Proyecto final procesamiento de imágenes y visión:

Detección de placas

Carolina Mercado
Mateo Gómez
David Calles

4 de Diciembre de 2020

Pontificia Universidad Javeriana de Bogotá

1. Introducción y problema

La detección de placas particulares procedente de imágenes de cámaras de transito tiene una problemática muy común en el área de visión artificial ya que estas tienen particularidades que no permiten la extracción óptima de características, así mismo, los obstáculos y el ruido generado por el método de captura del repertorio y su ubicación en un ambiente no controlado genera interferencias en el procesamiento de los datos para su análisis.

Por esta razón, se plantea como solución a la problemática y proyecto final de la asignatura *Procesamiento de imágenes y visión* de la Pontificia Universidad Javeriana, un algoritmo que identifique el área de interés (ROI por sus siglas en inglés *Region of interest*) es decir, las placas de automóviles particulares colombianos (Color amarillo de 3 letras y 3 números) y así con ayuda de librerías en *Python* y *OpenCV* realizar un reconocimiento de caracteres a partir del OCR *Tesseract*.

A continuación se documenta el desarrollo del mismo, así justificando su planteamiento y dando a conocer las conclusiones y resultados del experimento.

2. Diagrama de flujo de la solución

La figura 1 muestra el diagrama de flujo de la solución.

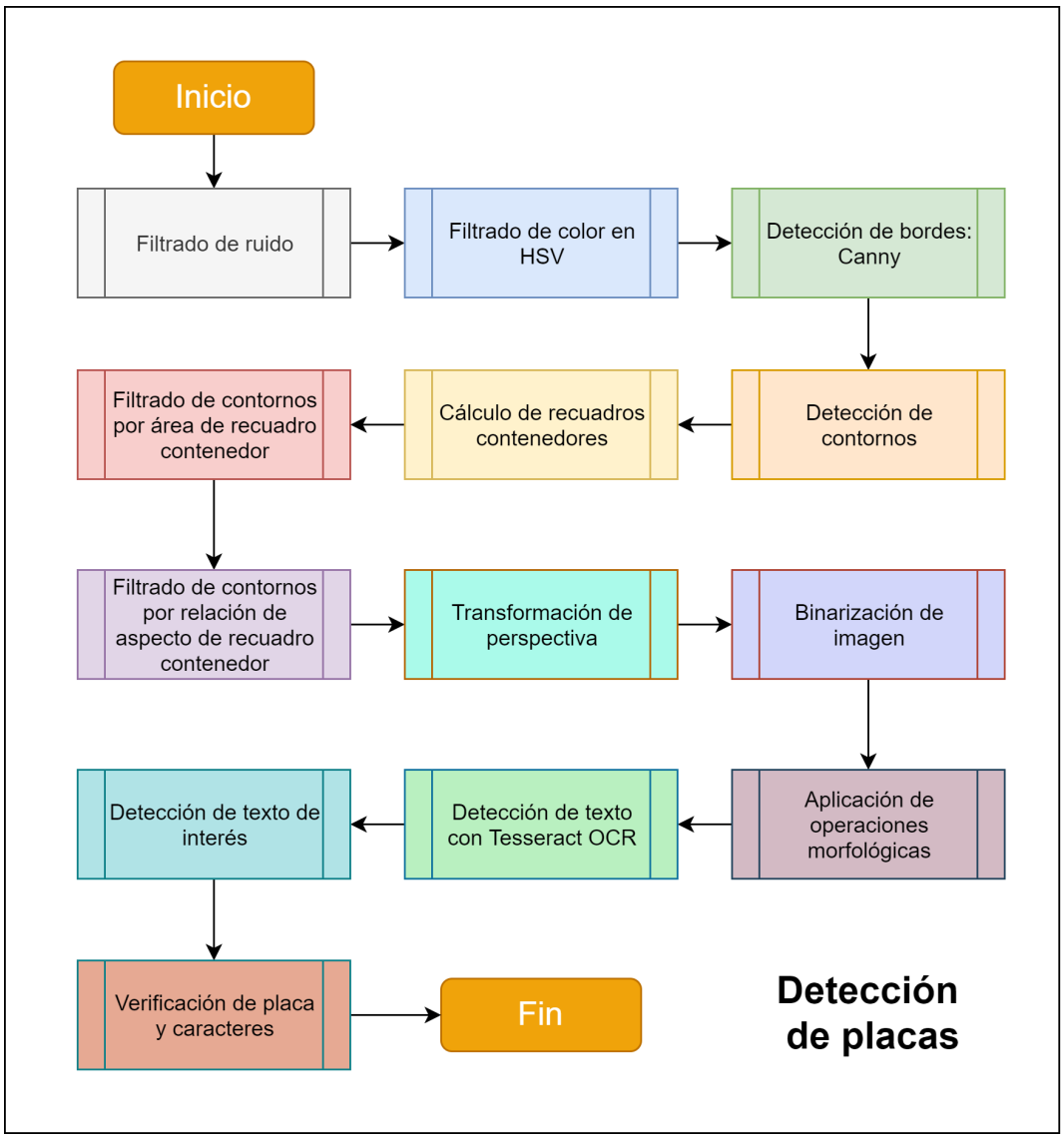

Figura 1: Diagrama de flujo con macro-bloques de la solución.

3. Desarrollo de la solución

Para la solución se plantea el diagrama de flujo con macro-bloques como un desarrollo secuencial, definiendo el paso a paso que sigue la imagen en el algoritmo para obtener el área de interés y su respectiva lectura de caracteres, así cada imagen pasa por el mismo proceso para obtener la información que se encuentra en ella.

1. **Filtrado de ruido:** Una vez importada la imagen (Figura 2) al algoritmo, se procede a la etapa de filtrado de ruido que tiene la imagen por el método de captura y otras particularidades en si propias de esta. Para ello se utiliza un filtro bilateral que utiliza un kernel adaptativo a la imagen para su limpieza según la intensidad de cada píxel de la imagen y el peso que le asigne, como muestra la figura 3. [1]

2. **Filtrado de color en HSV:** Se continua al bloque de filtrado de color en HSV, que según los valores en *Hue* (Tono), *Value* (Valor) y *Saturation* (Saturación) se permite el paso del color amarillo. Para esto se hizo una calibración manual de los valores para encontrar el rango de este color y poder limitar el paso de colores cercanos a este como lo son el rojo y el naranja. Se puede apreciar en a figura 4 el filtrado de la placa en comparación con los demás componentes de la imagen.[2]

3. **Detección de bordes Canny:** Una vez filtrada por color, se realiza un análisis de gradiente dentro de la imagen para detectar los bordes y así aislar los objetos del fondo del color que el filtro permitió pasar. Para ello se utiliza el algoritmo de Canny, que permite localizar los bordes de manera eficiente y continuar con la detección de otras características de la imagen. Como se muestra en la figura 5, se aprecian los bordes detectados en la imagen de prueba. [3]

4. **Detección de contornos:** Se prosiguió con la detección de contornos, que permite definir una secuencia de puntos sin espacios ni huecos entre ellos y así determinar que bordes son contornos cerrados y cuales no. Para garantizar que contornos se han encontrado se dibujan sobre la imagen original y se tiene una estimación de las zonas a las cuales considera como objetos definidos, como muestra la figura 6.[4]

5. **Calculo de recuadros Contenedores:** Con los contornos obtenidos en el proceso anterior, se les calcula los cuadros contenedores, es decir, los rectángulos que estos contornos pueden representar (Que pueden encontrarse rotados), como se muestra en la figura 7.

6. **Filtrado de contornos por área de recuadro contenedor:** El motivo del anterior proceso es filtrar por el área contenedora el tamaño de la placa como se muestra en la figura 8.

7. **Filtrado de contornos por relación de aspecto de recuadro contenedor:** Por medio de la relación 2:1 del ancho y alto de la placa se filtra a los posibles candidatos. Por ejemplo, una placa comúnmente tiene tamaño de 66cm de ancho por 33cm de largo (Relación 2:1), así que los rectángulos que cumplan la condición de esta relación son aquellos que pasan el filtrado. Así se puede visualizar en la figura 9.[5] [6]

8. **Transformación de perspectiva:** Con las coordenadas de las esquinas de los rectángulos se realiza una transformación de perspectiva, que permite en la imagen original obtener el rectángulo con mayor área donde se estima esté la placa ubicada, como muestra la figura 7. Para este punto, se ha detectado la placa.[7]

9. **Binarización de imagen:** Finalmente, para hacer el análisis de la imagen, se realiza el pre-procesamiento del área de interés con la binarización en un umbral definido de forma manual (Figura 11).

10. **Aplicación de operaciones morfológicas:** Se le aplica una operación morfológica de erosión para mejorar las letras y números visibles en el recuadro, como se muestra en la figura 12.[8].

11. **Detección de texto en Tesseract OCR:** Así utilizando la librería de Tesseract, se realiza la detección de texto con el OCR y ya que la imagen puede contener más texto del esperado, por ejemplo, la ciudad en la que se registra el vehículo, se filtra por la particularidad que tienen las placas colombianas de 3 caracteres seguidos de 3 números. [9]

12. **Verificación de placa y caracteres:** Por último, se programa el algoritmo para que permita al usuario calcular el error, así como los valores de las placas detectadas o si no detecto alguna, y la precisión de detección de caracteres.

4. Resultados

Luego de probar el algoritmo con una base de 122 placas, se obtienen los siguientes resultados:

La localización de la placa dentro de las imágenes originales se hace de manera exitosa, obteniendo un 94 % de exactitud, ya que al visualizar de forma manual la imagen procesada por los filtros, entre las 122 ingresadas al algoritmo, se obtienen 115 recuadros de placa con sus respectivas coordenadas.

Dentro del 94 % solo se detectan caracteres del tipo 3 letras 3 números en 49 imágenes, siendo así el 40 % de las placas detectadas por su localización. Este proceso se realiza ingresando los recuadros de las placas en el OCR Tesseract y con el patrón esperado se indica si se reconoció 3 caracteres seguido de 3 números o si en su defecto no detecta el texto esperado dentro de la imagen.

Individualmente los caracteres se detectan con una precisión del 82 %; esto se obtiene a partir de la selección de los caracteres de las placas en el nombre de los archivos y la posterior comparación con los resultados del OCR, es decir, en este caso de 294 caracteres (referente a 6 caracteres en 49 imágenes) se obtienen 243 correctos.

Sin embargo se consideran aspectos por mejorar ya que la cantidad de placas donde se detecta correctamente su valor es de 17 entre las 49 previamente detectadas, pues se aprecia que su patrón de 3 letras y 3 números corresponde con el valor real de la misma, mientras que, en las 32 placas restantes se tiene error en al menos 1 carácter dentro de la misma.

5. Conclusiones

- El resultado que se obtiene a partir de la medida de error y el set de imágenes permite concluir que la eficiencia del filtrado por color HSV así como el filtro de bordes y contornos es razonable con los alcances del curso y lo esperado para el proyecto, ya que presenta una segmentación asertiva del área de interés.

- A partir de la experimentación se puede concluir que los parámetros y valores en los umbrales dependen no solo de la aplicación sino de las imágenes a procesar, ya que su adquisición, digitalización y procesamiento afectan en los resultados que pasan por múltiples etapas.

- La extracción de características es muy importante para definir los criterios con los que se clasifican las imágenes y si estas son aptas para la analítica, por lo tanto, se debe visualizar

los resultados obtenidos en cada etapa para garantizar que es lo esperado y sea válido para las siguientes fases.

- Para utilizar los sistemas de OCR se debe hacer el procesamiento de la imagen que permita la lectura correcta del texto, ya que al ingresar figuras con característica indeseables se puede obtener error en los resultados, por ello, es esencial el tratamiento y acondicionamiento de la data al algoritmo.
- Hacer un algoritmo que se adapte al parámetro de entrada hace más practica la herramienta, ya que al caracterizar los valores con una sola imagen hacen inservible el mismo programa, por lo tanto que este se ajuste a la imagen que ingrese es una ventaja para este tipo de proyectos.

Referencias

[1] *Image Filtering*. dirección: `https://docs.opencv.org/master/d4/d86/group__imgproc__filter.html#ga9d7064d478c95d60003cf839430737ed`.

[2] *Color Space Conversions*. dirección: `https://docs.opencv.org/4.5.0/d8/d01/group__imgproc__color__conversions.html#ga397ae87e1288a81d2363b61574eb8cab`.

[3] *Feature Detection*. dirección: `https://docs.opencv.org/4.5.0/dd/d1a/group__imgproc__feature.html#ga2a671611e104c093843d7b7fc46d24af`.

[4] *Structural Analysis and Shape Descriptors*. dirección: `https://docs.opencv.org/4.5.0/d3/dc0/group__imgproc__shape.html#gadf1ad6a0b82947fa1fe3c3d497f260e0`.

[5] *Structural Analysis and Shape Descriptors*. dirección: `https://docs.opencv.org/master/d3/dc0/group__imgproc__shape.html#ga103fcbda2f540f3ef1c042d6a9b35ac7`.

[6] *Structural Analysis and Shape Descriptors*. dirección: `https://docs.opencv.org/4.5.0/d3/dc0/group__imgproc__shape.html#ga3d476a3417130ae5154aea421ca7ead9`.

[7] *Geometric Image Transformations*. dirección: `https://docs.opencv.org/4.5.0/da/d54/group__imgproc__transform.html#ga20f62aa3235d869c9956436c870893ae`.

[8] *Morphological Transformations*. dirección: `https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html`.

[9] *Pytesseract*. dirección: `https://pypi.org/project/pytesseract/`.

6. Anexos

Figura 2: Imagen Original a detectar.

Figura 3: Imagen con filtro bilateral.

Figura 4: Imagen con filtro HSV.

Figura 5: Imagen con Bordes Canny.

Figura 6: Imagen con todos los contornos.

Figura 7: Imagen con todos los recuadros contenedores.

Figura 8: Imagen con 5 recuadros contenedores de mayor área.

Figura 9: Imagen con relación 2:1.

Figura 10: Imagen con transformación de perspectiva.

Figura 11: Imagen Binarizada.

Figura 12: Imagen Erosionada.