Model Prediction

Note: If using the docker container from the factory image, you don't need to rebuild the environment. The environment is already set up, you just need to enter the docker according to the previous tutorial and run the corresponding function commands to use it.

1. Program Description

Use Python to call USB cameras or depth cameras to use your own trained models. Here we use our trained orange recognition model as an example. If you need personal training models, you can refer to the following two links, but training on Raspberry Pi is not recommended

Labeling: GitHub - HumanSignal/label-studio: Label Studio is a multi-type data labeling and annotation tool with standardized output format

Training: Use Ultralytics YOLO for Model Training - Ultralytics YOLO Documentation

2. Program Startup

2.1 Start Camera

Start the following program according to your camera model, enter in terminal:

```
ros2 launch orbbec_camera dabai_dcw2.launch.pyy
```

Open another terminal and enter the code folder:

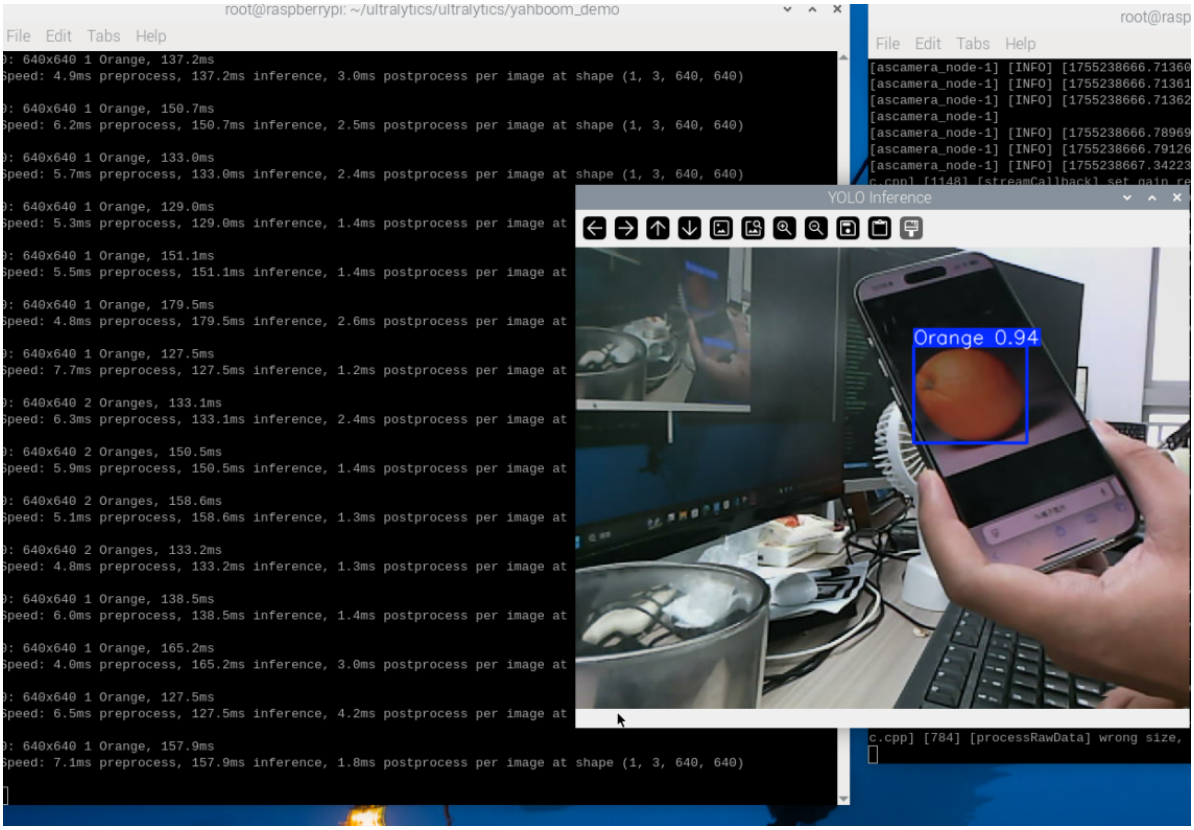
```
cd /root/ultralytics/ultralytics/yahboom_demo
```

Run the code: Click on the preview screen and press q key to terminate the program!

```
python3 06.orange_camera_usb.py
```

Effect Preview

yolo recognition output video location: /root/ultralytics/ultralytics/output/



Example code:

```
#!/usr/bin/env python3
import rclpy
from rclpy.node import Node
from sensor_msgs.msg import Image, CompressedImage
from cv_bridge import CvBridge
import cv2
from ultralytics import YOLO
import os

class Image_detection(Node):
    def __init__(self):
        super().__init__('Image_detection')
        self.model =
YOLO("/root/ultralytics/ultralytics/data/yahboom_data/orange_data/runs/detect/tr
ain/weights/best_ncnn_model")
        self.bridge = CvBridge()
        self.subscription =
self.create_subscription(Image,"/camera/color/image_raw",self.image_callback,10)

    def image_callback(self, msg):
        cv_image = self.bridge.imgmsg_to_cv2(msg, desired_encoding='bgr8')
        self.proecc(cv_image)


# Loop through the video frames
    def proecc(self,frame):
        # Run YOLO inference on the frame
        results = self.model(frame)

        # Visualize the results on the frame
        annotated_frame = results[0].plot()
```

```
        # Display the annotated frame
        cv2.imshow("YOLO Inference", cv2.resize(annotated_frame, (640, 480)))

        # Break the loop if 'q' is pressed
        cv2.waitKey(1) & 0xFF == ord("q")

    def cancel(self):
        cv2.destroyAllWindows()


def main(args=None):
    rclpy.init(args=args)
    node = Image_detection()
    try:
        rclpy.spin(node)
    except KeyboardInterrupt:
        pass
    finally:
        node.cancel()
        node.destroy_node()
        rclpy.shutdown()

if __name__ == '__main__':
    main()
```