用苹果控制乌龟 (ROS2)

rqt_graph

ROS2 工作空间配置

1. 创建 ROS2 工作空间

```bash
mkdir -p ~/dev_ws/src  # 创建 dev_ws/src 目录
cd ~/dev_ws  # 进入工作空间
```

2. 创建 ROS2 包

```bash
cd ~/dev_ws/src
```

```
# 方式1：克隆现有包（如你的代码仓库）
git clone <你的代码仓库地址>
# 方式2：手动创建新包（如 learning_topic）
ros2 pkg create --build-type ament_python learning_topic
cd learning_topic
mkdir -p learning_topic  # 创建Python子包目录
touch learning_topic/__init__.py  # 初始化文件
```

创建了后能看到这样的目录：

Code block

```
dev_ws/                    # ROS2 工作空间
├── build/                 # 编译生成的临时文件（自动创建）
├── install/               # 安装目录（含可执行文件）
├── log/                   # 编译日志（自动创建）
└── src/
    └── learning_topic/    # 你的ROS2包
        ├── learning_topic/    # Python包主目录
        │   ├── __init__.py    # Python包初始化文件（可为空）
        │   ├── apple_detector.py          # 苹果检测节点代码
        │   └── apple_controlled_turtle.py  # 海龟控制节点代码
        ├── package.xml    # 包定义文件（声明依赖和元数据）
        └── setup.py       # Python包的安装配置
```

要把apple_detector.py和apple_controlled_turtle.py新建到src/learning_topic里 （我们会用到的节点）

写程序

1. 添加依赖

`package.xml` 里要加这些依赖（dependencies）

package.xml

```
<depend>rclpy</depend>
<depend>geometry_msgs</depend>
<depend>turtlesim</depend>
```

2. 红色苹果检测节点

用于检测图像中的红色物体（如苹果），并通过 `/apple_detected` 话题发布检测结果（`True` 或 `False`）。它订阅摄像头图像，使用HSV颜色空间和轮廓检测判断是否存在红色物体，并在图像上标

记检测到的目标。

apple_detector.py

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import rclpy
from rclpy.node import Node
from sensor_msgs.msg import Image
from std_msgs.msg import Bool                          # use Bool message type
from cv_bridge import CvBridge
import cv2
import numpy as np

# HSV color range for red
lower_red = np.array([0, 90, 128])
upper_red = np.array([180, 255, 255])

class ImageSubscriber(Node):
    def __init__(self, name):
        super().__init__(name)
        self.sub = self.create_subscription(
            Image, 'image_raw', self.listener_callback, 10)
        self.pub = self.create_publisher(
            Bool, 'apple_detected', 10)              # publishing Bool
        self.cv_bridge = CvBridge()

    def object_detect(self, image):
        hsv_img = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
        mask_red = cv2.inRange(hsv_img, lower_red, upper_red)
        contours, _ = cv2.findContours(
            mask_red, cv2.RETR_LIST, cv2.CHAIN_APPROX_NONE)

        for cnt in contours:
            if cnt.shape[0] < 150:
                continue

            (x, y, w, h) = cv2.boundingRect(cnt)
            cv2.drawContours(image, [cnt], -1, (0, 255, 0), 2)
            cv2.circle(image, (int(x+w/2), int(y+h/2)), 5, (0, 255, 0), -1)

            return True  # apple detected

        return False  # no apple found

    def listener_callback(self, data):
```

```
        self.get_logger().info('Receiving video frame')
        image = self.cv_bridge.imgmsg_to_cv2(data, 'bgr8')

        apple_found = self.object_detect(image)

        msg = Bool()
        msg.data = apple_found
        self.pub.publish(msg)

        cv2.imshow("object", image)
        cv2.waitKey(50)

def main(args=None):
    rclpy.init(args=args)
    node = ImageSubscriber("apple_detector")
    rclpy.spin(node)
    node.destroy_node()
    rclpy.shutdown()

```

3. 海龟模拟器苹果控制节点

当检测到苹果时控制海龟画图以1.0的线速度和角速度移动，未检测到时停止。它订阅 `/apple_detected` 话题获取苹果检测状态，并通过 `/turtle1/cmd_vel` 话题发布控制指令。

apple_controlled_turtle.py

```python
#!/usr/bin/env python3

import rclpy
from rclpy.node import Node
from std_msgs.msg import Bool
from geometry_msgs.msg import Twist

class AppleControlledTurtle(Node):
    def __init__(self):
        super().__init__('apple_controlled_turtle')
        self.apple_detected = False

        # Subscriber to /apple_detected
        self.sub = self.create_subscription(
            Bool,
            'apple_detected',
            self.apple_callback,
            10
        )
```

```
20
21          # Publisher to /turtle1/cmd_vel
22          self.pub = self.create_publisher(Twist, 'turtle1/cmd_vel', 10)
23
24          # Timer to send velocity commands at 10 Hz
25          self.timer = self.create_timer(0.1, self.publish_cmd_vel)
26
27          self.get_logger().info('Apple-controlled turtle is running.')
28
29      def apple_callback(self, msg):
30          self.apple_detected = msg.data
31          if self.apple_detected:
32              self.get_logger().info('Apple detected - Turtle moving.')
33          else:
34              self.get_logger().info('No apple - Turtle stopping.')
35
36      def publish_cmd_vel(self):
37          twist = Twist()
38          if self.apple_detected:
39              twist.linear.x = 1.0
40              twist.angular.z = 1.0
41          else:
42              twist.linear.x = 0.0
43              twist.angular.z = 0.0
44          self.pub.publish(twist)
45
46  def main(args=None):
47      rclpy.init(args=args)
48      node = AppleControlledTurtle()
49      rclpy.spin(node)
50      node.destroy_node()
51      rclpy.shutdown()
52
53  if __name__ == '__main__':
54      main()
55
```

4. 添加接入口

在 `setup.py` 中添加 `entry_points` 的作用是将 Python 脚本注册为可执行命令，使其可以通过终端直接运行。

setup.py

```
1   entry_points={
2           'console_scripts': [
3             'apple_detector = learning_topic.apple_detector:main',
```

```
            'apple_controlled_turtle =
    learning_topic.apple_controlled_turtle:main',
        ],
    },
```

运行程序

（如果用虚拟机器要先把电脑的摄像头连上）

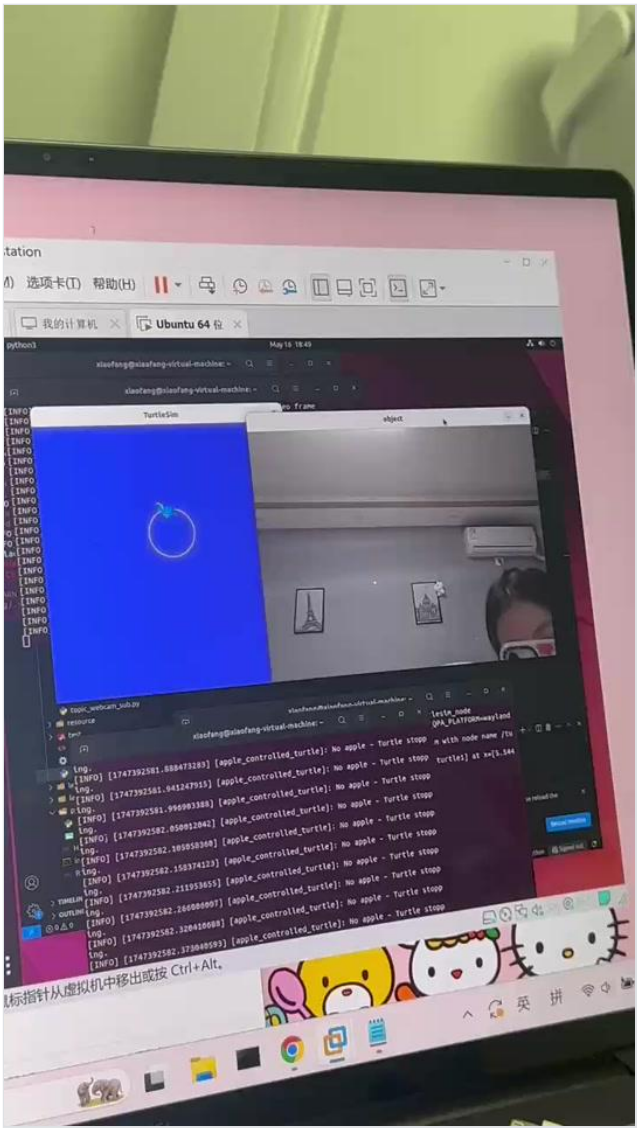
1. **重新编译工作空间**

```bash
cd ~/dev_ws  # 进入ROS2工作目录
colcon build --packages-select learning_topic  # 仅编译learning_topic包
source install/setup.bash  # 更新环境变量
```

2. **在别的的终端运行这些命令**

```bash
ros2 run v4l2_camera v4l2_camera_node #启动摄像头节点
ros2 run learning_topic apple_detector #启动苹果坚持节点
ros2 run turtlesim turtlesim_node #启动海龟模拟器
ros2 run learning_topic apple_controlled_turtle #启动苹果控制乌龟节点
```

最后的效果


选项卡(T) 帮助(H)
我的计算机
Ubuntu 64 位
TurtleSim
object
[INFO] [1747392581.888473383] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392581.941247915] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392581.996903388] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392582.050012042] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392582.105058368] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392582.158374123] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392582.211953655] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392582.266080007] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392582.320410608] [apple_controlled_turtle]: No apple - Turtle stopp
[INFO] [1747392582.373040593] [apple_controlled_turtle]: No apple - Turtle stopp
鼠标指针从虚拟机中移出或按 Ctrl+Alt。