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

TEMA:

Desarrollo de aplicaciones prácticas en la adquisición y procesamiento de imágenes en MatLab y LabVIEW

AUTOR:

Pallo Jiménez, Yesy Javier

Trabajo de Titulación previo a la obtención del título de

INGENIERO EN TELECOMUNICACIONES

TUTOR:

M. Sc. Zamora Cedeño, Néstor Armando

Guayaquil, Ecuador

10 de Marzo del 2021

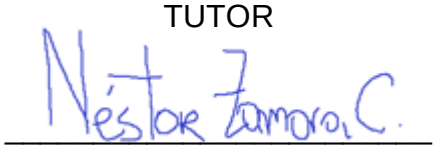
UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE TELECOMUNICACIONES

CERTIFICACIÓN

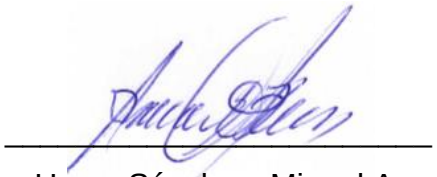
Certificamos que el presente trabajo fue realizado en su totalidad por la Srta. **Pallo Jiménez, Yesy Javier** como requerimiento para la obtención del título de **INGENIERA EN TELECOMUNICACIONES**.

TUTOR



M. Sc. Zamora Cedeño, Néstor Armando

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, **Pallo Jiménez, Yesy Javier**

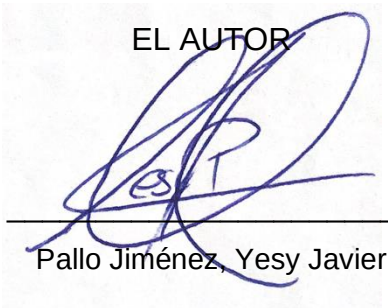
DECLARÓ QUE:

El trabajo de titulación **"Desarrollo de aplicaciones prácticas en la adquisición y procesamiento de imágenes en MatLab y LabVIEW"** previo a la obtención del Título de **Ingeniera en Telecomunicaciones**, 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



Pallo Jiménez, Yesy Javier

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE TELECOMUNICACIONES

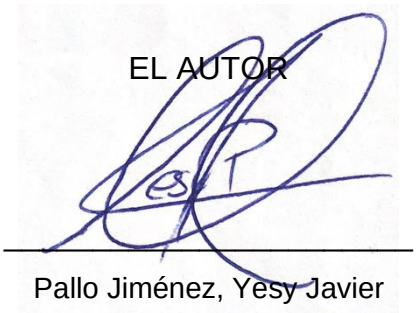
AUTORIZACIÓN

Yo, **Pallo Jiménez, Yesy Javier**

Autorizó a la Universidad Católica de Santiago de Guayaquil, la publicación, en la biblioteca de la institución del Trabajo de Titulación: **"Desarrollo de aplicaciones prácticas en la adquisición y procesamiento de imágenes en MatLab y LabVIEW"**, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

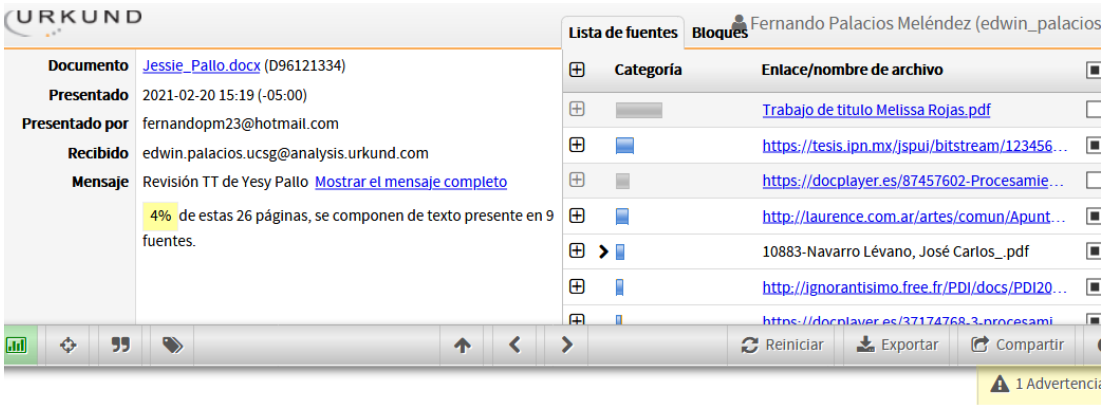
Guayaquil, a los 10 días del mes de marzo del año 2021

EL AUTOR



Pallo Jiménez, Yesy Javier

REPORTE DE URKUND



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

TEMA: Desarrollo de aplicaciones prácticas en la adquisición y procesamiento de imágenes en MatLab y LabVIEW

AUTOR: Pallo Jiménez, Yesy Javier

Trabajo de Titulación previo a la obtención del título de INGENIERO EN TELECOMUNICACIONES

TUTOR: M. Sc. Zamora Cedeño, Néstor Armando

Guayaquil, Ecuador

29 de Septiembre del 2020

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

DEDICATORIA

Dedico mi trabajo de titulación a mi familia, en especial a mis padres Eduardo Javier Pallo Ortiz y Nancy Laura Jiménez Olvera que estuvieron durante todo el proceso de formación académica, muchos de mis logras se los debo a ellos. Me formaron con reglas y con algunas libertades. Pero al final de todo fueron mi motivación para cumplir, alcanzar mis propósitos y mis metas.

A mi hermano, hermana y mi perrito (Sam) por ser un apoyo constante en este proyecto de la vida. También a mis compañeros de la universidad que me supieron ayudar en distintas formas académicas.

EL AUTOR

Pallo Jiménez, Yesy Javier

AGRADECIMIENTO

Agradezco primero a Dios por la sabiduría de poder culminar esta estaba estudiantil, a mi madre por estar todo el tiempo a mi lado apoyándome, a mi tutor de mi tesis M. Sc. Zamora Cedeño, Néstor Armando que ha estado en el proceso que he llevado mi trabajo de titulación.

EL AUTOR

Pallo Jiménez, Yesy Javier

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 JESUS

DECANO

f. 

M. Sc. HERASSÁNCHEZ MIGUEL ARMANDO

DIRECTOR DE CARREA

f. 

M. Sc. PALACIOS MELÉNDEZ, EDWIN FERNANDO

OPONENTE

Índice General

Índice de Figuras

Capítulo 2

Capítulo 3

Índice de Tablas

Capítulo 2

Capítulo 3

Resumen

Cada día el mundo está evolucionando muy rápido. El rápido desarrollo de la tecnología informática ha afectado a todas las áreas científicas. Medicina, automatización, análisis de datos, finanzas, biología, química, economía y muchos, muchos más se han beneficiado de la expansión de la tecnología. El mundo de los gráficos por ordenador evoluciona rápidamente todos los días. El mercado está lleno de aplicaciones de procesamiento de imágenes. Pero ¿cuántos de ellos realmente contribuyen al crecimiento de la ciencia y la tecnología? ¿Existe algún programa que combine funcionalidades de procesamiento de imágenes con técnicas de programación? La respuesta a estas preguntas es Matlab y LabVIEW. Matlab es un software que proporciona un lenguaje de programación de alto nivel, muchas bibliotecas temáticas y mecanismos de interfaz gráfica de usuario fáciles de implementar. El procesamiento de imágenes es una parte importante de la instrumentación virtual. El software LabVIEW se puede utilizar para el procesamiento de imágenes para obtener mejores resultados. De acuerdo con estos antecedentes, se desarrolla el trabajo de titulación que se basa en la implementación de aplicaciones prácticas en la adquisición y procesamiento de imágenes en las plataformas de simulación MatLab y LabVIEW.

Palabras claves: Procesamiento, Imágenes, Detección, Bordes, Resolución, Cuantificación.

Capítulo 1: Descripción General del Trabajo de Titulación

1.1. Introducción.

La enseñanza es el proceso de educar o instruir a quienes desconocen fenómenos particulares. La enseñanza siempre se ha considerado un servicio, por lo que, como cualquier otro servicio, cuando uno no es competente probablemente no sea ético. Hay tres tipos principales de estilos de aprendizaje: auditivo, visual y cinestésico. Prácticamente, una clase de estudiantes es un grupo de personas que tienen una combinación de estos tres estilos de aprendizaje.

En esta era de la tecnología de la información, los estudiantes ya no son auditivos, sino con un estilo de aprendizaje de estilos visuales y cinestésicos. Entonces, cuando un maestro enseña especialmente ciencia o ingeniería, las meras conferencias por sí solas no pueden ayudar a los estudiantes a comprender y apreciar los conceptos. Esto representa un desafío para los profesores de educación superior y los motiva a adaptar la enseñanza visual y basada en la experimentación.

El procesamiento de imágenes digitales es el proceso de procesar imágenes digitales con diversas técnicas como restauración, eliminación de ruido, segmentación, detección de bordes, etc. Hoy en día, el procesamiento de imágenes digitales juega un papel vital en la vida cotidiana, incluido comunicación multimedia, diagnóstico médico, astronomía, pronóstico del tiempo, coincidencia de patrones y reconocimiento (reconocimiento de matrículas de vehículos, reconocimiento de huellas dactilares y palmares) para aplicaciones de seguridad, análisis forense, sistemas de información geográfica, interfaces de computadora humana, inspección industrial, procesamiento de documentos, control remoto detección, procesamiento de imágenes de satélite y transmisiones de imágenes de manera alámbrica e inalámbrica.

La enseñanza del procesamiento de imágenes digitales es un desafío para los docentes que manejan estudiantes de todos los estilos de

aprendizaje. Cabe mencionar la confesión de Confucio, el gran filósofo chino de que “escucho y olvido. Veo y recuerdo. Lo hago y lo entiendo”. Por lo tanto, los métodos de aprendizaje visual y experimental pueden influir en los estudiantes mucho mejor que la mera conferencia. Los conceptos de procesamiento de imágenes digitales pueden entenderse fácilmente si se enseñan utilizando métodos visuales, experimentales e interactivos.

MATLAB (MATrix LABoratory) es una plataforma informática adecuada para desarrollar y probar diversas aplicaciones de ingeniería, ciencia y gestión. Es un entorno de programación de alto nivel que está equipado con un buen número de cajas de herramientas (ToolBox) con funciones para integrar algoritmos basados en MATLAB con aplicaciones y lenguajes externos como C, C ++, Java, .NET y Microsoft Excel.

El lenguaje MATLAB proporciona soporte nativo para las operaciones vectoriales y matriciales que son fundamentales para resolver problemas científicos y de ingeniería, lo que permite un rápido desarrollo y ejecución. Con el lenguaje MATLAB, los programas y algoritmos se pueden desarrollar más rápido que con los lenguajes tradicionales. En muchos casos, el soporte para operaciones vectoriales y matriciales elimina la necesidad de largos bucles for. Por lo tanto, una sola línea de código MATLAB a menudo puede reemplazar varias líneas de código C o C ++.

MATLAB también proporciona características de lenguajes de programación tradicionales, como control de flujo, manejo de errores y programación orientada a objetos (OOP). Además de los tipos de datos fundamentales, MATLAB también permite tipos de datos definidos por el usuario. Se pueden producir resultados inmediatos ejecutando comandos de uno en uno de forma interactiva. Este enfoque ayuda a explorar rápidamente múltiples opciones e iterar hacia una solución óptima.

1.2. Antecedentes.

Una imagen se define como la representación 2-D de la escena 3D. Una imagen digital se considera la representación numérica de la imagen 2D en

una forma muestreada y cuantificada. El elemento de imagen básico se llama píxel y una imagen $M \times N$ tiene M filas de píxeles y N columnas de píxeles. También se puede pensar en una cuadrícula o matriz 2D cuyos elementos están representados por f (x, y), donde x e y son las coordenadas de la cuadrícula o los índices de los elementos de la matriz.

1.3. Definición del Problema.

El procesamiento de imágenes digitales es un área de la ingeniería muy poco abordada en los trabajos de titulación de la Carrera de Telecomunicaciones, por eso, surge la necesidad de realizar aplicaciones prácticas en la adquisición y procesamiento de imágenes en MatLab y LabVIEW.

1.4. Justificación del Problema.

El procesamiento de imágenes digitales se ocupa del procesamiento de imágenes digitales con la ayuda de algoritmos informáticos. Hay varios métodos de procesamiento de imágenes basados en las aplicaciones. En este trabajo de titulación, se abordan los problemas relacionados con la enseñanza adquisición y tratamiento de imágenes para la segmentación y la detección de bordes de imágenes.

1.5. Objetivos del Problema de Investigación.

1.5.1. Objetivo General.

Desarrollar aplicaciones prácticas en la adquisición y procesamiento de imágenes en MatLab y LabVIEW

1.5.2. Objetivos Específicos.

- Describir los fundamentos teóricos del procesamiento de imágenes digitales.
- Diseñar las aplicaciones prácticas del tratamiento de imágenes utilizando el software MatLab y LabVIEW.
- Analizar los resultados obtenidos para los escenarios de simulación en el procesamiento de imágenes digitales propuestos.

1.6. Hipótesis.

El procesamiento de imágenes juega un papel importante en la vida diaria. La implementación de aplicaciones de procesamiento de imágenes usando MatLab se lleva a cabo en la práctica. La implementación basada en bloques de aplicaciones de procesamiento de imágenes se puede realizar con LabVIEW. El presente trabajo de titulación demostrará la funcionalidad en las operaciones básicas en imágenes como la extracción de los componentes RGB (rojo, verde y azul) en una imagen en color, la conversión de la imagen en escala de grises a una imagen binaria y el proceso de detección de bordes tanto en MATLAB como en LabVIEW.

1.7. Metodología de Investigación.

La metodología que se emplea en el procesamiento de imágenes es "comprender" automáticamente imágenes y escenas en 3-D por computadora. Los diversos pasos que conducen a este punto se dividen comúnmente en dos partes principales: procesamiento de imágenes, propiamente hablando y análisis de imágenes. Las técnicas de procesamiento de imágenes ya se han desarrollado en gran medida y son bien conocidas por la comunidad de Ingenieros en Telecomunicaciones, Electrónica y afines.

El enfoque que se utiliza en el presente trabajo de titulación es el cuantitativo y los métodos empleados son descriptivos y simulados. Es cuantitativo porque el procesamiento de imágenes son matrices algebraicas. Es descriptivo, porque se revisa las bases teóricas del procesamiento de imágenes. Y finalmente, es simulado porque los escenarios o aplicaciones prácticas que se desarrollaron fueron sobre dos plataformas de simulación, que son, MatLab y LabVIEW.

Capítulo 2: Fundamentación Teórica

2.1. Visión general del procesamiento de imágenes.

El procesamiento de imágenes se ocupa del procesamiento de imágenes o fotogramas de video. Mejora las características importantes de la imagen al tiempo que atenúa los detalles irrelevantes para una aplicación específica. El procesamiento de imágenes implica cambiar la naturaleza de una imagen, ya sea para mejorar su información pictórica para la interpretación humana o para hacerla adecuada para la interpretación mecánica. (Sonka et al., 2015)

El procesamiento de imágenes es cualquier forma de procesamiento de señales, donde la entrada es una imagen, como una fotografía o un cuadro de video; la salida puede ser una imagen o un conjunto de características o parámetros relacionados con la imagen (Gonzalez & Woods, 2018). La mayoría de las técnicas de procesamiento de imágenes implican tratar la imagen como una matriz bidimensional y aplicarle técnicas estándar de procesamiento de señales.

Las áreas de aplicación detrás del interés en los métodos de procesamiento de imágenes digitales son muy variadas y pueden clasificarse en dos categorías principales (a saber, mejora de la información pictórica para la interpretación humana; y procesamiento de datos de imágenes para almacenamiento, transmisión y representación para visión artificial).

Las aplicaciones espaciales, las imágenes médicas, las observaciones de recursos terrestres remotos y la astronomía son ejemplos de esas aplicaciones, en las que es evidente la mejora de la información pictórica para la interpretación humana. El estudio de los patrones de contaminación de imágenes aéreas y satelitales, la mejora de imágenes y los procedimientos de restauración para procesar imágenes degradadas de objetos irrecuperables son otras dos aplicaciones que básicamente pueden servir a los geógrafos y arqueólogos respectivamente. En las aplicaciones de visión artificial, el interés

se centra en los procedimientos para extraer información de una imagen en una forma adecuada para el procesamiento informático.

Otra forma, mediante la cual uno posiblemente pueda desarrollar una comprensión básica del alcance de las aplicaciones de procesamiento de imágenes, es categorizar las imágenes según sus fuentes. Las imágenes basadas en la radiación del espectro electromagnético son las más familiares, especialmente las imágenes en las bandas de rayos X y visuales del espectro. Las imágenes de otras fuentes incluyen imágenes acústicas, microscopía electrónica e imágenes sintéticas (generadas por computadora).

El procesamiento de imágenes digitales implica una serie de pasos fundamentales (por ejemplo, adquisición de imágenes, mejora y preprocesamiento de imágenes, detección y segmentación de bordes, representación y descripción, y coincidencia y reconocimiento). El resultado de estos pasos es una imagen o un atributo de imagen. Se puede encontrar un extenso cuerpo de literatura que incluye libros, artículos de revistas, informes técnicos y tesis de investigación que abordan el tema del procesamiento de imágenes digitales.

Los libros cubren normalmente todos o la mayoría de los pasos fundamentales del procesamiento de imágenes (Gonzalez & Woods, 2018; Sonka et al., 2015). Además de los libros de texto y de referencia, también se pueden encontrar trabajos publicados sobre diversas aplicaciones, tales como, detección de bordes (Bustacara-Medina et al., 2020; Vikram Mutneja, 2015), segmentación (Ramadan et al., 2020; Zaitoun & Aqel, 2015) y clasificación (Xin & Wang, 2019), entre otras decenas de artículos de revistas.

2.2. Componentes básicos de un sistema de procesamiento de imágenes digitales

Los modelos de sistemas de procesamiento de imágenes vendidos en todo el mundo a mediados de la década de 1980 consistían principalmente en dispositivos periféricos robustos conectados a computadoras principales (servidores). A fines de la década de 1980 y principios de la de 1990, los

sistemas de procesamiento de imágenes comenzaron a cambiar hacia la forma de placas de circuitos compatibles con computadoras personales.

Los sistemas de procesamiento de imágenes a gran escala todavía se utilizan ampliamente para aplicaciones de procesamiento de imágenes robustas, como el procesamiento de imágenes de sistemas satelitales. Pero hoy en día la tendencia es reducir el tamaño de los sistemas de procesamiento de imágenes especializados y, al mismo tiempo, hacer que las pequeñas computadoras de uso general sean compatibles con estas funciones. Los componentes de un sistema típico de propósito general utilizado para el procesamiento de imágenes digitales se muestran en la figura 2.1.

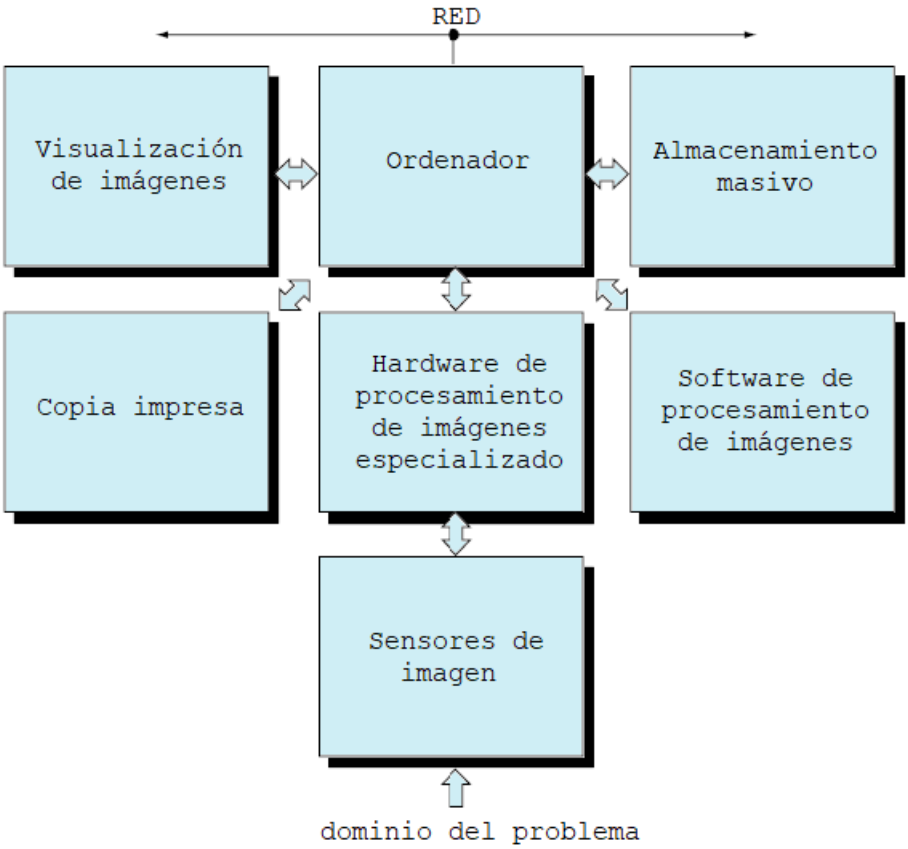


Figura 2. 1: Componentes básicos de un sistema de procesamiento de imágenes digitales.
Fuente: (Gonzalez & Woods, 2018)

a) Sensor de imagen:

Para obtener imágenes digitales, se necesita un dispositivo físico sensible a la energía iluminado por el objeto a ser fotografiado y elementos digitalizadores que conviertan la salida del dispositivo físico en forma digital.

Por ejemplo; los digitalizadores convierten las salidas eléctricas producidas por los receptores en proporción a la intensidad de la luz en una cámara de video digital en información digital.

b) Hardware de procesamiento de imágenes especializado:

Consta de elementos de sensor de imagen, digitalizador y otro hardware que realiza operaciones aritméticas y lógicas.

c) Software de procesamiento de imágenes:

El software producido para el procesamiento de imágenes consta de módulos especializados que realizan tareas específicas. Un programa de paquete bien diseñado debe incluir la capacidad de escribir códigos que hagan que el usuario utilice módulos especializados.

d) Almacenamiento masivo:

La capacidad de almacenamiento masivo es demasiado importante para pasarla por alto en las aplicaciones de procesamiento de imágenes. Una imagen sin comprimir de 1024x1024 píxeles con cada píxel de 8 bits de longitud requiere aproximadamente 1 Mbyte de memoria libre. Proporcionar suficiente memoria para un sistema de procesamiento de imágenes cuando se trata de miles o incluso millones de imágenes es una tarea que empuja los límites del pensamiento humano. La memoria digital utilizada para aplicaciones de procesamiento de imágenes se divide en tres categorías básicas.

1. Memoria a corto plazo utilizada en el procesamiento.
2. Memoria en línea para una recuperación relativamente rápida.
3. Memoria de archivo para entradas poco frecuentes.

Byte se mide en una unidad de medida de bytes de 8 bits de longitud. Un método para crear memoria a corto plazo es utilizar la memoria de acceso aleatorio *(Random Access Memory, RAM)* de la computadora. Otro método consiste en utilizar placas de circuito especializadas denominadas búferes de trama. Estos pueden almacenar una o más imágenes y, como un video, se puede acceder muy rápidamente a 30 imágenes completas por segundo.

El método de búfer de tramas permite acercar y alejar instantáneamente. Los búferes de tramas se guardan generalmente dentro del equipo de procesamiento de imágenes especializado que se utiliza en la figura 2.1. La memoria en línea suele adoptar la forma de discos magnéticos o memorias de medios ópticos. El acceso frecuente a la información almacenada es el factor principal que caracteriza la memoria en línea. En consecuencia, la memoria de archivo se caracteriza por requisitos de memoria que necesitan un acceso unitario pero escaso.

e) Visualización de imágenes:

Las imágenes que se utilizan hoy en día en su mayoría en color, preferiblemente monitores de TV de pantalla plana. Los monitores funcionan con salidas de video y tarjetas gráficas de video integradas con sistemas informáticos.

f) Almacenamiento de imágenes (copia impresa)

Para unidades digitales como impresoras láser, se utilizan cámaras de película, herramientas sensibles al calor, unidades de inyección de tinta y discos ópticos.

g) Red de información (Red),

Se utiliza actualmente en todos los sistemas informáticos. Debido a la gran cantidad de información disponible en las aplicaciones de procesamiento de imágenes, la clave para la transmisión de imágenes es el ancho de banda.

2.3. Conceptos básicos de imagen digital

En esta sección se presentan los conceptos básicos de imágenes digitales.

2.3.1. Imagen en blanco y negro.

Se puede pensar en una imagen en blanco y negro como una función bidimensional que muestra la intensidad de la luz. En esta función f (x, y), las coordenadas cartesianas x e y, el valor de la función en el punto (x, y) muestra el brillo de la imagen en ese punto. El brillo de una imagen en blanco y negro

en el punto (x, y) se denomina nivel de grises de la imagen en este punto. En teoría x, y ∈ R y f (x, y) es un número finito mayor que cero. f (x, y) tiene dos componentes:

1) Cantidad de luz en el ambiente (iluminación),

2) La velocidad a la que el objeto refleja la luz (refleja).

Si denotamos la iluminación con i (x, y) y la reflexión con r (x, y), podemos escribir f (x, y) como el producto de estos:

$$f(x,y) = i(x,y)r(x,y) \qquad (2.1)$$

Aquí teóricamente,

$$0 < i(x,y) < \infty \qquad (2.2)$$

y,

$$0 < r(x,y) < \infty \qquad (2.3)$$

La fuente de luz determina i (x, y) y las propiedades de reflexión de los objetos determinan r (x, y). El valor límite ’0’ para r (x, y) significa absorción total de luz, y el valor límite ‘1’ significa reflexión total de luz. La iluminación en un día promedio es de 9000 en un día despejado, 1000 en un día nublado, y 100 pies-candela en una oficina típica. Algunos valores encontrados como promedio para la reflexión son 0.01 para el objeto más oscuro, 0.65 para acero inoxidable y 0.93 para nieve. (Gonzalez & Woods, 2018)

Si se define el valor de la función f (x, y) en el punto (x, y) como λ, de las ecuaciones (2.1), (2.2) y (2.3) se ve que varía en el rango:

$$L_{mín} \leq \lambda \leq L_{máx} \qquad (2.4)$$

En teoría, el único requisito es que $L_{mín}$ sea positivo y $L_{máx}$ sea finito. En las aplicaciones, si se escribe $L_{mín} = i_{mín}r_{mín}$ y $L_{máx} = i_{máx}r_{máx}$ y teniendo en cuenta los valores anteriores, se encuentran $L_{mín} \cong 0.005$ y $L_{máx} \cong 100$ para espacios interiores. El rango de $[L_{mín}, L_{máx}]$ se llama escala de grises. En la práctica, este rango se desplaza al rango [0, L], en la que $\lambda = 0$ se considera

negro y $\lambda = L$ se considera blanco. Los valores intermedios corresponden a tonos de gris.

2.3.2. Imagen digital

El sistema de coordenadas y la amplitud de la función de imagen f (x, y) deben digitalizarse para poder operar en la imagen con una computadora. La digitalización del sistema de coordenadas cartesiana (x, y) se denomina muestreo de imágenes, y la digitalización de la amplitud se denomina cuantificación del nivel de grises. En este caso, la pintura digital es una función imagen en la que tanto las coordenadas cartesianas como el brillo están en forma discreta.

$$f(x,y) \cong \begin{pmatrix} f(0,0) & f(0,1) & \cdots & f(0,M-1) \\ f(0,0) & f(0,2) & \cdots & f(1,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,M-1) \end{pmatrix} \qquad (2.5)$$

En la ecuación (2.5), la función f (x, y) muestra una imagen continua. El lado derecho de la ecuación es la aproximación de esta imagen y se llama imagen digital. Se puede pensar en esta imagen digital como una matriz en la que los índices de fila y columna indican las coordenadas de un punto en la imagen, y el elemento en ese punto indica el valor de la escala de grises (i = 0,1, ..., N - 1; j = 0,1, ..., M −1). Los elementos de dicho plano de imagen se denominan píxeles.

En el proceso de digitalización, es necesario determinar los valores N, M y el número de tonos de gris discretos. Llamemos al número de tonos de gris G. En aplicaciones informáticas, este valor generalmente se elige potencias enteras positivas de 2:

$$N = 2^n, M = 2^m, G = 2^k$$

Suponiendo que los valores de escala de grises aquí están en el rango de [0, L] y valores discretos igualmente espaciados. Dado que las imágenes se procesan y almacenan en un entorno informático, se necesita una memoria de bits N * M * k para almacenar una imagen. Dado que la matriz de la imagen se aproxima a una función continua, nos enfrentamos al problema de cuáles

deberían ser las dimensiones de la matriz y el número de tonos grises para una buena similitud.

Cuanto mayor sean estos parámetros, más cerca estará la imagen digital de la imagen original. Sin embargo, también aumentará la cantidad de memoria necesaria para la imagen y el tiempo necesario para el procesamiento de la imagen. Es difícil definir una buena imagen porque las necesidades de varias áreas de aplicación son diferentes entre sí. Una imagen equivalente a una imagen de televisión en blanco y negro se puede encontrar con una imagen en escala de grises de 128 a 512 x 512 píxeles.

2.3.3. Imagen en color

El uso de imágenes en color en el procesamiento de imágenes tiene varias ventajas. En el análisis automático de imágenes, el color es una herramienta poderosa que facilita la identificación de objetos y su aislamiento de su entorno. Además, la esencia humana puede distinguir entre unas dos docenas de tonos de gris, pero miles de matices e intensidades.

2.3.4. Archivos de imagen

El primer paso en el procesamiento de imágenes es la adquisición y digitalización de imágenes. Las imágenes digitalizadas se almacenan en el entorno informático en forma de imagen en movimiento (vídeo) o archivo de imagen.

2.3.4.1. Gráficos vectoriales y de píxeles

Los archivos de imagen se pueden almacenar de dos formas. Estos archivos de imagen se denominan archivos vectoriales o basados en vectores si se almacenan como curvas, áreas y los colores con los que se rellenan. Cuando se crea una línea o pendiente con programas vectoriales, se crea una expresión matemática entre el punto inicial y el final. Los ángulos y distancias se incluyen como variaciones en esta ecuación. Cuando la forma dibujada se lleva a un nuevo tamaño, la ampliación se ingresa en las variables y la forma se recrea sin pérdida. Si la imagen se crea como una colección de puntos, dichas imágenes se denominan imágenes basadas en píxeles. El tamaño de

archivo de las imágenes basadas en píxeles aumenta al aumentar la ampliación.

2.3.4.2. Pixeles

Pixel es la unidad más pequeña de la imagen cuadrada. Las imágenes digitales consisten en una colección de píxeles uno al lado del otro, tal como se observa en la figura 3.2. En el espacio bidimensional, la imagen digital se define como una matriz que utiliza filas y columnas. Cada elemento de la matriz se alcanza en las intersecciones de filas y columnas en la forma (x, y). Pixel no tiene valores de ancho y alto por sí solo. Un solo píxel rectangular puede ser de 1x1 mm, 1x1 cm o 3x2 m. Los píxeles tienen la misma relación de aspecto a menos que se indique lo contrario.

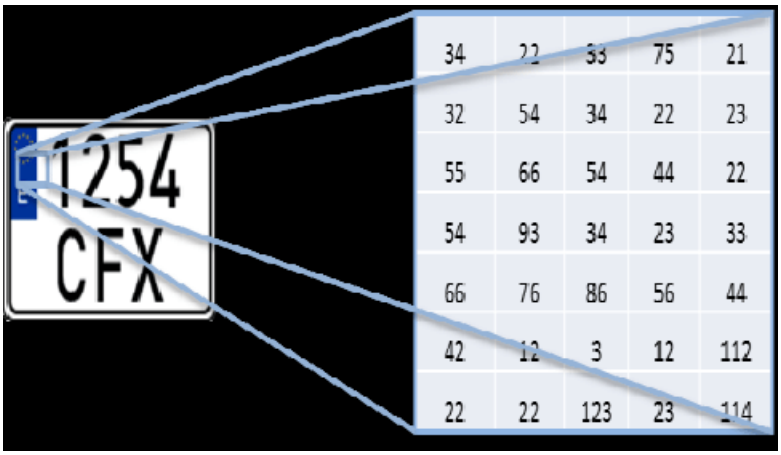


Figura 2. 2: Representación de una imagen digital en píxeles.
Fuente: (Rajendra et al., 2017)

La resolución es un concepto adicional necesario en las definiciones de dimensión. Cuando se determina el tamaño de los píxeles que no tienen una dimensión propia, se determina cuántos píxeles se encontrarán en la unidad de longitud. (Rajendra et al., 2017)

Independientemente del área que ocupe, cada píxel como unidad de visualización puede contener solo un valor de color. Los programas de procesamiento de imágenes digitales basados en píxeles no utilizan pulgadas ni centímetros al procesar imágenes. Todos los procesos de corte, pegado y desplazamiento se aplican a los píxeles. A diferencia de los archivos de imágenes vectoriales, la resolución y el número de píxeles son muy

importantes para los archivos de imágenes puntuales. Todas las operaciones del documento tienen un efecto al cambiar el color o la posición de los píxeles. La duración del proceso depende de la cantidad de píxeles y la complejidad del proceso.

2.3.4.3. Resolución

La resolución es el número de puntos que pueden distinguirlos en unidades de longitud. En los cálculos de resolución, las pulgadas o centímetros se utilizan como unidades de longitud.

2.4. Pasos básicos del procesamiento de imágenes digitales.

El procesamiento de imágenes digitales cubre los temas de convertir una imagen a digital y procesarla. El objetivo principal de los métodos de procesamiento de imágenes digitales es mejorar la imagen para que el ser humano pueda interpretar mejor la imagen y analizar la imagen para que la computadora pueda interpretar la imagen.

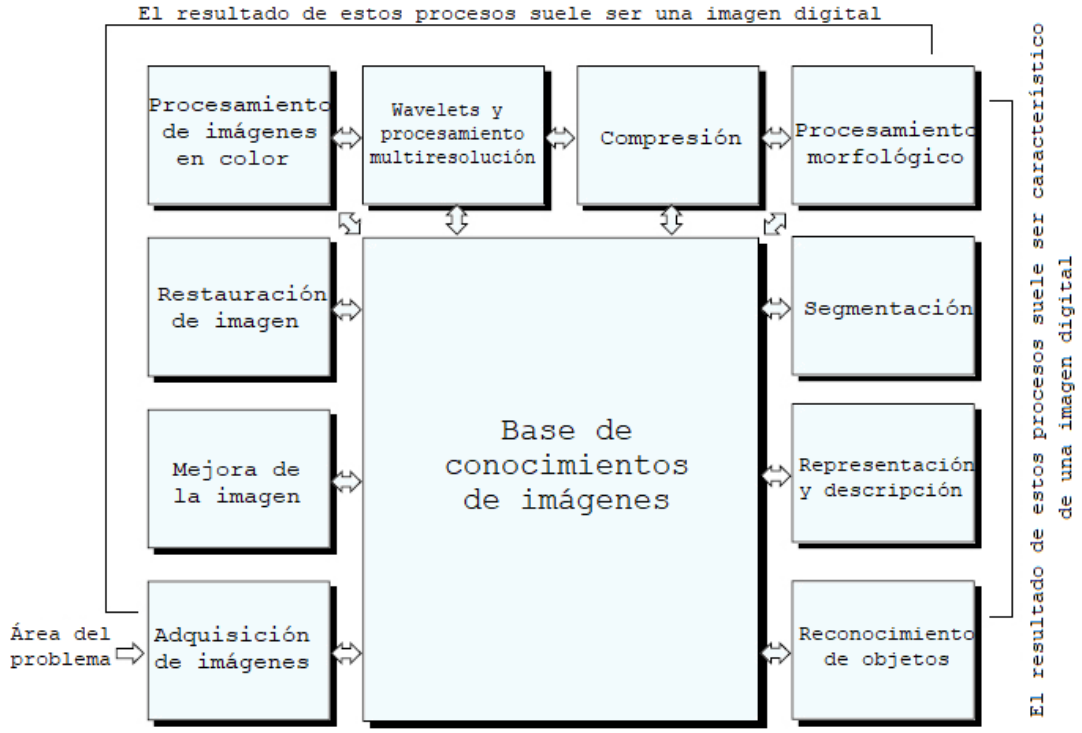


Figura 2. 3: Pasos básicos en el procesamiento de imágenes digitales.
Fuente: (Gonzalez & Woods, 2018; Ribeiro, 2014)

El objetivo de las técnicas de mejora de imágenes es llevar la imagen a un formato más adecuado que la imagen original para un propósito específico.

Tanto la entrada como la salida de un sistema de mejora de imágenes son imágenes digitales. Esta organización se resume en la Figura 3.3. Esta figura no muestra que cada método se aplique a una sola imagen digital, sino que expresa la idea de que todos los métodos pueden aplicarse a imágenes para diferentes propósitos.

2.4.1. Adquisición de imágenes.

El primer paso en una aplicación de procesamiento de imágenes es la adquisición de imágenes, y esto requiere un sensor de imagen. Este dispositivo puede ser un escáner, una cámara digital o una cámara. La selección de la técnica de adquisición de imágenes adecuada es el paso inicial en el desarrollo de cualquier sistema de visión artificial. Las imágenes se utilizan para adquirir información, ya que esta tecnología tiene como objetivo duplicar el papel de la visión humana al percibir y comprender electrónicamente una imagen. (Mahajan et al., 2015)

El sistema de visión artificial implementa cálculos teóricos y algorítmicos mediante los cuales se puede extraer y analizar automáticamente información útil sobre un objeto o escena a partir de una imagen adquirida. La figura 2.4 muestra una configuración típica de adquisición de imágenes y es lo todo investigador necesita para experimentar con aplicaciones de visión artificial. Todos los componentes esenciales están disponibles comercialmente y el precio del sistema elemental puede ser tan bajo como $ 3000.

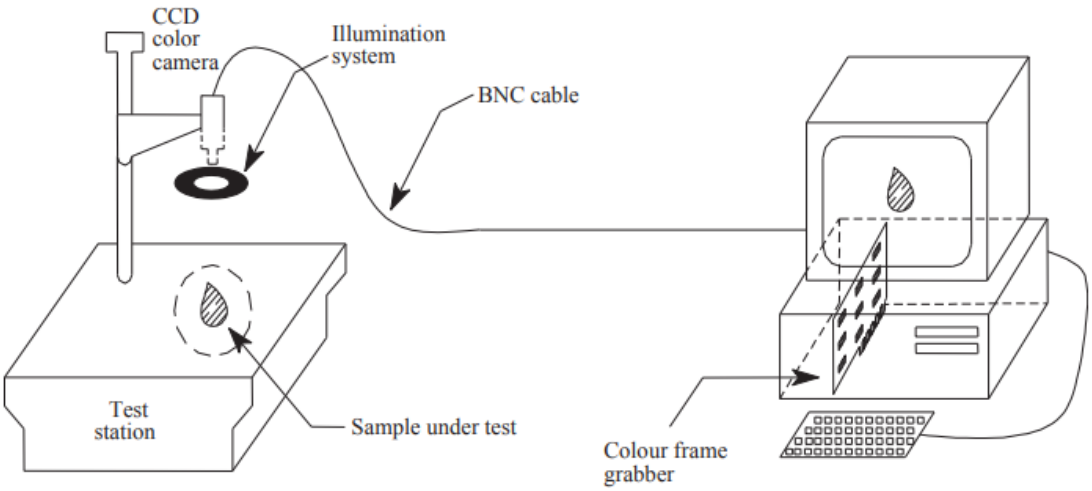


Figura 2. 4: Pasos básicos en el procesamiento de imágenes digitales.
Fuente: (Abdullah, 2016)

La figura 2.5 (a) muestra la geometría de una antena circular, mientras que la figura 2.5 (b) muestra el sistema de adquisición de imágenes, mostrando todos los componentes principales. En la figura 2.5 (b) se observa que el sistema de adquisición de imágenes consta de un multiplexor (32 a 1) que cambia automáticamente las líneas de transmisión y recepción. El multiplexor está conectado a un analizador de redes vectoriales (VNA), que proporciona las mediciones en el dominio del tiempo. También se utiliza un amplificador de banda ancha de 20 dB para mejorar la potencia de la señal recibida. (Abdullah, 2016)

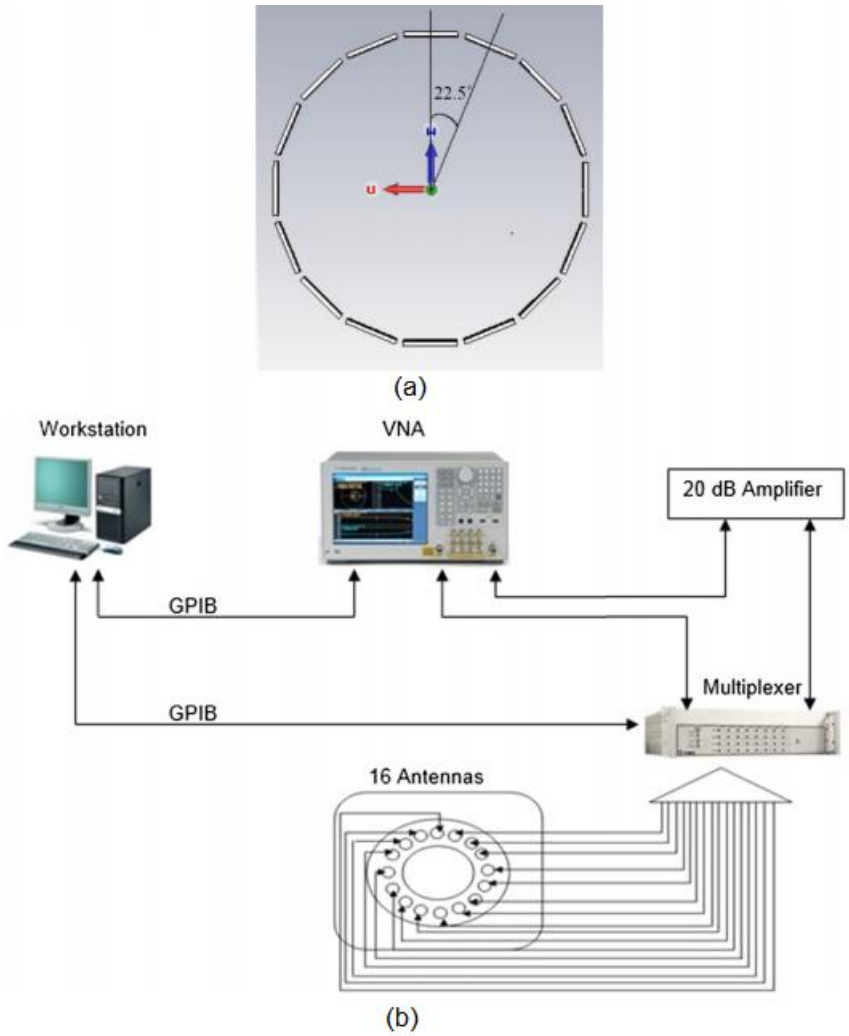


Figura 2. 5: Pasos básicos en el procesamiento de imágenes digitales.
Fuente: (Abdullah, 2016)

2.4.2. Mejoramiento de imágenes.

La mejora de imágenes es una de las áreas más simples y utilizadas del procesamiento de imágenes digitales. De hecho, la idea detrás de las técnicas de refuerzo es resaltar detalles discretos o resaltar algunas de las

características orientadas al objetivo en una sola imagen. Aumentar el contraste (contraste) de una imagen digital es uno de los métodos de mejora de imágenes más utilizados. La mejora de la imagen es un área subjetiva del procesamiento de imágenes.

La figura 2.6 es un ejemplo de mejora de imagen utilizando la imagen de Lena. (a) es la imagen original. (b) es el resultado del filtro de mediana adaptativo (AMF), (c) es el resultado del Filtro de Wallis (WF), y (d) es el resultado de la combinación de AMF y WF denominado AWF en el trabajo propuesto por (Tan, 2016). La imagen original de Lena contiene ruidos y los detalles en las regiones oscuras no son claros.

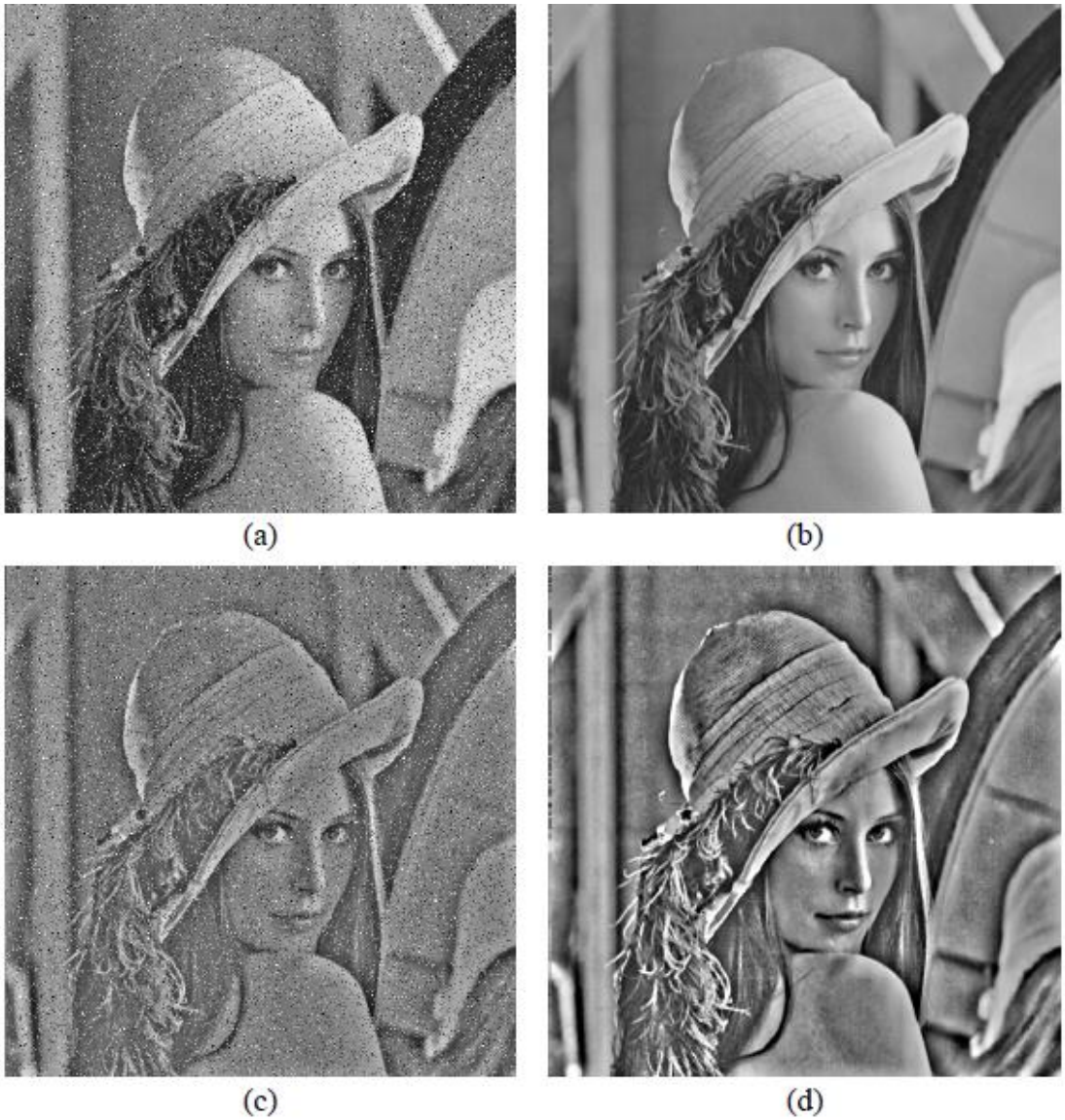


Figura 2. 6: Ejemplo de mejora de la imagen de Lena.
Fuente: (Tan, 2016)

Las herramientas de mejora de imágenes a menudo se clasifican en operaciones puntuales y operaciones espaciales. Las operaciones de puntos incluyen estiramiento de contraste, recorte de ruido, modificación de histograma y pseudocoloración. Las operaciones puntuales son, en general, operaciones simples no lineales que son bien conocidas en la literatura de procesamiento de imágenes y se tratan en otra parte. Las operaciones espaciales que se utilizan hoy en día en el procesamiento de imágenes son, por otro lado, operaciones típicamente lineales. (Arce et al., 2009)

El objetivo de la mejora de la imagen es mejorar la utilidad de una imagen para una tarea determinada, como proporcionar una imagen subjetivamente más agradable para la vista humana. En la mejora de la imagen, se hace poco o ningún intento por estimar el proceso real de degradación de la imagen, y las técnicas suelen ser ad hoc. Este proceso no aumenta el contenido de información inherente a los datos. Es un proceso subjetivo. Incluye manipulación de contraste y nivel de gris, reducción de ruido, nitidez de bordes, filtrado, interpolación y ampliación, y pseudocoloración. Las técnicas de mejora de imágenes se pueden dividir en dos categorías: métodos de dominio de frecuencia y métodos de dominio espacial. (Lu & Guo, 2017)

2.4.3. Restauración de imágenes.

La restauración de imágenes digitales es un campo de la ingeniería que se ocupa de los métodos utilizados para recuperar una escena original a partir de observaciones degradadas (Amudha et al., 2012). La restauración de imágenes es un proceso de restauración o recuperación de una imagen degradada utilizando algún conocimiento previo del método de degradación que ha degradado la imagen. Por tanto, el proceso de restauración de la imagen implica la estimación del modelo deteriorado, así como la relevancia del filtrado inverso para restaurar o recuperar la imagen original.

Aunque la imagen reconstruida puede no ser la forma exacta de la imagen original, será la aproximación de la imagen original. La figura 2.7 muestra un ejemplo de una aplicación propuesta por Ma et al., (2012) en la

que implementa un sistema de restauración de imágenes basada en el aprendizaje para imágenes comprimidas.

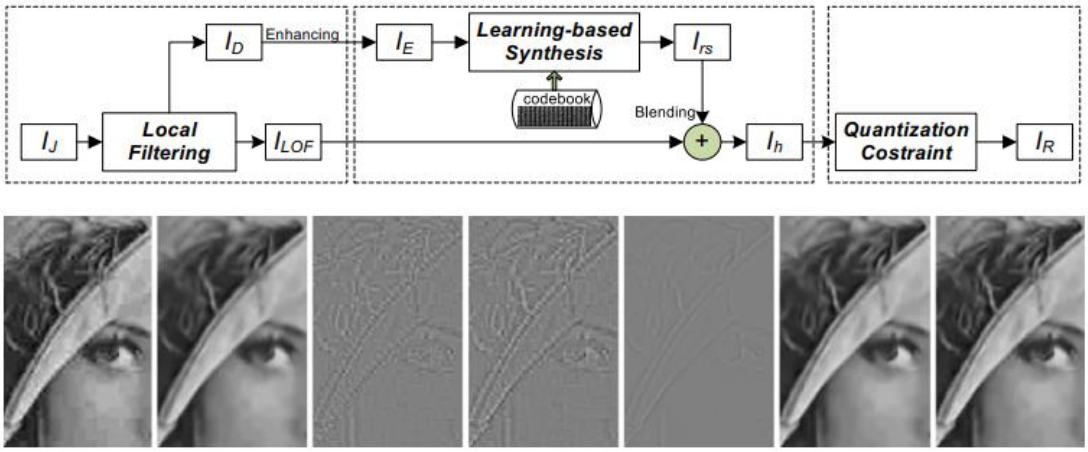


Figura 2. 7: Ejemplo de la restauración de imagen.
Fuente: (Ma et al., 2012)

2.4.4. Procesamiento de imágenes en color.

El primer comercio del procesamiento de imágenes en color de imágenes digitales fue en la industria de los periódicos y las imágenes se transmitían por cables submarinos entre Londres y Nueva York. En el procesamiento inicial de imágenes digitales se producía una cinta codificada mediante una impresora de telégrafos. En el comercio intermedio del procesamiento de imágenes en color que mejora el sistema Bartlane dio como resultado imágenes y procesos fotográficos de mayor calidad.

El objetivo del procesamiento de imágenes en color es mostrar información de imágenes para la percepción humana. También es un paso fundamental del procesamiento de imágenes digitales. Proporciona las aplicaciones de procesamiento de imágenes que extraen las características de los datos de la imagen, a partir de los cuales se puede obtener una descripción, comprensión e información por máquina.

El procesamiento de imágenes en color se utiliza para proporcionar procesamiento de imágenes digitales y está en formato bidimensional. El procesamiento de imágenes digitales define el procesamiento de imágenes que son de naturaleza digital por una computadora digital. La imagen en color proporciona información de imagen para cada píxel. Pixel es una pequeña

unidad de color programable en una imagen en color. Las imágenes en color se almacenan en la memoria como un mapa de tramas o imágenes de gráficos de tramas. En gráficos por computadora, las imágenes de gráficos de trama se almacenan en archivos de imagen con diferentes formatos. Las imágenes de gráficos de trama se pueden ver a través de un monitor, papel u otro medio de visualización. Es una estructura de datos de matriz de puntos que representa una cuadrícula rectangular de píxeles.

El procesamiento de imágenes digitales es muy utilizado en imágenes médicas, recursos terrestres remotos y astronomía, entre otras aplicaciones. El comercio actual de procesamiento de imágenes digitales ha crecido vigorosamente. Hoy se utiliza en geografía, biología, maquinaria nuclear, arqueología, aplicación de la ley, defensa e industria.

2.4.5. Procesamiento de operaciones morfológicas.

La morfología comúnmente denota una rama de la biología que se ocupa de la forma y estructura de animales y plantas. Aquí, la misma palabra morfología se utiliza como herramienta para extraer componentes de imagen que son útiles en la representación y descripción de la forma de la región. También se utiliza para el procesamiento previo o posterior, como el filtrado. El lenguaje de la morfología matemática utiliza la teoría de conjuntos para representar objetos en una imagen.

El procesamiento de imágenes morfológicas (o morfología) describe varias operaciones de procesamiento de imágenes que se ocupan de la forma de las características de una imagen. Estas operaciones eliminan imperfecciones en la imagen y se aplican en imágenes binarias. Las características de forma de la imagen (es decir, bordes, agujeros, esquinas, grietas) se pueden extraer en este proceso utilizando varios elementos estructurantes de forma. Este proceso se utiliza en muchas aplicaciones industriales de visión por computadora, como el reconocimiento de objetos, la segmentación de imágenes y la búsqueda de defectos (Thanki & Kothari, 2019).

Figura 2. 8: Diferencias entre (a) imagen original y (b) operación morfológica por erosión.
Fuente: (MathWorks, 2020)

2.5. Pasos fundamentales del procesamiento de imágenes.

El campo del procesamiento de imágenes digitales se refiere al procesamiento de imágenes digitales utilizando una computadora digital. Considerando que, una imagen digital se compone de un número finito de elementos, cada uno de los cuales tiene una ubicación y un valor particulares. Estos elementos se denominan elementos de imagen, elementos de imagen o píxeles. Antes de pasar a procesar una imagen, se convierte en formato digital. Entre los muchos y variantes posibles pasos de procesamiento y análisis realizados en una imagen digital, nosotros, en este artículo, estamos más interesados en la detección y segmentación de bordes de imágenes, y los procesos de reconocimiento y coincidencia de imágenes.

2.5.1. Detección de bordes

La detección de bordes es un tipo de técnicas de segmentación de imágenes, cuya principal tarea es determinar la presencia de un borde o una línea en una imagen y delinearlas de forma adecuada. La aplicación de detección de bordes conduce a simplificar el procesamiento de la imagen, mientras que se minimiza la cantidad de datos que más preocupan. Generalmente, un borde se define como el borde entre dos regiones de la imagen, donde ocurren grandes cambios de intensidad. La operación de detección comienza con el examen de la discontinuidad local en cada elemento de píxel en una imagen. La amplitud, orientación y ubicación de una

región particular en la imagen que es de interés son características esencialmente importantes de los posibles bordes. Basándose en estas características, el detector tiene que decidir si cada uno de los píxeles examinados es un borde o no.

Un método ampliamente utilizado para la detección de bordes derivados de primer orden se basa en evaluar los gradientes generados a lo largo de dos direcciones ortogonales. De acuerdo con este método, se considera que un borde está presente si el gradiente excede un valor de umbral definido, t=T. Para una ubicación (x, y), el gradiente se puede calcular como las derivadas a lo largo de ambos ejes ortogonales:

$$G(x,y) = \frac{\partial F(x,y)}{\partial x}\cos\theta + \frac{\partial F(x,y)}{\partial y}\sin\theta \qquad (2.1)$$

El gradiente se estima en una dirección normal al gradiente del borde. El gradiente medio espacial se puede escribir como:

$$G(j,k) = \sqrt{[G_R(j,k)]^2 + [G_C(j,k)]^2} \qquad (2.2)$$

En las ecuaciones (2.1) y (2.2), $F(x,y)$ es la imagen de entrada original, mientras que $G(j,k)$ se refiere a la imagen diferencial de salida. Los gradientes de fila y columna discretos más simples $(G_R\ y\ G_C)$ vienen dados por:

$$G_R(j,k) = F(j,k) - F(j,k-1) \qquad (2.3)$$

$$G_C(j,k) = F(j,k) - F(j+1,k) \qquad (2.4)$$

Basado en las ecuaciones (2.3) y (2.44), Roberts propuso un operador para la detección de bordes evaluando los gradientes y las direcciones, para los píxeles de la imagen, usando una matriz [2x2], que se conoce como operadores de máscara. Los elementos de los operadores de máscara de Roberts se muestran en la tabla 2.1.

Tabla 2. 1: Operadores de máscara de Roberts.

1	0
0	-1

eje -x-

0	1
-1	0

eje -y-

Elaborado por: Autor.

Prewitt, Sobel y Canny han desarrollado otros operadores de detectores de bordes, cuyos operadores de máscara para los ejes x e y están tabulados en la tabla 2.2. Los operadores de máscara son matrices [3x3]. Después de aplicar un detector de bordes, se puede llevar a cabo un proceso de suavizado, donde los bordes detectados se someten a un proceso de adelgazamiento y enlace para optimizar los límites que conducen a lo que se conoce como segmentación.

Tabla 2. 2: Operadores de máscara para tres operadores de detectores de bordes.

Nombre del operador	Máscara para eje -x-				Máscara para eje -y-		
Prewitt	-1	0	1		1	1	1
	-1	0	1		0	0	0
	-1	0	1		-1	-1	-1
Sobel	-1	0	1		-1	-2	-1
	-2	0	2		0	0	0
	-1	0	2		1	2	1
Canny	1	2	1		-1	0	1
	0	0	0		-2	0	2
	-1	-2	-1		-1	0	1

Elaborado por: Autor.

2.5.2. Emparejamiento de imágenes

El proceso de emparejamiento se centra en encontrar una correspondencia entre varios conjuntos de datos. Los conjuntos de datos pueden representar imágenes, fotografías, mapas o cualquier otra forma de modelo de objeto. El emparejamiento siempre ha sido un problema desafiante en el área de investigación y desarrollo de imágenes. Algunos factores que pueden plantear problemas en la coincidencia de imágenes incluyen, entre otros; cambios en el contenido de la imagen, rotación del plano, cambio de escala, cambio de iluminación y diferencias causadas por ruido electrónico. El principio básico de la coincidencia es buscar en todos los píxeles el área correcta que es idéntica a una imagen de plantilla determinada.

Capítulo 3: Diseño, configuración y resultados de simulación.

3.1. Descripción general de las aplicaciones prácticas a desarrollarse.

En esta sección se describe brevemente las aplicaciones que se van a desarrollar en las siguientes secciones del capítulo 3. El propósito del capítulo es disponer de herramientas de aplicación práctica que se van a implementar en el software MatLab y LabVIEW.

3.2. Aplicación práctica 1: operaciones básicas de imágenes en MatLab.

El propósito principal de esta parte es familiarizarse con las operaciones básicas de lectura/escritura en imágenes en el software MatLab R2018a y aprender el tipo de imágenes admitidas, así como algunas operaciones básicas sobre ellas.

Las imágenes tienen un tamaño en píxeles, que suelen expresarse como $N \times M$, es decir, que en MatLab, cada imagen no es más que una matriz, tal como se muestra en la figura 3.1. Cada elemento de la matriz es el tono gris para imágenes en B/N, o, si es una imagen en color, se puede representar mediante tres matrices, donde cada matriz representa uno de los tres colores primarios (rojo, verde, azul o RGB).

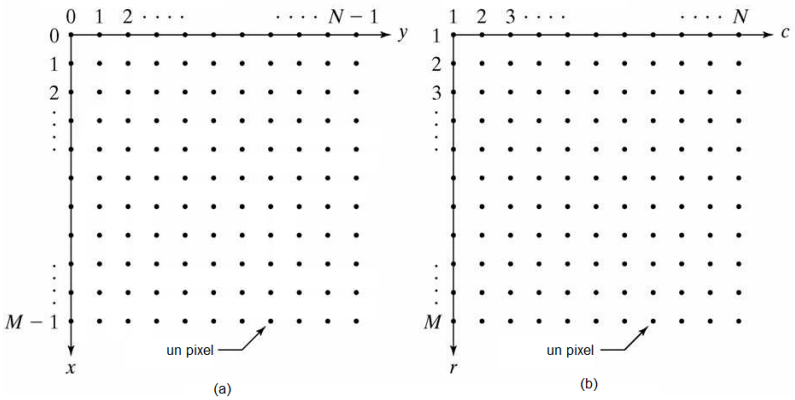


Figura 3. 1: Representación matricial del procesamiento de imágenes (a) en textos y (b) ToolBox de MatLab.
Elaborado por: Autor.

Para leer una imagen del disco, se utiliza la función imread (). A continuación, se muestra el script que permite leer desde el disco de un computador.

```
I=imread('mongolia.jpg'); % leer imagen directamente
I=imread('C:\imagenes\mongolia.jpg'); % leer imagen desde una ruta
I=imread('/home/user/mongolia.jpg');
```

Si la imagen es B/N, MatLab creará una matriz I, donde cada elemento tendrá un valor de gris. La matriz tendrá la siguiente forma:

$$I = \begin{pmatrix} I(1,1) & I(1,2) & \cdots & I(1,N) \\ I(2,1) & I(2,2) & \cdots & I(2,N) \\ \vdots & \vdots & \vdots & \vdots \\ I(M,1) & I(M,2) & \cdots & I(M,N) \end{pmatrix}$$

Ahora se puede comprobar el tamaño de la imagen usando la función size (), o también guardar el tamaño en una variable usando [M, N]=size (), o también guardar la cantidad de colores de la imagen con el mismo comando [M, N, c]=size (). Y si se desea información detallada sobre la imagen se utiliza la función 'whos'. En el siguiente script se muestra las diferentes formas de leer y guardar los datos de una imagen, y también ver los detalles de la imagen.

```
size(I)
[M, N] = size(I);
[M, N, c] = size(I);
whos I
```

Para visualizar imágenes en MatLab se utiliza la función imshow (). Es posible que quiera mostrar la imagen aplicándole efectos a través de la función imshow (I, G). Donde, I es la imagen y G es el número de niveles de intensidad a mostrar. Si no se especifica G, se utiliza por default 256. A continuación, se muestra el script que permite mostrar la imagen original (mongolia.jpg) a colores, después se la convierte a escala de grises y finalmente se muestra los efectos de intensidad de la imagen, tal como se muestra en la figura 3.2.

```
I=imread('mongolia.jpg'); % leemos imagen original
Irgb = rgb2gray(I); % convertimos imagen a escala de grises
figure(1)
subplot(1,3,1); imshow(I)
xlabel('(a) Imagen original')
subplot(1,3,2); imshow(Irgb)
xlabel('(b) Imagen a escala de grises')
subplot(1,3,3); imshow(Irgb,[0,150])
xlabel('(c) Imagen a escala de grises con intensidad 150')
```

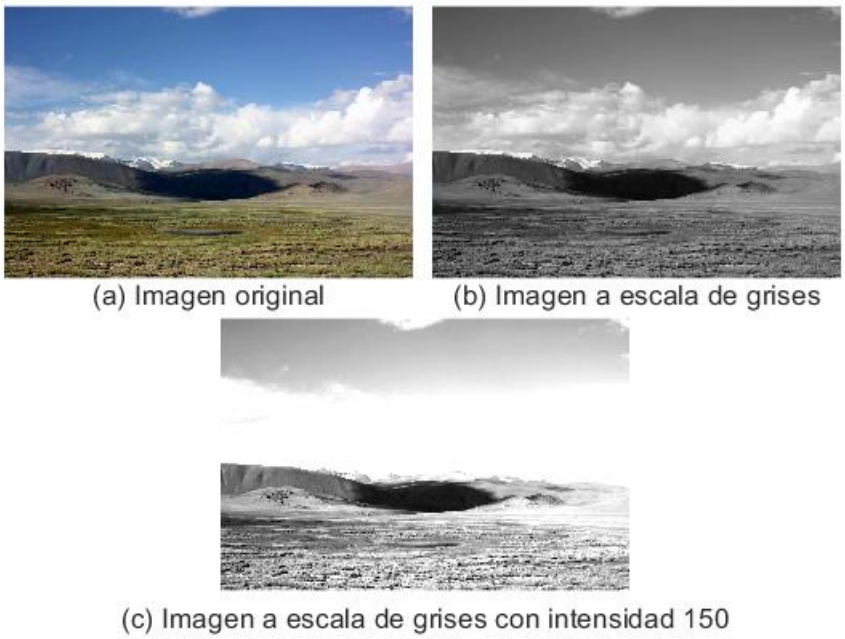


Figura 3. 2: Visualización de la imagen ‘mongolia.jpg’ (a) colores, (b) escala de grises, y (c) con intensidad 150.
Elaborado por: Autor.

Si se desea ampliar el rango dinámico, se puede dejar los corchetes vacíos, es decir, que con eso se utiliza el valor mínimo de intensidad de la imagen como límite bajo y el valor máximo como límite alto. En la figura 3.3 se visualiza la imagen con el rango dinámico de la imagen ampliado.

Figura 3. 3: Visualización de la imagen ‘mongolia.jpg’ con ampliación del rango dinámico de intensidad.
Elaborado por: Autor.

Otra función interesante es pixval, que afecta a la última imagen cargada. Cuando lo ejecuta en la consola, puede ver el valor de la intensidad cuando el mouse se desplaza sobre un píxel en particular. También puede medir la distancia euclidiana entre dos puntos.

Finalmente, cuando se carga una nueva imagen, MatLab sobrescribe la ventana de la imagen cargada previamente en workspace. Se puede grabar imágenes con extensiones ‘tif’, ‘jpg’, entre otras, y en cuyo caso también se puede especificar la calidad mediante imwrite (I, 'name.jpg', 'quality', q). Donde ‘q’ es un número entre 0 y 100 (0 = mayor compresión, peor calidad, 100 = menor compresión, mejor calidad). Para evitar esto y abrir una nueva imagen en una ventana independiente, se implementa el siguiente script en MatLab.

```
figure, imshow(I)
imwrite(I, 'practica1', 'tif')
imwrite(I, 'practica1.jpg', 'quality', 100)
imfinfo practica1.jpg
```

Al ejecutar esas líneas de programación se van a generar dos archivos una con extensión ‘tif’ y otra con extensión ‘jpg’. Abajo se muestra el resultado que se obtiene para saber de la información de la imagen ‘practica1.jpg’. En la tabla 3.1 se muestran los formatos de imagen admitidos.

```
          Filename: 'C:\Users\practica1.jpg'
      FileModDate: '14-Feb-2021 10:16:17'
         FileSize: 100821
           Format: 'jpg'
    FormatVersion: ''
            Width: 550
           Height: 366
         BitDepth: 24
        ColorType: 'truecolor'
  FormatSignature: ''
  NumberOfSamples: 3
     CodingMethod: 'Huffman'
    CodingProcess: 'Sequential'
          Comment: {}
```

De forma predeterminada, MatLab almacena todas las variables numéricas como valores de punto flotante de doble precisión de 8 bytes (64 bits). Estas variables tienen tipo de datos (clase) double. Por ejemplo, en la tabla 3.2 se muestra la clasificación de los tipos de variables numéricas que se generan al cargar imágenes en MatLab con su respectiva descripción y valores. Por ejemplo, una matriz cuyos valores se han escalado para

representar la intensidad pueden ser uint8 o uint16 (véase la tabla 3.2). Si son dobles, los valores se escalan en el rango [0,1].

Tabla 3. 1: Listado de formatos de imagen admitidos en MatLab.

Nombre de formato	Descripción	Extensiones reconocidas
TIFF	Formato de archivo de imagen etiquetado	'tif', 'tiff'
JPEG		'jpg', 'jpeg'
GIF	formato de gráficos intercambiables	'gif'
BMP	Mapa de bits de Windows	'bmp'
PNG	Gráficos de red portátiles	'png'

Elaborado por: Autor.

Tabla 3. 2: Descripción de las clases de datos en MatLab.

Tipos de variables	Descripción
double	Precisión doble, números en coma flotante que varían en un rango de -10308 a 10308 (8 bytes por elemento)
uint8	Enteros de 8 bits en el rango de [0,255] (1 byte por elemento)
uint16	Enteros de 16 bits en el rango de [0, 65535] (2 bytes por elemento)
uint32	Enteros de 32 bits en el rango de [0, 4294967295] (4 bytes por elemento)
int8	Enteros de 8 bits en el rango de [-128, 127] (1 byte por elemento)
int16	Enteros de 16 bits en el rango de [-32768, 32767] (2 bytes por elemento)
int32	Enteros de 32 bits en el rango de [-2147483648,2147483647] (4 bytes por elemento)
single	Número de coma flotante de precisión simple, con valores en el rango de -1038 a 1038 (4 bytes por elemento)
char	Caracteres (2 bytes por elemento)
logical	Los valores son 0 o 1 (1 byte por elemento)

Elaborado por: Autor.

Para el caso de imágenes binarias, cuyos valores son solo '0' o '1' se representan en MatLab utilizando valores lógicos. Se pueden convertir

matrices a diferentes formatos de acuerdo con el tipo de imágenes, para esto se debe revisar las funciones en la tabla 3.3.

Tabla 3. 3: Clasificación de conversión de tipos de imagen.

Comando	Convertir en	Tipo de entrada válido
im2uint8	uint8	logical, uint8, uint16 y double
im2uint16	uint16	logical, uint8, uint16 y double
mat2gray	double ([0,1])	double
im2double	double	logical, uint8, uint16 y double
im2double	logical	uint8, uint16 y double

Elaborado por: Autor.

3.3. Aplicación práctica 2: efectos de los cambios en la resolución espacial y la cuantificación en las imágenes de nivel gris.

En esta sección, se tienen dos propósitos que son el estudio del efecto de cambiar el número de niveles de gris sobre la calidad de las imágenes, y el estudio del efecto de cambiar la resolución espacial sobre la calidad de las imágenes, utilizando dos métodos: (1) interpolación del vecino más cercano, y (2) Interpolación bilineal. Antes de proceder con la parte práctica 2, se describe brevemente el cambio de niveles de grises y el cambio de la resolución espacial:

a. Cambiar el número de niveles de grises:

Se ha descubierto (tanto subjetiva como objetivamente) que la calidad de una imagen de niveles de gris se ve dramáticamente afectada por su resolución de nivel de grises. En otras palabras, aumentar el número de bits por píxel tiene un gran efecto en la mejora de la calidad de las imágenes de nivel de grises. Esto se debe a que un mayor número de niveles de grises proporcionaría una transición suave a lo largo de los detalles de la imagen y, por lo tanto, mejoraría su calidad para el ojo humano.

b. Cambiar la resolución espacial:

Cambiar la resolución espacial de una imagen digital, mediante la ampliación (zoom) o reducción, es una operación de gran importancia en una amplia gama de aplicaciones (es decir, en cámaras digitales,

procesamiento de imágenes biomédicas e imágenes astronómicas, entre otras). Simplemente, hacer zoom y encoger son las operaciones de sobremuestreo y submuestreo de una imagen digital, respectivamente. Hacer zoom en una imagen digital requiere dos pasos: la creación de nuevas ubicaciones de píxeles y la asignación de niveles de grises a esas nuevas ubicaciones.

La asignación de niveles de grises a las nuevas ubicaciones de píxeles es una operación de gran desafío. Se puede realizar utilizando dos enfoques:

1) Interpolación de vecino más cercano: a cada píxel de la imagen ampliada se le asigna el valor de nivel de grises de su píxel más cercano en la imagen original.
2) Interpolación bilineal: el valor de cada píxel en la imagen ampliada es un promedio ponderado de los valores de nivel de grises de los píxeles en la vecindad de 2 por 2 más cercana, en la imagen original.

En esta aplicación práctica 2, la resolución espacial de una imagen de nivel de grises se reducirá de 256 a 16, cada vez en un factor de 2. El detalle más pequeño discernible se medirá subjetivamente. Para comparar estos efectos de contracción, las imágenes reducidas deben redimensionarse nuevamente al tamaño original (256 X 256), luego, se observa la calidad de las imágenes y los efectos. En MatLab, esto se puede realizar utilizando la función imresize (). Esta función acepta un factor mayor que 1.0 para hacer zoom y menor que 1.0 para reducir.

Se va a desarrollar dos aplicaciones experimentales en MatLab, y se utiliza solo una imagen: Mongolia.jpg. A continuación, se muestra el script (parte 1) que convierte una imagen en niveles de grises o binaria en una imagen indexada y reducir el número de niveles de grises. En la figura 3.4 se muestran las imágenes convertidas y reducidas a niveles de grises.

```
% Trabajo de titulación - Práctica 2:
M=imread('mongolia.jpg'); % leemos la imagen mongolia.jpg
M=rgb2gray(M); % convertir imagen a escala de grises
% una imagen de 256 niveles de grises:
[I256,map256]=gray2ind(M,256);
% una imagen de 128 niveles de grises:
[I128,map128]=gray2ind(M,128);
% una imagen de 64 niveles de grises:
[I64,map64]=gray2ind(M,64);
figure(1);
subplot(221); subimage(M);
xlabel('(a) Imagen original');
subplot(222); subimage(I256,map256);
xlabel('(b) Imagen a 256 niveles');
subplot(223); subimage(I128,map128);
xlabel('(c) Imagen a 128 niveles');
subplot(224); subimage(I64,map64);
xlabel('(d) Imagen a 64 niveles');
```

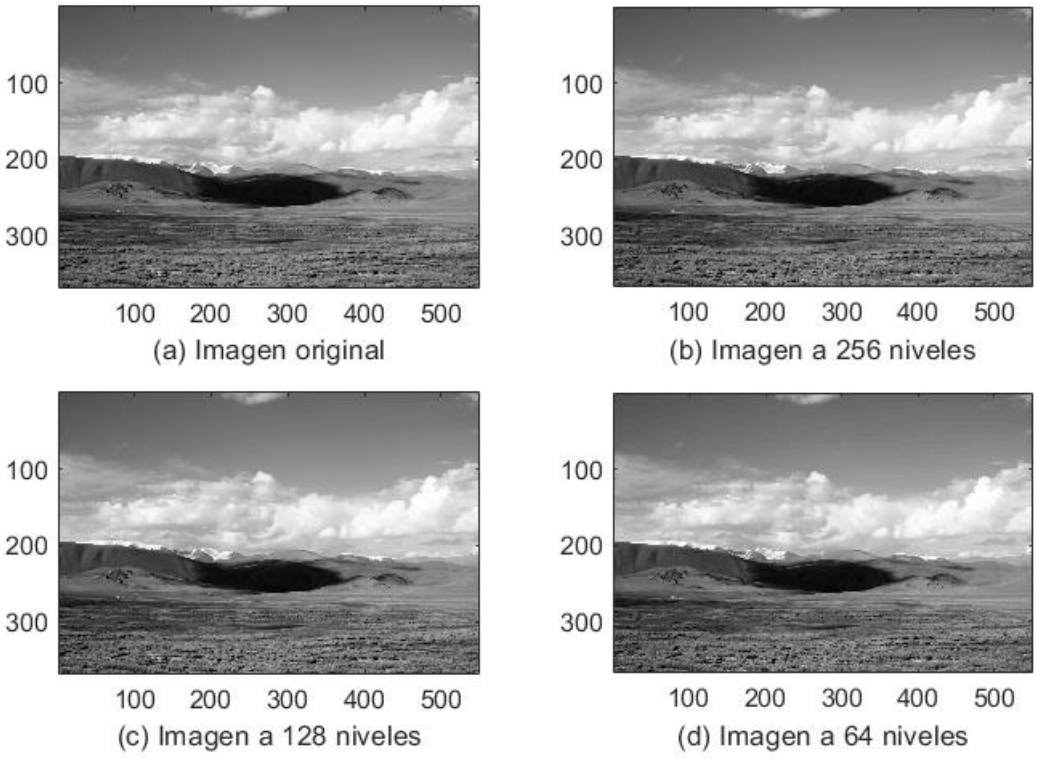


Figura 3. 4: Conversión (a) nivel de grises, y reducción a (b) 256, (c) 128 y (d) 64 niveles de grises.
Elaborado por: Autor.

A continuación, se muestra la segunda parte del código de la reducción a 32, 16 y 8 niveles de grises. La figura 3.5 muestra las imágenes convertidas y reducidas a niveles de grises. Se observa

claramente que las imágenes se van degradando debido a la reducción de niveles de grises por un factor de 2.

```
% una imagen a 32 niveles de grises:
[I32,map32]=gray2ind(M,32);
% una imagen a 16 niveles de grises:
[I16,map16]=gray2ind(M,16);
% una imagen a 8 niveles de grises:
[I8,map8]=gray2ind(M,8);
figure(2);
subplot(221); subimage(M);
xlabel('(a) Imagen original');
subplot(222); subimage(I32,map32);
xlabel('(b) Imagen a 32 niveles');
subplot(223); subimage(I16,map16);
xlabel('(b) Imagen a 16 niveles');
subplot(224); subimage(I8,map8);
xlabel('(b) Imagen a 8 niveles');
```

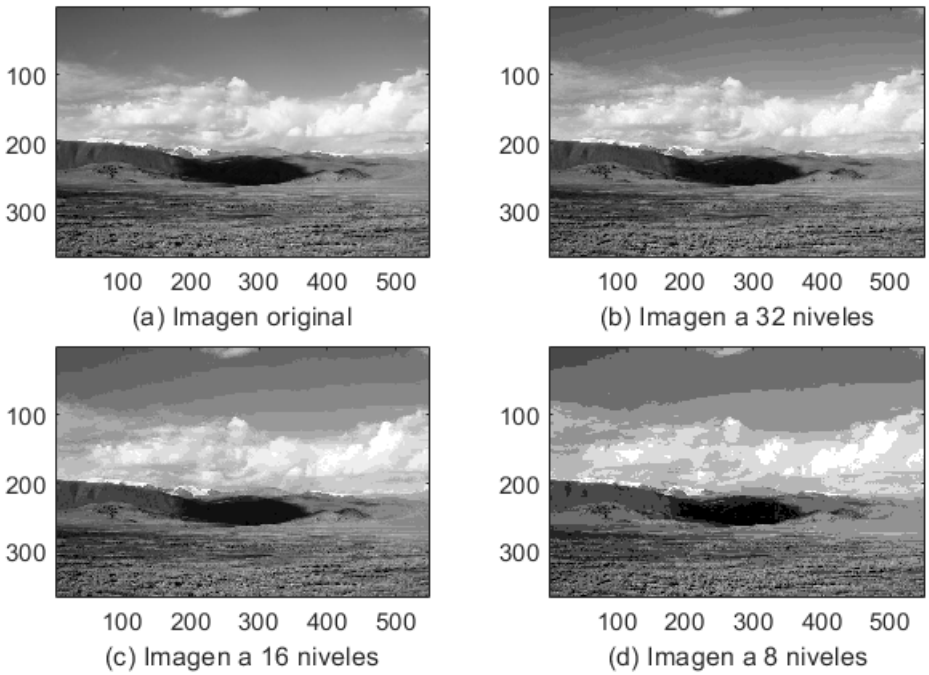


Figura 3. 5: Conversión (a) nivel de grises, y reducción a (b) 32, (c) 16 y (d) 8 niveles de grises.
Elaborado por: Autor.

Ahora se muestra la tercera parte del código de la reducción a 8, 4 y 2 niveles de grises. La figura 3.6 muestra las imágenes convertidas y reducidas a niveles de grises. En comparación con las figuras 3.4 y 3.5 se observa una mayor degradación de la imagen 'mongolia.jpg' para 8, 4 y 2 niveles de grises.

```
% una imagen a 4 niveles de grises:
[I4,map4]=gray2ind(M,4);
% una imagen a 2 niveles de grises:
[I2,map2]=gray2ind(M,2);
figure(3);
subplot(221); subimage(M);
xlabel('(a) Imagen original');
subplot(222); subimage(I8,map8);
xlabel('(b) Imagen a 8 niveles');
subplot(223); subimage(I4,map4);
xlabel('(c) Imagen a 4 niveles');
subplot(224); subimage(I2,map2);
xlabel('(d) Imagen a 2 niveles');
```

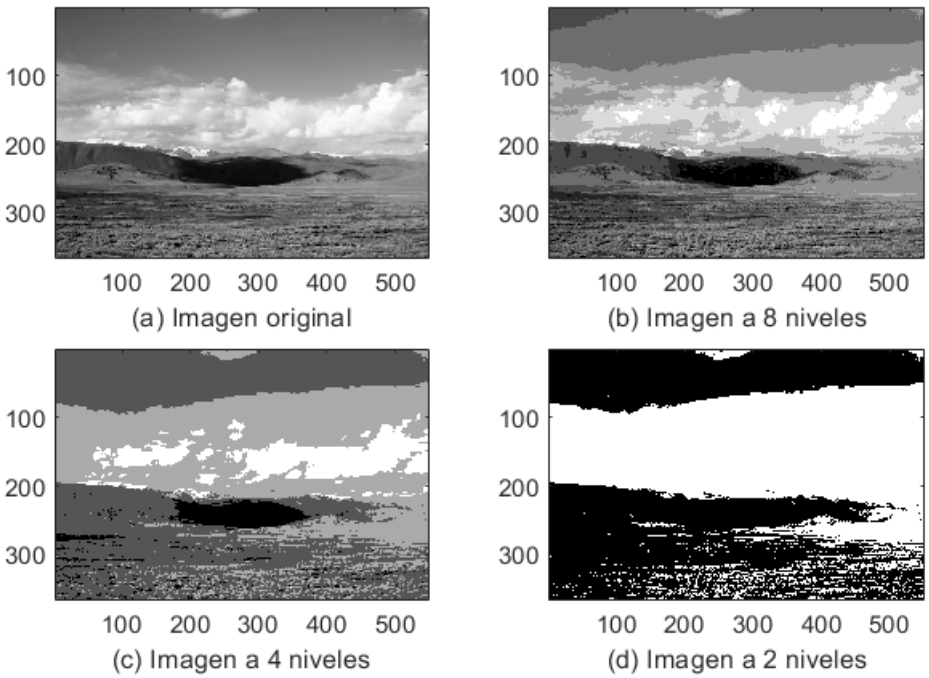


Figura 3. 6: Conversión (a) nivel de grises, y reducción a (b) 8, (c) 4 y (d) 2 niveles de grises.
Elaborado por: Autor.

Ahora, se va a desarrollar el script para la resolución espacial de la imagen ‘mongolia.jpg’ utilizando imresize () que devuelve la imagen M1 en una escala multiplicada por el tamaño de M. La imagen de entrada M puede ser una imagen en escala de grises, RGB o binaria. Si M tiene más de dos dimensiones, imresize solo cambia el tamaño de las dos primeras dimensiones. Si la escala está en el rango [0, 1], M1 es menor que M. Si la escala es mayor que 1, M1 es mayor que M. Por defecto, imresize () usa interpolación bicúbica. El siguiente script permite reducir el tamaño de imagen M (original ‘mongoli.jpg’) mediante

la interpolación del vecino más cercano. La figura 3.7 muestra cuatro imágenes reducidas en tamaño para en 0.5, 0.25, 0.125 y 0.0625 respetivamente. Como se puede apreciar cada imagen fue reducida del nivel de grises de 'mongolia.jpg'. Aunque, estás imágenes al final disminuye la resolución a medida que aumenta el porcentaje de reducción.

```
% Segunda aplicación - Práctica 2:
close all; clear all;
M=imread('mongolia.jpg'); % leemos la imagen mongolia.jpg
M1=rgb2gray(M); % convertir imagen a escala de grises
% Reducir el tamaño de M usando la interpolación de vecino
% más cercano
I128=imresize(M1,0.5); imshow(I128),pause
I64=imresize(M1,0.25);close,imshow(I64),pause
I32=imresize(M1,0.125);close,imshow(I32),pause
I16=imresize(M1,0.0625);close,imshow(I16),pause
```

Figura 3. 7: Reducción del tamaño de la imagen 'mongolia.jpg' en un factor de (a) 50%, (b) 25%, (c) 12.5% y (d) 6.25%.
Elaborado por: Autor.

En el siguiente script se procede a cambiar el tamaño de imágenes reducidas al tamaño original y se las compara en las figuras 3.8 y 3.9. La figura 3.8 muestra la comparación entre imagen original y reducida a 128x128.

```
I16=imresize(I16,16);
I32=imresize(I32,8);
I64=imresize(I64,4);
I128=imresize(I128,2);
figure(2);
subplot(121); subimage(M1);
xlabel('(a) Imagen original');
subplot(122); subimage(I128);
xlabel('(b) Imagen reducida 128x128');
```

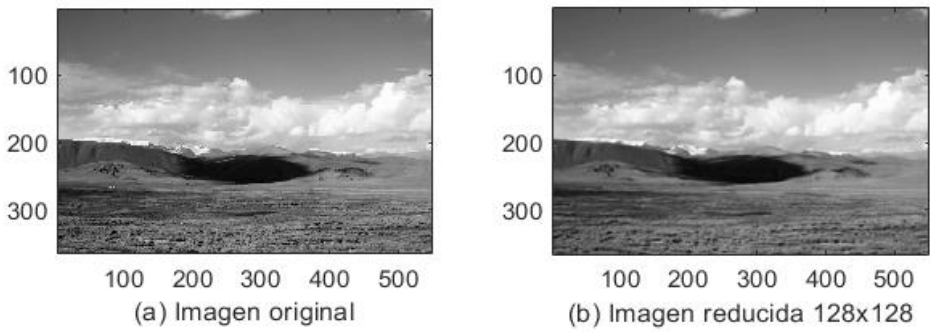


Figura 3. 8: Comparación de imagen (a) original y (b) reducida a 128x128.
Elaborado por: Autor.

La figura 3.9 muestra la comparación entre imagen original e imágenes reducidas a 64x64, 32x32 y 16x16. A diferencia de la figura 3.8 en la que no hay cambios significativos, la figura 3.9 muestra claramente los cambios en la resolución de las imágenes reducidas y la mayor distorsión sucede en la compresión de 16x16.

```
figure(3);
subplot(221); subimage(M1);
xlabel('(a) Imagen original');
subplot(222); subimage(I64);
xlabel('(b) Imagen reducida 64x64');
subplot(223); subimage(I32);
xlabel('(c) Imagen reducida 32x32');
subplot(224); subimage(I16);
xlabel('(d) Imagen reducida 16x16');
```

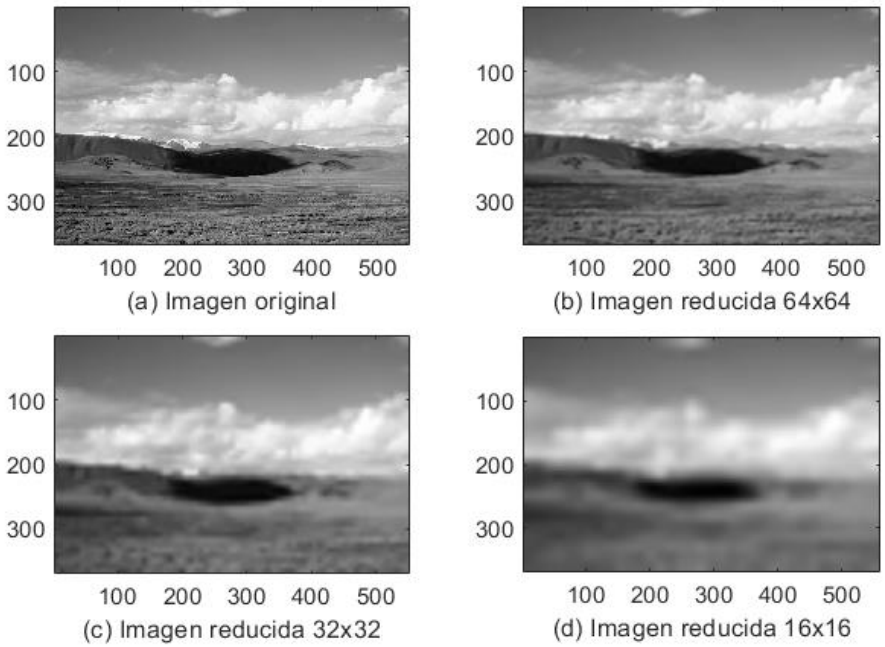


Figura 3. 9: Comparación de imagen (a) original y reducidas a (b) 64x64, (c) 32x32 y (d) 16x16.
Elaborado por: Autor.

Otro método de reducción de imágenes mediante la función imresize () es la interpolación bilineal (valor del píxel de salida es un promedio ponderado de píxeles en la vecindad más cercana de 2 por 2). A continuación, se muestra el script para interpolación bilineal.

```
I128_b=imresize(M1,0.5,'bilinear');
I128_b=imresize(I128_b,2,'bilinear');
I64_b=imresize(M1,0.25,'bilinear');
I64_b=imresize(I64_b,4,'bilinear');
I32_b=imresize(M1,0.125,'bilinear');
I32_b=imresize(I32_b,8,'bilinear');
I16_b=imresize(M1,0.0625,'bilinear');
I16_b=imresize(I16_b,16,'bilinear');
subplot(241); subimage(I128); axis off;
xlabel('(a) Imagen a 128x128');
subplot(242); subimage(I64); axis off;
xlabel('(b) Imagen a 64x64');
subplot(243); subimage(I32); axis off;
xlabel('(c) Imagen a 32x32');
subplot(244); subimage(I16); axis off;
xlabel('(d) Imagen a 16x16');
subplot(245); subimage(I128_b); axis off;
xlabel('(e) Imagen a 128x128 bilineal');
subplot(246); subimage(I64_b); axis off;
xlabel('(f) Imagen a 64x64 bilineal');
subplot(247); subimage(I32_b); axis off;
xlabel('(g) Imagen a 32x32 bilineal');
subplot(248); subimage(I16_b); axis off;
xlabel('(h) Imagen a 16x16 bilineal');
```

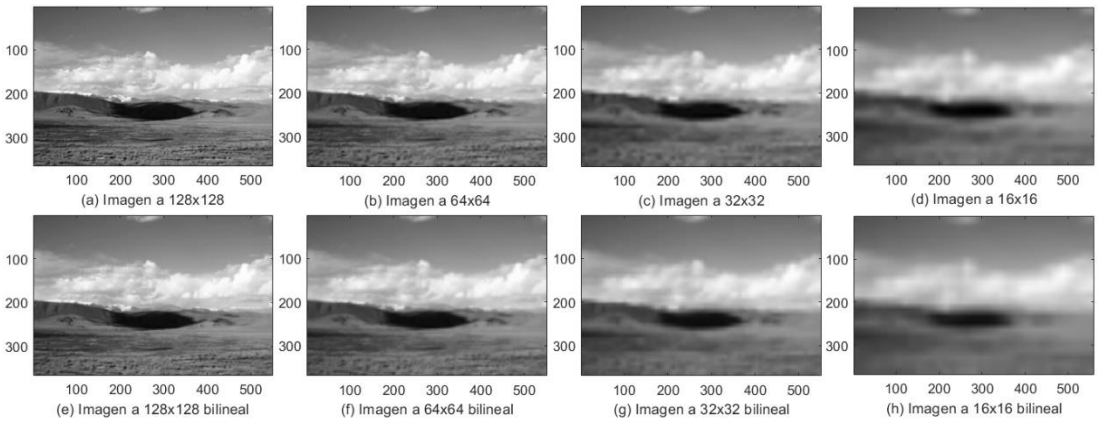


Figura 3. 10: Comparación de reducción de imágenes usando interpolación bilineal.
Elaborado por: Autor.

En la tabla 3.4 se muestran los tiempos de simulación para los tres algoritmos de interpolaciones vecino más cercano, bilineal y bicúbica.

Tabla 3. 4: Complejidad del tiempo de interpolación.

Algoritmos de interpolación	Tiempo (s)
Vecino más cercano	0.085
Bilineal	0.151
Bicúbica	0.181

Elaborado por: Autor.

La interpolación bilineal toma un promedio ponderado de los cuatro píxeles vecinos para calcular su valor interpolado final. El resultado es una imagen mucho más suave que la imagen original. Cuando todas las distancias de píxeles conocidas son iguales, entonces el valor interpolado es simplemente su suma dividida por cuatro. Esta técnica realiza una interpolación en ambas direcciones, horizontal y vertical. Esta técnica da mejores resultados que la interpolación del vecino más cercano y requiere menos tiempo de cálculo en comparación con la interpolación bicúbica.

La interpolación bicúbica va un paso más allá de lo bilineal al considerar la vecindad 4x4 más cercana de píxeles conocidos para un total de 16 píxeles. Dado que se encuentran a varias distancias del píxel desconocido, los píxeles más cercanos reciben una mayor ponderación en el cálculo. Bicúbica produce imágenes notablemente más nítidas que los dos métodos anteriores y es quizás la combinación ideal de tiempo de procesamiento y calidad de salida. Por esta razón, es un estándar en muchos programas de edición de imágenes, incluido Adobe Photoshop, controladores de impresora e interpolación en la cámara.

3.4. Aplicación práctica 3: espacios de color en imágenes e histogramas.

En esta sección se realiza la práctica para la representación de colores, RGB (rojo, verde y azul). Una imagen digital es una señal

bidimensional y se describe por el brillo o el color de los elementos de la imagen ("píxeles") indexados por coordenadas horizontales y verticales. De forma predeterminada, MatLab almacena una imagen en color utilizando 3 matrices, cada una de las cuales representa los componentes rojo, verde y azul de los píxeles.

Una componente R/G/B se representa por defecto usando 8 bits (entero de 8 bits sin signo o 'uint8' en el tipo de datos en MatLab). La representación de 8 bits da una escala de 0 a 255, siendo 0 el más oscuro y 255 el brillo en el componente de color particular. El propósito principal de esta aplicación práctica es demostrar cómo se descomponen los colores R/G/B en imágenes, así como el modelo de color de matiz-saturación-valor *(Hue-Saturation-Value, HSV)* y también visualizar el histograma de colores de imágenes.

```
% Trabajo de titulación - Práctica 3:
% leemos imagen artemoderna2.png
Art = imread('artemoderna2.png');
figure(1); imshow(Art); size(Art)
Art1_r = Art( :, :, 1); % separar color rojo
Art1_g = Art( :, :, 2); % separar color verde
Art1_b = Art( :, :, 3); % separar color azul

figure(2),
subplot(2,2,1); imshow(Art);
xlabel('(a) Imagen original');
subplot(2,2,2); imshow(Art1_r);
xlabel('(b) Separación de color rojo'); colorbar
subplot(2,2,3); imshow(Art1_g);
xlabel('(c) Separación de color verde'); colorbar
subplot(2,2,4); imshow(Art1_b);
xlabel('(d) Separación de color azul'); colorbar
```

En la figura 3.11 se muestra la imagen original ‘artemoderna2.png’ que tiene cuatro colores (rojo, verde, azul y amarillo). La figura 3.12 muestra las imágenes con su respectiva separación de color RGB. Se observa la separación de color rojo (ver figura 3.12 (b)), color verde (ver figura 3.12 (c)) y color azul (ver figura 3.12 (d)). Se puede ver que el color amarillo no fue descompuesto, ya que el algoritmo solo descompone de la imagen los colores RGB.

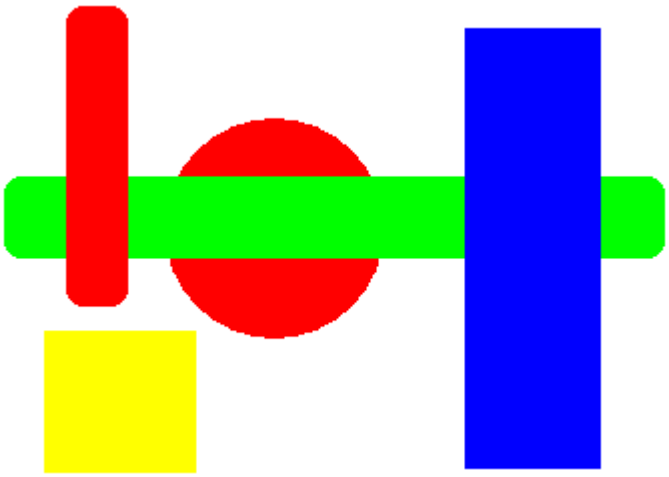


Figura 3. 11: Imagen original para separación del espacio de color.
Elaborado por: Autor.

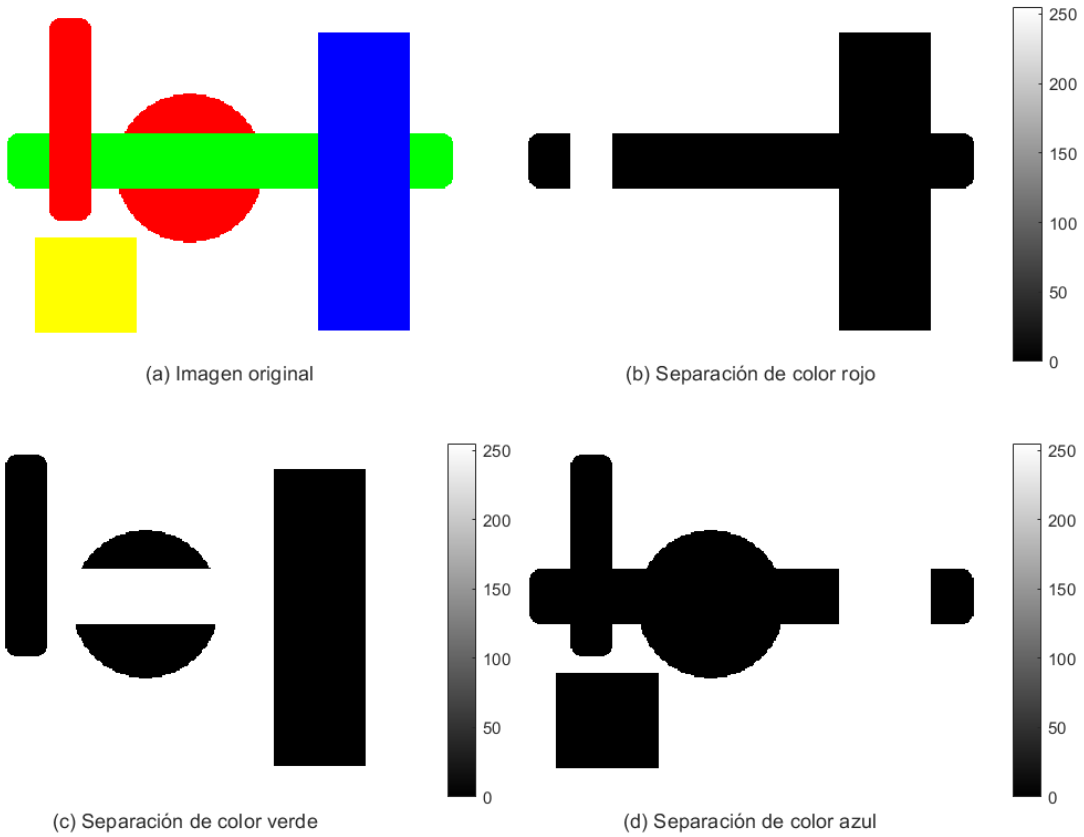


Figura 3. 12: Imagen original para separación del espacio de color.
Elaborado por: Autor.

En el procesamiento de imagen/video, también se lo conoce como canales R/G/B. Una matriz es esencialmente una matriz indexada por dos variables de indexación, típicamente para fila y columna. Cada una de las tres matrices de color se puede especificar mediante el tercer número de indexación; observe el número 1 en el ejemplo img (:,:, 1),

y así sucesivamente. Las dimensiones horizontal y vertical de cada matriz de color coinciden con el tamaño de la imagen en unidades de píxeles. Por lo tanto, la imagen de color general se almacena en una matriz de tres variables de indexación, tal como se puede ver en el resumen de Workspace en MatLab, o ejecutando la función size.

```
% Trabajo de titulación - Práctica 3 prueba 2:
close all; clear
Art1_1=imread('artemoderna2.png'); % Leemos imagen
figure(3)
imshow (Art1_1) %
Art1_1=im2double(Art1_1); % convierte imagen a doble precisión
xlabel('Imagen de objetos original')
R=Art1_1(:,:,1); G=Art1_1(:,:,2); B=Art1_1(:,:,3);
figure(2)
subplot(3,3,1), imshow(Art1_1)
% Creación de máscaras utilizando el espacio de color HSV
% Convertir formato de color RGB al formato de color HSV
HSV=rgb2hsv(Art1_1);
% Separación de las componentes de tono (H), saturación (S)
% y valor de intensidad (V)
H=HSV(:,:,1); % matiz o tono
S=HSV(:,:,2); % saturación
V=HSV(:,:,3); % valor de intensidad
% Los valores H se seleccionan según el rango de colores
Mam = ( H>0.15 & H<0.20 & S>0.4 ); % zona color amarillo
subplot(3,3,2), imshow(Mam), xlabel('(a) Zona color amarillo')
Mvr = ((H<0.10 | H>0.972) & S==1); % zona color rojo
subplot(3,3,4), imshow(Mvr), xlabel('(b) Zona color rojo')
Mvg = (H>0.25 & H<0.40 & S==1); % zona color verde
subplot(3,3,5), imshow(Mvg), xlabel('(c) Zona color verde')
Mvb = (H>0.55 & H<0.75 & S==1); % zona color azul
subplot(3,3,6), imshow(Mvb), xlabel('(d) Zona color azul')
MMam = repmat(Mam,[1 1 3]);
B=Art1_1; B(~MMam)=0;
subplot(3,3,3), imshow(B), xlabel('(e) Objeto Amarillo')
MMvr = repmat(Mvr,[1 1 3]); % repetir copia de matriz color rojo;
B=Art1_1; B(~MMvr)=0; % cargar color rojo a zona seleccionada de rojo
subplot(3,3,7), imshow(B); xlabel('(f) Objetos rojos')
MMvg = repmat(Mvg,[1 1 3]); % repetir copia de matriz color verde;
B=Art1_1; B(~MMvg)=0; % cargar color verde a zona seleccionada de verde
subplot(3,3,8), imshow(B), xlabel('(g) Objetos verdes')
MMvb = repmat(Mvb,[1 1 3]); % repetir copia de matriz color verde;
B=Art1_1; B(~MMvb)=0; % cargar color azul a zona seleccionada de azul
subplot(3,3,9), imshow(B), xlabel('(h) Objetos azules')
```

La figura 3.13 muestra la separación de las zonas de colores (a) amarillo, (b) rojo, (c) verde y (d) azul con fondo negro y las imágenes para objetos (e) amarillo, (f) rojos, (g) verdes y (h) azules. A diferencia con las imágenes de la figura 3.12, se observa que los objetos son separados primero seleccionando la zona de color requerida y después

mostrando el color en la zona escogida según el rango del modo de color HSV.

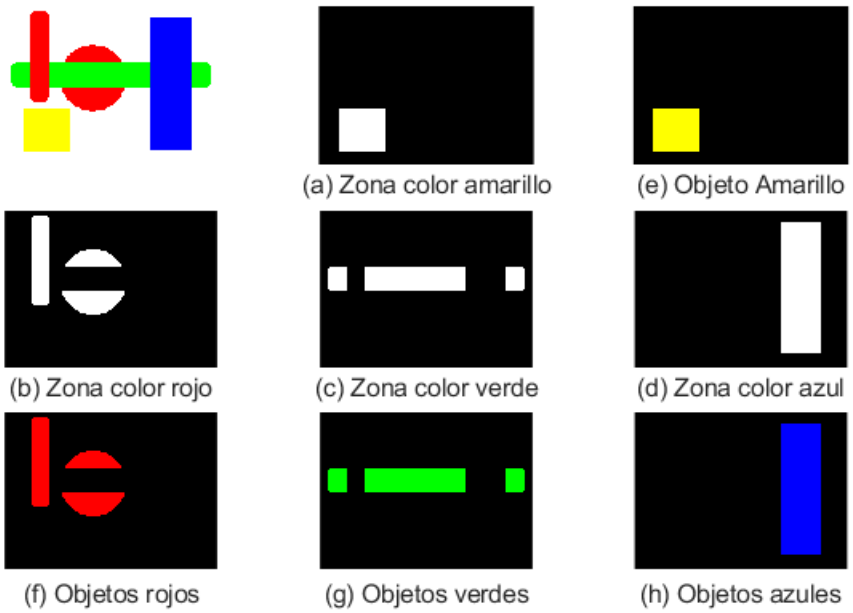


Figura 3. 13: Imagen original para separación del espacio de color.
Elaborado por: Autor.

Ahora, se va a mostrar el script que calcula el histograma de colores para la imagen ´mongolia.jpg' (ver aplicación práctica 2).

```
% Trabajo de titulación - Práctica 3 prueba 3:
M=imread('mongolia.jpg'); % Leer imagen
figure(4)
imshow(M) % mostrar imagen mongolia.jpg
title('Mongolia')
figure(5)
% Llamar función que calcula histograma de colores
% RGB de la imagen mongolia.jpg
[cR,cG,cB,xR,xG,xB]=rgbhist(M)
% Graficar histograma de colores RGB
plot(xR,cR,'r',xG,cG,'g',xB,cB,'b');
title('Histogramas de colores RGB')
grid;
% Graficar histograma de colores RGB y cada
% histograma de colores Rojo, Verde y Azul
figure(6); subplot(2,2,1);
plot(xR,cR,'r',xG,cG,'g',xB,cB,'b');
title('Histogramas de colores RGB'), grid
subplot(2,2,2); plot(xR,cR,'r');
grid; title('Histograma R');
subplot(2,2,3); plot(xG,cG,'g');
grid; title('Histograma G');
subplot(2,2,4); plot(xB,cB,'b');
grid; title('Histograma B')
```

La figura 3.14 muestra el histograma de colores RGB del procesamiento de la imagen ‘mongolia.jpg’. La figura 3.15 muestra cuatro histogramas que son RGB, rojo, verde y azul, respectivamente.

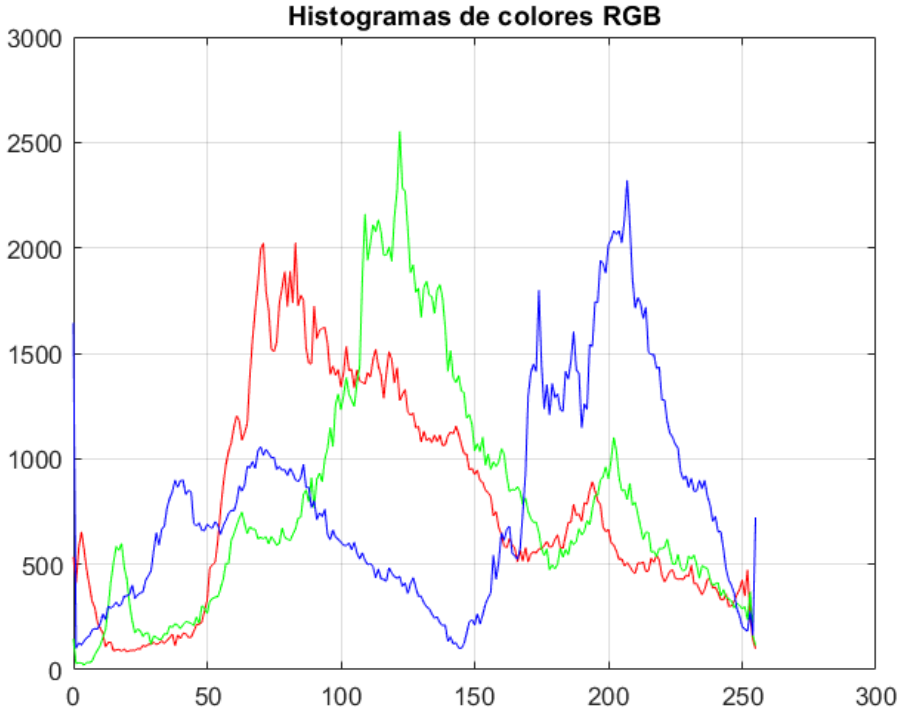


Figura 3. 14: Histograma de colores de la imagen ‘mongologia.jpg’.
Elaborado por: Autor.

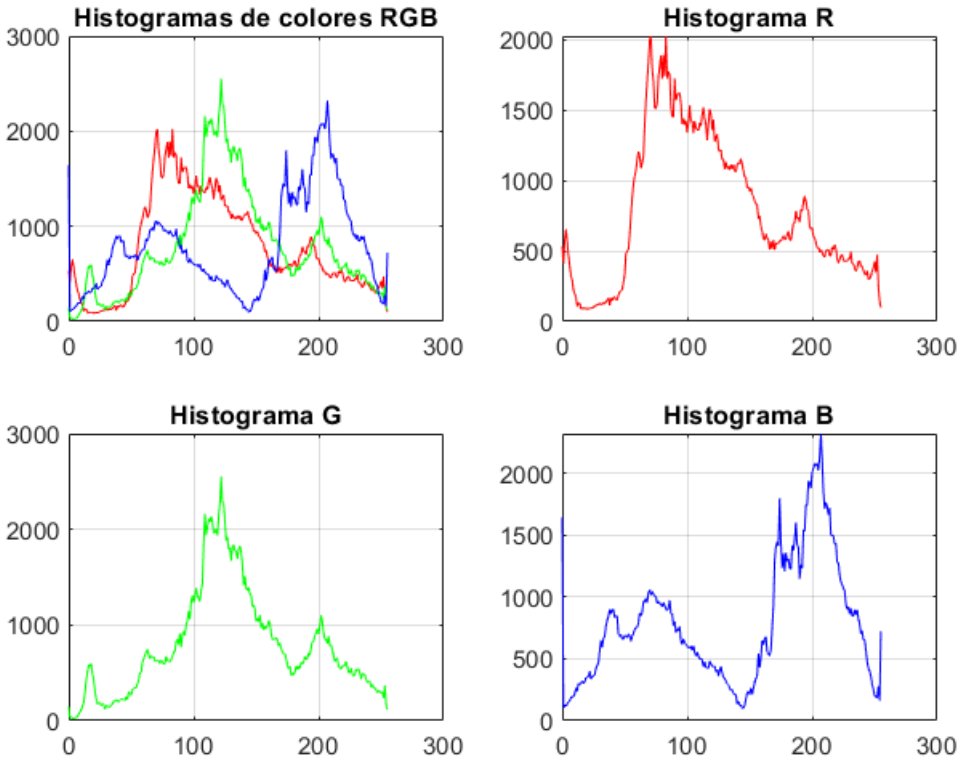


Figura 3. 15: Histogramas de colores RGB y para rojo, verde y azul.
Elaborado por: Autor.

Del histograma de la figura 3.14 se deducen los colores rojos, verdes y azules ya que están en áreas bien definidas de la imagen. Estos corresponden, obviamente, a los picos más altos para las imágenes definidas para los colores que se muestran en la figura 3.16 (b), (c) y (d).

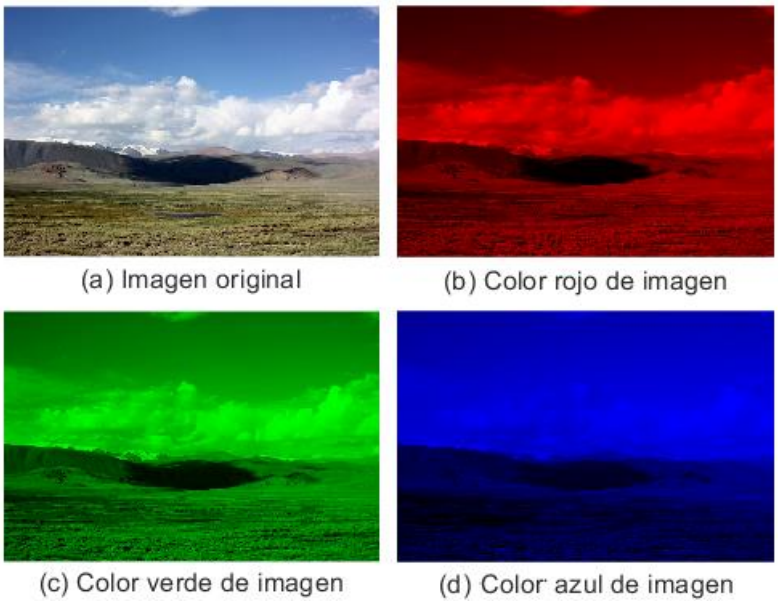


Figura 3. 16: Comparación de reducción de imágenes usando interpolación bilineal.
Elaborado por: Autor.

3.5. Aplicación práctica 4: Detección de bordes en imágenes utilizando la plataforma virtual LabVIEW.

En esta sección se presenta la aplicación práctica de detección de bordes en imágenes utilizando el banco de trabajo de ingeniería de instrumentos virtuales de laboratorio (LabVIEW), que es una plataforma de diseño de sistemas y un entorno de desarrollo para un lenguaje artificial visible de National Instruments. LabVIEW se utiliza generalmente para adquisición de datos, control de instrumentos y automatización industrial. LabVIEW simplifica la integración de hardware para que pueda adquirir y visualizar conjuntos de datos de prácticamente desde cualquier dispositivo de E/S. Combinado con una sintaxis de programación gráfica que reduce el tiempo de programación.

Hay dos formas de estrategias utilizadas para el procesamiento de imágenes en particular, el proceso de imágenes analógicas y digitales. El

proceso de imagen analógica se utiliza a menudo para las copias agotadoras como impresiones e imágenes. Los analistas de imágenes utilizan numerosos fundamentos de interpretación mientras que victimizan estas técnicas visuales. Las técnicas de procesamiento de imágenes digitales facilitan la manipulación de las imágenes digitales por computadoras.

La definición de bordes son cambios nativos críticos en la intensidad entre una imagen. Los bordes tienden a aparecer en el límite entre dos regiones completamente diferentes de una imagen. El objetivo de la detección de bordes, entonces, es proporcionar una delimitación de una escena trazando los límites/bordes de los objetos entre una imagen. Los bordes a menudo son causados por dos situaciones completamente diferentes; eventos geométricos correspondientes a un objeto o límite de superficie, o eventos no geométricos correspondientes a reflejos de luz y sombras entre la imagen

El lenguaje artificial utilizado en LabVIEW, conjuntamente presentado como programación grafica G, puede ser un lenguaje artificial de flujo de datos. La ejecución se establece mediante la estructura de un diagrama gráfico (el código fuente LV) en el que el ingeniero de software conecta nodos de función completamente diferentes dibujando cables. Estos cables propagan variables y cualquier nodo se ejecutará tan pronto como se ofrezca.

Dado que este puede ser el caso de varios nodos al mismo tiempo, la programación G es inherentemente capaz de ejecución en paralelo. El hardware de multiprocesamiento y subprocesos múltiples es explotado mecánicamente por el hardware intrínseco, que multiplexa múltiples subprocesos del sistema operativo en los nodos preparados para la ejecución.

En la figura 3.17 se muestra la programación G para la detección de bordes y la interface gráfica donde se visualizarán las imágenes con bordes detectados en escala de grises mediante filtrado de Sobel, Prewitt, Roberts, Sigma, Gradiente y Diferenciación.

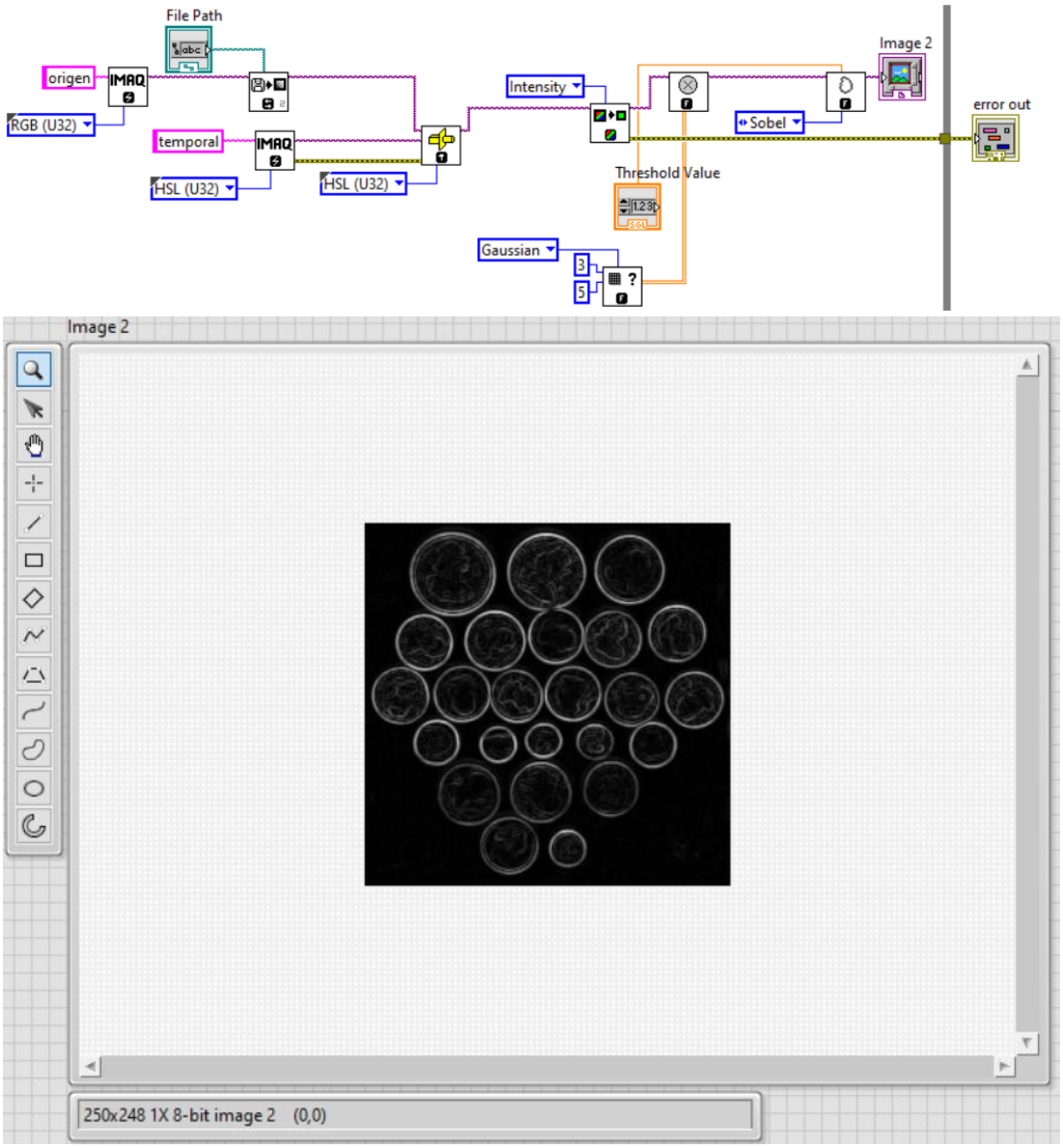


Figura 3. 17: Programación gráfica G e interfaz gráfica para detección de bordes.
Elaborado por: Autor.

La figura 3.18 muestra el instrumento virtual IMAQ EdgeDetection VI que permite extraer los contornos (detecta bordes) en valores de nivel de grises. Cualquier imagen conectada a la entrada 'Image Dst' debe ser del mismo tipo de imagen conectada a 'Image Src'. El tipo de imagen conectada a la entrada de 'Image Mask' debe ser una imagen de 8 bits. La imagen de origen conectada debe haber sido creada con un borde capaz de soportar el tamaño de la matriz de procesamiento. Por ejemplo, una matriz de 3×3 tiene un tamaño de borde mínimo de 1. El tamaño de borde de la imagen de destino no es importante.

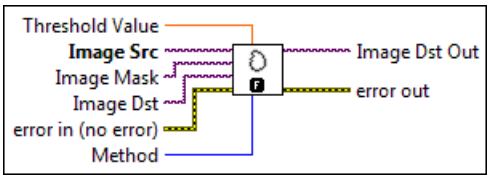


Figura 3. 18: Instrumento virtual para la detección de bordes.
Elaborado por: Autor.

El método (Method) especifica el tipo de filtro de detección de bordes que se utilizará. El valor predeterminado es diferenciación, aunque en la figura 3.17 se muestra el filtrado de Sobel. La tabla 3.5 muestra los métodos de filtros válidos en el software LabVIEW.

Tabla 3. 5: Complejidad del tiempo de interpolación.

Método del filtro	**Procesamiento**
Diferenciación	con una matriz 2×2
Gradiente	con una matriz 2×2
Prewitt	con una matriz de 3×3
Roberts	con una matriz 2×2
Sigma	con una matriz de 3×3
Sobel	con una matriz de 3×3

Elaborado por: Autor.

La figura 3.17 muestra los resultados obtenidos para cada filtrado, (a) sobel, (b) sigma, (c) Roberts, (d) Prewitt, (e) Gradiente y (f) diferenciación. De los resultados obtenidos se observa que el método de Prewitt (ver figura 3.20b) es el filtrado más eficiente en la relación a los otros métodos empleados, aunque el método de Sobel es también eficiente. Finalmente, el método Prewitt es similar a Sobel y se utilizan para detectar bordes verticales y horizontales en imágenes. Sin embargo, a diferencia del filtro de Sobel, este método no pone ningún énfasis en los píxeles que están más cerca del centro de la máscara.

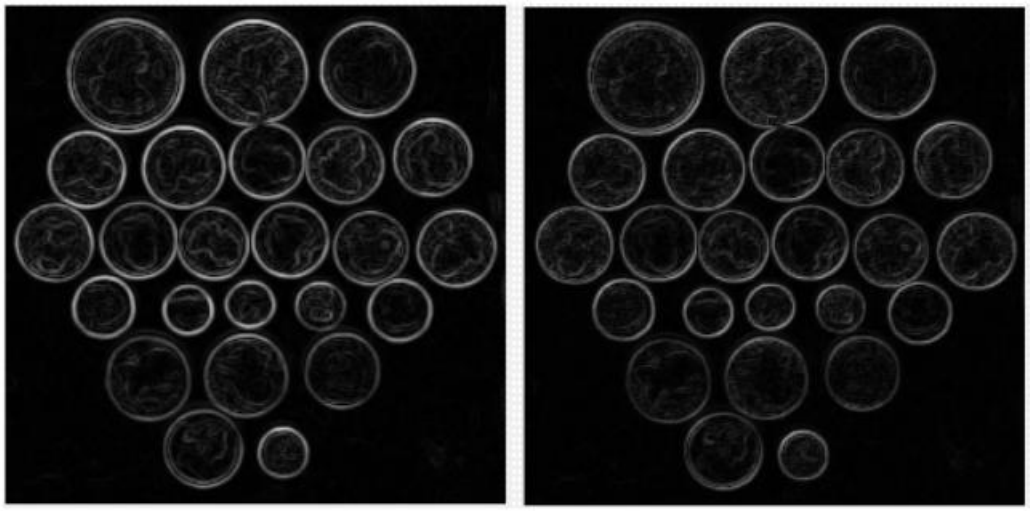


Figura 3. 19: Detección de bordes mediante filtrado (a) Sobel y (b) Sigma.
Elaborado por: Autor.

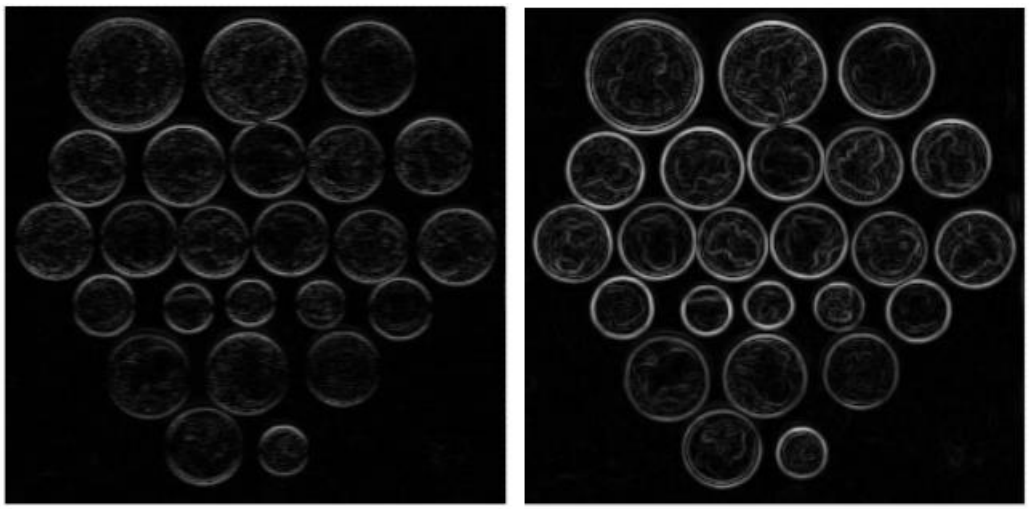


Figura 3. 20: Detección de bordes mediante filtrado (a) Roberts y (b) Prewitt.
Elaborado por: Autor.

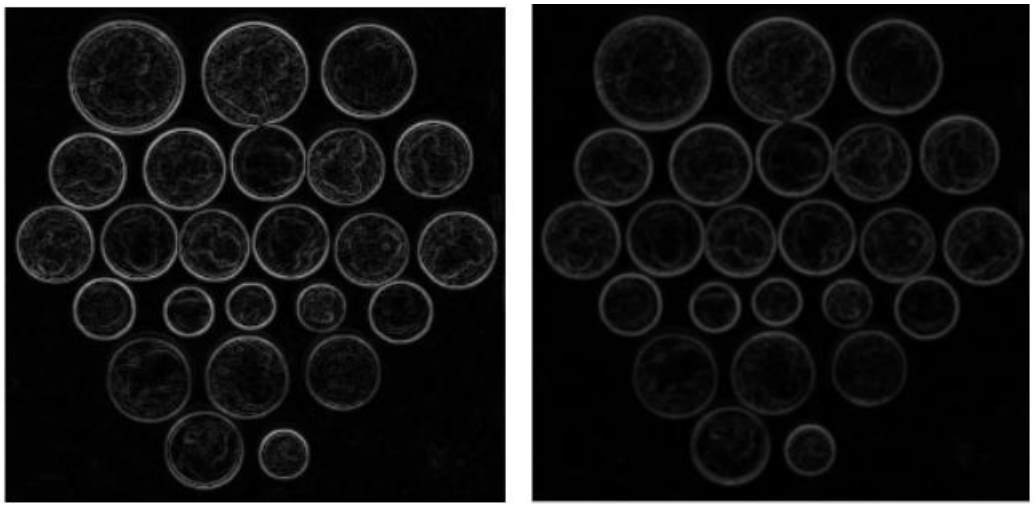


Figura 3. 21: Detección de bordes mediante filtrado (a) Gradiente y (b) Diferenciación.
Elaborado por: Autor.

Conclusiones.

- Después de amplios estudios de áreas relacionadas con el tema propuesto, se diseñó e implementó aplicaciones prácticas para el procesamiento de imágenes en Matlab y LabVIEW. La parte de programación real inicialmente tuve algunos desafíos relacionados principalmente con las funciones que se utilizaron para el desarrollo de las cuatro aplicaciones prácticas. Solo para mencionar un ejemplo: las operaciones responsables de rotar imágenes de forma vertical y horizontalmente funcionan solo para imágenes en escala de grises. Aprender sobre las limitaciones y las posibilidades de errores me ayudó a comprender lo importante que es planificar y pensar en todos los detalles del nuevo software al comienzo del trabajo. En resumen, el resultado del presente trabajo de titulación son aplicaciones prácticas sencillas que se implementan en Matlab y LabVIEW.

- La diferencia entre el método bilineal y bicúbica, es que bilineal utiliza 4 vecinos más cercanos para determinar la salida, mientras que bicúbica utiliza 16 (vecindario 4×4). Y que la distribución del peso se realiza de forma diferente. Entonces, lo único que necesitamos saber es cómo se distribuyen los pesos y el resto es igual que bilineal.

- Se ha observado que el detector de bordes Prewitt produce una mayor precisión en la detección de bordes de objetos con mayor entropía, PSNR, MSE y tiempo de ejecución en comparación con los métodos de Sobel, Roberts, Prewitt, Gradiente y Diferenciación.

Recomendaciones.

- Las aplicaciones prácticas desarrolladas sean utilizadas cómo un tópico en el tratamiento de señales e imágenes ya que existen múltiples investigaciones relacionados al procesamiento de imágenes digitales.

- Implementar una interfaz gráfica de usuario (GUI) para el tratamiento de imágenes mediante transformaciones morfológicas en MatLab/Simulink.

Bibliografía

Abdullah, M. Z. (2016). Image Acquisition Systems. En *Computer Vision Technology for Food Quality Evaluation* (pp. 3–43). Elsevier. https://doi.org/10.1016/B978-0-12-802232-0.00001-3

Amudha, J., Pradeepa, N., & Sudhakar, R. (2012). A Survey on Digital Image Restoration. *Procedia Engineering*, *38*, 2378–2382. https://doi.org/10.1016/j.proeng.2012.06.284

Arce, G. R., Bacca, J., & Paredes, J. L. (2009). Nonlinear Filtering for Image Analysis and Enhancement. En *The Essential Guide to Image Processing* (pp. 263–291). Elsevier. https://doi.org/10.1016/B978-0-12-374457-9.00012-3

Bustacara-Medina, C., Florez-Valencia, L., & Carlos Diaz, L. (2020). Improved Canny Edge Detector Using Principal Curvatures. *Journal of Electrical and Electronic Engineering*, *8*(4), 109. https://doi.org/10.11648/j.jeee.20200804.11

Gonzalez, R. C., & Woods, R. E. (2018). *Digital image processing*. Pearson.

Lu, Z.-M., & Guo, S.-Z. (2017). Introduction. En *Lossless Information Hiding in Images* (pp. 1–68). Elsevier. https://doi.org/10.1016/B978-0-12-812006-4.00001-2

Escriba aquí la ecuación.

Ma, L., Zhao, D., & Gao, W. (2012). Learning-based image restoration for compressed images. *Signal Processing: Image Communication*, *27*(1), 54–65. https://doi.org/10.1016/j.image.2011.05.004

Mahajan, S., Das, A., & Sardana, H. K. (2015). Image acquisition techniques for assessment of legume quality. *Trends in Food Science & Technology*, *42*(2), 116–133. https://doi.org/10.1016/j.tifs.2015.01.001

MathWorks. (2020). *Erode image—MATLAB imerode*. https://www.mathworks.com/help/images/ref/imerode.html

Rajendra, P., Sudheer, K., & Boadh, R. (2017). Design of a Recognition System Automatic Vehicle License Plate through a Convolution Neural Network. *International Journal of Computer Applications*, *177*(3), 47–54. https://doi.org/10.5120/ijca2017915703

Ramadan, H., Lachqar, C., & Tairi, H. (2020). A survey of recent interactive image segmentation methods. *Computational Visual Media*, *6*(4), 355–384. https://doi.org/10.1007/s41095-020-0177-5

Ribeiro, S. (2014). USING SIMPLECV FOR SEED METADATA EXTRACTION INTO XML DOCUMENT. *Iberoamerican Journal of Applied Computing*, *4*, 29.

Sonka, M., Hlavac, V., & Boyle, R. (2015). *Image processing, analysis, and machine vision* (Fourth edition). Cengage Learning.

Tan, D. (2016). Image Enhancement Based on Adaptive Median Filter and Wallis Filter. *Proceedings of the 2015 4th National Conference on Electrical, Electronics and Computer Engineering*. 2015 4th National Conference on Electrical, Electronics and Computer Engineering, Xi'an, China. https://doi.org/10.2991/nceece-15.2016.142

Thanki, R. M., & Kothari, A. M. (2019). Morphological Image Processing. En R. M. Thanki & A. M. Kothari, *Digital Image Processing using SCILAB* (pp. 99–113). Springer International Publishing. https://doi.org/10.1007/978-3-319-89533-8_5

Vikram Mutneja, D. (2015). Methods of Image Edge Detection: A Review. *Journal of Electrical & Electronic Systems*, *04*(02). https://doi.org/10.4172/2332-0796.1000150

Xin, M., & Wang, Y. (2019). Research on image classification model based on deep convolution neural network. *EURASIP Journal on Image and Video Processing*, *2019*(1), 40. https://doi.org/10.1186/s13640-019-0417-8

Zaitoun, N. M., & Aqel, M. J. (2015). Survey on Image Segmentation Techniques. *Procedia Computer Science*, *65*, 797–806. https://doi.org/10.1016/j.procs.2015.09.027

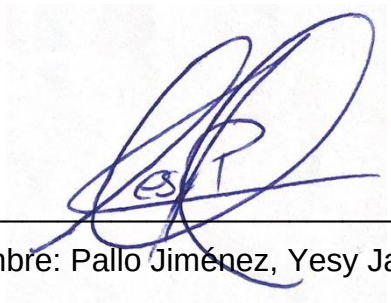
DECLARACIÓN Y AUTORIZACIÓN

Yo, **Pallo Jiménez, Yesy Javier** con C.C: # 200014794-8 autor del Trabajo de Titulación: **Desarrollo de aplicaciones prácticas en la adquisición y procesamiento de imágenes en MatLab y LabVIEW** 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: Pallo Jiménez, Yesy Javier

C.C: 200014794-8

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA			
FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN			
TÍTULO Y SUBTÍTULO:	Desarrollo de aplicaciones prácticas en la adquisición y procesamiento de imágenes en MatLab y LabVIEW		
AUTOR(ES)	Pallo Jiménez, Yesy Javier		
REVISOR(ES)/TUTOR(ES)	M. Sc. Zamora Cedeño, Néstor Armando		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad de Educación Técnica para el Desarrollo		
CARRERA:	De Telecomunicaciones		
TITULO OBTENIDO:	Ingeniero en Telecomunicaciones		
FECHA DE PUBLICACIÓN:	10 de marzo del 2021	**No. DE PÁGINAS:**	53
ÁREAS TEMÁTICAS:	Sistemas de Transmisión, Comunicaciones Ópticas		
PALABRAS CLAVES/ KEYWORDS:	Procesamiento, Imágenes, Detección, Bordes, Resolución, Cuantificación.		

RESUMEN/ABSTRACT (150-250 palabras):

Cada día el mundo está evolucionando muy rápido. El rápido desarrollo de la tecnología informática ha afectado a todas las áreas científicas. Medicina, automatización, análisis de datos, finanzas, biología, química, economía y muchos, muchos más se han beneficiado de la expansión de la tecnología. El mundo de los gráficos por ordenador evoluciona rápidamente todos los días. El mercado está lleno de aplicaciones de procesamiento de imágenes. Pero ¿cuántos de ellos realmente contribuyen al crecimiento de la ciencia y la tecnología? ¿Existe algún programa que combine funcionalidades de procesamiento de imágenes con técnicas de programación? La respuesta a estas preguntas es Matlab y LabVIEW. Matlab es un software que proporciona un lenguaje de programación de alto nivel, muchas bibliotecas temáticas y mecanismos de interfaz gráfica de usuario fáciles de implementar. El procesamiento de imágenes es una parte importante de la instrumentación virtual. El software LabVIEW se puede utilizar para el procesamiento de imágenes para obtener mejores resultados. De acuerdo con estos antecedentes, se desarrolla el trabajo de titulación que se basa en la implementación de aplicaciones prácticas en la adquisición y procesamiento de imágenes en las plataformas de simulación MatLab y LabVIEW.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** +593 969858478	E-mail: yesypallo@hotmail.com
CONTACTO CON LA INSTITUCIÓN: COORDINADOR DEL PROCESO DE UTE	**Nombre:** Palacios Meléndez, Edwin Fernando **Teléfono:** +593-9-67608298 **E-mail:** edwin.palacios@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA	
Nº. DE REGISTRO (en base a datos):	
Nº. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	