ROS Introductory Course
Tecnológico de Monterrey

Cheat sheet

Robot Modelling:

List of inertias

Depending on the design and the type of visualisation, the inertia matrix of the solid object must be added. The following link is for reference.

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

Gazebo tags

(For more information, visit the following link: http://wiki.ros.org/ros_control)

The following are the parameters/parameters that can be added/modified in gazebo tags.

Elements for Links

Name	Type	Description
material	value	Material of visual element
gravity	bool	Use gravity
dampingFactor	double	Exponential velocity decay of the link velocity - takes the value and multiplies the previous link velocity by (1-dampingFactor).
maxVel	double	maximum contact correction velocity truncation term.
minDepth	double	minimum allowable depth before contact correction impulse is applied
mu1	double	Friction coefficients µ for the principal contact directions along the contact surface as defined by the Open Dynamics Engine (ODE) (see parameter descriptions in ODE's user guide)
mu2		
fdir1	string	3-tuple specifying direction of mu1 in the collision local reference frame.
kp	double	Contact stiffness k_p and damping k_d for rigid body contacts as defined by ODE (ODE uses erp and cfm but there is a mapping between erp/cfm and stiffness/damping)
kd		

selfCollide	bool	If true, the link can collide with other links in the model.
maxContacts	int	Maximum number of contacts allowed between two entities. This value overrides the max_contacts element defined in physics.
laserRetro	double	intensity value returned by laser sensor.

Elements for Joints

Name	**Type**	**Description**		
stopCfm	double	Joint stop constraint force mixing (cfm) and error reduction parameter (erp) used by ODE		
stopErp				
provideFeedback	bool	Allows joints to publish their wrench data (force-torque) via a Gazebo plugin		
implicitSpringDamper	bool	If this flag is set to true, ODE will use ERP and CFM to simulate damping. This is a more stable numerical method for damping than the default damping tag. The cfmDamping element is deprecated and should be changed to implicitSpringDamper.		
springStiffness			double	Spring stiffness in N/m.
springReference	double	Equilibrium position for the spring.		
cfmDamping				
fudgeFactor	double	Scale the excess for in a joint motor at joint limits. Should be between zero and one.		

Elements for Transmissions

(taken for reference from https://wiki.ros.org/urdf/XML/Transmission)

The transmission has the following elements:

- **<type>** (one occurrence)
    - Specifies the transmission type.
- **<joint>** (one or more occurrences)
    - A joint the transmission is connected to. The joint is specified by its **name** attribute, and the following sub-elements:
    - **<hardwareInterface>** (one or more occurrences)

        - Specifies a supported joint-space hardware interface. Note that the value of this tag should be **EffortJointInterface** when this transmission is loaded in Gazebo and **hardware_interface/EffortJointInterface** when this transmission is loaded in RobotHW.
- **<actuator>** (one or more occurrences)
    - An actuator the transmission is connected to. The actuator is specified by its **name** attribute, and the following sub-elements:
    - **<mechanicalReduction>** (optional)
        - Specifies a mechanical reduction at the joint/actuator transmission. This tag may not be needed for all transmissions.
    - **<hardwareInterface>** (optional) (one or more occurrences)
        - Specifies a supported joint-space hardware interface. Note that **<hardwareInterface>** tag should only be specified here for ROS releases prior to Indigo. The correct place to specify this tag is in **<joint>** tag. More details about this can be found here.

Example for transmission

```
<transmission name="simple_trans">
  <type>transmission_interface/SimpleTransmission</type>
  <joint name="foo_joint">
    <hardwareInterface>EffortJointInterface</hardwareInterface>
  </joint>
  <actuator name="foo_motor">
    <mechanicalReduction>50</mechanicalReduction>
    <hardwareInterface>EffortJointInterface</hardwareInterface>
  </actuator>
</transmission>
```

Highlight!!: It is important to remember that ROS control counts with 3 types of joint interface. This parameter specifies how the controller interfaced with the joint :

- hardware_interface::JointStateInterface
- hardware_interface::EffortJointInterface
- hardware_interface::VelocityJointInterface

The example above is an example of a EffortJoint interface.

Gazebo plugin

(taken from http://gazebosim.org/tutorials/?tut=ros_control)

You need to add the following lines of code :

```
<gazebo>
  <plugin name="gazebo_ros_control" filename="libgazebo_ros_control.so">
    <robotNamespace>/MYROBOT</robotNamespace>
  </plugin>
</gazebo>
```

Here are the description of additional ellements that can be added:

- <robotNamespace>: The ROS namespace to be used for this instance of the plugin, defaults to robot name in URDF/SDF
- <controlPeriod>: The period of the controller update (in seconds), defaults to Gazebo's period
- <robotParam>: The location of the robot_description (URDF) on the parameter server, defaults to '/robot_description'
- <robotSimType>: The pluginlib name of a custom robot sim interface to be used (see below for more details) defaults to 'DefaultRobotHWSim'

Gazebo Interface reference

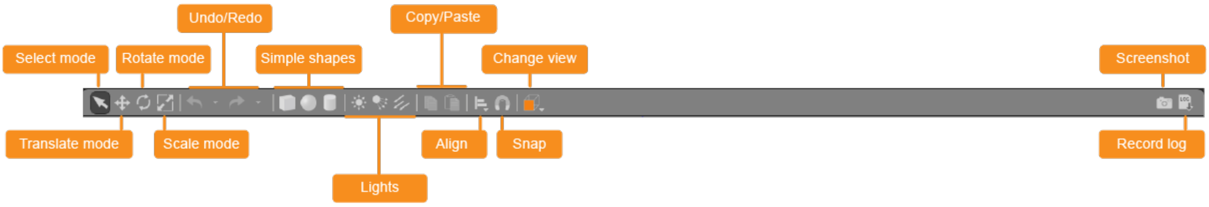
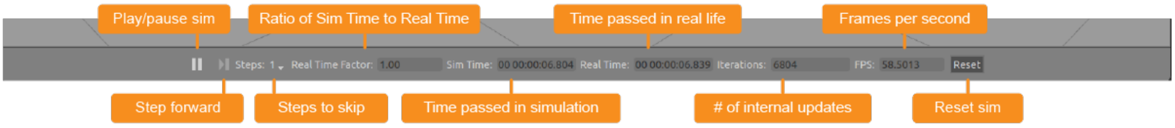
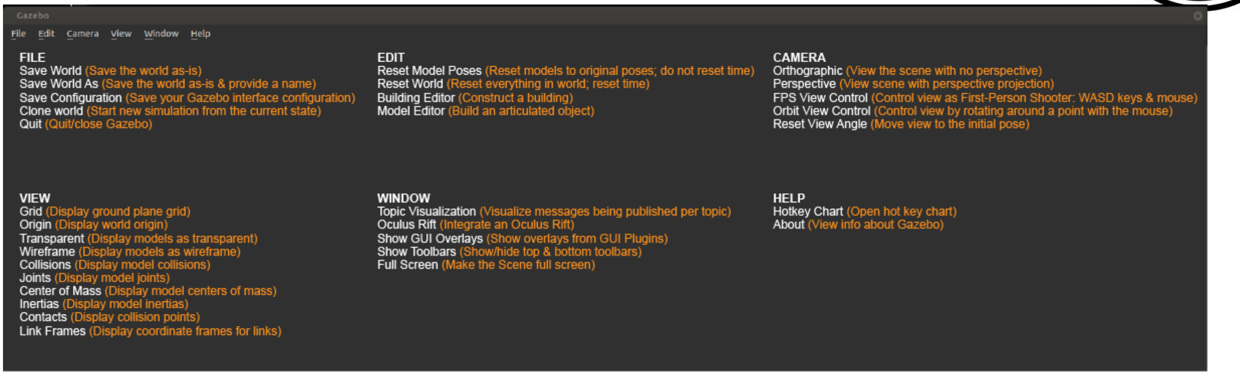


Figure 1: Upper tool bar(taken from gazebosim.org)



Steps: 1 Real Time Factor: 1.00 Sim Time: 00 00:00:06.804 Real Time: 00 00:00:06.839 Iterations: 6804 FPS: 58.5013

Figure 2: Lower tool bar(taken from gazebosim.org)



FILE
Save World (Save the world as-is)
Save World As (Save the world as-is & provide a name)
Save Configuration (Save your Gazebo interface configuration)
Clone world (Start new simulation from the current state)
Quit (Quit/close Gazebo)

VIEW
Grid (Display ground plane grid)
Origin (Display world origin)
Transparent (Display models as transparent)
Wireframe (Display models as wireframe)
Collisions (Display model collisions)
Joints (Display model joints)
Center of Mass (Display model centers of mass)
Inertias (Display model inertias)
Contacts (Display collision points)
Link Frames (Display coordinate frames for links)

EDIT
Reset Model Poses (Reset models to original poses; do not reset time)
Reset World (Reset everything in world; reset time)
Building Editor (Construct a building)
Model Editor (Build an articulated object)

WINDOW
Topic Visualization (Visualize messages being published per topic)
Oculus Rift (Integrate an Oculus Rift)
Show GUI Overlays (Show overlays from GUI Plugins)
Show Toolbars (Show/hide top & bottom toolbars)
Full Screen (Make the Scene full screen)

CAMERA
Orthographic (View the scene with no perspective)
Perspective (View scene with perspective projection)
FPS View Control (Control view as First-Person Shooter: WASD keys & mouse)
Orbit View Control (Control view by rotating around a point with the mouse)
Reset View Angle (Move view to the initial pose)

HELP
Hotkey Chart (Open hot key chart)
About (View info about Gazebo)

Figure 3: Menu options quick reference (taken from gazebosim.org)

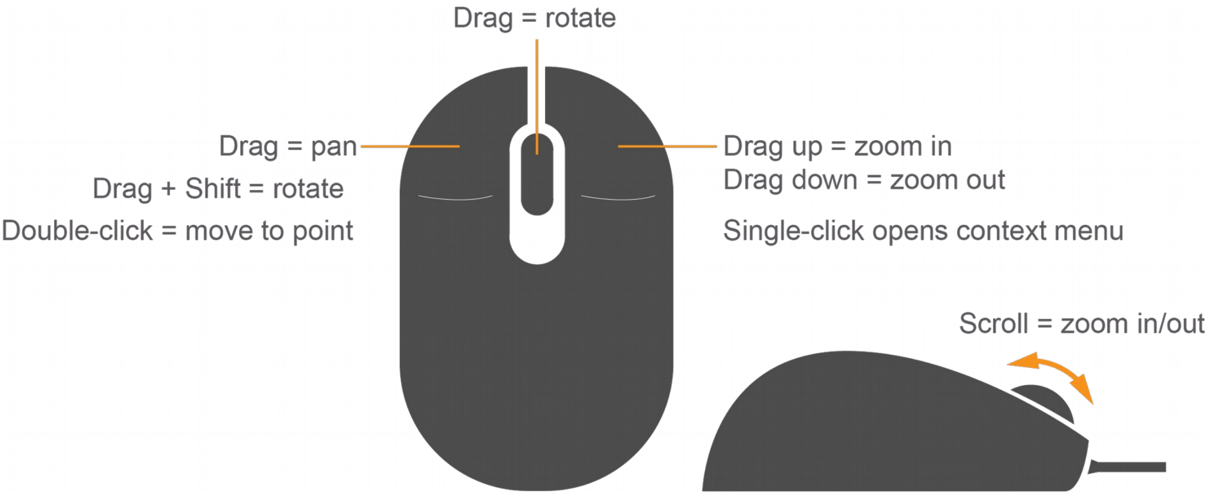


Figure 4: Mouse controls reference

For more info, check: http://gazebosim.org/tutorials?tut=guided_b2&cat=

FAQ:

Is URDF modelling compatible with closed link robots ?

This robot architecture is not natively supported by URDF, but some researchers have found ways to adapt it. Here are some references found about it:

Development of a URDF file for simulation and programming of a **delta robot** using **ROS**

RT Arrazate - Santiago de Querétaro, 2017 - 207.249.123.243

Delta robot controlled by **robotic** operating system

DR Rivas-Lalaleo, EE Galarza-Zambrano… - Iteckne, 2015 - scielo.org.co