Oriented Bounding Boxes Object Detection

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. Oriented Bounding Boxes Object Detection: Image

Use yolo11n-obb.pt to predict images that come with the ultralytics project.

Enter the code folder:
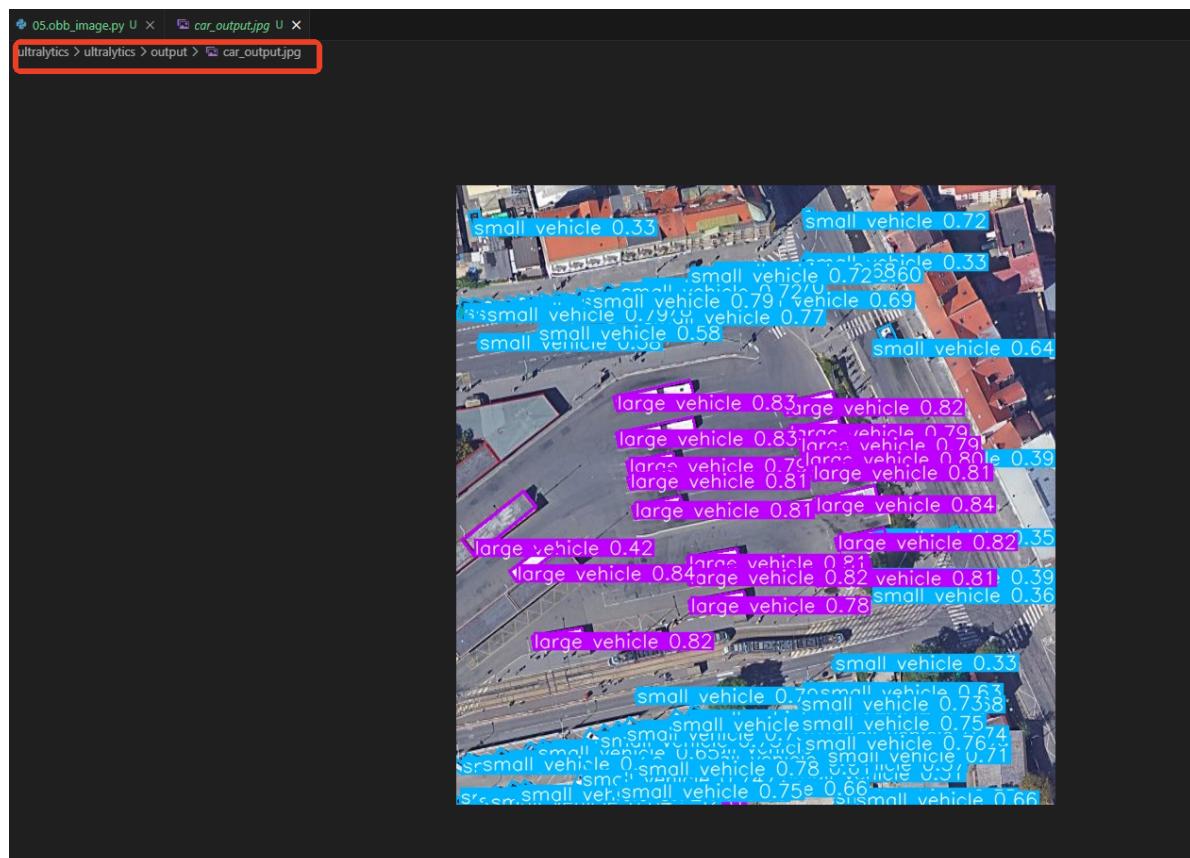
```
cd ~/ultralytics/ultralytics/yahboom_demo
```

Run the code:

```
python3 05.obb_image.py
```

Effect Preview

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

Example code:

```
from ultralytics import YOLO

# Load a model
model = YOLO("/root/ultralytics/ultralytics/yolo11n-obb.pt")
# model = YOLO("/root/ultralytics/ultralytics/yolo11n-obb.onnx")
```

```
# model = YOLO("/root/ultralytics/ultralytics/yolo11n-obb_ncnn_model")

# Run batched inference on a list of images
results = model("/root/ultralytics/ultralytics/assets/car.jpg")  # return a list
of Results objects

# Process results list
for result in results:
    boxes = result.boxes  # Boxes object for bounding box outputs
    # masks = result.masks  # Masks object for segmentation masks outputs
    # keypoints = result.keypoints  # Keypoints object for pose outputs
    # probs = result.probs  # Probs object for classification outputs
    obb = result.obb  # Oriented boxes object for OBB outputs
    result.show()  # display to screen
    result.save(filename="/root/ultralytics/ultralytics/output/car_output.jpg")
# save to disk
```

2. Oriented Bounding Boxes Object Detection: Video

Use yolo11n-obb.pt to predict videos in the ultralytics project (not videos that come with
ultralytics).

Enter the code folder:

```
cd /root/ultralytics/ultralytics/yahboom_demo
```
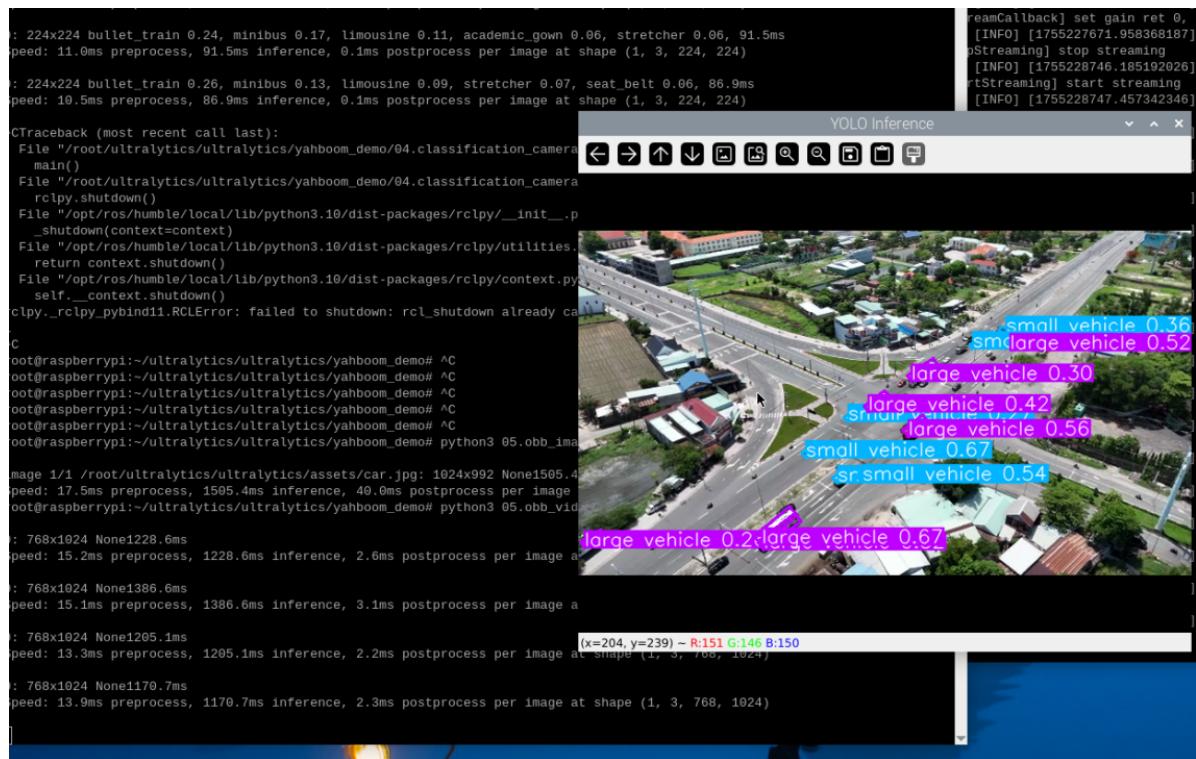
Run the code:

```
python3 05.obb_video.py
```

Effect Preview

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

Example code:

```python
import cv2
from ultralytics import YOLO

# Load the YOLO model
model = YOLO("/root/ultralytics/ultralytics/yolo11n-obb.pt")
# model = YOLO("/root/ultralytics/ultralytics/yolo11n-obb.onnx")
# model = YOLO("/root/ultralytics/ultralytics/yolo11n-obb_ncnn_model")

# Open the video file
video_path = "/root/ultralytics/ultralytics/videos/street.mp4"
cap = cv2.VideoCapture(video_path)

# Get the video frame size and frame rate
frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
fps = int(cap.get(cv2.CAP_PROP_FPS))

# Define the codec and create a VideoWriter object to output the processed video
output_path = "/root/ultralytics/ultralytics/output/05.street_output.mp4"
fourcc = cv2.VideoWriter_fourcc(*'mp4v')  # You can use 'XVID' or 'mp4v'
depending on your platform
out = cv2.VideoWriter(output_path, fourcc, fps, (frame_width, frame_height))

# Loop through the video frames
while cap.isOpened():
    # Read a frame from the video
    success, frame = cap.read()

    if success:
        # Run YOLO inference on the frame
        results = model(frame)

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

        # Write the annotated frame to the output video file
        out.write(annotated_frame)

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

        # Break the loop if 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord("q"):
            break
    else:
        # Break the loop if the end of the video is reached
        break
# Release the video capture and writer objects, and close the display window
cap.release()
out.release()
cv2.destroyAllWindows()
```

3. Oriented Bounding Boxes Object Detection: Real-time Detection

3.1 Start Camera

```
ros2 launch orrbec_camera dabai_dcw2.launch.py
```

Open another terminal and enter the code folder:
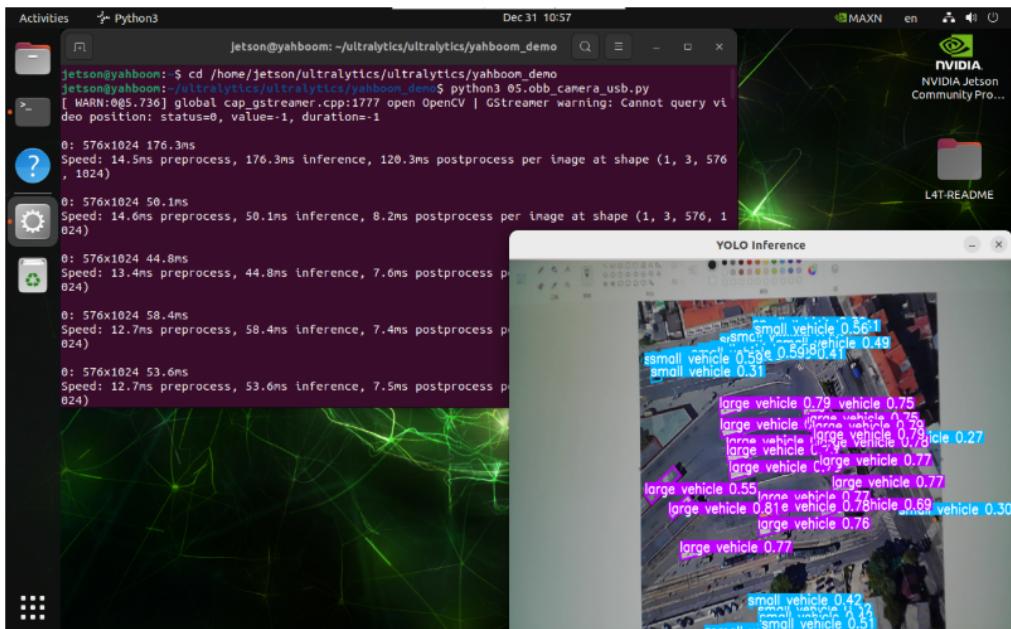
```
cd /root/ultralytics/ultralytics/yahboom_demo
```

Run the code: Click on the preview screen and press Ctrl+c key to terminate the program!

```
python3 05.obb_camera.py
```

Effect Preview

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

Example code:

```python
#!/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 = model = YOLO("/root/ultralytics/ultralytics/yolo11n-obb.onnx")
        self.bridge = CvBridge()
```

```python
        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()
        self.out.release()


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()
```