



Universidad de Concepción
Dirección de Posgrado
Facultad de Ingeniería
Programa de Magister en Ciencias de la Computación

Hibridización De Algoritmos ACO En Detección De Bandas En Imágenes DGGE

Tesis para optar al grado de Magíster en Ciencias de la Computación

HANS ENRIQUE VILLAGRÁN VIDAL
CONCEPCIÓN-CHILE

2015

Profesor Guía: María Angélica Pinninghoff Junemann
Depto. de Ingeniería Informática y Ciencias de la Computación, Facultad de Ingeniería
Universidad de Concepción

Agradecimientos

En primer lugar, agradezco a Dios por ayudarme en este camino de 2 años, por darme fortaleza en los momentos de debilidad y frustración, y brindarme momentos de alegría y felicidad junto a las personas que más quiero.

A mi familia, por su constante apoyo durante toda mi vida. A mis padres que siempre dieron lo mejor para mi crecimiento en todas las áreas con mucho esfuerzo, el cual lo he tomado como ejemplo para mi vida.

A los profesores que me guiaron en este camino de estudio y que me dieron las herramientas del conocimiento para afrontar mis metas propuestas en este programa. En especial a la profesora María Angélica Pinninghoff, quien me ha apoyado en toda la carrera de pregrado y postgrado, alumno ayudante en las clases dictadas a cursos inferiores y en este período como tesista, por su conocimiento, el cual fue completamente necesario para finalizar este trabajo de memoria; y su apoyo y aliento para llegar al término de esta tesis.

Agradezco también las gestiones de la Dirección de Postgrado del Depto. de Informática y Cs. de la Computación por financiar mis estudios de postgrado a través de la Beca de Articulación Pregrado-Postgrado y la Beca de Arancel y Estipendio.

A todas las personas, quienes ya sea preguntándome cómo estaba, ayudándome en distintas áreas o apoyándome les agradezco profundamente.

Resumen

El objetivo de esta investigación es desarrollar un algoritmo que permita detectar bordes de bandas pertenecientes a imágenes DGGE. Estas imágenes se utilizan en la identificación de microorganismos presentes en las muestras, las cuales pasan por un proceso de separación del ADN en distintas secuencias que se visualizan en forma de bandas, las cuales pertenecen a un carril. Luego, estas estructuras son fotografiadas y se obtiene la imagen propiamente tal. Los inconvenientes de estas imágenes es que presentan ruido en exceso y generalmente las bandas están muy difuminadas, producto de su generación. Estas características hacen que la detección de las bandas a través de métodos tradicionales sea muy deficiente.

Es por esto que al analizar los trabajos de la literatura se decidió utilizar un algoritmo de hormigas como base de una combinación de heurísticas para desarrollar el sistema para detectar bandas. El algoritmo de hormigas basado en Ant Colony System utiliza dos tipos de feromonas como característica principal, la primera como feromona usual para el algoritmo ACO y la segunda como información heurística. La mejora propuesta en este trabajo consiste en inicializar la feromona usual de ACO en base al gradiente y la segunda derivada de la imagen, en contraposición a la inicialización usual, que utiliza una constante y el uso de lógica difusa como matriz de información heurística, en donde se calcula el grado de pertenencia de un pixel a ser parte del borde de la imagen.

El algoritmo propuesto fue probado tanto en imágenes genéricas, a las cuales se les aplicó blurring a través de filtros de suavizado, las cuales entregan buenos resultados de cualquiera de las dos formas, la imagen original y la suavizada. Utilizando métricas se confirma esto en este tipo de imágenes.

En las imágenes DGGE se introduce la identificación de las bandas, además de su detección, en donde el algoritmo propuesto presenta un buen rendimiento en diferentes tipos de imágenes DGGE, las cuales van desde una imagen con bandas nítidas hasta imágenes con carriles inclinados, en comparación con técnicas tradicionales.

Índice

Agradecimientos	II
Resumen	III
1. Introducción	1
1.1. Motivación	1
1.2. Hipótesis	2
1.3. Objetivos	3
1.3.1. Objetivo General	3
1.3.2. Objetivos Específicos	3
1.4. Metodología	3
1.5. Organización de la Tesis	3
2. Marco Teórico	4
2.1. Imágenes Digitales	4
2.1.1. Compresión de las imágenes	5
2.1.2. Procesamiento de imágenes digitales	6
2.2. Imágenes DGGE	8
2.2.1. Obtención de imágenes DGGE	9
2.2.2. Aplicaciones	11
2.3. Detección de Bordes	11
2.3.1. Bordes	11
2.3.2. Segmentación de Imágenes	12
2.3.3. Proceso de detección de bordes	13
2.3.4. Técnicas de Detección de bordes	14
2.4. Blurring en Imágenes	17
2.4.1. Solución del problema de blurring	18
2.5. Estado del Arte en Detección de Bordes	20
2.5.1. Procesamiento en imágenes DGGE	22
3. Técnicas Utilizadas en la Solución	23
3.1. Algoritmos Basados en Hormigas	23
3.1.1. Principales Algoritmos ACO	25
3.2. Lógica Difusa	28

3.2.1.	Conjuntos Difusos	29
3.2.2.	Funciones de Pertenencia	31
3.2.3.	Proceso de Inferencia Difusa	32
4.	Diseño de la Solución	35
4.1.	Descripción del Problema	35
4.1.1.	Detección e Identificación de Bandas	35
4.2.	Propuesta de Solución	36
4.2.1.	<i>An ant-inspired algorithm for detection of image edge features</i>	37
4.2.2.	<i>A Hybrid Approach to Edge Detection using Ant Colony Optimization and Fuzzy Logic</i>	39
4.3.	Diseño de la Solución	42
5.	Resultados en Imágenes Genéricas	45
5.1.	Imágenes y Parámetros	45
5.1.1.	Parámetros utilizados	46
5.2.	Resultados en Imágenes de Prueba	47
5.2.1.	Pruebas con <i>figuras</i>	47
5.2.2.	Pruebas con <i>Lena</i>	49
5.2.3.	Comparación con técnicas tradicionales	50
5.3.	Análisis de Resultados	52
5.3.1.	Evaluación de los Resultados	52
5.3.2.	Evaluación del Efecto del Blurring	53
5.3.3.	Variaciones del Algoritmo Propuesto	54
6.	Resultados en Imágenes DGGE	58
6.1.	Proceso de Ubicación de Bandas en el Carril	58
6.2.	Resultados en Imagen Ideal	59
6.2.1.	División de la Imagen en Carriles	60
6.2.2.	Ubicación de bandas	60
6.3.	Imágenes DGGE con diferentes características	62
6.3.1.	Color de las bandas	63
6.3.2.	Bandas Difusas	64
6.3.3.	Carriles Inclínados	66
6.4.	Comparación con Técnicas Tradicionales	67
6.5.	Análisis de Resultados	69
7.	Conclusiones y Trabajo Futuro	70
	Bibliografía	72

Índice de figuras

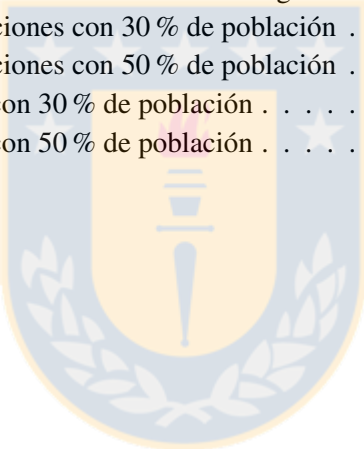
1.1. Muestra de un carril de una imagen DGGE	1
2.1. Imagen y su representación matricial	4
2.2. Tipos de Ruido	7
2.3. Tipos de Suavizado	7
2.4. DGGE en escala de grises	8
2.5. Muestra de un carril de una imagen DGGE	8
2.6. Tipos de Imágenes DGGE	9
2.7. Metodología de la Obtención de Imágenes DGGE	10
2.8. Tipos de Bordes: Escalones y Líneas	12
2.9. Tipos de Bordes: Unión en T	12
2.10. Tipos de Blurring	17
3.1. Experimentos de Doble Puente	24
3.2. Conjuntos Clásicos y Difusos	29
3.3. Operaciones en Conjuntos Difusos	30
3.4. Modelo de Inferencia Difusa	32
4.1. Detección de Bandas en una Imagen DGGE por parte del software Quantity One	35
4.2. Deficiencias en las imágenes DGGE	36
4.3. Diferencias de color en las bandas de distintas imágenes	36
4.4. Reglas Difusas para Calcular Edgeness	40
5.1. Figuras Geométricas	45
5.2. Lena	45
5.3. Cuadrado	45
5.4. Aplicación del algoritmo híbrido con los parámetros establecidos a <i>figuras</i>	47
5.5. Segmentación a <i>figuras</i>	47
5.6. Aplicación de suavizado a <i>figuras</i>	48
5.7. Resultantes de <i>figuras</i> suavizada	48
5.8. Aplicación del algoritmo híbrido con los parámetros ya establecidos a <i>Lena</i>	49
5.9. Segmentación a <i>Lena</i> con $T = 120$	49
5.10. Aplicación de suavizado a <i>Lena</i>	50
5.11. Resultantes de <i>Lena</i> suavizada	50

5.12. Métodos de detección de bordes en <i>figuras</i>	51
5.13. Métodos de detección de bordes en <i>Lena</i>	51
5.14. Imagen de Control del Cuadrado para evaluación	52
5.15. Cuadrado con Suavizado para Blurring	53
5.16. <i>Score</i> tras variación de iteraciones	55
5.17. <i>Performance</i> en diferentes algoritmos con 30 % de población	56
5.18. <i>Performance</i> en diferentes algoritmos con 50 % de población	57
6.1. Histograma de Promedio de Intensidades	58
6.2. Imagen DGGE ideal	59
6.3. Imagen Resultante Alg. Propuesto	59
6.4. Segmentación de Imagen con $T = 200$	59
6.5. División de Carriles de la imagen 6.2	60
6.6. Resultados por Grupo de Carriles	60
6.7. Identificación de Bandas en Grupo 1	61
6.8. Identificación de Bandas en Grupo 2	61
6.9. Identificación de Bandas en Carril 3	62
6.10. Identificación de Bandas en Grupo 4	62
6.11. Imagen DGGE con bandas blancas	63
6.12. Grupos de Carriles con Bandas Blancas y sus Resultantes	63
6.13. Identificación de Bandas en Carril 3 Blanco	64
6.14. Imagen DGGE con bandas difusas	64
6.15. Grupos de Carriles con Bandas Difusas y sus Resultantes	65
6.16. Identificación de Bandas en Carril 1 Difuso	65
6.17. Imagen DGGE con carriles inclinados	66
6.18. Identificación de Bandas en Carril Inclinado	66
6.19. Métodos de Detección de Bordos en Carriles Claros	67
6.20. Métodos de Detección de Bordos en Carriles con Bandas Blancas	67
6.21. Métodos de Detección de Bordos en Carriles con Bandas Borrosas	68
6.22. Métodos de Detección de Bordos en Carriles con Bandas Inclinadas	68

Índice de Tablas

- 3.1. Funciones de Pertenencia más utilizadas 31

- 5.1. Parámetros escogidos para el algoritmo propuesto 46
- 5.2. Evaluación de la resultante del *cuadrado* 53
- 5.3. Evaluación de la resultante del *cuadrado* con blurring 54
- 5.4. Score tras variación de iteraciones con 30 % de población 54
- 5.5. Score tras variación de iteraciones con 50 % de población 55
- 5.6. Performance de algoritmos con 30 % de población 56
- 5.7. Performance de algoritmos con 50 % de población 57



Capítulo 1

Introducción

1.1. Motivación

El análisis de imágenes es usado en diferentes áreas y para distintas aplicaciones, por ejemplo, en el área médica el análisis de radiografías, ecografías, etc. es ampliamente utilizado y determinante a la hora de diagnosticar alguna enfermedad o anomalía, también en el área militar, forense, forestal o científica se pueden encontrar un sin fin de aplicaciones en donde se utilizan imágenes para determinar patrones, anomalías, alguna característica específica de clasificación, etc.

Las imágenes DGGE (Denaturing Gradient Gel Electrophoresis) [19] son un tipo de imagen de ADN que se obtiene a través de una técnica que genera un perfil genético o *huella digital* el cual puede ser utilizado para identificar los miembros dominantes de una comunidad microbiana. En las imágenes DGGE los carriles representan a las muestras de ADN. Los carriles son empleados para indicar el peso molecular, medidos en bp (pares base), de las hebras de ADN. Cada carril está compuesto por bandas (como se observa en la Figura 1.1), que son las líneas verticales de los carriles. Las bandas representan la aglomeración de segmentos de la muestra de ADN que tienen el mismo tamaño (en bp).



Figura 1.1: Muestra de un carril de una imagen DGGE

El problema con las imágenes DGGE, es que en su proceso de generación son afectadas por factores físico-químicos, los que provocan alteraciones en las imágenes. Por eso es que se deben aplicar correcciones para analizar las imágenes de mejor manera e identificar sus elementos aplicando técnicas de segmentación de imágenes.

Un procesamiento común es la detección de bordes de una imagen, el cual consiste en detectar los píxeles donde ocurra un cambio brusco de la intensidad de la imagen, que corresponde a las fronteras de los objetos visualizados. Las técnicas que permiten detectar bordes se basan principalmente en aproximaciones discretas de la primera y segunda derivada de las imágenes en tonos de grises, pues el gradiente representa la variación local respecto a una variable. Esta es la base de las técnicas de análisis numérico sobre la representación matricial de las imágenes, pero también existen técnicas no tradicionales, que utilizan diferentes heurísticas.

Esta tarea de detectar los bordes y extraer características de las imágenes tiene una gran importancia. La

extracción de características puede variar desde detectar los bordes de las imágenes hasta tareas más complicadas como hacer matching entre imágenes, o usar características especializadas para la aplicación en que son construídas (por ejemplo, aproximar el número de personas que hay en una aglomeración detectando los rostros humanos).

En la obtención de imágenes digitales, frecuentemente ocurre que la imagen resulta difusa por el movimiento del objeto a fotografiar, por un inadecuado enfoque o el uso de una abertura que da poca profundidad de campo; o como en el caso de las imágenes DGGE, por la complejidad de los procesos para obtenerlas, esto se conoce como **blurring**. El principal problema de una imagen con blurring es que disminuyen el contraste de la imagen promediando las intensidades de los píxeles que están más cercanos a los bordes de líneas o áreas donde hay una significativa transición de intensidad, aunque esto se utiliza para reducir el efecto del ruido en las imágenes. Para solucionar el problema de blurring se han utilizado enfoques basados en la naturaleza matricial de la imagen, en la cual se soluciona la ecuación lineal que representa a la imagen aplicando los métodos iterativos de solución de ecuaciones lineales. Otros métodos consisten en aplicar técnicas de segmentación basados en el gradiente de la imagen.

El problema de detección de bandas en las imágenes DGGE es tratado como un problema de detección de bordes. La detección de las bandas a través de técnicas tradicionales genera resultados poco precisos. Si se considera que es difícil distinguir las bandas del fondo, los resultados empeoran más.

Varias técnicas se utilizan para solucionar el problema de detección de bordes. Las técnicas tradicionales consisten en utilizar las características matriciales de la imagen, las que se expresan en forma de filtros u operadores basados en las derivadas de la imagen. Pero al aplicar estas técnicas no se obtienen buenos resultados cuando la imagen presenta ruido o es difusa, siendo éste el común de los casos en las imágenes DGGE. Es por esto que se han usado variados algoritmos basados en distintas heurísticas para solucionar este problema.

Sin embargo, en la detección de bandas DGGE se ha probado solo con algoritmos basados en hormigas (ACO), los cuales corresponden a técnicas probabilísticas que se inspiran en el comportamiento de las hormigas en la naturaleza, las cuales buscan el camino más corto para encontrar su alimento y volver a su hormiguero. Esto se debe a la utilización de una sustancia llamada *feromona*, la que se deposita en el camino para que las demás hormigas sigan ese camino. Tras probar en varios caminos, la feromona depositada en los caminos más cortos será más fuerte que en los caminos más largos debido a la evaporación, lo que permite que se olviden los caminos más largos y las hormigas se concentren en el camino más corto.

El objetivo de este trabajo es proveer una técnica que detecte de mejor manera los bordes de las bandas en imágenes DGGE, ya que por el blurring y el ruido presente en ellas, aun hay fallos en la detección. Es por esto que se propone realizar una hibridización entre dos algoritmos, lo que consiste en combinar técnicas diferentes para solucionar un problema y así obtener mejores resultados aprovechando y combinando las ventajas de cada una de las estrategias individuales, basándose en los trabajos realizados en el área de detección de bordes en imágenes en general y aplicarla a las imágenes DGGE.

1.2. Hipótesis

La detección de los bordes de las imágenes DGGE mejora al aplicar un algoritmo de colonias de hormigas combinado con lógica difusa; al compararlo con técnicas tradicionales.

1.3. Objetivos

1.3.1. Objetivo General

Desarrollar un método de detección de bordes de las bandas de las imágenes DGGE combinando un algoritmo basado en hormigas con lógica difusa, que sea tolerante al blurring de la imagen.

1.3.2. Objetivos Específicos

- Investigar sobre técnicas de detección de bordes, específicamente aplicados a imágenes borrosas (problema de blurring).
- Desarrollar e implementar una técnica heurística hibridizada.
- Probar el algoritmo híbrido en imágenes con simulación de blurring, para evaluar su rendimiento.
- Realizar un conjunto de pruebas en imágenes DGGE con distintos tipos de problemas.
- Realizar un análisis comparativo de los resultados obtenidos en las diferentes pruebas.

1.4. Metodología

En primer lugar se realizará una investigación sobre el problema de detección de bordes en imágenes y se pondrá atención a técnicas que permitan disminuir el blurring en las imágenes. También se estudiarán las características de las imágenes DGGE y las técnicas utilizadas en la detección de bandas en este tipo de imágenes.

Más adelante, de acuerdo a lo revisado en el estado del arte, se escogerán las técnicas a utilizar para realizar la hibridización de los algoritmos, la cual se probará en imágenes genéricas con filtros para simular el blurring e imágenes DGGE con distintas características.

El algoritmo híbrido creado se evaluará teniendo como comparación técnicas tradicionales y las técnicas en que se basó el algoritmo diseñado, proveyendo una métrica para esta labor.

1.5. Organización de la Tesis

La tesis está organizada de la siguiente manera.

En el capítulo 2 se presentan las bases teóricas sobre los conceptos utilizados en esta tesis, los cuales son procesamiento de imágenes, las características de las imágenes DGGE, técnicas de detección de bordes, presentar el problema de blurring e indicar las técnicas actuales que resuelven estos problemas. El capítulo 3 se muestran las técnicas utilizadas como base para diseñar el algoritmo híbrido.

En el capítulo 4 se presentan los problemas que presenta el trabajo con imágenes DGGE, los papers de los trabajos en que se basa el diseño del algoritmo y el modelo de solución propuesto para resolver el problema de detección de bordes de bandas.

Los capítulos 5 y 6 muestran las pruebas y resultados obtenidos con el algoritmo en distintas imágenes; el capítulo 5 contiene las pruebas realizadas en imágenes genéricas, con simulación de blurring y el capítulo 6 presenta los resultados del algoritmo en las imágenes DGGE.

El capítulo 7 muestra las conclusiones y los trabajos futuros que dejó la realización de esta tesis.

Capítulo 2

Marco Teórico

2.1. Imágenes Digitales

Una imagen digital es una representación discreta de una imagen de dos dimensiones, a través de una función $f(x, y)$, donde x e y representan coordenadas en un plano. Esta función indica la intensidad o nivel de gris de la imagen en ese punto. Al ser x e y discretos se tiene una grilla o matriz, la cual es la representación usual de la imagen.

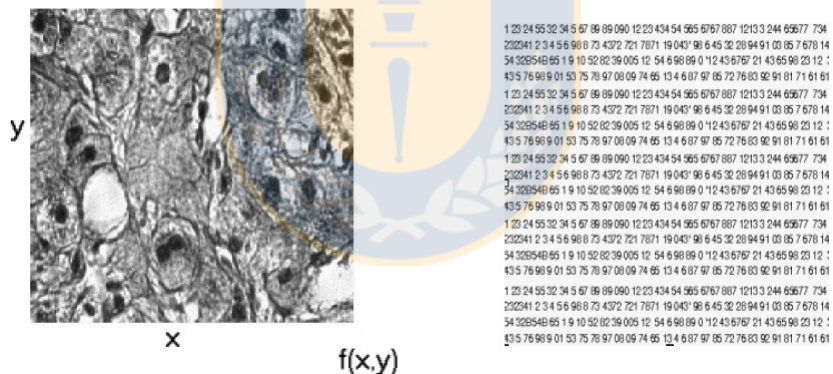


Figura 2.1: Imagen y su representación matricial

Cada elemento de la matriz (coordenada (x, y) aplicada la función f) se conoce como *pixel* (abreviación de picture element —elemento de la imagen) y almacena el valor de la intensidad de la imagen en ese punto. La información que se obtiene de la matriz y los pixeles son la *resolución de la imagen*, la cual es la cantidad de pixeles que define la imagen (el tamaño de la matriz) y la *profundidad de tono* que indica el valor de los pixeles (de 0 a $2^n - 1$, siendo n la cantidad de bits necesarios para representar un pixel). Estas imágenes se conocen como mapa de bits.

Otro tipo de imágenes son las imágenes vectoriales o gráficos orientados a objetos, las cuales generan la imagen a partir de trazos geométricos (segmentos, polígonos, arcos, etc.) determinados por cálculos y fórmulas matemáticas, tomando algunos puntos de la imagen para construir el resto. Al ser compuestas por diferentes elementos, es posible modificar una imagen (ampliarla, rotarla, entre otras operaciones) de manera relativamente sencilla. Las imágenes vectoriales se almacenan en una lista que describe cada uno de sus vectores componentes, su posición y sus atributos. Las desventajas de este tipo de imágenes se presentan en el ámbito fotográfico, pues

presentan dificultades al tratar efectos en las imágenes, sobre todo en imágenes complejas, pues requieren una gran cantidad de cálculos en el procesador.

Volviendo a las imágenes en mapa de bits, la profundidad de tono se define de acuerdo a la cantidad de bits para representar los píxeles.

- Las imágenes binarias o monocromáticas son aquellas en que cada píxel es representado por un bit, el píxel será blanco si su valor es 1 y negro si el valor es 0.
- Al usar ocho bits (un byte) para representar los píxeles se obtienen las imágenes en escala de grises, cuyos valores oscilan desde el negro (valor de píxel 0) hasta el blanco (valor de píxel 255), es decir, se tienen 256 matices de grises.
- Las imágenes a color se representan con más canales para obtener los colores, a través de un modelo aditivo (donde 0 es el negro y el valor máximo $2^n - 1$ es el blanco), los cuales se combinan para producir un color dentro de una gama determinada. El más utilizado es el RGB, que se compone de 3 canales (rojo, verde y azul) para representar los colores, cada uno de ellos utilizando 16 bits, con una cantidad determinada de bits para cada uno de los colores. Con una profundidad de 24 bits se utiliza 1 byte por color y en el caso de una profundidad con 32 bits, se utiliza un cuarto byte para tener un valor del grado de transparencia de la imagen ¹. Lo usual es utilizar una profundidad de 24 bits y en este caso se obtienen 16,7 millones de colores. Otra forma de representar los colores es con 4 canales en el modo CMYK (Cian, Magenta, Amarillo, Key o Negro) con usualmente 32 bits (1 byte para cada canal). Este es un modelo sustractivo, ya que a diferencia del modelo aditivo (RGB) el blanco es la ausencia de luz y se representa con el valor 0 y el negro es la combinación de los colores. Sin embargo este modelo es dependiente del dispositivo de control de color ², a diferencia del RGB que es independiente del dispositivo.
- Existe otro modo utilizado cuando se tiene que tener en consideración el tamaño del archivo o cuando se trabaja con imágenes con pocos colores. Se utiliza un número fijo de colores, 256 o menos, en cada punto para representar el color en ese punto. Por defecto intenta calcular una paleta óptima para representar mejor su imagen. Es el modo indexado y se utiliza en imágenes GIF.

2.1.1. Compresión de las imágenes

Para almacenar las imágenes de manera de preservar sus características en un tamaño que no sea muy grande se utiliza la compresión de las imágenes. Una forma de realizar la compresión de todo tipo de datos es eliminando la redundancia de datos, la cual se puede realizar de dos maneras:

- **Compresión sin pérdida** (lossless compression): son aquellas técnicas en las que la imagen puede recuperarse de su versión comprimida sin ningún tipo de alteración de los datos. Es una compresión reversible
- **Compresión con pérdidas** (lossy compression): en estas técnicas de compresión se realizan modificaciones irreversibles a la imagen, teniendo en consideración las características de la visión humana al momento de realizar las alteraciones de la imagen, por lo cual las pérdidas sufridas no son detectadas por el ojo humano (esto se conoce como compresión visualmente sin pérdidas).

¹El grado de transparencia es una máscara que de forma virtual permite componer imágenes opacas con otras que tienen un cierto nivel de transparencia.

²Los dispositivos de control de color se utilizan como periféricos para mostrar el color. Algunos de estos dispositivos son los monitores, impresoras, escáneres, pantallas y medios similares

Los formatos de almacenamiento de imágenes están dados por el tipo de compresión que utilicen.

- Formatos que no utilizan compresión: TIFF y BMP. Los archivos con estos formatos tienen un gran tamaño
- Formatos que utilizan compresión sin pérdida: PNG y GIF
- Formatos que utilizan compresión con pérdidas: JPEG, el formato más utilizado en almacenamiento de imágenes.

2.1.2. Procesamiento de imágenes digitales

El procesamiento de imágenes digitales se preocupa de la transformación de una imagen a un formato digital y de su procesamiento a través de computadores. El procesamiento de imágenes es uno de los componentes del tratamiento digital de imágenes, que está conformado también por el análisis. Mientras el procesamiento de imágenes digitales está referido a la realización de transformaciones para mejorar las imágenes, el análisis consiste en la extracción de características y propiedades de las imágenes, así como la clasificación e identificación de ellas y el reconocimiento de patrones [22].

Las áreas en que se aplica el procesamiento de imágenes digitales son muy variadas, desde la medicina hasta la astronomía, incluyendo biología, geología, aplicaciones industriales y militares, entre otras, pues casi no hay área del desarrollo técnico y científico que no esté impactada de alguna forma por el procesamiento de imágenes digitales.

2.1.2.1. Algunos procesamientos de imágenes usuales

Las principales transformaciones a las que son sujetas las imágenes son reflejar, trasladar, rotar, cambiar color, modificar el contraste, aplicar filtros a los píxeles, utilizando técnicas de procesamiento de imágenes combinadas con otras operaciones matemáticas. Técnicas más complejas son la segmentación, detección de bordes, patrones y otras características [22].

Uno de los problemas que tienen las imágenes digitales es el **ruido**, el cual es una variación aleatoria del brillo o la intensidad de las imágenes, producidos por el dispositivo que genera las imágenes. Matemáticamente se define de la siguiente manera:

$$g(x, y) = f(x, y) + n(x, y) \quad (2.1)$$

donde: $f(x, y)$ es la intensidad de la imagen, $n(x, y)$ es el ruido agregado y $g(x, y)$ es la intensidad total de la imagen.

Estas variaciones son indeseables al momento de procesar la imagen para distintos propósitos (ya que es información no deseada) y se muestran como *grano* en la imagen. Algunos tipos de ruido son el ruido *sal y pimienta* y el *ruido gaussiano*.

- **Ruido Sal y Pimienta:** Se define como ocurrencias aleatorias de píxeles entre valores muy altos (color blanco, sal) y valores muy bajos (color negro, pimienta). Este tipo de ruido afecta a una pequeña cantidad de píxeles de la imagen.
- **Ruido Gaussiano:** En este tipo de ruido, a cada píxel se le adiciona un pequeño valor, de acuerdo a la distribución normal de la imagen.

Estos tipos de ruido se pueden ver en la figura 2.2:



Figura 2.2: Tipos de Ruido

Una de las formas de eliminar el ruido de las imágenes es la utilización de **técnicas de suavizado**, las cuales consisten en reducir las variaciones de intensidad entre pixeles vecinos, utilizando filtros o máscaras. La desventaja de la aplicación de filtros para suavizar la imagen es la difuminación que sufren los bordes. Los filtros utilizados pueden ser lineales o no lineales.

- **Filtros Lineales:** En este tipo de filtros se realiza una operación de convolución³ entre la imagen a ser filtrada y la máscara o filtro que será aplicado. Algunos de estos filtros son el *filtro por promedio* y el **filtro gaussiano**. Éste último consiste en aplicar un filtro a los pixeles de la imagen, en donde los valores vecinos a cada pixel se promedian de acuerdo a un peso. Este tipo de filtros se utiliza principalmente para reducir el ruido gaussiano.
- **Filtros No Lineales:** Este tipo de filtros no se puede representar de forma matricial (una máscara) y no hay una estructura general para los distintos filtros no lineales. Un tipo de filtro no lineal es el *filtro de rango*, en el cual puede ocuparse el valor mínimo, máximo o de la mediana. El **filtro de mediana** actúa en los pixeles de la imagen calculando la mediana de los pixeles vecinos respecto al que se aplicó el filtro. Este tipo de filtros se utiliza para reducir el ruido aleatorio.

La aplicación de estos filtros a una imagen se muestra en la figura 2.3



Figura 2.3: Tipos de Suavizado

³Operación que transforma dos funciones f y g que genera una nueva función formada por la superposición de f y la traslación o inversión de g

Finalmente, otro tipo de procesamiento de imágenes es el **realce de contraste**, lo cual consiste en aumentar el contraste⁴ de la imagen para mejorar algunas de sus características visuales para las etapas posteriores del análisis automático de imágenes. Las técnicas de realzado se basan en operaciones de punto a punto, es decir, sin considerar la vecindad de los píxeles. Las técnicas de realce de contraste se basan en la ecualización del histograma⁵ de la imagen y en la transformación de las intensidades de la imagen.

2.2. Imágenes DGGE

DGGE (Denaturing Gradient Gel Electrophoresis) es una técnica para detectar diferencias entre fragmentos de ADN del mismo tamaño pero con diferentes secuencias de pares base [17]. Esto involucra la separación de las muestras en fragmentos utilizando métodos fisicoquímicos, principalmente electroforesis de las muestras utilizando una unidad de electroforesis con gel, radiación UV y una luz de excitación emitida al tomar una foto con una cámara CCD⁶, y se obtiene una imagen DGGE [48]. Estas imágenes se componen de carriles y bandas (Figura 2.4). Los carriles son las columnas verticales que aparecen en la imagen DGGE y representan las muestras de ADN, salvo los carriles de los extremos que son los carriles de referencia. Este tipo de carriles se utiliza para indicar el peso molecular del ADN, medido en pares base (bp)⁷. Cada carril está compuesto por bandas, las cuales representan la aglomeración de segmentos de una muestra de ADN con el mismo valor del pares base (bp)

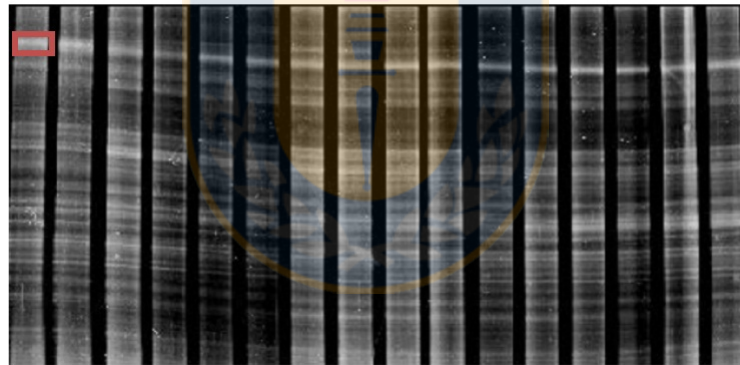


Figura 2.4: DGGE en escala de grises



Figura 2.5: Muestra de un carril de una imagen DGGE

⁴El contraste es la dispersión de los niveles de gris de la imagen

⁵Función que mapea la cantidad de ocurrencias de un nivel de gris en la imagen

⁶Las cámaras CCD (*Charge-coupled device*) son las cámaras digitales con celdas que registran la imagen a través del modelo RGB

⁷Un par base consiste en un par de nucleótidos (componentes del ADN) opuestos y complementarios conectados por un puente de hidrógeno

El procesamiento de separación de ADN puede provocar alteraciones a este tipo de imágenes. Pueden ser de distinta naturaleza, como el ruido, deformaciones y difusión, entre otras [41]. Algunos casos específicos son difusión de las bandas, deformación de los carriles y rotación la imagen, de carriles y bandas o ruido presente en la imagen.

De acuerdo a la disposición del gradiente de desnaturalización y el campo eléctrico se tienen dos tipos de imágenes DGGE:

- **DGGE Perpendicular:** el gradiente es perpendicular al campo eléctrico y se usa un amplio rango del desnaturalizante, típicamente de 0-100 % o de 20-70 %
- **DGGE Paralelo:** el gradiente es paralelo al campo eléctrico y el rango del desnaturalizante es difuso para permitir la mejor separación de los segmentos [32]. Este tipo de imágenes son utilizadas en este trabajo

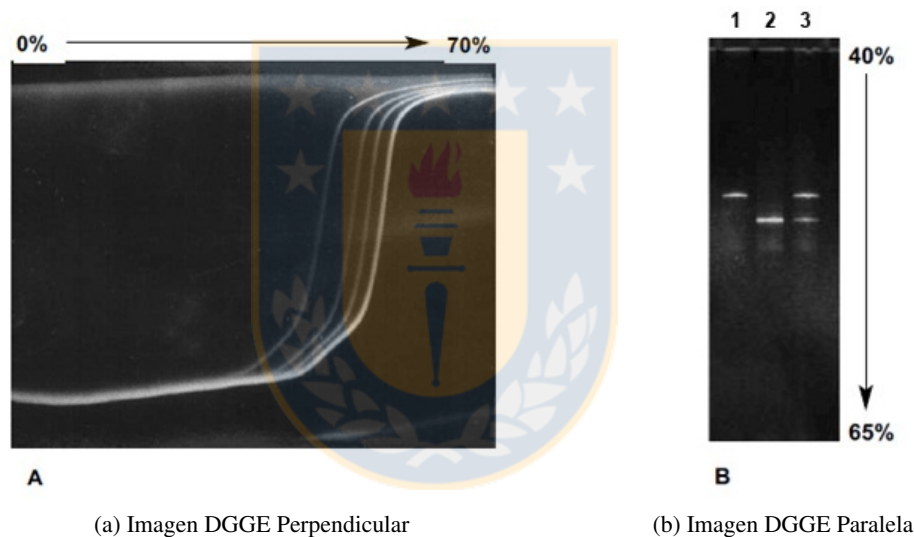


Figura 2.6: Tipos de Imágenes DGGE

En esta investigación se utilizarán imágenes DGGE paralelas.

2.2.1. Obtención de imágenes DGGE

Para detectar diferencias en las secuencias de ADN se necesita separar la molécula en segmentos específicos, para lo cual existen varios métodos y tecnologías, en un proceso conocido como desnaturalización.

En términos generales se utiliza un gel desnaturalizante, que puede ser gel de poliacríamida, el cual al aplicarle electroforesis permite ver las muestras de ADN bajo la luz ultravioleta. La electroforesis es un proceso de separación de moléculas, utilizando campos eléctricos, de acuerdo a la carga tamaño y forma. Otro gel utilizado es la agarosa, la cual carga positivamente a la muestra y permite el movimiento de la muestra desde el gel hacia el ánodo. Aplicada a moléculas orgánicas (ADN) la electroforesis permite mostrar la variabilidad genética de las muestras [34].

Estos procesos aplicados a DGGE constan de los siguientes pasos (los que se pueden ver en la figura 2.7):

- **Extracción de ADN:** el ADN es extraído desde una mezcla de ADN de todos los organismos presentes en la muestra. La muestra contiene ADN de microorganismos y cada molécula de ADN posee miles de genes que codifican variadas funciones celulares.
- **Amplificación:** acompañada de la reacción en cadena de las polimerasas (PCR), se selecciona una región específica de un gen y se generan muchas copias de ese segmento. La secuencia de ADN de esta región es usada para clasificar al organismo dentro de una familia o especie. El resultado de la amplificación es una muestra que contiene el mismo largo (o tamaño), pero puede contener distintas secuencias de ADN.
- **Separación:** las mezclas obtenidas en la amplificación se separan utilizando un calor constante (cerca de 60°C) y una concentración incremental de químicos desnaturalizantes que fuerzan a las moléculas de ADN a separar sus hélices. Además se utiliza un gel que permite que la muestra se cargue negativamente y sea atraída por un electrodo positivo (proceso de electroforesis). El ADN resultante es forzado a migrar a los polos del gel. Al alcanzar la concentración de desnaturalizante hasta un umbral se tiene una muestra de ADN mezclada [33]. El movimiento de los fragmentos de ADN genera estiramientos de los pares base con una temperatura de fusión idéntica, conocidos como dominios de fusión. Cualquier variación de secuencias de ADN dentro de estos dominios da lugar a diferentes temperaturas de fusión, lo que causará que las secuencias migren a distintas posiciones del gel.

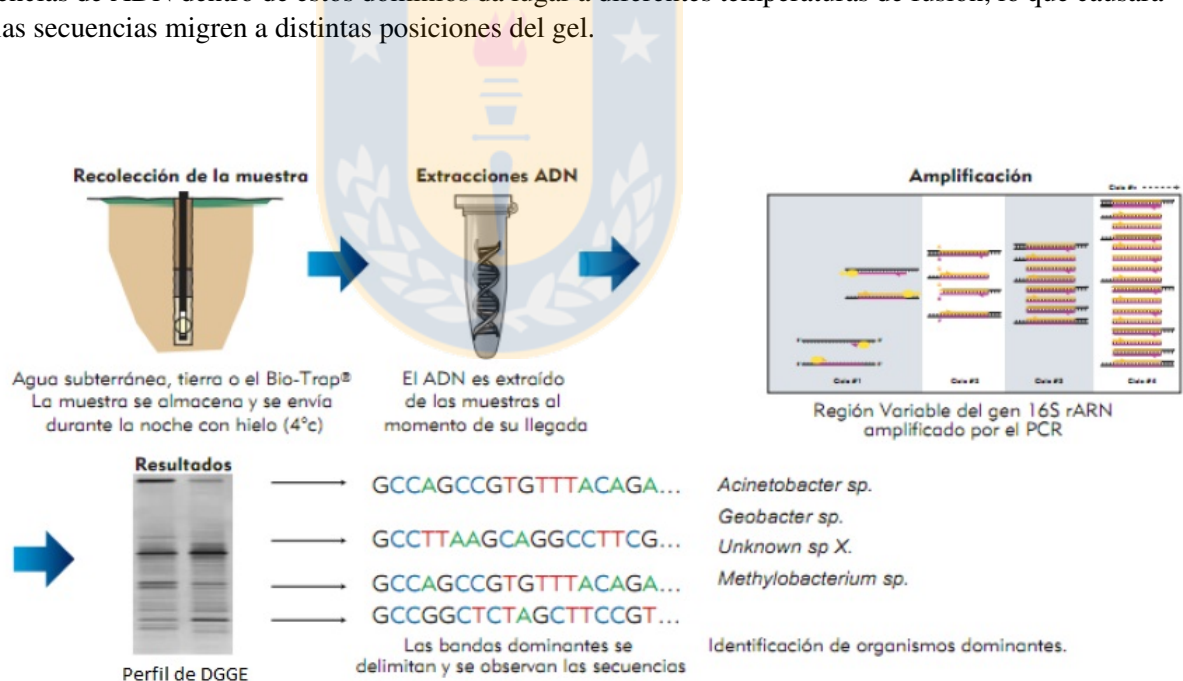


Figura 2.7: Metodología de la Obtención de Imágenes DGGE

2.2.2. Aplicaciones

La principal aplicación del análisis de las imágenes DGGE es la identificación precisa de genes funcionales presentes en poblaciones particulares de bacterias o especies específicas de bacterias y a partir de ello la determinación de las que son dominantes. Esta tarea se utiliza en la microbiología ambiental y clínica y en el tratamiento seguro de alimentos [30].

Otras aplicaciones de DGGE incluyen:

- Evaluación de Biorremediación ⁸
- Tratamiento de agua potable y aguas servidas [56]
- Formación de Biofilm (película bacteriana) [31]
- Corrosión provocada por microbios
- Identificaciones de contaminantes microbianos en productos comerciales e industriales.

Para la evaluación de Biorremediación, los perfiles DGGE y el análisis de secuencia son comúnmente usados para evaluar las similitudes/diferencias de la composición de una comunidad bacteriana (o de hongos). DGGE marca las diferencias entre las muestras y cambios en la composición de la población microbiana en tiempo adicional o en tratamientos de seguimiento. De otra forma, DGGE puede ser utilizado para determinar los grupos bacteriales que son estimulados dada una acción correctiva, como puede ser la adición de un sustrato de crecimiento o un nutriente.

2.3. Detección de Bordes

2.3.1. Bordes

Los bordes son límites entre diferentes texturas y proveen de información visual importante que corresponden a discontinuidades físicas, fotométricas o de las propiedades geométricas de los objetos. En la interpretación física de los bordes se incluyen variaciones significativas en la reflectancia, iluminación, orientación o profundidad de las superficies. En las imágenes digitales pueden ser definidos como discontinuidades en la intensidad desde un pixel a otro. Los bordes de una imagen marcan características importantes que ofrecen una indicación de una mayor frecuencia que otras [24].

Las variaciones de intensidad de una imagen se manifiestan en forma de pasos, líneas y uniones [63]:

- El tipo de borde más común es el de escalón o paso, el cual es el punto en el cual ocurre una discontinuidad del nivel de gris de la imagen a un pixel de distancia (es decir, un cambio abrupto). Esto ocurre al generar la sombra de un objeto, o cuando un objeto está escondido detrás de otro. Esto se aplica en imágenes sin ruido, es decir, en condiciones ideales. Al existir ruido o tener una imagen borrosa, se consideran rampas, que consisten en detectar la transición a una distancia de n píxeles, donde la inclinación de la rampa es inversamente proporcional al grado de difusión del borde (Figura 2.8).

⁸La *biorremediación* es un proceso que utiliza microorganismos para remover los contaminantes presentes de un sitio contaminado y que éste vuelva a su condición natural.

- Las líneas resultan de la iluminación entre objetos que están en contacto o de objetos localizados contra un fondo. Corresponden a los extremos de la imagen, donde hay cambios abruptos de intensidad y se vuelve al estado inicial. La aproximación real de las líneas corresponde a los bordes de tejado, que corresponden a la versión suavizada de una discontinuidad de línea con ruido. Se utilizan para detectar caminos o ríos en las imágenes satelitales o mapas (Figura 2.8).

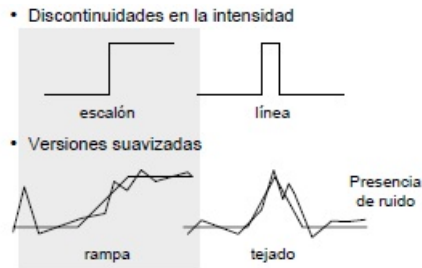


Figura 2.8: Tipos de Bordes: Escalones y Líneas

- Las uniones representan las esquinas de dos bordes de distintos objetos que están en contacto bajo distintas circunstancias de iluminación, por lo cual se tienen uniones en T (Figura 2.9), L, X o Y.

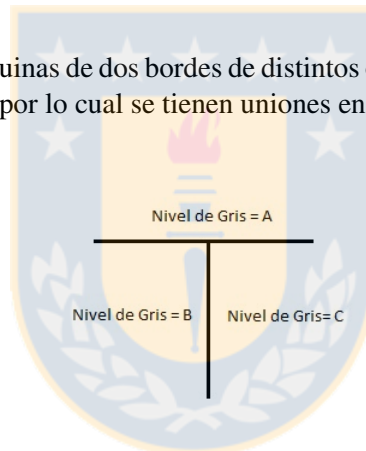


Figura 2.9: Tipos de Bordes: Unión en T

Se utiliza el término borde para todos estos tipos, pero las técnicas de detección de bordes consideran los bordes de escalón, al ser los más comunes.

Al obtener los bordes de una imagen se pueden extraer distintas características desde ellos (tales como líneas, curvas o esquinas) y ser utilizadas por algoritmos de visión computacional de alto nivel, tales como los utilizados en reconocimiento de patrones.

2.3.2. Segmentación de Imágenes

La detección de bordes es una de las técnicas que pertenece al proceso de segmentación de imágenes, el cual consiste en particionar una imagen en múltiples regiones o grupos de píxeles dado algún criterio, sea la textura, color o intensidad de la imagen. El resultado de la segmentación es un conjunto de regiones que colectivamente cubren la imagen completamente, o un conjunto de contornos extraídos desde la imagen. Todos los píxeles presentes en una región poseen características o propiedades similares. Las regiones adyacentes difieren significativamente respecto a las mismas características. Por ejemplo, considerando una segmentación de una imagen dada su intensidad, de acuerdo a algún criterio se pueden obtener n grupos, al aplicar la detección de bordes se obtienen dos grupos, los píxeles que pertenecen al borde y los que no.

Una técnica usual para segmentar de acuerdo a la intensidad es a través del uso de un valor de *umbral* T . Estos métodos binarizan la imagen, aplicando a cada pixel (con intensidad $f(x, y)$) de la imagen una comparación entre su valor y el valor umbral de la siguiente forma:

$$U(x, y) = \begin{cases} 0, & \text{si } f(x, y) < T \\ 1 & \text{si } f(x, y) \geq T \end{cases} \quad (2.2)$$

donde $U(x, y)$ es la intensidad final de la imagen binarizada. La elección del valor de umbral puede realizarse de manera manual, asignando un valor definido por el usuario. Otra forma de elegir el valor del umbral es escogerlo de acuerdo a la media de los dos valores máximos del histograma. Sin embargo, la técnica estándar para escoger el umbral es el **método de Otsu**, el cual utiliza la medida de dispersión de los niveles de gris. El método de Otsu calcula el valor umbral de forma que la dispersión dentro de cada segmento sea lo más pequeña posible, pero al mismo tiempo la dispersión sea lo más alta posible entre segmentos diferentes. Para ello se calcula el cociente entre ambas variantes y se busca un valor umbral para el que este cociente sea máximo [22].

Otra forma de segmentar una imagen es binarizar la imagen de acuerdo a un valor ya elegido de umbral y aplicar un preprocesamiento a la imagen, con los filtros ya explicados en la sección 2.1.2.1, por ejemplo utilizando el filtro de mediana.

2.3.3. Proceso de detección de bordes

La detección de bordes es un proceso que trata de capturar las propiedades significativas de una imagen. Estas propiedades son las relacionadas con las discontinuidades de las características físicas de los objetos, mostradas en las imágenes digitales, los bordes (en una imagen a escala de grises son las variaciones de los niveles de gris o la intensidad de la imagen). El propósito de la detección de bordes es localizar dónde ocurren estas variaciones. El resultado de este proceso es una imagen que proporciona información estructural útil de los bordes del objeto de la imagen, simplificando el análisis de ellas reduciendo drásticamente la cantidad de datos procesados. Este proceso debe ser eficiente y confiable, ya que puede ser utilizado en etapas posteriores donde se debe validar el producto obtenido para ser usado.

El proceso de detección de bordes contiene 4 pasos fundamentales [44] [22]:

- **Filtrado:** dado que las imágenes obtenidas no son ideales, puesto que contienen alteraciones como el ruido, es necesario mejorar la fuente, quitando el ruido, sin embargo, se tiene un trade-off entre reducir el ruido y la robustez del detector de bordes, pues se pueden borrar los verdaderos bordes.
- **Realce:** para facilitar la detección de bordes, se determinan cambios para mejorar la imagen como cambiar la intensidad de los vecinos de un punto o mejorar los puntos borrosos (conocido como sharpening). Para esto se realizan cambios locales en la intensidad para realzar los pixeles.
- **Detección:** se extraen los puntos que son potenciales candidatos para ser parte del borde de acuerdo al método utilizado sin post-procesamientos o tras la aplicación de un umbral.
- **Localización:** se localizan los puntos seleccionados desde los candidatos, los que forman el borde de la imagen

Existen varias maneras de realizar la detección de bordes. Sin embargo, la mayoría de los métodos pueden ser agrupados dentro de dos categorías [45]:

- **Método del Gradiente:** se detectan los bordes buscando el máximo y el mínimo usando la primera derivada de la imagen. También se puede discretizar el cálculo de la primera derivada. El gradiente (primera derivada) proporciona la dirección y magnitud del máximo cambio de intensidad en cada ubicación de píxeles (Ecuación 2.3).

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad (2.3)$$

- **Detección basada en el Laplaciano:** se buscan los puntos de inflexión de la segunda derivada de la imagen para encontrar los bordes (Ecuación 2.4).

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad (2.4)$$

2.3.4. Técnicas de Detección de bordes

Se han definido varios operadores que trabajan con la primera o segunda derivada para detectar los bordes de una imagen. Con la información que proporciona el gradiente de la imagen (primera derivada) se tienen los operadores de Robert, Prewitt y Sobel [39].

1. **Operador Sobel:** es una aproximación simple del concepto de gradiente con suavizado. Este operador aplica un filtro (o kernel) sobre una submatriz de 3x3 en las direcciones del eje x y del eje y ; y ha sido diseñado para contrarrestar al máximo la sensibilidad de los bordes en dirección vertical y horizontal respecto a la grilla de píxeles.

$$H_x^S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad (2.5)$$

$$H_y^S = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ -1 & 2 & 1 \end{bmatrix} \quad (2.6)$$

Donde H_x^S es el filtro horizontal y H_y^S es el filtro vertical. Al aplicar los filtros a las imágenes, se obtienen dos imágenes gradiente D_x y D_y , donde $D_x = H_x^S * I$, $D_y = H_y^S * I$, con I la imagen y $*$ es el operador de convolución.

2. **Operador Prewitt:** el filtro Prewitt es muy similar al filtro Sobel. Se utilizan filtros sobre matrices 3x3 para detectar el gradiente en las direcciones x e y . Se diferencia del operador Sobel en la respuesta espectral (spectral response).

$$H_x^P = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad (2.7)$$

$$H_y^P = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad (2.8)$$

Siendo H_x^P el filtro horizontal y H_y^P el filtro vertical. Al aplicar los filtros a las imágenes originales, se obtienen las imágenes D_x y D_y , donde $D_x = H_x^P * I$, $D_y = H_y^P * I$.

3. **Operador Roberts:** este operador es uno de los más simples detectores de bordes, ya que aplica una máscara de 2x2 en las diagonales de la matriz de la imagen.

$$H_1^R = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad (2.9)$$

$$H_2^R = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.10)$$

Al aplicar los filtros a las imágenes originales, se obtienen las imágenes D_x y D_y , donde $D_x = I * H_1^R$, $D_y = I * H_2^R$.

A diferencia de los operadores de Prewitt y Sobel, en Roberts Cross calcula el gradiente con las componentes rotadas en 45° .

En general, por cada dirección (x e y) se obtienen las imágenes de gradiente D_x y D_y separadamente en la imagen, lo que produce mediciones separadas de los componentes del gradiente en cada orientación y pueden ser combinados para encontrar la magnitud y orientación absoluta local del gradiente:

La magnitud del gradiente se calcula de la siguiente forma:

$$|\nabla(x, y)| = \sqrt{(D_x(x, y))^2 + (D_y(x, y))^2} \quad (2.11)$$

Y para obtener la orientación del gradiente se utiliza la siguiente ecuación

$$\phi(x, y) = \tan^{-1} \left(\frac{D_y(x, y)}{D_x(x, y)} \right) = \arctan(D_y(x, y), D_x(x, y)) \quad (2.12)$$

Al utilizar la segunda derivada se utiliza el operador laplaciano con un filtro gaussiano, llamado *LoG*. Este operador considera la aplicación de un filtro de suavizado realizada por convolución de una función gaussiana $g(x, y)$, para reducir su sensibilidad al ruido y luego se aplica la derivación a través del Laplaciano (2da derivada), el cual muestra las regiones con un cambio abrupto de la intensidad, lo que se utiliza para detectar los bordes (Ecuación 2.13).

$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \quad g(x, y) = e^{-\frac{(x^2 + y^2)}{2\sigma^2}} \quad (2.13)$$

2.3.4.1. Algoritmos avanzados de detección de bordes

Además de estas técnicas básicas de detección de bordes, existen algoritmos avanzados que mejoran la tarea de detectar bordes tomando en cuenta otros factores como la presencia de ruido en las imágenes o la naturaleza de los bordes. A continuación se muestran los algoritmos más conocidos que usan técnicas de primera y segunda derivada.

Marr-Hildreth(1980) [29]: es uno de los primeros algoritmos sofisticados de búsqueda y análisis de bordes. Utiliza la segunda derivada para encontrar los bordes (búsqueda de los puntos de inflexión). Marr y Hildreth argumentaban en su desarrollo que los cambios de intensidad no eran independientes de la escala de imagen y su detección requería el uso de operadores de distintos tamaños.

El algoritmo de Marr-Hildreth consta de los siguientes pasos:

1. Suavizar la imagen utilizando el Gaussiano. Este filtrado reduce los errores debido al ruido.
2. Aplicar el Laplaciano a la imagen
3. Encontrar los puntos de inflexión de la imagen a partir del paso 2 (los puntos en los cuales la segunda derivada es 0)

Canny (1986) [4]: el detector de Canny es considerado como el algoritmos estandar en la industria. Canny consideró el problema de detección de bordes como un problema de optimización de procesamiento de señales y desarrolló una función para ser optimizada. Para acercar este enfoque a la detección de bordes, Canny se propuso tres objetivos:

- Baja tasa de error: los bordes detectados deben ser lo más cercanos posible a los bordes verdaderos.
- Precisión en la localización de los puntos de borde: la distancia entre un punto marcado como borde por el detector y el centro del borde verdadero debe ser mínima.
- Respuesta de un solo punto de borde: el detector debe retornar solo un punto por cada punto del borde real. Esto significa que el detector no debe identificar varios pixeles de borde donde solo existe un punto de borde.

Al ser considerado el problema como optimización, Canny utiliza la primera derivada para encontrar los puntos máximos de la imagen para detectar los bordes.

El algoritmo de Canny consta de los siguientes pasos:

1. Suavizar la imagen usando el filtro Gaussiano.
2. Calcular el gradiente de la imagen para obtener los puntos de cambio de intensidad.
3. Aplicar supresión nonmaximal. Es decir, al detectar los bordes con el gradiente, los puntos considerados para ser parte del borde deben ser los máximos de la primera derivada y los que no cumplen esta condición son removidos (o *suprimidos*) del borde, estableciendo su valor de intensidad en 0. Por lo cual se calcula la magnitud del gradiente para cada punto y si la magnitud del gradiente es mayor a un pixel de distancia en la dirección positiva o negativa del gradiente, se continúa ese camino, si no se elimina ese pixel.

4. Aplicar un umbral al borde: en este algoritmo se utiliza un doble umbral, uno alto y otro bajo. Si un pixel sobrepasa el umbral alto, es parte de los pixeles del borde. Si el valor del pixel es sobre el umbral inferior y es vecino de un pixel de borde, también es parte de los pixeles de borde. Si el valor del pixel sobrepasa el umbral inferior, pero no es vecino de un pixel de borde o su valor está por debajo del umbral inferior, este pixel no forma parte de los pixeles de borde.

2.4. Blurring en Imágenes

En la obtención de imágenes digitales, frecuentemente ocurre que la imagen resulta difusa por el movimiento del objeto a fotografiar, por un inadecuado enfoque o el uso de una abertura que da poca profundidad de campo; o como en el caso de las imágenes DGGE, por la complejidad de los procesos para obtenerlas.

En general, una imagen difusa o con blurring viene de la degradación de la señal, dada por algún tipo de convolución lineal [35], dado por la siguiente ecuación:

$$G = H \otimes I + N \quad (2.14)$$

donde G representa a la imagen con blurring, I es la imagen original, \otimes es el operador de convolución, N es el ruido que se agrega a la imagen y H es la función de blurring, la que puede ser de los siguientes tipos [2]:

- **Blurring Promedio:** se utiliza para reducir el ruido, principalmente en forma de puntos de la imagen. Este filtro se distribuye en dirección horizontal y vertical de la imagen (Figura 2.10b).
- **Blurring Gaussiano:** el filtro utilizado es la función gaussiana, donde los pesos de los pixeles no son iguales, sino que decrecen desde el centro del kernel aplicado a los bordes en base a la campana de Gauss de la imagen. Este filtro es denso en el centro y disminuye al llegar a los bordes (Figura 2.10c).
- **Blurring por Movimiento:** es un filtro que hace que la imagen parezca estar en movimiento y se realiza aplicando el filtro en una dirección específica. El movimiento puede ser controlado por ángulo, dirección, distancia o intensidad (Figura 2.10d).



Figura 2.10: Tipos de Blurring

Otro tipo de blurring existente es el *blurring fuera de foco* que consiste en plasmar una escena en 3D en una imagen en 2D, en donde algunas partes de la escena están en el foco y otras no; lo cual depende de la distancia entre la cámara y el objeto y la longitud del foco.

El principal problema de una imagen con blurring es que disminuyen el contraste de la imagen promediando las intensidades de los pixeles que están más cercanos a los bordes de líneas o áreas donde hay una significativa transición de intensidad, aunque esto se utiliza para reducir el efecto del ruido en las imágenes [10].

2.4.1. Solución del problema de blurring

El proceso de **deblurring** es el procesamiento de una imagen para obtener una mejor representación de la escena capturada, lo cual es fundamental para hacer que las imágenes sean nítidas y útiles en las áreas que se ocupan (medicina, biología, astronomía, logística, entre otras).

Dada la naturaleza matricial de las imágenes, los algoritmos y técnicas de *deconvolución*, es decir, la realización de operaciones matemáticas para restaurar una señal, son los más utilizados en solucionar este problema. La deconvolución es la operación inversa a la convolución, la cual es la definición matemática del blurring, tal como lo muestra la ecuación 2.14. Basado en esta operación, la técnica para la deconvolución es aplicar la transformada de Fourier inversa a la imagen, ya que de acuerdo al teorema de convolución, la aplicación de la transformada de Fourier a la convolución es igual al producto punto de las transformadas. La función de blurring H también es conocida como *función de dispersión de punto* (PSF). Las funciones de dispersión de punto describen la respuesta de la imagen a una fuente de punto definida por la función, o de manera más general, la respuesta del sistema de la imagen enfocada a un impulso. En general, una PSF es la inversa de la transformada de Fourier de la función de transferencia óptica en el dominio de la frecuencia.

Las técnicas de deconvolución se dividen en dos grupos de acuerdo a la disponibilidad del PSF. Si ya es dado o se tiene un estimado de la PSF en la técnica, ésta es conocida como **deblurring no ciego**, en cambio, las técnicas donde se tiene que realizar una estimación de la PSF son conocidas como técnicas de **deblurring ciego**. Para cualquiera de estos grupos, se tiene el supuesto de deblurring espacialmente invariante, es decir, que el kernel o PSF de la imagen en proceso es uniforme [23].

2.4.1.1. Deconvolución No Ciega

En este tipo de técnicas, la PSF es conocida a priori o es dada una estimación de ella para realizar la deconvolución de la imagen para minimizar el blurring o el ruido, dado un parámetro de regularización. En general, se basan en la repetición de un proceso a través de iteraciones. Algunas de estas técnicas son:

- **Algoritmo Lucy-Richardson** [49] es un procedimiento iterativo para recuperar una imagen latente⁹ borrosa a través de una PSF conocida. La representación de la intensidad de un pixel en una posición i , denotado por d_i , es dada por la siguiente fórmula:

$$d_i = \sum_j p_{ij} u_j \quad (2.15)$$

donde p_{ij} es la función de dispersión de punto, definida como la fracción de luz que viene de la posición j a la posición del observador i y u_j es la intensidad del pixel j de la imagen latente. Los valores de u_j indican la presencia de ruido de disparo¹⁰ del pixel, modelado estadísticamente a través de la distribución de Poisson.

⁹Una imagen latente es una imagen invisible obtenida en material fotográfico a través de la exposición a la luz y al revelarse se forma la imagen.

¹⁰El *ruido de disparo* es un tipo de ruido presente cuando partículas (fotones o electrones) producen fluctuaciones en la intensidad de la imagen.

La idea es calcular el más probable u_j dado el observado d_i y el conocido p_{ij} a través de una ecuación que puede ser resuelta iterativamente de acuerdo a la siguiente fórmula:

$$u_j^{(t+1)} = u_j^{(t)} \sum_i \frac{d_i}{c_i} p_{ij} \quad (2.16)$$

donde

$$c_i = \sum_j p_{ij} u_j^{(t)}$$

Empíricamente la ecuación 2.16 converge al valor más probable de u_j

- **Métodos no basados en minimización:** En [7] se ocupa un método iterativo utilizando un filtro lineal estimado para obtener la solución a través de un problema de equilibrio de Nash, buscando de esta forma los valores mínimos para obtener una imagen sin blurring. También se han propuesto métodos, como el descrito en [50], que no utilizan iteraciones, sino que a través de la regresión de kernel, para estimar la PSF, utilizan transformaciones locales (series de Taylor, métodos de descenso de gradiente) en base a la información de los vecinos para obtener una imagen con poca presencia de blurring.

2.4.1.2. Deconvolución Ciega

En estos métodos, no se conocen la imagen objetivo I y el filtro utilizado H , por lo que se tiene que primero determinar la PSF o filtro para llegar a la imagen objetivo, lo cual se puede realizar de forma iterativa. Se agrega también un parámetro de regularización para detener este proceso iterativo. Las técnicas que utilizan este enfoque son las siguientes:

- **Estimación máxima a posteriori (MAP) [26]:** Esta técnica bayesiana consiste en maximizar la probabilidad posterior de los parámetros desconocidos de una entrada dada. En el caso del blurring, se quiere maximizar la probabilidad de obtener la imagen I y el filtro H simultáneamente (asumiendo que ambos parámetros son independientes), en base a la imagen borrosa G , de acuerdo a la fórmula de Bayes que se muestra en la ecuación 2.17:

$$p(I, H|G) \propto p(G|I, H)p(I)p(H) \quad (2.17)$$

La probabilidad previa $p(I)$ favorece imágenes naturales, usualmente basadas en la observación y en donde su gradiente es disperso. Una medida común para ella es:

$$\log p(I) = - \sum_i |g_{x,i}(I)|^\alpha + |g_{y,i}(I)|^\alpha + C \quad (2.18)$$

en donde $g_{x,i}(I)$ y $g_{y,i}(I)$ son las derivadas horizontales y verticales del pixel i y C es una constante de normalización. Valores de $\alpha > 1$ llevan a distribuciones previas dispersas y las imágenes naturales usualmente corresponden a α entre el rango $[0,5, 0,8]$

Si se usa la medida de la ecuación 2.18, el problema de optimización se presenta en la fórmula 2.19

$$(I, H) = \arg \min_{I, H} \lambda \|H \otimes I - y\|^2 + \sum_i |g_{x,i}(I)|^\alpha + |g_{y,i}(I)|^\alpha \quad (2.19)$$

Para optimizar la ecuación 2.19 de acuerdo a [26], $|I| \rightarrow 0$ y $|H| \rightarrow \infty$, lo que resulta en una imagen plana y sin gradientes.

Una forma para solucionar este problema consiste en asumir que la intensidad media de las imágenes borrosa y nítida sea igual y restringir la sumatoria de H a $\sum_i H_i = 1$. Otra propuesta de solución es utilizar un esquema de iteraciones de dos fases, que se calculen H e I por separado, marginalizando en la primera fase I para realizar una estimación MAP_H obteniendo el kernel y para obtener la imagen se usan técnicas de deconvolución no ciega [27].

- **Métodos basados en las estadísticas de la imagen** [25]: En las imágenes que presentan un blurring por movimiento en diferentes direcciones, el uso de diferentes kernels es considerado segmentando la imagen completa para ver si el cambio en las derivadas de ella es muy grande comparándolas con los diferentes filtros probados. Estos métodos buscan un modelo mixto que define la mejor distribución observada en la imagen. Este modelo consiste en buscar los kernel que modelen la imagen difusa a través de la distribución de probabilidad observada en la imagen, tras lo cual se obtiene un conjunto de dos kernel. Utilizando cada kernel con las capas suavizadas de la imagen, se segmenta ésta en pequeñas ventanas, de acuerdo a la similitud entre la imagen original y la aplicación de cada kernel a ella. Finalmente, con la aplicación de kernels y la segmentación realizada se maximiza la similitud de las ventanas y se suavizan. Los resultados dependen del uso de los filtros, los cuales pueden ser filtros promedio, determinar la dirección o el tamaño del filtro de blurring en base a las características de la imagen para aumentar el rendimiento del método.

2.4.1.3. Otras técnicas de deblurring

Además de los métodos basados en la deconvolución, se han utilizado también las redes neuronales para quitar el blurring de una imagen. Las redes neuronales son una forma de un sistema multiprocesador que se componen de elementos de procesamiento simples con un alto grado de interconexión e interacción entre ellos. Aprovechando la conversión matricial, las redes neuronales proveen una robusta herramienta para aproximar una función objetivo, dado un set de ejemplos y permiten la reconstrucción de una función a través de una clase de imágenes. Sus elementos constitutivos, conocidos como *perceptrones* y el método de aprendizaje llamado *backpropagation* usan técnicas de gradiente descendiente para ajustar la red con un conjunto de ejemplos de entradas y salidas para su entrenamiento, para luego, ser utilizadas en el procesamiento de las imágenes con blurring [1].

El trabajo con las redes neuronales permite extender el modelo de trabajo más allá de las funciones lineales, ya que su estructura permite que aprendan funciones no lineales complejas para producir mejores resultados en las zonas de alta intensidad de la imagen. Principalmente se utilizan redes con backpropagation de dos capas totalmente conectadas [51].

2.5. Estado del Arte en Detección de Bordes

Además de los métodos descritos en el capítulo 2.3.4, existen otras técnicas que van más allá del cálculo matricial para detectar los bordes. De acuerdo a la topología de la imagen, es posible segmentarla para encontrar similitudes entre píxeles y regiones. A través de las transformaciones divisorias por inundación, las cuales consisten en colocar una fuente de agua que fluya sobre el relieve topográfico de forma descendente, se convierte la magnitud del gradiente de la imagen en píxeles topográficos que tienen la máxima intensidad de la magnitud del gradiente (de acuerdo a la altura del relieve), representados como líneas divisorias, indicando las regiones de borde. Esto puede ser calculado de dos maneras, escogiendo el mínimo local del gradiente a través de marcadores, dados por la segmentación o por zonas de fusión de la imagen; o por el uso de una posición específica del marcador, definida explícitamente por el usuario o dada automáticamente por operadores morfológicos, detectan bordes siempre conectados y delgados [42].

Las transformadas de onda son otra forma para obtener una imagen segmentada en base a su contraste y también permiten estimar valores de bordes en un espacio de manera local o global. Definidas como la suma sobre todas las filas y columnas de la función de intensidad de la imagen multiplicada por una versión desplazada de una función de onda, para mapearla en dos dimensiones, frecuencia y espacio (filas y columnas). Esto permite mapear la intensidad de la imagen en alguna de estas dimensiones y obtener intervalos y puntos máximos, los cuales se obtienen de los peaks positivos y negativos de este mapeo. La obtención de puntos máximos en una pequeña escala permite determinar una posición de borde. Extendiendo el modelo a una situación con ruido, permite que los bordes solo dependan de la pendiente y se puedan distinguir, ya que los puntos máximos de intensidad, que son bordes, son mayores que los de la intensidad de la zona con ruido [61].

Las técnicas heurísticas, empleadas por sí solas o combinadas entre ellas, son otro enfoque de solución al problema de detección de bordes. El uso de lógica difusa proporciona diferentes posibilidades para tratar el problema de la detección de bordes [21]. Una forma es definir una función de membresía que indique por cada vecindad de píxeles la pertenencia a una zona de borde, modificándola en parte para ser determinada heurísticamente y realizar la decisión a través de un sistema difuso. Otra forma consiste en utilizar reglas *if-else* apropiadas que puedan desarrollarse en vecindades generales y específicas, medibles en base a la similitud u homogeneidad entre dos regiones segmentadas, que entreguen como respuesta una imagen con los bordes detectados.

Presentado como un problema de segmentación, se han utilizado los algoritmos genéticos en [43], minimizando una función objetivo en base a un conjunto de parámetros dependientes del espacio de soluciones de las configuraciones de borde de una imagen y obteniendo resultados a través de los operadores especializados de los algoritmos genéticos. Utilizando funciones de fitness que contribuyan a encontrar soluciones aceptables. Los criterios en que se basan las funciones de fitness son: el número de píxeles de borde localizados correctamente, el número de regiones detectadas, en el caso de una imagen con muchos elementos (como los granos de arroz) y el número de puntos considerados como borde en un borde fragmentado.

Las redes neuronales son utilizadas con un preprocesamiento para reducir el tamaño de las imágenes y ser utilizadas como inputs de las capas de la red. Este enfoque necesita de una fase supervisada la cual consiste en entrenar la red con imágenes de prueba o patrones para que realice la tarea de detección. Se han utilizado un conjunto de redes neuronales que procesan la saturación y la intensidad de los píxeles de la imagen. La mejor solución se determina a través de la mínima distancia respecto a la solución esperada [53].

Con máquinas de soporte vectorial (SVM) se presenta un algoritmo que emplea estimaciones de la intensidad de la imagen de una vecindad pequeña, utilizando la combinación de un conjunto de vectores para realizar la clasificación (segmentación) de los píxeles respecto a las zonas de borde, en base a un kernel gaussiano y aplicando la estimación en base al gradiente y los puntos donde la función es cero, obteniendo resultados similares a Canny [57].

Varios enfoques basados en algoritmos de enjambres han sido utilizados, tales como Optimización de Partículas (PSO) [5], algoritmos de hormigas (ACO) [18] y algoritmos de bacterias (BFA) [54]. Estas estrategias se basan en la colaboración de diversos agentes que individualmente tienen un procesamiento limitado, pero la unión de ellos produce buenas soluciones gracias a la adaptabilidad del conjunto de agentes. Estos algoritmos fueron probados en imágenes con presencia de ruido, los cuales tienen un buen comportamiento en estos casos. El movimiento de los agentes se debe a decisiones basadas en la probabilidad para escoger el mejor camino a tomar y maximizar la *energía* de ellos para realizar el recorrido, para así formar caminos, los cuales indicarán los bordes de la imagen obtenida.

2.5.1. Procesamiento en imágenes DGGE

Respecto a las imágenes DGGE, los principales problemas que surgen son el enderezamiento de los carriles de las bandas y la detección de las bandas propiamente tal. Debido al proceso de generación de imágenes DGGE, éstas presentan un alto nivel de ruido, bordes borrosos y a veces los carriles de las bandas tienen curvaturas, por lo que hay que enderezarlos, para utilizarlos en el proceso posterior de detección de bandas. Para enderezar los carriles se han utilizado enfoques adaptativos, como los algoritmos genéticos, utilizando un gran número de plantillas, para aprovechar la naturaleza combinatoria de estas técnicas y obtener buenos resultados [37]. También se han utilizado híbridos entre los algoritmos genéticos y otros métodos, como la búsqueda tabú y simulated annealing [38], lo cual ha ayudado en automatizar este proceso.

Ya que el ruido y bordes difuminados de las imágenes dificultan la detección de las bandas DGGE, se observaron algunas propiedades de las imágenes, en particular, la ubicación de las bandas a lo largo de ella, se propone un algoritmo para detectar la ubicación de las posibles bandas y separarlas, para obtener la información relevante a través del cálculo de un umbral [6].

Para la tarea de detección de las bandas, existen softwares que están basados en las técnicas tradicionales de detección de bordes, pero con resultados no muy satisfactorios, haciendo que esta tarea aún dependa del criterio de seres humanos para determinar qué es una banda en la imagen. Para automatizar este proceso se han utilizado principalmente técnicas evolutivas en imágenes con las mismas características, presencia de carriles y bandas; como la prueba de algoritmos basados en hormigas, Elitist Ant-System (EAS), comparándolo con el algoritmo de Canny y aplicando técnicas de segmentación para obtener una buena detección [6]; o comparar dos algoritmos de este tipo (Ant System (AS) y EAS) para evaluar el rendimiento en este tipo de imágenes, lo cual para mejorar la detección también requirió de técnicas de realce de imágenes [19]. En ninguno de estos trabajos se realizó preprocesamiento para tratar los problemas de blurring o ruido.

Capítulo 3

Técnicas Utilizadas en la Solución

3.1. Algoritmos Basados en Hormigas

Los algoritmos de Optimización de Colonias de Hormigas (Ant Colony Optimization, ACO) son técnicas metaheurísticas aplicadas a problemas de optimización. Se basan en el comportamiento de algunas especies de hormigas recolectoras, las cuales poseen la capacidad de encontrar el camino más corto desde su hormiguero hasta su comida y viceversa. Las hormigas no manifiestan un comportamiento individual, sino un comportamiento colaborativo entre ellas, bajo un sistema de comunicación indirecta, utilizando una sustancia que depositan en el camino llamada *feromona*. Las hormigas perciben la presencia de feromonas y tienden a seguir los caminos donde la concentración de feromonas es mayor, ya que esto indica que esos caminos han sido transitados frecuentemente por otras hormigas.

Este tipo de comportamientos entre insectos (como las hormigas, abejas) y otros animales (aves y peces) ha servido como modelo para un método de resolución de problemas conocido como *Inteligencia de Enjambre*; en el cual cada miembro del enjambre actúa de forma autónoma, pero existe una forma de comunicación indirecta entre ellos, la cual hace que exista una inteligencia social colectiva, que emerge desde la comunidad, pues cada individuo del enjambre no es capaz de desarrollar esta inteligencia por sí solo [18].

Este comportamiento de las hormigas fue estudiado por Deneubourg y su equipo en la especie *Linepithema humile*, realizando el experimento conocido como *doble puente*. En primer lugar, conectaron el nido de las hormigas con la fuente del alimento a través de dos puentes de la misma longitud y observaron que las hormigas elegían aleatoriamente uno de los dos caminos y después de un tiempo, uno de los dos puentes tenía una mayor concentración de feromonas, el cual era donde pasaba una mayor cantidad de hormigas. Goss modificó este experimento, cambiando las longitudes de los puentes, uno significativamente más largo que el otro. En este caso, las variaciones en la decisión de las hormigas fueron reducidas, pues ellas escogieron al poco tiempo el camino más corto y éste era el que contenía una mayor concentración de feromonas (Figura 3.1) [15].

Tomando como inspiración estos experimentos y los modelos propuestos por Deneubourg [9], Marco Dorigo y sus colaboradores diseñaron un modelo artificial de hormigas para emular el experimento del doble puente utilizando un grafo. Para esto, el grafo contenía dos nodos (el nido y la fuente de alimento) que estaban conectados por un arco corto y uno largo (siendo el arco largo r veces el corto). El movimiento de las hormigas se midió con tiempo discreto y ellas agregan una unidad de feromona en los arcos que usan, regidas a través de una probabilidad. Los investigadores llegaron a la misma conclusión que los experimentos realizados con hormigas naturales.

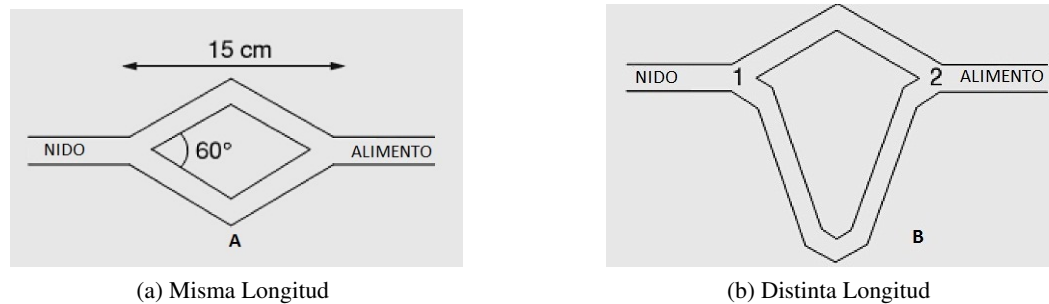


Figura 3.1: Experimentos de Doble Puente

Luego, Dorigo propuso su modelo para la solución de problemas de optimización, los cuales pueden ser estáticos o dinámicos. Los problemas estáticos son aquellos cuyas características están completamente determinadas desde el inicio de la construcción de una solución, como es el caso del vendedor viajero. Los problemas dinámicos son aquellos en que sus características no están determinadas y en ejecución van cambiando en función de diferentes valores que se van presentando en el problema, en este caso los algoritmos deben ser capaces de adaptarse a las nuevas condiciones que van surgiendo en el tiempo a medida que se va construyendo la solución al problema [12]. Dorigo utilizó su algoritmo para la solución del problema del vendedor viajero (Traveling Salesman Problem, TSP), un problema estático. A grandes rasgos, el TSP utiliza un grafo, cuyos nodos representan a un conjunto de ciudades y los arcos están rotulados con la distancia entre ellas. La solución a este problema es encontrar el camino más corto que permita recorrer todas las ciudades solo una vez. Las hormigas se mueven en el grafo de manera probabilística por los arcos y dejan su rastro de feromonas en los arcos que recorren. El algoritmo de Dorigo es iterativo, pues en cada iteración un número de hormigas es considerada y caminan por los vértices del grafo, con la condición de que no pueden visitar vértices anteriormente recorridos (es decir, poseen una memoria). En cada paso, la solución se construye eligiendo el siguiente vértice a visitar de acuerdo a un método estocástico que depende, en parte, de la concentración de feromona presente en los vértices y de un conocimiento de la convergencia del próximo estado de transición, disponible a priori. Al final de la iteración, en base a la calidad de las soluciones encontradas, la feromona es actualizada para mantener las mejores soluciones y mejorarlas a futuro.

La metaheurística general de los algoritmos ACO se muestra en el algoritmo 1.

Algoritmo 1: Metaheurística ACO

- 1 Fijar parámetros e inicializar rastro de feromona
 - 2 **while** *no se alcance la condición de término* **do**
 - 3 *las hormigas construyen sus soluciones*
 - 4 *aplicar búsqueda local (Opcional)*
 - 5 *actualizar feromona*
-

Muchos de los problemas computacionales que entran en la categoría NP-hard pueden ser tratados como problemas de optimización, por lo cual es posible aplicar algoritmos ACO para obtener soluciones cercanas al óptimo. Algunos ejemplos son el problema de enrutamiento vehicular (VRP), asignación de tareas, scheduling. También los algoritmos de hormigas son aplicables a problemas de telecomunicaciones o aplicaciones industriales [15].

3.1.1. Principales Algoritmos ACO

Diferentes algoritmos basados en hormigas se han propuesto. El algoritmo original fue propuesto en los primeros años de los 90 y es conocido como Ant System (AS). Desde entonces, otros algoritmos han sido introducidos, como el Elitist AS, MAX-MIN AS, Rank Based AS y el Ant Colony System. Todos estos algoritmos comparten la misma idea.

3.1.1.1. Ant System (AS)

Es el primer algoritmo ACO propuesto por Marco Dorigo en 1992 [11]; es la base de la implementación de otros algoritmos ACO que presentan buen rendimiento y efectividad para tratar los problemas de optimización. Inicialmente fueron propuestos tres tipos de algoritmos: *ant-density*, *ant-quality*, *ant-cycle*. En los dos primeros, la feromona se actualiza inmediatamente al movimiento de las hormigas de un nodo al otro. En *ant-cycle*, la actualización de las feromonas se realiza cuando todas las hormigas terminaron su camino. Al realizar pruebas de rendimiento sobre estos algoritmos, *ant-cycle* tuvo mejor rendimiento que los otros, por lo que fueron abandonados y *ant-cycle* se adoptó como el algoritmo Ant System.

En este algoritmo, m hormigas son puestas de forma aleatoria sobre el mapa (considerando el TSP) o grafo $G = (C, L)$, donde L es el conjunto de aristas que conectan completamente el conjunto de nodos C , para construir la solución del problema paso a paso. Una buena heurística en AS es inicializar el valor de la feromona presente en los arcos del grafo con un valor levemente mayor al que las hormigas pueden depositar, este valor es obtenido de acuerdo a la fórmula $\forall(i, j), \tau_{i,j} = \tau_0 = m/C^{mn}$, donde m es el número de hormigas y C^{mn} es el largo del tour generado con la heurística del *vecino-más-cercano*. La razón de la elección de este valor se debe a que si la feromona inicial τ_0 depositada es muy baja, la búsqueda se influenciará por los primeros caminos generados por las hormigas que por la exploración de otros recorridos. En cambio si τ_0 es muy alto, se pierden iteraciones esperando que se evapore la feromona depositada hasta un nivel que permita la exploración, afectando la búsqueda realizada por las hormigas. Las hormigas deciden de forma probabilística el siguiente nodo a visitar. La probabilidad que una hormiga k que está en un nodo i vaya a un nodo j está definida de la siguiente forma:

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}]^\alpha [\eta_{il}]^\beta}, \quad \text{si } j \in N_i^k, \quad (3.1)$$

donde $\eta_{ij} = 1/d_{ij}$ es un valor heurístico disponible a priori, α y β son dos parámetros que expresan la influencia del rastro de feromona en la información heurística y N_i^k es la vecindad factible de la hormiga k estando en la ciudad i , es decir, las ciudades que la hormiga k no ha visitado aún. Por medio de esta regla, la probabilidad de escoger un arco particular (i, j) aumenta con el valor τ_{ij} asociado y el valor de la información heurística η_{ij} . Los valores de α y β se eligen experimentalmente, y es importante escoger buenos valores para estos parámetros, pues una mala elección puede llevar al algoritmo a un estancamiento. El mejor valor para α es 1 y para β es entre 2 y 5 al usar algoritmos ACO en el TSP [15].

Cada hormiga k posee una memoria \mathcal{M}^k que contiene los nodos que ha visitado, en el orden que fueron visitados. Después que cada hormiga ha construido su solución, se actualizan los valores de feromonas de los arcos del grafo; primero por la disminución de la concentración de feromonas o evaporación, implementada a través de la siguiente ecuación:

$$\tau_{ij} \leftarrow (1 - \rho)\tau_{ij}, \quad \forall(i, j) \in L, \quad (3.2)$$

donde $0 < \rho \leq 1$ es la tasa de evaporación de la feromona, que se utiliza para evitar la acumulación ilimitada de ella y permite al algoritmo desechar las malas decisiones tomadas anteriormente.

Luego, las hormigas depositan feromonas solo en los arcos que han visitado en su tour (modelado en la ecuación 3.3):

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^m \Delta\tau_{ij}^k, \quad \forall (i, j) \in L, \quad (3.3)$$

con $\Delta\tau_{ij}^k$ la cantidad de feromona que la hormiga k deposita en los arcos que ha visitado y se define de la siguiente manera:

$$\Delta\tau_{ij}^k = \begin{cases} 1/C^k, & \text{si } (i, j) \in T^k \\ 0, & \text{e.o.c} \end{cases} \quad (3.4)$$

donde C^k es el largo del camino T^k construido por la hormiga k , calculado como la suma de los largos de los arcos pertenecientes a T^k .

3.1.1.2. Elitist-Ant System (EAS)

Es la primera modificación al AS hecha por Dorigo en 1992 [14], donde se agregó la *estrategia elitista*, cuya idea es añadir un reforzamiento especial a los arcos que componen la mejor ruta encontrada en una iteración, es decir, se agrega una mayor cantidad de feromona en la mejor ruta encontrada en cada iteración, denotada como T^{bs} (ruta *mejor – hasta – ahora*¹). La cantidad extra de feromona agregada a los arcos es e/C^{bs} , siendo e un parámetro dado a la mejor ruta encontrada hasta el momento (T^{bs}) y C^{bs} es el largo de esa ruta. La cantidad de feromona depositada por una hormiga en los arcos que recorre se calcula de acuerdo a la siguiente ecuación:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^m \Delta\tau_{ij}^k + e\Delta\tau_{ij}^{bs}, \quad \forall (i, j) \in L, \quad (3.5)$$

La evaporación de la feromona es la misma que en AS. El uso de esta estrategia con un valor aproximado de e , permite encontrar mejores resultados y en menos iteraciones. La asignación del valor de e se realiza en base a la experimentación y el mejor valor para e es el número de ciudades a recorrer en el TSP n , de acuerdo a [15].

3.1.1.3. MAX-MIN Ant System

En 1997 Stultze y Hoos proponen cuatro modificaciones al AS [46]:

- **Uso del mejor camino encontrado:** se deposita feromona en el mejor camino encontrado por una hormiga en cada iteración, de esta forma la deposición de feromonas es directa. En cambio, si se escoge la deposición por la mejor hormiga, se favorece la exploración de caminos, pero la deposición es menos directa. Esto se formaliza en la siguiente ecuación:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^{best}, \quad (3.6)$$

con $\Delta\tau_{ij}^{best} = 1/C^{best}$. La hormiga que agrega feromonas a los arcos, puede ser el mejor camino hasta ahora, en este caso $\Delta\tau_{ij}^{best} = 1/C^{bs}$ o la hormiga que obtuvo el mejor camino en la iteración actual, en el cual $\Delta\tau_{ij}^{best} = 1/C^{ib}$ (siendo C^{ib} el largo del mejor camino encontrado en la iteración actual).

¹En inglés, *best-so-far*, de ahí surge el superíndice *bs*

- **Limitación de la deposición de feromona:** lo anterior puede provocar un estancamiento en la búsqueda de la solución, por lo que se limita los posibles valores de feromonas en el intervalo $[\tau_{min}, \tau_{max}]$. El límite superior τ_{max} está acotado por $1/\rho C^*$, donde C^* es el largo de la mejor ruta encontrada, que se actualiza cada vez que se encuentra una ruta *mejor – hasta – ahora*. El límite inferior se calcula como $\tau_{min} = \tau_{max}/a$, siendo a un parámetro.
- **Inicialización de feromonas:** para aumentar la exploración en las primeras iteraciones del algoritmo, se inicializan los arcos con el valor τ_{max} , además de agregar una pequeña tasa de evaporación.
- **Reinicialización de caminos tras estancamientos:** cada vez que el sistema se acerca a un estancamiento o si por un número determinado de iteraciones no se ha generado alguna solución mejor de las que se llevan hasta ese momento, se reinician todos los caminos.

3.1.1.4. Rank Based Ant System (AS_{RANK})

Esta versión del AS, propuesta por Bullnheimer en 1997 [3], considera que cada hormiga deposita feromona en los arcos que recorre y decrece de acuerdo al ranking de la solución encontrada. Además, como en el EAS, la mejor hormiga hasta el momento siempre deposita una mayor cantidad de feromonas en cada iteración.

Antes de actualizar la cantidad de feromonas en los arcos, las hormigas son ordenadas por la longitud del camino (u otro criterio) encontrado de forma creciente y la cantidad de feromona que una hormiga deposita es pesada de acuerdo al ranking r de la hormiga. Los empates son resueltos aleatoriamente. En cada iteración, solo las $w - 1$ hormigas mejor rankeadas y la hormiga que produce la mejor solución hasta ahora (aunque no se encuentre en el grupo de las mejor rankeadas en la iteración actual) pueden depositar feromona (esta hormiga deposita feromona amplificada por un peso w). La regla de actualización de feromona está dada por:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{r=1}^{w-1} (w-r) \Delta\tau_{ij}^r + w \Delta\tau_{ij}^{bs}, \quad \forall (i, j) \in L, \quad (3.7)$$

con $\Delta\tau_{ij}^r = 1/C^r$ y $\Delta\tau_{ij}^{bs} = 1/C^{bs}$.

3.1.1.5. Ant Colony System (ACS)

Este algoritmo fue propuesto por Dorigo y Gambardella en 1997 [13] para aumentar el rendimiento del AS. Este algoritmo se diferencia en tres puntos del AS. Primero, se aprovecha la experiencia de búsqueda acumulada por las hormigas con más fuerza que el AS a través del uso de un regla de elección más agresiva. En segundo lugar, la feromona se deposita sólo en los arcos pertenecientes al mejor camino hasta el momento. En tercer lugar, cada vez que una hormiga utiliza un arco (i, j) para moverse de un nodo i a un nodo j , se elimina parte de la feromona del arco para aumentar la exploración de caminos alternativos.

Las hormigas son puestas en los nodos del grafo de acuerdo a algún criterio. Cada hormiga construye su ruta, moviéndose de un nodo i a un nodo j de acuerdo a la regla llamada *pseudorandom proportional*, definida por:

$$j = \begin{cases} \arg \max_{l \in N_i^k} \{\tau_{il} [\eta_{il}]^\beta\}, & \text{si } q \leq q_0 \\ J & \text{e.o.c} \end{cases} \quad (3.8)$$

donde q es una variable aleatoria con distribución uniforme en $[0, 1]$, q_0 es un parámetro ($0 \leq q_0 \leq 1$) que da mayor o menor importancia al movimiento determinista o probabilista de la hormiga y J una variable aleatoria

seleccionada de acuerdo a la distribución de probabilidad dada en (3.1). La función $\arg \max_{l \in N_i^k} \{\tau_{il} [\eta_{il}]^\beta\}$ elige el máximo valor encontrado para $\tau_{il} [\eta_{il}]^\beta$, donde $l \in N_i^k$ indica todas las ciudades l que pertenecen a la vecindad de i y que no han sido visitados.

Durante la construcción de la solución, inmediatamente después que una hormiga cruza un arco (i, j) , se actualiza la feromona presente a través de la siguiente regla local de actualización:

$$\tau_{ij} \leftarrow (1 - \xi)\tau_{ij} + \xi\tau_0 \quad (3.9)$$

siendo ξ , $0 < \xi < 1$ y τ_0 parámetros. Experimentalmente, un buen valor para ξ es 0.1 y para $\tau_0 = 1/nC^{mn}$, con n el número de ciudades y C^{mn} , el largo del camino encontrado con la heurística del *vecino-más-cercano*.

El efecto de la regla de actualización local es que cada vez que una hormiga visite un arco (i, j) la feromona sea reducida de manera que dicho arco sea menos “deseable” para las siguientes hormigas, esto le da más oportunidades a los arcos que aún no han sido visitados, lo que a su vez reduce el estancamiento del algoritmo.

Después que todas las hormigas han construido sus soluciones, se modifica nuevamente la cantidad de feromona a través de una regla global de actualización. En esta regla, solo una hormiga (la mejor hasta ahora) es la que añade feromona por cada iteración. Esta regla se modela a través de la siguiente ecuación:

$$\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \rho\Delta\tau_{ij}^{bs}, \quad \forall (i, j) \in T^{bs} \quad (3.10)$$

donde $\Delta\tau_{ij}^{bs} = 1/C^{bs}$

3.2. Lógica Difusa

Es frecuente encontrar procesos y sistemas que dada su naturaleza compleja o incierta no es posible desarrollar un modelo exacto para representar sus componentes y funcionamiento. Además, cotidianamente en el lenguaje natural se utilizan expresiones para describir situaciones ambiguas. En todos estos casos la toma de decisiones a partir de información imprecisa es un procedimiento usual, generalmente expresado en términos cualitativos. Un experto es capaz de controlar un proceso basado en su conocimiento y observación aun sin un modelo matemático. Son este tipo de situaciones las que se intentan emular a través de la *lógica difusa*, en donde se usan expresiones que no son totalmente verdaderas ni completamente falsas, sino que los conceptos aplicados en esta lógica puede tomar un valor cualquiera de veracidad dentro un conjunto de valores que se encuentran entre dos extremos: la veracidad absoluta y la falsedad total, y en base a la observación del entorno, la formulación de reglas lógicas y de los mecanismos de toma de decisión.

En términos generales la lógica difusa es una lógica multivaluada, es decir, en donde existen más de dos valores de verdad. Cuando Aristóteles propuso su *Principio del Tercero Excluido*, en el cual se indica que un término puede ser verdadero o falso, a lo largo de la historia han surgido objeciones, como las de Heráclito que indicó que las cosas pueden ser simultáneamente verdaderas o no verdaderas. Platón considera la existencia de una tercera región que va más allá de lo verdadero y lo falso. A principios del siglo XX, Lukasiewicz describe una lógica trivaluada en la que propone que el tercer valor de verdad sea conocido como *posible*, asignándole un valor numérico entre lo verdadero y lo falso, indicando su notación y el sistema axiomático que sustenta su modelo. Con esto dio las bases para indicar la existencia de lógicas multivaluadas [20].

Esta noción de lógica infinitamente valuada fue introducida por Lotfi A. Zadeh en 1965, donde presenta los conceptos y operaciones de los conjuntos difusos y su extensión a la lógica difusa. La lógica difusa usa funciones de pertenencia que operan sobre el rango $[0,1]$ y operaciones realizadas sobre esta nueva lógica, presentada al principio como una generalización de la lógica clásica [58]. El concepto de *difuso* no indica que la lógica tenga características imprecisas, sino los objetos que se estudian a través de ella.

3.2.1. Conjuntos Difusos

Un conjunto difuso es aquel que no tiene sus límites bien definidos. Es decir, existe una transición gradual que indica la pertenencia de un elemento al conjunto, medida a través de una función de pertenencia, la cual a cada elemento del conjunto le asigna un grado de pertenencia al conjunto; en contraposición a un conjunto clásico donde la pertenencia de un elemento a un conjunto está claramente definida por un límite abrupto. Esta diferencia se muestra en la figura 3.2.

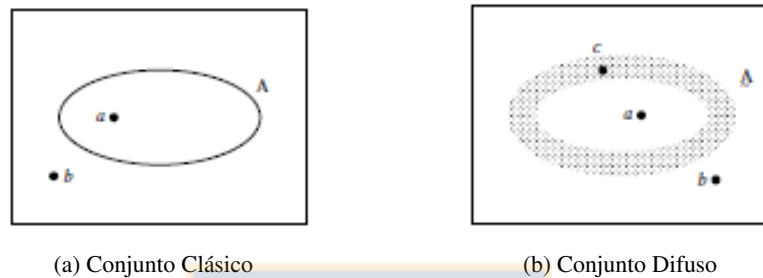


Figura 3.2: Conjuntos Clásicos y Difusos

Matemáticamente, un conjunto difuso A en un universo X de elementos x es definido de la siguiente forma:

$$A = \{(x, \mu_A(x)) \mid x \in X\} \quad (3.11)$$

donde $\mu_A(x)$ es la función de pertenencia de x en A , que representa el grado en que x pertenece al conjunto A en un valor perteneciente al intervalo continuo $[0,1]$. Un valor de 0 en la función de pertenencia indica que un elemento x definitivamente no pertenece al conjunto A . El valor de 1 en $\mu_A(x)$ implica que un elemento x pertenece definitivamente al conjunto y si el valor de la función de pertenencia está entre 0 y 1, el elemento x está en el borde del conjunto difuso. En un conjunto clásico, los valores de la función de pertenencia están restringidos a 0 y a 1, donde los límites están bien definidos [58].

3.2.1.1. Operaciones en Conjuntos Difusos

Sean A y B dos conjuntos difusos sobre el universo X , para un elemento x se definen las siguientes operaciones sobre X :

1. **Complemento:** el complemento de un conjunto difuso, denotado por A' se define así:

$$\mu_{A'}(x) = 1 - \mu_A(x) \quad (3.12)$$

2. **Intersección:** la intersección de dos conjuntos difusos A y B y sus respectivas funciones de pertenencia es un conjunto difuso $C = A \cap B$, denotado como:

$$\mu_C(x) = \mu_A(x) \wedge \mu_B(x) = \min(\mu_A(x), \mu_B(x)) \quad (3.13)$$

donde la relación \wedge propuesta por Zadeh busca que un elemento x que pertenezca a C debe pertenecer a ambos conjuntos de forma simultánea, es por esto que se utiliza la operación \min , pues el menor valor de las funciones de pertenencia .

3. **Unión:** la unión de dos conjuntos difusos A y B y sus respectivas funciones de pertenencia es un conjunto difuso $C = A \cup B$, denotado como:

$$\mu_C(x) = \mu_A(x) \vee \mu_B(x) = \max(\mu_A(x), \mu_B(x)) \quad (3.14)$$

la relación \vee propuesta por Zadeh permite que un elemento x pueda pertenecer a uno u otro de los conjuntos que se unen y el mayor valor de las funciones de pertenencia de ellos será el obtenido.

4. **Subconjunto:** A es un subconjunto de B si y solo si $\mu_A(x) \leq \mu_B(x)$ para todo x , denotado de la siguiente forma:

$$A \subseteq B \Leftrightarrow \mu_A(x) \leq \mu_B(x) \quad (3.15)$$

5. **Igualdad:** Dos conjuntos A y B son iguales si y solo si:

$$A = B \Leftrightarrow \mu_A(x) = \mu_B(x) \quad (3.16)$$

Gráficamente, estas relaciones se muestran en la figura 3.3

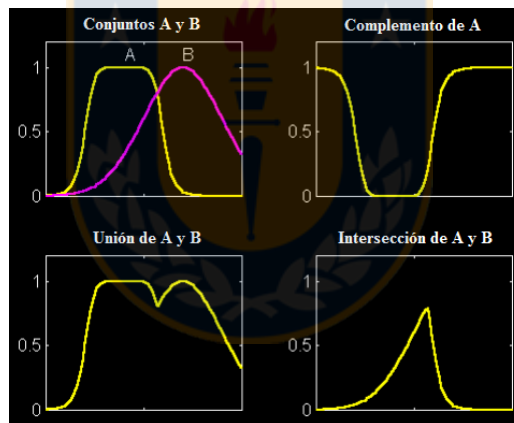


Figura 3.3: Operaciones en Conjuntos Difusos

Estas operaciones cumplen con la conmutatividad, asociatividad, distributividad, idempotencia, involución² y leyes de De Morgan al igual que los conjuntos clásicos. Sin embargo, el Principio del Tercero Excluido no se cumple en la lógica difusa, pues los conjuntos difusos con su complemento se superponen [40], como se muestra en la ecuación 3.17.

$$\begin{aligned} A \cup A' &\neq X \\ A \cap A' &\neq \phi \end{aligned} \quad (3.17)$$

²La involución es la aplicación del doble complemento $A'' = A$

3.2.2. Funciones de Pertenencia

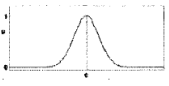
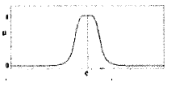
Como se definió anteriormente, una **función de pertenencia** representa el grado de pertenencia de un elemento a conjunto definido por una etiqueta, como también es la representación gráfica de un conjunto difuso. Las etiquetas son los valores que toma una *variable lingüística*, las cuales comprimen información, pues se componen de muchos valores numéricos. Por ejemplo, la variable *Edad* tiene como etiquetas *joven, muy joven, niño, anciano, bebé, adulto, etc.* [60].

Existen varias funciones que se utilizan para representar conjuntos difusos, pero generalmente se utilizan las siguientes, dada su capacidad para representar algunas situaciones recurrentes [62]:

1. **Funciones definidas por tramos:** son el tipo de funciones de pertenencia más simples. Se componen dos límites que son las bases de la gráfica. Al graficarlas, generalmente toman la forma *triangular* o *trapezoidal* y pueden ser simétricas o asimétricas. Son utilizadas para representar valores intermedios tales como *joven, de mediana edad, etc.*
2. **Funciones polinomiales:** son funciones cuyas curvas tienen una inclinación que es definida por sus puntos límites a y b . Las formas más comunes de este tipo de funciones son de S, Z y de pi (Π), que es la suma de las dos funciones anteriores. Se usan para representar conceptos extremos como *bebé, anciano, muy frío, etc.*
3. **Gaussiana** esta función transforma los valores originales a una distribución normal. El punto medio de la distribución normal determina la definición ideal para el conjunto. La pertenencia del resto de los valores de entrada disminuye a medida que se alejan del punto medio, tanto en la dirección positiva como en la negativa. El ancho de la función es definida por su desviación estandar σ y un parámetro c para indicar el centro de la función.
4. **Campana:** de forma más general que la función gaussiana, las funciones en forma de campana se componen de tres parámetros, lo que permite que sean adaptables a diferentes situaciones. Las funciones de campana y gaussianas son ampliamente utilizadas por ser curvas suaves y tener una notación concisa.

Tabla 3.1: Funciones de Pertenencia más utilizadas

Función	Gráfica	Función	Gráfica
<p>Triangular:</p> $f = u_{triang}(x, a, b, c)$ $f = \begin{cases} 0 & x \leq a \\ \frac{x-a}{b-a} & a \leq x \leq b \\ \frac{c-x}{c-b} & b \leq x \leq c \\ 0 & c \leq x \end{cases}$		<p>Trapezoidal:</p> $f = u_{trap}(x, a, b, c, d)$ $f = \begin{cases} 0 & x \leq a \\ \frac{x-a}{b-a} & a \leq x \leq b \\ 1 & b \leq x \leq c \\ \frac{d-x}{d-c} & c \leq x \leq d \\ 0 & d \leq x \end{cases}$	
<p>Forma de S:</p> $f = u_{smf}(x, a, b, c)$ $f = \begin{cases} 1 & x \leq a \\ 2 \left(\frac{x-a}{b-a} \right)^2 & a \leq x \leq \frac{a+b}{2} \\ 1 - 2 \left(\frac{x-a}{b-a} \right)^2 & \frac{a+b}{2} \leq x \leq b \\ 0 & b \leq x \end{cases}$		<p>Forma de Z:</p> $f = u_{zmf}(x, a, b)$ $f = \begin{cases} 1 & x \leq a \\ 1 - 2 \left(\frac{x-a}{b-a} \right)^2 & a \leq x \leq \frac{a+b}{2} \\ 2 \left(\frac{x-a}{b-a} \right)^2 & \frac{a+b}{2} \leq x \leq b \\ 0 & b \leq x \end{cases}$	

Función	Gráfica	Función	Gráfica
Gaussiana: $f = u_{gauss}(x, \sigma, c)$ $f = e^{-\frac{(x-c)^2}{2\sigma^2}}$		Campana: $f = u_{bell}(x, a, b, c)$ $f = \frac{1}{1 + \left \frac{x-c}{a} \right ^{2b}}$	

3.2.3. Proceso de Inferencia Difusa

Tal como en la lógica tradicional, la lógica difusa utiliza las variables lingüísticas en forma de proposiciones *imprecisas* basadas en la teoría de conjuntos difusos para producir un *razonamiento aproximado o difuso*. Para este objetivo se utiliza un **sistema de inferencia difuso** cuya idea básica es incorporar el conocimiento humano en un conjunto de reglas de inferencia difusas. Un sistema de inferencia difuso se compone de cuatro partes: una *interfaz de fusificación*, un *mecanismo de inferencia*, un *conjunto de reglas* y una *interfaz de defusificación* [36]. Este modelo de inferencia difusa se puede ver en la figura 3.4.

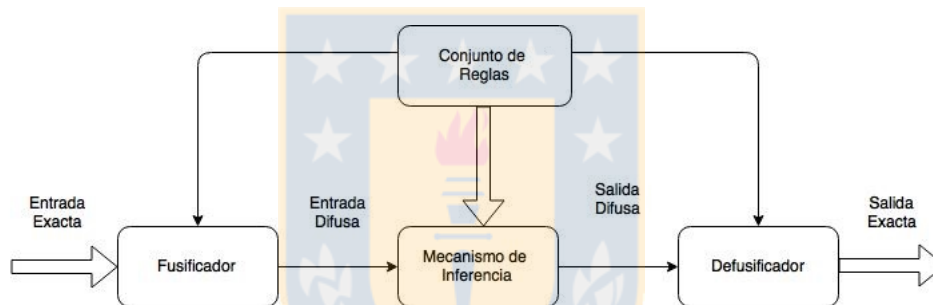


Figura 3.4: Modelo de Inferencia Difusa

3.2.3.1. Fusificación

Es el proceso de transformar las entradas o parámetros de un sistema, expresados en forma exacta, en variables difusas, las cuales pueden ser utilizadas en el sistema difuso. En este proceso se asignan grados de pertenencia a cada una de las variables de entrada en relación a los conjuntos difusos definidos previamente utilizando las funciones de pertenencia asociadas a esos conjuntos difusos.

El primer paso en la fusificación es dividir el universo de discurso, asignando etiquetas en cada variable difusa. Después, se establecen las funciones de pertenencia para dar significado numérico a cada etiqueta. En el proceso, las funciones de pertenencia definidas sobre las variables de entrada mapean a los datos presentes para determinar el grado de pertenencia.

3.2.3.2. Mecanismo de Inferencia Difusa

Tras fusificar las variables de entrada se realiza el proceso de realizar el mapeo de estas variables a una salida difusa a través del mecanismo de inferencia difusa. En esta etapa se proveen las bases para la toma de decisiones del sistema. Este proceso involucra la utilización de las funciones de pertenencia y las reglas generadas a partir de una base de conocimiento. Esta base de conocimiento se compone de dos elementos: una *base de datos* y un *conjunto de reglas*.

- **Base de Datos:** aquí están definidos el número, etiquetas y los tipos de las funciones de pertenencia de los conjuntos difusos utilizados como valores para cada variable del sistema, las que pueden ser de entrada o de salida. Los conjuntos difusos escogidos forman parte del universo de discurso de cada variable.
- **Conjunto de Reglas:** realiza el mapeo de las variables difusas de entrada a variables difusas de salida. La estrategia de control está presente en este conjunto de reglas, la cual pondera y combina los conjuntos difusos resultantes del proceso de inferencia en un valor exacto (0 ó 1) para cada salida. Las reglas utilizadas en este conjunto usualmente son del tipo IF-THEN, descritas de la siguiente forma general:

Regla i : IF x es A_i and y es B_i , THEN z es C_i

donde x e y son variables de entrada, z es una variable de salida; A_i, B_i y C_i son conjuntos difusos o términos lingüísticos como 'positivo', 'negativo', etc. Estas reglas se componen de un antecedente o premisa (la parte del IF) y un consecuente o acción (la parte del THEN) [59].

Existen diferentes modelos de inferencia a partir de las reglas IF-THEN, sin embargo usualmente se utilizan dos, los cuales varían en la forma en que las salidas son calculadas,;

- **Modelo de Mamdani:** Propuesto en 1975 por Ebrahim Mamdani [28] aplicado a un control de una máquina de vapor y una caldera, este modelo de inferencia no requiere de modelos matemáticos del sistema a controlar, pues las reglas se expresan en términos lingüísticos obtenidas de la experiencia de los expertos humanos. La forma de las reglas utilizadas en este modelo es la siguiente:

IF x_1 es A_{1i} AND ... AND x_n es A_{ni} , THEN y es B_i

siendo x_1, \dots, x_n variables de entrada, y es variable de salida y $A_{1i}, \dots, A_{ni}, B_i$ son conjuntos difusos.

Los métodos de inferencia utilizados en el modelo de Mamdani, en relación al significado del operador AND son **min-max** y **max-product**. El método *min-max* le da el significado al operador AND como el mínimo valor de los antecedentes (Ecuación 3.18), mientras que el método *max-product* usa el producto de los antecedentes como AND (Ecuación 3.19).

$$x_1 \text{ AND } \dots \text{ AND } x_n = \min\{x_1, \dots, x_n\} \quad (3.18)$$

$$x_1 \text{ AND } \dots \text{ AND } x_n = x_1 x_2 \dots x_n \quad (3.19)$$

Tras la aplicación del método de inferencia difusa se genera un conjunto difuso de salida B usando el operador de unión.

- **Modelo de Takagi-Sugeno:** El modelo propuesto por Takagi, Sugeno y Kant fue planteado como un esfuerzo para desarrollar una aproximación sistemática para generar reglas difusas a partir de un conjunto de datos de entrada y salida difusas [47]. Una regla típica que utiliza este modelo es la siguiente:

IF x_1 es A_{1i} AND ... AND x_n es A_{ni} , THEN $y = f(x_1, \dots, x_n)$

donde A_{1i}, \dots, A_{ni} son conjuntos difusos de entrada e $y = f(x_1, \dots, x_n)$ es una función nítida que describe apropiadamente la salida del modelo dentro de la región difusa especificada en el antecedente de la regla. Generalmente la función es un polinomio, cuando f es una constante, el modelo de Sugeno actúa como un caso especial del modelo de Mamdani, en el cual cada función del consecuente es especificada por un punto único difuso. En este modelo no se necesita una etapa de defusificación.

3.2.3.3. Defusificación

El proceso de defusificación implica la ponderación y combinación de los conjuntos difusos que resultan del proceso de inferencia difusa en calcular un valor discreto y exacto para cada output. Los métodos más comunes de defusificación son el *centro de área* (COA), la *media del máximo* y los *criterios del máximo*.

- **Centro de Área (COA):** Una salida exacta \hat{y} es elegida como el centro de área de la función de pertenencia μ_B del conjunto difuso B obtenido en la implicación. En un universo discreto Y se tiene que:

$$\hat{y} = \frac{\sum_{j=1}^m y_j \mu_B(y_j)}{\sum_{j=1}^m \mu_B(y_j)} \quad (3.20)$$

donde m es el número de elementos del conjunto B . Un método similar a éste es el *centro de sumas*, en donde se considera la obtención del conjunto B como la suma de los conjuntos difusos en el proceso de implicación. El centro de sumas es un proceso más rápido que el centro de área, por lo cual se utiliza en variadas aplicaciones actualmente. [16].

- **Media del Máximo:** El valor de salida \hat{y} es elegido para representar el valor promedio de los elementos cuyo grado de pertenencia en el conjunto difuso B sea el máximo. El cálculo discreto del valor de salida a través de este método es el siguiente:

$$\hat{y} = \frac{\sum_{j=1}^m y_j}{m} \quad (3.21)$$

siendo y_j el j -ésimo elemento del universo cuya función de pertenencia tiene el máximo valor y m el total de dichos elementos. Esta técnica de defusificación permite lograr los valores de salida mucho más rápidos que con el COA, sin embargo, no alcanza los valores extremos del universo de discurso, es decir, permite encontrar el valor máximo dentro de un intervalo definido [36].

- **Criterios del Máximo:** La salida \hat{y} tiene el máximo valor de todos aquellos puntos del conjunto difuso B que al evaluarlos con la función de pertenencia, ésta retorna el valor más alto. También es posible escoger el punto con el valor mínimo o la mediana de aquellos donde la función de pertenencia alcanza el valor máximo, cuando la función de pertenencia tiene varios puntos donde alcanza su máximo valor. En el caso que la función de pertenencia alcance su máximo en un solo punto, los diferentes criterios del máximo serán iguales, indicando ese punto.

Capítulo 4

Diseño de la Solución

4.1. Descripción del Problema

Al generar las imágenes DGGE tras el proceso de obtención (descrito en la Sección 2.2.1), se observa que para identificar las bandas de la imagen, los softwares de los investigadores adolecen de precisión, pues en un alto porcentaje de los casos, el usuario debe corregir manualmente lo que corresponde a una banda según el criterio de la persona, lo cual hace de este proceso una labor muy tediosa. Además, no se cuenta con una aplicación que permita cuantificarlas, lo que hace que el DGGE sea una técnica cualitativa. Por lo anterior los problemas a tratar en esta investigación son los relacionados con la detección e identificación de bandas. [19].

4.1.1. Detección e Identificación de Bandas

Uno de los problemas consiste en la falta de precisión y de sensibilidad a la hora de detectar y cuantificar las bandas de la imagen DGGE, por parte de un software de detección automática de bandas, ya que este opera a través de estándares, los cuales conllevan a arrojar resultados con algún porcentaje de falla, estos errores se pueden observar en las imágenes que se visualizan en la Figura 4.1. Se puede ver que en el carril de las figuras la detección automática de bandas es deficiente, ya que el usuario deberá borrar lo que a su parecer no corresponde a una banda.

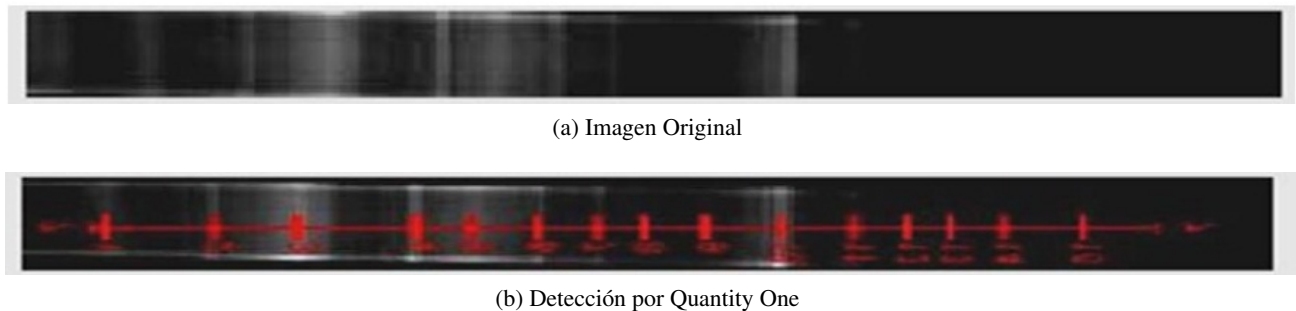


Figura 4.1: Detección de Bandas en una Imagen DGGE por parte del software Quantity One

En la figura 4.2 se puede apreciar de manera completa lo que se mencionó como deficiencia en el procesamiento de una imagen DGGE, en donde se aprecian los distintos problemas que se tienen al detectar las bandas: bandas muy acopladas entre sí (a) o bandas muy difusas (b) y (c) [19].

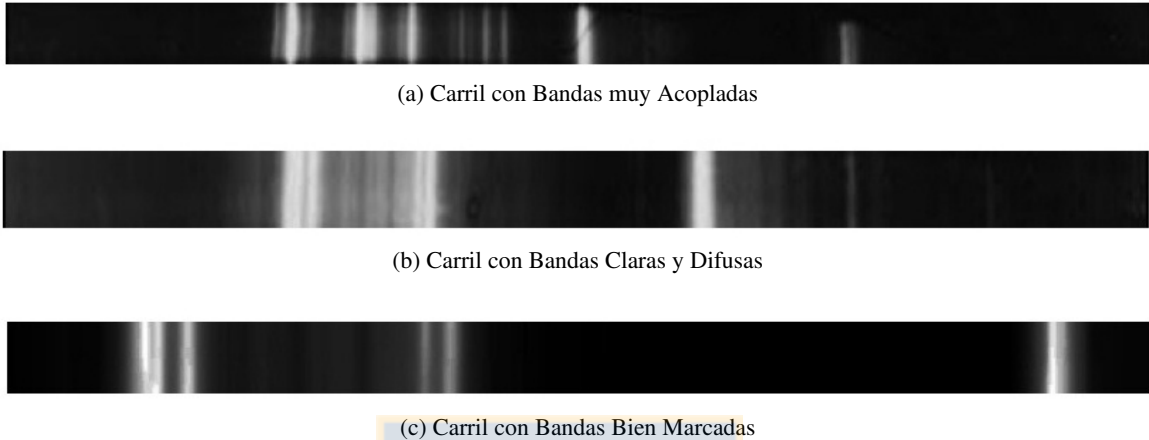


Figura 4.2: Deficiencias en las imágenes DGGE

4.2. Propuesta de Solución

Para resolver el problema de detección de bandas en imágenes DGGE se construirá un sistema de software que recibirá como input las imágenes DGGE, para las cuales se permiten los distintos formatos de imagen e imágenes con distintas profundidades de tono (sean en escala de grises o a color, ya que a estas últimas se les aplicará una transformación para que sean convertidas a escala de grises para su posterior procesamiento). El tratamiento de las diferencias entre el color de las bandas en distintas imágenes, sean blancas o negras, como se puede ver en la imagen 4.3, no requiere ningún tipo de tratamiento adicional (como la inversión de la imagen), ya que se espera que el sistema propuesto trate por igual ambos colores de bandas.

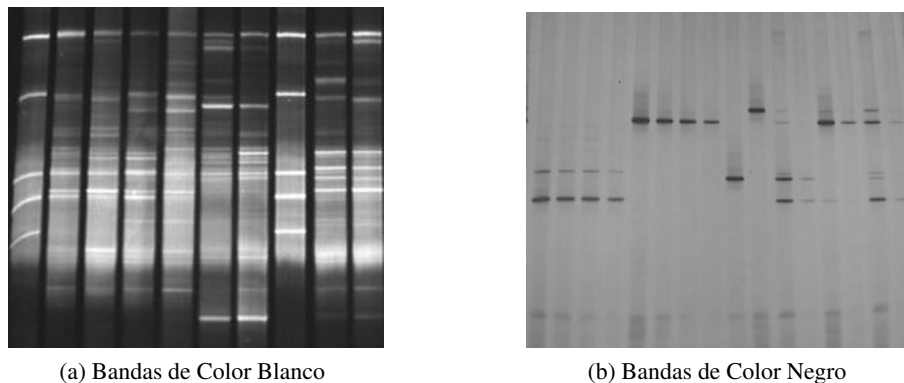


Figura 4.3: Diferencias de color en las bandas de distintas imágenes

Para la construcción del sistema, de acuerdo a la revisión bibliográfica (Capítulo 2.5) en donde se han utilizado algoritmos basados en hormigas para la solución del problema planteado; se optará por esta técnica, dado su buen rendimiento tanto en imágenes genéricas como en imágenes compuestas por bandas. Sin embargo, una de las variaciones que se proponen para una mejor detección de bandas es hibridizar el funcionamiento de los algoritmos ACO a través de la introducción del procesamiento de la imagen a través de la lógica difusa; la que proveerá información utilizable en la ejecución del algoritmo de hormigas. La solución obtenida a través de este sistema será una imagen en escala de grises, que presentará las bandas detectadas y su ubicación, interpretándose este resultado como una probabilidad de que lo detectado sea una banda, dejando en manos del experto la decisión final.

La base de la elección de esta solución consiste en dos iniciativas que ocupan algoritmos ACO en imágenes genéricas, con buenos resultados; los cuales se quieren probar en el dominio de las imágenes DGGE. Los detalles y elementos que proveen estas investigaciones se presentan a continuación [18, 52].

4.2.1. *An ant-inspired algorithm for detection of image edge features*

Este trabajo de S. Ali Etemad y Tony White *An Ant-inspired algorithm for detection of image edge features* utiliza una técnica basada en algoritmos de colonias de hormigas para procesar imágenes. El algoritmo se utiliza para mejorar la extracción de características de la imagen, principalmente la detección de bordes, incluso en presencia de ruido [18].

El algoritmo diseñado tiene por nombre *Ant-based Correlation for Edge Detection* (ACED) y está basado en el algoritmo Ant Colony System (ACS). En ACED las hormigas recorren el ambiente, representado por la imagen a escala de grises, y depositan feromonas en los pixeles de la imagen para marcar el camino (los bordes de la imagen). Antes de la acción de las hormigas, la imagen a tratar es transformada a una imagen en escala de grises cuando corresponda y luego se calcula el gradiente de la imagen, el cual será utilizado como la matriz de información heurística. También se recibe como parámetro el número de hormigas que compone la población y se define su ubicación en la matriz, siendo la más utilizada la distribución aleatoria de las hormigas. Otros parámetros de este algoritmo son el coeficiente de evaporación ρ , el máximo de iteraciones del algoritmo, denotado por *max_iterations*, un valor de deposición de feromonas después que una hormiga cruzó un arco, denotado por λ y un umbral T utilizado en el postprocesamiento para binarizar la imagen.

En este algoritmo hay dos tipos de feromonas que depositan las hormigas, la *feromona tipo-I*, denotada por τ , la cual es la feromona usual de los algoritmos ACO, la cual es depositada y se evapora al término de cada iteración, de acuerdo al coeficiente de evaporación ρ y la *feromona tipo-II*, llamada también *estímulo*, denotada por η , la cual junto a τ son responsables en la decisión del próximo lugar en que una hormiga se ubique en la próxima iteración y no transite en un loop en todo el proceso. Estos dos tipos de feromonas son independientes y no se combinan, ya que si se permitiera la combinación de ambas feromonas, las hormigas quedarían atrapadas en pequeños clusters de feromona. Además la feromona no se distribuye uniformemente, sino que se acumula en unos pocos pixeles.

Antes de entrar a la ejecución iterativa del algoritmo, se debe calcular el estímulo global S de la imagen, que es la matriz de feromonas tipo-II, lo cual se muestra en la ecuación (4.1)

$$S = \|\nabla IM\| = \sqrt{\left(\frac{\delta IM}{\delta x}\right)^2 + \left(\frac{\delta IM}{\delta y}\right)^2} \quad (4.1)$$

Cálculo de Estímulo Global

donde IM es la imagen en escala de grises que será tratada con ACED y ∇IM es el gradiente de la imagen.

Durante el comportamiento iterativo del algoritmo, el cual se aplica hasta llegar al máximo de iteraciones, cada hormiga se moverá de acuerdo al estímulo y feromona local, calculado según la ecuación (4.2), el cual considera los estímulos de sus vecinos, en base a la posición de éstos en la matriz de la ecuación 4.1 (de acuerdo a la vecindad Von Neumann¹), excepto de la posición anterior (i', j') en la que se encontraba la hormiga antes de cada iteración, lo que se expresa a través de la función $\sigma(i, j, i', j')$. La posición previa de cada hormiga es guardada en una memoria μ , la cual es actualizada en cada iteración.

$$S_{ij} = \sigma(i, j, i', j') = S((vecinos(i, j)) \wedge \neg(i', j')) \quad (4.2)$$

Cálculo de Estímulo Local

Luego, cada hormiga se mueve de acuerdo a la ecuación (4.3), utilizando los valores de feromona τ_{ij} y estímulo local (Ecuación 4.2) como η_{ij} la cual indica las probabilidades de cada vecino, manteniendo la restricción de no moverse directamente hacia la posición anterior en que estaba. La hormiga se mueve al pixel vecino que tiene la mayor probabilidad según la ecuación (4.3) en base a las reglas indicadas en el ACS y en ese punto deposita feromona (tipo-I).

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}]^\alpha [\eta_{il}]^\beta}, \text{ si } j \in N_i^k, \quad (4.3)$$

Probabilidad de desplazamiento de las hormigas

Después que todas las hormigas ya avanzaron hacia una nueva posición, se actualiza la matriz de feromonas (tipo-I), denotada por Λ , de acuerdo a la ecuación (4.4), en donde λ indica una deposición de feromona fresca al terminar el paso de las hormigas. El valor de λ es una constante típicamente ajustada a 1, de acuerdo a [18].

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \lambda \quad (4.4)$$

Actualización de Feromona

Al terminar la ejecución del algoritmo, los caminos trazados por las hormigas en la matriz de feromonas (tipo-I) Λ son los bordes de la imagen IM . A esta matriz se le aplica un postprocesamiento, el cual consiste en determinar un umbral T , elegido manualmente, para incluir otros pixeles al borde de la imagen y desechar aquellos que cumplan una serie de condiciones, de acuerdo a la ecuación (2.2).

¹En la vecindad de Von Neumann se consideran 4 vecinos. Para una celda (i, j) , sus vecinos Von Neumann son $(i - 1, j)$, $(i, j - 1)$, $(i, j + 1)$, $(i + 1, j)$

El algoritmo ACED se resume de la siguiente manera:

Algoritmo 2: Proceso de Extracción de Características

```

1 Convertir imagen a escala de grises, guardada como IM
2 Indicar los valores de  $\rho$ ,  $\lambda$  y el umbral  $T$ 
3 Determinar el número de hormigas y la posición inicial de éstas
4 Indicar número máximo de iteraciones  $max\_iterations$ 
5 Calcular gradiente de la imagen IM
6 Calcular estímulo de la matriz, de acuerdo a la ecuación (4.1)
7 while  $iteracion < max\_iterations$  do
8   for cada hormiga de la población do
9     Calcular estímulo local de acuerdo a la ecuación (4.2)
10    Moverse de acuerdo a la ecuación (4.3)
11    Depositar Feromona Tipo-I
12  Actualizar  $\Lambda$  de acuerdo a ecuación (4.4)
13   $iteracion = iteracion + 1$ 

```

Los resultados del algoritmo dependen de los parámetros escogidos, en donde es importante la distribución inicial de las hormigas, en la cual se propone ubicar al menos a una hormiga en cada ventana (una submatriz de 3x3 por ejemplo) para mejorar los resultados. En la entrega de resultados, se compara con las técnicas usuales de detección de bordes (Roberts, Sobel, Prewitt y Canny) en la que ACED entrega resultados balanceados, es decir, extrae las características principales de la imagen, no pocas (como Prewitt, Roberts y Sobel) ni demasiadas como Canny. Al estar en presencia de ruido, el algoritmo presenta una buena tolerancia a este elemento, pues no le afecta en gran manera, ya que la imagen se distingue a pesar de la presencia de ruido.

4.2.2. A Hybrid Approach to Edge Detection using Ant Colony Optimization and Fuzzy Logic

Este trabajo realizado por Y. Tyagi, T. A. Puntambekar, Preeti Sexena y Sanjay Tanwani presenta una técnica híbrida entre algoritmos ACO y lógica difusa; utilizando esta última como información heurística para el movimiento de las hormigas, y haciendo que la influencia de la feromona y la información heurística no sea fija, sino que se actualice dinámicamente [52].

La técnica propuesta en este trabajo presenta el concepto de **edgeness**, definido como el grado en que un pixel (i, j) pertenece a un borde cuando solamente la variación de la intensidad en su vecindad de Moore ² es tomada en cuenta. Primeramente, el edgeness de un pixel se calcula como el promedio de los pixeles de la ventana de 3x3 en la que está el pixel, tal como se muestra en la ecuación 4.5, con $f(x, y)$ el valor de la intensidad del pixel (i, j) y N el número de pixeles que están en una fila o columna de una ventana de 3x3.

$$Media = \frac{\sum f(i, j)}{N} \quad (4.5)$$

El primer paso de esta técnica es calcular los inputs necesarios para el método Mamdani de lógica difusa, denotados por $Mrow$, $Mcol$ e $Idiag$, los primeros dos calculados en base a la media de las filas y columnas, respectivamente, de la ventana de 3x3 e $Idiag$ es resultado del valor absoluto de la suma de la diferencia de cada vecino con el pixel objetivo, como se observa en las siguientes ecuaciones.

²Vecindad consistente en los 8 vecinos de un pixel

$$Mrow = Media_{1f} + Media_{3f} - 2 * Media_{2f} \quad (4.6)$$

$$Mcol = Media_{1c} + Media_{3c} - 2 * Media_{2c} \quad (4.7)$$

$$Idiag = |I_{i-1,j-1} - I_{i,j} + I_{i+1,j+1} - I_{i,j} + I_{i,j-1} - I_{i,j} + I_{i,j+1} - I_{i,j} + I_{i-1,j+1} - I_{i,j} \\ + I_{i+1,j-1} - I_{i,j} + I_{i-1,j+1} - I_{i,j} + I_{i-1,j} - I_{i,j} + I_{i+1,j} - I_{i,j}| \quad (4.8)$$

siendo $Media_{Nf}$ la media de la intensidad de los pixeles de la fila N y $Media_{Nc}$ la media de la intensidad de los pixeles de la columna N por cada ventana de 3×3 .

Cada uno de estos inputs se divide en tres clases (Alto: H, Medio: M y Bajo: L), los cuales son utilizados junto a una función de pertenencia gaussiana en el proceso de inferencia difusa utilizando el modelo de Mamdani con el método min-max (descrito en la sección 3.2.3.2 para obtener como resultado las clases del *Edgeness* en base a las 6 reglas difusas presentadas a continuación.

Regla 1: SI ($Mrow \in MrowH$) AND ($Idiag \in IdiagH$) AND ($Mcol \in McolH$) ENTONCES ($salida \in EdgenessH$)
Regla 2: SI ($Mrow \in MrowL$) AND ($Idiag \in IdiagH$) AND ($Mcol \in McolM$) ENTONCES ($salida \in EdgenessH$)
Regla 3: SI ($Mrow \in MrowM$) AND ($Idiag \in IdiagL$) AND ($Mcol \in McolL$) ENTONCES ($salida \in EdgenessL$)
Regla 4: SI ($Mrow \in MrowH$) AND ($Idiag \in IdiagL$) AND ($Mcol \in McolL$) ENTONCES ($salida \in EdgenessL$)
Regla 5: SI ($Mrow \in MrowL$) AND ($Idiag \in IdiagL$) AND ($Mcol \in McolL$) ENTONCES ($salida \in EdgenessL$)
Regla 6: SI ($Mrow \in MrowM$) AND ($Idiag \in IdiagM$) AND ($Mcol \in McolM$) ENTONCES ($salida \in EdgenessM$)

Figura 4.4: Reglas Difusas para Calcular Edgeness

Tras obtener las tres clases del edgeness, se les aplica el método de defusificación para obtener la matriz de edgeness. Esta matriz está normalizada en el rango $[0,1]$ y se ocupa en el algoritmo ACO como información heurística η .

El algoritmo ACO utilizado es el ACS, inicializado con K hormigas, definido como $K = \sqrt{M * N}$, siendo M el número de filas de la imagen y N el número de columnas y dispuestas de forma aleatoria en la imagen, cuidando que haya solo una hormiga en cada pixel por cada ciclo C , una hormiga recorre S pixeles. De acuerdo a la ecuación 4.3, los valores de α y β no son constantes, sino que actualizados en base a η_{ij} . Si la diferencia entre el máximo y el mínimo valor de η_{ij} en la vecindad de Moore es mayor que 0.00003, el valor de α decrece en un factor de 0.0001, mientras que el valor de β aumenta en el mismo factor cuando ocurre la condición inversa. El propósito de la actualización de exponentes de la ecuación 4.3 es controlar la influencia de la feromona y la información heurística.

La matriz de feromonas es actualizada de dos formas, de manera local y global. La actualización local diversifica la búsqueda haciendo que los bordes ya visitados sean menos atractivos para una nueva deposición de feromona, por lo que indirectamente se favorece la exploración de bordes no visitados y no se converge en un camino común para las hormigas, como se modela a través de la ecuación 4.9:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \rho\Delta\tau_{ij} \quad (4.9)$$

Actualización Local de Feromona

donde ρ es la tasa de evaporación de feromonas, τ_{ij} el valor actual de la matriz de feromona en la posición (i, j) y $\Delta\tau_{ij} = \eta_{ij}$, el valor en que la feromona se incrementa en cada iteración.

La actualización global se realiza después que todas las hormigas hayan realizado su movimiento de un pixel a otro por cada ciclo, de acuerdo a la ecuación 4.10.

$$\tau_{ij} = (1 - \psi)\tau_{ij} + \psi\tau_0 \quad (4.10)$$

Actualización Global de Feromona

con ψ el coeficiente de decaimiento de feromona y τ_0 el valor inicial de feromona.

Finalmente, la decisión de la pertenencia de un pixel al borde de la imagen se realiza a través de la aplicación de un umbral en la matriz de feromona resultante del algoritmo.

El algoritmo propuesto por Tyagi se resume de la siguiente manera:

Algoritmo 3: Método Híbrido de Detección de Bordes

- 1 Convertir imagen a escala de grises y normalizarla, guardada como IN
 - 2 Indicar los valores de ρ , ψ y el umbral T
 - 3 Indicar número de ciclos C y pasos S
 - 4 Segmentar la imagen en ventanas de 3×3
 - 5 **for cada** ventana de 3×3 **do**
 - 6 | Calcular M_{row} , M_{col} e I_{diag}
 - 7 Aplicar las reglas difusas mostradas en la figura 4.4 con los valores calculados como input
 - 8 Defusificar el output entregado por el sistema difuso
 - 9 Determinar el número de hormigas $K = \sqrt{M * N}$
 - 10 **while** ciclo $< C$ **do**
 - 11 | **while** pasos $< S$ **do**
 - 12 | **for cada** hormiga de la población **do**
 - 13 | Moverse de acuerdo a la ecuación (4.3)
 - 14 | Actualizar valores de α y β de acuerdo a la vecindad de η_{ij}
 - 15 | Actualizar feromona localmente de acuerdo a ecuación (4.9)
 - 16 | paso = paso + 1
 - 17 | Actualizar feromona globalmente de acuerdo a ecuación (4.10)
 - 18 | ciclo = ciclo + 1
 - 19 Aplicar umbral T a la matriz de feromonas resultante
-

4.3. Diseño de la Solución

La construcción del sistema solución se hará en base a los trabajos descritos anteriormente; tomando aspectos de Etemad [18] y Tyagi [52]; es decir, el sistema será un híbrido entre algoritmos ACO y lógica difusa, introduciendo nuevos elementos que participarán en la detección de bordes. Es por esto que el sistema de software constará de tres módulos: la obtención de la matriz de información heurística η usando la lógica difusa, la obtención de los bordes de la imagen con algoritmos ACO y la identificación de los bordes detectados buscando los máximos del histograma de la imagen. Aunque el objetivo del sistema es entregar una buena detección de bordes en imágenes DGGE, también puede recibir imágenes genéricas.

El sistema recibirá como parámetro único una *imagen*, la cual puede ser a escala de grises o a color, para las cuales se transformarán internamente en imágenes a blanco y negro, para ser utilizadas en el procesamiento posterior.

El módulo de **obtención de la matriz de información heurística** comienza por normalizar³ la imagen para que se apliquen los cálculos para obtener los inputs necesarios para el sistema de lógica difusa basado en el trabajo de Tyagi [52]. Luego, tras el procesamiento por lógica difusa, la aplicación de reglas difusas y la defusificación; se obtiene la matriz de información heurística η que será utilizada en el siguiente módulo.

El segundo módulo, de **obtención de la imagen de bordes**, está construido en base al algoritmo Ant Colony System propuesto por Dorigo y Gambardella en [13] y ocupado en los trabajos de Tyagi y Etemad, siendo éste último en donde se le introducen modificaciones para observar su rendimiento en imágenes genéricas [18]. Algunas de las modificaciones propuestas por Etemad utilizadas en este módulo son la utilización de la vecindad Von Neumann para los píxeles y la existencia de dos tipos de feromonas. Los parámetros utilizados para el funcionamiento del ACS modificado fueron determinados a través de pruebas con distintas imágenes de control, las cuales se muestran en el siguiente capítulo.

La utilización de los dos tipos de feromonas propuesta por Etemad será explorada con variaciones en el sistema propuesto. Etemad usa como feromona tradicional una inicialización de la matriz de feromonas a partir de una constante y como la matriz de información heurística introduce un nuevo tipo de feromona conocida como *estímulo* y es calculada a partir de la norma del gradiente de la imagen. Tal como se ha dicho anteriormente, la matriz de *edgeness*, calculada a partir del módulo de lógica difusa, ocupará el lugar del estímulo en el algoritmo propuesto por Etemad y como matriz de feromona se ocupará el **cuociente entre el gradiente y la segunda derivada de la imagen**. El gradiente de una imagen indica la dirección del cambio de intensidad, se calcula como la suma de las derivadas en ambos ejes, como lo indica la ecuación 4.11 y la segunda derivada de una imagen, conocida como *laplaciana*, se utiliza para destacar las regiones en donde exista un cambio brusco de intensidad. Se utiliza una forma distinta al laplaciano (mostrado en la ecuación 4.12), considerando la suma de las derivadas parciales y cruzadas en este sistema (Ecuación 4.13).

$$\hat{\nabla} f = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} \quad (4.11)$$

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad (4.12)$$

$$\hat{\nabla}^2 f = \frac{\partial^2 f}{\partial x^2} + 2 * \frac{\partial^2 f}{\partial xy} + \frac{\partial^2 f}{\partial y^2} \quad (4.13)$$

³Convertir la matriz en valores entre [0,1]

por lo cual, la matriz de feromonas Λ se inicializa de acuerdo a la ecuación 4.14:

$$\Lambda = \frac{\hat{\nabla}}{\hat{\nabla}^2} = \frac{\frac{\partial f}{\partial x} + \frac{\partial f}{\partial y}}{\frac{\partial^2 f}{\partial x^2} + 2 * \frac{\partial^2 f}{\partial xy} + \frac{\partial^2 f}{\partial y^2}} \quad (4.14)$$

Inicialización de la matriz de feromonas propuesta

Para elegir el movimiento de cada hormiga se utiliza la ecuación 4.3 para calcular las probabilidades y se escoge el valor resultante con el algoritmo de selección por ruleta, utilizado ampliamente con técnicas que utilizan algoritmos genéticos. Esta técnica consiste en asignar un segmento de una ruleta en base a la aptitud de la población, en el caso de las hormigas la aptitud es la probabilidad de cada individuo de moverse a una celda vecina, y girar la ruleta una vez para escoger donde moverse en la siguiente iteración. Esto se describe en el algoritmo 4 [8].

Algoritmo 4: Algoritmo de Selección por Ruleta

- 1 Sumar probabilidades de los vecinos del pixel actual (*sum*)
 - 2 Generar un número aleatorio entre 0 y 1 (*rand*)
 - 3 Inicializar variable acumuladora en 0 (*acum*)
 - 4 **while** *acum* < *rand* **do**
 - 5 **por cada vecino del pixel**
 - 6 *sumar la probabilidad del vecino a la variable acumuladora*
 - 7 Obtener valor (*x,y*) de la celda acorde a la probabilidad seleccionada
-

La actualización de feromonas se realiza de dos formas. La primera es la usual del algoritmo ACS, que se puede ver en la ecuación 3.10 se realiza al terminar el paso de todas las hormigas por cada iteración. La segunda actualización consiste en depositar una pequeña cantidad de feromona como se observa en la ecuación 4.4.

Tras la obtención de la imagen con los bordes detectados a través del algoritmo ACO, el módulo de **ubicación de bandas** calcula el *histograma de promedio de intensidades* por cada fila de la imagen, el cual se utiliza para determinar automáticamente el umbral T que se aplicará a la imagen y con la imagen umbralizada se buscan los máximos promedios de intensidad, los cuales se marcarán con una línea en la imagen indicando la posición de un límite de cada banda encontrada, además de guardar las posiciones en un archivo de texto. Cada par de líneas entregadas en una distancia cercana (menos de 10 píxeles) significa que en la región delimitada entre las líneas existiría una banda. La información entregada debe ser interpretada como una probabilidad que en el lugar indicado hay una banda DGGE, dejando la decisión al experto para su utilización posterior.

En resumen, el algoritmo propuesto para la detección de bandas en imágenes DGGE consiste en los siguientes pasos:

Algoritmo 5: Método Propuesto de Detección de Bordes en Imágenes DGGE

- 1 Convertir imagen a escala de grises , denotada como IBN
 - 2 Normalizar IBN para su utilización en el sistema difuso, denotada como $INorm$
 - 3 Recorrer $INorm$ a través de ventanas de 3×3
 - 4 **for cada** ventana de 3×3 **do**
 - 5 | Calcular $Mrow$, $Mcol$ e $Idiag$
 - 6 Aplicar las reglas difusas mostradas en el cuadro 4.4 con los valores calculados como input
 - 7 Defusificar el output entregado por el sistema difuso, denotado por η
 - 8 Indicar los valores de ρ , λ , número de hormigas y de iteraciones max_iter
 - 9 Calcular gradiente y segunda derivada en base a IBN
 - 10 Inicializar Λ de acuerdo a ecuación 4.14
 - 11 **while** iteraciones < max_iter **do**
 - 12 | **for cada** hormiga de la población **do**
 - 13 | Moverse de acuerdo a la ecuación (4.3)
 - 14 | Depositar Feromona
 - 15 | Actualizar feromona de acuerdo a ecuación (3.10)
 - 16 | Evaporación de la feromona de acuerdo a ecuación (4.4)
 - 17 | iteraciones = iteraciones + 1
 - 18 Calcular histograma de promedios h en base a Λ
 - 19 Calcular umbral T en base a h y aplicarlo a Λ
 - 20 En la imagen umbralizada, calcular histograma de promedios y buscar los máximos
 - 21 De acuerdo a los máximos, marcarlos con una línea para identificar las bandas
-

Capítulo 5

Resultados en Imágenes Genéricas

En este capítulo se muestran las pruebas realizadas utilizando el algoritmo 5 propuesto en el capítulo anterior con las imágenes de muestra en el mismo trabajo, las cuales son imágenes de figuras geométricas y *Lena*, y realizar mediciones y comparaciones con técnicas tradicionales, el algoritmo de Etemad [18] y la aplicación de métodos de forma separada.

5.1. Imágenes y Parámetros

Antes de realizar las pruebas con las imágenes DGGE (Capítulo 6), se utilizan las siguientes imágenes para probar el sistema de detección en ellas. También se les aplicará un filtro de suavizado, para simular el efecto del blurring en el rendimiento del sistema.

Imágenes Benchmark

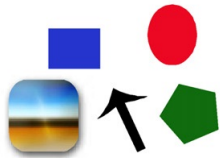


Figura 5.1: Figuras Geométricas

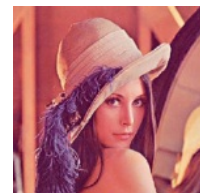


Figura 5.2: Lena

Imagen de Evaluación

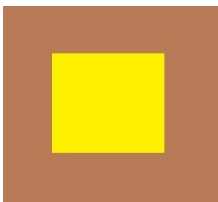


Figura 5.3: Cuadrado

La figura 5.3 será utilizada en las pruebas del algoritmo, la cual servirá para evaluar el rendimiento del algoritmo, a través del *terreno de la verdad*, para determinar los puntos correctos y fallidos de detección en base a una imagen que muestra los bordes del cuadrado interno.

5.1.1. Parámetros utilizados

Entre los dos trabajos en que se basa el algoritmo propuesto, se utiliza la versión del ACS ocupada en el trabajo de Etemad en [18], se aplicaron variaciones en cuatro parámetros para evaluar su comportamiento e incidencia en la detección de bordes sobre las imágenes escogidas y elegir los valores que produzcan mejores resultados. Estos parámetros son:

1. *Población*: Expresado como un valor porcentual en relación al tamaño de la imagen (en píxeles), corresponde a la cantidad de hormigas que se encuentran trabajando en paralelo en la detección de bordes.
2. *Número de Iteraciones*: Indica la cantidad de veces que las hormigas tendrán para realizar su camino en la imagen. Se supone que si el número de iteraciones es alto, aumentan las posibilidades que las hormigas recorran toda la imagen.
3. *Probabilidad de Movimiento*: en la ecuación (3.1), que modela la probabilidad que una hormiga se mueva de un punto a otro, se evaluarán los parámetros α y β , que indican la influencia del rastro de feromona, ya sea la usual de los algoritmos ACO (τ_{ij}) o el estímulo introducido en el trabajo de Etemad, que en este trabajo será definido por la matriz obtenida a partir de la lógica difusa (η_{ij}).
4. *Constante de Evaporación*: denotada por ρ , este valor determina la velocidad de evaporación de la feromona en cada iteración. Generalmente se ocupa $\rho = 0,1$, tal como se utiliza en [18] [52].
5. *Deposición de Feromona*: denotado por λ , indica la cantidad de feromona que las hormigas depositan después de cada iteración, formalizado en la ecuación (4.4) de actualización de feromona. Este valor es dependiente en el éxito que tiene la hormiga en solucionar el problema en consideración. De acuerdo a Etemad [18], su valor típicamente se fija en 1.

A partir de pruebas realizadas con este conjunto de variables, en donde se ejecutó el algoritmo combinando distintos valores de los parámetros indicados; la mejor combinación de valores para los parámetros se muestran en la tabla 5.1.

	Valor Seleccionado
Población	50 %
Número de Iteraciones	1000
Constante de Evaporación ρ	0.1
Deposición de Feromona λ	0.5
Influencia de Feromona α	0.5
Influencia de Estímulo β	2

Tabla 5.1: Parámetros escogidos para el algoritmo propuesto

Para obtener estos valores, se tuvieron como base los trabajos de Etemad [18], Tyagi [52] y el trabajo de memoria del autor [55], en donde la metodología de pruebas consistió en variar cada parámetro mientras los otros se mantienen constantes en un valor. Caso especial fue la determinación de los exponentes α y β , que indican la probabilidad del movimiento de las hormigas, los cuales se variaron, ambos, en distintas combinaciones, mientras los demás parámetros se mantuvieron constantes. Tras estas pruebas, se obtuvieron los parámetros mostrados en la tabla 5.1.

5.2. Resultados en Imágenes de Prueba

Con los valores establecidos en la tabla 5.1 se probó el algoritmo en las imágenes mostradas en la sección 5.1, añadiendo la aplicación de suavizado para simular el blurring en las imágenes. Sólo a la imagen original se le aplicó un postprocesamiento, el cual consiste en umbralizar la imagen. A las imágenes suavizadas no se les aplicó ningún tipo de postprocesamiento en este capítulo.

5.2.1. Pruebas con *figuras*

5.2.1.1. Imagen Original

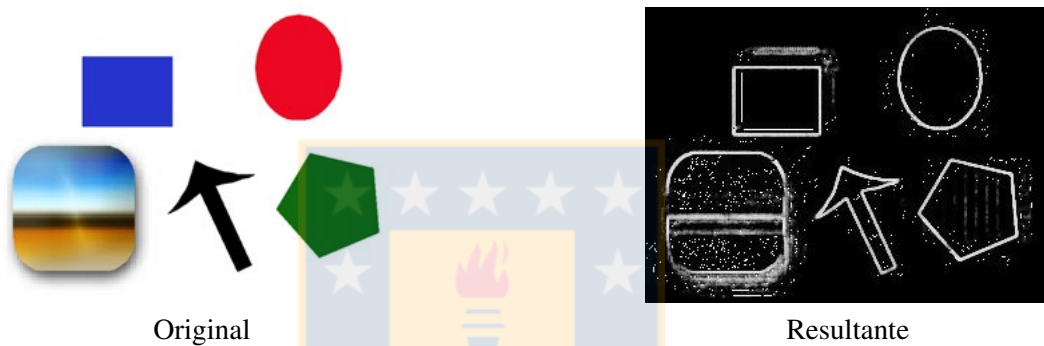


Figura 5.4: Aplicación del algoritmo híbrido con los parámetros establecidos a *figuras*

Se observa que la imagen resultante presenta los bordes de las figuras, sin embargo, presenta un poco de ruido, por lo cual se le aplicará *segmentación por intensidad*, en la cual se intensifican los píxeles que sobrepasan un umbral, determinado manualmente (de acuerdo a la ecuación 2.2). Los resultados de este postprocesamiento se observan en la figura 5.5

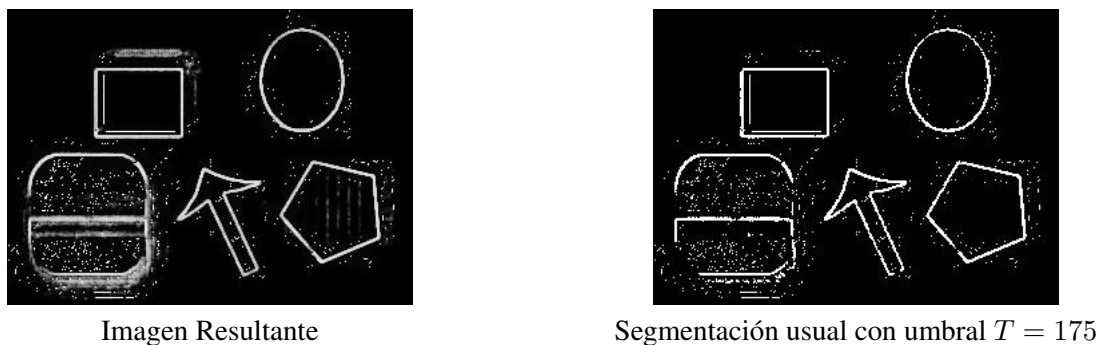


Figura 5.5: Segmentación a *figuras*

5.2.1.2. Simulación de Blurring

Para que la imagen fuese más difusa, se procedió a suavizarla. Para esto se probaron dos tipos de filtros para suavizar la imagen, el primero es utilizando el *filtro gaussiano* y el otro es ocupando el *filtro de mediana*. Las imágenes suavizadas se observan en la figura 5.6.

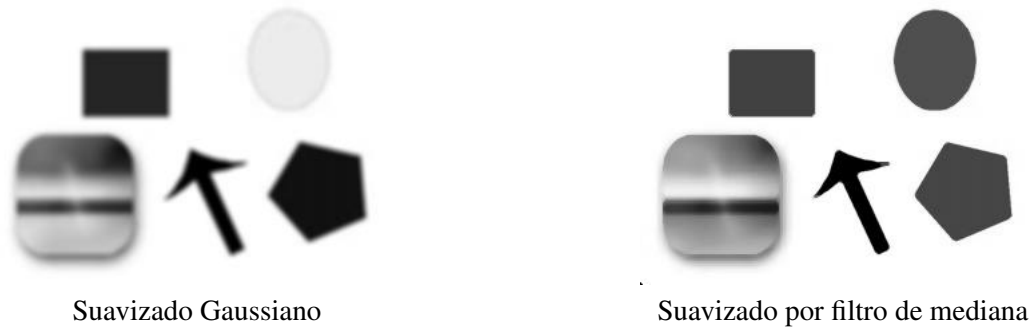


Figura 5.6: Aplicación de suavizado a *figuras*

En las imágenes resultantes, mostradas en la figura 5.7, se observa que la utilización de filtros de distinto tipo, para hacer que la imagen sea borrosa, en general realza los bordes en esta imagen. Un filtro lineal, como el aplicado en el suavizado gaussiano produce que los bordes se engrosen, mientras que un filtro no lineal, como el aplicado en el suavizado por mediana produce que los bordes detectados se realcen sin hacer que el grosor de ellos crezca mucho.

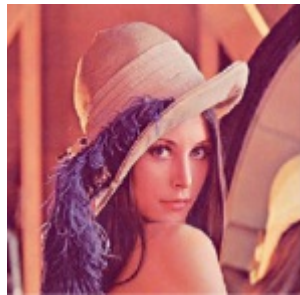


Figura 5.7: Resultantes de *figuras* suavizada

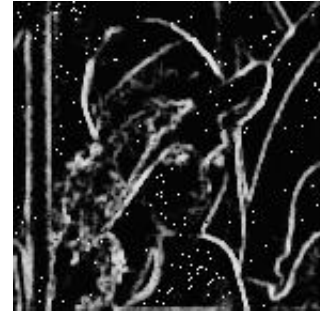
En general se observa que al suavizar la imagen antes de la aplicación del algoritmo no ocurren grandes variaciones, ya que se está aplicando sobre una figura simple. Los métodos de suavizado utilizados difieren en el grosor de los bordes detectados, siendo el método de suavizado por filtro de mediana el que genera bordes más delgados que el suavizado gaussiano.

5.2.2. Pruebas con *Lena*

5.2.2.1. Imagen Original



Original



Resultante

Figura 5.8: Aplicación del algoritmo híbrido con los parámetros ya establecidos a *Lena*

Con *Lena*, se observa que se generó una buena detección, pero no se observa una gran cantidad de detalles, de los que componen la imagen. La cantidad de ruido que se generó tras la detección, al igual que en *figuras*, es mínima, solo hay unos pocos puntos que no pertenecen a bordes. En este caso, la segmentación de la imagen (Figura 5.9), aplicada después de aplicar el algoritmo, solo elimina los píxeles con menor intensidad, manteniendo en gran parte los bordes de la figura.

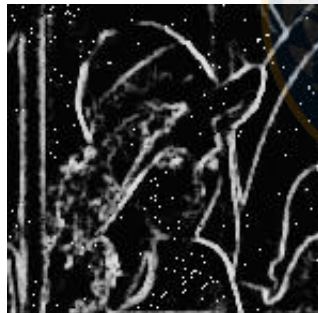


Imagen Resultante

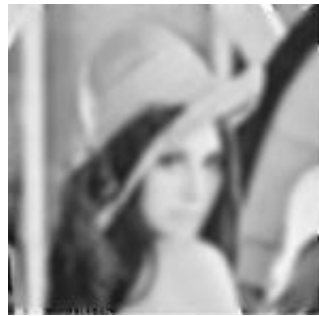


Segmentación usual con umbral

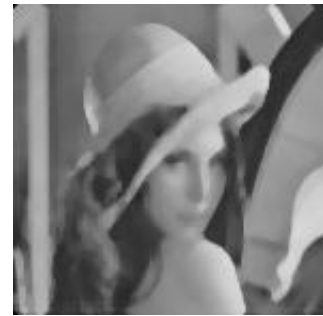
Figura 5.9: Segmentación a *Lena* con $T = 120$

5.2.2.2. Simulación de Blurring

Al igual que en la imagen de *figuras*, para hacer que *Lena* se vea más difusa, se suaviza la imagen original, utilizando el filtro gaussiano y el de mediana, provocando el primer filtro una imagen más difusa que el filtro de mediana. Las imágenes que resultan tras este preprocesamiento se muestran en la figura 5.10.



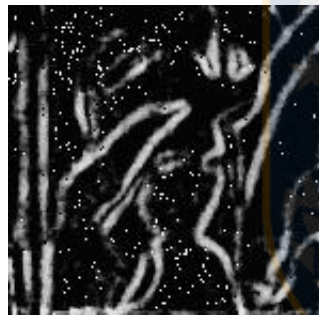
Suavizado Gaussiano



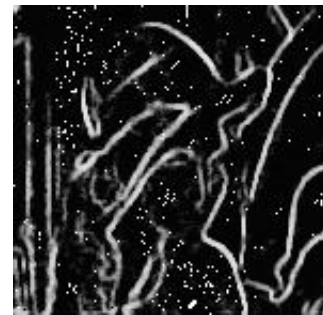
Suavizado por filtro de mediana

Figura 5.10: Aplicación de suavizado a *Lena*

Ya que el suavizado hace que la imagen sea más difusa, se pierden algunos detalles, sin embargo, la detección de bordes es buena, ya que se mantiene la imagen principal, sin importar el nivel de difuminación, siendo que la imagen contiene muchos más detalles que las figuras geométricas. Los resultados se aprecian en la figura 5.11.



Suavizado Gaussiano



Suavizado con filtro de mediana

Figura 5.11: Resultantes de *Lena* suavizada

En este caso, al ser una imagen más compleja que *figuras*, el suavizado disminuyó la cantidad de detalles capturados por la detección de bordes con el algoritmo híbrido en la imagen sin preprocesar, sin embargo, se mantiene el contorno global de la imagen. Comparando los métodos de suavizado, se observa que el suavizado con filtro de mediana obtiene una detección con bordes continuos y más finos que el suavizado gaussiano, dado que el filtro gaussiano difumina más la imagen que el filtro de mediana.

5.2.3. Comparación con técnicas tradicionales

En términos generales, el algoritmo propuesto realiza una buena detección de bordes en las imágenes presentadas en este capítulo. En cuanto a los resultados del algoritmo en las imágenes simples como *figuras* se observa que se marcan los bordes de forma continua, presentando ruido en forma dispersa a través de puntos. Comparando este resultado con las técnicas tradicionales (presentadas en la sección 2.3.4) se observa que la resultante tras el algoritmo propuesto presenta una detección mucho mayor que ellas. Salvo el método de Canny (Figura 5.12f), los otros métodos tradicionales no detectan la forma compleja de la imagen original (Figura 5.12a). Además, al observar la calidad de los bordes detectados, con el algoritmo propuesto éstos son continuos y su grosor es uniforme, en contraposición a las imágenes que resultan tras aplicar técnicas tradicionales, en las

cuales se observa que los bordes aparecen discontinuados, al haber diferencias en el grosor de los bordes.

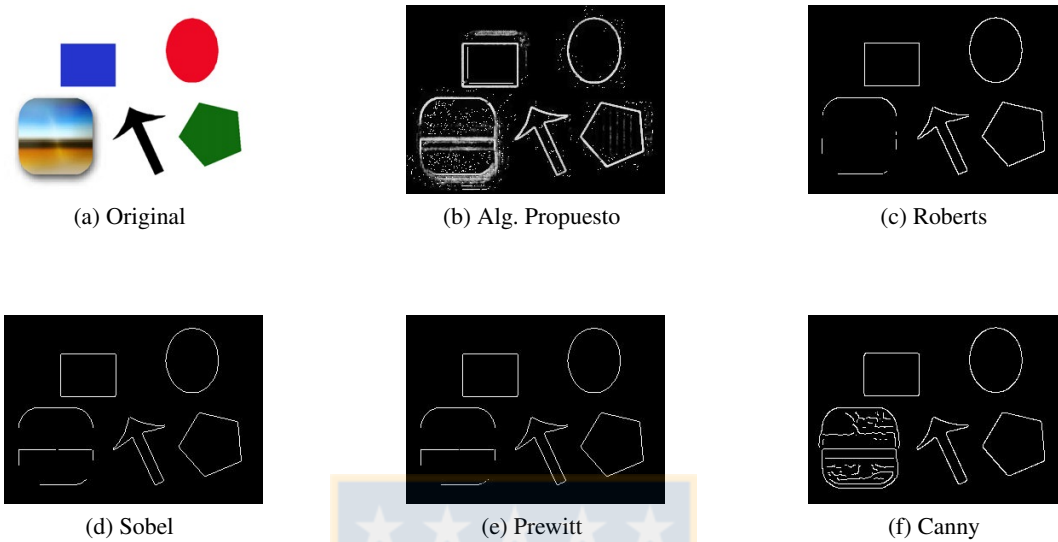


Figura 5.12: Métodos de detección de bordes en *figuras*

Considerando una imagen más compleja que *figuras*, en *Lena* se observa que el algoritmo propuesto presenta una detección con más detalles que los métodos tradicionales. Aunque el algoritmo de Canny (Figura 5.13f) detecta la totalidad de la figura, el algoritmo propuesto lo realiza de manera más marcada en los detalles principales, lo que permite que la imagen de bordes sea muy similar a la imagen original.

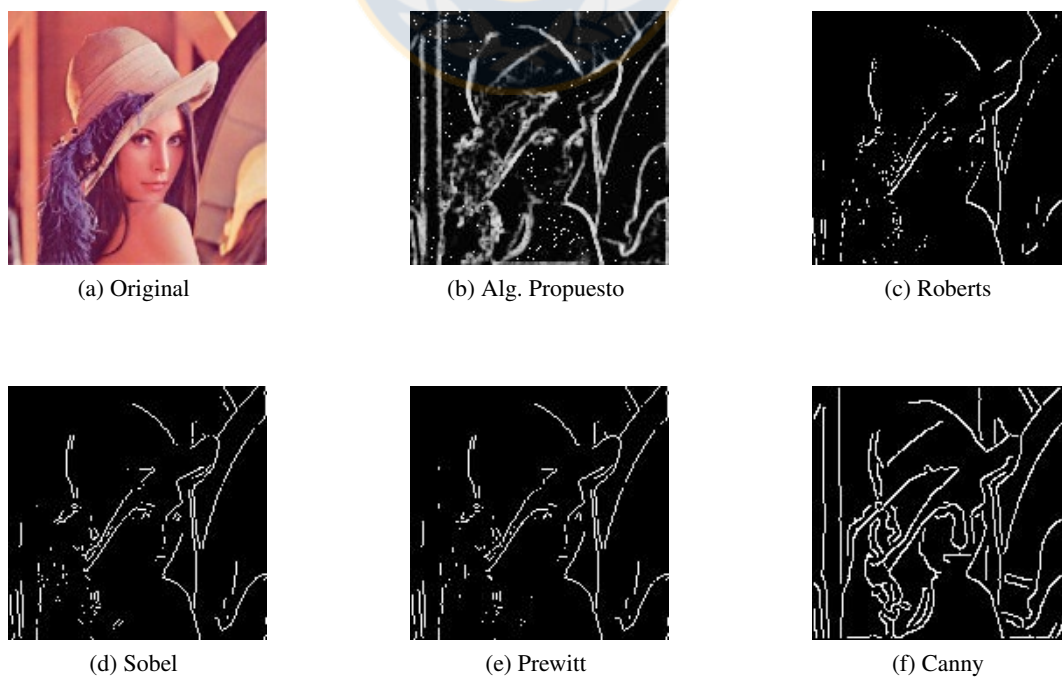


Figura 5.13: Métodos de detección de bordes en *Lena*

5.3. Análisis de Resultados

En esta sección, se evaluará el rendimiento del algoritmo teniendo como base la detección de bordes en una imagen simple, para lo cual se medirán los píxeles bien detectados y los errores que tuvo la técnica. También se compararán los resultados obtenidos en las imágenes mostradas con las técnicas tradicionales y con algoritmos similares.

5.3.1. Evaluación de los Resultados

Para medir la efectividad del algoritmo se utilizarán los conceptos de *pruebas de significación* para determinar la medida de performance, la cual se muestra en la ecuación (5.1)

$$Performance = \frac{TP}{TP + FP + FN} \quad (5.1)$$

donde:

TP: verdaderos positivos, es la cantidad de píxeles de bordes detectados correctamente

FP: falsos positivos, píxeles detectados como bordes que no lo son

FN: falsos negativos, píxeles de borde que no fueron detectados.

Otra medida utilizada es el *Score*, que mide la cantidad de píxeles reconocidos correctamente, es decir, consiste en la suma de los verdaderos positivos y negativos de la imagen [18]:

$$Score = TP + TN \quad (5.2)$$

donde se añade:

TN: verdaderos negativos, es la cantidad de píxeles que no son de bordes detectados correctamente.

Estas medidas de evaluación se utilizarán en la imagen *cuadrado* (Figura 5.14a), a la cual se le aplicará el algoritmo de hormigas en distintas condiciones y se utilizará como comparación la Figura 5.14b, la cual contiene **1054 píxeles**, los cuales se consideran como el total de los aciertos en la detección de bordes y **49378 píxeles** que no son de borde.

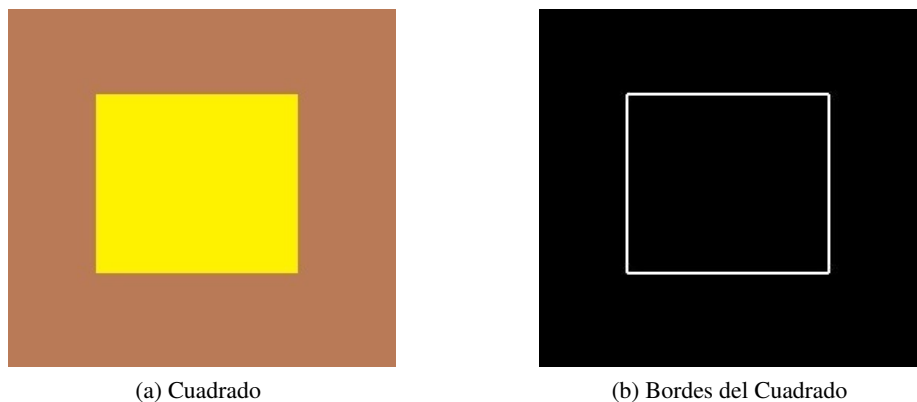


Figura 5.14: Imagen de Control del Cuadrado para evaluación

También se les aplicará la segmentación usual (Ecuación 2.2), con un umbral definido manualmente, a las imágenes para evaluar su rendimiento con este postprocesamiento.

En la tabla 5.2 se muestran los resultados obtenidos al aplicar el algoritmo a la imagen del cuadrado, en la cual se observa que se detectan todos los píxeles de borde de la imagen, sin embargo, se observan unas líneas adicionales en los lados horizontales que bajan la efectividad de la técnica aplicada. Al aplicarle segmentación a esta imagen, con un umbral de $T = 150$ o inferiores, no se observan cambios en el rendimiento, que es de 79.91 %. Umbrales mayores a 150 debilitan los bordes correctamente detectados del cuadrado y bajan su rendimiento.

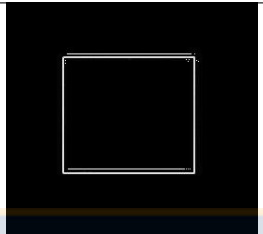
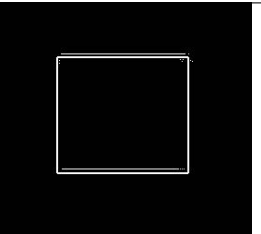
	Imagen Resultante	Imagen Segmentada
		
Aciertos (TP)	1054	1054
Verdaderos Negativos	49113	49113
Falsos Positivos	265	265
Falsos Negativos	0	0
Performance	79.91 %	79.91 %
Score	50167	50167

Tabla 5.2: Evaluación de la resultante del *cuadrado*

5.3.2. Evaluación del Efecto del Blurring

A continuación se muestran las pruebas realizadas al cuadrado con los filtros de suavizado para simular el blurring, éstos son el filtro gaussiano (figura 5.15a) y el filtro de mediana (figura 5.15b).



(a) Suavizado Gaussiano



(b) Suavizado por Filtro de Mediana

Figura 5.15: Cuadrado con Suavizado para Blurring

La aplicación de suavizado para simular el blurring afecta en gran manera el rendimiento del algoritmo propuesto. El suavizado gaussiano, que es el que más difumina la imagen, añade ruido en tres lados del cuadrado

y engrosa los bordes, lo cual considerando la imagen de control (Figura 5.14b), también se considera como ruido. Se observa que la aplicación del filtro de mediana en este caso disminuye un poco el grosor de los bordes y genera ruido en las esquinas de la imagen, sin embargo, su rendimiento es mejor que la imagen con suavizado gaussiano. En la tabla 5.3 se muestran los valores obtenidos tras la medición del performance.

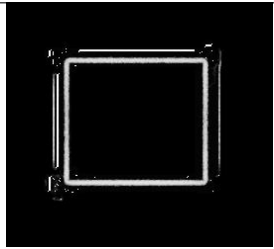
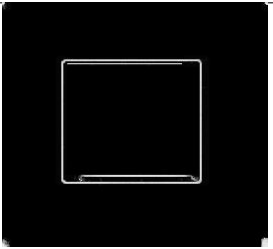
	Suavizado Gaussiano	Suavizado Mediana
		
Aciertos	1054	1041
Falsos Negativos	0	13
Falsos Positivos	1329	446
Evaluación	44.23 %	69.40 %

Tabla 5.3: Evaluación de la resultante del *cuadrado* con blurring

5.3.3. Variaciones del Algoritmo Propuesto

Las medidas introducidas anteriormente permiten evaluar el efecto de ciertos parámetros en el rendimiento del algoritmo propuesto y su comparación con otras técnicas similares.

5.3.3.1. Efecto del Número de Iteraciones

A través del *Score* se verá el efecto del número de iteraciones en el rendimiento del algoritmo propuesto, en distintos valores de población de hormigas. En las tablas 5.4 y 5.5 se observa la medición del *Score* tras variar el número de iteraciones en 100, 500 y 1000 considerando el 30 % y 50 % de hormigas respectivamente.

Efecto de Iteraciones con 30 % de Población

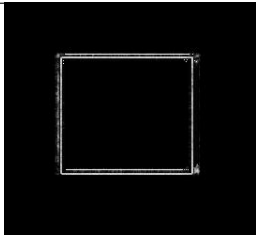
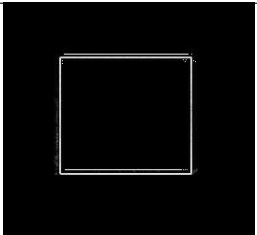
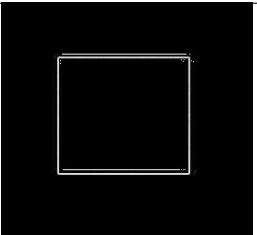
	100 iteraciones	500 iteraciones	1000 iteraciones
			
Aciertos (TP)	1045	1054	1054
Verdaderos Negativos	49108	49113	49113
Score	50153	50167	50167

Tabla 5.4: Score tras variación de iteraciones con 30 % de población

Efecto de Iteraciones con 50 % de Población

	100 iteraciones	500 iteraciones	1000 iteraciones
Aciertos (TP)	1054	1054	1054
Verdaderos Negativos	49106	49113	49113
Score	50160	50167	50167

Tabla 5.5: Score tras variación de iteraciones con 50 % de población

Al evaluar el efecto del número de iteraciones en el algoritmo propuesto se puede ver que el rendimiento se maximiza en un valor menor al escogido en el ajuste de parámetros (Tabla 5.1), sin embargo, se considera que ocurre esto porque es una imagen simple. Sin importar el número de hormigas de la población el resultado es el mismo, tras las 500 iteraciones se maximiza el *Score* del cuadrado. La figura 5.16 muestra esta relación.

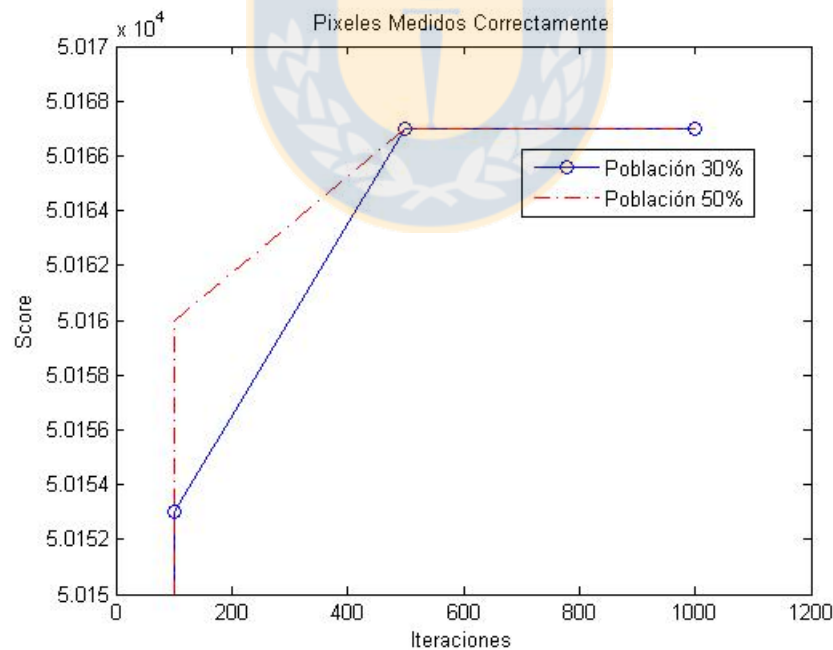


Figura 5.16: *Score* tras variación de iteraciones

5.3.3.2. Comparación con Variaciones del Algoritmo Propuesto

Con la medida del *Performance* se comparará el rendimiento de diferentes variaciones de algoritmos ACO basados en la metodología planteada por Etemad [18]. Se consideran dos variaciones para ser comparadas con la metodología propuesta, que consiste en la obtención de la matriz de información heurística a partir de lógica difusa y la inicialización de la matriz de feromonas como el cociente entre el gradiente y la segunda derivada; la primera es el algoritmo original de Etemad, como se describe en el algoritmo 2, en el cual se inicializa la feromona con una constante y la segunda es una primera variación planteada, considerando el cociente del gradiente y la segunda derivada, sin realizar la hibridización con lógica difusa, sino que manteniendo la definición de estímulo de Etemad, como la norma del gradiente. A cada uno de estos algoritmos, se evalúa el performance tras la aplicación de un umbral por cada valor del intervalo [1, 255] y se selecciona el máximo performance que proporcione uno de los umbrales aplicados.

Performance de Algoritmos con 30 % de Población

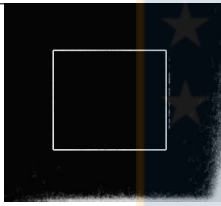
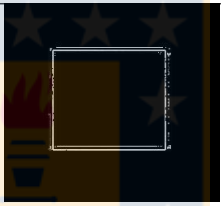
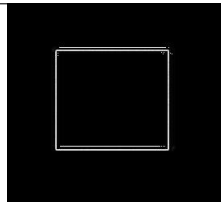
	Feromona Constante	Feromona Gradiente	Feromona Gradiente Híbrida
			
Umbral Máximo	197	121	145
Performance	61.95 %	76.19 %	79.91 %

Tabla 5.6: Performance de algoritmos con 30 % de población

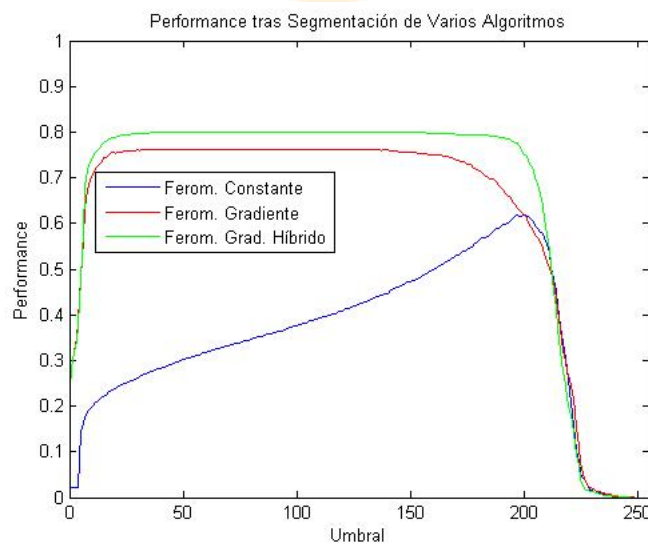


Figura 5.17: *Performance* en diferentes algoritmos con 30 % de población

Performance de Algoritmos con 50 % de Población

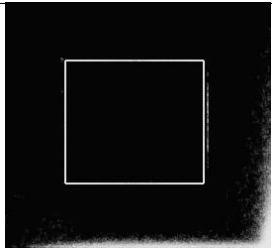
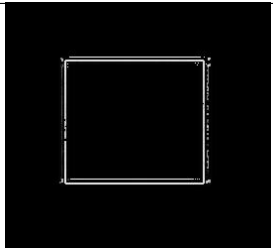
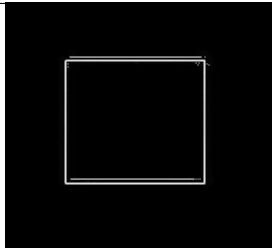
	Feromona Constante	Feromona Gradiente	Feromona Gradiente Híbrida
			
Umbral Máximo	211	147	165
Performance	61.90 %	78.67 %	79.91 %

Tabla 5.7: Performance de algoritmos con 50 % de población

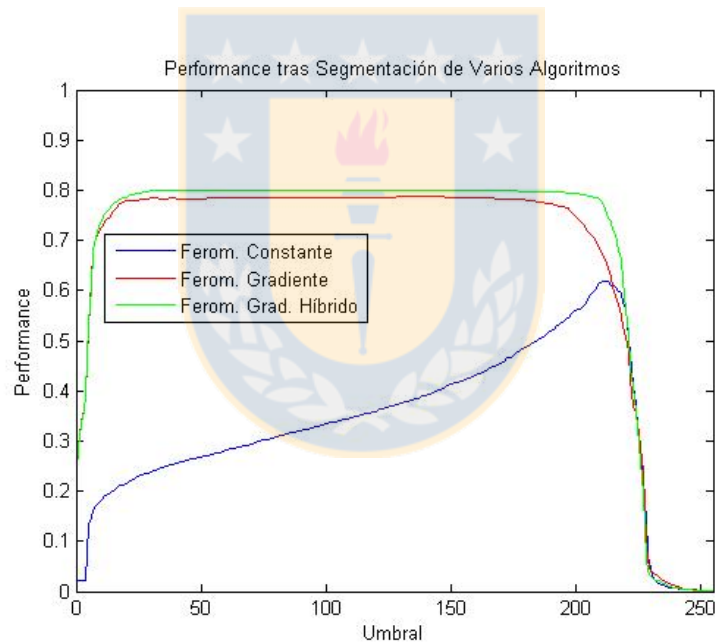


Figura 5.18: *Performance* en diferentes algoritmos con 50 % de población

Tras comparar estas variaciones de algoritmo ACO, se observa en las tablas 5.6 y 5.7 que la inicialización de la matriz de feromonas como el cociente entre el gradiente y la segunda derivada detecta los bordes de la imagen sin generar ruido en las esquinas de la imagen, por lo que el *Performance* es mayor. Además, los gráficos mostrados en las figuras 5.17 y 5.18 muestran el considerable aumento entre el trabajo base de Etemad, donde la matriz de feromonas se inicializa con una constante y la contribución propuesta de la inicialización de la matriz de feromonas con el cociente. La acción de la lógica difusa como la matriz de información heurística entrega los mejores valores en *Performance*, combinándola con la inicialización de feromonas propuesta. En síntesis, podemos deducir que el algoritmo propuesto presenta un mayor rendimiento que el algoritmo presente en la literatura.

Capítulo 6

Resultados en Imágenes DGGE

Tras probar y medir el rendimiento del algoritmo propuesto en imágenes genéricas, incluso simulando el efecto del blurring; en este capítulo se presentan los resultados obtenidos tras la aplicación de la técnica híbrida en las imágenes DGGE con diversas características. En este capítulo se introduce la acción del módulo de ubicación de bandas; el cual permite determinar la posición de una banda en la imagen procesada.

6.1. Proceso de Ubicación de Bandas en el Carril

Al utilizar un carril o conjunto de carriles DGGE para ser procesado por el algoritmo propuesto, además de la entrega de la imagen resultante, se ejecutan los siguientes pasos:

- Selección automática del *umbral* para segmentar la imagen resultante: en este paso, al tener la imagen resultante del algoritmo híbrido, se calcula el histograma de promedios de intensidades por cada fila para obtener con este conjunto de datos, un promedio de las intensidades más significativas que servirá como umbral para segmentar la imagen resultante.

El histograma de **promedios de intensidades** calcula por cada fila de la imagen (considerando un carril vertical) el promedio de las intensidades por cada pixel perteneciente a aquella fila. El resultado gráfico de esta operación se puede ver en la figura 6.1

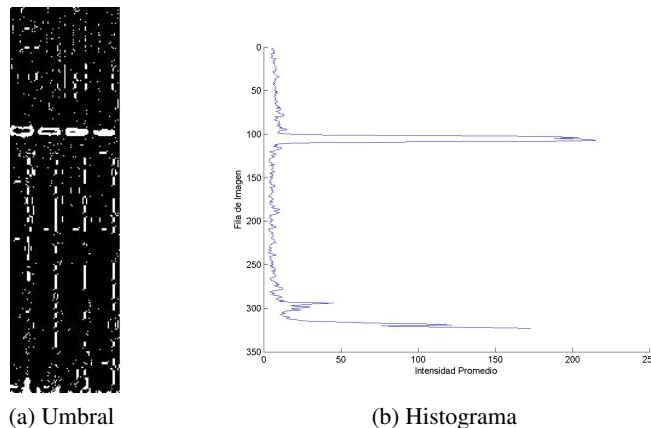


Figura 6.1: Histograma de Promedio de Intensidades

- Luego, tras segmentar la imagen, se calcula nuevamente el histograma de promedio de intensidades y se obtienen las posiciones de los peaks del gráfico, para determinar la ubicación de las bandas; las cuales se marcan en la imagen original y en la resultante segmentada.

6.2. Resultados en Imagen Ideal

Dados los parámetros de la tabla 5.1, primero se utilizará una imagen (Figura 6.2) en la cual se distinguen perfectamente las bandas en los distintos carriles. También se verá el efecto de la variación de algunos parámetros en la detección de las bandas.



Figura 6.2: Imagen DGGE ideal

La aplicación del algoritmo propuesto con los parámetros indicados en la tabla 5.1 en esta imagen produce como resultado la imagen de la figura 6.3 y tras la aplicación de un umbral $T = 200$, la imagen segmentada se puede apreciar en la figura 6.4. En este caso, el umbral aplicado se obtuvo de manera manual.

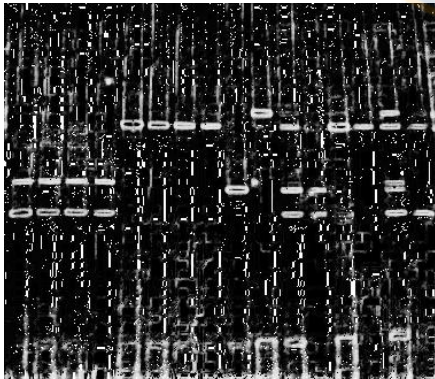


Figura 6.3: Imagen Resultante Alg. Propuesto

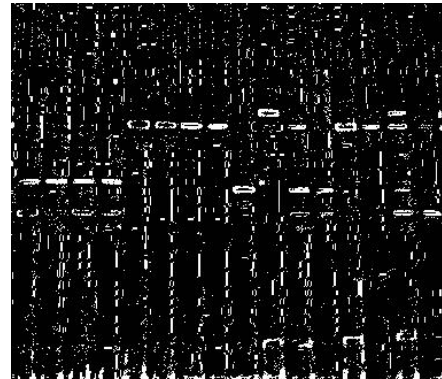


Figura 6.4: Segmentación de Imagen con $T = 200$

6.2.1. División de la Imagen en Carriles

Para probar el efecto de la variación de algunos parámetros y calcular el umbral de manera automática e identificar la posición de las bandas, se divide la figura 6.2 en cuatro grupos de carriles, que se pueden ver en la figura 6.5.

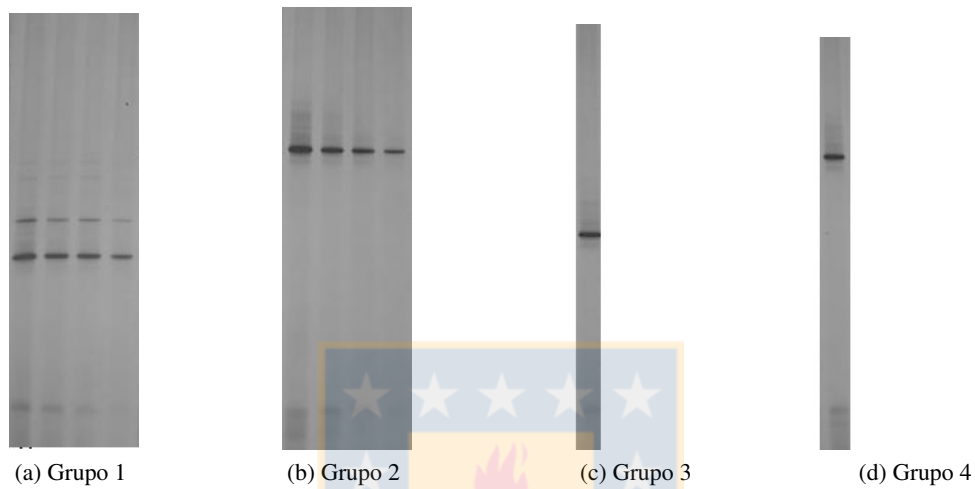


Figura 6.5: División de Carriles de la imagen 6.2

Los resultados tras la aplicación del algoritmo propuesto a cada grupo se aprecian en la figura 6.6.

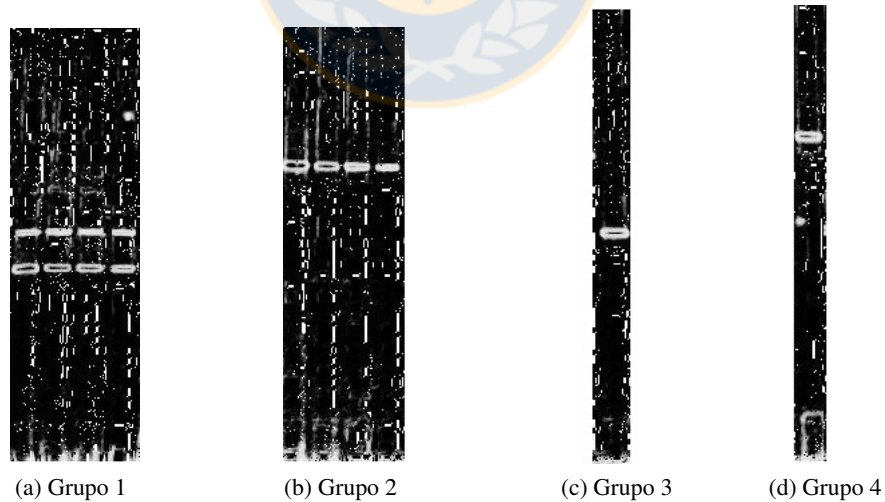


Figura 6.6: Resultados por Grupo de Carriles

6.2.2. Ubicación de bandas

Tras realizar el procesamiento a través del algoritmo como se observa en la figura 6.6, a continuación se aplican los métodos presentes en el módulo de ubicación de bandas: el cálculo del histograma de la imagen y la

detección de los puntos en que se encuentran los bordes de las bandas, por cada una de las divisiones mostradas en la figura 6.5, se muestra en las siguientes imágenes (Figuras 6.7 a 6.10).

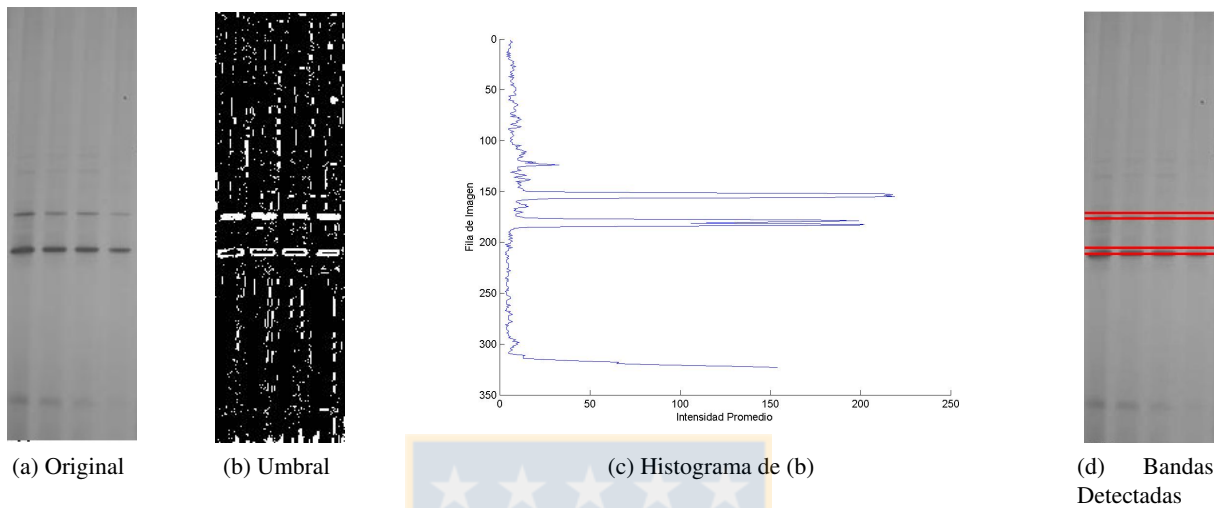


Figura 6.7: Identificación de Bandas en Grupo 1

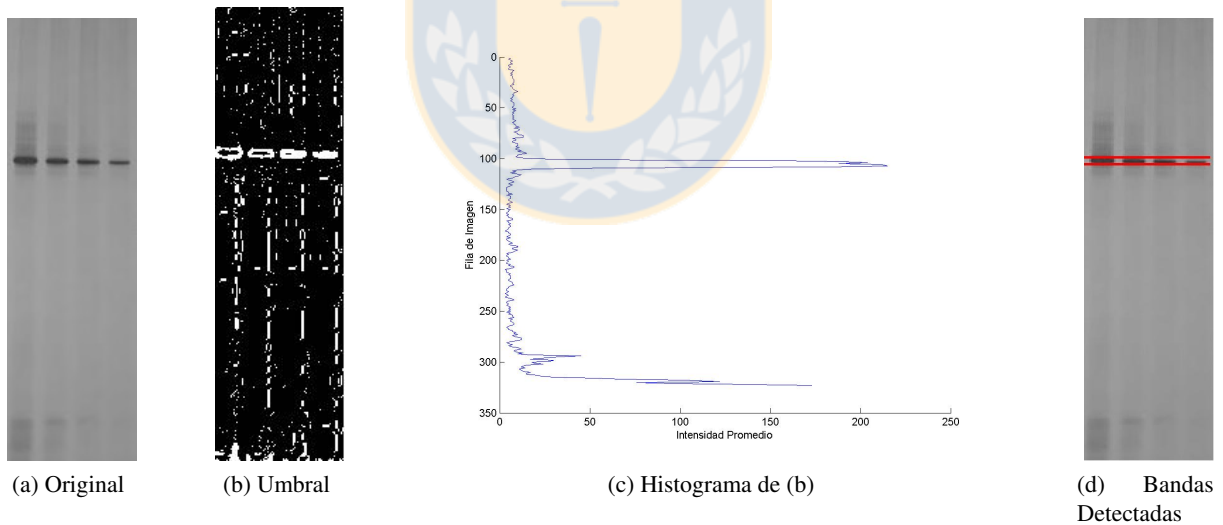


Figura 6.8: Identificación de Bandas en Grupo 2

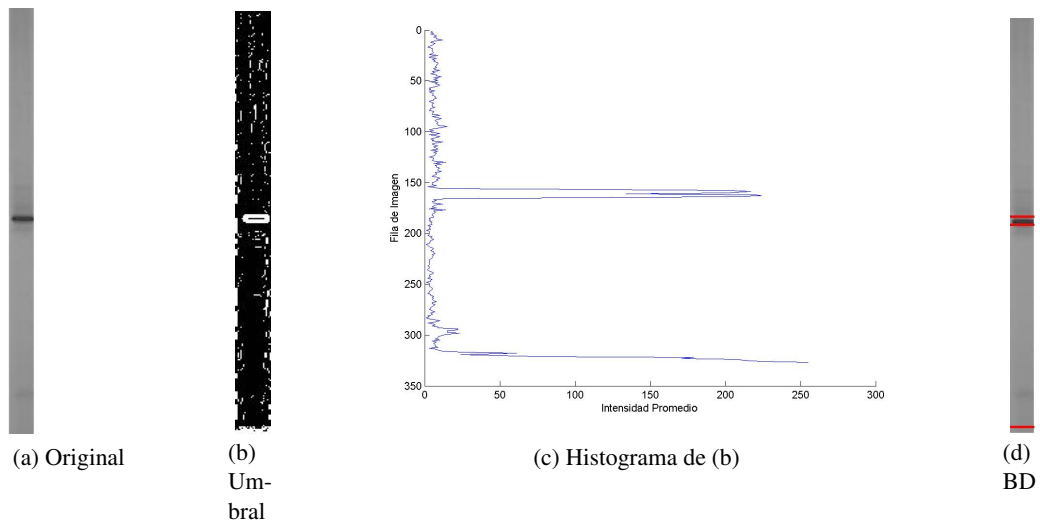


Figura 6.9: Identificación de Bandas en Carril 3

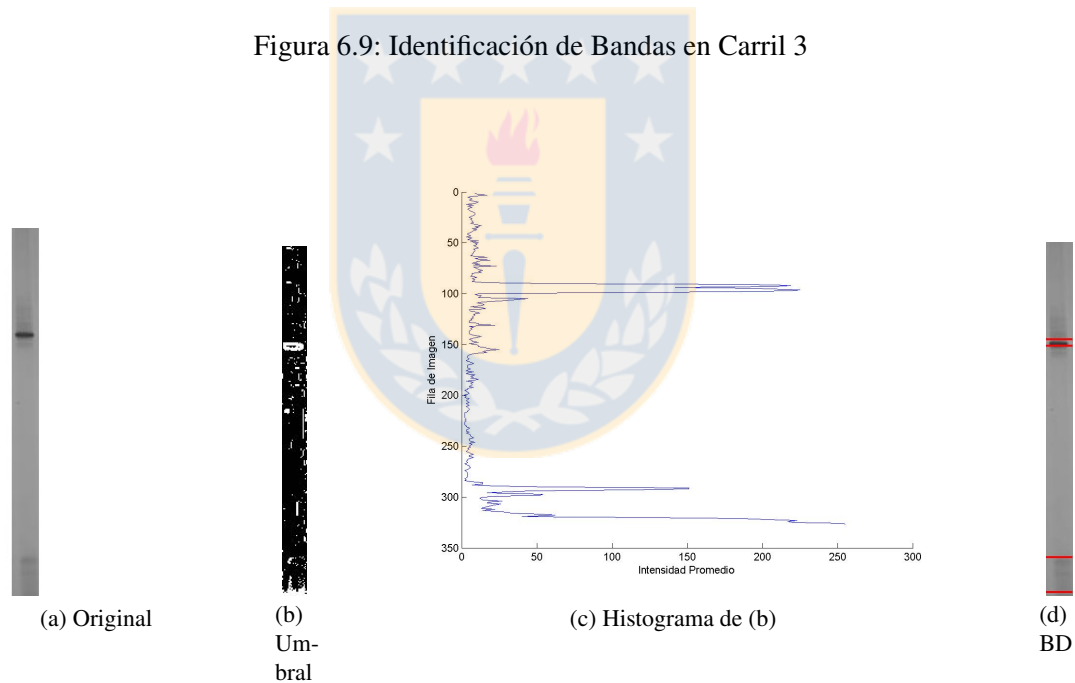


Figura 6.10: Identificación de Bandas en Grupo 4

6.3. Imágenes DGGE con diferentes características

Las imágenes DGGE generalmente no presentan sus bandas tan claras como se puede apreciar en la figura 6.2, sino que presentan difuminaciones, curvaturas en las bandas; o el cambio de color entre el fondo y las bandas. El efecto de estas características en la detección de las bandas se presenta en esta sección, realizando los procesamientos de detección, aplicación de umbral y ubicación de bandas.

6.3.1. Color de las bandas

En la figura 6.2 la imagen tenía bandas negras con fondo negro. Se presenta una variación a esto, que consiste en la inversa de la imagen presentada, bandas blancas con fondo negro, tal como se puede ver en la figura 6.11.

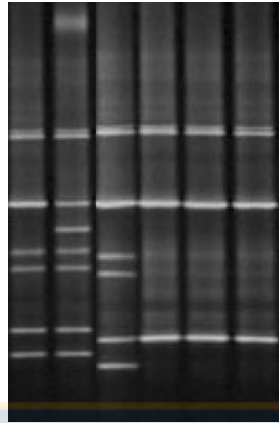


Figura 6.11: Imagen DGGE con bandas blancas

Para la identificación de bandas esta imagen también se dividió en distintos grupos de carriles, los que se muestran en la figura 6.12, junto a su resultante tras aplicar el algoritmo híbrido.

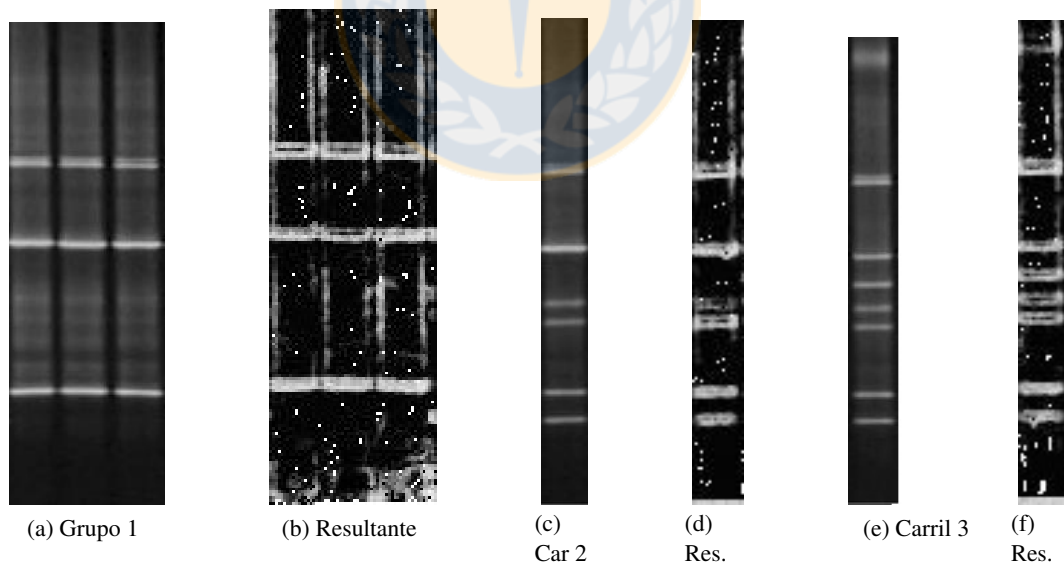


Figura 6.12: Grupos de Carriles con Bandas Blancas y sus Resultantes

Para ubicar las bandas en el carril se ocupará el carril 3, tal como se observa en la figura 6.13.

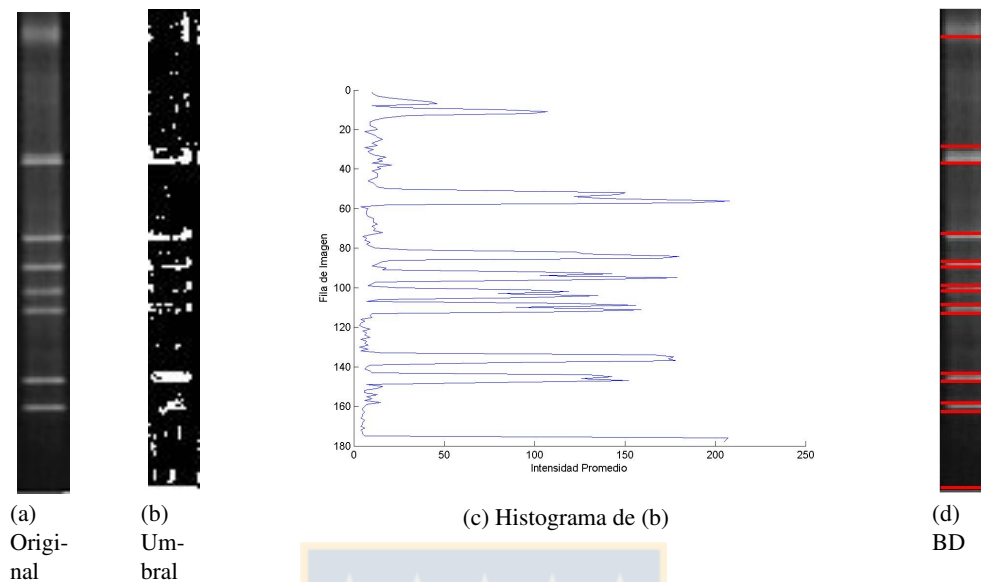


Figura 6.13: Identificación de Bandas en Carril 3 Blanco

6.3.2. Bandas Difusas

La siguiente característica a evaluar en las imágenes DGGE es la presencia de bandas difusas, las cuales son aquellos en que no hay una distinción total entre la banda y el fondo, principalmente generadas en la obtención de la imagen, cuando la banda se corre por el carril abarcando un área mucho mayor, como se muestra en la figura 6.14



Figura 6.14: Imagen DGGE con bandas difusas

Para la detección y ubicación de bandas se utilizaron algunos carriles de la imagen 6.14, los cuales junto a sus resultantes se muestran en la figura 6.15.

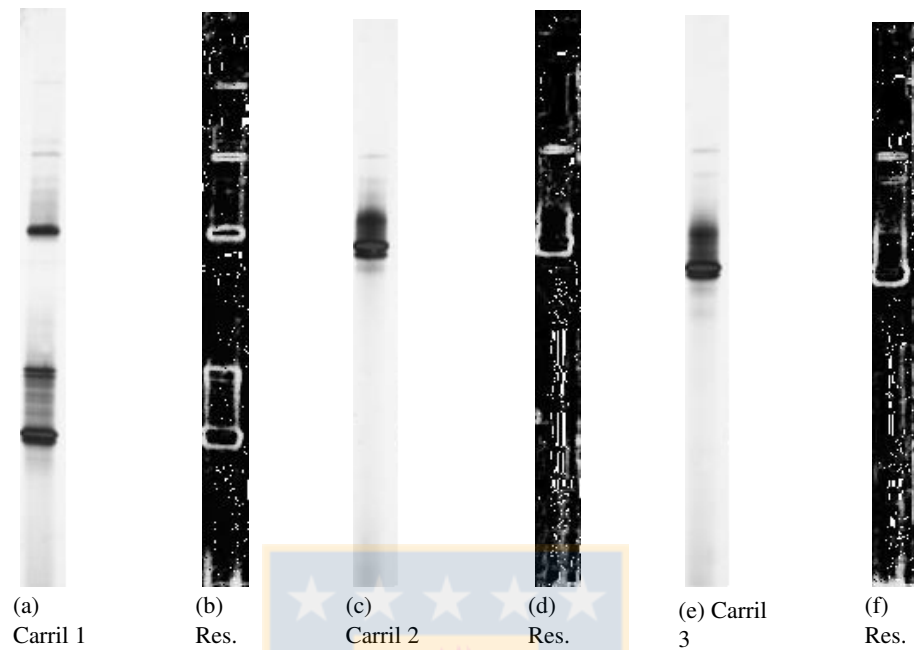


Figura 6.15: Grupos de Carriles con Bandas Difusas y sus Resultantes

El carril 1 (Figura 6.15a) será procesado para obtener la ubicación de las bandas detectadas con el algoritmo propuesto, como se puede ver en la figura 6.16.

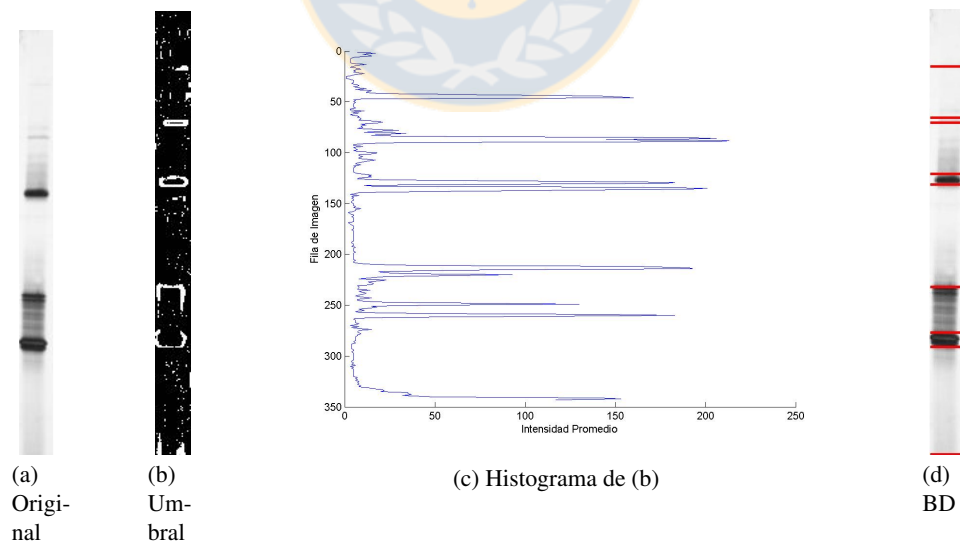


Figura 6.16: Identificación de Bandas en Carril 1 Difuso

6.3.3. Carriles Inclinados

Por último, se presenta la situación en que los carriles de la imagen aparecen inclinados, como se aprecia en la imagen 6.17. Como se puede apreciar, el primer carril de la imagen tiene una inclinación, producida durante la generación de la imagen; a la cual se le aplica el proceso de detección y ubicación de bandas, mostrado en la figura 6.18.

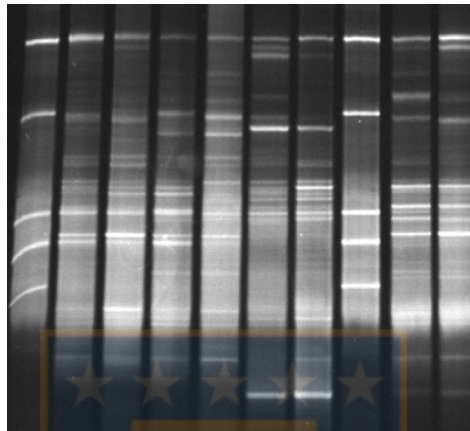


Figura 6.17: Imagen DGGE con carriles inclinados

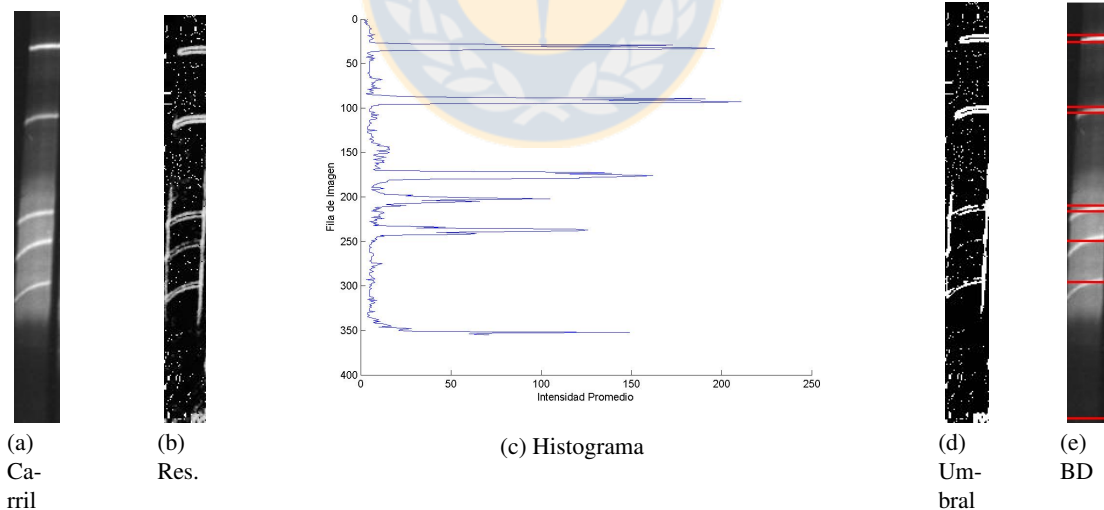


Figura 6.18: Identificación de Bandas en Carril Inclinado

6.4. Comparación con Técnicas Tradicionales

A continuación se muestran los resultados de la aplicación de las técnicas tradicionales Roberts, Prewitt, Sobel y Canny a los carriles seleccionados en las secciones anteriores, para evaluar su rendimiento y compararlos con los resultados obtenidos con el algoritmo propuesto.

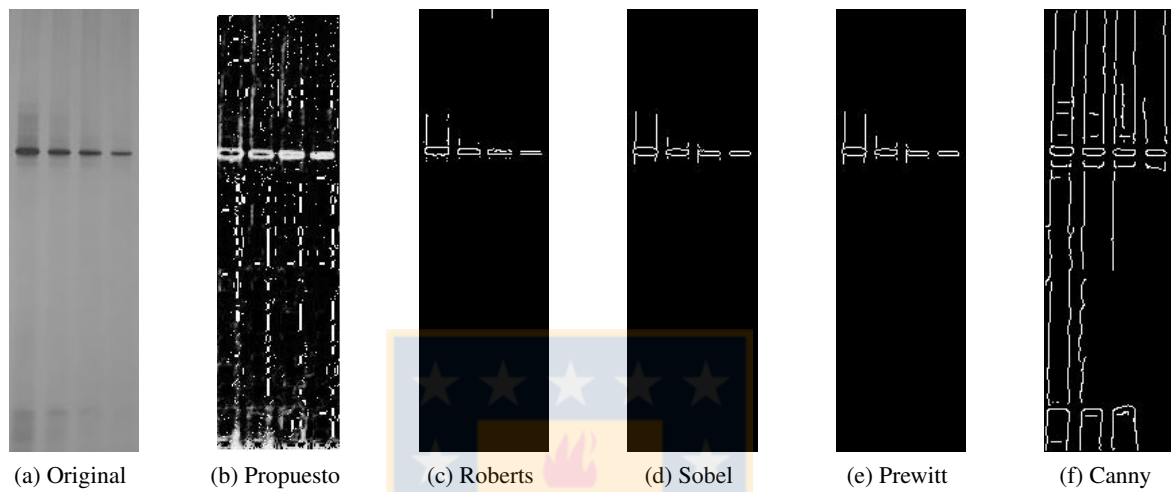


Figura 6.19: Métodos de Detección de Bordes en Carriles Claros

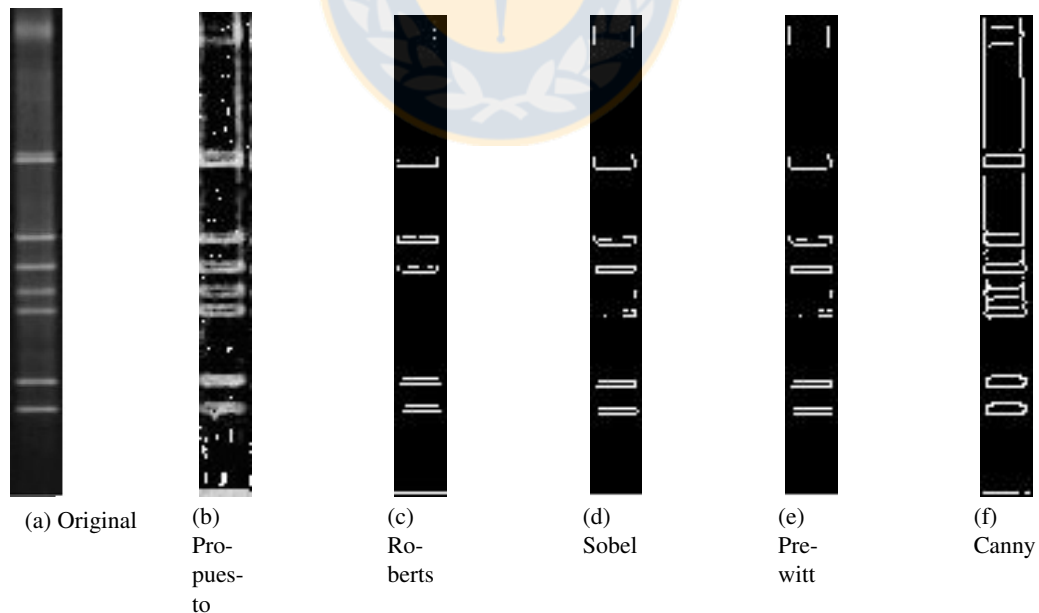


Figura 6.20: Métodos de Detección de Bordes en Carriles con Bandas Blancas

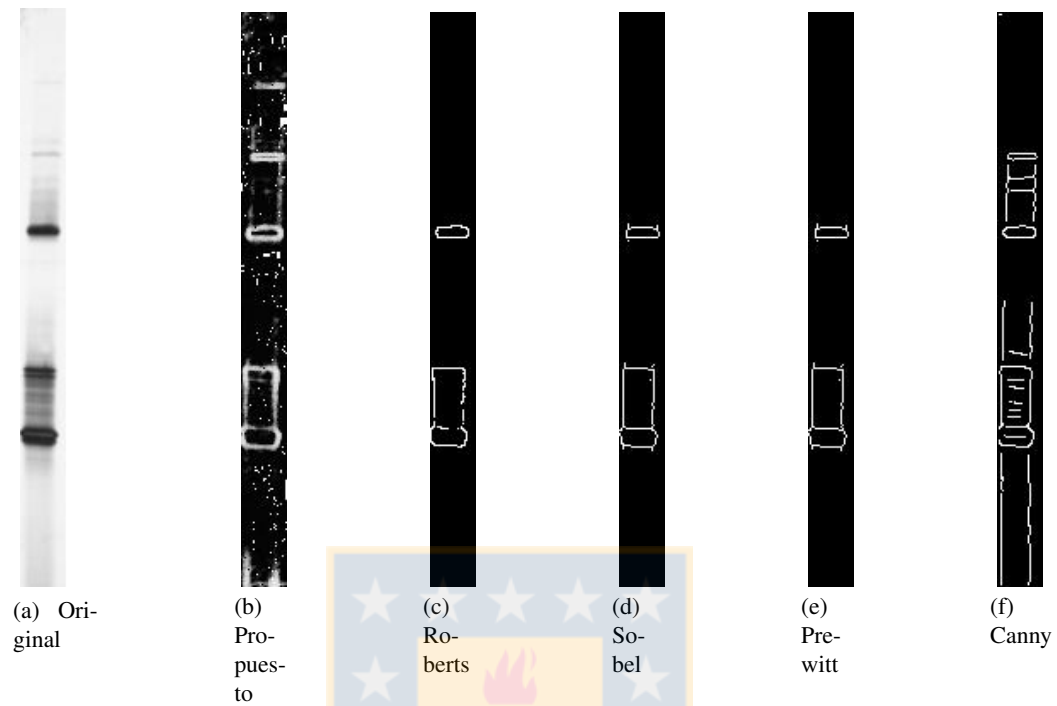


Figura 6.21: Métodos de Detección de Bordes en Carriles con Bandas Borrosas



Figura 6.22: Métodos de Detección de Bordes en Carriles con Bandas Inclinadas

6.5. Análisis de Resultados

Tras la aplicación del algoritmo propuesto y el módulo de ubicación de bandas en las distintas imágenes DGGE mostradas en las secciones 6.2 y 6.3 se puede afirmar que se entregan buenos resultados en la detección de las bandas en imágenes DGGE, cualquiera sea su naturaleza, ya que se observa claramente el borde de las bandas en los carriles. Incluso, es posible cargar una imagen completa en el sistema y se entrega una imagen con los bordes bien definidos, aunque, por la disposición de los carriles en las imágenes DGGE no se pueden identificar las posiciones de las bandas en este caso, sino que esta funcionalidad solo está presente cuando se ingresa un carril o un conjunto de carriles con características similares. Cabe señalar que el output entregado (las imágenes con líneas rojas) constituyen una probabilidad que en la zona marcada se encuentre una banda, ya que el sistema detecta como bordes las zonas en que hay una diferencia de intensidad con el fondo de la imagen; sin embargo, se reduce en gran manera los errores de detección que al utilizar un software que trabaje con técnicas tradicionales como el Quantity One (Figura 4.1).

En cuanto a los resultados en las distintas imágenes DGGE probadas en este capítulo, se puede señalar que:

- El color de las bandas en los carriles no afecta al resultado global del algoritmo, ya que éste se basa en encontrar las diferencias de intensidad existentes en la imagen, y las bandas, ya sean blancas o negras, están representadas bajo un fondo del color inverso a ellas.
- Al detectar las bandas de una imagen difusa, en las cuales las bandas no se distinguen bien del fondo, ya que éstas se esparcen por una zona del carril mucho mayor que una banda normal, el algoritmo detecta el inicio y el final del área que la banda esparcida abarca. Esta información puede ser utilizada por los expertos para acotar la zona en que se encuentran las bandas.
- Cuando se tiene un carril con bandas inclinadas, éstas son detectadas por el algoritmo, sin embargo, al momento de detectar la posición de las bandas, se muestra la línea en el pixel con mayor intensidad, lo cual permitiría indicar que en las zonas cercanas a la línea indicada en la imagen resultante habría una banda. En estos casos habría que enderezar las bandas para así aplicar el algoritmo de detección y ubicación de bandas.

Al comparar los resultados obtenidos con las técnicas tradicionales se puede señalar que los métodos utilizados, salvo Canny, detectan solo lo más marcado, dejando de lado muchos detalles en la imagen que pueden ser bandas, en contraposición a lo entregado con la técnica propuesta, que detecta más bandas que estos métodos; lo que afecta también al proceso de ubicación de bandas, las técnicas tradicionales solo entregarían la misma información que puede inferir un humano sobre este proceso. Al realizar la misma comparación con el algoritmo de Canny, si bien detecta de buena manera las bandas, genera mucho más ruido, en forma de líneas, que la técnica propuesta; el cual es reducible con el algoritmo propuesto al realizar la segmentación por umbral, funcionalidad que no se puede realizar en la imagen entregada por Canny, ya que esta es una imagen binaria. Además el ruido presente en la imagen detectada por Canny afecta el proceso de ubicación de las bandas, pues las líneas continuas que se generan en ella, el histograma también refleja el ruido presente y la ubicación de bandas se hace más difícil.

Capítulo 7

Conclusiones y Trabajo Futuro

La detección de bandas en imágenes DGGE fue la motivación para este trabajo de tesis en el cual se propuso una técnica híbrida para la solución de este problema. En la literatura revisada existen técnicas basadas en algoritmos ACO para detectar las bandas, y variadas combinaciones de algoritmos en el caso de la detección de bordes en imágenes genéricas. Una de estas combinaciones es la utilizada por Tyagi [52] donde se realiza una hibridización entre Ant Colony System y lógica difusa. Generalmente en los trabajos que utilizan algoritmos basados en hormigas inicializan la matriz de feromonas como una constante.

El método propuesto en este trabajo consiste en realizar una combinación entre un procesamiento de la imagen ocupada como input a través de la lógica difusa, para que sea utilizado como información heurística en la determinación del movimiento de una hormiga en un algoritmo ACS modificado, en donde la feromona es inicializada en base a las derivadas de la imagen, en forma del cociente del gradiente y la segunda derivada de la imagen. La elección de esta forma de inicialización fue realizada por las funciones que cumplen por separado estos elementos en la detección de bordes, ambos, por separado se utilizan en técnicas tradicionales para este propósito.

Los resultados del método propuesto en imágenes genéricas hubo una buena detección de los bordes, en las cuales solo aparece ruido en forma de una dispersión de puntos, pero los bordes detectados son finos y se aprecian los detalles de las imágenes probadas, de mejor manera que lo realizado con técnicas tradicionales. Al comparar este algoritmo con su base (inicialización de feromonas como constante) y quitando una componente (algoritmo ACO con matriz de feromonas como el cociente del gradiente y segunda derivada), el algoritmo híbrido obtuvo un mayor rendimiento que las otras dos técnicas y además, la introducción de la inicialización de la matriz de feromonas como se propuso en este trabajo mejora en gran manera la detección de bordes en imágenes genéricas, sin generar una gran cantidad de ruido que baje de forma considerable el rendimiento, como ocurre con el algoritmo ACS inicializado de forma tradicional. Incluso, con blurring simulado en estos casos, los detalles de las imágenes no se pierden tras la detección de bordes.

En las imágenes DGGE también se registró una buena detección de las bandas en diferentes características de la imagen. Además esta técnica permite identificar la posición de las bandas e indicarlas en la imagen original. El cambio de color de las bandas no influye en la detección de ellas; cuando existen bordes difusos que abarcan un gran área del carril se indica la posición de inicio y fin del área abarcada por la banda y al haber carriles inclinados, se detectan las bandas, pero no se puede identificar la posición de ellas, por lo cual para solucionar este problema se deben enderezar las bandas.

En términos generales, se ha construido una técnica híbrida que permite detectar bordes en imágenes genéricas que tolera el blurring presente en las imágenes y para las imágenes DGGE se tiene un sistema que detecta e identifica las bandas presentes en cada carril, lo que se debe interpretar como una probabilidad que en la zona indicada existe una banda, información que debe ser corroborada por un experto, pero implica una mejora a lo realizado por técnicas tradicionales.

Como trabajo futuro se propone lo siguiente:

- Paralelizar los módulos principales del algoritmo de detección de bandas: la obtención de la matriz heurística a partir de la lógica difusa y la detección de bordes a través de ACO, para reducir el tiempo de ejecución del algoritmo. Dado que en los algoritmos de hormigas, específicamente en el Ant Colony System utilizado en este trabajo, las hormigas recorren su camino de forma independiente. Con los procesos de inferencia y defusificación de la lógica difusa también es posible trabajar de forma paralela en las distintas ventanas que se generan al dividir la imagen (en este caso son ventanas de 3x3).
- Utilizar transiciones de n píxeles de distancia, conocidas como rampas, en vez de la detección de bordes usual que considera la variación entre un píxel y sus vecinos.
- Inicializar las posiciones de las hormigas en el algoritmo ACO, tratando que cada hormiga se ubique en una ventana de $i \times j$ píxeles, generalmente asumiendo $i = j = 3$, en vez de la inicialización usual, que consiste en la ubicación aleatoria de las hormigas.
- Probar con nuevas técnicas para hacer que el algoritmo sea capaz de detectar bandas en el caso que éstas no se distingan del fondo, situación que ocurre cuando las bandas están muy difusas y se expanden por un área mayor.

Bibliografía

- [1] I. Aizenberg, T. Bregin, C. Butakoff, V. Karnaukhov, N. Merzlyakov, and O. Milukova. Type of blur and blur parameters identification using neural network and its application to image restoration. In J. Dorransoro, editor, *Artificial Neural Networks, ICANN 2002*, volume 2415 of *Lecture Notes in Computer Science*, pages 1231–1236. Springer Berlin Heidelberg, 2002.
- [2] S. S. Al-amri, N. V. Kalyankar, and S. D. Khamitkar. Deblurred gaussian blurred images. *CoRR*, abs/1004.4448, 2010.
- [3] B. Bullnheimer, R. F. Hartl, and C. Strauss. A new rank based version of the Ant System. A computational study. 1997.
- [4] J. Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.
- [5] D. Chen, T. Zhou, and X. Yu. A new method of edge detection based on PSO. In *Advances in Neural Networks–ISNN 2012*, pages 239–246. Springer, 2012.
- [6] R. Contreras, M. Pinninghoff, and J. Ortega. Using ant colony optimization for edge detection in gray scale images. In J. M. Ferrandez Vicente, J. R. Alvarez Sanchez, F. de la Paz Lopez, and F. J. Toledo Moreo, editors, *Natural and Artificial Models in Computation and Biology*, volume 7930 of *Lecture Notes in Computer Science*, pages 323–331. Springer Berlin Heidelberg, 2013.
- [7] A. Danielyan, V. Katkovnik, and K. Egiazarian. Bm3d frames and variational image deblurring. *Image Processing, IEEE Transactions on*, 21(4):1715–1728, April 2012.
- [8] L. Davis. *Handbook of genetic algorithms*. VNR computer library. Van Nostrand Reinhold, 1991.
- [9] J.-L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels. The self-organizing exploratory pattern of the argentine ant. *Journal of insect behavior*, 3(2):159–168, 1990.
- [10] J. Domke and Y. Aloimonos. Image transformations and blurring. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):1000–9999, May 2009.
- [11] M. Dorigo. Optimization, learning and natural algorithms. *Ph. D. Thesis, Politecnico di Milano, Italy*, 1992.
- [12] M. Dorigo, G. D. Caro, and L. M. Gambardella. Ant algorithms for discrete optimization. *Artificial life*, 5(2):137–172, 1999.
- [13] M. Dorigo and L. M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on*, 1(1):53–66, 1997.

- [14] M. Dorigo, V. Maniezzo, A. Colorni, and V. Maniezzo. Positive feedback as a search strategy. 1992.
- [15] M. Dorigo and T. Stützle. *Ant Colony Optimization*. A Bradford book. University Press Group Limited, 2004.
- [16] D. Driankov, H. Hellendoorn, and M. Reinfrank. *An introduction to fuzzy control*. Springer Science & Business Media, 2013.
- [17] D. Ercolini. PCR-DGGE fingerprinting: novel strategies for detection of microbes in food. *J Microbiol Methods*, 56(3):297–314, 2004.
- [18] S. A. Etemad and T. White. An ant-inspired algorithm for detection of image edge features. *Applied Soft Computing*, 11(8):4883–4893, 2011.
- [19] C. Figueroa. Identificación y cuantificación de bandas en imágenes DGGE con algoritmos basados en hormigas. *Tesis de Magister en Ciencias de la Computación, Universidad de Concepción*, Mayo 2012.
- [20] R. Fullér. *Introduction to neuro-fuzzy systems*, volume 2. Springer Science & Business Media, 2013.
- [21] C. I. Gonzalez, P. Melin, J. R. Castro, O. Mendoza, and O. Castillo. An improved sobel edge detection method based on generalized type-2 fuzzy logic. *Soft Computing*, pages 1–12, 2014.
- [22] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006.
- [23] A. Gupta, N. Joshi, C. Lawrence Zitnick, M. Cohen, and B. Curless. Single image deblurring using motion density functions. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision ECCV 2010*, volume 6311 of *Lecture Notes in Computer Science*, pages 171–184. Springer Berlin Heidelberg, 2010.
- [24] M. Juneja and P. S. Sandhu. Performance evaluation of edge detection techniques for images in spatial domain. *methodology*, 1(5):614–621, 2009.
- [25] A. Levin. Blind motion deblurring using image statistics. In *Advances in Neural Information Processing Systems*, pages 841–848, 2006.
- [26] A. Levin, Y. Weiss, F. Durand, and W. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1964–1971, June 2009.
- [27] A. Levin, Y. Weiss, F. Durand, and W. Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2657–2664, June 2011.
- [28] E. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1):1 – 13, 1975.
- [29] D. Marr and E. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207(1167):187–217, 1980.
- [30] L. McAuliffe, R. J. Ellis, J. R. Lawes, R. D. Ayling, and R. A. Nicholas. 16S rDNA PCR and denaturing gradient gel electrophoresis; a single generic test for detecting and differentiating Mycoplasma species. *Journal of medical microbiology*, 54(8):731–739, 2005.

- [31] A. J. McBain, R. G. Bartolo, C. E. Catrenich, D. Charbonneau, R. G. Ledder, A. H. Rickard, S. A. Symmons, and P. Gilbert. Microbial characterization of biofilms in domestic drains and the establishment of stable biofilm microcosms. *Applied and environmental microbiology*, 69(1):177–185, 2003.
- [32] K. M. Miller, T. Ming, A. Schulze, and R. E. Withler. Denaturing gradient gel electrophoresis (DGGE): a rapid and sensitive technique to screen nucleotide sequence variation in populations. *Biotechniques*, 27:1016–1031, 1999.
- [33] G. Muyzer, E. C. De Waal, and A. G. Uitterlinden. Profiling of complex microbial populations by denaturing gradient gel electrophoresis analysis of polymerase chain reaction-amplified genes coding for 16S rRNA. *Applied and environmental microbiology*, 59(3):695–700, 1993.
- [34] R. M. Myers, T. Maniatis, and L. S. Lerman. Detection and localization of single base changes by denaturing gradient gel electrophoresis. In R. Wu, editor, *Recombinant DNA Part F*, volume 155 of *Methods in Enzymology*, pages 501 – 527. Academic Press, 1987.
- [35] S. Osher and L. I. Rudin. Feature-oriented image enhancement using shock filters. *SIAM J. Numer. Anal.*, 27(4):919–940, Aug. 1990.
- [36] K. M. Passino, S. Yurkovich, and M. Reinfrank. *Fuzzy control*, volume 42. Citeseer, 1998.
- [37] M. Pinninghoff, M. Valenzuela, R. Contreras, and M. Mora. Automatic lane correction in DGGE images by using hybrid genetic algorithms. In M. Polycarpou, A. de Carvalho, J.-S. Pan, M. Wozniak, H. Quintian, and E. Corchado, editors, *Hybrid Artificial Intelligence Systems*, volume 8480 of *Lecture Notes in Computer Science*, pages 221–232. Springer International Publishing, 2014.
- [38] M. A. Pinninghoff, D. Venegas, and R. Contreras. Genetic algorithms and tabu search for correcting lanes in dna images. In J. F. Martinez-Trinidad, J. A. Carrasco-Ochoa, and J. Kittler, editors, *Advances in Pattern Recognition*, volume 6256 of *Lecture Notes in Computer Science*, pages 144–153. Springer Berlin Heidelberg, 2010.
- [39] J. A. Richards. *Remote sensing digital image analysis*. Springer, 2012.
- [40] T. J. Ross. *Fuzzy logic with engineering applications*. John Wiley & Sons, 2009.
- [41] L. Rueda, O. Uyarte, S. Valenzuela, and J. Rodriguez. Processing random amplified polymorphism DNA images using the radon transform and mathematical morphology. In *Proceedings of the 4th international conference on Image Analysis and Recognition, ICIAR'07*, pages 1071–1081, Berlin, Heidelberg, 2007.
- [42] N. Samsudin, R. Hashim, and N. E. A. Khalid. Denoising and block gridding of microarray image using mathematical morphology. In *Computing and Convergence Technology (ICCCT), 2012 7th International Conference on*, pages 230–235. IEEE, 2012.
- [43] N. Senthilkumaran. Genetic algorithm approach to edge detection for dental x-ray image segmentation. *International Journal of Advanced Research in Computer Science and Electronics Engineering (IJARCSEE)*, 1(7):pp–179, 2012.
- [44] N. Senthilkumaran and R. Rajesh. Edge detection techniques for image segmentation-a survey of soft computing approaches. *International Journal of Recent Trends in Engineering*, 1(2):250–254, 2009.

- [45] G. Shrivakshan and C. Chandrasekar. A Comparison of various Edge Detection Techniques used in Image Processing. 2012.
- [46] T. Stutzle and H. Hoos. MAX-MIN ant system and local search for the traveling salesman problem. In *Evolutionary Computation, 1997., IEEE International Conference on*, pages 309–314. IEEE, 1997.
- [47] M. Sugeno. *Industrial Applications of Fuzzy Control*. Elsevier Science Inc., New York, NY, USA, 1985.
- [48] Y. Sugiyama, H. Saito, H. Takei, and K. Mori. A Similarity Analysis of DGGE Images using DP matching. In *TENCON 2005 2005 IEEE Region 10*, pages 1–5, 2005.
- [49] Y.-W. Tai, P. Tan, and M. Brown. Richardson-lucy deblurring for scenes under a projective motion path. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(8):1603–1618, Aug 2011.
- [50] H. Takeda, S. Farsiu, and P. Milanfar. Deblurring using regularized locally adaptive kernel regression. *Image Processing, IEEE Transactions on*, 17(4):550–563, April 2008.
- [51] S. Tiwari, V. P. Shukla, S. Biradar, and A. K. Singh. Blind restoration of motion blurred barcode images using ridgelet transform and radial basis function neural network. *Electronic Letters on Computer Vision and Image Analysis*, 13(3):63–80, 2014.
- [52] Y. Tyagi, T. Puntambekar, P. Sexena, and S. Tanwani. A hybrid approach to edge detection using ant colony optimization and fuzzy logic. *International Journal of Hybrid Information Technology*, 5(1):37–46, 2012.
- [53] J. Vasavada and S. Tiwari. Article: An edge detection method for grayscale images based on bp feedforward neural network. *International Journal of Computer Applications*, 67(2):22–28, April 2013.
- [54] O. P. Verma, M. Hanmandlu, P. Kumar, S. Chhabra, and A. Jindal. A novel bacterial foraging technique for edge detection. *Pattern recognition letters*, 32(8):1187–1196, 2011.
- [55] H. Villagrán. Detección de bordes en imágenes DGGE utilizando algoritmos híbridos. *Memoria de Título Ingeniería Civil Informática, Universidad de Concepción*, Enero 2014.
- [56] Q. Wu, X. Zhao, S. Zhao, et al. Application of PCR-DGGE in research of bacterial diversity in drinking water. *Biomedical and Environmental Sciences*, 19(5):371, 2006.
- [57] Z. Yu and H. R. Karimi. Edge detector design based on LS-SVR. *Mathematical Problems in Engineering*, 2014, 2014.
- [58] L. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338 – 353, 1965.
- [59] L. A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-3(1):28–44, Jan 1973.
- [60] L. A. Zadeh. Soft computing and fuzzy logic. *Software, IEEE*, 11(6):48–56, Nov 1994.
- [61] Y. Zhang and Q. Han. Edge detection algorithm based on wavelet transform and mathematical morphology. In *Control, Automation and Systems Engineering (CASE), 2011 International Conference on*, pages 1–3. IEEE, 2011.
- [62] J. Zhao and B. Bose. Evaluation of membership functions for fuzzy logic controlled induction motor drive. In *IECON 02 [Industrial Electronics Society, IEEE 2002 28th Annual Conference of the]*, volume 1, pages 229–234 vol.1, Nov 2002.

- [63] D. Ziou, S. Tabbone, et al. Edge detection techniques-an overview. *Pattern Recognition And Image Analysis C/C Of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, 8:537–559, 1998.

