

《ROS机器人开发技术》

**课程名称:ROS机器人开发技术**

**教师姓名:XXX**

**提交时间:2018年7月x日**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 课程信息 | 课程名称 | | 名称 | | 章数 | 节数 | | 课程类型 |
| 《ROS机器人开发技术》 | | URDF知识学习 | | 六 | 一 | | 授课(√ )  实训( ) |
| 教师 |  | | | 时长 |  | |
| 参考文献 | 1. ROS机器人程序设计（原书第二版） [西班牙]恩里克.费尔南德斯等著 ，刘锦涛 等译 | | | | | | | |
| 教学  目的  要求 | 掌握：使用统一机器人描述格式 | | | | | | | |
| 教学  重点  难点 | 重难点 | | | PPT页面 | | | 时间分配 | |
| 重点 | 1. gmapping功能包介绍 | | 3~7页 | | | 10分钟 | |
| 1. gmapping功能包演示 | | 8~16页 | | | 15分钟 | |
| 1. launch文件详解 | | 17~18页 | | | 10分钟 | |
| 1. 训练 | | 19~20页 | | | 10分钟 | |
| 难点 | 1. 训练 | | 19~20页 | | | 10分钟 | |
| 教学方法 | 本授课以课堂讲授为主，与课堂演示方式相结合 | | | | | | | |

|  |  |  |
| --- | --- | --- |
| 教学内容 | 操作演示 | 知识点 |
| **PPT第1页:**  今天我们给大家继续介绍movelt这款机器人，这讲课我们主要内容是movelt编程基础和Gazebo机械臂仿真两部分 |  |  |
| **PPT第2页:**  我们的教学内容主要分成4个部分：  1.movelt编程基础  2.Gazebo机械臂仿真 |  |  |
| **PPT第3页:**  首先是介绍movelt编程基础。 |  |  |
| **PPT第4页:**  我们先来回顾一下刚才讲到的movelt核心节点move\_group的一个框架图，在整个框架图中，我们可以看到之前已经讲过的GUI，那接下来我们就要去将C++和Python这样一个接口部分。这里的整个用户接口就是通过action或者service来和move\_group来进行数据上的交互的，最后move\_group完成了许多像路径规划或者控制的计算之后，会通过action和我们的机器人做一个数据上的交互。 |  |  |
| **PPT第5页：**  接下来我们来看一下movelt它的编程接口的相关内容，movelt主要是为我们提供了两种编程接口，一种是Python接口，一种是C++接口，其实这两种使用是非常类似的，它的API基本上保持了一致。在这里呢我们只介绍Python接口。首先呢是要去选择一个控制的话题，接下来给控制设置一个目标位姿。因为我们都是要让机械臂发生运动，这些运动就会有一个起始状态和一个目标状态，这里呢就是来设置目标状态的，都是通过ROS当中pose这样一个消息类型来去定义目标姿态的。参数设置完成之后就来使用set\_pose\_target，来把目标姿态设置到我们的机器人的控制组当中，接下来就可以使用plan这样的API来完成路径规划。如果成功的话会返回一个plan就是路径。这就是关于我们编程接口的大致情况。接下来我们就会用Python完成机械臂的控制，会教大家怎么样控制机械臂来完成关节空间运动，还有工作空间运动等等  **PPT第6页：**  大家主要是要去学习一个编码的流程，就是类似流程图这样一个形式，其实这个流程和使用C++还是Python没有关系，不管使用哪种语言流程都是一样的。首先我们先来一个最简单的运动规划，就是关节空间的运动规划。什么是关节空间，我们可以用机器人或者机器臂用关节空间来描述他的运动状态。比如说这里有一个六轴机器人，那我这个机器人的姿态是由留个电机的位置决定的，如果我已经知道了每个关节的位姿，我是通过运动学把整个机器人的姿态求解出来的。所以这就是关节空间，那我们在关节空间去做规划的话，就是已知了机器人起始的关节空间的状态，然后我们要让机器人运动到一个目标的位置，目标位置同样是使用六轴的关节来描述的，然后就是我们知道了六个轴对应的目标角度，让机器人做出相应的移动，就是完成机器人的关节空间的规划。关节空间规划的时候呢，是六个轴会发生相应的运动，六个轴之间虽然运动的时间是一样的，但是运动轨迹就是各自去考虑各自的轨迹，相互之间是没有什么联系的，所以也可以想到机器人终端走出的轨迹是一个任意轨迹，可能是一条直线，也可能是一条曲线，所以在关节的空间规划中终端姿态是保证不了的。关于机器人关节空间的概念性的东西，大家可以去看一下机器人导论等教材，这里就不详细讲述了。我们可以看一下ppt当中就是我们对应的关节空间规划的Python代码，整个代码我们做了一个比较详细的注释，大家下去之后也可以仔细的去理解一下，那我们在这里主要介绍一下整个流程。首先进入程序之后可以看到要去创建一个控制组的概念，就是说我们要去针对机械臂的哪个部分去做案例，然后因为我们这个仿真机器人里面有两个组，一个是六轴的机械臂这样一个组，还有一个是前端的夹手这样一个组，那我们就可以去创建组的一个对象，一个arm，一个gripper，来让我们的控制可以针对某一个具体的对象来完成，然后这里面实现的接口就是MoveGroupCommender，这里两个组名必须要和之前配置好的setupassistant中的组名对应，否则我们会找不到这个组的。接下来我们会这只机械臂和夹爪允许的误差值，这里都是0.001，它的单位是弧度，就是说我的机器人在运动到目标姿态以后，每个关节在0.001的弧度误差范围内，就认为机械臂到达了目标姿态。接下里我们就要让机器人开始真正的运动了。我们可以看到运动控制的接口并并不是很复杂，首先我们要针对哪一个组，比如先让arm组运动，就让arm组去调用里面的API中的函数set\_name\_target，其中’home’是在setupassistant里面已经配置好的预定义的目标点位，我们有配置一个arm和一个forward，如果有的话，我们可以通过这样一个形式让机械臂运动到这样一个姿态的。这里的home点是一个初始位姿，如果机械臂在任意位置的话，我们需要让他先回到home位姿，接下来设置了目标点之后就可以让机器人做运动了，我们可以使用的接口是go，我们可以将我们的代码和可视化操作对应起来。首先设置一个目标点对应于可视化界面中选择一个随机的位姿，<random valid>，go对应可视化操作里面的Plan and Execute这样一个按键，点击按键之后，机器人先完成一个运动规划，在完成一个运动控制，大家对照着看会好理解API中的含义。然后我们让sleep两秒钟。接下来是对夹手设定一个目标姿态，和设定机械臂初始位置差不多，紧接着就是制定机械臂的目标位置，其中set\_joint\_value\_target就是把指定的目标位置joint\_positions设置为机器人运动规划的目标姿态。然后同样是使用go这一命令就可以完成规划和控制。整个运动规划完成之后就是关闭并退出moveit，通过这样一个API | **可视化界面** |  |
| **PPT第7页：**  接下来我们来看一下这个关节空间规划是一个什么效果。大家首先要把我们之前的demo启动起来，启动了这个demo之后还是和上一讲课一样的有机器人模型，那我们现在要让这个机器人模型运动起来，用代码来控制它做运动，我们需要去运行下面这样一个Python代码rosrun marm\_planning moveit\_fk\_demo.py，大家可以看一下机器人模型发生什么变化，我们可以看到机器人在我们这个初始位姿运动到了一个目标位姿，这个运动的目标姿态，就是我们指定的一个目标空间下的目标姿态。这就是一个简单的目标规划。 | 启动demo  rviz界面 |  |
| **PPT第8页:**  对应于关节空间，机械臂的控制上面还存在着另一个常用的空间就是工作空间，工作空间它的描述对象主要是一个终端的这样一个姿态，也就是说在工作空间下，我去描述机器人的姿态使用的是终端的姿态，通过终端的x，y，z和绕着三轴的转动来描述姿态位置。但工作空间下面有一个问题，就是已知终端姿态反求各个关节位置的时候，各个关节的位姿解不一定唯一，这样就会涉及到选解的问题，就是我们到底要选哪个一个解比较合适，这个选解过程中就会涉及到最优化的原则，我们要使得机器人轨迹最优，功率最优还是其他等等原则来完成求解的选择。这些比较偏理论部分，如果大家有兴趣可以去看一下《机器人学》这本书当中的理论介绍。那我们回到movelt它的编程上面，我们可以通过给定接口来去指定工作空间下面的目标姿态，，可以通过x，y，z，rx，ry，rz来描述这样一个姿态，然后我们指定这个姿态之后可以让机器人从当前位置移动到这样一个姿态。我们来具体看一下API的流程，前面的内容还是一样的，要去创建一个控制组对象，接下来就有点不一样了，因为我们要去控制一个终端的位姿，所以我们要让我们的控制组去获取到到底是哪一个终端，所以用get\_end\_effector\_link这样一个函数来获取终端情况，接下来我们要去指定一个目标姿态，当然需要指定一个参考坐标系，这个坐标系我们指定base\_link，我们可以理解为一个机械臂的全局坐标系，那么接下来的所有位姿都是在这个坐标系下去做描述的，然后我们是允许运动规划失败后重新规划，然后同样是要设置一个允许的误差，接下来先让机械臂回到初始位姿，接下来就是要指定机械臂的一个姿态了，就是输入x,y,z,rx,ry,rz，但这边描述旋转是采用四元数的格式，其实这两者可以相互转化的，然后设置当前姿态为初始姿态，设定目标姿态。在上一个程序当中我们使用go来完成规划和控制，这里我们把go分解成plan和execute两部分，先规划出一条轨迹，在通过execute来执行。后面则是用程序来控制移动，大家可以自己详细对照着注释进行理解，最后关闭退出movelt是必要的。所以我们整体来看，就是设置参考坐标系，设置目标姿态然后让机器人指定目标姿态之后使用plan，execute或者直接使用go完成运动规划。这就是比较简单的工作空间下的路径规划。 |  |  |
| **PPT第9页:**  接下来我们运行一下例程来看一下效果是什么样的，先启动demo，接下来我们启动Python的程序，第一次启动Python程序的时候可能会出现一个warn，就是提示路径没有找到，我们可以重新再输入一遍命令就可以了。 | 启动demo  rviz界面  启动Python程序 |  |
| **PPT第10页:**  之前介绍的两种路径规划都是确定初始和目标位置，而无法确定整个运动过程的轨迹，比如现在我希望能够让终端直线运动，那前两种路径规划就不起作用了，在ROS当中我们采用笛卡尔路径规划的方法实现这样一种功能，我们先来运行一下例程，看看效果是什么样的。首先还是启动demo，为了我们等会看到终端运行轨迹是什么样的，我们添加一个插件Robotmodel，并在左边任务树种选择show trail。接下来我们就来运行笛卡尔空间规划的一个直线例程，我们可以看到机器人的运行轨迹都是直线。然后我们对照着看一下走曲线的效果，可以看到轨迹都是曲线。这个程序比较复杂，这里就不给大家具体介绍了，如果大家有兴趣可以自己去研究。 | 启动demo  rviz界面  添加插件  选择显示轨迹  走直线例程  走曲线例程 |  |
| **PPT第11页:**  另外就是机器人在做运动规划的时候，是可以去完成自主避障的，我们可以在仿真环境当中，添加一些障碍信息，机器人在规划轨迹的时候就会考虑到躲避障碍物，我们一样是先来看一下自主避障例程，一样的先来启动demo，然后来运行例程，这个例程启动时间会长一点，因为它会加入一些障碍物进来。我们可以看到rviz中出现绿色的桌子和一些障碍物，现在我们要把机械臂移到桌子中间，然后再躲避障碍物，移到另外一边。通过这个例程，我们就可以展示规划的躲避障碍物效果，通过movelt来完成自主避障的效果。可能规划出来的轨迹不一定是最优的路径，但是确实可以明确躲避障碍物的。movelt避障的API我们做了详细的注释，所以大家可以直接打开文件去查看。 | 启动demo  rviz界面  运行例程 |  |
| PPT第12页: 在上一小节我们主要给大家介绍了怎么样通过movelt中的Python代码接口，来实现对机械臂的运动控制，另外关于C++的接口大家下去可以去看一下关于movelt的API文档，大家可以去百度上搜索movelt来查看。那我们这一小节准备给大家介绍怎么样在Gazebo当中去实现一个机械臂的仿真系统，我们要把刚才创建的机器人放置到Gazebo环境当中，并且对它能够实现控制。 |  |  |
| PPT第13页: 我们要给机械臂去配置一个controller，这个controller也是ROS controller给我们提供的，这里用到的controller叫做关节轨迹控制器，这个控制器可以接收movelt规划出来的一条轨迹，最终把这条运动轨迹分解到每一个机器人的关节，来完成运动关节的控制。而这个关节控制器的配置文件就是trajectory\_control.yaml，我们打开就可以看到和PPT上一样的内容，在这个控制器插件当中我们添加了两个控制器，因为我们一方面需要去控制机械臂的六轴arm部分，另一部分就是要去控制机器人前端的夹爪，关于六轴的控制的话我们是用arm\_joint\_controller,里面包含了控制的类型和控制的关节，下面是每个关节增益的参数，也是我们可以去做配置的，对于夹爪的参数配置也差不多。 |  |  |
| PPT第14页: 那有了这个配置文件，我们就是要把插件和配置文件结合起来，我们都是在ROS中使用launch文件来完成的。我们可以看到已经编写好的launch文件，就是PPT中展示的内容。首先我们要使用rosparam把我们的配置文件加载进来，这个配置文件就记录了要用哪些controller和要用哪些参数，接下来我们要使用controller\_manager，这是一个ROS给我们提供的功能包，这个功能包的作用就是帮助我们去启动一个controller，到底启动哪个controller是和我们的配置文件有关系的。 PPT第15页: 这样我们就可以在Gazebo中建立机器人模型和机器人模型的控制器，结合我们在上一小节学到的movelt一系列运动规划的控制，在Gazebo中控制机器人模型。在控制之前我们需要去配置一下movelt的控制器，因为刚才的控制器我们是针对gazebo来配置的，现在要在movelt端配置控制器，能够让movelt与gazebo成功通信。那么配置文件是controllers.yaml，打开之后会发现和配置Gazebo文件基本上一样的，里面不一样的是该配置文件中多了action\_ns这一参数，这个参数的含义是每一个控制器都是通过一个action去做连接的，所以movelt和Gazebo之间通讯的接口，如果大家还有疑惑的话，一定要回到move\_group这个图当中去看一看，机器人之间的交互是通过action来完成的，这个action就是两者之间的接口。 |  |  |
| PPT第16页: 那我们接下来还是要把相关的控制器给启动起来，使用的是另外一个launch文件，具体内容已经放在PPT上了，我们可以看到这个launch文件和刚才是差不多的，同样是加载一个配置文件，和启动一个控制器管理器节点。由于我们保证movelt和Gazebo要能够通信，所以action的名称要一致，如果出现如图所示的错误，则说明action名称不一致。 |  |  |
| PPT第17页: 那接下来我们就可以去启动这样一个仿真环境了，这个是我们启动仿真环境的launch文件。 |  |  |
| **PPT第18页：**  而arm\_bringup\_movelt.launch文件则是我们启动的文件，这个文件当中第一步就是启动Gazebo这个仿真环境，然后第二步启动是坐标变换个关节信息，这一个启动项是可选的，如果不去启动，Gazebo环境当中就会缺少对应的信息，但并不会影响使用。然后第三个就是启动Gazebo这一端的controller，第四句则是启动movelt，并且把movelt这一端的controller启动起来。 |  |  |
| **PPT第19页:**  最终我们要实现的一个效果就是把整个的仿真环境给跑起来，大家可以用下面这句命令启动一下，可以打开Gazebo和movelt的rviz的一个可视化界面所以速度上回稍微慢一点，可能会启动失败都是正常现象，可以多启动几次。先是rviz打开了，之后Gazebo再打开，Gazebo打开界面就是这样子的。我们可以看到模型在微笑抖动，这是我们建立模型的不细致，但不妨碍我们去仿真验证。我们默认打开的rviz当中是没有什么显示内容的，我们要去调整一下。接下来我们可以按照之前讲述的通过可视化界面或者程序命令来控制movelt移动，这样Gazebo中的模型也会发生相应的移动，请大家自由尝试。  **PPT第20页** | 启动仿真环境  调整rviz  演示 |  |
| **PPT第21页**  今天的内容就讲到这里，谢谢大家的观看。 |  |  |