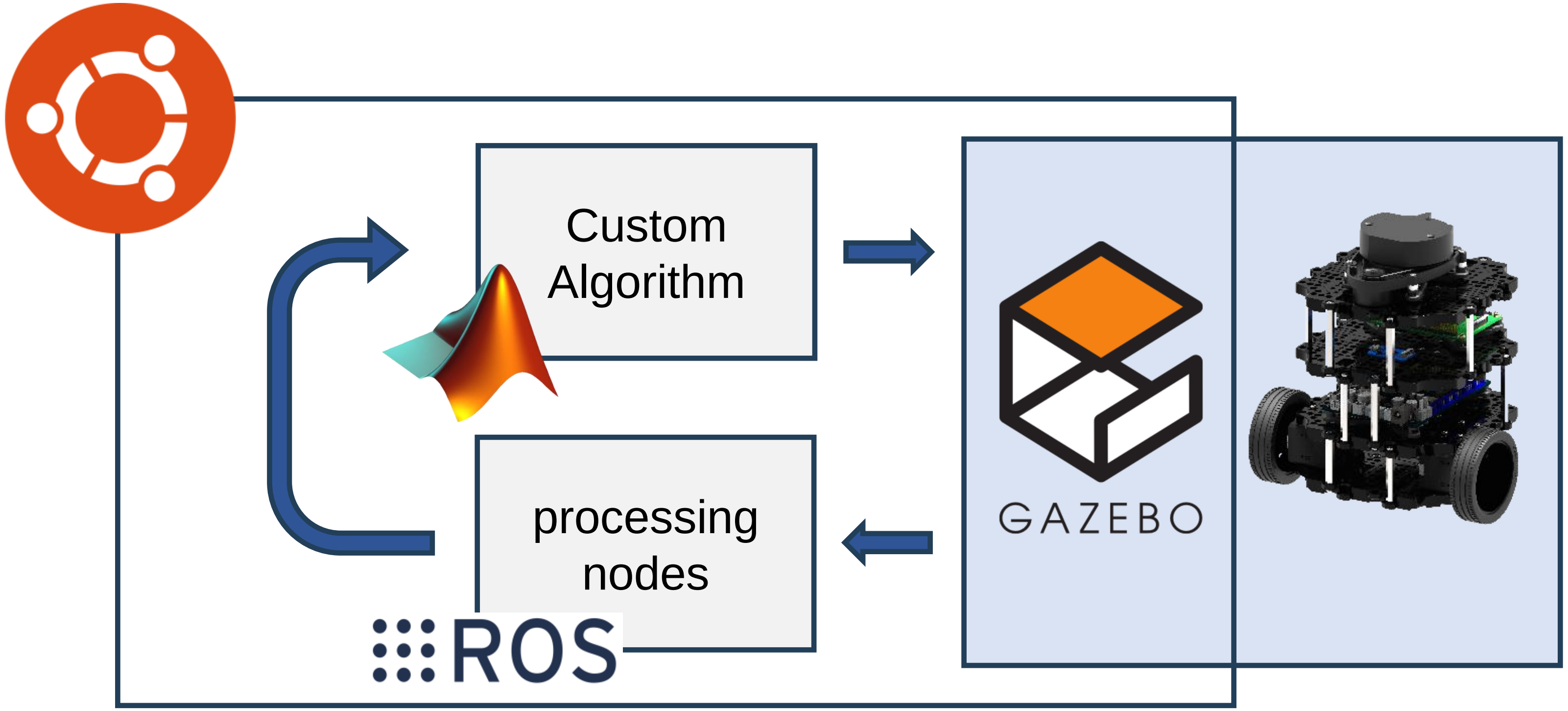
POLITECNICO DI MILANO

AUTONOMOUS VEHICLES

S. Arrigoni

Introduction

Full Architecture



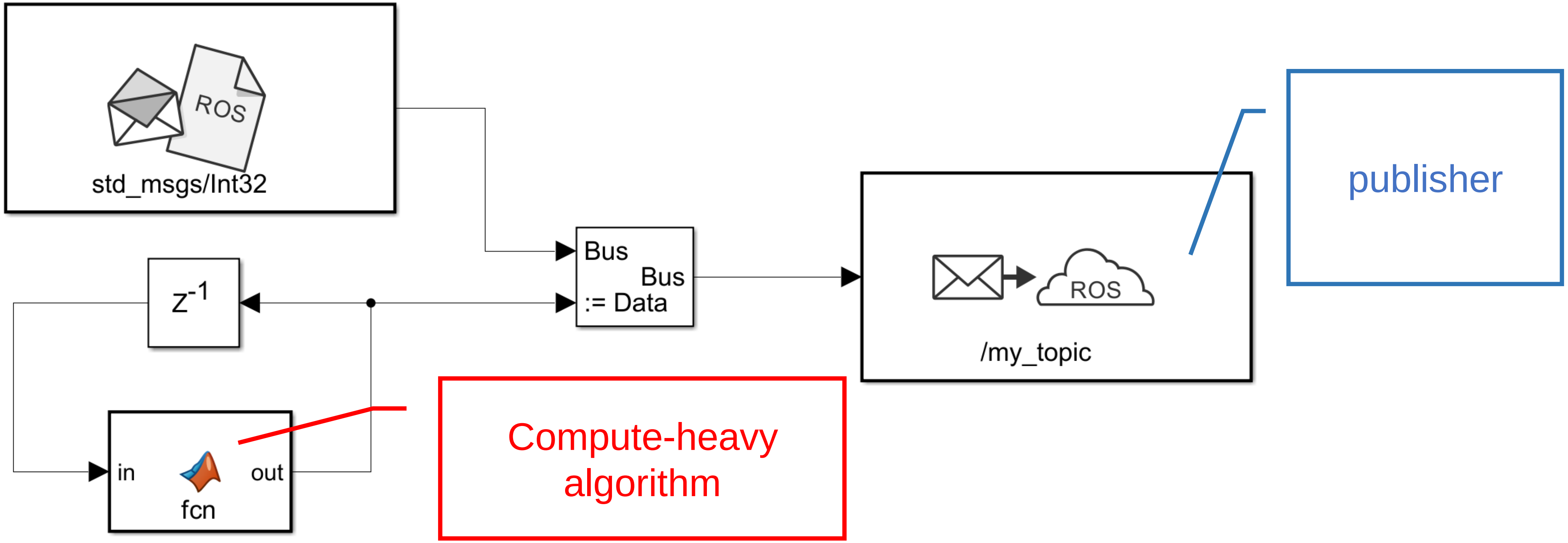
- ✓ Fast prototyping
- ✓ Easy debugging
- ✓ slow

Code Optimization & Real Testing

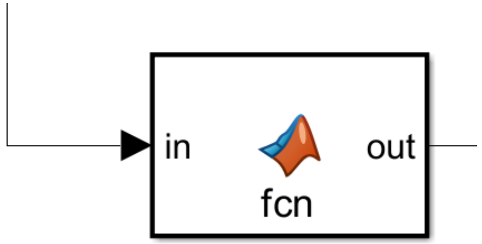
Code Optimization

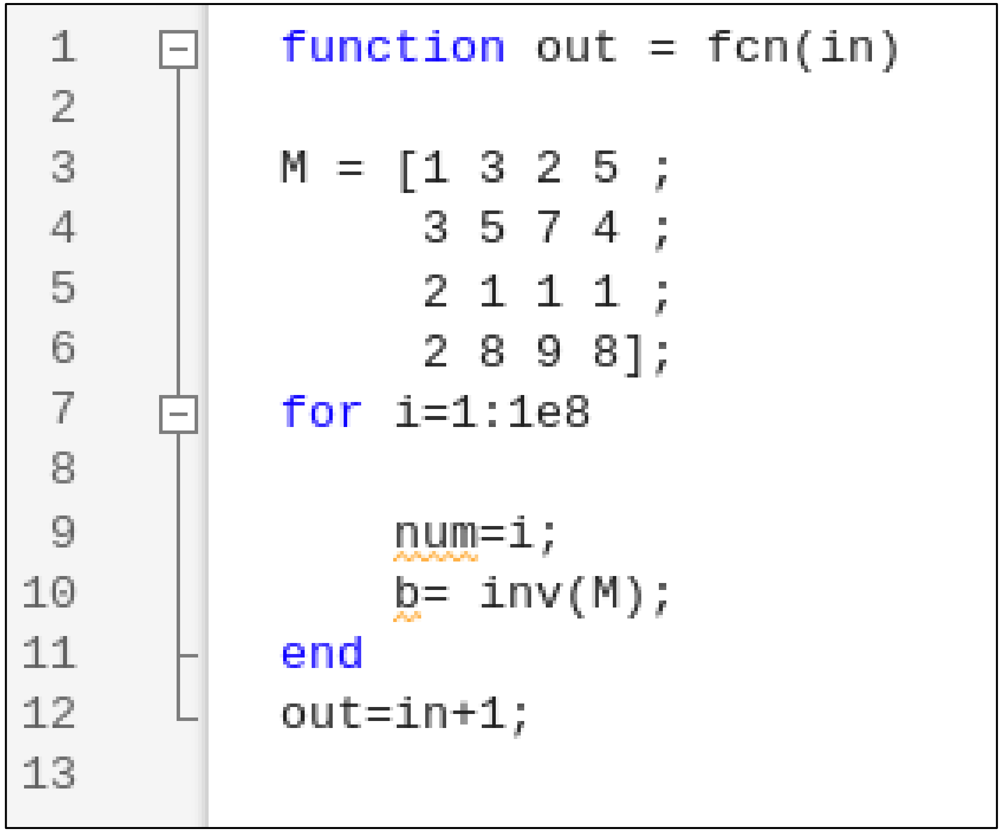
Is it possible to optimize the code?

Let’s try with an example:



Is it possible to optimize the code?





```
function out = fcn(in)

M = [1 3 2 5 ;
     3 5 7 4 ;
     2 1 1 1 ;
     2 8 9 8];
for i=1:1e8

    num=i;
    b= inv(M);
end
out=in+1;

```

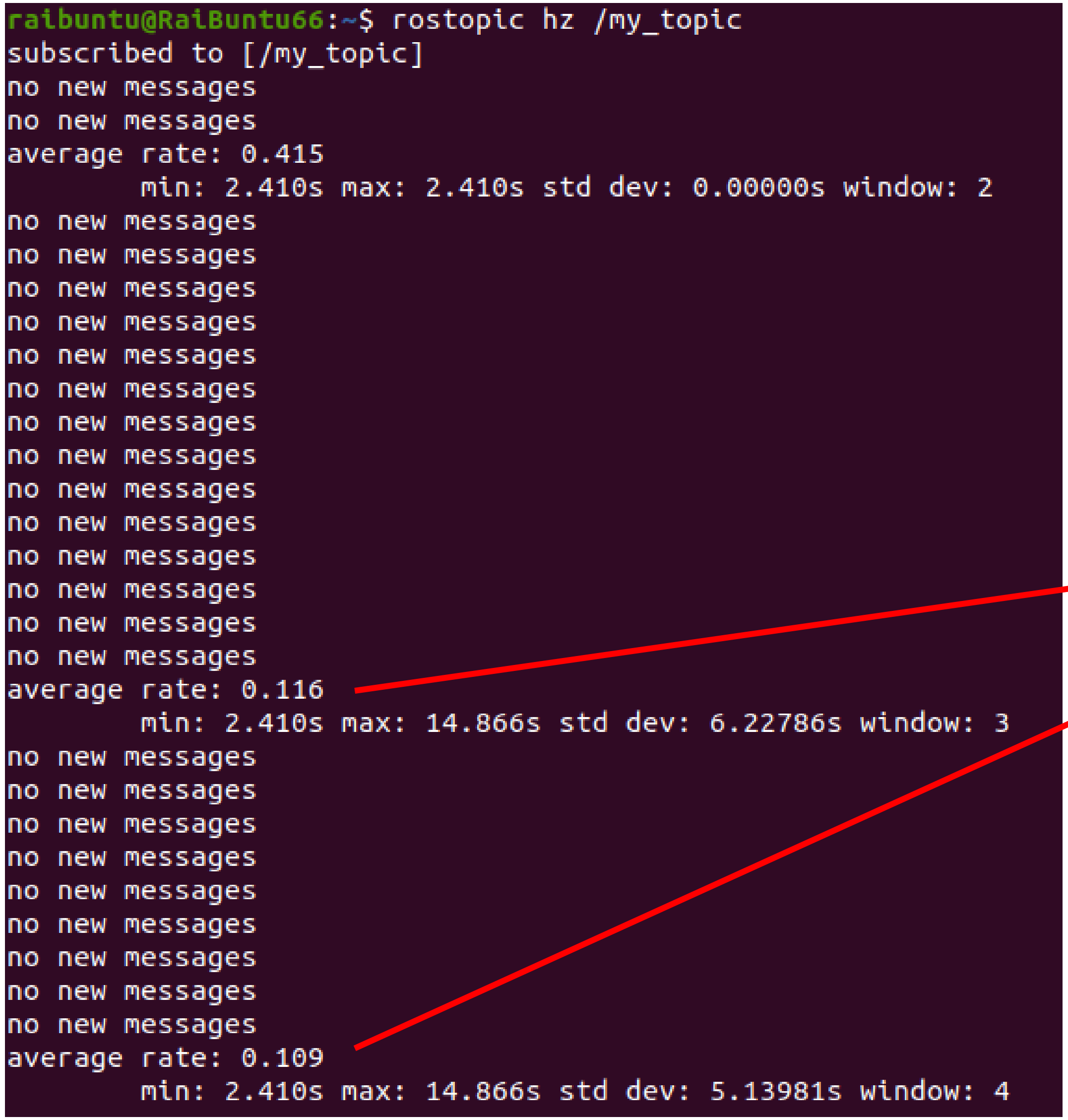
Output:

- 1,2,3,4,5,…

Frequency:

- 10Hz (?)

Is it possible to optimize the code?



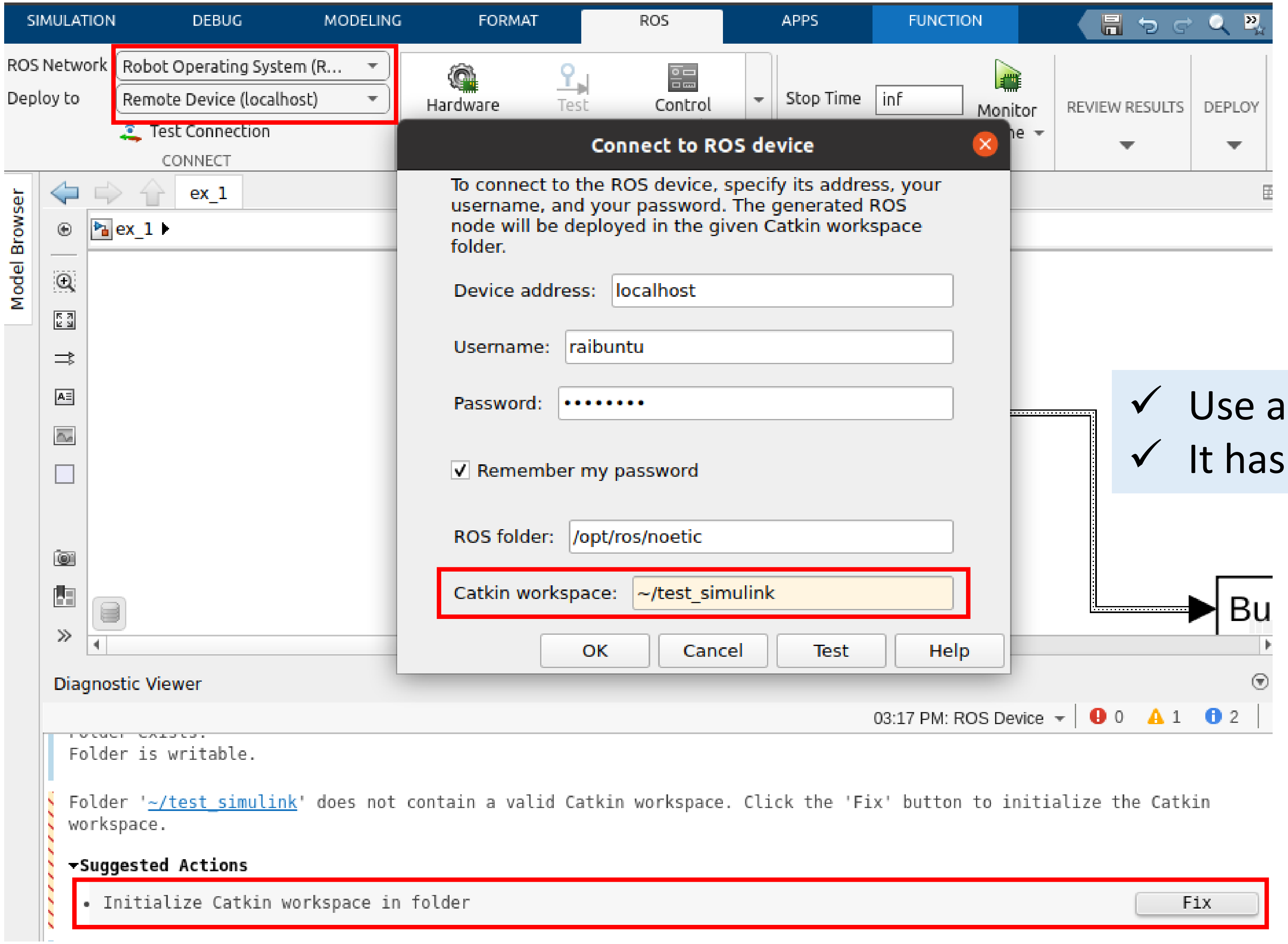
```
raibuntu@RaiBuntu66:~$ rostopic hz /my_topic
subscribed to [/my_topic]
no new messages
no new messages
average rate: 0.415
        min: 2.410s max: 2.410s std dev: 0.00000s window: 2
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
average rate: 0.116
        min: 2.410s max: 14.866s std dev: 6.22786s window: 3
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
no new messages
average rate: 0.109
        min: 2.410s max: 14.866s std dev: 5.13981s window: 4
```

Let’s try to measure it:

Less than 1 Hz!!

Is it possible to optimize the code?

Let’s optimize it:



ROS Network: Robot Operating System (R...

Deploy to: Remote Device (localhost)

Stop Time: inf

Connect to ROS device

To connect to the ROS device, specify its address, your username, and your password. The generated ROS node will be deployed in the given Catkin workspace folder.

Device address: localhost

Username: raibuntu

Password: ••••••••

☑ Remember my password

ROS folder: /opt/ros/noetic

Catkin workspace: ~/test_simulink

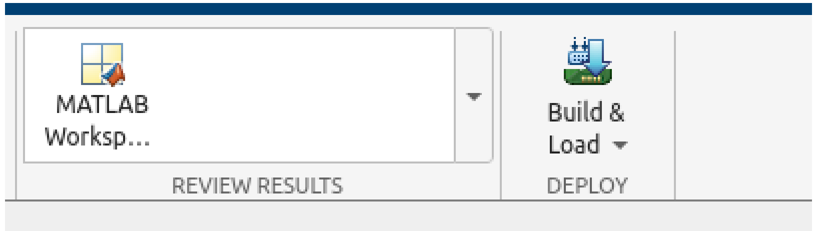
OK | Cancel | Test | Help

```
Folder exists.
Folder is writable.

Folder '~/test_simulink' does not contain a valid Catkin workspace. Click the 'Fix' button to initialize the Catkin
workspace.

▾Suggested Actions
 • Initialize Catkin workspace in folder                                                    Fix
```

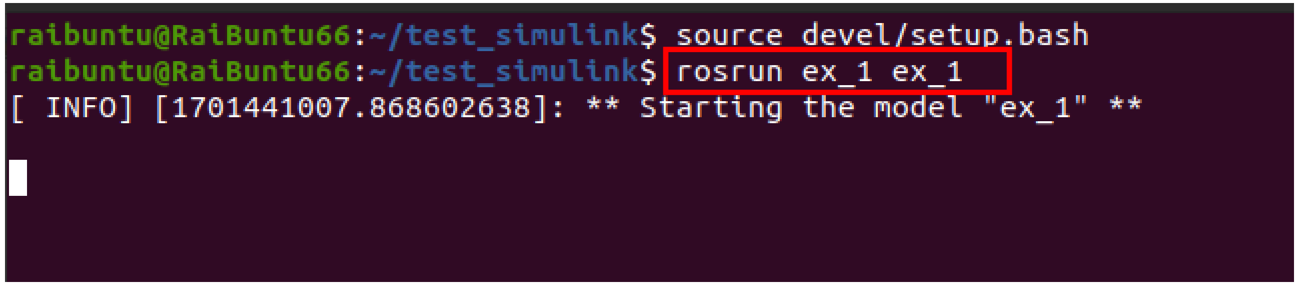
- ✓ Use a folder you are not into!
- ✓ It has to be a workspace

Is it possible to optimize the code?

Let’s create a new ROS node



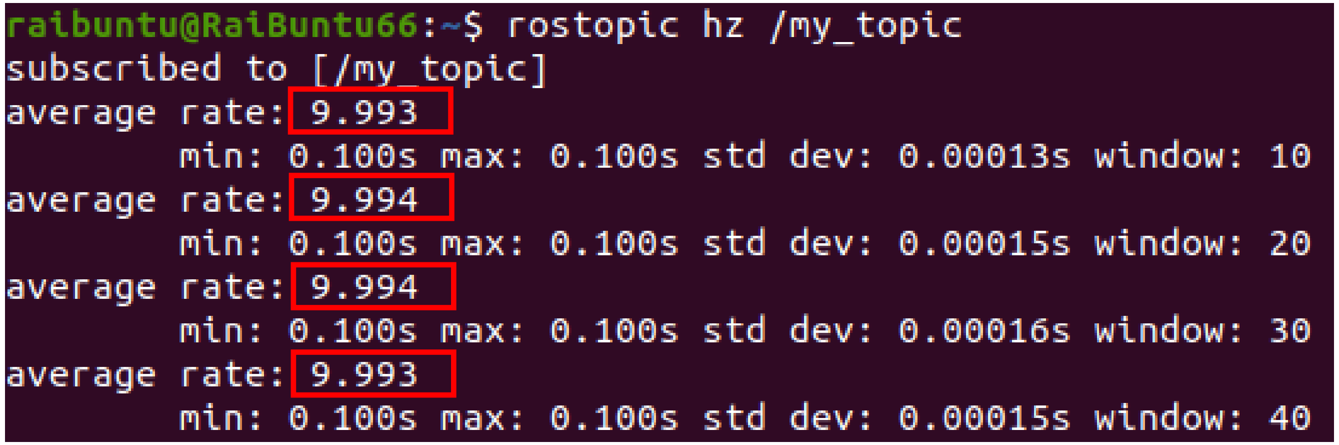
Let’s run it!



```
raibuntu@RaiBuntu66:~/test_simulink$ source devel/setup.bash
raibuntu@RaiBuntu66:~/test_simulink$ rosrun ex_1 ex_1
[ INFO] [1701441007.868602638]: ** Starting the model "ex_1" **
```

✓ The name you gave it

Is it possible to optimize the code?



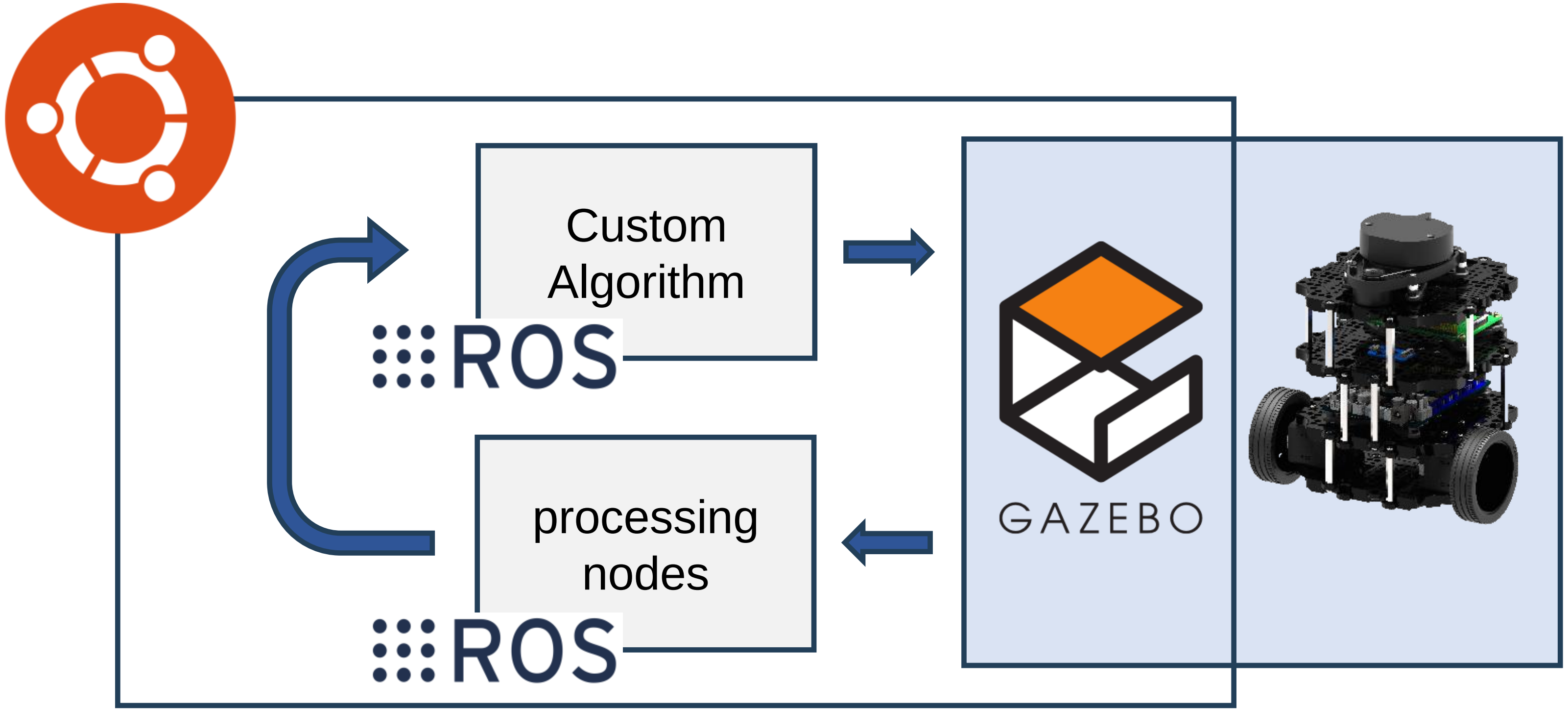
```
raibuntu@RaiBuntu66:~$ rostopic hz /my_topic
subscribed to [/my_topic]
average rate: 9.993
        min: 0.100s max: 0.100s std dev: 0.00013s window: 10
average rate: 9.994
        min: 0.100s max: 0.100s std dev: 0.00015s window: 20
average rate: 9.994
        min: 0.100s max: 0.100s std dev: 0.00016s window: 30
average rate: 9.993
        min: 0.100s max: 0.100s std dev: 0.00015s window: 40
```

Let’s try to measure it:

Stable around 10 Hz!!

Is it possible to optimize the code?

Full Architecture

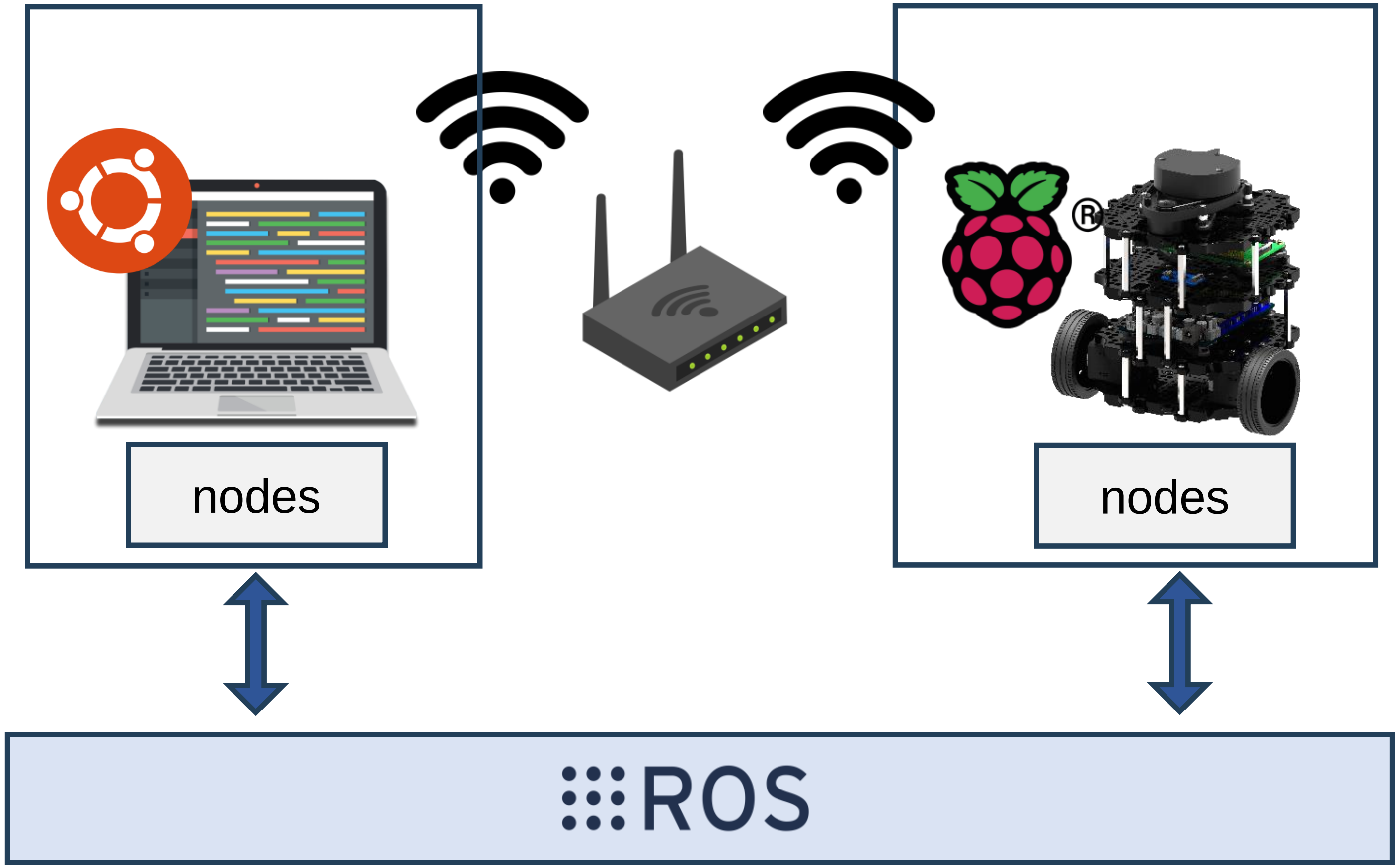


- ✓ Longer process
- ✓ no debug
- ✓ Fast & stable

Code Optimization & Real Testing

Real Testing

Real Testing



- ✓ PC+ Pi4 in the same network
- ✓ ROS master on PC
- ✓ Bring-up procedure on Pi4

Real Testing

ROS Setup (each side)

```
$ gedit ~/.bashrc

ROS_MASTER_URI = http://IP_of_PC:11311
ROS_HOSTNAME = IP_of_PC
```

```
$ ssh ubuntu@{IP_of_TB3}
$ nano ~/.bashrc

ROS_MASTER_URI = http://IP_of_PC:11311
ROS_HOSTNAME = IP_of_TB3
```

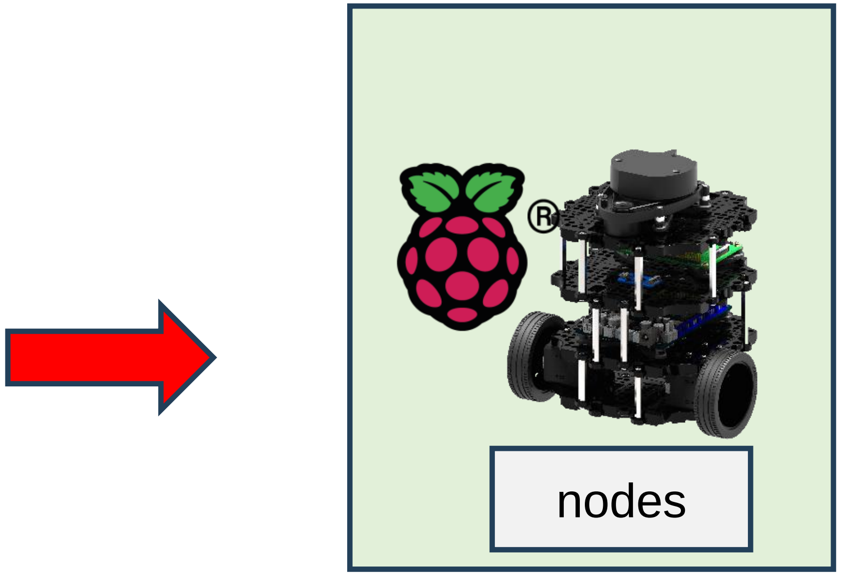
Real Testing

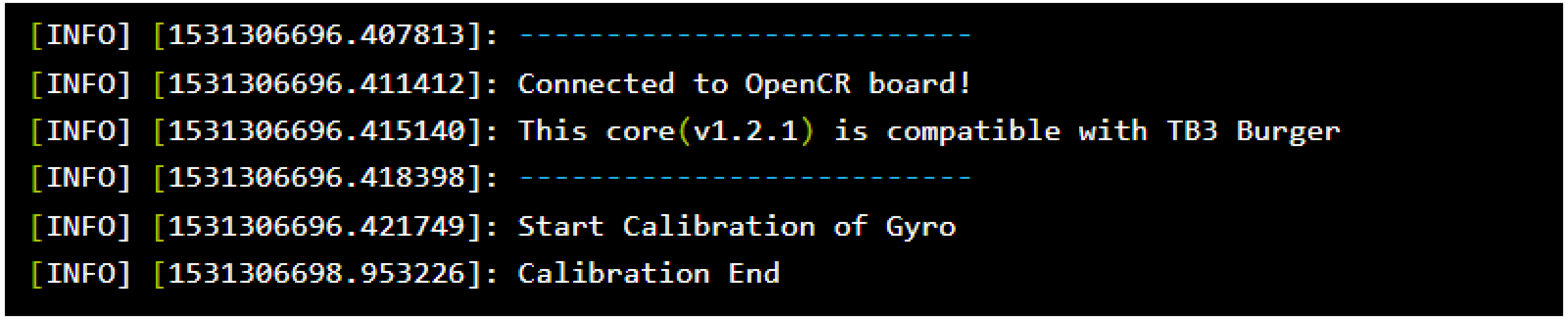
ROS Turn on TB3 (bringup) (*after roscore on PC*)

```
$ ssh ubuntu@{IP_of_TB3}
```

➔ Insert password

```
$ export TURTLEBOT3_MODEL=${TB3_MODEL}
$ roslaunch turtlebot3_bringup turtlebot3_robot.launch
…
```



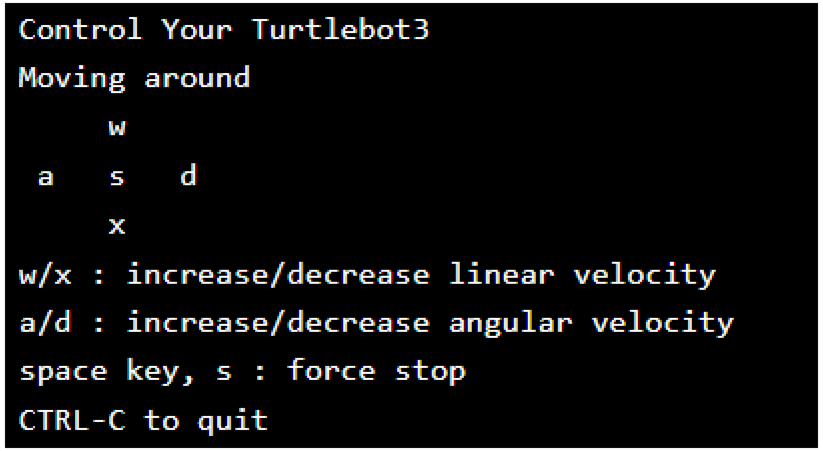


```
[INFO] [1531306696.407813]: --------------------------
[INFO] [1531306696.411412]: Connected to OpenCR board!
[INFO] [1531306696.415140]: This core(v1.2.1) is compatible with TB3 Burger
[INFO] [1531306696.418398]: --------------------------
[INFO] [1531306696.421749]: Start Calibration of Gyro
[INFO] [1531306698.953226]: Calibration End
```

Real Testing

ROS Run control node on PC

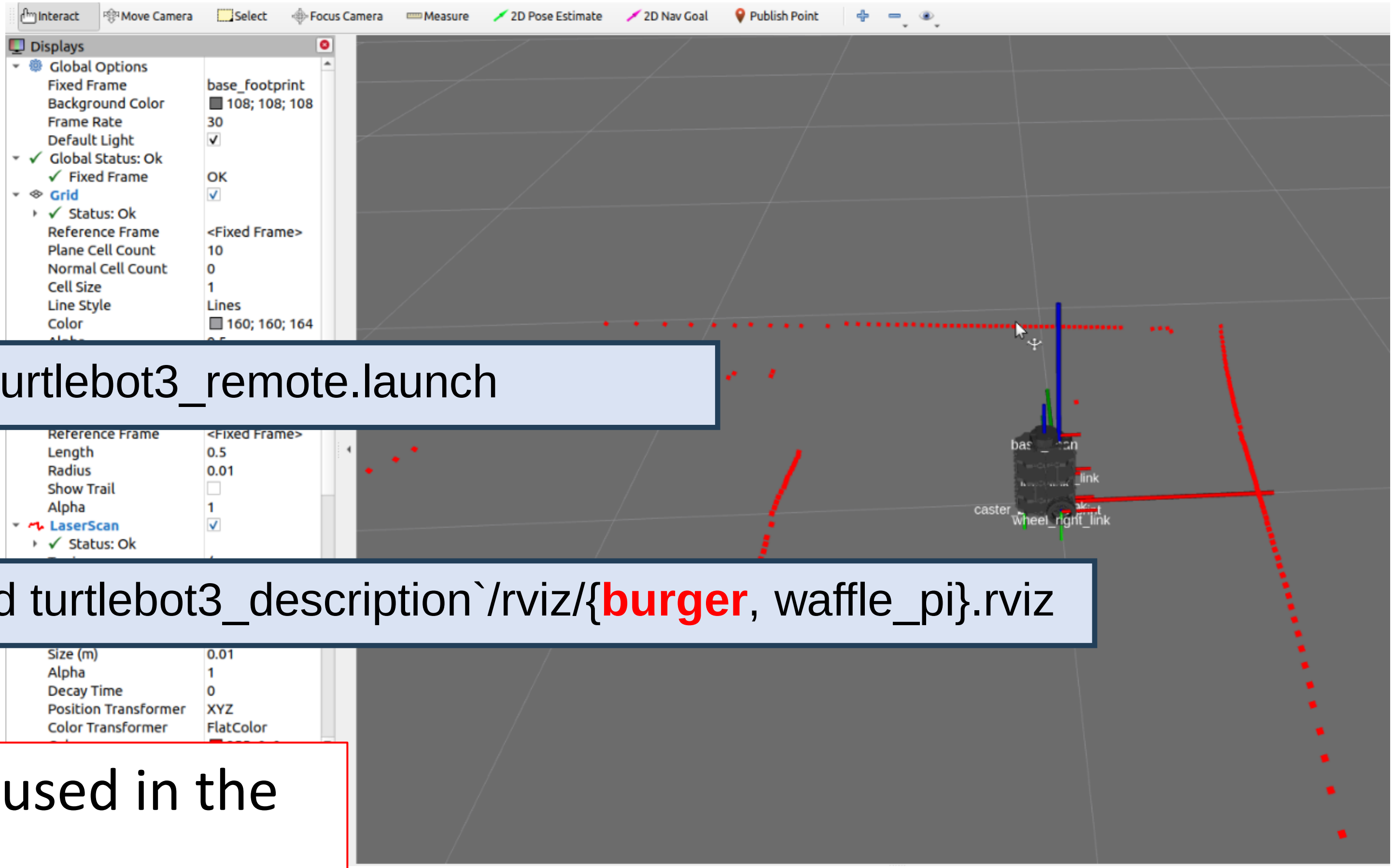
```
$ export TURTLEBOT3_MODEL=${TB3_MODEL}
$ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch
```



```
Control Your Turtlebot3
Moving around
     w
 a   s   d
     x
w/x : increase/decrease linear velocity
a/d : increase/decrease angular velocity
space key, s : force stop
CTRL-C to quit
```

Real Testing

ROS Run visualization on PC



Global Options	
Fixed Frame	base_footprint
Background Color	108; 108; 108
Frame Rate	30
Default Light	☑
✓ Global Status: Ok	
✓ Fixed Frame	OK
Grid	☑
✓ Status: Ok	
Reference Frame	<Fixed Frame>
Plane Cell Count	10
Normal Cell Count	0
Cell Size	1
Line Style	Lines
Color	160; 160; 164
Reference Frame	<Fixed Frame>
Length	0.5
Radius	0.01
Show Trail	□
Alpha	1
LaserScan	☑
✓ Status: Ok	
Size (m)	0.01
Alpha	1
Decay Time	0
Position Transformer	XYZ
Color Transformer	FlatColor

New terminal

```
$ roslaunch turtlebot3_bringup turtlebot3_remote.launch
```

New terminal

```
$ rosrun rviz rviz -d `rospack find turtlebot3_description`/rviz/{burger, waffle_pi}.rviz
```

...also previous example of rviz used in the course works...

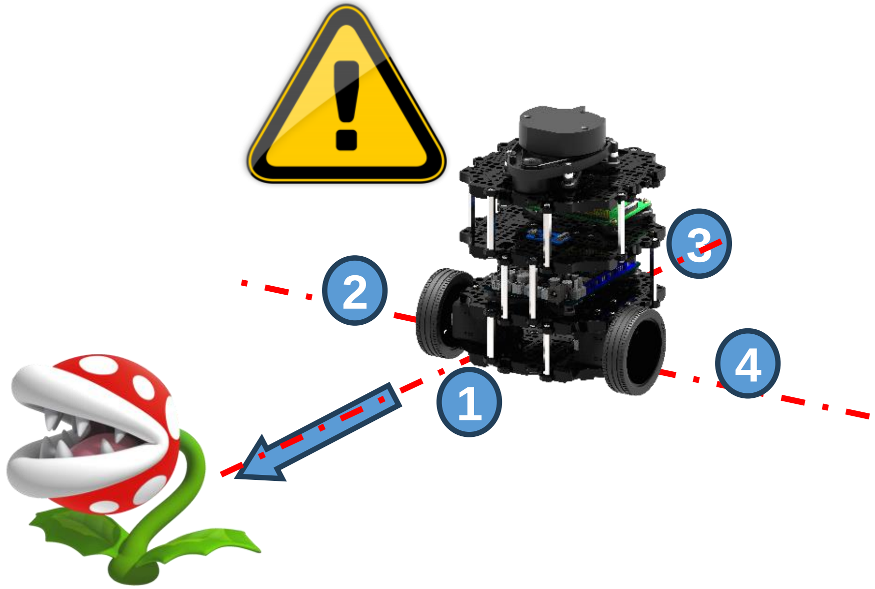
Code Optimization & Real Testing

All together: an example

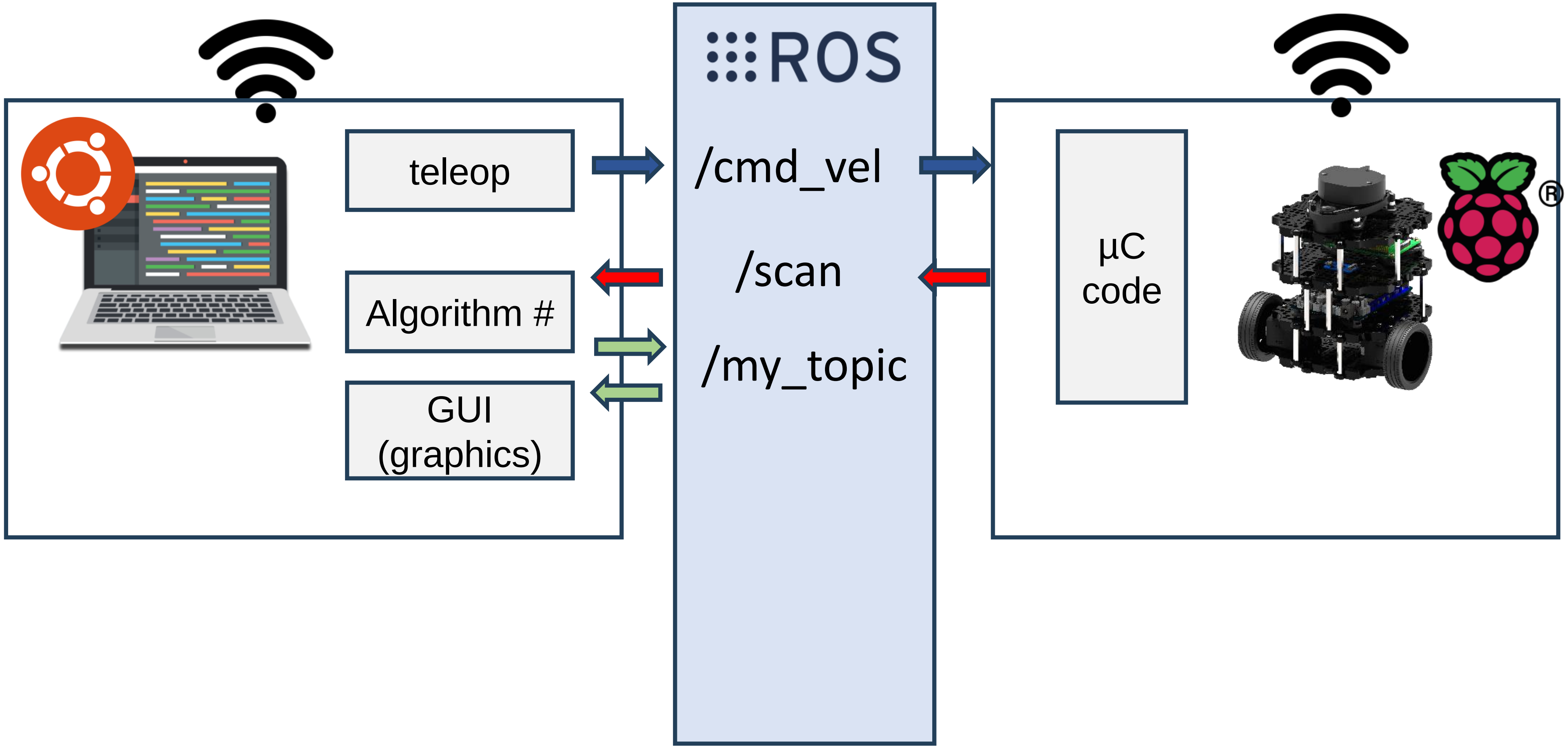
Real Example

Let’s define an algorithm that:

- ✓ Subscribes to /scan message
- ✓ Warns when an obstacle is too close
- ✓ Stable & reliable frequency
- ✓ Shows the potential collision area



Real Example



teleop

Algorithm #

GUI (graphics)

ROS

/cmd_vel

/scan

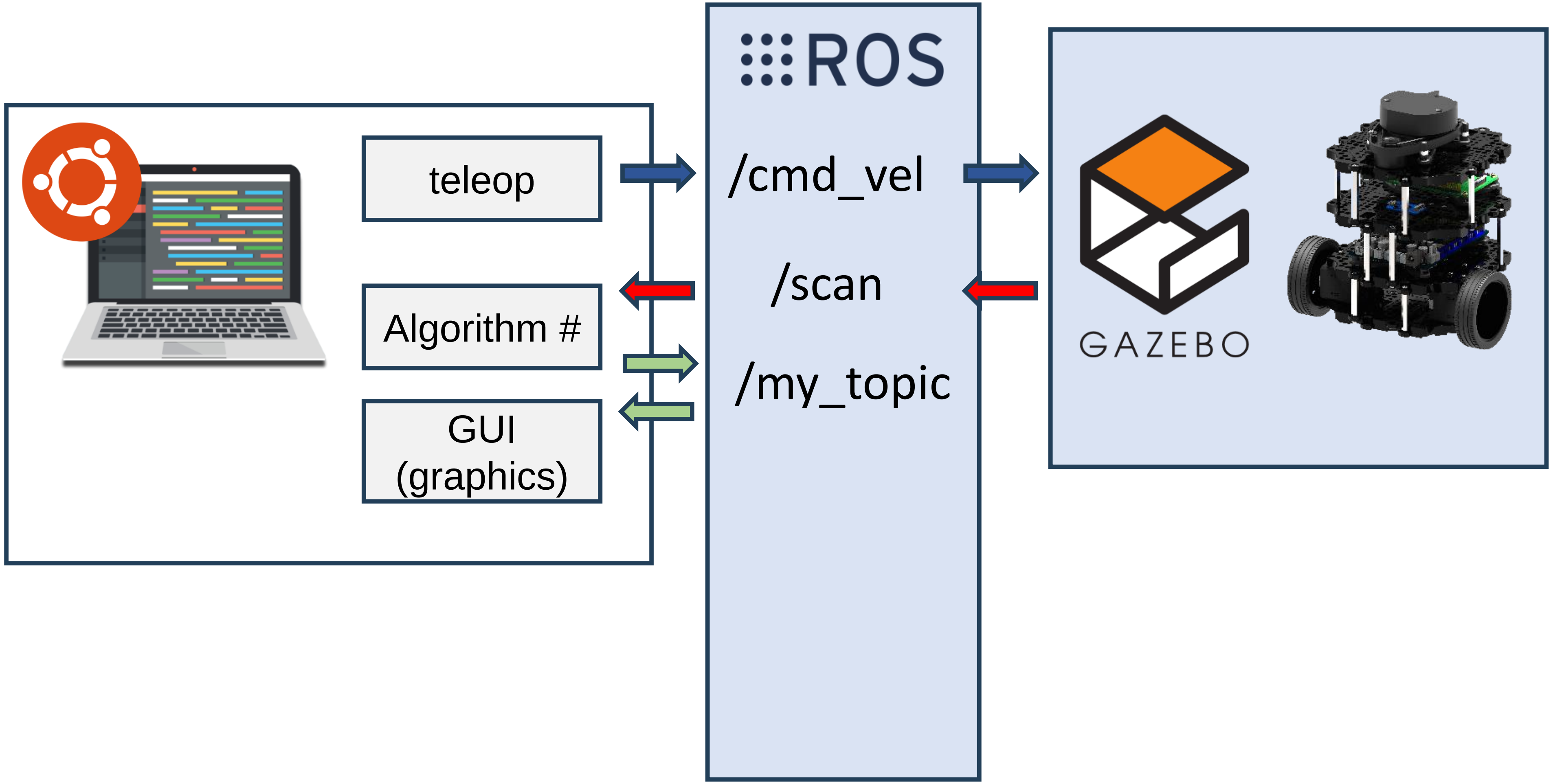
/my_topic

µC code

Real Example

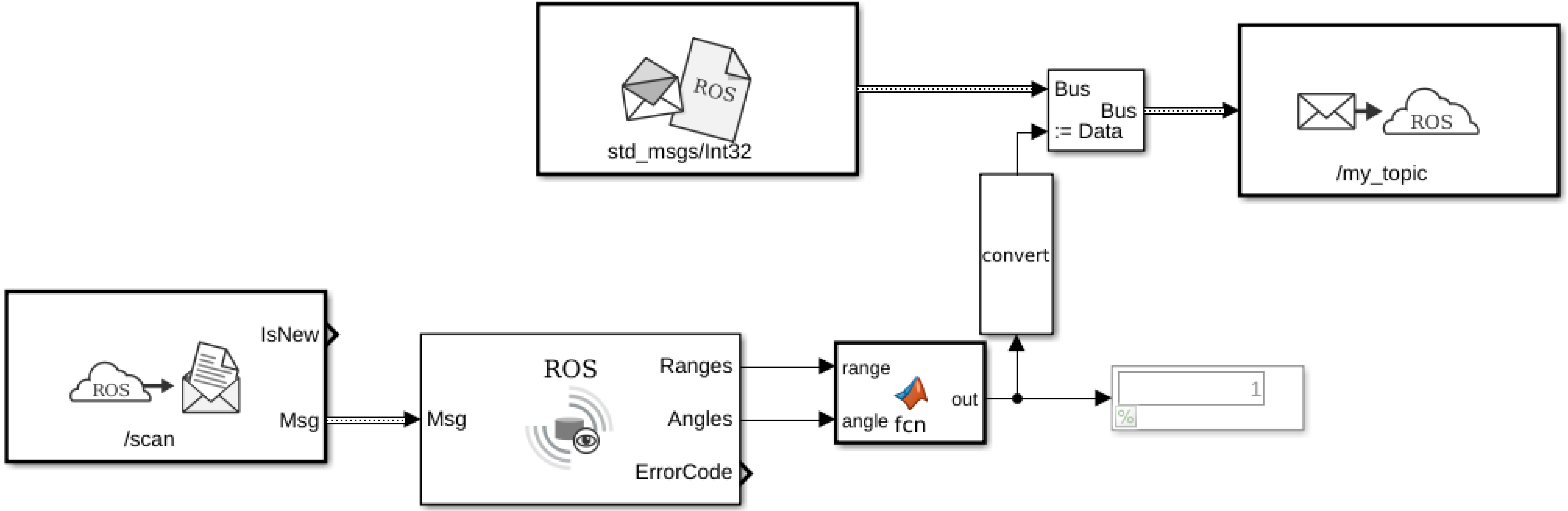
- ROS Setup (each side)
- ROS Turn on TB3 (bringup) (*after roscore on PC*)
- ROS Run control node on PC
- ROS Generate Algorithm # (*let's start testing in simulation*)
- ROS Run visualization on PC

Real Example

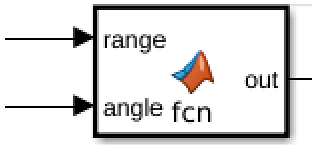


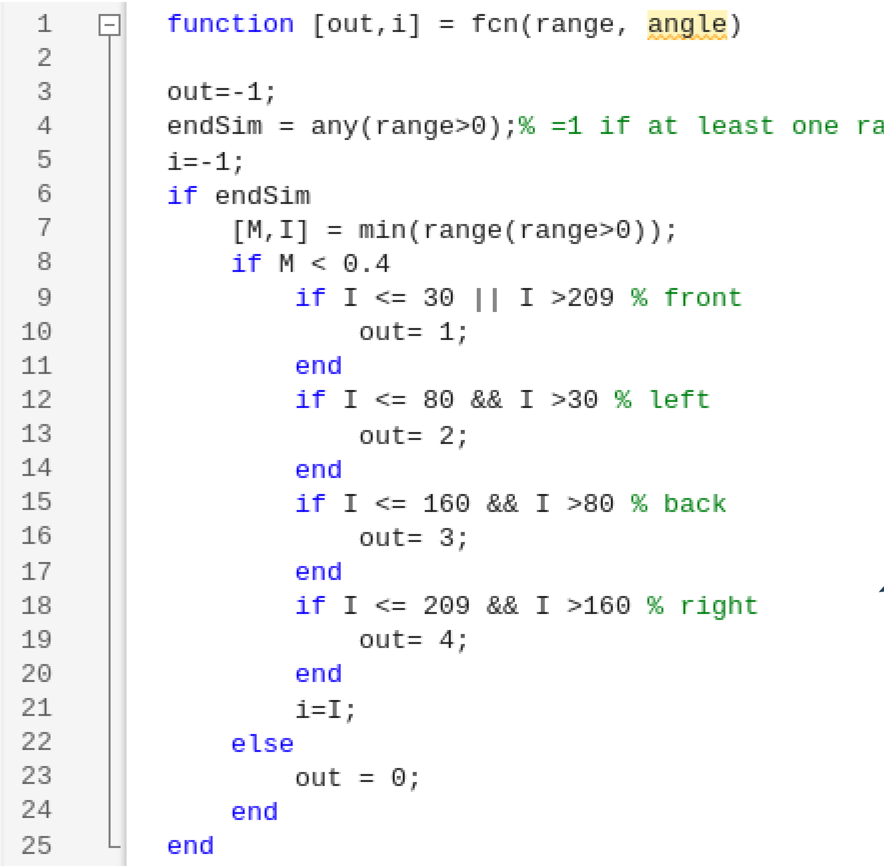
Real Example

ROS Generate Algorithm #

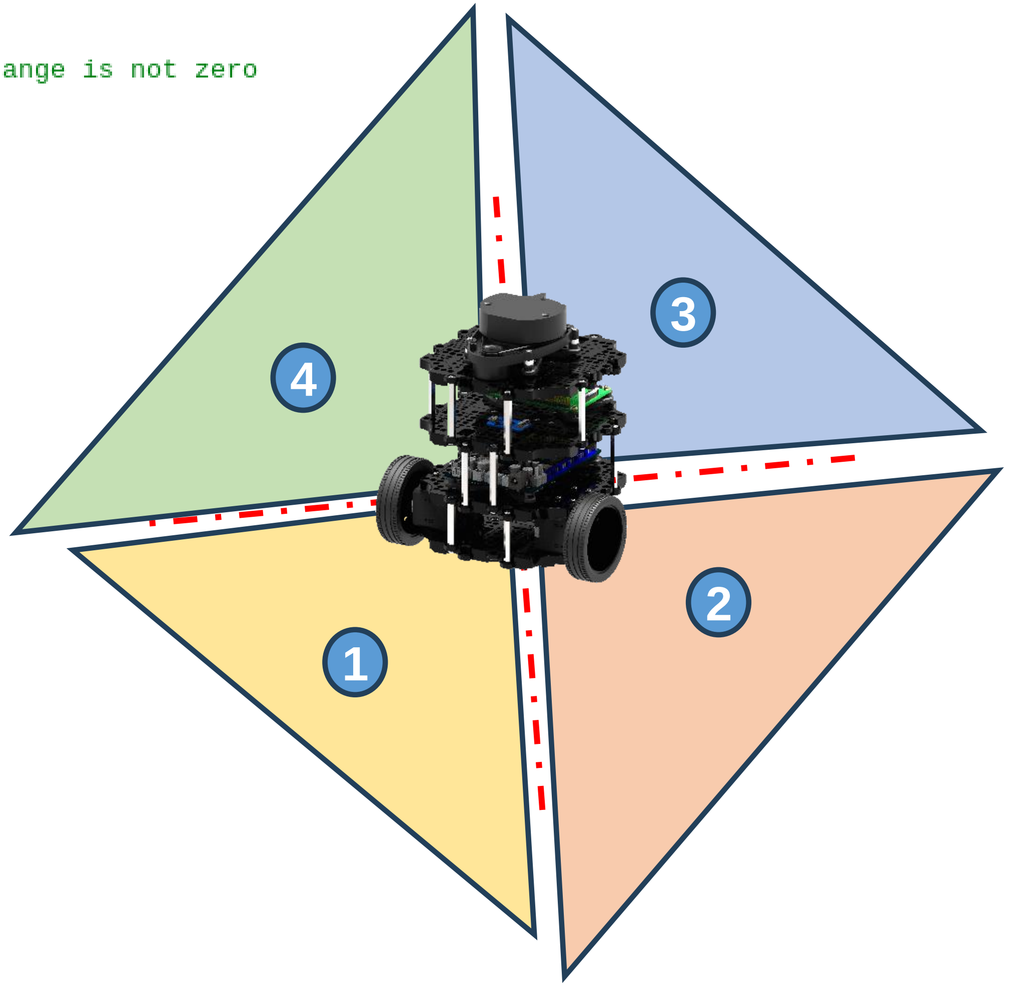


Real Example

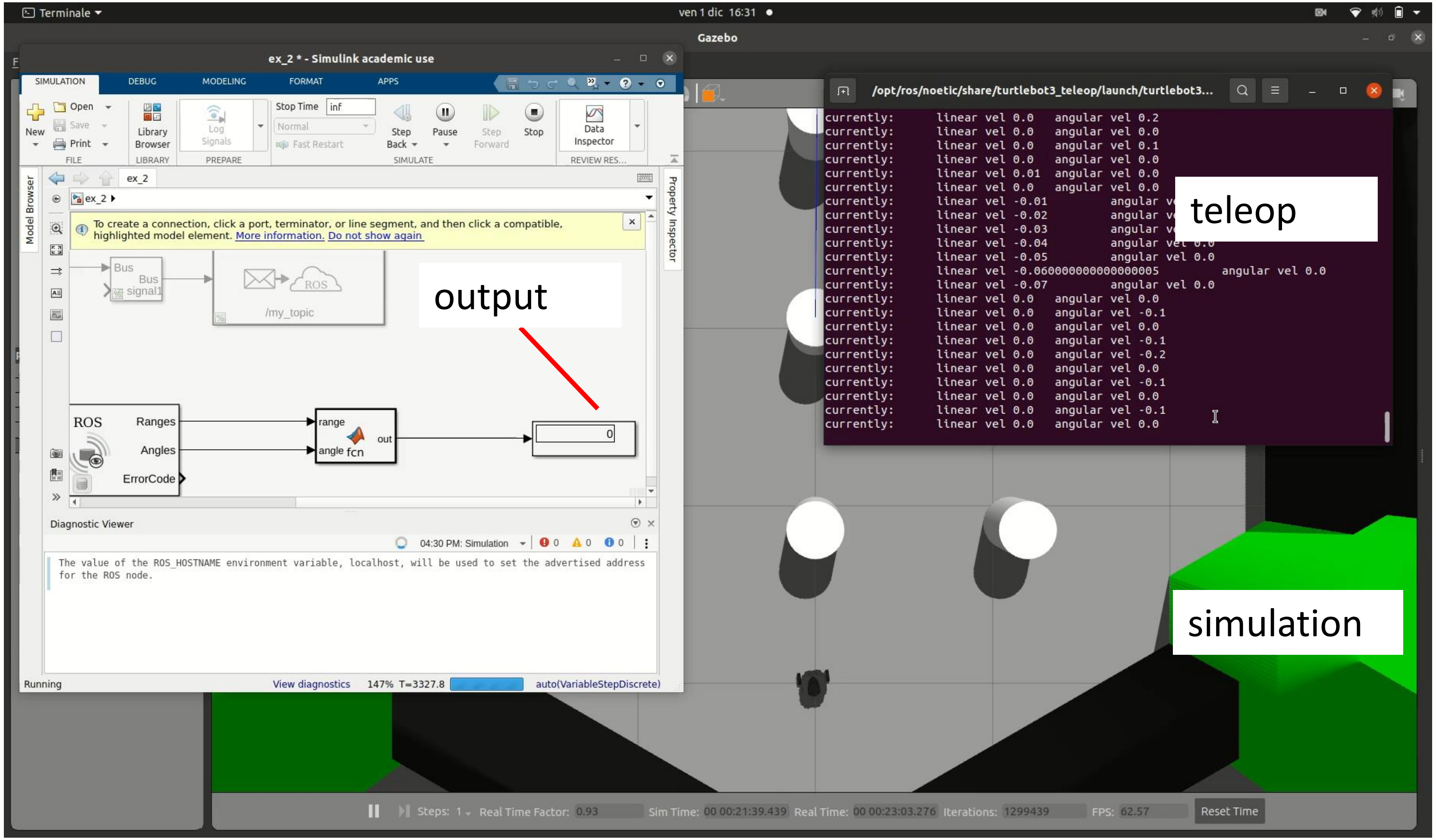




```
function [out,i] = fcn(range, angle)

out=-1;
endSim = any(range>0);% =1 if at least one range is not zero
i=-1;
if endSim
    [M,I] = min(range(range>0));
    if M < 0.4
        if I <= 30 || I >209 % front
            out= 1;
        end
        if I <= 80 && I >30 % left
            out= 2;
        end
        if I <= 160 && I >80 % back
            out= 3;
        end
        if I <= 209 && I >160 % right
            out= 4;
        end
        i=I;
    else
        out = 0;
    end
end
```



Real Example



To create a connection, click a port, terminator, or line segment, and then click a compatible, highlighted model element. More information. Do not show again

```
The value of the ROS_HOSTNAME environment variable, localhost, will be used to set the advertised address
for the ROS node.
```

/opt/ros/noetic/share/turtlebot3_teleop/launch/turtlebot3...

```
currently:      linear vel 0.0   angular vel 0.2
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.0   angular vel 0.1
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.01  angular vel 0.0
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel -0.01         angular ve
currently:      linear vel -0.02         angular ve
currently:      linear vel -0.03         angular ve
currently:      linear vel -0.04         angular vel 0.0
currently:      linear vel -0.05         angular vel 0.0
currently:      linear vel -0.060000000000000005         angular vel 0.0
currently:      linear vel -0.07         angular vel 0.0
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.0   angular vel -0.1
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.0   angular vel -0.1
currently:      linear vel 0.0   angular vel -0.2
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.0   angular vel -0.1
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.0   angular vel -0.1
currently:      linear vel 0.0   angular vel 0.0
```

Output:

- 0,1,2,3,4

Frequency:

- 10Hz (?)

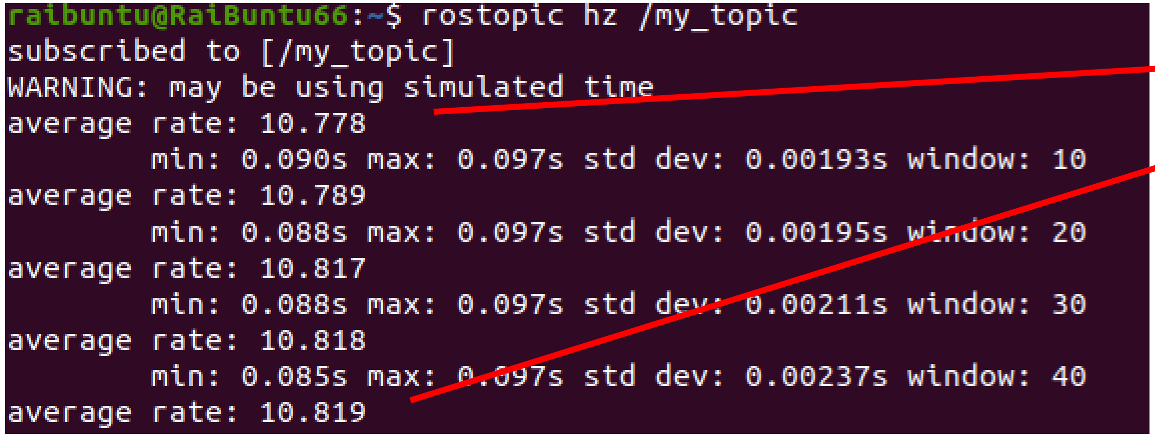
Real Example

Output:

- 0,1,2,3,4

Frequency:

- 10Hz (?)



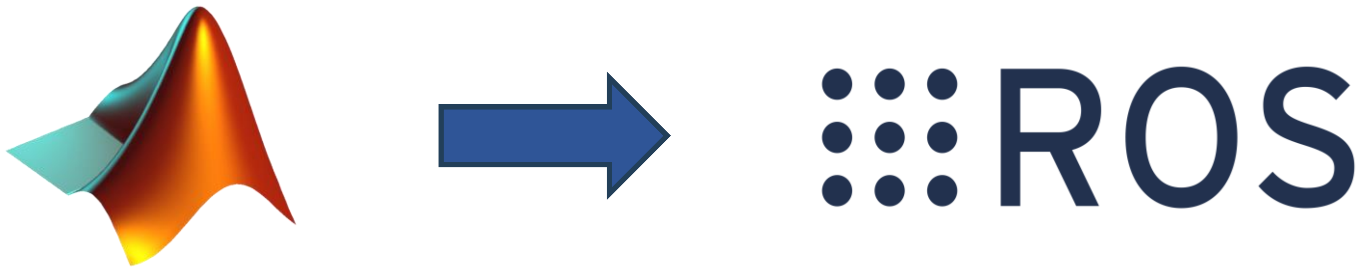
```
raibuntu@RaiBuntu66:~$ rostopic hz /my_topic
subscribed to [/my_topic]
WARNING: may be using simulated time
average rate: 10.778
        min: 0.090s max: 0.097s std dev: 0.00193s window: 10
average rate: 10.789
        min: 0.088s max: 0.097s std dev: 0.00195s window: 20
average rate: 10.817
        min: 0.088s max: 0.097s std dev: 0.00211s window: 30
average rate: 10.818
        min: 0.085s max: 0.097s std dev: 0.00237s window: 40
average rate: 10.819
```

Let's try to measure it:

Not stable and accurate!!

Real Example

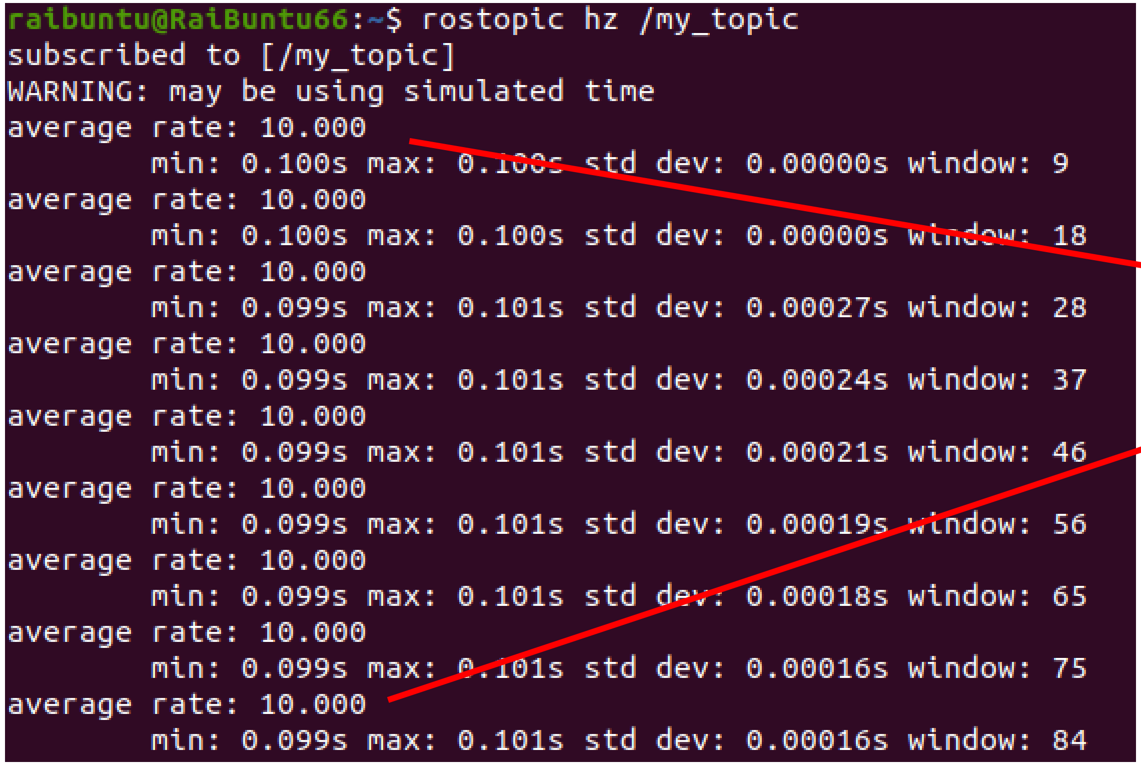
Let’s generate the node:



Let’s run the node:

```
$ rosrun ex_2 ex_2
```

Real Example



```
raibuntu@RaiBuntu66:~$ rostopic hz /my_topic
subscribed to [/my_topic]
WARNING: may be using simulated time
average rate: 10.000
        min: 0.100s max: 0.100s std dev: 0.00000s window: 9
average rate: 10.000
        min: 0.100s max: 0.100s std dev: 0.00000s window: 18
average rate: 10.000
        min: 0.099s max: 0.101s std dev: 0.00027s window: 28
average rate: 10.000
        min: 0.099s max: 0.101s std dev: 0.00024s window: 37
average rate: 10.000
        min: 0.099s max: 0.101s std dev: 0.00021s window: 46
average rate: 10.000
        min: 0.099s max: 0.101s std dev: 0.00019s window: 56
average rate: 10.000
        min: 0.099s max: 0.101s std dev: 0.00018s window: 65
average rate: 10.000
        min: 0.099s max: 0.101s std dev: 0.00016s window: 75
average rate: 10.000
        min: 0.099s max: 0.101s std dev: 0.00016s window: 84
```

Output:
- 0,1,2,3,4

Frequency:
- 10Hz (?)

Let's try to measure it:

stable and accurate!!

Real Example

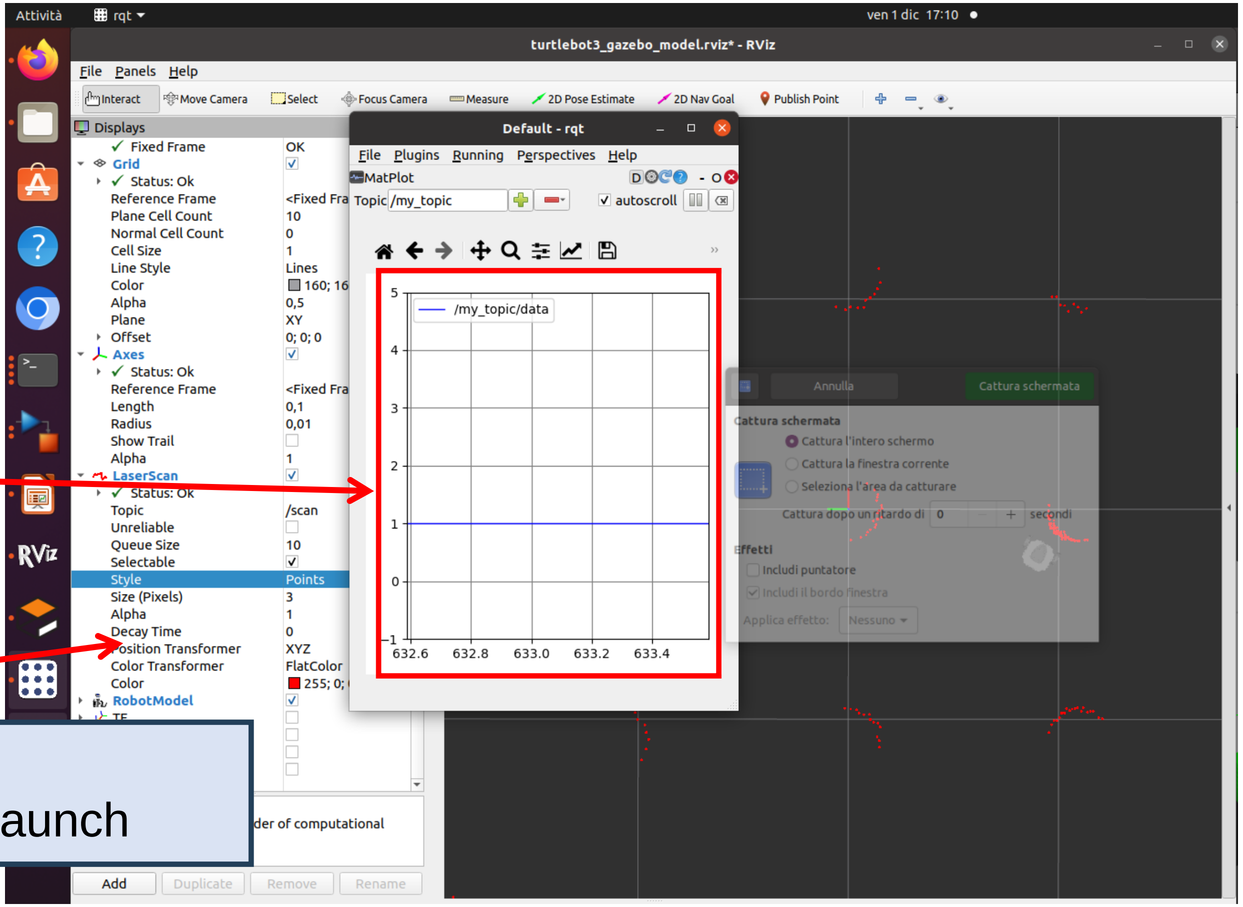
ROS Run visualization on PC

New terminal

```
$ rqt
```

New terminal

```
$ export TURTLEBOT3_MODEL=${TB3_MODEL}
$ roslaunch turtlebot3_gazebo turtlebot3_gazebo_rviz.launch
```



```
✓ Fixed Frame            OK
Grid                     ☑
  ✓ Status: Ok
  Reference Frame        <Fixed Fra
  Plane Cell Count       10
  Normal Cell Count      0
  Cell Size              1
  Line Style             Lines
  Color                  160; 16
  Alpha                  0,5
  Plane                  XY
  Offset                 0; 0; 0
Axes                     ☑
  ✓ Status: Ok
  Reference Frame        <Fixed Fra
  Length                 0,1
  Radius                 0,01
  Show Trail             □
  Alpha                  1
LaserScan                ☑
  ✓ Status: Ok
  Topic                  /scan
  Unreliable             □
  Queue Size             10
  Selectable             ☑
  Style                  Points
  Size (Pixels)          3
  Alpha                  1
  Decay Time             0
  Position Transformer   XYZ
  Color Transformer      FlatColor
  Color                  255; 0;
RobotModel               ☑
TF                       □
```

der of computational

Topic /my_topic

☑ autoscroll

/my_topic/data

Cattura schermata

● Cattura l'intero schermo

○ Cattura la finestra corrente

○ Seleziona l'area da catturare

Cattura dopo un ritardo di 0 secondi

Effetti

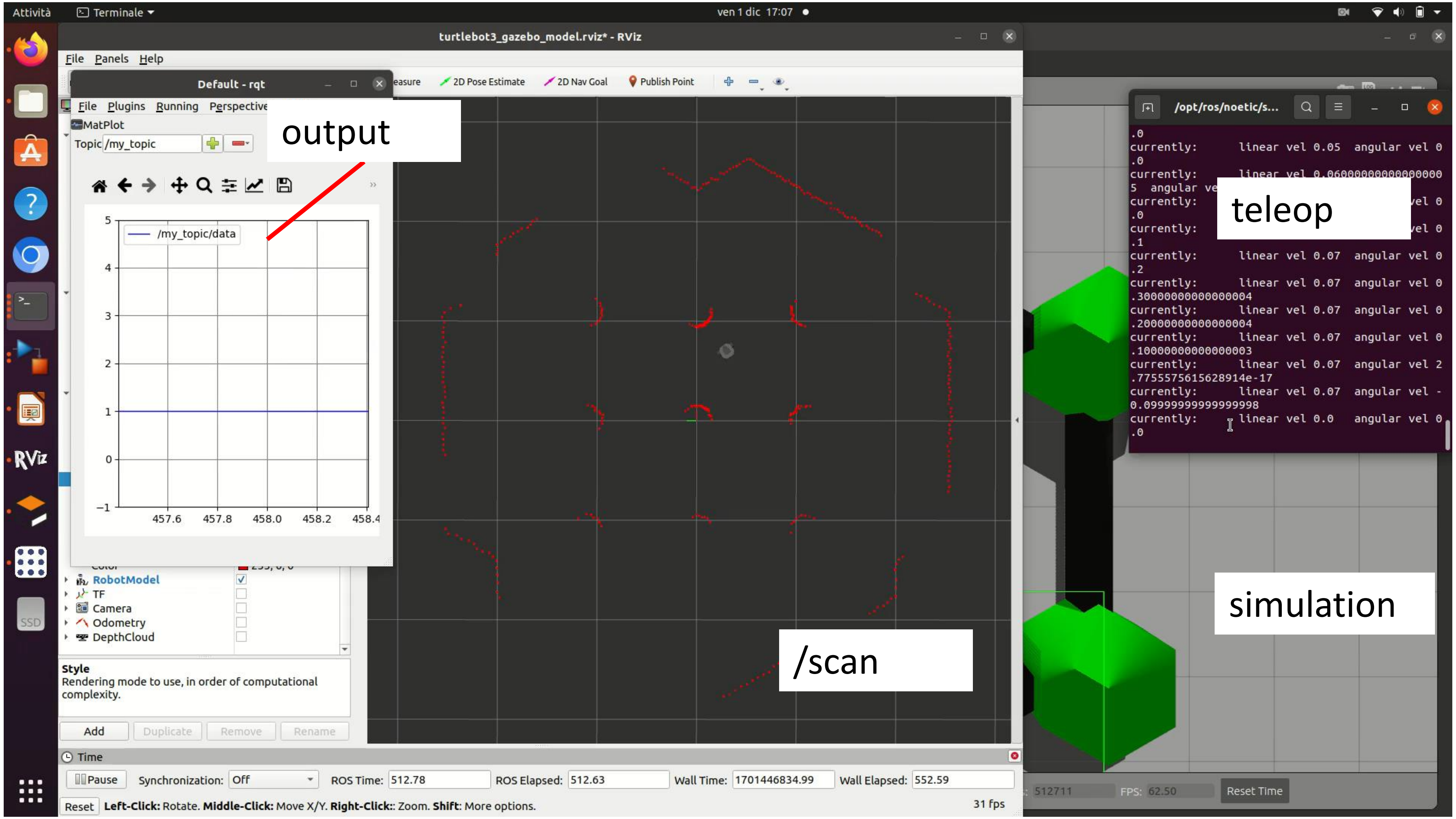
□ Includi puntatore

☑ Includi il bordo finestra

Applica effetto: Nessuno

Real Example

Results



Style
Rendering mode to use, in order of computational complexity.

☑ RobotModel

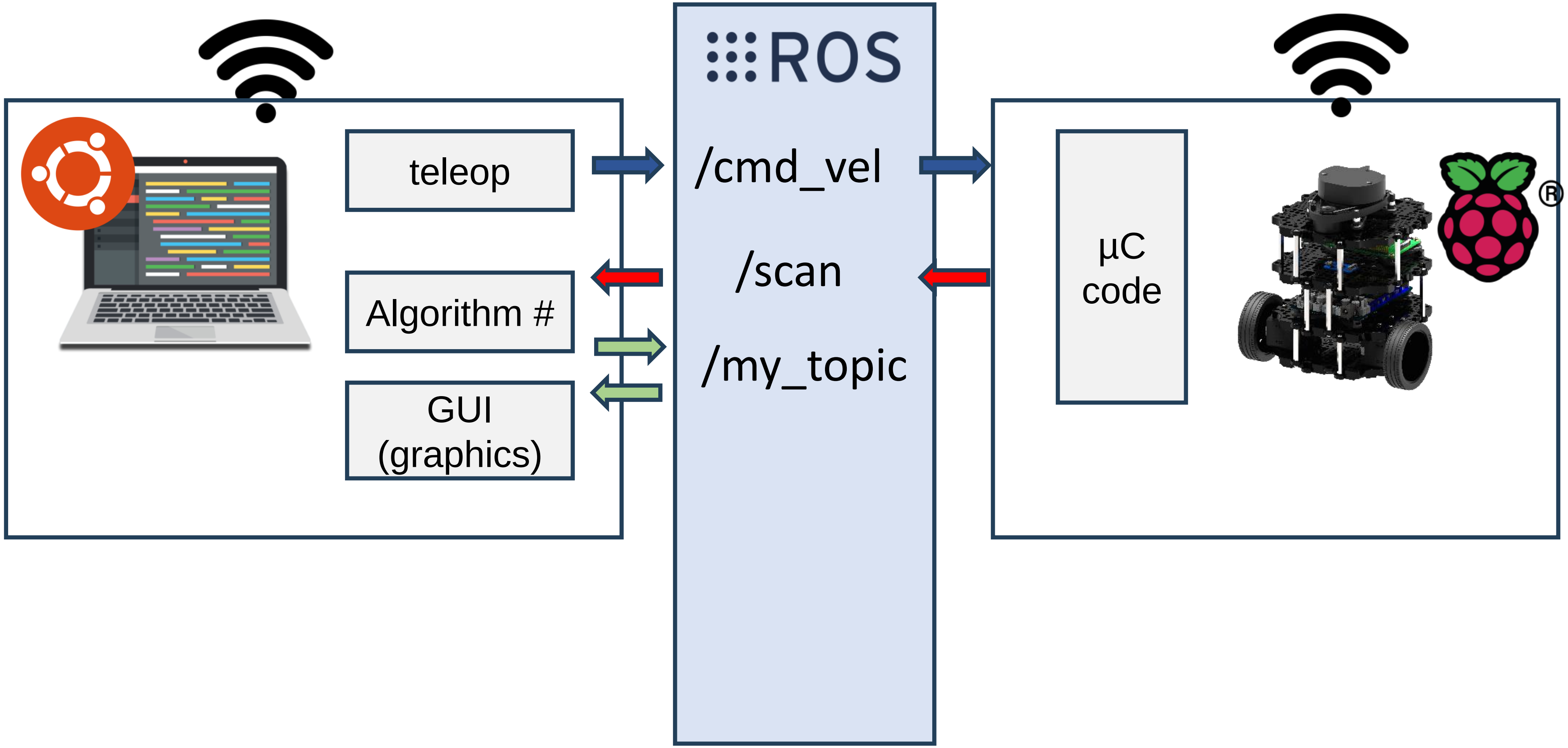
Synchronization: Off ROS Time: 512.78 ROS Elapsed: 512.63 Wall Time: 1701446834.99 Wall Elapsed: 552.59

FPS: 62.50

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Zoom. **Shift**: More options.

```
.0
currently:      linear vel 0.05  angular vel 0
.0
currently:      linear vel 0.060000000000000000
5  angular ve
currently:                                vel 0
.0
currently:                                vel 0
.1
currently:      linear vel 0.07  angular vel 0
.2
currently:      linear vel 0.07  angular vel 0
.30000000000000004
currently:      linear vel 0.07  angular vel 0
.20000000000000004
currently:      linear vel 0.07  angular vel 0
.10000000000000003
currently:      linear vel 0.07  angular vel 2
.7755575615628914e-17
currently:      linear vel 0.07  angular vel -
0.09999999999999998
currently:      linear vel 0.0   angular vel 0
.0
```

Let’s test it!

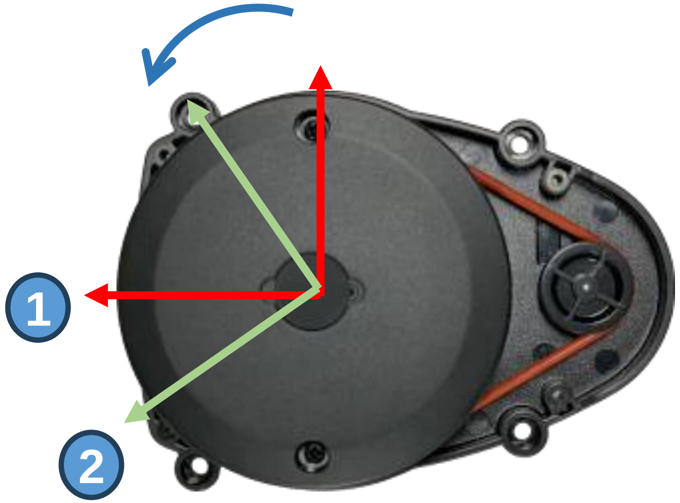


teleop
Algorithm #
GUI (graphics)
ROS
/cmd_vel
/scan
/my_topic
µC code

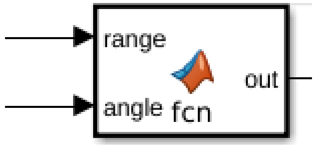
Let’s test it!

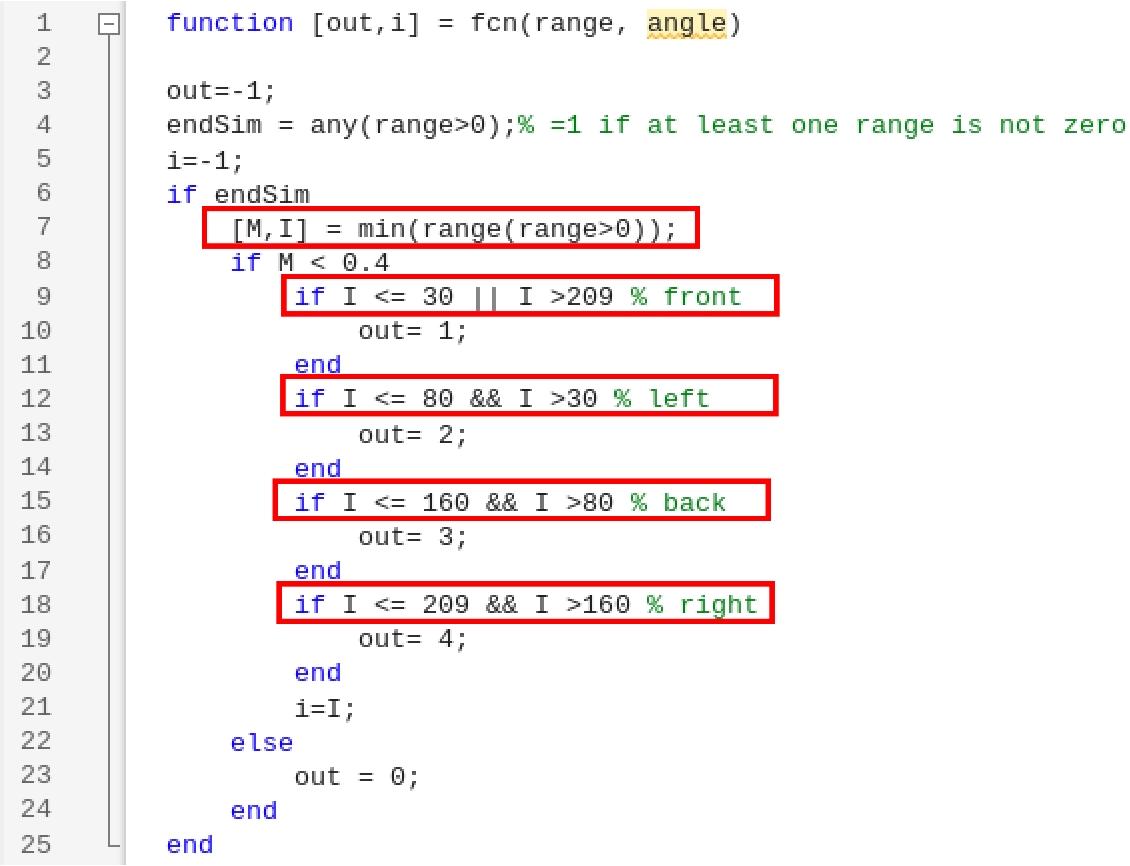
When using a real system:

- ✓ Misalignment of axis
- ✓ False positive if nothing detected (origin)



Let’s test it!





```
function [out,i] = fcn(range, angle)

out=-1;
endSim = any(range>0);% =1 if at least one range is not zero
i=-1;
if endSim
    [M,I] = min(range(range>0));
    if M < 0.4
        if I <= 30 || I >209 % front
            out= 1;
        end
        if I <= 80 && I >30 % left
            out= 2;
        end
        if I <= 160 && I >80 % back
            out= 3;
        end
        if I <= 209 && I >160 % right
            out= 4;
        end
        i=I;
    else
        out = 0;
    end
end
```

That’s it for today…

See you next time!

POLITECNICO
MILANO 1863

S. Arrigoni