《ROS机器人开发技术》

课程讲稿

|  |  |  |
| --- | --- | --- |
| 课程名称 | ： | ROS机器人开发技术 |
| 教师姓名 | ： | 刘梦倩 |
| 提交时间 | ： | 2018年7月15日 |

中国大学MOOC制

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 课程  名称 | 《ROS机器人开发技术》 | | 章 | | 第三章 | | | 课程类型 |
| 节 | | 第一节 | | | 授课(√ )  实训( ) |
| 名称 | 通信架构（一） | | | | | | |
| 教师 |  | 课时 | | | | 1课时 | | |
| 参考  资料 | 1. ROS wiki <http://wiki.ros.org/> 2. 《ROS机器人编程实践》 3. 《ROS机器人开发实践》 | | | | | | | |
| 教学  目的  要求 | 掌握：ROS的通信架构之topic | | | | | | | |
| 教学  重点  难点 |  | | | PPT页面 | | | 时间分配 | |
| 教学重点 | | | | | | | |
| 1. node&master、roscore、rosrun | | | 5~13页 | | | 20分钟 | |
| 1. launch文件 | | | 14~15页 | | | 5分钟 | |
| 1. topic&msg、rosmsg、rostopic | | | 16~27页 | | | 20分钟 | |
| 教学难点 | | | | | | | |
| 1. rosnode | | | 12~13页 | | | 5分钟 | |
| 1. topic | | | 17~19页 | | | 10分钟 | |
|  | 1. rostopic | | | 26页 | | | 5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：**  本章我们讲ROS的通信架构的第一部分。  本章也是ROS当中最重要的一个部分，所以大家一定要认真听讲，认真对待这部分内容。  ROS的通信架构包括各种数据的处理、进程的运行、消息的传递等等，是ROS中最重要的一个环节，没有通信，系统当中的不同的部分就无法组合在一起，整个系统便没有联系，若同一盘散沙。 |  |  |
| **PPT第2页:**  这是一个提纲。分别总结了本章将要介绍到的一些概念以及要讲解的一些命令。  现在先给大家看一下，让大家有一个大概的印象。 |  |  |
| **PPT第3页:**  通常一个机器人的系统是非常复杂的，比如ppt上这个简单的机器人，它包块摄像头、不同关节的控制、轮子的控制、以及一些独特的功能等等。想要控制这样复杂的系统是很困难的，因此ROS采取了点对点的设计。把每一个需要控制的部分都分成一个节点，通过节点和节点之间的通讯进行整个系统的控制。 |  |  |
| **PPT第4页:**  前面的章节在介绍ROS的时候已经提到过，ROS的一个理念是点到点，ROS是由大量互相连接并不断交换消息的小计算机程序组成，这些消息直接从一个程序传递到另一个程序，在ROS中，我们把这样的小程序叫做节点（nodes），ROS正是由许多个nodes组成的。不同的nodes通过通信进行数据的传递。 |  |  |
| **PPT第5页:**  接下来进入到本章的第一部分：Node&Master的介绍。 |  |  |
| **PPT第6页：**  在软件包当中有许多可执行文件，这些可执行文件运行之后就成了一个进程（process），也就是节点（node）。从程序角度来说，node就是一个可执行文件（通常为C++编译生成的可执行文件、Python脚本）被执行，加载到了内存之中；从功能角度来说，通常一个节点负责机器人的某一个单独的功能。由于机器人的功能模块非常复杂，我们往往不会把所有功能都集中到一个节点上，例如，一个节点控制一个激光测距仪，一个节点控制机器人的车轮电机，一个节点执行定位，一个节点执行路径规划，一个节点提供系统的图形视图，等等。  **PPT第7页：**  由于ROS当中有许多节点，就需要有节点管理器master来管理各个节点。  ROS Master向ROS系统中的其他节点提供命名和注册服务。它跟踪发布者（publisher）和订阅者（subscriber）的主题（topic）和服务（service）。Master的作用是使每个ROS节点能够彼此定位、建立连接，同时还为系统提供参数服务器，管理全局参数。  node首先在master处进行注册，之后master会将该node纳入整个ROS程序中。节点之间的通信也是先由master进行“牵线”，才能两两的进行点对点通信。当ROS程序启动时，第一步首先启动master，由节点管理器处理依次启动node。 |  | **node**  **Master** |
| **PPT第8页：**  roscore是一个向节点提供连接信息，以便节点间可以互相传递消息的服务程序。  我们下面仔细讲解一下roscore究竟是用来作什么的。这部分可能会比较难理解，大家注意下。  每个节点都在启动时连接到roscore并注册该节点发布和订阅的消息（这部分也就是本章的重点，后面会讲到，在这里先做大体了解）。当一个新的节点出现时，roscore向它提供与其他发布并订阅相同消息主题的节点建立点对点连接的必要信息。每一个ROS都需要一个roscore，没有它，节点无法相互找到彼此。  roscore就相当于一个类似于平台的地方，卖家和买家都在上面注册并且发布信息，当卖家卖的东西买家正好需要的时候，就在平台上面联系，但是最后的交钱和交货的部分和平台无关。 |  | **roscore** |
| **PPT第9页:**  已知roscore在网络中的位置后，节点在roscore中完成注册并请求roscore通过命名找到其他节点以及数据流。每个节点告诉roscore该节点提供和接收何种消息，roscore则提供了相关消息的发布者和接受者的地址。（再次提醒这是本章通讯的主要内容，本章就是讲如何以这种方式进行节点间的通讯。）  roscore同时也提供了一个被ROS节点广泛用于程序配置的参数服务器。参数服务器允许节点存储和获取任意数据结构，例如机器人的描述，算法的参数等等。有一个简单的命令行工具用于与参数服务器交互：rosparam。（会在通讯章节的第二部分讲到，现在提到只是让同学们有一个大致的了解，后来学到这个东西的时候知道是和roscore有关就行。到时候再返回来看这部分。） |  |  |
| **PPT第10页:**  roscore启动时，同时启动的还有rosout和parameter server,其中rosout是负责日志输出的一个节点，其作用是告知用户当前系统的状态，包括输出系统的error、warning等等，并且将log记录于日志文件中，parameter server即是参数服务器，它并不是一个节点，而是存储参数配置的一个服务器，后文我们会单独介绍。每一次我们运行ROS的节点前，都需要把master启动起来，这样才能够让节点启动和注册。 |  |  |
| **PPT第11页:**  既然已经介绍了节点，那么节点应该如何启动呢？这就设计到我们本章的第一个命令：rosrun。  ROS节点通常是可执行程序，用户可以在文件系统中通过cd命令手动切换到各个位置并启动节点。但是在一个很大的文件系统中，因为节点可能放在层级结构的文件系统中很深的位置，进入长的路径中会比较麻烦。因此ROS提供了一个命令行程序rosrun来寻找程序包中的可执行程序并向这个程序传递任何参数。rosrun可以使用包名直接运行一个包内的节点而不需要知道这个包的路径。  具体用法如下：  $ rosrun pkg\_name node\_name |  | **rosrun** |
| **PPT第12页:**  rosrun比较简单，下面我们来学习一下复杂一些的rosnode命令。  rosnode是一个命令行工具，用于显示关于ROS节点的调试信息，包括发布、订阅和连接。它还包含一个用于检索节点信息的实验库。本库仅供内部使用。  下面我们逐个学习rosnode命令的用法：  …… |  | **rosnode** |
| PPT第13页: ……  rosnode cleanup是一种临时解决方案，在正常操作中不鼓励使用它。它的好处是美观，也有潜在的未注册的功能节点的缺点。  以上命令中常用的为前三个，在开发调试时经常会需要查看当前node以及node信息，所以请记住这些常用命令。如果你想不起来，也可以通过rosnode help来查看rosnode命令的用法。 |  | rosnode |
| PPT第14页: 关于Node我们就讲到这里，下面我们进行本章的第二大部分：launch文件。  机器人是一个系统工程,通常一个机器人运行操作时要开启很多个node,对于一个复杂的机器人的启动操作应该怎么做呢?当然,我们并不需要每个节点依次进行rosrun,ROS为我们提供了一个命令能一次性启动master和多个node。  该命令是:  $ roslaunch pkg\_name file\_name.launch  roslaunch命令首先会自动进行检测系统的roscore有没有运行，也即是确认节点管理器是否在运行状态中，如果master没有启动，那么roslaunch就会首先启动master，然后再按照launch的规则执行。launch文件是描述一组节点以及它们话题重映射和参数的XML文件，里已经配置好了启动的规则。所以roslaunch就像是一个启动工具，能够一次性把多个节点按照我们预先的配置启动起来，减少我们在终端中一条条输入指令的麻烦。 |  | launch文件 |
| PPT第15页: launch文件同样也遵循着xml格式规范,是一种标签文本,它的格式包括以下标签:  其中有些标签很重要，经常用到，有的标签不是很常用，我们在后面的实训的章节当中会向大家演示如何建立一个launch文件，并使用这个文件。 PPT第16页: 下面我们开始进行本章最重要的部分：topic和msg |  | launch文件 |
| PPT第17页: 节点之间最重要的通信机制就是基于发布（publish）/订阅（subscrib）模型的消息（message）通信。每一个消息都是一种严格的数据结构，支持标准数据类型（整型、浮点型、布尔型等），也支持嵌套结构和数组，还可以根据需求由开发者自主定义。 |  | topic |
| **PPT第18页:**  topic在ROS中使用最为频繁。在ROS中有两个节点，一个是发布者（publisher），一个是订阅者（Subscriber），两个节点分别发布、订阅同一个topic，启动顺序没有强制要求，此处假设publisher先启动，可大致分成以下七个步骤来分析建立通信的详细过程。  ……  接下来我们通过图解的方式一步步让大家理解这七个步骤。 |  |  |
| PPT第19页: 1.publisher注册  publisher启动，向ROS Master注册发布者的信息，包含所发布消息的话题名，ROS Master会将节点的注册信息加入到注册列表中。  2.subscriber注册  subscriber启动，同样向ROS Master注册订阅者的消息，包含需要订阅的话题名。 |  | topic |
| PPT第20页: 3.ROS Master进行信息匹配  Master根据subscriber的订阅信息从注册列表中查找，如果没有找到匹配的发布者，则等待发布者的加入；如果找到匹配的发布者的信息，则向subscriber发送publisher的地址信息。  4.subscriber发送连接请求  subscriber接收到Master发回的subscriber的地址信息，尝试向publisher发送连接请求，传输订阅的话题名、消息类型以及通信协议。 |  |  |
| **PPT第21页：**  5.publisher确认连接请求  publisher接收到连接请求后，继续向subscriber确认连接信息，其中包含了自身的TCP地址信息。  6.subscriber尝试与publisher建立  publisher接收到确认信息后，使用TCP尝试与subscriber建立网络连接。 |  |  |
| **PPT第22页：**  7.publisher向subscriber发送数据  成功建立连接后，publisher开始向subscriber发送话题消息数据。  subscriber接收到消息会进行处理，一般这个过程叫做回调（callback）。就是提前定义好了一个处理函数(写在代码中),当有消息来就会触发这个处理函数,函数会对消息进行处理。 |  |  |
| **PPT第23页：**  ppt上是一个topic通讯的示意图。  我们根据这个示意图再回忆一遍到底有哪七个步骤：  1.publisher注册  2.subscriber注册  3.ROS Master进行信息匹配  4.subscriber发送连接请求  5.publisher确认连接请求  6.subscriber尝试与publisher建立  7.publisher向subscriber发送数据 |  | topic |
| PPT第24页: 消息（msg）文件就是一个描述ROS中所使用消息类型的简单文本。它们会被用来生成不同语言的源代码。msg文件存放在package的msg目录下，msg文件其实就是每行声明一个数据类型和变量名，可以使用的数据类型如下：  bool、int8、int16、int32、int64(以及uint)、float、float64、string、time、duration、header、可变长数组array[]、固定长度数组array[C]等等。ROS中还有一个特殊的数据类型：Header，含有时间戳和坐标系信息。在msg文件的第一行经常可以看到Header header的声明。  下面的图片是一个简单的msg文件的案例： |  | msg |
| PPT第25页: msg也有相对应的ros命令：  rosmsg  这个命令比较简单，只有两个子命令：  ……  我们可以通过rosmsg list命令 查看系统上所有的消息，  然后从当中随便选择一个msg运行rosmsg show msg\_name命令 |  | rosmsg |
| PPT第26页: 下面我们进入最后的一个十分重要的部分：  关于topic的命令：rostopic  这个命令的子命令比较多，需要同学们在后面的实训环节当中跟着老师多练习，才能记住每个命令的作用。  **PPT第27页:**  授课的部分就到此结束了，谢谢各位同学的收听，希望大家在后面实训的时候多回顾授课部分的讲义和ppt，起到记忆的效果。 |  | rostopic |
| 知识点框图 | | |
| 通讯架构之话题  概念  命令  rostopic  roslaunch  roscore  rosnode  rosrun  rosmsg  launch文件  msg  topic  Master  Node | | |