Técnicas de los Sistemas Inteligentes

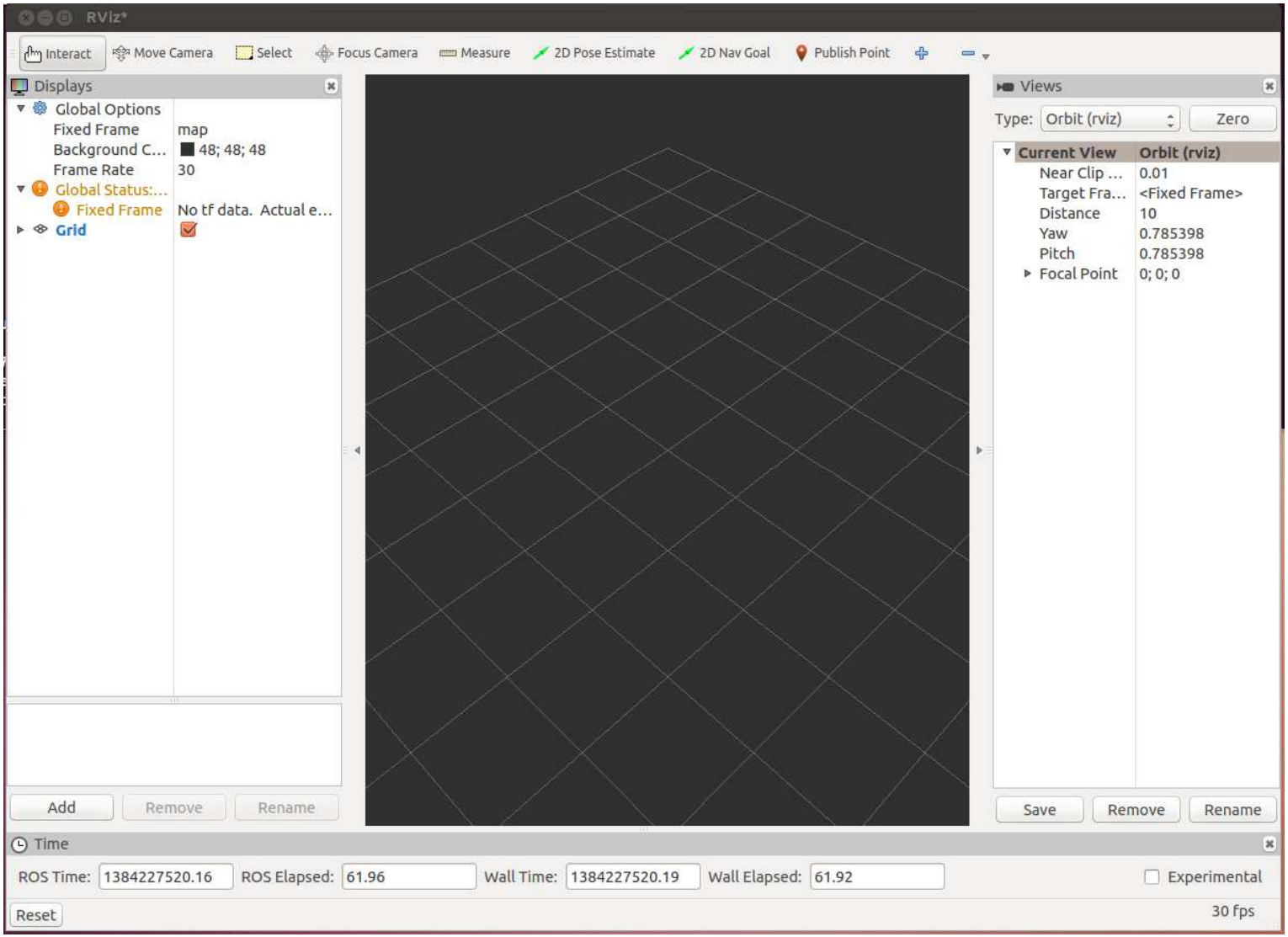
Práctica1: Robótica.
Sesion3. Visualización con **rviz**

RVIZ: visualización

- **rviz** es una herramienta de visualización 3D de ROS que nos permite ver el mundo desde la perspectiva del robot.
- Rviz user guide and tutorials

  http://wiki.ros.org/rviz
- Para ejecutar **rviz**

```
$ rosrun rviz rviz
```



- La primera vez se ve una vista 3D vacía
- A la izquierda hay un área de **Displays,** que contiene una lista de varios elementos en el mundo.
  - Ahora solo contiene opciones globales y la rejilla (grid).
  - Botón derecho o rueda para zoom in or out
  - Botón izquierdo para pan (shift-click) or rotate (click)

- Debajo de el área de Displays hay un botón **Add** button que permite añadir más elementos que visualizar
  - En generla asociados con los topics y/o mensajes que publican los nodos.

rviz Displays

Display name	Description	Messages Used
Axes	Displays a set of Axes	
Effort	Shows the effort being put into each revolute joint of a robot.	sensor_msgs/JointStates
Camera	Creates a new rendering window from the perspective of a camera, and overlays the image on top of it.	sensor_msgs/Image sensor_msgs/CameraInfo
Grid	Displays a 2D or 3D grid along a plane	
Grid Cells	Draws cells from a grid, usually obstacles from a costmap from the navigation stack.	nav_msgs/GridCells
Image	Creates a new rendering window with an Image.	sensor_msgs/Image
LaserScan	Shows data from a laser scan, with different options for rendering modes, accumulation, etc.	sensor_msgs/LaserScan
Map	Displays a map on the ground plane.	nav_msgs/OccupancyGrid

rviz Displays

Display name	Description	Messages Used
Markers	Allows programmers to display arbitrary primitive shapes through a topic	visualization_msgs/Marker visualization_msgs/MarkerArray
Path	Shows a path from the navigation stack.	nav_msgs/Path
Pose	Draws a pose as either an arrow or axes	geometry_msgs/PoseStamped
Point Cloud(2)	Shows data from a point cloud, with different options for rendering modes, accumulation, etc.	sensor_msgs/PointCloud sensor_msgs/PointCloud2
Odometry	Accumulates odometry poses from over time.	nav_msgs/Odometry
Range	Displays cones representing range measurements from sonar or IR range sensors.	sensor_msgs/Range
RobotModel	Shows a visual representation of a robot in the correct pose (as defined by the current TF transforms).	
TF	Displays the tf transform hierarchy.	

- Primero lanzar Gazebo (o Stage) con Turtlebot

```
$ roslaunch turtlebot_gazebo turtlebot_world.launch
```

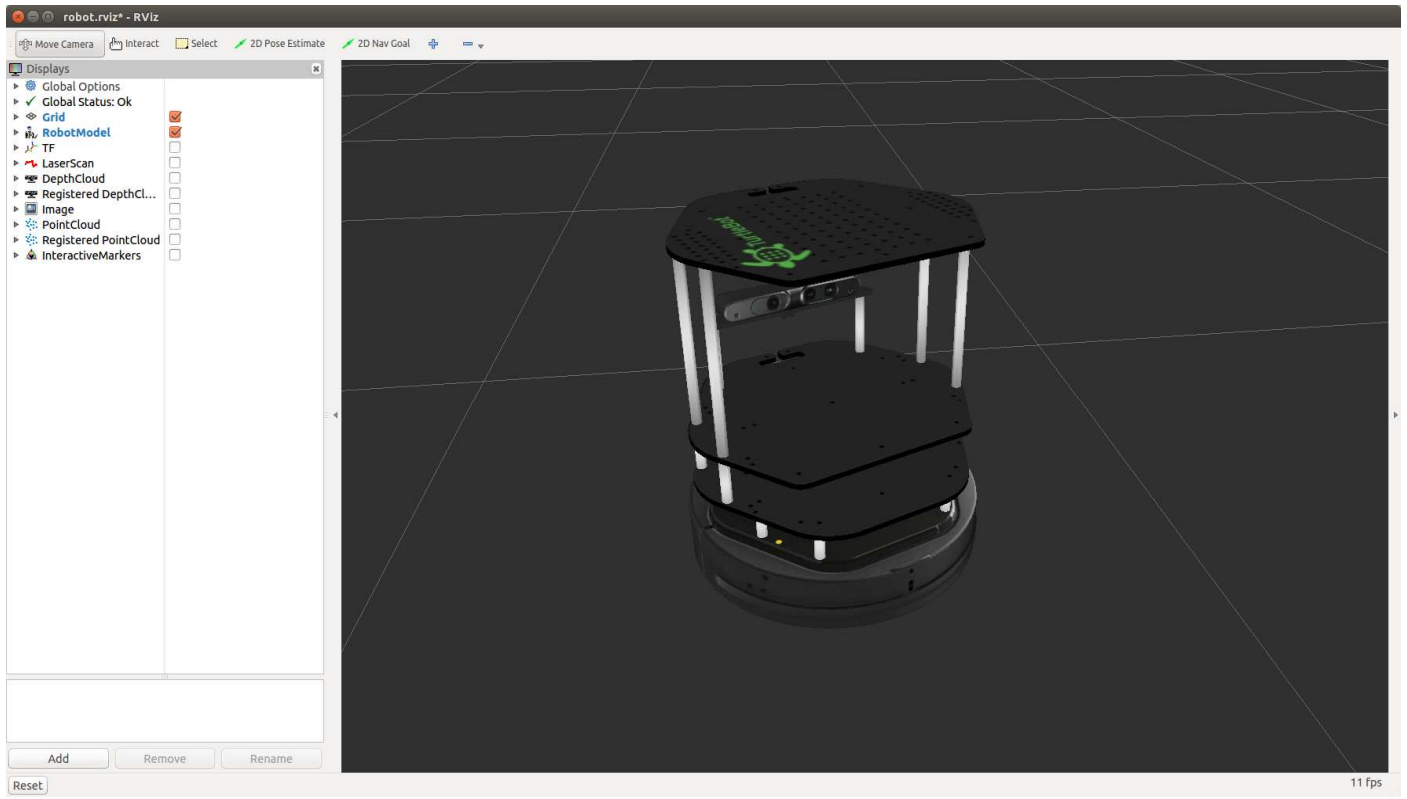
- Para Stage basta con lanzar

```
$ roslaunch turtlebot_stage turtlebot_in_stage.launch
```

rviz with TurtleBot

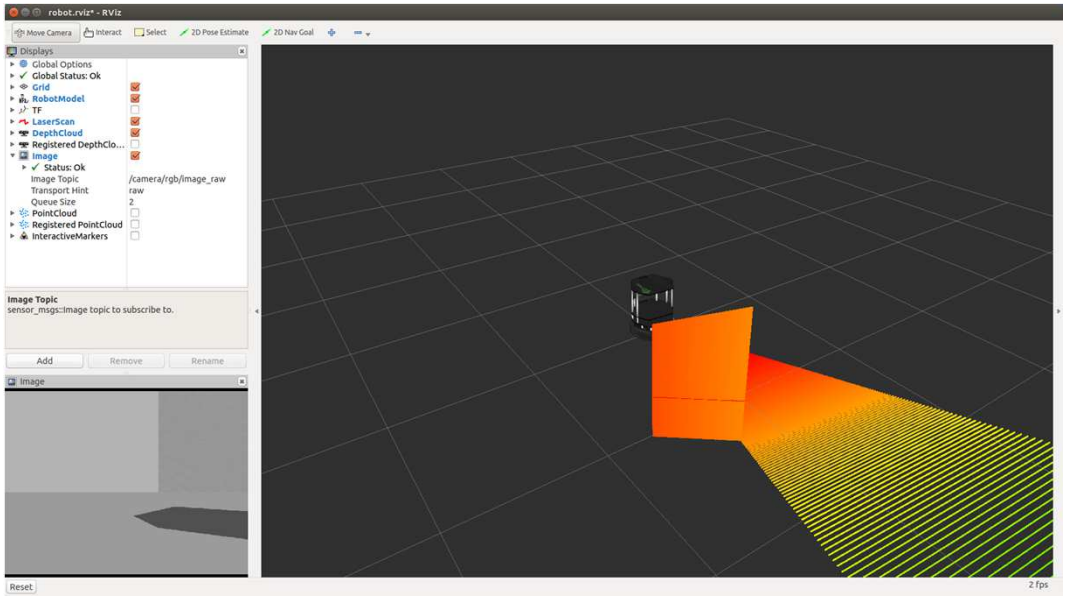
- You can start rviz already configured to visualize the robot and its sensor's output:

```
$ roslaunch turtlebot_rviz_launchers view_robot.launch
```



TurtleBot Image Display

- To visualize any display you want, just click on its check button



- Si no se observa imagen en el cuadro inferior derecho cambiar el topic del display “Image” (en Stage no hay posibilidad de ver imagen porque el modelo del robot no incluye una cámara).

Loading and Saving Configuration

- You can save your rviz settings by choosing File > Save Config from the menu
- Your settings will be saved to a .rviz file
- Then, you can start rviz with your saved configuration:

```
$ rosrun rviz rviz -d my_config.rviz
```

Launch File (Gazebo)

- Editar el fichero stopper.launch de la última sesión y guardarlo como “stopper-rviz.launch”.

```
<launch>
   <param name="/use_sim_time" value="true" />
   <!-- Launch turtle bot world -->
   <include file="$(find turtlebot_gazebo)/launch/turtlebot_world.launch"/>

   <!-- Launch stopper node -->
   <node name="stopper" pkg=“wander_bot" type="stopper" output="screen"/>

 <!-- Open rviz -->
  <include file="$(find turtlebot_rviz_launchers)/launch/view_robot.launch"/>
</launch>
```

- To run the launch file:

```
$ roslaunch wander_bot stopper-rviz.launch
```

- Si no funciona, ejecutar cada nodo por separado con rosrun.

Launch File (Stage)

- Editar el fichero stopper.launch de la última sesión y guardarlo como “stopper-rviz.launch”.

```
<launch>
   <param name="/use_sim_time" value="true" />
 <!- Launch Turtlebot in Stage>
 <include file = “$(find turtlebot_stage)/launch( turtlebot_in_stage.launch

 <!-- Launch stopper node -->
   <node name="stopper" pkg=“wander_bot" type="stopper" output="screen"/>

</launch>
```

Este launch lanza Stage, rviz y stopper node:

- To run the launch file:

```
$ roslaunch wander_bot stopper-rviz.launch
```

- Si no funciona, ejecutar cada nodo por separado con rosrun.

Si RVIZ no arranca a la primera...

- Desactivar aceleración hardware
  - Si vuestro sistema usa Mesa graphics drivers (e.g. para Intel GPUs, dentro de una VM), la aceleración hardware puede causar problemas.
  - Antes de ejecutar rviz hacer

```
$ export LIBGL_ALWAYS_SOFTWARE=1
$ rosrun rviz rviz
```

- Si persiste, usar opción -sync

```
$ export LIBGL_ALWAYS_SOFTWARE=1
$ rosrun rviz rviz -sync
```

- Si persiste, probar a borrar cualquier contenido de ~/.rviz:

```
$ rm -R  ~/.rviz/*
```