

# Object recognition

By calling yolov11 through opencv for object recognition detection, most object names can be recognized.

## 1. Code path

Code path:

```
~/dofbot_pro/src/dofbot_basic_visual/scripts/06.Object_Recognition.ipynb
```

## 2. File configuration

First, we need to import the yolo11n.engine network model structure cfg file, network weights file, and the txt file of the classification name of the COCO dataset. (Here we directly use the yolo4 official dataset and model)

## 3. Target detection

### 3.1 Build model network structure

First, use the **cv2.dnn.readNet()** function to construct the CSPDarknet53 network structure, pass in the model structure cfg file, and the network weights file. Opencv provides several methods for supporting image classification, detection, and segmentation for neural network modules, automatically implementing pre-processing and post-processing of input images. Here, the target detection module **cv2.dnn\_DetectionModel()** is used to pass in the network model.

```
robot = Robot_Controller()
robot.move_init_pose()
fps = FPS()
yolo_model = YOLO("./yolo11n.engine", task='detect')
model = "General"
```

### 3.2 Target Detection Method

```
results = yolo_model(img, save=False, verbose=False) # Object detection using YOLO11
annotated_frame = results[0].plot(
    labels = True, # Display labels
    conf = False, # Display confidence
    boxes = True, # Draw bounding boxes
)classids, scores, bboxes = self.model.detect(image, confThreshold,
numsThreshold)
```

**Parameters:**

**labels = True, # Display labels**

**conf = False, # Display confidence**

**boxes = True, # Draw bounding box**

**Return value:**

**classIds:**Category index

**scores:**Confidence, probability that the detection box belongs to a certain category

**boxes:**Detection box information, upper left corner coordinates (x, y), box width and height (w, h)

### 3.3 Set model input parameters

```
self.model.setInputParams(size=(320,320), scale=1/255)
```

**size** means scaling the input image to the specified size. The larger the size, the better the detection effect, but the slower the detection speed. **scale** indicates the scaling size of the pixel value.

## 4. Main code

Import various libraries and model files

```
#!/usr/bin/env python
# coding: utf-8
import Arm_Lib
import cv2 as cv
import threading
from time import sleep
import ipywidgets as widgets
from IPython.display import display
from Object_recognition import Object_recognition_identify
```

Object recognition function

```
def detect_image(self, image):

    classids, scores, bboxes = self.model.detect(image, 0.5, 0.3)

    for class_id, self.score, bbox in zip(classids, scores, bboxes):
        self.x, self.y, self.w, self.h = bbox
        self.class_name = self.classes[class_id]

        cv2.rectangle(image, (self.x,self.y), (self.x+self.w,self.y+self.h),
(255,255,0), 2)

        cv2.putText(image, self.class_name, (self.x,self.y+self.h+20),
cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,0), 2)

        cv2.putText(image, str(int(self.score*100))+'%', (self.x,self.y-5),
cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,255), 2)

    return image
```

List of object names:

```
1 person
2 bicycle
3 car
4 motorbike
5 aeroplane
6 bus
7 train
8 truck
9 boat
10 traffic light
11 fire hydrant
12 stop sign
13 parking meter
14 bench
15 bird
16 cat
17 dog
18 horse
19 sheep
20 cow
21 elephant
22 bear
23 zebra
24 giraffe
25 backpack
26 umbrella
27 handbag
28 tie
29 suitcase
30 frisbee
31 skis
32 snowboard
33 sports ball
34 kite
35 baseball bat
36 baseball glove
37 skateboard
38 surfboard
39 tennis racket
40 bottle
41 wine glass
42 cup
43 fork
```

Main Thread

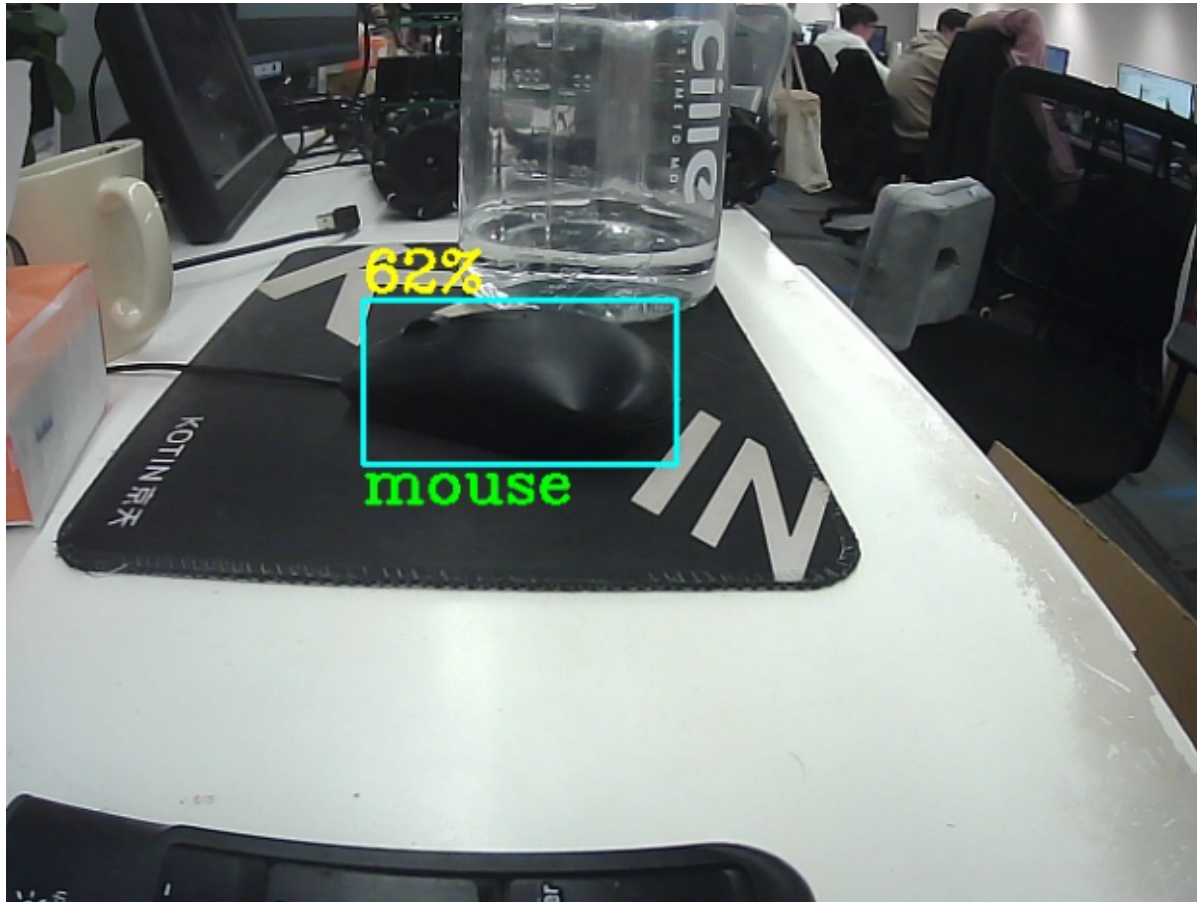
```
def camera():
    # 打开摄像头 Open camera
    capture = cv.VideoCapture(0)
    # 当摄像头正常打开的情况下循环执行
    while capture.isOpened():
        try:
            _, img = capture.read()
            img = cv.resize(img, (640, 480))
            img = ob_re.detect_image(img)
            if model == 'Exit':
```

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

Program Click the Run Entire Program button on the Jupyterlab toolbar, then scroll to the bottom to see the camera component display.



At this time, put the recognizable object into the camera screen, and the object name can be framed and displayed.



Exit

If you need to exit the program, please click the [Exit] button.

