Universidad Rey Juan Carlos

Práctica 3-B: Integración y estudio de dinámicas en Gazebo y ROS2

Modelado y Simulación de Robots

v 0.2

Objetivo:

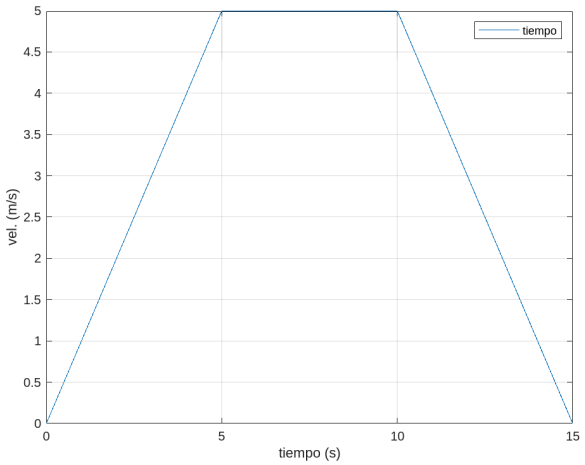
Realizar la simulación en dos entornos de gazebo diferentes usando el robot realizado en la práctica 2 para estudiar el comportamiento del robot y su modelo de colisión en mundos distintos.

Actividades:

En dos mundos de simulación preparados en Gazebo disponibles en el link:

https://gitlab.eif.urjc.es/juanscelyg/msr_world

Se debe realizar la simulación de un trayecto ejercido por una velocidad lineal del robot a través de un topic del tipo **Twist** de ROS, en torno al eje **X** del robot (*REP-103*). El comportamiento de la velocidad debe estar como se visualiza en la siguiente gráfica, dicho topic tiene que ser publicado por un nodo que esté programado en un en cualquier lenguaje de programación que esté habilitado en ROS (Python o C++), dicho nodo debe llamarse *"speed_controller"*:



***Nota**: Para simplificar la práctica se tendrá que eliminar el brazo robot.*

Control de la plataforma

Para controlar la plataforma se debe tener bien configurados la orientación de los ejes de cada rueda. A su vez se debe configurar los archivos de configuración de "*ros_control*" como se muestran en los anexos de este guión, publicados en aulavirtual.

Recuerde el orden de lanzamiento de los procesos debe ser, solo hasta cuando este completamente ejecutado lo solicitado en el launcher se debe continuar con el siguiente:

- *robot state publisher:* Publicación de la configuración en el topic robot_description. Solo informa que ha publicado información en el topic.

- *robot gazebo:* Spawn del robot en gazebo y en cada uno de los mundos. Se debe configurar para cada mundo. Este launcher quizás es el que más tiempo tarde en ejecución debido a que tenemos que esperar a que Gazebo visualice el robot por completo. Nota: Las ruedas pueden estar mal posicionadas y en color blanco, es normal. El tiempo de simulación debe estar contando

- *robot_controllers*: Lanzamiento de los controladores para mover el robot. Cuando se lance este launcher, las ruedas se ubicaran y tomarán su posición correspondiente. En la terminal donde se haya ejecutado deben visualizarse mensajes de color verde diciendo que los dos controladores están funcionando: *joint_state_broadcaster* y *DiffController* con le nombre seleccionado para tal fin.

El topic en el cual se publicará la velocidad deseada está en función de cómo se le haya denominado en el archivo de configuración del controlador. También se tendrá que almacenar la información de la odometría.

Obtención de Valores

Se debe recopilar durante todo el tiempo de la simulación los valores correspondientes a los valores de la articulaciones, suelen estar publicados en el topic ***joint_states***, también se requerirá guardar el valor de la información publicado por el sensor ***IMU*** cuyo nombre estará en función de cómo haya sido configurado en el **URDF**. El controlador emitirá la información concerniente a la odometría en un topic llamado: */"nombre_del_controlador"/odom.*

- los valores correspondientes a la posición y velocidad de las articulaciones de la cuatro ruedas
- los valores generados por el topic odom del controlador del robot
- los valores de la aceleración lineal, velocidad angular y magnetómetro que entrega la IMU.

Se recomienda usar rosbag para almacenar la información publicada en los topics, a continuación un link a como usar rosbag en ROS2: Link

Dentro del rosbag que se genere, es recomendable solo almacenar los topics que requerimos para dicha práctica, siendo los topics referentes a imágenes de no interés en este caso y no deberían guardarse en el rosbag.

Mundos a simular

Los dos mundos a simular son aportados en un repositorio (cuyo link está al inicio de este guión), en el cual basta con ubicarlo en el workspace. Uno de los mundos se llama ***sand*** y el otro se llama ***floor***. Tienen configuraciones diferentes y permiten hacer una comparativa respecto a dos valores diferentes de interacción con el suelo y el robot.

Ejecución

En la ejecución de la simulación deberíamos asegurarnos de que el desplazamiento del robot siga en la mayor medida el perfil de velocidad descrito en la imagen de este guión, las velocidades deben ser enviadas al robot a través del topic del controlador cuyo nombre suele ser: */"nombre_del_controlador"/cmd_vel_unstamped*.

Esta ejecución debería garantizar al menos el choque con uno de los cubos de la simulación y no debería exceder más de 15 segundos, tiempo que se inicia a contar desde que el robot recibe un valor de velocidad comandada en el topic del párrafo anterior.

Gráficas

En las gráficas que se generen deberían comparar el comportamiento del robot en cada uno de los escenarios. Siendo de especial interés las siguientes comparativas:

- Comparativa de las velocidades de las ruedas en los dos escenarios publicadas en *joint_states*
- Comparativa de la velocidad de las ruedas publicadas en *joint_states.* Y los desplazamientos que el controlador cree hacer, publicados en el topic de la odometría. Comparativa que se debe realizar entre los dos escenarios.
- Comparativa de las aceleraciones ejercidas por el robot en cada mundo, siendo de especial interés el momento en el cual choca con los cubos.

Para la obtención de estas gráficas a partir de un rosbag es altamente recomendable usar herramientas de visualización como pueden ser:

- Plotjuggler: https://github.com/facontidavide/PlotJuggler

En el repositorio anterior se encuentra la información suficiente para poder ejecutar ese paquete en ROS y ver las gráficas deseadas con tan solo dar clicks a los elementos a visualizar.