**智能轮椅仿真用户手册**

# 1.引言

## 1.1 编写目的

介绍智能轮椅仿真系统所提供的功能。

## 1.2 项目背景

软件系统名称：智能轮椅仿真系统

开发者：

用户：

## 1.3 手册的范围

使用范围：该系统基于ubuntu16.04+ROS Kinetic环境

软件版本：使用Gazebo 8软件

本手册主要介绍了智能轮椅仿真系统的软件概述、运行环境和使用说明等。

## 1.4 名词定义

urdf: Unified Robot Description Format，统一机器人描述格式。

xacro: xml macro，针对URDF模型的另外一种精简化、可复用、模块化的描述形式。

## 1.5 参考资料

1. <https://www.ros.org/>
2. <http://gazebosim.org/>
3. <http://sdformat.org/spec>
4. <https://www.theconstructsim.com/ros-qa-126-how-to-configure-the-differential-drive-ros-controller/>
5. <http://gazebosim.org/tutorials?tut=actor&cat=build_robot>

# 2 软件概述

## 2.1 系统框架

根据任务需求，智能轮椅仿真系统分为以下三部分：机器人状态模拟、传感器数据模拟和环境仿真场景。总体框架如图1所示。

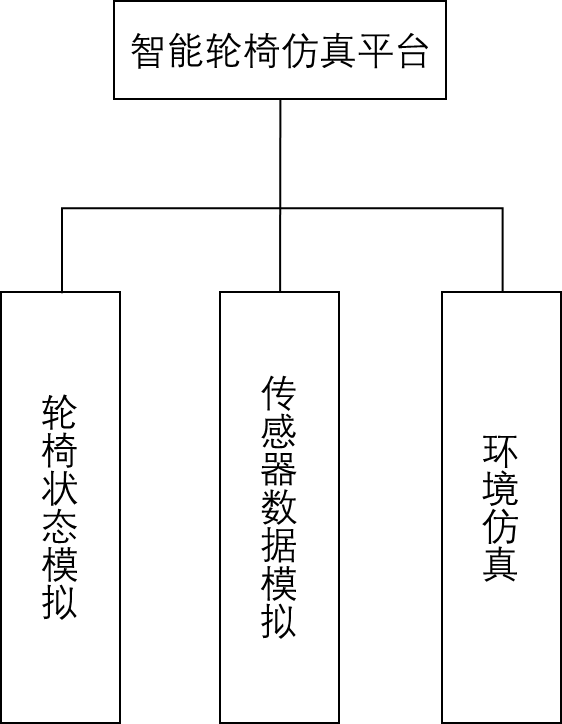


图1 系统总体框架

智能轮椅仿真主要提供轮椅环境仿真场景和机器人系统的三维仿真，能模拟机器人的状态和各传感器的数据，从而为移动机器人建图与定位提供验证平台。

## 2.2 系统结构

从构成上看，智能轮椅主要由电动轮椅加配计算机和传感器构成，结构图如图2所示。智能轮椅本体部分可简化为电机及减速器（本文以下简称为电机）、驱动轮以及轮椅车体三个部分。传感器部分包含二维激光传感器、三维激光传感器、视觉传感器等。其中，激光传感器均测量障碍物距离，这类传感器与视觉传感器一起作为外部传感器。

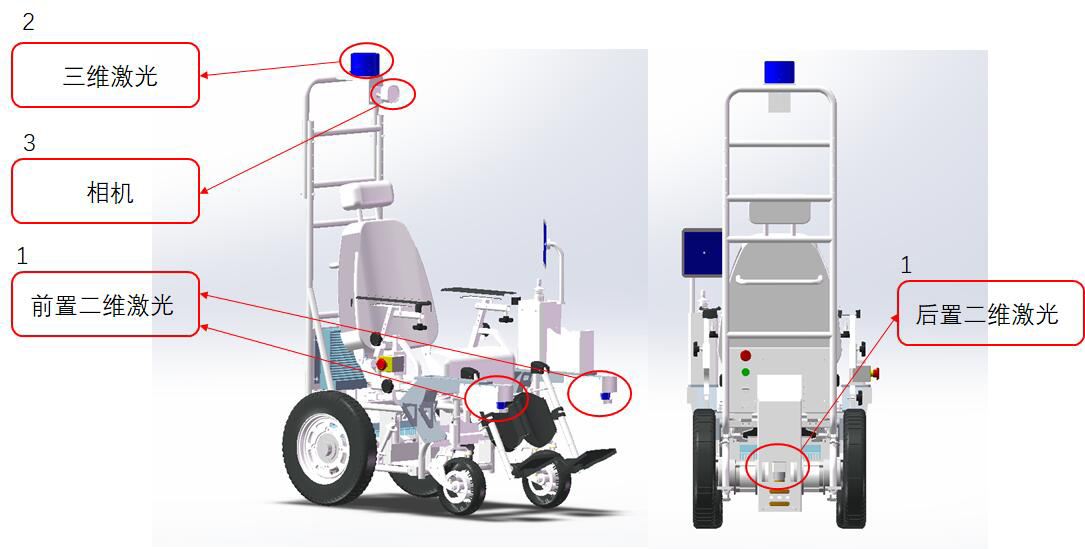


图2 智能轮椅结构图



图3 智能轮椅仿真平台层次图

对应于智能轮椅的通用硬件结构，图3展示了仿真平台的层次结构。

硬件层是智能轮椅在环境中运行时的各个硬件部分，包括环境、轮椅车体、传感器、执行器、输入设备以及传感器。

模型层上实现了其抽象模型，仿真平台对这些模型进行计算，并达到与真实硬件一致的特性。

抽象层在前述模型层的基础上，实现了轮椅的运动学和动力学。该层可以看成是轮椅底层硬件特性的抽象，不对应与具体实物，在仿真中可协调智能轮椅的多个部分。如通过编码器测量驱动轮转速完成速度闭环控制。同时，该层向上提供里程计及速度接口，该接口与真实轮椅的控制接口一致。

接口层将仿真平台的服务暴露给用户。GUI将仿真的场景直观地呈现于显示器上。用户通过修改XML文件，可快速修改仿真平台的参数配置。LaserScan为二维激光发布话题，PointCloud为三维激光点云信息，Image为相机发布的图像信息。最后，针对开发者，该平台采用消息传递方式开放传感器及控制接口。

# 3 运行环境

## 3.1 硬件设备

运行本仿真系统所要求的硬件配置：

1. 处理器：Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz 3.60GHz
2. 内存(RAM)：24.0GB
3. 鼠标 键盘

## 3.2 支持软件

运行本仿真系统所需要的支持软件：

　　(1)操作系统：Ubuntu 16.04

　　(2)机器人操作系统：ROS Kinetic

　　(3)仿真软件：Gazebo 8

　　(4)软件开发工具：Microsoft Visual Studio Code 2019

# 4.使用说明

## 4.1 安装与初始化

本系统是基于ROS Kinetic机器人操作系统以及Gazebo 8软件，因此，在使用改系统前需要预先安装好ros、Gazebo等。

本系统所有源代码均托管在Github上：xx

在最初使用时，需要将代码通过git clone下载至catkin\_ws/src目录下。

## 4.2 文件结构

源码包括wheelchair功能包：velodyne\_description、velodyne\_gazebo\_plugins、wheelchair\_witharm\_description。如图4所示。

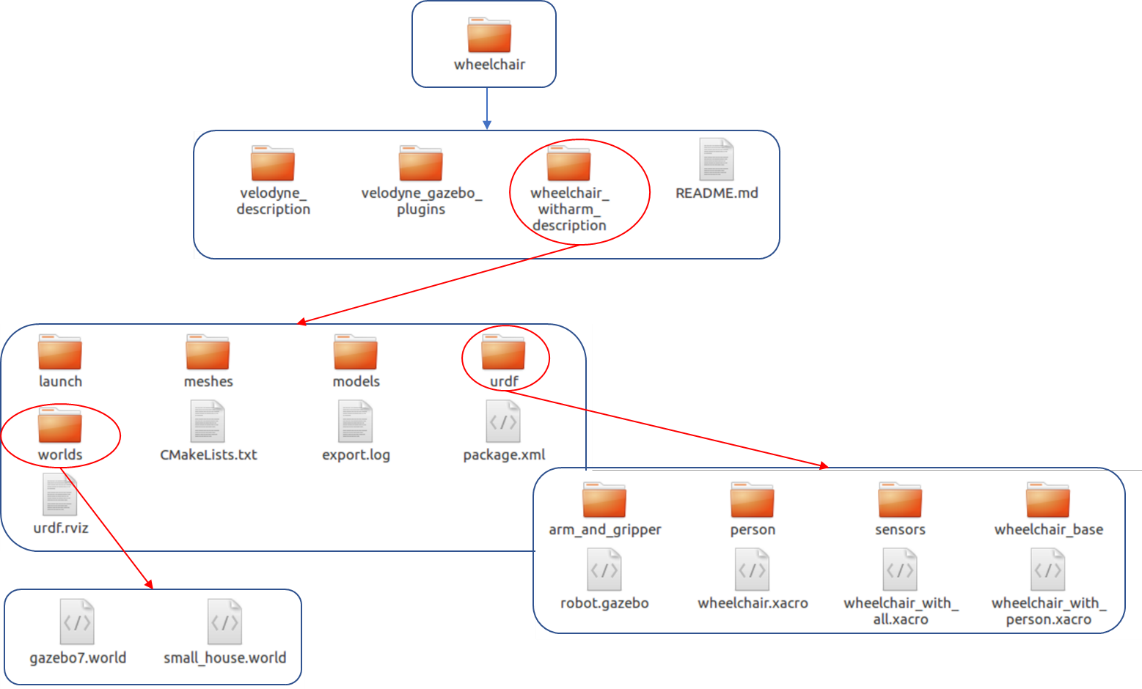


图4 文件结构

* wheelchair\_witharm\_description主要描述轮椅模型，包含对轮椅刚体外观、物理属性、关节类型等方面的描述。wheelchair\_witharm\_description功能包中包含urdf、meshes两个文件夹。
* urdf：用于存放机器人模型的URDF或xacro文件。urdf文件夹又包括wheelchair.xacro、wheelchair\_withperson.xacro、wheelchair\_with\_all.xacro等。
  + robot.gazebo：相关插件配置
* wheelchair.xacro：轮椅模型及其配置
* wheelchair\_with\_person.xacro：带有人的轮椅模型及其配置
* wheelchair\_with\_all.xacro：带有人和机械臂的轮椅模型及其配置
* meshes：用于放置URDF或xacro中引用的模型渲染文件。
* worlds主要描述轮椅所运行的环境，即医院环境。
  + gazebo7.world是不包括动态行人的环境，在gazebo7上可运行。
  + small\_house.world是包括动态行人的环境，在gazebo8上运行。
* velodyne\_description主要是velodyne描述模型。
* velodyne\_gazebo\_plugins主要是velodyne插件。

## 运行说明

### 4.3.1 环境模型



室内环境模型，包括餐厅、客厅、卧室和充电桩。

* 动态行人配置







### 4.3.2 轮椅模型

使用以下命令加载轮椅模型。



通过修改wheelchair/wheelchair\_witharm\_description/launch/gazebo.launch文件中的红色方框框选出的文件，可以打开不同的轮椅模型。



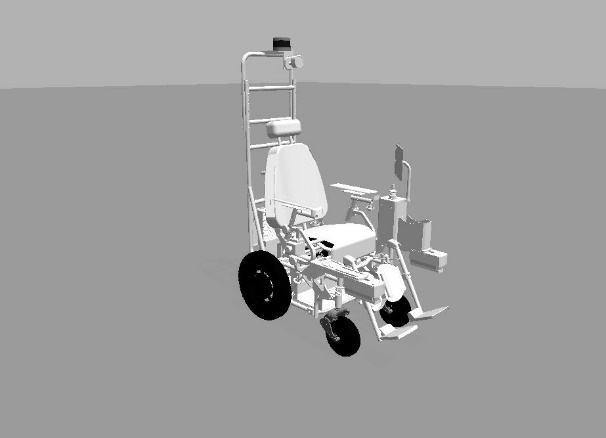
三种不同的轮椅模型：

wheelchair.xacro：轮椅模型及其配置

wheelchair\_with\_person.xacro：带有人的轮椅模型及其配置

wheelchair\_with\_all.xacro：带有人和机械臂的轮椅模型及其配置

启动效果分别如图5所示。



(a)轮椅模型

(b)带有人的轮椅模型 (c)带有人和机械臂的轮椅模型

图5 不同模型效果图

### 传感器参数

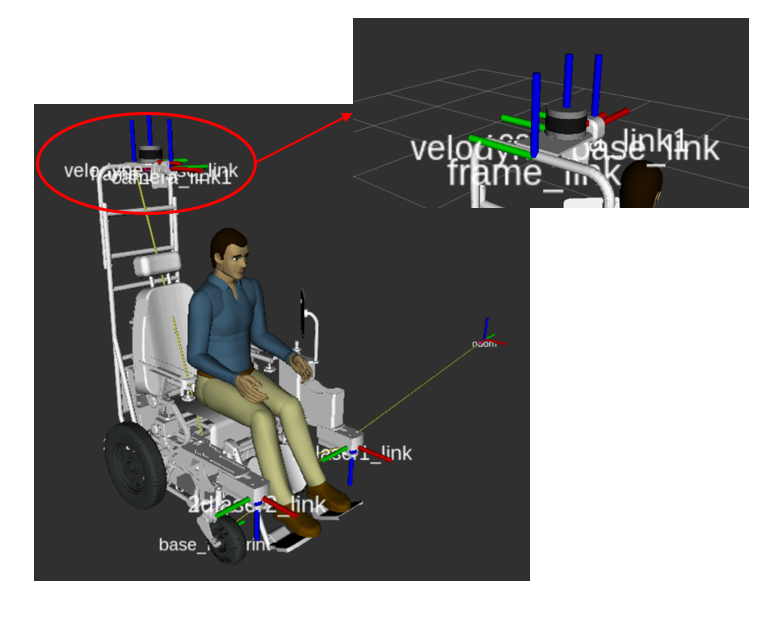


图7 传感器配置

#### 传感器配置

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 传感器名称 | child\_link | parent\_link | xyz | rpy |
| 二维激光 | 2dlaser1\_link | base\_link | "0.60054 0.29449 0.29895" | "-3.1416 0 0" |
| 2dlaser2\_link | base\_link | "0.60054 -0.29448 0.29896" | "3.1416 0 0" |
| 2dlaser3\_link | base\_link | "-0.50189 0 0.04275" | "3.1416 0 3.1416" |
| 三维激光 | velodyne\_base\_link | frame\_link | "0.13327 0 0.022" | "0 0 0" |
| 相机 | camera\_link | frame\_link | "0.27927 0 -0.04299" | "0 0 0" |

#### 二维激光雷达仿真

在SLAM和导航等机器人应用中，为了获取更精确的环境信息，会使用激光雷达作为主要传感器。在本系统中，为智能轮椅添加3个二维激光传感器，包括2个前置倒立二维激光和一个后置倒立二维激光。

1. 为hokuyo模型添加Gazebo插件

（以前置左侧二维激光为例，其他类似）本系统使用的激光雷达是hokuyo，在wheelchair\_witharm\_description/urdf/robot.gazebo中添加上述<gazebo>标签。



1. 查看二维激光数据

在Rviz中查看二维激光数据：

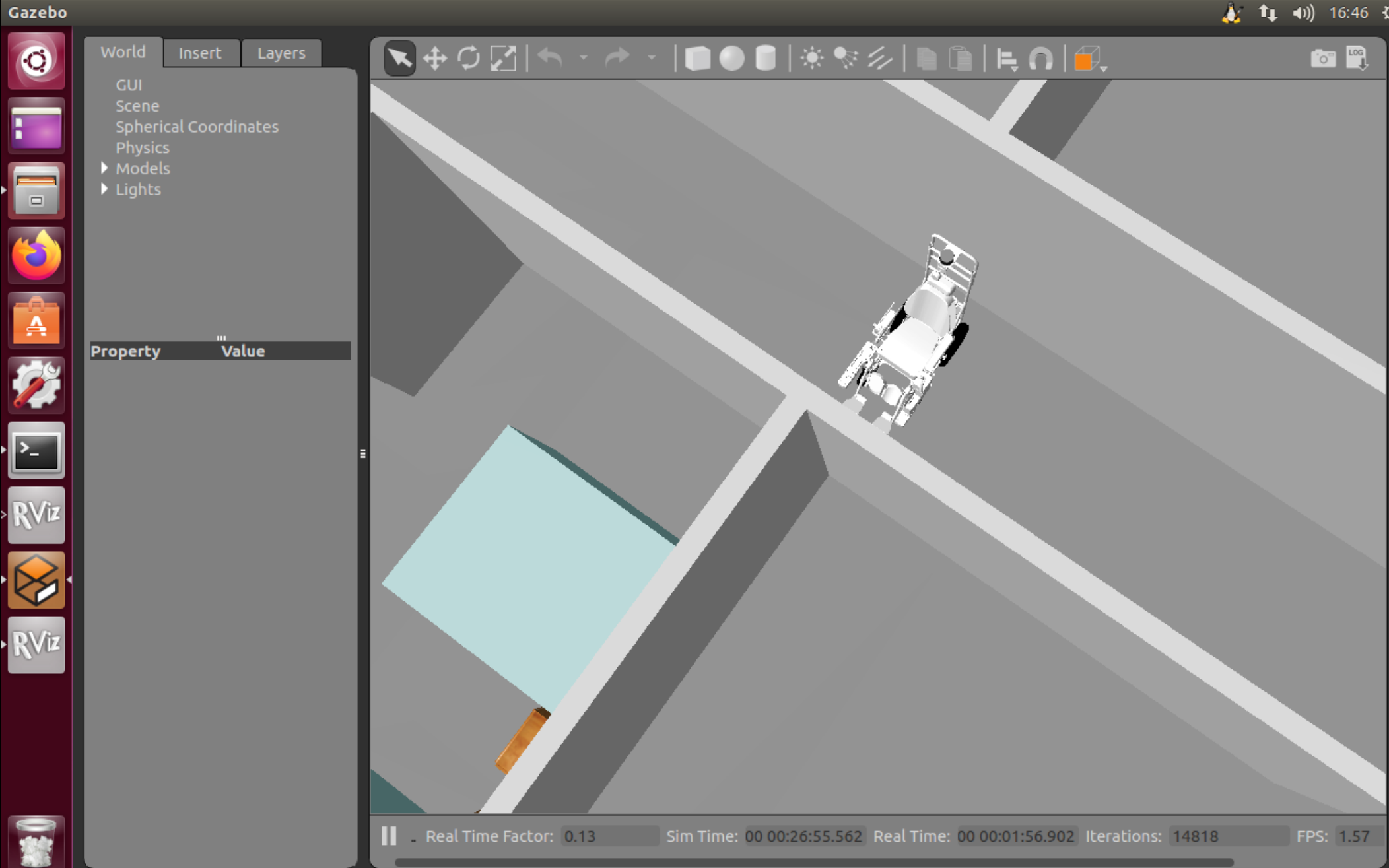
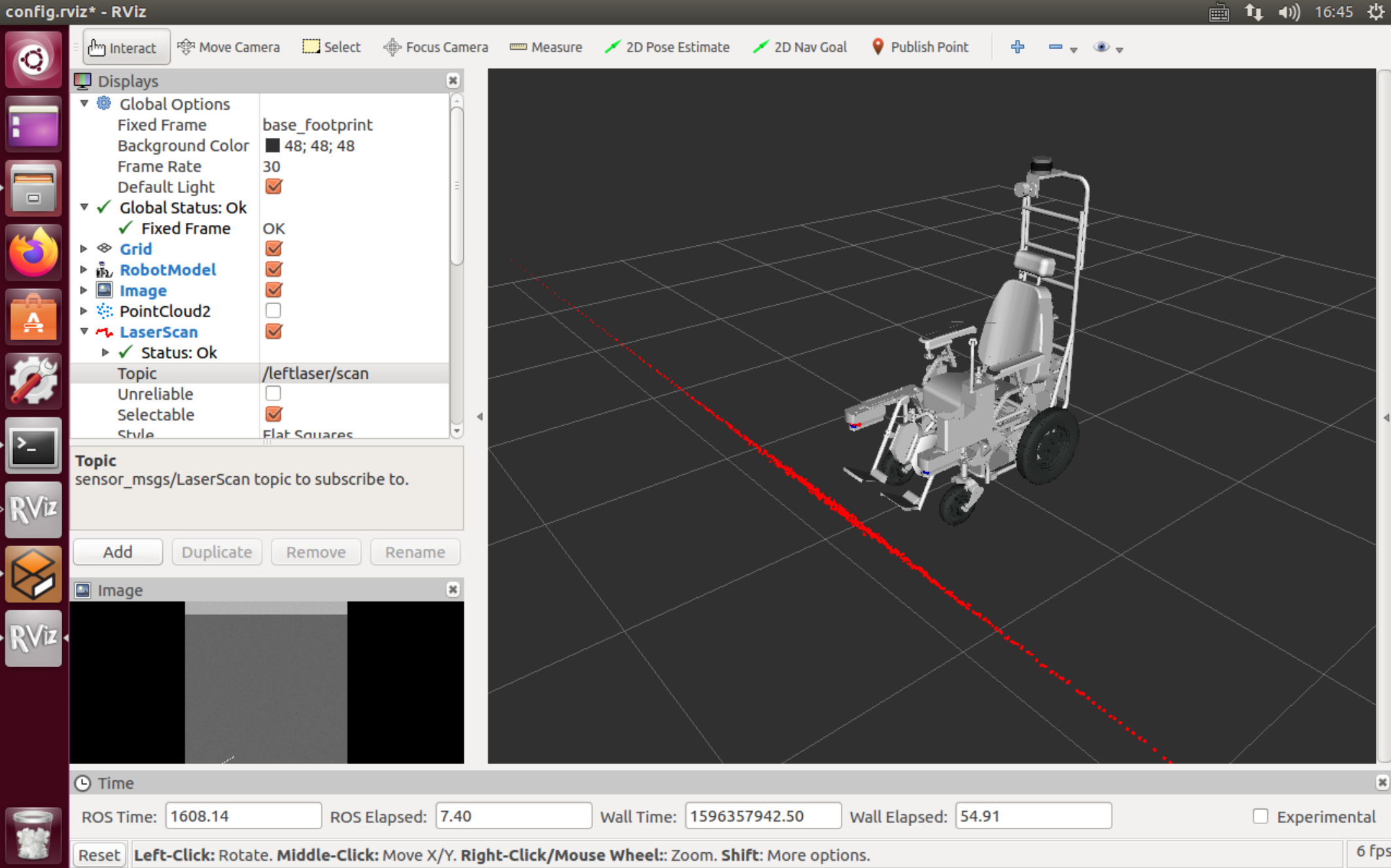
 

图7 仿真二维激光雷达发布的激光信息

#### 三维激光雷达仿真

1. velodyne模型

本系统采用的三维激光为VLP-16，具体velodyne激光测距仪的URDF文件和Gazebo插件均来自于<https://bitbucket.org/DataspeedInc/velodyne_simulator/src/master/>

使用时需要将其复制至wheelchair的目录下，（在本系统中已将其复制在wheelchair目录下，如下图所示）。



图8 velodyne模型

对于三维激光的配置主要包括以下相关参数， -180°-+180°的检测范围，单圈6000个采样点，最远130米的检测范围等。所发布的激光雷达的话题是“/velodyne\_points”。



1. 查看三维激光数据

在Rviz中查看二维激光数据：

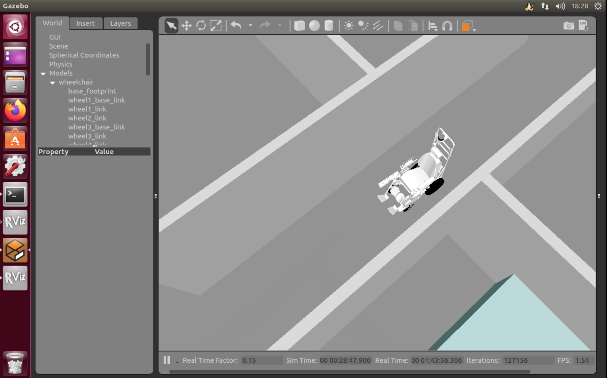
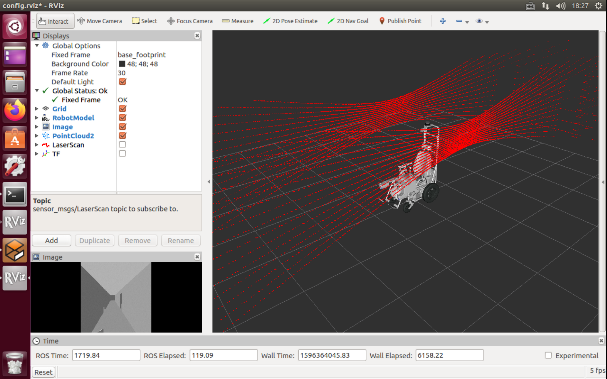
 

图9 三维激光仿真数据

#### 摄像头仿真

1. camera模型

1）普通相机

与激光传感器相同，在wheelchair\_witharm\_description/urdf/robot.gazebo添加以下<gazebo>相关标签。



2）realsense相机



2.查看摄像头数据

使用rqt工具查看摄像头相关数据：



选择仿真摄像头发布的图像话题“/wheelchair/camera/image\_raw”，即可看见如图10所示的图像信息。

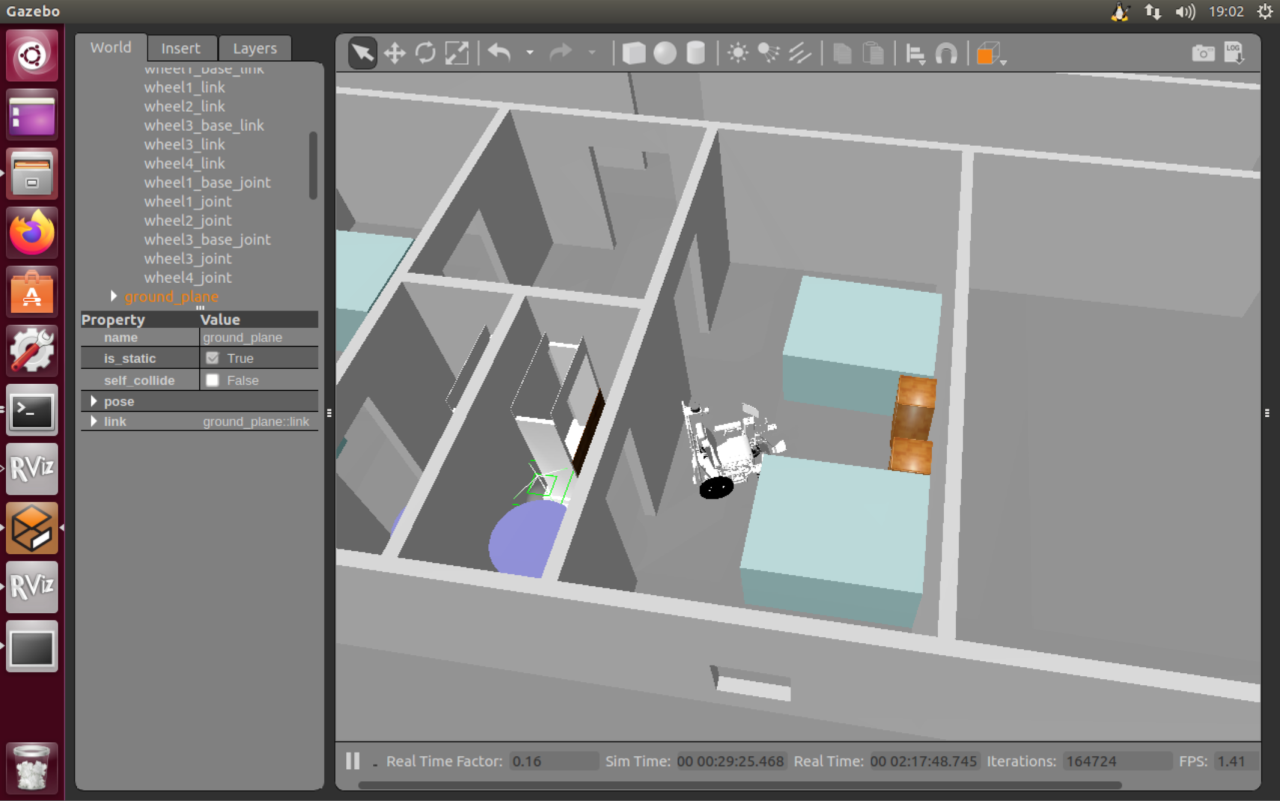
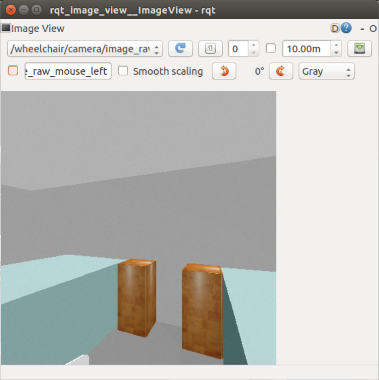
 

图10 相机数据

### 4.3.4 系统发布的话题

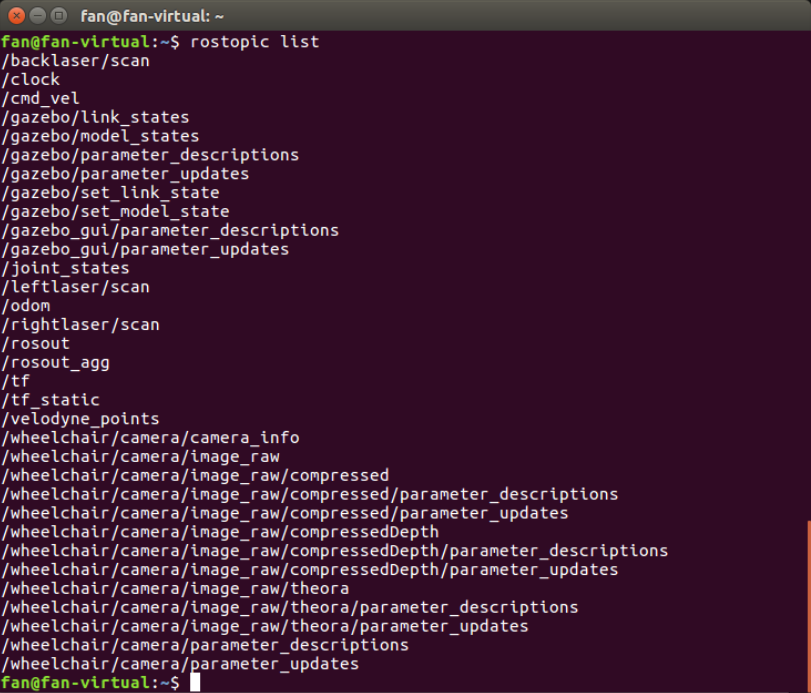


图6 系统发布的话题

运行该仿真系统，使用rostopic list命令可查看系统当前的话题列表，如图6所示。其中，

/leftlaser/scan：前置左侧二维激光发布的话题名

/rightlaser/scan：前置右侧二维激光发布的话题名

/backlaser/scan：后置二维激光发布的话题名

/velodyne\_points：三维激光发布的话题名

/wheelchair/camera/image\_raw：相机发布的话题名

/cmd\_vel：控制器订阅的速度控制指令

### 4.3.5 控制器



本系统采用diff\_drive\_controller控制器，其中wheelSeparation是指轮间距，wheelDiameter是车轮直径，wheelTorque是车轮转矩。

* 描述：控制器的主要输入是控制器名称空间中的geometry\_msgs :: Twist主题。
* 订阅的主题：cmd\_vel（geometry\_msgs / Twist）速度命令。
* 发布的话题：odom（nav\_msgs / Odometry）根据硬件反馈计算的里程表；

/ tf（tf / tfMessage）从odom转换为base\_footprint。

### 4.3.6 键盘控制——移动轮椅

首先，查看系统当前的话题列表（见图6）

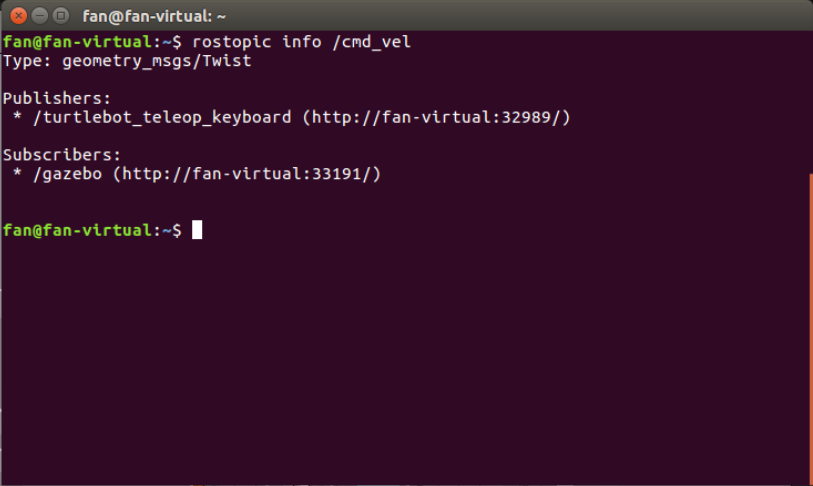


图11 查看ROS系统中的话题列表

由图11可以看出，Gazebo仿真中已开始订阅/cmd\_vel话题。

轮椅模型中存在diff\_drive\_controller控制器，可以使用差速控制器实现轮椅运动。运行键盘控制节点，发布该画图的速度控制消息，轮椅会在Gazebo中开始运动。

使用turtlebot自带的键盘控制节点，需要对/turtlebot/src/turtlebot/turtlebot\_teleop/launch/keyboard\_teleop.launch文件进行如下修改：



运行以下命令，启动键盘控制节点：



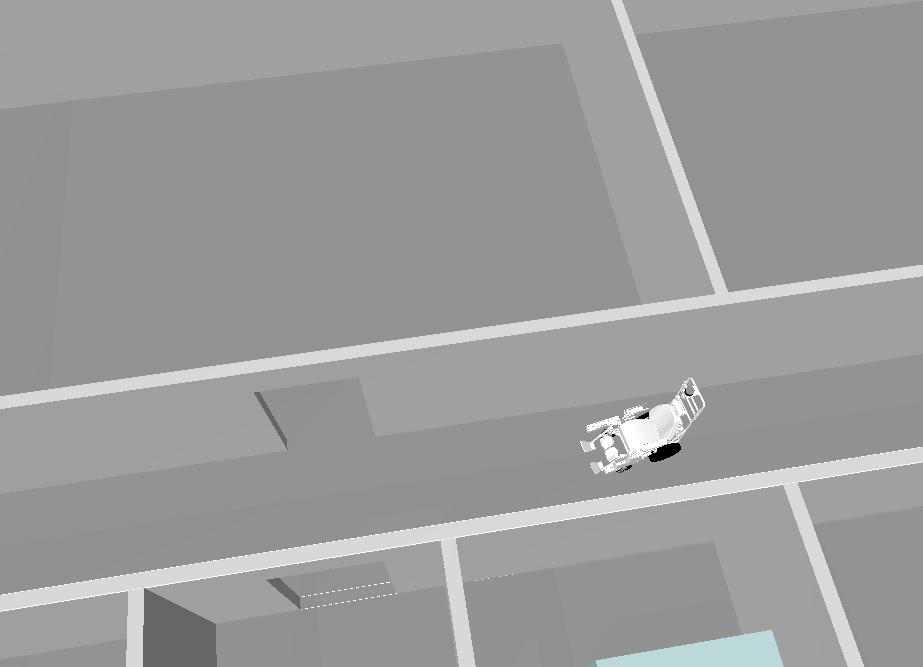


图12 通过键盘在轮椅在Gazebo中运动

# 注意事项：

1. 环境中添加了动态行走的人，GAZEBO版本需求为8+。
2. 机械臂不能在GAZEBO8上正常运行，机械臂需要在ubuntu16.04+GAZEBO7上运行。
3. 问题：二维激光扫不到动态的人？

答：1）将二维激光插件改为gup\_laser.so，但是这种方法会和三维激光的gup产生冲突

2）为动态行人添加collision

1. 关于开门问题：

使用apply force打开门