easymov
robotics

CONTRÔLE ET SIMULATION DE L'IPCAR

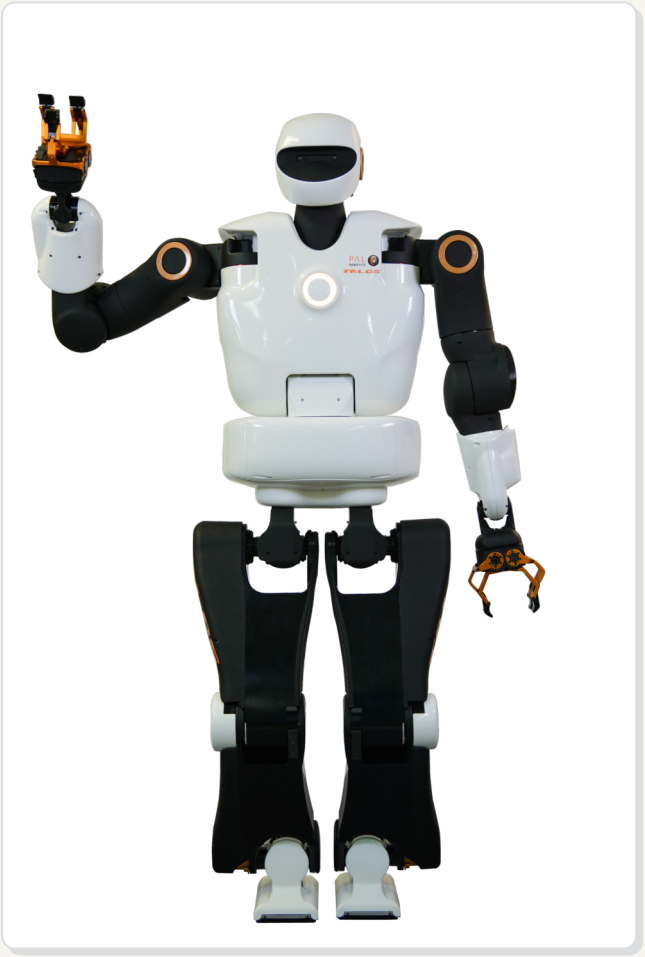
AVEC ROS ET GAZEBO

JOURNÉE TECHNIQUE - ROBOTS MOBILES TERRESTRES - 2018

Gérald Lelong

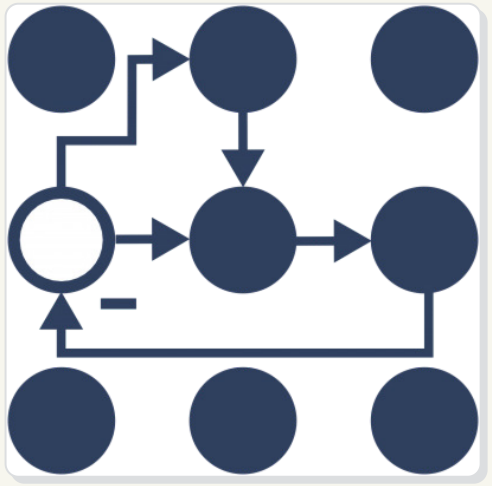
IPCAR

- commande en angle de braquage du train avant
- commande en vitesse du train arrière
- retour d'état par codeurs incrémentaux



TALOS - 32 DoF

ROS_CONTROL



POURQUOI ?

Pour découpler l'implémentation du driver d'un composant du contrôle de ce composant

ROS_CONTROL

COMMENT ?

Resource : une *resource* décrit une donnée capteur ou un élément commandable du robot

Handle : un *handle* permet d'interagir avec une *resource*

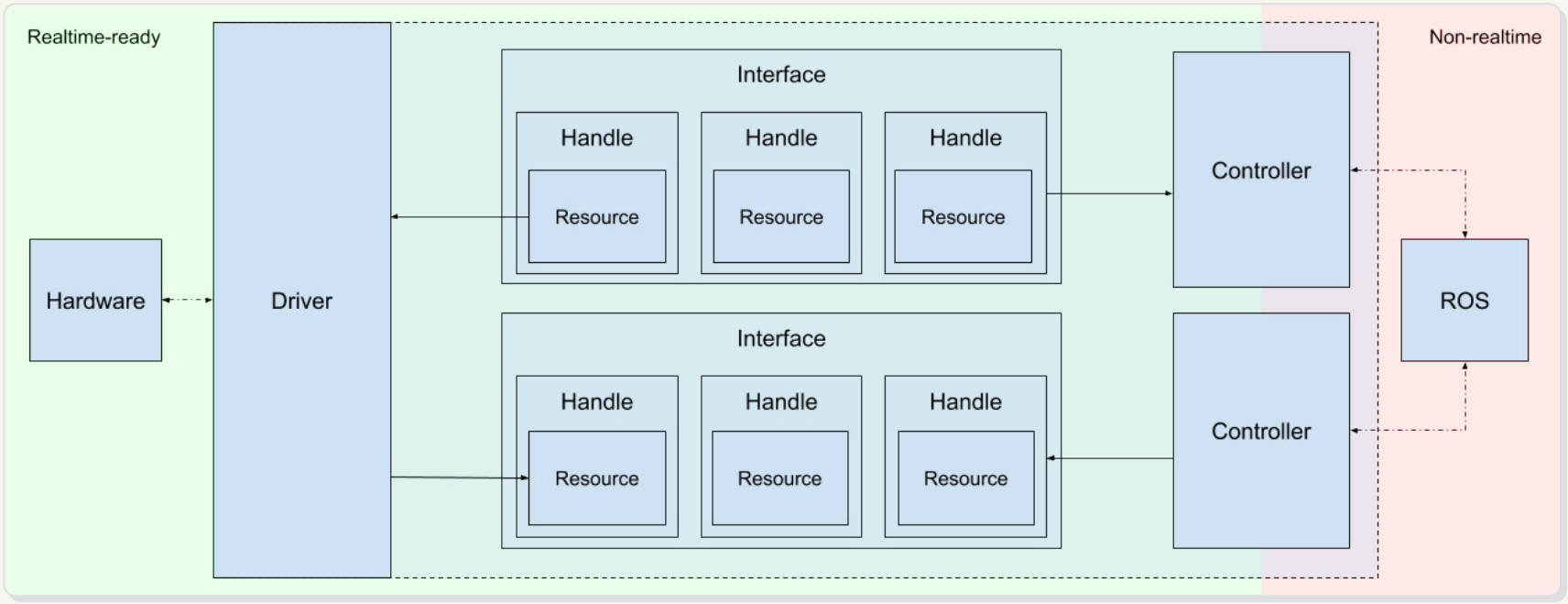
Interface : une *interface* regroupe les *handles* du même type et permet de les réclamer

Driver : le *driver* crée des *resources*, leur associe un *handle* et les enregistre dans les *interfaces* appropriées

Controller : un *controller* utilise des *handles* qu'il récupère depuis les *interfaces* pour interagir avec des *resources*

ROS_CONTROL

INTERACTIONS



ROS_CONTROL

EXEMPLE

Resource : la roue gauche du robot

Handle : *JointHandle* permet d'affecter une consigne à une articulation

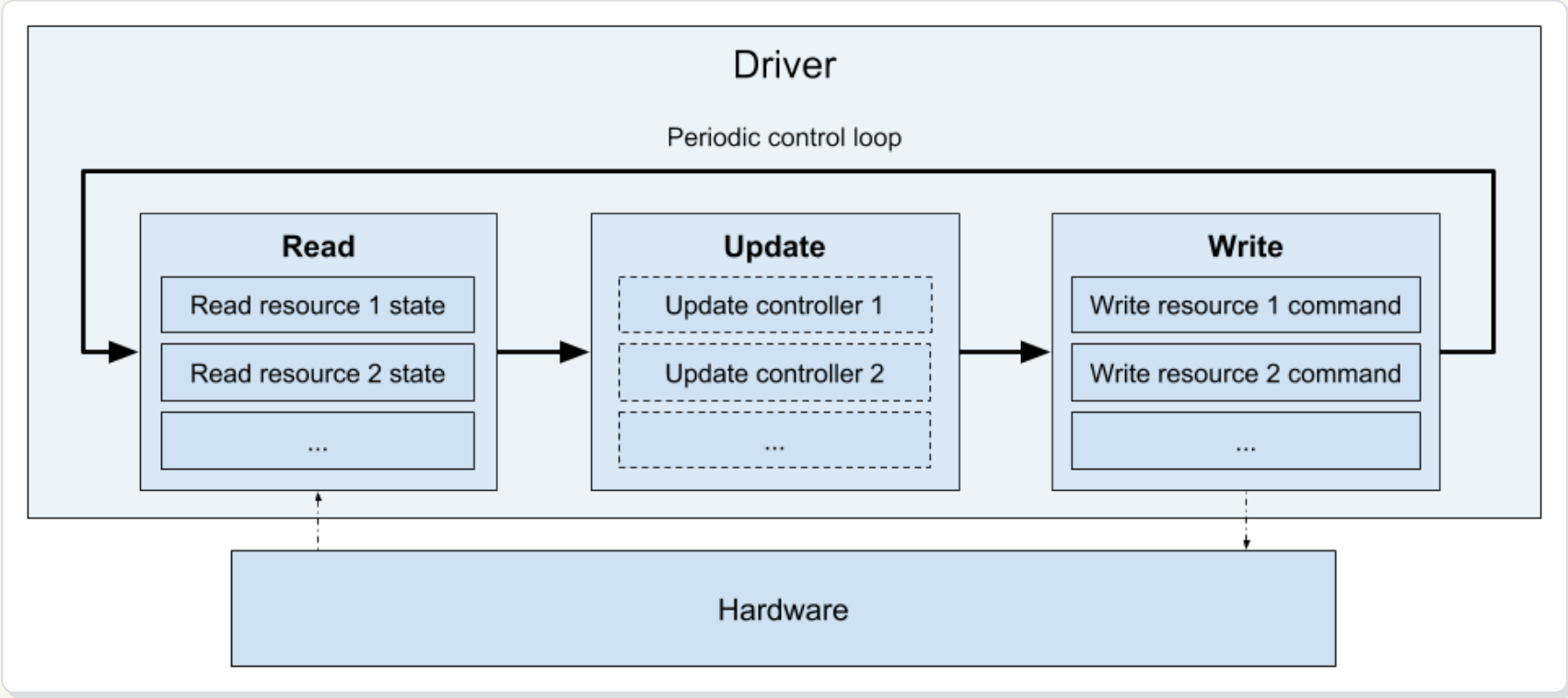
Interface : *VelocityJointInterface* regroupe les articulations commandables en vitesse

Driver : *lpcar driver* lit l'état de mon articulation sur le bus CAN et y écrit la consigne affectée

Controller : *ackermann controller* assigne à la roue gauche la vitesse demandée par l'utilisateur et publie l'odométrie

DRIVER

BOUCLE DE CONTRÔLE



DRIVER

JOINTS

Les articulations d'un robot sont regroupées sous le concept de joints.

Un *joint* possède un état (*state*) et peut être commandé :

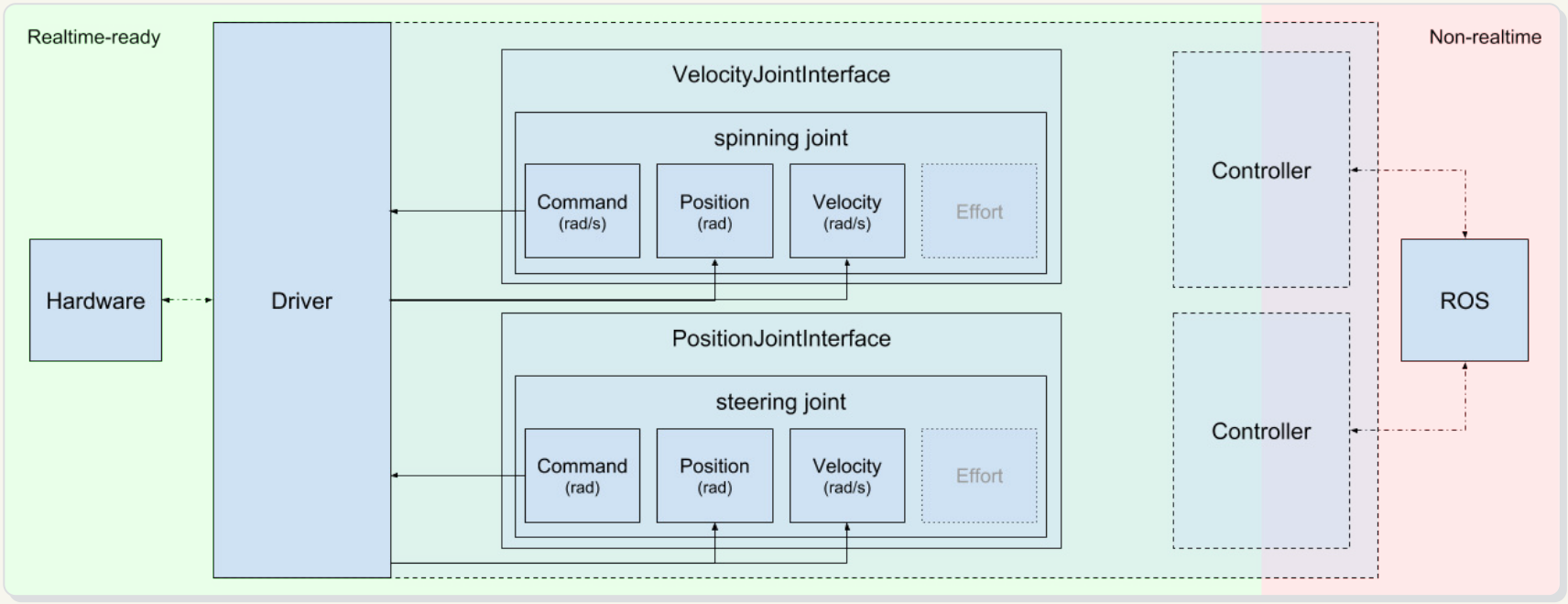
- en position
- en vitesse
- en couple

Chacun de ces modes de commande correspond à un type d'*interface* :

- *PositionJointInterface*
- *VelocityJointInterface*
- *EffortJointInterface*

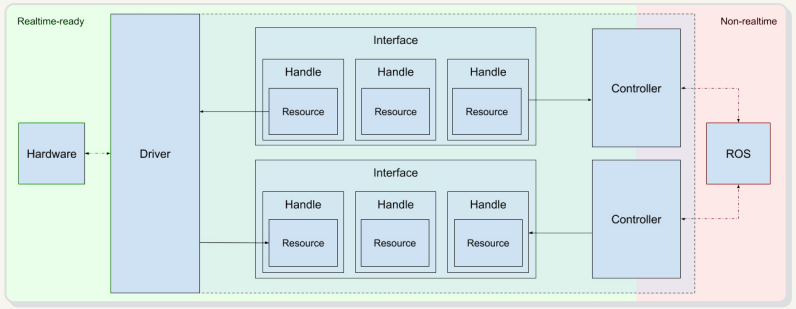
DRIVER

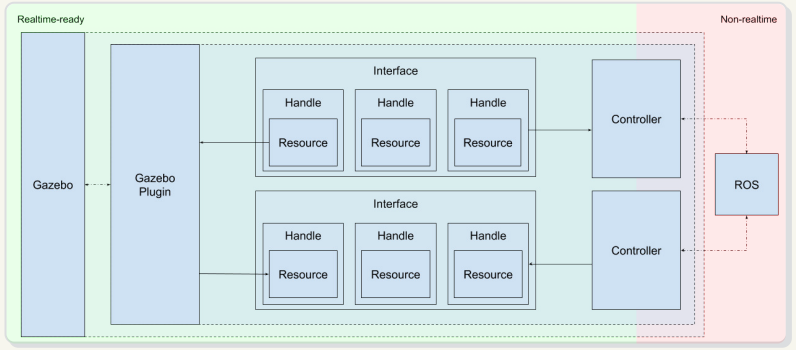
IPCAR




GAZEBO

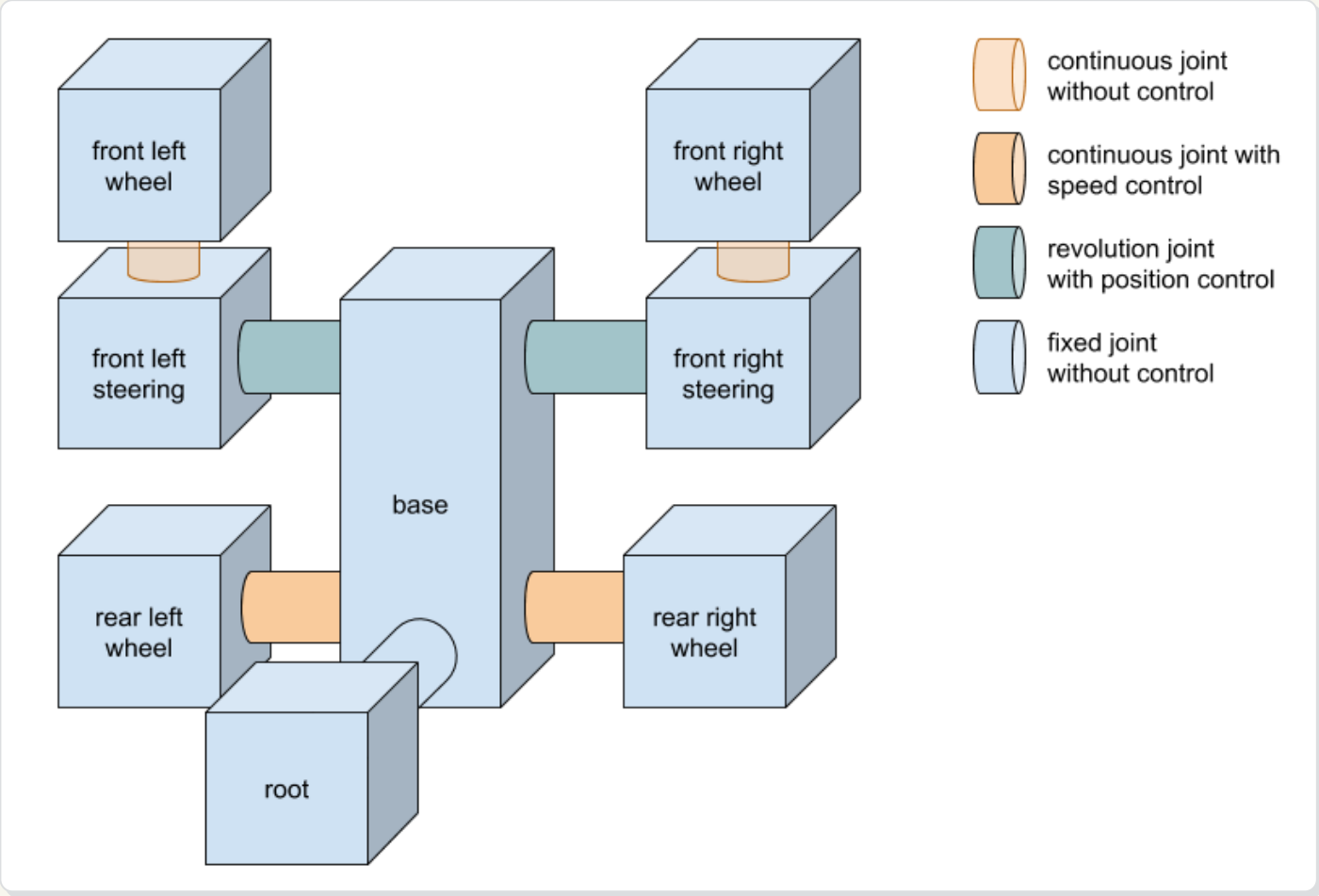
GAZEBO


Realtime-ready
Non-realtime
Hardware
Driver
Interface
Handle
Resource
Handle
Resource
Handle
Resource
Interface
Handle
Resource
Handle
Resource
Handle
Resource
Controller
Controller
ROS


Realtime-ready
Non-realtime
Gazebo
Gazebo Plugin
Interface
Handle
Resource
Handle
Resource
Handle
Resource
Interface
Handle
Resource
Handle
Resource
Handle
Resource
Controller
Controller
ROS

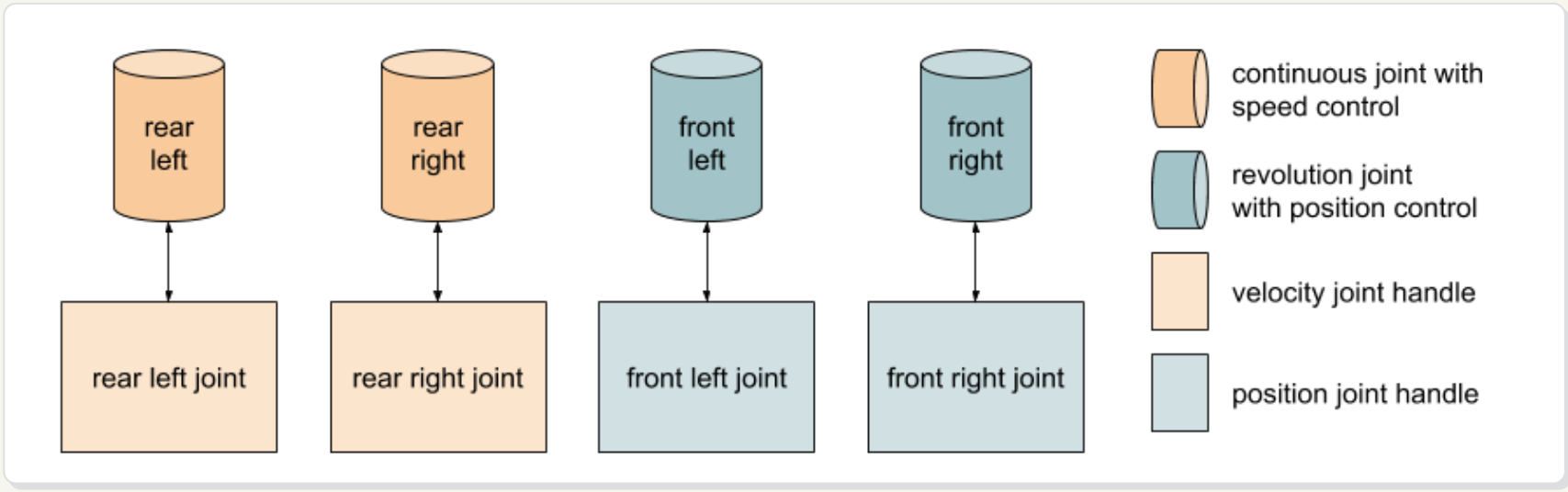
GAZEBO

URDF


front left wheel
front right wheel
front left steering
front right steering
base
rear left wheel
rear right wheel
root
continuous joint without control
continuous joint with speed control
revolution joint with position control
fixed joint without control

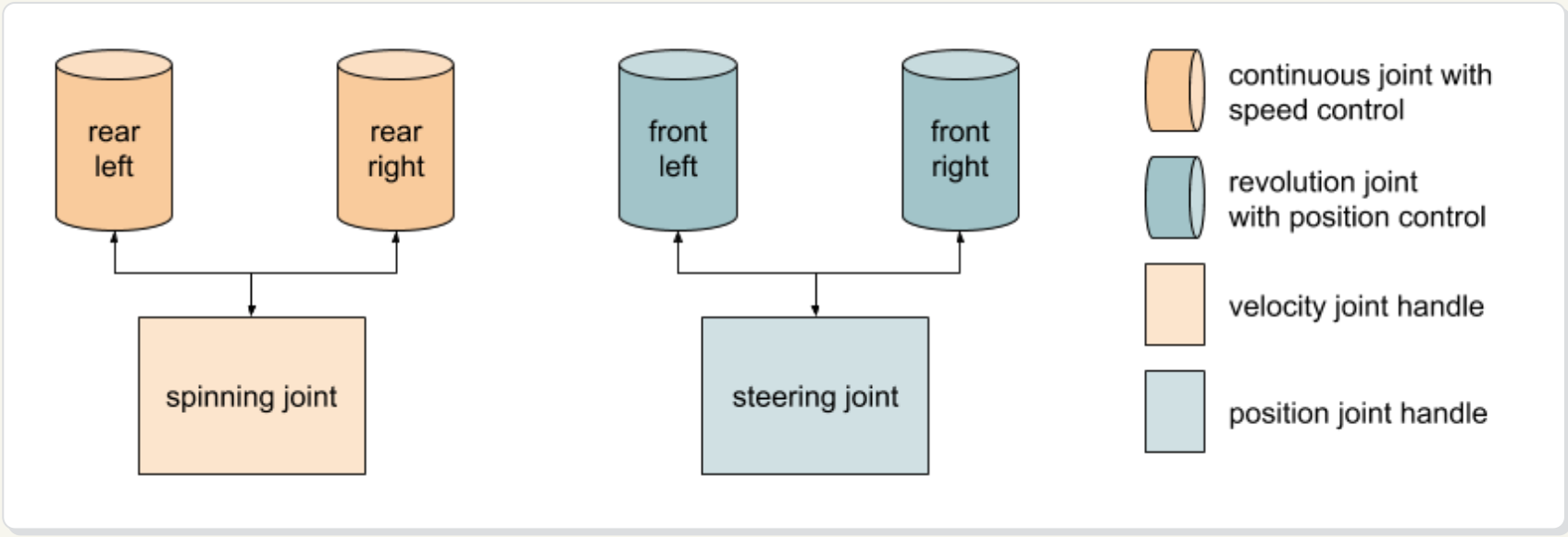
GAZEBO

DEFAULT *ROS_CONTROL* PLUGIN



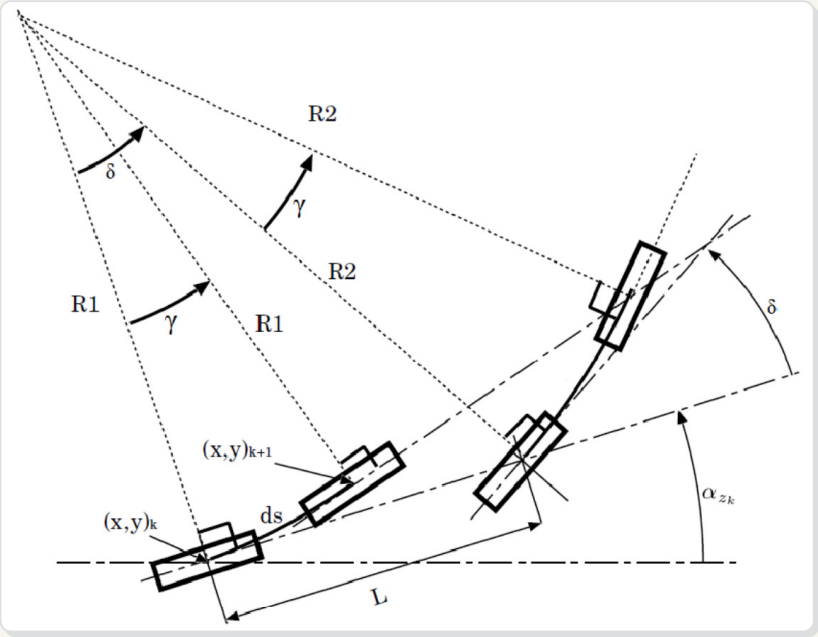
GAZEBO

CUSTOM *ROS_CONTROL* PLUGIN



CONTROLLERS

MODÈLE D'ÉVOLUTION

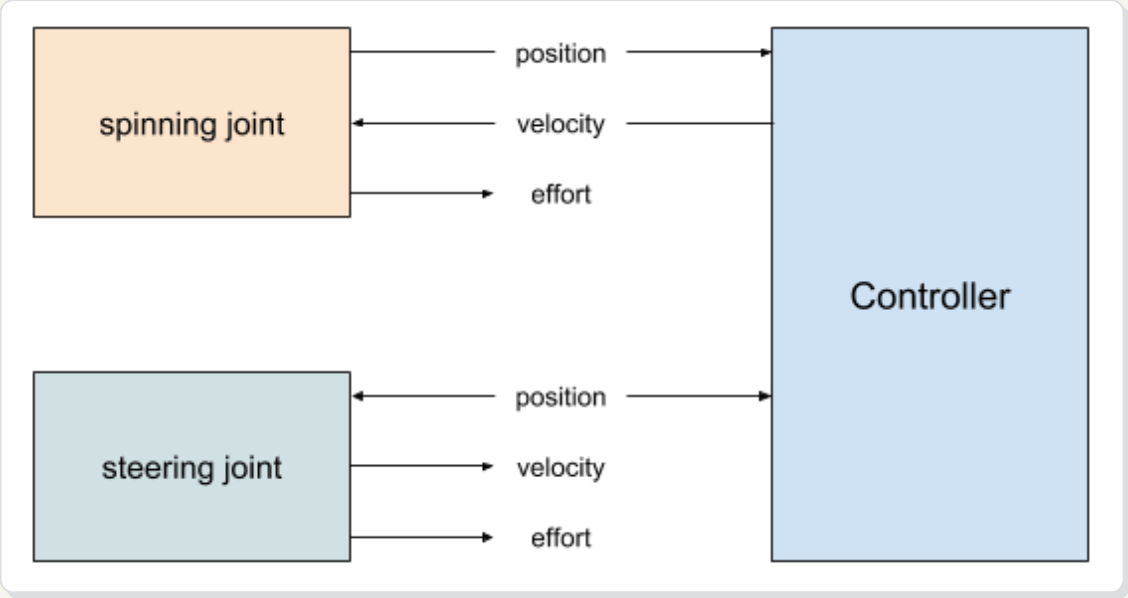


$$
\begin{aligned}
c &= 2R1\sin(\gamma/2) \\
x_{k+1} &= x_k + c\cos(\alpha_{z_k} + \gamma/2) \\
y_{k+1} &= y_k + c\sin(\alpha_{z_k} + \gamma/2) \\
\alpha_{z_{k+1}} &= \alpha_{z_k} + \gamma
\end{aligned}
$$

Thomas Feraud - 2013

CONTROLLERS

JOINTS



CONTROLLERS

MESSAGES

geometry_msgs/Twist

```
geometry_msgs/Vector3 linear
  float64 x
  float64 y
  float64 z
geometry_msgs/Vector3 angular
  float64 x
  float64 y
  float64 z
```

ackermann_msgs/AckermannDrive

```
float32 steering_angle
float32 steering_angle_velocity
float32 speed
float32 acceleration
float32 jerk
```

CONTROLLERS

TIMEOUT

CONTROLLERS

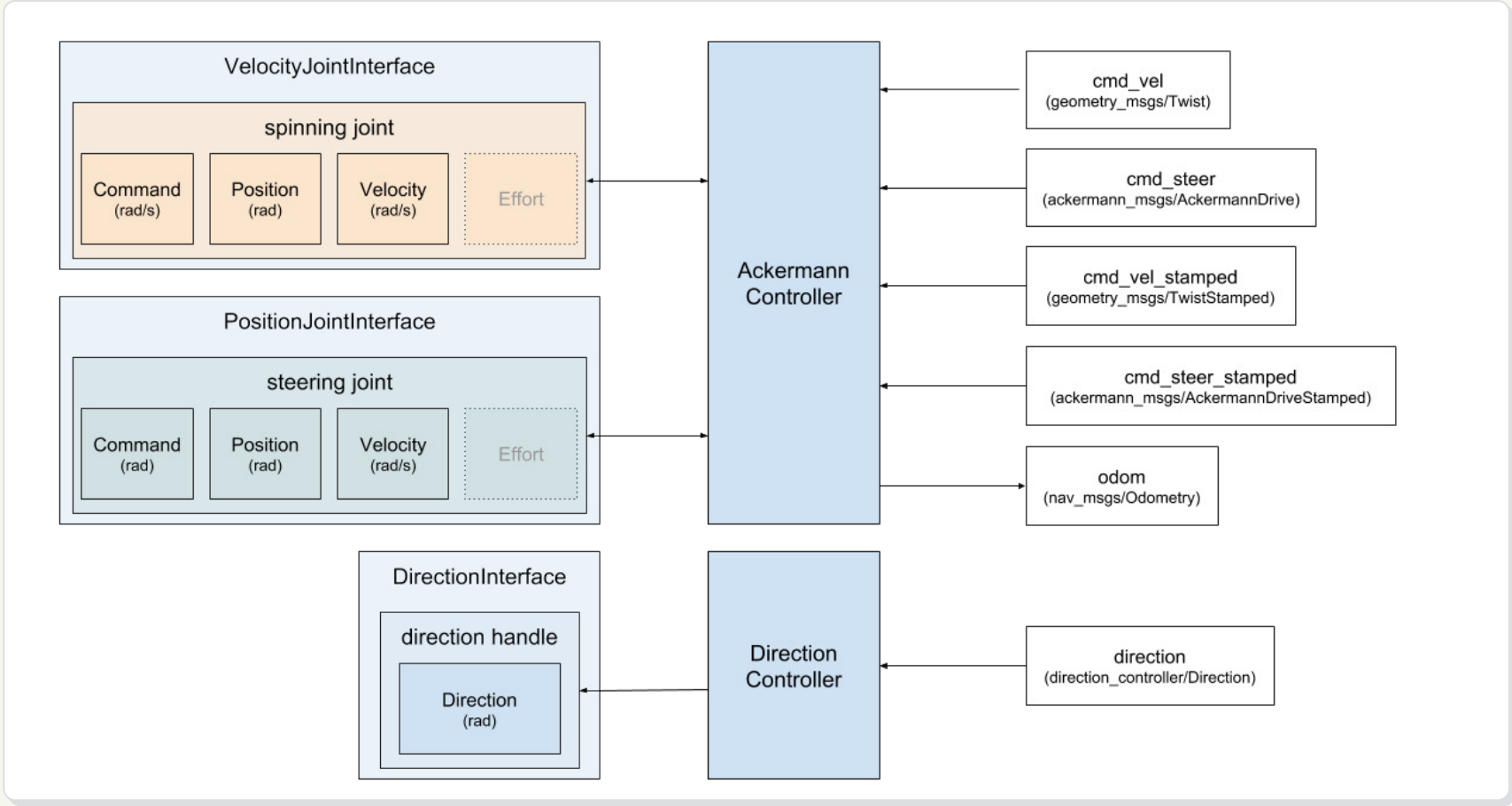
DIRECTION

direction_controller/Direction

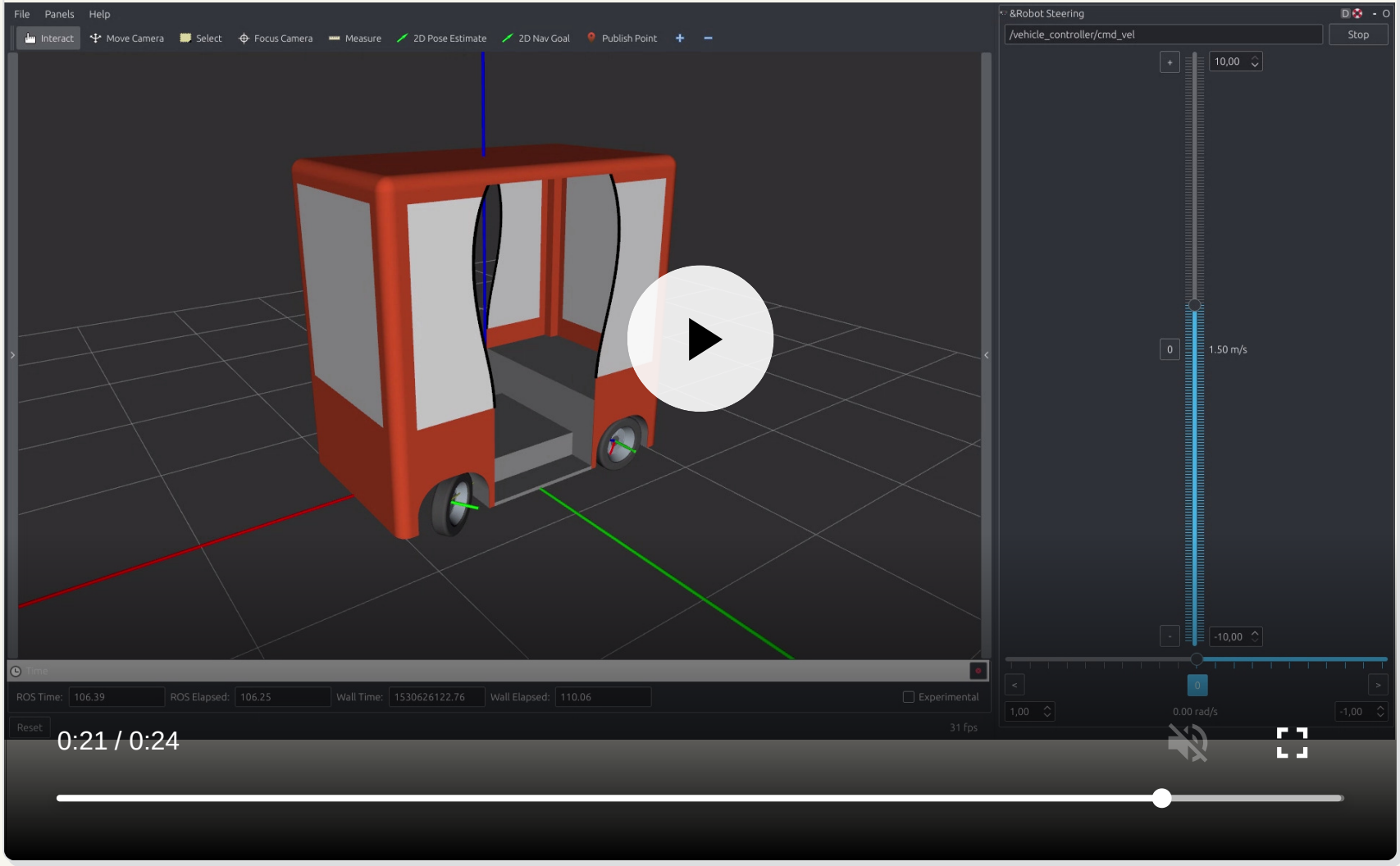
```
uint8 NO_DIRECTION=0
uint8 FORWARD=1
uint8 BACKWARD=2

uint8 direction
```

CONTROLLERS

IPCAR



RÉSULTATS


File
Panels
Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
&Robot Steering
/vehicle_controller/cmd_vel
Stop
10,00
1.50 m/s
-10,00
ROS Time: 106.39
ROS Elapsed: 106.25
Wall Time: 1530626122.76
Wall Elapsed: 110.06
Experimental
Reset
31 fps
0.00 rad/s
0:21 / 0:24

BÉNÉFICES

- maintenance des *drivers* et des *controllers* séparés
- *controllers* indépendants du matériel (donc réutilisables)
- passage de la simulation au réel sans modifier une ligne de code
- realtime-ready

- chargement de *controllers* à chaud
- méthode générique pour implémenter un *driver*
- *drivers* génériques (*CANopen DS402*)
- *controllers* génériques (*diff_drive_controller*)

- easymov.fr
- @easymovrobotics
- gitlab.com/easymov