2023 机器人软件工程学课程作业报告

基于语音交互任务
的情景智能 TurtleBot2

Scene Theme Orientation TurtleBot2
Based On Voice Interaction

姓名：2013365　颜铭

小组成员：

2013365 颜铭

2011023 杨宪

2011001 邓晨晖

完成时间：2023 年 6 月 3 日

摘　要

本开放性实验基于 TurtleBot2 和 YujinOs 集成机器人开发平台，融合了 turtlebot arm 机械臂，彩色深度摄像机 kinect，语音声卡，Ros 导航和建图包等功能，创新地纳入 Tensorflow 深度学习框架的物体识别 API，科大讯飞中文语言识别 SDK 进行以语音交互为核心节点，基于不同的任务和情景，通过期望意图的语言控制使得 Turtlebot2 可以搭配 Ros Noetic 上位机完成物体定位识别并且完成相应的交互动作等基础交互功能，同时在视觉、导航和机械臂控制等内容上加以扩充，使得机器人具备了完整的智能意图的反应链。最后，结合智能移动机器人软硬件技术上升的趋势，分析项目发展的不足和规划

关键词：物体识别，中文语言，情景智能，意图导向控制

目　录

一、项目背景与相关研究

1.1 项目背景

语音交互技术在智能机器人领域的应用，已经受到越来越多研究者的关注。随着科技的发展和人们生活水平的提高，越来越多的用户开始期望通过语音指令与机器人进行智能交互。

语音交互技术在机器人上的实际应用将为生活提供多功能的服务。例如，在医院、养老院等场景中，可以通过语音指令控制机器人送药、取物等操作，减轻医护人员的工作压力。在家庭生活中，用户可以通过语音指令控制机器人完成各种清扫、打扫等家务任务，提升生活质量。总之，语音交互技术在机器人领域的应用前景广阔，相信随着科技的不断进步和研究的深入，我们将会看到更加智能高效的机器人与人类之间的交互方式出现。

本项目着重设计一系列基于语音交互控制的机器人功能组块，并参考过往课程项目，探讨如何通过语音识别和并行结构，实现 turtlebot2 与人之间的语音交互和实际控制规划。该机器人项目结合了科大讯飞的语音程序和 turtlebot2 机器人平台，实现了跳舞、挥手、物体识别、小球跟随、导航定位，轨迹规划等多个功能模块的控制。用户可以通过语音指令，告诉机器人需要执行哪一个功能模块，并根据需要进行参数配置，最终实现对机器人的全面控制。

1.2 相关研究

在相关研究方面，近年来越来越多的研究者开始关注基于语音交互的机器人控制技术。例如，2020 年发表在《Speech Technology for Healthcare: Opportunities, Challenges, and State of the Art 》中，探讨了基于语音识别技术在医疗保健方面的机遇、挑战和技术水平，该研究认为，语音交互可以作为以后护理机器人实现人机共融的一种有效手段，可以方便患者与机器人进行智能交互。

同时，深度学习技术可以提高机器人对物体的识别准确率和反应速度，并通过机器学习算法不断优化模型性能。例如，2018 年张振亚等人在其研究论

文《基于 ROS 系统的家政服务机器人视觉导航识别算法》中，提出了一种基于 you Bot 机器人硬件平台和 ROS(Robot Operating System)操作系统,以 Kinect 为传感器,采用 Goog Le Net 深度学习识别模型,在 ROS 系统导航算法的支持下设计了拥有导航、定位与识别功能的家政机器人。

。

二、项目整体介绍

2.1 整体功能和框架（图）

本项目使用科大讯飞的语音程序，发布 turtlebot 机器人功能命令，是机器人执行相关程序。功能包括跳舞、挥手、物体识别、小球跟随、导航定位，轨迹规划等。以语音控制为主导，实现对各个功能模块控制。为了实现这些功能，我们需要利用 turtlebot 机器人的各个模块，并结合 ROS 的节点管理机制实现各个节点的控制和启动。同时，我们还需要编写相应的程序代码来实现机器人运动的控制、物体识别等功能。整个系统的协调与控制是由语音程序完成的，大大提高了机器人的自动化程度。

2.2 各模块之间关系与接口方法

语音模块：iat_publish 节点，向外发布消息给 voiceWords，可实现语音交互功能，调用了科大讯飞语音听写功能。

运动模块：GoForward 节点，订阅 voiceWords 消息和 odom 消息，发布 voiceWakeup 消息和 cmd_vel_mux/input/navi 消息给语音节点和机械臂节点，可实现跳舞功能。

机械臂模块：arm 节点，订阅 voiceWords 消息，发布消息给各个关节舵机，可实现招手功能。

导航模块：nav_to_point 节点，订阅 voiceWords 消息，发布运动指令，实现

定点导航和轨迹规划功能。

视觉识别模块：test_photo 节点，订阅 voiceWords 消息，实现物体识别。

小球跟随模块：ball_tracker 节点，通过识别小球，发布运动指令，实现 turtlebot 跟随小球运动。

三、具体功能原理和实现

3.1 功能一的原理和实现

核心功能为基于科大讯飞语音工具 SDK 的中文语音识别和对话，通过语音聊天作为核心通讯节点实现机器人基本的交互智能：

1. 在科大讯飞官网申请基于最新机器学习技术训练的语音听写工具 SDK，通过声学特征的筛选和循环神经网络以及集成分类器训练得到的可以识别中文并且转化为 UTF-8 编码的中文 wav 音频文件。
2. 下载 SDK 软件包到本地并解压，同时克隆开源机器人社区古月居的开源项目 https://github.com/guyuehome/guyueclass.git 到本地，进入到 robot_voice 子目录文件夹，将科大讯飞 SDK 下的 iat_record_sample 子文件夹和 libmsc.so 目录拷贝到/usr/lib/下。此外登录平台获取自动的用户端 appid 并将 robot_voice 的 iat_publis.cpp 和 voice_assistant.cpp 对应的收发接口保留报头，然后替换为自己的 appid。然后保留原有的 CmakeLists 进行 catkin build 编译。注意需要静态连接 sound 库和 pthread 库到动态连接 libmsc.so
3. iat_publish.cpp 承担的是语音听写的任务，创建订阅器订阅/voiceWakeup 话题的唤醒信号，然后发布/voiceWords 话题为识别到的中文编码汉字字符串。

   一般流程是使用 rostopic pub /voiceWakeup std_msgs/String “data: ‘any_string’ ”进行唤醒，然后可以对着麦克风说话，联网完成在线识别后将会对识别结果进行发布。
4. voice_assistant 承担的是语音助手的功能，运行节点前需要保证已经启动了/voiceWords 的发布节点确保可以获取语音识别的结果，之后可以通过

正则表达式处理，通过字符子串查找函数提取关键词，也即可以产生对话的效果。当然这种逻辑处理是单程和有限的，其余情况下语音会退化为复述词句。它和 iat_publish 发布的/voiceWords 话题共享的是 std_msgs/String 数据结构的信息。

5. 其他需要关注的科大讯飞语音听写 SDK 的功能包还有 tts_sample 和 ise_sample 等

   由于语音识别是核心节点，同时为了实现发声，其实可以手动发布/voiceWords 数据使得机器人可以在 voice_assistant 订阅到需要表达的内容。此外，可以将移动代码中的控制语言将 pocketsphinx 的英文语音替换为中文语言，实现“前进、后退和左移、右移”等操作。

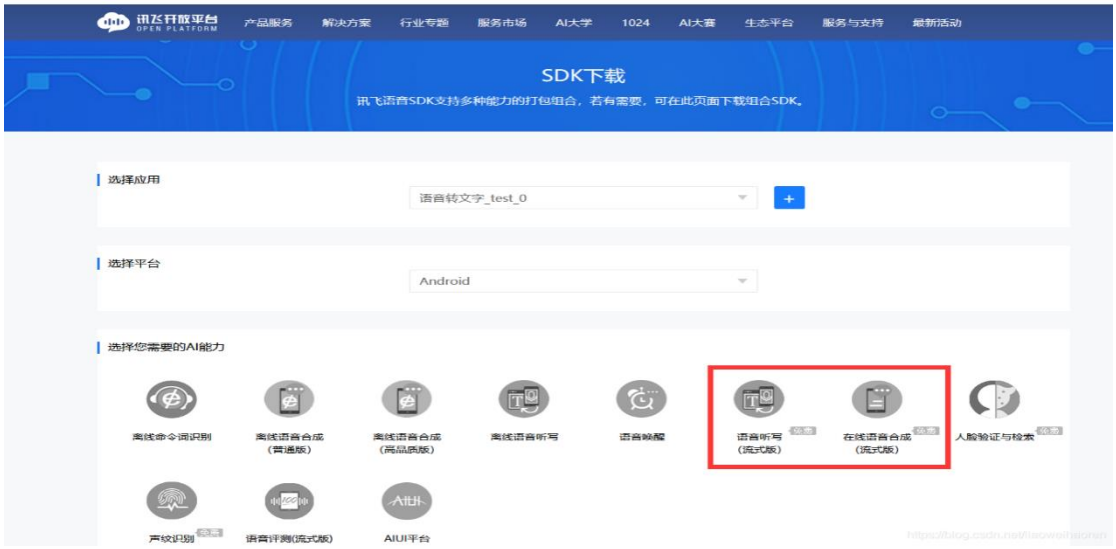

图 3.1.科大讯飞语音识别申请页面

这里我们在移动代码 GoToForwad.py 中拓展了“跳舞”这一功能，可以往前冲刺一段，模拟跳跃效果后开始左右摇头摆动的效果。也同时订阅/odom 话题获取里程计四元数数据转化的欧拉角转角信息。在转动一定角度使机器人意识到并发出语音和相应的机械臂动作。

同时由于移动功能的代码是关联到底盘启动模块最近的脚本，综合语音传输为总线的通讯，基本依靠的是循环节点的定期唤醒。因此，在节点未异常关闭前持续向/voiceWakeup 话题发布’any string’的唤醒信号

3.2 功能二的原理和实现

功能二为基于 Tensorflow Objection Detection Api 的视觉物体识别，可以识别物体同时基于定位框给出方位:

1、首先安装 cuda 和 tensorflow-gpu.。这里配置成功使用的是 2.4.4 版本。之后搜索 tensorflow model 克隆到本地，安装 protobuf 和 pycoco 工具包，按照命令执行 setup.py 执行本地预编译安装。python3 -c “import obeject_detection”并 python3 /object_detection/tf2_demo.py 测试即可。

图 3.2.TensorFlow 学习框架

2、下载 keras 的 coco 预训练权重文件，与测试用的图片。可以直接通过 html 连接下载也可以网络端下载解压到本地。之后配置路径导入 ndarray 格式的图像运行测试代码

```
    image_np = load_image_into_numpy_array(image_path)

    input_tensor = tf.convert_to_tensor(image_np)
    # The model expects a batch of images, so add an axis with
`tf.newaxis`.
    input_tensor = input_tensor[tf.newaxis, ...]

    # input_tensor = np.expand_dims(image_np, 0)
    detections = detect_fn(input_tensor)

    # All outputs are batches tensors.
```

```
    # Convert to numpy arrays, and take index [0] to remove the batch
dimension.
    # We're only interested in the first num_detections.
    num_detections = int(detections.pop('num_detections'))
    detections = {key: value[0, :num_detections].numpy()
                   for key, value in detections.items()}
    detections['num_detections'] = num_detections

    # detection_classes should be ints.
    detections['detection_classes']
=  detections['detection_classes'].astype(np.int64)

    image_np_with_detections = image_np.copy()

    viz_utils.visualize_boxes_and_labels_on_image_array(
          image_np_with_detections,
          detections['detection_boxes'],
          detections['detection_classes'],
          detections['detection_scores'],
          category_index,
          use_normalized_coordinates=True,
          max_boxes_to_draw=200,
          min_score_thresh=.30,
          agnostic_mode=False)

    plt.figure()
    plt.imsave('image{}.png'.format(cnt),image_np_with_detections)
    print('Done')
```

代码 3.1 物体识别调用

可以获得如下预测图像:

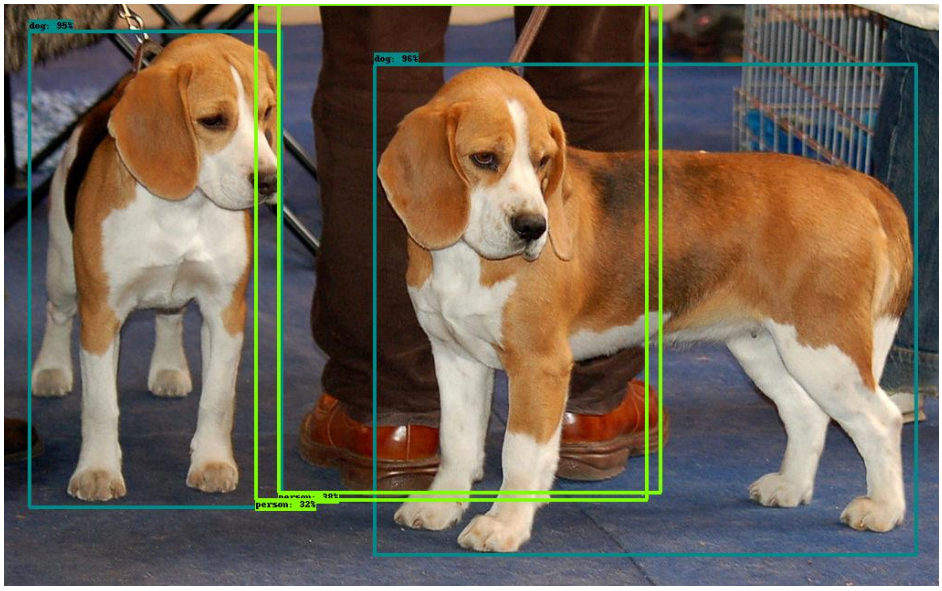

图 3.3 物体识别测试结果

因此，我们可以联合深度学习框架和 rospy 混合编程。

通过发出”你看到了什么？”的语音信号控制有条件地订阅 kinect 相机的 rgb/image_raw 话题下的 Image 格式的图片信息，得到需要预测的图片并获取预测类别编号，方框和置信度信息。

注意这里需要事先把编号和对应类别存储到相应的字典中方便索引。之后对置信度大于 0.6 的物体收集，并且根据方框的左上和右下四点信息定位到物体中点，判断是否超过图片大小的一半来确定是在机器人的左手还是右手边。

至此我们获得类别信息和方位信息，再生成关于/voiceWords 的发布器实现语音和视觉的交互。此外也尝试了订阅深度信息，处理四维数组获取距离信息以获取更准确的相对视觉信息。

预测的流程在于 ImageRecogniton.py 脚本可以订阅 rgbd 数据传输的 kinect 话题，同时订阅/voiceWords 话题按照语音需求可以得到触发信号完成相应的处理。最后实时预测结果如下:

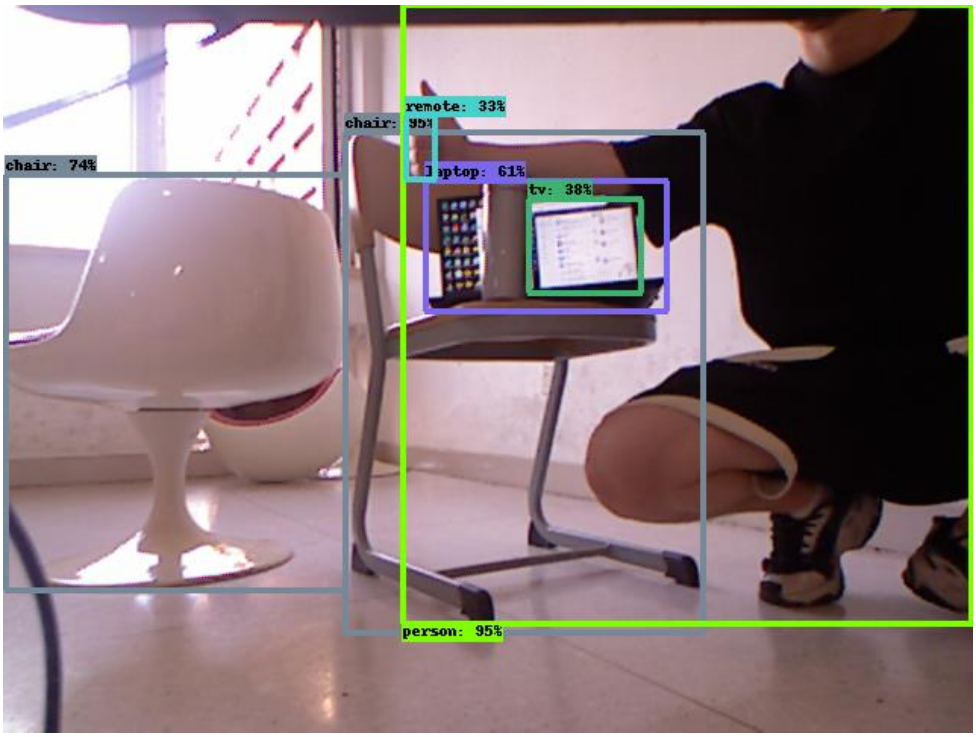

图 3.4 物体识别测试结果

此外，我们还设计了小球跟随的功能，通过 opencv 的 findcontour 函数提取球形轮廓，并指定色域为黄色也即对应一般常用的网球实现捕捉位置，然后划分网格后定位偏差位置与相关的激励速度，**最终实现利用小球引导然后灵活变向的机器人视觉移动功能**

3.3 功能三的原理和实现

功能三基于蒙特卡洛定位和 slam 建图的定点导航，使用 MoveBase 组进行多点的规划并执行几何任务。

首先融合建图导航过程使用的 amcl_demo 自适应蒙特卡洛定位和 gmapping 基于粒子滤波的算法。

进一步。根据建图的初始点坐标为[1.0,0.0]在教室外门口的世界坐标，以及运动轴方向，可以确定规划的目标点的 x 和 y 值均大于 0。下一步可以任选几组点，一样的关联中文语言控制，执行去第几个点的任务，最终得到满足预设定点控制和轨迹规划的路线信息。因此我们可以设置几何变换，如矩形和五芒星等形状。未确保节点通讯异步转同步的非阻塞，这里再每次导航定位中使用的是“启动导航”的控制命令。

此外，可以使用 tf 获取当前坐标并通过相对变化的斜线和旋转平移实现更加丰富的几何变换。比如绘制三角形只需 45 度斜运动接上 45 度运动即可。

3.4 功能四的原理和实现

功能四位基于 dynamixel 舵机定点控制器控制的固定机械臂构型的指挥。

和集成功能的目标一致，联合中文语言控制，实现机械臂的“招手”和“摆一下手”的机械臂构型。其中实验所用的机械臂为 turtlebot arm。**从下到上**分为五个 dynamixel 电机，分别对应 1-5 个关节。

因此，招手我们可以设置中间的 3 关节下垂。由于使用的是定点控制器。我们需要 echo /<dynamixel controller name>/state 话题获取到运动到理想点位的测距信息。最后使用代码控制在/command 话题上发布预期位置。由于定点控制器使用的是 PID 控制，因此参数较为敏感，在设计实验同时发现了其超调量较大。因而我们最后的对准效果归结到调一个较小的差值以期望实现运动期望位姿的相对位置的效果。

同样地，摆一下手，可以是 1 关节运动，将 5 关节的夹爪设置为一个 2.8 上下浮动 0.4 的定点弧度，也可以实现得到夹爪的展开。但是夹取这样夹爪反复闭合的功能并没有使用多线程成功实现。再加上考虑到重力的复杂因素的影响，在设计实验上没有进一步的地展开探究关节往复运动。

3.5 功能五的原理和实现

功能五为将上述所有功能集成到 launch 文件中

先导入加载数据流和算法特征的 launch 文件。最前面准备中启动底盘的 roslaunch turtlebot_bringup minimal.launch 文件，之后在运行 amcl_demo.launch 中启动了 kinect 摄像机获取深度传感数据。并且打开 rviz_launcher 的部署资源实现导航地图可视化。

启动节点中，由于语音听写和语音助手是全局核心，在第一位启动，然后需要循环地唤醒语音识别保持异步的通讯。因此，将 goforward 的移动唤醒节点放在之后，其余节点位置任意。考虑到视觉识别的 test_photo 节点需要代入 1 个 G 的预训练文件，大概使用 30s 时间，因此应该放到相对考前的位置，完整的 turtlebot2demo.launch 文件如下:

```
    <?xml version="1.0" encoding="UTF-8"?>
<launch>
    <include file="$(find turtlebot_bringup)/launch/minimal.launch"/>

    <include file="$(find
my_dynamixel)/launch/start_tilt_controller.launch"/>
    <include file="$(find
my_dynamixel)/launch/controller_manager.launch"/>
```

```
    <include file="$(find
turtlebot_navigation)/launch/amcl_demo.launch"/>
    <include file="$(find
turtlebot_rviz_launchers)/launch/view_navigation.launch"/>


    <node name="iat_publish" pkg="robot_voice" type="iat_publish"
required="true"  output="screen"/>
    <node name="GoForward" pkg="my_turtlebot2_demo"
type="GoToForward.py" required="true" output="screen"/>
    <node name="test_photo" pkg="my_turtlebot2_demo"
type="ImageRecognition.py" required="true" output="screen"/>
    <node name="voice_assistant" pkg="robot_voice"
type="voice_assistant" required="true" output="screen"  />



    <!-- <node name="ball_tracker" pkg="my_turtlebot2_demo"
type="ball_follower.py" required="true" output="screen"/> -->

    <node name="nav_to_point" pkg="my_turtlebot2_demo"
type="navigation_voice_control.py" required="true" output="screen"/>
     <!-- <node name="navTurtlebot" pkg="my_turtlebot2_demo"
type="navigation_control.py" required="true" output="screen"/> -->
    <node name="arm" pkg="my_turtlebot2_demo" type="ArmDemoControl.py"
required="true" output="screen"/>
    <node name="ball_tracker" pkg="my_turtlebot2_demo"
type="ball_follower.py" required="true" output="screen"/>
    <!-- <node name="object_detect" pkg="my_turtlebot2_demo"
type="object_detect.py" required="true" output="screen" /> -->
</launch>
```

代码 3.2 turtlebot2demoo.launch

四、个人完成的主要工作（不少于 3 页）

4.1 主要工作一

颜铭作为策划和领导者，负责将实验环境迁移到 ros noetic 且安装智能软件依赖以实现较为完备的功能设计

- **如何将实验环境迁移到 Ros Noetic**

对于项目软件开发环境从 Melodic 迁移到 Noetic，主要是以下几个部分:

（1）按照如下的

https://gist.github.com/jeremyfix/0c5973aba508ee8b6e8d3c3077c6db1e

[ros noetic turtlebot2]的 shell 命令指导安装需要的包

其主要按照一下在终端执行的命令

```
mkdir ~/catkin_ws
cd catkin_ws
mkdir -p src
catkin_make
cd src

# 将需要使用的开源包克隆到本地
git clone https://github.com/turtlebot/turtlebot.git
git clone https://github.com/turtlebot/turtlebot_msgs.git
git clone https://github.com/turtlebot/turtlebot_apps.git
git clone https://github.com/turtlebot/turtlebot_simulator

# 原问题 https://github.com/yujinrobot/kobuki/issues/427 的解答
git clone https://github.com/yujinrobot/yujin_ocs.git
# 移除所有的'yocs_cmd_vel_mux', 'yocs_controllers'和
'yocs_velocity_smoother'选项
mv yujin_ocs/yocs_cmd_vel_mux yujin_ocs/yocs_controllers
yujin_ocs/yocs_velocity_smoother .
rm -rf yujin_ocs

#安装电池驱动包
git clone https://github.com/ros-
drivers/linux_peripheral_interfaces.git
mv linux_peripheral_interfaces/laptop_battery_monitor ./
```

```
rm -rf linux_peripheral_interfaces
# You need to MANUALLY, for now, apply the changes proposed in
https://github.com/ros-drivers/linux_peripheral_interfaces/pull/18


#  kobuki 的 melodic 分支由于是 c++代码构建依然可以使用
git clone https://github.com/yujinrobot/kobuki.git

sudo apt install liborocos-kdl-dev -y
sudo apt install ros-noetic-joy
rosdep install --from-paths . --ignore-src -r -y

# catkin 构建包
cd ..
catkin_make

# LDS-01 激光驱动安装，可选
sudo apt install ros-noetic-hls-lfcd-lds-driver -y

# 3d 传感器软件安装，可选
sudo apt install ros-melodic-openni2-launch -y
sudo apt install ros-melodic-depthimage-to-laserscan -y

echo source "$HOME/catkin_ws/devel/setup.bash" >> ~/.bashrc
sudo adduser $USER dialout
```

（2）按照 https://aibegins.net/2020/11/22/give-your-next-robot-3d-vision-kinect-v1-with-ros-noetic/[freenect noetic]的提示安装可以在 ros noetic 上使用的

kinect-freenect 摄像机驱动：

安装依赖项

```
sudo apt-get install git-core cmake freeglut3-dev pkg-config build-essential libxmu-dev libxi-dev libusb-1.0-0-dev
```

本地安装 libfreenect 库

```
git clone git://github.com/OpenKinect/libfreenect.git
```

编译和安装

```
cd libfreenect
mkdir build
cd build
cmake -L ..
make
sudo make install
sudo ldconfig /usr/local/lib64/
```

更改 kinect 使用方式，不需每次都用 sudo

```
sudo adduser $USER video
sudo adduser $USER plugdev
```

添加 kinect 设备规则

```
sudo nano /etc/udev/rules.d/51-kinect.rules
```

粘贴以下代码后按下”ctrl+q”保存

```
# ATTR{product}=="Xbox NUI Motor"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e",
ATTR{idProduct}=="02b0", MODE="0666"
# ATTR{product}=="Xbox NUI Audio"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e",
ATTR{idProduct}=="02ad", MODE="0666"
# ATTR{product}=="Xbox NUI Camera"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e",
ATTR{idProduct}=="02ae", MODE="0666"
# ATTR{product}=="Xbox NUI Motor"
```

```
SUBSYSTEM=="usb", ATTR{idVendor}=="045e",
ATTR{idProduct}=="02c2", MODE="0666"
# ATTR{product}=="Xbox NUI Motor"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e",
ATTR{idProduct}=="02be", MODE="0666"
# ATTR{product}=="Xbox NUI Motor"
    SUBSYSTEM=="usb", ATTR{idVendor}=="045e",
ATTR{idProduct}=="02bf", MODE="0666"
```

我们还要安装 audio 驱动，进入 freenect 文件夹并打开终端运行

```
python3 src/fwfetcher.py
```

然后拷贝 audio.bin 到特殊的工作空间环境下

```
sudo cp src/audios.bin /usr/local/share/libfreenect
```

运行以下命令即可查看是否安装正常

```
freenect-micview

freenect-glview
```

（3）修改 dynamixel_motor 文件夹下的 python2 错误，主要工作为

1.使用 os.path.join 添加相对路径可以识别 package

2.print 修改为 print()

3.throw e 改为 throw Exception as e

此外在项目实现中，颜铭的主要工作包括物体识别功能、小球跟随功能、语音交互功能和总 launch 文件的代码编写。我们的目标是实现一个基于机器人的智能交互系统。

（4）实际上，在整体迁移中还碰到了一些安装包缺失的问题，比如 cv 和 qt 环境冲突，或者 freenect 的某个组件缺失。遇到如上问题时候可以使用 ros-noetic-debug 报错缺失的包名*进行安装，否则需要进一步在 stackoverflow 和 csdn 等社区寻找更为专业的解答。

对于任务实现，项目中我们需要实现一个当发布“你看到了什么”指令时，机器人能够识别摄像范围内物体种类，并通过语音告知提问者。为了实现这个功能，我们首先需要从机器人的摄像头中获取图像数据，并使用物体识别算法对图像进行处理。接着，我们需要根据识别结果，使用语音合成技术将机器人的回答转化成语音输出。

除了物体识别功能，我们还需要实现小球跟随功能。这个功能的实现需要先使用摄像头捕捉到小球的图像，并使用图像处理技术对小球进行识别和跟踪。接着，我们需要使用机器人运动控制技术，让机器人通过传感器数据来跟随小球的运动。

在项目的实现过程中，我们还需要考虑到语音交互功能的实现。我们利用科大讯飞语音听写功能，使机器人能够与用户进行中文语音交互。这个功能的实现需要借助语音处理技术和自然语言处理技术来实现。

最后我们需要编写总 launch 文件，将所有的功能模块集成在一起。在编写总 launch 文件的过程中，我们需要注意各个模块之间的依赖关系和启动顺序，确保系统能够顺利运行。

在项目测试阶段，我主要参与了各个功能模块的测试工作，包括物体识别、小球跟随、语音交互等功能的测试以及总系统的集成测试。我们不断进行调试和改进，最终成功实现了我们预期的功能。

4.2 主要工作二

杨宪的工作职责是项目资料整理和代码调整的执行者：

他负责编写机器人的运动模块跳舞和招手功能的代码，以及将各个功能模块整合到程序总 launch 文件中。

具体来说，在跳舞模块中，当语音模块发布“跳舞”指令时，机器人会进行跳舞动作，先向前俯冲，再开始不断摇摆。在这个过程中，杨宪遇到了转角控制的问题。他通过订阅里程计 odom 中的 yaw 信息，实现了对转角大小的控制。这种方法能够让机器人的跳舞动作更加精准和优美。

在招手模块中，当语音模块发布“招手”指令时，机械臂的各关节会运动到指定位置，从而完成招手的动作。然而，杨宪发现舵机的控制经常出现超调现象，需要进行优化。因此，他将舵机的 pos 位置设置在各关节收敛的位置，通过该方法，实现了对舵机的有效控制，进一步提高了机器人招手功能的稳定性和可靠性。

除了以上工作之外，杨宪还参与了各个功能模块的测试工作，不断改进和优化程序，帮助机器人实现更加智能、高效的交互功能。

总之，作为项目资料整理和代码调整的执行者，杨宪的工作贡献不可或缺。他通过不断探索和实践，为机器人的跳舞和招手功能提供了优秀的代码支持，并帮助实现了一个基于机器人的智能交互系统。

4.3 主要工作三

邓晨晖是一名 ROS Melodic 的完整实验员，他的主要工作是导航模块的代码编写。

该模块可以在语音模块发布“启动导航”指令时启动，并在语音模块发布“第一点”、“第二点”、“第三点”、“第四点”时将 turtlrbot 机器人移动到指定点，实现定点控制功能。此外，他还参与各模块功能测试工作，保证机器人能够顺利地执行各种任务。

导航模块中遇到的主要问题是初始点定位出现误差，导致机器人无法准确到达指定的点位。此外，设定点位 pos 时也容易出现位置不可达的现象，需要仔细设置点位以确保机器人能够到达。

为了解决这些问题，邓晨晖建议可以采用增加定位系统准确度的方法，例如使用更先进的 SLAM 算法或更精确的传感器。此外，设定点位 pos 时也可以考虑机器人的最大移动距离和避障问题，以确保指定点位可达。

除了导航模块之外，邓晨晖还参与了轨迹规划模块的编写。当语音模块发布“矩形”时，turtlrbot 机器人会按照矩形轨迹运动，实现轨迹规划功能。在这个模块中，经常会出现轨迹上的点不可达的情况，需要仔细设定轨迹上的点以确保机器人能够顺利移动。

为了解决轨迹规划中的问题，邓晨晖认为可以通过增加点位数量的方法来优化轨迹规划结果。此外，还可以采用遗传算法等优化算法来实现更加准确的轨迹规划。

总之，邓晨晖在机器人导航和轨迹规划方面拥有丰富的经验和技能，在开发过程中可以为团队提供宝贵的支持和建议。他的不懈努力和创造性的思维为整个机器人系统的优化和改进做出了重要贡献。

五、项目总结与展望

5.1 项目总结

本项目实现了基于语音交互的机器人控制系统，利用深度学习对物体识别功能进行优化、利用物体识别和 turtlebot 的运动学模块对小球跟随功能进行了设计、利用 turtlebot 的导航模块实现了导航定位和轨迹规划，同时还设计了其他多个功

能模块。我们利用 ROS 节点管理机制，将各个模块集成在一起，并使用科大讯飞语音程序来实现语音交互功能。通过不断地调试和优化，最终成功实现了预期的功能。

在整个项目开发过程中，我们重点关注了机器人技术与语音处理技术的结合应用，同时利用物体识别功能，探索了一种新的机器人交互方式。通过语音指令控制机器人，使得用户能够更加便捷地进行智能交互，为机器人应用场景的拓展提供了新的思路。

但同时也存在一些问题和不足之处，比如在物体识别和小球跟随方面，精度还有待提高；在导航定位和轨迹规划方面，还需要进一步完善算法和技术。未来的研究工作可以从这些方面入手，提高系统的性能和准确率。

5.2 项目展望

基于语音交互的机器人控制技术具有广阔的应用前景，未来可以在医疗、养老、家庭服务等领域得到广泛应用。在未来的研究中，我们可以探索更加复杂和智能化的机器人交互模式，例如情感表达、自主学习和社交交流等方面。

此外，在技术层面上，我们将继续深化对语音处理技术和机器人技术的融合，在物体识别、目标跟随、路径规划等方面进行进一步的优化和提升。同时，也将关注系统的安全性和可靠性问题，避免出现故障和意外事故。

总之，基于语音交互的机器人控制技术在未来的发展中具有广阔的前景和潜力，相信通过不断的研究和开发，我们将会看到更加智能、高效和便捷的机器人交互系统出现。

参考文献

[1] 胡春旭. ROS 机器人开发实践[M]. 机械工业出版社, 2018.

[2] Latif S, Qadir J, Qayyum A, et al. Speech technology for healthcare: Opportunities, challenges, and state of the art[J]. IEEE Reviews in Biomedical Engineering, 2020, 14: 342-356.

[3] Mustamo P. Object detection in sports: TensorFlow Object Detection API case study[J]. University of Oulu, 2018, 1.