登峰车队无人车解决方案

陈思聪 孙伟奇 阮正鑫 曾昱程

2019年9月

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

目录 Contents

目录 Contents

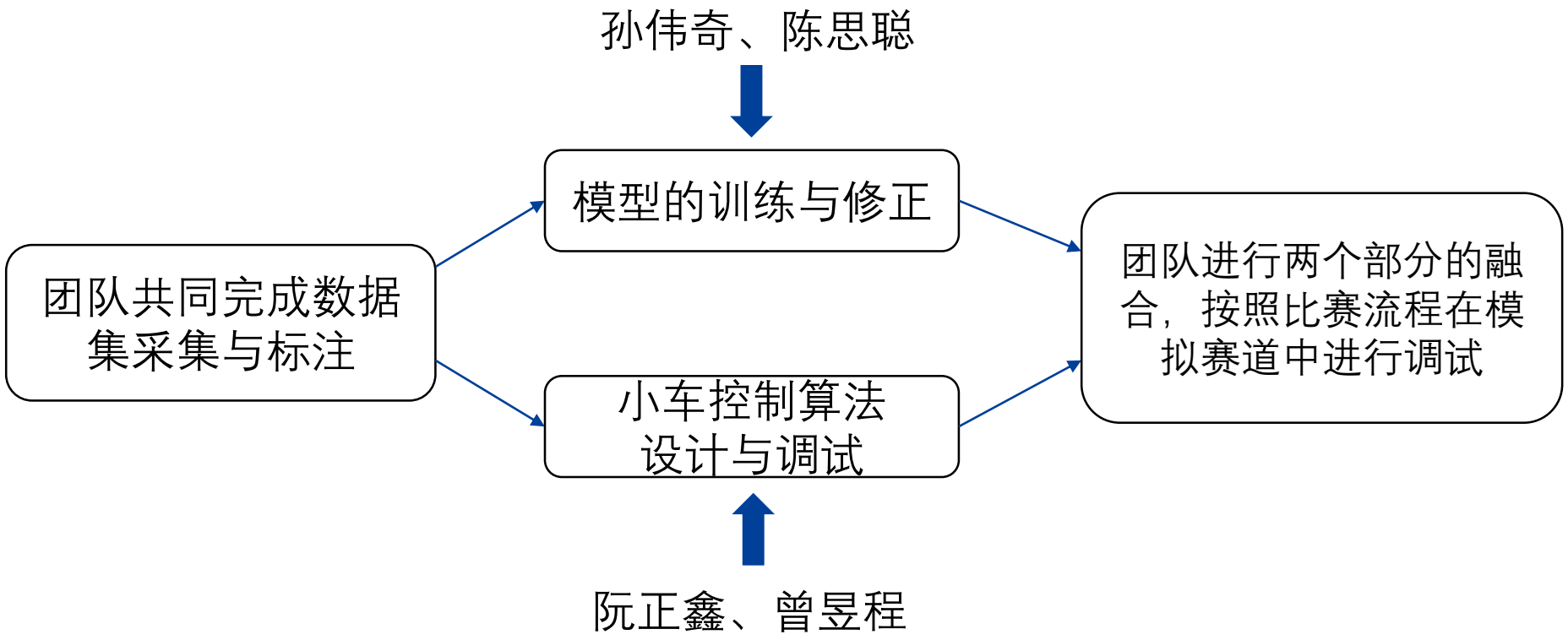
登峰车队团队成员

- **陈思聪（队长）**：上海交通大学电院自动化系2018级硕士学生，研究方向为自动驾驶、大数据。
- **孙伟奇**：上海交通大学电院自动化系2018级硕士学生，研究方向为自动驾驶。
- **阮正鑫：**上海交通大学自动化系2016级本科生。
- **曾昱程：**上海交通大学自动化系2016级人工智能方向本科生。

团队成员参赛经历
2018年上海交通大学校内无人驾驶挑战赛
2019年中美青年创客大赛
第十四届全国大学生交通科技大赛
第十四届全国大学生智能汽车竞赛

团队分工

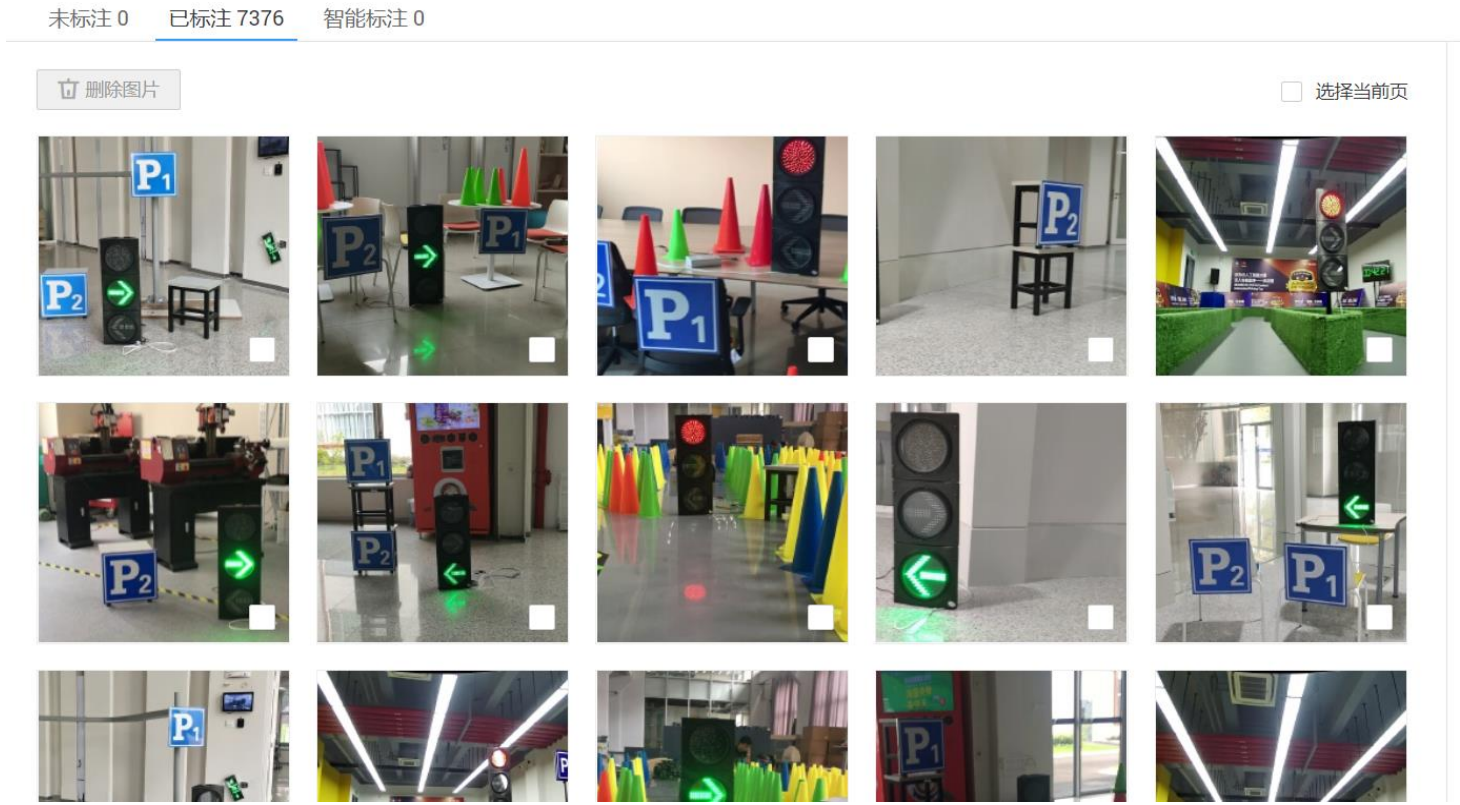
- 我们将这次竞赛的工作分为两个部分：**①模型的设计与训练，②小车的控制算法**



目录 Contents

数据集准备

- 手机拍摄 6213张——检测置信度约为0.6~0.7
- HiLens 补充拍摄 1163张——置信度提高到0.95+



数据集准备

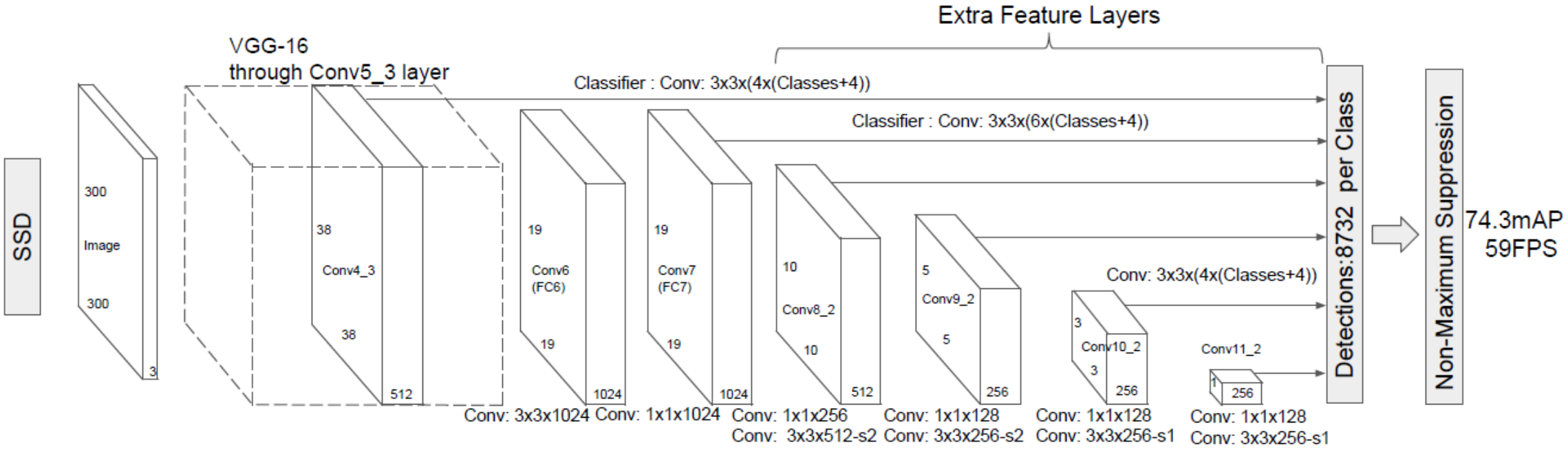
全部标签 5

标签	数量	操作
left	2299	
red	2449	
park2	5091	
right	2261	
park1	4592	

```
item {
  name: "none_of_the_above"
  label: 0
  display_name: "background"
}
item {
  name: "red"
  label: 1
  display_name: "red"
}
item {
  name: "left"
  label: 2
  display_name: "left"
}
item {
  name: "right"
  label: 3
  display_name: "right"
}
item {
  name: "park1"
  label: 4
  display_name: "park1"
}
item {
  name: "park2"
  label: 5
  display_name: "park2"
}
```

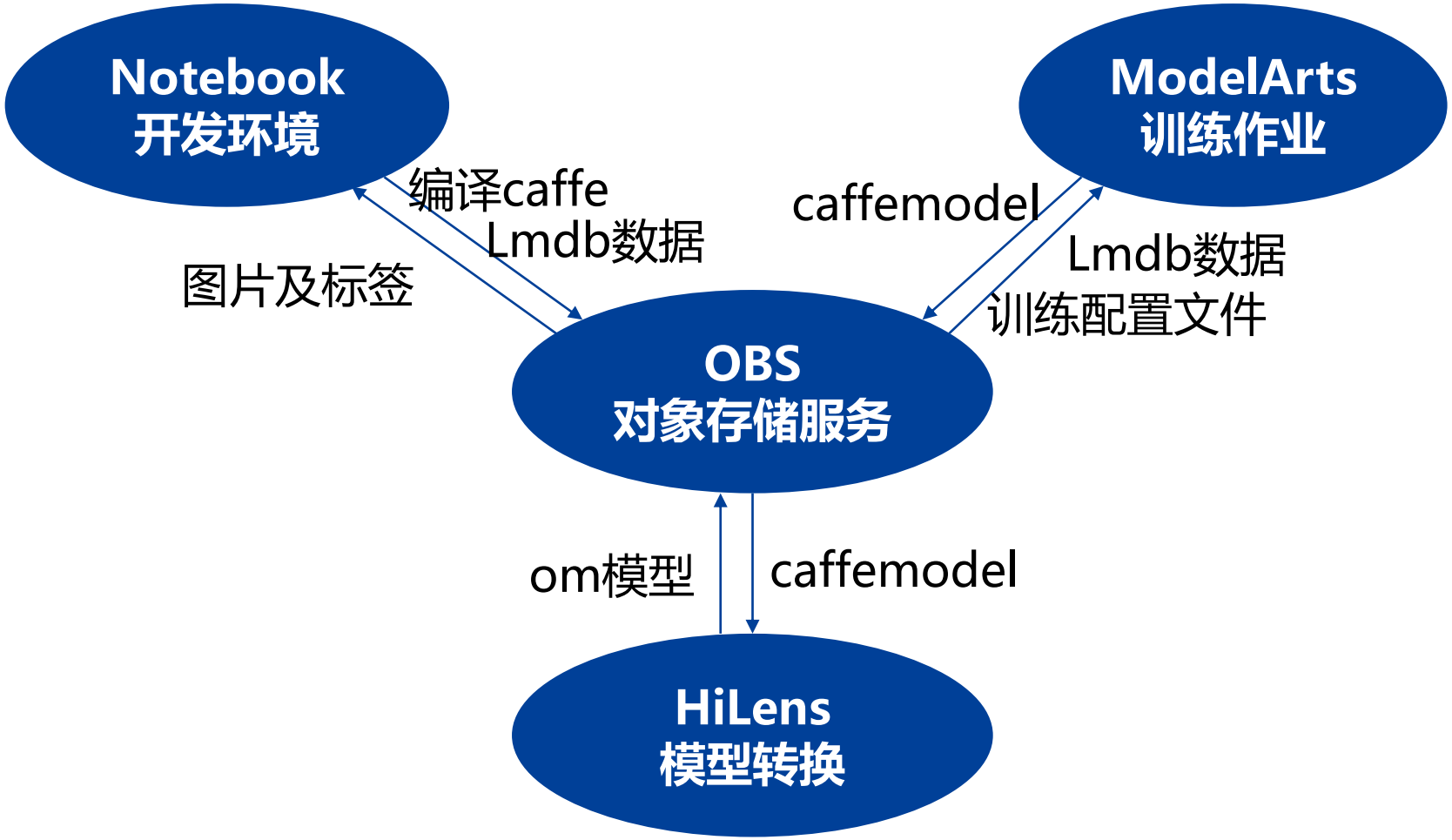
训练算法

- 采用SSD单步目标检测算法
- 相比Faster R-CNN有明显的的速度优势
- 相比YOLO有mAP的优势



W. Liu, D. Anguelov, D. Erham, C. Szegedy, S. Reed, C. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in Proceedings of European conference on computer vision, Springer, 2016, pp. 21-37.

使用华为云平台训练


Notebook
开发环境
ModelArts
训练作业
编译caffe
Lmdb数据
图片及标签
caffemodel
Lmdb数据
训练配置文件
OBS
对象存储服务
om模型
caffemodel
HiLens
模型转换

HiLens 部署

将转换后的om模型文件导入HiLens控制台

设备注册到HiLens平台，连接云侧端侧

从 OBS 导入模型

技能开发

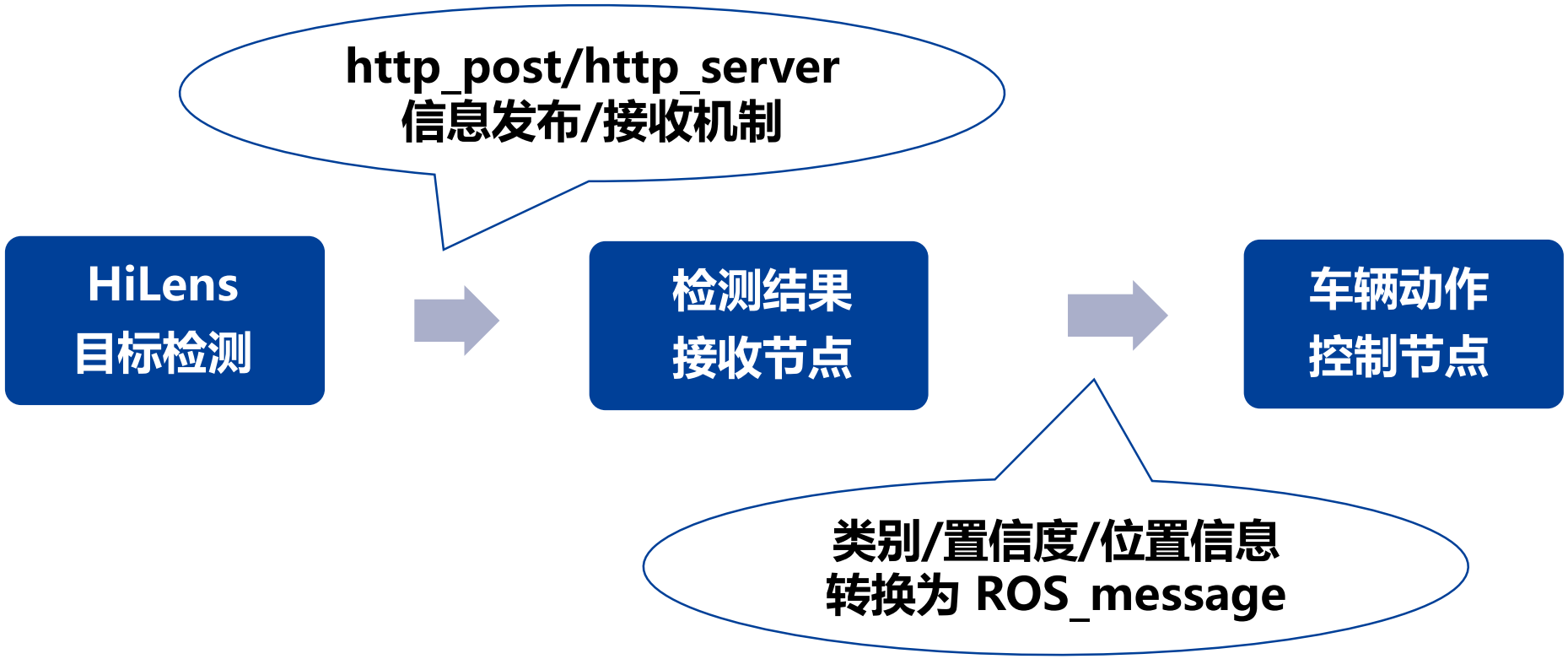
设备注册

技能部署

对HiLens摄像头的每帧图像进行推理

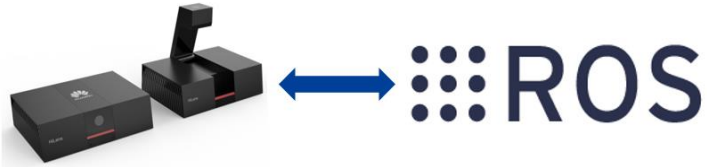
在端侧执行技能，并将检测结果输出

HiLens 与 ROS 通信



HiLens 与 ROS 通信

- **目标检测信息（HiLens端用http_post方式发送）：**



- **①报文信息初始化**

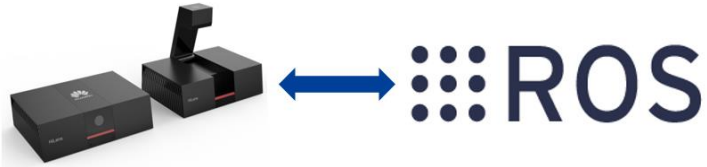
  在 HiLens (192.168.2.111) 端初始化报文信息，将 TX2 (192.168.2.1) 设置为接收报文消息的设备

- **②模型输出信息填充**

  在**循环推理**中，使用 urllib.request.Request 方法将检测结果的 **检测框位置、置信度、类别** 信息发送给TX2

HiLens 与 ROS 通信

- **检测结果预处理（ROS端将post消息转换到ROS消息）：**



- **①数据包解析**

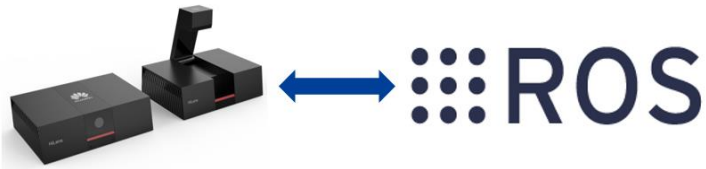
  TX2端接收到检测信息后，将其转换为字典格式，并使用字符分割的方式获取**检测框位置、置信度、类别** 的数值信息

- **②ROS消息发送**

  使用 rospy.Publisher.publish 函数将检测置信度和类别信息以 ROS_message 形式发出，供后续节点接收并控制车辆运动

HiLens 与 ROS 通信

- ROS 端下层节点响应（接收ROS消息并控制车辆运动）：



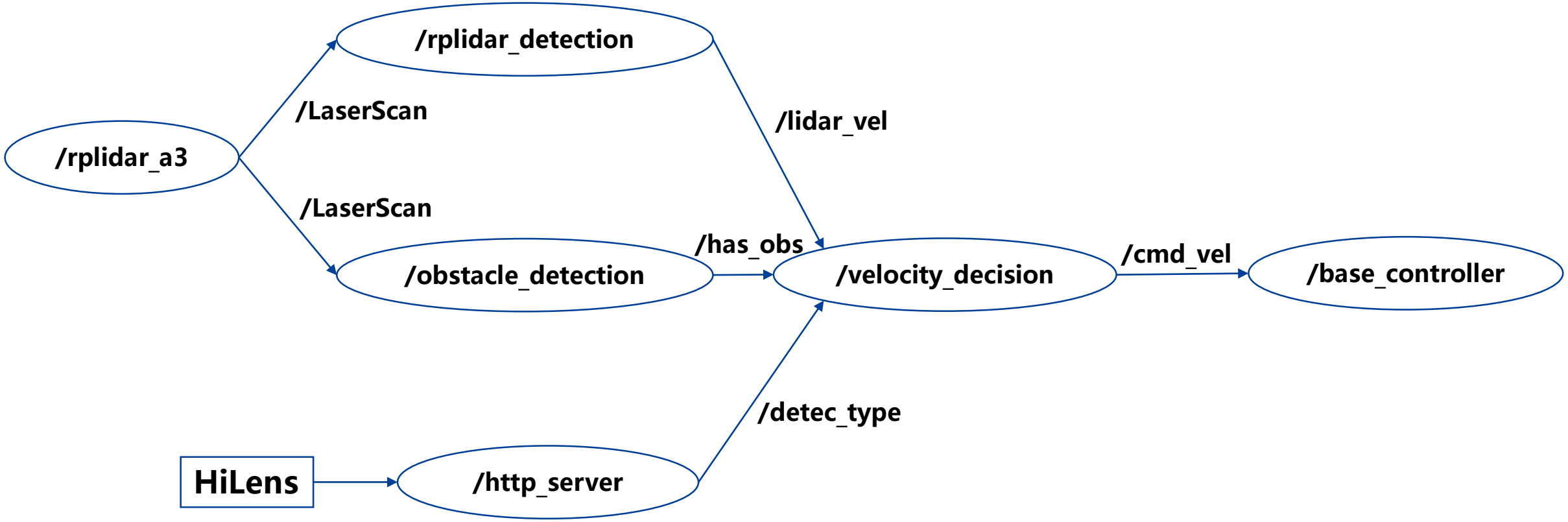
- ①ROS 消息接收

  用 rospy.Subscriber 函数接收预处理节点发出的检测结果

- ②运动指令发送，控制小车运动

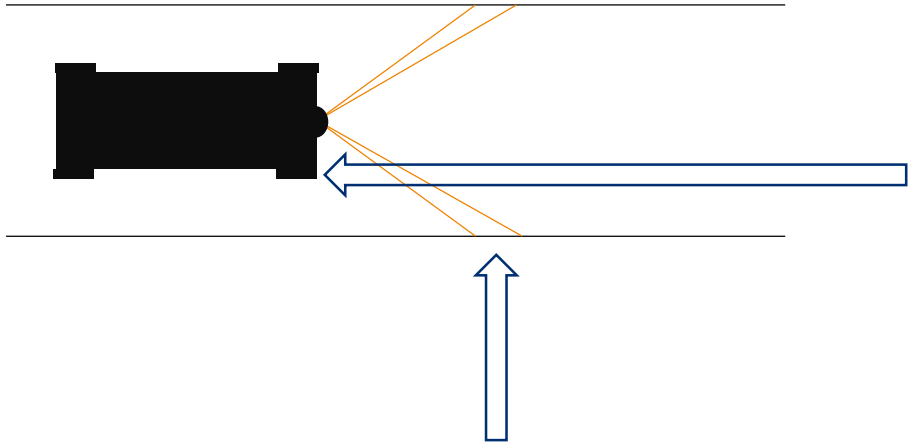
  在接收到左转、右转、停车位1、停车位2的检测结果后，将其他 ROS 节点计算出的速度指令赋给底盘，使其按照期望的控制指令运动

ROS通信拓扑图


/rplidar_detection
/LaserScan
/rplidar_a3
/lidar_vel
/LaserScan
/has_obs
/obstacle_detection
/velocity_decision
/cmd_vel
/base_controller
/detec_type
HiLens
/http_server

道路行驶方案

方案采取利用雷达探测左右前方的距离，控制距离相等，行驶于道路中央



舵机基础控制采用P控制，
计算左右前方的距离之差，
控制舵机的转向角度

探测侧前方40°~50°区间
取最小值，减小偶然误差

沿单边行驶：控制相应单边为一定值

红绿灯转向控制

1. 绿灯直接通行

定义全局变量，根据不同的绿灯方向进行更改，之后不再更改。

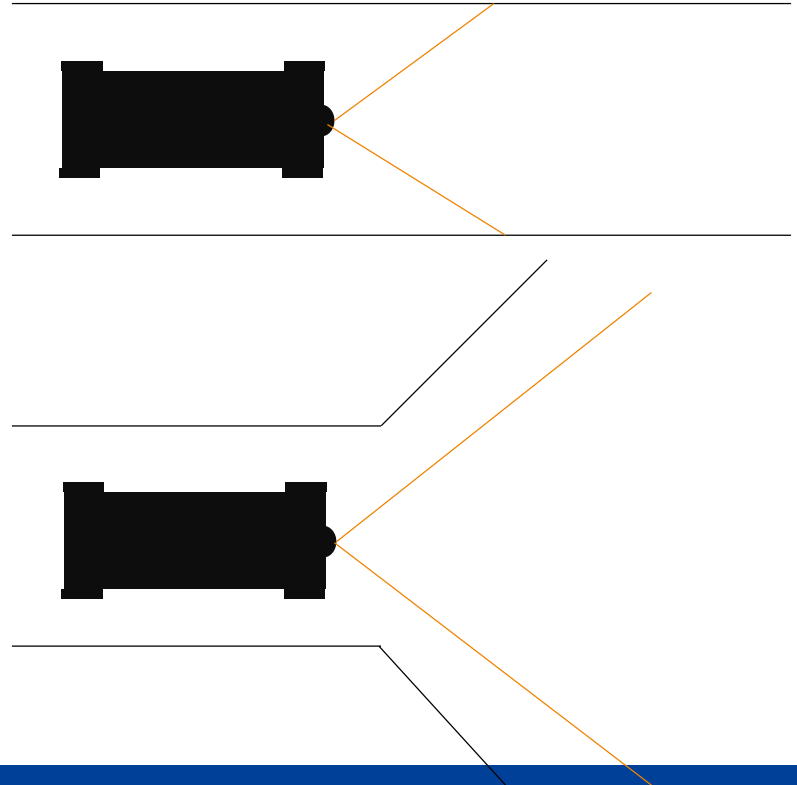
使用投票器进行判断，连续检测到多帧绿灯后才进行更改，避免误检测。

2. 路口进行相应转向

进行路口检测，检测到位于路口时，进行单边行驶，驶入相应道路为止。

完成环岛部分后，沿左墙行驶，以完成U型弯

路口检测：雷达扫描数据与普通道路有较大区别，距离增大



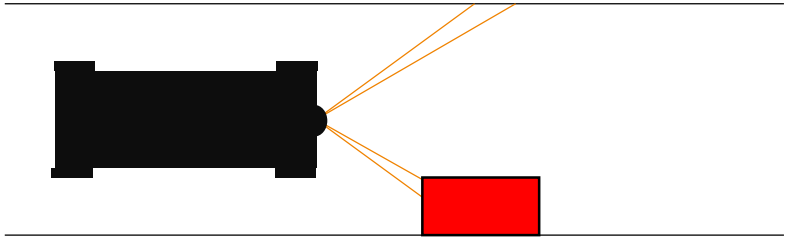
障碍物避障

1. 静态避障

提前检测到障碍物并进行响应

具有提前预见功能，可绕过障

碍物障碍物。

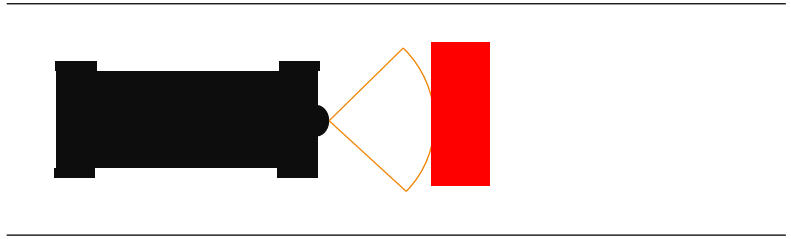


2. 动态避障

此任务具有最高优先级，实时检

正前方范围内障碍物，如出现

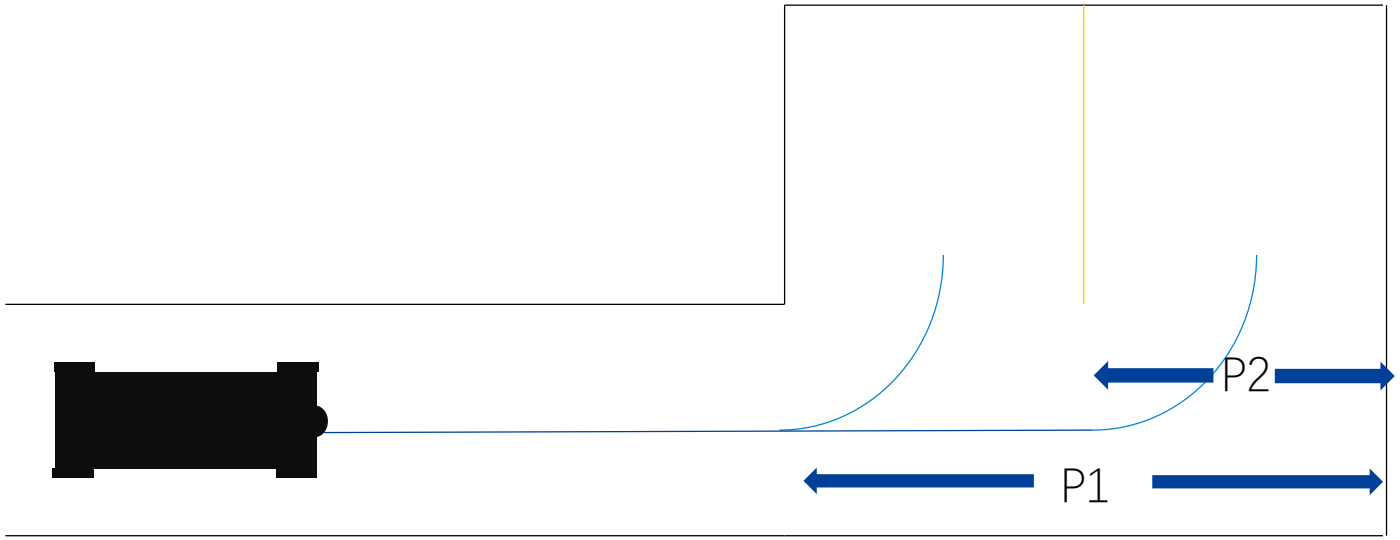
在区域内，则直接刹停。



自动泊车

- **步骤：**

1. 检测到车库号后，实时探测正前方距离信息，准备进行入库
2. 根据车库号选择合适的距离，当车辆到达距离时，启动入库程序
3. 根据测试参数，进行定时间的满舵转向，后直行至前方墙体距离达到设定要求，停车



目录 Contents

算法中三大亮点

标志物判断的投票器

- 当 HiLens 识别红绿灯与停车标志时，由于光线、角度、距离等等因素的影响，会出现一定的**误识别**。
- 针对这个问题，我们在 Python 中设计了一个**投票器**程序。
- 当 Hilens 传出 ROS 信息时，信息进入投票器类的数组中，投票器的输出是该数组中的**众数**。

Frame1	Frame2	Frame3	Frame4	frame5	输出
0	0	0	0	0	红灯
0	0	0	1	1	红灯
0	0	1	1	1	右转
1	1	1	1	1	右转

自适应舵机P控制参数

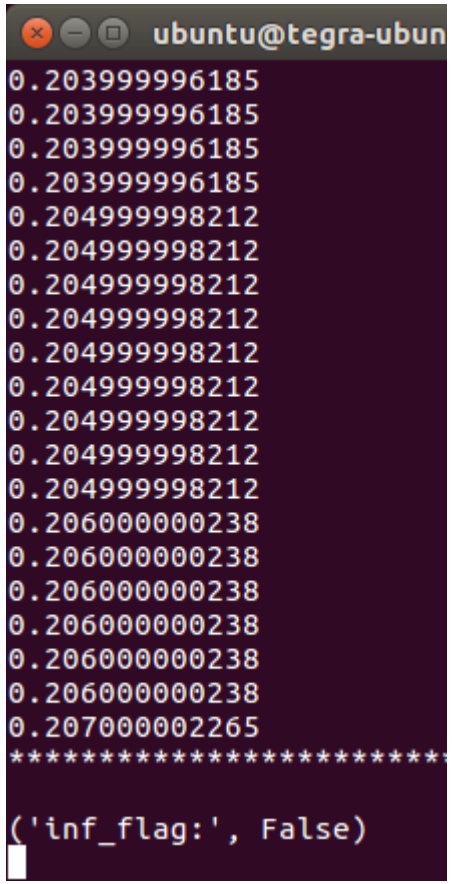
- 在小车控制算法的设计与调试中，我们设计了只有P参数的 PID 控制器。但在调试中我们发现，这个控制器能够完成巡线，但当遇到较大的急弯以及路边的障碍物时，会产生碰撞。

- 根据我们日常驾驶的经验来说，当车辆转弯时不能贴路边过近。针对这个问题，我们采用了**自适应的 P 控制参数**。

- 将前几时刻的舵机转角与当前时刻的控制量相关联，经取前几时刻控制量的均值，加入当前时刻的舵机控制量，达到了避免方向打的过急造成的擦碰问题。

雷达无效范围处理

- 在设计中，针对动态障碍物，我们设计的程序是使用雷达探测车正前方左右10°，当雷达探测到的距离的最小值小于0.4m时，认为车前方出现动态障碍物，小车急停。
- 由于雷达有一定的检测范围，当障碍物的距离小于15cm时，雷达无法正常的输出距离，而是INF的不可读取的输出。
- 针对这个问题，我们对这个雷达无效范围进行了处理。当小车一旦识别到动态障碍物后，且雷达输出数据为INF时保持停车状态，从而解决此问题。



目录 Contents

改进方向

- **建图与导航**

  本次比赛我们团队使用的方法主要是基于路沿距离检测的闭环反馈控制，但没有建立地图，也没有使用基于地图和车辆位置的导航功能。

  未来可以考虑通过 gmapping、hector-slam 等方式建立地图，用 AMCL 等方法导航来提高无人小车对于其他更加复杂场景的适应能力。

- **速度自适应**

  在小车运动控制方面，我们给的纵向线速度都是定值。未来可以加入一定的变速控制方法，例如直道加速、弯道降速通过等等，以使车辆既安全又高效地完成运行目标

目录 Contents

相关建议

- 本次比赛的赛道比较狭窄，因此在车辆运行的过程中对于车身姿态的要求比较高，如果出现一点点的偏差就有可能导致剐蹭墙体或无法绕过而停车。
- 考虑到室内场馆灯光较强，建议识别目标物如标志牌尽量采用哑光材质。
- 车载传感器较少，对于更加接近现实场景的无人驾驶场景，轮速里程计、惯性测量单元IMU、以及用于传统图像处理的摄像头也是需要配备的硬件，这样可以使开发出的功能更多样、更可靠。

相关建议

- 本次比赛中对视觉方面的处理要求较少，建议后续赛事可以加入车道线检测跟踪以及更多的交通元素的识别检测，例如前方限速、人行横道等警示标志，窄桥、隧道、立交等更为复杂的场景。
- ModelArts的训练作业中的预置算法框架种类单一，目前只有MXNET和TensorFlow。
- 常用框架中的只提供官方源码编译的caffe，用户需要在Notebook开发环境里编译好所需版本再经OBS传入训练作业，略繁琐。能否开放对训练作业的容器直接进行访问的交互界面。

谢谢！