4.Face recognition

1. Basic theory

Face detection algorithms can be divided into two categories according to methods, **feature-based algorithms and image-based algorithms**.

- **Feature-based algorithm**

  The feature-based algorithm extracts features from the image and matches them with face features. If they match, it means it is a face, and vice versa. The extracted features are artificially designed features, such as Haar and FHOG. After the features are extracted, the classifier is used to make judgments. In layman's terms, template matching is used, which uses a template image of a face to match each position in the image to be detected. The matching content is the extracted features, and then a classifier is used to determine whether there is a face.

- **Image-based algorithm**

Image-based algorithm, **divides the image into many small windows, and then determines whether each small window contains a face**. Usually image-based methods rely on statistical analysis and machine learning to find the statistical relationship between faces and non-faces through statistical analysis or learning process for face detection. The most representative one is CNN. **CNN is currently the best and fastest used for face detection**.

- **Haar Features**

We use **machine learning** methods to complete **face detection**. First, we need a large number of positive sample images (face images) and negative sample images (images without faces) to train the classifier**. We need to extract features from it. The** Haar features** in the figure below will be used, just like our convolution kernel, **each feature is a value**,This value is equal to the sum of the pixel values in the black rectangle minus the sum of the pixel values in the white rectangle.

This tutorial is about using it. Haar cascade classifier is one of the most common and efficient object detection methods. This machine learning method trains a cascade function based on a large number of positive and negative images, and then uses it to detect objects in other images. If you don't want to create your own classifier, OpenCV also includes many pre-trained classifiers for detection of faces, eyes, smiles, etc.

2.Main code

Code path：/home/yahboom/Dofbot/6.AI_Visuall/6.Face recognition.ipynb

The following code content needs to be executed according to the actual step. It cannot be run all at once. Running the last unit will directly exit the thread.

Import Haar cascade classifier xml file

```
self.faceDetect = cv.CascadeClassifier("haarcascade_frontalface_default.xml")
```

Filter faces

```
def face_filter(self, faces):
```

```
        '''
        Filter the face
        Filter faces
        '''
        if len(faces) == 0: return None
        # At present, we are looking for the face with the largest area in the
pictur
        # What we are currently looking for is the face with the largest area in
the picture.
        max_face = max(faces, key=lambda face: face[2] * face[3])
        (x, y, w, h) = max_face
        # Set the minimum threshold of face detection
        # Set minimum threshold for face detection
        if w < 10 or h < 10: return None
        return max_face
```

Recognize selected faces

```
 def follow_function(self, img):
        img = cv.resize(img, (640, 480))
        # Copy the original image to avoid interference during processing
        # Copy the original image to avoid interference during processing
        # img = img.copy()
        # Convert image to grayscale
        # Convert image to grayscale
        gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
        # Face detection
        # Face detection
        faces = self.faceDetect.detectMultiScale(gray, scaleFactor=1.3,
minNeighbors=5)

        if len(faces) != 0:
            face = self.face_filter(faces)
            # Face filtering
            # Face filtering
            (x, y, w, h) = face
            # Draw a rectangle on the original color map
            # Draw a rectangle on the original color map
            cv.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 4)
            cv.putText(img, 'Person', (280, 30), cv.FONT_HERSHEY_SIMPLEX, 0.8,
(105, 105, 105), 2)
```

Main thread:

```
def camera():
    global model
    # Open camera
    capture = cv.VideoCapture(0)
    while capture.isOpened():
        try:
            _, img = capture.read()
            img = cv.resize(img, (640, 480))
            img = follow.follow_function(img)
            if model == 'Exit':
```

```
                cv.destroyAllWindows()
                capture.release()
                break
            imgbox.value = cv.imencode('.jpg', img)[1].tobytes()
        except KeyboardInterrupt:capture.release()
```

After the program is run, you can see that the face has been recognized and selected.