

暨南大学本科实验报告专用纸

课程名称 机器人操作系统理论与实践 指导教师 李德平 成绩

实验项目名称 通过服务实现两个节点间的通信 实验项目编号 02

实验项目实验地点 验证 学院 智能科学与工程学院 专业 人工智能

学生姓名 王志涛 学号 2021102259 实验时间 2023 年 10 月 24 日

一、实验目的

- 1.1 熟练掌握建立工作空间、建立功能包的命令
- 1.2 熟练掌握编译功能包、建立节点的方式
- 1.3 实现一个复杂的服务通信，识别物体的位置

二、实验环境

2.1 Linux 虚拟机，版本为 Ubuntu22.0.4

三、实验内容

3.1 通过话题实现两个节点 A 直接的通信，节点 A 为客户端，节点 B 为服务器端。节点 A 请求目标物体的位置，节点 B 收到请求后，通过相机驱动节点的图像数据，返回位置给节点 A

四、实验及分析

4.1 建立工作空间

```
wzt@wzt-vpc:~$ mkdir -p ~/my_ws_service/src
wzt@wzt-vpc:~$
```

4.2 建立功能包

```
wzt@wzt-vpc:~$ cd my_ws_service/src/
wzt@wzt-vpc:~/my_ws_service/src$ ros2 pkg create --build-type ament_python learning_service
going to create a new package
package name: learning_service
destination directory: /home/wzt/my_ws_service/src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['wzt <wzt@todo.todo>']
licenses: ['TODO: License declaration']
build type: ament_python
dependencies: []
creating folder ./learning_service
creating ./learning_service/package.xml
creating source folder
creating folder ./learning_service/learning_service
creating ./learning_service/setup.py
creating ./learning_service/setup.cfg
creating folder ./learning_service/resource
creating ./learning_service/resource/learning_service
creating ./learning_service/learning_service/__init__.py
```

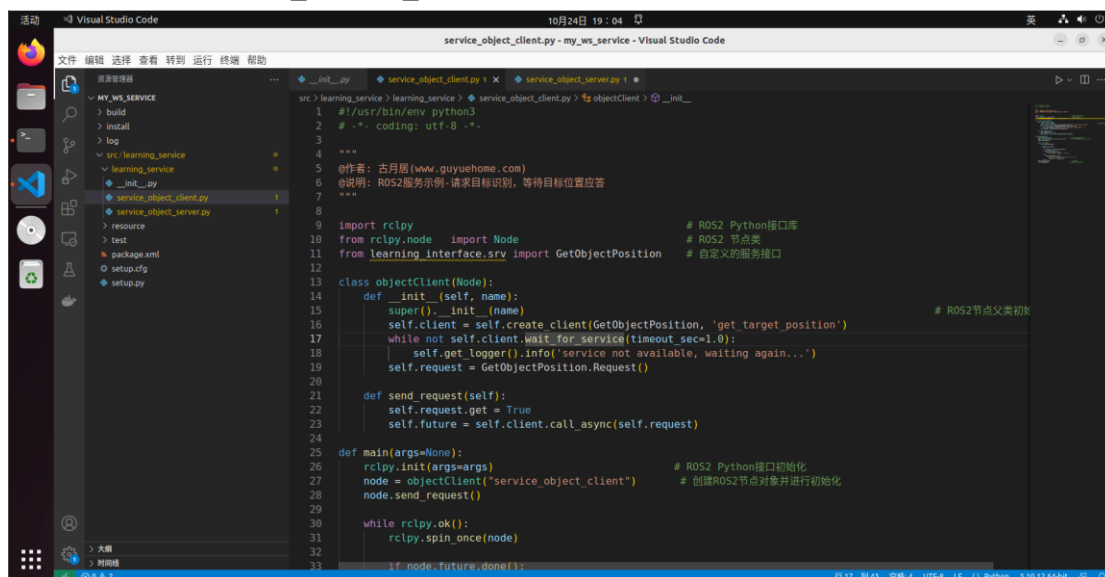
- 编译工作空间

```
wzt@wzt-vpc:~/my_ws_service$ colcon build
Starting >>> learning_service
Finished <<< learning_service [1.35s]

Summary: 1 package finished [1.69s]
```

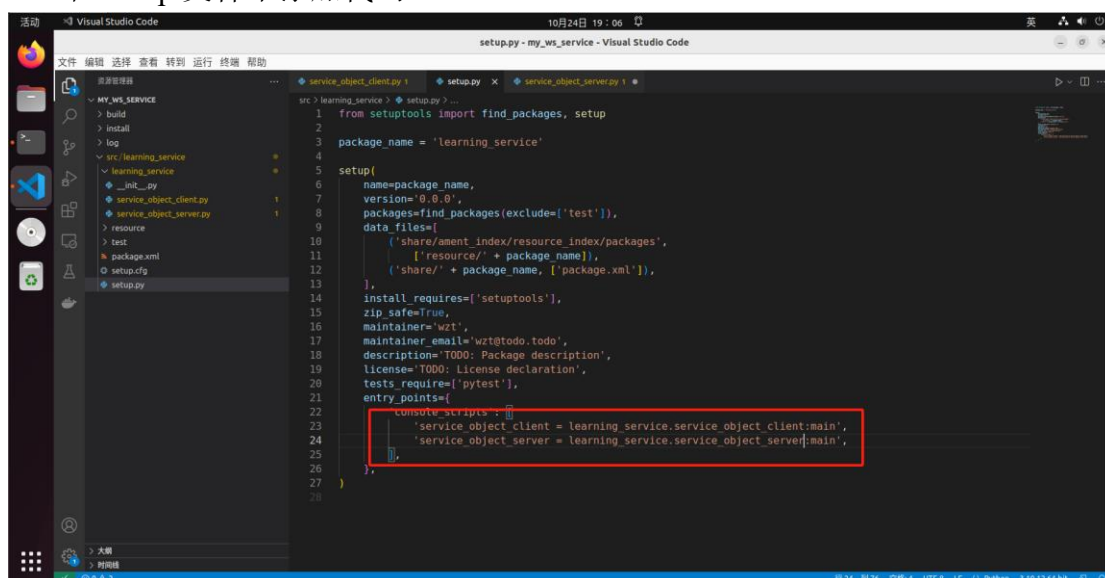
4.3 添加节点文件

- 在 learning_service 文件夹下添加两个文件，service_object_client(客户端)，service_object_server(服务端)



```
src> learning_service> learning_service ? service_object_client.py ? objectClient ? __init__
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 """
5 @作者: 古月居(www.guyuehome.com)
6 @说明: ROS2服务示例- 请求目标识别, 等待目标位置应答
7 """
8
9 import rclpy # ROS2 Python接口库
10 from rclpy.node import Node # ROS2 节点类
11 from learning_interface.srv import GetObjectPosition # 自定义的服务接口
12
13 class objectClient(Node):
14     def __init__(self, name):
15         super().__init__(name) # ROS2节点父类初始化
16         self.client = self.create_client(GetObjectPosition, 'get_target_position')
17         while not self.client.wait_for_service(timeout_sec=1.0):
18             self.get_logger().info('service not available, waiting again...')
19             self.request = GetObjectPosition.Request()
20
21     def send_request(self):
22         self.request.get = True
23         self.future = self.client.call_async(self.request)
24
25 def main(args=None):
26     rclpy.init(args=args) # ROS2 Python接口初始化
27     node = objectClient("service_object_client") # 创建ROS2节点对象并进行初始化
28     node.send_request()
29
30     while rclpy.ok():
31         rclpy.spin_once(node)
32
33     if node.future.done():
```

- 在 setup 文件中添加代码:



```
src> learning_service> setup.py ? setup.py ? service_object_server.py ?
1 from setuptools import find_packages, setup
2
3 package_name = 'learning_service'
4
5 setup(
6     name=package_name,
7     version='0.0.0',
8     packages=find_packages(exclude=['test']),
9     data_files=[
10         ('share/ament_index/resource_index/packages',
11          ['resource/' + package_name]),
12         ('share/' + package_name, ['package.xml']),
13     ],
14     install_requires=['setuptools'],
15     zip_safe=True,
16     maintainer='wzt',
17     maintainer_email='wzt@todo.todo',
18     description='TODO: Package description',
19     license='TODO: License declaration',
20     tests_require=['pytest'],
21     entry_points={
22         'console_scripts': [
23             'service_object_client = learning_service.service_object_client:main',
24             'service_object_server = learning_service.service_object_server:main',
25         ],
26     },
27 )
```

- service_object_client:

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

"""
@作者: 古月居(www.guyuehome.com)
@说明: ROS2服务示例-请求目标识别, 等待目标位置应答
"""

import rclpy  # ROS2 Python接口库
from rclpy.node import Node  # ROS2 节点类
from learning_interface.srv import GetObjectPosition  # 自定义的服务接口

class objectClient(Node):
    def __init__(self, name):
        super().__init__(name)
        self.client = self.create_client(GetObjectPosition, 'get_target_position')
        while not self.client.wait_for_service(timeout_sec=1.0):
            self.get_logger().info('service not available, waiting again...')
            self.request = GetObjectPosition.Request()

    def send_request(self):
        self.request.get = True
        self.future = self.client.call_async(self.request)

def main(args=None):
    rclpy.init(args=args)  # ROS2 Python接口初始化
    node = objectClient("service_object_client")  # 创建ROS2节点对象并进行初始化
    node.send_request()

    while rclpy.ok():
        rclpy.spin_once(node)

        if node.future.done():
            try:
                response = node.future.result()
            except Exception as e:
                node.get_logger().info(
                    'Service call failed %r' % (e,))
            else:
                node.get_logger().info(
                    'Result of object position:\n x: %d y: %d' %
                    (response.x, response.y))
            break

    node.destroy_node()  # 销毁节点对象
    rclpy.shutdown()  # 关闭ROS2 Python接口
```

● service_object_server

```

import rclpy # ROS2 Python接口库
from rclpy.node import Node # ROS2 节点类
from sensor_msgs.msg import Image # 图像消息类型
import numpy as np # Python数值计算库
from cv_bridge import CvBridge # ROS与OpenCV图像转换类
import cv2 # OpenCV图像处理库
from learning_interface.srv import GetObjectPosition # 自定义的服务接口
import threading

lower_red = np.array([0, 90, 128]) # 红色的HSV阈值下限
upper_red = np.array([180, 255, 255]) # 红色的HSV阈值上限

class ImageSubscriber(Node):
    def __init__(self, name):
        super().__init__(name) # ROS2节点父类初始化
        self.srv = self.create_service(GetObjectPosition, # 创建服务器对象（接口类型、服务名、服务器回调函数）
                                       'get_target_position',
                                       self.object_position_callback)

        self.objectX = 0
        self.objectY = 0

    def object_detect(self):
        cap = cv2.VideoCapture('/home/xiaowu/Videos/person.mp4')
        fgbg = cv2.createBackgroundSubtractorMOG2()
        while True:
            ret, frame = cap.read()
            if not ret:
                break
            fgmask = fgbg.apply(frame)
            (cnts, _) = cv2.findContours(fgmask.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
            maxArea = 0
            for c in cnts:
                Area = cv2.contourArea(c)
                if Area < maxArea :
                    (x, y, w, h) = (0,0,0,0)
                    continue
                else:
                    if Area < 1000*120:
                        (x, y, w, h) = (0,0,0,0)
                        continue
                    else:
                        maxArea = Area
                        m=c
                        (x, y, w, h) = cv2.boundingRect(m)
                        # 获取人物的位置
                        self.objectX = (int)(x + w/2)
                        self.objectY = (int)(y + h/2)
                        cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

            cv2.imshow('frame',frame)
            k = cv2.waitKey(30)&0xff
            if k==27:
                break
        cap.release()
        cv2.destroyAllWindows()

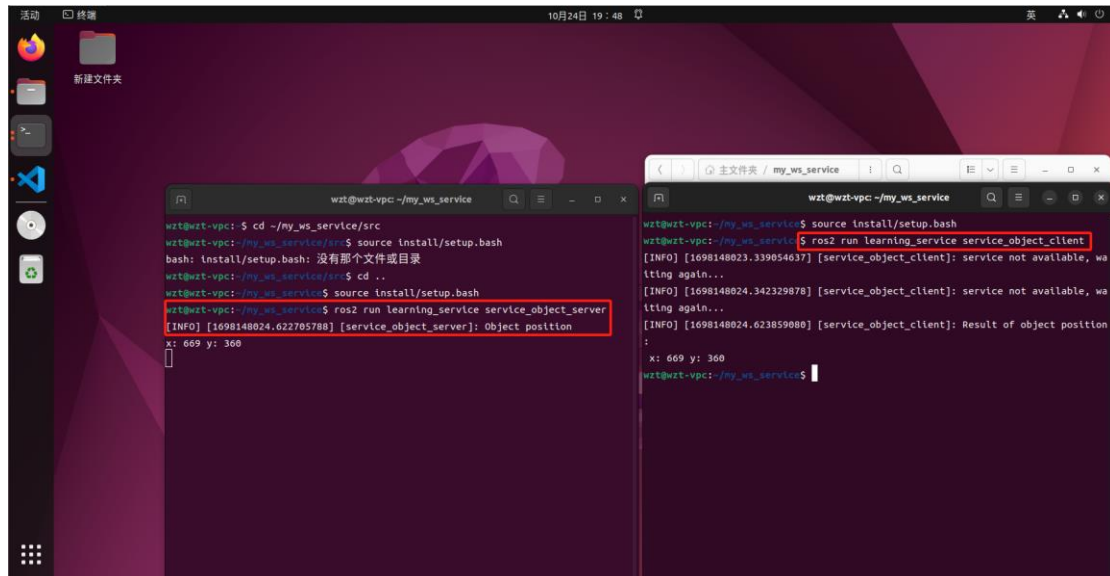
    def object_position_callback(self, request, response): # 创建回调函数，执行收到请求后对数据的处理
        if request.get == True:
            response.x = self.objectX # 目标物体的XY坐标
            response.y = self.objectY
            self.get_logger().info('Object position\nx: %d y: %d' % # 输出日志信息，提示已经反馈
                                   (response.x, response.y))
        else:
            response.x = 0
            response.y = 0
            self.get_logger().info('Invalid command') # 输出日志信息，提示已经反馈
        return response

def main(args=None): # ROS2节点主入口main函数
    rclpy.init(args=args) # ROS2 Python接口初始化
    node = ImageSubscriber("service_object_server") # 创建ROS2节点对象并进行初始化
    video = threading.Thread(target=node.object_detect) # 创建视频线程
    video.start() # 启动视频线程
    rclpy.spin(node) # 循环等待ROS2退出
    node.destroy_node() # 销毁节点对象
    rclpy.shutdown() # 关闭ROS2 Python接口
    video.join() # 清理视频线程

```

4.4 运行效果

● 输入命令行:

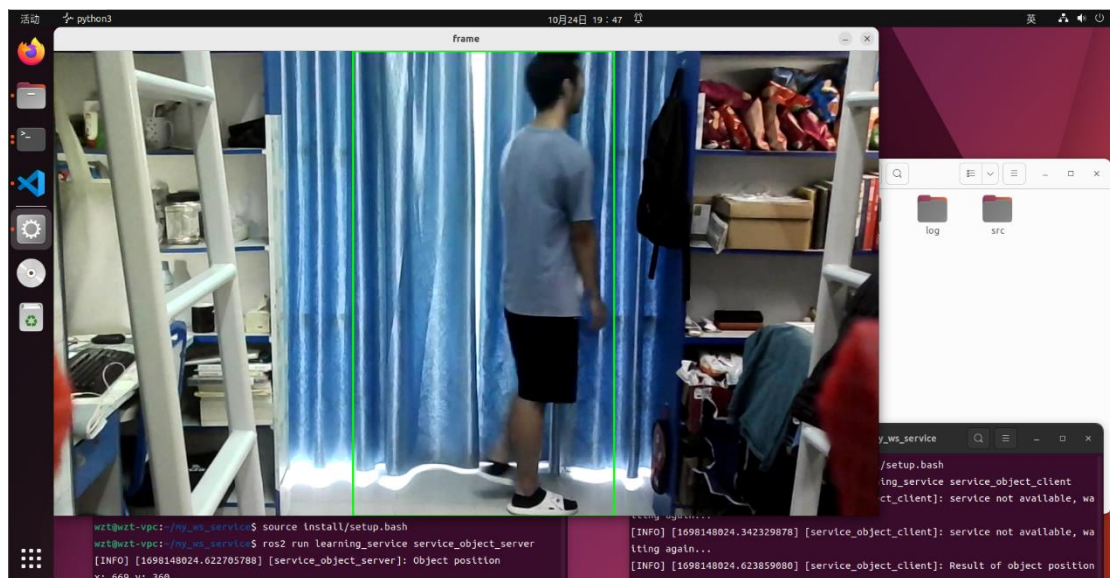


The screenshot shows a Linux desktop environment with two terminal windows. The left terminal shows the setup process: navigating to the source directory, sourcing the setup script, and running the service server. The right terminal shows the client running, which initially fails to find the service and then successfully receives the object position data (x: 669, y: 360) after the server is started.

```
wzt@wzt-vpc: ~/my_ws_service
wzt@wzt-vpc:~/my_ws_service$ cd ~/my_ws_service/src
wzt@wzt-vpc:~/my_ws_service/src$ source install/setup.bash
bash: install/setup.bash: 没有那个文件或目录
wzt@wzt-vpc:~/my_ws_service/src$ cd ..
wzt@wzt-vpc:~/my_ws_service$ source install/setup.bash
wzt@wzt-vpc:~/my_ws_service$ ros2 run learning_service service_object_server
[INFO] [1698148024.622705788] [service_object_server]: Object position
x: 669 y: 360

wzt@wzt-vpc:~/my_ws_service$
wzt@wzt-vpc:~/my_ws_service$ source install/setup.bash
wzt@wzt-vpc:~/my_ws_service$ ros2 run learning_service service_object_client
[INFO] [1698148023.339054637] [service_object_client]: service not available, waiting again...
[INFO] [1698148024.342329878] [service_object_client]: service not available, waiting again...
[INFO] [1698148024.623859080] [service_object_client]: Result of object position
x: 669 y: 360
wzt@wzt-vpc:~/my_ws_service$
```

开始运行:



客户端发送一个服务请求人的位置，服务端返回位置。

五、实验总结

5.1 服务可以实现一对多通信，服务器是唯一的，但是客户端可以是多个。

5.2 服务使用了客户端/服务器模型，是一种同步通信的方式。