2024 机器人软件工程学课程作业报告

项目名称：基于示教学习的复杂场景导航

英文名称：Navigation in Complex Scenarios Based on Demonstration Learning

姓名：2111381 石若川

小组成员：

2111176 柴　颖

2110705 姜雨杉

2111381 石若川

完成时间：2024 年 6 月

摘　要

近年来，随着人工智能和机器人技术的迅速发展，机器人在各个领域的应用日益广泛。然而，机器人的智能水平仍远不及人类。在复杂和动态环境中实现高效、智能的自主导航仍然面临诸多挑战。模仿学习作为一种更高效且低成本的学习方式，提供了一种潜在的解决方案，通过模仿人类的示范行为，机器人能够快速学习和适应复杂的导航任务。

基于此，本项目提出了一种结合语音交互、行人跟踪、建图导航、视觉识别和机械臂控制的综合解决方案。本项目通过示教学习实现复杂场景下的机器人导航，包括三个阶段：建图阶段、示教阶段和物品抓取阶段。建图阶段，使用建图算法对场景进行建图；示教阶段，机器人跟随示教者，并通过语音命令记录关键点位坐标；物品抓取阶段，机器人根据示教路径自主导航到目标位置，并通过视觉反馈进行位置调整，完成物品抓取。这种方法不仅提高了导航的精度和效率，还增强了机器人的环境适应能力和人机协作能力。

总结来看，本项目展示了模仿学习在机器人导航中的应用潜力，通过示教学习和多模块协同工作，机器人能够快速学习和适应复杂环境中的导航任务。未来，随着技术不断进步，模仿学习在机器人自主导航中的应用前景将更加广阔，有望实现更高水平的智能自主导航。

关键词：示教学习；模仿学习；建图导航；目标检测；PID 控制。

目　录

一、项目背景与相关研究

1.1 项目背景

近年来，人工智能和机器人技术的飞速发展，推动了机器人在各个领域的广泛应用。从工业制造到家庭服务，从医疗辅助到自动驾驶，机器人技术的进步显著提升了各行各业的效率和生产力。具身智能和人形机器人的崛起更是为机器人未来的发展描绘了宏伟蓝图。然而，尽管机器人在自主导航、语音交互、视觉识别和机械臂控制等方面取得了巨大进展，尤其是在大模型技术的支持下，机器人的功能愈加完善，初步具备了决策智能，然而其智能水平仍远不及人类。要实现机器人在复杂和动态环境中与人类相媲美的高效、智能自主导航，仍然面临诸多严峻挑战。

当前的机器人导航技术主要依赖于详细的环境建模和精确的路径规划。这些方法通常需要机器人在未知环境中构建地图（SLAM，Simultaneous Localization and Mapping）并进行定位。然而，在实际应用中，环境的动态性和复杂性使得这些方法难以完全满足需求。机器人需要处理不断变化的障碍物和动态的人群，这对环境建模和路径规划提出了极高的要求。此外，精确的环境建模和路径规划往往需要大量的计算资源和时间，难以满足实时导航的需求。

学习类算法的火热发展为问题的解决提供了新思路。机器人使用学习的方式模仿人类获得知识与技能的过程，在学习所得先验知识的加持下更高效地完成指定任务。然而，传统的强化学习方法虽然可以帮助机器人在复杂环境中自主探索并学习导航策略，但其高昂的探索成本和长时间训练需求在实际应用中具有显著的局限性。特别是在任务复杂、探索代价高或环境不允许大量试错的情况下，强化学习的应用变得更加困难。因此，我们或许可以采取一种更为高效且低成本的学习方式，即模仿学习。

我们提出了一种基于示教学习的复杂场景导航技术，旨在通过模仿人类行动路径，实现机器人在复杂场景下的自主导航。

这一创新思路的提出将有望为今后机器人导航功能的优化提供一种新范式。我们不再需要过度依赖算法性能的调优，而是将重心转向让机器人通过自主行人跟踪来积累实际场景中的经验数据。通过这种方法，机器人能够在复杂环境中实时学习并记忆人类的导航行为。当再次面对类似任务时，机器人能够迅速调取记

忆中相关场景的关键信息，实现高效稳定的导航。这种基于经验积累和模仿学习的方法，有望显著提升机器人在复杂环境中的适应性和实用性。

1.2 相关研究

模仿学习作为一种革新性的机器学习方法，与传统的强化学习有着显著的区别。其核心思想在于通过观察和模仿示范者的行为来学习执行特定任务，而非依赖于试错和奖励的反馈机制。模仿学习使得机器人能够直接复制示范者的动作和决策，从而快速获取所需的知识和技能。这样的方法显著加速了学习过程，并大幅减少了可能发生的不必要错误，使得系统在实际应用中更加可靠和稳定。

近年来，模仿学习在多个领域取得了显著的成果，展示了其在机器人技术中的巨大潜力和应用前景。例如，OpenAI 公司利用模仿学习训练机械手玩魔方。通过精准的观察和模仿人类在解决魔方时的手部动作，极大地提高了机械手的灵活度和操作精度，使得机器人能够有效应对复杂的操作任务。

在家庭服务机器人领域，斯坦福大学的华人团队采用模仿学习技术，让 Mobile ALOHA 机器人学习了多项家庭服务技能，如洗衣、做饭、打扫等[1]。通过观察和模仿人类在家庭环境中的行为，机器人能够自主地执行这些任务，不仅提升了家庭生活的便利性，也展示了模仿学习在实现复杂机器人任务中的广泛适用性。

模仿学习在机器人技术中的应用，不仅显著加速了机器人学习复杂任务的速度，而且极大地提升了其在实际应用中的可靠性和稳定性。随着技术的不断进步，模仿学习在机器人技术领域的应用前景愈发广阔，通过仿生的方法，机器人精准地观察和模仿人类行为，高效习得各项技能，展现出强大的潜力和无限的可能性。

二、项目整体介绍

2.1 整体功能和框架（图）

本项目主要实现了一种基于示教学习的复杂场景导航技术，旨在通过模仿人类行动路径，实现机器人在复杂场景下的自主导航。具体任务流程分为三个阶段：

- **建图阶段：**使用建图算法，对场景地图进行建立。

- **示教阶段：**机器人跟踪前方的示教者，示教者通过语音向机器人发出保存当前点位坐标信息的指令，机器人存储当前点位坐标并通过语音回复示教者。
- **物品抓取阶段：**机器人根据多个保存的点位坐标，模仿示教者的路径，自主导航至终点位置。在终点位置，机器人根据视觉反馈进行 PID 位置调整，完成目标物体的抓取。最后机器人原路返回至起点，将目标物体带回起点位置。

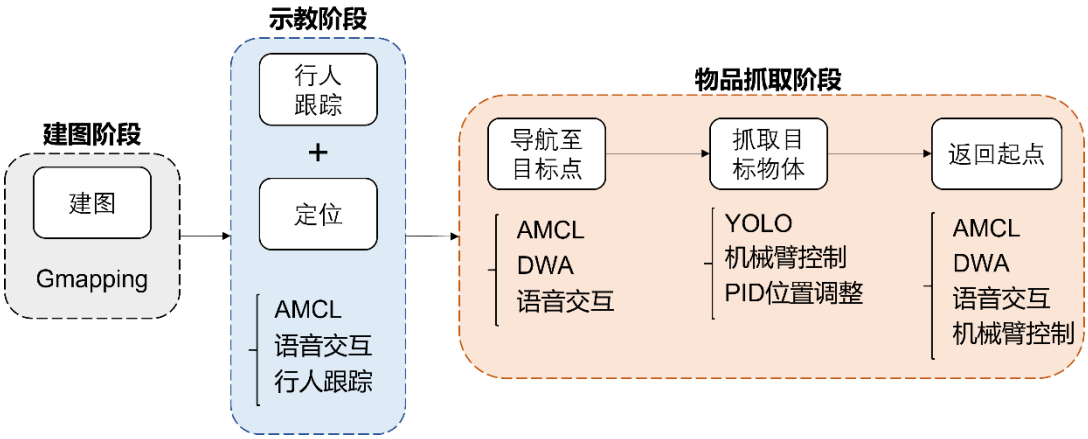

图 1.1 整体框架图

2.2 各模块之间关系与接口方法

项目中包括以下五大模块：**语音交互模块、行人跟踪模块、建图导航模块、视觉识别模块和机械臂模块**，具体的模块关系与接口方法如下：

- 建图阶段中，单独调用建图导航模块完成对搭建场景的建图。
- 示教阶段中，调用语音交互模块、行人跟踪模块和建图导航模块。将行人跟踪模块与建图导航模块相结合，在跟踪行人的同时获取机器人的实时坐标。语音交互模块会将语音识别的结果发布至/voiceWords 话题中。当识别到关键字时，将播放合成的语音并记录下当前的坐标信息。
- 物品抓取阶段中，调用语音交互模块、建图导航模块、视觉识别模块和机械臂模块。首先通过建图导航模块到达目标物体的位置，再通过视觉识别模块调整机器人的抓取位置，利用机械臂模块完成目标物品的抓取，最后通过语音交互模块在完成抓取和返回起始点后播报合成的语音。

三、具体功能原理和实现

3.1 语音交互的原理和实现

语音作为最自然便捷的交流方式，一直是人机通信和交互最重要的研究领域之一。项目中语音交互功能的实现主要依托语音识别与语音合成两个功能。

语音识别的本质就是将语音序列转换为文本序列，所要解决的问题是让计算机能够“听懂”人类的语音，将语音中包含的文字信息提取出来。语音识别的流程大体为：

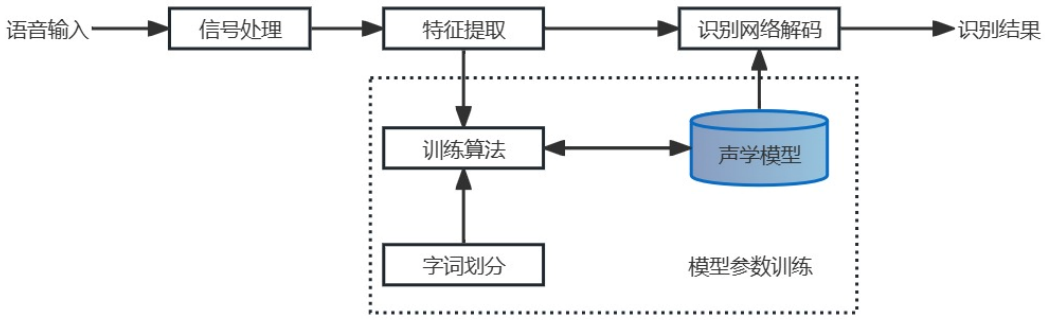

图 3.1.1 语音识别流程图

语音合成技术，又称文语转换（Text to Speech，简称 TTS）技术，是一项能够将任意文字信息实时转化为标准、流畅的语音朗读出来的技术，通过模拟人类语音，提供了更加自然和个性化的交互体验。语音合成的流程大体为：

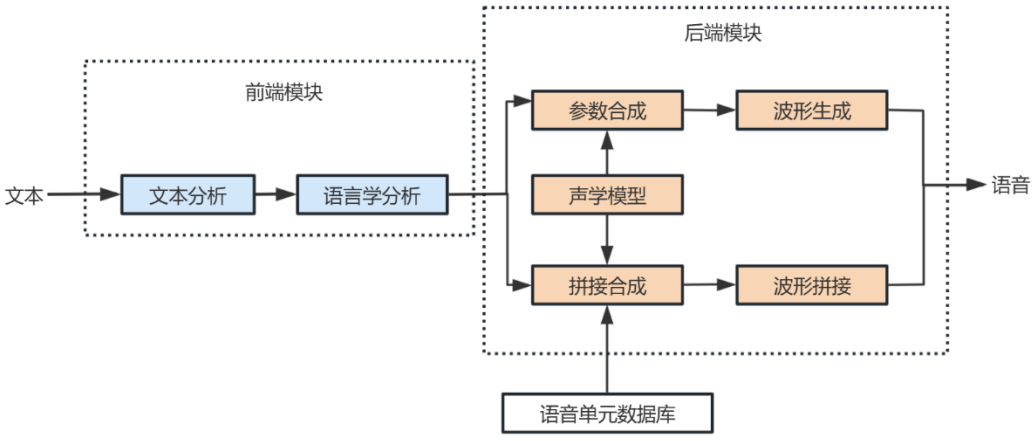

图 3.1.2 语音合成流程图

项目使用科大讯飞提供 API 完成精准的在线语音识别和流畅的在线语音合

成。此外，我们使用了一个对话管理模块，该模块根据语音识别得到的文本信息进行对话流程控制。通过预设的对话剧本，对话管理模块理解用户的意图和请求，决定机器人如何有效地回应，并对任务流程进行控制。语音交互具体实现流程如下：

1. 下载科大讯飞官方 SDK 文件，安装相关依赖包，并修改 APPID。

2. 将该功能模块与 ROS 融合，将关键代码放入 rospkg 并修改 CmakeList 后进行编译。

3. 编写对话管理模块，该模块根据接收到的识别文本，依据预设的对话剧本，生成回复语句并通过消息的发送对后续任务流程进行控制。

4. 开启麦克风，对环境进行录音，通过科大讯飞的语音识别 API，将用户语音输入发送至远程服务器进行处理。服务器利用深度学习和实时信号处理技术，将语音数据准确识别为文本，并将识别结果作为话题信息进行发布。

5. 对话管理模块根据语音识别得到的文本信息，生成回复文本，进行语音合成并播放。同时，对话管理模块将流程信息作为消息进行发布。

6. 将合成的音频进行播放。同时，其余节点收到消息进行指定运动。

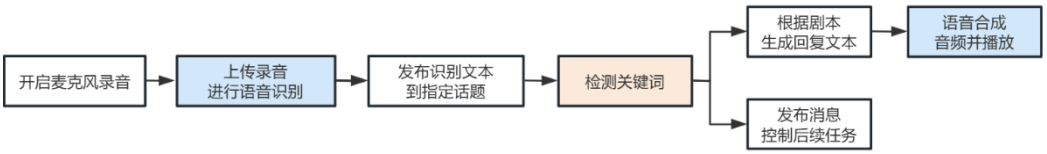

图 3.1.3 语音交互流程图

3.2 行人跟踪的原理和实现

项目中行人跟踪的实现是通过获取 Kinect 摄像头的数据，根据预先设定的参数获取感兴趣的区域（Region of Interest, ROI），根据该区域的物体质心位置控制机器人的运动，具体原理和实现如下。

1. 获取感兴趣的区域 ROI

ROI 的设置通过定义 x、y、z 的最小和最大值来实现。这些参数在初始化阶段通过参数服务器设置，也可以通过动态参数调整。根据这些参数，在摄像头的视野中会构建出一个长方体区域，后续将计算该区域中的质心位置。

2. 将相机坐标系中的点映射到世界坐标系中

视场角（Field of View，FOV）是描述相机或其他成像设备在空间中的视野范围的一个参数。它通常用角度来表示，可以分为水平视场角（Horizontal FOV，HFOV）和垂直视场角（Vertical FOV，VFOV），如图 3.2.1 所示。水平视场角是相机在水平方向上能看到的最大角度范围，通常用于描述相机从左到右能够捕捉到的图像宽度；垂直视场角是相机在垂直方向上能看到的最大角度范围。通常用于描述相机从上到下能够捕捉到的图像高度。

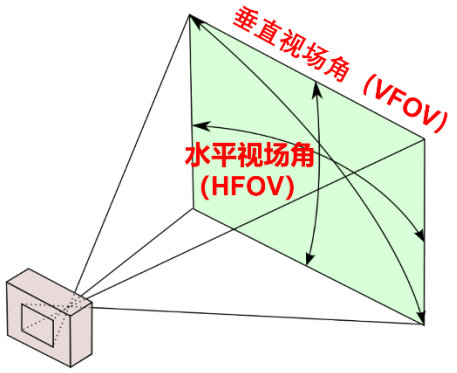

图 3.2.1 视场角示意图

实验中使用 Kinect 摄像头获得的图像，设摄像头的水平视场角的弧度值为 HFOV，垂直视场角的弧度值为 VFOV，图像的像素大小为 height×width。那么每个像素值对应的水平和垂直方向弧度x_{radian}、y_{radian}为：

$$x_{radian} = \frac{HFOV}{width} \tag{3.1}$$

$$y_{radian} = \frac{VFOV}{height} \tag{3.2}$$

设图像中某像素(x_{pixel}, y_{pixel})，那么它对应的水平和垂直方向弧度值α、β分别为：

$$\alpha = \left(x_{pixel} - \frac{width}{2}\right) x_{radian} \tag{3.3}$$

$$\beta = (y_{pixel} - \frac{height}{2}) y_{radian} \tag{3.4}$$

若已知该像素的深度值d，则在世界坐标系下该像素对应位置与摄像头的水平和垂直距离x_{real}、y_{real}为：

$$x_{real} = d\sin\alpha \tag{3.5}$$

$$y_{real} = d\sin\beta \tag{3.6}$$

图 3.2.2 展示了水平视场角与世界坐标系的对应关系。

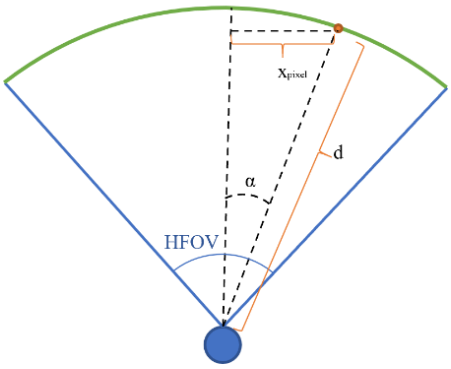

图 3.2.2 水平视场角与世界坐标系对应关系示意图

3. 筛选 ROI 中的点并计算质心

利用步骤 1 设定的 ROI 区域对摄像头采集的像素点进行筛选，得到符合要求的点集。设在 ROI 中的点集为

$$(x_{real_1}, y_{real_1}, d_1), (x_{real_2}, y_{real_2}, d_2) \cdots (x_{real_n}, y_{real_n}, d_n)$$

该点集质心位置的公式如下：

$$x_c = \frac{1}{n}\sum_{i=1}^{n} x_{real_i} \tag{3.7}$$

$$y_c = \frac{1}{n}\sum_{i=1}^{n} y_{real_i} \tag{3.8}$$

$$z_c = \frac{1}{n}\sum_{i=1}^{n} d_i \tag{3.9}$$

4. 生成控制命令

设目标距离为z_{goal}，线速度比例系数为k_z，角速度比例系数为k_x。根据质心坐标生成线速度v和角速度ω的比例控制命令，计算公式如下：

$$v = k_z(z_c - z_{goal}) \tag{3.10}$$

$$\omega = -x_c k_x \tag{3.11}$$

3.3 建图导航的原理和实现

项目中建图导航模块主要包括建图、定位和轨迹规划三部分，其中建图算法

为 Gmapping 算法，定位算法为 AMCL 算法，轨迹规划算法为 DWA 算法。

一、 建图算法——Gmapping

Gmapping 是一种基于粒子滤波的 SLAM 算法。[2]其基于 RBPF（Rao-Blackwellized Particle Filter）粒子滤波算法，并针对 RBPF 的缺点提出了改进，提升了 SLAM 的性能。SLAM 的核心问题是机器人在未知环境中同时进行自身定位和地图构建。

SLAM 的概率描述如下：

$$p(x_{1:t}, m|u_{1:t}, z_{1:t}) \tag{3.12}$$

其中，$x_{1:t}$表示机器人的路径，m表示地图，$u_{1:t}$表示控制输入，$z_{1:t}$表示观测数据。FastSLAM 算法通过 RBPF 方法将 SLAM 问题分解成两个子问题，即机器人定位问题和已知机器人位姿下进行地图构建的问题，其分解过程如下：[3]

$$p(x_{1:t}, m|u_{1:t}, z_{1:t}) = p(x_{1:t}|u_{1:t}, z_{1:t})p(m|x_{1:t}, u_{1:t}, z_{1:t}) \approx p(x_{1:t}|u_{1:t}, z_{1:t})p(m|x_{1:t}) \tag{3.13}$$

FastSLAM 算法采用粒子滤波对机器人的位姿进行估计，并为每个粒子构建一个地图。根据贝叶斯公式推导，我们可以得到：

$$p(x_{1:t}|u_{1:t}, z_{1:t}) = \eta p(z_t|x_t, m)\int p(x_t|x_{t-1}, u_t)p(x_{1:t-1}|u_{1:t-1}, z_{1:t-1})dx_{t-1} \tag{3.14}$$

经过上述公式，机器人位姿的估计问题被转化为一个增量估计的问题，通过观测数据进行归一化处理，得到每个粒子的预测轨迹，然后根据估计的轨迹和观测数据进行地图构建。

FastSLAM 采用粒子滤波方法存在易爆炸和粒子耗散的问题。Gmapping 算法针对这些问题提出了两种解决方案：

1. 粒子权重的更新：首先通过前次估计找到高概率估计点：

$$x_t^* = \arg\max_x p(z_t|x, m)p(x|x_{t-1}, u_t) \tag{3.15}$$

然后在估计点附近进行采样，对每个位姿进行打分，假设它们服从高斯分布，计算出均值和方差后使用正态分布重新计算粒子位姿。

2. 粒子重采样：为了解决粒子耗散的问题，Gmapping 算法根据粒子权重的高低确定是否进行重采样：

$$N_{\text{eff}} = \frac{1}{\sum(w^i)^2} \tag{3.16}$$

当 Neff 小于某个阈值时进行重采样。通过这种方式，保持粒子的多样性和有效性。

二、 定位算法——AMCL 算法

自适应蒙特卡洛定位（Adaptive Monte Carlo Localization，AMCL）是一种基于粒子滤波的定位算法，用于机器人在已知地图中的自定位。其基本思想是通过一组随机分布的粒子来表示机器人的可能位置，并通过逐步修正这些粒子的位置和权重来逼近机器人的真实位置。

AMCL 算法的主要步骤包括：

1. 初始化粒子集：在地图上的可能区域内随机生成一定数量的粒子，每个粒子表示机器人可能的一个位置和朝向。
2. 运动更新：根据机器人的运动模型（如里程计数据），预测每个粒子在运动后的新位置。
3. 测量更新：使用传感器数据（如激光雷达测距数据）来计算每个粒子的权重，权重越大表示粒子的位置越接近机器人的真实位置。
4. 重采样：根据粒子的权重，重新采样生成新的粒子集，保留高权重的粒子，丢弃低权重的粒子，从而集中在高概率区域。
5. 自适应调整粒子数量：根据算法的收敛情况，自适应地调整粒子的数量，以提高计算效率和定位精度。

三、 轨迹规划——DWA 算法

DWA 是基于预测控制理论的一种次优方法，因其在未知环境下能够安全、有效的避开障碍物，同时具有计算量小, 反应迅速、可操作性强等特点。DWA 算法的核心思想是根据移动机器人当前的位置状态和速度状态在速度空间（v, ω）中确定一个满足移动机器人硬件约束的采样速度空间，然后计算移动机器人在这些速度情况下移动一定时间内的轨迹，并通过评价函数对该轨迹进行评价，最后选出评价最优的轨迹所对应的速度来作为移动机器人运动速度，如此循环直至移动机器人到达目标点。

3.4 视觉识别的原理和实现

本项目基于 YOLO 进行了目标物体的识别。在具体的技术路线上，我们主要基于 darknet_ros 开源视觉识别神经网络实现。

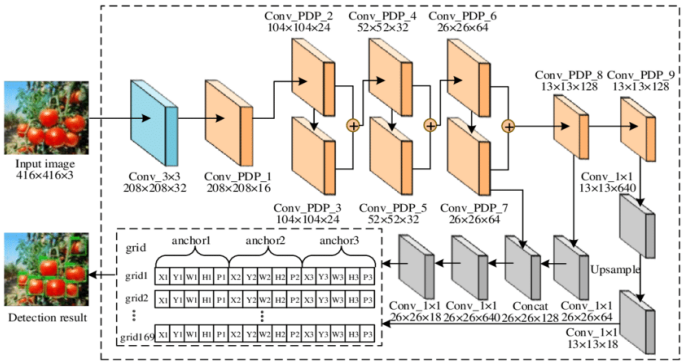

图 3.4.1 YOLOv3-tiny Network 结构图

darknet_ros 是一个将 Darknet 实时对象识别框架集成到 ROS（Robot Operating System，机器人操作系统）环境中的包。通过这种集成，用户可以在机器人系统中直接使用 YOLO（You Only Look Once）算法进行对象检测，这对于导航、交互和避障等应用非常有用。

darknet_ros 的核心节点负责加载 YOLO 模型，接收图像输入，执行对象检测，并发布检测结果的核心操作。这个节点能够处理大量的图像数据，并通过 CUDA 进行 GPU 加速的实时分析，确保实时物体识别处理的速度和效率。

在 ROS 中，darknet_ros 节点主要与两类话题互动。它订阅来自 openni2 摄像头的/camera1/image_raw 话题（其中 camera1 是为了与 Kinect 的相关话题区分而进行的人为指定）以获取机器人相机的实时视频流，并基于此数据执行对象检测；检测结果通过两个发布话题向其他 ROS 节点提供：**/darknet_ros/bounding_boxes 发布检测到的对象的边界框信息，而/darknet_ros/detection_image 则发布带有边界框标记的图像。**在实际运行物体识别时，与相机和 YOLO 相关的整体话题关系如图 3.4.2 所示。

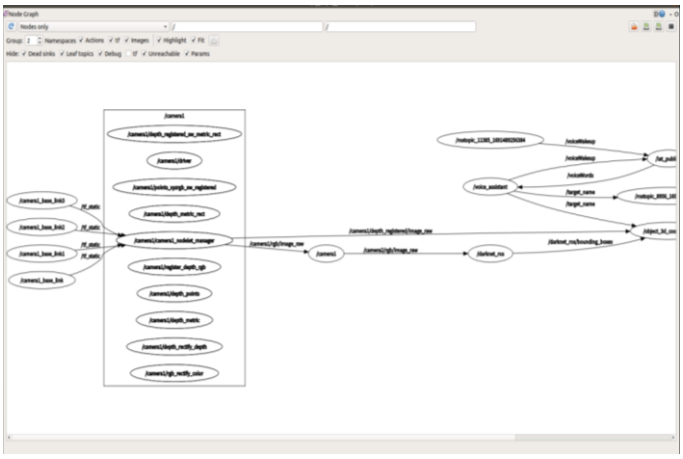

图 3.4.2 视觉识别相关节点与话题关系图

在实现基础的物体识别后，为了实现指定物体的识别，我们以实验室中红色，蓝色，绿色三种不同颜色的烧瓶为目标物体进行了数据集采集，每个物体采集 50 张图片进行标注。考虑到笔记本电脑性能限制，我们最终选择 yolov3tiny 的网络进行训练，经过 10000 轮训练后，可以得到权重文件 10000.weight。

在对应的 yolo_config 文件夹中修改使用的权重文件和现有标签，再次运行 roslaunch darknet_ros darknet_ros.launch 即可实现指定物体的识别和框选。最终效果如图 3.4.3 所示。

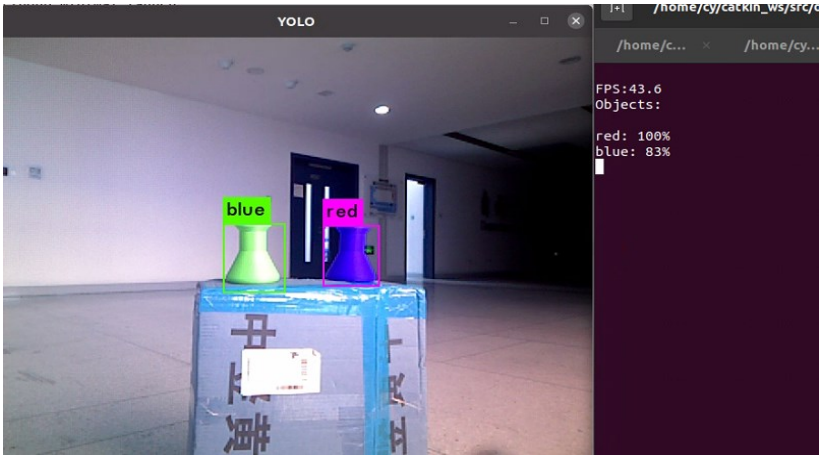

图 3.4.3 YOLO 目标物识别效果

最终视觉识别效的准确性较高，可以准确的框选出目标物体。YOLO 物体识别中候选框这一功能的实现可以帮助我们获取物体的位置，在结合深度传感器信息后可以完美实现后期的 PID 位置调整。

3.5 PID 微调的原理和实现

在导航到最后一个目标点后，由于导航存在一些位置和姿态上的偏差，我们使用 PID 进行机器人位姿的调整。

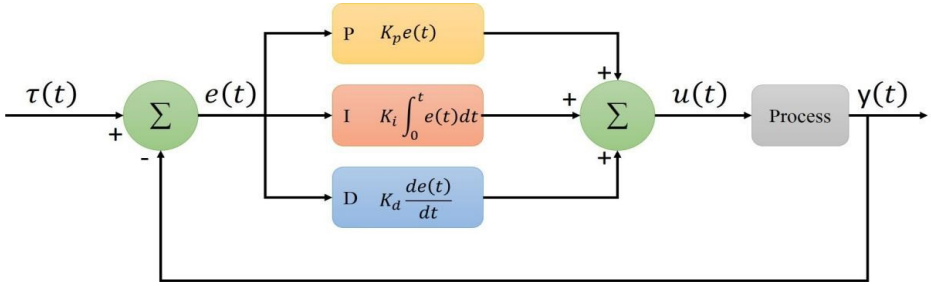

图 3.5.1 PID controller 原理图

微调功能的实现主要分为对象检测与深度信息获取和 PID 调整两大部分。

对象检测与深度信息获取是通过 RGBD_myImage 类实现的，该类负责从 ROS 主题接收 RGB 图像和深度图像，并通过 darknet_ros 接收对象的边界框信息。类的构造函数订阅了相应的 ROS 主题，包括用于获取 RGB 图像的 /camera1/rgb/image_raw 和深度图像的/camera1/depth_registered/image_raw，以及 darknet_ros 返回的对象边界框信息的 /darknet_ros/bounding_boxes 。在 dete_call_back 方法中，首先通过已接收的深度图像计算每个检测到的对象的中心点深度，考虑到实际实验过程中深度图中经常存在大量 NaN 值，我们计算候选框内深度最小值来估计到对象的距离，只统计有效的深度数据。该类将图像和深度信息转换为 OpenCV 格式，方便后续处理。

PID 位姿调整主要通过 bottle_Detection 类和它的 x_registration 方法实现。该类利用 RGBD_myImage 类提供的图像和深度信息来定位特定标签的物体，并计算机器人需要进行的位姿调整。x_registration 方法用于调整机器人的横向位置（x 坐标），以使其与目标物体对齐。通过比较目标的 x 坐标与预设的目标 x 坐标（goal_x），方法决定机器人应该左转、右转还是保持当前方向。使用 Twist 消息类型，发布到 cmd_vel_mux/input/navi 主题来控制机器人的旋转和前进速度。这里的 **PID 控制逻辑体现在如何基于位置误差调整旋转速度：误差越大，旋转动作越显著。一旦误差在可接受范围内，机器人将停止旋转，完成位置调整。**

这两个部分共同确保了机器人能够准确地检测目标对象，并根据实际环境中的距离和位置信息做出有效的移动和操作决策。通过这种方式，机器人能够自动导航至目标位置，并进行抓取目标物的动作。

最终我们成功实现了基于视觉识别的 PID 位姿微调，实验过程中的图像如图 3.5.2 所示。

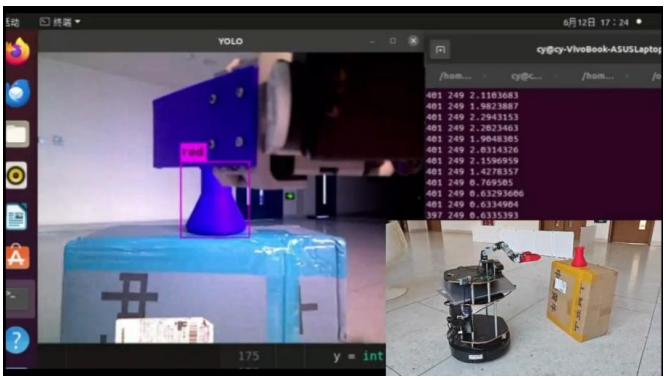

图 3.5.2 基于视觉的 PID 位姿微调效果

3.6 机械臂抓取的原理和实现

TurtleBot 机器人机械臂拥有五个关节。这些关节通过各自的电机驱动，从而精确控制机械臂的运动，使其能够定位到空间中的特定位置。本项目机械臂控制主要两个任务：首先，在 PID 位置精调的基础上，使用机械臂抓取目标物体。抓取后持举物品递交给用户。

本项目采取开环控制的方式，为机械臂设定若干预设点位与姿态，通过精心设计机械臂运动过程中的关键位姿，我们可以控制五个关节精确地运动到这些预设位置，从而完成各种复杂的工作任务。机械臂具体运动流程如下图所示。

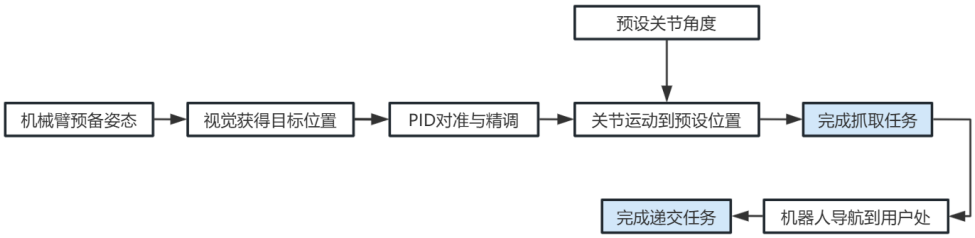

图 3.6.1 机械臂抓取功能流程图

四、个人完成的主要工作

4.1 建立场景地图

项目中首先搭建了一个较复杂场景，如图所示 4.1 所示。

图 4.1 场景搭建

使用 Turtlebot 自带的 Gmapping 包进行地图建立，终端中运行以下命令即可通过键盘控制机器人完成建图，并保存地图。建图结果如图所示。

```
1.  roslaunch turtlebot_bringup minimal.launch
2.  roslaunch turtlebot_navigation gmapping_demo.launch
3.  roslaunch turtlebot_rviz_launchers view_navigation.launch
4.  roslaunch turtlebot_teleop keyboard_teleop.launch
5.  rosrun map_server map_saver -f /tmp/my_map
```

使用 Turtlebot 自带的 Gmapping 包进行地图建立，终端中运行以下命令即可通过键盘控制机器人完成建图，并保存地图。建图结果如图 4.2 所示。

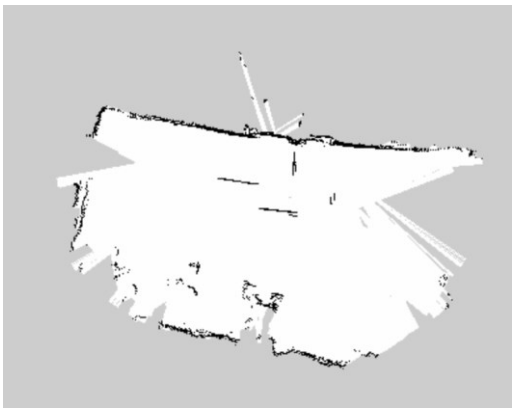

图 4.2 建图结果

4.2 整合行人跟踪、机器人定位和语音交互

Turtlebot 自带的包中提供了实现行人跟踪的 launch 文件 follow.launch 和利用 AMCL 算法实现机器人定位的 amcl.launch 文件。但是同时运行这两个文件时，会出现报错。这是由于这两个 launch 文件均需要调用 Kinect 摄像头，并且传入的参数有所区别，导致节点冲突。

项目中，我将原先的两个 launch 文件进行整合，确定了能够保证跟踪与定位功能的对应摄像头参数，对启动摄像头的部分进行修改，得到了能同时实现跟踪与导航的 amcl_follower.launch 文件。

在示教过程中，示教者通过语音控制机器人记录点位坐标。通过科大讯飞识别到示教者的命令“Remember this point”后，会将识别到的文字发布至/voiceWords 话题中。我编写了 Python 代码 store_pose.py 和 get_pose.py。运行 store_pose.py 程序时，当/voiceWords 话题中出现关键词时，store_pose.py 会调用

get_pose.py 读取/amcl 话题中的坐标信息，并存入 pose.txt 文件中，以供后续按照示教者的路径进行导航。

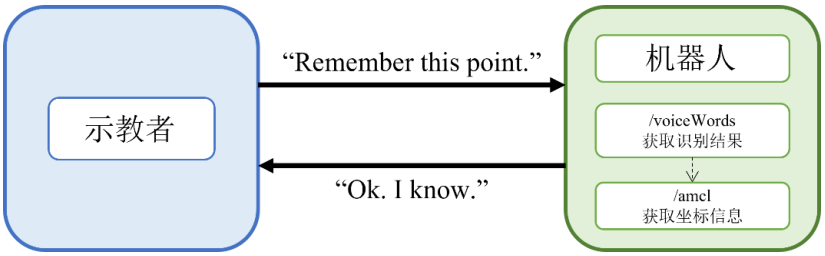

图 4.3 语音控制保存坐标信息

终端中运行以下命令即可完成机器人的行人跟踪，以及语音控制坐标存储。

```
1. roslaunch turtlebot_bringup minimal.launch
2. roslaunch turtlebot_follower amcl_follower.launch
3. roslaunch xf_voice reply.launch
4. rosrun my_dunamixel store_pose.py
```

4.3 机械臂关节的信息获取与控制

项目中机器人的关节由 6 个电机关节组成，在实验中使用了其中五个关节，分别对应 tilt_controller、shoulder_controller、elbow_controller、wrist_controller、hand_controller。经过关节位置的调试，得到项目中所用到的关节参数，如表 4.1 所示。

表 4.1 关节参数表

动作	tilt	shoulder	elbow	wrist	hand
初始化	4.203	3.482	5.062	3.973	2.756
准备抓取	4.172	0.0	3.886	1.641	1.917
抓取	4.172	0.0	3.886	1.641	3.017
收回	4.213	1.820	4.407	3.073	3.068
准备递送	4.310	1.600	4.535	2.782	3.078
递送	4.310	1.600	4.535	2.782	1.917

4.4 编写抓取物品阶段的代码

抓取物品阶段涉及到语音、导航、视觉和机械臂等多个功能的整合。在该阶段中我编写了 all.py 和 control_arm.py 文件，运行 all.py 文件将按照设定顺序执行任务，在其中抓取物品环节将调用 control.py 文件中的 Manipulation 类，执行设定的关节动作。all.py 文件的代码架构具体如下。

1. **读取目标点位坐标信息：**

   从 pose.txt 文件中读取设定的目标点位，存储在 GoalPoints 列表中。

2. **RGBD_myImage 类：**

   1) 订阅 RGB 图像、深度图像和物体检测结果，并提供获取这些数据的方法。

   2) image_call_back、depth_call_back、dete_call_back 分别处理图像和检测结果的回调函数。

3. **bottle_Detection 类：**

   1) 使用 RGBD_myImage 实例化对象，处理特定标签物体（这里标签从文件中读取）的检测和定位。

   2) get_position 获取物体位置及其在深度图中的距离。

   3) get_color 获取指定位置像素的 RGB 颜色值。

   4) x_registration 通过 PID 控制机器人朝特定 X 轴目标位置移动。

4. **move_forward 和 move_backward 函数：**

   向前和向后移动给定的距离。

5. **assign_goal 函数：**

   将给定的位置和姿态信息转换为 ROS MoveBaseGoal 消息。

6. **主程序：**

   1) 初始化 ROS 节点。

   2) 使用 Actionlib 库发送 MoveBaseGoal，使机器人导航到预定义的多个目标点。

   3) 读取颜色标签并实例化 bottle_Detection 类进行物体检测。

   4) 执行一系列动作：前进、定位、抓取、返回等。

   5) 使用语音（通过/voiceWords 话题）进行状态提示。

五、项目总结与展望

5.1 项目总结

一、项目完成情况

我们的项目成功实现了基于人类示教的 Turtlebot 导航，充分利用了模仿学习的原理来提高机器人在复杂环境中的导航能力。我们让 Turtlebot 在实际环境中跟随人类，并精确记录了导航过程中的关键点。这些关键点后续被用于机器人的自主导航，使其能够独立完成从起点到终点的路径复现。

在技术实施方面，我们整合了多种技术以增强系统的实用性和效率。使用 Kinect 进行实时行人跟踪，并通过 YOLO 网络对环境中的对象进行识别。此外，我们还实现了基于视觉信息的 PID 位置调控，确保机器人在导航过程中的位置精确性，同时语音交互功能的加入，使得操作者能够通过简单的语音命令与机器人进行交互，进一步提升了用户体验。

二、项目创新性

- 高效学习能力

我们的机器人展示了在短时间内快速学习复杂导航路径的能力，这一点在机器人技术中尤为重要。通过高效的学习算法，机器人能够迅速吸收并复现在示教阶段获得的导航知识和技能，**这不仅减少了机器人对长时间训练的依赖，也显著提高了其在新环境中的部署效率**，使机器人能够在观察几次操作后，就能独立完成复杂的导航任务。

- 环境适应性的提升

项目通过示教数据，显著提高了机器人的环境适应能力。**在复杂的实际环境中，我们的机器人能够根据示教数据，自动优化其导航策略以适应环境。**通过学习特定环境中人类的避障方式，机器人能够在类似环境中更加高效和安全地导航，从有限的示例中迅速学习并优化其行为模式。

- 人机协作的强化

我们的项目在人机交互方面也进行了创新。通过实时行人跟踪和语音交互技术，机器人不仅能够模仿人类行为，还能与人类操作者进行有效的沟通和协作。操作者可以通过简单的语音命令调整机器人的导航路径，机器人则能理解这些指

令并做出相应的反应，这种交互**提高了工作效率，在需要快速反应和适应复杂任务指令的场景中，展现了极大的灵活性和应用潜力。**

5.2 项目展望

模仿学习（Imitation Learning）是一种机器学习方法，它通过观察和模仿人类或其他智能体的行为来学习特定任务。与传统的强化学习方法相比，模仿学习**不需要智能体通过反复试验来探索环境和获得奖励，而是通过学习示范者的行为轨迹直接获取解决任务的策略。**这种方法在任务复杂、探索代价高或环境不允许大量试错时尤为有效。

我们在项目中实现的基于人类示教的 Turtlebot 导航，正是模仿学习思想的一种体现。通过让 Turtlebot 跟随人类，并记录导航过程中轨迹中的关键点，我们使 Turtlebot 能够学习人类的导航行为，并重复这些轨迹。这种方法不仅简单直观，而且能够有效地将人类的导航经验直接传递给机器人。

在我们的项目中，**人类示教过程实际上是示范者行为的记录和重现过程，这与模仿学习的核心理念一致。**通过模仿学习，我们能够减少机器人在导航情景中自主探索的需求，提高导航效率和安全性。

展望未来，模仿学习从人类学习中学习复杂的技能在更广泛的人工智能领域有着广阔的应用前景。

- 复杂环境中的导航：模仿学习可以帮助机器人在更加复杂和动态的环境中进行导航。例如，在城市环境中，机器人可以通过模仿人类行走路径来避开障碍物和人流，实现更加智能的路径规划[4], [5], [6]。
- 人机协作：模仿学习还可以用于提高人机协作的效率。在工厂或仓库中，机器人可以通过模仿工人的操作来完成搬运、组装等任务，减少工人的劳动强度并提高生产效率。
- 自动驾驶：在自动驾驶领域，模仿学习可以通过学习人类驾驶员的行为来优化车辆的驾驶策略，提高行车安全性和舒适性。自动驾驶系统可以通过观察和模仿大量人类驾驶数据，学习如何在复杂的交通环境中做出正确决策。
- 家庭服务机器人：在家庭环境中，服务机器人可以通过模仿家庭成员的行为来完成打扫、做饭等家务活动，使得家庭生活更加便利。

- 医疗辅助：在医疗领域，模仿学习可以帮助机器人学习医生的操作技巧，用于手术辅助、康复训练等场景，提高医疗服务质量。

在我们的项目中，通过记录并复现人类的导航轨迹，我们使 Turtlebot 能够学习并重复人类的导航行为，我们展示了模仿学习这一概念的实际应用。这种方法直接将人类的导航经验传递给机器人，减少了导航过程中容易出错的复杂探索，提高了导航的效率和安全性。这种从人类学习复杂技能的模仿学习方法在人工智能的多个领域，如复杂环境导航、人机协作、自动驾驶、家庭服务及医疗辅助等，都具有广泛的应用潜力。

参考文献

[1] Z. Fu, T. Z. Zhao, and C. Finn, “Mobile ALOHA: Learning Bimanual Mobile Manipulation with Low-Cost Whole-Body Teleoperation.” arXiv, Jan. 04, 2024. Accessed: Jun. 15, 2024. [Online]. Available: http://arxiv.org/abs/2401.02117

[2] G. Grisetti, C. Stachniss, and W. Burgard, “Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters,” *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, Feb. 2007, doi: 10.1109/TRO.2006.889486.

[3] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, “FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem,” *Aaai/iaai*, vol. 593598, 2002.

[4] H. Karnan, G. Warnell, X. Xiao, and P. Stone, “VOILA: Visual-Observation-Only Imitation Learning for Autonomous Navigation,” in *2022 International Conference on Robotics and Automation (ICRA)*, Philadelphia, PA, USA: IEEE, May 2022, pp. 2497–2503. doi: 10.1109/ICRA46639.2022.9812316.

[5] M. Moder, F. Özgan, and J. Pauli, “Model-based Imitation Learning for Real-time Robot Navigation in Crowds,” in *2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, Busan, Korea, Republic of: IEEE, Aug. 2023, pp. 513–519. doi: 10.1109/RO-MAN57019.2023.10309382.

[6] B. Cèsar-Tondreau, G. Warnell, E. Stump, K. Kochersberger, and N. R. Waytowich, “Improving Autonomous Robotic Navigation Using Imitation Learning,” *Front. Robot. AI*, vol. 8, p. 627730, Jun. 2021, doi: 10.3389/frobt.2021.627730.