ROBÓTICA – G1

ROBOT OPERATING SYSTEM (ROS) - TURTLEBOT3

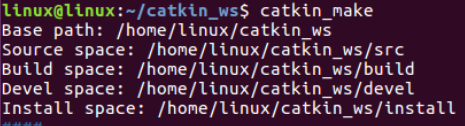
15/08/2022

Enlace del repositorio:

<https://github.com/luisanthony196/Robotica-201209>

Enlace del video: <https://drive.google.com/file/d/1TvkoxtTwseo8cculyb9jPQTPHx1MyU83/view?usp=sharing>

1. **Ejercicios**
2. Configurar un espacio de trabajo – catkin workspace



1. Instale los paquetes correspondientes a Turtlebot3: ros-kinectic-turtlebot3 y ros-kinect-turtlebot3-gazebo



1. Crear un paquete llamado turtle\_control, que dependa de rospy geometry\_msgs,sensor\_msgs, turtlebot3 y turtlebot3-gazebo.

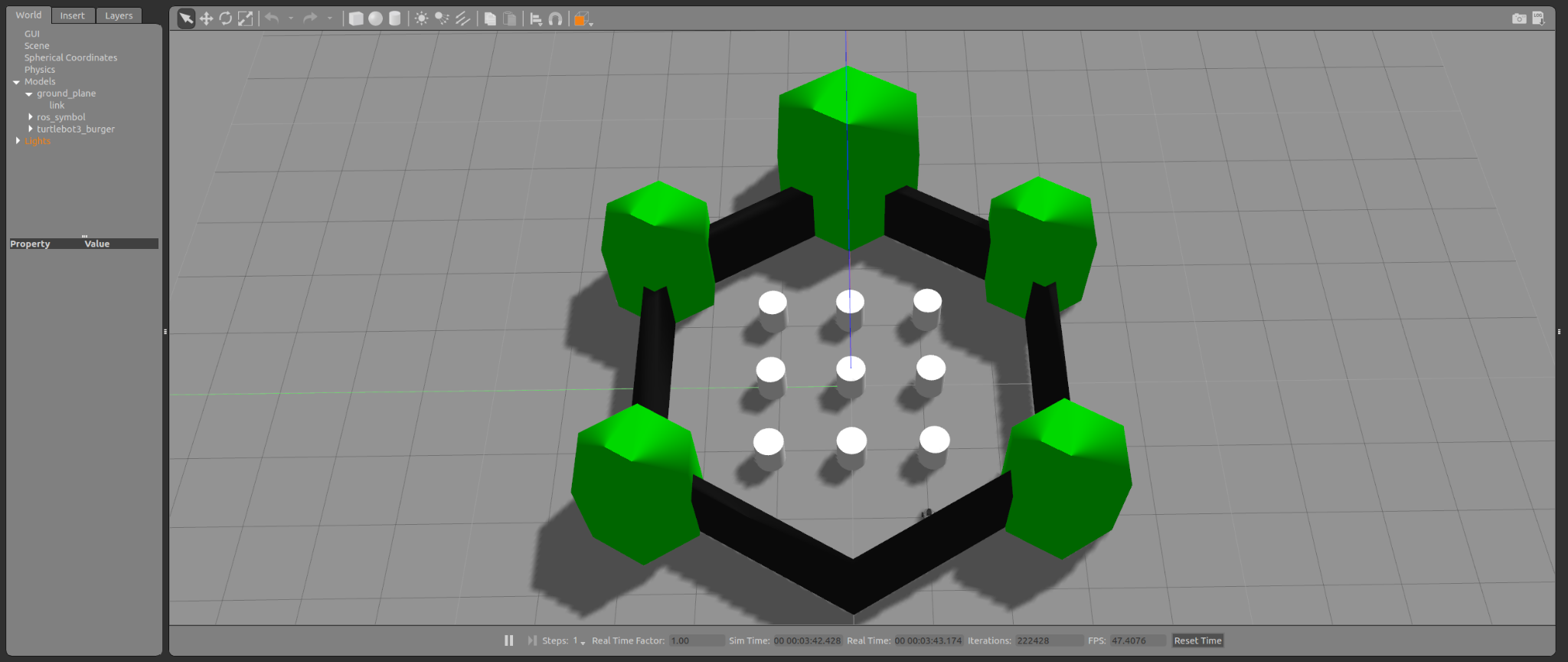


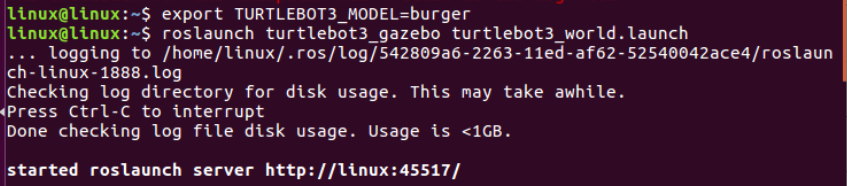
1. Defina la siguiente variable:

$ export TURTLEBOT3\_MODEL=burger

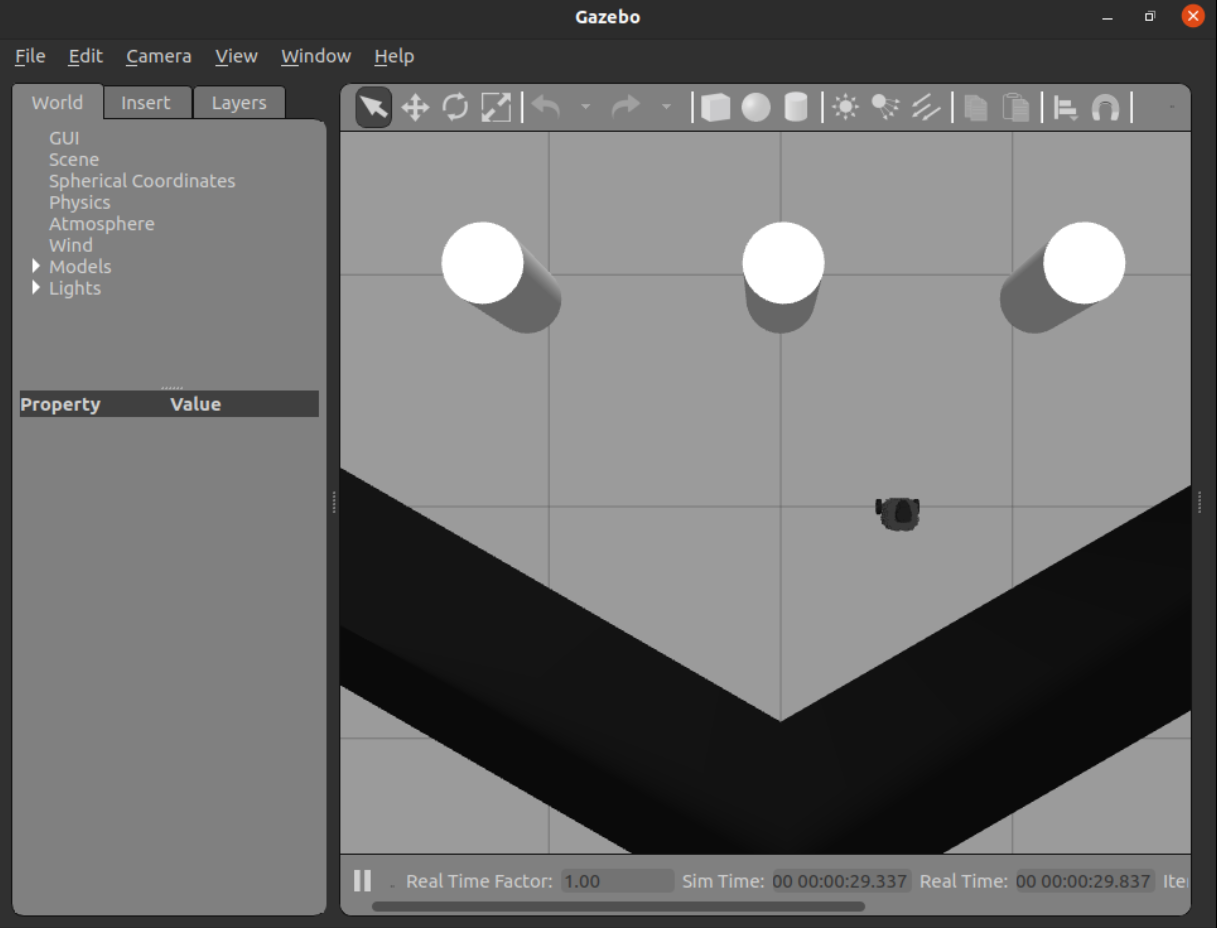


1. Identifique el paquete que contiene el archivo “turtlebot3\_world.launch”. Ejecute el archivo launch y asegúrese que el ambiente simulado es el siguiente:

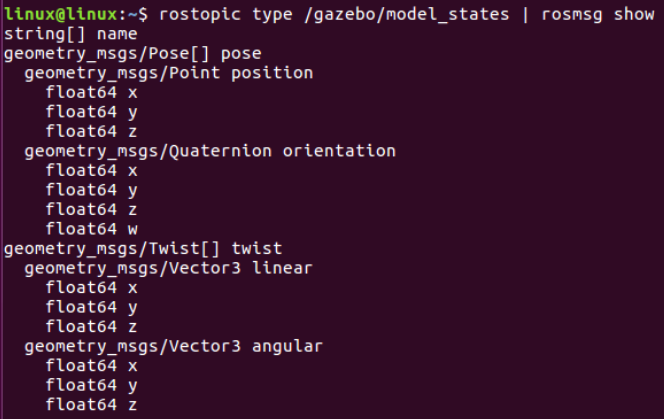




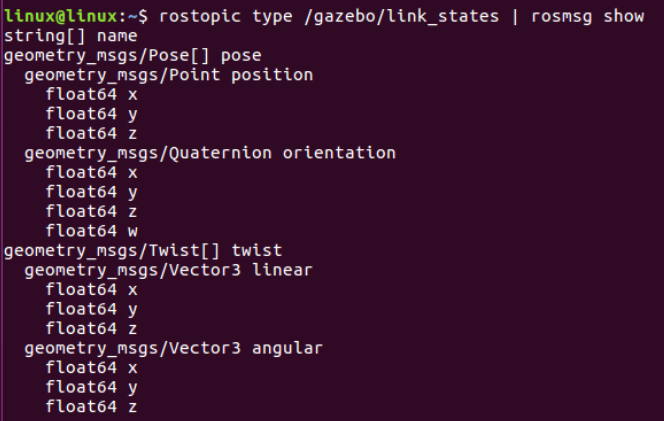
Gazebo funcionando



1. Muestre desde consola la estructura de los mensajes de cada uno de los tópicos asociados al robot.

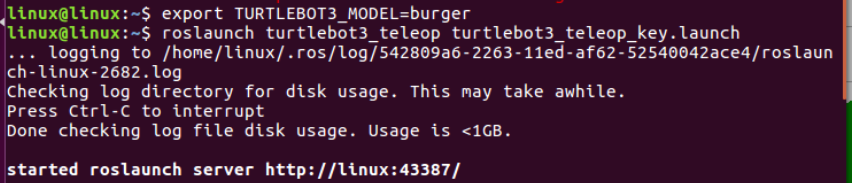


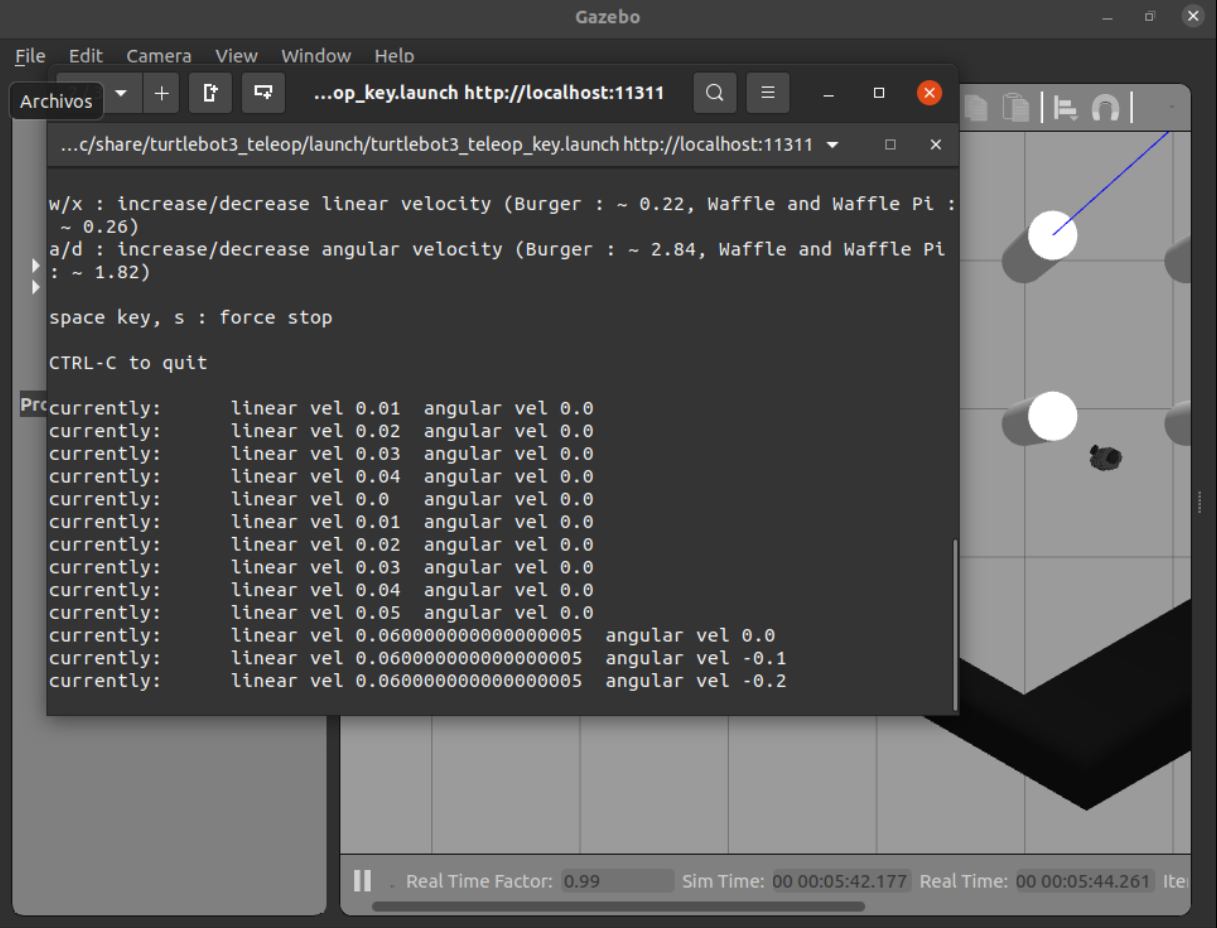
El tópico /gazebo/model\_states es aquel que se controla la posición y la velocidad del turtlebot3\_burger



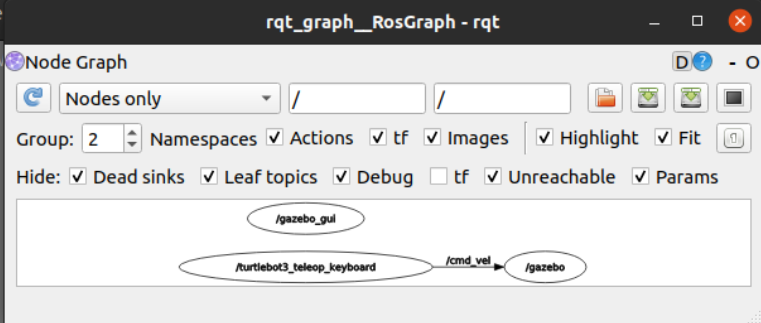
El tópico /gazebo/link\_states controla la estructura del turtlebot3\_burger, como lo son la base del robot y sus dos ruedas

1. Verifique que tiene instalado el paquete turtlebot3\_teleop, el cual tiene implementado el nodo “turtlebot3\_teleop\_key.launch”, ejecute el nodo para teleoperar el robot.





1. Muestra el grafo de nodos y tópicos, describa las funciones de cada tópico.



El nodo /gazebo está suscrito al tópico /cmd\_vel el cual obtiene sus datos a partir del nodo /turtlebot3\_teleop\_keyboard. Este tópico entrega data relacionada a la velocidad lineal y angular del elemento

1. Lea los datos del sensor LIDAR implementado en el robot. Cree un nodo python que se suscriba al tópico \**scan**, para que el robot navegue evitando obstáculos. Identifique el obstáculo que se encuentra frente al robot con las siguientes ecuación:

range\_ahead = instancia.ranges[len(instancia.ranges)/2]

closest\_range = min(instancia.ranges)

Para la navegación del robot utilice las siguientes funciones: **rospy.Duration()** y rospy.Time.Now()

Código del script en Python

///////////////////////////////////////////////////////////////

import rospy

from geometry\_msgs.msg import Twist

from sensor\_msgs.msg import LaserScan

class TurtleBot:

def \_\_init\_\_(self):

# Inicia nodo burger

rospy.init\_node('burger', anonymous=True)

def \_\_init\_\_(self):

# Guarda el publicador de la velocidad del robot

self.velocity\_publisher = rospy.Publisher('/cmd\_vel', Twist, queue\_size=10)

# Se subscribe a la información del sensor de distancia

self.prox\_subscriber = rospy.Subscriber('/scan', LaserScan, self.update\_prox)

self.prox = LaserScan()

self.rate = rospy.Rate(10)

def update\_prox(self, data):

self.prox = data

def move(self):

vel\_msg = Twist()

t0 = rospy.Time.now().to\_sec()

vel\_msg.linear.y = 0

vel\_msg.linear.z = 0

vel\_msg.angular.x = 0

vel\_msg.angular.y = 0

while(True):

range\_ahead = 0

if (len(self.prox.ranges) > 0):

mini = self.prox.range\_min

ahead = self.prox.ranges[0]

left = self.prox.ranges[15]

right = self.prox.ranges[345]

print(ahead, right, left)

if (ahead - mini <= 0.25 or left - mini <= 0.15 or right - mini <= 0.15):

if (left > right):

vel\_msg.linear.x = 0.1

vel\_msg.angular.z = 0.6

else:

vel\_msg.linear.x = 0.1

vel\_msg.angular.z = -0.6

else:

vel\_msg.linear.x = 0.2

vel\_msg.angular.z = 0

self.velocity\_publisher.publish(vel\_msg)

self.rate.sleep()

vel\_msg.linear.x = 0

vel\_msg.angular.z = 0

self.velocity\_publisher.publish(vel\_msg)

if \_\_name\_\_ == '\_\_main\_\_':

try:

burger = TurtleBot()

burger.move()

except rospy.ROSInterrupt

///////////////////////////////////////////////////////////

1. Cree un archivo roslaunch para iniciar el nodo python creado en el paso anterior.

Código del archivo .launch

--------------------------------------------------------------------------------------------

<launch>

<node pkg="turtle\_control" name="burger" type="obstacles.py"/>

</launch>

1. **Entregable**

En el repositorio agregue las siguientes carpetas:

- Informe 03: contiene un informe paso a paso de los ítems solicitados.

- Laboratorio 03: incluye el paquete creado con el script generado.

Grabe un video de su trabajo de max 2min.