THE AUTOWARE FOUNDATION

Autoware Core: 自動運転のための安定したROS 2ベース基盤

Ryohsuke Mitsudome

2025/09/09

What is Autoware?

THE AUTOWARE FOUNDATION

The first all-in-one open source software for autonomous driving

- Originally created in 2015 at Nagoya University
- Based on the open-source ROS middleware
- Autoware is:
  - Complete AD software stack
  - Independent of vehicle type or electronic hardware
  - Governed by an independent foundation
  - Completely open-source (licensed under Apache License 2.0)

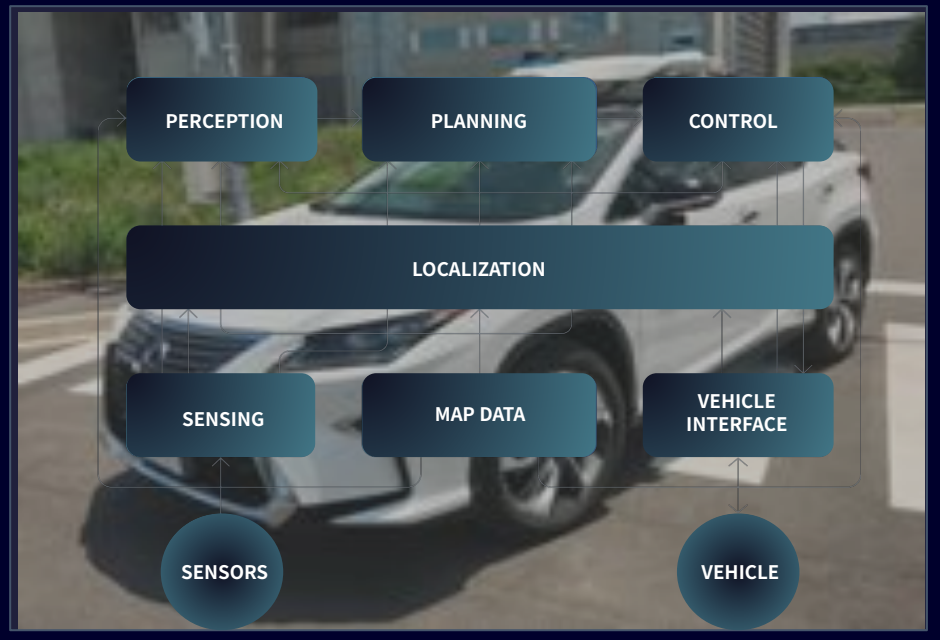
What is Autoware Core?

THE AUTOWARE FOUNDATION

Base Platform for all Autoware Systems

Defines core interface for Autonomous Driving Components

Provides fundamental implementation for Autonomous driving



Past Distributions of Autoware

THE AUTOWARE FOUNDATION

2015 ~ 2020

+ ROS1
+ various packages combined to realize AD stack

- No architecture design
- limited documents and test codes

2018 ~ 2022

+ ROS2
+ rewriting the code from scratch
+ Defining target use case (AVP, cargo delivery, etc)
+ Writing design documents and test codes

- High entry barriers for new engineers
- Making large change is difficult
- no platforms for experimental packages

Autoware – Core/Universe

THE AUTOWARE FOUNDATION

Autoware Core

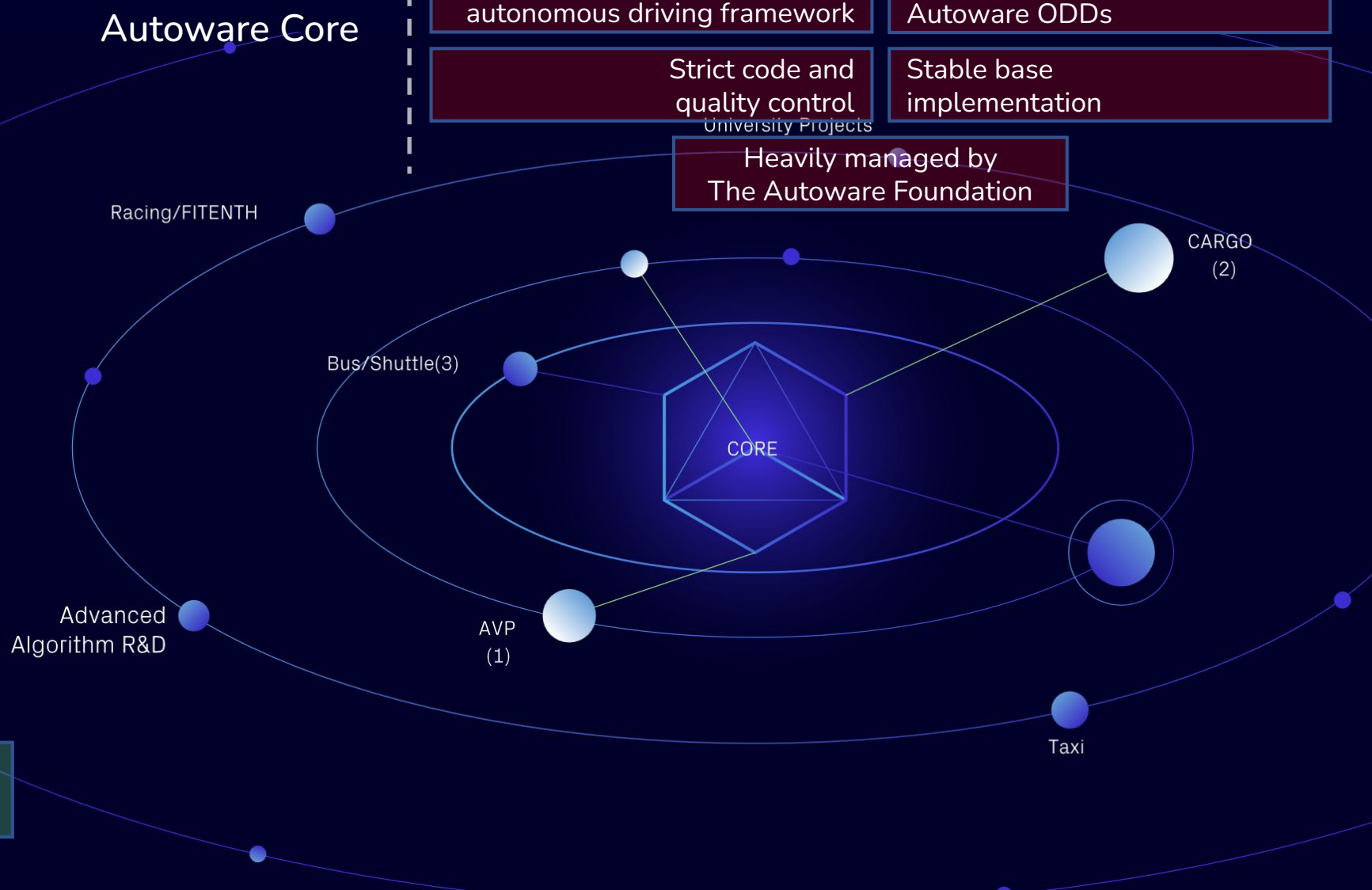
- Complete end-to-end autonomous driving framework
- Supports all current Autoware ODDs
- Strict code and quality control
- Stable base implementation

University Projects

Heavily managed by The Autoware Foundation

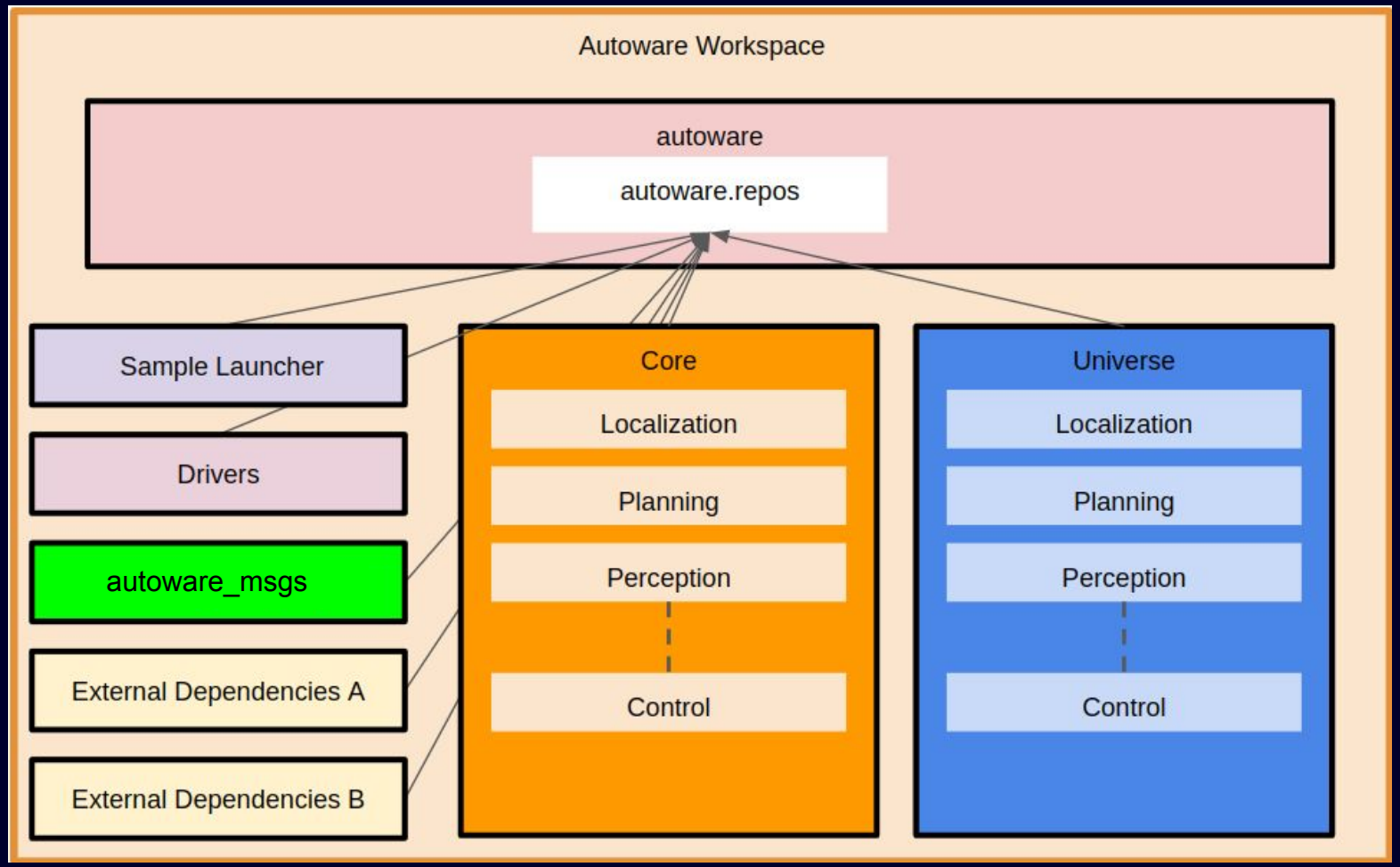
Autoware Universe

- Additional packages built on top of Autoware Core
- Extends Autoware functionality beyond Autoware ODDs
- Dependent on Autoware Core functionality
- Enables quick experimentation and prototype testing
- Managed by the Autoware Community



Autoware – Core/Universe Repository Structure

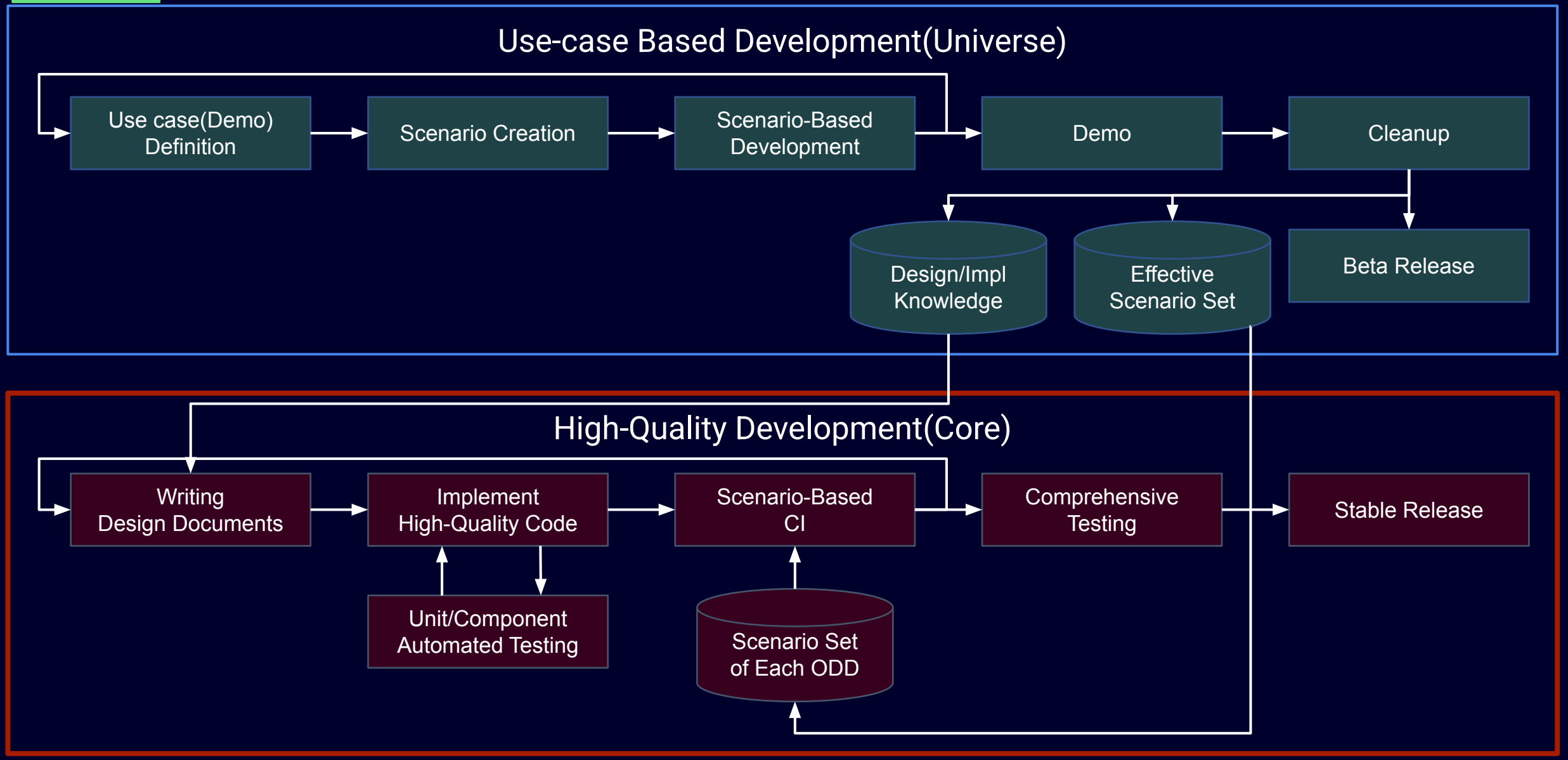
THE AUTOWARE FOUNDATION



🔗 GitHub https://github.com/autowarefoundation/autoware
🔗 Documentation https://autowarefoundation.github.io/autoware-documentation/main/

Autoware – Core/Universe Development

THE AUTOWARE FOUNDATION



Autoware High Level ODD Roadmap

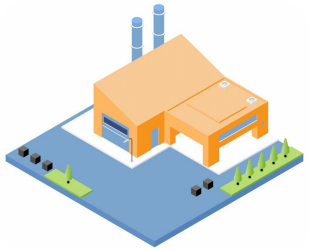
THE AUTOWARE FOUNDATION

2020

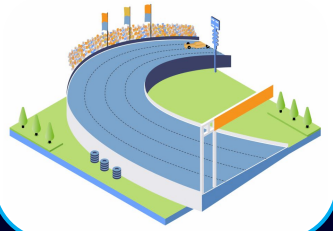
Autonomous Valet Parking (AVP) support in Autoware.Auto

2021

Cargo Delivery support in Autoware.Auto



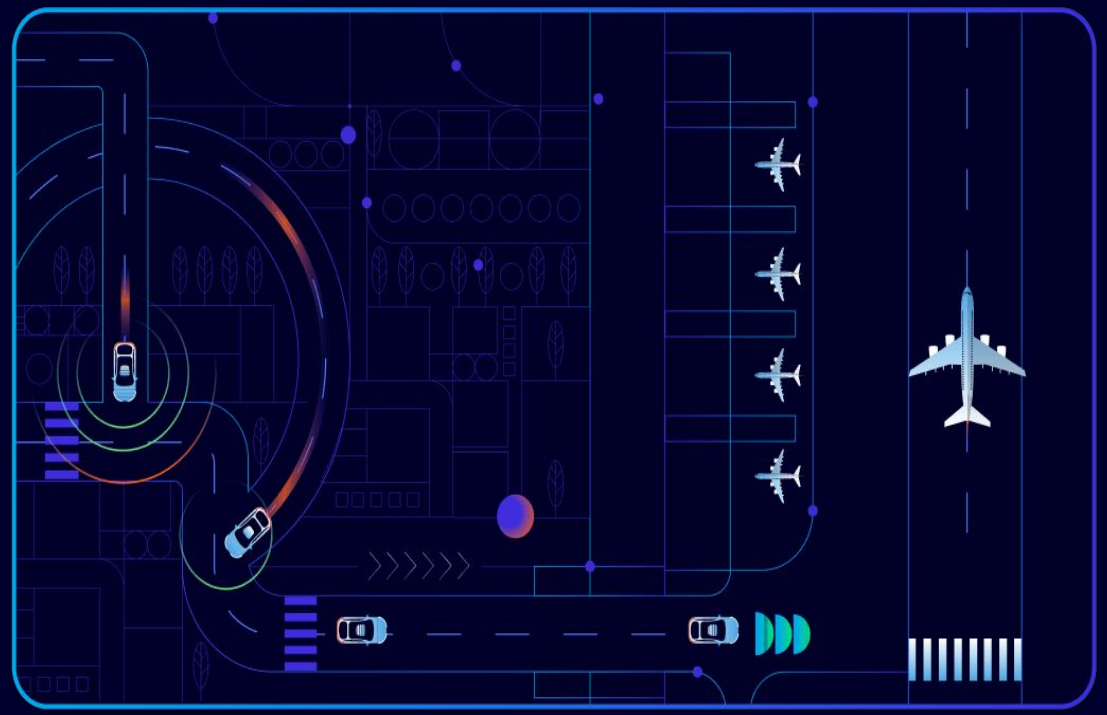
Autonomous Racing
Autoware-based package for the IAC and F1Tenth



2022 - 2023

Autonomous Bus ODD support in Autoware Universe

2024 (and onwards)



Integrating dense urban areas, highways and final service destinations to offer a full self-driving experience

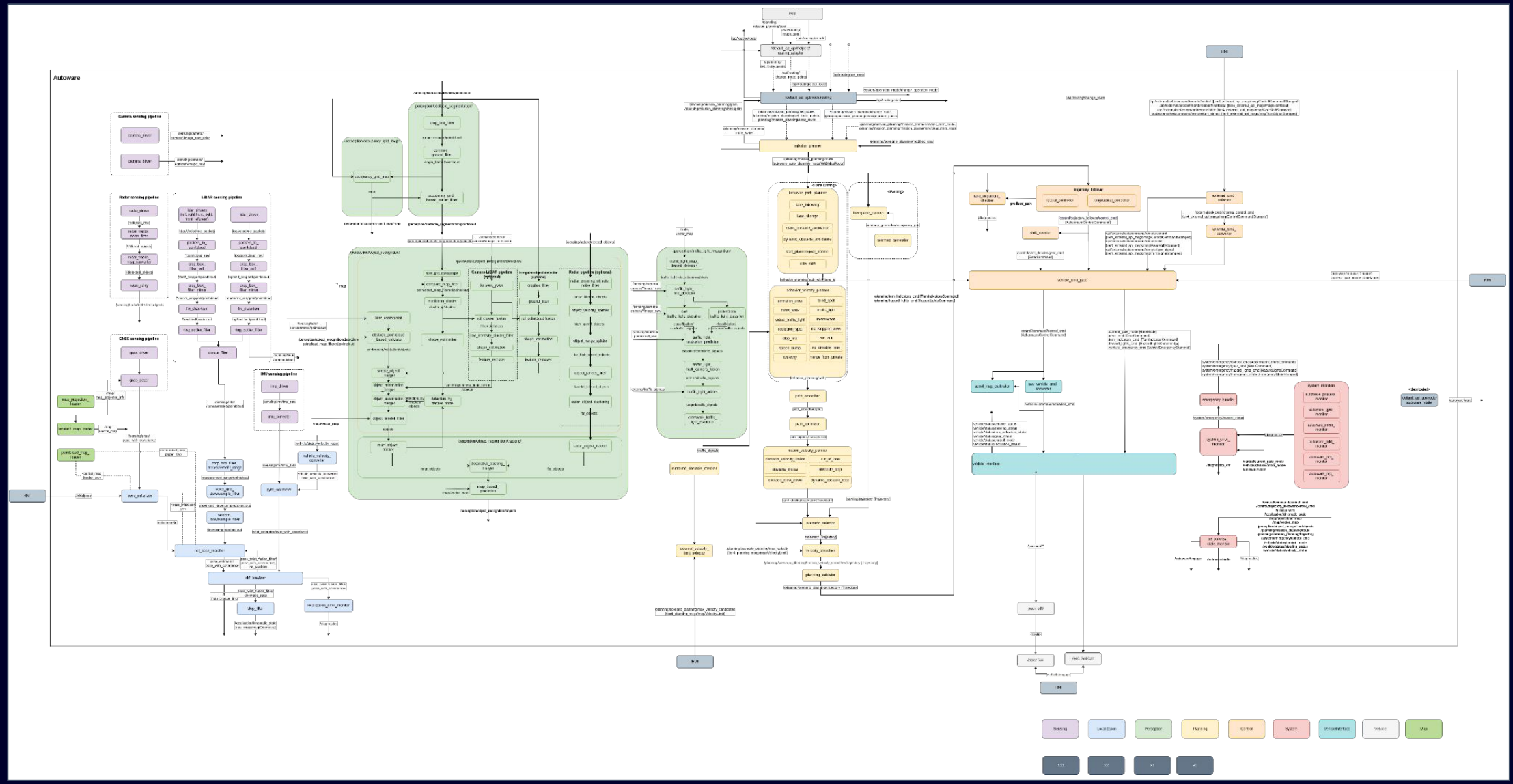
Port essential packages from Bus ODD to Autoware Core

Bus ODD Demo with Autoware Universe

THE AUTOWARE FOUNDATION

Now, Porting Codes to Autoware Core...?

THE AUTOWARE FOUNDATION

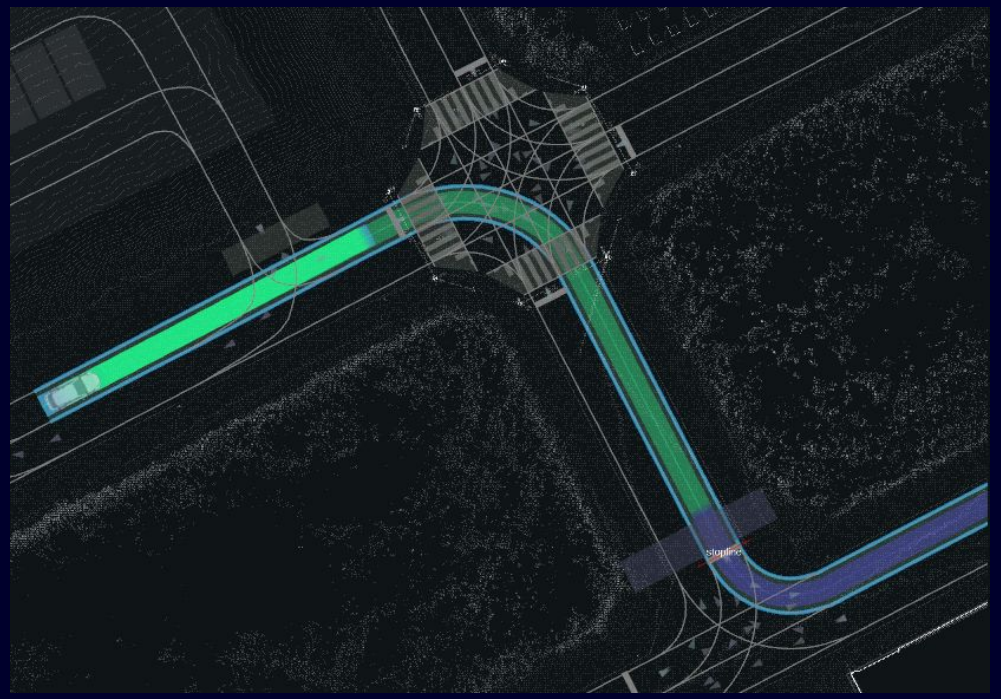


Define Scope

THE AUTOWARE FOUNDATION

Extract minimum functions for autonomous driving

- Localization using GNSS + LiDAR + IMU
- Primitive object detection with LiDARs
- Driving along a single lane
- Stopping against stopline
- Stopping against obstacles
- Extend features using Autoware Universe

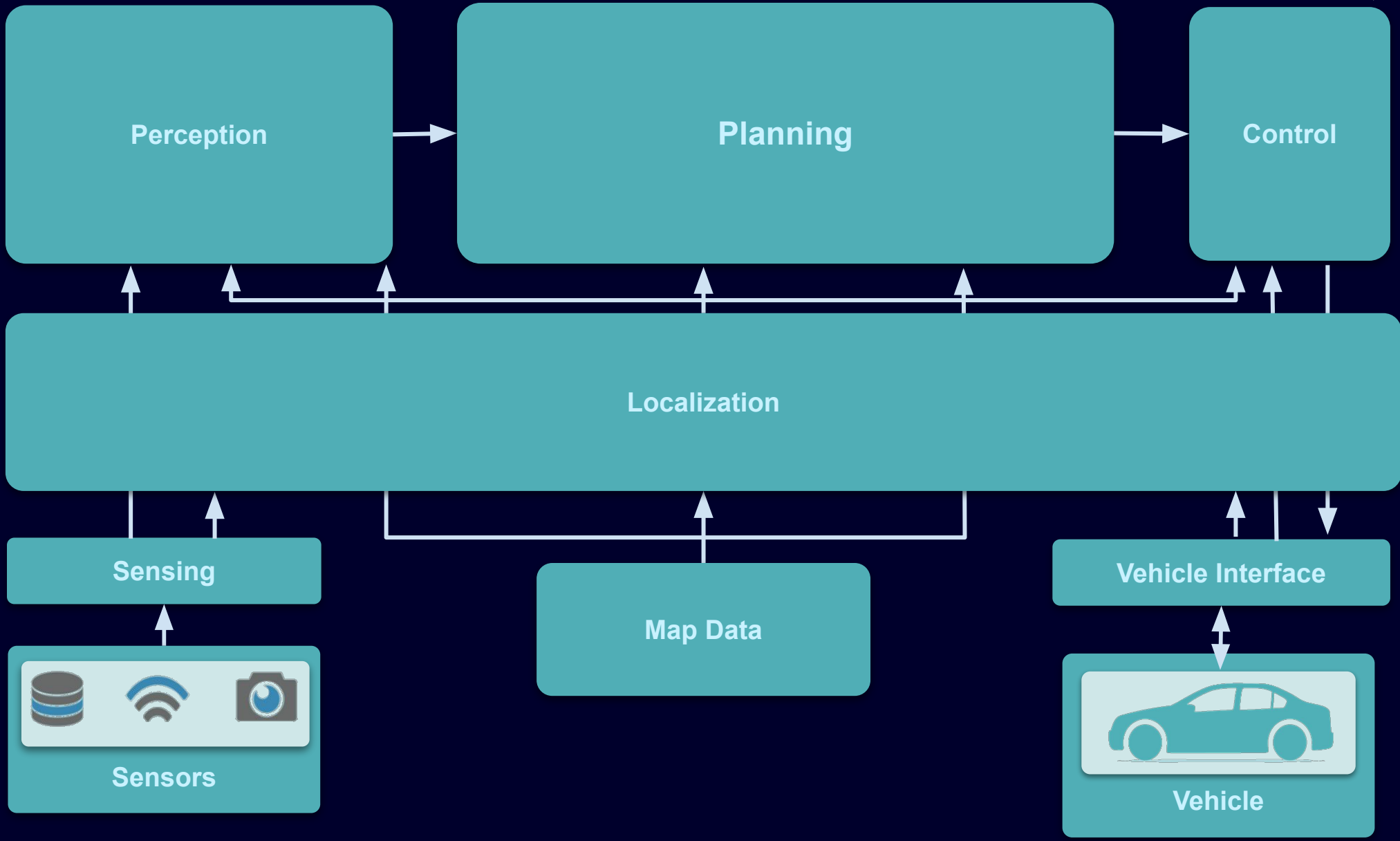


Who is it for?

- People who find Autoware Universe too complex
- People who needs stable version of Autoware without having breaking changes to interface

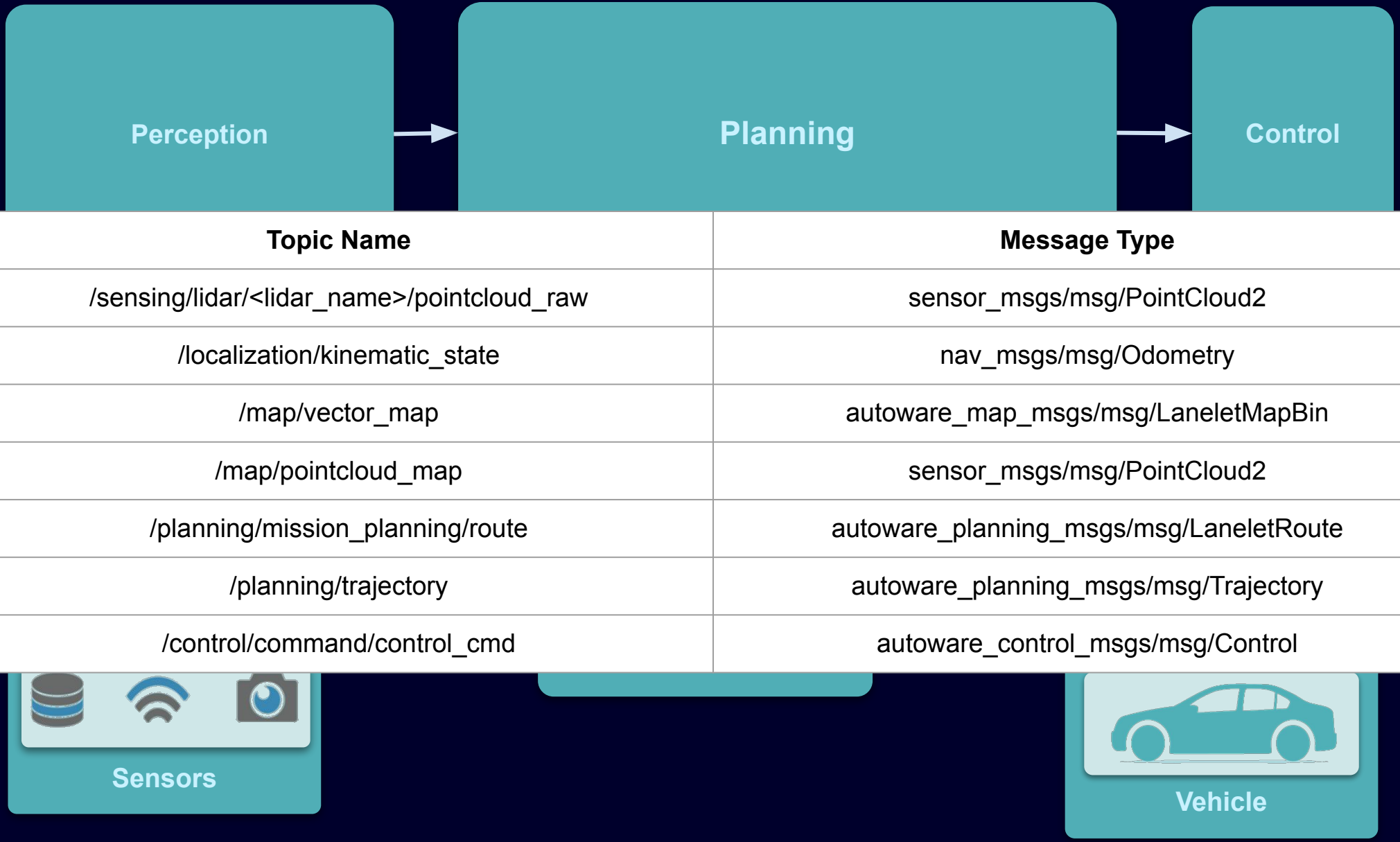
Autoware Architecture

THE AUTOWARE FOUNDATION



Autoware Architecture

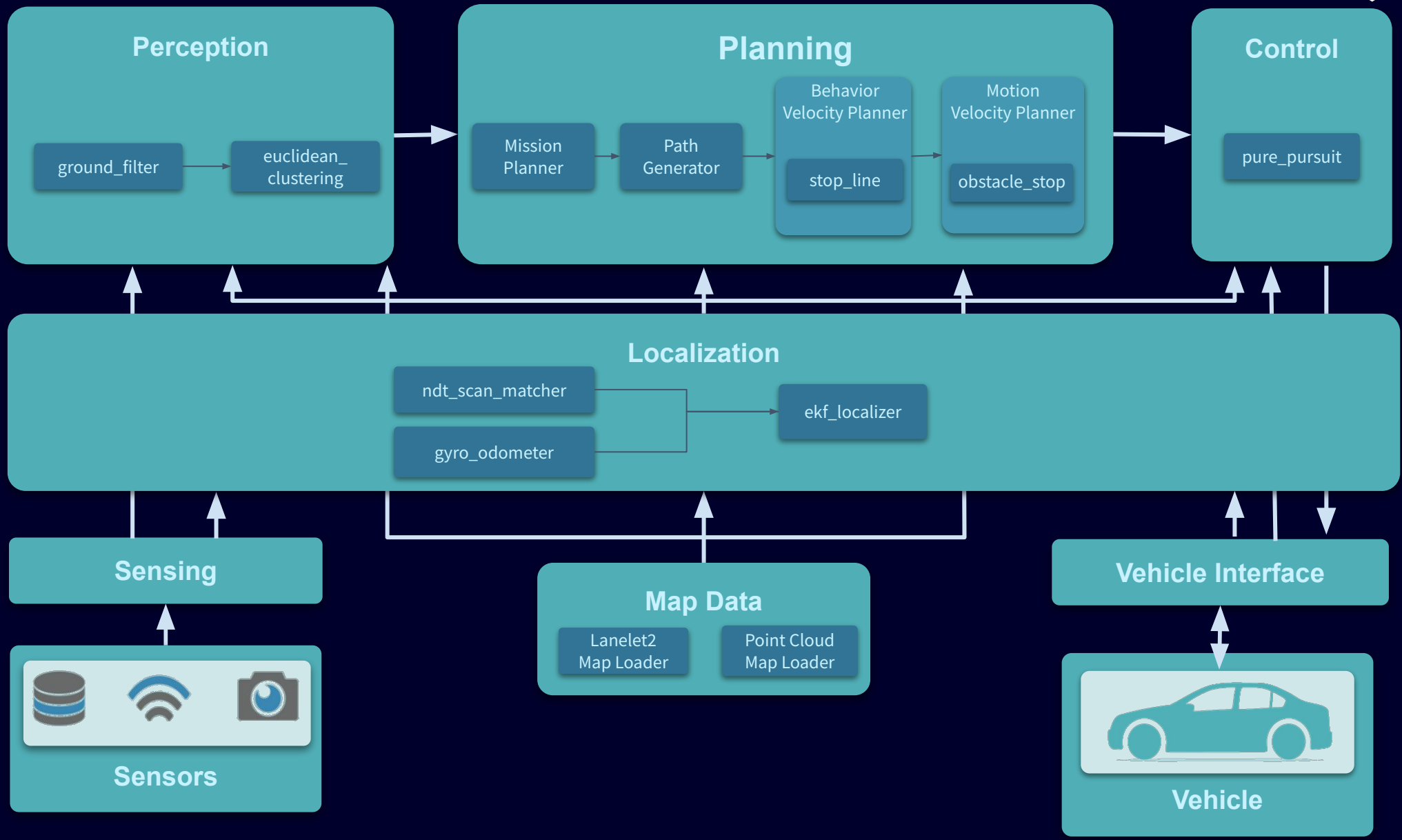
THE AUTOWARE FOUNDATION



Topic Name	Message Type
/sensing/lidar/<lidar_name>/pointcloud_raw	sensor_msgs/msg/PointCloud2
/localization/kinematic_state	nav_msgs/msg/Odometry
/map/vector_map	autoware_map_msgs/msg/LaneletMapBin
/map/pointcloud_map	sensor_msgs/msg/PointCloud2
/planning/mission_planning/route	autoware_planning_msgs/msg/LaneletRoute
/planning/trajectory	autoware_planning_msgs/msg/Trajectory
/control/command/control_cmd	autoware_control_msgs/msg/Control

Autoware Core Implementation

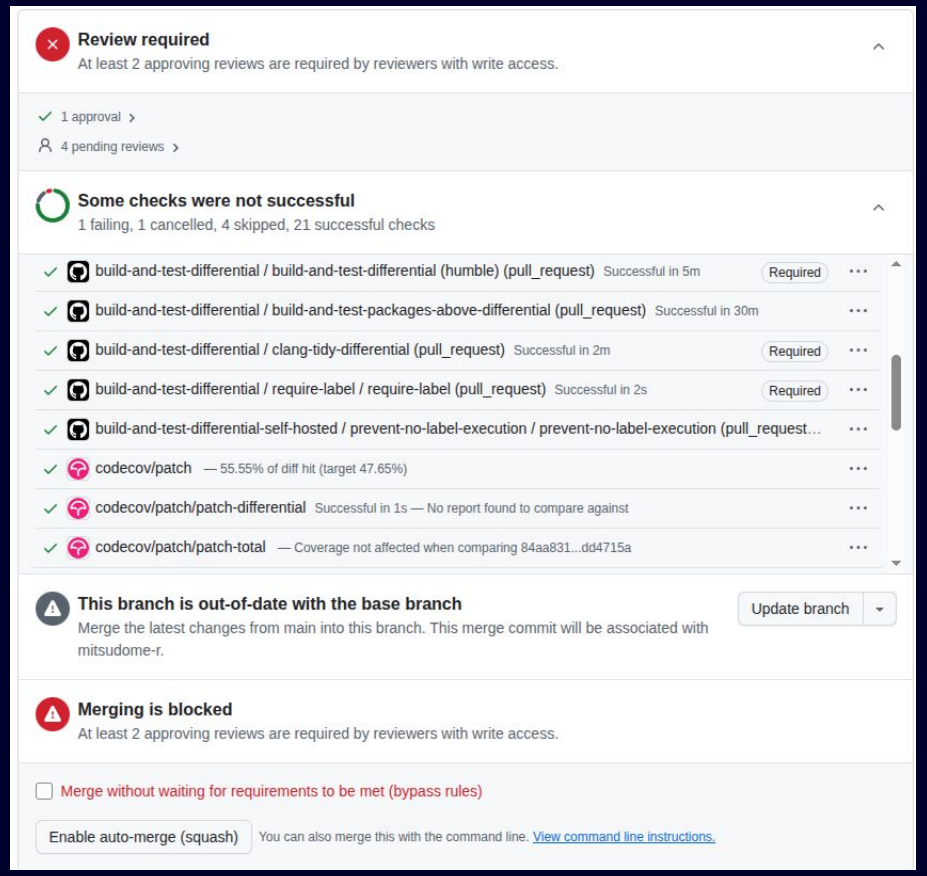
THE AUTOWARE FOUNDATION



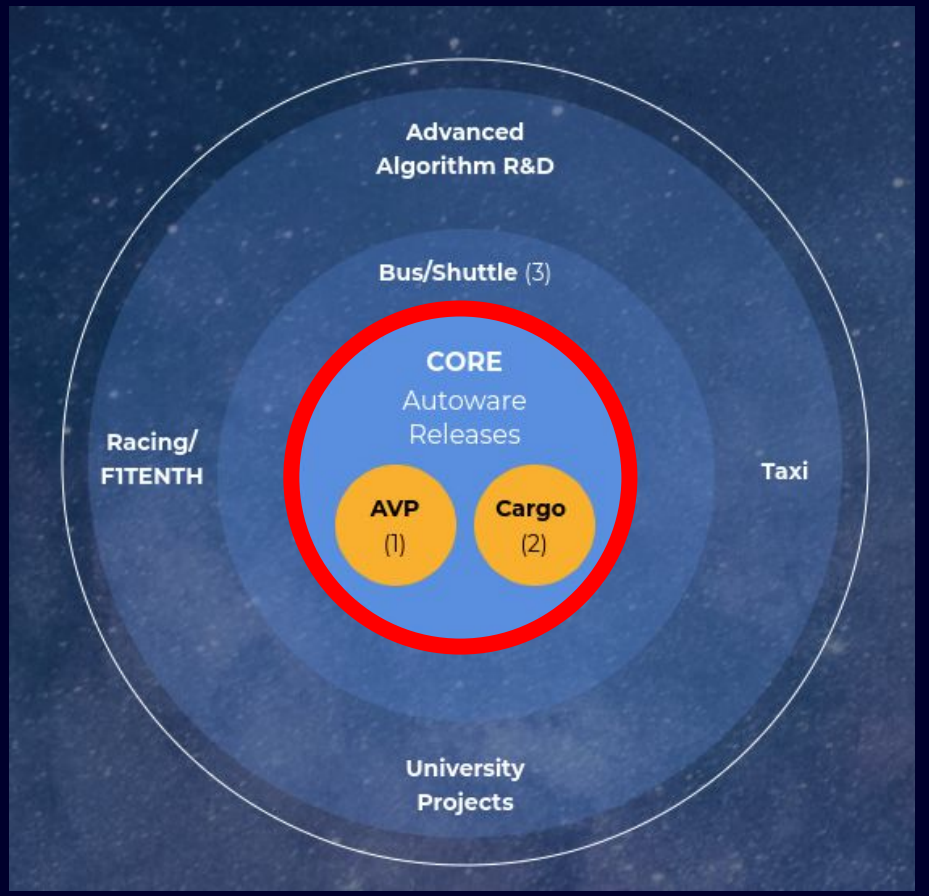
Autoware Core Development Activities

THE AUTOWARE FOUNDATION

Setting CI for Autoware Core
Add unit tests for packages



Making Autoware self-contained (remove dependency to TIER IV repositories)



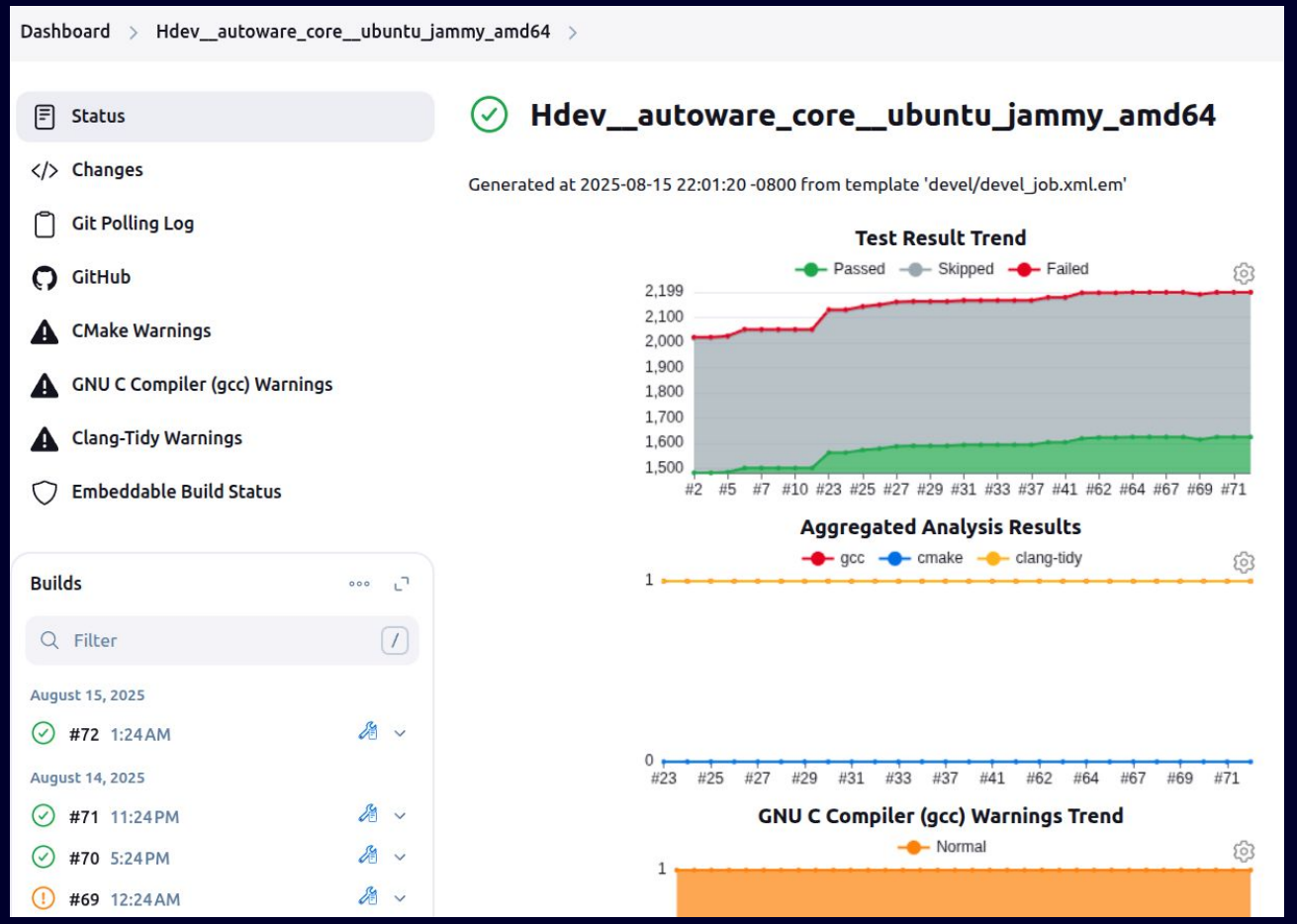
Autoware Core Development Activities

THE AUTOWARE FOUNDATION

Reduce technical debts
Refining package structure
Refine Interface to match with Autoware Design Docs

```
autoware_gnss_poser
├─ package.xml
├─ CMakeLists.txt
├─ README.md
│
├─ config
│   ├─ gnss_poser.param.yaml
│   └─ another_non_ros_config.yaml
│
├─ schema
│   └─ gnss_poser.schema.json
│
├─ doc
│   ├─ foo_document.md
│   └─ foo_diagram.svg
│
├─ include  # for exporting headers
│   └─ autoware
│       └─ gnss_poser
│           └─ exported_header.hpp
│
├─ src
│   ├─ include
│   │   ├─ gnss_poser_node.hpp
│   │   └─ foo.hpp
│   ├─ gnss_poser_node.cpp
│   └─ bar.cpp
│
├─ launch
│   ├─ gnss_poser.launch.xml
│   └─ gnss_poser.launch.py
│
└─ test
    ├─ test_foo.hpp  # or place under an `include` folder here
    └─ test_foo.cpp
```

Registering to ROS Buildfarm



Autoware Core Binary Release

THE AUTOWARE FOUNDATION

First release of Autoware from ROS Buildfarm!

```
$ sudo apt update
$ sudo apt install ros-humble-autoware-core
$ sudo apt install ros-humble-autoware-global-parameter-loader
```

…Jazzy porting is in progress

Using Binary Release

THE AUTOWARE FOUNDATION

Install Autoware Core from apt

```
$ sudo apt update
$ sudo apt install ros-humble-autoware-core
$ sudo apt install ros-humble-autoware-global-parameter-loader
```

Setup configuration files for AWSIM

```
$ mkdir -p ~/colcon_ws/src/
$ cd ~/colcon_ws/src
$ git clone https://github.com/autowarefoundation/autoware_launch.git
$ git clone https://github.com/tier4/sensor_component_description.git
$ cd ~/colcon_ws
$ colcon build --symlink-install --cmake-args -DCMAKE_BUILD_TYPE=Release
```

Using Binary Release

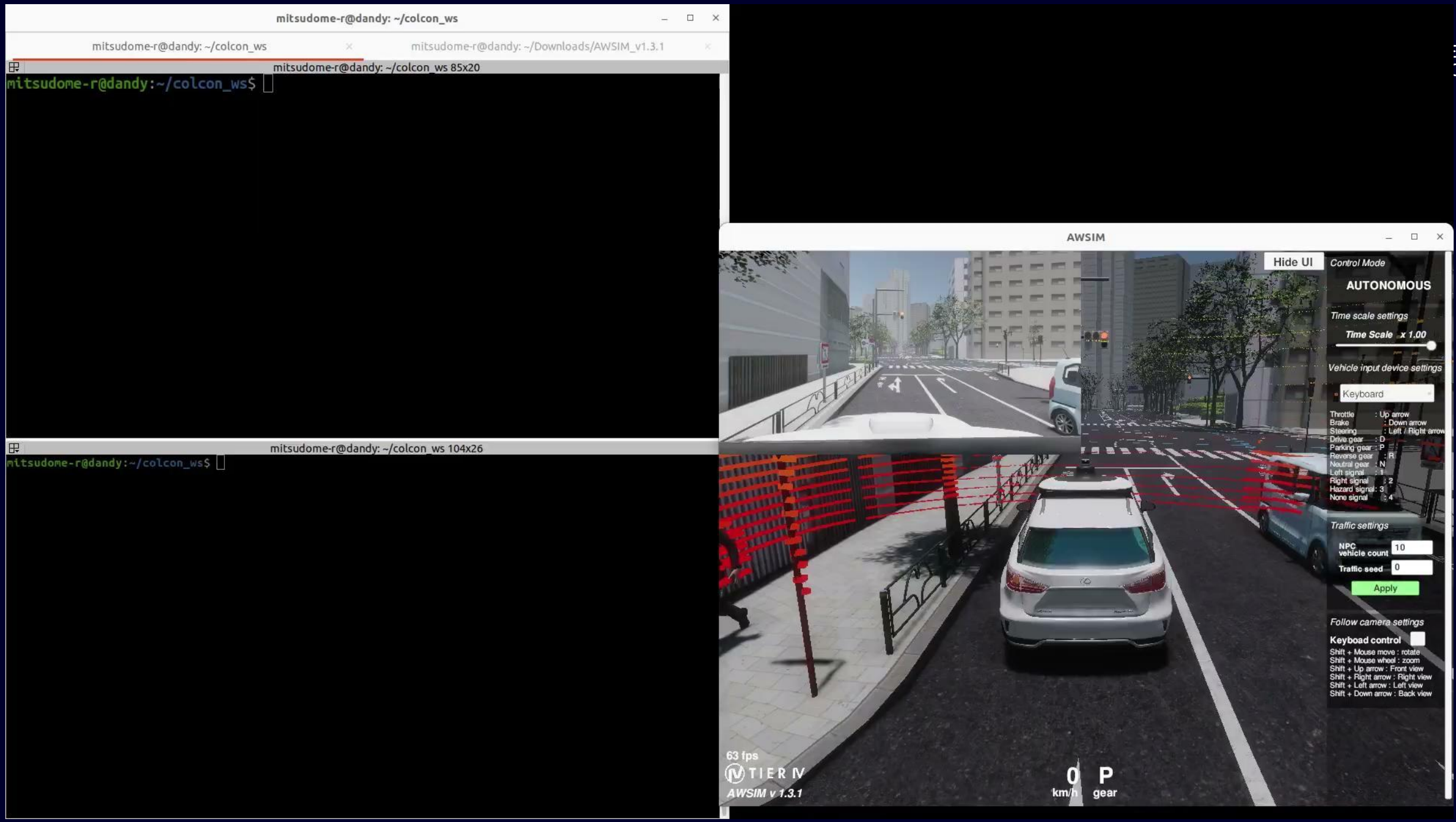
THE AUTOWARE FOUNDATION

Launch Autoware

```
$ ros2 launch autoware_core autoware_core.launch.xml map_path:=<map_path>
use_sim_time:=true vehicle_model:=sample_vehicle sensor_model:=awsim_sensor_kit
```

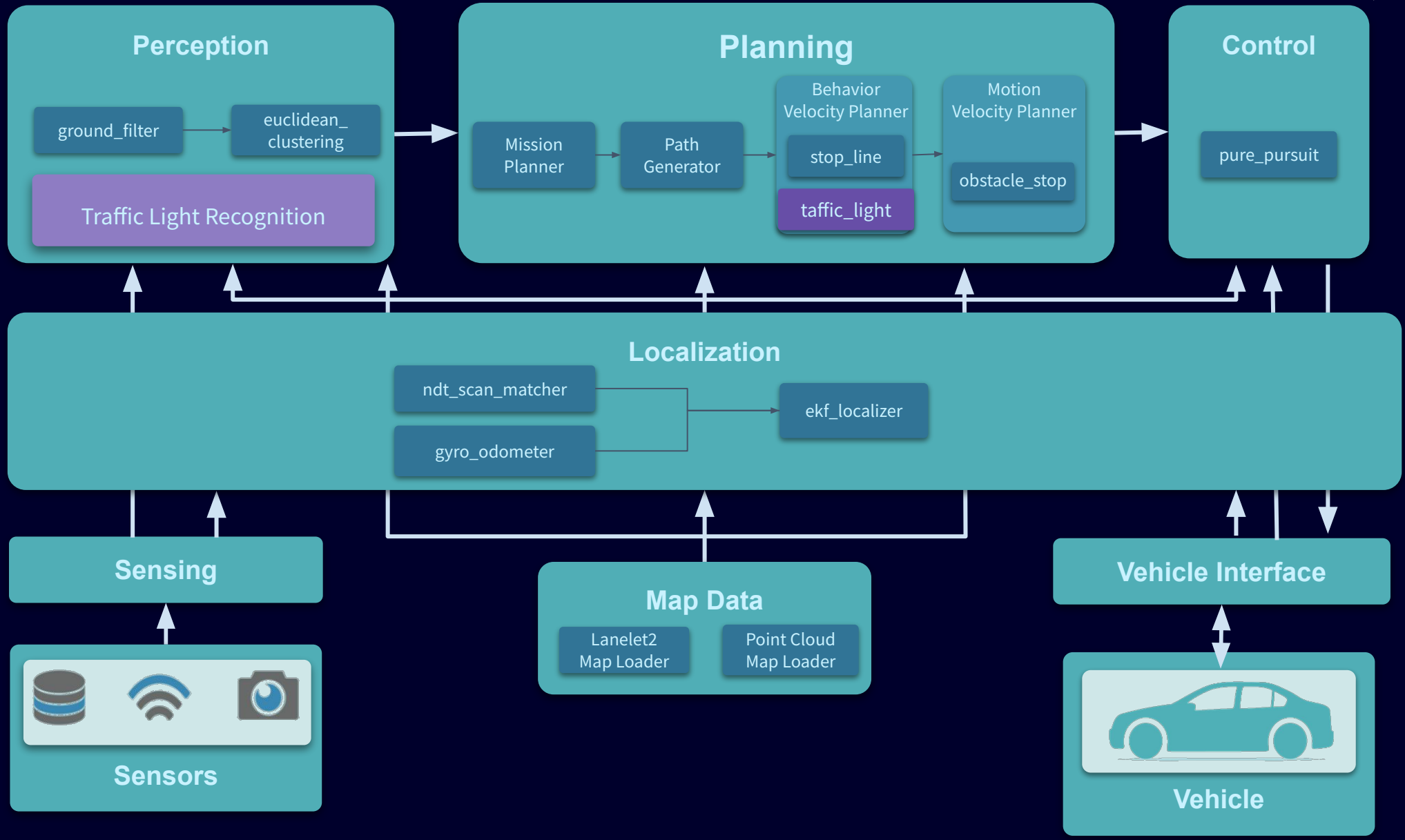
Run AWSIM (https://tier4.github.io/AWSIM)

```
$ ./AWSIM.x86_64
```


mitsudome-r@dandy: ~/colcon_ws
mitsudome-r@dandy: ~/colcon_ws
mitsudome-r@dandy: ~/Downloads/AWSIM_v1.3.1
mitsudome-r@dandy: ~/colcon_ws 85x20
mitsudome-r@dandy:~/colcon_ws$
mitsudome-r@dandy: ~/colcon_ws 104x26
mitsudome-r@dandy:~/colcon_ws$
AWSIM
Hide UI
Control Mode
AUTONOMOUS
Time scale settings
Time Scale x 1.00
Vehicle input device settings
Keyboard
Throttle : Up arrow
Brake : Down arrow
Steering : Left / Right arrow
Drive gear : D
Parking gear : P
Reverse gear : R
Neutral gear : N
Left signal : 1
Right signal : 2
Hazard signal : 3
None signal : 4
Traffic settings
NPC vehicle count 10
Traffic seed 0
Apply
Follow camera settings
Keyboad control
Shift + Mouse move : rotate
Shift + Mouse wheel : zoom
Shift + Up arrow : Front view
Shift + Right arrow : Right view
Shift + Left arrow : Left view
Shift + Down arrow : Back view
63 fps
TIER IV
AWSIM v 1.3.1
0 km/h
P gear

Adding Traffic Light Recognition

THE AUTOWARE FOUNDATION



Adding Traffic Light Recognition

THE AUTOWARE FOUNDATION

Build Autoware Univerese from source

Add Traffic Light Recognition Launch from Autoware Universe to autoware_core_perception.launch.xml

```
diff --git a/perception/autoware_core_perception/launch/autoware_core_perception.launch.xml b/perception/autoware_core_perception/launch/autoware_core_perception.launch.xml
index 11108ea6..6a687d9b 100644
--- a/perception/autoware_core_perception/launch/autoware_core_perception.launch.xml
+++ b/perception/autoware_core_perception/launch/autoware_core_perception.launch.xml
@@ -23,5 +23,38 @@
         <arg name="output_topic" value="/perception/object_recognition/objects"/>
       </include>
     </group>
+
+    <!-- Traffic light module -->
+    <let name="data_path" value="/home/mitsudome-r/autoware_data"/>
+    <group>
+      <push-ros-namespace namespace="traffic_light_recognition"/>
+      <let name="traffic_light_config_dir" value="$(find-pkg-share autoware_launch)/config/perception/traffic_light_recognition"/>
+
+      <include file="$(find-pkg-share tier4_perception_launch)/launch/traffic_light_recognition/traffic_light.launch.xml">
+        <arg name="fusion_only" value="false"/>
+        <arg name="camera_namespaces" value="[traffic
+        <arg name="use_high_accuracy_detection" va
+        <arg name="high_accuracy_detection_type
+        <arg name="each_traffic_light_map_b
+        <arg name="traffic_light_fine_de
+
+
+
+
+
+
+
+
+        <arg name="traffic_light_arbiter_param_path" value="$(var traffic_light_config_dir)/traffic_light_arbiter/traffic_light_arbiter.param.yaml" />
+        <arg name="crosswalk_traffic_light_estimator_param_path" value="$(var traffic_light_config_dir)/crosswalk_traffic_light_estimator/crosswalk_traffic_light_estimator.param.yaml"/>
+        <arg name="whole_image_detection/model_path" value="$(var data_path)/tensorrt_yolox/yolox_s_car_ped_tl_detector_960_960_batch_1.onnx"/>
+        <arg name="whole_image_detection/label_path" value="$(var data_path)/tensorrt_yolox/car_ped_tl_detector_labels.txt"/>
+        <arg name="fine_detection/model_path" value="$(var data_path)/traffic_light_fine_detector/tlr_car_ped_yolox_s_batch_6.onnx"/>
+        <arg name="fine_detection/label_path" value="$(var data_path)/traffic_light_fine_detector/tlr_labels.txt"/>
+        <arg name="classification/car/model_path" value="$(var data_path)/traffic_light_classifier/traffic_light_classifier_mobilenetv2_batch_6.onnx"/>
+        <arg name="classification/car/label_path" value="$(var data_path)/traffic_light_classifier/lamp_labels.txt"/>
+        <arg name="classification/pedestrian/model_path" value="$(var data_path)/traffic_light_classifier/ped_traffic_light_classifier_mobilenetv2_batch_6.onnx"/>
+        <arg name="classification/pedestrian/label_path" value="$(var data_path)/traffic_light_classifier/lamp_labels_ped.txt"/>
+      </include>
+    </group>
+
   </group>
 </launch>
```

<include file="$(find-pkg-share tier4_perception_launch)/launch/traffic_light_recognition/traffic_light.launch.xml/>

Adding Traffic Light Recognition

THE AUTOWARE FOUNDATION

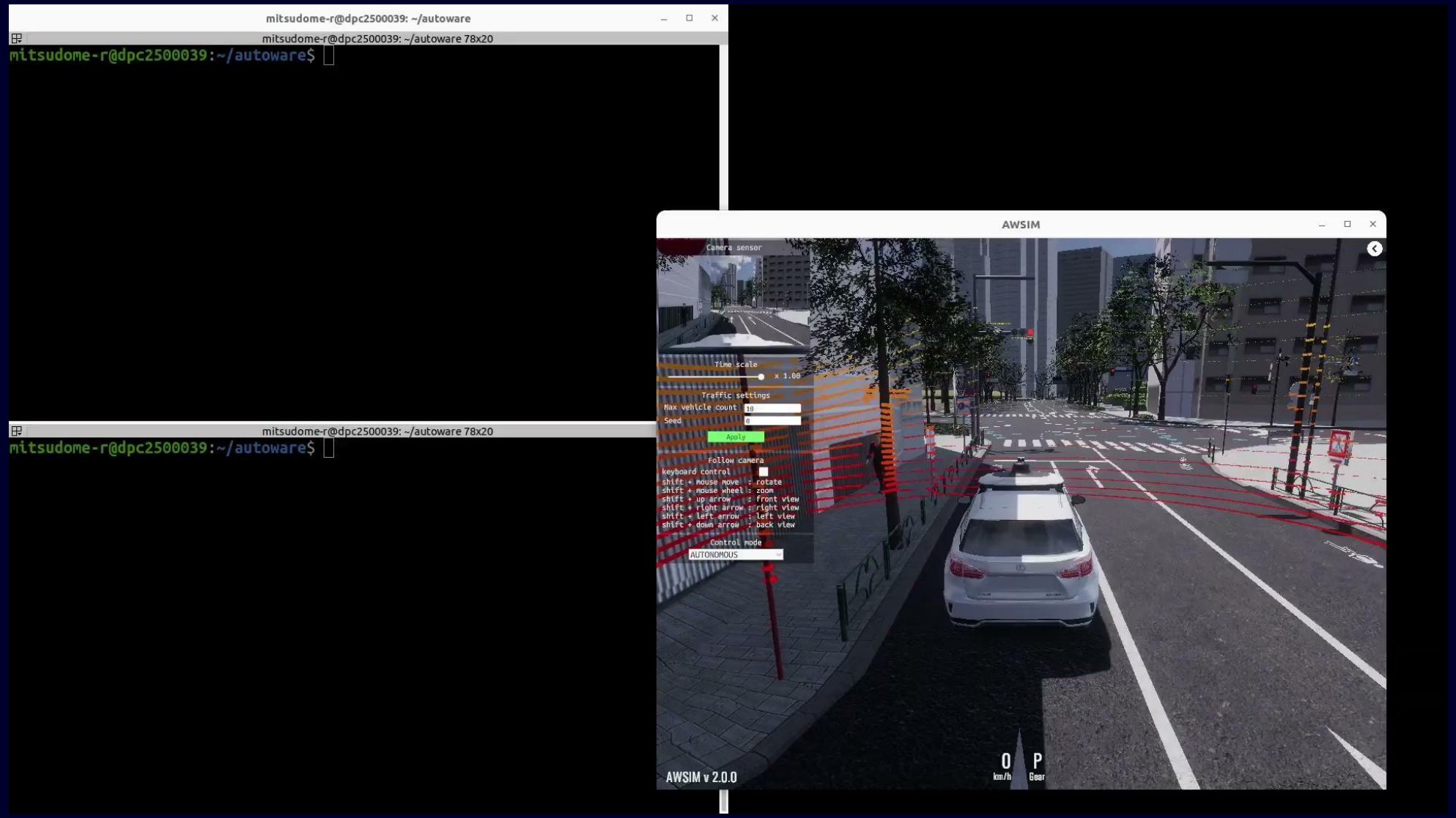
Load behavior_velocity_planner_traffic_light_module in autoware_core_planning.launch.xml

<param name="launch_modules" value="[autoware::behavior_velocity_planner::StopLineModulePlugin,autoware::behavior_velocity_planner::TrafficLightModulePlugin]"/>

```
-       <param name="launch_modules" value="[autoware::behavior_velocity_planner::StopLineModulePlugin]"/>
+       <param name="launch_modules" value="[autoware::behavior_velocity_planner::StopLineModulePlugin,autoware::behavior_velocity_planner::TrafficLightModulePlugin]"/>
        <param name="enable_all_modules_auto_mode" value="$(var enable_all_modules_auto_mode)"/>
        <param name="is_simulation" value="$(var is_simulation)"/>
        <!-- load config -->
@@ -83,6 +83,7 @@
        <param from="$(var behavior_velocity_planner_param_path)"/>
        <param from="$(var behavior_velocity_planner_common_param_path)"/>
        <param from="$(var behavior_velocity_planner_stop_line_module_param_path)"/>
+       <param from="$(var behavior_velocity_planner_traffic_light_module_param_path)"/>
      </node>
    </group>
```

Autoware Core + Autoware Universe

THE AUTOWARE FOUNDATION



Summary

THE AUTOWARE FOUNDATION

- Autoware made the first official release of Autoware Core from ROS Buildfarm
- Autoware Core provides is a base platform for Autonomous Driving
- Autoware Core can extend its capabilities with packages from Autoware Universe (Community Maintained Packages)

Future Work:

- More test coverage
- Makes regular release of Autoware Core
- Port more packages from Autoware Universe to Core
- Port Autoware Core to latest ROS Distributions

Using Autoware Core

THE AUTOWARE FOUNDATION

Source Install: Follow the instructions in Autoware Documentation

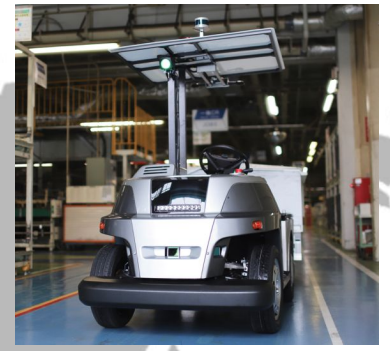
```
$ git clone https://github.com/autowarefoundation/autoware.git
$ vcs import src < autoware.repos
$ vcs import src < autoware-nightly.repos
$ rosdep install -y --from-paths src --ignore-src --rosdistro $ROS_DISTRO
$ colcon build --symlink-install --cmake-args -DCMAKE_BUILD_TYPE=Release
```

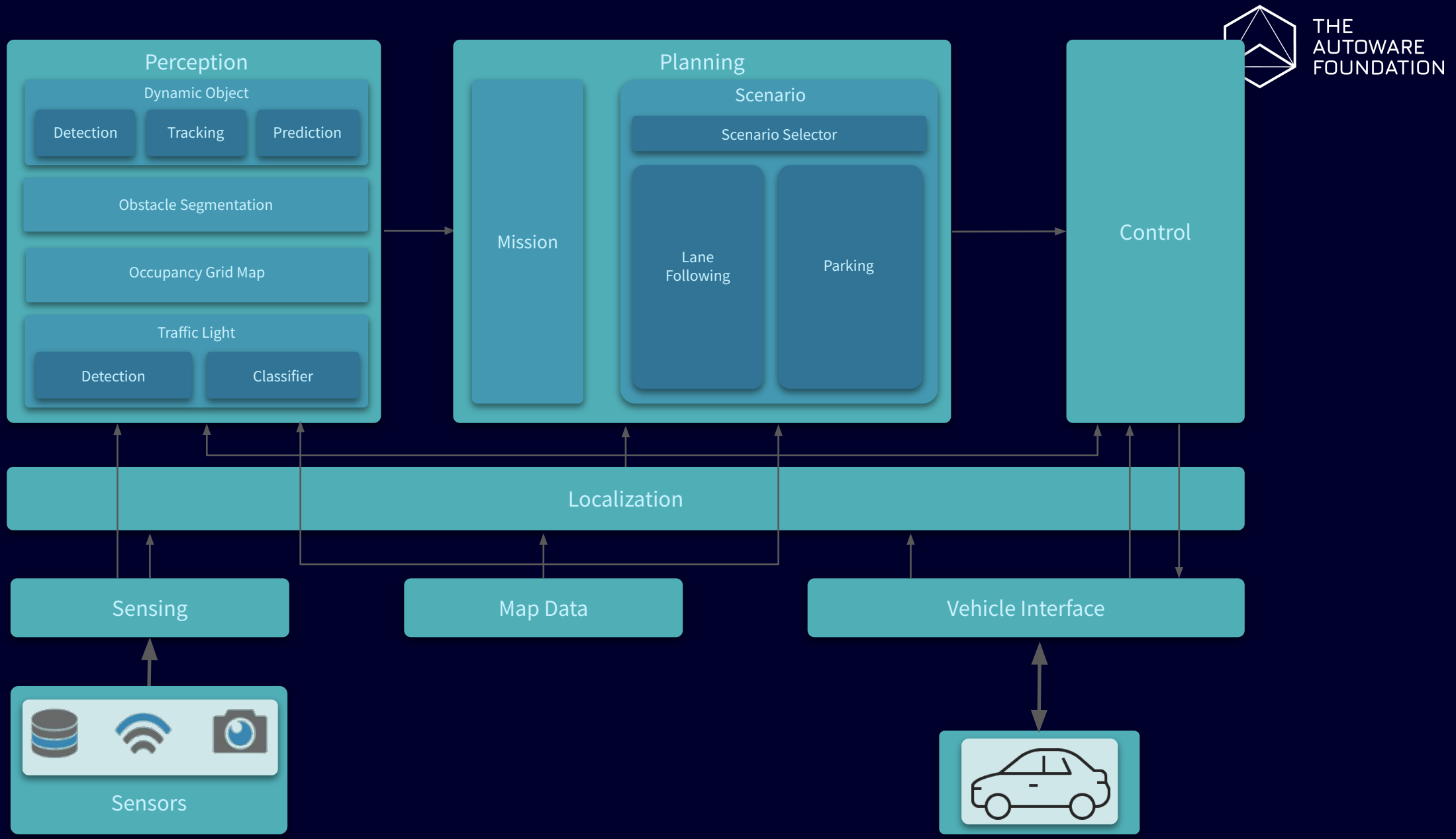
Using Docker

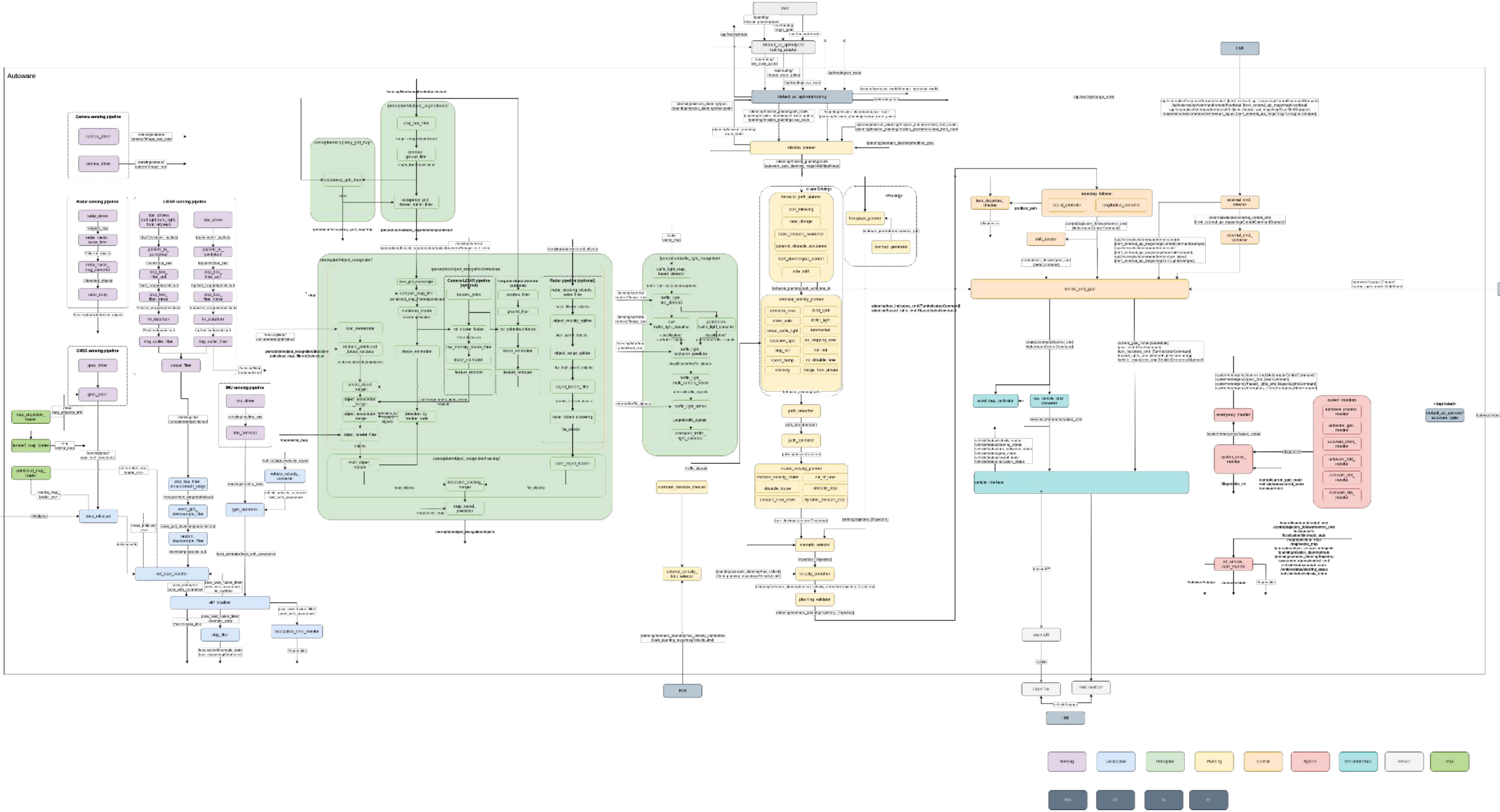
```
$ docker pull ghcr.io/autowarefoundation/autoware:universe-devel-cuda
$ docker run -it --name autoware_latest --gpus all --net=host --privileged -v
/your/autoware/path/out/of/docker:/wc_ws -w /wc_ws/ -e DISPLAY -e
QT_X11_NO_MITSHM=1 -v /tmp/.X11-unix:/tmp/.X11-unix
ghcr.io/autowarefoundation/autoware:universe-devel-cuda bash
```

Autoware Deployments

THE AUTOWARE FOUNDATION




THE AUTOWARE FOUNDATION
Perception
Dynamic Object
Detection
Tracking
Prediction
Obstacle Segmentation
Occupancy Grid Map
Traffic Light
Detection
Classifier
Planning
Mission
Scenario
Scenario Selector
Lane Following
Parking
Control
Localization
Sensing
Map Data
Vehicle Interface
Sensors


Autoware
Camera sensing pipeline
Radar sensing pipeline
LiDAR sensing pipeline
GNSS sensing pipeline
IMU sensing pipeline
Radar pipeline (optional)
Camera-LiDAR pipeline (optional)
Sensing
Localization
Perception
Planning
Control
System
Vehicle Interface
Vehicle
Map

Autoware Core Architecture

THE AUTOWARE FOUNDATION