CascadeClassifier :: CascadeClassifier

Carga un clasificador de un archivo.

**C ++:**CascadeClassifier::**CascadeClassifier**( cadena constante y **nombre de archivo**)

**Python:**cv2.**CascadeClassifier**( [ nombre de archivo ] ) → <objeto CascadeClassifier>

|  |  |
| --- | --- |
| **Parámetros:** | **filename** : nombre del archivo desde el que se carga el clasificador. |

CascadeClassifier :: detectMultiScale

Detecta objetos de diferentes tamaños en la imagen de entrada. Los objetos detectados se devuelven como una lista de rectángulos.

**C ++:**void CascadeClassifier::**detectMultiScale**( const Mat & **image** , vector <Rect> & **objects** , double **scaleFactor** = 1.1, int **minNeighbors** = 3, int **flags** = 0, Size **minSize** = Size (), Size **maxSize** = Size () )

**Python:**cv2.CascadeClassifier.**detectMultiScale**( image [ , scaleFactor [ , minNeighbors [ , flags [ , minSize [ , maxSize ] ] ] ] ] ) → objetos

**Python:**cv2.CascadeClassifier.**detectMultiScale**( imagen, niveles de rechazo, pesos de nivel [ , factor de escala [ , vecinos mínimos [ , banderas [ , tamaño mínimo [ , tamaño máximo [ , niveles de rechazo de salida ] ] ] ] ] ] ) → objetos

**C:**CvSeq \* **cvHaarDetectObjects**( const CvArr \* **imagen** , CvHaarClassifierCascade \* **cascada** , CvMemStorage \* **almacenamiento** , doble **factor de escala** = 1.1, int **min\_neighbors** = 3, int **flags** = 0, CvSize **min\_size** = cvSize (0,0), CvSize **max\_size** = cvSize (0 0) )

**Python:**cv.**HaarDetectObjects**( imagen, cascada, almacenamiento, factor\_escala = 1.1, vecinos\_mín = 3, banderas = 0, tamaño\_mín = (0, 0) ) → objetos detectados

|  |  |
| --- | --- |
| **Parámetros:** | * **cascade** - **Cascada del** clasificador Haar (solo API OpenCV 1.x). Se puede cargar desde un archivo XML o YAML usando [**Load()**](https://docs.opencv.org/2.4/modules/core/doc/old_xml_yaml_persistence.html#void*%20cvLoad(const%20char*%20filename,%20CvMemStorage*%20memstorage,%20const%20char*%20name,%20const%20char**%20real_name)). Cuando la cascada ya no sea necesaria, suéltela usando cvReleaseHaarClassifierCascade(&cascade). * **imagen** : matriz del tipo que CV\_8U contiene una imagen en la que se detectan objetos. * **objetos** : vector de rectángulos donde cada rectángulo contiene el objeto detectado. * **scaleFactor** : parámetro que especifica cuánto se reduce el tamaño de la imagen en cada escala de imagen. * **minNeighbors** : parámetro que especifica cuántos vecinos debe tener cada rectángulo candidato para retenerlo. * **flags** - Parámetro con el mismo significado para una cascada antigua que en la función cvHaarDetectObjects. No se utiliza para una nueva cascada. * **minSize** : tamaño de objeto mínimo posible. Los objetos más pequeños que eso se ignoran. * **maxSize** : tamaño máximo de objeto posible. Los objetos más grandes que eso se ignoran. |

La función se paraleliza con la biblioteca TBB.

**Nota**

* (Python) Se puede encontrar un ejemplo de detección de rostros usando clasificadores en cascada en opencv\_source\_code / samples / python2 / facedetect.py

**CONCEPTOS**

La detección de objetos mediante clasificadores en cascada basados en funciones de Haar es un método eficaz de detección de objetos propuesto por Paul Viola y Michael Jones en su documento,“Rapid Object Detection using a Boosted Cascade of Simple Features” en 2001. Se trata de un enfoque basado en el aprendizaje automático en el que la función en cascada se entrena a partir de muchas imágenes positivas y negativas. Luego se utiliza para detectar objetos en otras imágenes.

Aquí trabajaremos con la detección de rostros o caras. Inicialmente, el algoritmo necesita muchas imágenes positivas (imágenes de caras) y negativas (imágenes sin caras) para entrenar al clasificador. Entonces necesitamos extraer rasgos de él. Para ello se utilizan las características de Haar mostradas en la imagen inferior. Son como nuestro núcleo convolucional. Cada característica es un valor individual obtenido restando la suma de píxeles bajo rectángulo blanco de la suma de píxeles bajo rectángulo negro.{