

2023  
IPN - ESCOM

# Análisis de imágenes

## Morfología matemática

### Comunidad 1 - Color Naranja

- Hernández Domínguez Angel Alan
- Morán Pablo Aliss
- Rivera Sánchez Perla Amarilis
- Salazar Gómez Andrés
- Villegas Dorantes Edwin Ivan

**Grupo: 3CM14**

**Maestra: Cruz Meza María Elena**





Resumen - En este proyecto de procesamiento de imágenes, se utiliza la morfología matemática como una técnica fundamental para analizar y tratar estructuras geométricas presentes en las imágenes. La morfología matemática se basa en la teoría de conjuntos, retículos, topología y funciones aleatorias, y ofrece un conjunto de operadores y técnicas para realizar transformaciones en las imágenes. Estas transformaciones incluyen operaciones como dilatación, erosión, apertura, cierre, gradiente morfológico, entre otras. En el presente proyecto se ha desarrollado un prototipo en Python que implementa las operaciones morfológicas y muestra la imagen procesada. Este proyecto demuestra la versatilidad y utilidad de la morfología matemática en el análisis de imágenes, especialmente cuando se requiere resaltar y segmentar objetos de interés en una imagen.



## ÍNDICE DE CONTENIDO

<b>1. MARCO TEÓRICO .....</b>	<b>4</b>
<b>1.1. Procesamiento de imágenes .....</b>	<b>4</b>
<b>1.2. Morfología matemática.....</b>	<b>5</b>
<b>1.2.1. Historia de la morfología matemática .....</b>	<b>5</b>
<b>1.2.2. Operaciones básicas de la morfología matemática.....</b>	<b>7</b>
<b>2. DEFINICIÓN DEL PROBLEMA .....</b>	<b>8</b>
<b>3. OBJETIVOS.....</b>	<b>8</b>
<b>3.1. Objetivo General.....</b>	<b>8</b>
<b>3.2. Objetivos Específicos .....</b>	<b>8</b>
<b>4. PROTOTIPO .....</b>	<b>9</b>
<b>5. PRUEBAS .....</b>	<b>15</b>
<b>6. CONSLUSIONES .....</b>	<b>19</b>
<b>7. REFERENCIAS .....</b>	<b>20</b>



## ÍNDICE DE FIGURAS

Figura 1 Efectos de las operaciones más comunes de morfología matemática .....	6
Figura 2. Bibliotecas usadas para el proyecto.....	9
Figura 3. Código de la función de redimensión de imágenes.....	10
Figura 4. Código de la función erosión. ....	10
Figura 5. Código de la función dilatación. ....	11
Figura 6. Código de la función apertura. ....	11
Figura 7. Código de la función cerradura.....	12
Figura 8. Código de la función gradiente. ....	12
Figura 9. Código de la función start. ....	13
Figura 10. Código de la función choose.....	13
Figura 11. Código de la interfaz gráfica. ....	14
Figura 12. Imagen a la que le aplicaremos la transformación.....	15
Figura 13. Imagen en grises. ....	15
Figura 14. Imagen Binarizada. ....	16
Figura 15. Imagen con un filtro promedio aplicado utilizando un kernel de 3x3. ....	16
Figura 16. Imagen binarizada a la que se le aplicó el filtro de cerradura.....	17
Figura 17. Imagen binarizada a la que se le aplicó el filtro blackhat. ....	17
Figura 18. Uso de contornos para la imagen procesada con los métodos correspondientes.....	18
Figura 19 Imágenes seleccionadas para hacer la operación OR. ....	18
Figura 20 Resultados obtenidos .....	18



# 1. MARCO TEÓRICO

## 1.1. Procesamiento de imágenes

El procesamiento de imágenes es un campo de estudio y aplicación que se ocupa de manipular imágenes digitales con el objetivo de mejorar su calidad, extraer información relevante o realizar tareas específicas relacionadas con el análisis y reconocimiento de patrones visuales. Consiste en aplicar una serie de técnicas y algoritmos computacionales para modificar, analizar o interpretar imágenes capturadas por dispositivos de visión artificial, como cámaras o escáneres.

El procesamiento digital de imágenes abarca un amplio espectro de técnicas y algoritmos que se aplican a imágenes digitales. Estas técnicas van desde procesos que toman imágenes como entrada y generan imágenes como salida, hasta aquellos que extraen atributos o características cuantificables de una imagen, e incluso procedimientos que permiten reconocer objetos individuales o en grupo.

Históricamente, el procesamiento de imágenes ha experimentado un notable avance y evolución a lo largo del tiempo. Sus raíces se remontan al siglo XIX, cuando se comenzaron a utilizar técnicas fotográficas para capturar imágenes. Sin embargo, fue a partir de la década de 1950 con el desarrollo de los primeros computadores digitales cuando se sentaron las bases para el procesamiento de imágenes tal como lo conocemos hoy en día.

En la década de 1960, con el surgimiento de la era digital, se comenzaron a utilizar técnicas más avanzadas en el procesamiento de imágenes. La computación gráfica y la visualización de imágenes se convirtieron en áreas de investigación y desarrollo activas. En este período se introdujeron conceptos como la transformada de Fourier para el análisis de frecuencias, los filtros lineales para la eliminación de ruido y los métodos estadísticos para el análisis de imágenes.

A medida que la tecnología digital avanzaba, se desarrollaron algoritmos más sofisticados y poderosos para el procesamiento de imágenes. En las décadas siguientes, el procesamiento de imágenes se benefició del crecimiento exponencial de la capacidad computacional y el desarrollo de algoritmos más complejos. Se introdujeron técnicas de visión por computadora, reconocimiento de patrones y aprendizaje automático, lo que permitió la detección y clasificación automática de objetos en imágenes.

Actualmente, el procesamiento de imágenes abarca una amplia gama de aplicaciones, desde el diagnóstico médico asistido por computadora y el reconocimiento facial, hasta la realidad aumentada y la robótica. Se continúa investigando y desarrollando nuevas técnicas para mejorar la precisión y eficiencia en el procesamiento de imágenes, con el objetivo de aprovechar todo su potencial en diferentes campos de aplicación.

En este contexto, tres aspectos principales del procesamiento de imágenes merecen especial atención. En primer lugar, el mejoramiento de imágenes se refiere a la aplicación de técnicas para eliminar ruido, realzar detalles y mejorar la calidad visual de las imágenes. La restauración de imágenes, por otro lado, busca recuperar imágenes dañadas o degradadas debido a condiciones adversas como el ruido, el desenfoque o la compresión. Por último, la identificación de objetos implica la detección y clasificación de objetos de interés en las imágenes.



En el campo del procesamiento de imágenes, se utilizan diversas herramientas y técnicas. Entre ellas, se destacan el análisis de Fourier, que permite descomponer una imagen en frecuencias y trabajar en el dominio de la frecuencia para aplicar filtros y realizar operaciones específicas. Los filtros lineales también son ampliamente empleados para eliminar ruido o realzar características de interés. Además, los métodos estadísticos brindan herramientas para el análisis y la extracción de información basada en la distribución de los valores de los píxeles. Por último, los patrones sintácticos se utilizan para el reconocimiento de objetos, empleando técnicas de aprendizaje automático y clasificación.

## 1.2. Morfología matemática

La morfología matemática es una teoría y metodología utilizada en el procesamiento de imágenes para analizar y manipular las formas y estructuras presentes en las imágenes.

### 1.2.1. Historia de la morfología matemática

El procesamiento de imágenes es un campo multidisciplinario que abarca una amplia gama de metodologías destinadas a abordar diversos enfoques. Entre estas metodologías se destaca la morfología matemática, la cual desempeña un papel fundamental en el procesamiento de imágenes debido a su carácter algebraico y su implementación basada en la teoría de conjuntos y retículos. La importancia de las propiedades algebraicas radica en su capacidad para evaluar la validez de las transformaciones implicadas en esta teoría.

La morfología matemática tuvo sus inicios en 1964, como resultado de la colaboración entre Georges Matheron y Jean Serra en la École des Mines de Paris, Francia. El trabajo de Serra, desarrollado como parte de su tesis doctoral, se centró en la cuantificación de las características minerales en una sección delgada de la mina de la Mourière, lo cual condujo a nuevos avances teóricos y enfoques prácticos en geometría integral y topología.

Durante la década de 1960 y la década de 1970, la morfología matemática se centró principalmente en imágenes binarias tratadas como conjuntos, generando una amplia variedad de operadores binarios y técnicas como la transformación de localización, dilatación, erosión, apertura, cierre, granulometría, adelgazamiento, cálculo del esqueleto, erosión final, bisectriz condicional, entre otros.

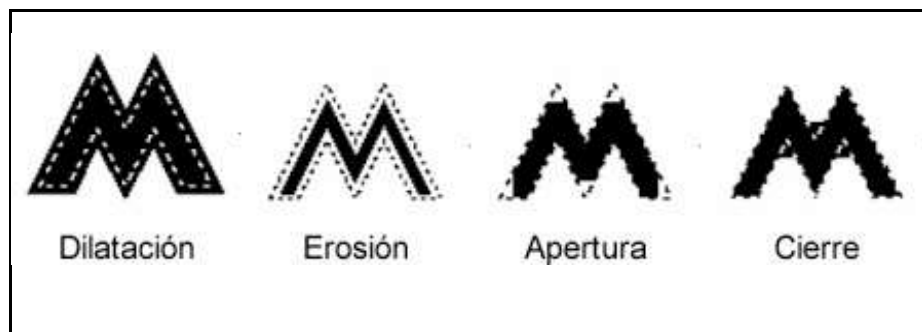
A partir de mediados de la década de 1970 hasta mediados de la década de 1980, la morfología matemática se extendió también a funciones e imágenes en escala de grises. Esta generalización no solo amplió los conceptos principales (como dilatación y erosión) a funciones, sino que también introdujo nuevos operadores, como el gradiente morfológico, la transformación sombrero de copa y la divisoria (watershed), que se convirtieron en enfoques fundamentales para la segmentación en la morfología matemática.

La morfología matemática ha evolucionado desde sus inicios en la década de 1960, pasando de imágenes binarias a imágenes en escala de grises, y ha desarrollado una amplia gama de operadores y técnicas para el análisis y procesamiento de formas e estructuras en imágenes. Su enfoque basado

en la teoría de conjuntos y retículos, así como en propiedades algebraicas, la convierte en una herramienta poderosa y versátil en el campo del procesamiento de imágenes.

La morfología matemática es comúnmente aplicada más a las imágenes digitales, pero puede ser empleada también en gráficos, mallas poligonales, sólidos y muchas otras estructuras espaciales.

Conceptos topológicos y geométricos de espacio continuo, tales como tamaño, forma, convexidad, conectividad y distancia geodésica, se pueden caracterizar por la morfología matemática en espacios continuos y discretos. La morfología matemática es también la base del procesamiento de imágenes morfológicas, que consiste en un conjunto de operadores que transforman las imágenes de acuerdo a las caracterizaciones anteriores.



*Figura 1 Efectos de las operaciones más comunes de morfología matemática*

La idea intuitiva que tenemos sobre la estructura de los objetos no es suficiente, ya que no siempre puede ser precisa. Además, resulta prácticamente imposible proporcionar una descripción objetiva y completa de cualquier objeto. Al observar un objeto, cada persona enfatizará ciertas características que le resulten interesantes, transformándolo así en otra representación resaltando ciertos detalles, el observador puede mencionar que la parte del objeto que le interesa tiene picos, es cuadrada o redonda, pequeña o grande, estos atributos son el resultado de comparar tal vez una parte específica del objeto con una estrella, un cuadrado, un círculo u otros objetos similares en forma y tamaño, a estos últimos objetos se les puede identificar como elementos estructurales y su relación con un objeto se puede ver como una transformación morfológica.

Por tanto, la morfología matemática es una teoría que considera los aspectos geométricos locales que resulten de interés al analizar uno o más objetos. Actualmente, se considera que la morfología matemática es una herramienta versátil en el análisis de imágenes, especialmente en aplicaciones donde los aspectos geométricos son relevantes.

Durante el desarrollo de la aplicación se presentará de manera detallada y visual el funcionamiento de algunas propiedades elementales relativas a las operaciones fundamentales como erosión, dilatación, apertura y cerradura morfológica, se mostrarán ejemplos con diferentes imágenes las cuales serán procesadas, se aplicarán diversas combinaciones específicas a las imágenes. Estas aplicaciones además incluyen la detección de bordes, la eliminación del ruido aleatorio, así como la segmentación de objetos para el reconocimiento de formas.



## 1.2.2. Operaciones básicas de la morfología matemática

Dentro del marco de la morfología matemática, se han desarrollado una serie de operaciones básicas que permiten modificar la forma y estructura de una imagen. Estas operaciones son fundamentales para realizar tareas de procesamiento y análisis de imágenes.

A continuación, se describen las operaciones básicas más comunes en la morfología matemática:

- **Dilatación**

La dilatación es una operación que expande los objetos en una imagen. Consiste en desplazar un elemento estructurante sobre la imagen y asignar el valor máximo encontrado en la vecindad de cada píxel al píxel central del elemento estructurante. Esto produce un efecto de expansión de los objetos presentes en la imagen, aumentando su tamaño. La dilatación es útil para rellenar huecos, unir objetos cercanos y resaltar características de interés.

- **Erosión**

La erosión es la operación opuesta a la dilatación. Consiste en desplazar un elemento estructurante sobre la imagen y asignar el valor mínimo encontrado en la vecindad de cada píxel al píxel central del elemento estructurante. Esto tiene el efecto de reducir el tamaño de los objetos presentes en la imagen, eliminando detalles finos y separando objetos conectados. La erosión es útil para eliminar ruido, eliminar regiones pequeñas o delgadas y separar objetos adyacentes.

- **Apertura y cierre**

La apertura y el cierre son combinaciones de las operaciones de dilatación y erosión. La apertura consiste en aplicar una erosión seguida de una dilatación, mientras que el cierre se compone de una dilatación seguida de una erosión. La apertura suaviza los contornos de los objetos, elimina pequeños detalles y divide objetos conectados. El cierre, por otro lado, rellena huecos, une objetos separados y suaviza bordes irregulares.

- **Gradiente morfológico**

El gradiente morfológico es una operación que permite resaltar los bordes o contornos de los objetos en una imagen. Se obtiene calculando la diferencia entre la dilatación y la erosión de la imagen original. El resultado es una imagen que resalta las transiciones abruptas de intensidad, lo que facilita la detección de bordes.

- **Top-hat y Black-hat**

El top-hat y el black-hat son operaciones que permiten resaltar estructuras de tamaño y forma específicas en una imagen. El top-hat se obtiene restando la imagen original a su apertura, lo que resalta estructuras más pequeñas que el elemento estructurante utilizado en la apertura. El black-hat, por otro lado, se obtiene restando el cierre de la imagen original, lo que resalta estructuras más grandes que el elemento estructurante utilizado en el cierre. Estas operaciones son útiles para detectar y analizar estructuras de interés en una imagen.





## 2. DEFINICIÓN DEL PROBLEMA

La problemática abordada en esta tesis se centra en la identificación de letras en una imagen específica de una camioneta estacionada sobre tierra y vegetación circundante. Se busca lograr la segmentación precisa de las letras presentes en la placa de la camioneta. Esta tarea plantea desafíos debido a la presencia de elementos distractorios y posibles variaciones en la apariencia de las letras.

La segmentación de las letras en la placa de la camioneta es de gran importancia en aplicaciones como el reconocimiento automático de matrículas y el control de acceso vehicular. Sin embargo, esta problemática requiere el desarrollo de técnicas y metodologías específicas para superar los obstáculos mencionados y lograr una segmentación precisa y confiable.

En este contexto, se explorarán y aplicarán técnicas de morfología matemática, seleccionando cuidadosamente los elementos de estructura adecuados y se propone el desarrollo de una aplicación que permita procesar las imágenes de placas de vehículos con el fin de identificar las letras presentes en ellas, para lograrlo, se utilizarán técnicas de procesamiento de imágenes y morfología matemática.

Al abordar esta problemática, se espera contribuir al desarrollo de sistemas de reconocimiento de matrículas más eficientes y confiables, con potenciales aplicaciones en áreas como la seguridad vial, la gestión del tráfico y la seguridad en general.

## 3. OBJETIVOS

### 3.1. Objetivo General

Desarrollar una aplicación en Python para procesar y segmentar de manera precisa las placas de vehículos en una imagen, utilizando propiedades morfológicas para obtener una mejor visión de dicha imagen.

### 3.2. Objetivos Específicos

- Identificar las características de las letras en la placa para diseñar algoritmos de procesamiento de imágenes.
- Evaluar y seleccionar las propiedades más adecuadas de la morfología matemática para el procesamiento de la imagen.
- Diseñar algoritmos de segmentación basados en las características identificadas en las letras de la placa.
- Aplicar las operaciones morfológicas seleccionadas para mejorar la segmentación de la imagen.
- Presentar los resultados obtenidos al aplicar los algoritmos desarrollados.
- Integrar todos los algoritmos en una aplicación funcional para facilitar su uso como herramienta de procesamiento de imágenes.

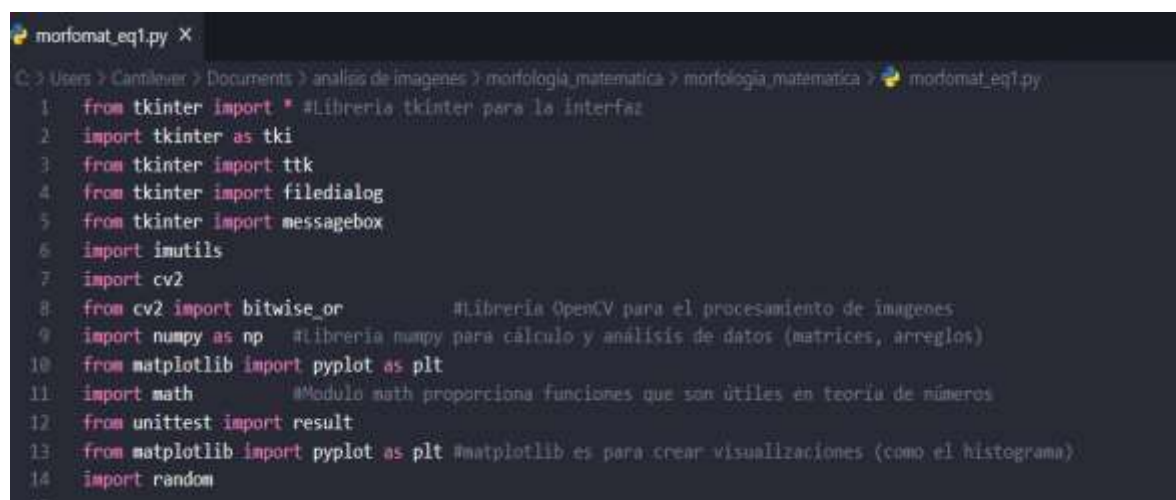
## 4. PROTOTIPO

El desarrollo del prototipo se llevó a cabo utilizando el poderoso lenguaje de programación Python, lo cual resultó ser una elección acertada. Python, con su amplia gama de bibliotecas especializadas, nos brindó una sólida base para implementar todas las funcionalidades requeridas en nuestro proyecto. El uso de bibliotecas como tkinter, opencv y numpy nos permitió aprovechar las capacidades gráficas, de procesamiento de imágenes y de análisis de datos de Python de una manera eficiente y efectiva. Esta elección estratégica nos proporcionó una ventaja significativa al desarrollar el prototipo, permitiéndonos implementar de manera fluida y robusta las operaciones y propiedades necesarias para lograr un procesamiento de imágenes preciso y de alto rendimiento. Gracias a Python, pudimos aprovechar las fortalezas del lenguaje y las bibliotecas disponibles, lo que contribuyó en gran medida al éxito de nuestro proyecto.

En el prototipo se programaron las operaciones y propiedades asignadas a la comunidad, estas operaciones son: erosión, dilatación, apertura, cierre y gradiente morfológico; el resultado de aplicar estas operaciones sobre las imágenes seleccionadas y mostrar la imagen ya procesada, en primer lugar, se identificarán las características específicas de las letras en la placa y se diseñarán los algoritmos de procesamiento necesarios, posteriormente se aplicarán las propiedades morfológicas seleccionadas para mejorar la segmentación de la imagen y los resultados obtenidos serán presentados, mostrando los algoritmos testeados y evaluando cuál ha sido el más efectivo para resolver este problema. Finalmente, todos los algoritmos se integrarán en una aplicación única y funcional, que facilitará su uso como herramienta de procesamiento de imágenes para resolver la problemática planteada.

La manera en la que funciona el prototipo es la siguiente:

La siguiente imagen muestra todas las bibliotecas que ocupamos para el desarrollo del proyecto.



```
morformat_eq1.py X
C:\Users\> Cantilever\Documents> analisis de imagenes> morfologia_matematica> morfologia_matematica> morformat_eq1.py
1  from tkinter import * #Libreria tkinter para la interfaz
2  import tkinter as tki
3  from tkinter import ttk
4  from tkinter import filedialog
5  from tkinter import messagebox
6  import imutils
7  import cv2
8  from cv2 import bitwise_or #Libreria OpenCV para el procesamiento de imagenes
9  import numpy as np #Libreria numpy para cálculo y análisis de datos (matrices, arreglos)
10 from matplotlib import pyplot as plt
11 import math #Modulo math proporciona funciones que son útiles en teoría de números
12 from unittest import result
13 from matplotlib import pyplot as plt #matplotlib es para crear visualizaciones (como el histograma)
14 import random
```

Figura 2. Bibliotecas usadas para el proyecto.

Como vemos en las imágenes, usamos las bibliotecas tkinter para desarrollar la interfaz gráfica, opencv para el procesamiento de imágenes y numpy para el cálculo y análisis de datos, siendo estas las más importantes.

En la siguiente imagen se muestra la función `ResizeWithAspectRatio`:

```
16 def ResizeWithAspectRatio(image, width=None, height=None, inter=cv2.INTER_AREA):
17     dim = None
18     (h, w) = image.shape[:2]
19
20     if width is None and height is None:
21         return image
22     if width is None:
23         r = height / float(h)
24         dim = (int(w * r), height)
25     else:
26         r = width / float(w)
27         dim = (width, int(h * r))
28
29     return cv2.resize(image, dim, interpolation=inter)
30
```

Figura 3. Código de la función de redimensión de imágenes.

Esta función se encarga de redimensionar la imagen que recibe como entrada al tamaño especificado.

En la siguiente imagen se muestra la función erosión:

```
31 def erosion(img):
32     imagen = img
33     # Agregue este para reajustar el tamaño
34     #resized = ResizeWithAspectRatio(imagen, width=300)
35     # Erosión
36     kernel = np.ones((5,5),np.uint8)
37     erosion = cv2.erode(imagen,kernel,iterations = 1)
38
39     cv2.imshow('Original',imagen)
40     cv2.imshow('Erosion',erosion)
41
42     filenameErosion = file+"_erosion.jpg"
43     cv2.imwrite(filenameErosion, erosion)
44
45     cv2.waitKey(0)
46     cv2.destroyAllWindows()
47     result.config(text="Erosión Realizada")
48
```

Figura 4. Código de la función erosión.

En esta función aplicamos la operación de erosión con una estructura que es un cuadrado.

En la siguiente función se muestra la función dilatación:

```
49 def dilatacion(img):
50     imagen = img
51     kernel = np.ones((5,5),np.uint8)
52     dilatacion = cv2.dilate(imagen,kernel,iterations = 1)
53
54     cv2.imshow("Original", imagen)
55     cv2.imshow("Dilatacion", dilatacion)
56
57     filenameDilatada = file+"_dilatacion.png"
58     cv2.imwrite(filenameDilatada, dilatacion)
59
60     cv2.waitKey(0)
61     cv2.destroyAllWindows()
62     result.config(text="Dilatación Realizada")
63
```

Figura 5. Código de la función dilatación.

En esta función aplicamos la operación de dilatación con una estructura que es un cuadrado.

En la siguiente imagen se muestra la función apertura:

```
64 def apertura(img):
65     imagen = img
66
67     #Filtro Apertura
68     opening = cv2.morphologyEx(imagen, cv2.MORPH_OPEN, np.ones((15,15),np.uint8))
69
70     #Filtro Top Hat
71     tophat = cv2.morphologyEx(imagen, cv2.MORPH_TOPHAT, np.ones((15,15),np.uint8))
72
73     cv2.imshow('Original',imagen)
74     cv2.imshow('Apertura',opening)
75     cv2.imshow('Top Hat',tophat)
76
77     filenameApertura = file+"_apertura.png"
78     cv2.imwrite(filenameApertura, opening)
79
80     filenameTH = file+"_tophat.png"
81     cv2.imwrite(filenameTH, tophat)
82
83     cv2.waitKey(0)
84     cv2.destroyAllWindows()
85     result.config(text="Apertura Realizada")
86
```

Figura 6. Código de la función apertura.

En esta función aplicamos la operación de apertura con una estructura que es un cuadrado, además, usamos la operación top hat.

En la siguiente imagen se muestra la función apertura:

```
88 def cerradura(img):
89     imagen = img
90
91     closing = cv2.morphologyEx(imagen, cv2.MORPH_CLOSE, np.ones((15,15),np.uint8))
92
93     #filtro Black Hat
94     blackhat = cv2.morphologyEx(imagen, cv2.MORPH_BLACKHAT, np.ones((15,15),np.uint8))
95
96     cv2.imshow('Original',imagen)
97     cv2.imshow('Cerradura', closing)
98     cv2.imshow('Black Hat', blackhat)
99
100     filenameCerradura = file+"_cerradura.png"
101     cv2.imwrite(filenameCerradura, closing)
102
103     filenameBH = file+"_blackhat.png"
104     cv2.imwrite(filenameBH, blackhat)
105
106     cv2.waitKey(0)
107     cv2.destroyAllWindows()
108     #Permite a los usuarios destruir o cerrar todas las ventanas en cualquier momento después de salir del script.
109     result.config(text="Cerradura Realizada")
110
```

Figura 7. Código de la función cerradura.

En esta función aplicamos la operación de cerradura con una estructura que es un cuadrado, además, usamos la operación black hat.

En la siguiente imagen se muestra la función gradiente:

```
111 #funcion para la ecualización hiperbolica
112 def gradiente(img):
113     imagen = img
114     kernel = np.ones((5,5),np.uint8)
115
116     gradiente = cv2.morphologyEx(imagen, cv2.MORPH_GRADIENT, kernel)
117
118     cv2.imshow("Imagen Original",imagen)
119     cv2.imshow("Gradiente Morfológico",gradiente)
120
121     filenameGradiente = file+"_gradiente.png"
122     cv2.imwrite(filenameGradiente, gradiente)
123
124     cv2.waitKey(0) #Mostrará la ventana infinitamente hasta que se presione cualquier tecla.
125     cv2.destroyAllWindows()
126     result.config(text="Gradiente Morfológico Realizado")
127
```

Figura 8. Código de la función gradiente.

En esta función aplicamos la operación de gradiente con una estructura que es un cuadrado.

En la siguiente imagen se muestra la función start:

```
127
128 def start(): #Menu de opciones
129     if (file == "No se ha seleccionado el archivo") or (file2 == "No se ha seleccionado el archivo"):
130         print("No se ha seleccionado el archivo")
131         messagebox.showerror(title="Error", message="Inserte una imagen valida")
132     else:
133
134         try:
135             if modo.get() == "Erosión":
136                 erosion(image)
137             if modo.get() == "Dilatación":
138                 dilatacion(image)
139             if modo.get() == "Apertura":
140                 apertura(image)
141             if modo.get() == "Cerradura":
142                 cerradura(image)
143             if modo.get() == "Gradiente Morfológico":
144                 gradiente(image)
145         except:
146             result.config(text="No se puede realizar esa operación")
147             print("No se puede realizar esa operación")
148
```

Figura 9. Código de la función start.

Esta función decide qué hacemos con base en la operación seleccionada.

En la siguiente imagen se muestra la función choose y choose2:

```
150 file = ""
151 def choose():
152     global file
153     file = filedialog.askopenfilename(filetypes = [
154         ("image", ".jif"),
155         ("image", ".jpeg"),
156         ("image", ".png"),
157         ("image", ".jpg"),
158         ("image", ".bmp")]) #Lista de tipos de archivos admitidos por este programa
159     if len(file) > 0:
160         global image
161         image = cv2.imread(file)
162         fileLabel.configure(text=file)
163
164 file2 = ""
165 def choose2():
166     global file2
167     file2 = filedialog.askopenfilename(filetypes = [
168         ("image", ".jif"),
169         ("image", ".jpeg"),
170         ("image", ".png"),
171         ("image", ".jpg"),
172         ("image", ".bmp")]) #Lista de tipos de archivos admitidos por este programa
173     if len(file2) > 0:
174         global image2
175         image2 = cv2.imread(file2)
```

Figura 10. Código de la función choose.

Estas funciones se utilizan para cargar las imágenes que vamos a analizar.





En la siguiente imagen se muestra la parte del código que utilizamos para crear la interfaz gráfica.

```
177 #Ventana principal
178 window = Tk()
179 window.title("Morfología Matemática Comunidad 1")
180
181 window.config(bg = "SteelBlue2")
182 window.iconbitmap("./root/morfoimg.ico")
183
184 image = tki.PhotoImage(file="IPN.png")
185 imageS = image.subsample(6)
186 widget = tki.Label(image=imageS, bg = "SteelBlue2")
187 widget.place(x=45,y=4)
188
189 image2 = tki.PhotoImage(file="ESCOM.png")
190 imageS2 = image2.subsample(14)
191 widget2 = tki.Label(image=imageS2, bg = "SteelBlue2")
192 widget2.place(x=440,y=5)
193 window.geometry("600x500")
194
195 lbl = Label(window, text="ESCOM - IPN\n\n Morfología Matemática\n\n Operaciones Disponibles\n", font=("Arial", 15))
196 lbl.place(x=160, y=20)
197
198 s = ttk.Style()
199 s.configure("Peligro.TCombobox", foreground="black", width=20)
200 s.map("Peligro.TCombobox", foreground=[("active", "#FFA500")])
201
202 #menú de opciones para seleccionar la operación
```

Figura 11. Código de la interfaz gráfica.

Como vemos, creamos la ventana y la personalizamos.

## 5. PRUEBAS

Al final para poner a prueba nuestro programa aislamos en la siguiente imagen de un automóvil las letras de la matrícula que se puede ver en la Figura 12.



*Figura 12. Imagen a la que le aplicaremos la transformación.*

Convertimos la imagen a grises para poder operarla de mejor forma como vemos en la Figura 13.



*Figura 13. Imagen en grises.*



Posteriormente se binariza la imagen en grises para poder aplicar los filtros de morfología matemática.



*Figura 14. Imagen Binarizada.*

Con la intención de reducir el ruido en la imagen intentaremos utilizar un filtro promedio con un kernel de 3x3.



*Figura 15. Imagen con un filtro promedio aplicado utilizando un kernel de 3x3.*

Intentaremos además aplicar el filtro cerradura utilizando la imagen binarizada.



*Figura 16. Imagen binarizada a la que se le aplicó el filtro de cerradura.*

Aplicamos el filtro blackhat a este resultado



*Figura 17. Imagen binarizada a la que se le aplicó el filtro blackhat.*

Después se usa la aplicación de los contornos, de esta manera obtenemos la siguiente imagen en la cual se observa con menor ruido la escena y obtenemos las letras de las placas de manera resaltada.



*Figura 18. Uso de contornos para la imagen procesada con los métodos correspondientes.*

Para concluir nuestro proceso, utilizamos aritmética de operaciones para la imagen, en este caso usamos el operador OR, con las siguientes imágenes:



*Figura 19 Imágenes seleccionadas para hacer la operación OR.*

Obtenemos cómo resultado:



*Figura 20 Resultados obtenidos*



## 6. CONSLUSIONES

El objetivo del trabajo es lograr la segmentación y procesamiento de las letras en la placa utilizando la aplicación que se propuso en la cual se aplican ciertos procedimientos, propiedades y operaciones con la finalidad de ayudarnos a lograr el objetivo. Como se pudo observar en la sección del prototipo, hemos sido capaces de alcanzar dicho objetivo.

Los procesos que se han programado y probado por esta comunidad son: erosión, dilatación, apertura, cierre y gradiente morfológico; Al final estos sirvieron para resolver el problema, pues, como se vio en la parte del desarrollo del prototipo, se obtuvieron mejores resultados en la segmentación, pero es importante saber cómo combinar y aplicar los filtros para poder visualizar un buen resultado, con nuestro objeto bien segmentado.

Al realizar este proceso se fue capaz de lograr la segmentación. Ahora, también se probaron las aplicaciones que realizaron el resto de las comunidades para saber si resuelven de mejor forma nuestro problema o qué problemáticas más se pueden resolver con morfología matemática.

En resumen, el prototipo ha sido exitoso en términos de su capacidad para realizar las operaciones morfológicas de manera precisa y obtener resultados satisfactorios en la mejora y procesamiento de las imágenes seleccionadas, los resultados obtenidos han sido prometedores y nos impulsan a continuar explorando nuevas oportunidades y mejoras dentro del campo de la morfología matemática aplicada al procesamiento de imágenes.



## 7. REFERENCIAS

- Aguilar, M., & Hernández Lamonedá, L. (Eds.). (2015). Aportaciones Matemáticas: Memorias de la sociedad matemática universitaria. Recuperado 20 de junio de 2023, de [https://www-optica.inaoep.mx/~gurcid/AM201549\\_41-77.pdf](https://www-optica.inaoep.mx/~gurcid/AM201549_41-77.pdf)
- colaboradores de Wikipedia. (2021). Morfología matemática. Wikipedia, la enciclopedia libre. [https://es.wikipedia.org/wiki/Morfolog%C3%ADa\\_matem%C3%A1tica#:~:text=La%20morfolog%C3%ADa%20matem%C3%A1tica%20se%20desarroll%C3%B3,te%C3%B3rico%20de%20la%20morfolog%C3%ADa%20matem%C3%A1tica](https://es.wikipedia.org/wiki/Morfolog%C3%ADa_matem%C3%A1tica#:~:text=La%20morfolog%C3%ADa%20matem%C3%A1tica%20se%20desarroll%C3%B3,te%C3%B3rico%20de%20la%20morfolog%C3%ADa%20matem%C3%A1tica)
- González, R. (2010, 14 mayo). Capítulo 2: Procesamiento de imágenes. udlap.mx. [http://catarina.udlap.mx/u\\_dl\\_a/tales/documentos/mel/gonzalez\\_g\\_ra/capitulo2.pdf](http://catarina.udlap.mx/u_dl_a/tales/documentos/mel/gonzalez_g_ra/capitulo2.pdf)
- Mares Javier, M. (2014). Morfología matemática: Un enfoque al procesamiento digital de imágenes [Tesis de licenciatura]. Universidad Autónoma De Puebla. <https://repositorioinstitucional.buap.mx/handle/20.500.12371/6921?locale-attribute=en>
- Procesamiento Morfológico de Imágenes en Color. Aplicación a la Reconstrucción Geodésica. (s. f.). [https://rua.ua.es/dspace/bitstream/10045/10053/5/Ortiz-Zamora-Francisco-Gabriel\\_4.pdf](https://rua.ua.es/dspace/bitstream/10045/10053/5/Ortiz-Zamora-Francisco-Gabriel_4.pdf)
- Tema 5: Morfología. (s. f.). alojamientos.us.es. <http://alojamientos.us.es/gtocom/pid/tema5-1.pdf>