ROS学习笔记

ROS是什么? 通信机制 + 开发工具 + 应用功能 + 生态系统

> **通信机制**：松耦合分布式通信,ROS把机器人的功能框架抽象成一个个节点 ,通过节点进行通讯
> **开发工具**：机器人可视化工具,Qt工具箱等等
> **应用功能**：Navigation、SLAM、MoveIt等
> **生态系统**: 发行版、软件源、博客等

ROS的作用：增加机器人研发过程中软件复用率。(***不要重复造轮子***)

1. 基本概念

节点和节点管理器

节点:

- 执行具体任务的进程
- 不同节点可使用不同编程语言,可分布式运行在不同主机
- 节点在系统中的名称必须唯一

节点管理器(ROS Master):

- 为节点提供命名和注册服务
- 跟踪和记录话题/服务通信,辅助节点互相查找、建立连接
- 提供参数服务器,节点使用此服务器存储和检索运行时的参数

话题通信

- **话题(Topic)————异步通信机制**:
  - 节点间用来传输数据的重要总线
  - 使用发布/订阅模型,数据有发布者传输到订阅者,同一个话题的订阅者或发布者可以**不唯一**
- **消息(Message)————话题数据**:
  - 具有有一定的类型和数据结构,包括ROS提供的标准类型和用户自定义类型
  - 使用编程语言无关的.msg文件定义,编译过程中生成对应的代码文件

服务通信————同步通信机制

- 使用客户端/服务器(C/S)模型,客户端发送请求数据,服务器完成处理后返回应答数据
- 使用编程语言无关的.srv文件定义请求和应答数据结构,编译过程中生成对应的代码文件

参数(Parameter)

- 可通过**网络**访问的共享、多变量字典
- 节点可使用此服务器来存储和检索运行时的参数
- 适合存储静态、非二进制的配置参数,不适合存储动态配置的数据

2.基本命令行工具使用

- `rqt_graph`:图形化界面,显示出计算图,含有节点和话题
- `rosnode`: 打印出节点信息
- `rostopic`: 打印出ROS话题信息,也可以发布话题
- `rosmsg`: 显示ROS消息的类型
- `rosservice`: ...
- `rosbag`: 可以用`bag`记录调试信息(record),在当前终端路径下生成一个.bag文件,之后再使用该文件复现(play)

3.创建工作空间与功能包

> `src`: 代码空间
> `build`: 编译空间
> `devel`: 开发空间
> `install`: 安装空间

> 工作空间目录:
>
> - `src`
> - `CmakeLists`
> - 各种功能包
> - `CmakeLists`
> - 自定义消息类型
> - `package.xml`: 功能包的描述信息
> - 源码(`.cpp`或`.py`)
> - `devel`
> - 各种`setup.*sh`
> - `lib`
> - 编译生成的功能包
> - share
> - `build`
> - `install`

创建工作空间:

1. 创建`src`文件夹,在目录下`catkin_init_workspace`,会生成`Cmake`文件
2. 编译工作空间,到`src`上级目录中`catkin_make`,生成`build`文件夹和`devel`文件夹
3. 执行`devel/setup.*sh`设置环境变量
4. `catkin_make install`生成安装空间

创建功能包

1. 在`src`路径下使用`catkin_create_pkg <package_name> [depend1] [depend2] [depend3]`创建功能包并指明依赖
2. 编译功能包,在**根目录**下`catkin_make`编译功能包

4.发布者Publisher的编程实现(让小海龟动起来)

- 创建功能包(包含`rospy,roscpp,std_msgs,geometry_msgs,turtlesim`依赖)
- 如何实现一个发布者
  - 初始化ROS节点
  - 向ROS Master注册节点信息,包括发布的话题名和话题中的消息类型
  - 创建消息数据
  - 按照一定频率发布消息
- 功能包编写完成后,回到根目录下进行编译(使用`cpp`编写的文件会在`/devel/lib`文件夹下生成功能包,包中有同名可执行文件)
- 先启动海龟节点,然后`rosrun`自己的功能包。小海龟就可以动起来

---

5. 订阅者Subscriber的编程实现

- 如何实现一个订阅者
  - 初始化ROS节点
  - 订阅需要的话题
  - 循环等待话题消息,接收到消息后进入回调函数
  - 在回调函数里完成消息处理

---

6. 自定义话题消息

- 定义msg文件(与编程语言无关,在功能包中创建msg文件夹,在其中定义`*.msg`)
- 在`package.xml`中添动态生成话题消息的功能包依赖:

```
    <build_depend>message_generation</build_depend>
    <exec_depend>message_runtime</exec_depend>
```

注意：`python`到这就可以了,在python文件中可以直接`import`消息类型

---

7. 客户端Client的编程实现

- 初始化ROS节点
- 创建一个Client实例
- 发布服务请求数据
- 等待Server处理之后的应答结果

> `python`中使用`rospy.ServiceProxy()`进行服务的定义

---

8. 服务端Server的编程实现

- 初始化ROS节点
- 创建一个Server实例
- 循环等待服务请求,进入回调函数

- 在回调函数中完成服务功能的处理,并反馈应答数据

9.Topic通信和Service通信的理解

- **Topic**: node到node之间的单向通信方式,Publisher发布消息后并不会关心Subscriber是否接收到消息,不同的node可以向同一个topic上发送、接收消息,发送数据的node不知道数据据发送到哪个节点,接受数据的node也不知道数据是哪个node发出的,node只需要负责自己的功能实现以及外部接口,不需要关心其他node的行为,是一种异步通信
- **Service**: 消息的传输只涉及两个node,发送请求的一端成为client,提供服务的一端称为server Service通信是双向的,server处理完请求后会反馈回一个response,client发布请求后会阻塞等待reply,直到server处理完请求并完成response,client才会继续执行,是一种同步通信