



Universidad Nacional de Rosario

Facultad de Ciencias Exactas, Ingeniería y Agrimensura

Tecnicatura

Universitaria en Inteligencia Artificial

Procesamiento de Imágenes I - IA 4.4

TRABAJO PRÁCTICO N°2 - Año 2024 - 2º Semestre

Alumnos:

Antuña, Franco A-4637/1

Gallardo, Jonatan G-5970/6

Orazi, Roberto O-1815/5

Problema 1 - Detección y clasificación de monedas y datos.....	3
Descripción del problema.....	4
Implementación.....	4
Resultado.....	9
Problema 2 - Detección de patentes.....	10
Descripción del problema.....	10
Implementación.....	10
Conclusión Final.....	12

Problema 1 - Detección y clasificación de monedas y dados

En la figura 1 se muestra la imagen del archivo *monedas.jpg*, la cual consiste en un fondo de intensidad no uniforme sobre el cual se hallan dados y monedas de distinto valor y tamaño.

Se debe elaborar un algoritmo capaz de procesar dicha imagen y resolver los siguientes puntos sobre ella:

- A. Procesar la imagen de manera de segmentar las monedas y los dados de manera automática.
- B. Clasificar los distintos tipos de monedas y realizar un conteo automático
- C. Determinar el número del valor que representa la cara superior de cada dado y realizar un conteo automático.

Aviso: En cada punto, el script elaborado debe informar y mostrar los resultados en cada una de las etapas de procesamiento



Figura 1: Imagen con monedas y dados del archivo *monedas.jpg*.

Descripción del problema

El objetivo principal del proyecto es desarrollar un algoritmo capaz de procesar una imagen denominada *monedas.jpg*, la cual contiene monedas de distintos valores y tamaños, así como dados con caras visibles en un fondo de intensidad no uniforme. Se busca segmentar automáticamente estos objetos, clasificar las monedas según su valor nominal, determinar el número visible en la cara superior de cada dado y realizar un conteo total. Para cada paso, el script debe generar resultados visuales y numéricos.

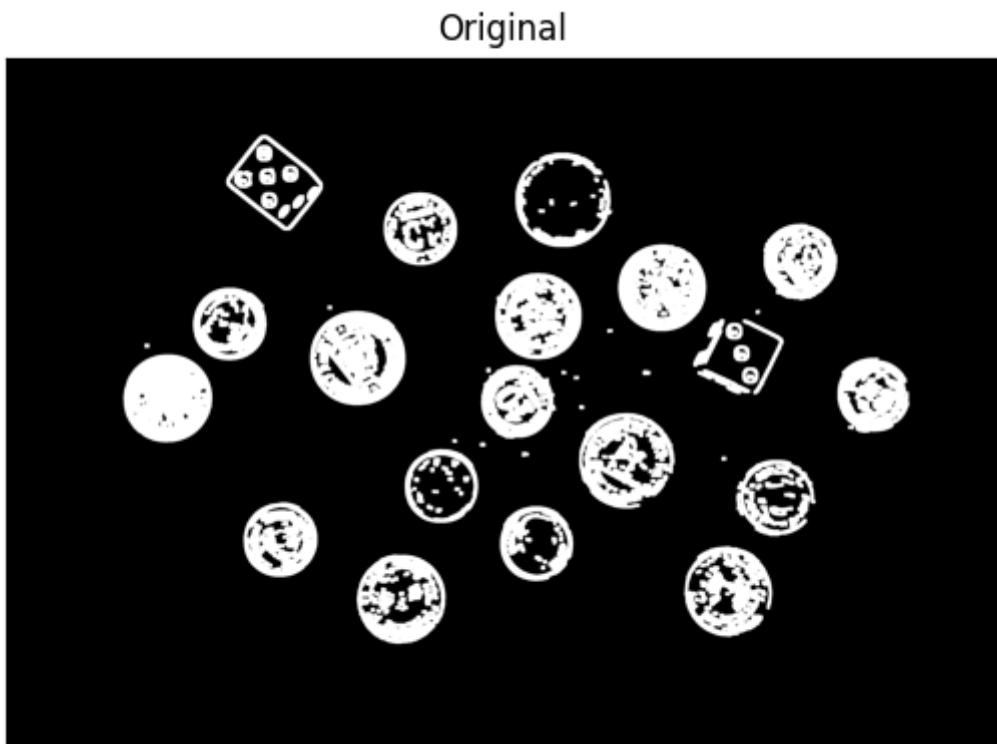
Implementación

La implementación del proyecto se llevó a cabo mediante un conjunto de etapas diseñadas para procesar y resaltar las características relevantes de la imagen original. En primer lugar, se adquirió la imagen de entrada y se preparó adecuadamente mediante su conversión a escala de grises, lo que permitió simplificar el análisis eliminando la información de color. Posteriormente, se aplicó un filtro Gaussiano para reducir el ruido presente en la imagen, mejorando así la definición de los bordes en las etapas posteriores.

Para identificar los contornos principales, se utilizó el operador *Canny*, el cual generó un mapa inicial de bordes basado en las diferencias de intensidad. Este mapa fue refinado mediante una operación de dilatación que amplió los bordes detectados, haciéndolos más prominentes y continuos. Finalmente, se llevó a cabo un cierre morfológico que permite llenar pequeñas discontinuidades en los bordes, logrando una representación más uniforme y preparada para su análisis.

La figura 1 muestra el resultado intermedio obtenido tras aplicar la operación de dilatación sobre la imagen procesada. En esta etapa, los bordes detectados previamente mediante el operador *Canny* se amplían utilizando un kernel de tamaño 13 x 13. Esto permite resaltar las áreas delimitadas por los contornos, haciéndolas más prominentes y uniformes. La dilatación es especialmente útil para mejorar la continuidad de los bordes y facilitar etapas posteriores de procesamiento, como la segmentación o el análisis de formas.

Figura 1. Resultado intermedio después de la dilatación aplicada a la imagen original.



Para reducir el ruido y preservar las características esenciales de los objetos, se implementa un filtro contraharmónico. Este filtro es eficaz para eliminar ruido de tipo sal sin afectar los bordes, lo cual resulta fundamental en imágenes donde los objetos tienen bordes suaves o finos. Una vez limpia la imagen, se procede a llenar huecos en las regiones segmentadas mediante un procedimiento de reconstrucción morfológica iterativa. Este paso garantiza que las monedas y dados queden completamente definidos, incluso si inicialmente presentan interrupciones en sus bordes.

El desarrollo del proyecto incluyó la creación de varias funciones personalizadas para abordar tareas específicas en el procesamiento de la imagen. A continuación, se presenta un resumen de las mismas:

La función **imreconstruct** implementa la reconstrucción morfológica a partir de un marcador y una máscara. Este proceso iterativo combina dilataciones sucesivas con intersecciones, permitiendo restaurar o reconstruir regiones de interés según las restricciones impuestas por la máscara. Es fundamental para operaciones como la restauración de formas específicas en imágenes binarias.

La función **imfillhole** se encarga de llenar los huecos en las regiones de una imagen binaria. Utilizando la reconstrucción morfológica como núcleo, esta función invierte la imagen y utiliza los bordes como marcadores para identificar y completar las áreas internas que corresponden a huecos. Al final, el resultado es invertido nuevamente para devolver la imagen con los huecos rellenos.

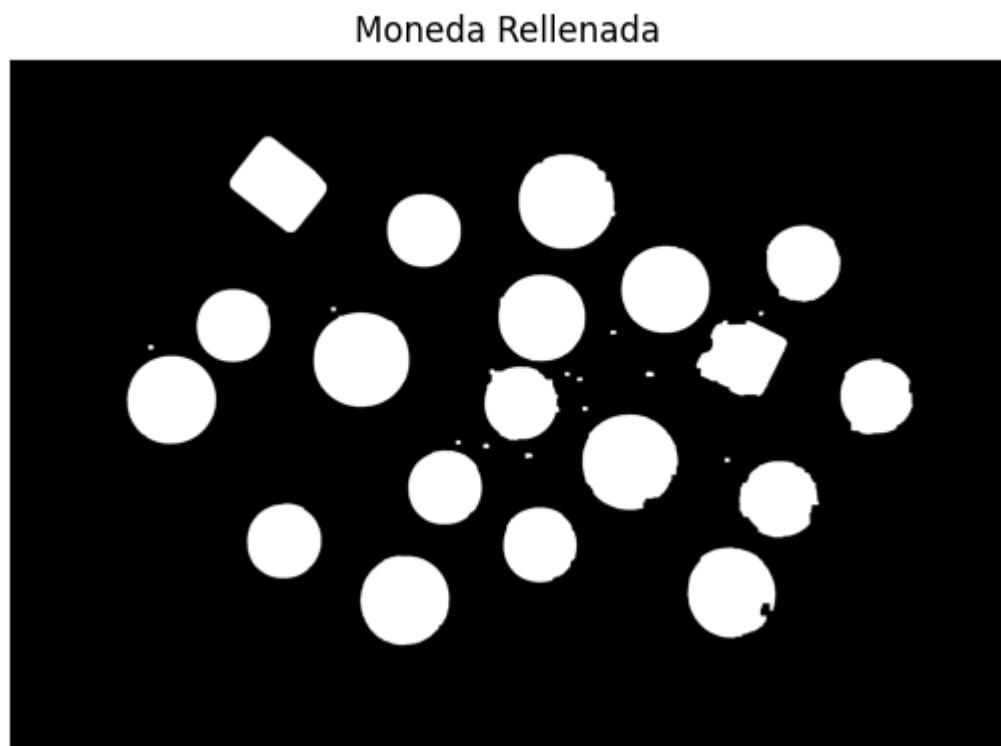
La función ***menor_factor_de_forma*** permite identificar el objeto con el menor factor de forma dentro de un conjunto de registros. Esto es útil en aplicaciones donde se analiza la forma de los objetos para clasificar o filtrar elementos específicos.

Finalmente, la función ***menor_area*** localiza el objeto con el área más pequeña dentro de un conjunto de objetos. Esto se utiliza para determinar características relevantes relacionadas con el tamaño de los elementos en la imagen, permitiendo discriminar entre diferentes regiones de interés.

Posteriormente, se aplicó un proceso de relleno de huecos en la imagen procesada. Para ello, se utilizó la función personalizada ***imfillhole***. Este proceso asegura que los objetos detectados no presenten discontinuidades internas, facilitando la identificación precisa de sus características.

La Figura 2 muestra el resultado de este paso, donde los huecos dentro de las formas detectadas han sido completamente llenados, destacando así las regiones sólidas correspondientes a las monedas y otros objetos en la imagen procesada.

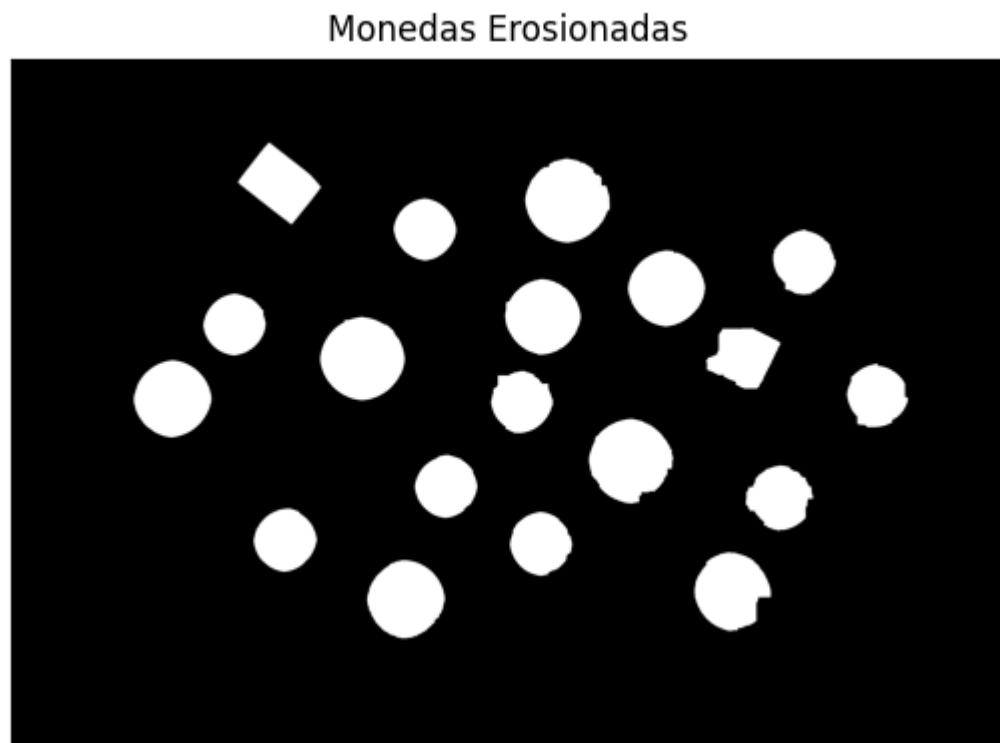
Figura 2. Imagen binaria tras el proceso de relleno de huecos aplicado a los bordes detectados.



A continuación, se realizó un proceso de erosión sobre la imagen resultante del relleno de huecos. Este paso tiene como objetivo reducir el tamaño de las regiones detectadas, eliminando posibles irregularidades en los bordes externos y refinando las

formas. Para ello, se utilizó un kernel de 41x41, lo que permite aplicar una erosión significativa, ajustada al tamaño de los objetos presentes en la imagen.

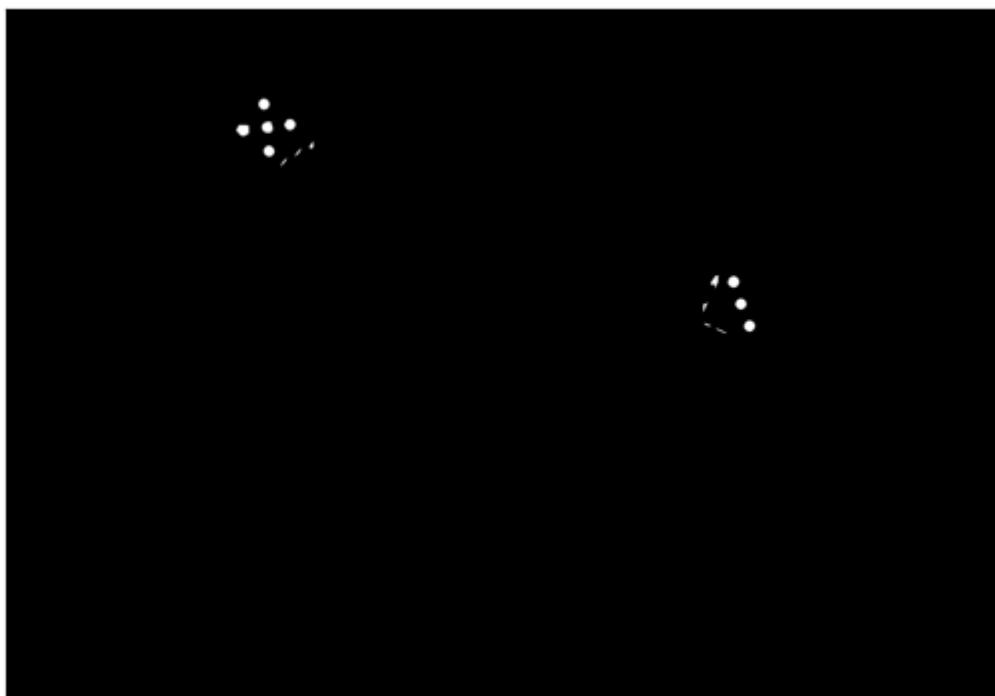
Figura 3. Imagen binaria tras el proceso de erosionado



Por otra parte, se aisló la región correspondiente a los datos mediante un procesamiento adicional. A estas regiones se les aplicó un refinamiento morfológico, que incluyó erosiones sucesivas y relleno de huecos. Esto permitió mejorar la precisión en la segmentación de los datos, eliminando detalles redundantes y asegurando la uniformidad de las regiones extraídas.

La Figura 4 muestra el resultado de esta etapa, donde los datos han sido erosionados y procesados para una mejor visualización y análisis.

Dados erosionados



En la última fase del procesamiento, se llevó a cabo la identificación y etiquetado de las monedas y los dados mediante un análisis más preciso de las regiones segmentadas. Para ello, se utilizaron las estadísticas generadas por la función **cv2.connectedComponentsWithStats** para los objetos identificados como dados. Se filtraron los componentes con un área superior a 1000 píxeles, lo que permitió enfocar la atención únicamente en las regiones relevantes.

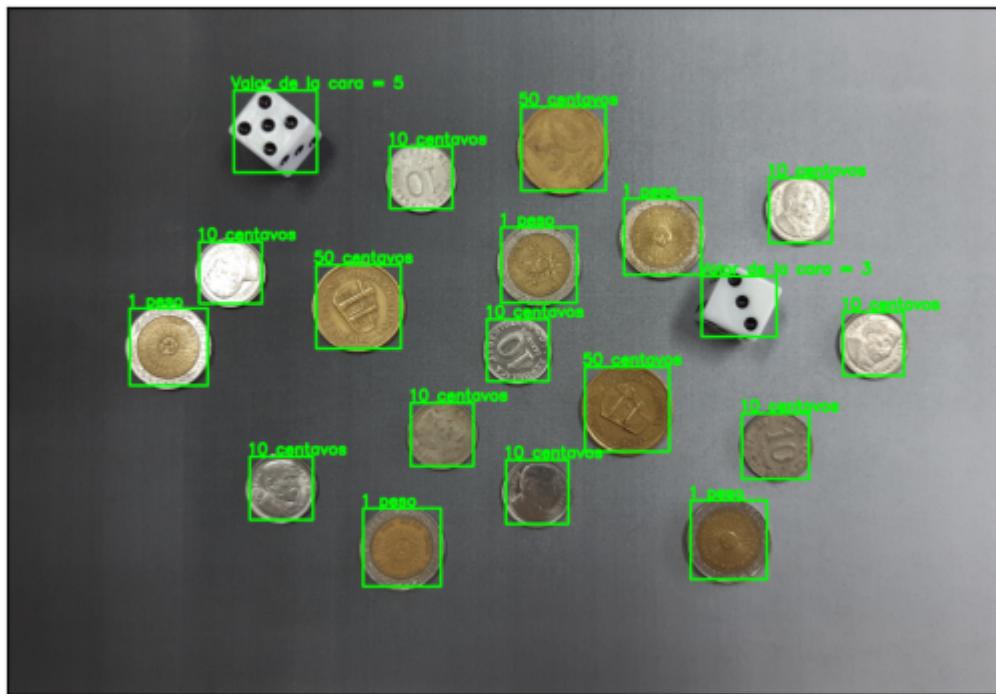
Además, se analizaron las caras de los dados, y a partir de la cantidad de componentes conectados en estas áreas, se asignaron valores a cada cara. Esta información se utilizó para etiquetar los dados con su valor correspondiente, que fue incorporado a la imagen.

Por otro lado, las monedas fueron etiquetadas y clasificadas en tres categorías según su tamaño. Se dibujan rectángulos alrededor de cada objeto, y se añadieron etiquetas con los valores correspondientes de las monedas, como "10 centavos", "1 peso" y "50 centavos". De manera similar, los dados también fueron etiquetados y rodeados por rectángulos, mostrando el valor de la cara detectada en cada uno.

Resultado

La figura 5 muestra la imagen final, donde tanto las monedas como los dados han sido etiquetados y destacados mediante rectángulos, con las correspondientes etiquetas visualizadas sobre cada objeto.

Dados y monedas etiquetados por valor



Se logró implementar un sistema eficaz para la detección, clasificación y etiquetado de monedas y dados a partir de imágenes utilizando técnicas de procesamiento de imágenes. El proceso comenzó con la segmentación de las regiones de interés mediante la conversión a escala de grises, el filtrado, y la detección de bordes. Posteriormente, se aplicaron operaciones morfológicas como la dilatación y la erosión para refinar la imagen y eliminar el ruido, mejorando la segmentación de los objetos.

El uso de componentes conectados permitió identificar y clasificar los objetos en la imagen según su área y forma, lo que facilitó la distinción entre las monedas y los dados. Las monedas fueron etiquetadas por valor, y se asignaron valores a las caras de los dados mediante el análisis de las componentes conectadas.

Se logró un etiquetado claro y preciso de los objetos detectados, destacando las monedas de diferentes denominaciones y los valores de los dados, lo que demuestra la efectividad de las técnicas de procesamiento de imágenes empleadas.

Problema 2 - Detección de patentes

En la figura 2 se muestra una de las doce imágenes (archivos img<id>.png), las cuales representan la vista anterior o posterior de diversos vehículos. En cada una de ellas se puede visualizar las correspondientes patentes.

Se debe elaborar un algoritmo capaz de procesar cada una de estas imágenes y resolver los siguientes puntos en cada uno de ellas:

- A. Detectar automáticamente la placa patente y segmentar la misma. Informar las distintas etapas de procesamiento y mostrar los resultados de cada etapa
- B. Implementar un algoritmo de procesamiento que segmenta los caracteres de la placa patente detectada en el punto anterior. Informar las distintas etapas de procesamiento y mostrar los resultados de cada etapa.



Figura 2: Ejemplo de una de las imágenes (archivo *img05.png*) de la carpeta *Patentes*.

Descripción del problema

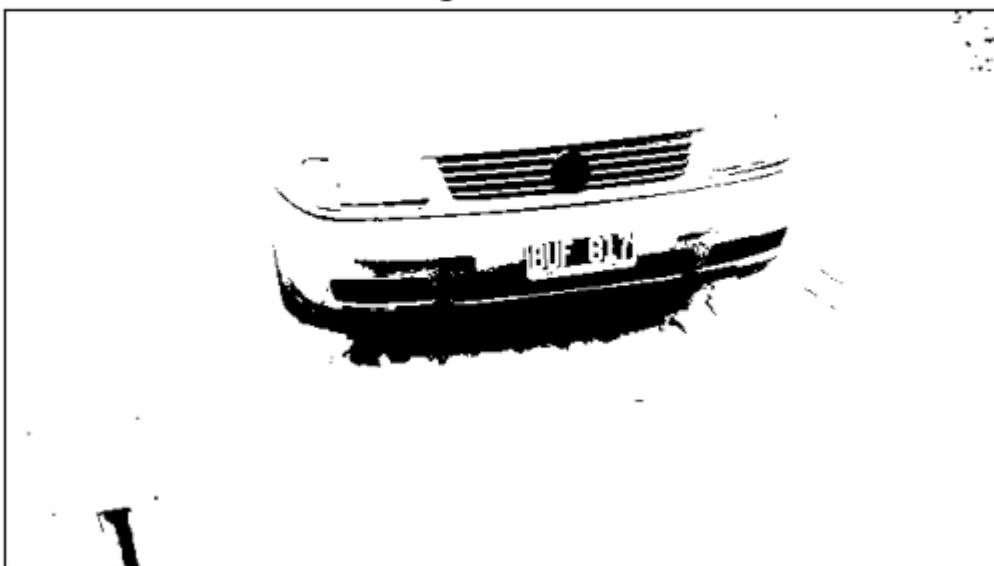
El propósito principal del script es procesar imágenes de vehículos para identificar y extraer posibles regiones que correspondan a las patentes, utilizando técnicas de procesamiento digital de imágenes y segmentación.

Implementación

En este segundo ejercicio, se utilizará la "Imagen 1" como base para todo el proceso. No repetiremos las etapas iniciales de preprocesamiento.

Se realiza un proceso iterativo para detectar y filtrar componentes en imágenes que representan patentes. El procedimiento comienza con la lectura de varias imágenes en formato RGB y la conversión a escala de grises para facilitar el análisis. Luego, se aplica un umbral binario inicial de 50, que convierte la imagen en blanco y negro, separando los objetos del fondo. Este proceso de umbralización se repite, aumentando el valor del umbral progresivamente, hasta que se detectan al menos seis componentes conectados que coinciden con las características de una patente.

Imagen binaria



Para identificar los componentes, se utilizan componentes conectadas, que etiqueta las regiones conectadas y calcula estadísticas sobre ellas, como el área y la relación de aspecto. Se filtran los componentes que tienen un área entre 17 y 200 píxeles y cuya relación de aspecto se encuentra entre 1.5 y 3.0, lo que permite identificar formas que se asemejan a las patentes.

A continuación, se agrupan los componentes que están cerca unos de otros, basándose en las distancias entre los centros de sus cuadros delimitadores. Si se encuentran entre seis y ocho componentes agrupados, el algoritmo asume que ha identificado una patente correctamente.

Una vez que se han seleccionado los componentes correctos, se dibujan cuadros delimitadores verdes sobre la imagen original para resaltar las patentes. Además, se calcula una caja delimitadora que envuelve todos los componentes de la patente detectada, y se dibuja un cuadro azul alrededor de esa área global.

El resultado final muestra la imagen con los cuadros delimitadores, indicando que las patentes han sido correctamente identificadas. Este proceso es iterativo y permite ajustar el umbral para mejorar la detección en cada imagen.

Conclusión Final

El algoritmo desarrollado para la detección y segmentación de patentes vehiculares ha demostrado ser eficaz en casi la totalidad de las imágenes procesadas. A pesar de los desafíos técnicos identificados, como la sensibilidad a ciertas características de las imágenes (color del vehículo, iluminación, y contraste) y la complejidad de la etapa final de agrupamiento, los resultados obtenidos validan el enfoque implementado.

En todos los casos, el sistema logra:

- Identificar componentes relevantes de las patentes en la mayoría de las imágenes, filtrando adecuadamente elementos no deseados.
- Detectar y reconstruir correctamente las patentes completas o, en su defecto, las letras y números que las componen en escenarios más complicados.
- Funciona de manera robusta bajo diversas configuraciones, con un buen equilibrio entre precisión y eficiencia.

