**System Specification**

Cooperative Research Platform

1. Goal of the document

This document summarizes the details of how the Cooperative Research Platform (CRP) is built up. It consists of the multiple layers:

* System Functionality,
* Architecture,
* Scenario Coverage and demonstration basis

1. Function Specification

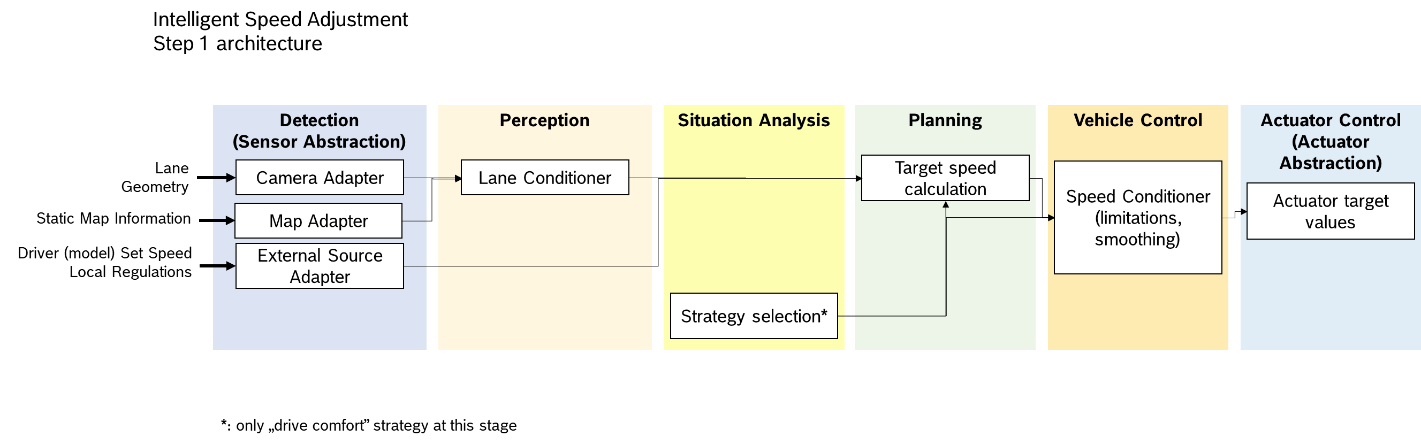
This Section describes the high-level specifications of the covered functionality.

2.1. Intelligent Speed Adjustment

* Step 1 functionality: longitudinal speed control adjusted to static information, such as curve and local regulations (speed limit).
* Step 2 functionality: step 1 + speed adjustment on dynamic information, such as moving objects (e.g., followed vehicle).

For both: speed range is 0 <= vx <= 150 kph, which therefore includes automatic start/stop functionality.

Step 1 architecture:

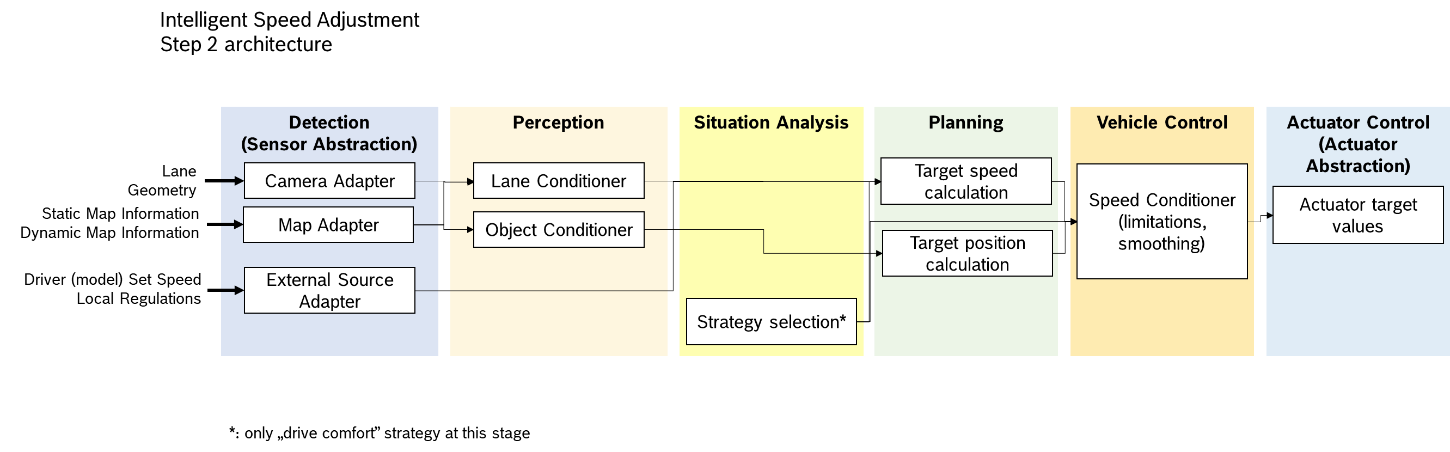


Corresponding Autoware Universe packages:

|  |  |  |  |
| --- | --- | --- | --- |
| Component | Node/Package | Inputs | Outputs |
| **Planning / Target speed calculation** | path\_smoother | behavior\_planning / path; <type: Path > | path\_smoother / path <type: Path > |
| **Planning / Target speed calculation** | obstacle\_velocity\_limiter | motion\_planning / obstacle\_avoidance\_planner / trajectory  <type: Trajectory> | motion\_planning / obstacle\_velocity\_limiter / trajectory  <type: Trajectory> |
| **Planning / Target speed calculation** | motion\_velocity\_smoother | trajectory  <type: Trajectory> | /planning / scenario\_planning / motion\_velocity\_smoother / trajectory  <type: Trajectory> |
| **Detection / MapAdapter** | map\_projection\_loader |  | map\_projector\_info  <type: > |
| **Detection / MapAdapter** | lanelet2\_map\_loader | map\_projector\_info  <type: > | /map  /vector\_map |
| **Vehicle Control / Speed Conditioner** | motion\_velocity\_smoother | trajectory  <type: Trajectory> | /planning / scenario\_planning / motion\_velocity\_smoother / trajectory  <type: Trajectory> |
| Vehicle Control / Speed Conditioner | trajectory\_follower / longitudinal\_controller | /planning / scenario\_planning / trajectory  <type: Trajectory> | /control / trajectory\_follower / control\_cmd  <type: AckermannControlCommand> |

Note: even if we only control the vehicle longitudinally, the lateral path shall be filled with dummy values. Idea: add a straight line with no offset. Later, it must be solved that the vehicle is longitudinally controlled by the system, but laterally by the driver.

Step 2 architecture:



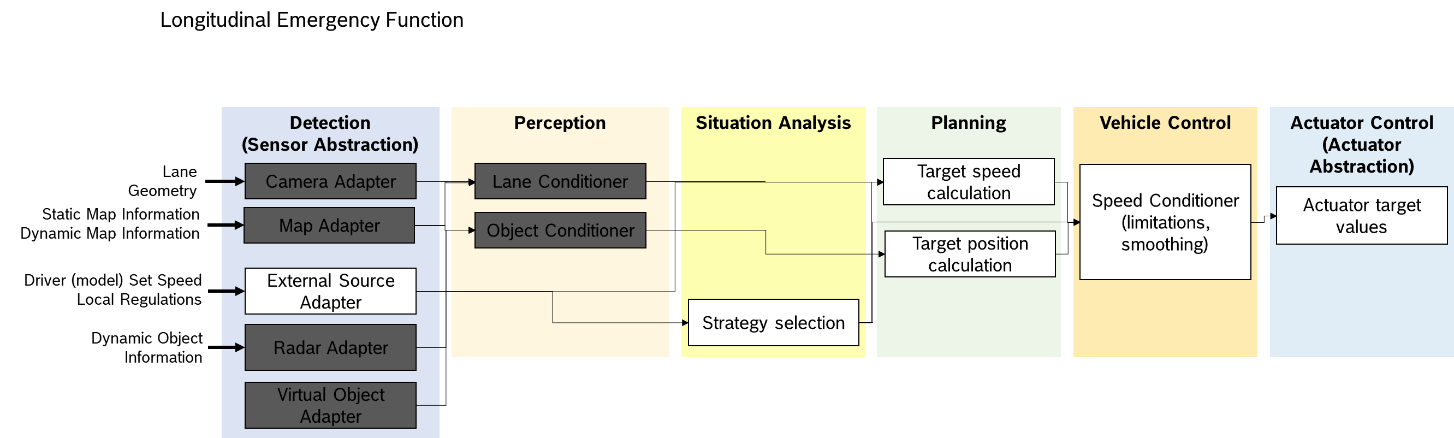
Corresponding Autoware Universe packages:

|  |  |  |  |
| --- | --- | --- | --- |
| Component | Node/Package | Inputs | Outputs |
| Target speed calculation |  |  |  |

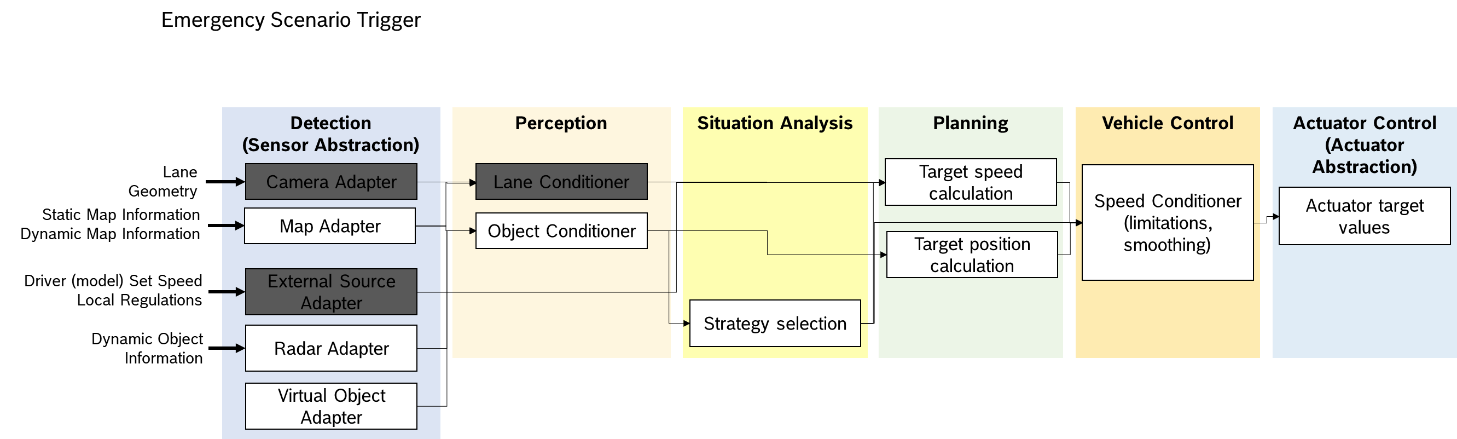
Interfaces:

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | Name | Type | Content |
| Detection | det\_road\_geometry |  |  |
|  |  |  |  |
|  |  |  |  |

2.2. Longitudinal emergency function



2.3. Emergency Scenario Trigger



Based on distributed sensor data calculate the trigger of the emergency scenario.

1. Message definitions

3.1. Path

This is a tier4 autoware message extension, with the following definition:

|  |
| --- |
| std\_msgs/Header header  tier4\_planning\_msgs/PathPoint[] points  nav\_msgs/OccupancyGrid drivable\_area |

3.2. Path point

|  |
| --- |
| uint8 REFERENCE=0  uint8 FIXED=1  geometry\_msgs/Pose pose  geometry\_msgs/Twist twist  uint8 type |

3.3. Trajectory

|  |
| --- |
| std\_msgs/Header header  tier4\_planning\_msgs/TrajectoryPoint[] points |

3.4. Trajectory point

|  |
| --- |
| geometry\_msgs/Pose pose  geometry\_msgs/Twist twist  geometry\_msgs/Accel accel |