John Pabon

Proyecto 1: Encontrar líneas de carril en la carretera

Este programa fue desarrollado en Jupiter que viene instalado en Anaconda Navigator versión 1.10. Adicionalmente se instaló en Anaconda Navigator los siguientes programas: Matplolib v.3.3.1, Numpy v1.19.1, Math, Opencv 3.4.2, y Python 3.6.

Pasos para encontrar líneas de carril.

1. ***Importar paquetes***
   #importing some useful packages
   import *matplotlib.pyplot as plt*
   import *matplotlib.image as mpimg*
   import *numpy as np*
   import *cv2*
   *%**matplotlib inline***

2. ***Leer en una imagen***
   *Para leer la imagen original, sólo tiene que llamar al **imread** función del cv2 módulo.*

3. ***Tamaño de la imagen.***
   *En las aplicaciones de procesamiento de imágenes, a menudo es necesario conocer el tamaño de una imagen que se carga. Para obtener la forma o el tamaño de la imagen, use **image.shape** para obtener las dimensiones de la imagen.*

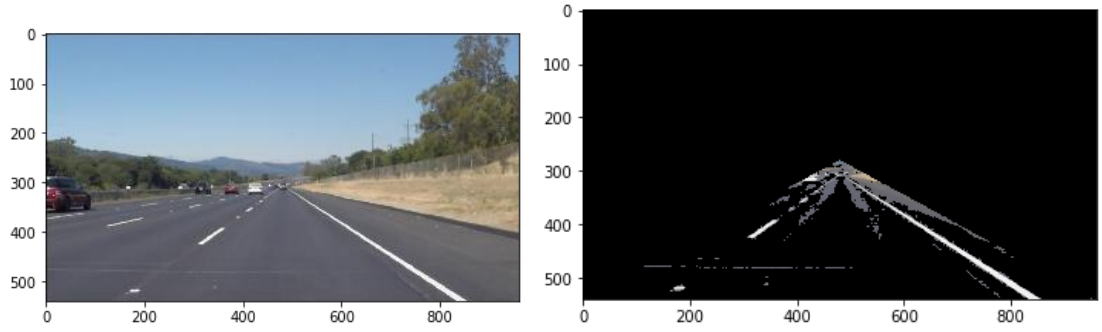
   *Podemos acceder a la altura, el ancho y el número de canales desde **image.shape.** La altura está en el índice 0, la anchura está en el índice 1; y número de canales en el índice 3. Número de canales = 3 representa los canales o colores Azul, Rojo, y Verde.*

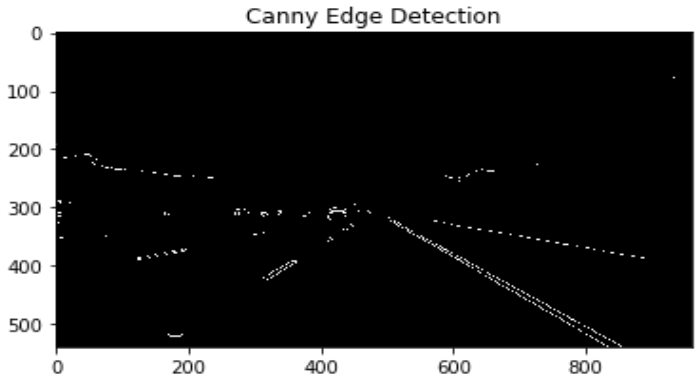
4. ***Copia de la imagen.***
   Debemos hacer una copia de las matrices u otras variables en Python.

   region_select*=np.copy (image)*
   line_image*=np.copy(image)*
   color_select*=np.copy(image)*

5. ***Cambiar umbral de la imagen.***
   Definir el valor del umbral de color en las variables red_threshold, green_threshold, blue_threshold y complete rgb_threshold.
   Este vector deberá contener los valores mínimos para rojo, verde y azul (R, G, B).
   Para lograr separar la región deseada se establece un valor que define el umbral, los pixeles cuya intensidad superen el umbral serán rechazado o aceptados, según sea el caso.

   Ejemplos de imágenes cambiando sus valores.
   red_threshold *= 0 green_threshold = 0 blue_threshold = 0*
   red_threshold *= 100 green_threshold = 100 blue_threshold = 100*
   red_threshold *= 200 green_threshold = 200 blue_threshold = 200*





Con una simple selección de color hemos logrado eliminar casi todo en la imagen excepto las líneas de los carriles.

6. ***Ahora debemos definir nuestra región de interés***.
Estos parámetros define el área de un triángulo.

left_bottom *= [0, 539]*
right_bottom *= [900, 300]*
apex *= [400, 0]*

7. *Ajustar líneas.*

Ajustemos las líneas (y = Ax + B) para identificar el Región de interés de 3 lados
np.polyfit () devuelve los coeficientes [A, B] del ajuste

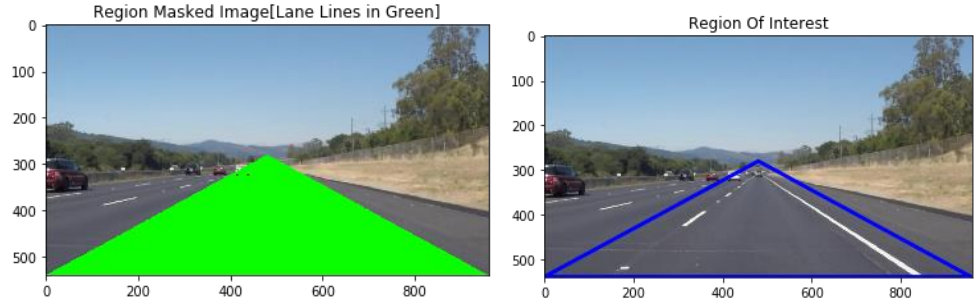
fit_left *= np.polyfit((left_bottom[0], apex[0]), (left_bottom[1], apex[1]), 1)*
fit_right *= np.polyfit((right_bottom[0], apex[0]), (right_bottom[1], apex[1]), 1)*
fit_bottom *= np.polyfit((left_bottom[0], right_bottom[0]), (left_bottom[1], right_bottom[1]), 1)*

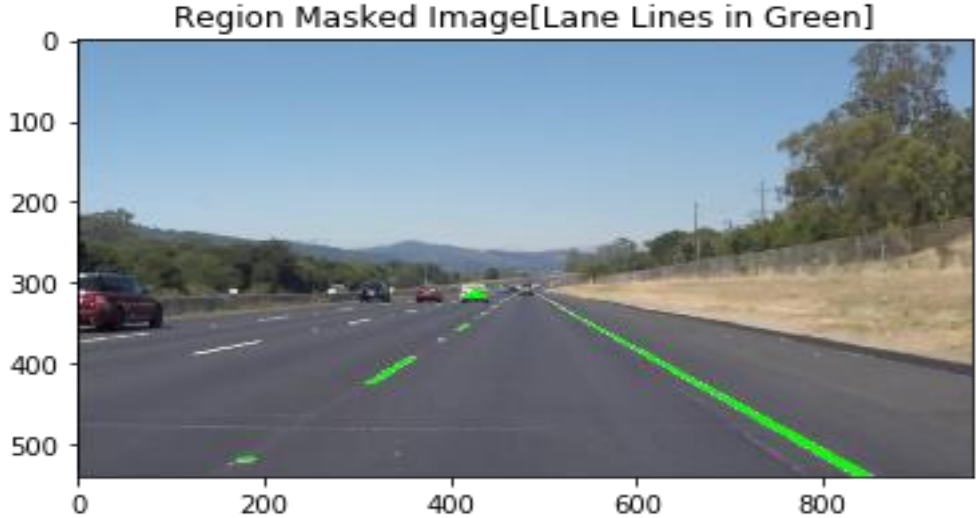
8. *Encontrar la región dentro de las líneas*

XX, YY *= np.meshgrid(np.arange(0, xsize), np.arange(0, ysize))*
*region_thresholds = (YY > (XX*fit_left[0] + fit_left[1])) & *

*(YY > (XX*fit_right[0] + fit_right[1])) & *
*(YY < (XX*fit_bottom[0] + fit_bottom[1]))*

Ahora hemos encontrado o detectado las líneas de la carretera

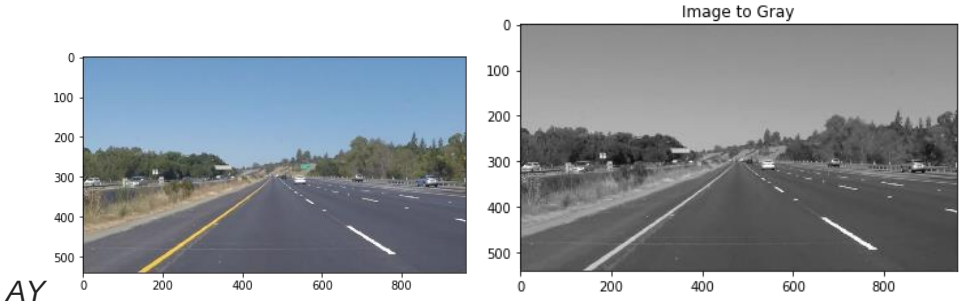




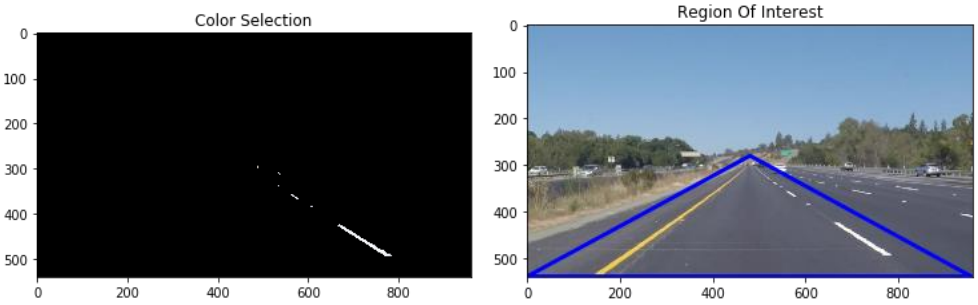
9. *Aplicar Canny.*

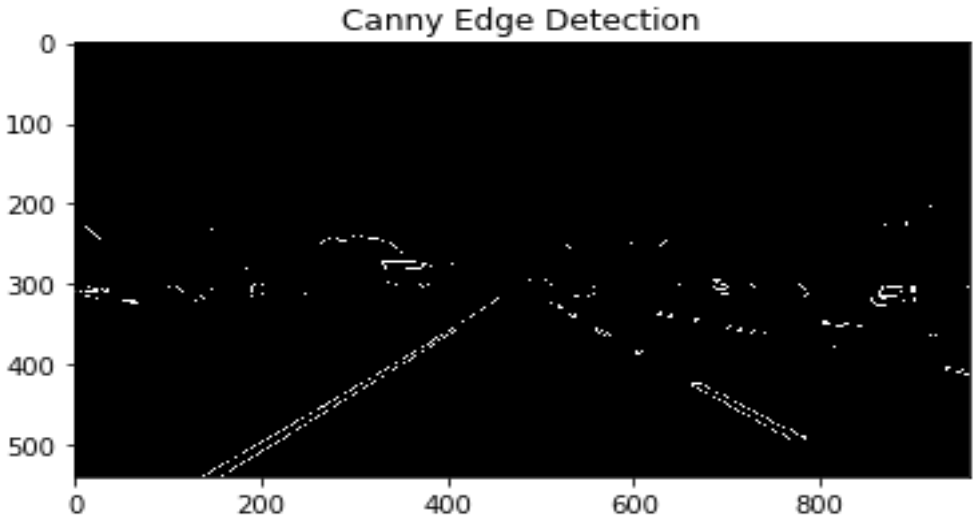
Ahora aplicaremos Canny a la imagen en escala de grises y nuestra salida será otra imagen llamada bordes. low_threshold y high_threshold son sus umbrales para la detección de bordes.

Como primera entrada, esta función recibe la imagen original. Como segunda entrada, recibe el código de conversión del espacio de color. Como queremos convertir nuestra imagen original del espacio de color BGR a gris, usamos el código COLOR_BGR2GRAY



AY

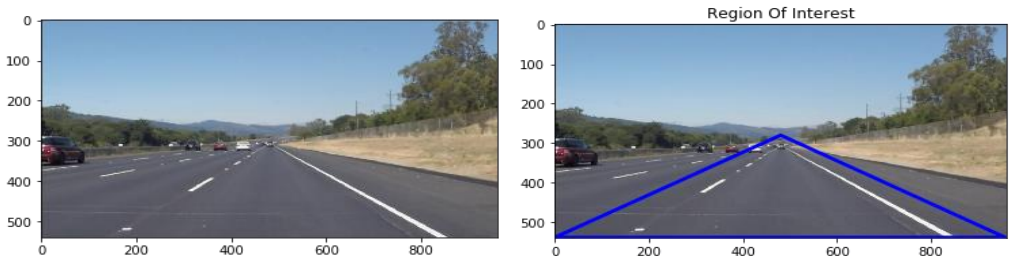


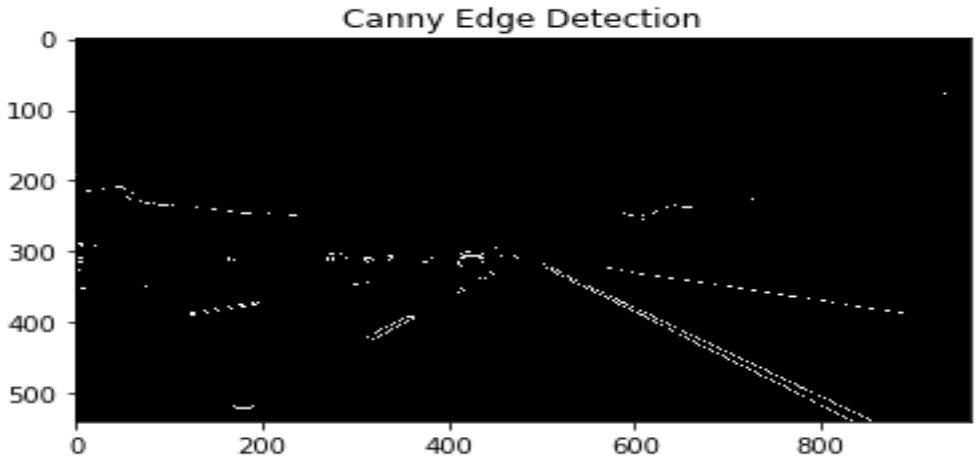


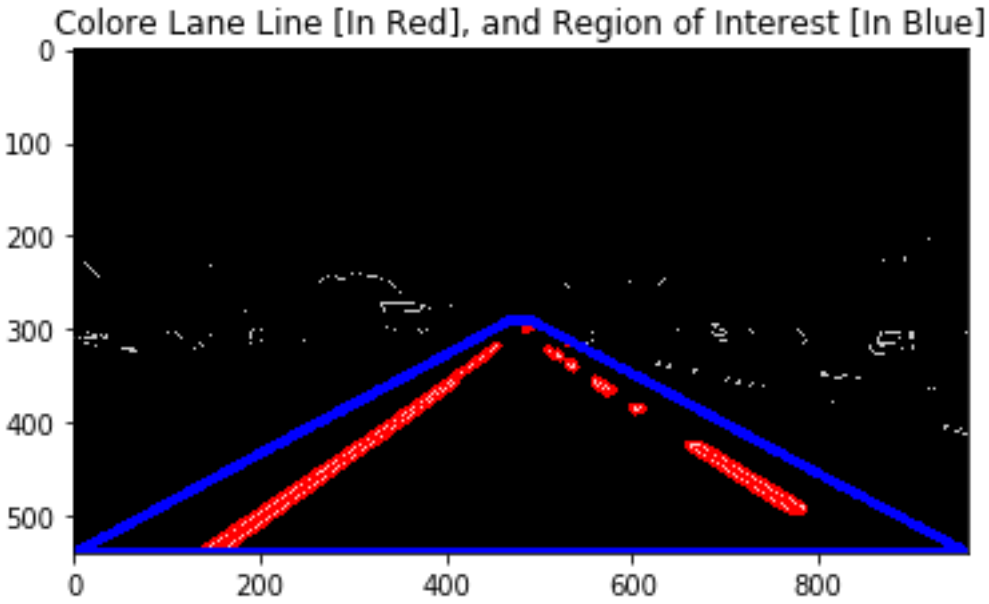
10. *Implementación de una transformación de Hough.*

La tarea es encontrar líneas de carril. Necesitamos especificar algunos parámetros para decir qué tipo de líneas queremos detectar (es decir, líneas largas, líneas cortas, líneas flexibles, líneas discontinuas, etc.).

Para hacer esto, usaremos una función OpenCV llamada HoughLinesPque toma varios parámetros.

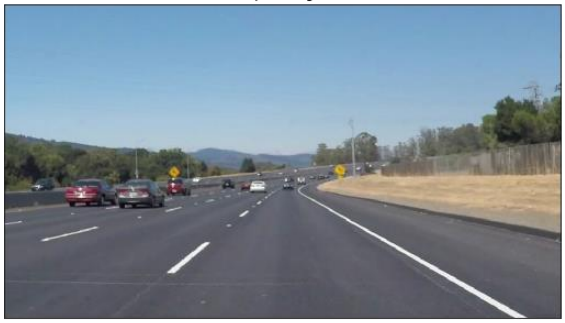






11. ***Ahora se puede efectuar pruebas en diferentes imágenes y efectuar pruebas en video.***

Input Image



Output Image [Lane Line Detected]

Input Image

Output Image [Lane Line Detected]

Input Image

Output Image [Lane Line Detected]

Input Image

Output Image [Lane Line Detected]