepto de *pertenencia* empleado en la teoría de conjuntos.

En la teoría de conjuntos clásica, el concepto de *pertenencia* es empleado para expresar si un elemento forma parte de un conjunto y tiene dos valores: cero para la no-pertenencia y uno para la total pertenencia. Basados en el concepto de *pertenencia*, la lógica difusa propone que existen grados parciales de pertenencia y no únicamente valores duros de cero y uno, de esta manera, la pertenencia en lógica difusa se conoce como la evaluación una valor mediante *funciones de membrecía* o *funciones de pertenencia* de cada conjunto difuso.

Ejemplo: Si en el universo de hombres se plantearan dos conjuntos para catalogar a los hombres como “altos” o “bajos”, la representación de su membrecía seria la mostrada en Fig. 3.1.

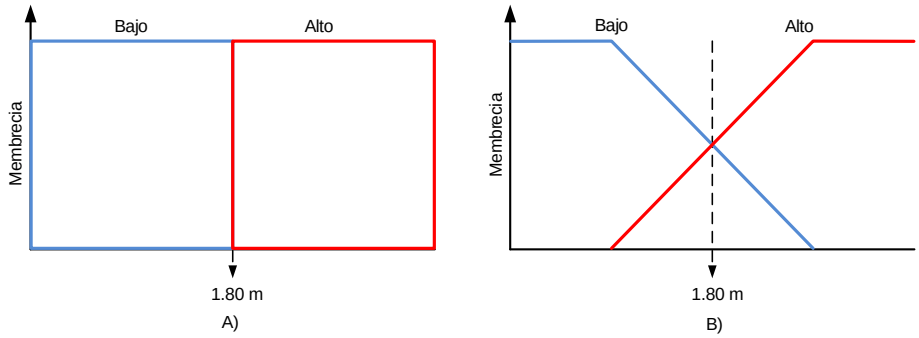


Figura 3.1 A) Conjuntos tradicionales bivaluados, B) Conjuntos difusos

Como se puede observar, 1.80 metros se podría considerar un punto a partir del cual una persona se puede considerar alta, sin embargo mientras que la lógica tradicional muestra al conjunto “bajo” y “alto” como mutuamente excluyentes esto representa una inconsistencia con el razonamiento humano, debido a que una persona que mida 1.79 m seria catalogada como baja y una persona de 1.81 cm seria catalogada alta sin considerar que la diferencia de estaturas no es significativa.

La lógica difusa es capaz de aproximarse más al razonamiento humano mediante las funciones de membrecía debido a que es capaz de agregar grados de certidumbre a los conjuntos, esto permite manejar información parcial y poder tomar decisiones con información inexacta o con perturbaciones.

Es precisamente la capacidad de tomar decisiones con información inexacta o parcial, que le otorga a los sistemas difusos un alto grado robustez a perturbaciones.

3.1.1. Funciones de membrecía.

Dentro de la lógica difusa, existe una gran variedad de funciones de membrecía que buscan aproximar el razonamiento humano de diferentes formas, donde la evaluación de las funciones de membrecía se asocia a grados de pertenencia de las llamadas variables lingüísticas, mismas que suelen expresarse con prefijos tales como “muy”, “gran”, “poco” etc. , a continuación se presentan las más empleadas:

Función triangular.

$$\mu_{triangular}(x,a,b,c) = \begin{cases} 0 & x < a \\ \frac{x-a}{b-a} & a < x < b \\ \frac{c-x}{c-b} & b < x < c \\ 0 & x > c \end{cases} \tag{3.1}$$

Función trapezoidal

$$\mu_{trapezoidal}(x,a,b,c,d) = \begin{cases} 0 & x < a \\ \frac{x-a}{b-a} & a < x < b \\ 1 & b < x < c \\ \frac{d-x}{d-c} & c < x < d \\ 0 & x > d \end{cases} \tag{3.2}$$

Función Gaussiana

$$\mu_{gaussiana}(x,m,\delta) = e^{-\frac{1}{2}\left(\frac{x-m}{\delta}\right)^2} \tag{3.3}$$

Función GBell

$$\mu_{gbell}(x,a,b,c) = \frac{1}{1-\left|\frac{x-c}{a}\right|^{2b}} \tag{3.4}$$

Como se puede observar existen gran variedad de funciones de membrecía, todas con la característica de ser funciones matemáticas continuas, de esta manera se busca la mejor representación del razonamiento humano, al conjunto de funciones de membrecía que describen una variable se le conoce como *variable lingüística*.

3.1.2. Operadores de lógica difusa.

Debido a que la lógica difusa se basa en los conceptos de la teoría de conjuntos, los operadores difusos como tal son expansiones de los operadores de lógica tradicional, y son los siguientes.

Para funciones de membrecía A y B tal que (3.5).

$$A = \{(x, \mu_A(x)/x \epsilon X)\}\, y\, B = \{(x, \mu_B\,(x)/x \epsilon X)\} \tag{3.5}$$

Complemento de A

Como se puede observar en la ec. (3.6), la operación de complemento se comporta de la misma forma que un complemento en lógica clásica, la operación se muestra en la Fig. 3.2.

$$\mu'_A = 1 - \mu_A \tag{3.6}$$

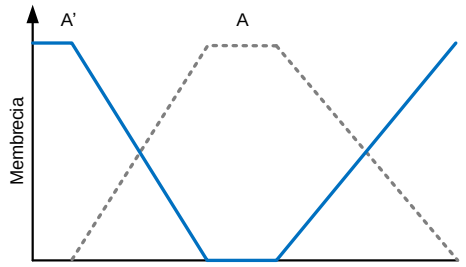


Figura 3.2 Complemento

Intersección entre A y B

La intersección entre dos funciones de membrecía se denomina *T- Norma* en lógica difusa y se denota por $\otimes$. Las T-Normas más empleadas son las expresadas en (3.7).

$$A \otimes B = \min(A, B)$$
$$A \otimes B = A * B \tag{3.7}$$

La representación gráfica de la T-Norma mínimo es la observada en la Fig. 3.3.

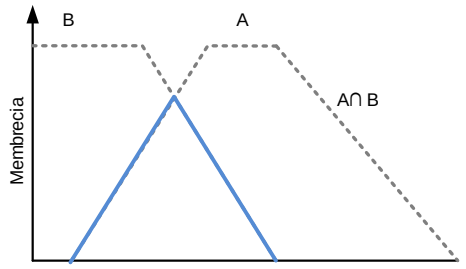


Figura 3.3 Intersección entre A y B

Unión entre A y B

La unión en lógica difusa se nombra como *S-Norma* y se denota por $\oplus$. Las S-Normas más empleadas son las expresadas en (3.8). Su interpretación grafica se muestra en la Fig. 3.4.

$$A \oplus B = \max(A, B)$$
$$A \oplus B = A + B - AB \tag{3.8}$$

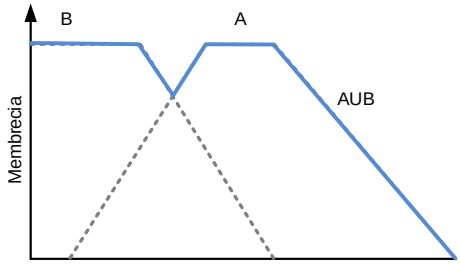


Figura 3.4 Unión entre A y B

3.1.3. Sistema de inferencia difuso de Mamdani

Haciendo uso de la lógica difusa surgen los llamados sistemas difusos, uno de los más empleados es el sistema difuso de Mamdani [18], cuyo énfasis original era el de control de plantas sin embargo en la actualidad ha sido empleado en diferentes aplicaciones [19], [20] que se muestra en la Fig. 3.5.

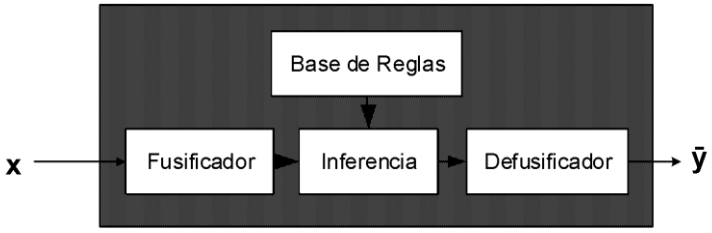


Figura 3.5 Sistema difuso de Mamdani

Este sistema se caracteriza por emplear la relación entre variables lingüísticas de entrada con variables lingüísticas de salida mediante reglas *si- entonces* conocidas como “Base de Reglas”, los bloques que lo componen se describen a continuación:

Fusificador

Este bloque se encarga de evaluar las funciones de membrecía de entrada con el fin de traducir las entradas duras a variables lingüísticas con el fin de poder ser relacionadas en etapas posteriores mediante relaciones difusas.

Base de Reglas

La base de reglas contiene el conocimiento del sistema, puede ser planteado por un experto o bien por medio de aprendizaje. El conocimiento del sistema puede ser total o parcial, siendo conocimiento completo a la cantidad de reglas equivalente a la combinatoria de funciones de membrecía del sistema.

Inferencia

El proceso de inferencia se refiere a la evaluación de la relación difusa descrita en las reglas mediante la "y" que en este caso corresponde a la T-Norma y como resultado obtiene las funciones de membrecía de cada regla, también conocidas como fuerzas de disparo.

Por ejemplo:

Considerando la regla

$$\mathbb{R}: si\ A\ y\ B\ entonces\ \ldots$$

A este razonamiento se le conoce como *Modus Ponens* [21].Se representa en la Fig. 3.6.

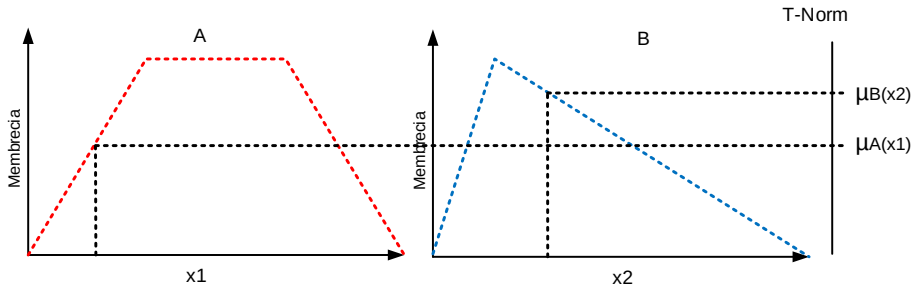


Figura 3.6 Regla

Donde considerando empleando las T-Norma min y producto se obtiene el resultado de la fuerza de disparo de la regla (3.9).

$$fr = \min\left(\mu_A(x1), \mu_B(x2)\right)\ \text{ó}\ fr = \mu_A(x1) * \mu\_B\ (x2) \qquad (3.9)$$

Defusificador

Existen diferentes métodos para llevar a cabo el proceso de defusificación, mismo que consiste en la agregación del conocimiento parcial obtenido por cada regla evaluada y su posterior traducción del dominio lingüístico a un valor duro [22].

La agregación se lleva a cabo mediante el operador difuso S-Norma de las funciones de membrecía de los consecuentes asociados a sus respectivas fuerzas de disparo mediante el operador T-Norma, como se desarrolla a continuación:

Conocimiento parcial de cada regla

La ecuación (3.10) muestra el conocimiento parcial para la *r*-esima regla, para cada sistema difuso se realizará la evaluación de *r* reglas, previo a su agregación.

$$Br = Fr \otimes Cr(y) \in y\ en\ Y \tag{3.10}$$

Siendo *Fr* la fuerza de disparo asociada a la regla y $Cr(y)$ la función de membrecía del consecuente, a continuación se muestra la apreciación grafica en Fig. 3.7.

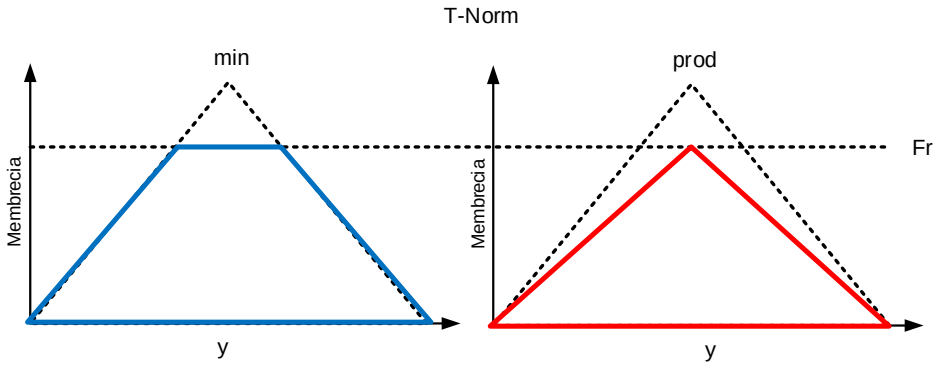


Figura 3.7 Conocimiento parcial de la regla r

Donde el conocimiento parcial resultante es una función de membrecía.

Agregación del conocimiento

La agregación se realiza mediante el operador S-Norma, de la siguiente manera (3.11).

$$B = \oplus_{r=1}^{R} Br \tag{3.11}$$

En la Fig. 3.8 se muestra un ejemplo gráfico:

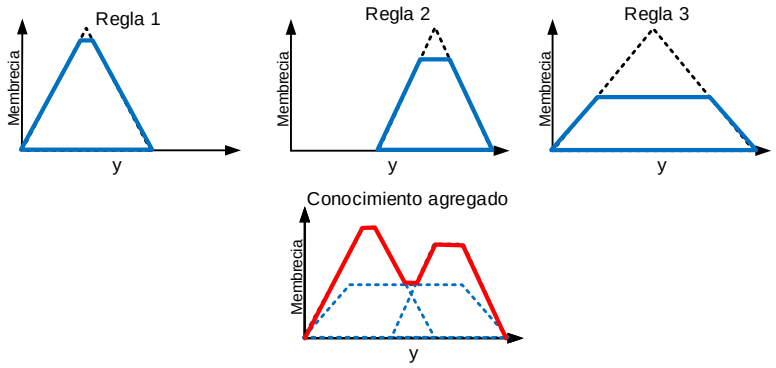


Figura 3.8 Conocimiento agregado

Los métodos para este proceso se pueden dividir de manera generalizada en dos categorías:

Defusificación por Centroide.

Este método consiste en la evaluación del Centroide de la agregación de los conocimientos parciales de cada regla. Se expresa en la ec. (3.12):

$$\tilde{y} = \frac{\sum \mu_B(y) * y}{\sum \mu_B(y)} \tag{3.12}$$

Donde μ_B es la función de membrecía del agregado de las reglas y la variable de salida se representa con y.

Defusificación aproximada

Debido al alto costo computacional empleado en la defusificación por centroide existen algunas alternativas que permiten aproximar el resultado mediante sumas ponderadas con diferentes consideraciones. En la ec. (3.12) se muestra la forma generalizada.

$$\tilde{y} = \frac{\sum_{i=1}^{i=R} Ki * \mu_{Bi}(y)}{\sum_{i=1}^{i=R} \mu_{Bi}(y)} \tag{3.12}$$

Donde $\mu_{Bi}(y)$ representa la función de membrecía del conocimiento asociado de cada regla, y *Ki* representa el método empleado para la ponderación.

Defusificación por alturas

La defusificación por alturas es uno de los casos de defusificación por ponderación, en este caso, se emplea la altura de la función de membrecía del conocimiento de cada regla (3.13).

$$\tilde{y} = \frac{\sum_{i=1}^{i=R} h_i * \mu_{Bi}(y)}{\sum_{i=1}^{i=R} \mu_{Bi}(y)} \tag{3.13}$$

Defusificación por centro de Conjuntos

La defusificación por centro de conjuntos emplea el centroide de la función de membrecía asociada al consecuente de cada regla (3.14).

$$\tilde{y} = \frac{\sum_{i=1}^{i=R} h_i * \mu_{Bi}(y)}{\sum_{i=1}^{i=R} \mu_{Bi}(y)} \tag{3.14}$$

Lógica difusa tipo 2 por intervalos.

La lógica difusa tipo 2 surge como resultado de los principios de lógica difusa, tales como las funciones de membrecía para evaluar los grados de pertenencia, los

operadores difusos y los sistemas de inferencia, sin embargo permite considera en su planteamiento las incertidumbres del sistema [23].

Está compuesto de 4 bloques fundamentales como se muestran en Fig. 3.9.

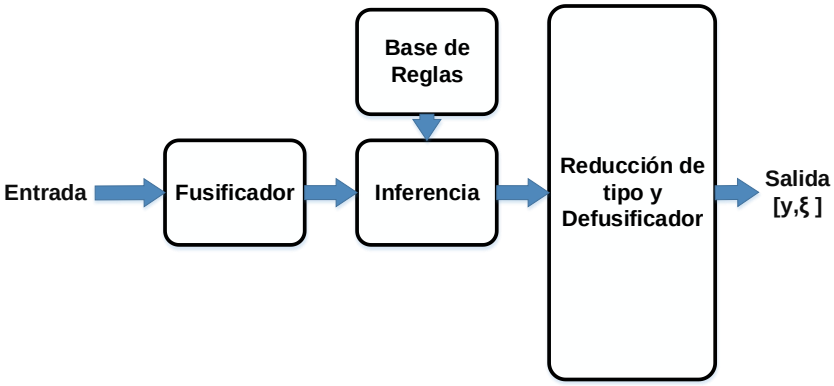


Figura 3.9 Sistema difuso tipo 2 por intervalos

Fusificador T2

El fusificador es el bloque encargado de la conversión de una variable real a variables lingüísticas para su posterior procesamiento, En la Fig. 3.10 se muestran los conjuntos difusos correspondientes a una variable lingüística. En el caso de lógica difusa, surge el concepto de FOU (Footprint of Uncertainty) como el intervalo dos funciones de membrecía tipo 1 que constituyen el intervalo superior e inferior [24].

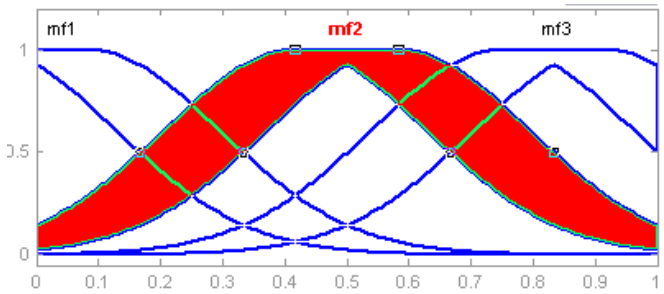


Figura 3.10 Fusificador IT2 Fuzzy

Como se puede observar en la figura, los conjuntos difusos constituidos por dos funciones de membrecía contienen dentro de si el área entre ellas conocida como FOU.

Inferencia/Reglas T2

El bloque de inferencia hace uso de la base de reglas para calcular así el conocimiento parcial resultado de la evaluación de cada regla establecida. En la Fig. 3.11 se muestra la evaluación de la base de reglas así como el conocimiento parcial que arroja dicha evaluación y su posterior agregación.

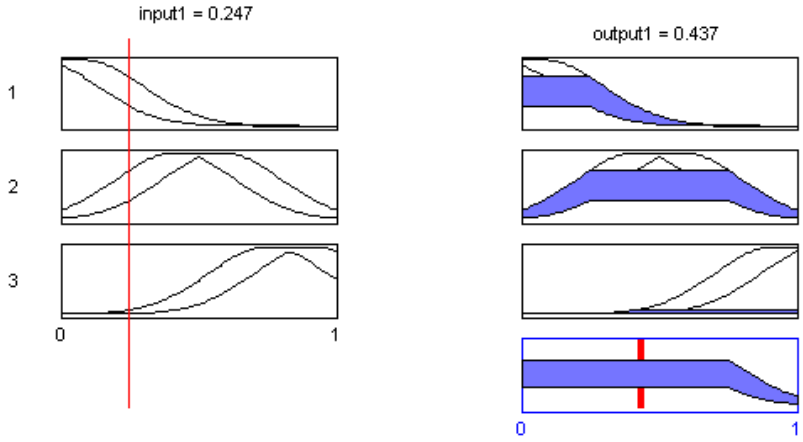


Figura 3.11 Inferencia IT2 Fuzzy

Reducción de tipo/ Defusificación

El bloque de reducción de tipo y defusificación es la encargada de convertir los conocimientos parciales en una respuesta concreta así como su grado de incertidumbre o bien un sistema difuso tipo 1. En la Fig. 3.12 se muestra la reducción de tipo y defusificación, la línea azul corresponde al resultado obtenido del sistema difuso y el área sombreada de verde corresponde al grado de incertidumbre obtenida en cada punto.

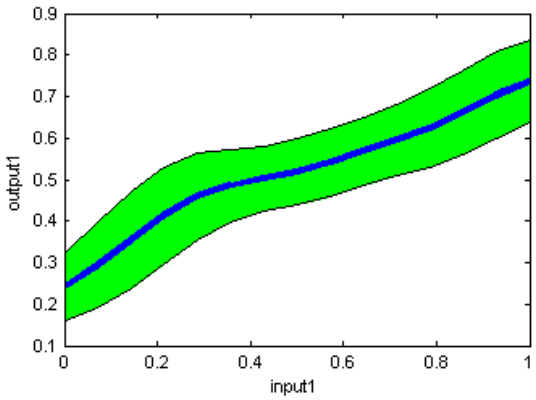


Figura 3.12 Reducción de tipo IT2 Fuzzy

Como se ha observado la lógica difusa tiene a su favor el hecho de poder representar el grado de incertidumbre de un sistema.

La desventaja de los sistemas difusos tipo 2 es el alto coste computacional que representan pues este es superior al doble del costo ya elevado de un sistema difuso tradicional.

Capítulo 4

Hardware Reconfigurable y Arquitecturas Basadas en Lógica Difusa

Debido a los demandantes recursos computacionales requeridos para los sistemas de inferencia difusa, estos no representan en muchas ocasiones una opción viable en el contexto de procesamiento digital de imágenes en el contexto de hardware, es por ello, que en el presente trabajo se propone la implementación de estos sistemas mediante sistemas reconfigurables de hardware llamados FPGA.

Un dispositivo FPGA (Field Programmable Gate Array) se trata de un dispositivo con bloques lógicos que tienen la ventaja de ser reconfigurables, haciendo de este un versátil dispositivo que puede albergar operaciones tan complejas como él mismo y dichos módulos operativos pueden ser descritos mediante el lenguaje VHDL, se trata de un lenguaje descriptor de hardware [25]**.**

Los FPGA han alcanzado utilidad en gran variedad de aplicaciones tales como la criptografía, el procesamiento digital de señales, y sistemas de cómputo bio-inspirados debido a su naturaleza reconfigurable y bajo consumo de potencia como explica [26]–[29]. Es precisamente por estas ventajas que se propone como herramienta para el diseño e implementación de arquitecturas diseñadas estratégicamente para el sistema de inferencia difuso.

En la Fig. 4.1 se encuentra el esquema básico de un FPGA el cual ofrece la versatilidad como principal característica al ser reconfigurable, así mismo es capaz de contener sistemas digitales tan complejos como el mismo FPGA.

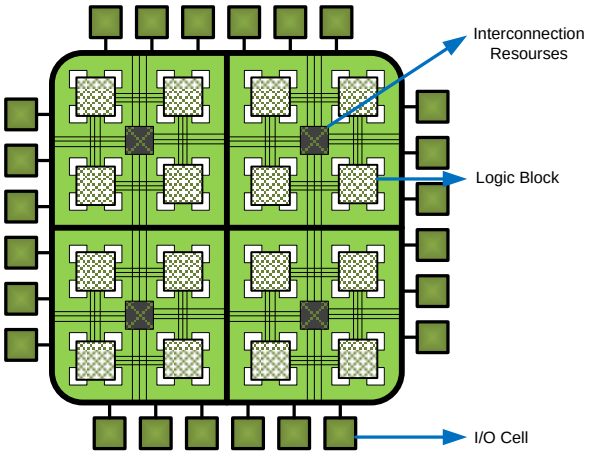


Figura 3.13: Esquema básico de FPGA

A su vez, el FPGA permite sintetizar sistemas cuya complejidad únicamente es limitada por los recursos del mismo, en este caso los llamados Look-Up Tables (LUT's), es decir los

bloques lógicos elementales del mismo, y los módulos especializados en multiplicación, mismos que pueden ser multiplicadores o módulos DSP, dependiendo el fabricante.

4.1.1. Estrategias de diseño en dispositivos reconfigurables

Al tratarse de un dispositivo reconfigurable es posible obtener ciertas ventajas sobre los dispositivos de procesamiento centralizado, las ventajas principales se explican a continuación.

Paralelismo

Quizá una ventaja muy poderosa para la reducción del tiempo de procesamiento y como su nombre lo indica consiste en la ejecución simultánea de 2 o más procesos Fig. 4.2.

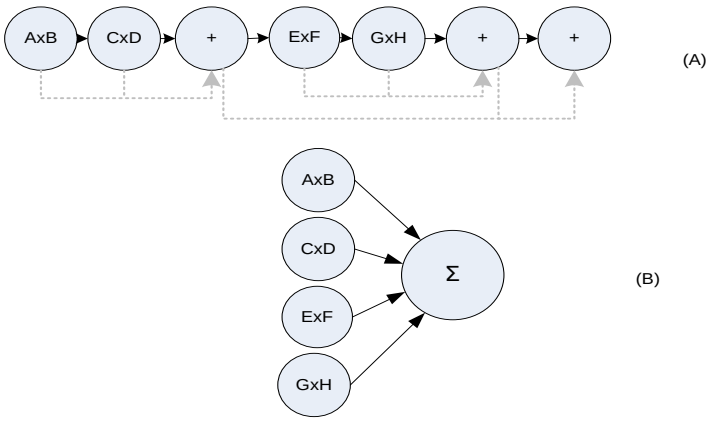


Figura 3.14 Explicación grafica del paralelismo

Segmentación

Se puede expresar como multiplexión en tiempo y podría ser considerado como otra forma de paralelismo puesto que consiste en la realización de más de un sub procesos de un proceso mayor, con diferentes datos a fin de aumentar la tasa de procesamiento.

Memoria distribuida

Consiste en la posibilidad de manejar recursos de memoria o registros de manera individual o segmentada, esto constituye un área de oportunidad puesto que es posible acceder a más de una direcciones de memoria simultáneamente, agilizando así el procesamiento o flujo de datos.

Notación de punto fijo

La notación de punto fijo surge de la necesidad de representar números fraccionarios dentro de un entorno digital cuyo manejo de datos es a nivel de bits.

La estrategia consiste en emplear los denominados bits fraccionarios mediante la operación de desplazamiento a la izquierda, en los operandos y posteriormente el desplazamiento a la derecha de los resultados, esto en una representación decimal se podría interpretar como la multiplicación de los operandos en función del número de bits fraccionarios como muestra la ecuación (4.1)

$$truncación(2^n a)truncación(b) = \frac{truncación(c)}{2^n} \quad (4.1)$$

Donde n representa el número de bits fraccionarios. Para solo un número con números fraccionarios. Si se consideran 2 números con números fraccionarios la ecuación seria (4.2).

$$truncación(2^n a)truncación(2^n b) = \frac{truncación(c)}{2^{n+1}} \quad (4.2)$$

A continuación, se explica un ejemplo para observar los efectos de la notación de punto fijo. Observemos el siguiente producto (4.3).

$$(13.9999)(90.9999) = 1273.9895 \quad (4.3)$$

Sin embargo, en un entorno con números binarios el factor con números fraccionales se vería truncado y por lo tanto se realizaría de la siguiente forma (4.4).

$$(13)(90) = 1170 \quad (4.4)$$

Como se puede observar, debido al truncamiento, el error obtenido es aproximadamente el 8%, sin embargo, empleando la notación de punto fijo el resultado sería el siguiente. Para un bit fraccionario, equivale a multiplicar el factor por 2 y dividir la respuesta entre 4, como sigue (4.5).

$$(27)(181) = \frac{4887}{4} = 1221 \quad (4.5)$$

Como se puede observar, el resultado se aproxima más al valor real, con un error del 4%, empleando dos bits fraccionales sería el equivalente a multiplicar y dividir por 8, como sigue (4.6)

$$(55)(363) = \frac{19965}{8} = 1247 \quad (4.6)$$

El valor de nueva cuenta se aproxima mayormente al valor real, con error del 2%, ahora bien, considerando 3 bits fraccionales, el equivalente seria multiplicar y dividir por 8, como se muestra a continuación (4.7).

$$(111)(727) = \frac{80697}{64} = 1260 \quad (4.7)$$

Finalmente para 3 bits fraccionarios el error se reduce al 1%

Como se puede observar, el FPGA ofrece ventajas que pueden ser empleadas para compensar el alto orden de los algoritmos de lógica difusa, y como principal reto está el hecho de que los lenguajes HDL a menudo representan un alto grado de complejidad por tratarse de lenguajes descriptores de hardware.

Capítulo 5

Diseño de arquitectura de lógica difusa

5.1. Metodología propuesta

Para el diseño de la arquitectura FLED-T1 es necesario seguir una serie de pasos previos con el fin de establecer criterios para la toma de decisiones y guiar el diseño hacia una arquitectura que cumpla con las metas propuestas.

En primer lugar, es necesario establecer el entorno de aplicación hacia el cual está dirigida la arquitectura, sin comprometer versatilidad pero considerando las capacidades y limitaciones de los sistemas de hardware.

Una vez establecido el entorno de aplicación se procede con el diseño del sistema difuso FLED-T1 -T1 y extendiendo hacia el sistema FLED-T2, realizando el desarrollo teórico e implementando el sistema en software para evaluar su desempeño y realizar los ajustes necesarios.

5.1.1. Entorno de aplicación

Nos referimos a entorno de aplicación a aquellos criterios de hardware importantes que influyen en el diseño de la arquitectura, incluyendo los rendimientos esperados por la misma, dichos criterios están sintetizados en la Tabla 5.5.

Tabla 5.1 Entorno de aplicación

Criterio	
FPGA	**Spartan 6**
Resolución de los pixeles	**8 bits**
Tiempo real	**>24 FPS**
Resolución de imagen	**640 x 480 pixeles**
Cifras significativas	**3**

Se proponen este entorno de aplicación con fines de experimentación, sin embargo, la arquitectura no es acotada a un solo FPGA (o la tecnología FPGA) ni al contexto de imágenes establecido, sino que ofrece la versatilidad característica de las arquitecturas basadas en lenguajes HDL.

5.1.2. Diseño FLED-T1

En este apartado se realiza el desarrollo teórico del sistema difuso de detección de bordes, para lo cual es necesario definir de antemano las características del sistema difuso, estás son resumidas en la Tabla 5.2.

Tabla 5.2 Sistema difuso FLED-T1

Característica	
Tipo	**Mamdani**
T-Norma / S-Norma	**Min / Max**
Funciones de membrecía	**Trapezoidales**
Entradas	**4**
Salidas	**1**

Habiendo definido las características del sistema difuso, a continuación se procede con el diseño del sistema, dicho sistema se propone adaptable, con el fin de que pueda ser ajustado para la obtención de los mejores resultados.

Fuzzificador-T1

El fusificador se encarga de la evaluación de todas las funciones de membrecía del antecedente por lo tanto, su costo computacional es proporcional a la cantidad de funciones de membrecía del antecedente en el sistema. La Fig. 5.1, muestra la cantidad de funciones de membrecía y distribución en el sistema propuesto.

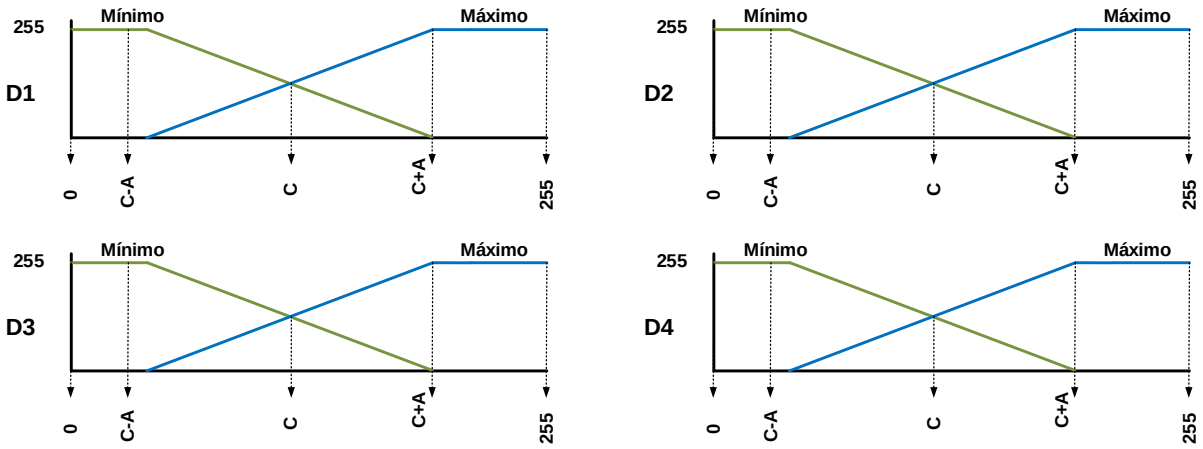


Figura 5.1 Distribución de conjuntos difusos de entrada

La distribución de las funciones de membrecía puede ajustarse en función de dos variables, la variable C, que representa el centro de la intercepción entre las dos funciones de membrecía, y la A, que representa el dominio de la intercepción de las funciones de membrecía.

Recordando que el dominio de entrada del sistema difuso son diferencias absolutas, el dominio se encuentra entre 0 y 255 para las imágenes propuestas de 8 bits de

resolución, en este contexto, las variables C y A se relacionan con el brillo y el contraste respectivamente y pueden ser decididos con ayuda del histograma de la imagen con el fin de obtener los resultados más óptimos.

Inferencia y reglas

La inferencia se encarga de la relación antecedente-consecuente de las reglas planteadas en función de las funciones de membrecía del sistema. En este trabajo se hace uso de la relación causal mediante la T-Norma mínimo; las reglas del sistema que permiten la identificación de bordes fueron propuestas en base a conocimientos empíricos del experto, y son las siguientes (5.1).

$$\begin{aligned}
&if\ (D1\ is\ \min) and\ (D2\ is\ \max)\ and\ (D3\ is\ \max)\ and\ (D4\ is\ \max)\ then\ Y\ is\ Edge\\
&if\ (D1\ is\ \max) and\ (D2\ is\ \min)\ and\ (D3\ is\ \max)\ and\ (D4\ is\ \max)\ then\ Y\ is\ Edge\\
&if\ (D1\ is\ \max) and\ (D2\ is\ \max)\ and\ (D3\ is\ \min)\ and\ (D4\ is\ \min)\ then\ Y\ is\ Edge\\
&if\ (D1\ is\ \max) and\ (D2\ is\ \max)\ and\ (D3\ is\ \min)\ and\ (D4\ is\ \max)\ then\ Y\ is\ Edge\\
&if\ (D1\ is\ \max) and\ (D2\ is\ \max)\ and\ (D3\ is\ \max)\ and\ (D4\ is\ \min)\ then\ Y\ is\ Edge\\
&if\ (D1\ is\ \max) and\ (D2\ is\ \max)\ and\ (D3\ is\ \max)\ and\ (D4\ is\ \max)\ then\ Y\ is\ Edge
\end{aligned} \tag{5.1}$$

Es importante señalar que todas las reglas hacen referencia a un mismo consecuente, esto es interesante porque de esta manera, el costo computacional de la defusificación se reduce drásticamente, todo esto mediante un desarrollo teórico - matemático como se describe a continuación.

Defusificación

El método de defusificación FoM (First of Maxima), como se describe en el capítulo 3, emplea el primer valor del núcleo del conocimiento agregado *B* como el valor defusificador.

Recordando, el conocimiento agregado se obtiene mediante la aplicación de la operación S-Norma a los conocimientos parciales de cada regla como se muestra en (5.2).

$$B = \oplus Br_{r=1}^{6} \tag{5.2}$$

Este conocimiento parcial de cada regla no es otro que la fuerza de disparo de la regla relacionada con la función de membresía del consecuente mediante el operador T-Norma, como se muestra en (5.3).

$$B = \oplus(Gr \otimes Fr)_{r=1}^{6} \tag{5.3}$$

Recordando que todas las reglas son asociadas al a misma función de membrecía del consecuente, el conocimiento agregado *B* se puede expresar como (5.4).

$$B = G_{edge} \otimes \oplus Fr_{r=1}^{6} \quad (5.4)$$

Sustituyendo los operadores T-Norma y S-Norma propuestos en el trabajo, la ecuación del conocimiento agregado queda como sigue (5.5).

$$B = \min(G_{borde}, FR\ Máxima) \quad (5.5)$$

La Fig. 5.2 muestra la distribución de las funciones de membresía del consecuente, con dominios establecidos en función de nuestras variables de entrada y salida, es decir, datos de 8 bits de resolución.

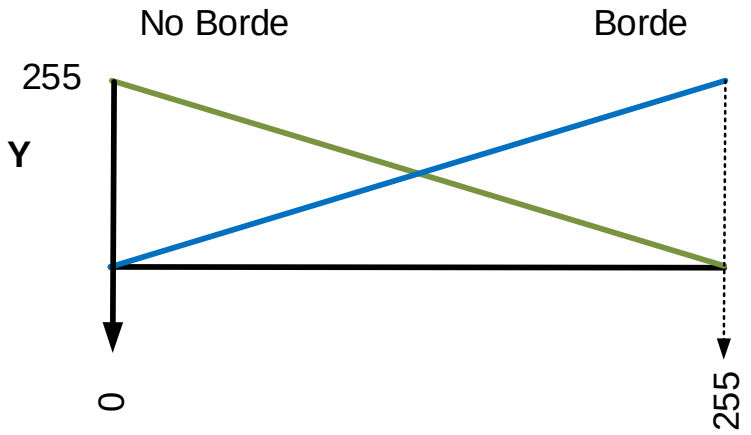


Figura 5.2 Distribución de conjunto difuso de salida.

Basados en la distribución propuesta y la ecuación (6), la Fig. 5.3 muestra una apreciación grafica del conocimiento agregado del sistema para cualquier caso.

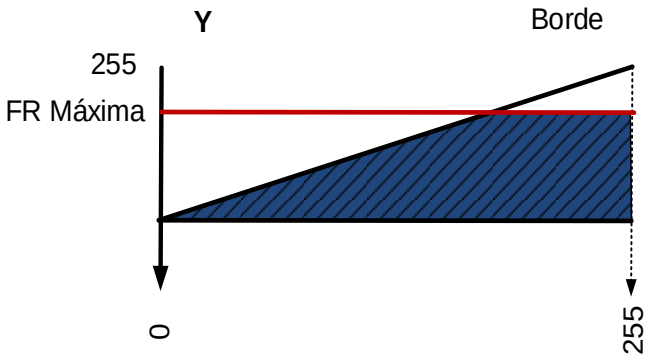


Figura 5.3 Conocimiento agregado del sistema

Se puede observar que es posible el uso del teorema de triángulos semejantes para la inferencia de la salida aproximada como se muestra en (5.6).

$$\frac{\bar{\tilde{y}}}{MaxFR} = \frac{255}{255} \quad \therefore \quad \tilde{y} = FR\ Máxima \tag{5.6}$$

De esta manera, el proceso de defusificación se reduce matemáticamente a la obtención del valor máximo de las fuerzas de disparo de las reglas.

5.1.3. Detección de bordes empleando IT2FL

En el presente trabajo proponemos un sistema IT2FL basado en el sistema T1 propuesto, es decir, con las mismas características citadas en la tabla X, sin embargo con las consideraciones teóricas adicionales que caracterizan a los sistemas IT2. A continuación se procede con el desarrollo de las etapas del sistema difuso.

Fusificación

La Fig. 5.4 muestra la distribución de los conjuntos difusos de entrada y como se puede observar se emplean los mismos parámetros del sistema T5.

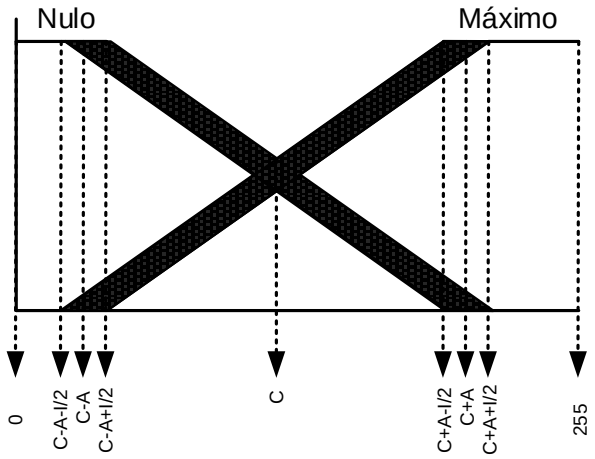


Figura 5.4 Funciones de membrecía de entrada

Se puede observar que la distribución depende de las variables C y A explicadas en el apartado X, sin embargo, se incluye la variable I para hacer presente la huella de incertidumbre FOU.

En la presente tesis, se consideran los conjuntos difusos tipo 2 como dos conjuntos difusos tipo 1 superpuestos y que generan la huella de incertidumbre. Las implicaciones en costo computacional son que el uso de dos conjuntos difusos tipo 1 para generar el conjunto difuso tipo 2 duplica el costo computacional del proceso de fusificación.

Base de reglas e inferencia

La base de reglas e inferencia funciona de igual forma que en un sistema difuso tipo 1 sin embargo, aunque se evalúan la misma cantidad de reglas, el hecho de tener dos conjuntos difusos tipo 1 por cada sistema difuso tipo 2, implica que por cada regla es necesario el cálculo de dos fuerzas de disparo en lugar de una, duplicando el costo computacional al igual que en el proceso de fusificación.

Reducción de tipo y defusificación

La reducción de tipo es uno de los procesos con mayor costo computacional y del cual existen diversas variante como los realizados en [10].

El método de reducción de tipo y defusificación empleado se propone como una aproximación del mismo con el fin de reducir el coste computacional. Pará poder emplear este método es importante que se cumpla la siguiente condición dada por la ecuación (5.7).

$$hl = \frac{\overline{\mu}_l + \underline{\mu}_l}{2}, \Delta l = \frac{\overline{\mu}_l - \underline{\mu}_l}{2} \tag{5.7}$$

$$\therefore \overline{Y} = \frac{\overline{y}_l + \underline{y}_l}{2} \leftrightarrow \frac{\sum_{l=1}^{M} \Delta_l}{\sum_{l=1}^{M} h_l} \ll 1$$

De acuerdo a la distribución y parámetros planteados se llega a las conclusiones expresadas en la ec. (5.8).

$$hl = \frac{m(2x+\frac{I}{2})}{2}, \Delta l = \frac{mI}{4}, \quad \frac{\Delta l}{hl} = \frac{\frac{mI}{4}}{\frac{m(2x+\frac{I}{2})}{2}} = \frac{I}{4x+I} \tag{5.8}$$

En la Fig.5.5 se realiza una simulación de la ec. (5.8) para la verificación de la condición para el uso de la reducción de tipo aproximada, acotada en el dominio de amplitudes he incertidumbres representativas del sistema a emplear, se puede observar que el resultado oscila entre 0 y 0.2 por lo cual se concluye que puede aproximarse la reducción de tipo de la manera expresada en la ec. (5.7).

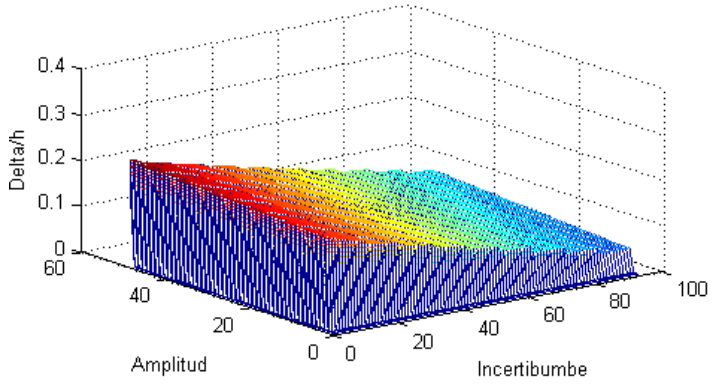


Figura 5.5: Superficie Δ/h

5.2. Arquitectura de Sistema Difuso T1

Habiendo definido el sistema difuso se procede con el desarrollo de la arquitectura, a continuación se desarrolla el diseño de la arquitectura del sistema difuso T5.

Es importante mencionar que la estrategia de diseño se basa en las estrategias mencionadas en el Capítulo 4.

5.2.1. Módulo TrapMF

El primer módulo es el TrapMF, como su nombre lo indica es el encargado de la evaluación de la función de membrecía trapezoidal. En este módulo se opta por hacer uso del paralelismo, como se explicó en el Capítulo 3, la función trapezoidal es una función no lineal que consiste en 4 diferentes intervalos, con el fin de acelerar el cálculo, el módulo TrapMF que se muestra en la Fig. 5.6 realiza la

evaluación de los 4 intervalos simultáneamente al tiempo que evalúa el intervalo correspondiente al valor de entrada.

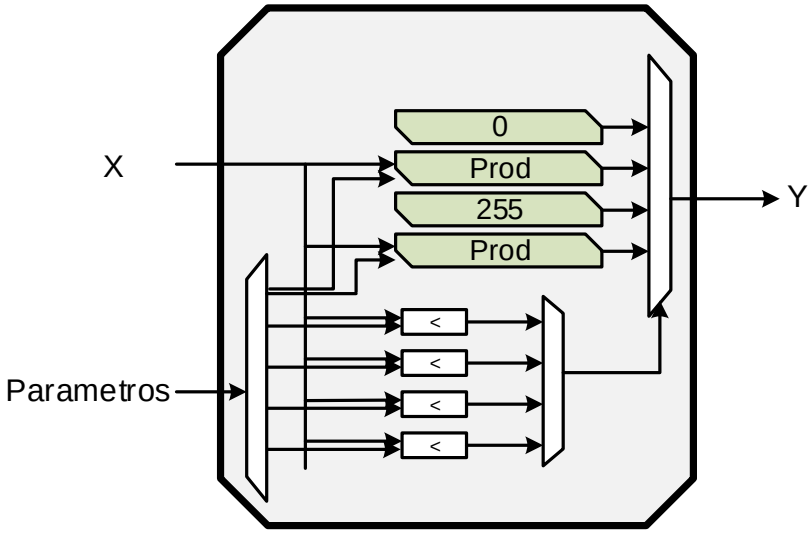


Figura 5.6 Modulo TrapMF

Cabe mencionar que el módulo TrapMF contiene dentro de sus entradas un dato de 42 bits que sintetiza la información correspondiente a la función de membrecía que se desea evaluar, es decir, se trata de un módulo parametrizado que puede ser usado en diferentes contextos y adaptado sin necesidad de una nueva síntesis. La Tabla 5.3 muestra los recursos empleados en el módulo.

Tabla 5.3 Recursos TrapMF

Característica	
Multiplicadores	**1**
Slices	**0%**
LUT	**0%**

5.2.2. Módulo Fusificación

El modulo Fuzz es el encargado de la evaluacion de todas las funciones de membrecia de entrada, haciendo uso del diseño jerarquico pues contiene dentro de si un bloque TrapMF, la estrategia empleada para la fusificacion es un diseño basado en la segmentacion o pipelining, se evaluan las funciones de membrecia secuencialmente y posteriormente se almacenan en registros para cada evaluación. La Fig. 5.7 muestra el esquema general del modulo.

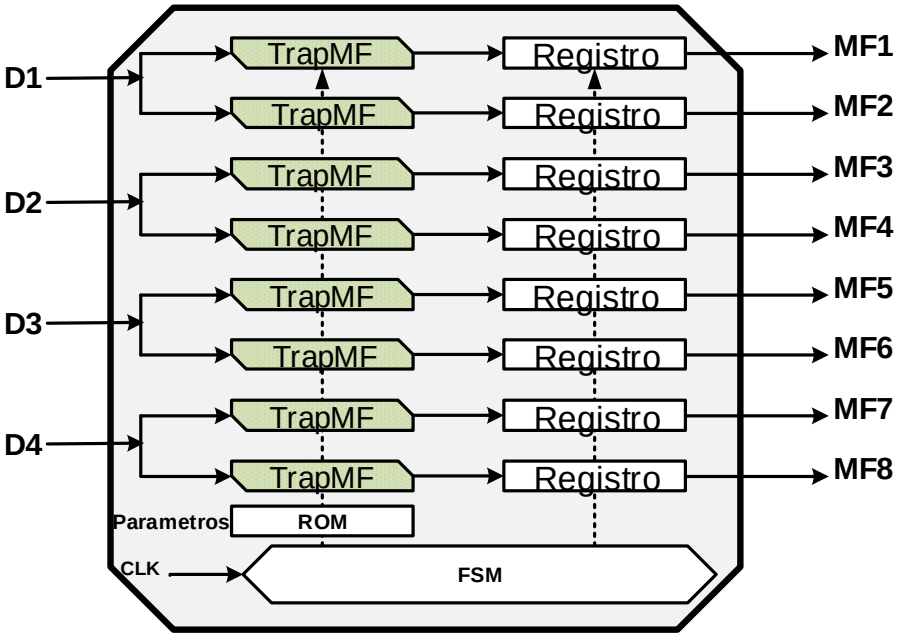


Figura 5.7 Modulo Fusificación.

El desarrollo secuencial se logra mediante una maquina de estados finitos (FSM) que se puede observar en la Fig. 5.8.

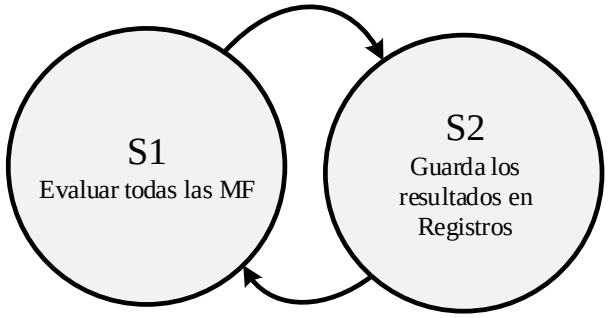


Figura 5.8 FSM Fusificación

La Tabla 5.4 muestra los recursos empleados en el módulo.

Tabla 5.4 Recursos Fusificación

Característica	
Multiplicadores	**8**
Slices	**0%**
LUT	**0%**

5.2.3. Modulo Inferencia/Regla

El módulo de Inferencia/Reglas se puede observar en la Fig. 5.9, realiza por medio del paralelismo/pipelining la evaluación simultánea de todas las reglas, la

evaluación simultanea se realiza usando paralelismo y se emplea pipelining para poder realizar la evaluación/almacenado en un mismo tiempo de ejecución, en consecuencia, los datos contienen un tiempo de retraso, es decir, una latencia de 5.

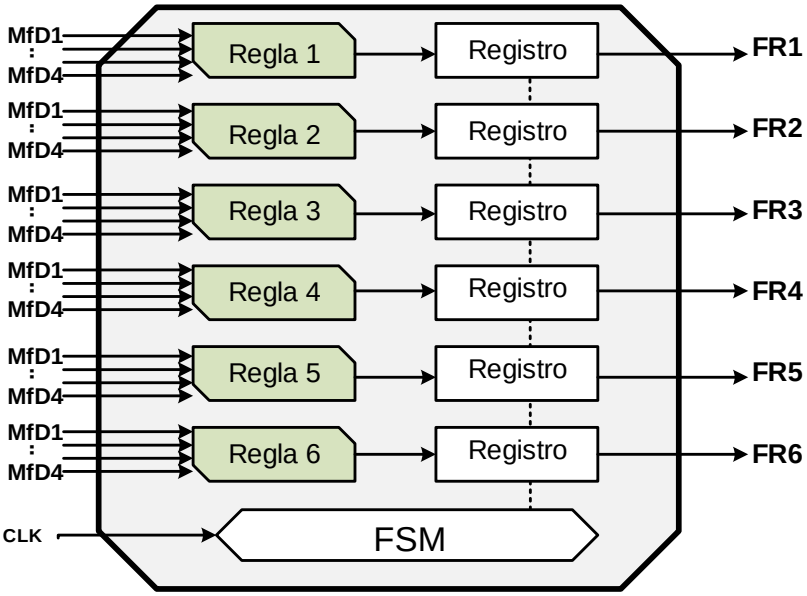


Figura 5.9 Modulo Inferencia

Siguiendo la misma estrategia empleada en el módulo TrapMF, la FSM empleada se muestra en la Fig. 5.10

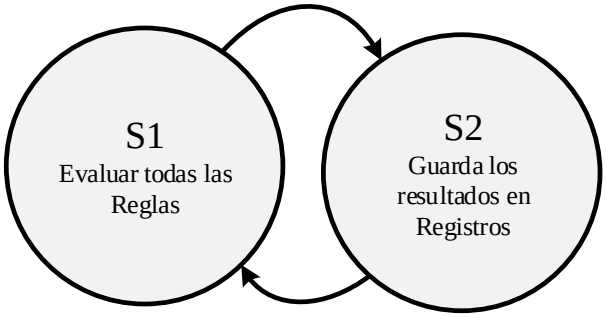


Figura 5.10 FSM Inferencia

La Tabla 5.5 muestra los recursos empleados en el módulo.

Tabla 5.5 Recursos Inferencia

Característica	
Multiplicadores	**0**
Slices	**0%**
LUT	**0%**

5.2.4. Módulo Defusificación

El módulo Defusificación se encarga de realizar la agregación del conocimiento y su posterior conversión a números duros.

El funcionamiento se basa en la ecuación (5.6) y la misma estrategia empleada en los anteriores módulos, la Fig. 5.11 muestra la organización interna del módulo.

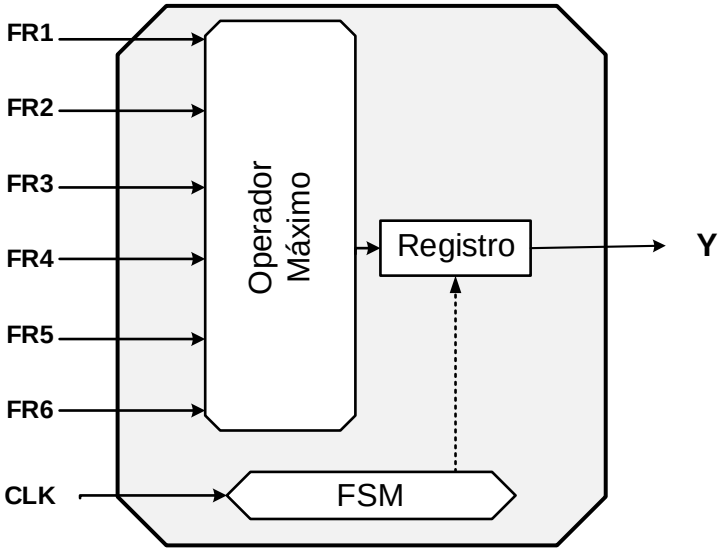


Figura 5.11 Modulo Defusificación

En este módulo de nueva cuenta se hace uso de una FSM para el manejo de los datos y esta es mostrada en la Fig. 5.12.

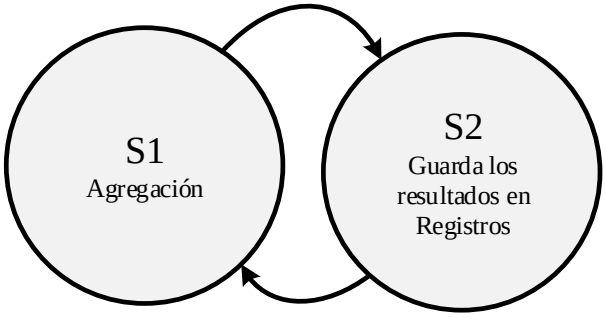


Figura 5.12 FSM Defusificación.

La Tabla 5.6 muestra los recursos empleados en el módulo.

Tabla 5.6 Recursos Defusificación

Característica	
Multiplicadores	**1**
Slices	**0%**
LUT	**0%**

5.2.5. Fuzzy Edge Detector Architecture

Haciendo uso de los módulos mediante el diseño jerárquico, se realiza la integración de la arquitectura del sistema difuso FLED-T1 , donde los módulos se conjuntan y se coordinan mediante una máquina de estados finitos que controla el flujo de los datos mediante la activación o desactivación de etapas internas de cada módulo, haciendo uso de pipelining se consigue la mayor tasa de procesamiento posible para estos módulos, siendo el módulo de fusificación el cuello de botella y obteniendo una tasa de procesamiento de 2 ciclos de reloj. La Fig. 5.13 muestra el esquemático de la arquitectura FLED-T1.

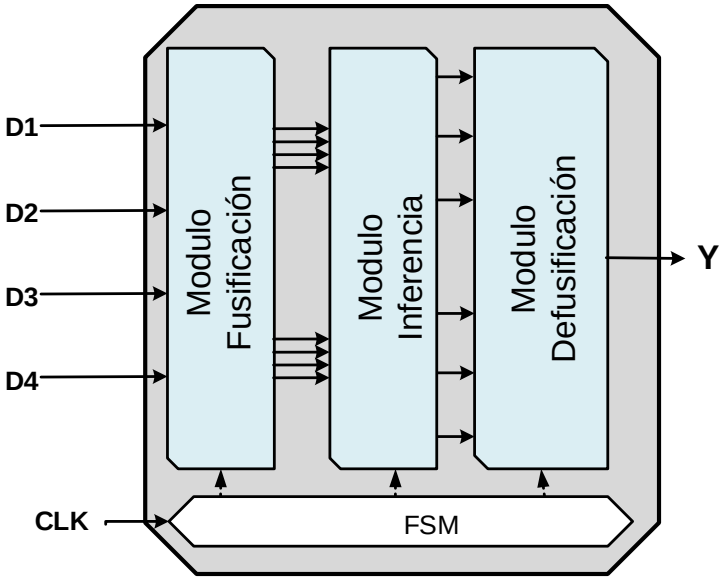


Figura 5.13 Arquitectura FLED-T1

La Tabla 5.7 muestra los recursos empleados en el módulo.

Tabla 5.7 Recursos Arquitectura FLED-T1

Característica	
Multiplicadores	**8**
Slices	**1%**
LUT	**1%**

5.2.6. Arquitectura FLED-T2

Para el diseño de la arquitectura FLED-T2 se hace uso del diseño jerárquico, y como se observó en el capítulo 3, el sistema difuso tipo 2 por intervalos se puede aproximar mediante el promedio de dos evaluaciones de tipo 1, la Fig. 5.14 muestra la arquitectura FLED-T2.

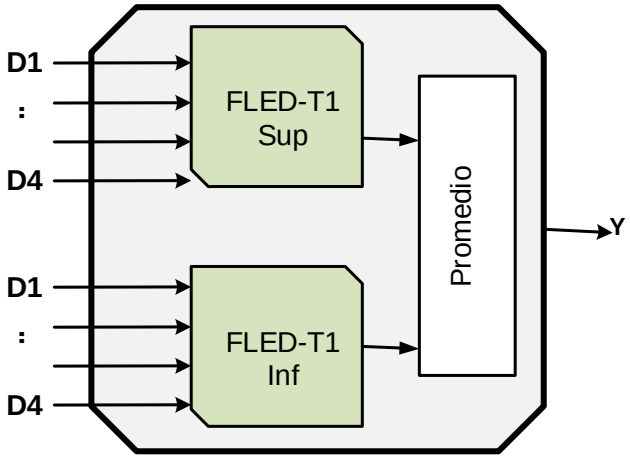


Figura 5.14 Arquitectura FLED-T2

La Tabla 5.8 muestra los recursos empleados en la arquitectura FLED-T2.

Tabla 5.8 Recursos Arquitectura FLED-T2

Característica	
Multiplicadores	**16**
Slices	**2%**
LUT	**2%**

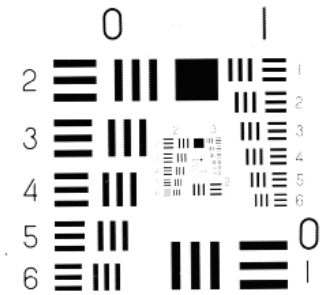
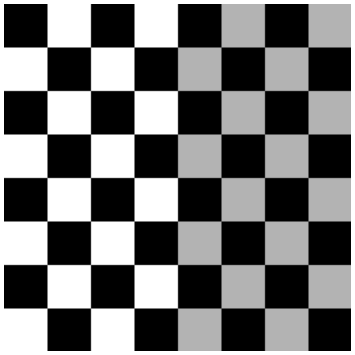
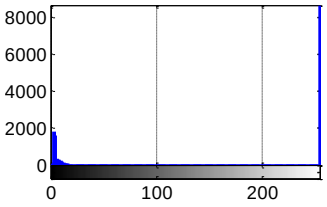
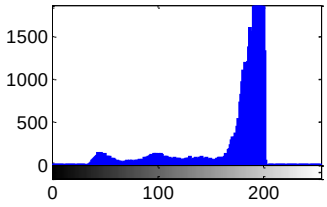
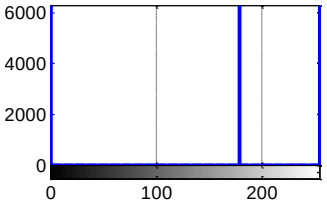
Capítulo 6

Experimentos y Evaluación de Rendimiento

En el presente capítulo se presentarán los experimentos realizados con el fin de evaluar el rendimiento de la arquitectura FLED-T2 y poder de esta forma poder realizar un análisis de resultados con respecto a las metas propuestas.

El experimento consiste en evaluar la tasa de procesamiento de la arquitectura en la plataforma de implementación, el experimento 2 consiste en la evaluación de la calidad de la misma con respecto a diferentes detectores de bordes convencionales, el experimento 3 consiste en evaluar la calidad de la detección de bordes de la FLED-T2 para diferentes niveles de perturbaciones gaussianas, el experimento 4 consiste en la evaluación de la fidelidad de la implementación del sistema difuso en la arquitectura de hardware con respecto al modelo teórico sin errores de truncamiento. A continuación la Tabla 6.1 muestra las imágenes empleadas en las pruebas y sus respectivos histogramas.

Tabla 6.1 Imágenes empleadas

"ResolutionChart.tiff"	"Jellybeam.tiff"	"Chessboard"
		
		

6.1. Experimento 1 Tasa de procesamiento FPGA

En esta prueba se realizara la confirmación de la tasa de procesamiento, en este experimento se abordara el funcionamiento individual de cada módulo y posteriormente el procesamiento en la arquitectura completa. Los experimentos son realizados con ayuda de la plataforma de simulación de Xilinx Sim Scope. El periodo de reloj empleado es 10ns de acuerdo a la frecuencia ofrecida por el FPGA seleccionado como plataforma.

6.1.1. TrapMF

El módulo TrapMF encargado de evaluar las funciones de membrecía, de acuerdo al ciclo de reloj disponible en el FPGA seleccionado, la Fig. 6.1 muestra la evaluación de una función de membrecía en diferentes puntos.

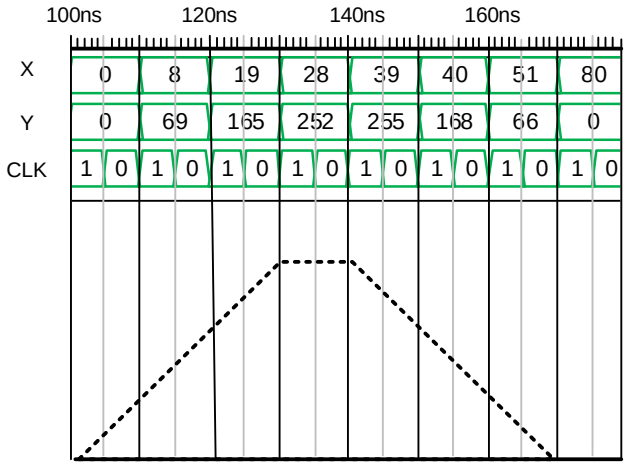


Figura 6.1 Grafica de tiempo de TrapMF

Se puede observar que los resultados de la evaluación son consistentes para una función simétrica. La Tabla 6.2 sintetiza los resultados observados en las simulaciones desarrolladas para el módulo TrapMF.

Tabla 6.2 Rendimiento TrapMF

Característica	
Tasa de procesamiento	**100 MHz**
Latencia	**0**
Consumo de Recursos	**0%**

6.1.2. Fusificador

El módulo fusificador se encarga de la evaluación de todas las funciones de membrecía del antecedente, en este experimento, es evaluado para las mismas entradas que el experimento para el módulo TrapMF, la Fig. 6.2 muestra la respuesta del módulo y su tasa de procesamiento.

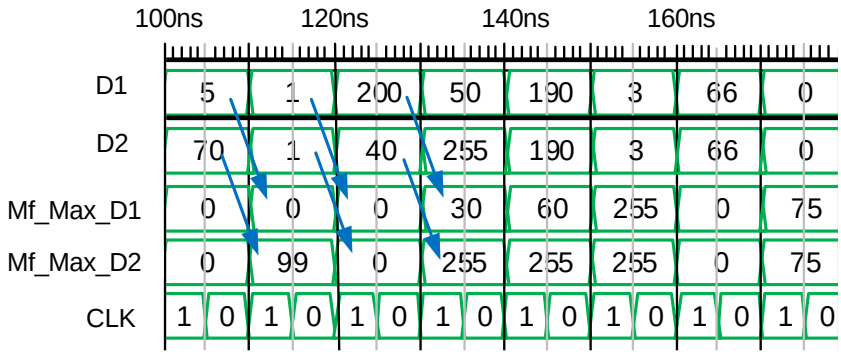


Figura 6.2 Diagrama de tiempo Fusificación

Como se resalta en las flechas azules, los resultados para la fusificación se ven reflejados con un retraso de un ciclo de reloj. La Tabla 6.3 sintetiza los resultados observados en las simulaciones desarrolladas para el módulo fusificador.

Tabla 6.3 Rendimiento Fusificación

Característica	
Tasa de procesamiento	**50 MHz**
Latencia	**2**
Consumo de Recursos	**0%**

6.1.3. Inferencia

Para el módulo de inferencia/reglas, se realiza la simulación introduciendo los valores obtenidos de la fusificación, la Fig. 6.3muestra la respuesta del módulo y su tasa de procesamiento.

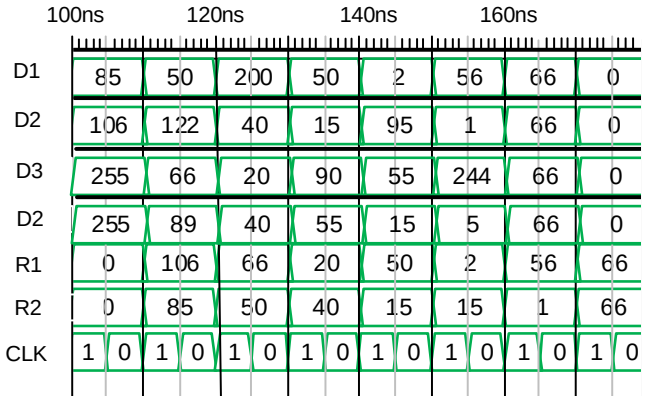


Figura 6.3 Diagrama de tiempo Inferencia

La Tabla 6.4 sintetiza los resultados observados en las simulaciones desarrolladas para el módulo inferencia.

Tabla 6.4 Rendimiento Inferencia

Característica	
Tasa de procesamiento	**50 MHz**
Latencia	**2**
Consumo de Recursos	**0%**

6.1.4. Defusificación

Para el módulo de Defusificación, se realiza la simulación introduciendo los valores obtenidos de la inferencia, la Fig. 6.4 muestra la respuesta del módulo y su tasa de procesamiento.

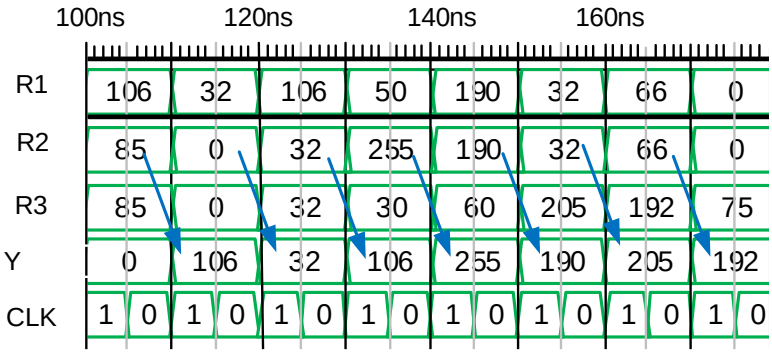


Figura 6.4 Grafica de tiempo Defusificación

La Tabla 6.5 sintetiza los resultados observados en las simulaciones desarrolladas para el módulo Fusificación.

Tabla 6.5 Rendimiento Defusificación

Característica	
Tasa de procesamiento	**50 MHz**
Latencia	**2**
Consumo de Recursos	**0%**

6.1.5. FLED-T2

Para el módulo de inferencia/reglas, se realiza la simulación introduciendo los valores obtenidos de la fusificación, la Fig. 6.5 muestra la respuesta del módulo y su tasa de procesamiento.

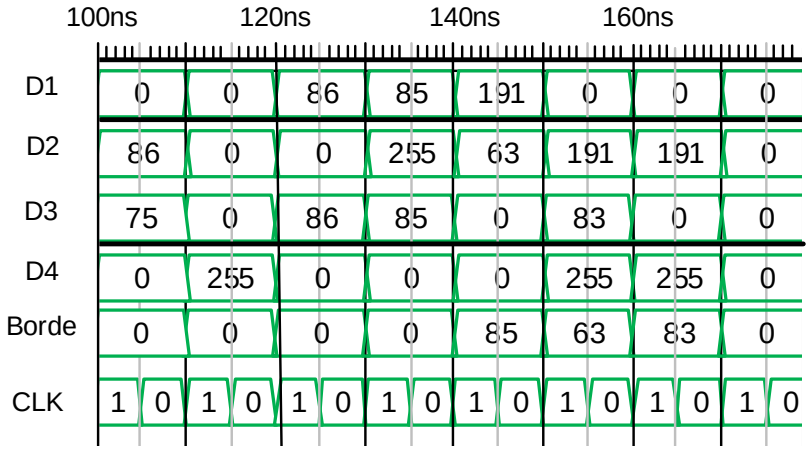


Figura 6.5 Grafica de tiempo de la arquitectura FLED-T2

La Tabla 6.6 muestra los resultados del rendimiento de la arquitectura conjunta.

Tabla 6.6 Rendimiento Arquitectura FLED-T2

Característica	
Tasa de procesamiento	**50 MHz**
Latencia	**2**
Consumo de Recursos	**0%**

6.1.6. Análisis de resultados experimento 1

Del experimento 1 se obtuvo como resultado destacable la latencia de la arquitectura así como la tasa de procesamiento de la arquitectura.

De la tabla X y en el contexto de lógica difusa podemos encontrar que tenemos 50 MFLIPS, es decir, 50 millones de inferencias por segundo, considerando que para la detección de bordes se necesita una inferencia por pixel, la Fig. 6.6 muestra la cantidad de FLIPS necesaria para realizar la detección de bordes en tiempo real para las diferentes resoluciones estándar.

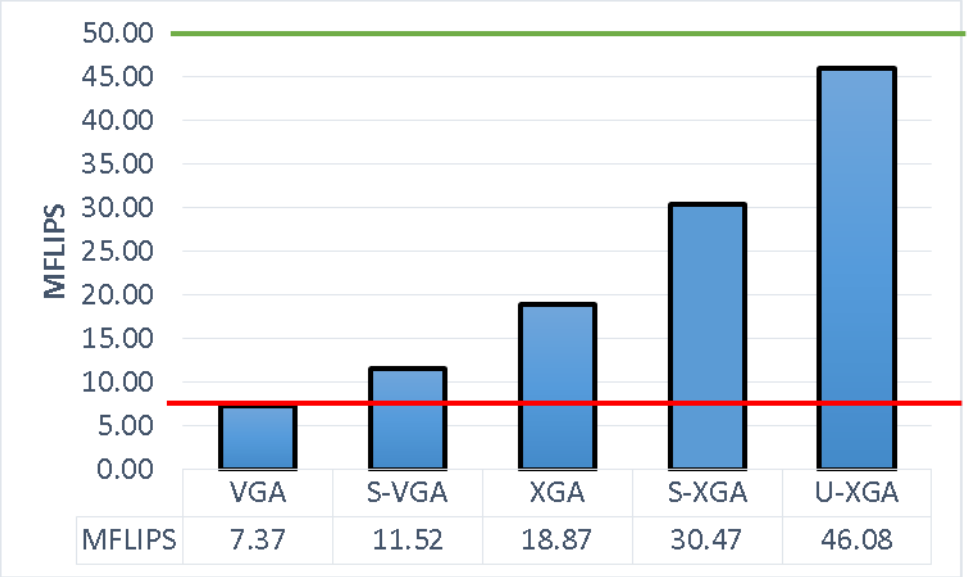


Figura 6.6 Requerimientos en MFLIPS para tiempo real

En la figura se puede observar la proporción en la que crece la demanda de FLIPS para las diferentes resoluciones estándar, la línea roja muestra la meta propuesta, que es aproximadamente 7.4 MFLIPS y la línea verde correspondiente a los resultados obtenidos, es decir, 50 MFLIPS, de lo anterior podemos determinar que los resultados obtenidos permiten realizar un procesamiento en tiempo real para imágenes de una resolución superior a lo propuesto, es decir, alta definición.

6.2. Experimento 2 Calidad de bordes detectados

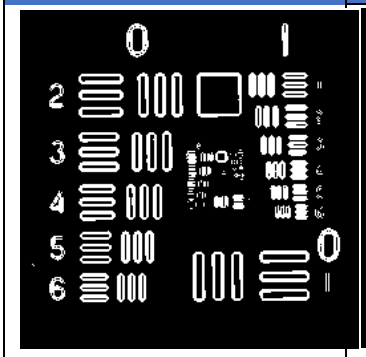
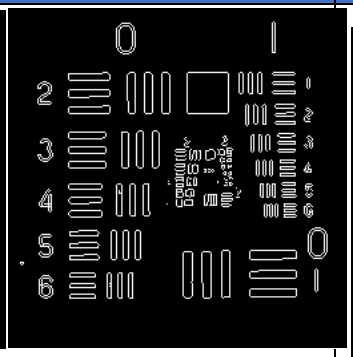
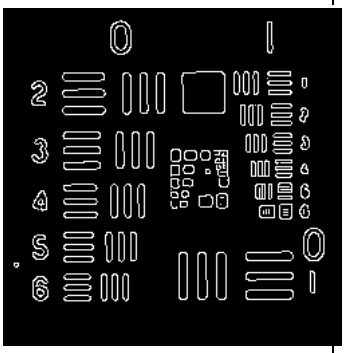
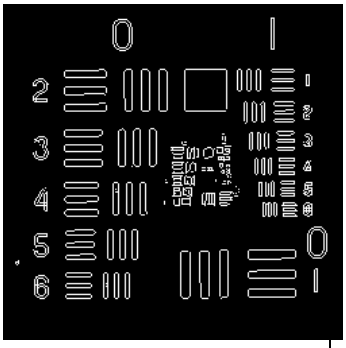
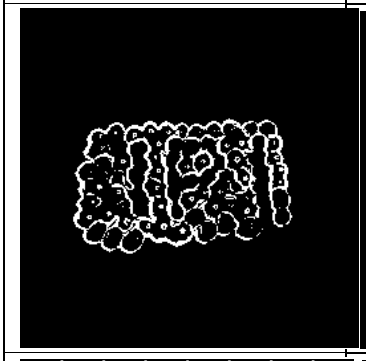
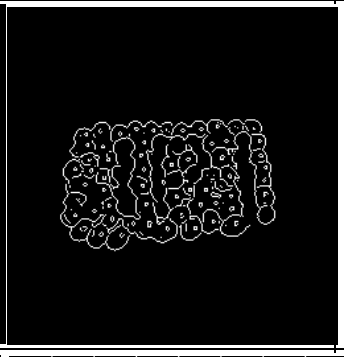
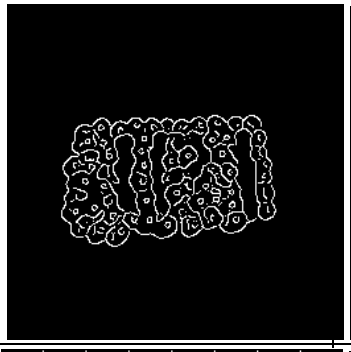
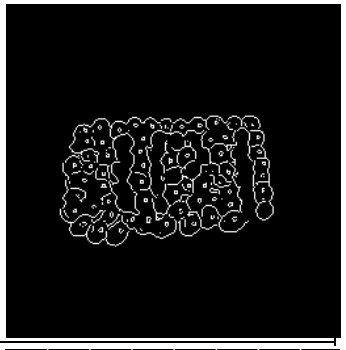
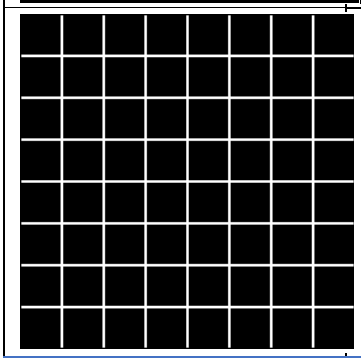
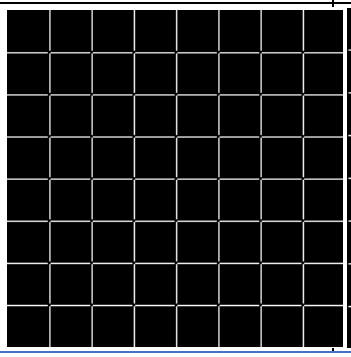
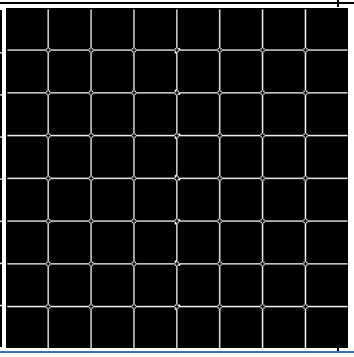
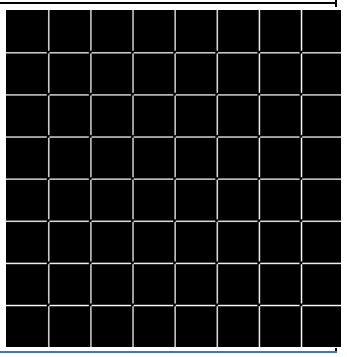
Para hacer uso de la métrica de la FOM de Pratt, es necesaria una imagen de referencia para la evaluación del detector de bordes, en el presente experimento se evalúa la calidad de bordes detectados mediante la FOM de Pratt con respecto a tres de los detectores de bordes más empleados actualmente. Los resultados se pueden observar en la Tabla 6.7.

Tabla 6.7 Calidad del FLED-T2 con respecto a detectores convencionales

FOM de Pratt			
Imagen	Sobel	Canny	Prewitt
Resolution Chart	0.8571	0.8488	0.8553
Jelly	0.8826	0.8534	0.8806
Chess	0.8691	0.8862	0.8691

Los bordes evaluados en la Tabla 6.7 se pueden observar en la Tabla 6.8.

Tabla 6.8 Resultados obtenidos por detectores de bordes

FLED-T2	Sobel	Canny	Prewitt
			
			
			

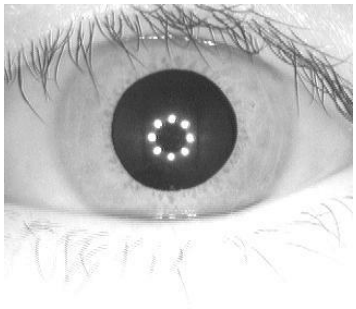
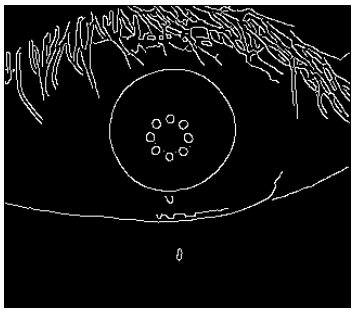
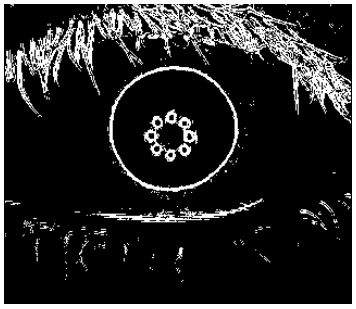
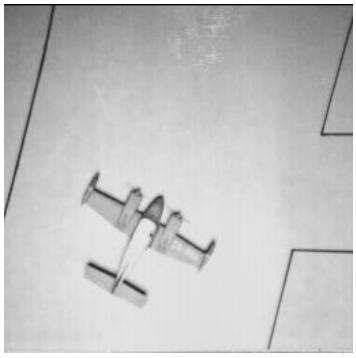
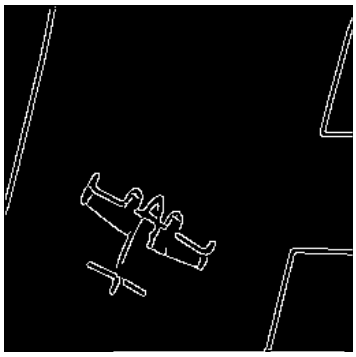
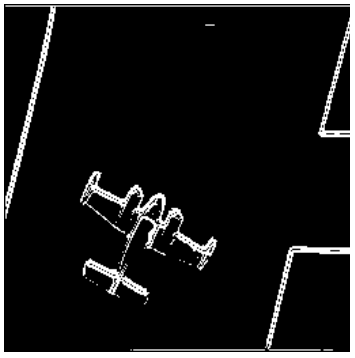
6.2.1. Análisis de resultados experimento 2

La calidad evaluada en el experimento 2, la tabla 6.7, se observa como el índice de calidad, la FOM de Pratt, obtiene valores cercanos al 90% del total.

Se puede decir que la calidad de un borde detectado, en imágenes convencionales, es una medición relativa, puesto que el índice de calidad depende de la imagen ideal considerada.

Sin embargo, la imagen ideal puede ser decidida de acuerdo al contexto de aplicación, por ejemplo, la Tabla 6.9 muestra imágenes de diferentes contextos para los cuales el detector FLED-T2 podría ser considerado como el detector ideal o mejor que los detectores convencionales. Las imágenes corresponden a aplicaciones de biometría y protección civil.

Tabla 6.9 Detección de bordes en diferentes contextos

Imagen	Bordes por Canny	Bordes por FLED-T2
		
		
		

6.3. Experimento 3: Rendimiento ante perturbaciones

La robustez de un algoritmo es un atributo importante, en procesamiento de imágenes toma mayor importancia debido a los ambientes no controlados a los que se someten los algoritmos.

Para evaluar el rendimiento del FLED-T2 ante perturbaciones, se evalúa la FOM de Pratt para la detección de bordes de una imagen contaminada con ruido gaussiano, usando como imagen ideal la detección de bordes de la misma imagen sin perturbaciones. La Tabla 6.10 muestra la FOM de Pratt para imágenes contaminadas con diferentes varianzas de ruido gaussiano.

Tabla 6.10 FOM de imágenes contaminadas

Varianza	Resolution Chart	Jellybean	Chessboard
0	1	1	1
0.001	0.9742	0.8544	0.9999
0.002	0.9538	0.8175	0.9938
0.003	0.9421	0.8263	0.9738
0.004	0.9352	0.7824	0.9586
0.005	0.9267	0.7023	0.9381
0.01	0.9126	0.5844	0.8346
0.02	0.9219	0.518	0.738

Los resultados obtenidos en la Tabla 6.10 se pueden graficar mediante una gráfica logarítmica como se muestra en la Fig. 6.7.

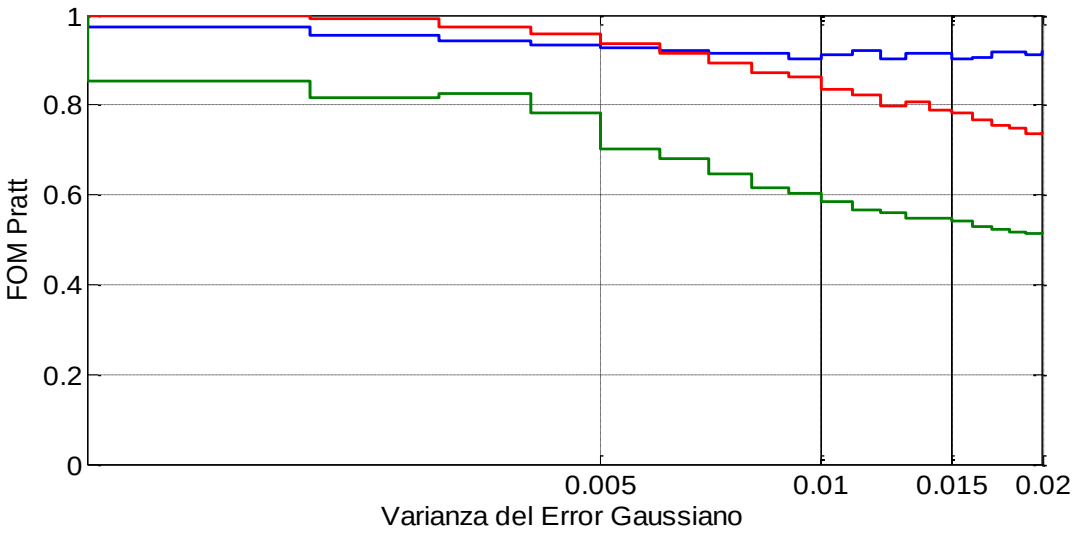
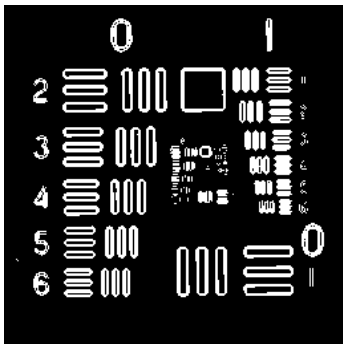
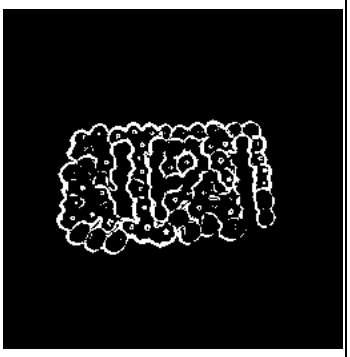
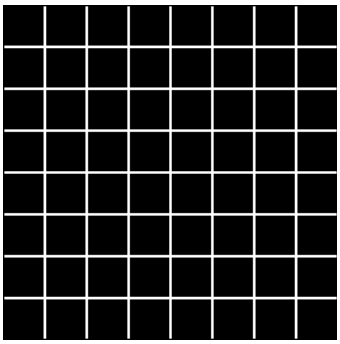
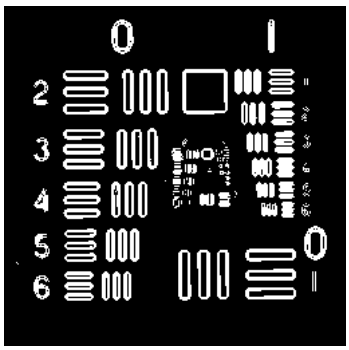
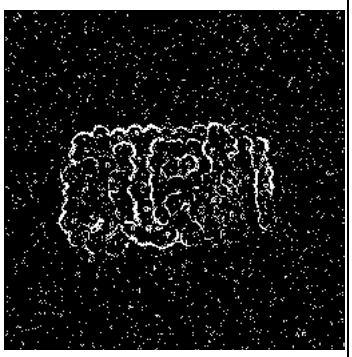
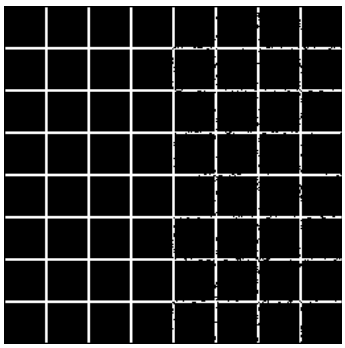
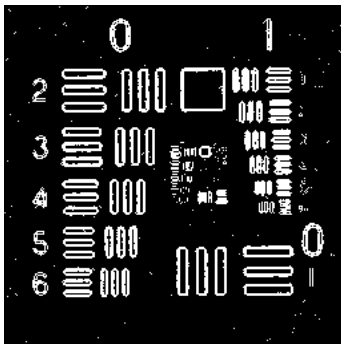
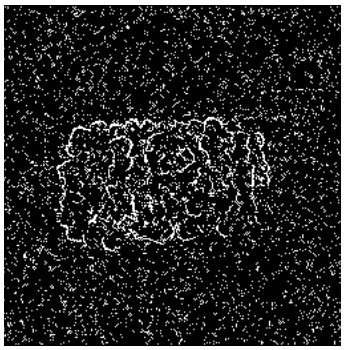
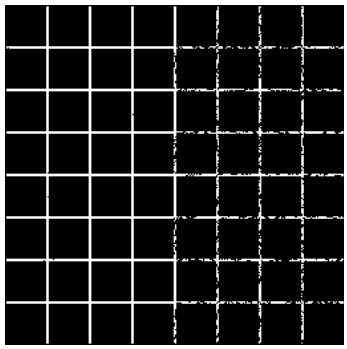
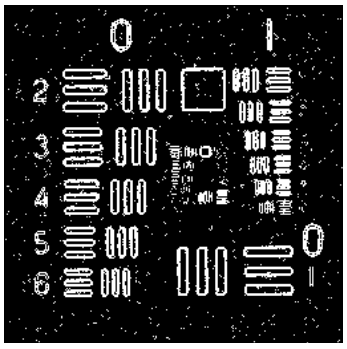
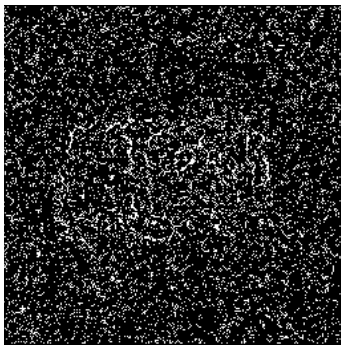
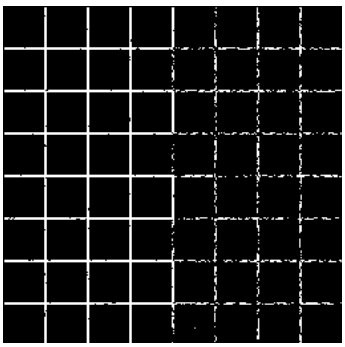


Figura 6.7 Rendimiento del FLED-T2 ante perturbaciones gaussianas.

En la Tabla 6.11 se muestra un ejemplo de la evaluación de bordes para las imágenes contaminadas con ruidos de diferentes varianzas.

Tabla 6.11 Detección de bordes para imágenes contaminadas con ruido gaussiano

Var.	"ResolutionChart.tiff"	"Jellybeam.tiff"	"Chessboard"
Referencia			
0.0005			
0.0010			
0.0020			

6.3.1. Análisis de resultados experimento 3

En términos del rendimiento ante perturbaciones hace falta poner en contexto este rendimiento, en primer lugar ante los detectores convencionales, a continuación, la Tabla 6.12 y la Fig. 6.8 muestran los rendimientos del detector de Canny en las mismas condiciones de perturbación.

Tabla 6.12 FOM Imágenes contaminadas Canny

Varianza	Resolution Chart	Jellybean	Chessboard
0	1	1	1
0.001	0.9953	0.5782	0.8287
0.002	0.8542	0.5712	0.6895
0.003	0.7173	0.5707	0.6267
0.004	0.6742	0.5685	0.5607
0.005	0.6578	0.5686	0.5478
0.01	0.6367	0.5671	0.5295
0.02	0.629	0.565	0.5152

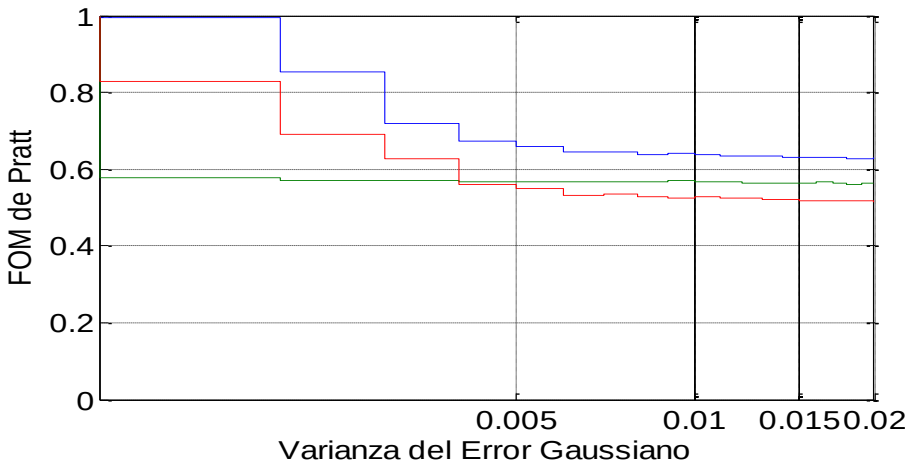


Figura 6.8 Rendimiento de Canny ante perturbaciones gaussianas.

Como se puede observar el rendimiento obtenido por la FLED-T2 es considerablemente superior al rendimiento del detector Canny, en términos de tolerancia a perturbaciones.

Adicionalmente a esto se puede observar algo muy interesante en la prueba 3 y es el hecho de que el sistema se ve seriamente afectado para imágenes de bajo contraste, por ejemplo, véase la Tabla 6.10. Para la imagen “Resolution Chart.tiff” el rendimiento es muy bueno independientemente de las perturbaciones, sin embargo, el rendimiento es muy bajo para la imagen “Jellybeans.tiff” la cual es de bajo contraste, y esto se puede observar claramente con la imagen sintética “Chess” la cual tiene alto contraste y bajo contraste en igual medida y dividido mitad y mitad, se observa que el bajo rendimiento se obtiene en la regio de bajo contraste. Una posible explicación es que el ruido gaussiano tiene un mayor impacto en regiones de bajo contraste debido a que es un ruido de bajo contraste y en esas condiciones

la relación señal a ruido es mucha, por lo tanto, se maximiza el efecto de las perturbaciones.

6.4. Experimento 4: Fidelidad al modelo teórico.

A causa de la discretización inherente a los sistemas digitales, es importante la evaluación de la fidelidad o el error de la aproximación en hardware contra el modelo teórico, esto es debido a que pequeños errores en cualquier etapa del sistema difuso, pueden provocar grandes perturbaciones en el sistema a la salida.

En el FPGA como en todos los sistemas digitales, uno de las mayores fuentes de error es el truncamiento, en FPGA la información empleada se maneja a nivel de bits, esto significa que en ese nivel se pierde toda la información contenida en la parte fraccionaria, es por ello que en el sistema propuesto se maneja la estrategia de notación de punto fijo, en un intento por rescatar cifras significativas de la parte fraccionaria.

En la arquitectura propuesta, la mayor Fuente de error se encuentra en la etapa de fusificación, esto debido a que los errores de truncamiento se magnifican en los productos y como tal, la evaluación de las funciones de membrecía trapezoidales se centran en ecuaciones de la recta, tal y como se expresa en la ecuación (6.1).

$$y = mx + b \tag{6.1}$$

El error se puede expresar como la diferencia entre la evaluación de los productos de la variable independiente por la pendiente real y la pendiente con error de truncamiento como sigue (6.2).

$$e = m_r x - m_t x \tag{6.2}$$

Donde mr es la pendiente real, mt representa la pendiente discretizada con errores de truncamiento y x es la variable independiente, en nuestro contexto, la entrada de la función de membrecía es decir, la magnitud del gradiente. La ecuación se puede expresar como (6.3) hacienda uso de la factorización.

$$e = x(m_r - m_t) \tag{6.3}$$

La pendiente es diferente para cada intervalo de la función de membrecía pero surge del concepto básico de pendiente, expresado en (6.4).

$$m = \frac{\Delta y}{\Delta x} \tag{6.4}$$

La pendiente truncada puede expresarse en función de la pendiente real como sigue (6.5) se reemplaza el valor de la dY por 255 debido a que el contexto de aplicación del sistema nos dice que trabajaremos con datos de 8 bits por lo cual el mayor número decimal seria 255

$$m_r = \frac{255 - m_t \Delta x}{\Delta x} + m_t \, , x \leq \frac{255}{m_t} \tag{6.5}$$

Reemplazando (13) en la ecuación del error se obtiene (6.6).

$$e = x\left(\frac{255 - m_t \Delta x}{\Delta x} + m_t - m_t\right) \tag{6.6}$$

Y finalmente, simplificando se puede expresar como sigue.

$$e = x\left(\frac{255 - m_t \Delta x}{\Delta x}\right), x \leq \frac{255}{m_t} \tag{6.7}$$

Como se puede observar en (15) el comportamiento del error es no lineal y es directamente proporcional a x, pero a sus ves, inversamente proporcional a mt, lo cual lo hace tener un error creciente con respecto a las x, la evaluación de (6.7) se observa en la Fig. 6.9.

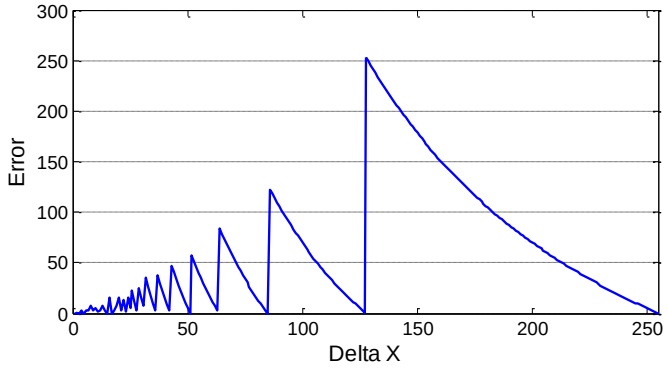


Figura 6.9 Comportamiento del error.

El error observado es cercano al 255, es decir, cerca del 100% y esto es no deseable, debido a esto se emplea la estrategia de notación de punto fijo por lo cual la ecuación de la pendiente se expresa en (6.8).

$$m = \frac{2^b * 255}{\Delta x}, x \in X, (1{,}255) \tag{6.8}$$

Este cambio tiene un impacto en la ecuación del error como se muestra (6.9).

$$e = \frac{\left(x\left(\frac{2^b * 255 - m_t \Delta x}{\Delta x}\right)\right)}{2^b}, x \leq \frac{255}{m_t} \tag{6.9}$$

La Fig. 6.10 y la Fig. 6.11 muestran el comportamiento para 1 y 3 bits fraccionales respectivamente.

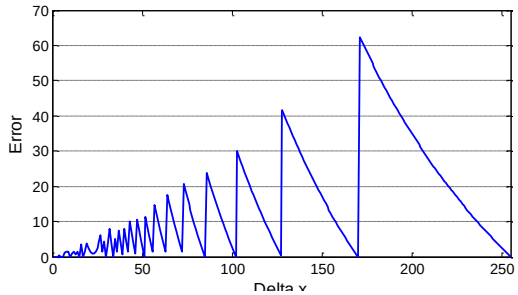


Figura 6.10 Comportamiento del error para 1 bit fraccional.

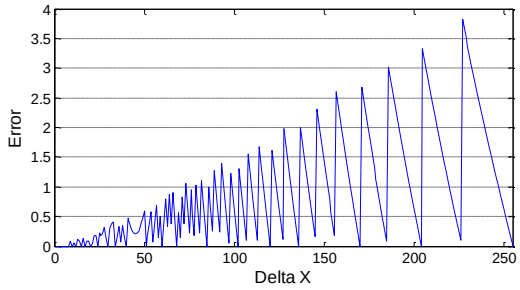


Figura 6.11 Comportamiento del error para 3 bits fraccionales.

El error se reduce a 63 y 4 respectivamente, esto representa cerca del 25% y 1.5%.

Poniendo en contexto las ecuaciones obtenidas, es importante observar que la delta x tiene relación directa con la A es decir, la amplitud del conjunto difuso.

De acuerdo a esto, la amplitud máxima es 127 y por ende, el caso critico de error por ser directamente proporcional al mismo de acuerdo a la ecuación (6.8).

La Fig. 6.12 muestra la superficie de error para una imagen evaluada para C=127 y A=127.

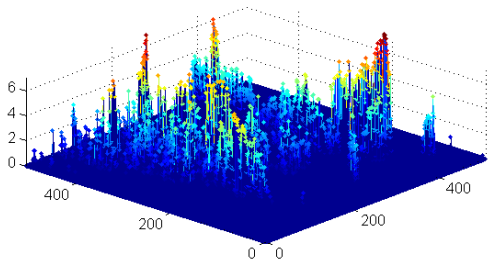


Figura 6.12 Máxima superficie de error

Obteniendo un error máximo cercano al 7% pero no obteniendo una detección de bordes adecuada, la Fig. 6.13 muestra una superficie de error para una distribución realista de los conjuntos de entrada, para una A=30 y C=40.

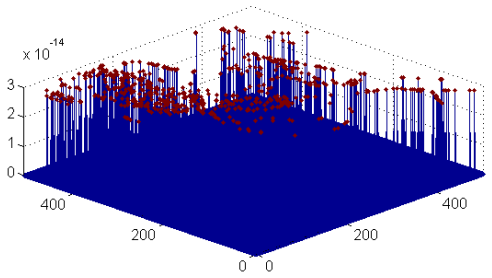


Figura 6.13 Superficie de error para sistema propuesto

De esta manera se observa un error muy aproximado a 0.

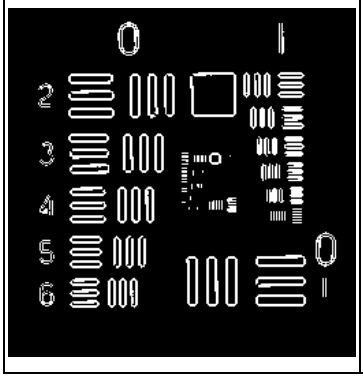
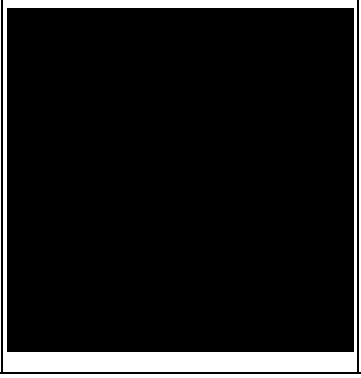
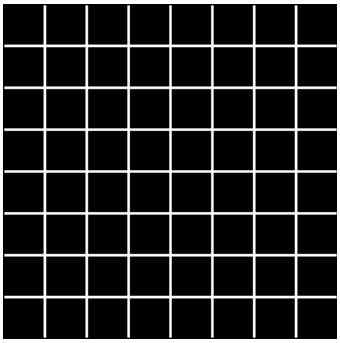
6.4.1. Análisis de resultados Experimento 4

De acuerdo al experimento 4, se demuestra la fiabilidad de la arquitectura FLED-T2 con respecto al modelo teórico, esto es de gran importancia debido a que los resultados obtenidos anteriormente pierden su valor si en caso de la síntesis en hardware se presentan errores adicionales relativos al entorno de aplicación.

Se comprueba que los errores máximos obtenidos por truncamiento son aproximadamente de un 7%, esto puede ser significativo pero considerando el contexto practico de la detección de bordes es tolerable.

Por otro lado se demuestra que para los sistemas prácticos de detección de bordes, el error tiende a cero, para comprobar la poca viabilidad del caso crítico de error se muestra la evaluación de bordes para las imágenes antes manejadas. La Tabla 6.13 muestra los resultados obtenidos.

Tabla 6.13 Detección de bordes para distribución de máximo error

Resolution Chart	Jellybean	Bordes por FLED-T2
		

El caso critico es poco práctico puesto que representaría una imagen muy alto contraste y en esa condición se podrían obtener resultados con una distribución con una menor A, es decir menor superficie de error por truncamiento. En el caso de Jellybean, la cual es una imagen con un histograma distribuido más uniformemente y por ello con mayor contraste, se observa que el detector no fue capaz de determinar ningún borde, esto evidencia la poca practicidad de la distribución que presenta el mayor error de truncamiento.

Capítulo 7

Conclusiones

El presente trabajo demuestra que es viable la implementación de sistemas de lógica difusa, parametrizables en arquitecturas FPGA, con un consumo de recursos controlable, y capaz de realizar procesamiento en tiempo real de imágenes de alta resolución, en contraste con otras estrategias que requieren procesamiento fuera de línea.

Se ha demostrado que la arquitectura propuesta presenta un comportamiento de calidad de detección de bordes equiparable y en condiciones específicas, incluso superior respecto a las técnicas tradicionales, puntualizando al respecto de otras conclusiones podemos decir:

- En el Capítulo 5, el desarrollo del sistema difuso se realizó de manera sencilla y basada en el conocimiento empírico del experto, esto se destaca debido a que a pesar de ser algoritmos de mayor complejidad, los sistemas de lógica difusa de Mamdani pueden ser formulados con base en el conocimiento humano.

- El sistema difuso de la arquitectura FLED tiene la característica de ser ajustable, debido a que es parametrizado, por medio de dos variables *C* e *I* como se explica en el Capítulo 5, estos parámetros se pueden relacionar con el brillo y contraste de la imagen adaptándola así al contexto en que sea necesario, esta característica es importante debido a la gran cantidad de variaciones en imágenes debido al ambiente no controlado de las mismas.

- En cuanto al diseño de la arquitectura se destacan las herramientas o estrategias de diseño que ofrecen los lenguajes HDL, en este caso VHDL, se observa cómo fue posible el diseño modular de la arquitectura gracias a la estrategia de diseño jerárquico y empleando herramientas como paralelismo o pipelining para reducir los ciclos de ejecución de cada módulo de manera individual así como el sistema integrado.

- En cuanto a la cantidad de recursos empleados para la arquitectura se puede observar que el porcentaje de ocupación medido en LUT's es prácticamente 1%, es decir, la limitante es únicamente los bloques multiplicadores, dichos bloques pueden ser sustituidos por módulos multiplicadores ya sea ofrecidos por el fabricante o bien diseñados por el experto, esto permitiría prescindir de los bloques multiplicadores pagando un costo en LUT's, de cualquier manera, la arquitectura como tal puede ser replicada y esto permitiría, de ser necesario, manejar *n* arquitecturas FLED en paralelo con el fin de multiplicar la tasa de procesamiento, tan solo con una correcta administración de los

datos, la esbelta arquitectura puede ser usada como simplemente un módulo de una arquitectura de mayor complejidad.

- Del experimento 1 explicado en el Capítulo 6, se pudo constatar la tasa de procesamiento lograda por la arquitectura FLED, tanto de forma modular como el sistema integrado y se observa que en términos de FLIPS se obtuvo una tasa de procesamiento de 50 MFLIPS, mismo que supera con creces la meta propuesta de 7.3MFLIPS, observando la Fig. 6.X se encuentra que de hecho, la tasa de procesamiento lograda, es incluso la suficiente para lograr una detección de bordes en tiempo real para una imagen U-XGA, esto sin considerar que esta tasa de procesamiento puede ser multiplicada empleando dos arquitecturas FLED como módulos de un sistema de mayor jerarquía.

- En términos de la calidad de bordes detectados, como se evalúa en el experimento 2, es importante recordar que hablar de la calidad de los bordes en imágenes convencionales es una calificación relativa, esto debido a que es necesario una imagen parámetro para comparar. En la presente tesis se decidió por usar la FOM de Pratt y se reportaron resultados cercanos al 0.9, como se explicó en el Capítulo X, los resultados obtenidos son calificaciones relativas a los métodos convencionales de detección de bordes, sin embargo, en diferentes contextos es posible que la arquitectura FLED logre mejor calidad de bordes que los mismos, esto nos lleva a concluir que los resultados obtenidos fueron buenos debido a que guardan consistencia con los métodos convencionales independientemente del contexto.

- El rendimiento a perturbaciones reportado en el experimento 3 y analizado a profundidad en el Capítulo 6, muestra lo que de antemano es conocido de los sistemas difusos, el hecho de ser algoritmos que pueden manejar información incompleta, es decir, la inherente robustez a perturbaciones que los caracteriza. En el experimento se sometió a perturbaciones del tipo gaussiano y se reportaron resultados muy positivos, sin embargo cabe destacar que la arquitectura propuesta es adaptable, esto implica que los resultados reportados no son los mejores posibles, puesto que si se adapta el controlador a la imagen perturbada, este podría responder con mejores rendimientos puesto que la variación del contraste de la imagen son cubiertos con las distribuciones parametrizadas del sistema como se explica en el Capítulo 5.

- De la fidelidad de la arquitectura al sistema difuso, se abordó a profundidad en el Capítulo 6 y se observó cómo influía el error de truncamiento en la detección de bordes, algo que en software parece trivial es importante en hardware y el uso de la notación de punto fijo trajo como resultado aproximaciones muy buenas al sistema difuso teórico, se decidió por hacer uso de 3 bits fraccionarios y se lograron resultados de errores menores al 6%, además se demostró que en realidad los errores máximos se daban para

distribuciones no-prácticas para detección de bordes y por lo tanto se puede concluir que la aproximación de la arquitectura al modelo es muy buena.

Referencias

[1] M. Alimohammadi, J. Pourdeilami, y A. A. Pouyan, “Edge detection using fuzzy inference rules and first order derivation”, en *2013 13th Iranian Conference on Fuzzy Systems (IFSC)*, 2013, pp. 1–5.

[2] W. Barkhoda, F. A. Tab, y O.-K. Shahryari, “Fuzzy edge detection based on pixel’s gradient and standard deviation values”, en *International Multiconference on Computer Science and Information Technology, 2009. IMCSIT ’09*, 2009, pp. 7–10.

[3] Z. Talai y A. Talai, “A fast edge detection using fuzzy rules”, en *2011 International Conference on Communications, Computing and Control Applications (CCCA)*, 2011, pp. 1–5.

[4] X. Chen y Y. Chen, “An Improved Edge Detection in Noisy Image Using Fuzzy Enhancement”, en *2010 International Conference on Biomedical Engineering and Computer Science (ICBECS)*, 2010, pp. 1–4.

[5] Y. Zeng, C. Tu, y X. Zhang, “Fuzzy-Set Based Fast Edge Detection of Medical Image”, en *Fifth International Conference on Fuzzy Systems and Knowledge Discovery, 2008. FSKD ’08*, 2008, vol. 3, pp. 42–46.

[6] J. Wu, Z. Yin, y Y. Xiong, “The Fast Multilevel Fuzzy Edge Detection of Blurry Images”, *IEEE Signal Process. Lett.*, vol. 14, núm. 5, pp. 344–347, may 2007.

[7] S. Sarangi y N. P. Rath, “Performance Analysis of Fuzzy-Based Canny Edge Detector”, en *International Conference on Conference on Computational Intelligence and Multimedia Applications, 2007*, 2007, vol. 3, pp. 272–276.

[8] O. Mendoza y P. Melin, “Interval type-2 fuzzy integral to improve the performance of edge detectors based on the gradient measure”, en *Fuzzy Information Processing Society (NAFIPS), 2012 Annual Meeting of the North American*, 2012, pp. 1–6.

[9] P. Melin, C. I. Gonzalez, J. R. Castro, O. Mendoza, y O. Castillo, “Edge-Detection Method for Image Processing Based on Generalized Type-2 Fuzzy Logic”, *IEEE Trans. Fuzzy Syst.*, vol. 22, núm. 6, pp. 1515–1525, dic. 2014.

[10] C. Gonzalez, J. R. Castro, P. Melin, y O. Castillo, “An edge detection method based on generalized type-2 fuzzy logic”, en *2013 IEEE Symposium on Advances in Type-2 Fuzzy Logic Systems (T2FUZZ)*, 2013, pp. 39–44.

[11] R. C. Gonzalez y R. E. Woods, *Digital Image Processing*, 3 edition. Pearson, 2012.

[12] A. M. Mahmood, H. H. Maras, y E. Elbasi, “Measurement of edge detection algorithms in clean and noisy environment”, en *2014 IEEE 8th International Conference on Application of Information and Communication Technologies (AICT)*, 2014, pp. 1–6.

[13] E. Ataman, V. Aatre, y K. Wong, “Some statistical properties of median filters”, *IEEE Trans. Acoust. Speech Signal Process.*, vol. 29, núm. 5, pp. 1073–1075, oct. 1981.

[14] M. Sharifi, M. Fathy, y M. T. Mahmoudi, “A classified and comparative study of edge detection algorithms”, en *International Conference on Information Technology: Coding and Computing, 2002. Proceedings*, 2002, pp. 117–120.

[15] A. A. Khodaskar y S. A. Ladhake, “Pattern recognition: Advanced development, techniques and application for image retrieval”, en *2014 International Conference on Communication and Network Technologies (ICCNT)*, 2014, pp. 74–78.

[16] I. E. Abdou y W. K. Pratt, “Quantitative design and evaluation of enhancement/thresholding edge detectors”, *Proc. IEEE*, vol. 67, núm. 5, pp. 753–763, may 1979.

[17] L. A. Zadeh, “Fuzzy logic = computing with words”, *IEEE Trans. Fuzzy Syst.*, vol. 4, núm. 2, pp. 103–111, may 1996.

[18] E. H. Mamdani, “Application of fuzzy algorithms for control of simple dynamic plant”, *Proc. Inst. Electr. Eng.*, vol. 121, núm. 12, pp. 1585–1588, dic. 1974.

[19] F. X. Aymerich, P. Sobrevilla, E. Montseny, y A. Rovira, “Detection of hyperintense regions on MR brain images using a Mamdani type Fuzzy Rule-Based System: Application to the detection of small multiple sclerosis lesions”, en *2011 IEEE International Conference on Fuzzy Systems (FUZZ)*, 2011, pp. 751–758.

[20] B. M. Gayathri y C. P. Sumathi, “Mamdani fuzzy inference system for breast cancer risk detection”, en *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, 2015, pp. 1–6.

[21] C. Alsina, S. Guadarrama, E. Renedo, y E. Trillas, “Studying Fuzzy Modus Ponens”, en *NAFIPS 2006 - 2006 Annual Meeting of the North American Fuzzy Information Processing Society*, 2006, pp. 426–429.

[22] W. V. Leekwijck y E. E. Kerre, “Defuzzification: criteria and classification”, *Fuzzy Sets Syst.*, vol. 108, núm. 2, pp. 159–178, dic. 1999.

[23] Q. Liang y J. M. Mendel, “Interval type-2 fuzzy logic systems: theory and design”, *IEEE Trans. Fuzzy Syst.*, vol. 8, núm. 5, pp. 535–550, oct. 2000.

[24] J. M. Mendel y R. I. B. John, “Type-2 fuzzy sets made simple”, *IEEE Trans. Fuzzy Syst.*, vol. 10, núm. 2, pp. 117–127, abr. 2002.

[25] J. J. Rodriguez-Andina, M. J. Moure, y M. D. Valdes, “Features, Design Tools, and Application Domains of FPGAs”, *IEEE Trans. Ind. Electron.*, vol. 54, núm. 4, pp. 1810–1823, ago. 2007.

[26] F. Cabello, J. Leon, Y. Iano, y R. Arthur, “Implementation of a fixed-point 2D Gaussian Filter for Image Processing based on FPGA”, en *Signal Processing:*

Algorithms, Architectures, Arrangements, and Applications (SPA), 2015, 2015, pp. 28–33.

[27] B. Ding, R. M. Stanley, B. S. Cazzolato, y J. J. Costi, “Real-time FPGA control of a hexapod robot for 6-DOF biomechanical testing”, en *IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society*, 2011, pp. 252–257.

[28] J. Leuchter, P. Bauer, V. Rerucha, y P. Bojda, “Dc-Dc converters with FPGA control for photovoltaic system”, en *Power Electronics and Motion Control Conference, 2008. EPE-PEMC 2008. 13th*, 2008, pp. 422–427.

[29] P. K. Dash, S. Pujari, y S. Nayak, “Implementation of edge detection using FPGA amp; model based approach”, en *2014 International Conference on Information Communication and Embedded Systems (ICICES)*, 2014, pp. 1–6.