Adriano Miranda Seoane

MEMORIA PRACTICA 2 VA:

Trabajo 6. Reconocimiento y detección de señales de tráfico:

El objetivo de este proyecto es reconocer y detectar diferentes señales de tráfico en imágenes. Para ello pensé cuales eran la particularidades que diferencian una señal del resto de elementos de la imagen. Las dos características en las que me centro para detectar cada señal entre los distintos elementos que puede tener una imagen son el color y la forma.

La estrategia que utilizo es un proceso en cascada en el cual filtro primeramente la imagen por colores y después por formas e intento identificarlas (probé a filtrar primero por formas y luego por colores pero el resultado era significativamente peor).

Todas las señales que pretendo identificar son azules o rojas por lo tanto aplico un filtrado de color HSV, aplicando una máscara con rangos de color que son los que se van a filtrar del resto de la imagen. Todos los elementos que no se encuentran en este rango de colores pasan a valer 0. Utilizo este filtrado porque es fácil de implementar y de aplicar. Aquí me encontré con el primer problema al darme cuenta de que el umbral que debía aplicar para señales rojas (por ejemplo) no es el mismo dependiendo de factores como lo lejos o cerca que esté la señal o si hay más luz o menos. Esto quiere decir que para dos señales con el mismo color en imágenes distintas, en una imagen igual tengo que utilizar un umbral mínimo y máximo distinto que el que aplicaría en la otra. De esta forma lo que hago es para cada imagen que quiero filtrar por color le paso un umbral bajo y otro alto para filtrar los colores de esa imagen.

Una vez que tengo la imagen filtrada por color, tengo que filtrarlas por formas para identificar los triángulos, círculos y un caso particular (explico más adelante porqué) que son las señales de STOP. Para detectar círculos utilizo la transformada de Hough vista en clase. La función que utiliza cv2 para aplicar esta transformada (cv2.HoughCircles) tiene varios parámetros, después de probar muchos valores para estos parámetros encontré unos con los que funcionaba bastante bien para cualquier caso, excepto para el radio máximo del circulo. En algunas imágenes con señales redondas grandes debía aumentar este parámetro y en otras más pequeñas tuve que bajarlo para que la detectase. Los círculos detectados se pintan en color verde en la imagen.

Para la detección de formas triangulares, rectangulares, cuadradas u octogonales (señales STOP) cree una función que detecta estas formas dependiendo de su número de lados, aplicando un filtrado gaussiano a la imagen y luego Canny para detectar los contornos del objeto (Por alguna razón la señal de STOP la detecta con 4 lados en vez de 8). Está función a pesar de aplicarla después del filtrado por color detectaba demasiados objetos a parte de las señales, por esta razón se me ocurrió hacer una función para detectar objetos en la imagen después del filtrado por color y antes de la detección de formas aplicando Canny. El resultado mejoró significativamente, sobre todo porque detecta muchos menos objetos a parte de las señales de los que identificaba antes.

Finalmente, para identificar las señales de cada imagen tenemos un proceso en cascada en el cual realizamos los siguientes pasos:

1.- Filtro por color con HSV (varía el umbral alto y bajo dependiendo del color de las señales)
2.- Aplico el detector de bordes de Canny (también puedo variar sus parametros para detectar más o menos objetos), en este punto tengo una imagen solo con los colores de las señales y solo con los objetos detectados.
3.- Aplico el detector de circulos (si hay señales redondas, sino no debería detectar nada)
4.- Aplico el detector de formas geométricas (si quiero identificar una señal triangular le paso como parámetro 3 lados, si es una señal con 4 lados o STOP le paso 4 lados)

Como resultado tenemos la imagen de entrada con todos estos filtros aplicados, de forma que deberían estar todos los pixeles que no son del color de las señales en negro (valor 0) , los elementos detectados con el detector de bordes de Canny en rojo y las señales detectadas tanto con el detector de círculos como el detector de formas geométricas con el contorno verde.

No funciona al 100%, es decir, hay señales que por algún motivo no las detecta (señales que están muy lejos y apenas se ven, señales recortadas que solo se ven una parte de ellas etc.)

Al final del código se encuentran comentados ejemplos de ejecución para las imágenes de prueba.