Tutorial: Creating a Velodyne sensor
The Future of Robocup Rescue Simulation Workshop
March 1, 2016
Humanoids

Atlas

http://gazebosim.org/tutorials?cat=drcsim
Issue: No open-source controller

Nao (via Robocup soccer simulation league)

https://github.com/robocup-logistics/gazebo-rcll
Issue: Again, no open-source controller

Others

Hubo, Robonaut 2, Valkyrie
Velodyne HDL-32

Tutorial for demonstration purposes

Collect information

- http://velodynelidar.com/hdl-32e.html
- Physical properties
  Dimension, mass, joints
- Sensor properties
  Type of sensor (sonar, camera, lidar)
  Accuracy, range, etc.
Create the SDF model

http://gazebosim.org/tutorials?tut=guided_i1

Start simple and build the model progressively
  Use simple shapes
  Ignore joints, inertia

Use available tools
  Start simulation paused (-u command line argument)
  Visualize model properties: collision, joint, inertia
  Joint command widget to verify joint properties

Future tools
  Graphical model editor
  Plotting utility
Create the SDF model: Steps

http://gazebosim.org/tutorials?tut=guided_i1

Step 1: Create a simple model

Step 2: Add Inertia

Step 3: Add the joint

Step 4: Add the sensor
Model appearance

http://gazebosim.org/tutorials?tut=guided_i2

Importance

- Improve user experience
- Improve sensor data, such as from cameras

How?

- Use pre-generated 3D meshes, create your own, use an artist
- Same applies for textures

Advanced

- Normal maps, for improved lighting effects
- Custom GL shaders
Model appearance: Steps

http://gazebo.org/tutorials?tut=guided_i2

Step 1: Mesh Acquisition

Step 2: Add meshes to SDF

Step 3: Textures
Sensor Noise

http://gazebosim.org/tutorials?tut=guided_i3

Why?

Data from simulation can be too perfect
Modify output to match physical properties
Lens effects, noise

How?

A: Use Gazebo’s internal noise models
B: Write a plugin to modify sensor data
C: Pass data through a ROS node
Sensor Noise: Steps

http://gazeboSim.org/tutorials?tut=guided_i3

**Step 1: Visualize data**

**Step 2: Add noise**
Contribute model
http://gazebosim.org/tutorials?tut=guided_i4

Why?
- Let Gazebo manage your resources
  - Auto-download
  - Share materials and meshes between models
- Don’t re-invent the wheel
  - Other users benefit from your contribution

How?
- Fork https://bitbucket.org/osrf/gazebo_models
- Add your model
- Create pull-request
Plugin

http://gazebosim.org/tutorials?tut=guided_i5

Purpose

Attach custom code to simulation

Types

Model: Control the model & its joints
Sensor: Modify data generation
World: Monitor/modify models and world properties
System: Control system startup

API: http://gazebosim.org/api
Plugin: Steps

http://gazeboSim.org/tutorials?tut=guided_i5

Step 1: Create workspace
Step 2&3: Write the plugin & build script
Step 4: Attach plugin to model
Step 5: Create an API
Step 6: Test
Connect to ROS
http://gazebosim.org/tutorials?tut=guided_i6

Access to the ROS ecosystem
   Rviz, MoveIt, RQT, SLAM, etc

Approaches
   Use or write a plugin for http://wiki.ros.org/gazebo_ros_pkgs
   Directly add ROS to your Gazebo plugin

Step 1: Add ROS transport
Step 2: Control Velodyne