

# **车载自组网的通信和控制模块设计和实现**

**毕业答辩**

01

## 研究背景与目的

Research background and  
purposend

02

## 整体研究思路

Overall research idea

03

## 研究内容与方法

Research contents and  
methods

04

## 结果分析

Analysis of the result

05

## 总结与展望

Summary and prospect



# 01

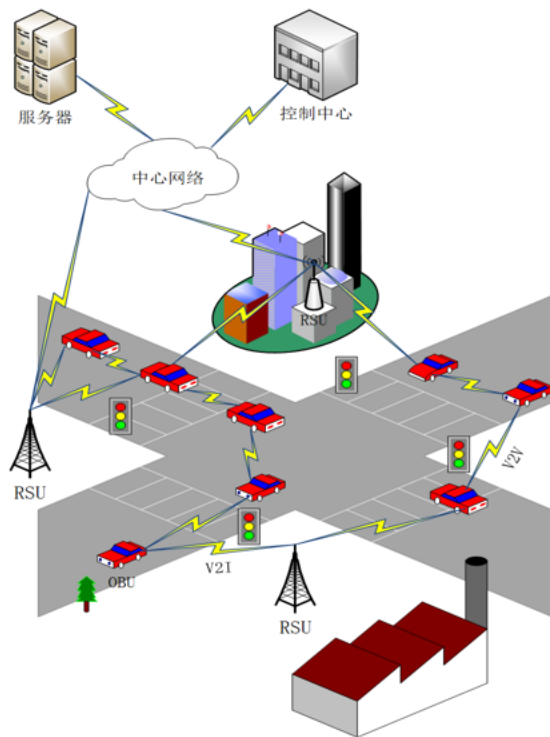
## 研究背景与目的

Research background and  
purposend



# 车载自组网简介

- 车载自组网 (Vehicular Ad Hoc Networks, VANET) 是一种无线自组网络 (AD Hoc Network), 它是基于移动自组织网络 (MANET, Mobile Ad Hoc Networks) 中专用于车辆交通、道路交通的网络形式。
- VANET的目标是在道路上搭建一个自组织的、临时分布形式、部署方便、扩展性强、抗毁性强, 同时时延相对较低, 成本相对较低, 且能够依靠现有道路设备的车辆间通信网络。



# 车载自组网应用

## 道路交通&智能驾驶

---

车载自组网是智能交通系统应用的重要组成部分，其在交通事故预警、交通流控制以及为驾驶人员提供信息服务和增值业务等方面都具有巨大的潜力。其在道路交通中的应用主要分为道路安全应用、交通效率应用、商业和信息应用三种。

## 事故救援

---

车载自组网因其本身独特的优势，能够被应用在很多场合。比如在应急救援场合，当地震、火山、泥石流等自然灾害发生时，公网信号往往会在事故区域直接消失或者减弱。此时事故救援人员必须立刻结合现场情况，针对性的组建无线传输通信网络，实现事故区域救援人员信息交互、事故区域-指挥中心信息联通等。

# 车载自组网的改进空间

- 目前大多无人车的车载自组网的应用中，自组网通信模块和车辆控制模块是两个**独立**的部分，车辆控制信号往往通过其他方式发送传递，这在部分场合中会出现一定的问题。比如在放射性事故中，人员无法进入事故现场，同时事故现场往往具备强大的信号干扰，此时事故现场外的基站，很难远距离的发送控制信号控制搜救车或搜救机器人，导致了严重的后果。
- 举例：日本核电站泄漏事件
- 因此，将**车载自组网通信模块与车辆的控制模块结合**，实现通过车载自组网传递控制信号具有一定的意义。

# 02

## 整体研究思路

Overall research idea



# 整体研究思路



## 车载自组网通信&控制系统

核心研究内容是将**车载自组网通信模块**与**车辆的控制模块**相结合，设计车载自组网的通信&控制系统。围绕该系统的**整体方案设计**、**软件联合仿真**、**真实场景应用**共计三个方面进行研究。



# 整体研究思路

## 整体方案设计

---

即如何从地面站通过人工的方式，实现控制车辆的过程。包括客户端、微控制器、传感器、协调器、车载端的配置。



## 软件联合仿真

---

设计并实现、完善基于Ardupilot等软件的车辆SITL联合在环仿真方法，并结合该方法对Ardupilot等开源平台进行再开发。



## 真实场景应用

---

进行了协同作业中车载自组网的节点部署研究。提出了基于改进A\*算法的路径规划和节点部署策略。

# 03

## 研究内容与方法

Research contents and  
methods

# 研究内容与方法

---

“

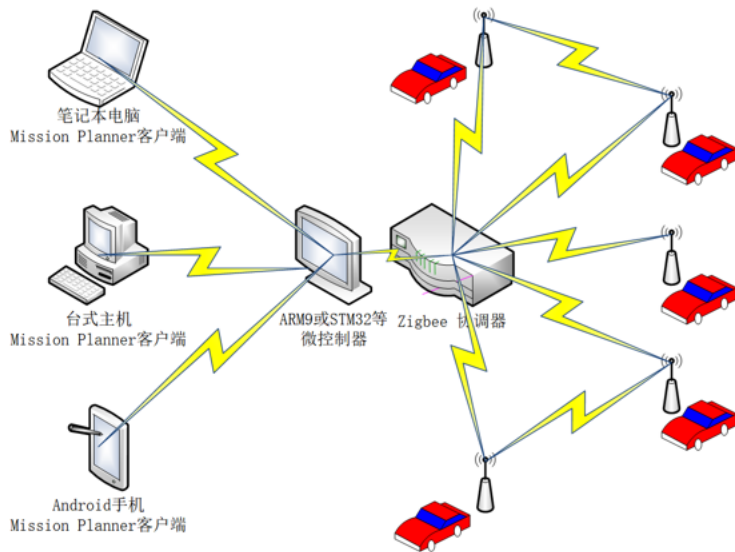
整体方案设计

”

---

# 整体方案设计

- 车载自组网通信、控制系统整体的组成主要由三个部分构成，分别是地面站、车载部分，以及通信协议，如图所示。

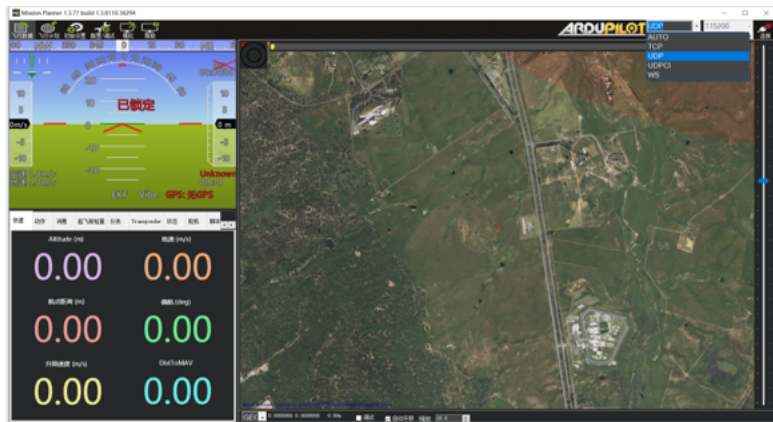


## 通信协议选取

- 本文在设计自组网时选用了Zigbee技术，而非目前车载自组网使用最多的Wifi技术。二者的区别主要体现在IEEE 802.11 和 IEEE 802.15.4 的不同。
- 其一，功耗。WIFI技术相比于Zigbee技术，功耗是一个劣势。
- 其二，WIFI技术相比于Zigbee技术的劣势在于其链路层不公平的现象。
- 其三，利用WIFI技术网络的车载自组网，设备若是试图接入网络，需要利用无线网络的部署。Wifi的优势则体现在其传输距离，速率。因此普通的车辆自组网，多用Wifi技术，而在部分场合下，Zigbee技术有着无可比拟的优势。本文也因此选用Zigbee技术搭建自组网。

# 地面站部分

- 移动设备上安装地面站 Mission Planner客户端，通过网络或者有线连接与微控制器通信，微控制器部分与 Zigbee协调器相连，通过 Zigbee协调器经路由器将信息传递给车载自组网的各个小车。

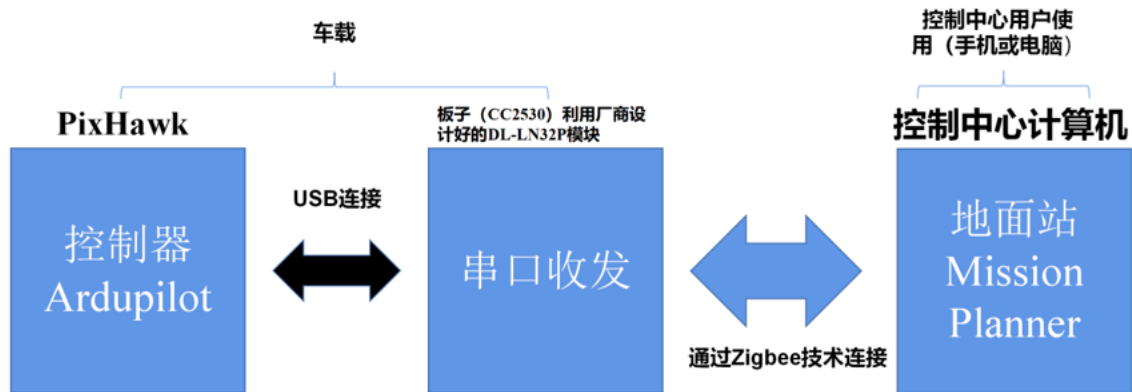


# 车载部分

- 在车辆节点处，采用Ardupilot飞控和基于Zigbee的CC2530开发板，Ardupilot负责控制小车，Zigbee技术用于传递控制信息，即通过车载自组网实现控制信息的传递，传统的设计中自组网通信网络和遥控通信网络是分开的。对于基于Zigbee的CC2530开发板，很多厂商给出了完整的基于TI的CC2530芯片的电路板，可以直接使用在小车模型上，例如DL-LN32P



# 整体方案流程





# 研究内容与方法

---

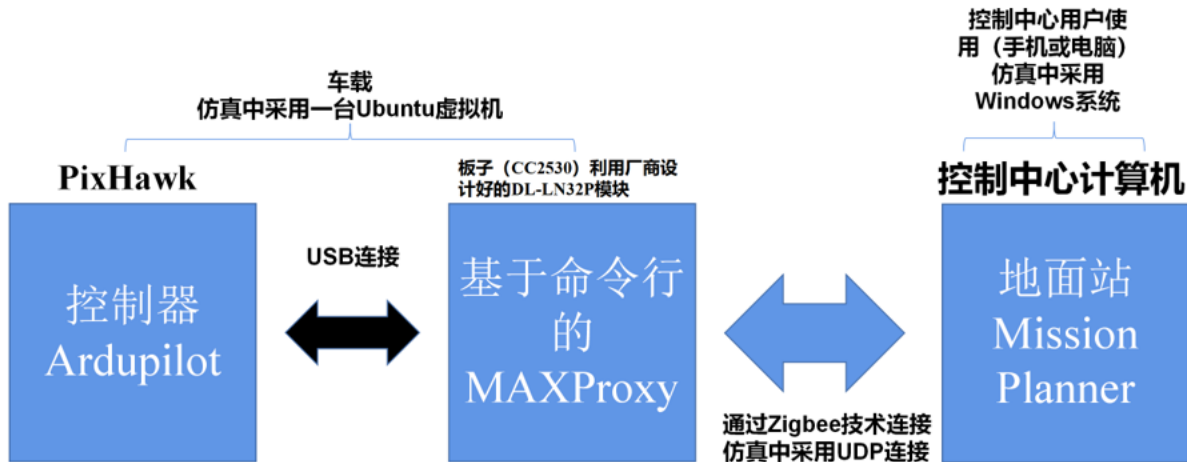
“

软件联合仿真

”

---

# 软件联合仿真



# Ardupilot解析及代码改进

- 由于Ardupilot小车Rover 部分的应用较少，很多Rover部分的代码都是来自于飞行单位的控制，很多地方出现了一些表达不清，或者有可能不合理之处，在本文的实现过程中，进行部分修改，使之更加适合本文的项目。
- Rover部分的test环节，很明显来自于直升机代码部分，甚至出现了“pitch forward to fly north”（俯身飞向北方）这种内容，予以删除并修改，同时对该部分代码进行改进。由于小车同飞机的差异，加大测试复杂度，定义新函数“复杂测试”。复杂测试不仅直接包括原来的所有测试，同时距离也进行了扩充。

# 核心参数结构

- 对于Ardupilot和Mission Planner 以及MAVProxy来说，最重要的数据莫过于航点数据。由于Ardupilot最初是运用在飞行单位上，因此有了航点这一名词，它表示的是“行驶轨迹的点”这一概念。其数据结构，无论是Mission Planner还是MAVProxy，均是一致的

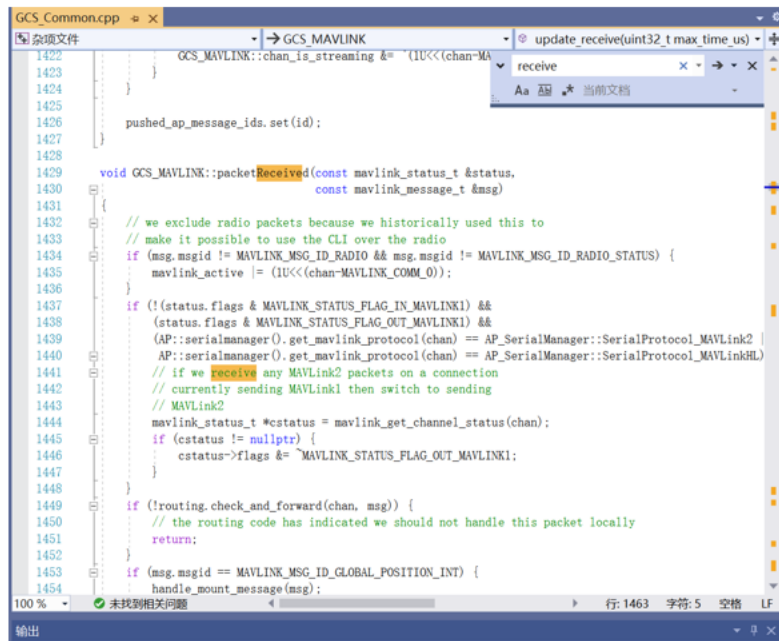
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

QGC WPL 110

0	1	0	16	0	0	0	0	-35.3618041	149.1586995	583.930000	1
1	0	3	16	0.00000000	0.00000000	0.00000000	0.00000000	-35.36224610	149.16034700	100.000000	1
2	0	3	16	0.00000000	0.00000000	0.00000000	0.00000000	-35.36350600	149.16212800	100.000000	1
3	0	3	16	0.00000000	0.00000000	0.00000000	0.00000000	-35.36459090	149.15721420	100.000000	1
4	0	3	16	0.00000000	0.00000000	0.00000000	0.00000000	-35.36145860	149.15575500	100.000000	1
5	0	3	16	0.00000000	0.00000000	0.00000000	0.00000000	-35.35995370	149.15991780	100.000000	1
6	0	3	16	0.00000000	0.00000000	0.00000000	0.00000000	-35.36340100	149.16517500	100.000000	1

# MAVLink控制信息分析

- 1) 收到mavlink消息
  - 2) Ardupilot控制器接受消息
  - 3) Ardupilot程序进行控制
- handleMessage就是刷新接收的关键内容，对应的是GCS\_mavlink.cpp中的函数。HandleMessage通过switch-case处理不同的事件，根据收到的mavlink消息id来划分事件：

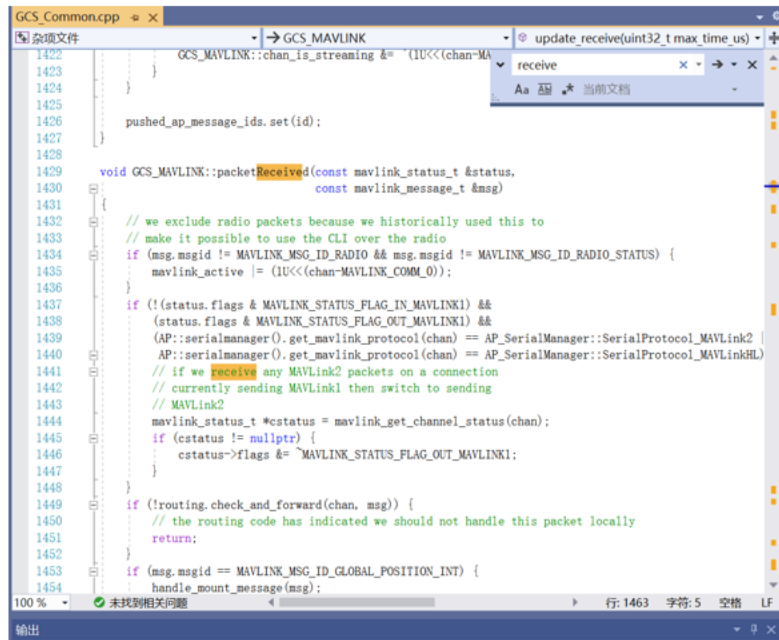


The screenshot shows a code editor window titled 'GCS\_Common.cpp'. A search bar at the top right contains the text 'receive'. The code is in C++ and shows a function 'void GCS\_MAVLINK::packetReceived(const mavlink\_status\_t &status, const mavlink\_message\_t &msg)'. The code includes comments and logic for handling MAVLink messages, such as checking for radio status and handling different message IDs. The status bar at the bottom indicates '100%' zoom, '未找到相关问题' (No problems found), and '行: 1463 字符: 5 空格 LF' (Line: 1463, Character: 5, Space, LF).

```
1422 GCS_MAVLINK::chan_is_streaming &= "(IU<<<(chan-MA
1423
1424
1425
1426 pushed_ap_message_ids.set(id);
1427
1428
1429 void GCS_MAVLINK::packetReceived(const mavlink_status_t &status,
1430                                 const mavlink_message_t &msg)
1431 {
1432     // we exclude radio packets because we historically used this to
1433     // make it possible to use the CLI over the radio
1434     if (msg.msgid != MAVLINK_MSG_ID_RADIO && msg.msgid != MAVLINK_MSG_ID_RADIO_STATUS) {
1435         mavlink_active |= (IU<<<(chan-MAVLINK_COMM_0));
1436     }
1437     if (!(status.flags & MAVLINK_STATUS_FLAG_IN_MAVLINK1) &&
1438         (status.flags & MAVLINK_STATUS_FLAG_OUT_MAVLINK1) &&
1439         (AP::serialmanager().get_mavlink_protocol(chan) == AP_SerialManager::SerialProtocol_MAVLink2 |
1440         AP::serialmanager().get_mavlink_protocol(chan) == AP_SerialManager::SerialProtocol_MAVLinkHL))
1441     {
1442         // if we receive any MAVLink2 packets on a connection
1443         // currently sending MAVLink1 then switch to sending
1444         // MAVLink2
1445         mavlink_status_t *cstatus = mavlink_get_channel_status(chan);
1446         if (cstatus != nullptr) {
1447             cstatus->flags &= ~MAVLINK_STATUS_FLAG_OUT_MAVLINK1;
1448         }
1449     }
1450     if (!routing.check_and_forward(chan, msg)) {
1451         // the routing code has indicated we should not handle this packet locally
1452         return;
1453     }
1454     if (msg.msgid == MAVLINK_MSG_ID_GLOBAL_POSITION_INT) {
1455         handle_mount_message(msg);
1456     }
```

# MAVLINK控制信息分析

## ■ MAVLINK初始化中的SYSID块



# 软件联合仿真



软件联合仿真  
详细内容请见  
软件展示视频

# 研究内容与方法

---

“

真实场景应用

”

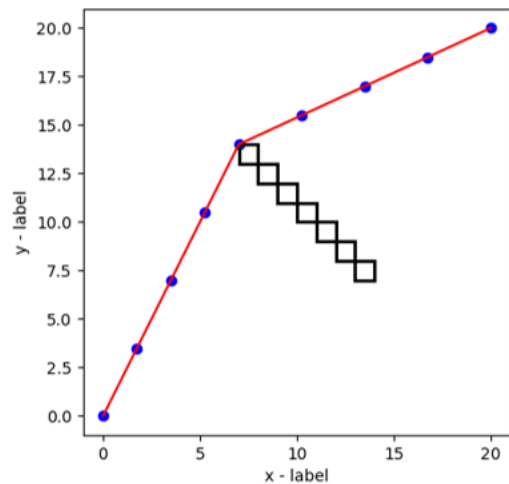
---



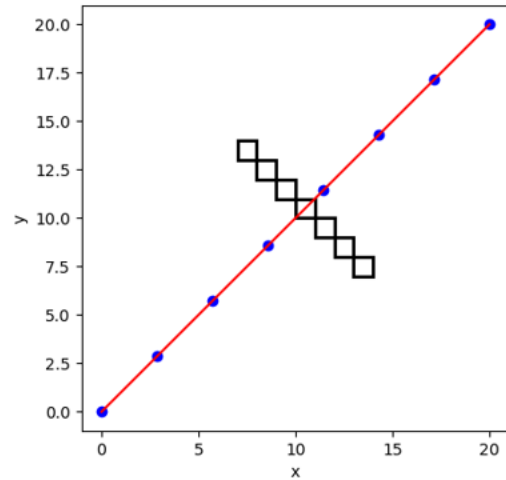
## 真实场景应用

前面的内容主要为车载自组网的具体设计及仿真方案，而在实际的工程项目中，因为自组网自身的特点，真正能够实现并利用本文设计的车载自组网结构仍然需要很多其他算法、策略的辅助。车载自组网在车辆协同中有着重要的意义，本文因此进行了协同作业中车载自组网的节点部署研究。提出了**基于改进A\*算法的路径规划和节点部署策略**。

# 真实场景应用



VANET OBU deployment diagram based on improved algorithm



# 真实场景应用

针对自组网的特点，本文提出了路径规划的一个新概念——可跳跃点这一概念，即在最终部署的时候在通信上能够跨越的障碍，称为可跨越障碍，该点为可跳跃点。但是其最终部署，并不能完全不考虑所有可跳跃点，因为其同样有一个调度成本。在任务执行车的行驶路径不变的情况下，其余中继车辆节点在行驶之初，仍要在一定程度上按照基础A星算法得到的路径去行进。这部分可以通过车辆协同算法实现，在本文的改进A\*算法中，对此进行了一定的简化，即通过设置函数——调度成本，来表示这个问题。即表达从初始轨迹移动到最终部署点的成本，这个函数可以是一个比例参数，也可以是一个复杂函数。满足调度成本要求的可跳跃点，称之为跳跃点。

- 在本文中，将调度成本设定为任务执行车距离目标点的最优距离/直线距离，意图粗略的表示其障碍的复杂度。该比值越大，障碍越复杂，同时调度的成本越高。



# 真实场景应用

- 本文给出了两个方法，其一是静态判定，即设置基于规则的判定方法，判定障碍是否可跨越。该规则可以设置为长度或其他单位。不过静态方法存在一定的弊端，即车辆在实际运动中是动态的，不一定会经过该节点，同时经过该节点不一定能达成最短路径。另一个则一个动态的方法，即双步长策略（Double-step strategy）。对车辆节点模型进行一定的简化，假定某场合下自组网稳定通信最大距离（或安全距离）为X米，设定 $X/2.8$ 为一单位，即在该模拟中，面对障碍能够实现通信。



## 真实场景应用



算法运行示例  
详细内容请见  
软件展示视频

# 04

## 结果分析

Analysis of the result

# 结果分析

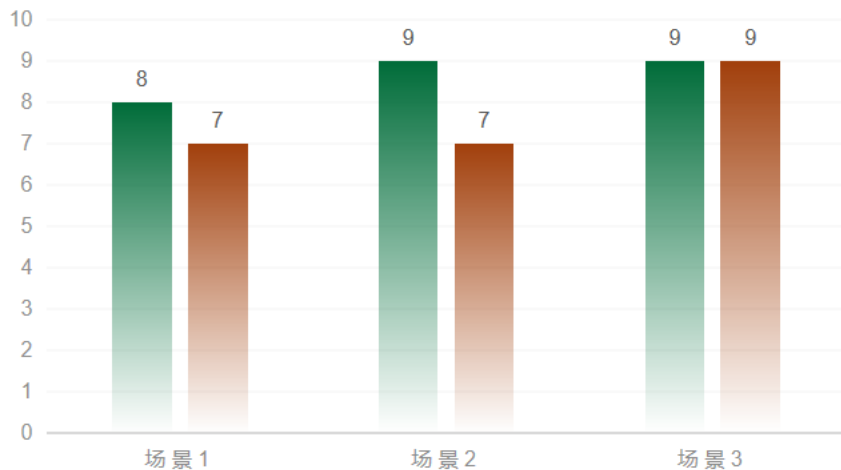
- 软件仿真部分能够较好的模拟出车辆的行驶及通信
- 选取三个场景对基于改进A\*算法的路径规划和节点部署策略进行评测：

```
(11,0) (11,1) (11,2) (11,3) (10,4) (9,5) (8,5) (7,6) (6,7) (5,8) (4,9) (3,10) (2,10) (1,10) (0,11)
0 0 0 0 0 0 0 0 0 0 0 *
0 0 0 0 0 0 0 0 0 0 0 * 0
0 0 1 0 0 0 0 0 0 0 0 * 0
0 0 1 0 0 0 0 0 0 0 0 * 0
0 0 1 0 0 0 0 0 0 0 * 0 0
0 0 1 1 1 0 0 0 0 * 0 0 0
0 0 0 1 0 0 0 * 1 0 0 0
0 0 0 1 0 0 * 0 0 0 0 0
0 0 0 1 0 * 1 0 0 0 0
0 1 0 1 0 * 1 0 0 0 0
0 0 0 1 * 0 1 0 0 0 0
* * * * 0 0 0 0 0 0 0
```

```
(11,0) (10,1) (9,2) (7,4) (6,5) (5,6) (4,7) (3,8) (2,9) (1,10) (0,11)
0 0 0 0 0 0 0 0 0 0 0 *
0 0 0 0 0 0 0 0 0 0 0 * 0
0 0 1 0 0 0 0 0 0 0 * 0 0
0 0 1 0 0 0 0 0 * 0 0 0
0 0 1 0 0 0 0 * 0 0 0 0
0 0 1 1 1 0 * 0 0 0 0
0 0 0 1 0 * 0 0 1 0 0 0
0 0 0 1 * 0 0 0 0 0 0
0 0 0 1 0 0 1 0 0 0 0
0 1 * 1 0 0 1 0 0 0 0
0 * 0 1 0 0 1 0 0 0 0
* 0 0 0 0 0 0 0 0 0 0
```

# 结果分析

## 改进A\*算法评测



## 结论

左侧绿色柱形图为原算法，红色为改进A\*算法，由图表可知，改进A\*算法能够在一定程度上减少车辆节点应用。或者说，能够利用有限的车辆节点，完成更复杂的任务



# 05

## 总结与展望

Summary and prospect



# 总结

- 主要围绕该系统的整体方案设计、软件联合仿真、真实场景应用三个方面进行研究。本文首先对车载自组网及相关技术进行分析探讨，之后对整个车载自组网的通信、控制模块进行了设计，同时搭建了仿真平台，分析了Ardupilot代码Rover部分的架构及原理，进一步实现了Ardupilot、MAVProxy、Mission Planner等软件的联合仿真。同时对官方说明未详细阐述、未考虑到的仿真部分进行了补充，详细说明了Ardupilot结合其他软件进行仿真的方法。该部分内容对于因疫情或其他实际情况，导致无法进行实车实验的研究人员提供了参考。
- 同时，本文从宏观角度出发，研究通信和车辆控制模块整合的车载自组网在车辆协同作业的应用，针对该系统的自身特点，提出了基于改进A\*(A-Star)算法的路径规划、节点部署策略。该方法考虑到了Zigbee技术、自组网技术自身的特点，同本文设计的小车相配套。同时提出了可跳跃点、双步长策略等新概念。

## 不足和展望

- 本文虽然分析了通过自组网传递的车辆控制信息，但是受限于疫情等实际条件的限制，未能进行实车测试，因此对于Zigbee通信的封装数据包内容未能详细设计。
- 本文设计的基于改进A\*算法的路径规划和节点部署策略，在调度函数上进行了一定的简化，该部分内容在一定程度上会影响算法在部分实例中的表现，同时部分极端情况下，无法给出最优方案，该部分内容也是未来研究的重点。