TECNICATURA UNIVERSITARIA EN INTELIGENCIA ARTIFICIAL (TUIA)

PROCESAMIENTO DE IMÁGENES (PDI)

Docentes de la cátedra:

Dr. Ing. Gonzalo Sad

Ing. Facundo Reyes

Ing. Julian Álvarez

Trabajo Práctico 2

Estudiantes (GRUPO 15):

Max Eder

Martín L. Perrone Leone

Pablo Pistarelli

AÑO 2023 (2do CUATRIMESTRE), Rosario.

INTRODUCCIÓN

En el siguiente informe presentamos el proceso de trabajo realizado para el TP2.

Al inicio abordamos el problema 1, en el cual se requiere segmentemos entre monedas y dados, que clasifiquemos a las primeras por su tamaño y que determinemos el número que presentan los segundos.

Luego tratamos el problema 2, en el que se nos requiere implementar un algoritmo que detecte automáticamente patentes y las segmente, y además de cada una de ellas que se segmenten los caracteres.

PROBLEMA 1 – Detección y clasificación de Monedas y Dados

Lo primero que hicimos fue pasar la imagen, que estaba en colores, a escala de grises ya que como vimos en la teoría, tiene varias ventajas por utilizar un solo canal en lo que respecta a la simplicidad computacional y al contenido estructural.

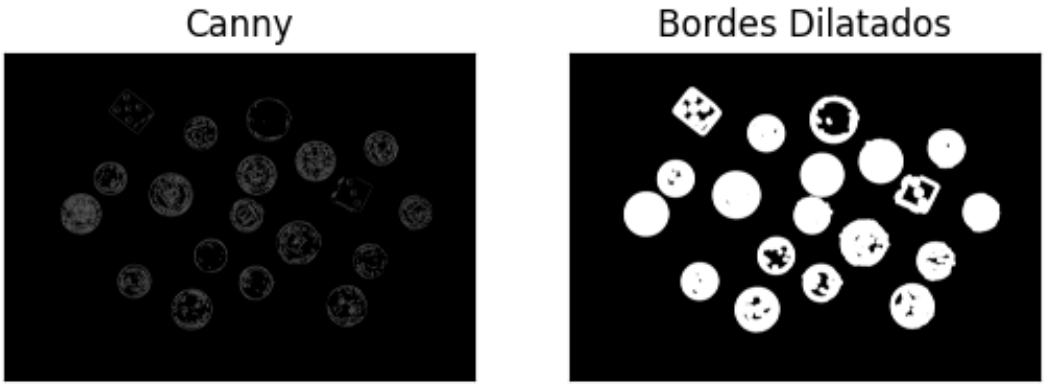
Luego le aplicamos un suavizado gaussiano para reducir el ruido, eliminar detalles finos, suavizar los bordes y preparar la imagen para las siguientes operaciones de procesamiento.

El resultado luego de estos dos pasos es el siguiente:

El siguiente paso fue detectar los bordes para lo que decidimos utilizar Canny en lugar del umbralado simple ya que el primero ofrece detección de bordes más precisa, robusta frente al ruido y con mejor definición y además, al ser una única imagen y pequeña no se requiere de grandes recursos computacionales. Luego de hacer algunas pruebas en las que o los bordes no se detectaban o aparecía mucho ruido, encontramos el equilibrio en threshold1=20 y threshold2=150.

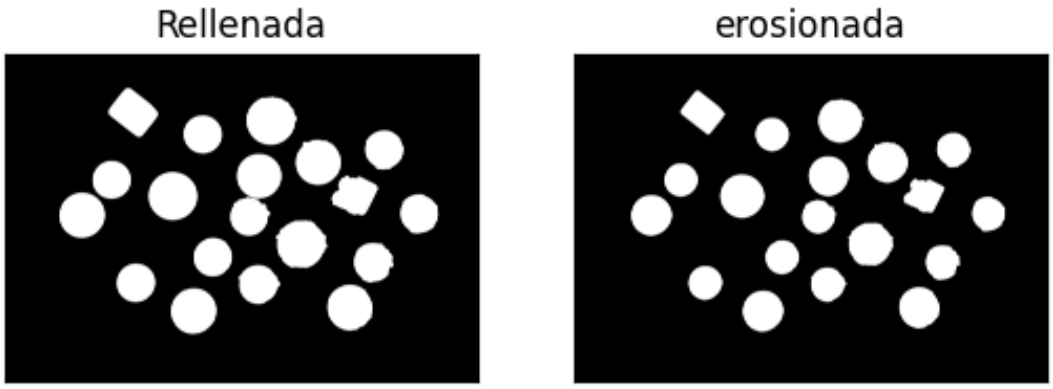
A esta imagen le aplicamos una dilatación utilizando un elemento estructural con forma de elipse de (40, 40) luego de probar varias veces y nuevamente encontrando un equilibrio, en este caso para que deje el borde bien definido pero no se terminen uniendo con otras figuras cercanas.

Las imágenes obtenidas en cada caso fueron las siguientes:

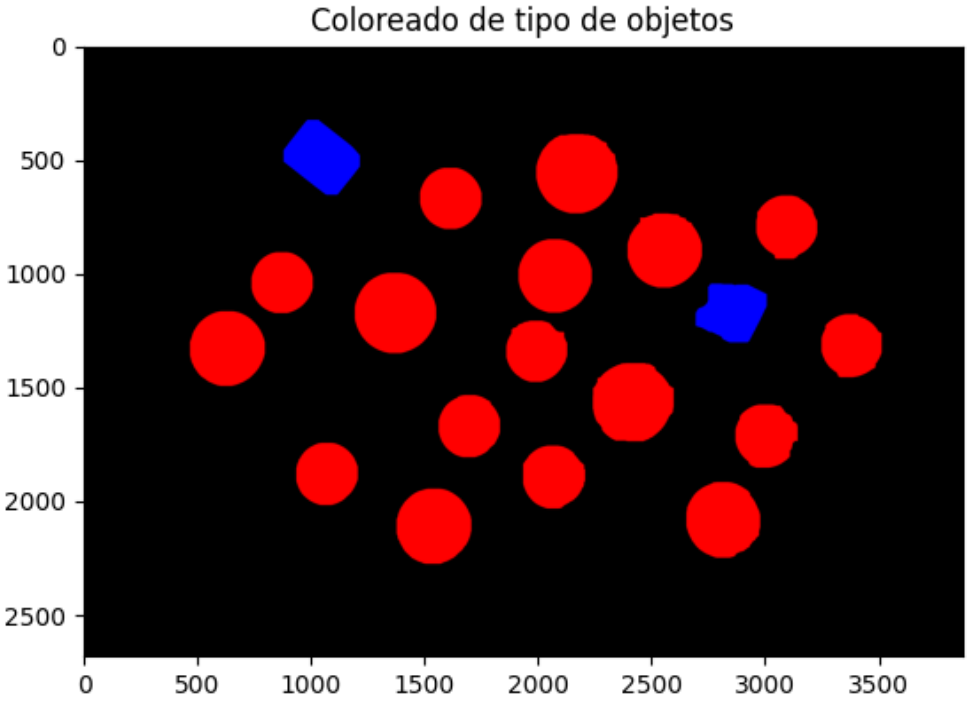


Una vez obtenido esto, procedimos a rellenar los objetos con una función, en la cual definimos al marcador como el complemento de los bordes, a la máscara como el complemento de la imagen, reconstruimos y por último nos queda la imagen con los huecos rellenos igual al complemento de la reconstrucción. Esta función fue brindada por el profesor en clases.

A la imagen rellena le aplicamos la operación morfológica de erosión con un kernel de 10x10 y por 4 iteraciones para eliminar detalles pequeños y suavizar los bores. También probamos con varios parámetros hasta encontrar el equilibrio en que no se deformaban o carcomían mucho los objetos y además las deformidades desaparecían.
Las imágenes obtenidas son las siguientes:



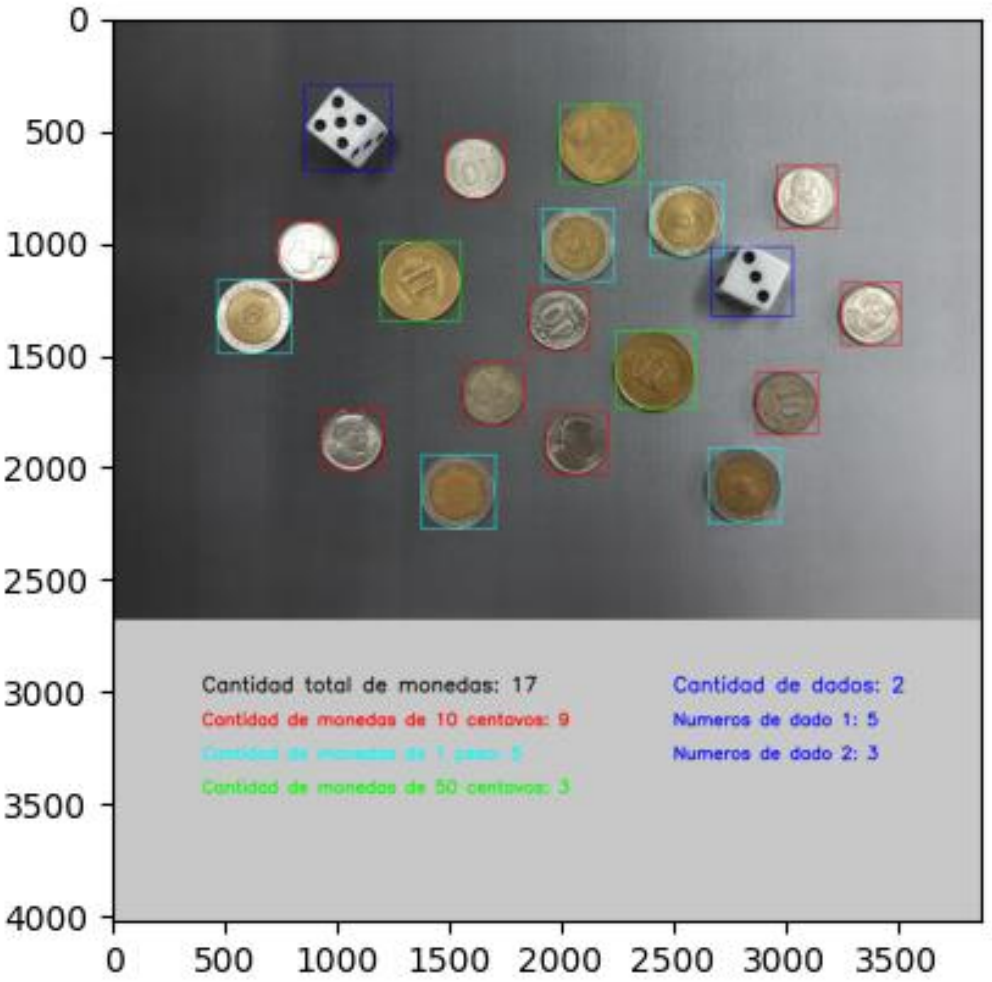
Luego aplicamos la operación morfológica de apertura para suavizar el contorno con un elemento estructural de 50x50 y con forma rectangular. De esta imagen obtenemos los componentes conectados y por cada uno de ellos aplicamos el factor de forma y la función provista por los profesores para detectar objetos circulares y no circulares. En la siguiente imagen pasada nuevamente a color, se muestra el resultado pintando de color azul a los no circulares y de color rojo a los circulares:



El paso siguiente fue recorrer todos los objetos y trabajarlos:

a) En caso de que sean circulares (monedas) determinamos el tipo de moneda según el área del objeto y de acuerdo al tipo las pintamos de un color (verde, rojo o azul). Estos valores para la clasificación fueron determinados mediante pruebas.
b) En caso de que no sean circulares asumimos que es un dado y lo aislamos de la imagen utilizando una máscara rectangular, recortando su región y volviendo a convertirla en escala de grises para continuar trabajándola. Detectamos y contamos los puntos en cada dado mediante el cálculo del Rho para asegurarnos que sean los de la cara superior donde se ven más circulares aplicándole previamente dilatación y componentes conectados para identificar los distintos puntos.

Por último, generamos la imagen final con las de las monedas, las de los dados y los textos con todo lo obtenido dentro del for en el que se recorren los distintos objetos. La imagen definitiva es la siguiente:

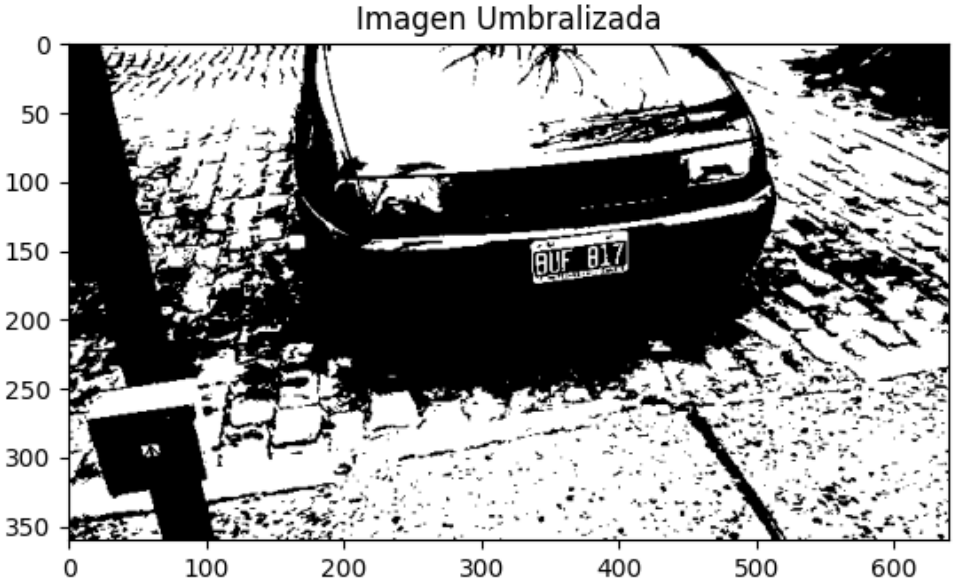


PROBLEMA 2 – Detección de patentes

Para hacer la primera aproximación del código trabajamos con la imagen img01.png. Nuevamente aquí lo primero que hicimos fue pasar la imagen a escala de grises.

Luego la convertimos en binaria aplicando cv2.threshold() con el parámetro de umbralado inferior en el valor de la mediana – 5 por pruebas que fuimos haciendo y fue el valor que nos servía para la mayor cantidad de imágenes distintas.

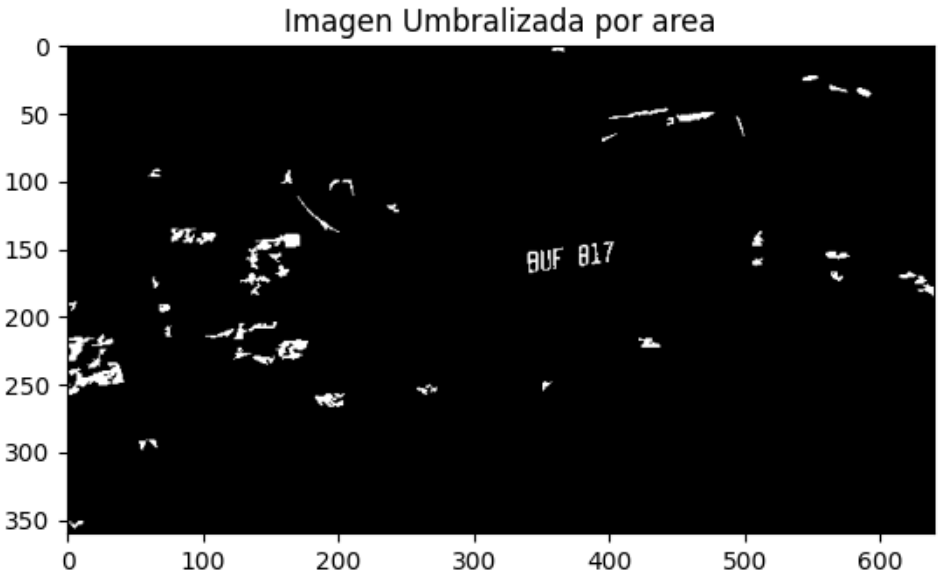
La imagen obtenida es la siguiente:



A esta le aplicamos cv2.connectedComponentsWithStats() para segmentar la imagen etiquetando los componentes conectados y proporcionando información estadística sobre estos componentes para luego trabajar los que sean de nuestro interés.

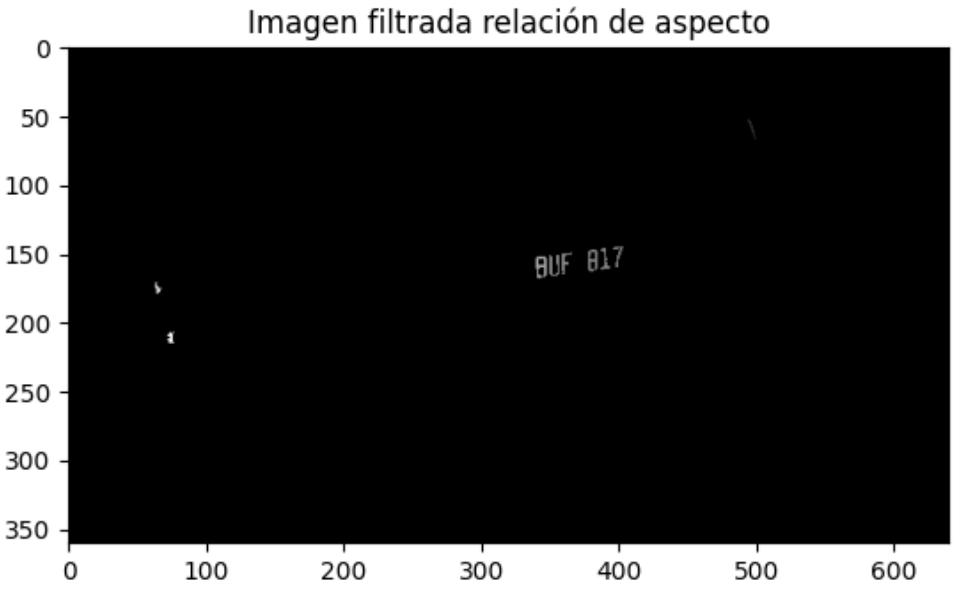
Creamos una máscara que conserva solo los componentes conectados cuyas áreas están dentro del rango definido entre 17 y 500 según las pruebas que fuimos haciendo y eliminando los componentes que no cumplen con estos criterios de área. Utilizamos esta máscara para filtrar y retener los componentes de interés en la imagen binaria.

La imagen obtenida es la siguiente:

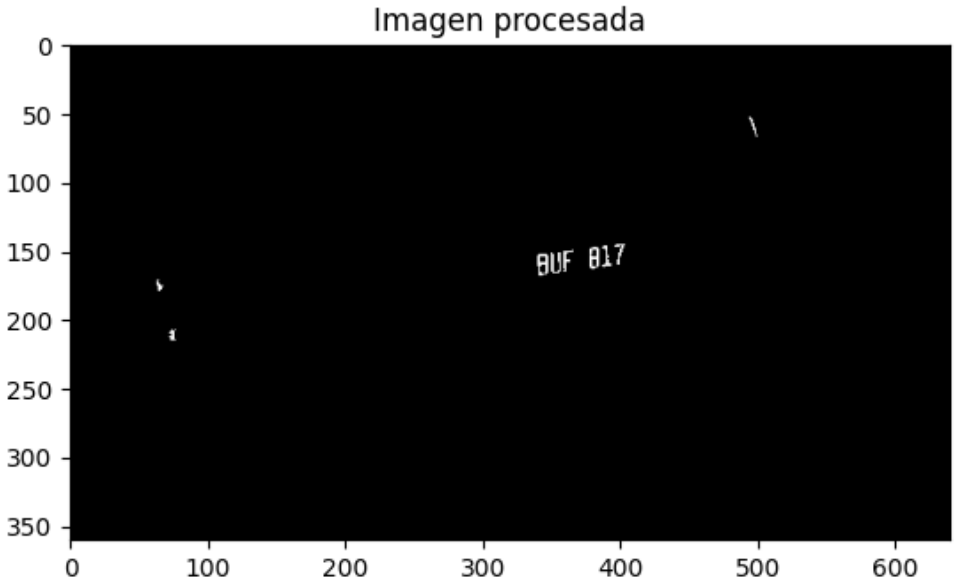


Volvemos a filtrar los componentes conectados, pero esta vez basándonos en su relación de aspecto, eliminando aquellos cuya relación de aspecto está fuera del rango especificado (menor a 1.5 o mayor a 3). Si la relación de aspecto de un componente no cumple con el rango

deseado, se asigna el valor 0 a la etiqueta correspondiente en la imagen labels_aspect_ratio_filtered, eliminándolos de la imagen. El resultado es el siguiente:



Volvemos a aplicar el umbralado para convertir la imagen a una imagen binaria, donde los píxeles que cumplen con el umbral se establecen en blanco (255) y el resto en negro (0), para segmentar y resaltar áreas específicas de interés en la imagen.

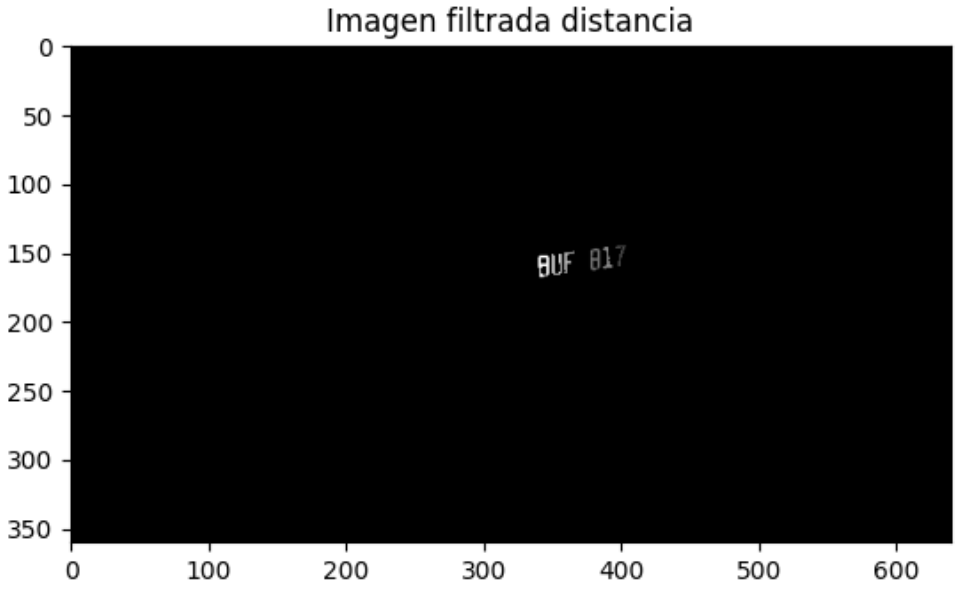


Como al hacer este procesamiento perdimos información, volvemos a ejecutar cv2.connectedComponentsWithStats() para trabajar la imagen.

Iteramos sobre cada componente y por cada uno, volvemos a iterar para calcular las distancias en las coordenadas *x* e *y* entre cada par de componentes.

Se calcula la distancia en coordenada *y* la distancia en coordenada *x* entre los componentes. Si la distancia *x* está entre 5 y 50 y si la distancia *y* es menor a 10 lo tomamos como cercano y sumamos a un contador para luego, si este contador es menor a 2 (hay pocos componentes cercanos y por lo tanto no son parte de la

patente), establecemos el valor de este en 0 para eliminarlos de la imagen para obtener la siguiente:



Pasamos la imagen a uint8 y volvemos a ejecutar cv2.connectedComponentsWithStats() y ordenamos los componentes según su coordenada x inicial.

Convertimos la imagen original de un espacio de color BGR a RGB utilizando cv2.cvtColor() para luego aplicar una máscara basándonos en las coordenadas de estos componentes conectados para recuadrar la zona de la patente.

Con esto cumplimos con el ítem a) del ejercicio.

Para el item b) recorremos los componentes de izquierda a derecha (ordenados por su coordenada de inicio *x*) y con los datos de posición *x*, posición *y*, ancho y altura dibujamos rectángulos delimitadores alrededor de cada uno de ellos en la imagen original:

Conclusión: En este proceso nos hemos dado cuenta de que debemos trabajar mucho con las pruebas de los parámetros en el trabajo de la imagen y en las operaciones morfológicas para lograr identificar y segmentar los objetos de manera correcta. Lo interesante de esto es que con las herramientas que se nos brindaron podemos lograr algo que antes de conocerlas nos parecía imposible.