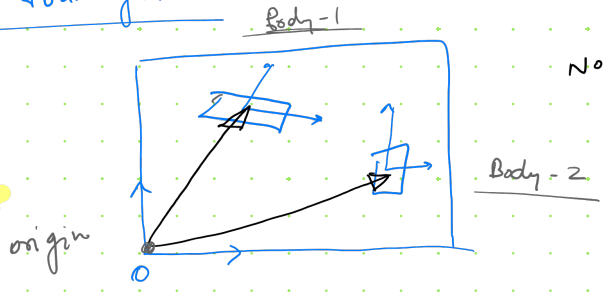
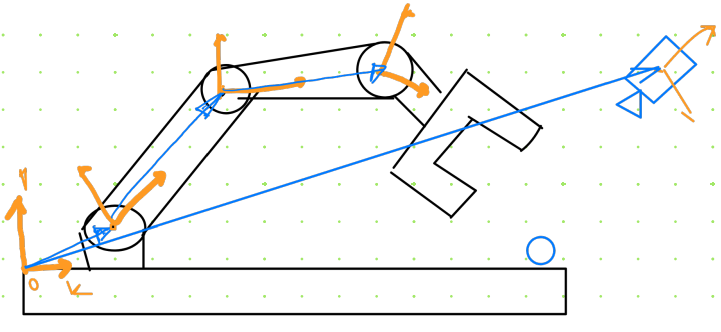
class 4

ROS-6

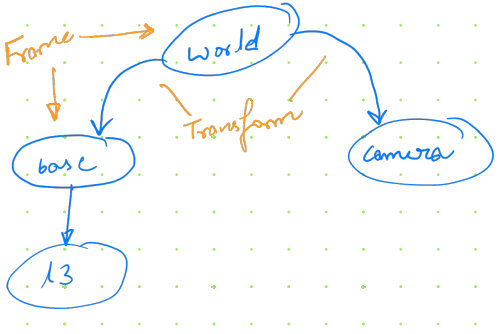
Coordinate transformation



No closed loop



ROS TF → Broadcast transform



base & camera are defined w.r.t world

Broadcaster transform static & dynamic

Broadcasting & Listening

Example of static transform

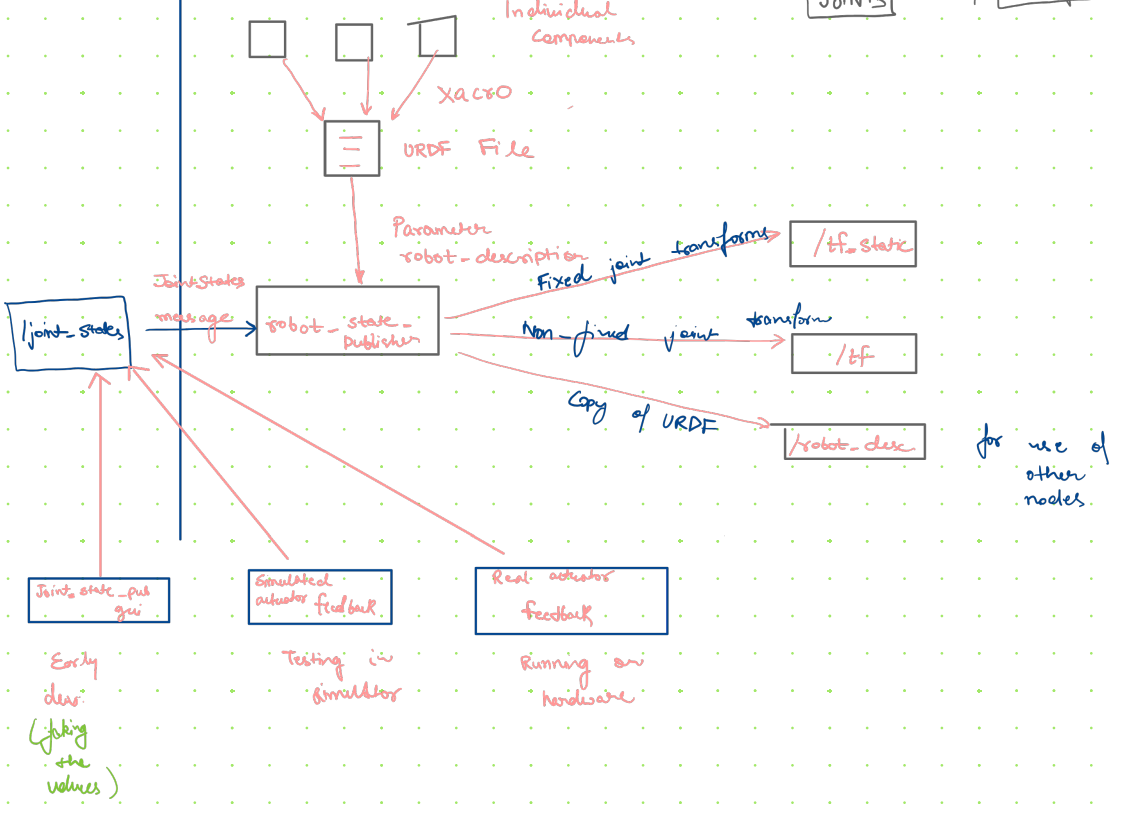
- ros2 run tf2_ros static_transform_publisher ------ world robot1
- " " " " " ------ robot1 robot2

Dynamic transform

- URDF File : Inertia, joint, color

URDF
A tree of ....
LINKS
Connected by
JOINTS

TF
A tree of
Frames
connected by
Transforms



> Add display from rviz , tf & robotModel

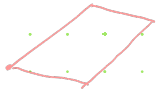
> Debugging with view_frames

- generate the pdf

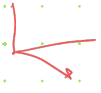
-

Kos-7

URDF

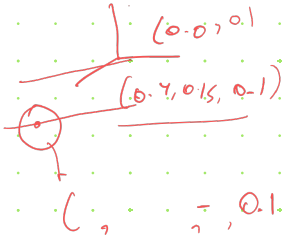


> Break physical obj into separate into components



>> If two parts move differently

>> obj.obj



> ros2 launch urdf_tutorial display.launch.py model:=tb3_waffle.urdf

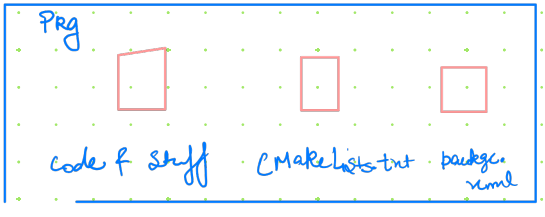
18 Nov | Morgen |

→

✓ Task for today : →

✓ I have (rough) URDF File of physical system, now i am creating testing on turtlebot3, but have to configure for SLAM, Planning, Control.

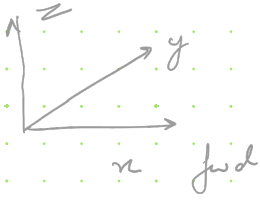
> Simulation environment / Real environment



> creating a common base for working.

---

creating a rough 3D model



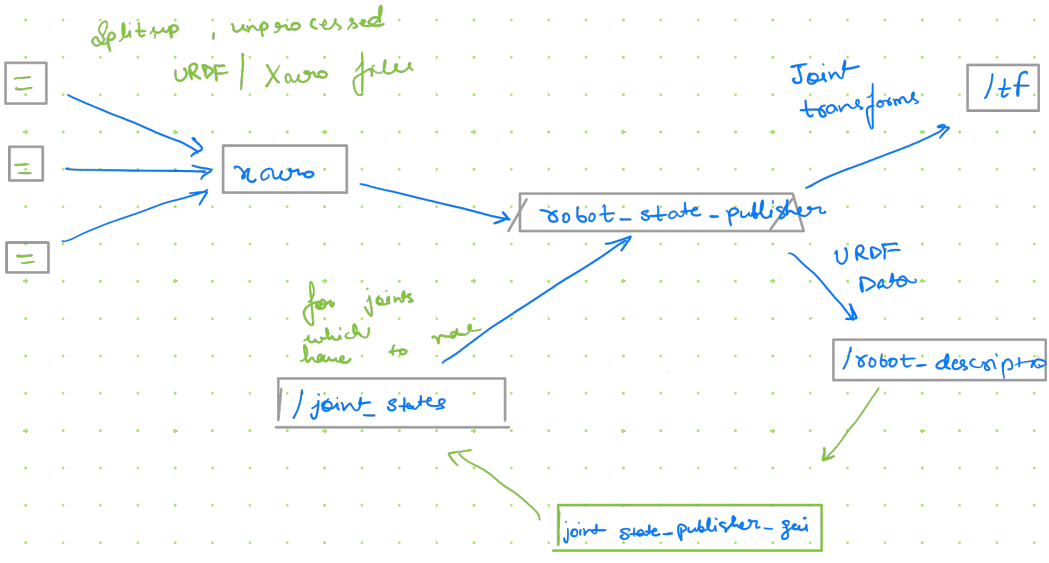


Abb.

/name : Its a topic

Rule

-- symlink-install : if used then you don't have to build xacro file again & again , if no new added

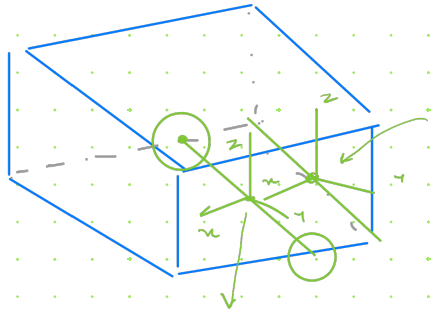
- rosdep : if rviz2 not fucking update

URDF

→ Bot , Lidar , — : different files for everything

Base link choice

It can selected anywhere COM , etc. But for differential robot its better to choose in the center of powering wheels which is also center of rotation



→ when you are going to place box, then its centered around "origin of chassis", which we don't want.

So, we will box → by xyz.

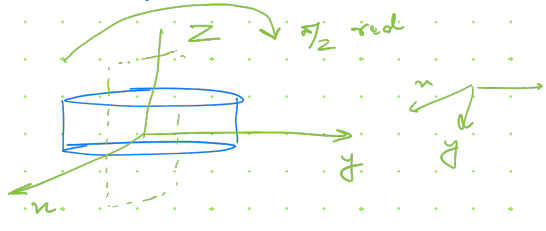
Changes in RVIZ, IN ORDER TO SEE

✓ tf, RobotModel

- change fixed frame → base_link
- In Robot Model → add description topic to → /robot_description

---

In ROS cylinders are by default oriented towards z-direction



but, we want along y-axis

→ Writing URDF, requires clarity in geometry of bot

Error using log

> work on this —, error message on cli, always mention something on launch files, which is not right.

After saving rviz file, we can launch that file —

> rviz2 -d path to the file

> ros2 run joint_state_publisher_gui joint_state_publisher_gui
(for launching fake joint publisher)

How to re-run?

1. Open Cli
2. go to ws
3. > source install/setup.bash
4. > colcon build --symlink-install
5. > ros2 launch my_bot rsp.launch.py
6. In other Cli > ros2 run joint_state_publisher_gui joint_state_publisher_gui
fake joint publishing
7. In other cli > rviz2

Note: opening the rviz file from config doesn't work, not able to publish tf msg of joints

Driving your wheeled robot

✓ ros2 launch my_bot rsp_launch.py use_sim_time:=true

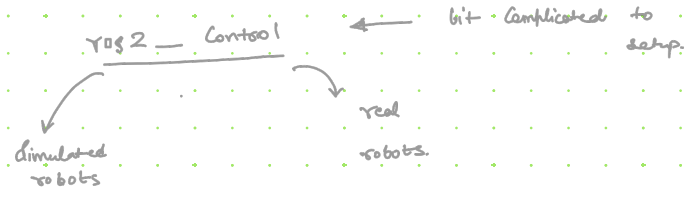
simulation synchronise time

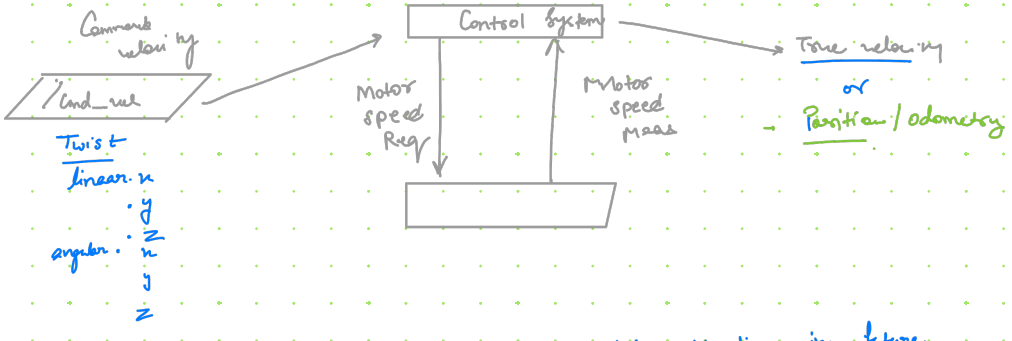
* Gazebo → cmake → not opening
  → uninstall & installed using binaries

launch file for launching in gazebo

> ros2 launch my_bot launch_sim.launch.py

> gazebo path setup is not correct





→ Robot position: estimating the robot estimation in future
Dead reckoning

→ Odometry: Resulting position estimate from dead reckoning

* Always pass full path starting from home → /home/user/.bash →

To launch bot & world →

> ros2 launch my_bot launch_sim.launch.py world:=pwd

How to integrate Rplidar → view results in Rviz

(1)
> setting up fixed-frame → manually →
> setting up topic
> launch file for Rplidar

Use Camera

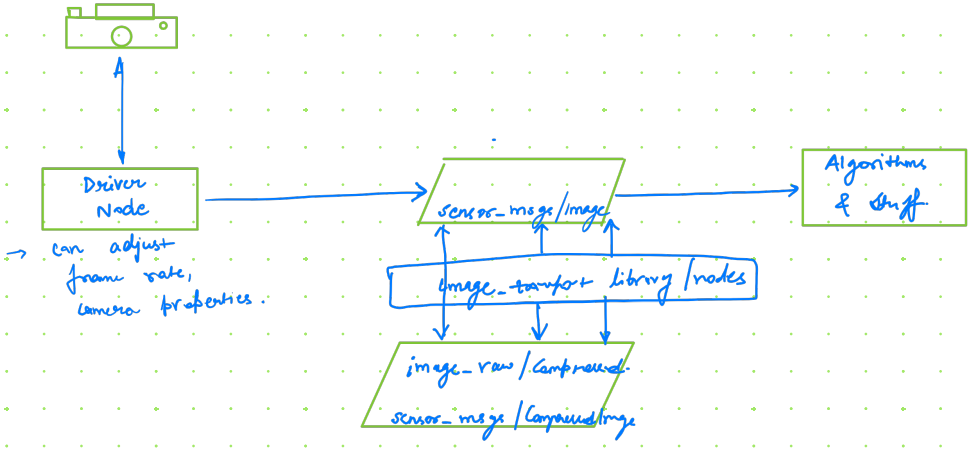
1. Intro 2. Camera → Ros2 3. Simulate virtual camera
4. Real.

1. Compression → png : lossless → exactly same (decompression)
   → jpeg : lossy → ...

focal length : dist of lens from its screen

Horizontal FOV :

$$h_{FOV} = 2\tan^{-1}\left(\frac{\text{sensor-width}}{2 \times \text{focal-length}}\right)$$



Coordinate frame

ROS Body standard

Image standard

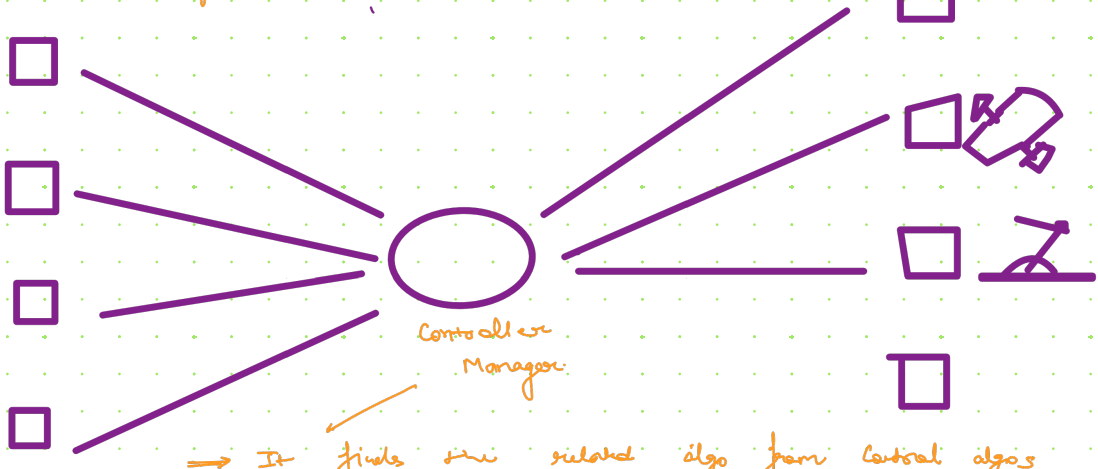
camera_link

camera_link_optical

19th Nov

Hardware / drivers

Control Algorithms



→ It finds the related algo from Control algos and links it to hardware.

→ It uses plugin system.

(things we control) (things we measure)

→ Command interface vs. State interface

- ✓ Resource Manager / Controller Manager { for multiple motors / actuators this offers all command & state interface and exposes it to hardware interface
- ✓ Controllers

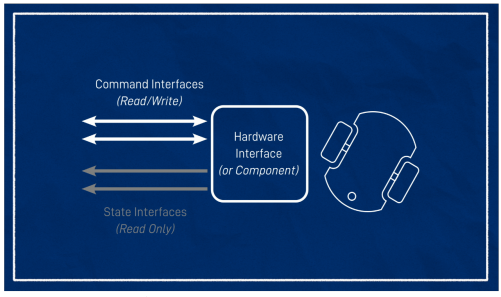
Q. How to write a hardware interface / hardware component for your robot?

Interacting with Controller Manager

1. via ROS Services
2. via ros2 Control CLI tool
3. via specialised nodes / scripts

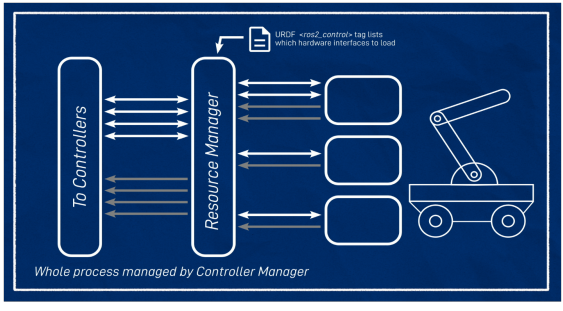
Q. How to write ROS2_ Control file?

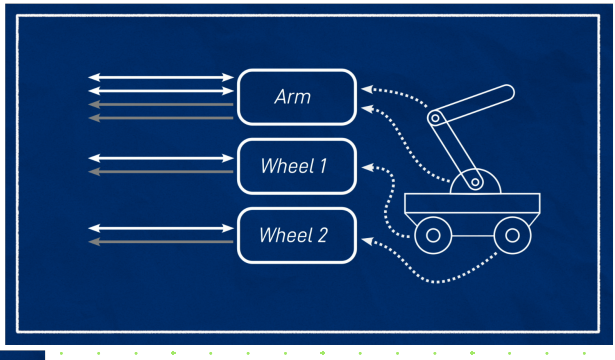
- ✓ plugin name →
- ✓ hardware interface

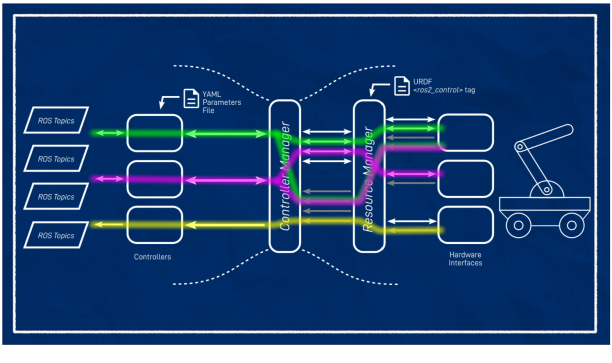


- ✓ command interface is the way through which we control the hardware
- ✓ state interface are used to read the hardware values.

```
dev@dev-PC:~/dev_ws$ ros2 control list_hardware_interfaces
command interfaces
        left_wheel_joint/velocity [unclaimed]
        right_wheel_joint/velocity [unclaimed]
state interfaces
        left_wheel_joint/position
        left_wheel_joint/velocity
        right_wheel_joint/position
        right_wheel_joint/velocity
dev@dev-PC:~/dev_ws$
```







> when not using Gazebo, we will need to run "ros2 run controller_manager ros2_control_node"

> To check for controllers

- ros2 control list_hardware_interfaces
  - ✓ command interfaces
    - left_wheel_joint/velocity [unclaimed]
    - right_wheel_joint/velocity [unclaimed]
  - ✓ state interfaces
    - left_wheel_joint/horizon
    - left_wheel_joint/velocity
    - right_wheel_joint/horizon
    - right_wheel_joint/velo
- ros2 control list_controllers
  only shows controllers which are running
- ros2 run control [tab]
- ros2 screen controller_manager spawner diff_cont

Running things

ROS2_CONTROL EXTRA BITS (for implementing in real bot)

How to launch basic setup of bot with controller & indoor Map?

T1 → dev_ws $ source install/setup.bash

T1 → ros2 launch my_bot launch_sim.launch.py world:= Complete path

T2 → rviz2

T3 → ros2 run teleop_twist_keyboard teleop_twist_keyboard

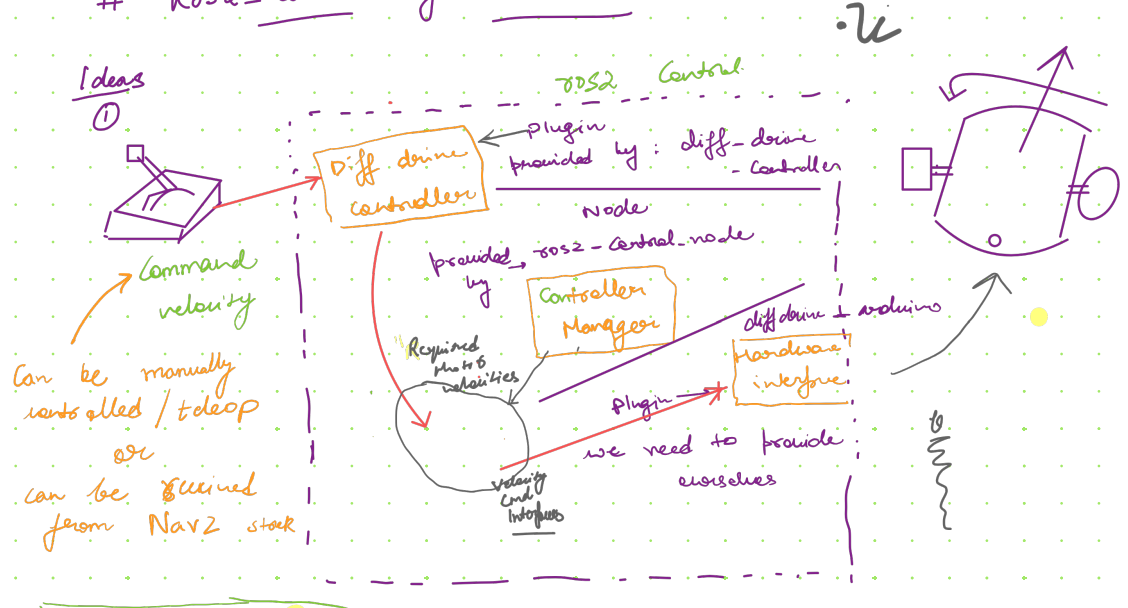
--ros-args -r /cmd_vel:=/diff_cont/cmd_vel_unstamped

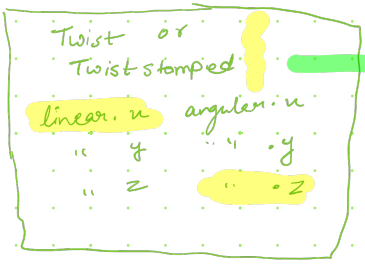
Etc → ros2 topic list

Sol~ • Adjust the update rate very high by creating gazebo_sim.yaml file and then updating launch file

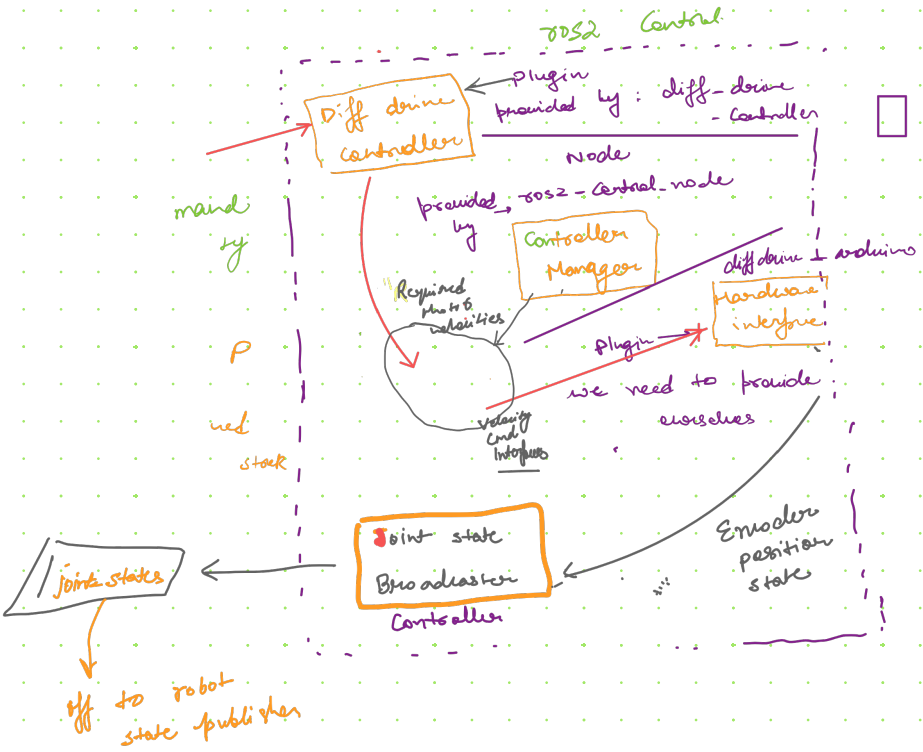
Q. wheel slippage

Ros2_Control for real robot





ros topic here is of type : /twist or



Hardware Interface (diff_drive_arduino)

- ✓ for Ros2 ← something new
- ✓ vs editor remote deploy

✓ Steps for allocating Custom hardware interface

1. update ros2_control.xacro , add parameters

2. update launch file → launch_robot.launch.py

so to call controller manager

pkg : controller_manager

exe : ros2_control_node

param : [ ]

URDF → Controller Manager ← param.yaml

* gazebo_does

add line of robot_description = ( )

and controller_params_file = my_controller.yaml

3. Time_delay or Timer → for delaying launch of controller manager

4. spawn problem delay.

5. launch rviz2 from configuration { TF, Robot Model, LaserScan, Camera

6. teleop (Moving in rviz also)

7. encoder check → 360°

my_bot

8. Camera check

- ✓ source ~/.bashrc
- ✓ ros2 launch my_bot camera_launch.py

9. LiDAR

10. odometry issue → enc_count_per_rev = $\frac{\text{change}}{\text{}}$

- ✓ Game Controller
- ✓ Camera feed on phone

Controller plugin : which Ackermann based steering is good for the hardware work?

TEB RPP MPPI VP

Controller	Dynamic obstacle Handling	Path Accuracy	Speed Handling	computational demand	Comment
TEB	Excellent	Moderate	Moderate	High	dynamic obstacle avoidance & smooth navigation
RPP	Limited	Excellent	Moderate	Low	Exact path following in static environment

Controller	Dynamic obstacle Handling	Path Accuracy	Speed Handling	computational demand	Comment
MPPI	Excell	Exc.	high	very high	Complex dynamic environment & precise Controle.
VP	Mod.	Mod.	Exc.	Mod.	High speed path tracking in open environment