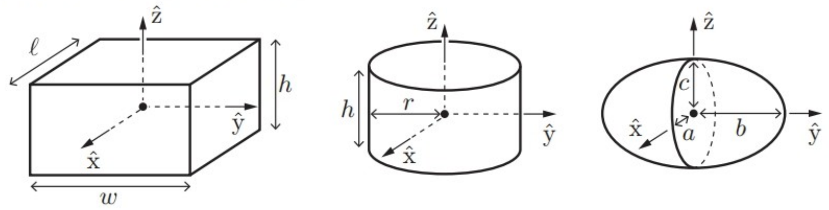
本次作业第一题
在第一个终端运行
roslaunch mbot_gazebo view_mbot_gazebo_empty_world.launch
第二个终端运行
roslaunch mbot_teleop mbot_teleop.launch
通过按键实现小车 j 旋转 i 前进 k 停止
此外也可以直接执行 learning_xacro 里的 launch 文件，通过 xacro 代替 urdf，即在终端输入
roslaunch learning_xacro display_mbot_base.launch

本次作业第二题
对于 camera
第一个终端输入
roslaunch mbot_gazebo view_mbot_with_camera_gazebo.launch
第二个终端输入
roslaunch learning_xacro display_mbot_base.launch
通过按键实现小车 j 旋转 i 前进 k 停止
第三个终端输入
qt_image_view
选择 camera_image_raw 话题，观察相机采集信息
对于 kinect
第一个终端输入
roslaunch mbot_gazebo view_mbot_with_kinect_gazebo.launch
第二个终端输入
roslaunch learning_xacro display_mbot_base.launch
通过按键实现小车 j 旋转 i 前进 k 停止
第三个终端输入
rosrun rviz rviz
添加 pointcloud2，选择/kinect/depth/points 话题
添加 robotmodel
固定坐标系为 odom，观察 kinect 采集的点云信息
对于 laser
第一个终端输入
roslaunch mbot_gazebo view_mbot_with_laser_gazebo.launch
第二个终端输入
roslaunch learning_xacro display_mbot_base.launch
通过按键实现小车 j 旋转 i 前进 k 停止
第三个终端输入
rosrun rviz rviz
添加 laserscan，选择/scan 话题
添加 robotmodel
固定坐标系为 odom，观察 laser 采集的雷达信息

实现步骤
1、先在工作空间的 src 文件夹下创建功能包
终端输入 catkin_create_pkg learning_xacro urdf xacro
2、在 learning_xacro 功能包下手动创建 xacro（存放 xacro）、meshes（存放 solidworks 文件）、launch（launch 启动文件）和 config（rviz 配置文件）四个文件夹
3、在 xacro 文件夹中，touch mbot_base_gazebo.xacro，创建文件夹 sensors，touch camera_gazebo.xacro，touch kinect_gazebo.xacro，touch lidar_gazebo.xacro

4、在 xacro 文件夹中，touch mbot_gazebo.xacro，调用 mbot_base_gazebo.xacro；
touch mbot_with_camera_gazebo.xacro，调用 camera_gazebo.xacro；
touch mbot_with_kinect_gazebo.xacro，调用 kinect_gazebo.xacro；
touch mbot_with_laser_gazebo.xacro，调用 lidar_gazebo.xacro；
5、核心代码为自己的机器人模型，主要包括四个步骤
第一，为上节已经含有 visual 外观的 link 添加惯性参数 matrix 和碰撞属性 collision
matrix 主要有三大类

下图分别是长方体、圆柱体、椭球体以质心坐标系为参考的质量惯性矩（转动惯量）计算公式：




rectangular parallelepiped:
volume $= abc$,
$\mathcal{I}_{xx} = \mathfrak{m}(w^2 + h^2)/12$,
$\mathcal{I}_{yy} = \mathfrak{m}(\ell^2 + h^2)/12$,
$\mathcal{I}_{zz} = \mathfrak{m}(\ell^2 + w^2)/12$

circular cylinder:
volume $= \pi r^2 h$,
$\mathcal{I}_{xx} = \mathfrak{m}(3r^2 + h^2)/12$,
$\mathcal{I}_{yy} = \mathfrak{m}(3r^2 + h^2)/12$,
$\mathcal{I}_{zz} = \mathfrak{m}r^2/2$

ellipsoid:
volume $= 4\pi abc/3$,
$\mathcal{I}_{xx} = \mathfrak{m}(b^2 + c^2)/5$,
$\mathcal{I}_{yy} = \mathfrak{m}(a^2 + c^2)/5$,
$\mathcal{I}_{zz} = \mathfrak{m}(a^2 + b^2)/5$

collision 与 visual 类似
第二，为 link 添加 gazebo 标签，格式固定
第三，为 joint 添加传动装置
第四，添加 gazebo 控制器插件
mbot_base_gazebo.xacro 代码内容如下：

```
<?xml version="1.0"?>
<robot name="mbot" xmlns:xacro="http://www.ros.org/wiki/xacro">

    <!-- PROPERTY LIST -->
    <xacro:property name="M_PI" value="3.1415926"/>
    <xacro:property name="base_mass"   value="2" />
    <xacro:property name="base_chang"   value="0.3" />
    <xacro:property name="base_kuan"   value="0.3" />
    <xacro:property name="base_gao"   value="0.02" />

    <xacro:property name="wheel_mass"   value="2" />
    <xacro:property name="wheel_radius" value="0.05"/>
    <xacro:property name="wheel_length" value="0.02"/>
    <xacro:property name="wheel_joint_x" value="0.1"/>
    <xacro:property name="wheel_joint_y" value="0.15"/>
    <xacro:property name="wheel_joint_z" value="0"/>

    <xacro:property name="caster_mass"    value="0.1" />
    <xacro:property name="caster_radius"  value="0.02"/>
    <xacro:property name="caster_joint_x" value="0.12"/>
    <xacro:property name="caster_joint_z" value="0.03"/>

    <!-- Defining the colors used in this robot -->
    <material name="yellow">
        <color rgba="1 0.4 0 1"/>
```

```
  </material>
  <material name="black">
    <color rgba="0 0 0 0.95"/>
  </material>
  <material name="white">
    <color rgba="1 1 1 0.9"/>
  </material>

  <!-- Macro for inertia matrix -->
  <xacro:macro name="sphere_inertial_matrix" params="m r">
    <inertial>
      <mass value="${m}" />
      <inertia ixx="${2*m*r*r/5}" ixy="0" ixz="0"
        iyy="${2*m*r*r/5}" iyz="0"
        izz="${2*m*r*r/5}" />
    </inertial>
  </xacro:macro>

  <xacro:macro name="cylinder_inertial_matrix" params="m r h">
    <inertial>
      <mass value="${m}" />
      <inertia ixx="${m*(3*r*r+h*h)/12}" ixy = "0" ixz = "0"
        iyy="${m*(3*r*r+h*h)/12}" iyz = "0"
        izz="${m*r*r/2}" />
    </inertial>
  </xacro:macro>

  <xacro:macro name="box_inertial_matrix" params="m x y z">
    <inertial>
      <mass value="${m}" />
      <inertia ixx="${m*(y*y+z*z)/12}" ixy = "0" ixz = "0"
        iyy="${m*(x*x+z*z)/12}" iyz = "0"
        izz="${m*(x*x+y*y)/12}" />
    </inertial>
  </xacro:macro>

  <!-- Macro for robot wheel -->
  <xacro:macro name="wheel" params="prefix reflect">
    <joint name="${prefix}_wheel_joint" type="continuous">
      <origin xyz="${-wheel_joint_x} ${reflect*wheel_joint_y*(-1)} ${wheel_joint_z}" rpy="0 0
0"/>
      <parent link="base_link"/>
      <child link="${prefix}_wheel_link"/>
      <axis xyz="0 1 0"/>
    </joint>

    <link name="${prefix}_wheel_link">
      <visual>
```

```
        <origin xyz="0 0 0" rpy="${M_PI/2} 0 0" />
        <geometry>
          <cylinder radius="${wheel_radius}" length = "${wheel_length}"/>
        </geometry>
        <material name="white" />
      </visual>
      <collision>
        <origin xyz="0 0 0" rpy="${M_PI/2} 0 0" />
        <geometry>
          <cylinder radius="${wheel_radius}" length = "${wheel_length}"/>
        </geometry>
      </collision>
      <cylinder_inertial_matrix  m="${wheel_mass}" r="${wheel_radius}" h="${wheel_length}" />
    </link>

    <gazebo reference="${prefix}_wheel_link">
      <material>Gazebo/Gray</material>
    </gazebo>

    <!-- Transmission is important to link the joints and the controller -->
    <transmission name="${prefix}_wheel_joint_trans">
      <type>transmission_interface/SimpleTransmission</type>
      <joint name="${prefix}_wheel_joint" >
        <hardwareInterface>hardware_interface/VelocityJointInterface</hardwareInterface>
      </joint>
      <actuator name="${prefix}_wheel_joint_motor">
        <hardwareInterface>hardware_interface/VelocityJointInterface</hardwareInterface>
        <mechanicalReduction>1</mechanicalReduction>
      </actuator>
    </transmission>
  </xacro:macro>

  <!-- Macro for robot caster -->
  <xacro:macro name="caster">
    <joint name="front_caster_joint" type="continuous">
      <origin xyz="${caster_joint_x} 0 ${-caster_joint_z}" rpy="0 0 0"/>
      <parent link="base_link"/>
      <child link="front_caster_link"/>
      <axis xyz="0 1 0"/>
    </joint>

    <link name="front_caster_link">
      <visual>
        <origin xyz="0 0 0" rpy="0 0 0"/>
        <geometry>
          <sphere radius="${caster_radius}" />
        </geometry>
```

```
      <material name="black" />
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0"/>
      <geometry>
        <sphere radius="${caster_radius}" />
      </geometry>
    </collision>
    <sphere_inertial_matrix  m="${caster_mass}" r="${caster_radius}" />
  </link>

  <gazebo reference="front_caster_link">
    <material>Gazebo/Black</material>
  </gazebo>
</xacro:macro>

<xacro:macro name="mbot_base_gazebo">
  <link name="base_footprint">
    <visual>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <box size="0.001 0.001 0.001" />
      </geometry>
    </visual>
  </link>
  <gazebo reference="base_footprint">
    <turnGravityOff>false</turnGravityOff>
  </gazebo>

  <joint name="base_footprint_joint" type="fixed">
    <origin xyz="0 0 ${base_gao/2+caster_radius*2}" rpy="0 0 0" />
    <parent link="base_footprint"/>
    <child link="base_link" />
  </joint>

  <link name="base_link">
    <visual>
      <origin xyz=" 0 0 0" rpy="0 0 0" />
      <geometry>
        <box size="${base_chang} ${base_kuan} ${base_gao}"/>
      </geometry>
      <material name="yellow" />
    </visual>
    <collision>
      <origin xyz=" 0 0 0" rpy="0 0 0" />
      <geometry>
        <box size="${base_chang} ${base_kuan} ${base_gao}"/>
      </geometry>
```

```
      </collision>
      <box_inertial_matrix  m="${base_mass}" x="${base_chang}" y="${base_kuan}" z="$
{base_gao}" />
    </link>

    <gazebo reference="base_link">
      <material>Gazebo/Blue</material>
    </gazebo>

    <wheel prefix="left"  reflect="-1"/>
    <wheel prefix="right" reflect="1"/>
    <caster/>
    <!-- controller -->
    <gazebo>
      <plugin name="differential_drive_controller"
          filename="libgazebo_ros_diff_drive.so">
        <rosDebugLevel>Debug</rosDebugLevel>
        <publishWheelTF>true</publishWheelTF>
        <robotNamespace>/</robotNamespace>
        <publishTf>1</publishTf>
        <publishWheelJointState>true</publishWheelJointState>
        <alwaysOn>true</alwaysOn>
        <updateRate>100.0</updateRate>
        <legacyMode>true</legacyMode>
        <leftJoint>left_wheel_joint</leftJoint>
        <rightJoint>right_wheel_joint</rightJoint>
        <wheelSeparation>${wheel_joint_y*2}</wheelSeparation>
        <wheelDiameter>${2*wheel_radius}</wheelDiameter>
        <broadcastTF>1</broadcastTF>
        <wheelTorque>30</wheelTorque>
        <wheelAcceleration>1.8</wheelAcceleration>
        <commandTopic>cmd_vel</commandTopic>
        <odometryFrame>odom</odometryFrame>
        <odometryTopic>odom</odometryTopic>
        <robotBaseFrame>base_footprint</robotBaseFrame>
      </plugin>
    </gazebo>
  </xacro:macro>

</robot>
```

6、添加 mbot_gazebo 功能包和 mbot_teleop 功能包
7、在 mbot_gazebo 功能包中选择对应的 launch 文件，将其修改为自己的机器人模型，即选择 learning_xacro 中的四个 xacro 文件分别为 mbot_gazebo.xacro，mbot_with_camera_gazebo.xacro，mbot_with_kinect_gazebo.xacro，mbot_with_laser_gazebo.xacro
8、最后按开头步骤进行执行终端命令