



PROGRAMMING FOR AUTONOMOUS SYSTEMS

Fred Livingston, PhD
Work Shop 001

CONTACT INFO

Fred Livingston, PhD

Email: fjliving@ncsu.edu

Mobile: 919.795.4710

Web: <https://livingston.wordpress.ncsu.edu/>

Bitbucket: [https://bitbucket.org/livingston ai/](https://bitbucket.org/livingston_ai/)

SPRING 2023 WORKSHOP SERIES

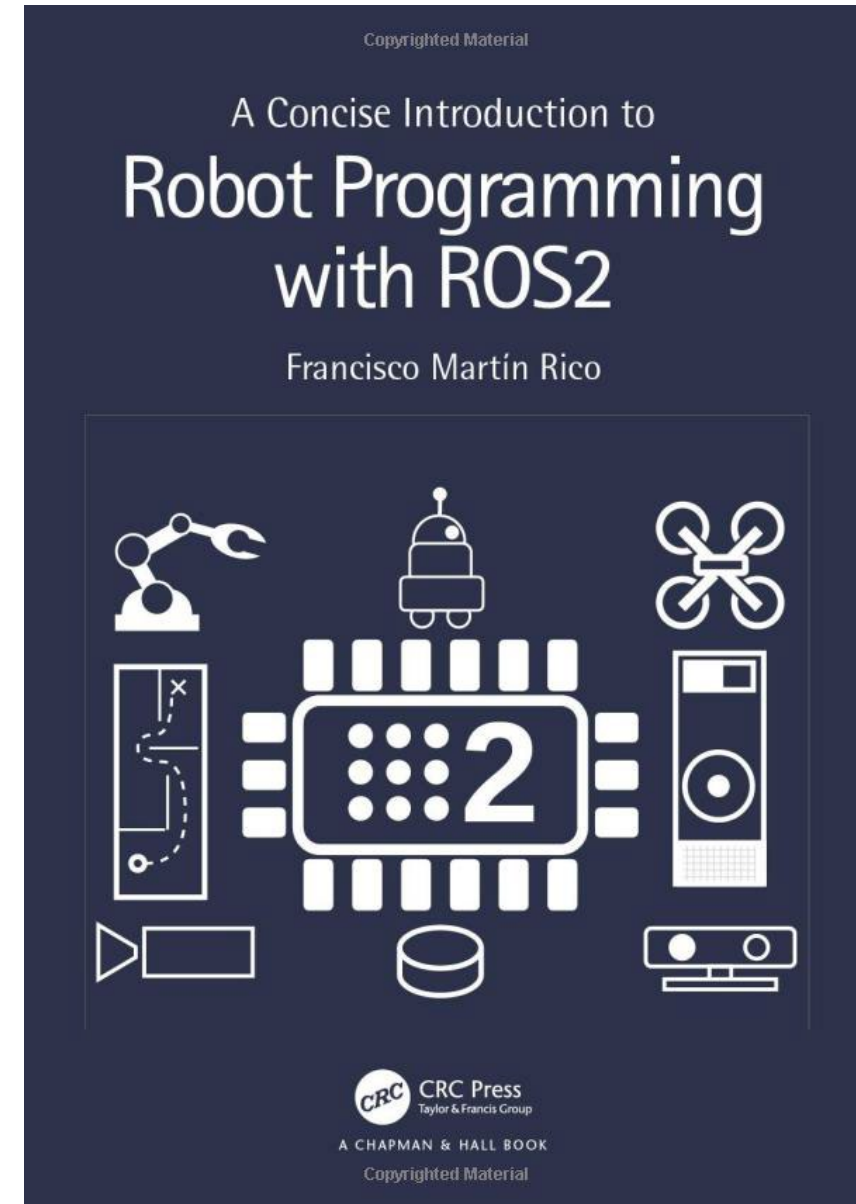
- WS 001 – Introduction to Robot Programming using ROS2 [Feb 17th, 2023]
- WS 002 – Navigation [March 3rd, 2023]
- WS 003 – Reactive Behaviors [TBD]

ROBOT PROGRAMMING USING ROS2

- Introduction to ROS2
- ROS Client Layer
- Simulation Tools
- robot program: architectures
- Task1: Basic control of ground vehicle

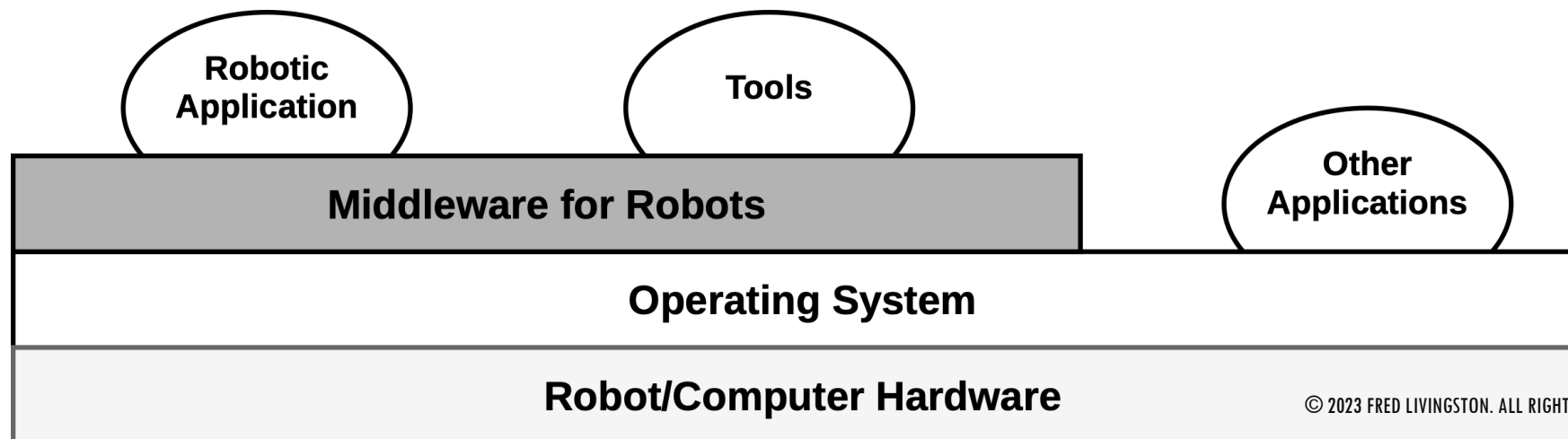
RESOURCES

- ROS/2 Tutorials
 - <http://docs.ros.org/en/rolling/Tutorials.html>
 - <https://roboticsbackend.com/category/ros2/>
 - https://www.theconstructsim.com/robotigniteacademy_learnros/ros-courses-library/

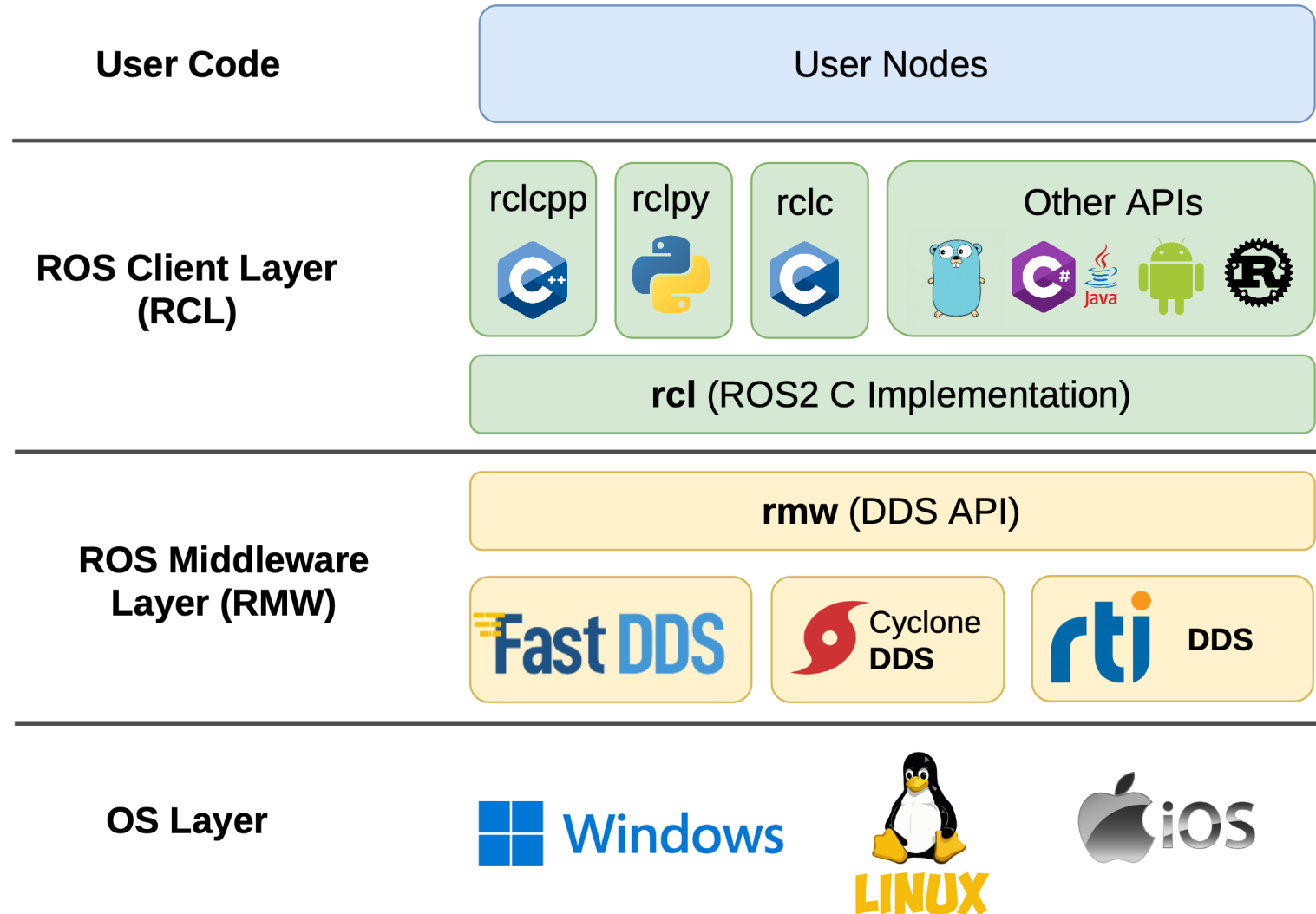


PROGRAMMING ROBOTS

- Robots must be programmed to be useful
- We need Middlewares
- Robot programming middlewares provide drivers, libraries, and methodologies
- Few of them have survived the robot for which they were designed or have expanded from the laboratories where they were implemented
- The big difference is the ROS developers community around the world.



ROS2 DESIGN



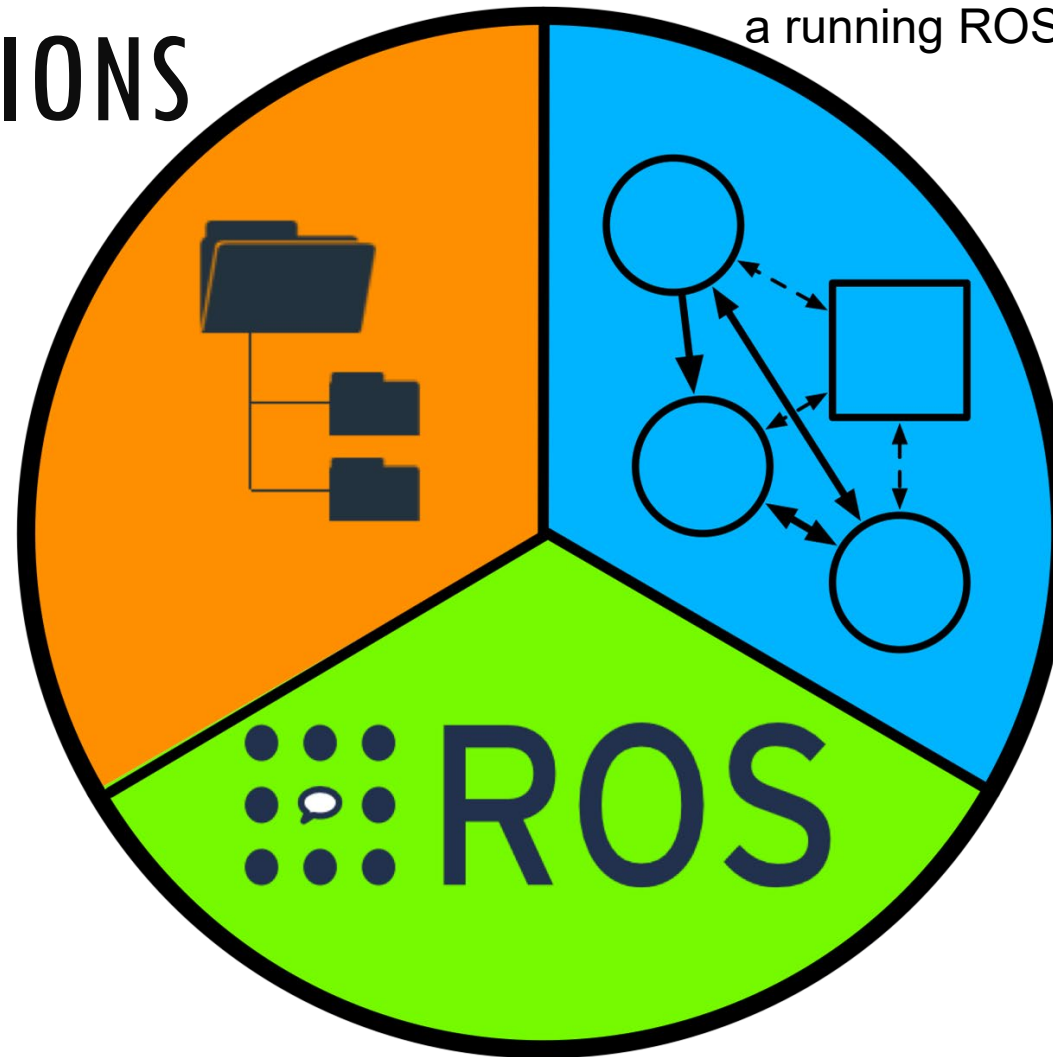
Computation Graph:

a running ROS2 application

ROS DIMENSIONS

Workspace:

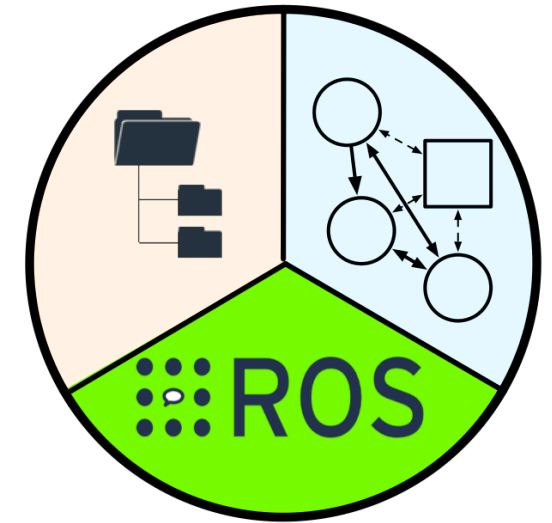
the set of software installed on the robot or computer, the programs that the user develops, and tools to build



Community: vast community of developers who contribute with their own applications and utilities through public repositories, to which other developers can contribute

THE COMMUNITY

- Open Source and Licenses
- ROS2 organizes software development in federal model
- Packages and distributions
- Online resources

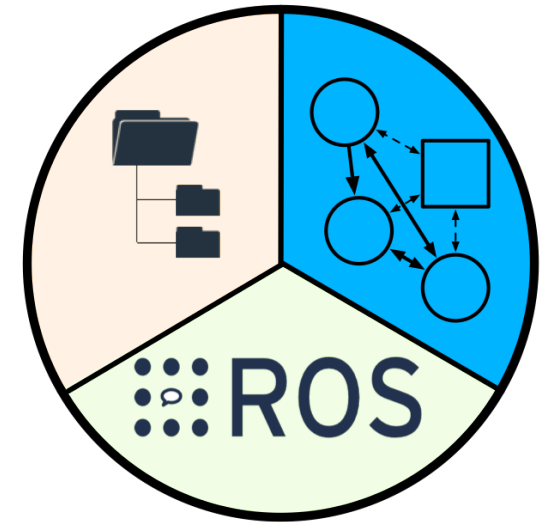


MIT LICENSE GNU LICENSE BSD-2
OPEN SOURCE LICENSE
APACHE LICENSE 2.0 BSD-3



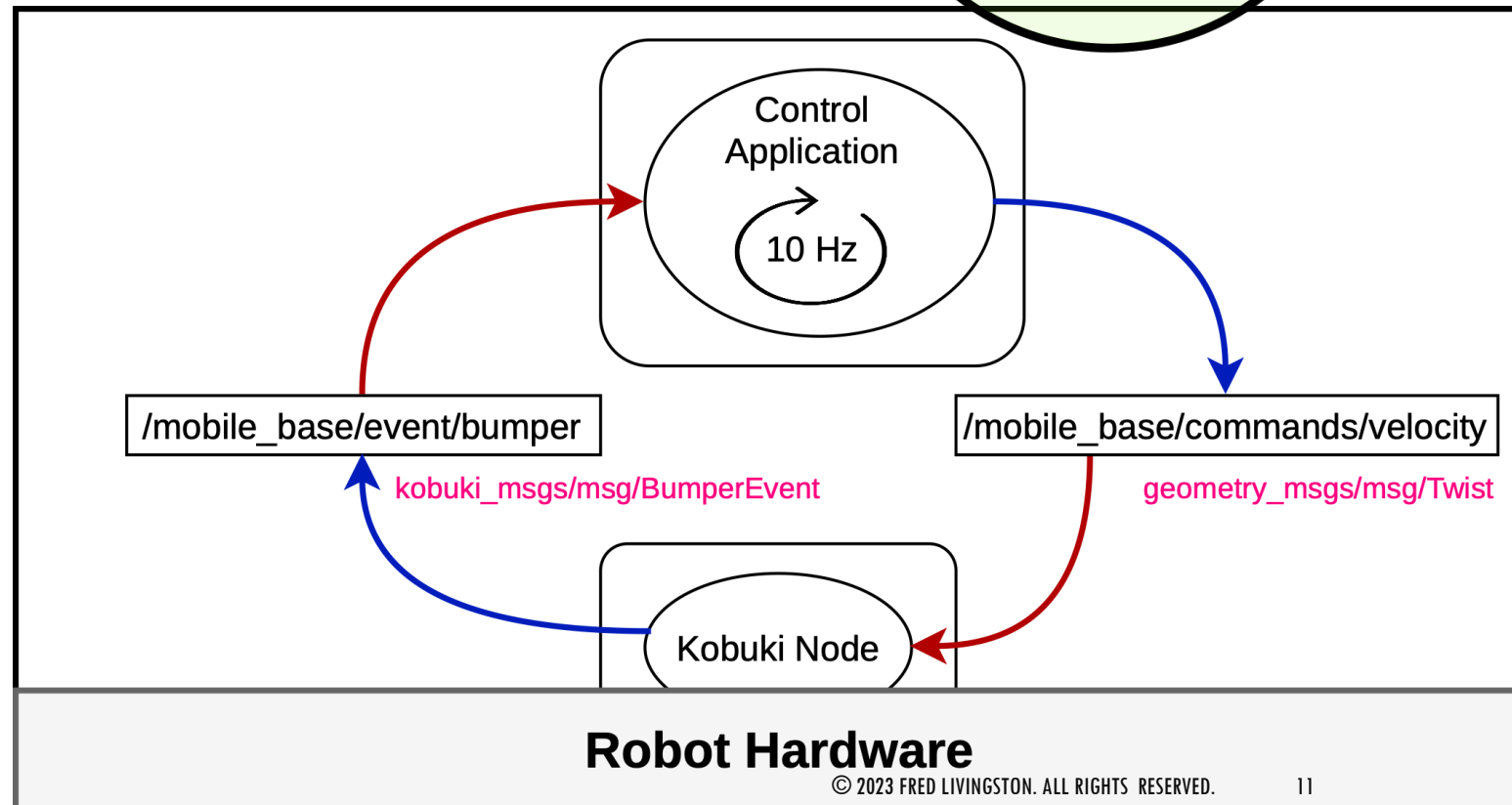
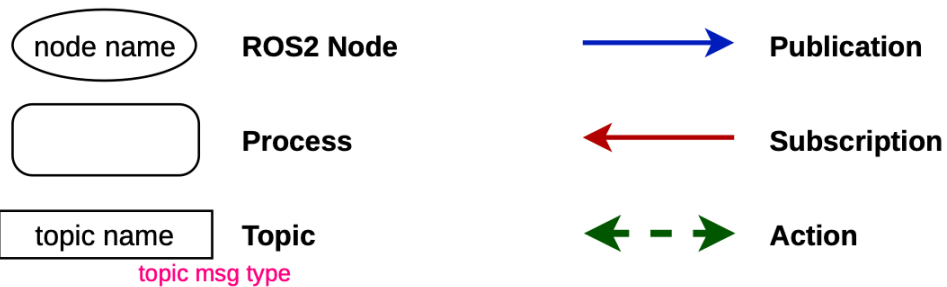
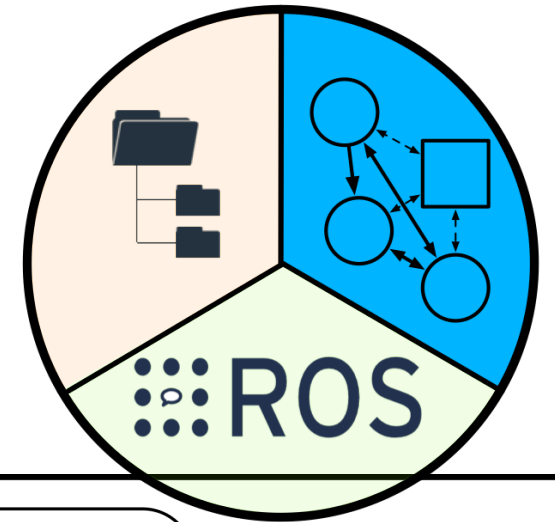
THE COMPUTATION GRAPH

- A robot's software looks like during its execution.
- A Computation Graph contains ROS2 nodes that communicate with each other so that the robot can carry out some tasks.
- The logic of the application is in the nodes, as the primary elements of execution in ROS2.
- Communication mechanisms:
 - **Publication/Subscription:** Asynchronous N:M
 - **Services:** Synchronous 1:1
 - **Actions:** Asynchronous 1:1

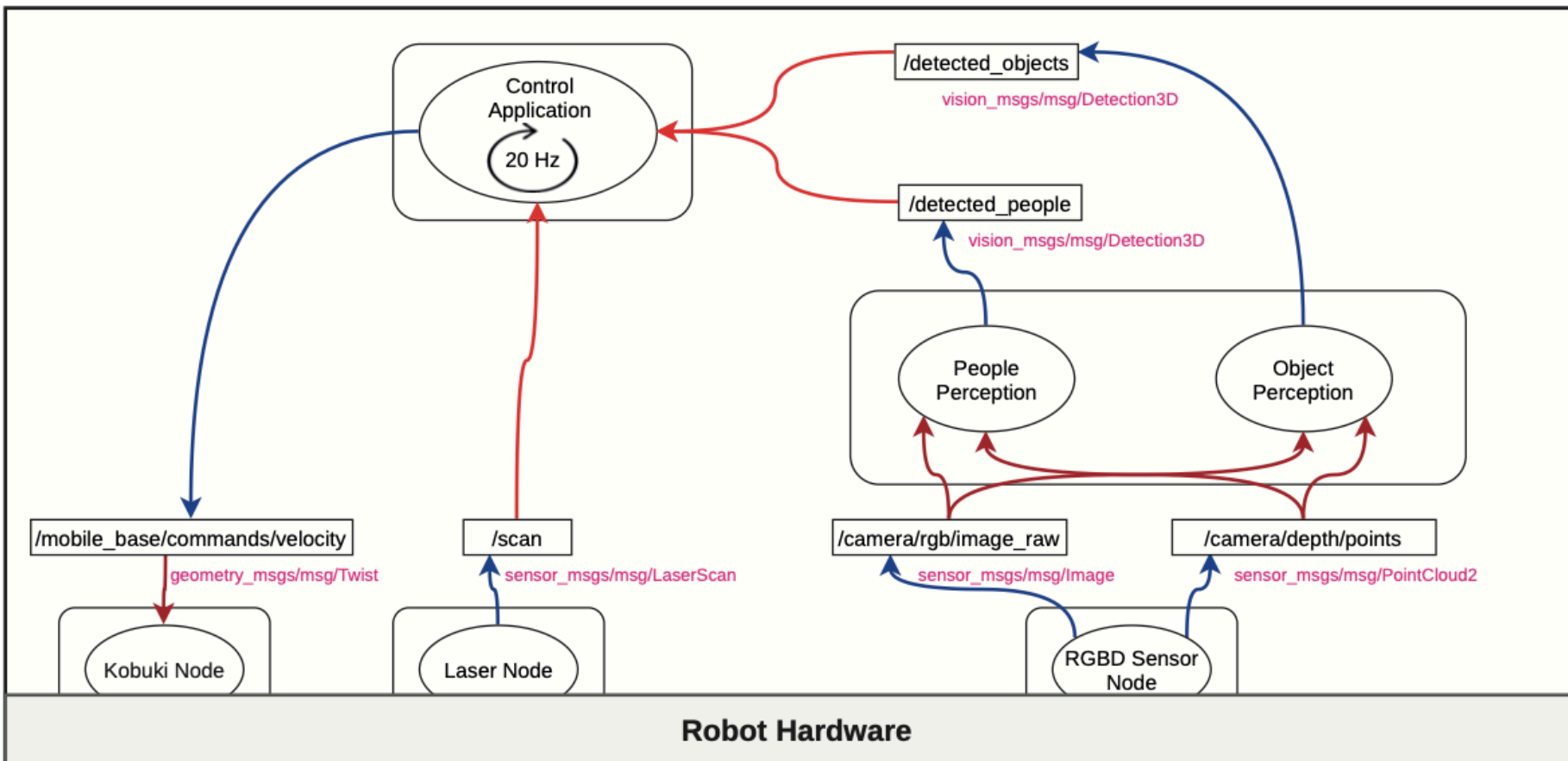
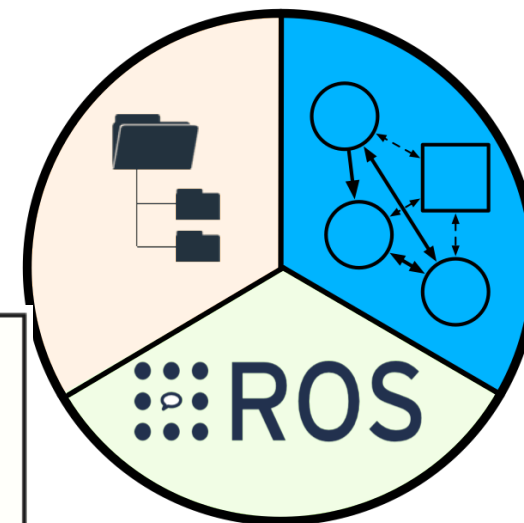


THE COMPUTATION GRAPH

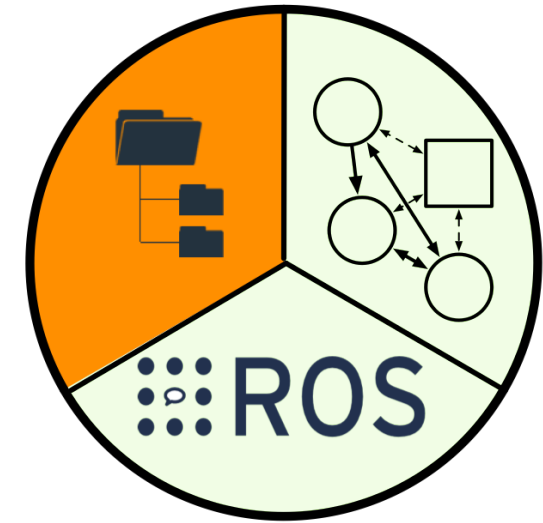
- Execution model
 - ↳ Iterative
 - ↳ Event-Oriented



THE COMPUTATION GRAPH - EXAMPLE



THE WORKSPACE



- Approaches ROS2 software from a static point of view.
- Where the ROS2 software is installed, organized, and all the tools and processes that allow us to launch a computing graph.
- This includes the build system and node startup tools.
- Elements:
 - **Package:**
 - It is the minimum functional set of software.
 - Contains executables, libraries, or message definitions with a common purpose.
 - **Workspace:**
 - A directory that contains packages.
 - Activable to be available to use.
- ***Underlay y overlay***

ROS2::FOXY INSTALATION

- ❑ VirtualBox (Option 1)

- ❑ <https://www.virtualbox.org/wiki/Downloads>

- ❑ VMWare (Option 2 – Preferred)

- ❑ <https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html>

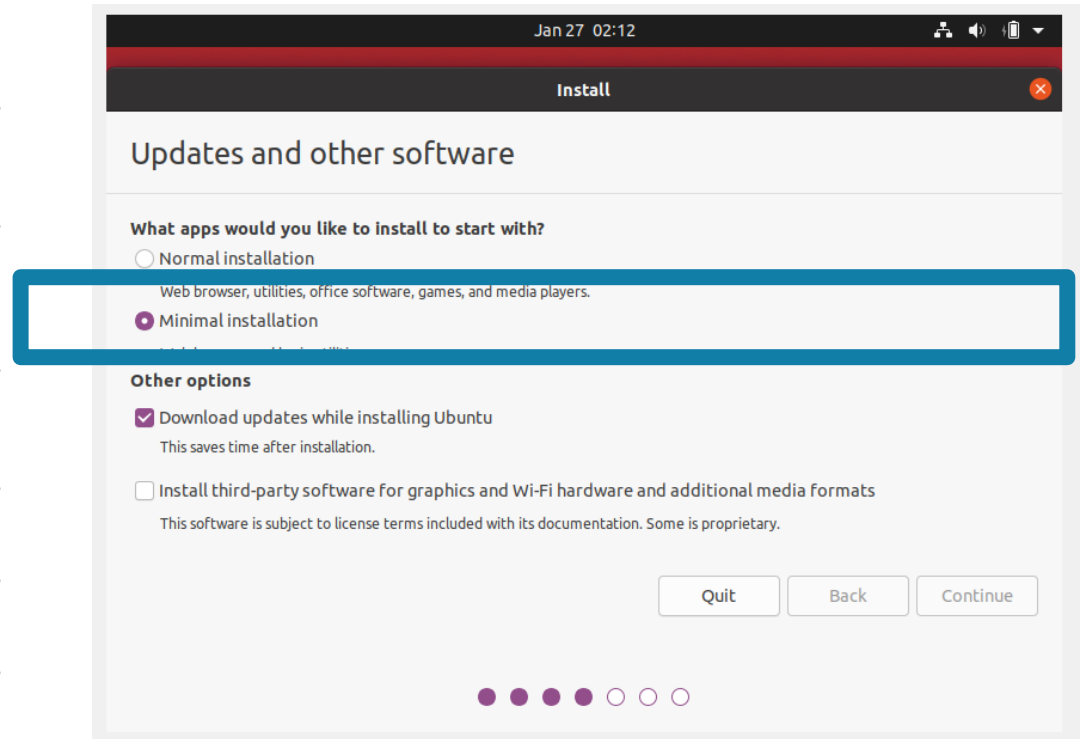
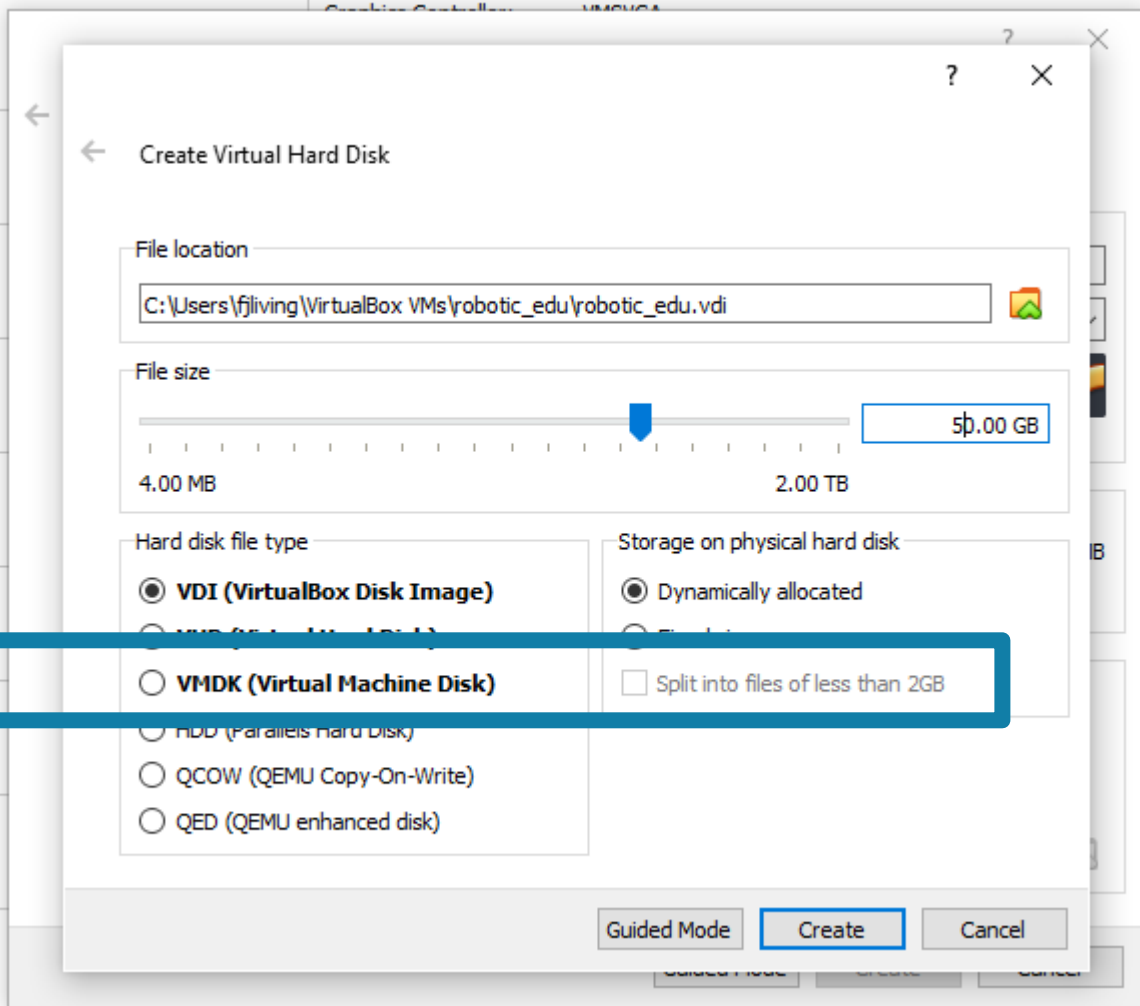
- ❑ Ubuntu **20.04**

- ❑ <https://releases.ubuntu.com/22.04/ubuntu-22.04.1-desktop-amd64.iso>

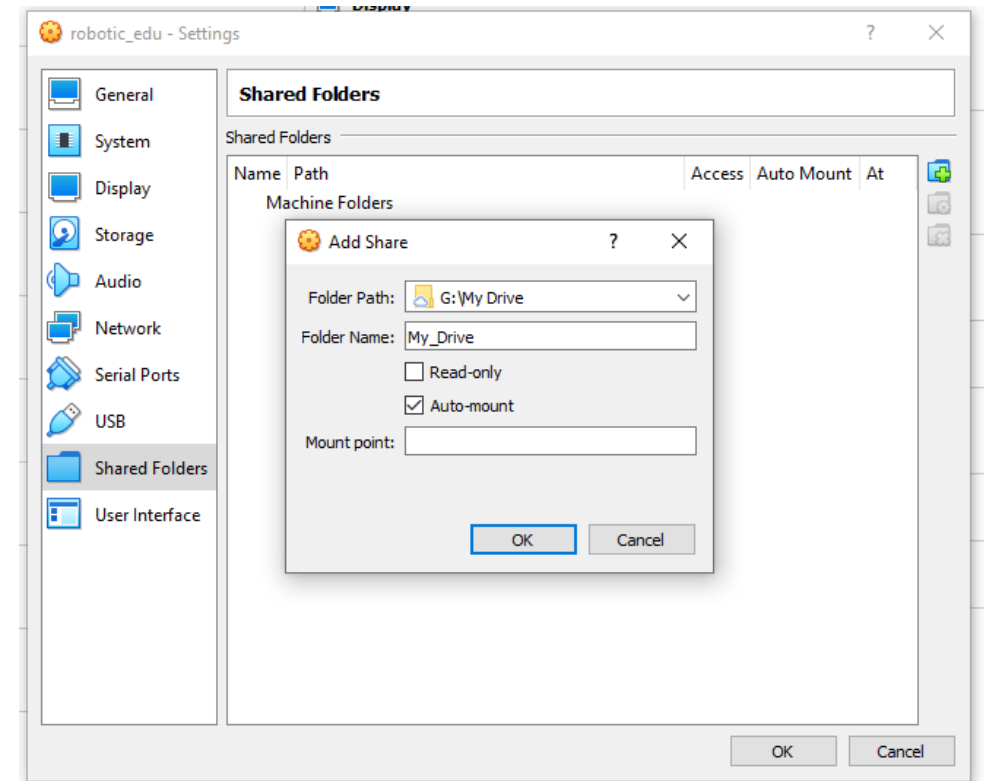
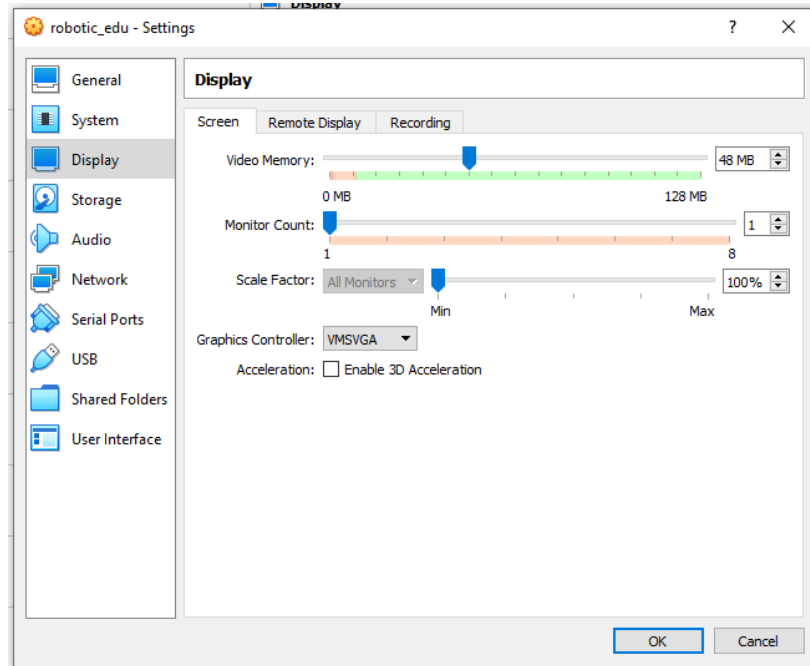
- ❑ ROS2::Foxy (LTS)

- ❑ <https://docs.ros.org/en/foxy/Installation/Ubuntu-Install-Debians.html>

VM – RECOMMENDED SETTINGS



VM — RECOMMENDED SETTINGS



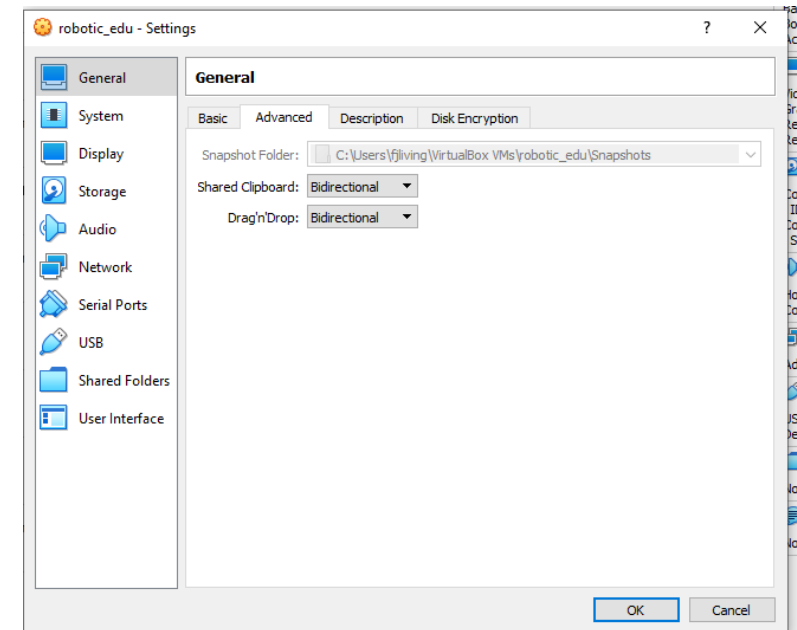
☐ Make sure 3D Acceleration is Disabled

VM – RECOMMENDED SETTINGS



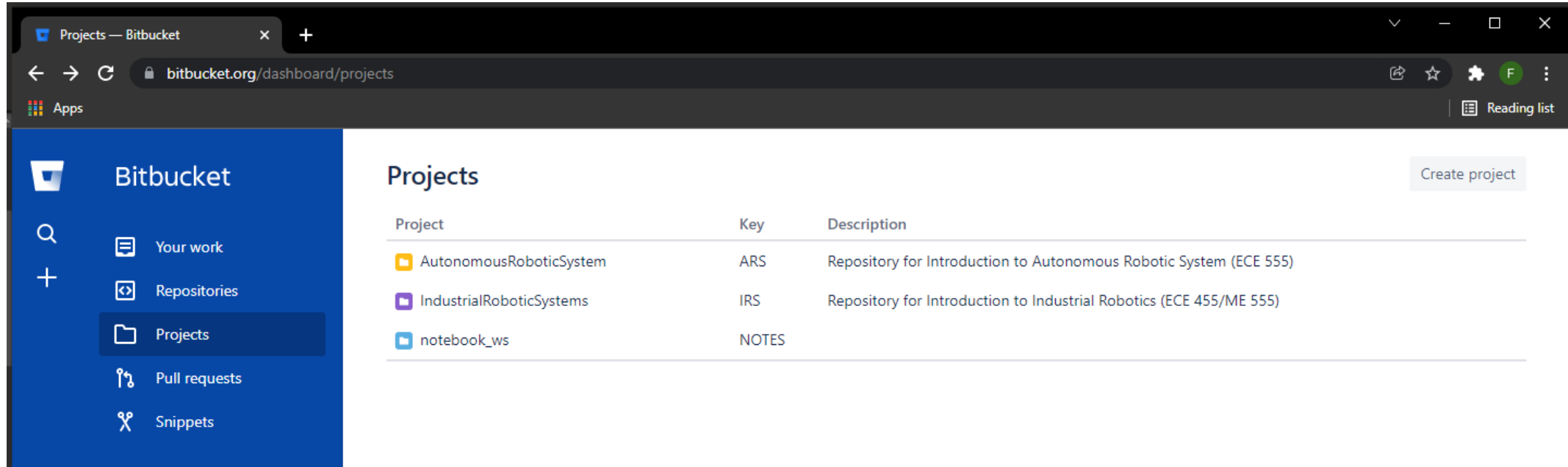
```
VirtualBox Guest Additions installation
Verifying archive integrity... All good.
Uncompressing VirtualBox 6.1.32 Guest Additions for Linux.....
VirtualBox Guest Additions installer
Copying additional installer modules ...
Installing additional modules ...
VirtualBox Guest Additions: Starting.
VirtualBox Guest Additions: Building the VirtualBox Guest Additions kernel
modules. This may take a while.
VirtualBox Guest Additions: To build modules for other installed kernels, run
VirtualBox Guest Additions: /sbin/rcvboxadd quicksetup <version>
VirtualBox Guest Additions: or
VirtualBox Guest Additions: /sbin/rcvboxadd quicksetup all
VirtualBox Guest Additions: Building the modules for kernel 5.13.0-27-generic.

This system is currently not set up to build kernel modules.
Please install the gcc make perl packages from your distribution.
VirtualBox Guest Additions: Running kernel modules will not be replaced until
the system is restarted
Press Return to close this window...
```



Install Guest Additions

BITBUCKET



The screenshot shows the Bitbucket dashboard at bitbucket.org/dashboard/projects. The left sidebar contains navigation options: Your work, Repositories, Projects (selected), Pull requests, and Snippets. The main content area displays a table of projects with columns for Project, Key, and Description. A 'Create project' button is visible in the top right corner of the main area.

Project	Key	Description
AutonomousRoboticSystem	ARS	Repository for Introduction to Autonomous Robotic System (ECE 555)
IndustrialRoboticSystems	IRS	Repository for Introduction to Industrial Robotics (ECE 455/ME 555)
notebook_ws	NOTES	

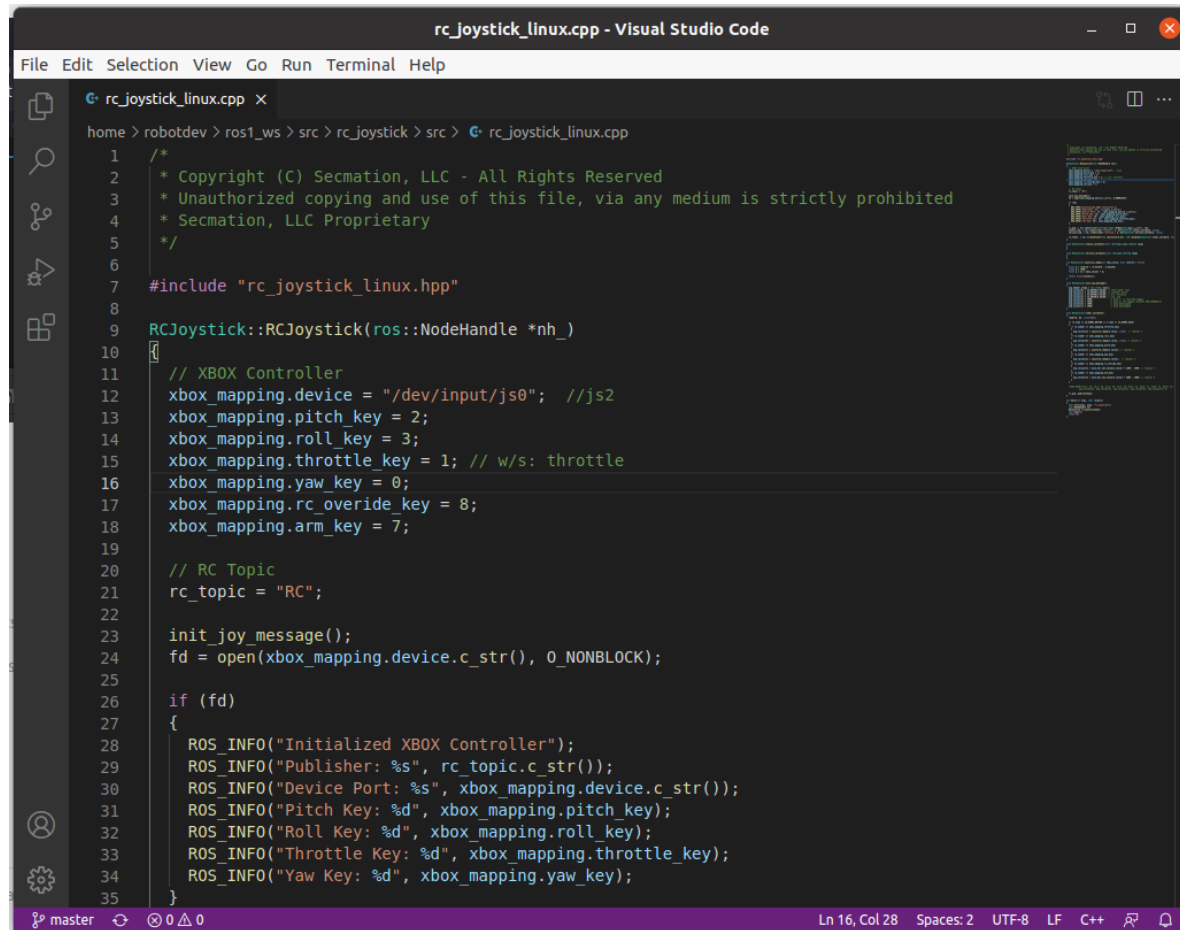
https://bitbucket.org/livingston_ai/

PYTHON 2 UNINSTALL (OPTIONAL)

- ❑ <https://askubuntu.com/questions/1242702/how-to-remove-python-2-from-ubuntu-20-04>
- ❑ `$sudo apt-get install python-is-python3`
- ❑ `$sudo apt-get autoremove --purge`

VSCODE (OPTIONAL)

❑ \$sudo snap install code --classic



```
rc_joystick_linux.cpp - Visual Studio Code
File Edit Selection View Go Run Terminal Help
rc_joystick_linux.cpp x
home > robotdev > ros1_ws > src > rc_joystick > src > rc_joystick_linux.cpp
1  /*
2  * Copyright (C) Secmation, LLC - All Rights Reserved
3  * Unauthorized copying and use of this file, via any medium is strictly prohibited
4  * Secmation, LLC Proprietary
5  */
6
7  #include "rc_joystick_linux.hpp"
8
9  RCJoystick::RCJoystick(ros::NodeHandle *nh_)
10
11  // XBOX Controller
12  xbox_mapping.device = "/dev/input/js0"; //js2
13  xbox_mapping.pitch_key = 2;
14  xbox_mapping.roll_key = 3;
15  xbox_mapping.throttle_key = 1; // w/s: throttle
16  xbox_mapping.yaw_key = 0;
17  xbox_mapping.rc_override_key = 8;
18  xbox_mapping.arm_key = 7;
19
20  // RC Topic
21  rc_topic = "RC";
22
23  init_joy_message();
24  fd = open(xbox_mapping.device.c_str(), 0_NONBLOCK);
25
26  if (fd)
27  {
28      ROS_INFO("Initialized XBOX Controller");
29      ROS_INFO("Publisher: %s", rc_topic.c_str());
30      ROS_INFO("Device Port: %s", xbox_mapping.device.c_str());
31      ROS_INFO("Pitch Key: %d", xbox_mapping.pitch_key);
32      ROS_INFO("Roll Key: %d", xbox_mapping.roll_key);
33      ROS_INFO("Throttle Key: %d", xbox_mapping.throttle_key);
34      ROS_INFO("Yaw Key: %d", xbox_mapping.yaw_key);
35  }
```

ROS2::FOXY INSTALLATION

□ https://bitbucket.org/livingston_ai/ros2_robot_programming/src/master/ros2_foxy_setup.ipynb?viewer=nbviewer

ROS 2 Cheats Sheet

Command Line Interface

All ROS 2 CLI tools start with the prefix 'ros2' followed by a command, a verb and (possibly) positional/optional arguments.

For any tool, the documentation is accessible with,

```
$ ros2 command --help
```

and similarly for verb documentation,

```
$ ros2 command verb -h
```

Similarly, auto-completion is available for all commands/verbs and most positional/optional arguments.

E.g.,

```
$ ros2 command [tab][tab]
```

Some of the examples below rely on:

[ROS 2 demos package](#).

action Allows to manually send a goal and displays debugging information about actions.

Verbs:

info Output information about an action.

list Output a list of action names.

send_goal Send an action goal.

show Output the action definition.

Examples:

```
$ ros2 action info /fibonacci
```

```
$ ros2 action list
```

```
$ ros2 action send_goal /fibonacci \  
action_tutorials/action/Fibonacci "order: 5"
```

```
$ ros2 action show action_tutorials/action/Fibonacci
```

bag Allows to record/play topics to/from a rosbag.

Verbs:

info Output information of a bag.

play Play a bag.

record Record a bag.

list Output a list of running containers and components.

load Load a component into a container node.

standalone Run a component into its own standalone container node.

types Output a list of components registered in the ament index.

unload Unload a component from a container node.

Examples:

```
$ ros2 component list
```

```
$ ros2 component load /ComponentManager \  
composition composition::Talker
```

```
$ ros2 component types
```

```
$ ros2 component unload /ComponentManager 1
```

daemon Various daemon related verbs.

Verbs:

start Start the daemon if it isn't running.

status Output the status of the daemon.

stop Stop the daemon if it is running

doctor A tool to check ROS setup and other potential issues such as network, package versions, rmw middleware etc.

Alias: **wtf** (where's the fire).

Arguments:

--report/-r Output report of all checks.

--report-fail/-rf Output report of failed checks only.

--include-warning/-iw Include warnings as failed checks.

Examples:

```
$ ros2 doctor
```

```
$ ros2 doctor --report
```

```
$ ros2 doctor --report-fail
```

```
$ ros2 doctor --include-warning
```

interface Various related verbs. Intended for the following options: 'ros2 interface list', 'ros2 interface package', 'ros2 interface packages', 'ros2 interface proto', 'ros2 interface show'.

Verbs:

list List

package Output

packages Output

proto Print

show Output

Examples:

```
$ ros2 interface
```

```
$ ros2 interface
```

```
$ ros2 interface
```

```
$ ros2 interface
```

```
$ ros2 interface
```

launch Allows to launch without to 'cd' the package.

Usage:

```
$ ros2 launch <
```

Example:

```
$ ros2 launch de
```

lifecycle Various

Verbs:

get Get li

list Outp

nodes Outp

set Trigg

msg (deprecated)

messages.

Verbs:

```

$ ros2 msg list
$ ros2 msg package std_msgs
$ ros2 msg packages
$ ros2 msg show geometry_msgs/msg/Pose

```

multicast Various multicast related verbs.

Verbs:

- `receive` Receive a single UDP multicast packet.
- `send` Send a single UDP multicast packet.

node Displays debugging information about nodes.

Verbs:

- `info` Output information about a node.
- `list` Output a list of available nodes.

Examples:

```

$ ros2 node info /talker
$ ros2 node list

```

param Allows to manipulate parameters.

Verbs:

- `delete` Delete parameter.
- `describe` Show descriptive information about declared parameters.
- `dump` Dump the parameters of a given node in yaml format, either in terminal or in a file.
- `get` Get parameter.
- `list` Output a list of available parameters.
- `set` Set parameter

Examples:

```

$ ros2 param delete /talker /use_sim_time
$ ros2 param get /talker /use_sim_time
$ ros2 param list
$ ros2 param set /talker /use_sim_time false

```

pkg Create a ros2 package or output package(s)-related information.

Verbs:

- `create` Create a new ROS2 package.

```

$ ros2 pkg executables demo_nodes_cpp
$ ros2 pkg list
$ ros2 pkg prefix std_msgs
$ ros2 pkg xml -t version

```

run Allows to run an executable in an arbitrary package without having to 'cd' there first.

Usage:

```

$ ros2 run <package> <executable>

```

Example:

```

$ ros2 run demo_node_cpp talker

```

security Various security related verbs.

Verbs:

- `create_key` Create key.
- `create_permission` Create keystore.
- `generate_artifacts` Create permission.
- `list_keys` Distribute key.
- `create_keystore` Generate keys and permission files from a list of identities and policy files.
- `distribute_key` Generate XML policy file from ROS graph data.
- `generate_policy` List keys.

Examples (see [sros2 package](#)):

```

$ ros2 security create_key demo_keys /talker
$ ros2 security create_permission demo_keys /talker \
  policies/sample_policy.xml
$ ros2 security generate_artifacts
$ ros2 security create_keystore demo_keys

```

service Allows to manually call a service and displays debugging information about services.

Verbs:

- `call` Call a service.
- `find` Output a list of services of a given type.
- `list` Output a list of service names.

srv (deprecated)

Verbs:

- `list` Ot
- `package` Ot
- `packages` Ot
- `show` Ot

test Run a ROS2

topic A tool for d topics, including i and messages.

Verbs:

- `bw` Displa
- `delay` Displa
- `echo` Outpu
- `find` Find t
- `hz` Displa
- `info` Outpu
- `list` Outpu
- `pub` Publis
- `type` Outpu

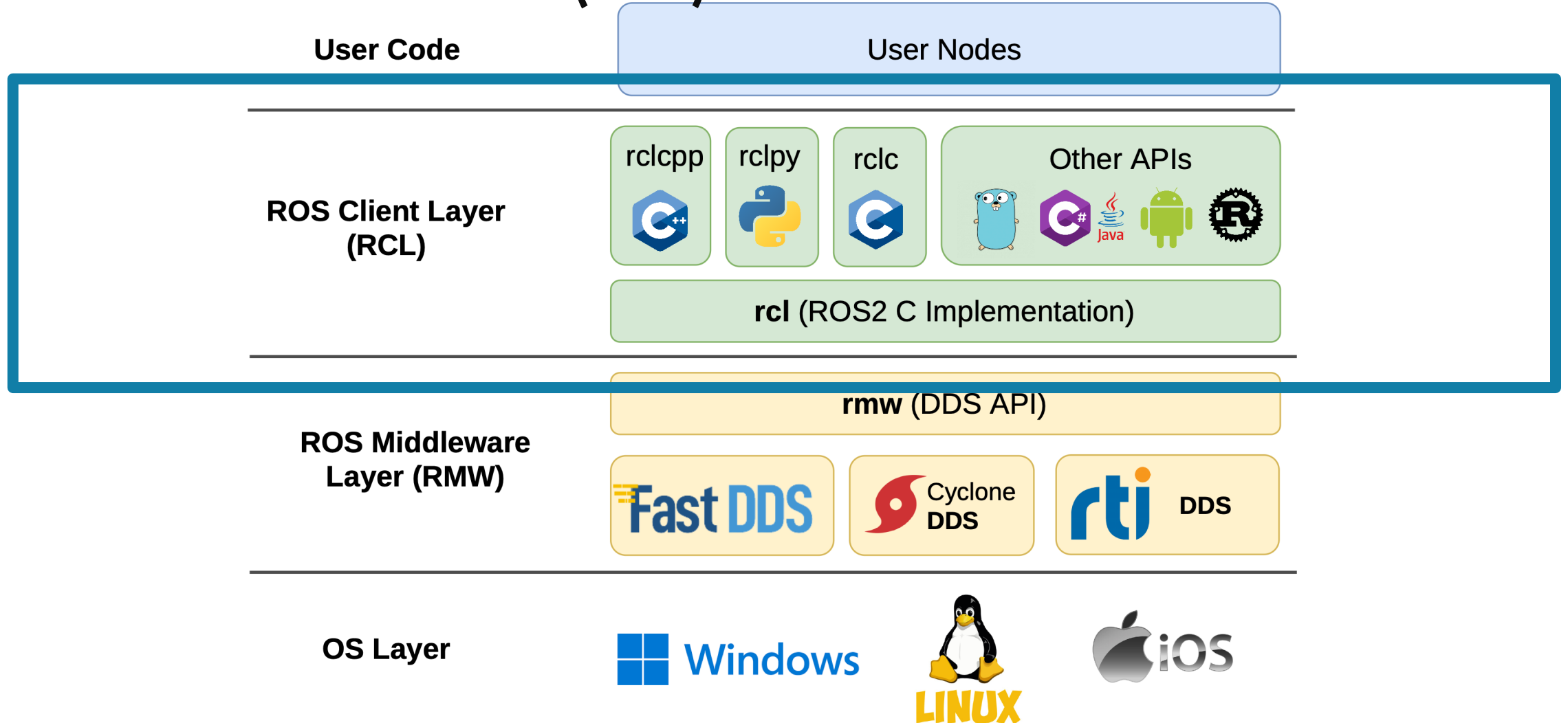
Examples:

```

$ ros2 topic bw
$ ros2 topic ech
$ ros2 topic finc
$ ros2 topic hz ,
$ ros2 topic infc
$ ros2 topic list
$ ros2 topic put 'data: Hello RC
$ ros2 topic typ

```

ROS CLIENT LAYER (RCL)




```
$ ros2
usage: ros2 [-h] Call 'ros2 <command> -h' for more detailed usage. ...
ros2 is an extensible command-line tool for ROS 2.
...
```

```
ros2 <command> <verb> [<params>|<option>]*
```

action	extension_points	node	test
bag	extensions	param	topic
component	interface	pkg	wtf
launch	run	daemon	lifecycle
security	doctor	multicast	service

Further readings:

- <https://github.com/ros2/ros2cli>
- https://github.com/ubuntu-robotics/ros2-cheats-sheet/blob/master/cli/cli_cheats_sheet.pdf

```
$ ros2 pkg list
```

```
ackermann_msgs
```

```
action_msgs
```

```
action_tutorials_cpp
```

```
...
```

```
$ ros2 pkg executables demo_nodes_cpp
```

```
demo_nodes_cpp add_two_ints_client
```

```
demo_nodes_cpp add_two_ints_client_async
```

```
demo_nodes_cpp add_two_ints_server
```

```
demo_nodes_cpp allocator_tutorial
```

```
...
```

```
demo_nodes_cpp talker
```

```
...
```

RUNNING A ROS2 PROGRAM (PUBLISHER)

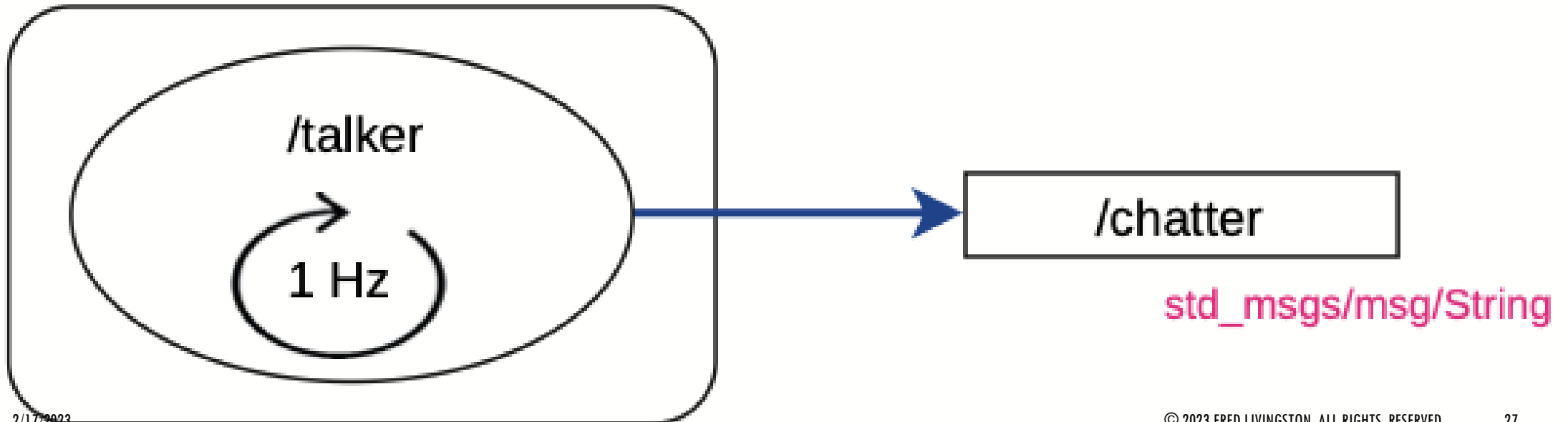
```
$ ros2 run demo_nodes_cpp talker
```

```
[INFO] [1643218362.316869744] [talker]: Publishing: 'Hello World: 1'
```

```
[INFO] [1643218363.316915225] [talker]: Publishing: 'Hello World: 2'
```

```
[INFO] [1643218364.316907053] [talker]: Publishing: 'Hello World: 3'
```

```
...
```



RUNNING A ROS2 PROGRAM

```
$ ros2 node list
```

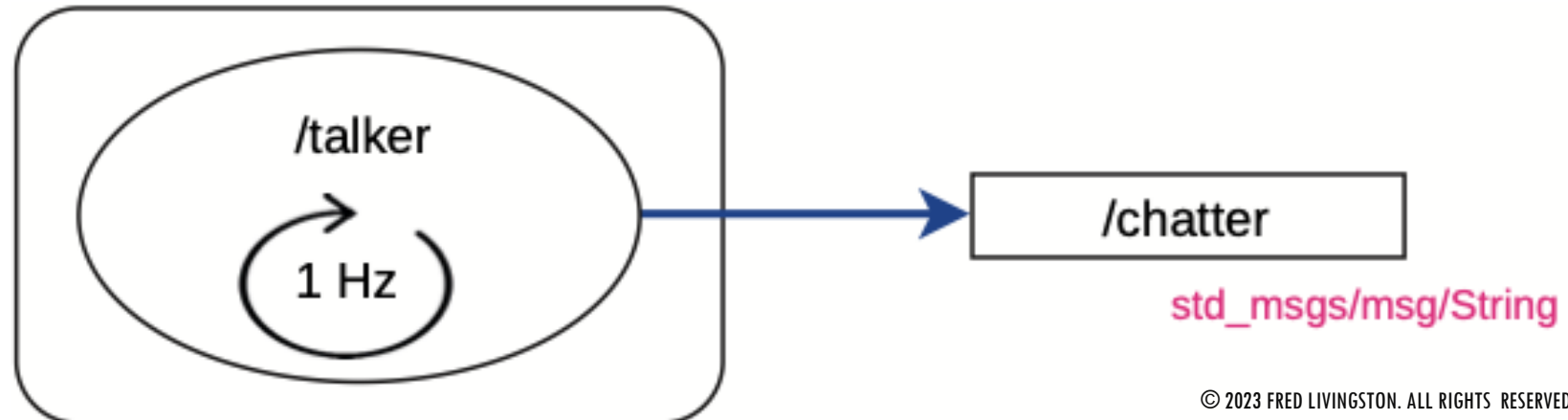
```
/talker
```

```
$ ros2 topic list
```

```
/chatter
```

```
/parameter_events
```

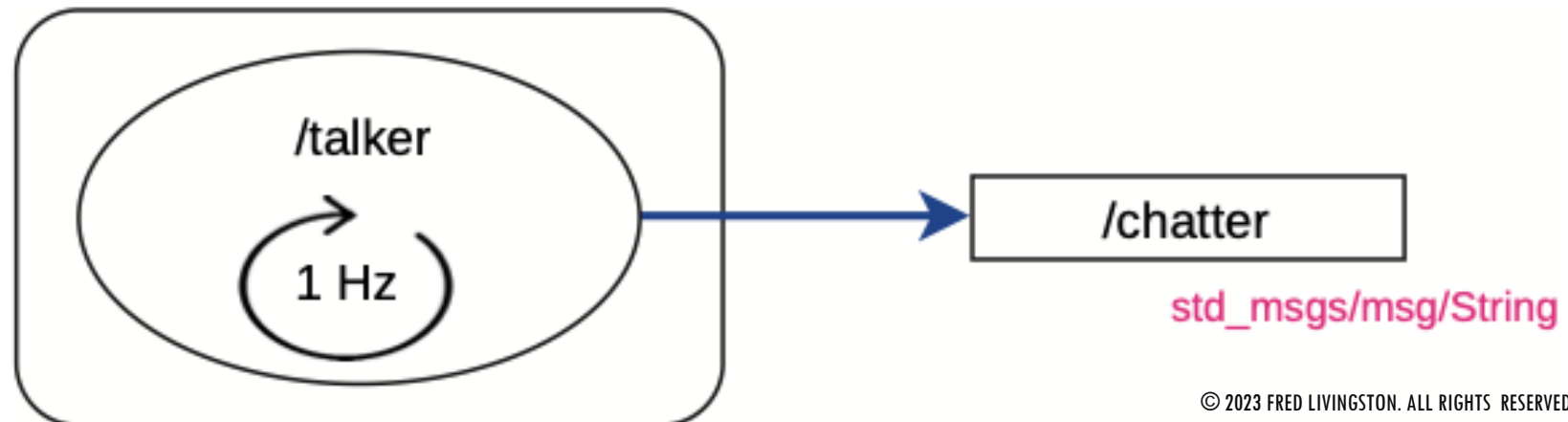
```
/rosout
```



RUNNING A ROS2 PROGRAM

```
$ ros2 node info /talker

/talker
  Subscribers:
    /parameter_events: rcl_interfaces/msg/ParameterEvent
  Publishers:
    /chatter: std_msgs/msg/String
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /rosout: rcl_interfaces/msg/Log
  Service Servers:
  ...
```



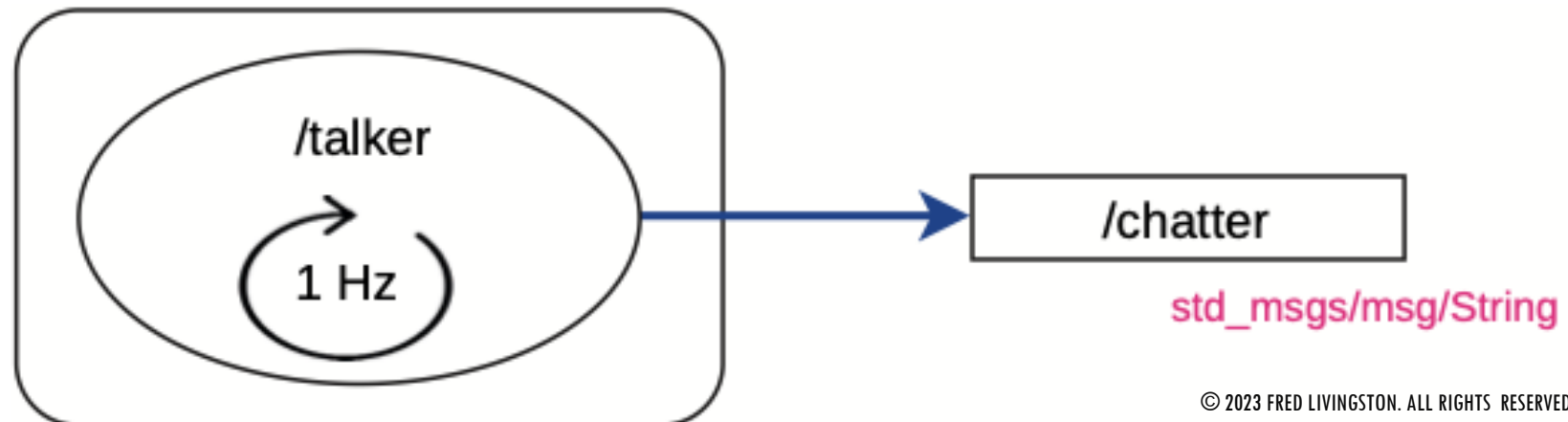
RUNNING A ROS2 PROGRAM

```
$ ros2 topic info /chatter
```

```
Type: std_msgs/msg/String
```

```
Publisher count: 1
```

```
Subscription count: 0
```



INTERFACES

```
$ ros2 interface list
```

```
Messages:
```

```
  ackermann_msgs/msg/AckermannDrive  
  ackermann_msgs/msg/AckermannDriveStamped  
  ...  
  visualization_msgs/msg/MenuEntry
```

```
Services:
```

```
  action_msgs/srv/CancelGoal  
  ...  
  visualization_msgs/srv/GetInteractiveMarkers
```

```
Actions:
```

```
  action_tutorials_interfaces/action/Fibonacci  
  ...
```

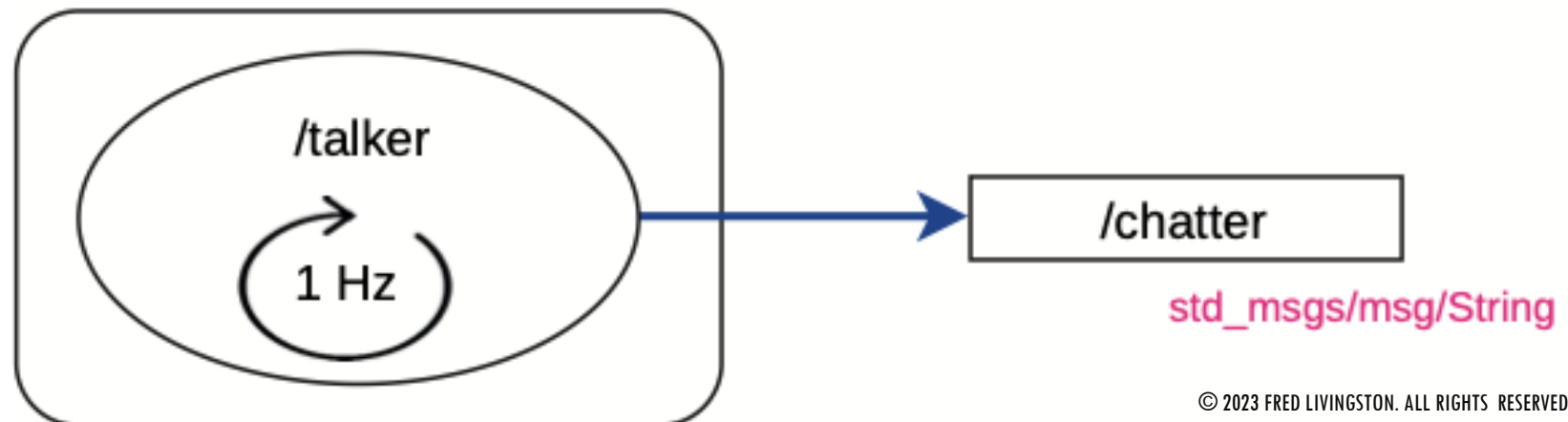
```
$ ros2 interface show std_msgs/msg/String
```

```
... comments
```

```
string data
```

TOPICS

```
$ ros2 topic echo /chatter  
data: 'Hello World: 1578'  
---  
data: 'Hello World: 1579'  
...
```



RUNNING A SUBSCRIBER

```
$ ros2 run demo_nodes_py listener
```

```
[INFO] [1643220136.232617223] [listener]: I heard: [Hello World: 1670]
```

```
[INFO] [1643220137.197551366] [listener]: I heard: [Hello World: 1671]
```

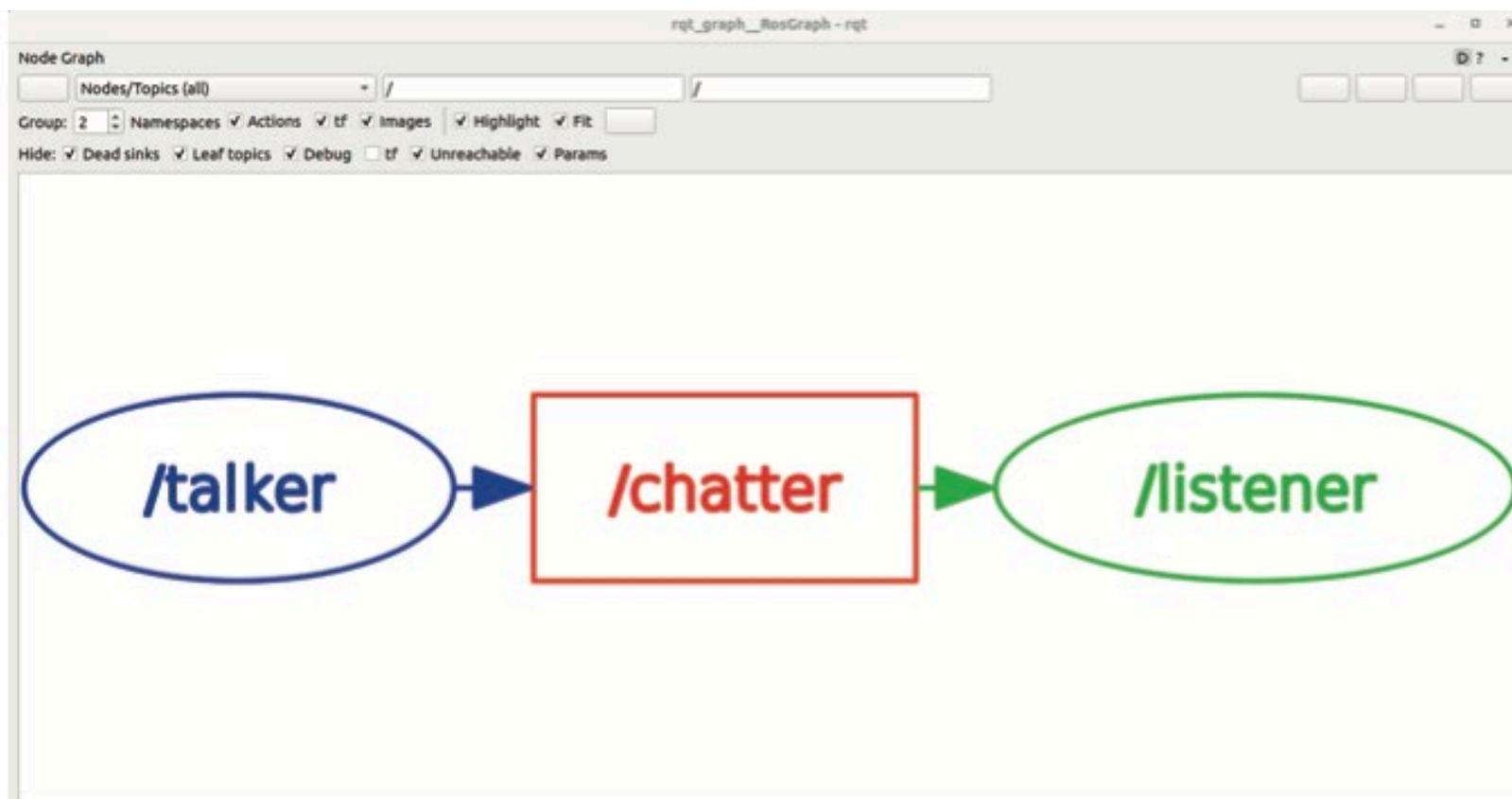
```
[INFO] [1643220138.198640098] [listener]: I heard: [Hello World: 1672]
```

```
...
```



RQT TOOLS

```
$ ros2 run rqt_graph rqt_graph
```



F1TENTH GYM

https://github.com/f1tenth/f1tenth_gym_ros

Native on Ubuntu 20.04

Install the following dependencies:

- ROS 2 Follow the instructions [here](#) to install ROS 2 Foxy.
- F1TENTH Gym

```
git clone https://github.com/f1tenth/f1tenth_gym
cd f1tenth_gym && pip3 install -e .
```

Installing the simulation:

- Create a workspace: `cd $HOME && mkdir -p sim_ws/src`
- Clone the repo into the workspace:

```
cd $HOME/sim_ws/src
git clone https://github.com/f1tenth/f1tenth_gym_ros
```

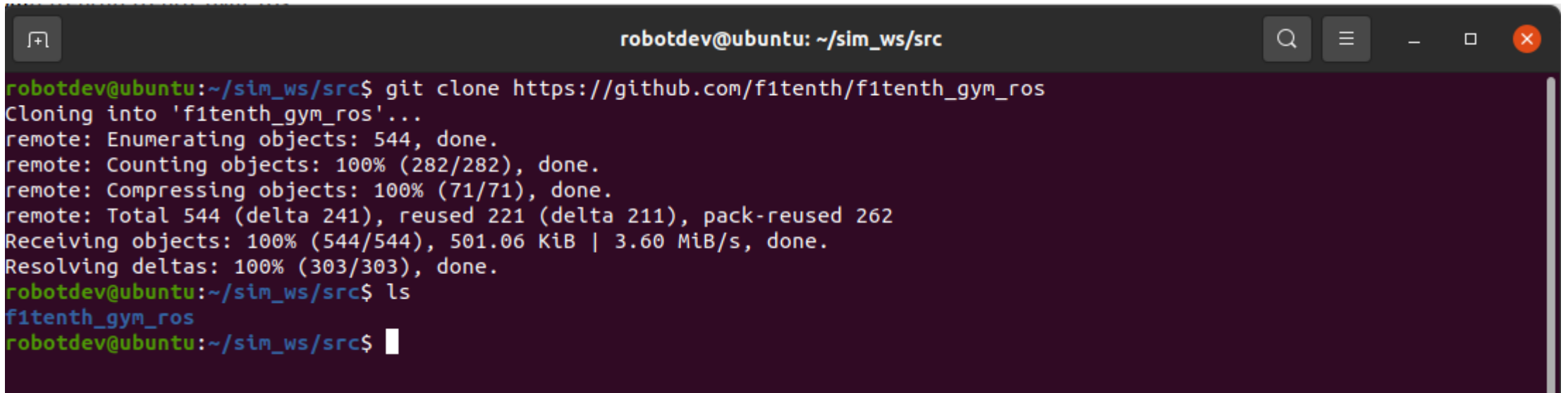
- Update correct parameter for path to map file: Go to `sim.yaml` https://github.com/f1tenth/f1tenth_gym_ros/blob/main/config/sim.yaml in your cloned repo, change the `map_path` parameter to point to the correct location. It should be `'<your_home_dir>/sim_ws/src/f1tenth_gym_ros/maps/levine'`
- Install dependencies with rosdep:

```
source /opt/ros/foxy/setup.bash
cd ..
rosdep install -i --from-path src --rosdistro foxy -y
```

- Build the workspace: `colcon build`

F1TENTH GYM (CLONE REPO)

```
$ git clone https://github.com/f1tenth/f1tenth_gym_ros
```

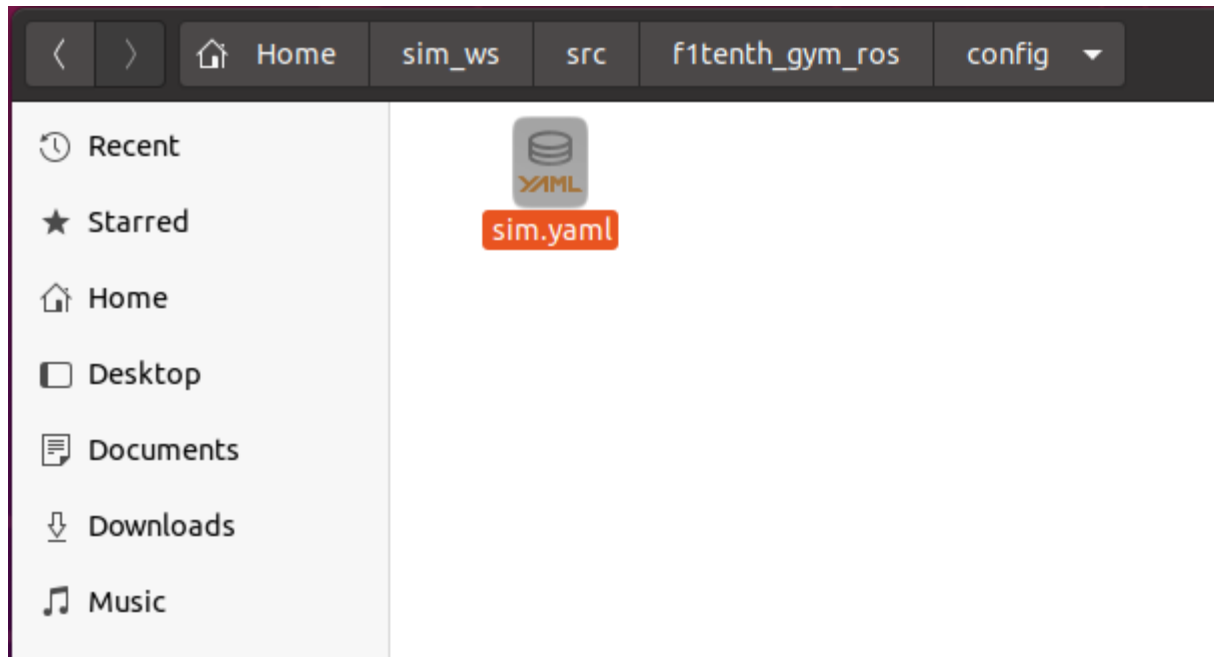
A terminal window with a dark purple background. The title bar reads "robotdev@ubuntu: ~/sim_ws/src". The terminal output shows the execution of the git clone command and its progress, including object enumeration, counting, and compression. The final output shows the directory "f1tenth_gym_ros" has been created.

```
robotdev@ubuntu:~/sim_ws/src$ git clone https://github.com/f1tenth/f1tenth_gym_ros
Cloning into 'f1tenth_gym_ros'...
remote: Enumerating objects: 544, done.
remote: Counting objects: 100% (282/282), done.
remote: Compressing objects: 100% (71/71), done.
remote: Total 544 (delta 241), reused 221 (delta 211), pack-reused 262
Receiving objects: 100% (544/544), 501.06 KiB | 3.60 MiB/s, done.
Resolving deltas: 100% (303/303), done.
robotdev@ubuntu:~/sim_ws/src$ ls
f1tenth_gym_ros
robotdev@ubuntu:~/sim_ws/src$
```

FITENTH GYM (CONFIGURATION)

Map_path: must contain full path name

Num_agent: 1 or 2



```
sim.yaml
~/sim_ws/src/f1tenth_gym_ros/config

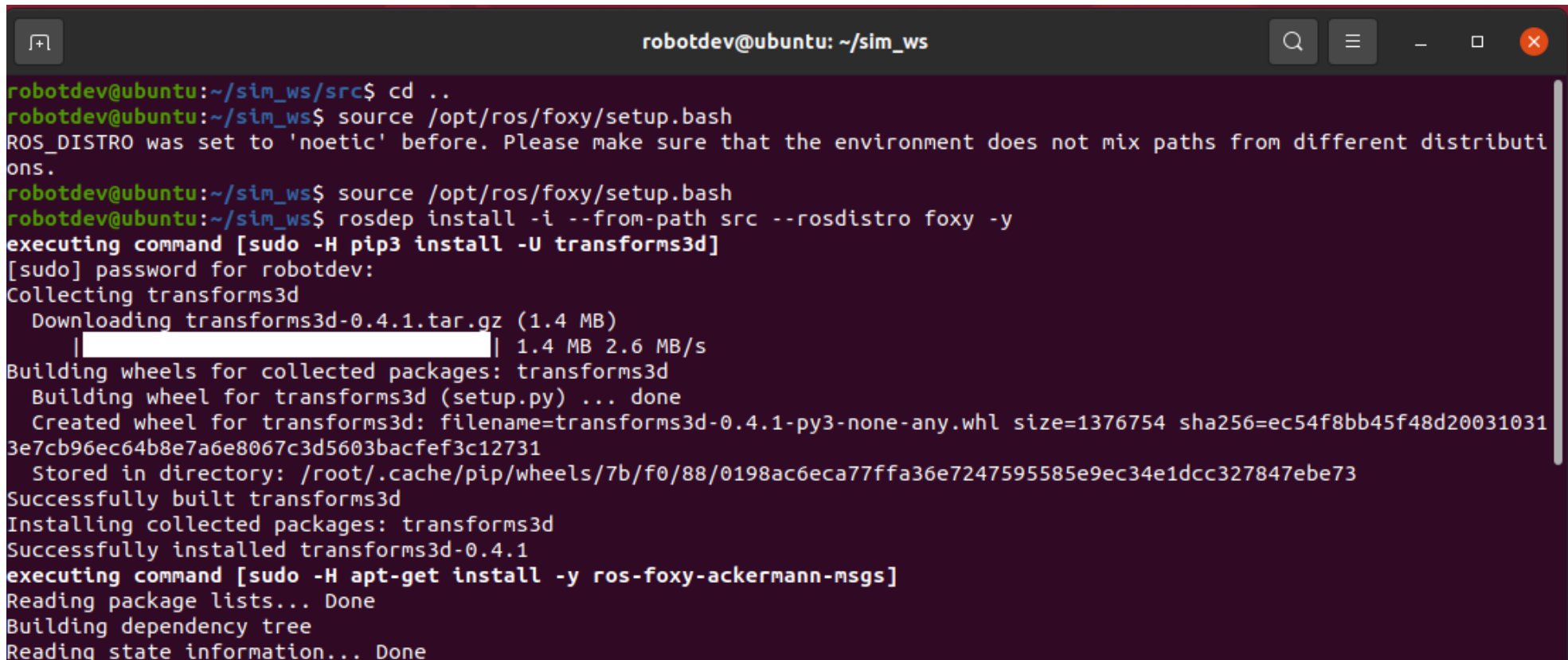
13 # copies or substantial portions of the Software.
14
15 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
16 # IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
17 # FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
18 # AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
19 # LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
20 # OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
21 # SOFTWARE.
22
23 bridge:
24   ros_parameters:
25     # topics and namespaces
26     ego_namespace: 'ego_racecar'
27     ego_scan_topic: 'scan'
28     ego_odom_topic: 'odom'
29     ego_opp_odom_topic: 'opp_odom'
30     ego_drive_topic: 'drive'
31     opp_namespace: 'opp_racecar'
32     opp_scan_topic: 'opp_scan'
33     opp_odom_topic: 'odom'
34     opp_ego_odom_topic: 'opp_odom'
35     opp_drive_topic: 'opp_drive'
36
37   # transform related
38   scan_distance_to_base_link: 0.0
39
40   # laserscan parameters
41   scan_fov: 4.7
42   scan_beams: 1080
43
44   # map parameters
45   map_path: '/home/robotdev/sim_ws/src/f1tenth_gym_ros/maps/levine'
46   map_img_ext: '.png'
47
48   # opponent parameters
49   num_agent: 1
50
51   # ego starting pose on map
52   sx: 0.0
53   sy: 0.0
54   stheta: 0.0
55
56   # opp starting pose on map
57   sx1: 2.0
58   sy1: 0.5
59   stheta1: 0.0
60
61   # teleop
62   kb_teleop: True
```

FIFTEENTH GYM (INSTALL DEPENDENCIES)

```
$ cd ..
```

```
$ source /opt/ros/foxy/setup.bash
```

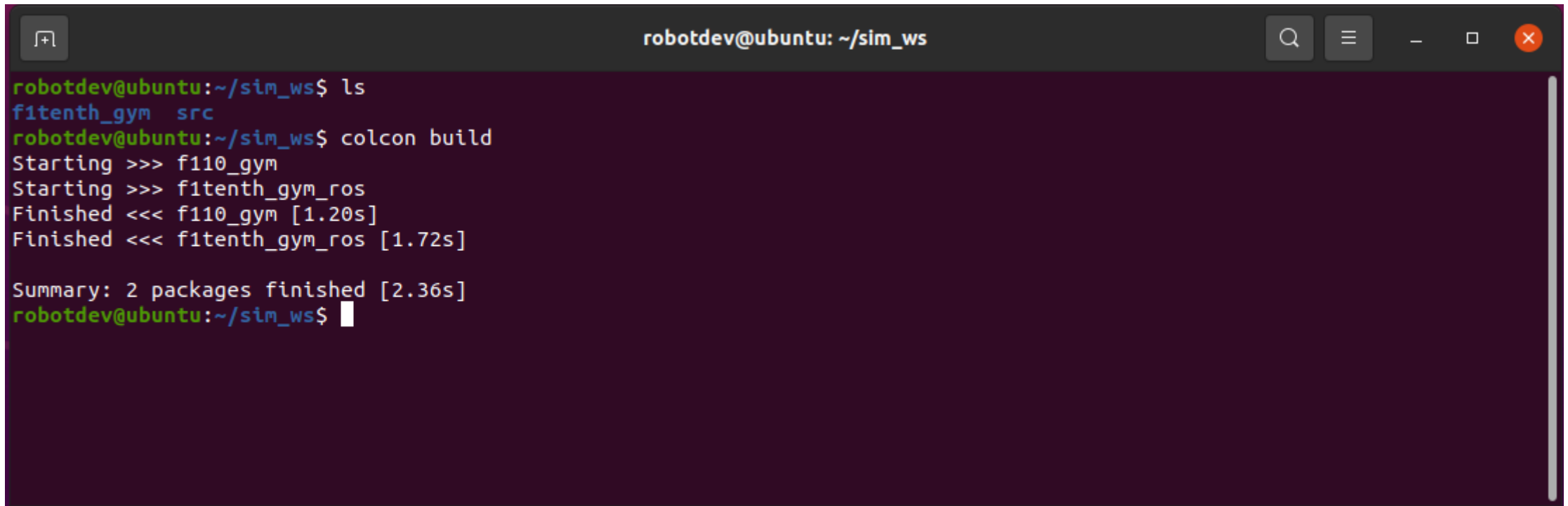
```
$ rosdep install -i --from-path src --rosdistro foxy -y
```



```
robotdev@ubuntu: ~/sim_ws
robotdev@ubuntu:~/sim_ws/src$ cd ..
robotdev@ubuntu:~/sim_ws$ source /opt/ros/foxy/setup.bash
ROS_DISTRO was set to 'noetic' before. Please make sure that the environment does not mix paths from different distributions.
robotdev@ubuntu:~/sim_ws$ source /opt/ros/foxy/setup.bash
robotdev@ubuntu:~/sim_ws$ rosdep install -i --from-path src --rosdistro foxy -y
executing command [sudo -H pip3 install -U transforms3d]
[sudo] password for robotdev:
Collecting transforms3d
  Downloading transforms3d-0.4.1.tar.gz (1.4 MB)
    |████████████████████████████████████████| 1.4 MB 2.6 MB/s
Building wheels for collected packages: transforms3d
  Building wheel for transforms3d (setup.py) ... done
  Created wheel for transforms3d: filename=transforms3d-0.4.1-py3-none-any.whl size=1376754 sha256=ec54f8bb45f48d200310313e7cb96ec64b8e7a6e8067c3d5603bacfef3c12731
  Stored in directory: /root/.cache/pip/wheels/7b/f0/88/0198ac6eca77ffa36e7247595585e9ec34e1dcc327847ebe73
Successfully built transforms3d
Installing collected packages: transforms3d
Successfully installed transforms3d-0.4.1
executing command [sudo -H apt-get install -y ros-foxy-ackermann-msgs]
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

FITENTH GYM (COMPILE SRC)

\$ colcon build

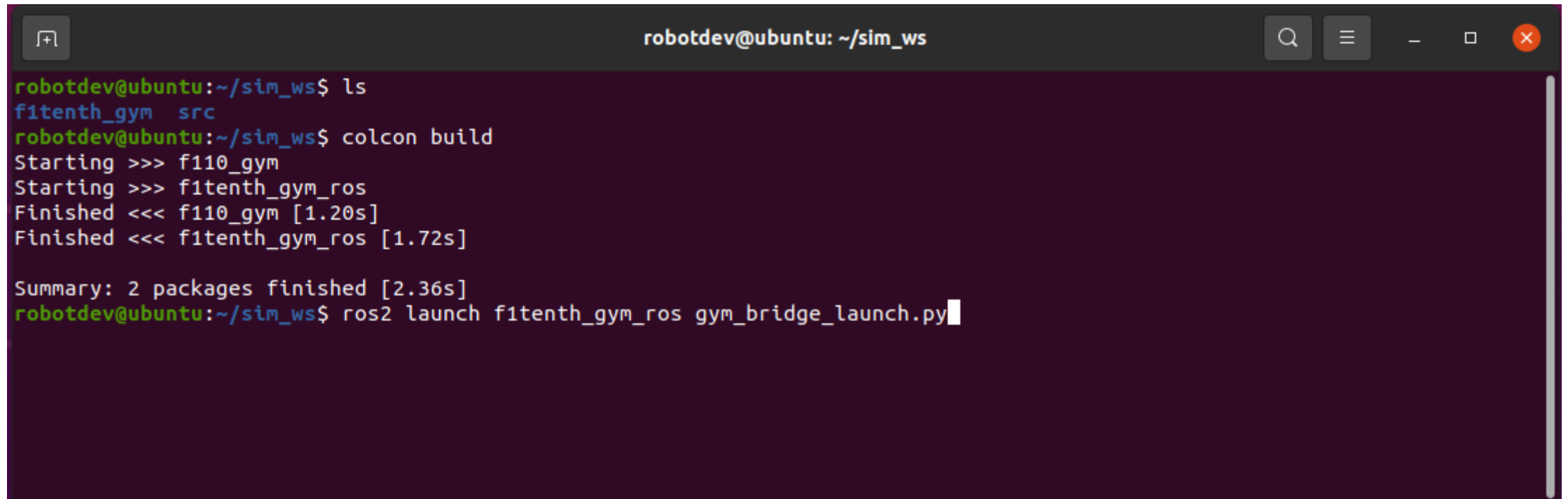
A terminal window titled 'robotdev@ubuntu: ~/sim_ws' with search, menu, and window control icons. The terminal shows the execution of 'colcon build' in a directory containing 'f1tenth_gym' and 'src'. The output indicates successful compilation of 'f110_gym' and 'f1tenth_gym_ros' packages.

```
robotdev@ubuntu:~/sim_ws$ ls
f1tenth_gym  src
robotdev@ubuntu:~/sim_ws$ colcon build
Starting >>> f110_gym
Starting >>> f1tenth_gym_ros
Finished <<< f110_gym [1.20s]
Finished <<< f1tenth_gym_ros [1.72s]

Summary: 2 packages finished [2.36s]
robotdev@ubuntu:~/sim_ws$
```

F1TENTH GYM (RUN SIMULATOR)

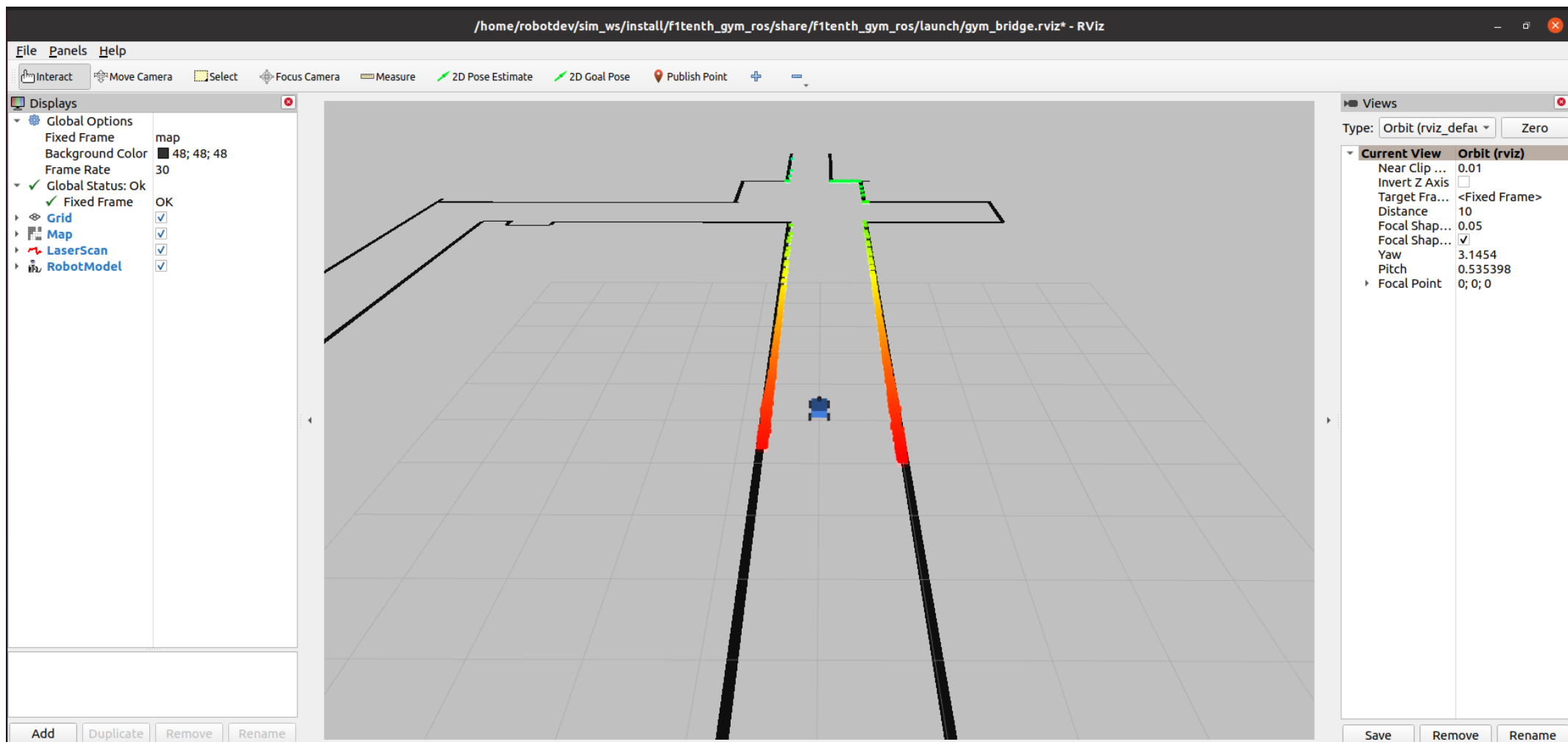
```
$ ros2 launch f1tenth_gym_ros gym_bridge_launch.py
```

A terminal window titled "robotdev@ubuntu: ~/sim_ws" with standard window controls. The terminal shows the following output:

```
robotdev@ubuntu:~/sim_ws$ ls
f1tenth_gym  src
robotdev@ubuntu:~/sim_ws$ colcon build
Starting >>> f110_gym
Starting >>> f1tenth_gym_ros
Finished <<< f110_gym [1.20s]
Finished <<< f1tenth_gym_ros [1.72s]

Summary: 2 packages finished [2.36s]
robotdev@ubuntu:~/sim_ws$ ros2 launch f1tenth_gym_ros gym_bridge_launch.py
```


F1TENTH GYM (RUN SIMULATOR)



TOPICS PUBLISHED BY THE SIMULATION

In single agent:

/scan: The ego agent's laser scan

/ego_racecar/odom: The ego agent's odometry

/map: The map of the environment

In two agents:

In addition to the topics available in the single agent scenario, these topics are also available:

/opp_scan: The opponent agent's laser scan

/ego_racecar/opp_odom: The opponent agent's odometry for the ego agent's planner

/opp_racecar/odom: The opponent agents' odometry

/opp_racecar/opp_odom: The ego agent's odometry for the opponent agent's planner

TOPICS SUBSCRIBED BY THE SIMULATION

In single agent:

/drive: The ego agent's drive command via **AckermannDriveStamped** messages

/initialpose: This is the topic for resetting the ego's pose via RViz's 2D Pose Estimate tool. Do NOT publish directly to this topic unless you know what you're doing.

In two agents:

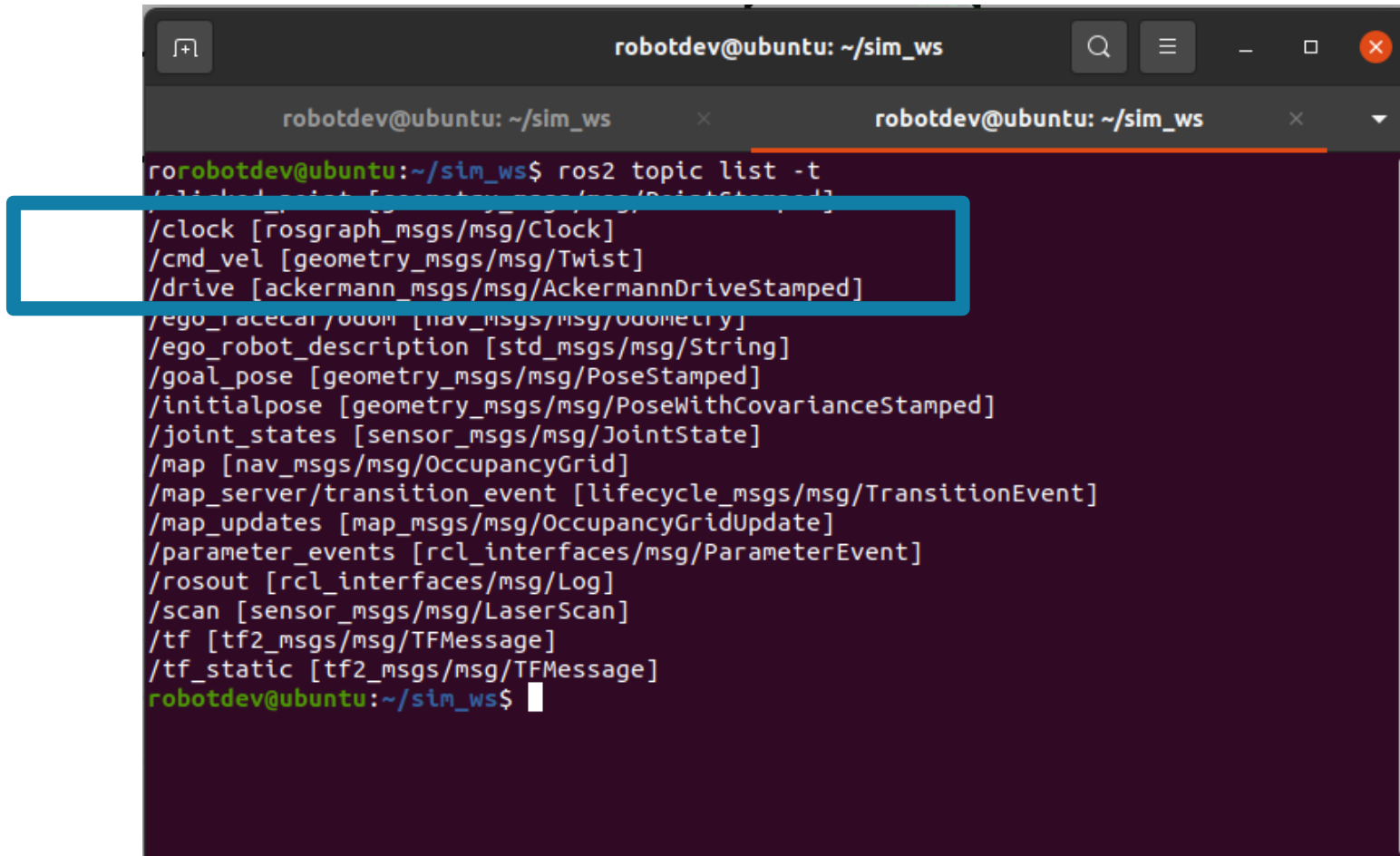
In addition to all topics in the single agent scenario, these topics are also available:

/opp_drive: The opponent agent's drive command via **AckermannDriveStamped** messages

/goal_pose: This is the topic for resetting the opponent agent's pose via RViz's 2D Goal Pose tool. Do NOT publish directly to this topic unless you know what you're doing

FIFTEENTH GYM (TOPICS)

\$ ros2 topic list -t



```
robotdev@ubuntu: ~/sim_ws
robotdev@ubuntu: ~/sim_ws
robotdev@ubuntu: ~/sim_ws$ ros2 topic list -t
/clock [rosgraph_msgs/msg/Clock]
/cmd_vel [geometry_msgs/msg/Twist]
/drive [ackermann_msgs/msg/AckermannDriveStamped]
/ego_racecar/odom [nav_msgs/msg/Odometry]
/ego_robot_description [std_msgs/msg/String]
/goal_pose [geometry_msgs/msg/PoseStamped]
/initialpose [geometry_msgs/msg/PoseWithCovarianceStamped]
/joint_states [sensor_msgs/msg/JointState]
/map [nav_msgs/msg/OccupancyGrid]
/map_server/transition_event [lifecycle_msgs/msg/TransitionEvent]
/map_updates [map_msgs/msg/OccupancyGridUpdate]
/parameter_events [rcl_interfaces/msg/ParameterEvent]
/rosout [rcl_interfaces/msg/Log]
/scan [sensor_msgs/msg/LaserScan]
/tf [tf2_msgs/msg/TFMessage]
/tf_static [tf2_msgs/msg/TFMessage]
robotdev@ubuntu: ~/sim_ws$
```

GEOMETRY_MSGS/MSG/TWIST

http://docs.ros.org/en/noetic/api/geometry_msgs/html/msg/Twist.html

geometry_msgs/Twist Message

File: `geometry_msgs/Twist.msg`

Raw Message Definition

```
# This expresses velocity in free space broken into its linear and angular parts.  
Vector3 linear  
Vector3 angular
```

Compact Message Definition

```
geometry_msgs/Vector3 linear  
geometry_msgs/Vector3 angular
```

autogenerated on Wed, 02 Mar 2022 00:06:53

ACKERMANN_MSGS/MAG/ACKERMANNDRIVESTAMPED

http://docs.ros.org/en/melodic/api/ackermann_msgs/html/msg/AckermannDriveStamped.html

[ackermann_msgs/AckermannDriveStamped Message](#)

File: `ackermann_msgs/AckermannDriveStamped.msg`

Raw Message Definition

```
## Time stamped drive command for robots with Ackermann steering.
# $Id$

Header          header
AckermannDrive  drive
```

Compact Message Definition

```
std_msgs/Header header
ackermann_msgs/AckermannDrive drive
```

autogenerated on Mon, 28 Feb 2022 21:32:24

F1 TENTH GYM (TELEOP)

```
$ ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

The screenshot displays a ROS2 teleoperation interface. The main window shows a 3D environment with a robot (a small blue cube) on a grey floor. A path is visible, starting from the robot and extending into the distance, colored with a gradient from yellow to red. The interface includes several panels:

- Displays:** Global Options (Fixed Frame: map, Background Color: 48; 48; 48, Frame Rate: 30), Global Status: Ok.
- Views:** Type: Orbit (rviz_defat), Zero. Current View: Orbit (rviz). Parameters: Near Clip ... 0.01, Invert Z Axis , Target Fra... <Fixed Frame>, Distance 10, Focal Shap... 0.05, Focal Shap... , Yaw 3.1454, Pitch 0.535398, Focal Point 0; 0; 0.
- Terminal:** robotdev@ubuntu: ~/sim_ws. Shows the command `ros2 run teleop_twist_keyboard teleop_twist_keyboard` and its output, including a warning about RTPS transport and a list of keyboard controls for moving around and adjusting speeds.

Terminal output:

```
robotdev@ubuntu: ~/sim_ws
robotdev@ubuntu: ~/sim_ws
robotdev@ubuntu: ~/sim_ws
~/rosout
robotdev@ubuntu:~/sim_ws$ ros2 run teleop_twist_keyboard teleop_twist_keyboard
2023-02-17 00:47:02.980 [RTPS_TRANSPORT_SHM Error] Failed init_port fastrtps_port7421: open_and_lock_file failed -> Function open_port_internal

This node takes keypresses from the keyboard and publishes them
as Twist messages. It works best with a US keyboard layout.
-----
Moving around:
  u   i   o
  j   k   l
  m   ,   .

For Holonomic mode (strafing), hold down the shift key:
-----
  U   I   O
  J   K   L
  M   <   >

t : up (+z)
b : down (-z)

anything else : stop

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%

CTRL-C to quit
```

FIFTEENTH GYM (TELEOP)

```
$ ros2 topic echo /cmd_vel
```

The image shows a ROS2 teleop simulation environment. On the left, a terminal window displays the output of the `ros2 run teleop_twist_keyboard teleop_twist_keyboard` command. The output includes a warning about `RTPS_TRANSPORT_SHM Error` and a help message for the teleop node. The help message lists keyboard controls for moving around and for holonomic mode (strafing), along with speed adjustment keys. On the right, another terminal window shows the output of the `ros2 topic echo /cmd_vel` command. The output displays the current velocity commands for linear and angular motion, showing values for x, y, and z coordinates and angular velocities.

```
robotdev@ubuntu: ~/sim_ws
```

```
robotdev@ubuntu: ~/sim_ws$ ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

```
2023-02-17 00:47:02.980 [RTPS_TRANSPORT_SHM Error] Failed init_port fastrtps_port7421: open_and_lock_file failed -> Function open_port_internal
```

```
This node takes keypresses from the keyboard and publishes them as Twist messages. It works best with a US keyboard layout.
```

```
-----
```

```
Moving around:
```

```
u i o
```

```
j k l
```

```
m , .
```

```
For Holonomic mode (strafing), hold down the shift key:
```

```
-----
```

```
U I O
```

```
J K L
```

```
M < >
```

```
t : up (+z)
```

```
b : down (-z)
```

```
anything else : stop
```

```
q/z : increase/decrease max speeds by 10%
```

```
w/x : increase/decrease only linear speed by 10%
```

```
e/c : increase/decrease only angular speed by 10%
```

```
CTRL-C to quit
```

```
robotdev@ubuntu: ~/sim_ws$ ros2 topic echo /cmd_vel
```

```
Call `ros2 topic <command> -h` for more detailed usage.
```

```
robotdev@ubuntu:~/sim_ws$ ros2 topic echo /cmd_vel
```

```
linear:
```

```
  x: 0.0
```

```
  y: 0.0
```

```
  z: 0.0
```

```
angular:
```

```
  x: 0.0
```

```
  y: 0.0
```

```
  z: 0.0
```

```
---
```

```
linear:
```

```
  x: -0.5
```

```
  y: 0.0
```

```
  z: 0.0
```

```
angular:
```

```
  x: 0.0
```

```
  y: 0.0
```

```
  z: 0.0
```

```
---
```

```
linear:
```

```
  x: 0.0
```

```
  y: 0.0
```

```
  z: 0.0
```

```
17 fps
```


DEVELOPING PROGRAMS

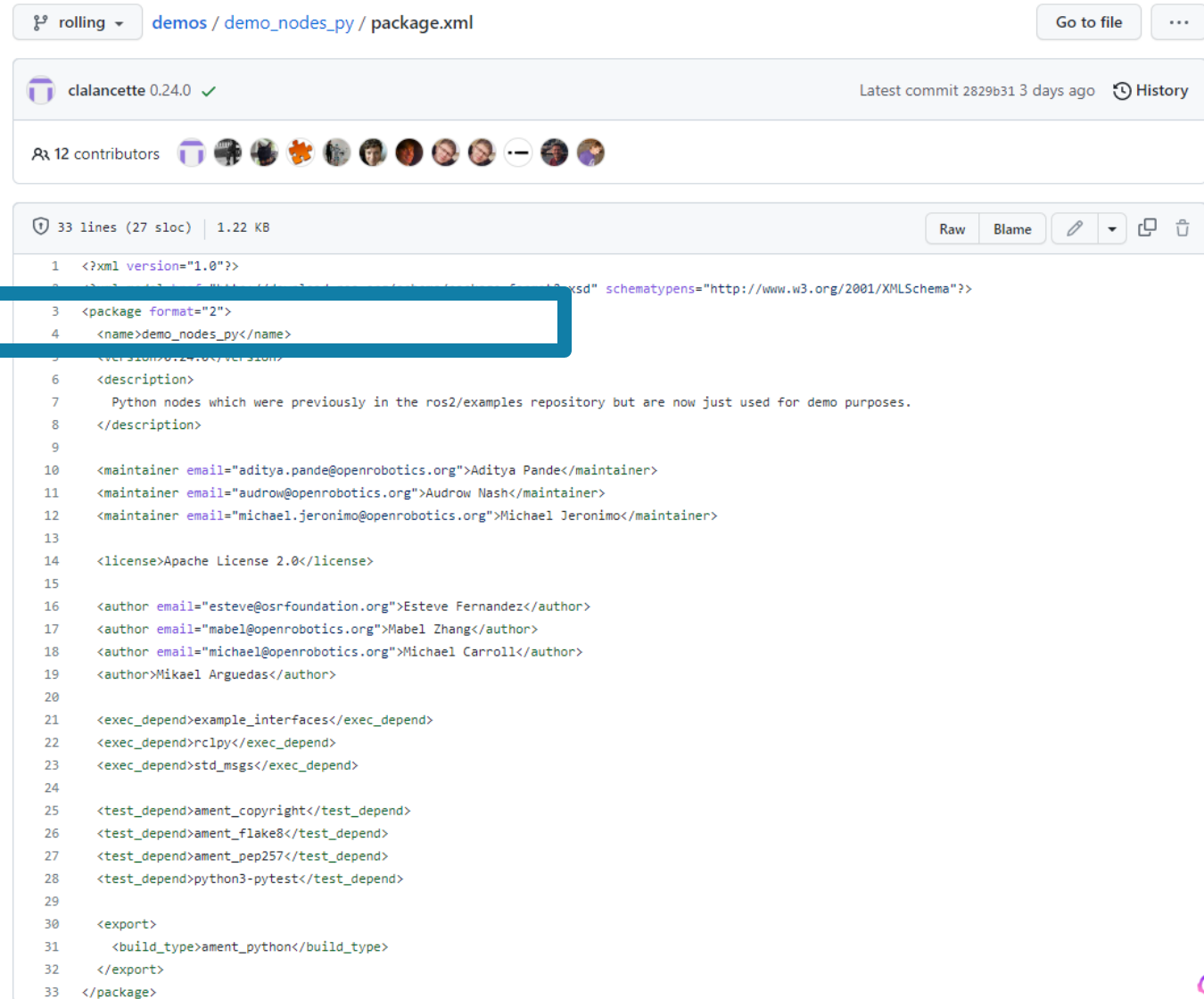
ROS2 Demos

<https://github.com/ros2/demos>

The screenshot shows the GitHub repository page for `ros2/demos`. The repository is public and has 265 forks and 331 stars. The current branch is `rolling`. The file structure is as follows:

File/Folder	Commit Message	Commit Date
demo_nodes_py	Demo for pre and post set parameter callback support (#565)	4 months ago
img	Added README.md for demo_nodes_py (#600)	2 weeks ago
resource	install data_files	6 years ago
test	more verbose test_flake8 error messages (same as ros2/laun...	3 years ago
CHANGELOG.rst	0.24.0	3 days ago
README.md	Added README.md for demo_nodes_py (#600)	2 weeks ago
package.xml	0.24.0	3 days ago
setup.cfg	Use underscores instead of dashes in setup.cfg (#502)	2 years ago
setup.py	0.24.0	3 days ago

PACKAGE.XML



The screenshot shows a GitHub repository page for the file `package.xml` in the `demos / demo_nodes_py` directory. The repository is named `clalancette` and is at version `0.24.0`. The file is 33 lines long (27 sloc) and 1.22 KB in size. The code content is as follows:

```
1 <?xml version="1.0"?>
2 <?xml-stylesheet href="http://www.w3.org/2001/XMLSchema.xsd" schematypens="http://www.w3.org/2001/XMLSchema"?>
3 <package format="2">
4   <name>demo_nodes_py</name>
5   <version>0.24.0</version>
6   <description>
7     Python nodes which were previously in the ros2/examples repository but are now just used for demo purposes.
8   </description>
9
10  <maintainer email="aditya.pande@openrobotics.org">Aditya Pande</maintainer>
11  <maintainer email="audrow@openrobotics.org">Audrow Nash</maintainer>
12  <maintainer email="michael.jeronimo@openrobotics.org">Michael Jeronimo</maintainer>
13
14  <license>Apache License 2.0</license>
15
16  <author email="estev@osrfoundation.org">Esteve Fernandez</author>
17  <author email="mabel@openrobotics.org">Mabel Zhang</author>
18  <author email="michael@openrobotics.org">Michael Carroll</author>
19  <author>Mikael Arguedas</author>
20
21  <exec_depend>example_interfaces</exec_depend>
22  <exec_depend>rclpy</exec_depend>
23  <exec_depend>std_msgs</exec_depend>
24
25  <test_depend>ament_copyright</test_depend>
26  <test_depend>ament_flake8</test_depend>
27  <test_depend>ament_pep257</test_depend>
28  <test_depend>python3-pytest</test_depend>
29
30  <export>
31    <build_type>ament_python</build_type>
32  </export>
33 </package>
```

CREATE A NEW PACKAGE

```
$ cd ~/sim_ws/src
```

```
$ ros2 pkg create my_robot_controller --build-type ament_python
```

```
robotdev@ubuntu:~/sim_ws/src$ ls
fitenth_gym_ros
robotdev@ubuntu:~/sim_ws/src$ ros2 pkg create my_robot_controller --build-type ament_python
going to create a new package
package name: my_robot_controller
destination directory: /home/robotdev/sim_ws/src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['robotdev <Fred.Livingston@gmail.com>']
licenses: ['TODO: License declaration']
build type: ament_python
dependencies: []
creating folder ./my_robot_controller
creating ./my_robot_controller/package.xml
creating source folder
creating folder ./my_robot_controller/my_robot_controller
creating ./my_robot_controller/setup.py
creating ./my_robot_controller/setup.cfg
creating folder ./my_robot_controller/resource
creating ./my_robot_controller/resource/my_robot_controller
creating ./my_robot_controller/my_robot_controller/__init__.py
creating folder ./my_robot_controller/test
creating ./my_robot_controller/test/test_copyright.py
creating ./my_robot_controller/test/test_flake8.py
creating ./my_robot_controller/test/test_pep257.py
robotdev@ubuntu:~/sim_ws/src$
```

SIMPLE PUBLISHER (TALKER.PY)

https://github.com/ros2/demos/blob/rolling/demo_nodes_py/demo_nodes_py/topics/talker.py

Line 25:

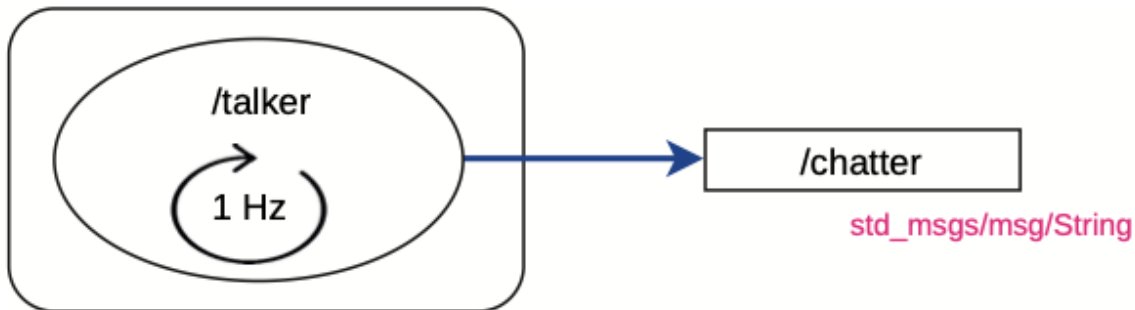
```
super().__init__('NAME_OF_PROCESS')
```

Line 27:

```
self.create_publisher(String, 'chatter', 10)
```

Line 29:

```
self.create_timer(1, self.timer_callback)
```



```
14
15 import rclpy
16 from rclpy.executors import ExternalShutdownException
17 from rclpy.node import Node
18
19 from std_msgs.msg import String
20
21
22 class Talker(Node):
23
24     def __init__(self):
25         super().__init__('talker')
26         self.i = 0
27         self.pub = self.create_publisher(String, 'chatter', 10)
28         timer_period = 1.0
29         self.tmr = self.create_timer(timer_period, self.timer_callback)
30
31     def timer_callback(self):
32         msg = String()
33         msg.data = 'Hello World: {}'.format(self.i)
34         self.i += 1
35         self.get_logger().info('Publishing: "{}"'.format(msg.data))
36         self.pub.publish(msg)
37
```

CREATE A PYTHON PROGRAM

```
$ cd my_robot_controller/my_robot_controller/  
$ touch move_robot.py  
$ gedit move_robot.py
```

```
robotdev@ubuntu:~/sim_ws/src$ ls  
fitenth_gym_ros  my_robot_controller  
robotdev@ubuntu:~/sim_ws/src$ cd my_robot_controller/my_robot_controller/  
robotdev@ubuntu:~/sim_ws/src/my_robot_controller/my_robot_controller$ touch move_robot.py  
robotdev@ubuntu:~/sim_ws/src/my_robot_controller/my_robot_controller$
```

MOVE_ROBOT.PY

```
$ cd my_robot_controller/my_robot_controller/  
$ touch move_robot.py  
$ gedit move_robot.py
```

```
robotdev@ubuntu:~/sim_ws/src$ ls  
fitenth_gym_ros  my_robot_controller  
robotdev@ubuntu:~/sim_ws/src$ cd my_robot_controller/my_robot_controller/  
robotdev@ubuntu:~/sim_ws/src/my_robot_controller/my_robot_controller$ touch move_robot.py  
robotdev@ubuntu:~/sim_ws/src/my_robot_controller/my_robot_controller$
```

MOVE_ROBOT.PY

```
move_robot.py x
home > robotdev > sim_ws > src > my_robot_controller > my_robot_controller > move_robot.py > ...
1 # move_robot.py
2 # Fred Livingston (fjliving@ncsu.edu) 2-17-2023
3
4 import rclpy
5 from rclpy.executors import ExternalShutdownException
6 from rclpy.node import Node
7
8 from geometry_msgs.msg import Twist
9
10
11 class Controller(Node):
12
13     def __init__(self):
14         super().__init__('move_robot')
15         self.pub = self.create_publisher(Twist, 'cmd_vel', 10)
16
17         # move robot fwd
18         msg = Twist()
19         msg.linear.x = 0.5
20         msg.linear.y = 0.0
21         msg.linear.z = 0.0
22         msg.angular.x = 0.0
23         msg.angular.y = 0.0
24         msg.angular.z = 0.0
25         self.pub.publish(msg)
26
27         timer_period = 10.0
28         self.tmr = self.create_timer(timer_period, self.timer_callback)
29
30     def timer_callback(self):
31         # stop robot
32         msg = Twist()
33         msg.linear.x = 0.0
34         msg.linear.y = 0.0
35         msg.linear.z = 0.0
36         msg.angular.x = 0.0
37         msg.angular.y = 0.0
38         msg.angular.z = 0.0
39         self.pub.publish(msg)
```

```
move_robot.py x
home > robotdev > sim_ws > src > my_robot_controller > my_robot_controller > move_robot.py > ...
27         timer_period = 10.0
28         self.tmr = self.create_timer(timer_period, self.timer_callback)
29
30     def timer_callback(self):
31         # stop robot
32         msg = Twist()
33         msg.linear.x = 0.0
34         msg.linear.y = 0.0
35         msg.linear.z = 0.0
36         msg.angular.x = 0.0
37         msg.angular.y = 0.0
38         msg.angular.z = 0.0
39         self.pub.publish(msg)
40
41
42 def main(args=None):
43     rclpy.init(args=args)
44
45     node = Controller()
46
47     try:
48         rclpy.spin(node)
49     except (KeyboardInterrupt, ExternalShutdownException):
50         pass
51     finally:
52         node.destroy_node()
53         rclpy.try_shutdown()
54
55
56 if __name__ == '__main__':
57     main()
58
```

SETUP.PY

```
Open setup.py Save
~/sim_ws/src/my_robot_controller

1 from setuptools import setup
2
3 package_name = 'my_robot_controller'
4
5 setup(
6     name=package_name,
7     version='0.0.0',
8     packages=[package_name],
9     data_files=[
10         ('share/ament_index/resource_index/packages',
11          ['resource/' + package_name]),
12         ('share/' + package_name, ['package.xml']),
13     ],
14     install_requires=['setuptools'],
15     zip_safe=True,
16     maintainer='robotdev',
17     maintainer_email='Fred.Livingston@gmail.com',
18     description='TODO: Package description',
19     license='TODO: License declaration',
20     tests_require=['pytest'],
21     entry_points={
22         'console_scripts': [
23             'move_robot = my_robot_controller.move_robot:main'
24         ],
25     },
26 )

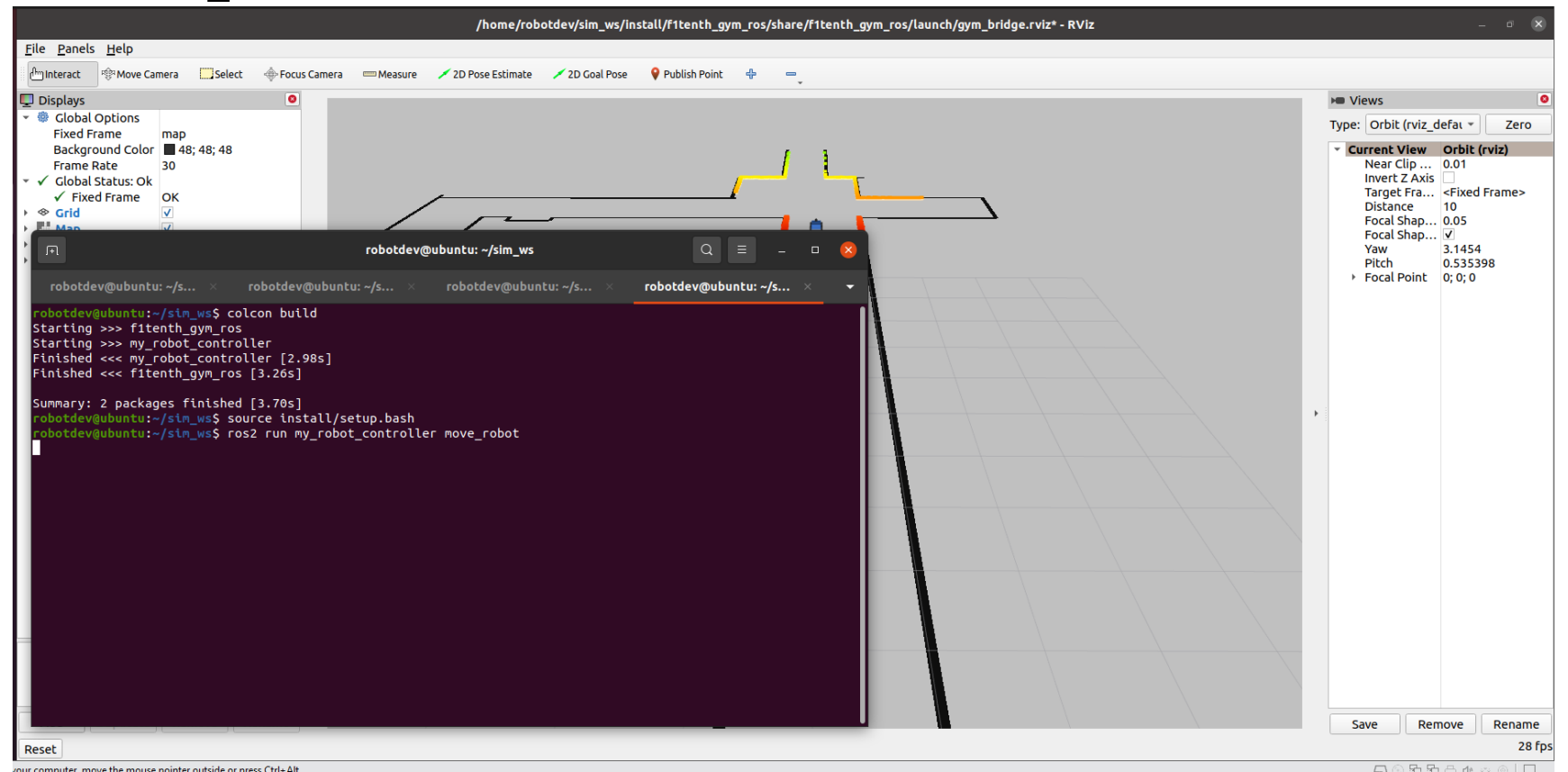
Python Tab Width: 8 Ln 23, Col 20 © 2023 FRED LIVINGSTON. ALL RIGHTS RESERVED.
```


BUILD AND EXECUTE ROBOT CONTROLLER

\$ colcon build

\$ source install/setup.bash

\$ ros2 run my_robot_controller move_robot



END OF WORKSHOP

Fred Livingston (fjliving@ncsu.edu)