The Robot Operating System

Day 3 – Mobile robots, SLAM and navigation

Dr. Murilo Fernandes Martins

Department of Electrical Engineering
Centro Universitário da FEI

29 January 2014

Outline

Gazebo simulator

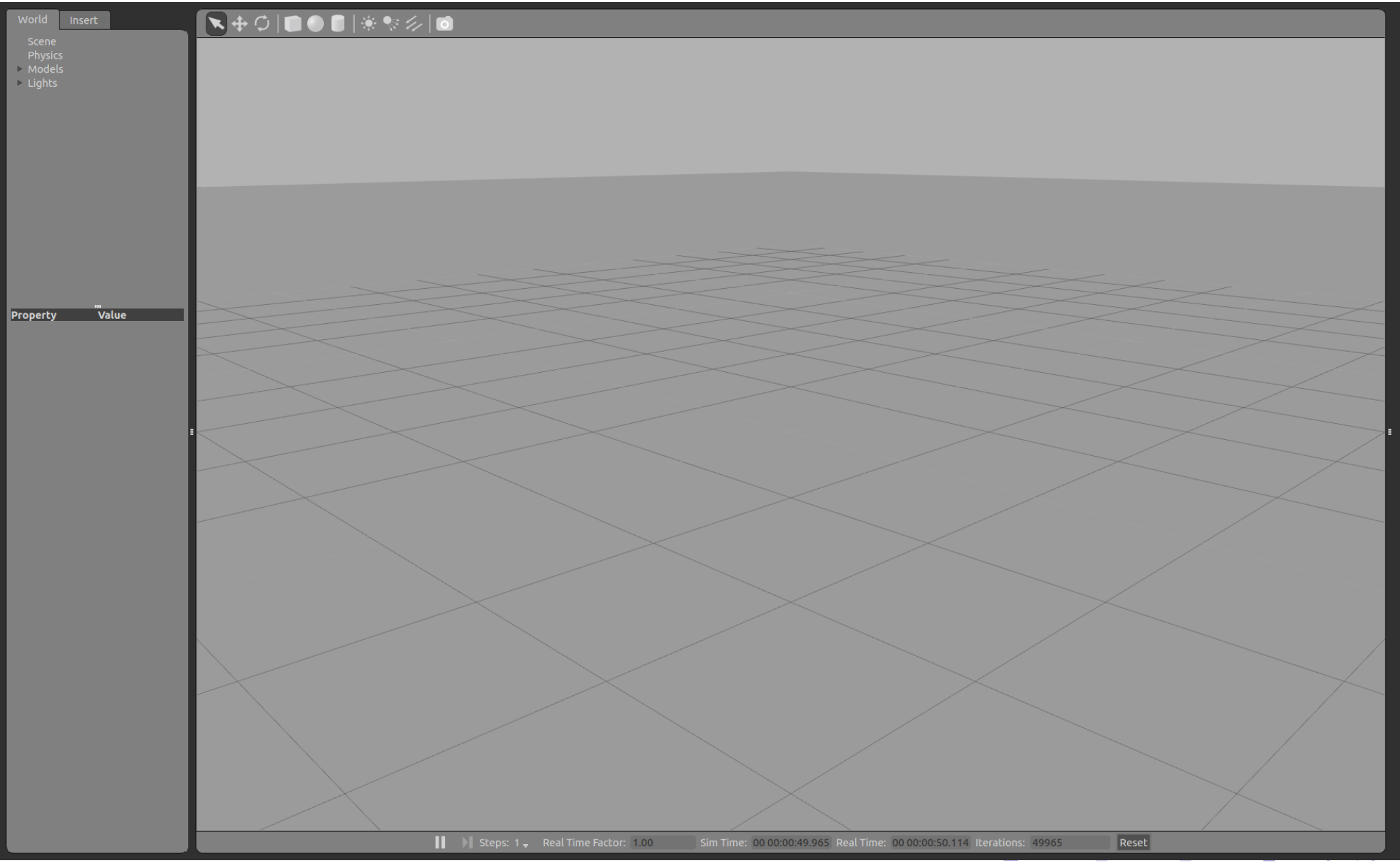
Launching Gazebo with an empty world

```
1 $ roslaunch gazebo_ros empty_world.launch
```

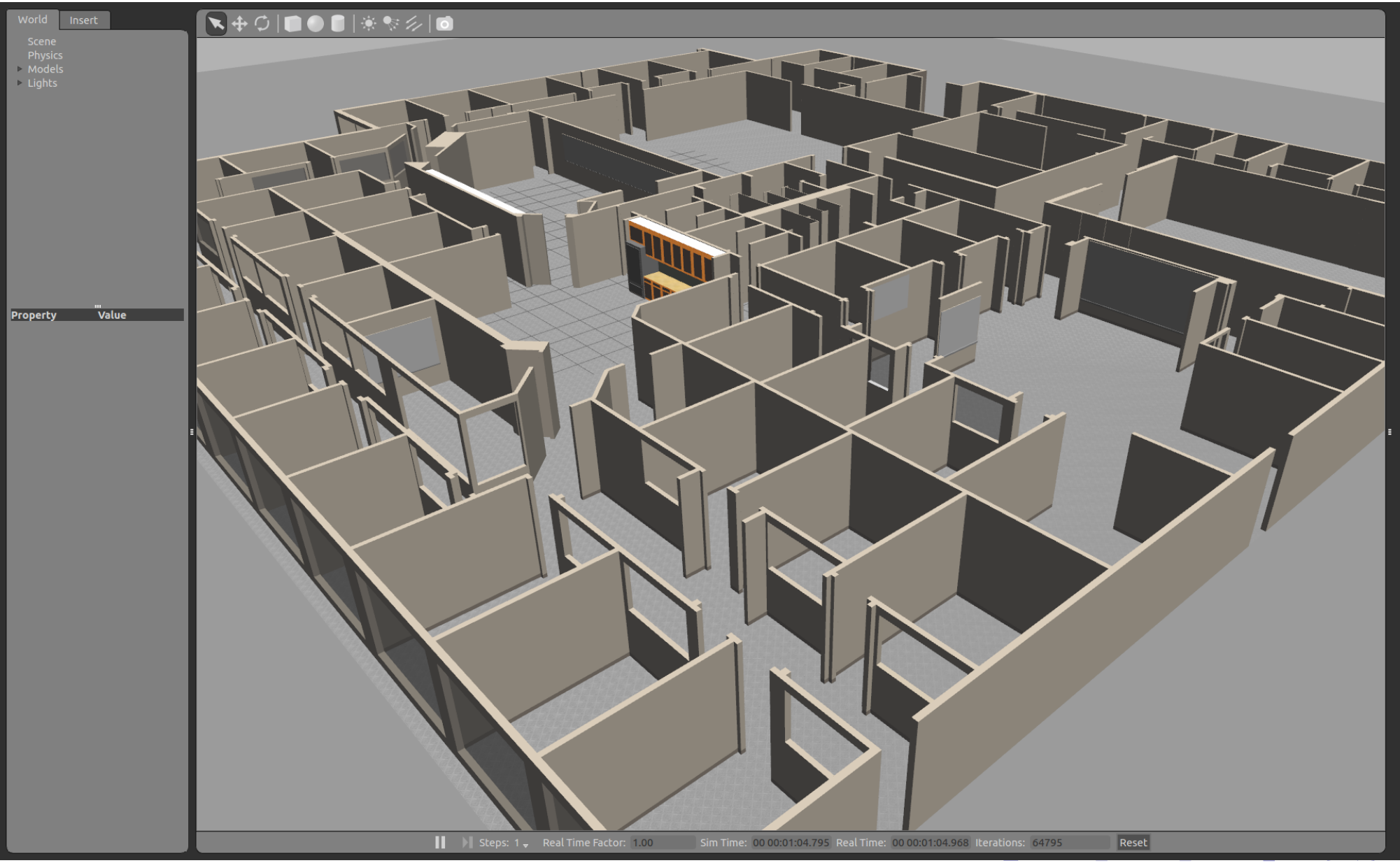
Launching Gazebo with willowgarage map

```
1 $ roslaunch gazebo_ros willowgarage_world.launch
```

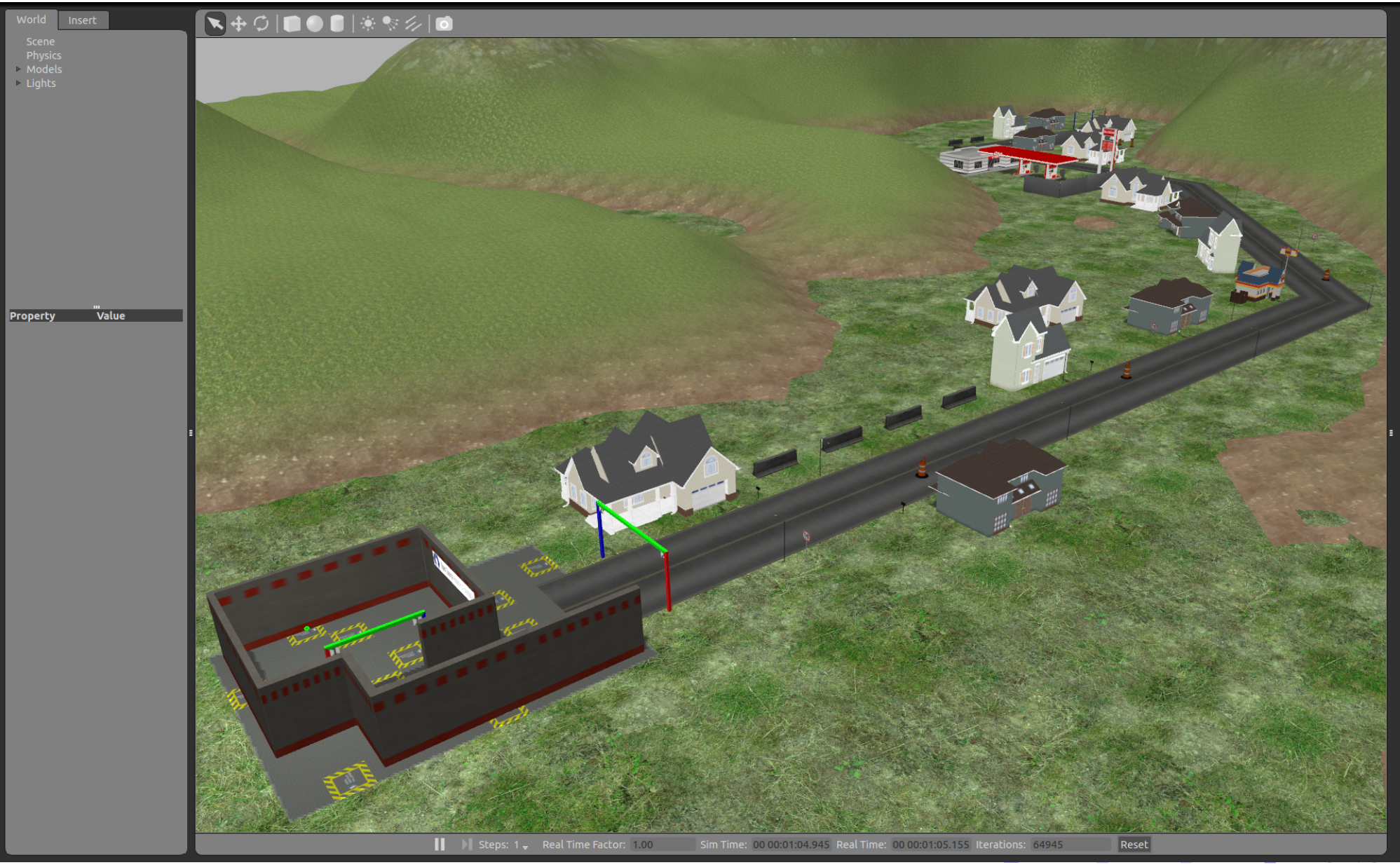
Gazebo with an empty world



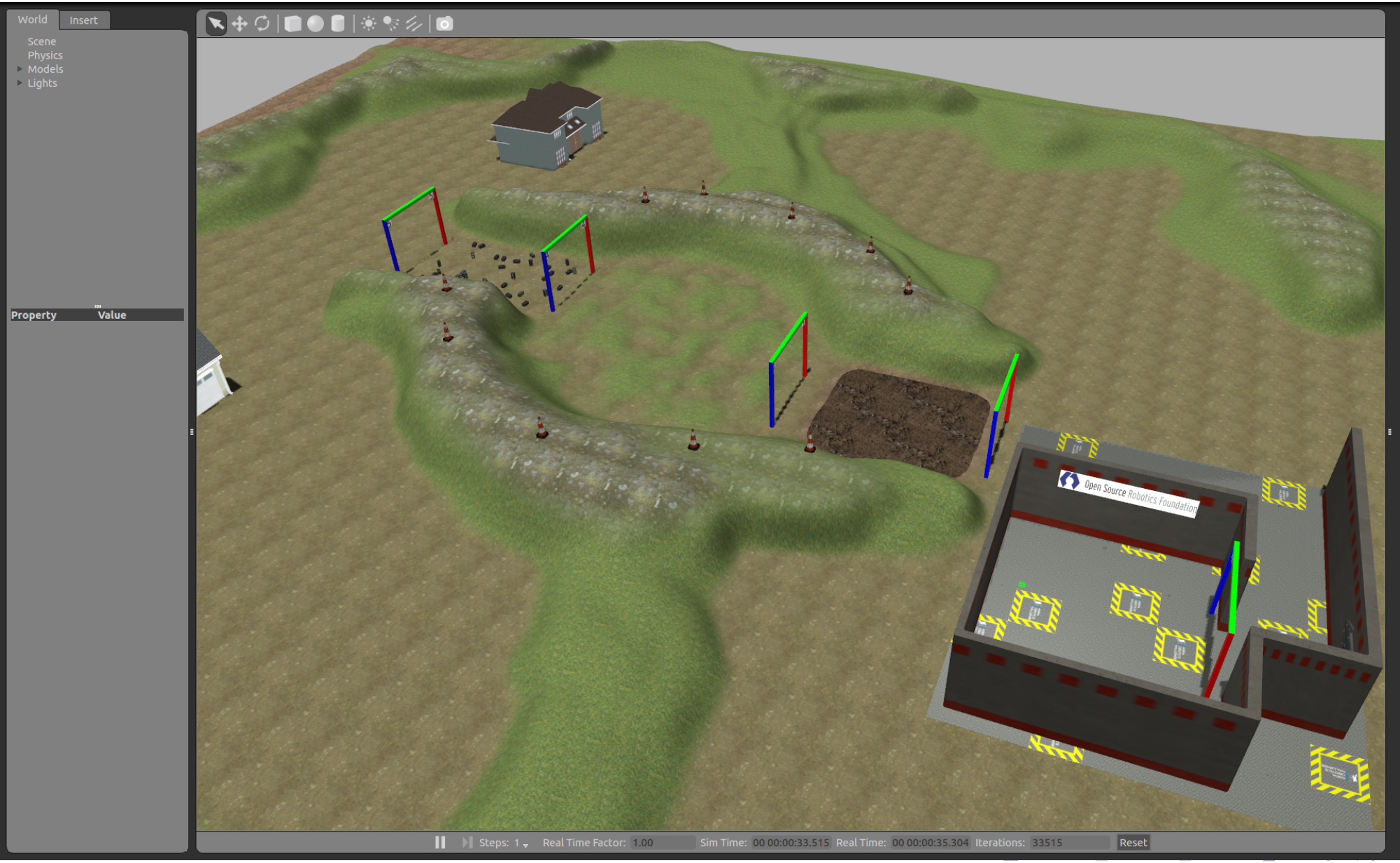
Gazebo with willowgarage map



Gazebo with VRC Final Task 1



Gazebo with VRC Final Task 2



Before we proceed...

Fetch some packages from fei-ros-pkg

```
$ cd ~/catkin_ws/src

$ git clone https://github.com/fei-ros-pkg/fei_gazebo.git
$ git clone https://github.com/fei-ros-pkg/husky_fei_robot.git
$ git clone https://github.com/fei-ros-pkg/husky_fei_simulator.git
$ git clone https://github.com/fei-ros-pkg/husky_fei_navigation.git

$ cd ../
$ catkin_make
```

Quick recap: roslaunch

Firstly, execute the following commands in separate terminals

```
$ roscore

$ rosrun gazebo_ros gzserver `rospack find gazebo_ros`/worlds/
    willowgarage.world _respawn:=false _output:=screen

$ rosrun gazebo_ros gzclient _respawn:=false _output:=screen
```

From a practical perspective...

- Several terminals
- Several commands and arguments to be typed in
- Tedious and slow task, more prone to errors

Quick recap: roslaunch

Definition

- Tool for easily launching multiple ROS nodes (locally and remotely)
- . . . and for setting parameters on the Parameter Server

Quick recap: roslaunch

Definition

- Tool for easily launching multiple ROS nodes (locally and remotely)
- . . . and for setting parameters on the Parameter Server
- Includes options to automatically respawn processes that have died

Quick recap: roslaunch

Definition

- Tool for easily launching multiple ROS nodes (locally and remotely)
- . . . and for setting parameters on the Parameter Server
- Includes options to automatically respawn processes that have died
- Takes in one or more XML configuration files, specifying:
  - nodes to launch
  - parameters to set
  - which machines such nodes should run on

Quick recap: roslaunch

Definition

- Tool for easily launching multiple ROS nodes (locally and remotely)
- . . . and for setting parameters on the Parameter Server
- Includes options to automatically respawn processes that have died
- Takes in one or more XML configuration files, specifying:
  - nodes to launch
  - parameters to set
  - which machines such nodes should run on
- *roscore* is, in fact, a specialisation of roslaunch tool to bring up the "core" of ROS system (that is, the node Master)

Quick recap: ROS launch files

roslaunch – Usage

```
$ roslaunch [options] [package] <filename> [arg_name:=value ...]
```

Example .launch XML config file

```
<launch>
  <node name="talker" pkg="day2_talker" type="talker_node" />
</launch>
```

Our first launch file: gazebo_wg.launch

```
<?xml version="1.0" ?>
<launch>
  <!-- start gazebo server with willowgarage world -->
  <node name="gazebo" pkg="gazebo_ros" type="gzserver" args="worlds/willowgarage.world"
        respawn="false" output="screen"/>

  <!-- start gazebo client -->
  <node name="gazebo_gui" pkg="gazebo_ros" type="gzclient" respawn="false" output="screen"/>
</launch>
```

Quick recap: ROS launch files

Load Gazebo with willowgarage world

```
1 $ roslaunch gazebo_wg.launch
```

Quick recap: ROS launch files

Load Gazebo with willowgarage world

```
$ roslaunch gazebo_wg.launch
```

Spawn the Husky-FEI into willowgarage world

```
$ roslaunch husky_fei_gazebo spawn_husky_fei.urdf.gazebo.launch
```

Quick recap: ROS launch files

Load Gazebo with willowgarage world

```
$ roslaunch gazebo_wg.launch
```

Spawn the Husky-FEI into willowgarage world

```
$ roslaunch husky_fei_gazebo spawn_husky_fei.urdf.gazebo.launch
```

Teleoperate the Husky-FEI robot

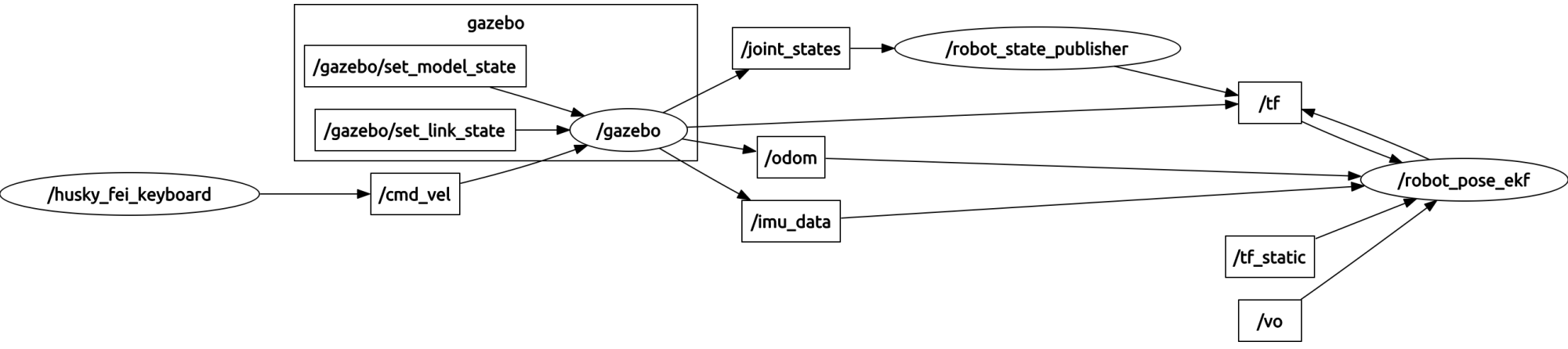
```
$ roslaunch husky_fei_teleop teleop_keyboard.launch
```

Quick recap: ROS launch files

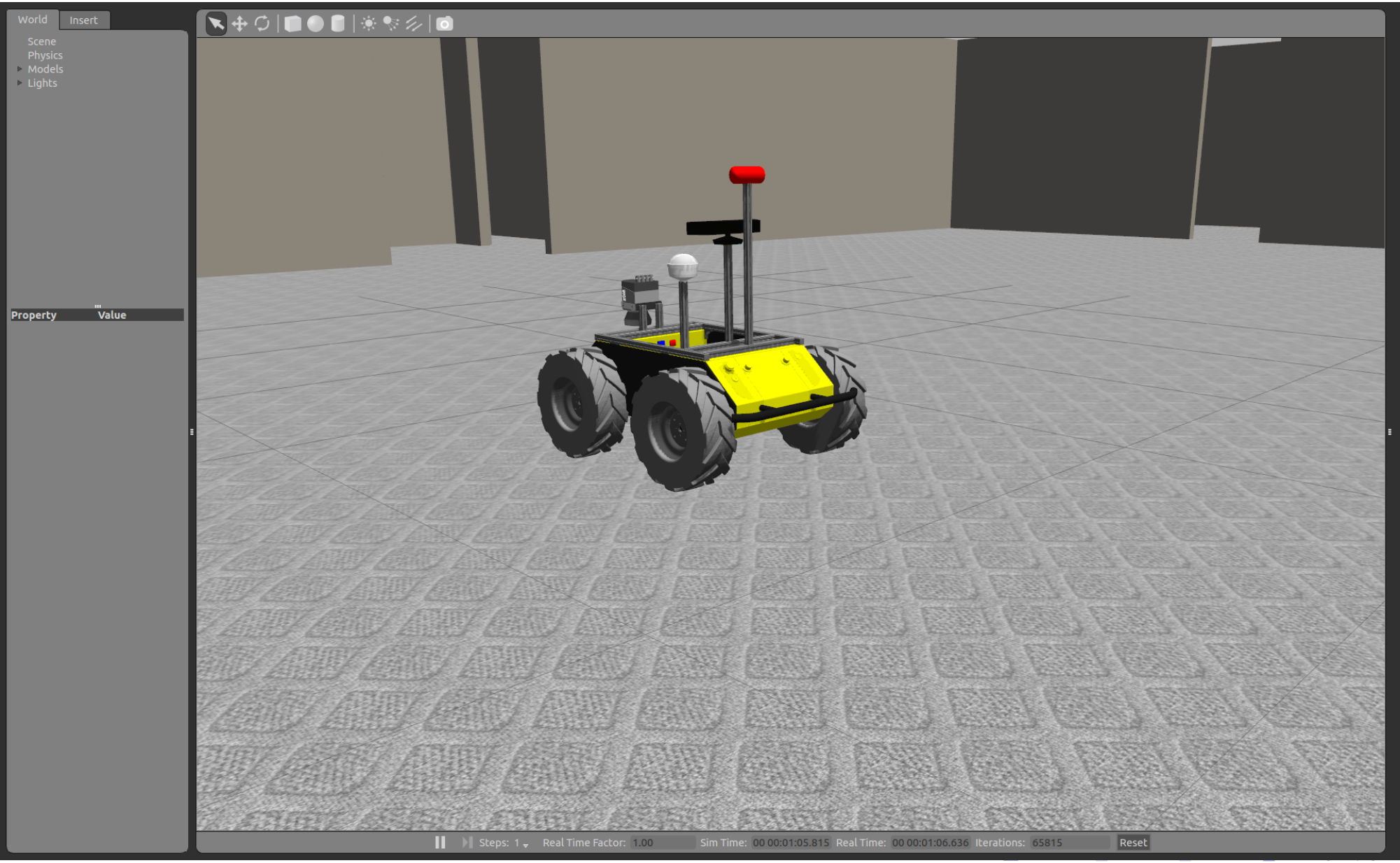
Automatically spawn a Husky-FEI into our willowgarage world

```
1 <?xml version="1.0" ?>
2 <launch>
3   <!-- start gazebo server with willowgarage world -->
4   <node name="gazebo" pkg="gazebo_ros" type="gzserver" args="worlds/willowgarage.world"
5         respawn="false" output="screen"/>
6
7   <!-- spawn Husky-FEI -->
8   <include file="$(find husky_fei_gazebo)/launch/spawn_husky_fei.urdf.gazebo.launch">
9    <arg name="pos_x" value="-5.0"/>
10   <arg name="pos_y" value="5.0"/>
11   <arg name="pos_z" value="0.0"/>
12   <arg name="rot_r" value="0.0"/>
13   <arg name="rot_p" value="0.0"/>
14   <arg name="rot_y" value="3.14159265359"/>
15  </include>
16
17  <!-- start gazebo client -->
18  <node name="gazebo_gui" pkg="gazebo_ros" type="gzclient" respawn="false" output="screen"/>
19 </launch>
```

ROS Computation Graph



Husky-FEI in Gazebo with willowgarage map



RViz: 3D visualisation for robots using ROS

Open two terminals and load...

- Gazebo with the willowgarage map (from husky_fei_gazebo)
- Husky-FEI teleoperation node

RViz: 3D visualisation for robots using ROS

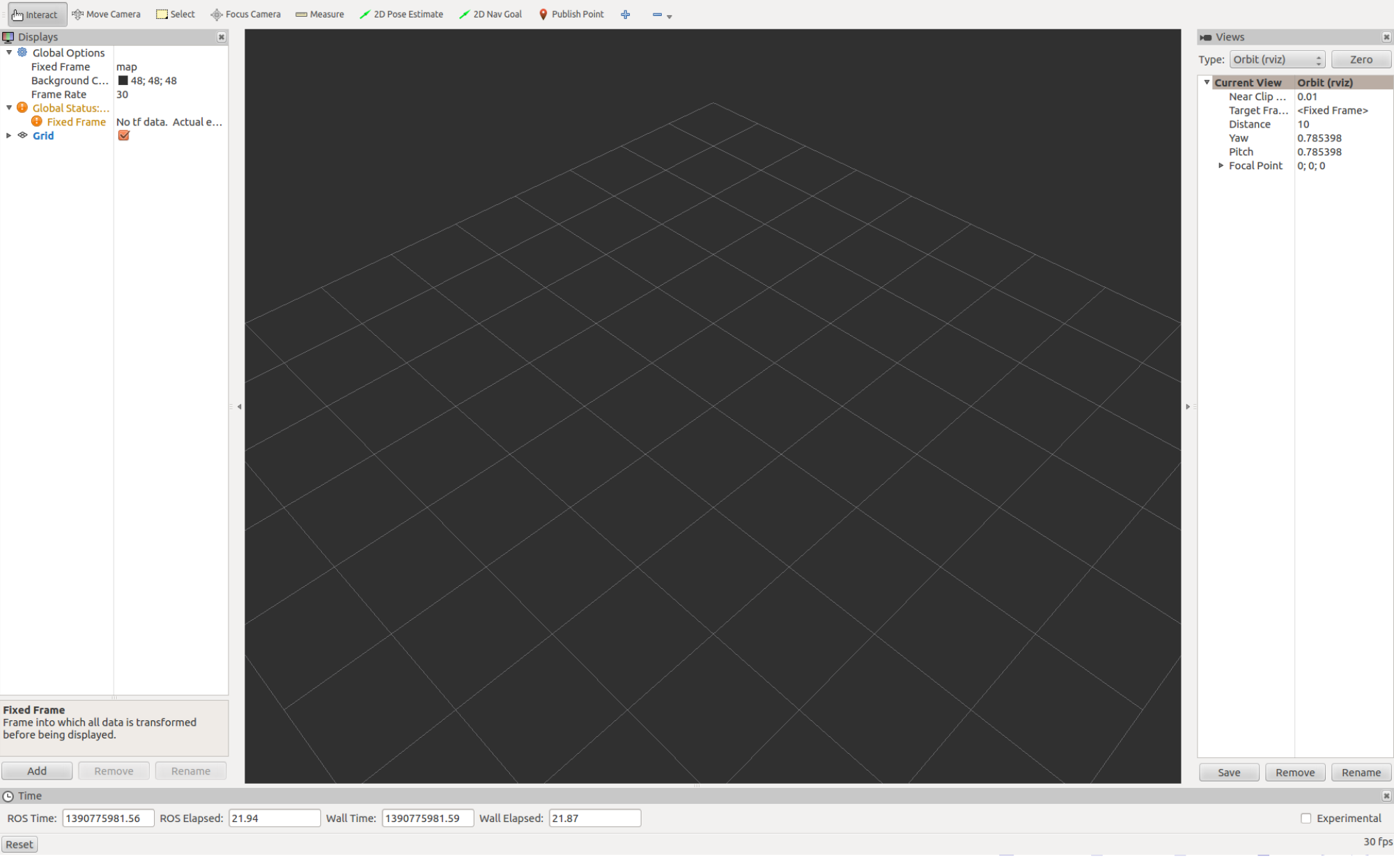
Open two terminals and load...

- Gazebo with the willowgarage map (from husky_fei_gazebo)
- Husky-FEI teleoperation node

Load RViz

```
1    $ rosrun rviz rviz
```

RViz GUI – empty



RViz GUI

Set the correct fixed frame

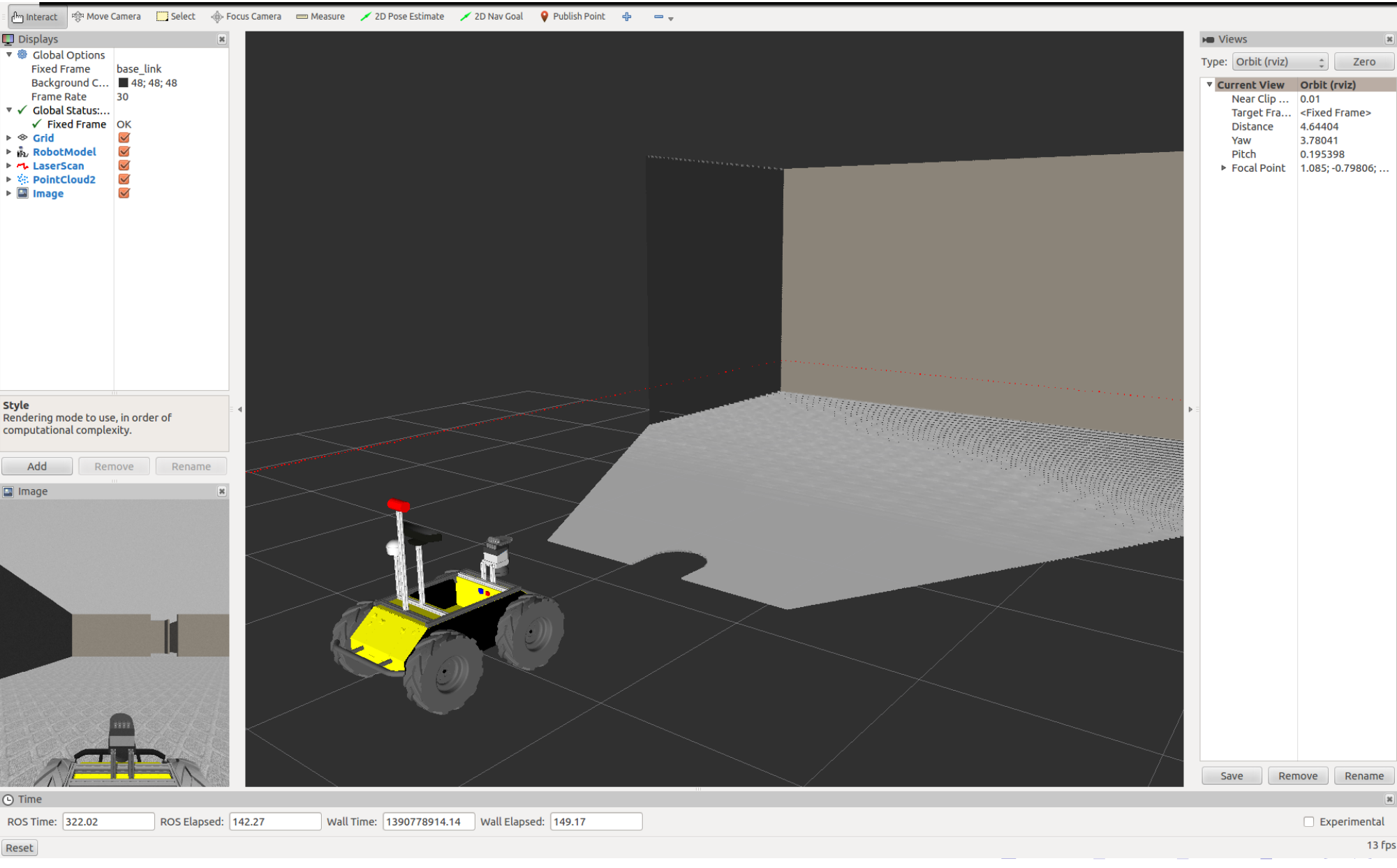
In **Global Options** → **Fixed Frame**: set to /base_link

Some Displays to add...

- Robot Model
- LaserScan (/laser_data)
- PointCloud2 (/camera/depth/points)
- Image (/stereo_camera/left/image_raw)

Now, play with rviz and teleoperate the Husky-FEI around

RViz GUI – configured



rosbag: recording and playing back data

rosbag is. . .

- a set of tools for recording from and playing back to ROS topics
- intended for high performance avoiding (de)serialisation of messages

```
murilo@muhrix: ~
murilo@muhrix:~$ rosbag -h
Usage: rosbag <subcommand> [options] [args]

A bag is a file format in ROS for storing ROS message data. The rosbag command can record, replay and manipulate bags.

Available subcommands:
   check        Determine whether a bag is playable in the current system, or if it can be migrated.
   compress     Compress one or more bag files.
   decompress   Decompress one or more bag files.
   filter       Filter the contents of the bag.
   fix          Repair the messages in a bag file so that it can be played in the current system.
   help
   info         Summarize the contents of one or more bag files.
   play         Play back the contents of one or more bag files in a time-synchronized fashion.
   record       Record a bag file with the contents of specified topics.
   reindex      Reindexes one or more bag files.

For additional information, see http://wiki.ros.org/rosbag

murilo@muhrix:~$
```

Recording data to build a map

Create a folder for the bags

```
1 $ mkdir ~/catkin_ws/bagfiles
2 $ cd ~/catkin_ws/bagfiles
```

Recording data to build a map

Create a folder for the bags

```
1 $ mkdir ~/catkin_ws/bagfiles
2 $ cd ~/catkin_ws/bagfiles
```

Open two terminals and load...

- Gazebo with the willowgarage map (from husky_fei_gazebo)
- Husky-FEI teleoperation node

Recording data to build a map

Create a folder for the bags

```
$ mkdir ~/catkin_ws/bagfiles
$ cd ~/catkin_ws/bagfiles
```

Open two terminals and load...

- Gazebo with the willowgarage map (from husky_fei_gazebo)
- Husky-FEI teleoperation node

Using rosbag to record the necessary data from the Husky-FEI

```
$ rosbag record -O mylaserdata /laser_data /tf
```

Now drive the robot around, explore the environment...

Stop recording. . . and playing back data

After navigating around the environment. . .

Stop recording data
(press **Ctrl-C** on the terminal which rosbag is running)

Stop recording. . . and playing back data

After navigating around the environment. . .

Stop recording data
(press **Ctrl-C** on the terminal which rosbag is running)

Examine the bag file

```
1    $ rosbag info <your bag file>
```

Stop recording. . . and playing back data

After navigating around the environment. . .

Stop recording data
(press **Ctrl-C** on the terminal which rosbag is running)

Examine the bag file

```
$ rosbag info <your bag file>
```

Play back recorded data

Kill all nodes, then execute in separate terminals:

```
$ roscore
$ rosbag play <your bag file>
$ rosrun rqt_graph rqt_graph
```

Simultaneous Localisation and Mapping (SLAM)

gmapping package

- ROS wrapper of gmapping, from OpenSlam[a]

[a] http://openslam.org

Simultaneous Localisation and Mapping (SLAM)

gmapping package

- ROS wrapper of gmapping, from OpenSlam[a]
- Provides laser-based SLAM as a ROS node named *slam_gmapping*

[a] http://openslam.org

Simultaneous Localisation and Mapping (SLAM)

gmapping package

- ROS wrapper of gmapping, from OpenSlam[a]
- Provides laser-based SLAM as a ROS node named *slam_gmapping*
- Generates 2D occupancy grids (like a building floorplan) from laser and pose data collected by a mobile robot

[a] http://openslam.org

Simultaneous Localisation and Mapping (SLAM)

gmapping package

- ROS wrapper of gmapping, from OpenSlam[a]
- Provides laser-based SLAM as a ROS node named *slam_gmapping*
- Generates 2D occupancy grids (like a building floorplan) from laser and pose data collected by a mobile robot
- Subscribes to topics *tf* and *scan*

[a] http://openslam.org

Simultaneous Localisation and Mapping (SLAM)

gmapping package

- ROS wrapper of gmapping, from OpenSlam[a]
- Provides laser-based SLAM as a ROS node named *slam_gmapping*
- Generates 2D occupancy grids (like a building floorplan) from laser and pose data collected by a mobile robot
- Subscribes to topics *tf* and *scan*
- Publishes topics *map_metadata*, *map* and *~entropy*

[a] http://openslam.org

Simultaneous Localisation and Mapping (SLAM)

gmapping package

- ROS wrapper of gmapping, from OpenSlam[a]
- Provides laser-based SLAM as a ROS node named *slam_gmapping*
- Generates 2D occupancy grids (like a building floorplan) from laser and pose data collected by a mobile robot
- Subscribes to topics *tf* and *scan*
- Publishes topics *map_metadata*, *map* and *~entropy*
- Provides service *dynamic_map*

[a] http://openslam.org

Building a map using gmapping

Playing back your bag file

Make sure ROS is not running! Then, in separate terminals...

```
1 roscore
2 rosrun gmapping slam_gmapping scan:=laser_data _odom_frame:=
      odom
3 rosbag play <your bag file>
```

Building a map using gmapping

Playing back your bag file

Make sure ROS is not running! Then, in separate terminals...

```
1 roscore
2 rosrun gmapping slam_gmapping scan:=laser_data _odom_frame:=
      odom
3 rosbag play <your bag file>
```

Save the map to map.pgm image file

Once *rosbag* has finished and exited, save the generated map:

```
1 rosrun map_server map_saver
```

Now, open the image file and see the result of your exploration!

Map Server package

map_saver

- Saves a map to disk (e.g., from a SLAM mapping service)
- Generates two files:
  - map.pgm: an ordinary image
  - map.yaml: metadata wrapping image file

Map Server package

map_saver

- Saves a map to disk (e.g., from a SLAM mapping service)
- Generates two files:
  - map.pgm: an ordinary image
  - map.yaml: metadata wrapping image file

YAML format: a simple, yet complete, example

```
1 image: map.pgm
2 resolution: 0.1
3 origin: [0.0, 0.0, 0.0]
4 occupied_thresh: 0.65
5 free_thresh: 0.196
6 negate: 0
```

More details about map_saver can be found at:
http://ros.org/wiki/map_server

Map Server package

map_server

- ROS node that reads a map from disk and offers it via a ROS service
- Simply converts colour values from (grey scale) image into ternary occupancy values:
  - free (0)
  - occupied (100)
  - unknown (-1)
- Publishes topics map_metadata and map
- Provides service static_map
- Usage:

```
rosrun map_server map_server mymap.yaml _respawn:=true
```

Adaptive Monte Carlo Localisation (AMCL)

AMCL

- Probabilistic localisation system for a robot moving in 2D

Adaptive Monte Carlo Localisation (AMCL)

AMCL

- Probabilistic localisation system for a robot moving in 2D
- Implements adaptive (or KLD-sampling) Monte Carlo localisation

Adaptive Monte Carlo Localisation (AMCL)

AMCL

- Probabilistic localisation system for a robot moving in 2D
- Implements adaptive (or KLD-sampling) Monte Carlo localisation
- It is a particle filter to track pose of a robot against a known map

Adaptive Monte Carlo Localisation (AMCL)

AMCL

- Probabilistic localisation system for a robot moving in 2D
- Implements adaptive (or KLD-sampling) Monte Carlo localisation
- It is a particle filter to track pose of a robot against a known map
- Subscribes to topics scan, tf, initial_pose and map

Adaptive Monte Carlo Localisation (AMCL)

AMCL

- Probabilistic localisation system for a robot moving in 2D
- Implements adaptive (or KLD-sampling) Monte Carlo localisation
- It is a particle filter to track pose of a robot against a known map
- Subscribes to topics scan, tf, initial_pose and map
- Publishes topics amcl_pose, particlecloud and tf

Adaptive Monte Carlo Localisation (AMCL)

AMCL

- Probabilistic localisation system for a robot moving in 2D
- Implements adaptive (or KLD-sampling) Monte Carlo localisation
- It is a particle filter to track pose of a robot against a known map
- Subscribes to topics scan, tf, initial_pose and map
- Publishes topics amcl_pose, particlecloud and tf
- Provides service global_localization

Adaptive Monte Carlo Localisation (AMCL)

AMCL

- Probabilistic localisation system for a robot moving in 2D
- Implements adaptive (or KLD-sampling) Monte Carlo localisation
- It is a particle filter to track pose of a robot against a known map
- Subscribes to topics scan, tf, initial_pose and map
- Publishes topics amcl_pose, particlecloud and tf
- Provides service global_localization
- Calls service static_map

Adaptive Monte Carlo Localisation (AMCL)

AMCL

- Probabilistic localisation system for a robot moving in 2D
- Implements adaptive (or KLD-sampling) Monte Carlo localisation
- It is a particle filter to track pose of a robot against a known map
- Subscribes to topics scan, tf, initial_pose and map
- Publishes topics amcl_pose, particlecloud and tf
- Provides service global_localization
- Calls service static_map
- Has a very long list of parameters to fine tune...

Adaptive Monte Carlo Localisation (AMCL)

Running AMCL node

- Notice that AMCL subscries to scan
- However, Husky-FEI publishes the base laser scans on laser_data
- Therefore, we need to remap the arguments!

```
1   $ rosrun amcl amcl scan:=laser_data
```

Adaptive Monte Carlo Localisation (AMCL)

Running AMCL node

- Notice that AMCL subscries to scan
- However, Husky-FEI publishes the base laser scans on laser_data
- Therefore, we need to remap the arguments!

```
1   $ rosrun amcl amcl scan:=laser_data
```

Use the launch files from husky_fei_navigation package

- There are way too many parameters to bother typing
- The package husky_fei_navigation contains launch files prepared for this exercise
- Examine launch file named **amcl_demo.launch**
- Examine launch file named **amcl.launch**

Playing with AMCL and RViz using *husky_fei_navigation*

In separate terminals, run

```
1  roslaunch husky_fei_gazebo husky_fei_wg.launch
2  roslaunch husky_fei_navigation amcl_demo.launch
3  rosrun rviz rviz
```

Playing with AMCL and RViz using *husky_fei_navigation*

In separate terminals, run

```
roslaunch husky_fei_gazebo husky_fei_wg.launch
roslaunch husky_fei_navigation amcl_demo.launch
rosrun rviz rviz
```

Add the following Displays in RViz

- RobotModel
- LaserScan
- Map
- PoseArray

Playing with AMCL and RViz using *husky_fei_navigation*

Set up robot current pose estimate

- Select 2D Pose Estimate button (top of window)
- Within the map, left click and hold to set position...
- then move mouse to set direction and release button
- A particle cloud (red arrows) will appear

Playing with AMCL and RViz using *husky_fei_navigation*

Set up robot current pose estimate

- Select 2D Pose Estimate button (top of window)
- Within the map, left click and hold to set position...
- then move mouse to set direction and release button
- A particle cloud (red arrows) will appear

Teleoperate the Husky-FEI and observe the particle cloud

In case you forgot, here is the command:

```
1 roslaunch husky_fei_teleop teleop_keyboard.launch
```

2D Navigation and the Husky-FEI

Close all ROS nodes and start over, in separate terminals

```
1  roslaunch husky_fei_gazebo husky_fei_wg.launch
2  roslaunch husky_fei_navigation odom_navigation_demo.launch
3  roslaunch husky_fei_teleop teleop_keyboard.launch
```

move_base

- Implementation of an action that, given a goal in the world, will attempt to reach it with a mobile base
- Global and local planner to accomplish navigation[a]

[a] http://www.ros.org/wiki/move_base

That's all for today, folks!

Today we have learnt how to:

- Run Gazebo with the Husky-FEI

That's all for today, folks!

Today we have learnt how to:

- Run Gazebo with the Husky-FEI
- Create launch files and use roslaunch

That's all for today, folks!

Today we have learnt how to:

- Run Gazebo with the Husky-FEI
- Create launch files and use roslaunch
- Visualise data using RViz

That's all for today, folks!

Today we have learnt how to:

- Run Gazebo with the Husky-FEI
- Create launch files and use roslaunch
- Visualise data using RViz
- Record and play back data using rosbag

That's all for today, folks!

Today we have learnt how to:

- Run Gazebo with the Husky-FEI
- Create launch files and use roslaunch
- Visualise data using RViz
- Record and play back data using rosbag
- Map unknown environments using gmapping

That's all for today, folks!

Today we have learnt how to:

- Run Gazebo with the Husky-FEI
- Create launch files and use roslaunch
- Visualise data using RViz
- Record and play back data using rosbag
- Map unknown environments using gmapping
- Localise a mobile robot within a known map

That's all for today, folks!

Today we have learnt how to:

- Run Gazebo with the Husky-FEI
- Create launch files and use roslaunch
- Visualise data using RViz
- Record and play back data using rosbag
- Map unknown environments using gmapping
- Localise a mobile robot within a known map
- Use the Husky-FEI navigation package (mostly move_base)

Question time

Thank you very much!
Questions?