



**UNIVERSIDAD CATÓLICA  
DE SANTIAGO DE GUAYAQUIL  
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL  
DESARROLLO  
CARRERA DE TELECOMUNICACIONES**

TEMA:

**Análisis comparativo de los operadores de detección de  
bordes en el procesamiento de imágenes utilizando MatLab**

AUTOR:

León Falconí, Shirley Andrea

Componente práctico del examen complejo previo a la  
obtención del grado de **INGENIERO EN  
TELECOMUNICACIONES**

REVISOR:

M. Sc. Romero Paz, Manuel de Jesús

Guayaquil, Ecuador

10 de marzo del 2021



**UNIVERSIDAD CATÓLICA  
DE SANTIAGO DE GUAYAQUIL  
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO  
CARRERA DE TELECOMUNICACIONES**

**CERTIFICACIÓN**

Certificamos que el presente **componente práctico del examen complejo**, fue realizado en su totalidad por **León Falconí, Shirley Andrea** como requerimiento para la obtención del título de **INGENIERA EN TELECOMUNICACIONES**.

REVISOR

M. Sc. Romero Paz, Manuel de Jesús

DIRECTOR DE CARRERA

M. Sc. Heras Sánchez, Miguel Armando

Guayaquil, a los 10 días del mes de marzo del año 2021



**UNIVERSIDAD CATÓLICA  
DE SANTIAGO DE GUAYAQUIL  
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO  
CARRERA DE TELECOMUNICACIONES**

**DECLARACIÓN DE RESPONSABILIDAD**

Yo, **León Falconí, Shirley Andrea**

**DECLARÓ QUE:**

**El componente práctico del examen complejo, Análisis comparativo de los operadores de detección de bordes en el procesamiento de imágenes utilizando MatLab**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 10 días del mes de marzo del año 2021

EL AUTOR

---

LEÓN FALCONÍ, SHIRLEY ANDREA



**UNIVERSIDAD CATÓLICA  
DE SANTIAGO DE GUAYAQUIL  
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO  
CARRERA DE TELECOMUNICACIONES**

**AUTORIZACIÓN**

**Yo, León Falconí, Shirley Andrea**

Autorizó a la Universidad Católica de Santiago de Guayaquil a la **publicación** en la biblioteca de la institución del **componente práctico del examen complejo, Análisis comparativo de los operadores de detección de bordes en el procesamiento de imágenes utilizando MatLab** cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, a los 10 días del mes de marzo del año 2021

EL AUTOR



LEÓN FALCONÍ, SHIRLEY ANDREA

# REPORTE DE URKUND

**URKUND** Fernando Palacios Meléndez (edwin\_palacios)

**Documento** León Shirley.docx (D97411748)  
**Presentado** 2021-03-06 00:35 (-05:00)  
**Presentado por** fernandopm23@hotmail.com  
**Recibido** edwin.palacios.ucsg@analysis.orkund.com  
**Mensaje** Revisión EC Shirley León [Mostrar el mensaje completo](#)

1% de estas 17 páginas, se componen de texto presente en 3 fuentes.

| Lista de fuentes | Bloques | Categoría | Enlace/nombre de archivo  |
|------------------|---------|-----------|---|
|                  |         |           | Ordóñez Páez (Electrónica).docx   |
|                  |         |           | <a href="https://es.wikipedia.org/wiki/Detector_de_bordes">https://es.wikipedia.org/wiki/Detector_de_bordes</a>   |
|                  |         |           | <a href="https://core.ac.uk/download/pdf/270313891.pdf">https://core.ac.uk/download/pdf/270313891.pdf</a>         |
|                  |         |           | <a href="https://es.mathworks.com/help/images/ref/edge...">https://es.mathworks.com/help/images/ref/edge...</a>   |
|                  |         |           | <a href="https://digitum.um.es/xmlui/bitstream/10201/86...">https://digitum.um.es/xmlui/bitstream/10201/86...</a> |
|                  |         |           | submission.pdf  |
|                  |         |           | <a href="https://es.paz.wiki/wiki/Canny_edge_detector">https://es.paz.wiki/wiki/Canny_edge_detector</a>           |

0 Advertencias. Reiniciar Exportar Compartir

UNIVERSIDAD CATÓLICA DE SANTIAGO DE GUAYAQUIL FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TEMA: Análisis comparativo de los operadores de detección de bordes en el procesamiento de imágenes utilizando MatLab

AUTOR: León Falconí, Shirley Andrea

Componente práctico del examen complejo previo a la obtención del grado de INGENIERO EN TELECOMUNICACIONES

**REVISOR**



**M. Sc. Romero Paz, Manuel de Jesús**

## **DEDICATORIA**

A mis queridos Padres por su amor, valores, sacrificio y esfuerzo en todo momento, logrando convertirme en la persona que soy.

A mi hija por ser la inspiración de terminar este proyecto

Gracias por confiar en mí, este es el resultado de todo lo que han sembrado en mí.

**EL AUTOR**

**LEÓN FALCONÍ, SHIRLEY ANDREA**

## **AGRADECIMIENTO**

En primer lugar, a Dios, por darme la vida porque cada amanecer es un nuevo comienzo y por permitirme el privilegio de cumplir este gran sueño.

A mis padres Galo León y Dolores Falcón que son el pilar fundamental en mi vida. Su ayuda incondicional hacia mí durante toda mi carrera profesional.

A los Docentes que, con sus enseñanzas, consejos, dedicación, paciencia y tiempo, fueron un ejemplo a seguir para mi desarrollo profesional.

MI esposo Marcos Tapia Onofre por el estímulo y apoyo incondicional en todo momento

EL AUTOR

LEÓN FALCONÍ, SHIRLEY ANDREA



**UNIVERSIDAD CATÓLICA  
DE SANTIAGO DE GUAYAQUIL  
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO  
CARRERA DE TELECOMUNICACIONES**

**TRIBUNAL DE SUSTENTACIÓN**

f.   
M. Sc. **ROMERO PAZ, MANUEL DE JESÚS**  
DECANO

f.   
M. Sc. **HERAS SÁNCHEZ, MIGUEL ARMANDO**  
DIRECTOR DE CARRERA

f.   
M. Sc. **PALACIOS MELÉNDEZ, EDWIN FERNANDO**  
OPONENTE



## ÍNDICE GENERAL

|   |      |
|---|------|
| Índice de Figuras .....   | XI   |
| Índice de Tablas .....  | XII  |
| Resumen .....   | XIII |
| Capítulo 1: Descripción general del componente práctico .....               | 2    |
| 1.1. Introducción.....  | 2    |
| 1.2. Objetivo General. ....   | 3    |
| 1.3. Objetivos Específicos. ....  | 4    |
| Capítulo 2: Fundamentación teórica. ....                                    | 5    |
| 2.1. Introducción de la detección de bordes. ....                           | 5    |
| 2.2. Técnicas de detección de bordes tradicionales. ....                    | 6    |
| 2.2.1. Operador Sobel para detección de bordes.....                         | 7    |
| 2.2.2. Operador Prewitt para detección de bordes.....                       | 9    |
| 2.2.3. Operador Roberts para detección de bordes. ....                      | 9    |
| 2.2.4. Operador Laplaciano de Gauss (LoG) para detección de<br>bordes. .... | 10   |
| 2.2.5. Operador Canny para detección de bordes. ....                        | 10   |
| 2.3. Análisis del software de simulación MatLab. ....                       | 11   |
| 2.3.1. El entorno de desarrollo.....  | 12   |
| 2.3.2. Ventajas del entorno MatLab contra C.....                            | 13   |
| 2.3.3. Desventajas del entorno MatLab contra C. ....                        | 14   |
| 2.3.4. Diferencias entre entorno MatLab y C.....                            | 14   |
| Capítulo 3: Diseño. Simulación y Resultados.....                            | 15   |
| 3.1. Breve descripción del desarrollo del componente práctico. ....         | 15   |
| 3.2. Revisión del estado del arte. ....                                     | 15   |
| 3.3. Procesamiento de imágenes para la detección de bordes. ....            | 16   |
| 3.4. Metodología propuesta para la detección de bordes. ....                | 17   |

|        |  |    |
|--------|--|----|
| 3.5.   | Implementación de operadores de detección de bordes utilizando MatLab. ....                    | 18 |
| 3.5.1. | Práctica 1: Aplicación manual del filtro de Sobel para la detección de bordes de monedas. .... | 19 |
| 3.5.2. | Práctica 2: Aplicación de detección de bordes a una imagen de gradiente binarizada. ....       | 20 |
| 3.5.3. | Práctica 3: Aplicación de operadores de detección de bordes. ....                              | 21 |
| 3.5.4. | Práctica 4: Detección de bordes para contar monedas. ....                                      | 24 |
| 3.6.   | Discusión de resultados para diferentes imágenes en la detección de bordes. ....               | 25 |
| 3.6.1. | Número de bordes detectados. ....  | 26 |
| 3.6.2. | Relación de rendimiento (PR). ....   | 26 |
| 3.6.3. | Medida de precisión de pruebas. ....   | 27 |
| 3.6.4. | Relación señal a ruido pico (PSNR). ....   | 28 |
|        | Conclusiones .....   | 30 |
|        | Recomendaciones. ....  | 32 |
|        | Bibliografía.....  | 33 |

## Índice de Figuras

### Capítulo 2:

|  |    |
|--|----|
| Figura 2. 1: Aplicación de operadores Sobel y Gaussiano en imágenes odontológicas (rayos X). .....                         | 6  |
| Figura 2. 2: (a) imagen original, y detección de bordes utilizando los operadores (b) Sobel, (c) Prewitt y (d) Canny. .... | 7  |
| Figura 2. 3: Imagen original de una carretera. ....  | 7  |
| Figura 2. 4: Núcleos de convolución de Sobel. ....   | 8  |
| Figura 2. 5: Umbral del operador Sobel de manera automática. ....  | 8  |
| Figura 2. 6: Núcleos de convolución de Prewitt. ....   | 9  |
| Figura 2. 7: Núcleos de convolución de Roberts. ....   | 10 |
| Figura 2. 8: Umbral del operador Roberts de manera automática. ....  | 10 |
| Figura 2. 9: Núcleos de convolución de Canny. ....   | 11 |
| Figura 2. 10: Umbral del operador Canny de manera automática. ....   | 11 |
| Figura 2. 11: Interfaz de inicio del software MatLab R2018a. ....  | 13 |

### Capítulo 3:

|   |    |
|---|----|
| Figura 3. 1: Metodología propuesta para el componente práctico. ....                                      | 18 |
| Figura 3. 2: Resultado obtenido de la práctica 1. ....  | 19 |
| Figura 3. 3: Aplicación manual de filtrado de Sobel mediante gradientes. ...                              | 20 |
| Figura 3. 4: Imagen de bordes detectado en monedas mediante gradiente total y binarización. ....          | 21 |
| Figura 3. 5: Resultados obtenidos de los operadores de detección de bordes en monedas. ....               | 22 |
| Figura 3. 6: Resultados obtenidos de los operadores de detección de bordes en monedas para imagen T. .... | 23 |
| Figura 3. 7: Conteo de monedas. ....  | 25 |

## Índice de Tablas

### Capítulo 3:

|   |    |
|---|----|
| Tabla 3. 1: Comparativa de los operadores de detección de bordes.....                 | 24 |
| Tabla 3. 2: Comparativa del número de bordes detectados. ....                         | 26 |
| Tabla 3. 3: Comparativa de la relación de rendimiento de la detección de bordes. .... | 27 |
| Tabla 3. 4: Comparativa de la medida de precisión en la detección de bordes. ....     | 28 |
| Tabla 3. 5: Comparativa de la medida de precisión en la detección de bordes. ....     | 29 |

## Resumen

El objetivo del presente estudio es comparar varios operadores de detección de bordes y analizar su desempeño y también el desempeño de tales técnicas se lleva a cabo para diferentes imágenes utilizando el software de simulación MatLab R2018a. En el capítulo 2 se presenta una base teórica y matemática para la detección de bordes. En el capítulo 3 se presenta el enfoque propuesto y proporciona los resultados experimentales y la discusión de los resultados. Se ha demostrado que el algoritmo de detección de bordes de Canny funciona mejor que todos estos operadores en casi todos los escenarios. La evaluación de las imágenes mostró que en condiciones ruidosas Canny, LoG (Laplaciano de Gauss), Robert, Prewitt, Sobel exhiben un mejor desempeño, respectivamente. Se ha observado que el algoritmo de detección de bordes de Canny es computacionalmente más caro en comparación con el operador de LoG (Laplaciano de Gauss), Sobel, Prewitt y Robert.

**Palabras claves:** Detección, Bordes, Filtros, Algoritmo, Operadores, Procesamiento

## Capítulo 1: Descripción general del componente práctico

### 1.1. Introducción.

Las imágenes contienen información diferente sobre el objeto o la escena, como el tamaño, la forma, el color y su orientación. El resultado de cambios en la luz, el color, la sombra y la textura se llama borde. Cuando una imagen se corrompe por el ruido, la detección de bordes es muy difícil de obtener y el proceso tomará tiempo. Los objetos se pueden encontrar bien planificados y ejecutados fácilmente solo cuando los bordes de las imágenes se pueden reconocer de manera eficiente.

La detección de bordes es una de las principales técnicas utilizadas en el procesamiento de imágenes. La aplicación de la detección de bordes utilizada en el procesamiento de imágenes es amplia, como el reconocimiento de patrones, el reconocimiento de objetos, el procesamiento de imágenes médicas y el análisis de movimiento, etc. (Kumari et al., 2020)

El punto en el que el brillo de la imagen cambia bruscamente se organiza típicamente en un conjunto de segmentos de líneas curvas denominados bordes. La detección de bordes es el nombre de un conjunto de métodos matemáticos que tienen como objetivo identificar puntos en una imagen digital en los que el brillo de la imagen cambia bruscamente y tiene discontinuidades. El mismo problema de encontrar discontinuidades en la señal 1D se conoce como detección de paso y el problema de encontrar discontinuidades de señal a lo largo del tiempo se conoce como detección de cambio. La detección de bordes es una herramienta fundamental en el procesamiento de imágenes, versión de máquina y versión de computadora, particularmente en áreas de detección de características y extracción de características

Aunque se han desarrollado muchos métodos de evaluación de detección de bordes en los últimos años. Por ejemplo, Lakhani et al., (2016) realiza el análisis de técnicas de detección de bordes para extracción de características en radiografías dentales. Dagar & Dahiya, (2020) utilizan la

técnica de detección de bordes con optimización de enjambre de partículas binarias, aunque este sigue siendo un problema desafiante y sin resolver.

El mayor desafío proviene de la dificultad de elegir una medida de rendimiento adecuada de los resultados de detección de bordes. En la mayoría de las aplicaciones, la detección de bordes se utiliza como un paso de preprocesamiento para extraer algunas características de límites de bajo nivel, que luego se introducen en otros pasos de procesamiento, como la búsqueda y el reconocimiento de objetos. Por lo tanto, el rendimiento de la detección de bordes es difícil de definir sin incorporarlo en determinadas aplicaciones.

Sin embargo, si la detección de bordes se evalúa basándose en el rendimiento de una aplicación especial dicha evaluación puede no ser aplicable a otras aplicaciones. Esto introduce un dilema bien conocido inherente a la evaluación de detección de bordes: la evaluación de propósito general es difícil de definir, mientras que la evaluación basada en una aplicación específica reduce la generalidad del método de evaluación.

Los bordes caracterizan los límites y, por lo tanto, son un problema de fundamental importancia en el procesamiento de imágenes. La detección de bordes de imagen reduce significativamente la cantidad de datos y filtra la información inútil, al tiempo que conserva las importantes propiedades estructurales de una imagen. Dado que la detección de bordes está a la vanguardia del procesamiento de imágenes para la detección de objetos, es fundamental tener una buena comprensión de los algoritmos de detección de bordes.

## **1.2. Objetivo General.**

Realizar el análisis comparativo de los operadores de detección de bordes en el procesamiento de imágenes que son implementados en el software MatLab.

### **1.3. Objetivos Específicos.**

- a. Describir las bases teóricas de la detección de bordes y de los métodos de operación para su detección.
- b. Implementar prácticas experimentales de algoritmos de operadores de detección en MatLab
- c. Analizar comparativamente los diferentes operadores de detección de bordes en MatLab.



## Capítulo 2: Fundamentación teórica.

### 2.1. Introducción de la detección de bordes.

La detección de bordes es un método de extracción de características muy importante que se ha utilizado ampliamente en muchas aplicaciones de procesamiento de imágenes y visión por computadora. La idea básica de la mayoría de los detectores de bordes disponibles es localizar cierta información local sobre los límites del objeto en una imagen mediante la creación de umbrales y esqueletización del mapa de variación de intensidad de píxeles.

Desde el primer trabajo de Julez en 1959, se han desarrollado una gran cantidad de detectores de bordes desde diferentes perspectivas. Entonces, una pregunta muy natural e importante es: ¿qué detector de bordes y qué ajustes de parámetros del detector pueden producir mejores resultados de detección de bordes? Esto motiva fuertemente el desarrollo de una forma general y sistemática de evaluar los resultados de detección de bordes.

Los métodos de evaluación de detección de bordes anteriores se pueden clasificar de varias maneras. Primero, pueden clasificarse como métodos subjetivos y objetivos. El primero utiliza la observación y la decisión visual humana para evaluar el rendimiento de la detección de bordes. Dada la inconsistencia inherente en la percepción humana, los resultados de la evaluación subjetiva pueden mostrar una gran variación para diferentes observadores. En los métodos objetivos, las medidas cuantitativas se definen basándose únicamente en imágenes y los resultados de detección de bordes.

En segundo lugar, los métodos de evaluación de detección de bordes se pueden clasificar de acuerdo con su requisito de la realidad del terreno. Con la realidad del terreno, la detección de bordes se puede evaluar cuantitativamente de una manera más creíble. Sin la realidad básica, se suele utilizar cierta información de coherencia local para medir el rendimiento. En tercer lugar, los métodos de evaluación de detección de bordes se pueden clasificar en función de las imágenes de prueba: métodos basados en

imágenes sintéticas y métodos basados en imágenes reales. Se puede encontrar una discusión más detallada sobre varios detectores de bordes y métodos de evaluación de detección de bordes.

Por ejemplo, la figura 2.1 muestra la detección de bordes en aplicaciones odontológicas realizado por (Lakhani et al., 2016). En este artículo, se han discutido los problemas obtenidos al utilizar diferentes metodologías de extracción de bordes de procesamiento de imágenes en imágenes de rayos X dentales desalineadas.

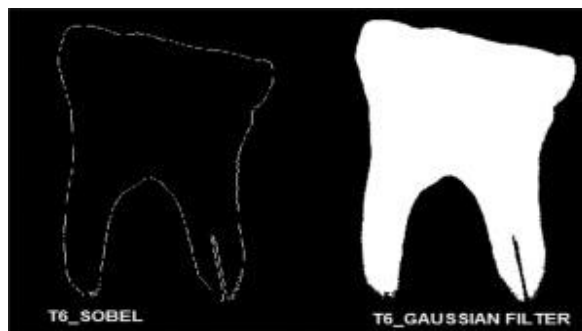


Figura 2. 1: Aplicación de operadores Sobel y Gaussiano en imágenes odontológicas (rayos X).

Fuente: (Kumari et al., 2020)

Las técnicas convencionales de extracción de bordes, es decir, Sobel y Prewitt, parecen ser inadecuadas para obtener con éxito las características de los bordes que cuando se aplica el filtro gaussiano en la imagen de Sobel, se formó una imagen mucho más clara y, por lo tanto, la extracción de características ahora se puede aplicar fácilmente sobre la imagen gaussiana así formada, que se puede comparar fácilmente con la imagen del diente normal para identificar el problema real en el diente. Además, se encontró que el suavizado y el afilado de imágenes tienen un mayor efecto en el diagnóstico de enfermedades en imágenes médicas.

## 2.2. Técnicas de detección de bordes tradicionales.

En esta sección se describe brevemente los métodos de detección de bordes clásicos. tales como, los operadores de Sobel (véase figura 2.2 b), Prewitt (véase figura 2.2 c), Roberts, Laplaciano de Gauss (LoG), Canny (véase figura 2.2 d), Rothwell y Edison. Aunque para la implementación

práctica en el capítulo 3 no se consideran los detectores de bordes Rothwell y Edison para la evaluación.

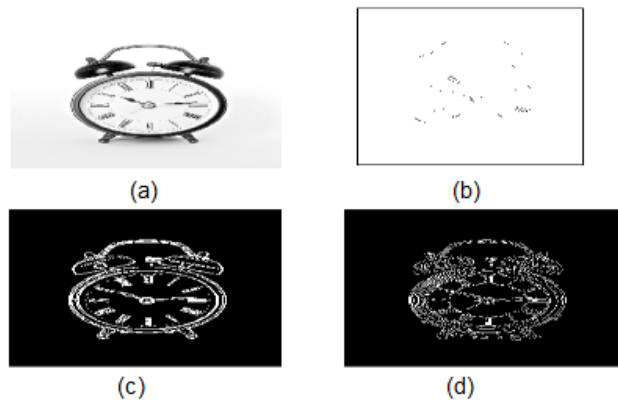


Figura 2. 2: (a) imagen original, y detección de bordes utilizando los operadores (b) Sobel, (c) Prewitt y (d) Canny.

Fuente: (Kumari et al., 2020)

Cada detector tiene su propia configuración de parámetros. Además, en el capítulo 3 se utiliza las funciones de la caja de herramientas de procesamiento de imágenes en Matlab para los detectores de bordes Sobel, LoG y Canny. La figura 2.3 muestra la imagen de una carretera descrita en el trabajo de Talib et al., (2013) en donde aplican los operadores de Sobel, Roberts y Canny en las secciones 2.2.1, 2.2.3 y 2.2.5.



Figura 2. 3: Imagen original de una carretera.

Fuente: (Talib et al., 2013)

### 2.2.1. Operador Sobel para detección de bordes.

Cuando se duplica el peso en los píxeles centrales, para ambas plantillas de Prewitt, se obtiene el famoso operador de detección de bordes de Sobel que, nuevamente, consta de dos máscaras para determinar el borde en forma vectorial. El operador de Sobel fue el operador de detección de bordes más

popular hasta el desarrollo de técnicas de detección de bordes con una base teórica. Resultó popular porque proporcionó, en general, un mejor rendimiento que otros operadores contemporáneos de detección de bordes, como el operador Prewitt. (Nixon & Aguado, 2012)

El detector de borde de sobel calcula el gradiente utilizando las diferencias discretas entre filas y columnas de una vecindad de una matriz de 3X3. El operador de Sobel se basa en convolucionar la imagen con un filtro pequeño, separable y de valor entero. Las plantillas para el operador Sobel se pueden encontrar en la figura 2.4.

|    |   |    |
|----|---|----|
| -1 | 0 | +1 |
| -2 | 0 | +2 |
| -1 | 0 | +1 |

G<sub>x</sub>

|    |    |    |
|----|----|----|
| +1 | +2 | +1 |
| 0  | 0  | 0  |
| -1 | -2 | -1 |

G<sub>y</sub>

Figura 2. 4: Núcleos de convolución de Sobel.  
Fuente: (Kekre & Gharge, 2010; Othman et al., 2009)

Una ligera variación de estas dos ecuaciones utiliza un peso de 2 en el coeficiente central que utiliza el operador de sobel. Las ecuaciones relacionadas con el operador de Sobel se definen como, (Gill et al., 2012).

$$G_x = (z_3 + 2z_6 + z_9) - (z_1 + z_4 + z_7)$$

$$G_y = (z_1 + 2z_2 + z_3) - (z_7 + 2z_8 + z_9)$$

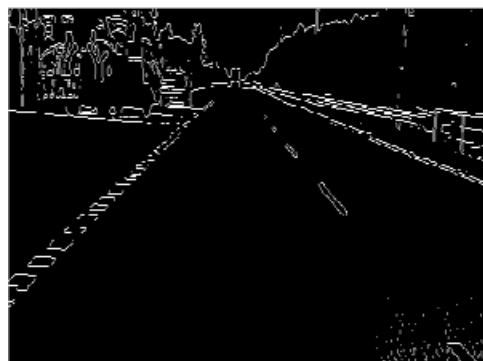


Figura 2. 5: Umbral del operador Sobel de manera automática.  
Fuente: (Talib et al., 2013)

Por ejemplo, la figura 2.5 muestra una aplicación práctica del operador de detección de Sobel de manera automática tomada del trabajo propuesto por Talib et al., (2013).

### 2.2.2. Operador Prewitt para detección de bordes.

Prewitt es un operador de diferenciación discreto que calcula una aproximación del gradiente de la función de intensidad de la imagen en cada punto de la imagen. El resultado del operador Prewitt es el vector de gradiente correspondiente de la norma de este vector. El operador Prewitt se basa en convolucionar la imagen con un filtro pequeño, separable y de valor entero en dirección horizontal y vertical y, por lo tanto, es relativamente caro en términos de cálculo. Fue desarrollado por M.S Prewitt. (Priyam et al., 2016)

Las máscaras de detección de bordes de operador de Prewitt son uno de los métodos más antiguos y mejor comprendidos para detectar bordes en imágenes. El detector de bordes de Prewitt utiliza la siguiente máscara (ver figura 2.6) para aproximar digitalmente las primeras derivadas  $G_x$  y  $G_y$ .

|       |    |    |       |   |    |
|-------|----|----|-------|---|----|
| -1    | -1 | -1 | -1    | 0 | +1 |
| 0     | 0  | 0  | -1    | 0 | +1 |
| +1    | +1 | +1 | -1    | 0 | +1 |
| $G_x$ |    |    | $G_y$ |   |    |

Figura 2. 6: Núcleos de convolución de Prewitt.  
Fuente: (Kekre & Gharge, 2010; Othman et al., 2009)

Las ecuaciones relacionadas con el operador Prewitt se definen como, (Gill et al., 2012)

$$G_x = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$
$$G_y = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

### 2.2.3. Operador Roberts para detección de bordes.

En la detección de bordes de Robert, los bordes verticales y horizontales se resaltan individualmente y luego se unen para la detección de bordes resultante. El detector de bordes de Roberts utiliza las siguientes máscaras (ver figura 2.5) para aproximar digitalmente las primeras derivadas como diferencias entre píxeles adyacentes. La forma de implementar la derivada de primer orden es utilizar los operadores de gradiente cruzado de Robert. Es la forma más sencilla de utilizar 2-D con preferencia diagonal.

Las ecuaciones para el operador de Robert se definen como, (Gill et al., 2012).

$$G_x = z_9 - z_5$$

$$G_y = z_8 - z_6$$

|    |   |
|----|---|
| -1 | 0 |
| 0  | 1 |

$G_x$

|   |    |
|---|----|
| 0 | -1 |
| 1 | 0  |

$G_y$

Figura 2. 7: Núcleos de convolución de Roberts.

Fuente: (Gill et al., 2012)

La figura 2.8 muestra una aplicación práctica del operador de detección de Roberts de manera automática tomada del trabajo propuesto por Talib et al., (2013).

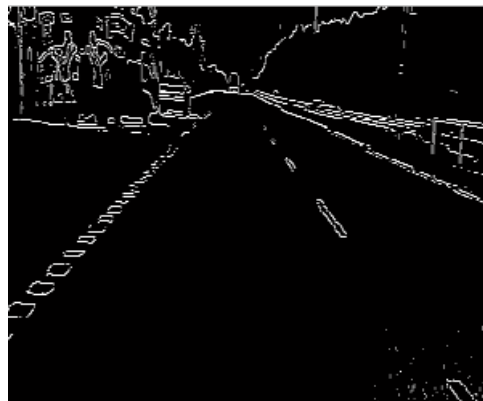


Figura 2. 8: Umbral del operador Roberts de manera automática.

Fuente: (Talib et al., 2013)

#### 2.2.4. Operador Laplaciano de Gauss (LoG) para detección de bordes.

Este detector encuentra bordes buscando cruces por cero después de filtrar  $f(x, y)$  con un filtro Laplaciano de Gauss. En este método, el filtrado gaussiano se combina con el laplaciano para descomponer la imagen donde la intensidad varía para detectar los bordes de manera efectiva. Encuentra el lugar correcto de los bordes y prueba un área más amplia alrededor del píxel.

#### 2.2.5. Operador Canny para detección de bordes.

La detección inteligente de bordes es un algoritmo de varias etapas para detectar una amplia gama de bordes en las imágenes. Este detector

encuentra bordes buscando máximos locales del gradiente de  $f(x,y)$ . El gradiente se calcula utilizando la derivada de un filtro gaussiano. El método utiliza dos umbrales para detectar bordes fuertes y débiles e incluye los bordes débiles en la salida solo si están conectados a bordes fuertes.

En el método de Canny, los bordes se encuentran buscando los máximos locales del gradiente de la imagen ( $I$ ). La derivada de un filtro gaussiano se usa para calcular el gradiente. Este método utiliza dos umbrales, umbral bajo y umbral alto para detectar bordes fuertes y débiles. Este método tiene más supresión de ruido que otros y es más probable que detecte verdaderos bordes débiles. La figura 2.9 muestra los núcleos de convolución de Canny.

|    |    |    |    |   |   |
|----|----|----|----|---|---|
| 1  | 2  | 1  | -1 | 0 | 1 |
| 0  | 0  | 0  | -2 | 0 | 2 |
| -1 | -2 | -1 | -1 | 0 | 1 |

Figura 2. 9: Núcleos de convolución de Canny.

Fuente: (Gill et al., 2012)

La figura 2.10 muestra una aplicación práctica del operador de detección de Canny de manera automática tomada del trabajo propuesto por Talib et al., (2013).

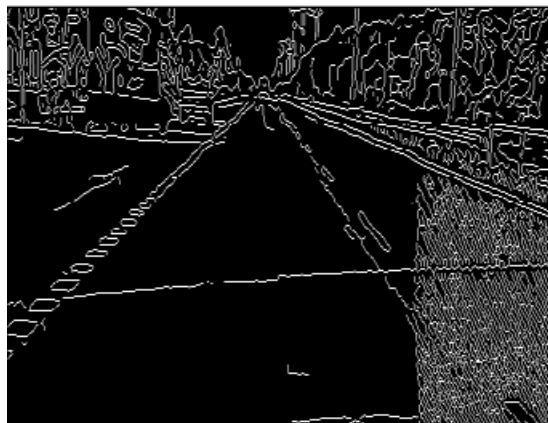


Figura 2. 10: Umbral del operador Canny de manera automática.

Fuente: (Talib et al., 2013)

### 2.3. Análisis del software de simulación MatLab.

Algunas investigaciones en el procesamiento de imágenes o de visión por computadora utilizan la biblioteca OpenCV, que tiene una colección de

funcionalidades de procesamiento de imágenes muy poderosas. Sin embargo, el software MatLab también proporciona un buen conjunto de herramientas conocidas como "Herramientas de Procesamiento de Imágenes", que ha sido suficiente para el propósito del componente práctico. Además, MatLab tiene algunas capacidades de carga de imágenes integradas que permiten una carga y manipulación de imágenes fácil y rápida.

Por ejemplo, la función "imread ()" de MatLab lee imágenes de muchos formatos comunes y las almacena en una matriz de 2x2 píxeles para cada canal de color. Esta funcionalidad nos permitió llegar directamente al trabajo de procesamiento de imágenes sin tener que desarrollar métodos de procesamiento previo para las imágenes que se utilizan en el capítulo 3.

Se utiliza una laptop (hardware, Intel Core i7 8th Generation) bastante modesto para realizar el procesamiento e incluso los métodos robustos en el peor de los casos resultaron en el capítulo 3 no tardar más de unos minutos en su procesamiento.

### **2.3.1. El entorno de desarrollo.**

El entorno de desarrollo tiene una influencia importante en la implementación de un sistema e incluso puede alterar la salida requerida. Al elegir el entorno para este documento, se pensó en considerar C y MatLab. Estas herramientas pueden manejar problemas similares y MatLab incluso puede interactuar con C.

Sin embargo, debido a la naturaleza compleja de las imágenes requeridas y la incapacidad del lenguaje de programación para manejar de manera eficiente manipulaciones matriciales complejas, se utilizó MatLab R2018a (ver figura 2.7). Además, MatLab también tiene algunas funciones integradas que facilitan la implementación de ciertas funciones y la hacen menos costosa.



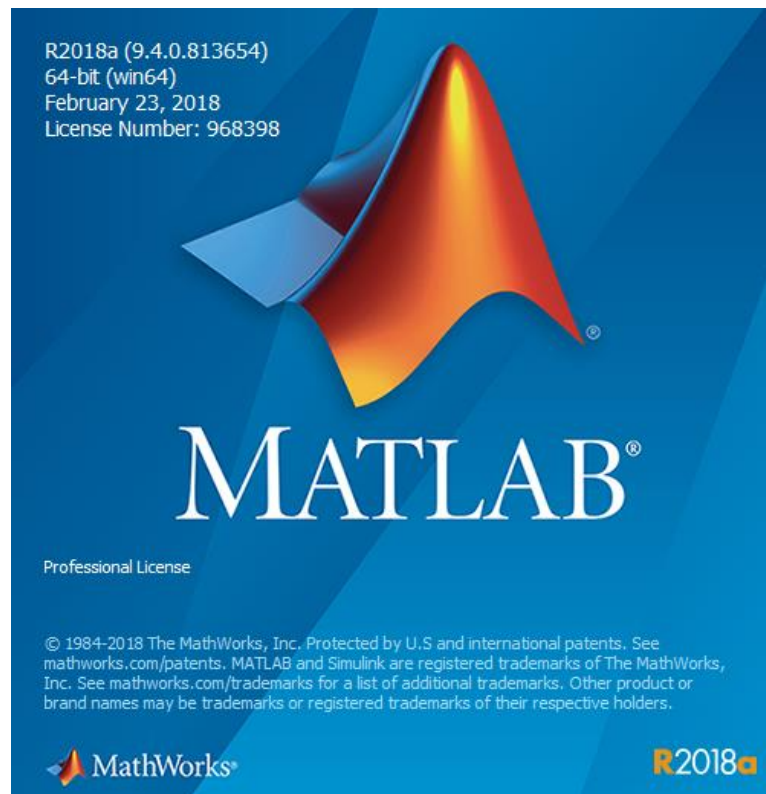


Figura 2. 11: Interfaz de inicio del software MatLab R2018a.  
Elaborado por: Autor.

### 2.3.2. Ventajas del entorno MatLab contra C.

A continuación, se describen las ventajas de utilizar el entorno de simulación MatLab.

- Como lenguaje interpretado, MatLab permite al programador una mayor flexibilidad en términos de codificación y actualización de datos mientras el programa está en ejecución.
- Es un lenguaje basado en funciones, lo que facilita el desarrollo y la documentación de programas.
- El sistema opera con matrices en lugar de cantidades escalares, lo que implica muchas operaciones matemáticas y es especialmente útil para el procesamiento de imágenes.
- No requiere la determinación de tamaño y tipo de variable antes de su uso.
- Con funciones integradas, permite al programador realizar cálculos y recibir resultados visuales.
- MatLab permite el uso de valores indefinidos, como la división por cero. Para el procesamiento de imágenes donde las características

como el gradiente de las líneas verticales tienden a estar indefinidas, esta es una característica muy importante.

### **2.3.3. Desventajas del entorno MatLab contra C.**

A continuación, se describen las ventajas de utilizar el entorno de simulación MatLab.

- C permite códigos que son más eficientes que MATLAB en muchas áreas, especialmente en comparación con el uso de algunas características integradas de MATLAB.
- A diferencia de C, MATLAB puede ser lento con ligeras ineficiencias en la programación.
- Con variables no declaradas, aumentan las posibilidades de que los programadores cometan errores.

### **2.3.4. Diferencias entre entorno MatLab y C.**

Algunas de las razones consideradas al seleccionar MATLAB en lugar de C para el proyecto se dan a continuación:

- Aunque C presenta una mayor eficiencia en el tiempo, se prefiere MATLAB durante el desarrollo.
- Sin la necesidad de declarar variables por adelantado, MATLAB permite una codificación rápida.
- MatLab no tiene restricciones en las estructuras de datos a diferencia de C, y puede realizar ciertos cálculos de matriz utilizando comandos predefinidos.
- MATLAB tiene funciones y bibliotecas integradas para referencia que son más confiables que las muchas bibliotecas disponibles de C

## Capítulo 3: Diseño. Simulación y Resultados.

### 3.1. Breve descripción del desarrollo del componente práctico.

El presente capítulo se desarrollan algoritmos de aplicaciones prácticas del procesamiento de imágenes, tales como la detección de bordes.

### 3.2. Revisión del estado del arte.

En la búsqueda de información, se pudo encontrar diferentes trabajos publicados en revistas arbitradas relacionadas al procesamiento de imágenes. A continuación, se presenta el estado del arte o revisión de literatura de tres trabajos que empelan la detección de bordes.

1. Tong et al., (2011) presentan una nueva técnica de detección de bordes basada en algoritmos genéticos (*Genetic Algorithm, GA*) y operador de sobel. La detección de bordes de Sobel incorporada en DSP Builder se usa primero para determinar los límites de los objetos dentro de una imagen. Luego, el algoritmo genético que utiliza SOPC Builder propone un nuevo algoritmo de umbral para el procesamiento de imágenes. Finalmente, el rendimiento de la nueva técnica de detección de bordes basada en los mejores enfoques de umbral en el software DSP Builder y Quartus II se compara tanto cualitativa como cuantitativamente con el operador de Sobel único. Demostraron que la nueva técnica de detección de bordes funciona muy bien en términos de resistencia al ruido, capacidad de búsqueda de bordes y calidad de la imagen de borde final.
2. Jin-Yu et al., (2009) presentan el operador de detección de bordes de Sobel y su algoritmo mejorado se analizan por primera vez en términos de umbral óptimo. Luego, basado en algoritmos genéticos y un operador Sobel mejorado, se propone un nuevo algoritmo de umbral automático para el procesamiento de imágenes. Finalmente, los experimentos de detección de bordes de dos imágenes reales se realizan mediante dos algoritmos. Los resultados del experimento comparativo muestran que el nuevo algoritmo de umbral automático es muy eficaz. Los resultados también son mejores que los métodos clásicos de Otsu.

3. Feng et al., (2017) proponen un nuevo método de detección de bordes basado en el algoritmo de Canny. En primer lugar, para procesarlo se adopta el filtrado de media y filtrado basado en el método de distancia euclidiana; en segundo lugar, utilizan el algoritmo de Freichen para calcular la amplitud del gradiente; finalmente, utilizando el algoritmo de Otsu para calcular la operación de amplitud de gradiente parcial para obtener imágenes del valor de los umbrales, luego encuentran el promedio de todos los umbrales calculados, la mitad del promedio es el valor de umbral alto. Los resultados muestran que este nuevo método puede suprimir eficazmente las perturbaciones por ruido, mantener la información de los bordes y también mejora la precisión de detección de los bordes.

### **3.3. Procesamiento de imágenes para la detección de bordes.**

La detección de bordes es un problema fundamental de la visión por computadora y del procesamiento de imágenes. Ha sido un tema de gran preocupación en la segmentación de imágenes y para los investigadores. El propósito de la segmentación de imágenes es dividir una imagen en regiones significativas con respecto a una aplicación particular donde los bordes de las imágenes digitales son áreas con fuertes contrastes de intensidad y un salto en la intensidad de un píxel al siguiente puede crear una gran variación en la calidad de la imagen y segmentación de imagen.

Para que los sistemas de procesamiento de imágenes y visión por computadora interpreten una imagen, primero deben poder detectar los bordes de cada objeto en la imagen. Hay varios operadores de detección de bordes disponibles para la segmentación de imágenes y la extracción de límites de objetos de imágenes digitales. Cada operador está diseñado para ser sensible a ciertos tipos de bordes. Entre ellos, Sobel, Roberts, Prewitt, LoG y Canny son los operadores más importantes. La geometría del operador determina una dirección característica en la que es más sensible a los bordes.

La presencia de ruido es un problema para la segmentación de imágenes. Las imágenes son muy propensas a verse afectadas por una serie

de ruidos como el ruido gaussiano, el ruido de Rayleigh, el ruido de impulso y el ruido Speckle. Se ha encontrado que, en presencia de ruido, el operador de detección de bordes Canny ha producido la mejor vista segmentada subjetiva de la imagen de prueba en lo que respecta a la apariencia y la localización de los límites del objeto, luego de los operadores de Sobel, Roberts, Prewitt y LoG.

La entropía, que es una medida estadística de aleatoriedad que se puede utilizar para caracterizar la textura de la imagen de entrada, se estudia junto con la relación señal/ruido pico (PSNR), la relación cuadrática media (MSE) y los tiempos de ejecución también se estudian en este componente práctico.

#### **3.4. Metodología propuesta para la detección de bordes.**

En este paso, la imagen preprocesada se segmenta con operadores de detección de bordes. Los objetos se detectan con los cinco operadores de detección de bordes diferentes. Se utilizan cinco operadores de detección de bordes diferentes: Robert, Prewitt, Sobel, LoG y Canny. La metodología propuesta tiene cinco etapas. La figura 3.1 describe un enfoque en forma de diagrama de flujo.

A continuación, se describe cada etapa del método propuesto para el desarrollo de aplicaciones prácticas.

1. Imagen original: la imagen original es una imagen capturada por la cámara digital de alta resolución.
2. Preprocesamiento: en este paso, la imagen capturada por la cámara (imagen en color verdadero) se convierte en una imagen en escala de grises y el ruido se elimina mediante un filtro de mediana 2D.
3. Detección de objetos: en este paso, la imagen preprocesada se segmenta con operadores de detección de bordes. Los objetos se detectan con los cinco operadores de detección de bordes diferentes.

Se utilizan cinco operadores de detección de bordes diferentes: Robert, Prewitt, Sobel, LoG y Canny.

4. Postprocesamiento: la operación de dilatación se ha aplicado en la etapa de posprocesamiento. La operación de dilatación ayuda a engrosar los bordes. Después de este relleno de agujeros se ha realizado lo que mejora la apariencia visual de una imagen.
5. Imagen final: en esta imagen de salida se muestra después de la eliminación de componentes adicionales.

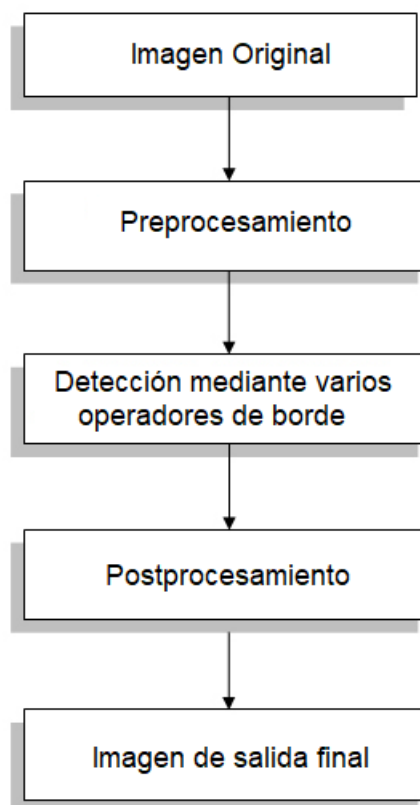


Figura 3. 1: Metodología propuesta para el componente práctico.  
Elaborado por: Autor.

### 3.5. Implementación de operadores de detección de bordes utilizando MatLab.

En esta sección se presenta el desarrollo de algoritmos implementados en el software MatLab utilizando la caja de herramientas de procesamiento de imágenes. La parte práctica fue facilitada por M. Sc. Edwin Palacios que

recibió en sus clases del Doctorado en Ingeniería Eléctrica en ESPOL. El desarrollo de cada uno de los algoritmos y sus resultados se presentan en las siguientes secciones de la sección 3.5.

### 3.5.1. Práctica 1: Aplicación manual del filtro de Sobel para la detección de bordes de monedas.

Para la aplicación manual de detección de bordes mediante los núcleos de convolución de la figura 2.3, primero se procede a leer la imagen “coing.png” mediante la función “imread ()” y realizar la normalización en el intervalo de [0; 1] utilizando la función “im2double ()”. Después, se aplica manualmente el filtro u operador de Sobel ( $G_x$  y  $G_y$ ) utilizando la función “filter2 ()”. Finalmente, se representan los gradientes absolutos ( $|G_x|$  y  $|G_y|$ ) y el valor total:  $|G_x| + |G_y|$

A continuación, se muestra el algoritmo implementado (parte1) en MatLab para la aplicación del operador de Sobel de manera manual. La figura 3.2 muestra la imagen de monedas original y la imagen convertida de intensidad (I) a doble precisión, respectivamente.

```
clear all, close all clc
C1=imread('coins.png'); % leer imagen
C2=im2double(C1); % convertir imagen a doble precisión
figure(1)
subplot(2,2,1); imshow(C1);
xlabel(sprintf('(a) Imagen original \n de coins'))
subplot(2,2,2); imshow(C2);
xlabel(sprintf('(b) Imagen de coins \n con formato double'))
```

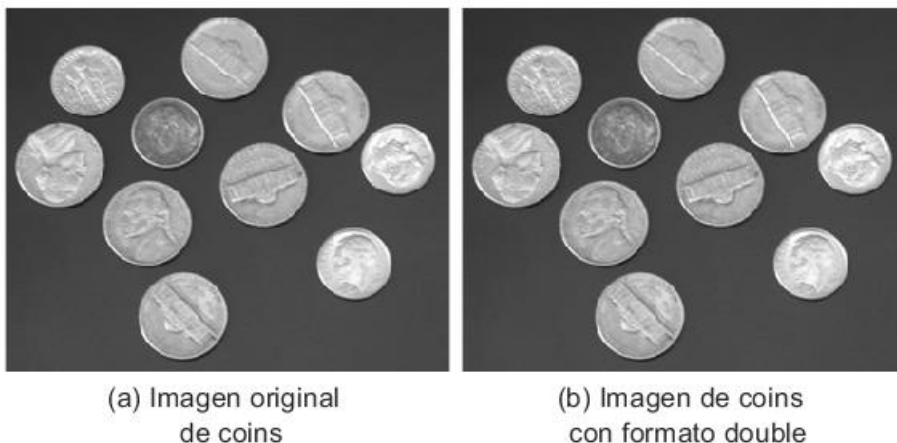


Figura 3. 2: Resultado obtenido de la práctica 1.  
Elaborado por: Autor.

Ahora se implementa la parte2 del algoritmo, en la que se aplica de forma manual el filtro de Sobel y graficar los gradientes absolutos horizontal, vertical y total. En la figura 3.3 se muestran las imágenes con sus respectivos gradientes (horizontal y vertical) y la suma de los gradientes.

```
figure(2)
% matriz Gx para filtrado de Sobel manual
Gx=[-1 0 1;-2 0 2;-1 0 1];
% matriz Gy para filtrado de Sobel manual
Gy=[1 2 1;0 0 0;-1 -2 -1];
subplot(1,3,1);
Cx=abs(filter2((Gx),C2)); imshow(Cx);
xlabel(sprintf('(a): Gradiente \n horizontal (abs)'))
subplot(1,3,2);
Cy=abs(filter2((Gy),C2)); imshow(Cy);
xlabel(sprintf('(b): Gradiente \n vertical (abs)'))
subplot(1,3,3);
Ct=imgradient(Cx,Cy); imshow(Ct);
xlabel(sprintf('(c): Suma de \n los gradientes'))
```

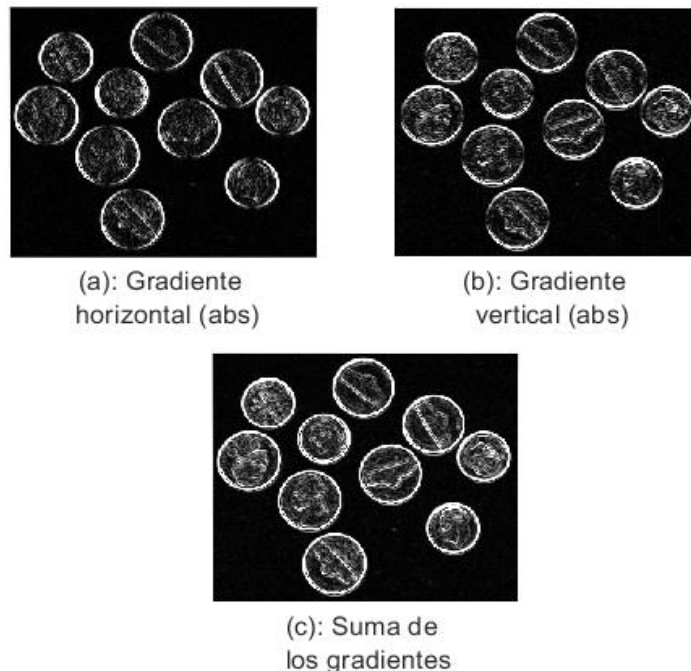


Figura 3. 3: Aplicación manual de filtrado de Sobel mediante gradientes.  
Elaborado por: Autor.

### 3.5.2. Práctica 2: Aplicación de detección de bordes a una imagen de gradiente binarizada.

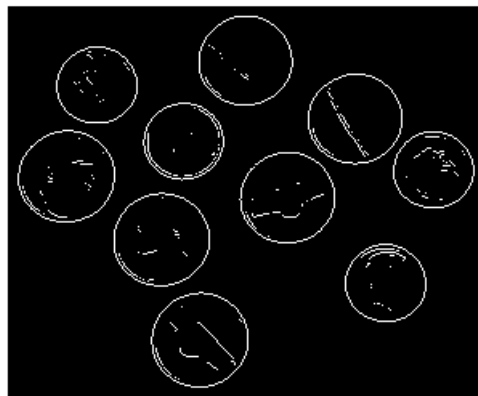
Para obtener los “bordes” o “contornos” se debe binarizar la imagen de gradientes. La función “edge ()” en Matlab realiza el proceso (gradiente total +



binarización) de detectar bordes de una imagen. Es decir, que devuelve una imagen binaria (BW) que contiene 1's donde la función encuentra bordes en la imagen de entrada I y 0's en otros lugares. Por defecto, edge utiliza el método de detección de bordes de Sobel.

Por lo tanto, a partir de la imagen de grises (coins o monedas), se obtienen los “bordes” con: B=edge (A, 'sobel'), detecta bordes en la imagen I utilizando el algoritmo de detección de bordes especificado por el método. A continuación, se implementa el algoritmo en Matlab para la aplicación directa de la función “edge ()” que detecta bordes. La figura 3.4 muestra la imagen de monedas con sus bordes detectados.

```
figure(3)
B=edge(C2,'sobel');
imshow(B);
xlabel(sprintf('Contornos de coins \n usando edge()'))
```



Contornos de coins  
usando edge()

Figura 3. 4: Imagen de bordes detectado en monedas mediante gradiente total y binarización.

Elaborado por: Autor.

### 3.5.3. Práctica 3: Aplicación de operadores de detección de bordes.

En esta sección se sigue trabajando con la detección de bordes de la misma imagen (monedas) pero utilizando los operadores de detección. La parte teórica de estos cinco métodos de operación para detección de bordes se describió en el capítulo 2. A continuación, se muestra el algoritmo implementado en MatLab donde se especifica cinco imágenes con sus respectivos bordes detectados mediante Sobel, Canny, Prewitt, LoG y Roberts, tal como se muestra en la figura 3.5.

```

figure(4)
subplot(2,3,1);
C_1=edge(C2,'sobel'); imshow(C_1);
xlabel(sprintf('(a): Bordes de coins \n usando filtro SOBEL'))
subplot(2,3,2);
C_2=edge(C2,'canny'); imshow(C_2);
xlabel(sprintf('(b): Bordes de coins \n usando filtro CANNY'))
subplot(2,3,3);
C_3=edge(C2,'prewitt'); imshow(C_3);
xlabel(sprintf('(c): Bordes de coins \n usando filtro PREWITT'))
subplot(2,3,4);
C_4=edge(C2,'log'); imshow(C_4);
xlabel(sprintf('(d): Bordes de coins \n usando filtro LoG'))
subplot(2,3,5);
C_4=edge(C2,'roberts'); imshow(C_4);
xlabel(sprintf('(e): Bordes de coins \n usando filtro ROBERTS'))

```

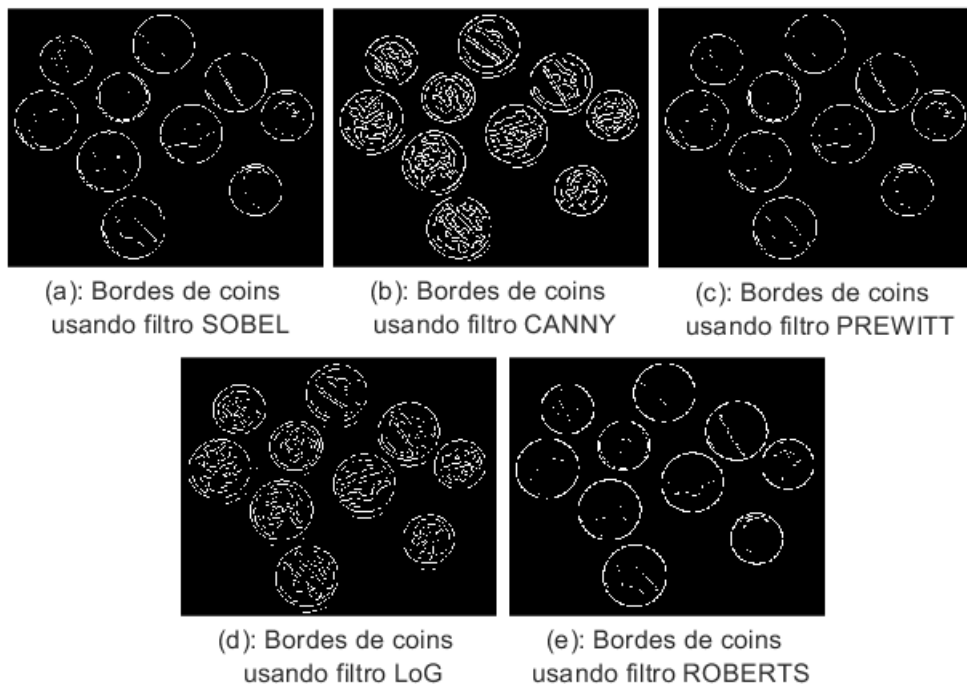


Figura 3. 5: Resultados obtenidos de los operadores de detección de bordes en monedas.

Elaborado por: Autor.

El propósito principal de esta sección es la implementación de diversas técnicas de detección de bordes. Por ejemplo, en las figuras 3.5 (a), (c) y (e) muestran los resultados de los operadores de bordes Sobel, Prewitt y Roberts, respectivamente. Estos tres operadores pueden calcular los bordes a lo largo de su orientación fácilmente, pero la detección de bordes es inexacta y más sensible al ruido.

Para comprobar la funcionalidad del algoritmo de los operadores de detección de bordes ya implementados, se procede a utilizar otra imagen para identificar los contornos. La figura 3.6 muestra los resultados obtenidos de los cinco operadores de detección de bordes.

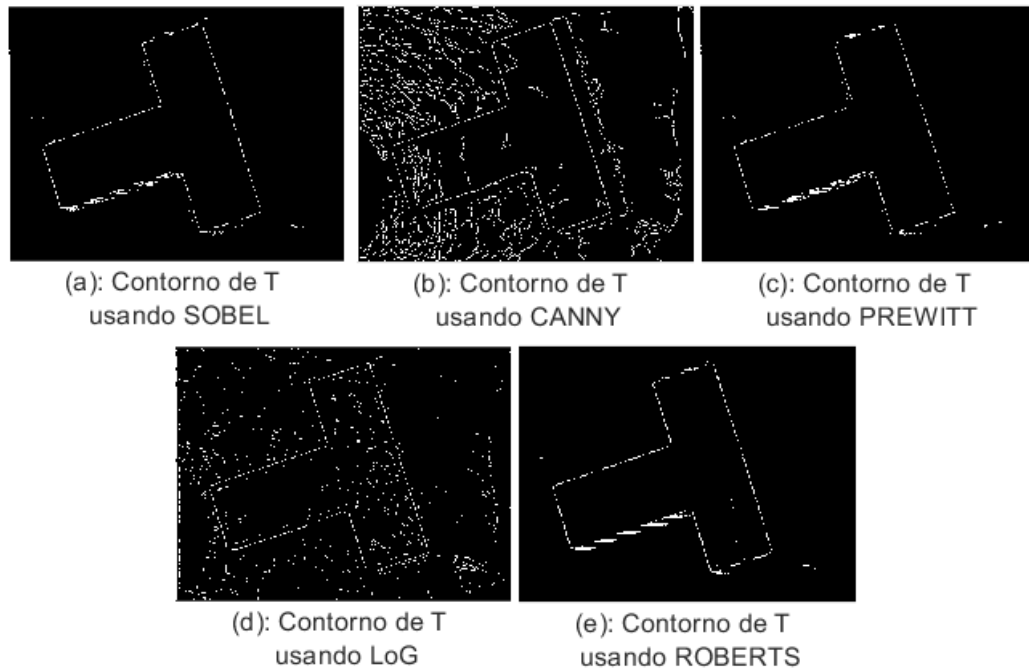


Figura 3. 6: Resultados obtenidos de los operadores de detección de bordes en monedas para imagen T.

Elaborado por: Autor.

La detección de bordes es más compleja en comparación con Sobel, Robert y Prewitt, su cálculo lleva mucho tiempo y la detección de bordes es inmune al ruido. Pero después del análisis de diferentes tipos de técnicas de detección de bordes, se concluye que la detección basada en los operadores Canny (ver figura 3.5 (b)) y LoG (ver figura 3.5 (d)) dan mejores resultados en comparación con la detección de bordes Sobel, Prewitt y Roberts. Sin embargo, el resultado experimental más eficiente resulta ser el operador de Canny (véanse las figuras 3.5 (b) y 3.6 (b)).

Dado que la detección de bordes es un paso fundamental en el procesamiento de imágenes y visión por computadora, es necesario señalar los verdaderos bordes para obtener los mejores resultados del proceso de emparejamiento. Por eso es importante elegir los detectores de bordes que mejor se adapten a la aplicación. En este sentido, primero se presentan

algunas ventajas y desventajas de las técnicas de detección de bordes en el contexto de clasificación en la tabla 3.1.

Tabla 3. 1: Comparativa de los operadores de detección de bordes.

| Operadores | Ventajas |          | Sensibilidad al ruido | Bordes falsos |
|------------|----------|----------|-----------------------|---------------|
|            | Tiempo   | Espacio  |                       |               |
| Sobel      | más bajo | alto     | menor sensibilidad    | mayor         |
| Canny      | alto     | alto     | sensibilidad mínima   | menor         |
| Prewitt    | bajo     | más bajo | sensibilidad mínima   | mayor         |
| LoG        | bajo     | menor    | sensibilidad mínima   | mayor         |
| Roberts    | alto     | alto     | mayor sensibilidad    | mayor         |

Elaborado por: Autor.

#### 3.5.4. Práctica 4: Detección de bordes para contar monedas.

Aunque la detección de bordes sirve para un sinnúmero de aplicaciones. A continuación, se implementa un algoritmo que permite detectar los bordes de monedas, reconoce el tipo de moneda y realiza el conteo respectivo de las monedas.

```

C1 = imread('coins.png');
C2 = im2bw(C1); % binarización de imagen
% Rellenar regiones y agujeros de imagen
C2 = imfill(C1,'holes');
% Etiquetar y contar componentes conectados
[L Ne]=bwlabel(double(C2));
% Métricas de área y centro de monedas
prop=regionprops(L,'Area','Centroid');
total=0; % inicializamos contador en 0
imshow(C1); % leer imagen de monedas original
hold on;

```

Se utiliza la imagen original “coins.png” (monedas de 5 y 10 centavos) para la implementación del algoritmo, aunque se puede realizar varias modificaciones si se desean detectar monedas de otra clasificación (1

centavo, 25 centavos, 50 centavos y 1 dólar). Después, se realiza el conteo de monedas basada en el área de las monedas.

```

% Contar monedas basado en el área
for n=1:size(prop,1) % de 1 al total de monedas
    cent=prop(n).Centroid;
    X=cent(1);Y=cent(2);
    if prop(n).Area>2000
        text(X-10,Y,'5 C')
        total=total+5;
    else
        total=total+10;
        text(X-10,Y,'10 C')
    end
end
end
hold on
title(['Total en USD: ',num2str(total),' Cents'])

```

La figura 3.7 muestra el resultado final del conteo de monedas dando un total de 70 centavos. El conteo de monedas se basa en reconocimiento de regiones de área y centro del círculo.



Figura 3. 7: Conteo de monedas.  
Elaborado por: Autor.

### 3.6. Discusión de resultados para diferentes imágenes en la detección de bordes.

A continuación, se realiza la discusión de los resultados obtenidos para cada operador de detección de bordes propuestos en la sección 3.5. Se discuten los resultados del número de bordes detectados, relación de

rendimiento (Performance Ratio, PR), medida F de precisión de prueba, relación señal a ruido pico (PSNR).

### 3.6.1. Número de bordes detectados.

Es el número de regiones blancas que se considera detectado como bordes en la salida. Para cada operador, calcula el número de bordes detectados de varias imágenes y tabulados en la tabla 3.2.

Tabla 3. 2: Comparativa del número de bordes detectados.

| <b>Operadores de detección</b> | <b>Sobel</b> | <b>Canny</b> | <b>Prewitt</b> | <b>LoG</b> | <b>Roberts</b> |
|--------------------------------|--------------|--------------|----------------|------------|----------------|
| Imagen 1                       | 11           | 15           | 5              | 12         | 2              |
| Imagen 2                       | 155          | 200          | 93             | 160        | 50             |
| Imagen 3                       | 7            | 19           | 3              | 10         | 2              |
| Imagen 4                       | 16           | 17           | 12             | 18         | 3              |
| Imagen 5                       | 22           | 24           | 13             | 25         | 10             |

Elaborado por: Autor.

Según la tabla 3.2, el número de bordes detectados es muy pequeño para el operador de Roberts en comparación con todos los demás operadores y muy alto para el operador de Canny. Para Sobel, el número de bordes detectados es alto en comparación con el operador de Roberts. El número de bordes detectados es similar para el operador Sobel y Prewitt.

### 3.6.2. Relación de rendimiento (PR).

Es un parámetro eficaz para medir el rendimiento de los detectores de bordes. Cuanto más altos sean los valores de PR, refleja una mejor salida de borde. La comparación con la realidad del terreno de una imagen PR se define como la relación entre el número de veces que tanto la realidad del terreno como la imagen detectada en el borde tienen una salida distinta de cero y el

número de veces que tanto la realidad del terreno como la imagen detectada en el borde tienen una salida opuesta.

Para las imágenes de realidad en el terreno, las relaciones públicas se consideran infinitas. Los valores de PR calculados para cada operador se muestran en la tabla 3.3.

Tabla 3. 3: Comparativa de la relación de rendimiento de la detección de bordes.

| <b>Operadores de detección</b> | <b>Sobel</b> | <b>Canny</b> | <b>Prewitt</b> | <b>LoG</b> | <b>Roberts</b> |
|--------------------------------|--------------|--------------|----------------|------------|----------------|
| Imagen 1                       | 8.999        | 14.050       | 8.930          | 10.015     | 8.542          |
| Imagen 2                       | 15.410       | 21.920       | 15.465         | 19.113     | 14.681         |
| Imagen 3                       | 9.979        | 16.310       | 10.059         | 14.256     | 8.262          |
| Imagen 4                       | 9.626        | 17.560       | 9.549          | 14.016     | 8.884          |
| Imagen 5                       | 11.170       | 16.760       | 11.108         | 14.675     | 10.249         |

Elaborado por: Autor.

### 3.6.3. Medida de precisión de pruebas.

Es la medida de la precisión de una prueba. Cuanto más altos sean los valores de la medida F, este refleja una mejor salida de borde. El valor máximo de la medida F es 1 para imágenes reales del terreno. Teniendo en cuenta tanto la precisión como la medida F de recuperación se define como,

$$F = 2 \frac{P * R}{P + R}$$

Donde,  $P$  es la precisión (también llamada valor predictivo positivo), es decir, la fracción de instancias recuperadas que son relevantes. Mientras que  $R$  es la recuperación (también conocida como sensibilidad), es decir, la fracción de instancias relevantes que se recuperan. Por tanto, tanto la precisión como la recuperación se basan en la comprensión y la medida de la relevancia.

Por ejemplo, suponga que un programa de computadora para reconocer perros en escenas de un video identifica 7 perros en una escena que contiene 9 perros y algunos gatos. Si 4 de las identificaciones son correctas, pero 3 son en realidad gatos, la precisión del programa es  $4/7$  mientras que su recuperación es  $4/9$ . Otro ejemplo, sería cuando un motor de búsqueda devuelve 30 páginas, de las cuales solo 20 eran relevantes y no devuelve 40 páginas relevantes adicionales, su precisión es  $20/30 = 2/3$  mientras que su recuperación es  $20/60 = 1/3$ .

Los valores de la medida F calculados para cada operador se muestran en la tabla 3.4. Tanto la relación de rendimiento como las medidas F son altas para el detector de bordes de Canny en comparación con los otros cuatro operadores. Los operadores Sobel y Prewitt tienen casi los mismos valores para dos parámetros.

Tabla 3. 4: Comparativa de la medida de precisión en la detección de bordes.

| <b>Operadores de detección</b> | <b>Sobel</b> | <b>Canny</b> | <b>Prewitt</b> | <b>LoG</b> | <b>Roberts</b> |
|--------------------------------|--------------|--------------|----------------|------------|----------------|
| Imagen 1                       | 0.0061       | 0.0137       | 0.0058         | 0.01010    | 0.0035         |
| Imagen 2                       | 0.0044       | 0.0076       | 0.0042         | 0.0066     | 0.0018         |
| Imagen 3                       | 0.0101       | 0.0173       | 0.0099         | 0.0135     | 0.0072         |
| Imagen 4                       | 0.0052       | 0.0089       | 0.0050         | 0.0067     | 0.0029         |
| Imagen 5                       | 0.0118       | 0.0139       | 0.0114         | 0.0105     | 0.0114         |

Elaborado por: Autor.

#### **3.6.4. Relación señal a ruido pico (PSNR).**

Es la relación entre la potencia máxima de la señal y la potencia de ruido. Cuanto mayor sea el valor de PSNR, mejor será la capacidad de reconstrucción de imágenes. Los valores de comparación de la PSNR entre diferentes detectores de bordes de una imagen en particular se muestran en la tabla 3.5.



Tabla 3. 5: Comparativa de la medida de precisión en la detección de bordes.

| <b>Operadores de detección</b> | <b>Sobel</b> | <b>Canny</b> | <b>Prewitt</b> | <b>LoG</b> | <b>Roberts</b> |
|--------------------------------|--------------|--------------|----------------|------------|----------------|
| Imagen 1                       | 0.0061       | 0.0137       | 0.0058         | 0.01010    | 0.0035         |
| Imagen 2                       | 0.0044       | 0.0076       | 0.0042         | 0.0066     | 0.0018         |
| Imagen 3                       | 0.0101       | 0.0173       | 0.0099         | 0.0135     | 0.0072         |
| Imagen 4                       | 0.0052       | 0.0089       | 0.0050         | 0.0067     | 0.0029         |
| Imagen 5                       | 0.0118       | 0.0139       | 0.0114         | 0.0105     | 0.0114         |

Elaborado por: Autor.

## Conclusiones

Dado que la detección de bordes es el paso inicial en el reconocimiento de objetos, es importante conocer las diferencias entre las técnicas de detección de bordes. En este componente práctico se ha estudiado las técnicas de detección de bordes más utilizadas de la detección de bordes basada en gradientes y laplaciano.

El software donde se desarrollan los algoritmos fue MatLab R2018a. Los algoritmos basados en gradientes, como el filtro Prewitt, tienen el principal inconveniente de ser muy sensibles al ruido. El tamaño del filtro del núcleo y los coeficientes son fijos y no se pueden adaptar a una imagen determinada. Es necesario un algoritmo de detección de bordes adaptativo para proporcionar una solución sólida que se adapte a los diferentes niveles de ruido de estas imágenes para ayudar a distinguir los contenidos de imágenes válidos de los artefactos visuales introducidos por el ruido.

El rendimiento del algoritmo para el operador Canny depende en gran medida de los parámetros ajustables,  $\sigma$ , que es la desviación estándar del filtro gaussiano, y los valores de umbral, "T1" y "T2".  $\sigma$  también controla el tamaño del filtro gaussiano. Cuanto mayor sea el valor de  $\sigma$ , mayor será el tamaño del filtro gaussiano. Esto implica más borrosidad, necesaria para imágenes ruidosas, además de detectar bordes más grandes. Sin embargo, como era de esperar, cuanto mayor sea la escala del gaussiano, menos precisa será la localización del borde.

Los valores más pequeños de  $\sigma$  implican un filtro gaussiano más pequeño que limita la cantidad de desenfoque, manteniendo los bordes más finos en la imagen. El usuario puede personalizar el algoritmo ajustando estos parámetros para adaptarse a diferentes entornos. El algoritmo de detección de bordes de Canny es computacionalmente más caro en comparación con el operador de Sobel, Prewitt y Robert. Sin embargo, el algoritmo de detección de bordes de Canny funciona mejor que todos estos operadores en casi todos los escenarios. La evaluación de las imágenes mostró que, en condiciones

ruidosas, Canny, LoG, Sobel, Prewitt y Roberts presentan un mejor rendimiento, respectivamente.

## **Recomendaciones.**

Evaluación de los operadores de detección de bordes utilizando el software LabVIEW

Desarrollar un dispositivo que permita procesar imágenes y enviar imágenes con detección de bordes para Rayos X odontológicos

## Bibliografía.

- Dagar, N. S., & Dahiya, P. K. (2020). Edge Detection Technique using Binary Particle Swarm Optimization. *Procedia Computer Science*, 167, 1421–1436. <https://doi.org/10.1016/j.procs.2020.03.353>
- Feng, Y., Zhang, J., & Wang, S. (2017). *A new edge detection algorithm based on Canny idea*. 040011. <https://doi.org/10.1063/1.5005213>
- Gill, R., Kaur, N., & Singla, A. K. (2012). A Comparative Analysis of Edge Detection Operators: Application for Text Detection. *International Journal of Computer Science and Technology*, 3(4), 530–533.
- Jin-Yu, Z., Yan, C., & Xian-Xiang, H. (2009). Edge detection of images based on improved Sobel operator and genetic algorithms. *2009 International Conference on Image Analysis and Signal Processing*, 31–35. <https://doi.org/10.1109/IASP.2009.5054605>
- Kekre, H. B., & Gharge, S. M. (2010). Image Segmentation using Extended Edge Operator for Mammographic Images. *International Journal on Computer Science and Engineering*, 2(4), 1086–1091.
- Kumari, R., Kumar, A., Mandoria, H. L., Pandey, B. K., & Prasad, S. (2020). A Review on Comparative Study of Different Edge Detection Techniques. *International Research Journal of Engineering and Technology*, 7(10), 748–754.
- Lakhani, K., Minocha, B., & Gugnani, N. (2016). Analyzing edge detection techniques for feature extraction in dental radiographs. *Perspectives in Science*, 8, 395–398. <https://doi.org/10.1016/j.pisc.2016.04.087>
- Nixon, M. S., & Aguado, A. S. (2012). Low-level feature extraction (including edge detection). En *Feature Extraction & Image Processing for Computer Vision* (pp. 137–216). Elsevier. <https://doi.org/10.1016/B978-0-12-396549-3.00004-5>

- Othman, Z., Haron, H., & Kadir, M. (2009). *Comprison of canny and sobel edge detection in MRI images*. Undefined. /paper/Comprison-of-canny-and-sobel-edge-detection-in-MRI-Othman-Haron/3d972e54ad6518a19738e54a640de4002e86c4ab
- Priyam, Dey, D., Shreya, & Polley D. (2016). Edge Detection by Using Canny and Prewitt. *International Journal of Scientific & Engineering Research*, 7(4), 251–254.
- Talib, M. L., Rui, X., Ghazali, K. H., Mohd. Zainudin, N., & Ramli, S. (2013). Comparison of Edge Detection Technique for Lane Analysis by Improved Hough Transform. En H. B. Zaman, P. Robinson, P. Olivier, T. K. Shih, & S. Velastin (Eds.), *Advances in Visual Informatics* (Vol. 8237, pp. 176–183). Springer International Publishing. [https://doi.org/10.1007/978-3-319-02958-0\\_17](https://doi.org/10.1007/978-3-319-02958-0_17)
- Tong, X., Ren, A., Zhang, H., Ruan, H., & Luo, M. (2011). *Edge detection based on genetic algorithm and sobel operator in image*. 828572. <https://doi.org/10.1117/12.913510>



## DECLARACIÓN Y AUTORIZACIÓN

Yo, **León Falconí, Shirley Andrea** con C.C: # 095047967-5 autor del Trabajo de Titulación: **Análisis comparativo de los operadores de detección de bordes en el procesamiento de imágenes utilizando MatLab** previo a la obtención del título de **INGENIERA EN TELECOMUNICACIONES** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 10 de marzo del 2021

f. 

Nombre: León Falconí, Shirley Andrea

C.C: 095047967-5

## **REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA**

### **FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN**

|   |   |   |    |
|---|---|---|----|
| <b>TÍTULO Y SUBTÍTULO:</b>  | Análisis comparativo de los operadores de detección de bordes en el procesamiento de imágenes utilizando MatLab |   |    |
| <b>AUTOR(ES)</b>  | León Falconí, Shirley Andrea  |   |    |
| <b>REVISOR(ES)/TUTOR(ES)</b>  | M. Sc. Romero Paz, Manuel de Jesús  |   |    |
| <b>INSTITUCIÓN:</b>   | Universidad Católica de Santiago de Guayaquil   |   |    |
| <b>FACULTAD:</b>  | Facultad de Educación Técnica para el Desarrollo  |   |    |
| <b>CARRERA:</b>   | De Telecomunicaciones   |   |    |
| <b>TÍTULO OBTENIDO:</b>   | Ingeniera en Telecomunicaciones   |   |    |
| <b>FECHA DE PUBLICACIÓN:</b>  | 10 de Marzo del 2021  | <b>No. DE PÁGINAS:</b>  | 34 |
| <b>ÁREAS TEMÁTICAS:</b>   | Procesamiento de señales e imágenes, Herramientas de simulación   |   |    |
| <b>PALABRAS CLAVES/<br/>KEYWORDS:</b>   | Detección, Bordes, Filtros, Algoritmo, Operadores, Procesamiento  |   |    |
| <b>RESUMEN/ABSTRACT (150-250 palabras):</b>   |   |   |    |
| <p>El objetivo del presente estudio es comparar varios operadores de detección de bordes y analizar su desempeño y también el desempeño de tales técnicas se lleva a cabo para diferentes imágenes utilizando el software de simulación MatLab R2018a. En el capítulo 2 se presenta una base teórica y matemática para la detección de bordes. En el capítulo 3 se presenta el enfoque propuesto y proporciona los resultados experimentales y la discusión de los resultados. Se ha demostrado que el algoritmo de detección de bordes de Canny funciona mejor que todos estos operadores en casi todos los escenarios. La evaluación de las imágenes mostró que en condiciones ruidosas Canny, LoG (Laplaciano de Gauss), Robert, Prewitt, Sobel exhiben un mejor desempeño, respectivamente. Se ha observado que el algoritmo de detección de bordes de Canny es computacionalmente más caro en comparación con el operador de LoG (Laplaciano de Gauss), Sobel, Prewitt y Robert.</p> |   |   |    |
| <b>ADJUNTO PDF:</b>   | <input checked="" type="checkbox"/> SI  | <input type="checkbox"/> NO   |    |
| <b>CONTACTO CON AUTOR/ES:</b>   | <b>Teléfono:</b> +593 99 660 9926   | <b>E-mail:</b> <a href="mailto:sleon@telectrodat.net">sleon@telectrodat.net</a> |    |
| <b>CONTACTO CON LA INSTITUCIÓN:<br/>COORDINADOR DEL PROCESO DE UTE</b>  | <b>Nombre:</b> Palacios Meléndez Edwin Fernando   |   |    |
|   | <b>Teléfono:</b> +593-9-67608298  |   |    |
|   | <b>E-mail:</b> <a href="mailto:edwin.palacios@cu.ucsg.edu.ec">edwin.palacios@cu.ucsg.edu.ec</a>                 |   |    |
| <b>SECCIÓN PARA USO DE BIBLIOTECA</b>   |   |   |    |
| <b>Nº. DE REGISTRO (en base a datos):</b>   |   |   |    |
| <b>Nº. DE CLASIFICACIÓN:</b>  |   |   |    |
| <b>DIRECCIÓN URL (tesis en la web):</b>   |   |   |    |