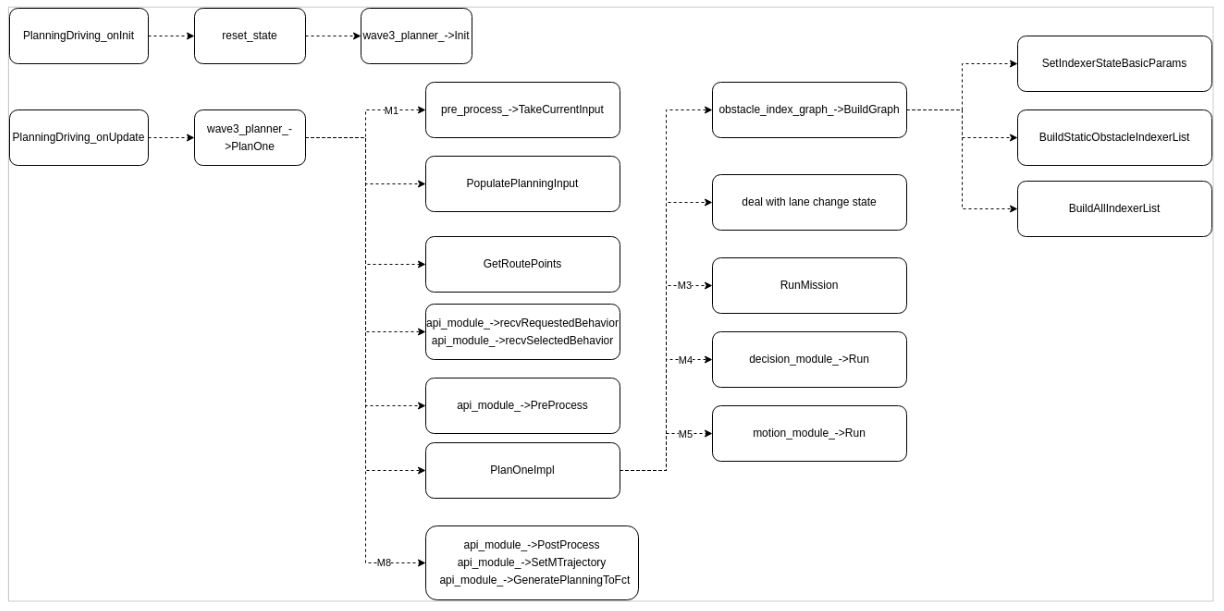
# 软件框架

## Pipeline



## Mission Module

The mission module is formed by two part: locate rote generator and lane change decider.

Local Route Group contains the lane candidates(left/right/center), and the target side is the side of the selected lane. Key Positions and Promote side are the planner's internal status that is stored in State Frame and could be queried by all planner's modules in the next frame.

The goal of the mission module: provide available road paths and the selected candidate(by routing or human command) for following submodules like decision and motion.

Locate rote generator part (part 1): generating available road paths(defined as Local Route) with given localization and semap frame.

Lane change decider (part 2): choose the best candidate to make sure ego can achieve the best progress, follow routing and overtake slow vehicles.

任务模块由两个部分组成：本地路径生成器和车道变换决策器。

本地路径组包含车道候选项（左/右/中），目标侧是所选车道的一侧。关键位置和促进侧是规划器内部状态，存储在状态框架中，并可以由下一帧中的所有规划器模块查询。

任务模块的目标：为后续的决策和动作等子模块提供可用的道路路径和所选的候选项（通过路径规划或人工命令）。

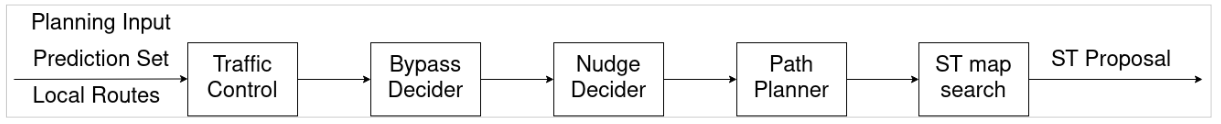
本地路径生成器部分（第1部分）：利用给定的定位和语义地图框架生成可用的道路路径（定义为本地路径）。

车道变换决策器部分（第2部分）：选择最佳的候选项，以确保自动驾驶车辆能够取得最佳进展，遵循路径规划并超越缓慢的车辆。

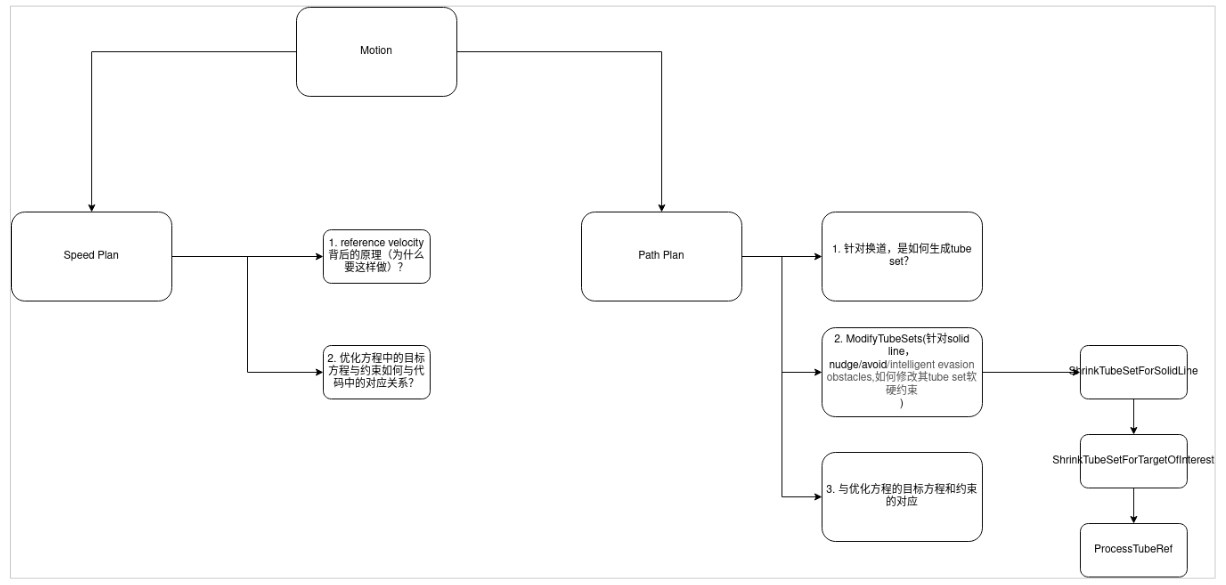
## Decision Module

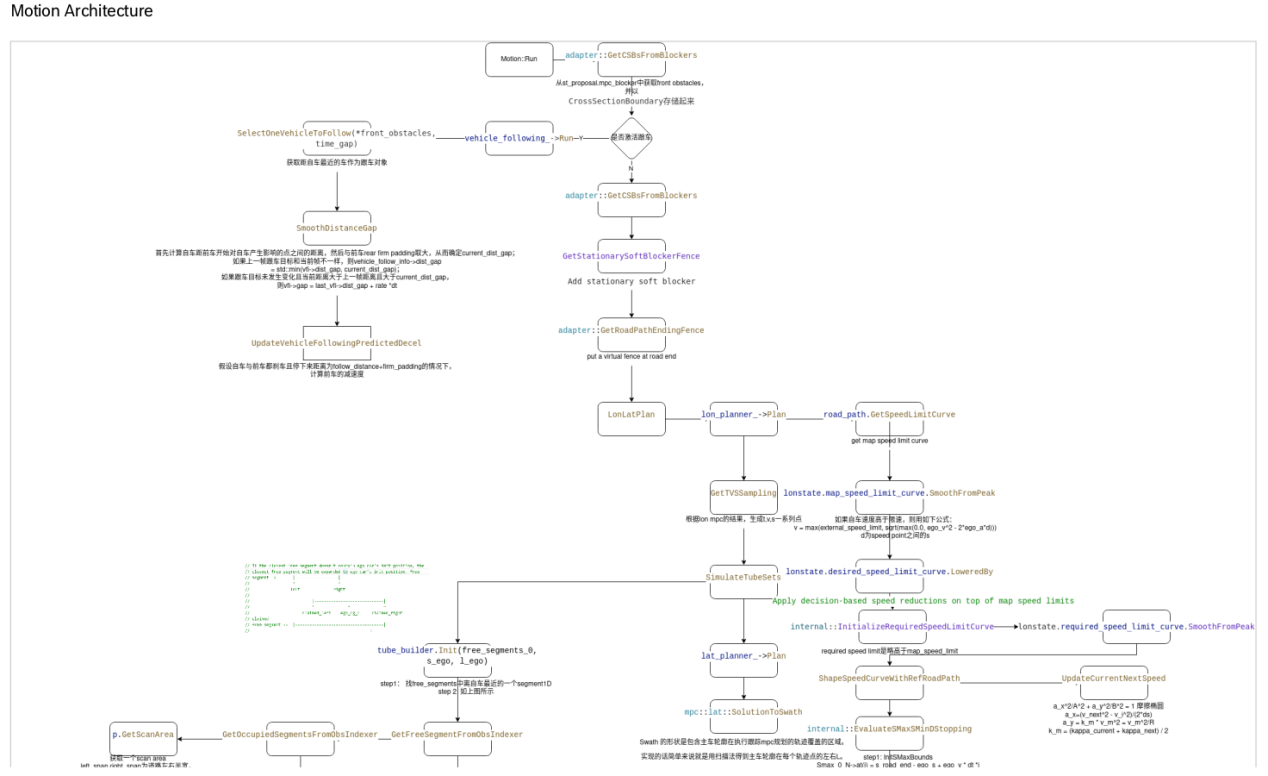
The overview goal of the decision module is to provide the path for the motion module. Four sub-tasks form the decision module: traffic control to process the traffic rules-related issues (traffic lights, stop sign, and crosswalk, etc.), bypass decider to figure out whether ego vehicle could bypass a static obstacle or not, path planner to generate enough pace to bypass necessary obstacles, and the st map will be presented in the ST map search module.

决策模块的总体目标是为动作模块提供路径。决策模块由四个子任务组成：交通控制处理与交通规则相关的问题（红绿灯、停车标志和人行横道等）、绕行决策器确定自动驾驶车辆是否能够绕过静态障碍物、路径规划器生成足够的速度以绕过必要的障碍物，以及ST地图将在ST地图搜索模块中呈现。

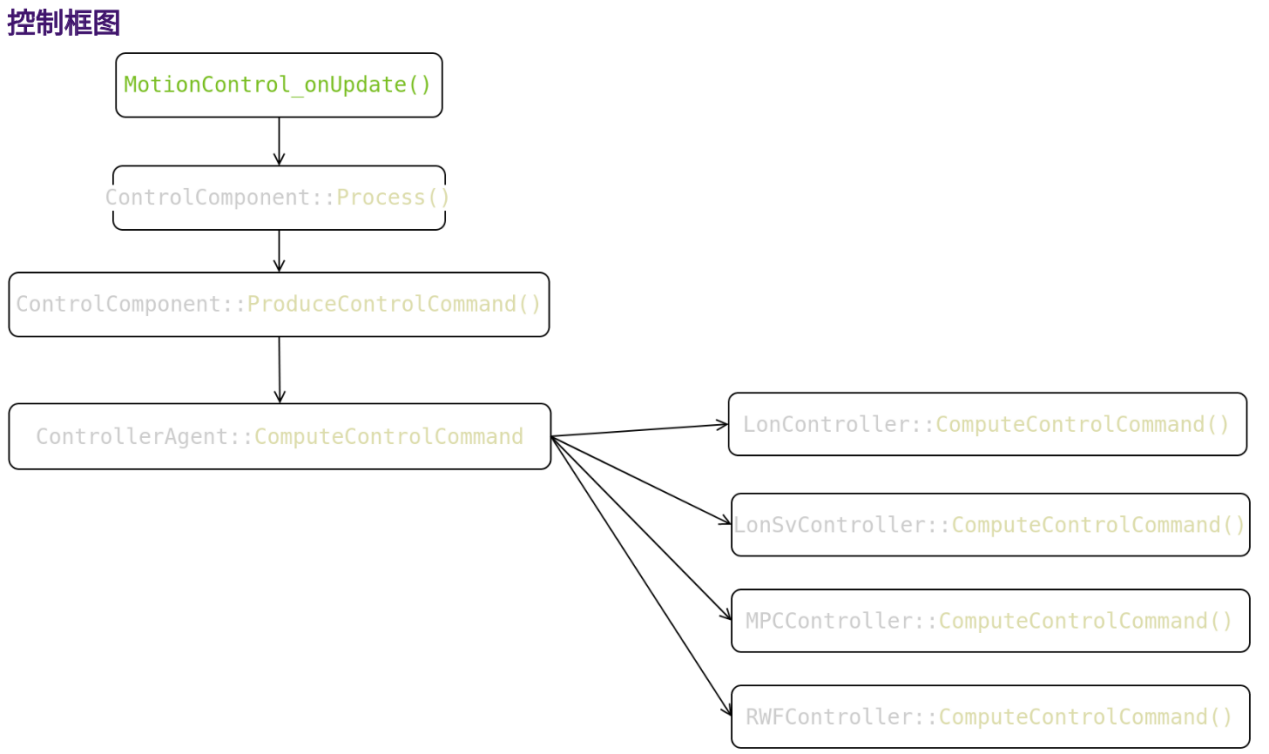


## Motion Module

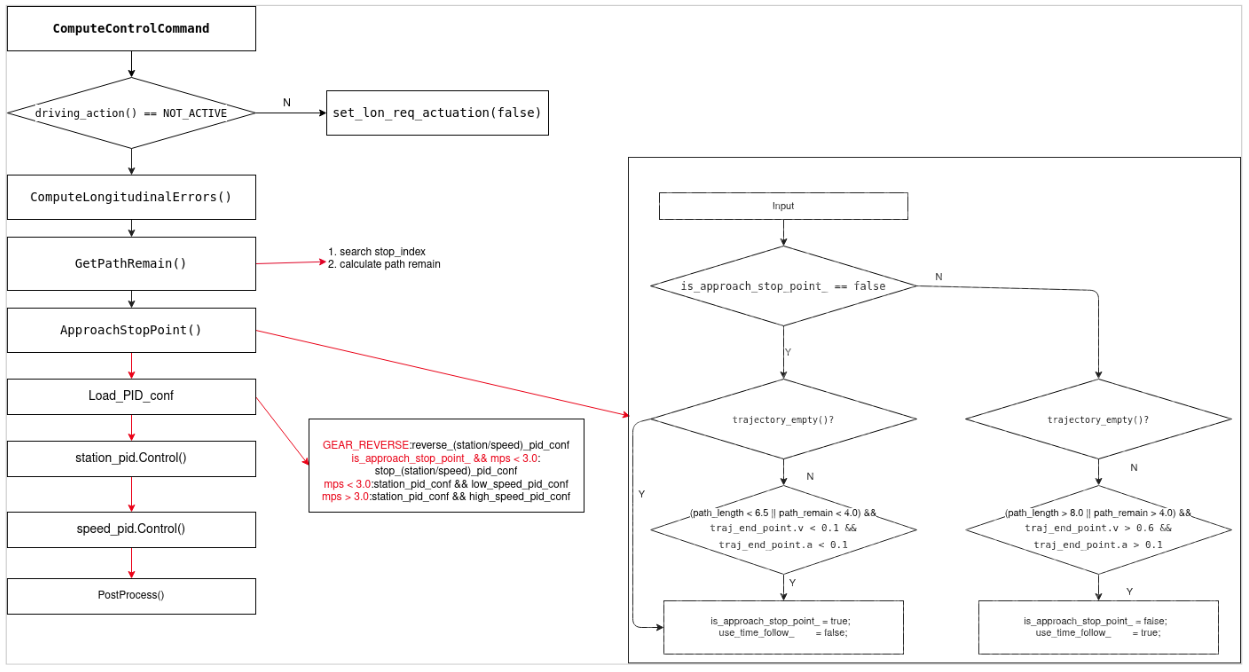




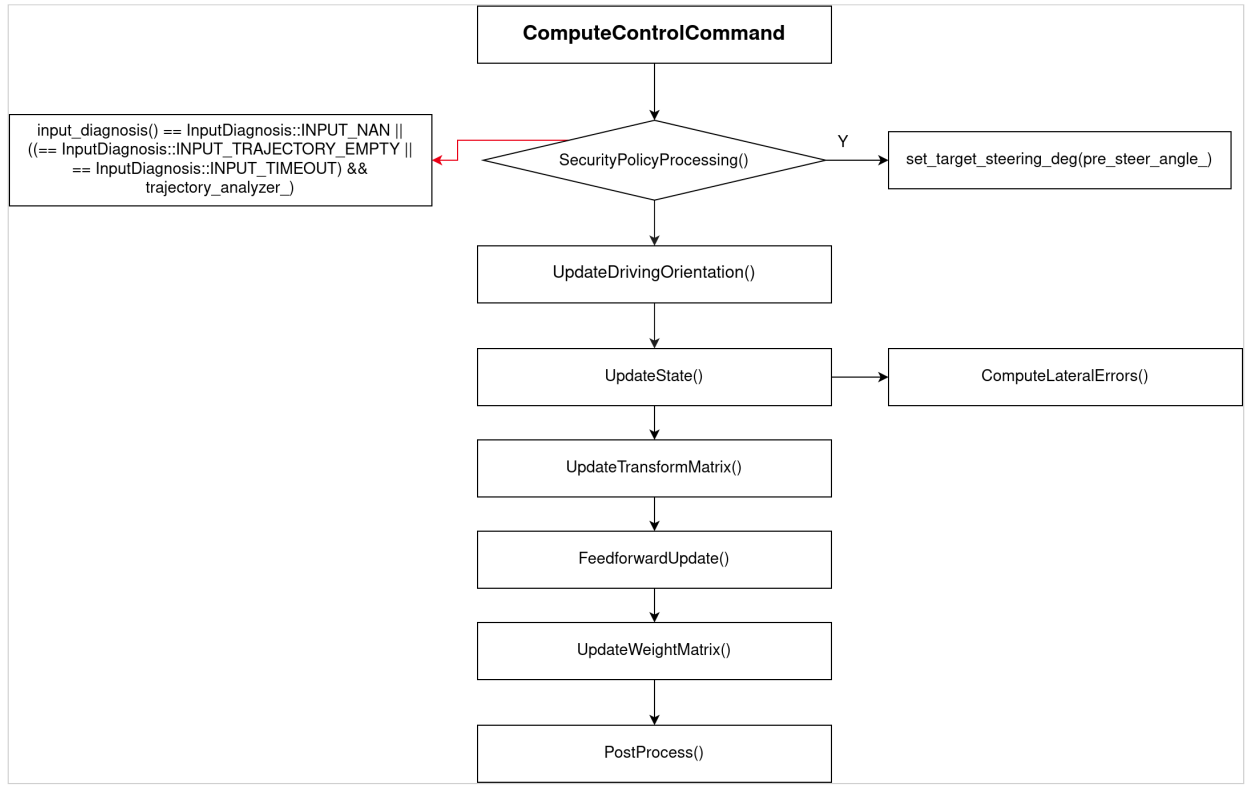
## Control Module



纵向控制LON\_CONTROLLER



横向控制MPC\_CONTROLLER



# 算法

## MPC控制

MPC在车辆运动控制的一般求解流程：

1. 问题建模：

- 系统建模：对车辆系统进行建模，包括车辆的动力学模型、约束条件以及性能指标等。

- 控制目标：明确控制的目标，例如跟踪一条参考轨迹。

2. 预测模型：

- 将车辆的动力学模型离散化，以便在有限的预测时段内进行预测。

- 建立状态空间模型，包括状态变量、控制输入、系统约束等。

3. 目标函数构建：

- 设计目标函数，通常由两部分组成：跟踪误差项和控制输入项。

- 跟踪误差项用于衡量车辆状态与参考轨迹之间的差异。

- 控制输入项用于衡量控制输入的大小和变化量，以保持控制的平滑性。

4. 约束条件添加：

- 添加约束条件，包括状态约束（如速度、加速度约束）、控制输入约束（如转向角、加速度约束）、系统动力学约束等。

5. 优化求解：

- 将MPC问题转化为一个优化问题，一般是二次规划（QP）问题。

- 使用优化求解算法（如内点法、梯度下降法等）求解优化问题，得到最优的控制输入序列。

## 碰撞检测

这三种碰撞检测算法——GJK、AABB和OBB

1. GJK (Gilbert-Johnson-Keerthi) 算法：

- GJK算法是一种用于检测两个凸形状是否相交的算法。

- 它通过迭代寻找最近点对来判断两个凸形状是否相交。

- GJK算法的核心是通过迭代寻找包络球（Minkowski差集的包围球）来逼近凸形状的交点，直到找到交点或确定两个凸形状不相交。

2. AABB (Axis-Aligned Bounding Box) 算法：

- AABB算法是一种用于快速检测碰撞的简单而有效的算法。

- 它将物体用轴对齐的最小边界框（AABB）来表示，使得边框的轴与坐标轴对齐。

- AABB算法通过比较两个AABB是否相交来判断物体是否相交。虽然不如OBB精确，但计算简单且高效，特别适用于快速排除不可能的碰撞情况。

3. OBB (Oriented Bounding Box) 算法：

- OBB算法是一种用于检测物体碰撞的更精确的算法，相对于AABB，它允许包围盒的旋转。

- OBB算法通常由包围盒的中心点、方向向量和尺寸组成，这些信息可以描述盒子的方向和形状。

- OBB算法在需要更准确的碰撞检测时特别有用，因为它能够更好地适应物体的形状，但与之相关的计算复杂度相对较高。

在实际应用中，这些算法可能会结合使用，根据具体情况选择合适的算法来进行碰撞检测。例如，可以使用AABB快速排除大部分不可能的碰撞，然后再使用更精确的GJK或OBB算法来进行更细致的检测。

# 涉及的概念

1. Continuous time lateral dynamic model
2. Bilinear/Eular transfomation to discretize a continuous state space model to a discrete model.
3. QP求解 CVXGEN