**中国大学慕课**

**实训文档**

**ROS机器人开发技术**

**实训6.2 为可视化模型的关节增加**

**运动属性和物理属性**

|  |  |
| --- | --- |
| **姓名:** | **组号:** |
| **实训负责人:** | **日期:** |

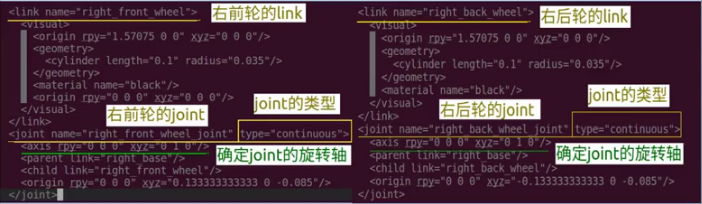
# 背景知识

在前面的实训课程中，我们搭建了R2D2的机器人可视化模型，然而，之前搭建的模型就像是积木一样，不能够运动，并不能满足我们实际想要的需求。在实际的生活中，我们看到过商城的扫地机器人、公司中的智能对话机器人等等，他们能够运动，能够识别，甚至能够像人一样对话，这里我们需要从简单的操作开始，希望为可视化模型的关节增加运动属性。

# 子任务1：修改代码为URDF模型增加joint属性，教师演示后让学生自己演示。 (15min)

**第一步操作：增加continuous连续不断旋转的joint**

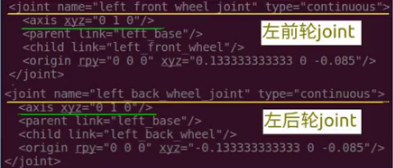
为了能够给模型增添类似于我们轮子的属性。这次实训课中的代码修改是在前面实训课的基础代码上进行修改的，为joint增加相应的移动属性。我们首先为r2d2的车轮joint增加continuous属性，这样车轮就可以旋转运动了。

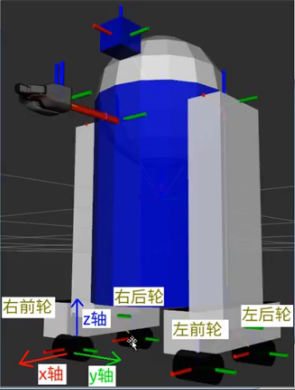


从上图的代码中可以看出我们将右前轮和右后轮的joint类型从fixed改为continus，这样joint就可以不断的旋转了，但是如果要想旋转就需要确定joint的旋转轴，因此增加了axis属性，这里rpy属性没用可设置为0或者不写也行，对于xyz我们将y设置为1，说明我们需要让该joint围绕y轴进行旋转。同理，右后轮也是类似的操作。

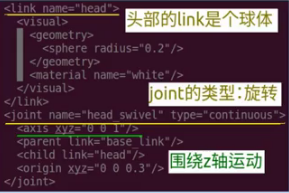
问题：为什么我们要增加axis增加为属性，并且把y置为1？

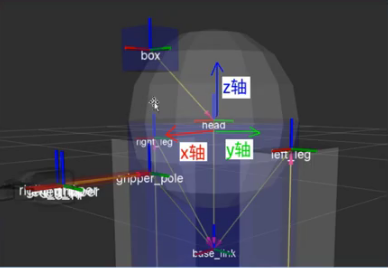
答：我们之所以想要车轮的joint围绕y轴进行旋转是由于x轴朝前，y轴是横向的，如下图所示，我们可以得知4个车轮的坐标轴方向都相同，我们需要4个车轮都围绕坐标轴的y轴旋转，同理我们修改左前后轮的代码也应该相同，在这里我们将axis的rpy属性删除也是可以的，因此我们只需要xyz属性。





接下来我们为r2d2的头部joint也增加continuous属性，这样头部也可以旋转了，修改的代码与车轮相似。





问题：为什么这个机器人模型能够左右摇头？

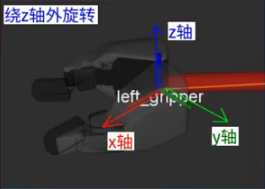
答：如上图所示，我们就明白为何头部的joint在旋转时我们要围绕z轴了，因为只有围绕z轴旋转才能让脑袋左右旋转，就跟人的脑袋一样，围绕着脖子旋转脑袋才能左右扭头，如果我们改为围绕y轴旋转那么就会实现成让脑袋上下点头的运动了。

**第二步操作：增加revolute外向旋转的joint**

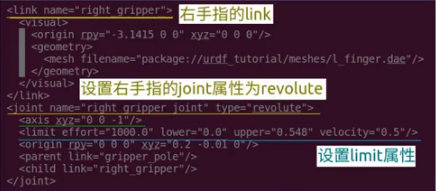
这里修改的属性类似于我们手指头的运动属性。

接下来我们为r2d2的抓取手指joint设置运动属性revolute，这是一种新的运动属性，类似于我们人的手指，就像拇指和食指的运动特性，只能向外旋转和向内合拢，这样就能夹住物体。



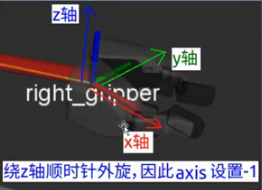


对于limit属性只有revolute和prismatic joint才能设置，对于effort属性其实设置的是抓取的力量限制大小，单位是N.m只有revolute这种抓取的joint才用设置effort属性。对于lower属性设置的是外旋转的下限，单位是弧度rad，同理upper设置向外旋转的上限，1.548rad约是89度。对于velocity设置的是手指运动的速度，对于revolute设置的是rad/s，velocity设置对于prismatic设置的是m/s。



问题：左右手指的设定有什么不同？

答：同样也需要设置右边手指的joint属性为revolute，同样也需要增加旋转轴axis属性和限制属性limit，包括限制revolute的夹取力量大小effort，revolute外旋时的最小值lower和最大值upper，这里需要注意这里用的是弧度表示，最后限制的就是revolute在夹区时的速度大小velocity，单位为rad/s。不过，我们可以发现，和左边手指的设定稍有不同，绕着z轴是顺时针旋转，xyz的z轴应该置成-1。



在本次实训课中，对于limit中的effort和velocity属性没有起到明显作用，后面会有更加详细的介绍，这里我们只需要关心手指的张开最大值upper和合拢起来lower最小值各是多少就行。

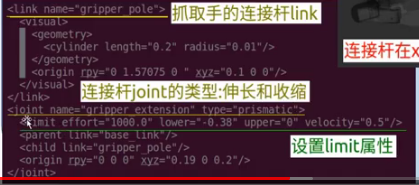
**第三步操作：增加prismatic可以伸长缩短的joint**

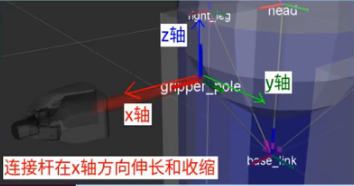
类似于机器人机械臂的伸长和缩短的运动属性。

最后来介绍最后一种joint属性prismatic，这种属性不同于上面两种，因为continuous和revolute其实都是围绕某一坐标轴旋转运动，但是prismatic是在坐标轴的某一方向上运动。

唯一的不同就是数值的单位不同，prismatic的单位是米m，revolute的单位是弧度rad。

相关示意图和坐标系如下：





# 背景知识

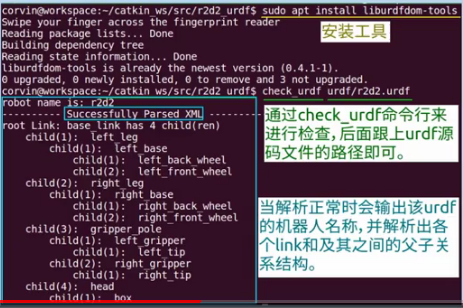
在本次实训中，我们将要创建一个看起来像星球大战里的R2D2的机器人可视化模型，事实上，在游戏中，这些机器人能随着键盘的操作，会产生一些复杂的交互。我们在上一个子任务中，已经添加了一些机器人的运动属性，现在我们需要真正的在Rviz中可视化我们的机器人结构，并且能够让它动起来。

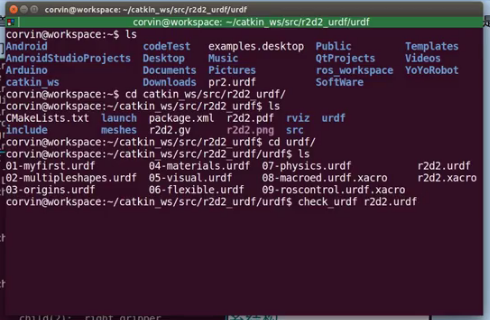
# 子任务2：运行URDF模型，查看修改后的结果。 (15min)

**第一步操作：**

介绍第一个小工具，安装liburdfdom-tool命令行工具来检查代码语法。用法是：check\_urdf urdf/r2d2.urdf

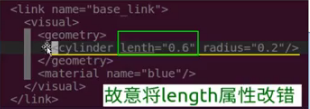
编写完代码后，我们首先需要检查下编写的代码是否有错误，这就需要liburdfdom-tools工具，该工具是不在ros安装包中的，我们需要在ubuntu中手动安装，具体操作如下：

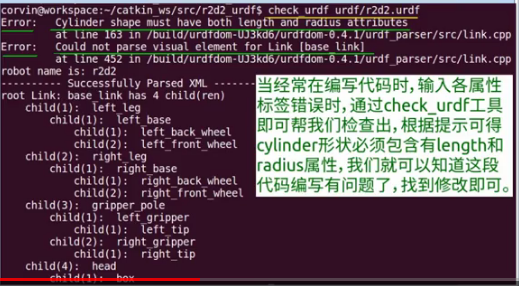
****

****

问题：假如我们在编写urdf文件时不小心将属性标签写错，check\_urdf工具如何帮我们检查该错误呢？

答：在这里我们模拟一下经常遇到的错误类型就是拼写错误。check\_urdf命令行工具会帮我们检查代码的语法是否正确，这样我们就有助于我们快速定位错误改正。

****

****

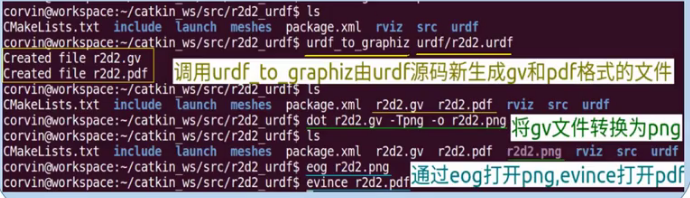
就可以检查我们urdf编写的语法是否正确

**第二步操作：**

当介绍完如何检查代码语法后，介绍第二个小工具，接下来我们就可以尝试使用graphiz将URDF可视化，具体操作如下：

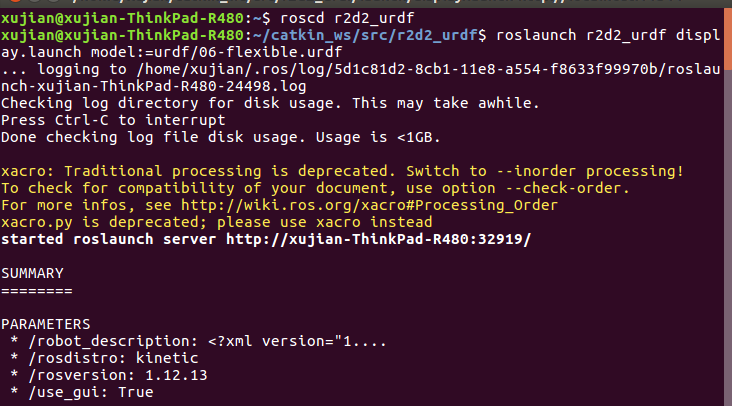
urdf\_to\_graphiz这个工具在indigo以后的版本中都是liburdffdom-tools工具集中的，只要我们在前面已经安装了第一个小工具的话，这里就可以直接调用urdf\_to\_graphiz了。

这里新生成两种文件gv和pdf文件，一般情况下我们只需要pdf就可以了，但是如果需要png格式的图片的话，我们可以将gv格式给转换一下就可以了。gv 文件本身就是一个类似于txt的文本，和刚刚那个pdf 的数状图一样，只不过是png格式。

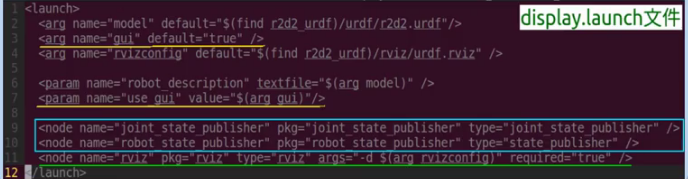
****

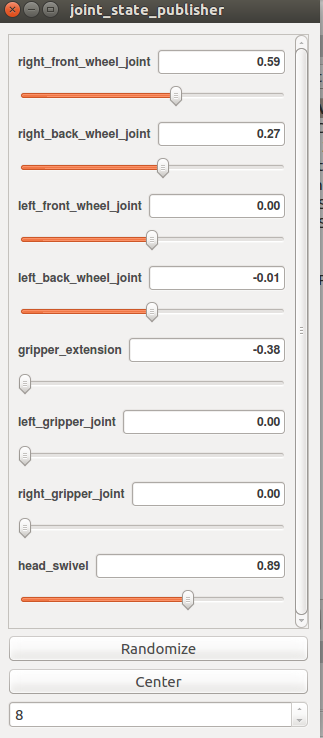
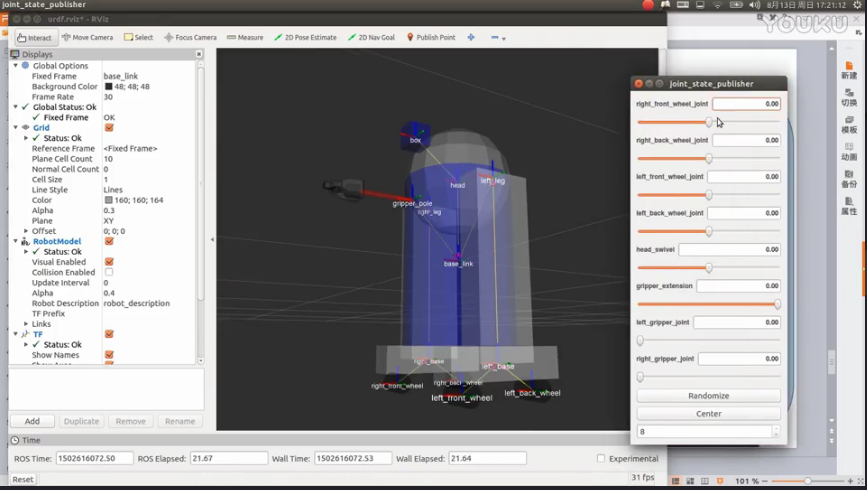
我们可以在命令行中输入命令：evince r2d2.pdf，就能看到里面树状的结构。

最后，启动一下文件运行：

****

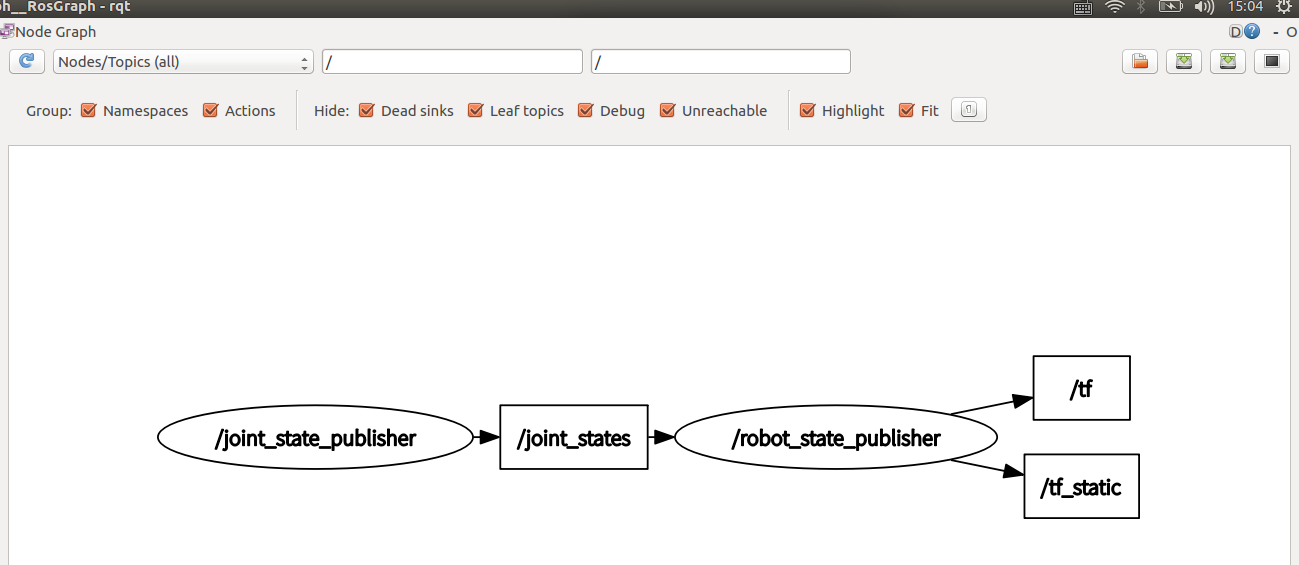
具体的display.launch文件

****

****

**第三步操作：**

接下来我们查看topic和node之间的连接图，通过rqt\_graph查看，可以在终端输入命令：rosrun rqt\_graph rqt\_graph



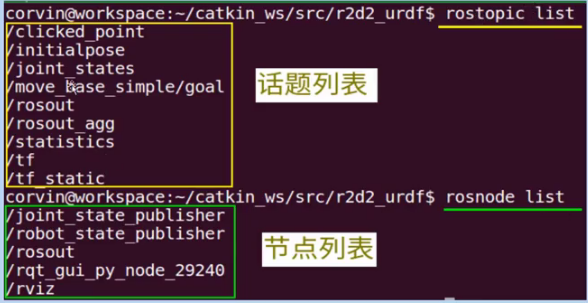
问题：当我们使用rviz中的gui控制面板来控制各个joint运动，背后机制如何实现呢？

答：gui首先解析urdf文件，找到所有的非固定joint和相应的limit属性。

当滑动gui上的滑动条时，将要发送/sensor\_msga/ JointState消息到/joint\_states话题。

节点/robot\_state\_publisher订阅/joint\_states话题，该节点会计算所有的坐标转换，将计算得到的转换树发送到rviz中进行显示。这样就可以看到机器人当前的位姿信息。

最后，可以查看所有的话题和节点列表。

****

# 

**背景知识：**

前面的实训课程中，我们学会了如何给机器人增加一些运动属性，这次我们需要更加丰富一下我们的模型，让它更加接近真实的机器人。因为，真实机器人会有一些物理属性：它的身体和手可能会发生碰撞，连接的关节在运动时会产生摩擦力，关节在运动的过程中会有一个阻尼系数。为了能够让机器人达到我们想要的目的，我们需要增加一些物理属性。

# 子任务3：在URDF模型中增加物理和碰撞检测属性，教师演示后，让学生自己练习 (15min)。

**第一步操作：**为link增加collision属性

问题：如何编写代码增加collision？

答：到目前为止，我们仅仅为我们的link增加了一个标签visual，这个功能就是显示机器人的基本形状。然而为了能够让各个link之间具备碰撞检测或者为了可以让机器人在Gazebo中进行仿真，我们就需要定义collision标签，这样该link就具备了碰撞检测功能。

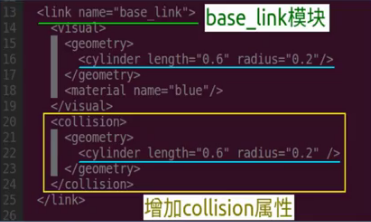
Collsion元素是link对象的子元素，和visual标签在同一个等级。

Collision元素也需要定义它的形状，跟visual元素差不多也是需要geometry标签。这个geometry标签的格式和visual中使用方式相同。

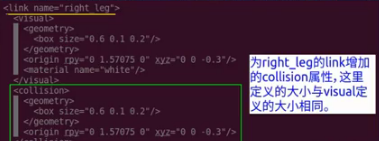
大部分情况下，我们想要碰撞检测区域跟visual中的geometry一样，但是有些情况下不想一样：

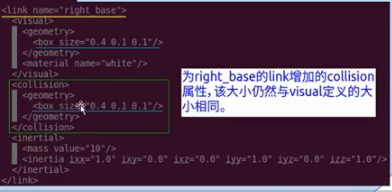
1、更快速的处理：由于为这两个模型做碰撞检测需要进行很复杂的运算，为了简化运算，我们就使用简单的模型。

2、安全区域：当想为敏感的link设置限制区域时，防止其他link靠近，就需要设置一个比link更大的空间才行。

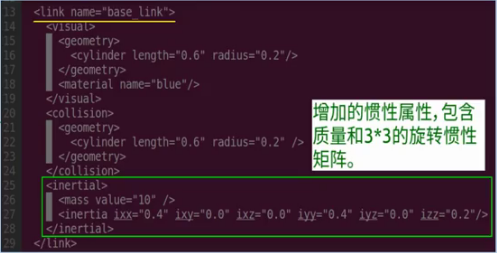
****

其他link也需要增加的collision属性，由于代码较多且各个link部分增加的代码雷同，下面只取其中部分演示：

****

****

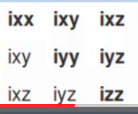
为了使模型能够正常仿真，需要为机器人定义若干物理属性，因为这些属性在物理仿真引擎（Gazebo）需要，每一个link元素在进行物理仿真的时候都需要inertial惯性标签，代码如下图所示：

****

Inertial元素也是link对象的子元素，跟collision同等级。

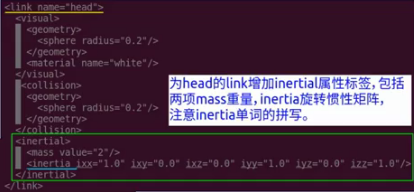
Mass标签的单位是：千克

在inertia元素中需要指定个3\*3的旋转惯性矩阵，因此只需指定上三角的六个元素即可。



由于在各个link中增加inertial属性的代码相同，这里只列出部分作为演示：





问题：对于3\*3的旋转惯性矩阵是怎么得到的呢？

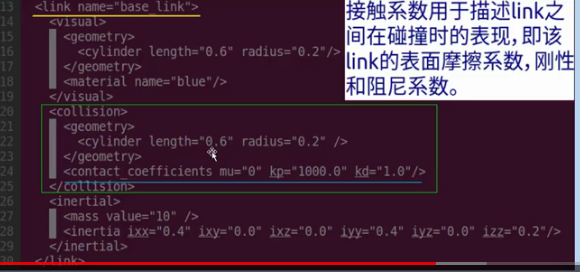
我们可以通过模型编程软件（Meshlab）来计算得到，它是一款处理和编辑三维模型的开源软件，它提供处理3d数字化工具产生的原始数据的功能。事实上，Meshlab软件可以计算一些不规则物体的旋转惯性矩阵数据，如果是一些基本的常规物体，可以通过以下网址来查看：

<https://en.wikipedia.org/wiki/List_of_moments_of_inertia>

我们也可以定义在link之间的接触系数，需要在collsion标签内定义的子元素，这里有3个属性需要指定：

Mu: 摩擦系数 kp:刚性系数 kd:阻尼系数

下面的代码示例如下，里面的系数仅作演示用，具体的系数需要根据实际情况测试得到：

****

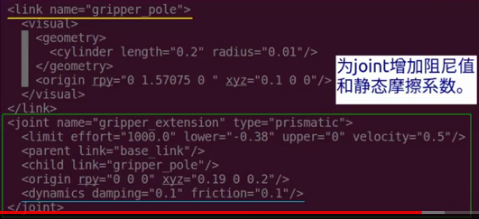
对于关节的动力学模型我们也需要指定dynamics标签，在关节运动时主要用到两个属性：

Friction：物体的静态摩擦力，对于prismatic关节，单位是：牛（N）。对于revolving关节，单位是（N.m）

Damping：物体的阻尼值。

如果关节的这些属性没有指定，那么默认情况就是0。

在代码中添加的示例如下，代码中数值仅供演示用，具体数值需要根据实际情况测量得到：

****

最终修改好的代码在rviz中演示查看结果，发现没有区别，主要是因为增加的这些属性主要是在进行物理仿真时用到的，在后面的gazebo中运行时就会发现这次增加的代码的效果。

**作业：**

学生尝试给之前的机器人模型添加一些运动属性和物理属性。

答案：

参考<实训源码>文件夹下的urdf\_tutorial-kinetic文件夹下的urdf文件夹下的：

06-flexible.urdf 和 07-physics.urdf 里面的代码。

06-flexible.urdf 主要增加了运动属性。

07-physics.urdf 主要增加了物理属性。