Rospy

Michael Bosello

michael.bosello@unibo.it

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Smart Vehicular Systems A.Y. 2021/2022

Disclaimer

- We will see the basics of rospy publish/subscribe
- Check out the documentation for additional functions/parameters

Rospy: the ROS client library for Python

How to create a Node in Pyhton

- Ensure the node is started as a Python script
  - The python file must start with:

```
#!/usr/bin/env python
```

- Import rospy

```
import rospy
```

- Initialize the node
  - You can call this function only once
  - Name should be unique

```
rospy.init_node('node_name')
```

Publisher

- Import the message type

```
from ackermann_msgs.msg import AckermannDriveStamped
```

- Create the publisher
  - Queue size 0 is infinite, be careful

```
drive_pub = rospy.Publisher('/nav', AckermannDriveStamped, queue_size=0)
```

- Create a message

```
ack_msg = AckermannDriveStamped()
ack_msg.drive.speed = 0.5
ack_msg.drive.steering_angle = 0
```

- Send the message

```
drive_pub.publish(ack_msg)
```

Subscriber

- Import the message type

```
from sensor_msgs.msg import LaserScan
```

- Create a callback

```
def lidar_callback(lidar_data):
        for i in range(len(lidar_data.ranges)):
        #do something
```

- Create the subscriber

```
lidar_subscriber = rospy.Subscriber("scan", LaserScan, lidar_callback)
```

Node Cycle

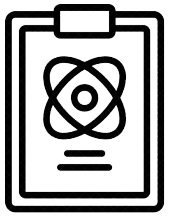
- *Cycle until shutdown is requested*

```
while not rospy.is_shutdown():
#do/pub something
```

- *Spin()* sleeps until *is_shutdown* is True

```
#setup callbacks
rospy.spin()
```

Lab Exercise 1



- Automatic Emergency Braking (see next slide)
  - Create a package
  - Create a node that takes the Lidar ranges, computes the time-to-collision, and stops the car if it is below a threshold
  - Build and run the package

Automatic Emergency Braking: TTC

Time-to-collision

- TTC is the time it would take for the ego-vehicle and an object to interchept one another given that they maintain current heading and velocity

$$TTC_i(t) = \frac{r_i(t)}{[v_x(t)cos(\theta_i)]_+}$$

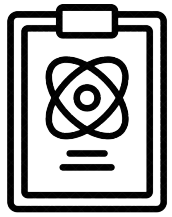
The denominator is the range-rate which is the time derivative of distance (projecting the velocity onto the vector)

Where:

- r_i is the distance measured by beam i
- v_x is the forward speed
- θ_i is the angle of beam i
- $[]_+$ is the operator $\max(x, 0)$

When the TTC of one of the beams is below a safety threshold: **Brake**

Lab Exercise 2



Pick one:

- Wall Following

  https://f1tenth-coursekit.readthedocs.io/en/stable/assignments/labs/lab3.html

- Follow The Gap

  https://f1tenth-coursekit.readthedocs.io/en/stable/assignments/labs/lab4.html

- Read the text and implement the node
  - Ignore “Deliverables and Submission” as we do not grade the lab exercises