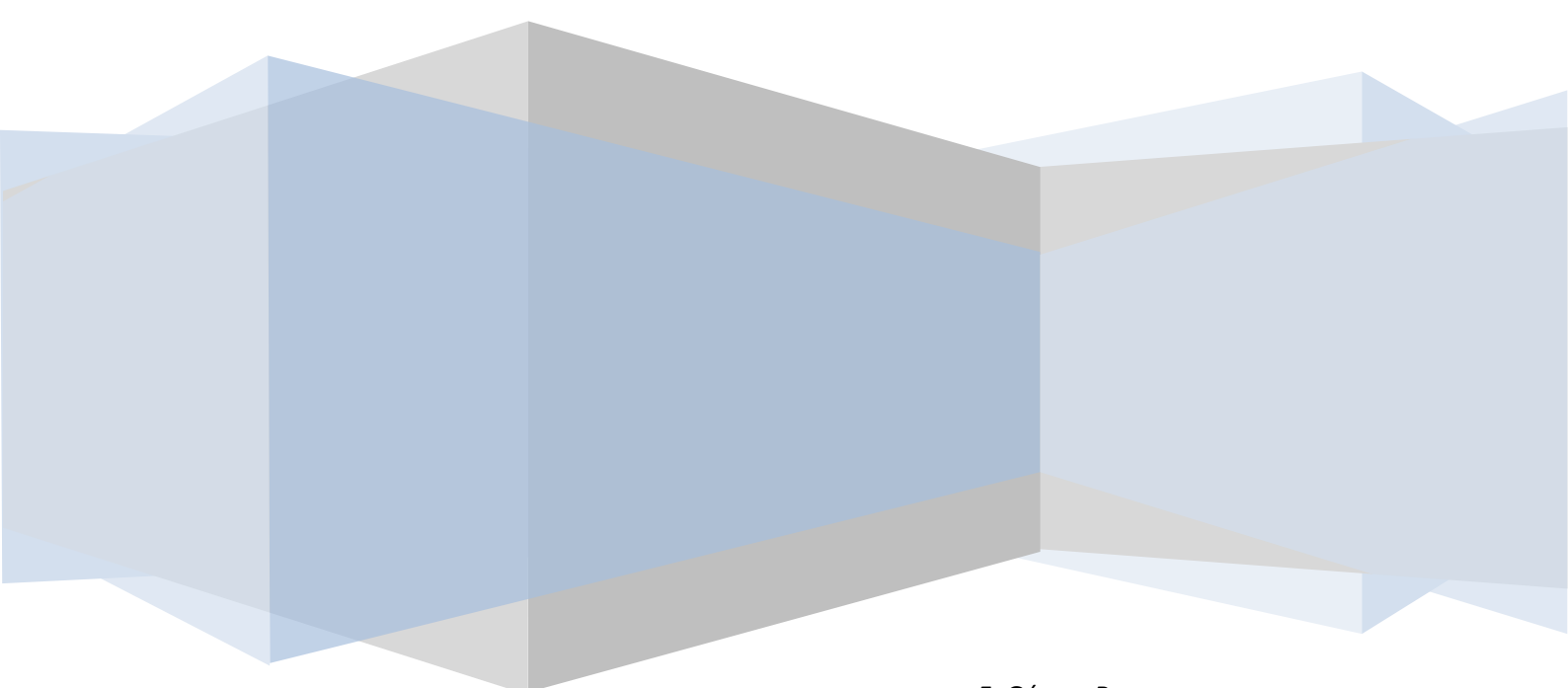


ROBÓTICA  
CUARTO CURSO DEL GRADO EN  
INGENIERÍA INFORMÁTICA



# Práctica 4

ROS: RVIZ y ROSBAG



F. Gómez Bravo  
R. López de Ahumada  
José Manuel Lozano

En esta práctica se describen los conceptos básicos del sistema de la herramienta *rviz*, así como el uso de *roscor* para depurar el funcionamiento de un robot

## 1. ¿Qué es RVIZ?

*Rviz* es un software de visualización en 3D (tres dimensiones) diseñado para el sistema operativo ROS. Este software debe ser ejecutado como un nodo más en el entorno de ROS aunque requiere de ciertas condiciones para que los elementos sean representados correctamente..

## 2. Primeros pasos con *rviz*

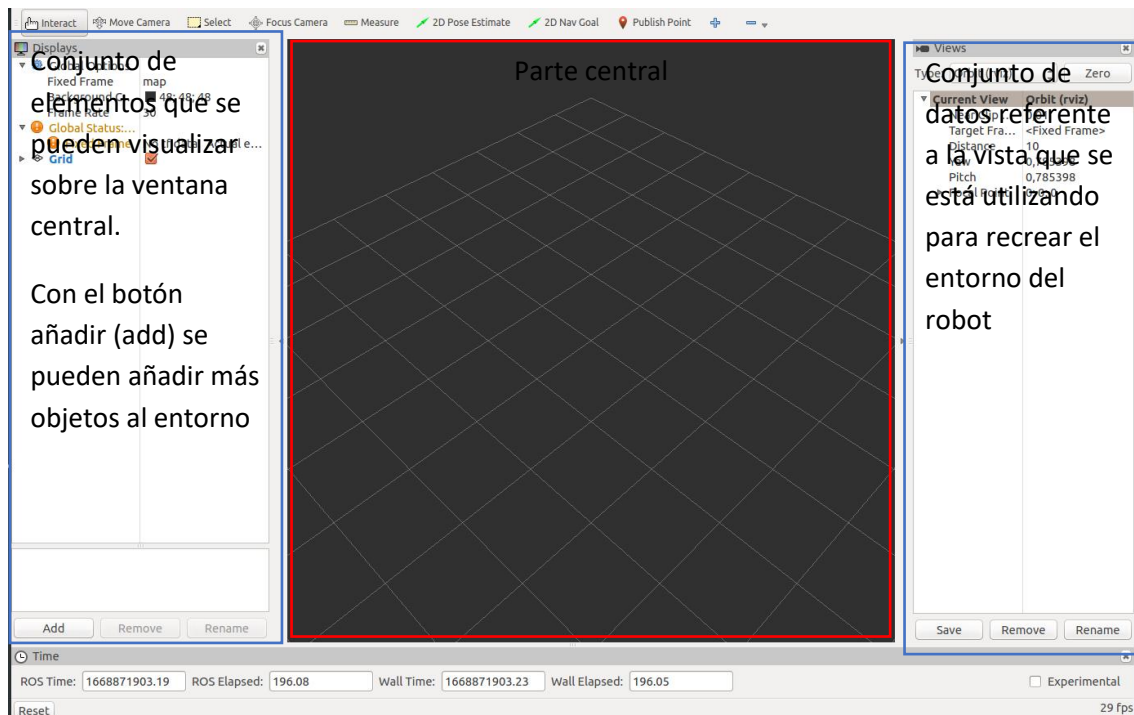
Como se ha comentado en el apartado anterior, *rviz* se ejecuta como un nodo más dentro del entorno de ROS, por ello en primer lugar debe estar lanzado el maestro de ROS. Para ello es necesario utilizar la instrucción *roscor* como se vio en la guía anterior.

```
turtlebot@rosindigo:~$ roscor
```

Una vez lanzado el maestro ya se podría lanzar *rviz*. Para lanzar *rviz* se debe hacer uso de la instrucción *roscor* del siguiente modo:

```
turtlebot@rosindigo:~$ roscor rviz rviz
```

Una vez carga el software *rviz*, se observa la vista básica del software. En el centro de la ventana se observa un fondo negro y una malla de cuadros en color gris. Esta zona es donde se encontrarán todos los elementos que compongan el mundo del robot. Dicho de otra forma, en ella se podrá observar el robot o los ejes que lo representan y los posibles obstáculos que se encuentren en el entorno del robot. En la parte izquierda de la ventana se observan un conjunto de elementos desplegables, mientras que en la derecha se observan las principales opciones de visualización del entorno. En la parte superior de la ventana se observan las opciones para poder mover la cámara, mover o rotar la matriz o malla y por tanto orientar el entorno, así como otras opciones como la opción de medir o indicar nuevos objetivos para el robot entre otras opciones. Por último, en la parte inferior se observa un registro temporal.



Resultado 1: lanzamiento de rviz

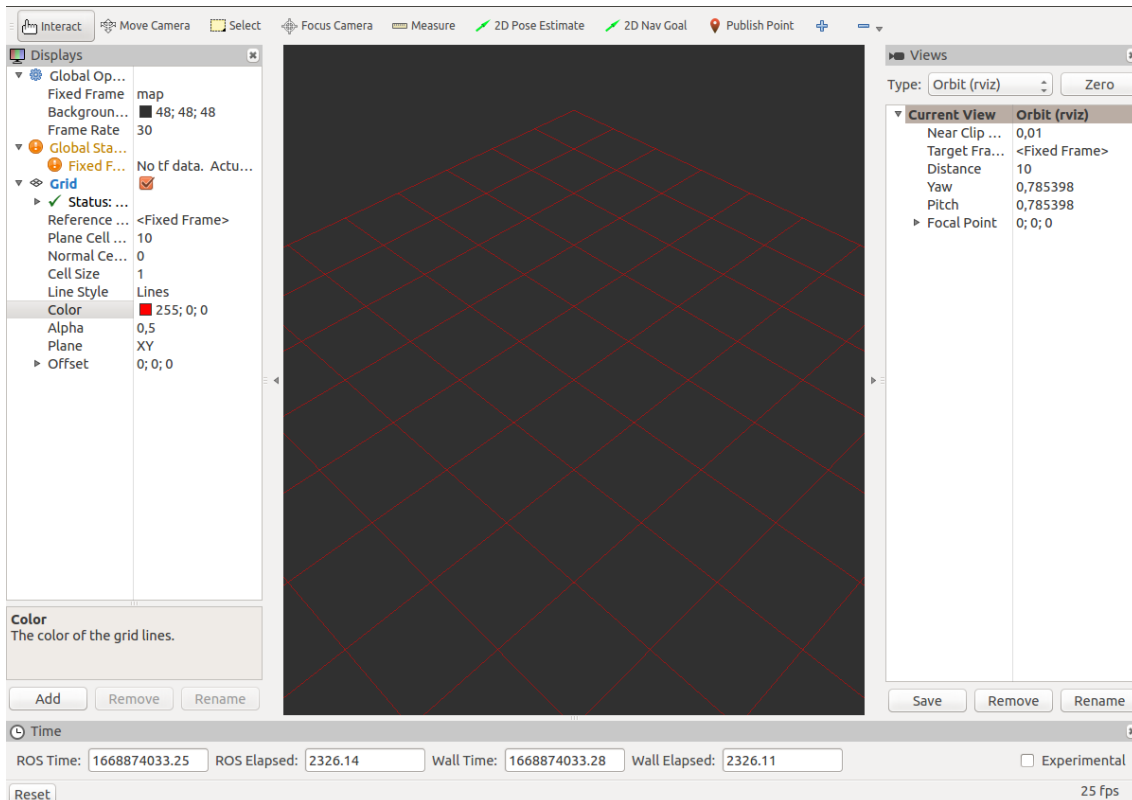
## 2.1. Elementos representables

Todos los elementos que se pueden representar se encuentran disponibles al pulsar el botón añadir (*add*) de la parte izquierda de la ventana. Entre ellos se pueden destacar los siguientes:

- **Rejilla o malla (*Grid*):** permite dibujar la rejilla para identificar más fácilmente los movimientos del robot
- **Ejes (*Axes*):** permite representar los ejes X, Y y Z, ya sean ejes locales o globales
- **Modelos de Robot (*RobotModel*):** permite cargar modelos de robot en función del robot que se esté utilizando
- **Escáner laser (*LaserScan*):** permite representar los datos detectados por un sensor laser en caso del que el robot utilizado lo incluya

Todos los elementos que se pueden representar tienen un conjunto de propiedades que se pueden modificar en función de las necesidades de cada uno de los entornos o características que sean necesarias. Un ejemplo de estas propiedades que pueden configurarse puede ser el color de la rejilla o malla que actualmente es gris y puede representarse en rojo si lo deseamos.

Antes de lanzar la instrucción *roscore*, se procederá a abrir una terminal y pulsar botón derecho y crear al menos una división de la terminal para poder trabajar con *roscore* y otra con las instrucciones que se expondrán más adelante. Al lanzar *roscore* en una de las dos terminales abiertas sobre la misma ventana ofrece en una de las terminales la siguiente salida:

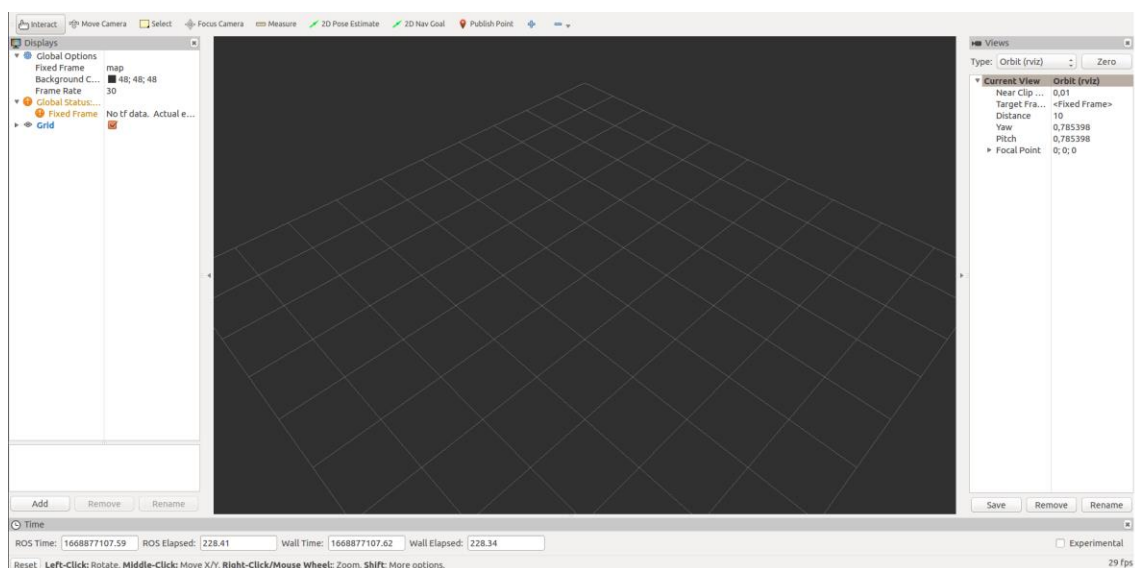


Para ello tan solo es necesario desplegar el menú del elemento representable *Grid* y modificar sus parámetros.

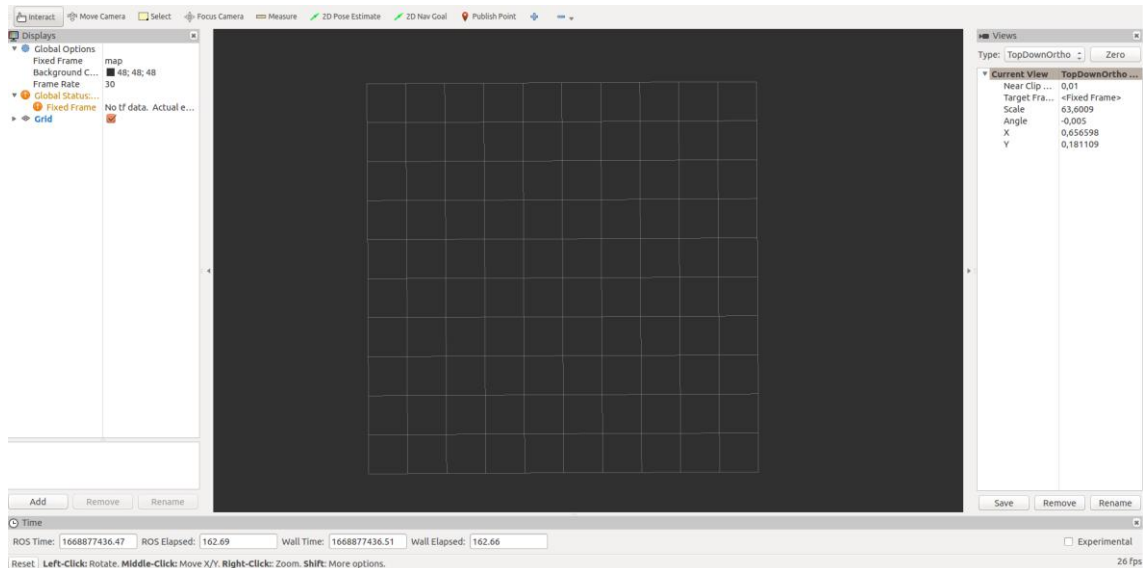
## 2.2. Cámaras

El programa cuenta también con un conjunto de cámaras que permite seleccionar la vista más adecuada en función de la operación a realizar con el robot que se esté trabajando. Las cámaras de las que se disponen son:

- Cámara orbital: es la cámara por defecto. Esta cámara rota sobre un punto focal y siempre se encuentra enfocando al mismo punto.



- Cámara primera persona o FPS (*first-person*): es una cámara en primera persona y su rotación permite observar lo que rodea al robot como si girara la cabeza. Es la cámara más parecida a la percepción del humano.
- Cámara Top-down Orthographic: esta cámara permite observar solo los ejes X e Y desde arriba es una vista en 2D. Esta cámara es muy útil cuando el robot solo se desplaza en el eje X e Y como por ejemplo *Turtlesim*.



En función de la cámara seleccionada, la parte inferior de la ventana indica como mover la cámara, rotarla o aumentar o disminuir el zoom.

### 2.3. Representación de Turtlesim en rviz

Los nodos de simulación de robot *Turtlesim* pueden ser representados en el software *rviz*. Para ello es necesario conocer que *rviz* requiere que la información se le envíe de una manera concreta. Esta información se envía a través de unos nodos denominados *broadcasters* que permiten realizar una superposición de espacios vectoriales o sistemas de coordenadas frente al sistema de coordenadas de un mundo y al sistema de coordenadas del propio robot. Dicho de otro modo, es necesario que todas las coordenadas de ubicación se establezcan frente a la posición 0,0,0 (x,y,z) del mundo virtual del entorno de *rviz*. Para ello es necesario hacer uso de un paquete denominado *tf*, el cual se mostrará en los siguientes pasos.

En primer lugar, *roscore* debe estar lanzado antes de comenzar a trabajar con ningún nodo de ROS. Luego se debe lanzar el nodo *turtlesim\_node* y su controlador por teclado *turtle\_teleop\_key* del siguiente modo (cada instrucción debe lanzarse en una consola o ventana de consola independiente):

```
turtlebot@rosindigo:~$ rosrund turtlesim turtlesim_node
```

```
turtlebot@rosindigo:~$ rosrund turtlesim turtle_teleop_key
```

Una vez lanzado tanto el simulador de robot móvil (*turtlesim*) como el nodo que lo controla por teclado, se puede observar que el robot simulado (tortuga) se encuentra en la posición

$x=5,544445$  e  $y=5,544445$ . Esto quiere decir que el centro la pantalla de *turtlesim* tiene esas coordenadas, las cuales serán necesarias para su posterior uso en *rviz*.

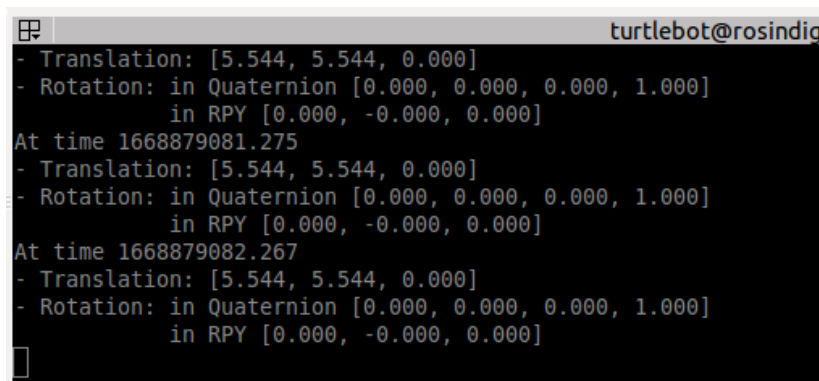
Para poder trabajar con *rviz*, el primer paso es la creación del nodo *broadcaster* encargado de traducir los movimientos de la tortuga en un sistema de coordenadas local a un sistema de coordenadas global. Para ello se hace uso del paquete *turtle\_tf*, el cual contiene un nodo *broadcaster* que usa como base la posición de la tortuga publicada en el topic */turtle1/pose*. Para que este nodo *broadcaster* funcione de manera correcta, es necesario indicarle el nombre de la tortuga deseada (en el presente caso como solo hay una tortuga el nombre es "turtle1"). En el caso de haber más de una tortuga sería necesario indicar el nombre de la tortuga que se quiera utilizar. Es importante recordar que este nodo debe ser lanzado en una nueva terminal.

```
turtlebot@rosindigo:~$ rosruncat turtle_tf broadcaster "turtle1"
```

La conversión de coordenadas puede observarse a través de otro nodo que las muestra por consola. Para ello, es necesario indicar qué eje de coordenadas es el que se utiliza como global y cuál como relativo. Todo ello se recoge en la siguiente instrucción, donde */world* es el eje global y */turtle1* es el eje relativo.

```
turtlebot@rosindigo:~$ rosruncat tf_echo /world /turtle1
```

En el resultado mostrado por pantalla se puede observar cómo se convierte la posición de la tortuga en una rotación y una posición frente al eje de coordenadas global.

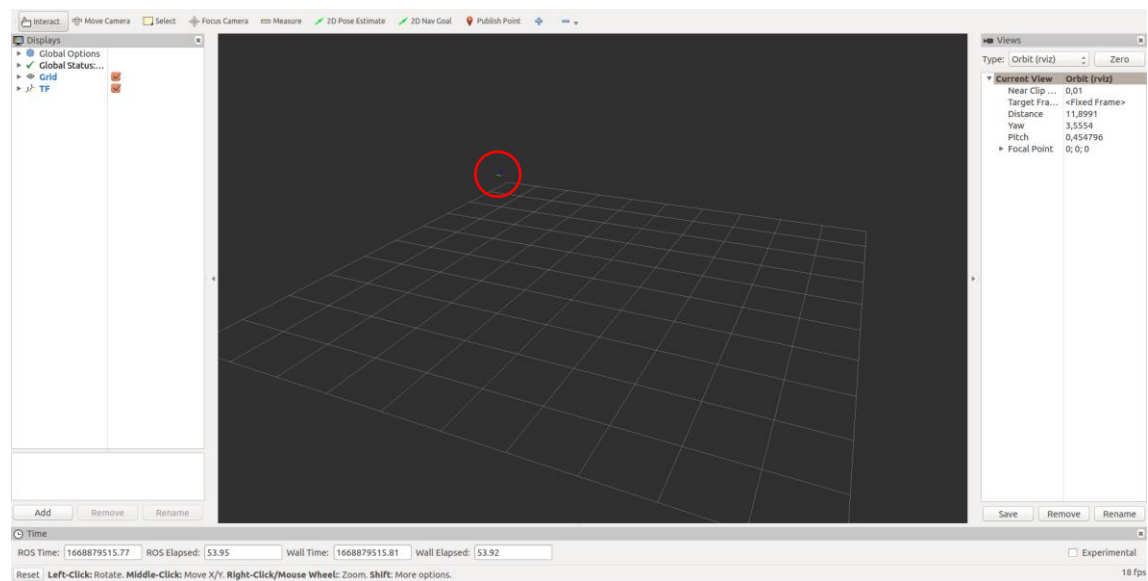


```
turtlebot@rosindigo:~$ rosruncat tf_echo /world /turtle1
- Translation: [5.544, 5.544, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
           in RPY [0.000, -0.000, 0.000]
At time 1668879081.275
- Translation: [5.544, 5.544, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
           in RPY [0.000, -0.000, 0.000]
At time 1668879082.267
- Translation: [5.544, 5.544, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
           in RPY [0.000, -0.000, 0.000]
```

Resultado 2: lanzamiento de `rosruncat tf_echo /world /turtle1`

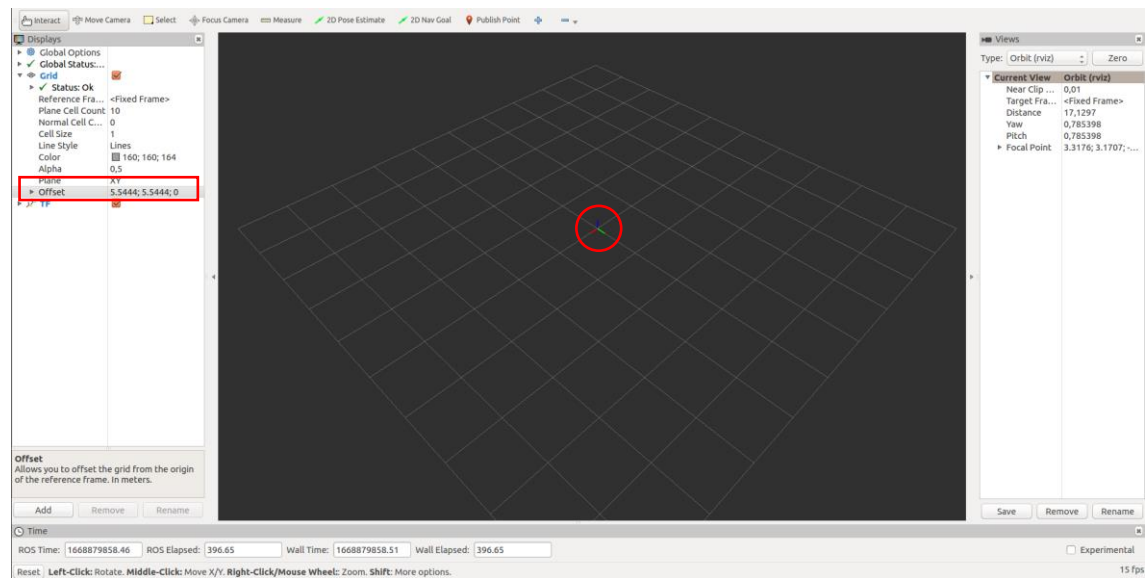
Una vez en este punto, tan solo es necesario lanzar *rviz* para poder representar el movimiento de la tortuga. Se recomienda utilizar la siguiente instrucción para lanzar *rviz*, ya con un entorno precargado donde la tortuga simulada será representada como un eje de coordenadas. La instrucción inicia *rviz* con el entorno contenido en X, el cual está preconfigurado para utilizar *turtlesim* como fuente de información.

```
turtlebot@rosindigo:~$ rosrn rviz rviz -d `rospack find turtle_tf`/rviz/turtle_rviz.rviz
```



Resultado 3: lanzamiento de rosrn rviz rviz con entorno para turtlesim

Como se puede observar en la figura anterior, *turtlesim* está representado por un eje de coordenadas con los colores rojo, verde y azul (ejes: x,y,z), para hacer más visible el eje ha sido remarcado por un círculo rojo. Esto se debe al tipo de cámara seleccionada por defecto, así como que la rejilla o malla no esta centrada y es necesario ubicarla. En primer lugar, se procederá a centrar la rejilla, para ello será necesario desplegar en el menú de la izquierda las opciones del elemento *Grid* y en *offset* establecer los valores 5,544445, 5,544445,0 (ejes: x,y,z). De este modo la rejilla quedará alineada junto a *Turtlesim*.

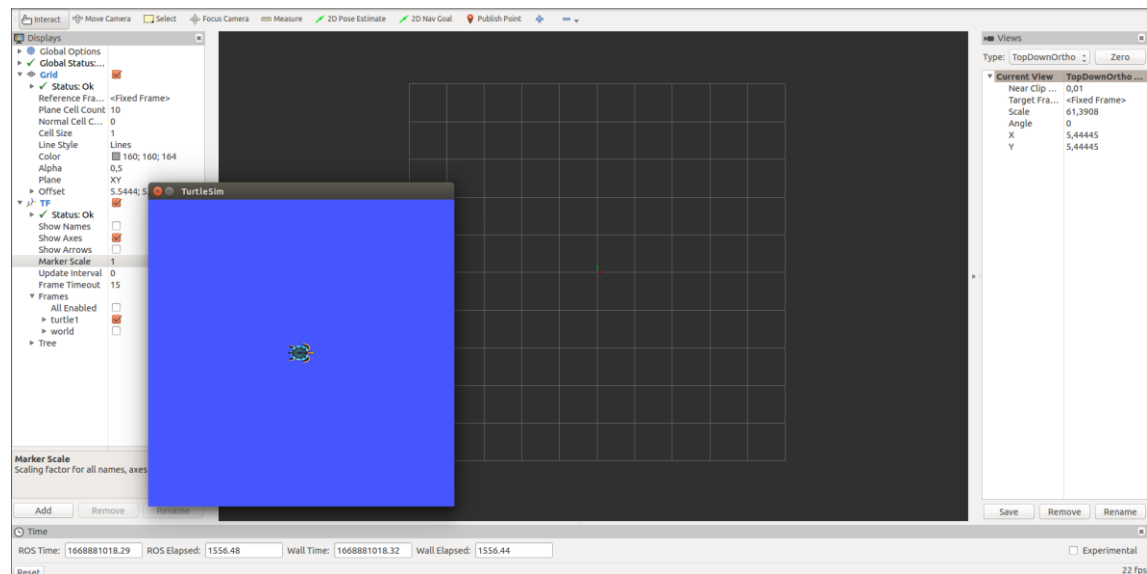


Resultado 4: modificaciones para poder observar adecuadamente el movimiento de Turtlesim

De este modo, el eje que representa a *turtlesim* queda en el centro de la rejilla y ahora al controlar a *turtlesim* mediante teclado siempre se mueve por la rejilla. Es importante destacar la orientación de la tortuga en función de los ejes representados, en rojo se representa el eje

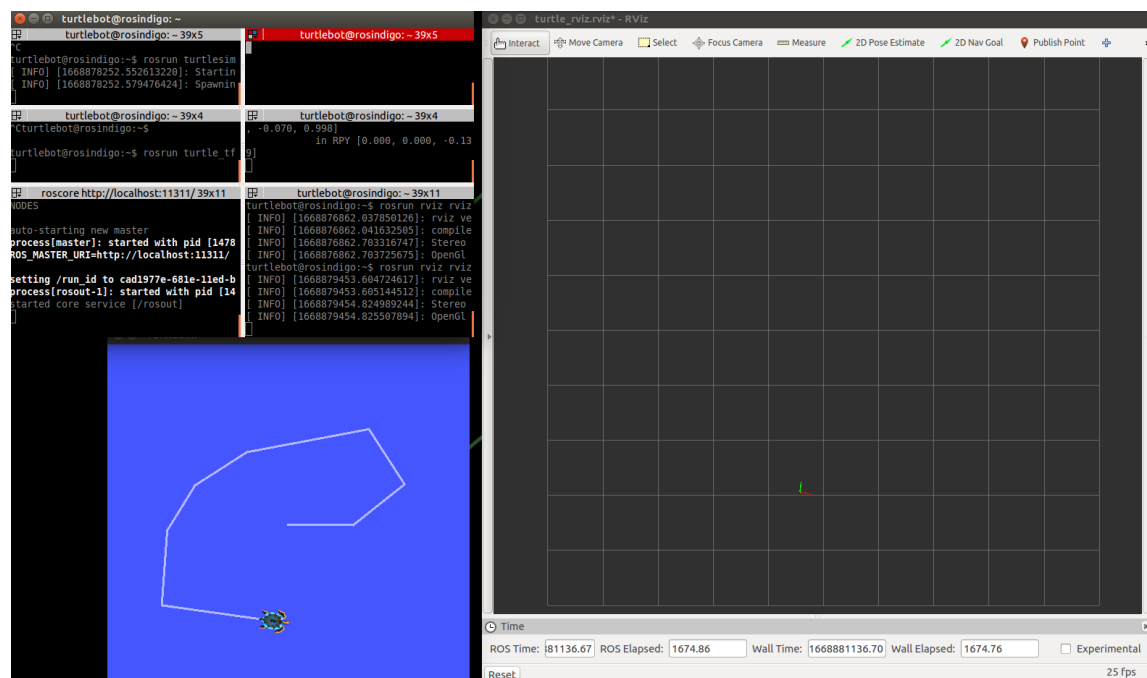
X, lugar donde apunta la cabeza de la tortuga *turtlesim*, en verde se representa el eje Y y en azul eje Z.

Como es conocido, *turtlesim* solo se desplaza en los ejes X e Y, por tanto, se podría cambiar a la vista Top-Down Orthographic, y el movimiento de la tortuga quedaría representado frente a los ejes X e Y. Para ello es necesario cambiar la cámara e indicar que la cámara enfoque al punto 5,54445 tanto para el eje X como para el eje Y. De este modo el movimiento de la tortuga será exactamente igual que el representado en la ventana de *turtlesim*.



Resultado 5: modificaciones de cámara y comparativa con turtlesim

Después de haber desplazado un poco la tortuga con el control por teclado, se puede observar cómo ha cambiado la orientación de la tortuga en la ventana de *Turtlesim*, así como en *rviz*.



Resultado 6: comparativa de ubicación de rviz con turtlesim



Se puede hacer uso de la instrucción *rostopic pub* de la primera guía para indicar a la tortuga que realice giros de manera constante y observar cómo se mueve igual en ambos entornos de representación. Para ello es necesario utilizar la siguiente instrucción:

```
turtlebot@rosindigo:~$ rostopic pub -r 1 /turtle1/cmd_vel geometry_msgs/Twist '[2.0,0.0,0.0]' '[0.0,0.0,2]'
```

A modo de conclusión, el entorno de visualización 3D no es solo interesante para el uso con *turtlesim*, sino que es interesante para el uso con cualquier otro robot soportado en ROS, ya que *rviz* permite interactuar al completo con el robot. Un ejemplo puede ser *turtlebot* que tan solo es necesario cargar el entorno necesario para comprobar cómo se muestra el robot al completo y cómo se cargan los elementos del entorno<sup>1</sup>.

### 3. Instrucción rosbag

La instrucción *rosbag* permite recoger todos los topics del sistema o algunos topics específicos. Esta información es almacenada de forma estructurada en un fichero. La instrucción es muy útil en el caso de estar depurando el funcionamiento de un robot, ya que esta permite almacenar toda la información relevante a los movimientos y detecciones que puede realizar el robot. Gracias a ello, en caso de error del robot, mediante la depuración se puede aislar el problema y corregirlo. La instrucción *rosbag* está sobrecargada con diferentes operadores como son:

- *rosbag record*: permite recoger los datos producidos en función de los topics
- *rosbag info*: permite visualizar la información que contiene un fichero .bag
- *rosbag play*: permite publicar la información de uno o más ficheros .bag en los topics correspondientes
- *rosbag check*: comprueba si el fichero .bag puede ser ejecutado en el sistema o bien si puede ser migrado a otro sistema
- *rosbag fix*: permite corregir posibles fallos en un fichero .bag para que sea ejecutado en el sistema actual
- *rosbag filter*: convierte un fichero .bag a expresiones de Python
- *rosbag compress*: comprime uno o más ficheros .bag
- *rosbag decompress*: descomprime uno o más ficheros .bag
- *rosbag reindex*: se produce una nueva indexación del fichero .bag sobre ficheros que están dañados

Las instrucciones más útiles de todas son *rosbag record*, *rosbag info* y *rosbag play*. En primer lugar, la instrucción *rosbag record* permite recoger los datos producidos en el sistema a través de todos los topics o bien sobre uno o varios topics concretos. Por ejemplo, para recoger el movimiento de *turtlesim* se pueden recoger todos los topics del sistema o tan solo */turtle1/cmd\_vel*, ya que este topic recoge la velocidad lineal y la velocidad angular enviada a la tortuga, así como también se podría recoger esta información junto con la posición de la tortuga en cada momento a través del topic */turtle1/pose*. Para iniciar la recogida de topics,

<sup>1</sup> <https://learn.turtlebot.com/2015/02/03/3/>

primero se ha lanzado *turtlesim* y se ha hecho uso de *rostopic pub* para generar la información. A continuación, se exponen las instrucciones utilizadas (cada una en una terminal diferente):

```
turtlebot@rosindigo:~$ roscore
turtlebot@rosindigo:~$ rosrunc turtlesim turtlesim_node
turtlebot@rosindigo:~$ rostopic pub -r 1 /turtle1/cmd_vel geometry_msgs/Twist '[2.0,0.0,0.0]' '[0.0,0.0,2]'
```

La instrucción necesaria para recoger la información de todos los topics es:

```
turtlebot@rosindigo:~$ rosbag record -a
```

La instrucción necesaria para recoger la información de todos los topics pero almacenándola en un fichero concreto es:

```
turtlebot@rosindigo:~$ rosbag record -a -O recogidaDatos.bag
```

La instrucción necesaria para recoger la información de los topics */turtle1/cmd\_vel* y */turtle1/pose* es:

```
turtlebot@rosindigo:~$ rosbag record /turtle1/cmd_vel /turtle1/pose
```

La instrucción necesaria para recoger la información de los topics */turtle1/cmd\_vel* y */turtle1/pose*, almacenando la información en un fichero concreto es:

```
turtlebot@rosindigo:~$ rosbag record -O recogidaDatos.bag /turtle1/cmd_vel /turtle1/pose
```

El resultado de cualquiera de estas instrucciones debe ser similar, en el se debe observar a qué topics se ha suscrito *rosbag record* y sobre qué fichero se están escribiendo estos datos.

```
^Cturtlebot@rosindigo:~$ rosbag record -O recogidaDatos.bag /turtle1/cmd_vel /turtle1/pose
[INFO] [1668883838.357320385]: Subscribing to /turtle1/cmd_vel
[INFO] [1668883838.368655469]: Subscribing to /turtle1/pose
[INFO] [1668883838.389018926]: Recording to recogidaDatos.bag.
^Cturtlebot@rosindigo:~$
```

Resultado 7: lanzamiento de *rosbag record*

La instrucción *rosbag info nombrefichero.bag* permite obtener un resumen de la información contenida en el fichero. Entre la información disponible se encuentra la ruta del fichero, la versión, la duración de la toma de datos (tiempo transcurrido desde el inicio al fin de la toma), fecha de inicio y fin de la toma de datos, el tamaño del fichero, cuántos mensajes se han capturado, si el fichero está comprimido, así como los tipos de mensajes y los topics recogidos (incluyendo el número de mensajes por topics). El resultado se puede observar en la siguiente figura:

```
turtlebot@rosindigo:~$ rosbag info recogidaDatos.bag
```

```
turtlebot@rosindigo:~$ rosbag info recogidaDatos.bag
path:      recogidaDatos.bag
version:   2.0
duration:  5.2s
start:     Nov 19 2022 19:50:38.68 (1668883838.68)
end:       Nov 19 2022 19:50:43.85 (1668883843.85)
size:      25.9 KB
messages:  262
compression: none [1/1 chunks]
types:     geometry_msgs/Twist [9f195f881246fdfa2798d1d3eebca84a]
           turtlesim/Pose      [863b248d5016ca62ea2e895ae5265cf9]
topics:    /turtle1/cmd_vel    5 msgs      : geometry_msgs/Twist
           /turtle1/pose       257 msgs   : turtlesim/Pose
turtlebot@rosindigo:~$
```

Resultado 8: lanzamiento de `rosbag info recogidaDatos.bag`

Una de las grandes ventajas de la recogida de datos es que posteriormente se pueden reproducir para depurar el funcionamiento del robot. Para ello, es necesario utilizar la instrucción `rosbag play nombrefichero.bag`. Con ello se reproduce todo el intercambio de mensajes capturados de los diferentes topics. En el caso de los topics capturados `/turtle1/cmd_vel` y `/turtle1/pose`.

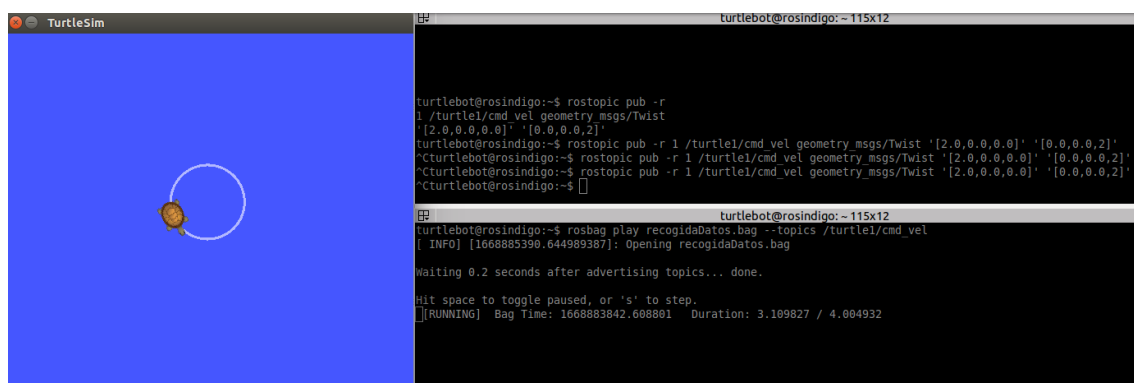
Se pueden reproducir todos los topics mediante la instrucción:

```
turtlebot@rosindigo:~$ rosbag play recogidaDatos.bag
```

Así mismo, se pueden reproducir un topic o varios topics específicos mediante el uso de:

```
turtlebot@rosindigo:~$ rosbag play recogidaDatos.bag --topics /turtle1/cmd_vel
```

Mediante esta instrucción, el robot simulado en `turtlesim_node` se mueve como se movió anteriormente en el tiempo.



Resultado 9: lanzamiento de `rosbag play recogidaDatos.bag` para un topic específico

Para la representación de datos del topic `/turtle1/pose`, en lugar de usar la ventana creada por `turtlesim`, se usará la herramienta `rqt_plot` descrita en la anterior guía. Esto se debe a que `turtlesim_node` utiliza el topic `/turtle1/pose` como salida, es decir, `turtlesim_node` es un nodo

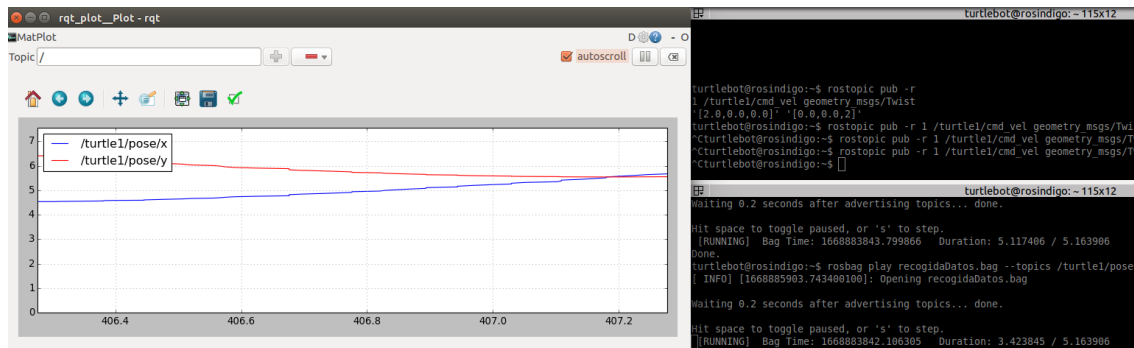
emisor o publicador y no acepta recibir o subscribirse a datos de `/turtle1/pose`. El funcionamiento de la herramienta `rqt_plot` se explicó en la guía anterior, aunque en esta se indicará una nueva instrucción para lanzar `rqt_plot` donde se incluye el filtro del topic en la misma instrucción. La instrucción es la siguiente: `roslaunch rqt_plot rqt_plot /topicFiltrar`

```
turtlebot@rosindigo:~$ roslaunch rqt_plot rqt_plot /turtle1/pose/x:y
```

Una vez lanzada la herramienta `rqt_plot`, se procede a lanzar la reproducción de los datos del topic `/turtle1/pose`

```
turtlebot@rosindigo:~$ rosbag play recogidaDatos.bag --topics /turtle1/pose
```

El resultado de ambas instrucciones es el siguiente:



Resultado 10: lanzamiento de `roslaunch rqt_plot rqt_plot /turtle1/pose/x:y` para un topic específico y mostrado con `rqt_plot`