Practica 3

Detección de objetos

David Vacas Miguel (GII+GIS)
Pedro Redondo Rabanal (GII+GIS)

Índice

1.-Introducción

Esta práctica se basaba en tres partes, la primera trata de la detección del centro de un coche mediante puntos de interés y votación a la Hough, la segunda consiste en la detección de coches usando cascadeClassifier utilizando un clasificador ya entrenado, y en última instancia la detección de coches en videos utilizando ambos métodos (de la parte 1 y de la parte 2).

2.-Explicación del algoritmo desarrollado

En este apartado pasaremos a explicar los diferentes algoritmos mencionados en la introducción:

- Parte 1: Detección de coches mediante puntos de interés. Lo primero que debemos realizar es inicializar el FlannBasedMatcher, que es una clase de opencv que se utiliza para emparejar puntos de interés, así como el detector de puntos de interés ORB, para inicializar este último metemos los parámetros 100 (características que utiliza), 1.3 (escala de la pirámide) y 4 (número de niveles de la pirámide). Una vez realizado esto pasamos a la fase de entrenamiento, en la cual lo primero que realizamos es pedir al usuario que introduzca por consola la dirección de donde el programa tiene que coger las imágenes de entrenamiento (por ejemplo nuestra dirección podría seria "C:\Users\usr\Desktop\VA\Practica1\training\training"). De esta carpeta cogeremos todas las imágenes y por cada imagen en la carpeta guardaremos sus descriptores, obtenidos de los puntos de interés, gracias a ORB y guardamos un vector que apunta hacia el centro de la imagen, puesto que todas las imágenes son del mismo tamaño y el centro del coche esta situado exactamente en el centro de la imagen, y los añadiremos al flannBasedMatcher. Una vez añadidos todos los descriptores, llamamos al método train para entrenar al flannBasedMatcher. Con esto finalizamos la fase de entrenamiento y entramos la fase de procesado. Para esta segunda fase volveremos a pedir al usuario que introduzca por consola la dirección del directorio donde se encuentren las imágenes a procesar. Por cada una de ellas crearemos un vector de votación, a continuación sacaremos los descriptores de esa imagen y los compararemos con los que ha entrenado flannBasedMatcher. Una vez obtenido los 6 descriptores más parecidos, gracias a los vectores guardados en la fase de entrenamiento, realizamos una votación de donde se cree que se encuentra el centro del coche (teniendo en cuenta el tamaño de ambos puntos de interés). Para que la acumulación sea evidente, dividimos por diez las dimensiones del vector en el que se apuntan las votaciones. Una vez realizadas todas las votaciones, se busca el punto de máxima votación donde se cree que esta el centro y este lo señalamos con un circulo en la imagen que se mostrara en una ventana.

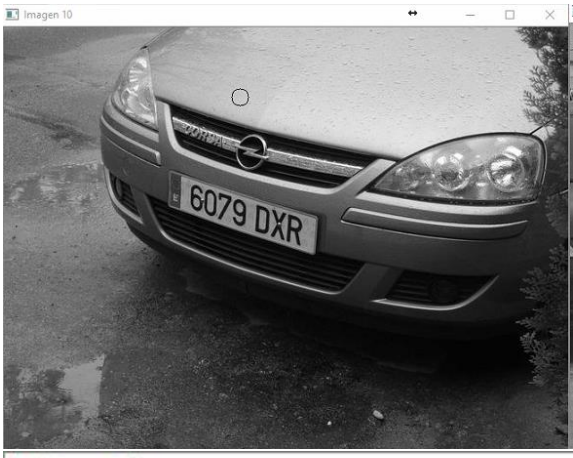
- Parte 2: Detección de coches usando cv2.cascadeClassifier. En primer lugar pediremos al usuario que introduzca por consola donde se encuentra el clasificador ya entrenado (en nuestro caso, y como ejemplo, la dirección seria la siguiente "C:\Users\usr\Desktop\VA\Practica1\haar/coches.xml"), y a continuación pediremos el directorio donde se encuentran las imágenes a procesar. Después

de esto creamos una instancia de la clase de opencv CascadeClassifier. Una vez hecho esto cada imagen localizada en el directorio la pasaremos al método de CascadeClassifier 'detectMultiScale' junto con tres parámetros más: el número de vecinos que tiene que tener para que, en nuestro caso, el rectángulo que envuelve a un coche sea válido (inicializado a 2), cuanto reducimos la escala de cada imagen (asignado con el valor 1,4) y por último el tamaño mínimo que debe tener el objeto (este valor esta proporcionado por el segundo parámetro de entrada). Este método nos devolverá los rectángulos que pintaremos sobre la imagen y nos dirá donde se encuentra el coche.

- Parte 3: Detección del coche en secuencias de video. Para comenzar con esta parte preguntaremos por consola que método desea utilizar para la detección del coche en el video (cascadeClassifier o algoritmo desarrollado por nosotros en la parte 1). Una vez elegido el tipo de algoritmo deseado, se vuelven a pedir los directorios que estos algoritmos necesitan para entrenar y el directorio junto con el nombre del video que se quiera procesar (como ejemplo de esto último el nuestro seria "C:\Users\usr\Desktop\VA\Practica1\Videos\Videos/video1.avi"). La única diferencia con los métodos anteriores es que en vez de procesar muchas imágenes de una carpeta, procesamos cada frame del video.

3.-Capturas de pantalla

- Parte 1



- Parte 2

- Parte 3
  - Resultado utilizando algoritmo parte 1

  - Resultado utilizando algoritmo parte 2

4.-Estadísticas

Las estadísticas obtenidas para la parte 1 de la práctica son las siguientes:

Precisión -> 84% (detección de un solo coche, es decir no detecta más de un centro de coche)

Las estadísticas de la parte 2 de la práctica son:

Precisión -> 81%