Project 1

本项目希望大家根据 Stanley 算法实现车辆的横向控制，结合上一章的 PID 算法实现轨迹跟踪。基本的系统框架已经给出，仅需要完成 `src/stanley_control/src/stanley_control.cpp` 中 todo 部分。

```
// /** to-do **/ 计算需要的控制命令，实现对应的stanley模型,并将获得的控制命令传递给汽车
// 提示，在该函数中你需要调用计算误差 ComputeLateralErrors
void StanleyController::ComputeControlCmd(
    const VehicleState &vehicle_state,
    const TrajectoryData &planning_published_trajectory, ControlCmd &cmd) {

}

// /** to-do **/ 计算需要的误差，包括横向误差，纵向误差
void StanleyController::ComputeLateralErrors(const double x, const double y,
                                              const double theta, double &e_y,
                                              double &e_theta) {

}
```

实现 todo 后，需到项目的根目录运行 `catkin build` 进行编译，编译通过后运行如下命令：

控制台 1：启动 `carla_ros_bridge`

```
source ./devel/setup.bash
roslaunch carla_ros_bridge carla_ros_bridge_with_example_vehicle.launch
```

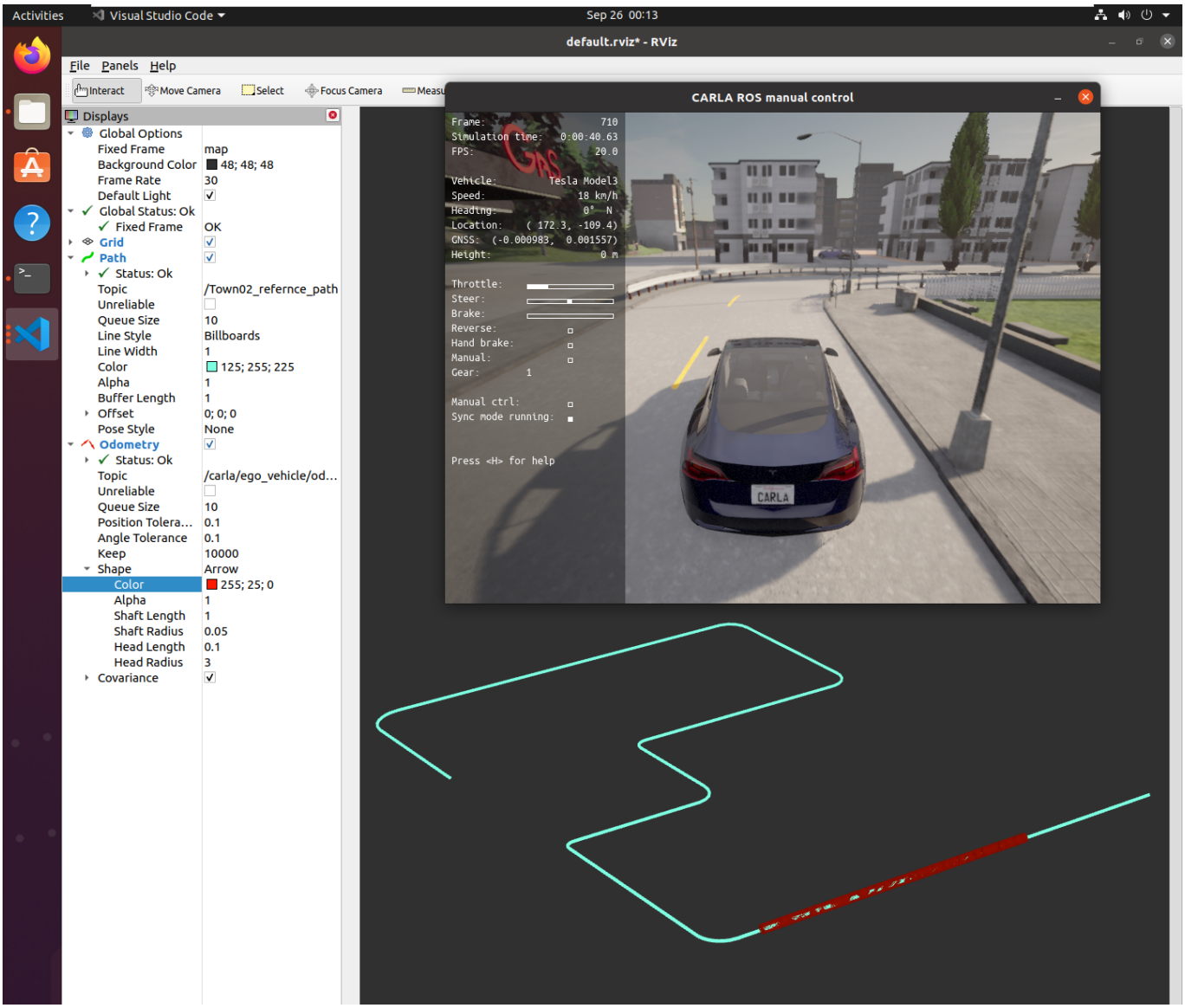
控制台 2：启动 `stanley_control` 节点

```
source ./devel/setup.bash
rosrun stanley_control stanley_control
```

控制台 3：打开 `rviz` 进行可视化

```
rviz
```

在 rviz 中可以通过订阅 `/carla/ego_vehicle/odometry` 获得车辆参考的定位消息，订阅 `Town02_refernce_path` 获得车辆参考路径的可视化消息。在左侧工具栏可根据自己的喜好调整曲线样式。运行效果如下：



作业以 用户名 + 第二章作业 的格式命名，需上传控制效果的录屏以及修改后的 `stanley_control.cpp` 文件。