TECNICATURA UNIVERSITARIA EN INTELIGENCIA ARTIFICIAL

Procesamiento de Imágenes

INFORME TP N°3

Estudiantes:

- **Domingo, Francisco**
- **Valeri, Lara**

Detección de carril

El objetivo del presente trabajo práctico es desarrollar un algoritmo para detectar automáticamente el carril por el cual está circulando un auto, mediante la detección de las dos líneas que lo delimitan. Luego, generar videos que muestran las líneas que definen el carril en color azul.

El trabajo consta de 4 archivos con extensión .py:

- **frame_ruta_1.py**: procesamiento de un frame del video ruta_1.mp4
- **ruta_1.py**: extensión del codigo de frame_ruta_1.py a todo el video
- **frame_ruta_2.py**: procesamiento de un frame del video ruta_2.mp4
- **ruta_2.py**: extensión del codigo de frame_ruta_2.py a todo el video

En primer lugar comenzamos con una versión sencilla, para la detección de líneas. Con el fin de analizar si al analizar el primer frame de la imagen obtenemos los mismos resultados al aplicarlo a todo el video y además para poner a prueba la creación del video con las líneas.

Consideramos que esta fue una primera aproximación, pero que el resultado obtenido no era el deseado. A continuación podemos ver una captura del video que se obtuvo al procesar el video ruta_1.mp4.

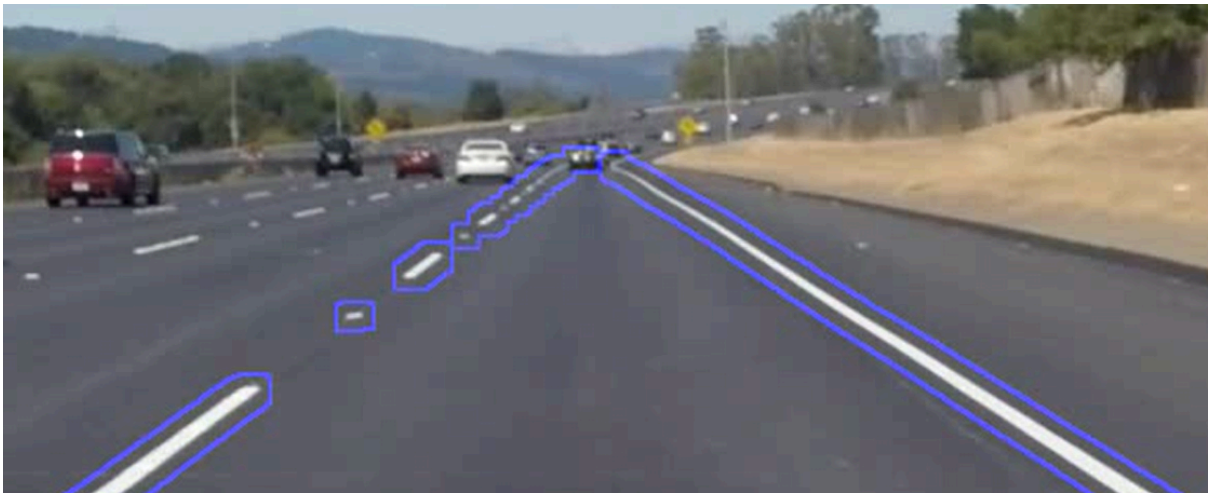

Fig 1: Imagen de primera aproximación

La detección de líneas en esta primera instancia se hizo con detección de puntos, a continuación una parte del código utilizado.

```
    w = -1 * np.ones((3, 3))
    w[1, 1] = 8
    fp = cv2.filter2D(f_mg, cv2.CV_64F, w)
    fpn = abs(fp)
    fpn = np.uint8(fpn)
```

En la Fig. 1 podemos observar que si bien se detectan las líneas quedaban cosas por mejorar, como por ejemplo, unir los segmentos de la línea de la izquierda, que las líneas azules se acercan más a la línea real y además, en algunos momentos del video, se encerraba al auto que estaba delante en azul.

Con esta primera versión nos enfocamos en analizar un solo frame del video (esto se puede ver en el archivo frame_ruta_1.py), utilizando la primera ayuda que había sido dada en el trabajo e intentando mejorarla en cada ejecución.

Utilizando la ayuda mencionada, logramos una gran aproximación al resultado deseado pero aún quedaban cosas por mejorar.

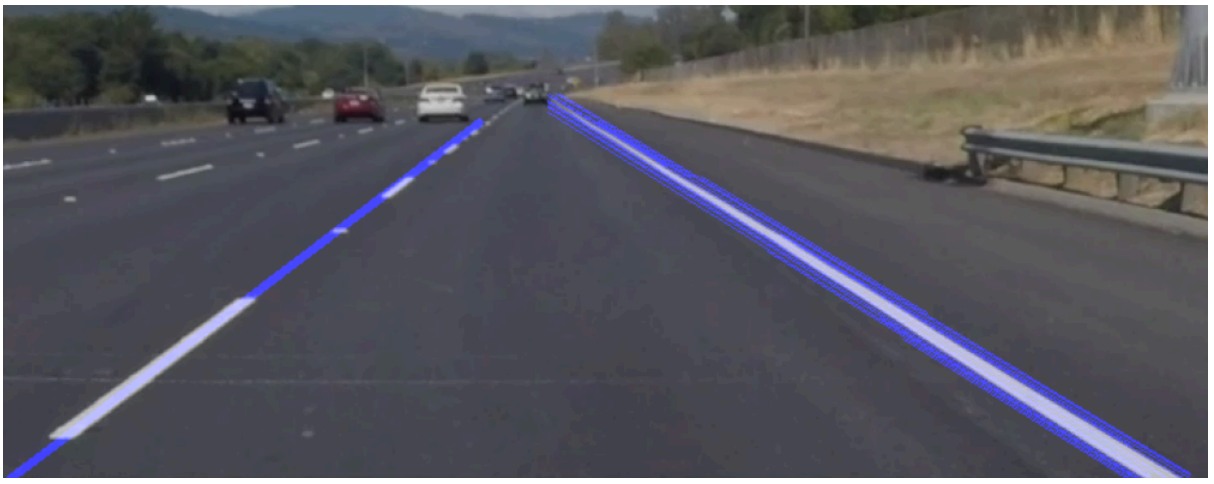

Fig 2: Primer resultado obtenido utilizando cv2.houghLines()

Problemas de esta segunda versión: la línea del lado izquierdo es más corta que la línea del lado derecho y la línea de lado derecho esta formada por muchas líneas juntas, la idea es obtener una sola línea azul.

Con algunas mejoras del código, principalmente ajustando parámetros logramos un resultado que consideramos adecuado y procedimos a pasar dicho código al archivo correspondiente para procesar todo el video obteniendo también un resultado con el cual estuvimos conforme.

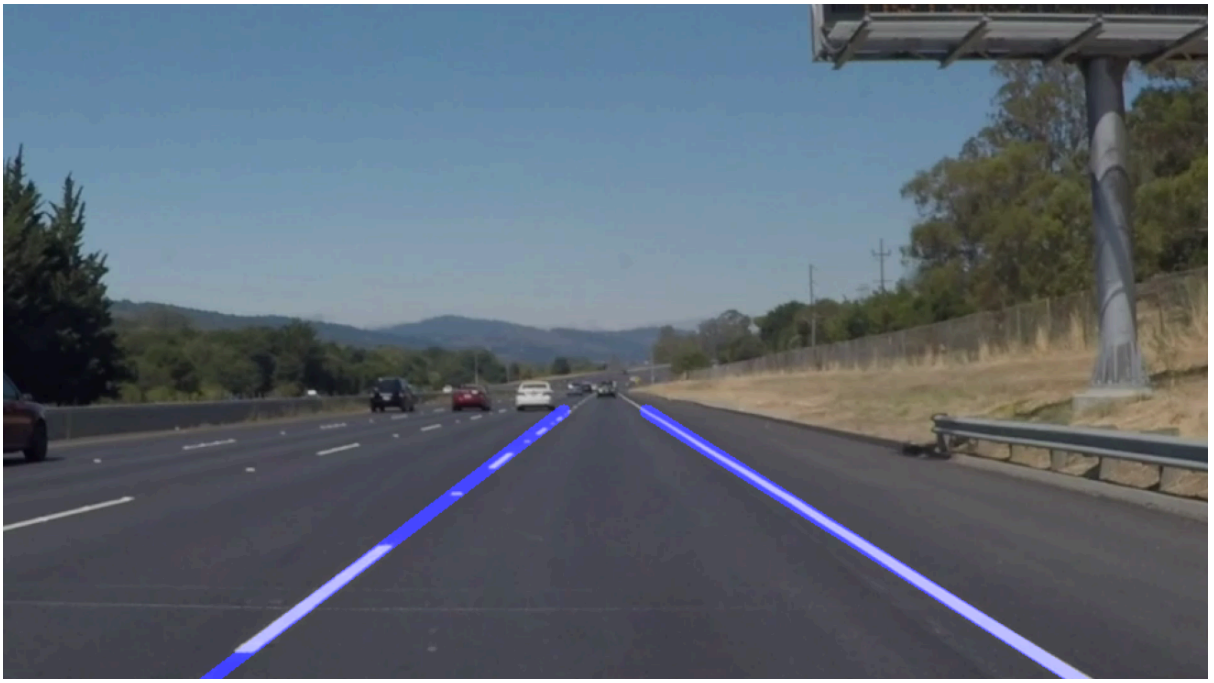

Fig 3: Frame obtenido antes de procesar todo el video

Con respecto al video ruta_2.py el procesamiento del mismo y la detección de las líneas fue más sencilla habiendo realizado todo el trabajo para el primer video, hubo que hacer algunas modificaciones, ya que en este caso la línea del lado derecho era la completa y la del lado izquierdo la línea cortada.
Pero ya en el primer procesamiento de un frame logramos buen resultado.
Una dificultad con la que nos encontramos en el procesamiento del segundo video, es que la línea del lado derecho al final, se iba bastante para el medio de la ruta, quedando despegada de la línea que teníamos que marcar, como se puede observar en la siguiente imagen.

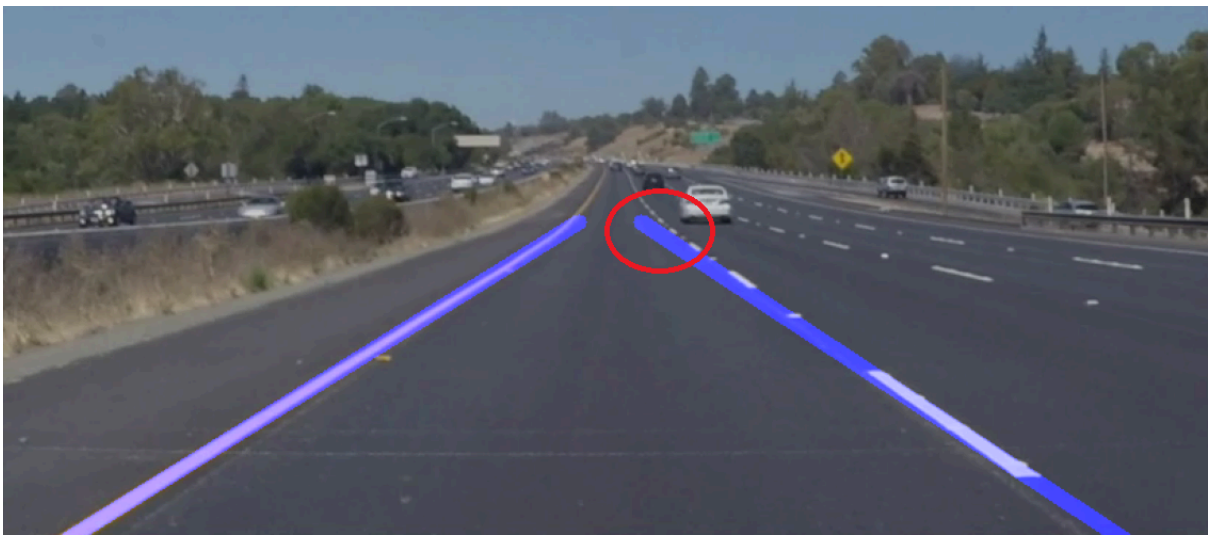

Fig 4: Frame del segundo video donde se puede ver el problema.

Este problema lo pudimos solucionar ajustando los parámetros de la detección de bordes con Canny y el gradiente morfológico.

Al pasar el código a todo el video los resultados no fueron los esperados ya que la línea de la derecha se movía todo el tiempo, a veces quedaba sobre la línea izquierda, otras paralela a ella y en otros casos desaparecía. En la siguiente imagen se puede ver una captura del video.

Fig 5: Captura del video generado para ruta_2

Ajustando la región de interés y algunos valores de las funciones logramos que la línea no se corra del lugar que le correspondía, pero no logramos que deje de aparecer y desaparecer.

Esto último lo pudimos solucionar guardando las coordenadas y realizando promedios. La idea es suavizar la variabilidad de las líneas detectadas

mediante el uso de un promedio móvil con un almacenamiento de líneas detectadas a lo largo de varios cuadros, incluso cuando no se detectan líneas nuevas en cada cuadro. Esto nos ayudó a mantener una línea más estable y a lograr el resultado que esperamos