# 动力学模型开发计划

## 目标

构建乘用车/卡车动力学模型，满足自动驾驶仿真的需求，包含PnC及传感器等算法的仿真需求。

模型具有模拟不同动力系统车型的能力，如燃油车，纯电动，混动等车型（混动车型变体较多，优先支持纯油和纯电）；具有模拟不同驱动方式车型的能力，如前驱，后驱，四驱等；支持为不同车型配置不同参数集。

动力学模型和动力学参数分开存储，动力学模型可以加载不同的动力学参数集。动力学参数集以json文件格式存储（protobuf工具链生成的json），由前端编辑生成。

## 动力学模型开发

### 2.1 模型库的组成

包含以下模型库：

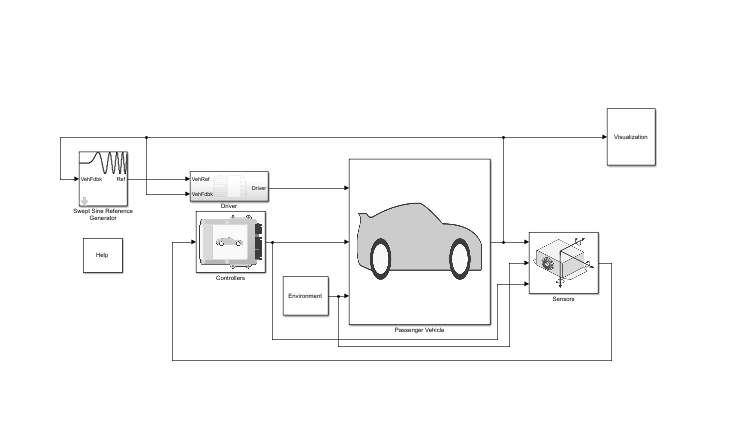
* 发动机
* 变速箱
* 电池
* 电机
* 转向
* 分动箱，差速器
* 悬架
* 制动
* 车身
* 自动驾驶员

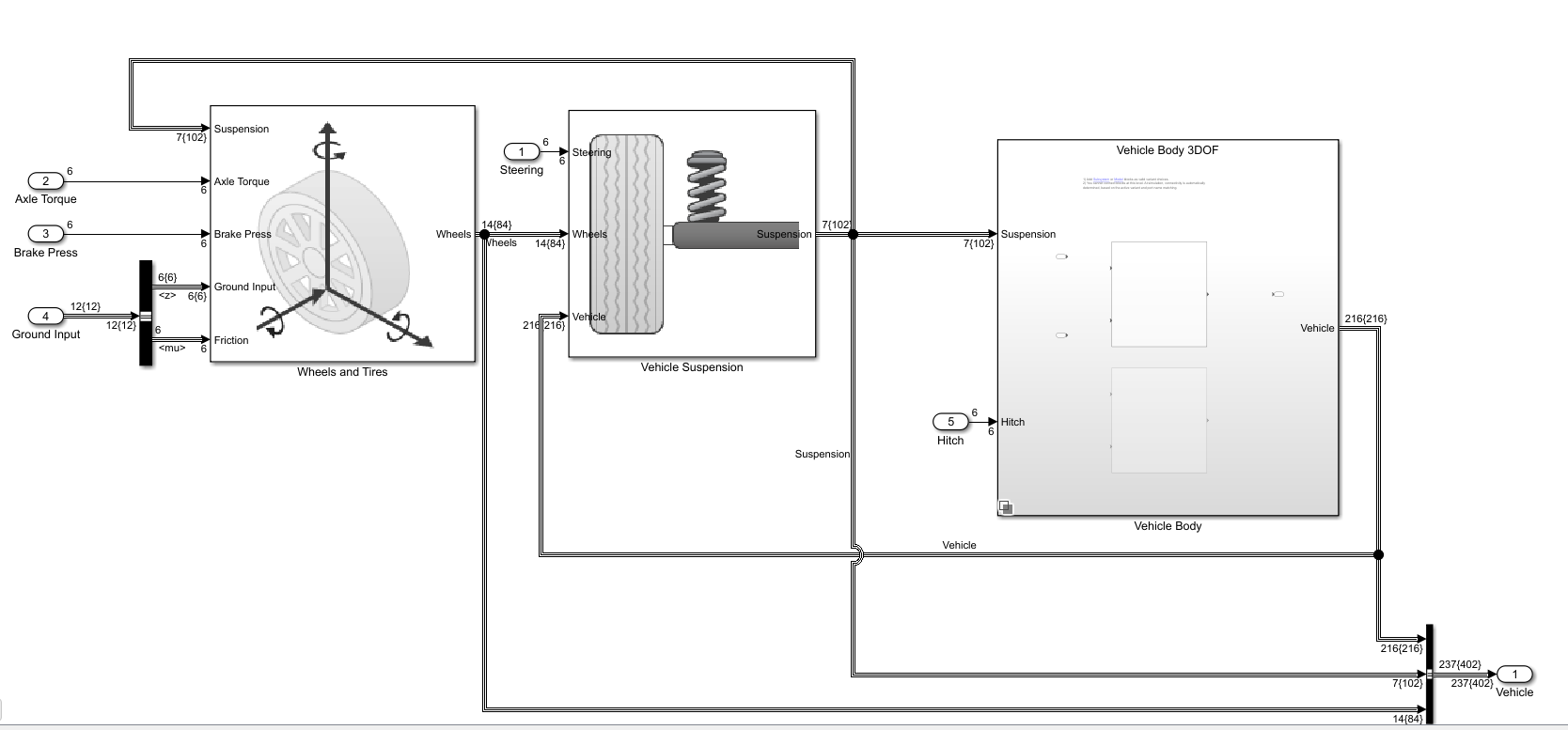
各个模型子系统通过simulink s-function封装，单输入单输出（均为simulink bus结构体），能够从动力学参数集（protobuf json文件）中加载对应的子系统模型参数，并进行初始化。模型能适配不同大小的数组型参数。子系统模型自动代码生成后，需要立即开展此项工作。

各个子系统要提前定义好输入输出的simulink bus结构体，及模型参数集解析功能。

### 2.2 构建动力学模型

构建纯油和纯电的车辆动力学模型，支持前驱，后驱，四驱。开发过程中，逐个模块替换为s-function，保证功能正常。





### 2.3 参数化工具开发

提供给前端动力学参数集的json文件，前端加载，修改和保存动力学参数集。

### 2.4 其他

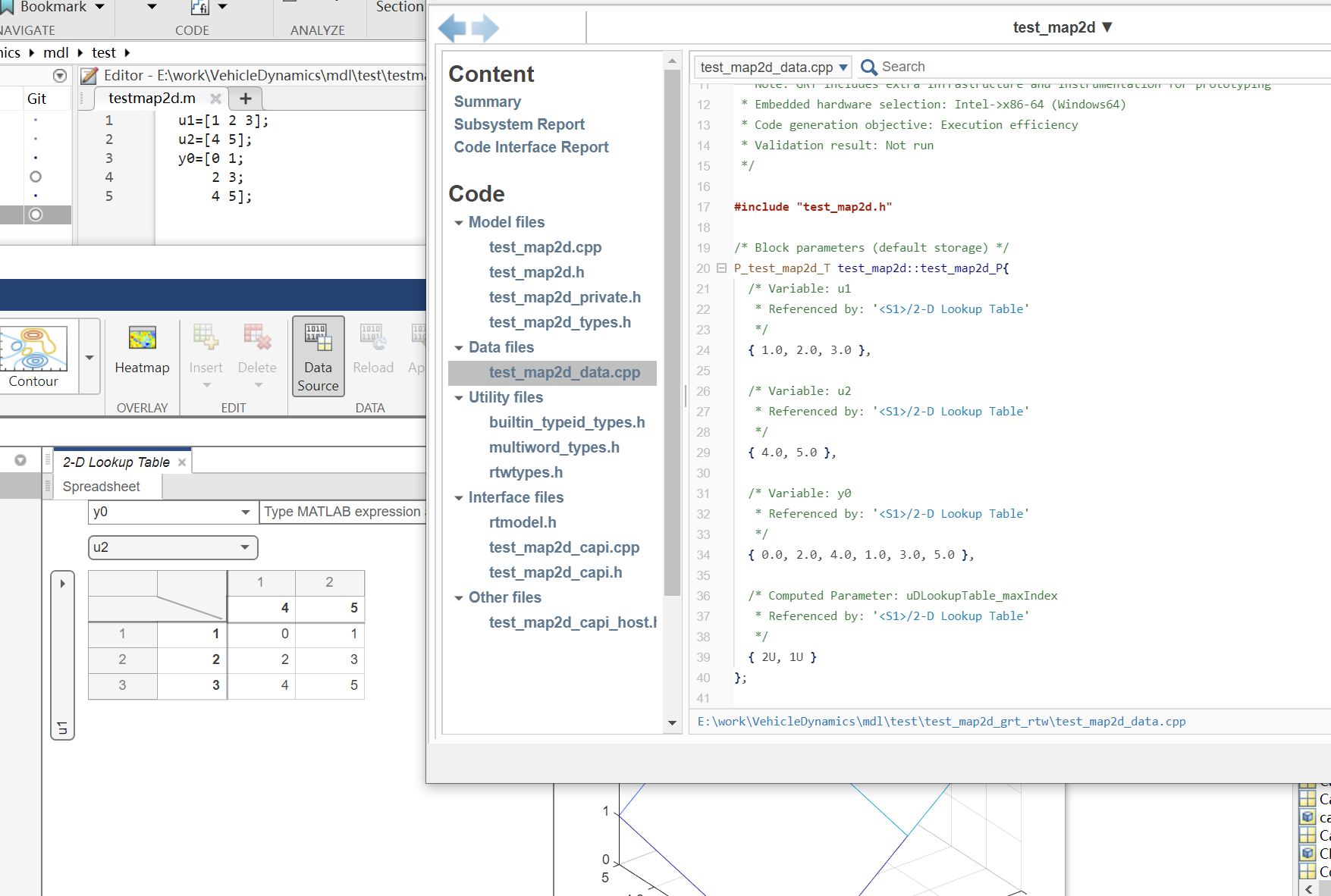
#### 2.4.1 simulink模型规则

* Simulink bus 结构体中的变量名为小写，单词之间以”\_”分割，且必须带有单位后缀，如motor\_speed[rad|s]。
* Simulink中的子系统模型集成到车辆动力学模型中，必须以Reference Model的形式。
* 每个子系统的s-function只能又两个参数，均为文件路径，即数据结构为char \*，第一个为license文件路径，第二个为参数集（protobuf json文件）路径。

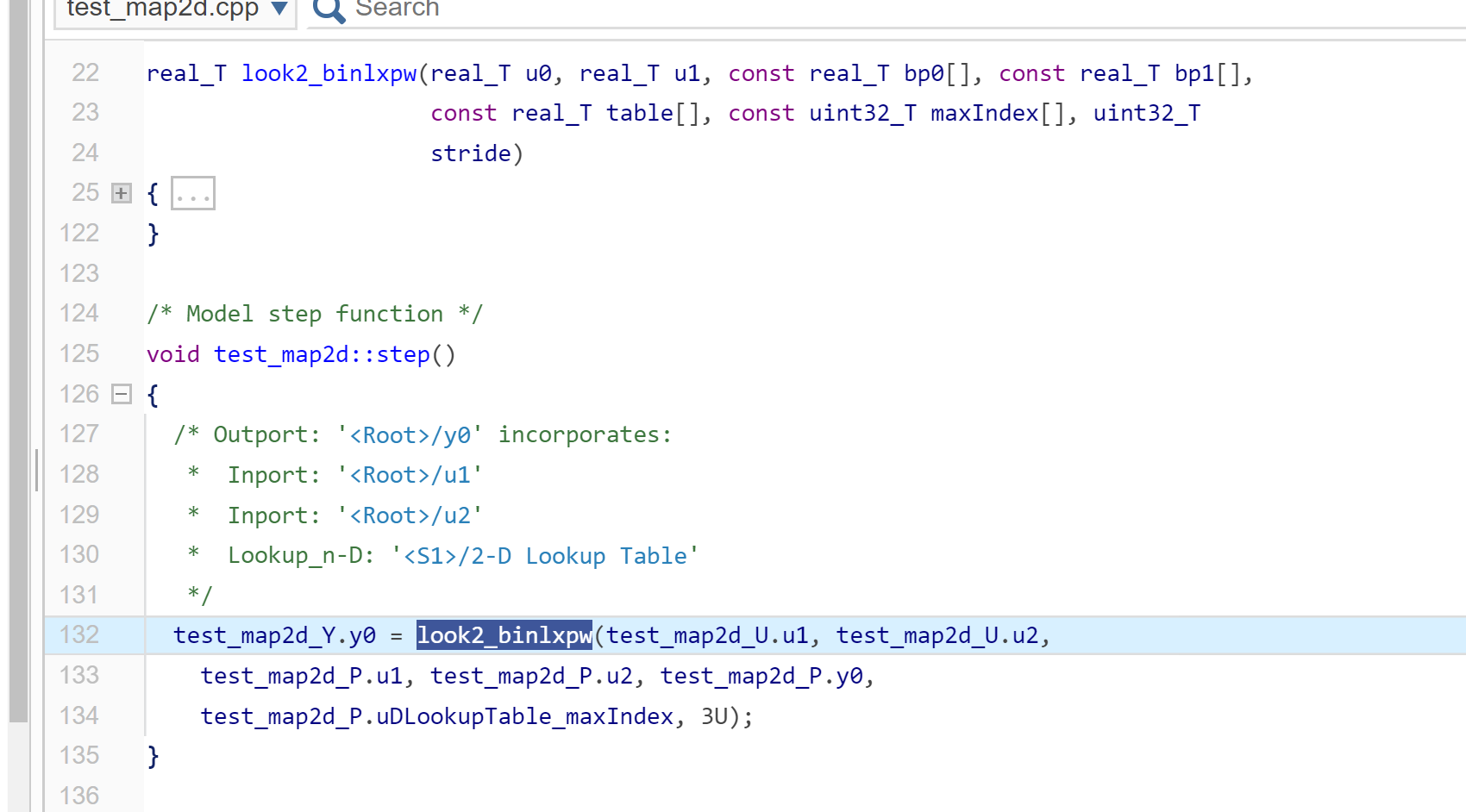
#### 2.4.2 C++命名规则

* 类名以大写开头，如class Foo{};
* 变量名为小写，单词之间以”\_”分割，如int x\_var\_name=0;
* 类成员变量命名以m\_开头，如 int m\_x, m\_y;
* 函数采用驼峰命名，如 void setValue(const int &x\_input); int getValue();
* 公共namespace为tx\_car, 每个子系统都属于此命名空间，且有自己的namespace， 如tx\_car::chassis::Foo, 子系统命名空间由各个开发工程师确认
* 不允许使用using namespace

#### 2.4.3 Simulink 2-d lookup table



2d map按照列排列，row\_num为u1的size，col\_num为u2的size，生成的map 2d数组在一维数组中按列排列。



Map2d的查表函数中的stride为u1的size

## 开发排期

* 动力学模型子系统划分：

1. 动力（包含虚拟ECU，电池模型等）
2. 传动（包含虚拟ECU）
3. 转向（包含虚拟ECU）
4. 底盘及车身
5. 其他，如自动驾驶员

* 构建支持纯油，纯电，前驱，后驱，四驱的动力学模型
  + 开发人员负责各自的子系统开发
  + 子系统接口bus定义，有需要的话开发人员同步确认
  + 子系统开发完之后，通过model reference形式接入到动力学模型中
  + 子系统模型protobuf参数定义及s-function化开发工作同步进行