20-5-2020

VISIÓN ARTIFICIAL PRÁCTICA 2

Jorge Alonso Vivar, Álvaro Robles Sánchez,
Chengian Li.

Tabla de contenido

1. Apartado 2

En primer lugar, umbralizamos las imágenes con el método "adaptativeThreshold" y usamos los archivos xml proporcionados para localizar los coches y sus matrículas.

Si se ha detectado un coche, se localiza su matrícula y se señalan con un recuadro, para después usar la función "detectorMat" para sacar sus contornos con "findContours"y seleccionar los posibles caracteres con una serie de restricciones.

Si no se ha localizado la matrícula, con el método "detectorCar" trataremos de identificar los contornos que puedan ser caracteres y para después obtener los que estén alineados con "Ransac".

En el caso de que no se halle ningún coche o no se hayan detectado 7 caracteres en la matrícula, se llamará al método "encontrarMatricula" para detectar la matrícula basándose en los contornos en lugar de en los xml y tratar de localizar los caracteres dentro de ella.

2. Apartado 3

En primer lugar, leemos todas las imágenes de entrenamiento, las umbralizamos y obtenemos sus contornos.

Llamamos a la función "detectorCaracter" para leer los contornos obtenidos y seleccionar los caracteres.

Tras ello, añadimos cada carácter a en una matriz 10x10, la convertimos en 1x100 y guardamos en una lista el nombre de las etiquetas obtenidas con el nombre de las imágenes.

Por último, creamos el LDA y el clasificador Gaussiano y usamos los métodos "fit" y "transform" para entrenarlos.

3. Apartado 4

Una vez obtenida lista de posibles caracteres y entrenado el clasificador, llamamos al método "escribirCaracteres". Transformamos la matriz de caracteres con el LDA y creamos el predic con ella. Posteriormente, dibujamos el texto de la matrícula en la imagen justo debajo de esta.

Finalmente, con el método "saveImg" guardamos todos el nombre, centro, texto y longitud de cada matrícula en un archivo llamado "testing_full_system".

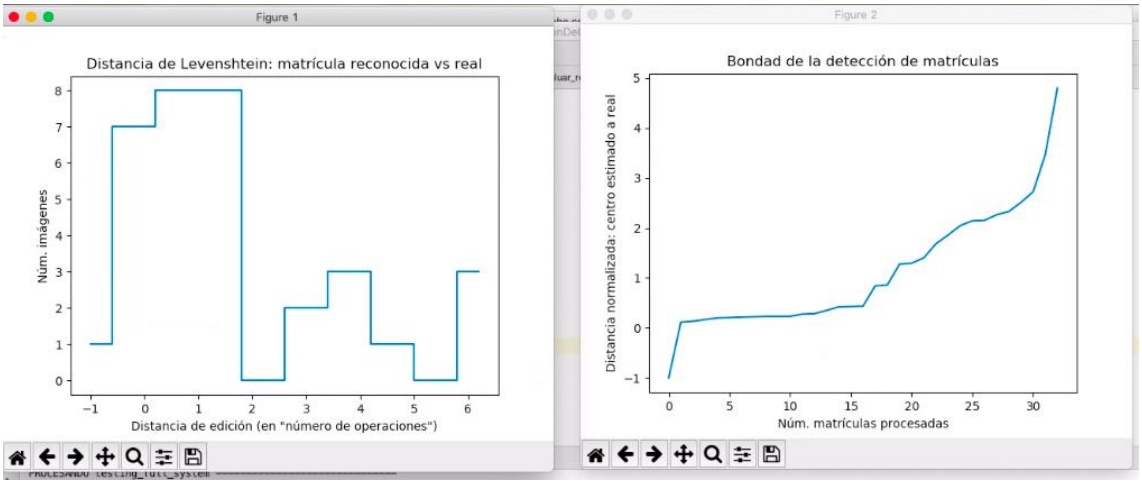
Todo esto se repite para crear el archivo "testing_ocr".

Una vez tenemos ambos archivos ejecutamos el comparador y obtenemos las gráficas.

4. Imágenes


Caracteres
2117 CWP


Caracteres
8821 DFH



5. Complicaciones

El principal problema con el que nos hemos encontrado a la hora de realizar la práctica ha sido el de detectar correctamente los caracteres cuando el detector de matrícula usado con el xml no funcionaba adecuadamente.

Para solucionarlo, si detectamos el coche, utilizamos “Ransac” para seleccionar una serie de contornos alineados ya filtrados previamente para asegurar que son posibles caracteres.

El otro problema surgido fue si no detectábamos tampoco el coche, para lo cual tuvimos que filtrar todos los contornos de la imagen para tratar de localizar la matrícula, y, a partir de ella, identificar sus caracteres.

6. Bibliografía

- Angle : https://likegeeks.com/es/procesar-de-imagenes-en-python/#Detecte-y-corrija-la-inclinacion-del-texto

- FindContours:
  https://hetpro-store.com/TUTORIALES/opencv-findcontours/

- SortContours:
  https://www.pyimagesearch.com/2015/04/20/sorting-contours-using-python-and-opencv/

- PropiedadContornos:
  https://unipython.com/propiedades-los-contornos/

- Morfologia:
  https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/pymorphological_ops/py_morphological_ops.html

- Libreria os:
  https://unipython.com/operaciones-con-archivos-y-carpetas-en-python/

- Blur:
  https://unipython.com/suavizando-imagenes-con-opencv/

- Set:
  https://blog.elcodiguero.com/python/eliminar-objetos-repetidos-de-una-lista.html