



---

Universidad Nacional de Rosario

## **Trabajo práctico N° 02**

### **Procesamiento de Imágenes**

Tecnicatura Universitaria en Inteligencia Artificial

Periodo: Diciembre 2025

Integrantes:

- de Brito, Nicolás - Legajo D-4416/4
- Giacone, Agustín - Legajo G-5917/1
- Taborda, Matías - Legajo T-3158/5

# Índice

<b>Introducción.....</b>	<b>2</b>
<b>Problema 1 - Detección y clasificación de monedas y datos.....</b>	<b>3</b>
Contexto del problema.....	3
Resolución del problema.....	4
Identificación de elementos - Mascara.....	4
Identificación de elementos - Segmentación.....	7
Clasificación de monedas.....	9
Detectar valores de los datos.....	10
<b>Problema 2 - Detección de patentes.....</b>	<b>12</b>
Contexto del problema.....	12
Resolución del problema.....	12
Segmentación de patentes.....	12
Segmentación de caracteres.....	19
Resultados integrales.....	22
Visualización final.....	23
Cuadro resumen de estadísticas.....	29
<b>Conclusión.....</b>	<b>30</b>

## Introducción

Este informe detalla la resolución de los problemas propuestos, aplicando diversas técnicas estudiadas en la asignatura. Para cada ejercicio, se explican detalladamente los pasos seguidos, algunos obstáculos encontrados y las soluciones implementadas.

Se incluyen recursos visuales, como imágenes, para facilitar la comprensión de los resultados y, en ciertos casos, ejemplos prácticos del funcionamiento de los programas desarrollados. El trabajo se basa en un repositorio de GitHub que contiene un archivo README con las instrucciones de ejecución, dos archivos Python con las soluciones y el material didáctico proporcionado por la cátedra.

# Problema 1 - Detección y clasificación de monedas y dados

## Contexto del problema

El trabajo consiste en el desarrollo de un algoritmo capaz de procesar una imagen que contiene monedas de distinto valor y tamaño, y dados. Las tareas específicas requeridas son:

1. **Segmentación:** Separar las monedas de los dados en la imagen.
2. **Clasificación y Conteo de Monedas:** Distinguir los diferentes tipos de monedas y determinar su cantidad total.
3. **Detección y Conteo de Dados:** Identificar la cara superior de cada dado para determinar el valor representado y realizar un conteo total de los dados.

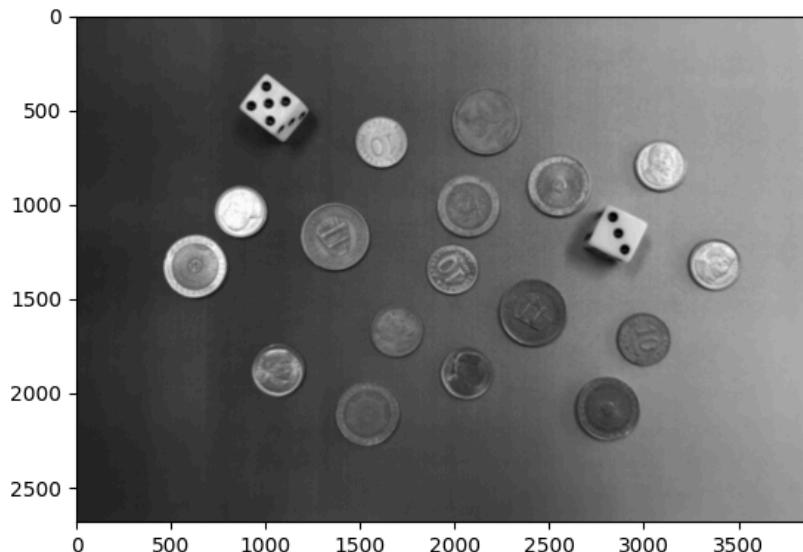
La imagen presenta un desafío adicional debido a que los objetos (dados y monedas) se encuentran sobre un fondo con intensidad de color no uniforme.



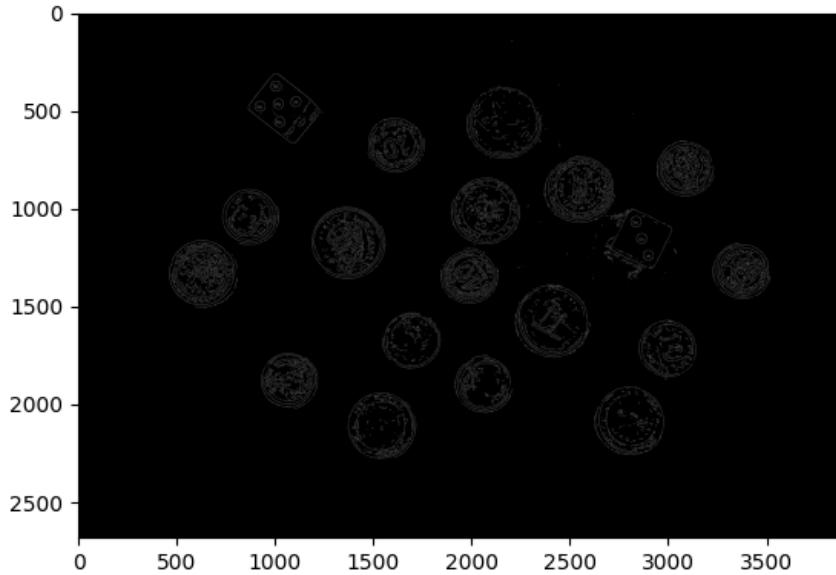
## Resolución del problema

### Identificación de elementos - Mascara

El primer objetivo fue generar una máscara que identifique los objetos presentes sobre el fondo. Para esto definimos la función `generar_mascara_identificacion()`, que recibe la imagen indicada en el enunciado y devuelve una máscara binaria con fondo negro y los objetos en blanco. El primer inconveniente que encontramos fue el fondo no uniforme, lo que dificulta separar correctamente las regiones de interés. Como primer paso aplicamos un suavizado fuerte para reducir ruido y eliminar detalles finos. Utilizamos `GaussianBlur` con un kernel de  $25 \times 25$  y obtuvimos el siguiente resultado:

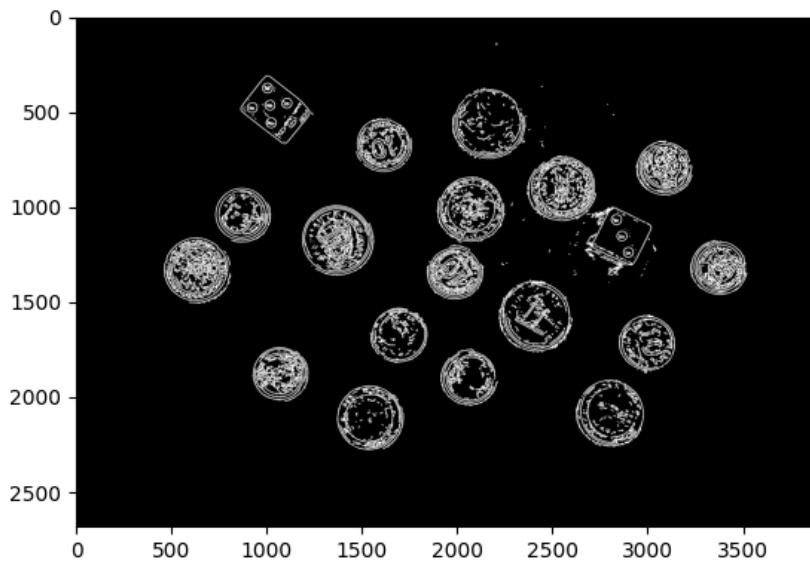


Para identificar los bordes de los objetos, se optó por aplicar el algoritmo Canny. Tras la experimentación, se determinó que los umbrales bajos ofrecían el mejor ajuste para el enfoque deseado. Específicamente, se seleccionó un umbral inferior de 10 y un umbral superior de 17, obteniendo el siguiente resultado:



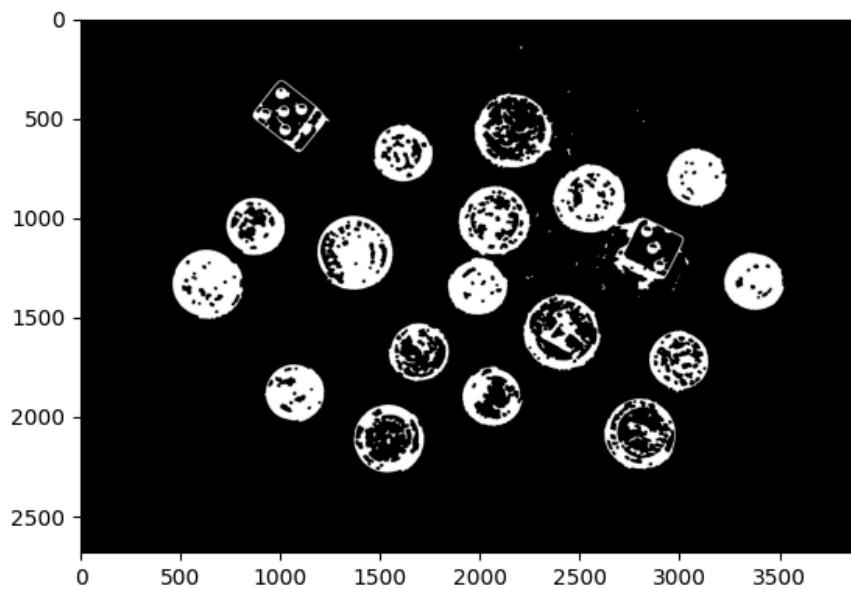
La aplicación del algoritmo Canny resultó en bordes sutiles en la imagen, lo cual fue beneficioso para reducir el ruido sin comprometer la integridad de las formas de los objetos a detectar.

Posteriormente, se inició el procesamiento morfológico para generar la máscara binaria. El primer paso consistió en una dilatación, destinada a acentuar los bordes previamente identificados por Canny. Tras varias pruebas, se determinó que un kernel elíptico de  $5 \times 5$  era el elemento estructural más adecuado para este propósito.

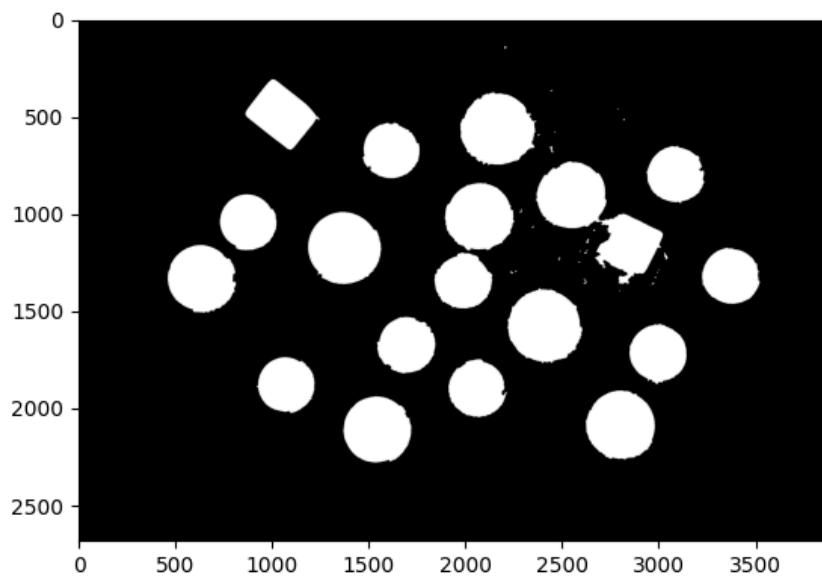


La dilatación expuso una pequeña cantidad de ruido que había quedado oculto debido a los umbrales bajos y el tamaño reducido del kernel de Canny. Además, se observó que los elementos ya estaban tomando la forma deseada. Para

proporcionarles más cuerpo, engrosar y suavizar los contornos, y conectar sus partes componentes, se aplicó una operación de clausura con un kernel elíptico de 15x15.

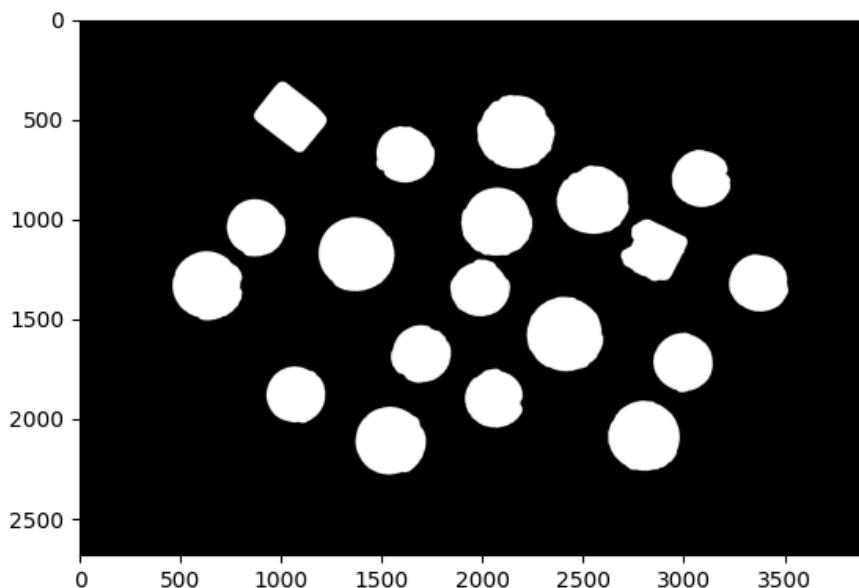


Al unir las partes de los elementos, sus contornos se cerraron, lo que permitió el relleno y la aproximación a la máscara deseada. No obstante, este proceso de clausura provocó que algunos elementos adyacentes se fusionaran y persistiera el problema del ruido (pequeños puntos blancos).



Tras el proceso de relleno, los elementos se mostraron significativamente más próximos a una representación que facilitaría su diferenciación. A continuación, se procedió a eliminar el ruido y a corregir las deformaciones en los contornos de algunos objetos, específicamente el dado de la derecha y la moneda superpuesta.

Se optó por aplicar una operación de apertura, ya que esta técnica es efectiva para eliminar el ruido de pequeño tamaño y separar objetos que se encuentran apenas en contacto. Tras la experimentación, se determinó que el mejor resultado se obtuvo utilizando un kernel elíptico con dimensiones de 65x65 píxeles.



Aunque el contorno final de los objetos no resultó completamente exacto, se consideró que la calidad obtenida era suficiente para el posterior procesamiento y para alcanzar los objetivos establecidos en el enunciado.

Cabe destacar que la combinación de operaciones morfológicas y los elementos estructurales seleccionados fueron el resultado de una extensa experimentación. Se evaluaron múltiples combinaciones, algunas arrojando resultados similares, hasta dar con la estrategia descrita, la cual se consideró la más óptima.

## Identificación de elementos - Segmentación

Para diferenciar entre monedas y dados a partir de su forma, se observó inicialmente que los dados tienden a ser más cuadrados, mientras que las monedas son más circulares. Por lo tanto, se decidió utilizar el concepto de circularidad y

establecer un umbral para clasificar los elementos según su proximidad a un círculo perfecto.

El procedimiento implementado fue el siguiente:

1. Se detectaron los elementos en la imagen usando `connectedComponentsWithStats`.
2. Con los `labels` obtenidos, se identificaron los contornos mediante `findContours`.
3. Utilizando el contorno externo, se calcularon el área (`contourArea`) y el perímetro (`arcLength`) de cada figura.
4. Finalmente, se calculó la circularidad.

Tras una fase de experimentación, se definió un umbral de 0.8. Las figuras con una circularidad igual o superior a 0.8 se clasificaron como monedas. Sus estadísticas (provenientes de `connectedComponentsWithStats`) se almacenaron en un *array* específico para su posterior identificación en la imagen original.

Los elementos con una circularidad inferior a 0.8 se consideraron dados y sus estadísticas se separaron en otro *array*.

Con las coordenadas de los *bounding boxes* guardadas, se graficaron los elementos de ambos *arrays* sobre la imagen original utilizando colores distintos, obteniendo el siguiente resultado:

Elementos detectados: dados y monedas



## Clasificación de monedas

La clasificación de los distintos tipos de monedas se realizó aprovechando la diferencia en sus tamaños, medida a través del área de los objetos obtenida con `connectedComponentsWithStats`. Se identificó la moneda de mayor área (denominada `$max_area_moneda$`), correspondiente a la moneda de 50 centavos, ya que es la de mayor tamaño físico.

Experimentalmente, se establecieron los siguientes umbrales para la clasificación:

- **Monedas de 50 centavos:** Aquellas con un área mayor o igual al 95% de `$max_area_moneda$`.
- **Monedas de 1 peso:** Aquellas con un área que se encuentra en el rango del 80% al 95% de `$max_area_moneda$`.
- **Monedas de 10 centavos:** Aquellas con un área menor al 80% de `$max_area_moneda$`.

Finalmente, se utilizó esta clasificación para graficar las diferentes monedas en distintos colores sobre la imagen original.

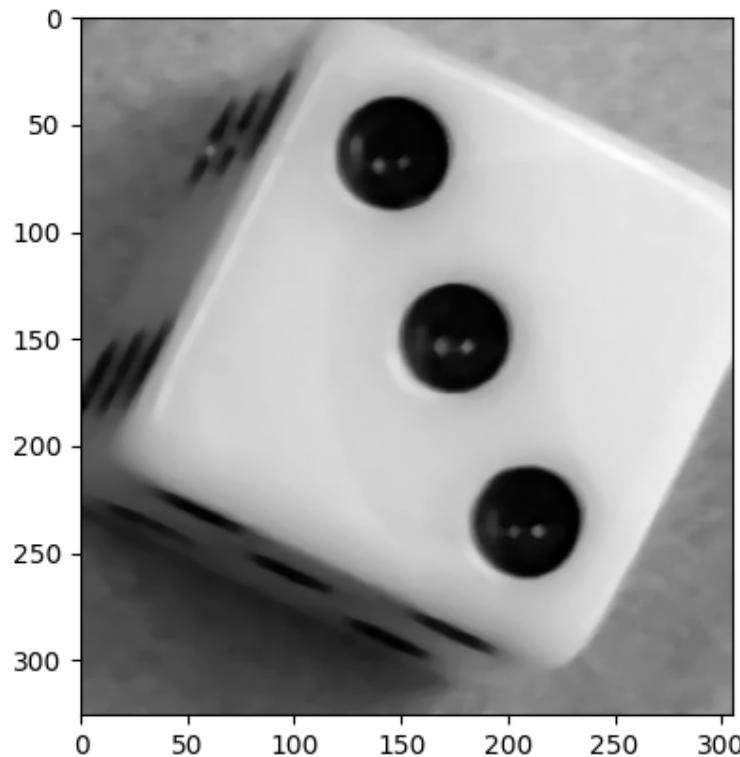


Además de la imagen, por consola se observa este mensaje:

```
*****
Conteo de monedas
*****
Se contabiliza un total de 17 monedas.
Monedas de $1: 5 unidades.
Monedas de $0.50: 3 unidades.
Monedas de $0.10: 9 unidades.
*****
Las monedas suman un total de $ 7.4
*****
```

## Detectar valores de los datos

A continuación, se detalla el procedimiento para este paso: a partir de la imagen original y las estadísticas obtenidas durante la segmentación, se procedió a recortar el objeto de interés. Este recorte se realizó sobre una versión en escala de grises de la imagen original proporcionada en el enunciado. Finalmente, se aplicó un filtro `medianBlur` con el objetivo de reducir el ruido presente en la imagen.



Dada la claridad de la imagen, se optó por aplicar directamente el algoritmo `HoughCircles`. Se ajustaron los parámetros de forma experimental para optimizar

la detección. Posteriormente, se contabilizó el número de círculos identificados y este valor se superpuso en la imagen original junto al boundingbox del dado.



Finalmente la consola muestra un mensaje similar al siguiente:

```
*****
Conteo de dados
*****
Se contabiliza un total de 2 dados.
Valor dado 1: 5
Valor dado 2: 3
*****
Los dados suman un total de 8
*****
```

# Problema 2 - Detección de patentes

## Contexto del problema

El objetivo de este trabajo es desarrollar un algoritmo capaz de procesar un conjunto de 12 imágenes de vehículos para llevar a cabo las siguientes tareas:

1. Detección y segmentación automática de la placa patente.
2. Implementación de un algoritmo de procesamiento para segmentar los caracteres de la placa patente detectada.

Las imágenes utilizadas son reales y presentan diversas complejidades, como variados colores de vehículos, presencia de ruido, obstrucciones (objetos que se interponen entre el vehículo y la cámara) y características particulares en las patentes que pueden dificultar la detección.

## Resolución del problema

### Segmentación de patentes

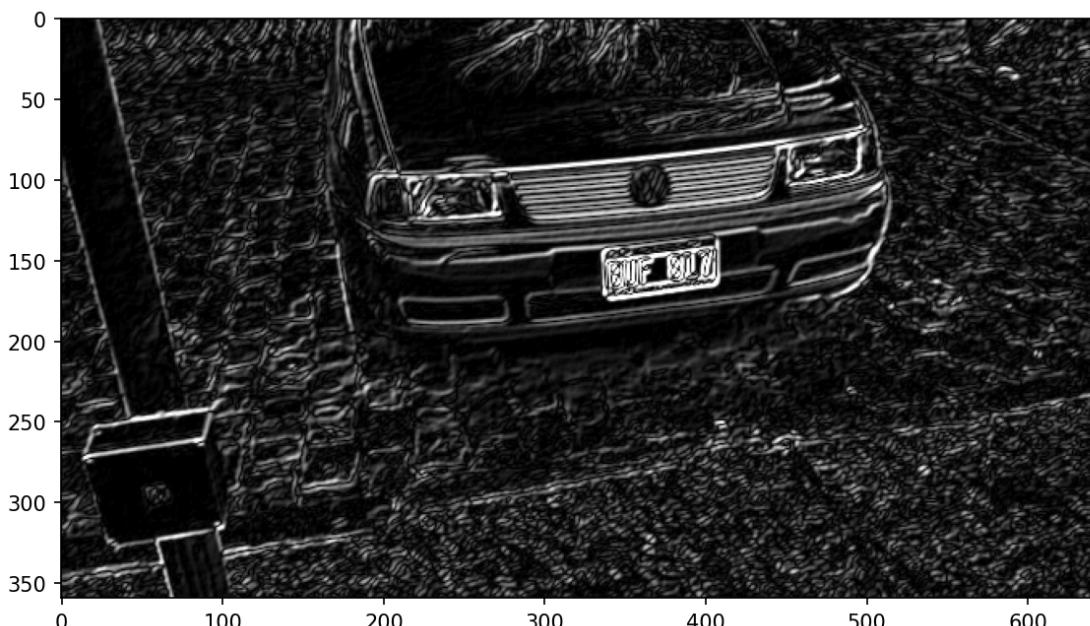
Para el seguimiento de este informe tomaremos como referencia una de las imágenes del conjunto



Dada la naturaleza distintiva de los colores en las patentes (una combinación de blanco y negro con bordes bien definidos), se optó por una estrategia de procesamiento de imágenes intensiva. El objetivo era eliminar otros elementos de la imagen, logrando aislar una forma rectangular única que sirviera como una aproximación inicial a la ubicación de la patente.

El proceso comenzó con la aplicación de un filtro GaussianBlur para suavizar la imagen y mitigar el ruido. Posteriormente, se exploraron métodos de detección de bordes. Inicialmente, se probó el algoritmo Canny seguido de operaciones morfológicas, pero los resultados no fueron satisfactorios. A pesar de varios experimentos, las combinaciones probadas no lograron un rendimiento óptimo de manera general.

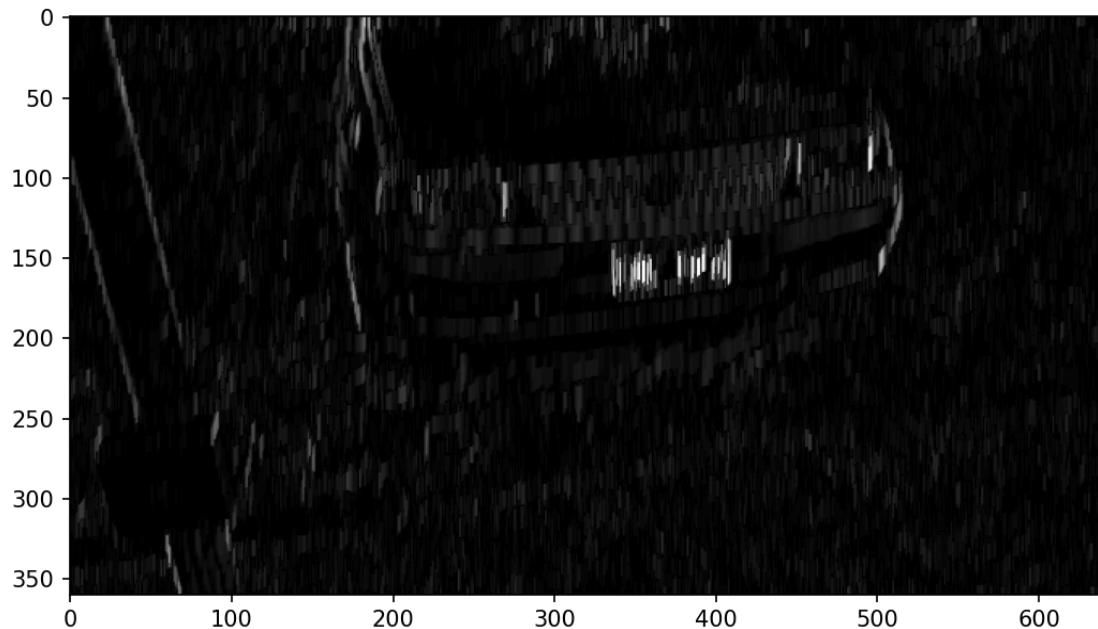
Ante esta limitación, se decidió probar otros enfoques de detección de bordes, lo que condujo a la experimentación con el método Sobel.



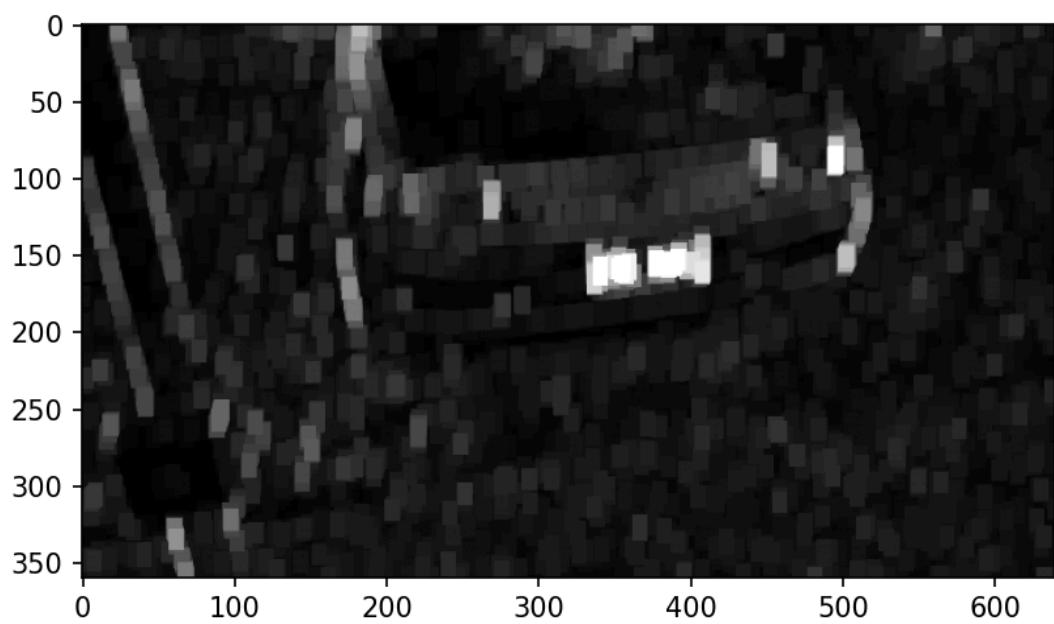
Como se mencionó anteriormente, el objetivo era obtener un recuadro blanco sobre la patente. Aunque la detección de bordes con el operador Sobel no fue tan nítida y precisa como con Canny, resultó ser más que suficiente para nuestro propósito, ya que lograba resaltar las áreas blancas de la patente sobre las negras, enfocándose fuertemente en el contraste generado.

Posteriormente, la etapa de experimentación con operaciones morfológicas en escala de grises para alcanzar el objetivo fue extensa y se convirtió en uno de los puntos más críticos del trabajo. No fue sencillo lograr una secuencia de pasos que funcionara con una efectividad superior al 90% de los casos.

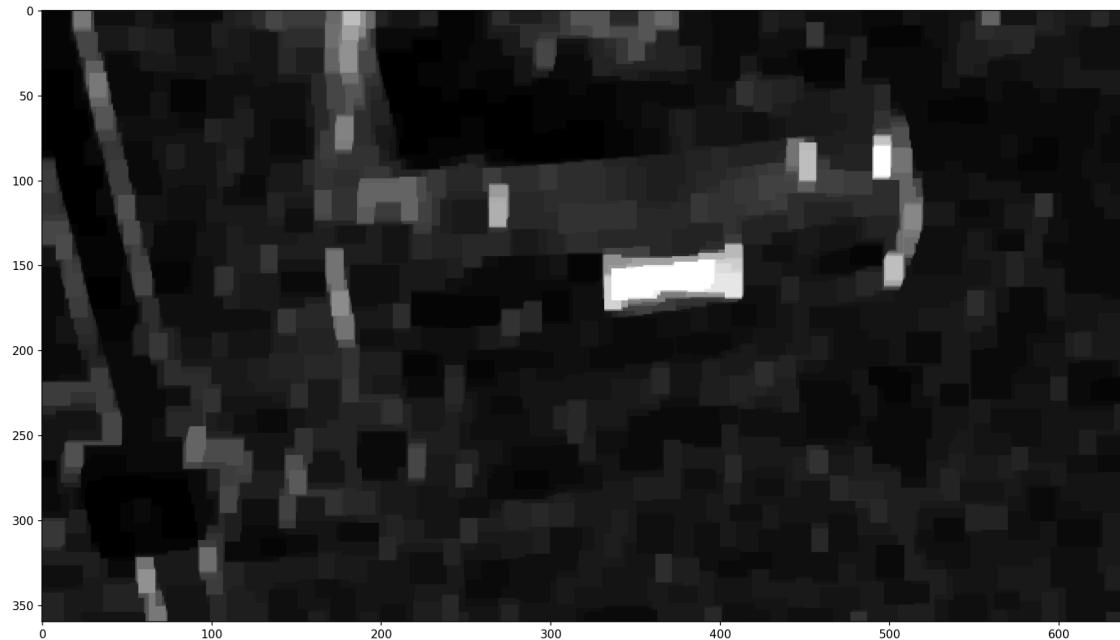
Comenzamos aplicando una operación de apertura utilizando un kernel vertical de 1x10. Esto ayudó a acentuar las líneas verticales y a eliminar pequeños detalles no deseados.



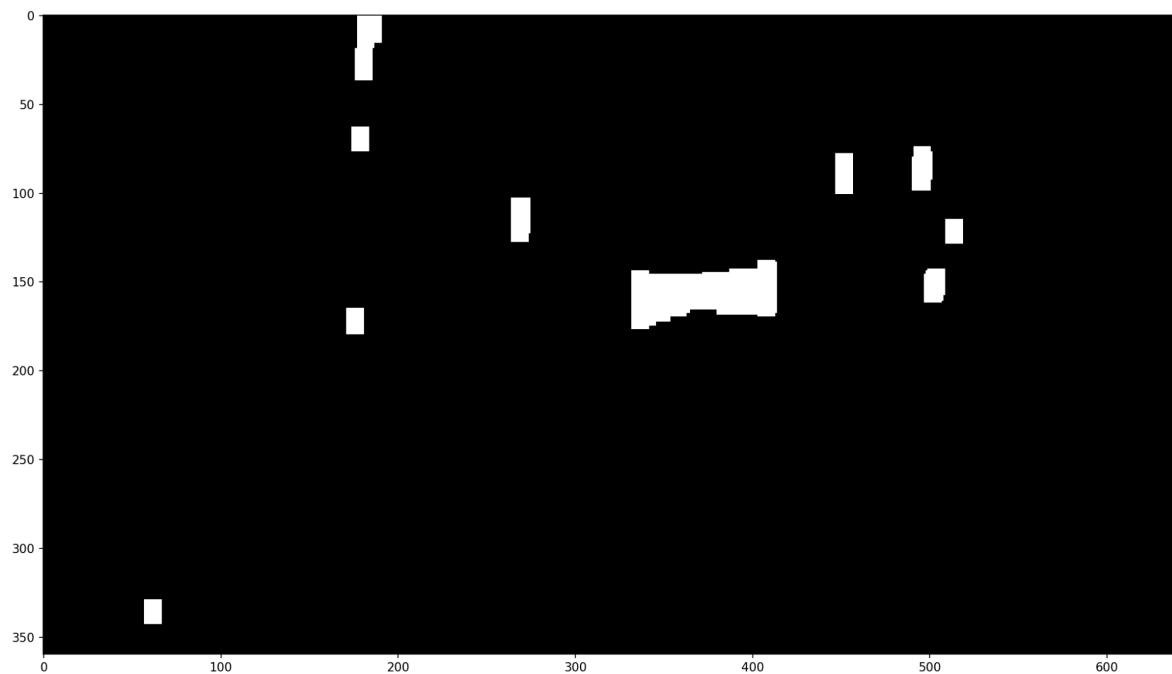
Para enfatizar aún más el resultado de la apertura, aplicamos una dilatación con un kernel rectangular de 10x5, con el objetivo de engrosar las líneas verticales que se habían resaltado previamente.



Dado que las líneas se encontraban relativamente cerca, consideramos adecuado aplicar una operación de clausura utilizando un kernel rectangular de 15x10. El objetivo de esta operación fue lograr una mejor unión entre las mencionadas líneas.



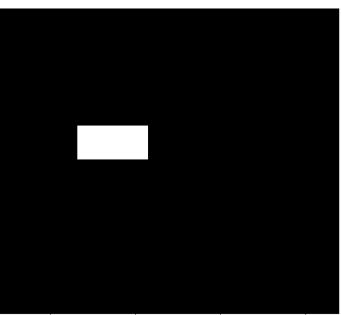
Finalmente, al alcanzar un estado muy cercano al objetivo deseado, consideramos oportuno aplicar un umbral a la imagen. El valor de umbral de 129 se determinó mediante la experimentación con las imágenes utilizadas.



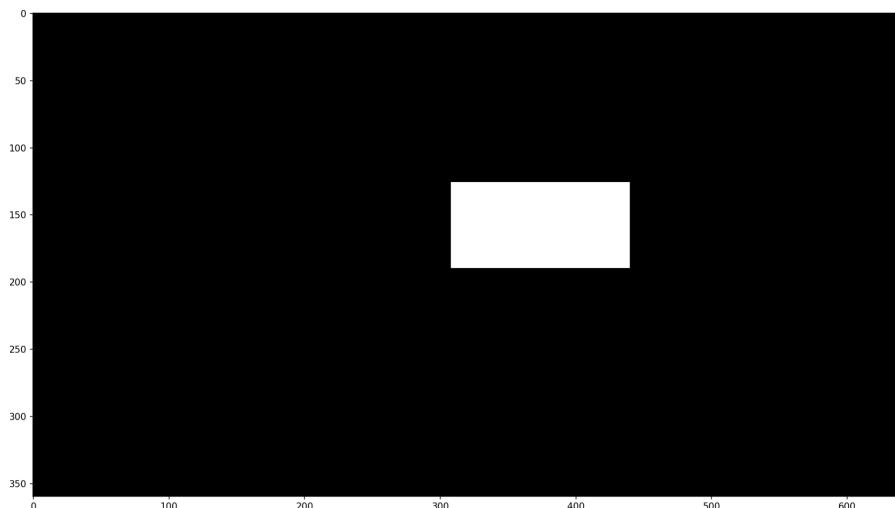
Tras umbralizar la imagen, se procedió a la detección de bordes utilizando la función `findContours`. Esto permitió obtener datos específicos de los elementos, como las componentes del *bounding box* con `boundingRect` y el área con `contourArea`. El objetivo principal era aislar y almacenar tanto el contorno como el *bounding box* de la patente. Un desafío importante en este proceso fue la necesidad de eliminar elementos no deseados, que, si bien eran pequeños en el ejemplo actual, en otros casos presentaban mayor tamaño y una morfología más variada.

Para aislar el elemento principal, se aplicó un filtrado basado en la relación ancho/alto (ratio) y el área del *bounding box*. Dado que las patentes son rectangulares, se esperaba un ratio mayor a 1. Se implementó un criterio más estricto, seleccionando elementos con un ratio entre 2 y 3.5. Adicionalmente, para descartar elementos demasiado pequeños, se estableció un umbral de área mínima de 200, ya que las patentes con áreas inferiores a este valor resultaban indistinguibles y difíciles de segmentar.

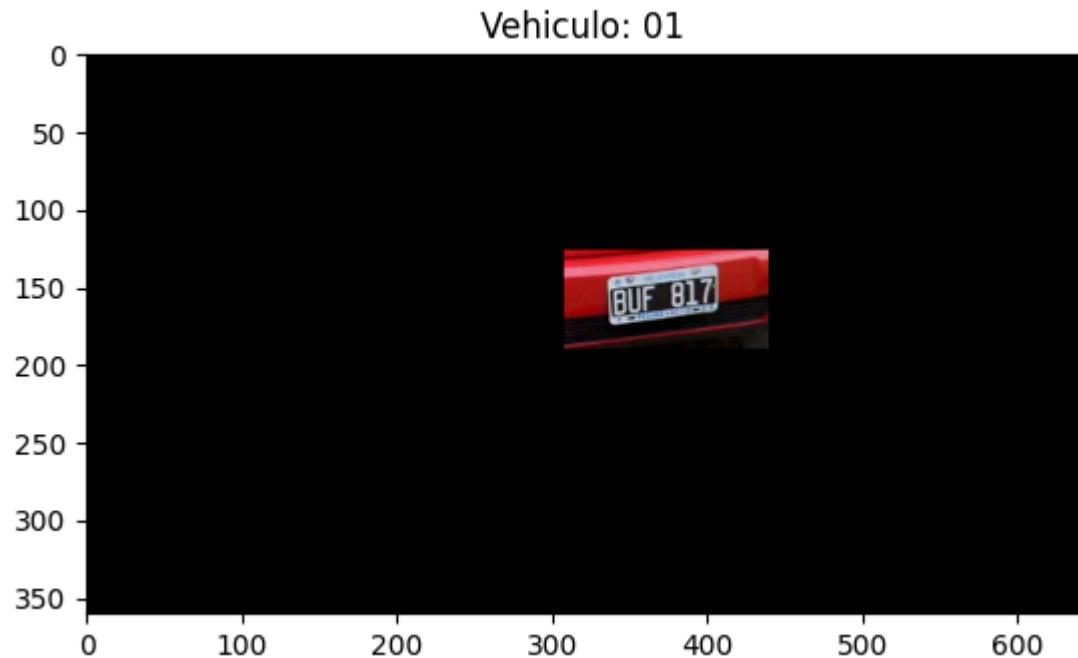
Para este ejemplo, el resultado fue el siguiente:

Elemento resultante	Boundingbox del elemento resultante
	

Para mayor seguridad, decidimos dilatar un poco el boundingbox detectado:

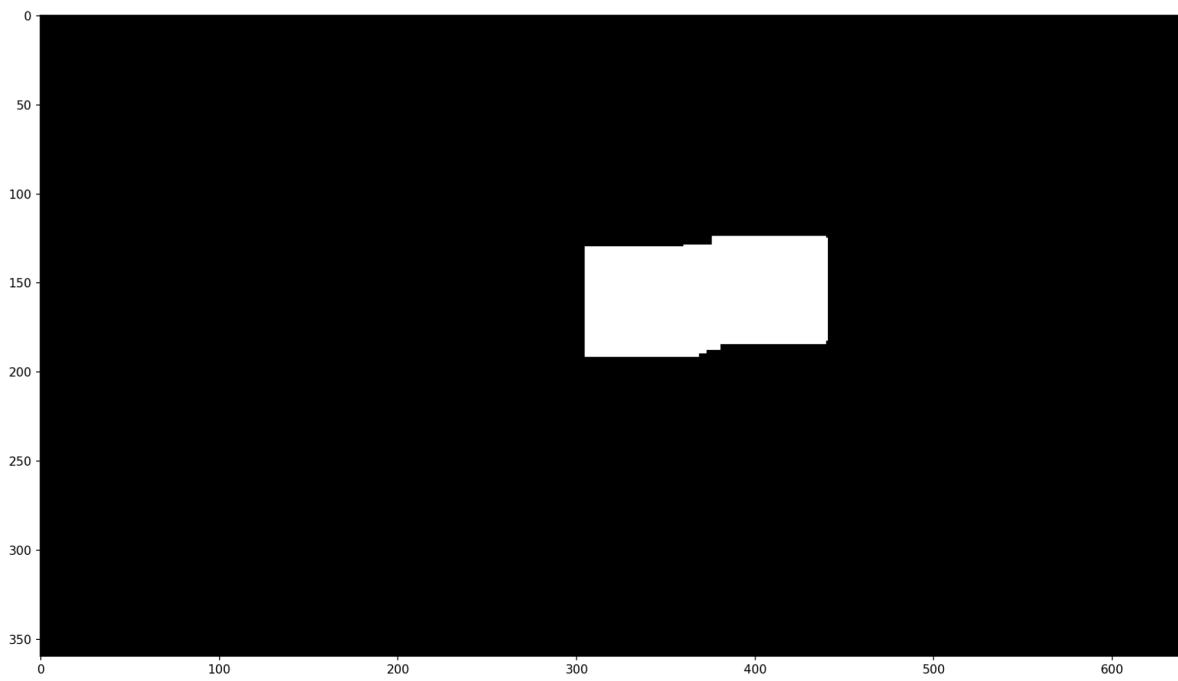


Finalmente realizamos el recorte sobre la imagen original:



Muchas patentes presentaban perfiles laterales (izquierdo o derecho) en las fotografías. Para mejorar la visibilidad y permitir una segmentación correcta, se procedió a enderezar los recortes de dichas imágenes.

Para ello dilatamos un poco el elemento detectado



Utilizamos la función approxPolyDP con el objetivo de aproximar un polígono de cuatro lados. Esta aproximación fue un paso previo necesario para aplicar una homografía y lograr la "rectificación" del recorte.



En algunos casos ayudó más que en otros, y por estar en una instancia de aprendizaje consideramos dejarlo pero realmente no parece ser un paso que aporte una gran ayuda a la posterior segmentación.

Finalmente, se logra un recorte de la patente que sirve como input del paso posterior (segmentación de caracteres):





## Segmentación de caracteres

Partimos de la misma premisa que en el paso anterior: la patente se caracteriza por caracteres blancos sobre un fondo negro, lo que genera un alto contraste.

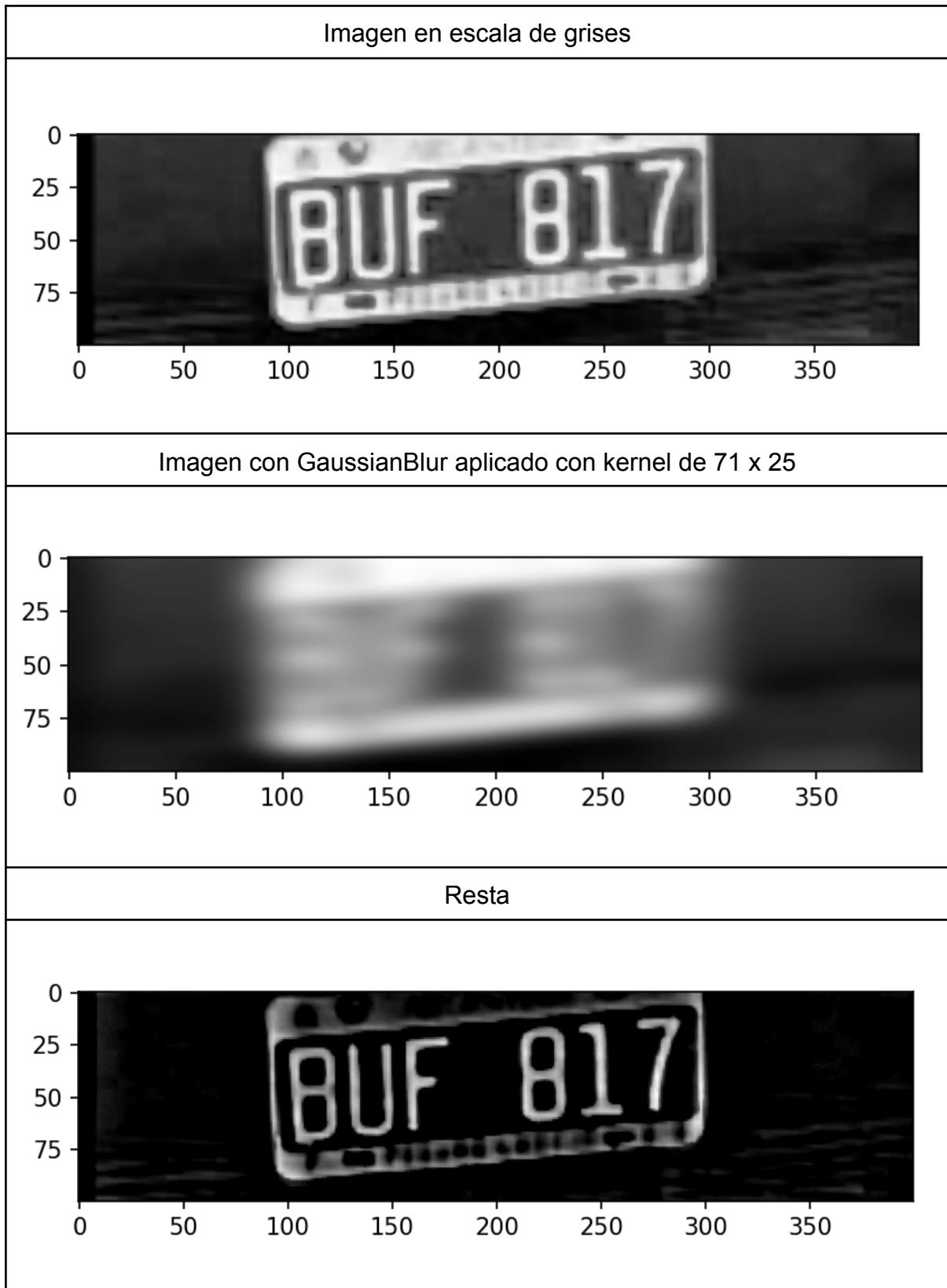
Inicialmente, la aplicación directa de un umbral (thresholding) a la imagen no ofreció resultados satisfactorios, ya que no se logró un umbral que generalizara correctamente todas las situaciones. Por lo tanto, exploramos métodos alternativos para acentuar el contraste. Intentamos con la ecualización de histograma, tanto global como local, pero sin obtener los resultados esperados.

Finalmente, la solución más efectiva se encontró al trabajar con el canal V de la imagen. Tras diversas pruebas, se optó por aplicar una ecualización de histograma específicamente sobre este canal.

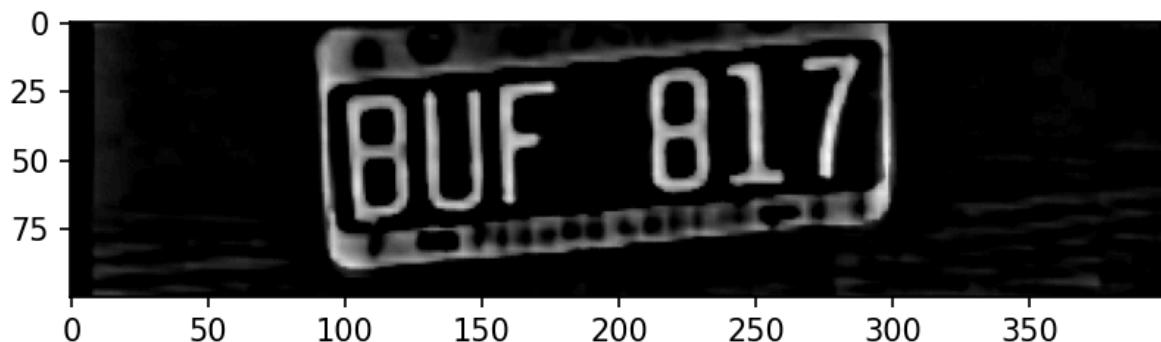
El resultado de esta técnica se presenta a continuación:



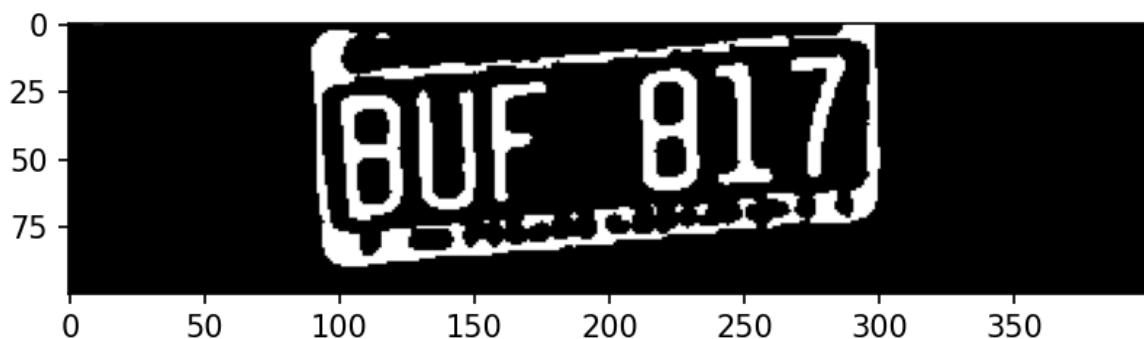
Luego pasamos la imagen a escala de grises y aplicamos un suavizado grande con GaussianBlur. Como la idea es resaltar los elementos blancos que contrastan fuertemente con el fondo, realizamos una resta entre la imagen en escala de grises y la imagen con el suavizado :



Probamos diversas operaciones para intentar resaltar aún más las partes blancas de la patente, y la que ofreció una ligera mejora fue Top-Hat. Aunque el cambio no resulta evidente en este ejemplo específico, en otras instancias ayudó a destacar áreas pequeñas que de otra forma permanecían más grisáceas.



Luego, consideramos que ya estábamos en condiciones de aplicar el umbral. Experimentalmente, determinamos que un umbral bajo de 40 resultaba ser el más óptimo, ya que contribuía significativamente en patentes donde las áreas blancas eran reducidas y predominaban los tonos gris claro.



A continuación, se detallan los filtros aplicados sobre los resultados de `connectedComponentsWithStats` para aislar los elementos con una morfología similar a la de un carácter de patente:

1. **Relación de aspecto (ratio):** La razón entre la altura y el ancho del *bounding box* debía estar en el rango de 1 a 4. Esto prioriza elementos más altos que anchos, incluyendo también los cuadrados.
2. **Ancho:** El ancho del *bounding box* debía representar entre el 5% y el 25% del ancho total de la imagen.
3. **Altura:** La altura del *bounding box* debía situarse entre el 20% y el 66% (dos tercios) de la altura total de la imagen.
4. **Área:** El área del *bounding box* debía estar comprendida entre el 1% y el

10% del área total de la imagen.

Además, a medida que se detectaban nuevos elementos que cumplían con la condición establecida, se comparaban con los ya identificados. Si un elemento recién detectado se superponía significativamente con uno existente, particularmente en el eje horizontal, se descartaba la adición a la lista de elementos detectados, bajo el criterio de que no representaba un nuevo elemento independiente.

Por otro lado, solo se agregaría un nuevo carácter detectado si la cantidad total de elementos no superaba los 6. Esta restricción se implementó con un propósito estético: en casos donde el algoritmo pudiera fallar, se buscaba evitar la representación gráfica de más de 6 elementos detectados en la imagen resultante.

Finalmente el resultado es el siguiente:



## Resultados integrales

Una vez completadas las etapas de segmentación de patentes y de caracteres, se incorporó un visualizador de resultados final que ofrece una comparación integral de las fases del procesamiento.

Imagen: 01



Es importante destacar, y apartándonos del ejemplo guía de este informe, que el algoritmo desarrollado no alcanza una detección del 100% de las patentes y sus caracteres.

#### Visualización final

La visualización final arroja las 12 imágenes siguientes. de las cuales se observa:

Imagen: 01

Vehículo



Patente



Caracteres detectados



Patente: OK  
Caracteres: OK

Imagen: 02

Vehículo



Patente



Caracteres detectados



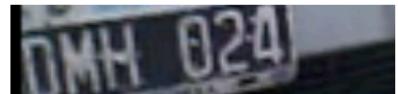
Patente: OK  
Caracteres: OK

Imagen: 03

Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: MAL (4 de 6)

Imagen: 04

Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: OK

Imagen: 05

Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: OK

Imagen: 06

Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: OK

Imagen: 07

Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: OK

Imagen: 08

Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: MAL (5/6)

Imagen: 09

Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: OK

Imagen: 10

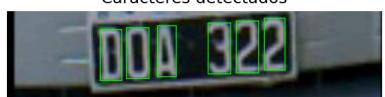
Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: OK

Imagen: 11

Vehículo



Patente



Caracteres detectados



Patente: OK

Caracteres: OK

Imagen: 12

Vehículo



Patente

Patente  
NO detectada

Caracteres detectados

Sin elementos

Patente: MAL

Caracteres: MAL (0/6)

Cuadro resumen de estadísticas

Vehículo	Detección de patente	Segmentación de todos los caracteres sobre la patente	Cantidad de caracteres detectados	Porcentaje de caracteres detectados
1	OK	OK	6/6	100%
2	OK	OK	6/6	100%
3	OK	DETECCIÓN INCOMPLETA	4/6	66,66%
4	OK	OK	6/6	100%
5	OK	OK	6/6	100%
6	OK	OK	6/6	100%
7	OK	OK	6/6	100%
8	OK	DETECCIÓN INCOMPLETA	5/6	83,33%
9	OK	OK	6/6	100%
10	OK	OK	6/6	100%
11	OK	OK	6/6	100%
12	NO DETECTADA	DETECCIÓN INCOMPLETA	0/6	0%
TOTAL	11/12	9/12	63/72	N/A
	91,66%	75%	87,5%	

# Conclusión

A lo largo de este trabajo aplicamos distintos conceptos y técnicas de procesamiento de imágenes para abordar dos problemas de naturaleza muy diferente.

En el primer problema nos enfrentamos al desafío de separar diversos elementos sobre un fondo no uniforme para poder analizarlos individualmente. Una vez detectados, fue necesario clasificarlos en dos categorías (dados y monedas) y luego aplicar un procesamiento específico a cada una. En el caso de las monedas, debíamos reconocer sus distintos tipos, mientras que para los dados era necesario identificar el valor de la cara superior. Para ello trabajamos con filtros, operaciones morfológicas, detección de bordes, contornos y técnicas de segmentación. Al operar sobre una única imagen, la experimentación con parámetros y combinaciones de herramientas resultó relativamente acotada y manejable.

El segundo problema representó el mayor desafío del trabajo: el reconocimiento de patentes y la segmentación de sus caracteres. El primer obstáculo fue que esta vez no contábamos con una sola imagen, sino con un conjunto de doce, lo que exigía que el algoritmo generalizara correctamente frente a distintos contextos. El segundo obstáculo fue la naturaleza real de las imágenes: variaciones de color en los vehículos, fondos cambiantes, caracteres deteriorados o poco visibles, iluminación desigual, entre otros factores. Lograr el nivel de resultado alcanzado no fue sencillo. Requirió mucha experimentación con herramientas, parámetros y secuencias de procesamiento para determinar qué usar y en qué etapa. A pesar de momentos frustrantes (propios de no obtener de inmediato el resultado esperado) la experiencia resultó enriquecedora, y aunque nos quedamos con las ganas de seguir experimentando hasta alcanzar una efectividad ideal, consideramos que el algoritmo funcionó muy bien y cumplió con los objetivos planteados.

En comparación, concluimos que el segundo problema presentó una complejidad significativamente mayor que el primero, tanto por la variabilidad de las imágenes como por la necesidad de un proceso más robusto, adaptable y generalizable.

En definitiva, este trabajo nos permitió comprender de manera práctica las dificultades reales del procesamiento de imágenes y el valor de la experimentación constante. Más allá de los resultados obtenidos, la experiencia fortaleció nuestro conocimiento teórico de algunos conceptos y herramientas sumado a la capacidad para diseñar, ajustar y evaluar soluciones, y dejó en claro que este tipo de desafíos requieren tanto conocimiento teórico y técnico como perseverancia y criterio para tomar decisiones en cada etapa del proceso.