Vehicle interface

What is the vehicle interface?

The purpose of the vehicle interface package is to convert the control messages calculated from the autoware into a form that the vehicle can understand and transmit them to the vehicle (CAN, Serial message, etc.) and to decode the data coming from the vehicle and publish it through the ROS 2 topics that the autoware expects. So, we can say that the purposes of the vehicle_interface are:

1. Converting Autoware control commands to a vehicle-specific format. Example control commands of autoware:
    - lateral controls: steering tire angle, steering tire rotation rate
    - longitudinal controls: speed, acceleration, jerk
2. Converting vehicle status information in a vehicle-specific format to Autoware messages.

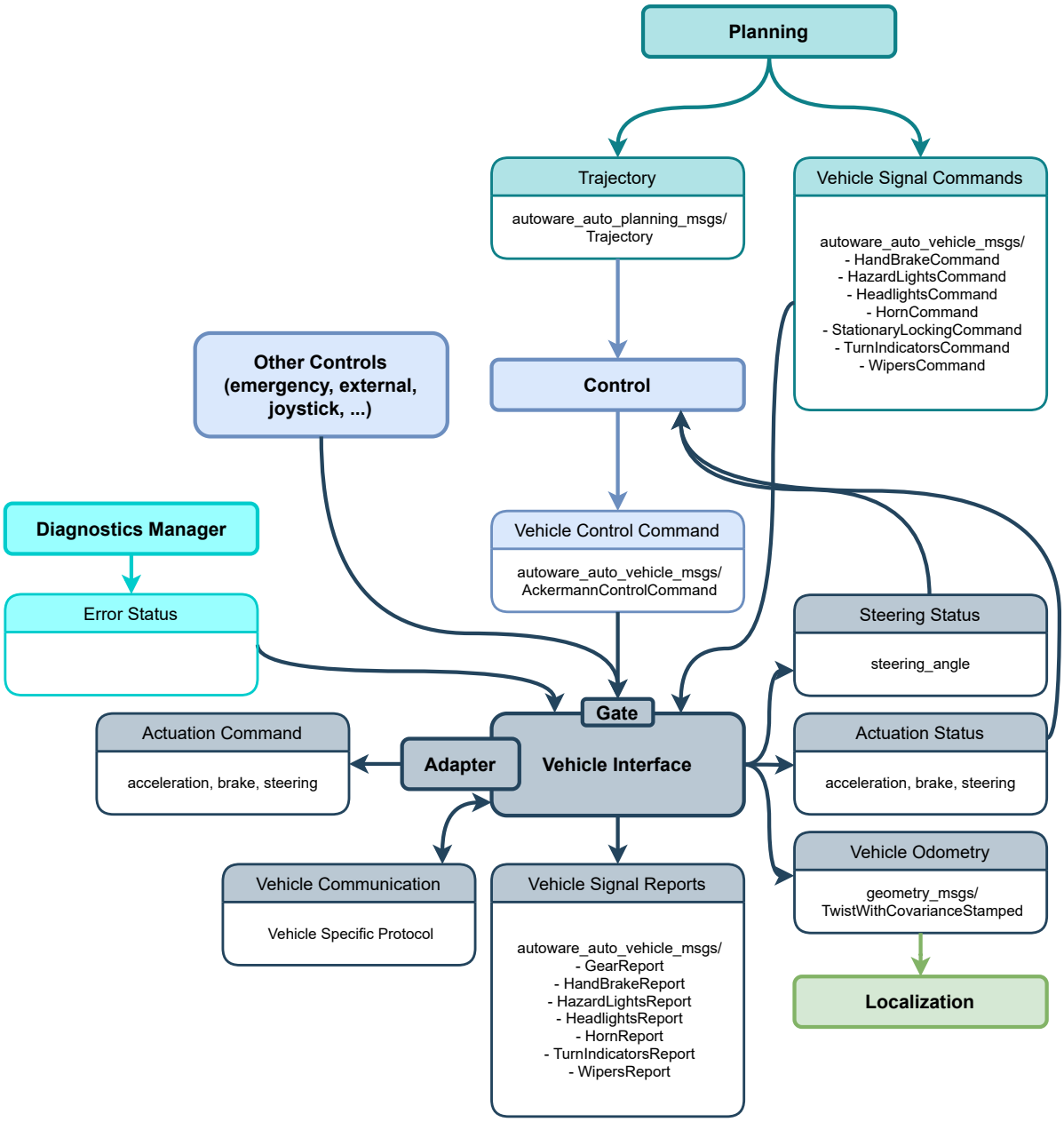
Autoware publishes control commands such as:

- Velocity control
- Steering control
- Car light commands
- etc.

Then, the vehicle interface converts these commands into actuation such like:

- Motor and brake activation
- Steering-wheel operation
- Lighting control
- etc.

So think of the vehicle interface as a module that runs the vehicle's control device to realize the input commands provided by Autoware.



An example of inputs and outputs for vehicle interface

There are two types of interfaces for controlling your own vehicle:

1. Target steering and target velocity/acceleration interface. (It will be named as Type A for this document)
2. Generalized target command interface (i.e., accel/brake pedal, steering torque). (It will be named as Type B for this document)

For Type A, where the control interface encompasses target steering and target velocity/acceleration.

- So, at this type, speed control is occurred by desired velocity or acceleration.
- Steering control is occurred by desired steering angle and/or steering angle velocity.

On the other hand, Type B, characterized by a generalized target command interface (e.g., accel/brake pedal, steering torque), introduces a more dynamic and adaptable control scheme. In this configuration, the vehicle controlled to direct input from autoware actuation commands which output of the `raw_vehicle_cmd_converter`, allowing for a more intuitive and human-like driving experience.

If you use your own vehicle like this way (i.e., controlling gas and brake pedal), you need to use raw_vehicle_cmd_converter package to convert autoware output control cmd to brake, gas and steer map. In order to do that, you will need brake, gas and steering calibration. So, you can get the calibration with using accel_brake_map_calibrator package. Please follow the steps for calibration your vehicle actuation.

The choice between these control interfaces profoundly influences the design and development process. If you are planning to use Type A, then you will control velocity or acceleration over drive by-wire systems. If type B is more suitable for your implementation, then you will need to control your vehicle's brake and gas pedal.

Communication between the vehicle interface and your vehicle's control device

- If you are planning to drive by autoware with your vehicle, then your vehicle must satisfy some requirements:

Type A	Type B
Your vehicle can be controlled in longitudinal direction by the target velocity or acceleration.	Your vehicle can be controlled in longitudinal direction by the specific target commands. (i.e., brake and gas pedal)
Your vehicle can be controlled in lateral direction by the target steering angle. (Optionally, you can use steering rate as well)	Your vehicle can be controlled in lateral direction by the specific target commands. (i.e., steering torque)
Your vehicle must provide velocity or acceleration information and steering	Your vehicle must provide velocity or acceleration information and steering

Type A	Type B
information which is described at the vehicle status topics above.	information which is described at the vehicle status topics above.

- You can also use mixed Type A and Type B, for example, you want to use lateral controlling with a steering angle and longitudinal controlling with specific targets. In order to do that, you must subscribe `/control/command/control_cmd` for getting steering information, and you must subscribe `/control/command/actuation_cmd` for gas and brake pedal actuation commands. Then, you must handle these messages on your own vehicle_interface design.
- You must adopt your vehicle low-level controller to run with autoware.
- Your vehicle can be controlled by the target shift mode and also needs to provide Autoware with shift information.
- If you are using CAN communication on your vehicle, please refer to ros2_socketcan package by Autoware.
- If you are using Serial communication on your vehicle, you can look at serial-port.