Tecnicatura Universitaria en Inteligencia Artificial

Procesamiento de Imágenes

“TRABAJO PRÁCTICO N° 3”

Balverdi, Valentina - (B-6588/9)

Caballero, Franco - (C-7328/8)

Grimaldi, Damián - (G-5977/3)

1° Semestre - Año 2025

Problema - Detección de carril

Introducción al problema

El presente trabajo práctico aborda el problema de detección de carriles en videos capturados desde el interior de un automóvil en movimiento, utilizando técnicas de procesamiento digital de imágenes. El objetivo principal consiste en diseñar un algoritmo en Python que permita identificar de manera automática las líneas que delimitan el carril de circulación, a partir de la información visual extraída de los videos provistos (ruta_1.mp4 y ruta_2.mp4). Como resultado, se espera generar nuevas versiones de estos videos en las que las líneas detectadas se encuentren resaltadas visualmente en color azul.

Desarrollo y resolución

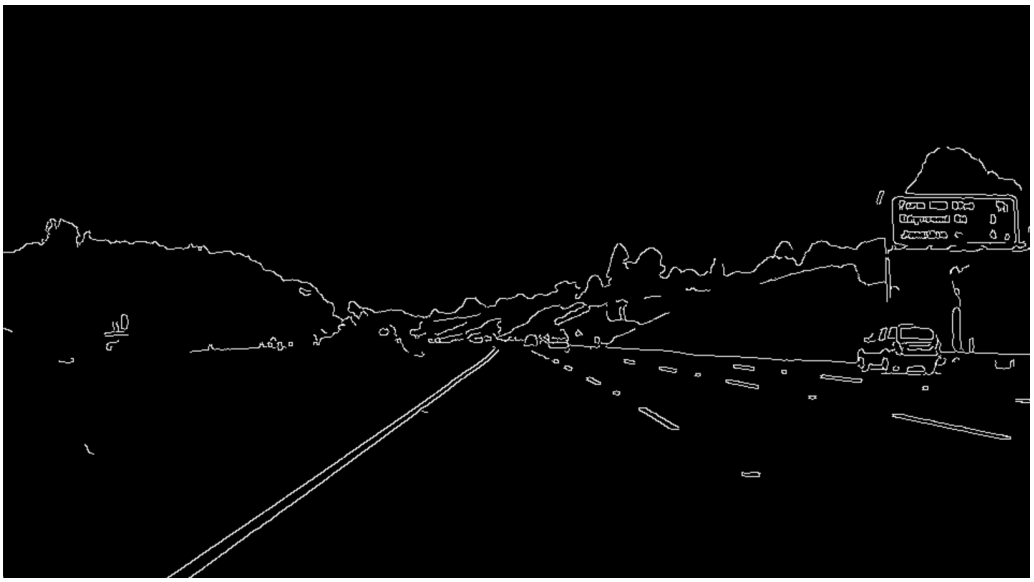
El procesamiento comienza con la apertura del video mediante OpenCV y la obtención de las dimensiones de cada cuadro, lo que resulta útil para definir regiones de interés (ROI) y establecer coordenadas relativas al tamaño de la imagen.

Cada frame se convierte a escala de grises para reducir la información redundante de color y enfocarse en la estructura de bordes, facilitando la detección de contornos. Esta imagen en escala de grises se suaviza aplicando un filtro *GaussianBlur*, el cual ayuda a reducir el ruido y evitar que pequeños detalles irrelevantes generen detecciones indeseables en etapas posteriores.

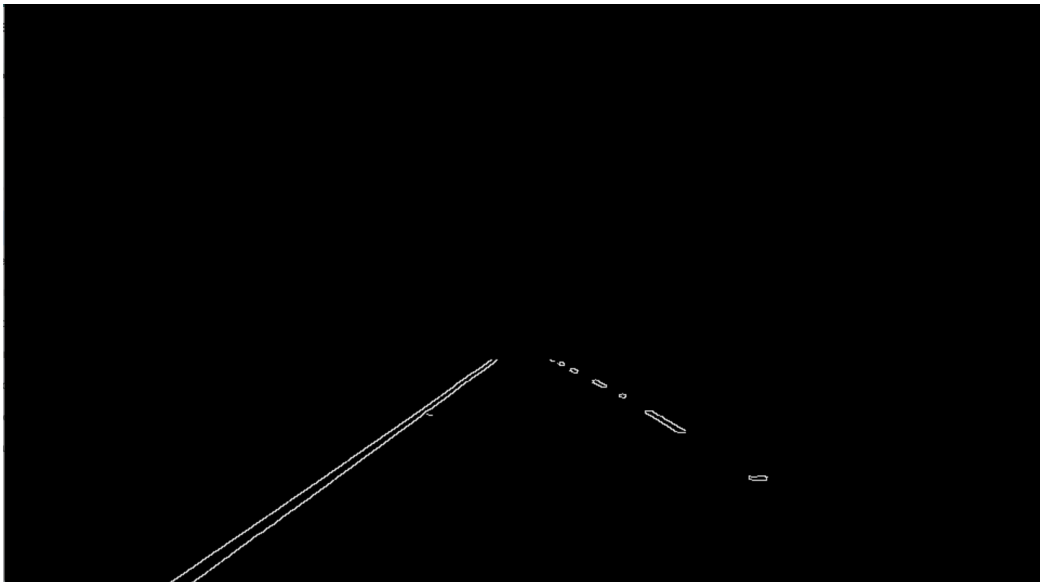
A continuación, se define una región de interés (ROI) con forma de trapecio. Esta ROI simula el área de la ruta más relevante para la detección de carriles, evitando que elementos fuera del camino (como árboles o vehículos en otros carriles) interfieran en el análisis. La ROI se implementa como una máscara binaria que delimita el área donde se aplicará el algoritmo de detección de bordes.

A continuación, la detección de bordes se realiza utilizando el algoritmo de Canny, que permite resaltar las transiciones bruscas de intensidad en la imagen, típicas de los límites del carril. La imagen resultante se combina con una máscara de región de interés (ROI) con forma de trapecio. Esta ROI simula el área de la ruta más relevante para la detección de carriles, evitando que elementos fuera del camino (como árboles

o vehículos en otros carriles) interfieran en el análisis. La ROI se implementa como una máscara binaria que delimita el área donde se aplicará el algoritmo de detección de bordes y se aplica mediante una operación *bitwise AND*, lo cual garantiza que solo se conserven los bordes ubicados dentro de la región de interés.



Detección de bordes con Canny.



Aplicación del ROI.

Uno de los principales desafíos encontrados fue el orden en que se aplicaba la ROI. Inicialmente, al aplicar la región de interés antes del algoritmo de Canny, se producía un efecto no deseado: el contorno del propio trapecio usado como máscara se detectaba como un borde válido. Esto interfería con la detección real de los bordes del carril. Para solucionarlo, se decidió invertir el orden de las operaciones, aplicando primero la detección de bordes con Canny sobre toda la imagen suavizada y luego

restringiendo el resultado mediante la máscara de ROI, lo cual evitó este efecto indeseado.

Con los bordes obtenidos, se aplica la transformada de Hough probabilística (*HoughLinesP*), que detecta líneas rectas en base a los segmentos de borde. A partir de las líneas detectadas, se filtran aquellas que tienen una pendiente cercana a cero (es decir, casi horizontales), ya que no representan carriles. También se descartan líneas verticales puras por no aportar información útil al análisis del carril.

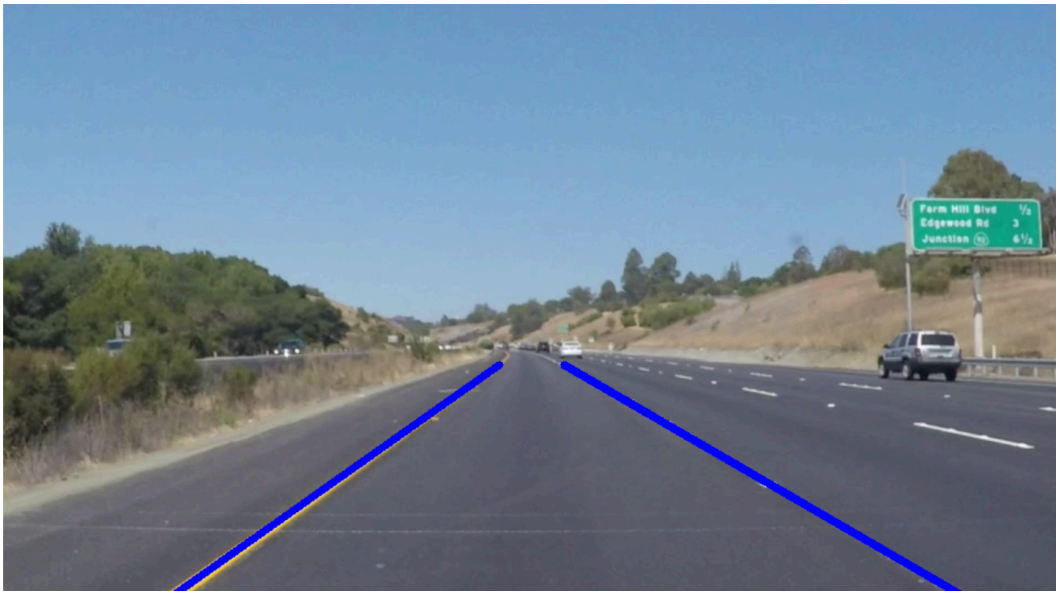
Las líneas restantes se clasifican como pertenecientes al lado izquierdo o derecho del carril, dependiendo del signo de su pendiente y de su posición horizontal respecto al centro del frame. Esto permite separar los dos bordes del carril que se busca representar.

Detección de líneas del lado izquierdo y derecho.

Dado que durante las pruebas, se observó que en algunos cuadros no se detectaban líneas en uno de los lados, el algoritmo conserva en memoria las líneas detectadas en

el frame anterior. Si en el nuevo cuadro no se detecta ninguna línea en un lado, se reutiliza la información previa, otorgando mayor estabilidad visual al resultado.

Finalmente, se llamó a la función **dibujar_linea_representativa()**, que calcula una línea promedio para cada lado del carril utilizando regresión lineal sobre los puntos extremos de las líneas detectadas. Estas líneas promedio se dibujaron sobre el frame original en color azul y con un grosor destacado, proporcionando una representación clara y continua del carril por el que circula el auto.



Resultado final con las líneas del carril detectadas y resaltadas en azul sobre el video original.

Conclusión

El desarrollo de este trabajo permitió poner en práctica diversas técnicas de procesamiento de imágenes con el objetivo de detectar las líneas que delimitan un carril de circulación en un entorno real. A lo largo del proceso, se lograron implementar etapas como la detección de bordes, la definición de una región de interés y la detección de líneas mediante la transformada de Hough, combinándolas de manera coherente para obtener un resultado funcional.

Durante la implementación surgieron algunos desafíos que requirieron especial atención, como la correcta definición del orden de operaciones para evitar la detección del borde del ROI, el ajuste de los parámetros del detector de bordes Canny y de los valores utilizados en la transformada de Hough para lograr líneas estables y representativas. La resolución de estos aspectos permitió mejorar la calidad del resultado y cumplir con el objetivo planteado.