本章目录

关键通信协议与开发接口

第九章 多机通信与智能组网

9.1 组网通信基础与背景需求

9.2 硬件选型指标与通信机制

9.3 软件分层与通信协议

9.4 通信建模与干扰仿真

9.5 RflySim组网框架与通信仿真

9.6 关键通信协议与开发接口

9.7 本章实验案例与综合验证

9.8 习题与课堂讨论

本节 4 个教学单元

9.6.1 MAVLink协议

教材对应：第9.2.2.1节

9.6.2 DDS中间件

教材对应：第9.2.2.2节

9.6.3 OLSR路由协议

教材对应：第9.2.2.3节

9.6.4 常用接口及二次开发

教材对应：第9.2.3节

9.6.1 MAVLink协议：关键通信协议分工

消息结构、程序匹配和路由维护各有职责

要回答的问题	本节机制	直接产物
接收方如何解释字节?	MAVLink消息编码	可解析的消息帧
哪些程序接收哪类数据?	DDS发布/订阅及QoS	匹配的Writer与Reader
IP包下一跳去哪里?	OLSR主动维护路由	本地目的地址→下一跳表

机制组合由实际架构决定，不能把三者画成强制串行链。

9.6.1 MAVLink协议：设计目标与版本

MAVLink用紧凑帧携带有类型的消息

MAVLink v1 Frame (8 – 263 bytes)

STX	LEN	SEQ	SYS ID	COMP ID	MSG ID	PAYLOAD (0 - 255 bytes)	CHECKSUM (2 bytes)

MAVLink v2 Frame (12 – 280 bytes)

STX	LEN	INC FLAGS	CMP FLAGS	SEQ	SYS ID	COMP ID	MSG ID (3 bytes)	PAYLOAD (0 - 255 bytes)	CHECKSUM (2 bytes)	SIGNATURE (13 bytes)

图9.9 局部：v1与v2帧结构

消息机制： MsgID对应字段结构；生成库完成编码、解析和CRC计算。

版本演进： v2扩展消息ID与字段，支持签名和末尾零字节截断；与旧节点通信要核对能力。

9.6.1 MAVLink协议：MAVLink传输流程

图9.9的核心是编码、传输和解析

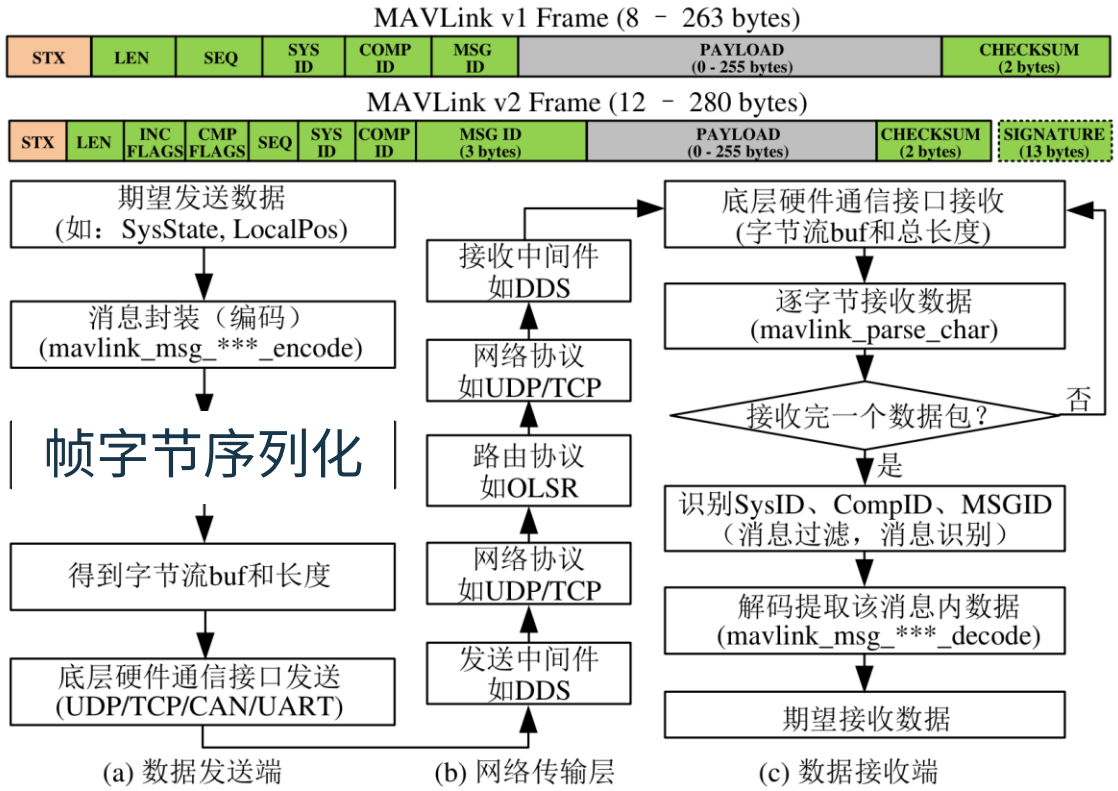


图9.9 MAVLink数据帧与传输流程

左列：结构体→编码→帧字节→实际链路发送。

右列：字节→完整有效帧→按消息ID解码。

中央组合：DDS为可选中间件；OLSR仅用于相应的多跳IP网络。

9.6.1 MAVLink协议：字段表：偏移0–4

固定帧头首先说明版本、长度和序号

字节偏移	字段 / 字节数	含义与处理
0	Magic / 1	0xFD表示MAVLink v2
1	Length / 1	实际载荷字节数，记为n；范围0–255
2	IncompatFlags / 1	未知的不兼容标志：拒绝该帧
3	CompatFlags / 1	未知的兼容标志：可忽略
4	Seq / 1	0–255循环；辅助分析丢包、乱序等

n表示载荷长度；偏移表示字节位置。

9.6.1 MAVLink协议：字段表：偏移5–9

源系统、源组件与消息类型不能互换

字节偏移	字段 / 字节数	区分的对象
5	SysID / 1	发送系统，例如载具1
6	CompID / 1	发送组件，例如自驾仪或吊舱
7–9	MsgID / 3	消息类型；按低、中、高字节发送

例：LOCAL_POSITION_NED的MsgID=32；不是“32号载具”。
帧头源ID、载荷中的目标ID、网络IP与UDP端口各有独立含义。

9.6.1 MAVLink协议：字段表：变长部分

载荷、CRC和签名从长度n确定位置

字节偏移	字段 / 字节数	解释
$10 \ldots (9+n)$	Payload / n	由消息定义生成；n=0时无此区
$(n+10) \ldots (n+11)$	CRC / 2	含CRC_EXTRA参与的帧校验
$(n+12) \ldots (n+24)$	Signature / 可选13	签名标志开启时附加

CRC：检错，不等于重传或认证。签名：认证与完整性，不等于加密。
自定义消息需在两端使用兼容的XML定义及生成库。

9.6.1 MAVLink协议：编码长度算例

帧长由实际载荷和是否签名共同决定

$$L_{v2} = 10 + n + 2 + 13s, \quad s \in \{0,1\}, \quad n = 28$$

L_{v2}为帧长（字节），n为载荷字节数；$s = 0$不签名，$s = 1$带签名。

部分	本例偏移（含两端）	长度
固定帧头	0–9	10B
载荷	10–37	28B
CRC / 可选签名	38–39 / 40–52	2B / 13B

算得未签名40B、签名53B；不含UDP/IP或链路层额外开销。

9.6.1 MAVLink协议：编码与解析接口

先生成帧字节，解析成功后才读取字段

```
mavlink_msg_heartbeat_encode(
    sysid, compid, &msg, &hb);
frame_len =
    mavlink_msg_to_send_buffer(
        buf, &msg);
send_bytes(buf, frame_len);
```

```
if (mavlink_parse_char(
    chan, byte, &msg, &status)) {
  if (msg.msgid == 0)
    mavlink_msg_heartbeat_decode(
        &msg, &hb);
}
```

hb、msg、buf与status须按生成库类型初始化；frame_len是整帧字节数。
send_bytes为自定义发送函数；局部片段还需源过滤、异常处理和有效帧后的应用逻辑。

9.6.1 MAVLink协议：位置消息的接口映射

安装版把位置消息解码到状态对象

```
if msg.get_type() == "LOCAL_POSITION_NED":
    self.uavPosNED[0] = msg.x
    self.uavPosNED[1] = msg.y
    self.uavPosNED[2] = msg.z
    self.netForwardData()
```

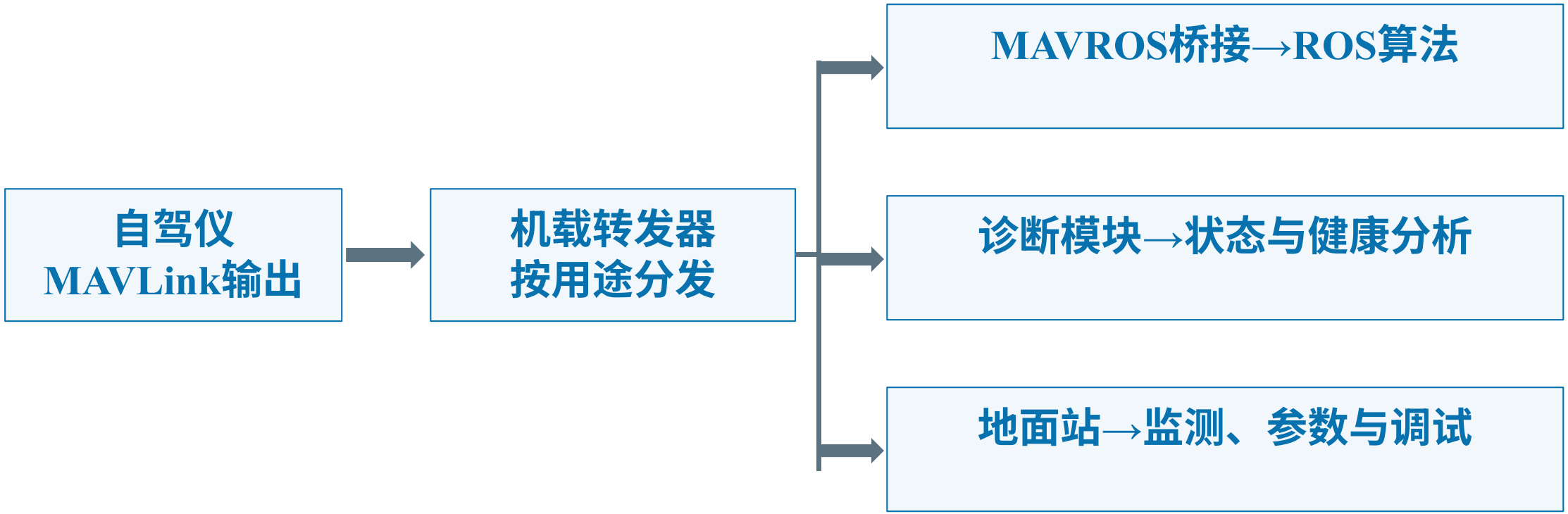
输入：已解析的MAVLink消息，类型ID为32。

输出：位置进入uavPosNED，再触发后续转发。

安装版定位：RflySimSDK/ctrl/PX4MavCtrlV4.py的getMavMsg接收分支。
实际完整分支还更新速度、全局参考位置及事件；此处截取位置和转发语句。
NED坐标（m）：msg.x向北、msg.y向东、msg.z向下；均相对局部原点。

9.6.1 MAVLink协议：多节点转发

MAVLink流可以按用途并行分发



各路消息集合、更新率与权限按任务配置；桥接、过滤和重复流量都影响时效。

本章目录

关键通信协议与开发接口

第九章 多机通信与智能组网

9.1 组网通信基础与背景需求

9.2 硬件选型指标与通信机制

9.3 软件分层与通信协议

9.4 通信建模与干扰仿真

9.5 RflySim组网框架与通信仿真

9.6 关键通信协议与开发接口

9.7 本章实验案例与综合验证

9.8 习题与课堂讨论

本节 4 个教学单元

9.6.1 MAVLink协议

教材对应：第9.2.2.1节

9.6.2 DDS中间件

教材对应：第9.2.2.2节

9.6.3 OLSR路由协议

教材对应：第9.2.2.3节

9.6.4 常用接口及二次开发

教材对应：第9.2.3节

9.6.2 DDS中间件：DCPS概念模型

DDS以主题和数据类型连接生产者与消费者

数据分发服务（DDS）；以数据为中心的发布/订阅（DCPS）。



域与参与者：Domain是逻辑通信范围；DomainParticipant管理本端实体与生命周期。

样本通路：Writer写入样本→已匹配Reader接收；Topic不是中央消息队列。

9.6.2 DDS中间件：发现与匹配

同一Domain只是能够匹配的必要条件之一

参与者发现
Domain与可达性
→
端点匹配
Topic、类型、QoS
→
样本交付
Writer→Reader

配置或现象	可以说明什么	不能据此推出什么
同Domain	处在同一逻辑域	主题、类型或权限一定正确
回调报告新增匹配	端点建立匹配关系	应用已收到新鲜有效样本
收到有效样本	数据到达并能解析	满足全部时延与控制要求

9.6.2 DDS中间件：服务质量策略

QoS分别约束可靠性、历史和时间语义

策略	控制的行为	状态流中的判断
Reliability	BEST_EFFORT / RELIABLE	丢包容忍与重传开销
History / ResourceLimits	KEEP_LAST深度或保留全部 并限制缓存资源	避免无限积压；保留需要的历史
Deadline / Lifespan	更新间隔监测 / 样本有效期	发现停更 / 排除过期样本
LatencyBudget	允许的通信延迟预算意图	不能替代端到端实测与调度

服务质量（QoS）需要两端兼容；应用还必须检查时间戳与任务截止期。

9.6.2 DDS中间件：新鲜度算例

可靠地保留旧数据也可能使控制滞后

$$f_{\text{pub}} = 50\ \text{Hz}, \quad f_{\text{use}} = 10\ \text{Hz}, \quad \Delta Q = 5 - 1 = 4$$

f_{pub}为发布频率，f_{use}为消费频率；每次各1个样本，全部到达且不丢弃。
按先进先出处理；每0.1s产生5个、处理1个，ΔQ为待处理数净增量（个）。

业务需求	可选设计	还需验证
最新载具状态	KEEP_LAST(1)；读取最新样本	数据年龄、丢弃策略、连续性
必须保存的事件	可靠传输与有界持久记录	处理能力、资源上限与补发机制

KEEP_LAST(1)限制历史深度，不能消除正在处理的旧任务；应用队列也要管理。

9.6.2 DDS中间件：应用消费与状态年龄

相同历史深度，应用队列仍会改变新鲜度



图：空心点为执行中样本的采样时刻，竖线为当前时刻

KEEP_LAST(1)只限每实例历史；应用待处理区与在执行任务需要分别管理。

9.6.2 DDS中间件：开源实现与传输

Fast DDS由DDS实体映射到RTPS端点



协议：实时发布/订阅（RTPS）规定DDS端点间发现与交换数据的通信规则。
对应关系：RTPSParticipant组织端点；RTPSWriter与RTPSReader在底层交换变化。

实现条件：相同标准不代表所有版本和配置都自动互通；发现方式、类型与QoS需兼容。

时延预算：查当前版本是否实现相关策略，再检查实际网络、缓存和应用处理时延。

9.6.2 DDS中间件：例程中的DCPS实体

HelloWorld源码先注册类型，再创建主题

```
data_type = HelloWorld.HelloWorldPubSubType()
data_type.setName("HelloWorldDataType")
support = fastdds.TypeSupport(data_type)
participant.register_type(support)
topic = participant.create_topic(
    "myTopic", data_type.getName(), qos)
```

类型：生成的PubSubType负责类型与序列化支持。

主题：名称与类型一起决定可匹配对象。

片段省略self前缀并重命名局部变量；fastdds、HelloWorld和participant已初始化。后续创建Writer/Reader，注册监听器，再写入或读取样本。

9.6.2 DDS中间件：最小入口与排错

先确认端点匹配，再判断样本内容和时效

[安装目录]/RflySimAPIs/9.RflySimComm/
0.ApiExps/e2.DDSDepsTestExps/Demo/
在上级目录双击两次Python38Run.bat；两窗口均cd Demo，先订阅后发布。

```
python -u HelloWorldExample.py -p subscriber -d 0
python -u HelloWorldExample.py -p publisher -d 0
```

参数与结束： -d 0指定域0；发布端发10条后结束，订阅端按回车退出。
排错： 依赖加载→同Domain/网络可达→Topic/类型/QoS→匹配→收包。

判据： 双方出现matched；接收端应依次显示Received Hello World : 0至9。
验收： 核对读取返回值与info.valid_data；记录到达时间，不能仅看进程未退出。

本章目录

关键通信协议与开发接口

第九章 多机通信与智能组网

9.1 组网通信基础与背景需求

9.2 硬件选型指标与通信机制

9.3 软件分层与通信协议

9.4 通信建模与干扰仿真

9.5 RflySim组网框架与通信仿真

9.6 关键通信协议与开发接口

9.7 本章实验案例与综合验证

9.8 习题与课堂讨论

本节 4 个教学单元

9.6.1 MAVLink协议

教材对应：第9.2.2.1节

9.6.2 DDS中间件

教材对应：第9.2.2.2节

9.6.3 OLSR路由协议

教材对应：第9.2.2.3节

9.6.4 常用接口及二次开发

教材对应：第9.2.3节

9.6.3 OLSR路由协议：邻居发现与总体流程

OLSR主动维护路由，HELLO只探测局部邻域

HELLO
一跳与二跳 → MPR选择
减少冗余泛洪 → 拓扑控制（TC）
通告拓扑关系 → 本地路由表
目的→下一跳

HELLO：只在一跳内交换；区分单向与对称链路，严格二跳集排除自身及一跳邻居。

MPR：Multipoint Relay，即多点中继；不是“多路径路由”的缩写。

维护代价：短更新周期有利于追踪变化，也增加控制流量；信息仍有失效与收敛过程。

9.6.3 OLSR路由协议：MPR覆盖目标

式9.5用少量一跳中继覆盖严格二跳邻居

$$\min|\mathrm{MPR}(i)| \quad \text{s.t.} \bigcup_{j\in\mathrm{MPR}(i)} R(j) \supseteq N_2(i) \tag{式(9.5)}$$

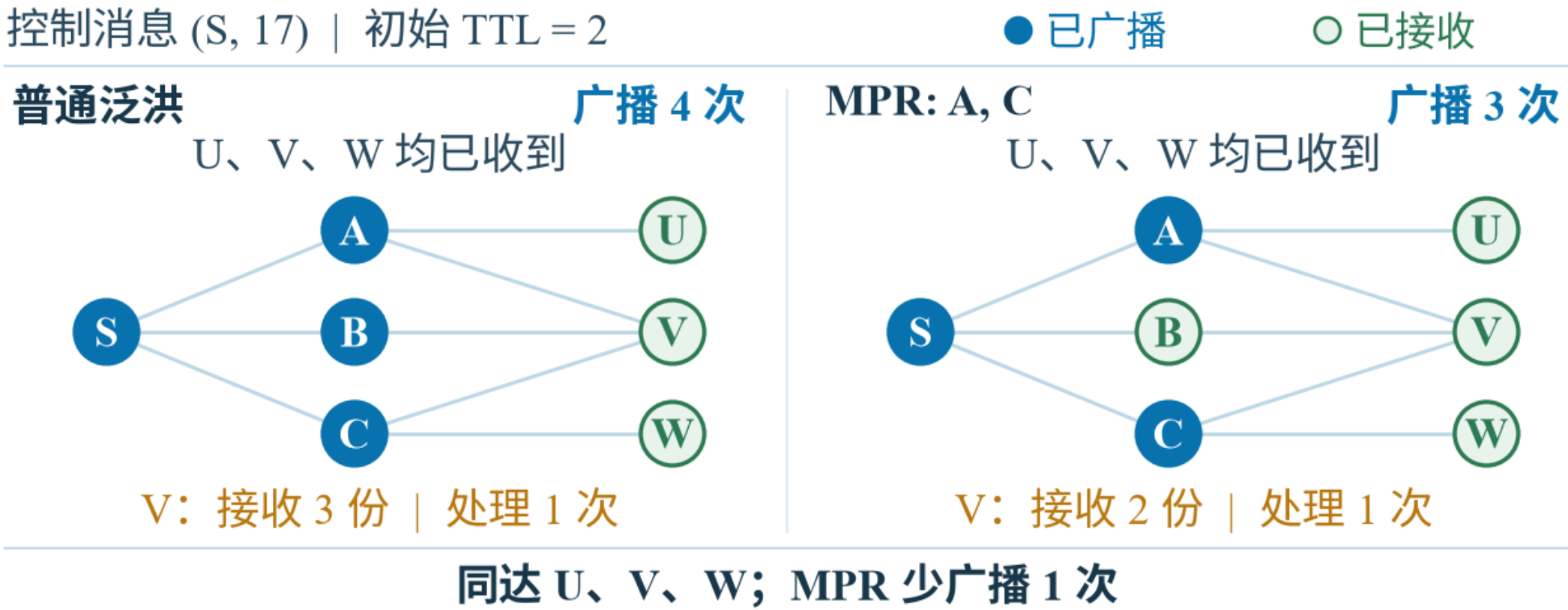
集合含义： i为本节点；$\mathrm{MPR}(i)$为选中的对称一跳中继集，$|\cdot|$表示节点数。
$R(j)$为邻居j能覆盖的严格二跳集；$N_2(i)$为本节点的全部严格二跳邻居。

覆盖要求： 所选覆盖集取并集，须覆盖全部严格二跳邻居；目标是减少重复广播。

启发式： 先处理必须选择的候选，再扩大未覆盖集合的覆盖；不保证全局最小。

9.6.3 OLSR路由协议：MPR覆盖与泛洪

覆盖相同二跳邻域，MPR减少重复广播



图：本例普通泛洪广播4次，MPR广播3次；两侧均识别重复消息

9.6.3 OLSR路由协议：集合覆盖计算

覆盖表给出完整的MPR选择过程

候选或步骤	覆盖集合	尚未覆盖 / 判断
A / B / C	{U,V} / {V} / {V,W}	初始：{U,V,W}
选择A	{U,V}	剩余：{W}；U只有A能覆盖
选择C	{V,W}	剩余为空；W只有C能覆盖
本例最少数量	单个候选最多覆盖2个	至少2个；{A,C}达到此下界

本例假设候选转发意愿相同且均可选；MPR选择是每个节点自己的局部计算。

9.6.3 OLSR路由协议：TC与下一跳

TC减少拓扑通告，数据包仍按路由表转发

阶段	交换或计算的内容	得到什么
A选择B为MPR	A在HELLO中告知邻居关系	B知道A是自己的选择者
B生成TC	至少通告选择它的邻居集合	远端获得必要拓扑关系
每个节点计算路由	本地邻居表与TC拓扑信息	目的地址、下一跳、距离
业务包转发	按本地IP路由表发送	经过可达下一跳，逐跳到目的

MPR集合、MPR选择者集合和业务路由表是三个不同对象。

9.6.3 OLSR路由协议：ETX链路代价

式9.6将双向交付概率换成期望传输次数

$$\mathrm{ETX}(u,v)=\frac{1}{d_f d_r} \qquad \text{式(9.6)}$$

d_f：前向交付概率；d_r：反向确认交付概率；均在(0,1]内。
ETX无量纲；给定独立、稳定的重复尝试模型。

$$\mathrm{ETX}_{\mathrm{direct}}=\frac{1}{0.4\times 0.8}=3.125$$

u、v为链路两端节点；下标direct表示本例直连链路。
零交付概率表示链路不可用；低ETX不自动等于最低时延或最高吞吐。

9.6.3 OLSR路由协议：最短路径模型

式9.7求的是指定边权下的最小总代价

$$d(s,v)=\min_{p\in\mathcal{P}(s,v)}\sum_{(x,y)\in p}w(x,y) \qquad \text{式(9.7)}$$

s、v为起点与终点；p为一条候选路径；(x,y)为该路径上的一条边。

$\mathcal{P}(s,v)$是候选路径集合；$w(x,y)$为非负边权；$d(s,v)$是最小总代价。

A到C候选路径	跳数	ETX和
A→C；成功率0.4 / 0.8	1	3.125
A→B→C；每段0.9 / 0.9	2	$2/(0.9\times0.9)\approx2.469$

算例输入为教学设定；按ETX选A→B→C，按跳数选A→C。

9.6.3 OLSR路由协议：完整三节点计算

Dijkstra逐步更新代价与下一跳

步骤	已确定节点	B的代价	C的代价 / 下一跳
初始化	A（代价0）	1.235	3.125 / C
确定B并松弛	A、B	1.235	2.469 / B
确定C	A、B、C	1.235	2.469 / B

每轮确定代价最小的未确定节点，再用经它的更小代价更新邻居（松弛）。
最终A的路由：目的B→下一跳B；目的C→下一跳B。
基础OLSR按跳数构路；ETX扩展须另有链路测量、通告和选路实现。

9.6.3 OLSR路由协议：实现与优化方向

基础OLSR与质量、预测等扩展要分别验证

```
case 1:
    list.Add(olsr, 100);
    break;
```

源码映射： RflyNet.cc中rt=1选择OlsrHelper；含自定义模块依赖，不能据此承诺直接编译。

基础机制： HELLO、MPR、TC与按跳数路由；不自动带有ETX。

扩展方向	引入的信息或结构	主要代价
质量 / 预测	交付率 / 位置与运动模型	测量误差、通告量及预测偏差
分簇 / 混合	簇间组织 / 主动与按需切换	维护和切换复杂度

本章目录

关键通信协议与开发接口

第九章 多机通信与智能组网

9.1 组网通信基础与背景需求

9.2 硬件选型指标与通信机制

9.3 软件分层与通信协议

9.4 通信建模与干扰仿真

9.5 RflySim组网框架与通信仿真

9.6 关键通信协议与开发接口

9.7 本章实验案例与综合验证

9.8 习题与课堂讨论

本节 4 个教学单元

9.6.1 MAVLink协议

教材对应：第9.2.2.1节

9.6.2 DDS中间件

教材对应：第9.2.2.2节

9.6.3 OLSR路由协议

教材对应：第9.2.2.3节

9.6.4 常用接口及二次开发

教材对应：第9.2.3节

9.6.4 常用接口及二次开发：单点通信接口

目的载具ID和UDP端口是两种参数

接口调用	参数含义	本例实际结果
enUavForward(2)	目标载具ID	224.0.0.10:60002
enNetForward([60002], ip)	目标端口列表与IP	ip:60002
StartNetRecOwn()	自身CopterID=2	监听60002
StartNetRec(60002)	接收端口	直接绑定60002

NetSimAPI(mav)可按状态更新事件转发；NetSimAPI(1)仅赋予源ID，不自动采集状态。

9.6.4 常用接口及二次开发：最小点对点调用

先建立接收，再把一号状态发给二号

发送端：一号载具

```
mav = PX4MavCtrl.PX4MavCtrler(1)
net = NetSimAPIV4.NetSimAPI(mav)
mav.InitMavLoop()
net.enUavForward(2)
```

接收端：二号载具

```
net = NetSimAPIV4.NetSimAPI(2)
net.StartNetRecOwn()
for _ in range(100):
    if "1" in net.UavDict: break
    time.sleep(0.1)
uav1 = net.UavDict.get("1")
```

模块：time、PX4MavCtrlV4 as PX4MavCtrl、NetSimAPIV4；两段分属两进程。
启动SITLRun2MavlinkFull.bat，初始化后依次运行UAV2Ctrl.py、UAV1Ctrl.py。
判据：uav1非空为收包成功；约10 s仍为空为超时。持续控制另查时间戳。

安装版：9.RflySimComm/0.ApiExps/e1.RflyNetAPIExps/1.P2PSendData

9.6.4 常用接口及二次开发：状态包解析

转发包的源ID、时间与坐标各有职责

部分	字段或格式	解释
32B包头	iiiiQd；源ID、模式、起始ID 64位掩码、发包墙钟	供识别、筛选和时间统计使用
104B状态	d6f9d；状态时间、姿态 速度、GPS原点、两种位置	角度为弧度；速度为m/s NED位置为m；GPS经纬为度
接收容器	UavDict['1']；UavData[0]	前者按源ID；后者按首次到达顺序

固定12345678只作格式识别；自定义跨平台格式还应明确字节序、版本和长度。

9.6.4 常用接口及二次开发：广播与载具筛选

公共端口分发与应用筛选作用在不同位置

发送一次
224.0.0.10:60000
→
加入同一组播组
多个接收进程
→
按源ID使用
各自所需状态

公共分发： enUavForward(0)与StartNetRec()使用默认公共端口60000。

物理目的： PortList决定sendto发往哪里；发送多个端口会实际发多份包。

逻辑筛选： 源ID过滤发生在应用端；目的掩码需由对应转发器解析才约束分发。

9.6.4 常用接口及二次开发：按需状态请求

请求列表说明向谁要，发送列表说明给谁发

2号请求者
请求1号状态 → 1号匹配自身ID
将2号加入发送表 → 1号状态流
随后发给2号

对象	列表或操作	需要理解的边界
请求侧	StartReqUavData([1])	当前循环约1Hz发请求
响应侧	netAddUavSendList([2])	保存目标ID供后续转发
取消与退出	本地清表或停止请求	不等于远端自动删除目标

9.6.4 常用接口及二次开发：自定义数据转发

自定义数据先约定字节，再检查解包条件

```
payload = struct.pack("<Bff", 1, 2.5, -1.0)
net.netForwardBuf(payload)
# 接收端：新包事件触发后
buf = bytes(net.bufData)
net.bufEvent.clear()
if len(buf) == 9:
    ver, a, c = struct.unpack("<Bff", buf)
    if ver == 1:
        use_values(a, c)
```

格式： import struct；小端、1B版本、两个4B浮点数。

前置： 发送端启用enNetForward；接收端启用StartNetRecOwn。

应用： use_values处理数据；控制量须验证范围。

解包结果：ver=1、a=2.5、c=-1.0；外层包头仍按iiiiQd格式处理。

9.6.4 常用接口及二次开发：时间与生命周期

收到过、收到新数据与正常退出是不同状态

检查对象	常见误判	正确判据
新鲜度	字典存在就继续使用	采样时间更新；超时后降级或退出
跨机延迟	收发墙钟之差就是网络时延	统一时基并估计同步误差
资源退出	设置停止标志就立即结束	唤醒接收；结束线程并关闭套接字

API入口： [安装目录]/RflySimAPIs/9.RflySimComm/API.pdf

练习与讨论

数据到达失败时，先定位是哪一个机制

1. **MAVLink**：CRC通过后，能否直接宣布控制指令已被执行？还缺什么证据？

2. **DDS**：Domain相同但一直无样本，接下来分别查哪些配置与状态？

3. **OLSR**：C未成为A的MPR，是否仍可发送和转发业务包？

4. **ETX**：两跳2.469小于直连3.125，能否据此证明时延更低？

参考要点

帧有效、端点匹配、路径可达与执行成功分层判断

MAVLink：检查源与目标、消息语义、所用命令的确认机制和实际响应；CRC不等于执行成功。

DDS：网络可达→Domain→Topic/类型/QoS→匹配→有效样本→采样时刻与使用时刻。

OLSR：C仍是网络节点；MPR只约束相应控制泛洪，业务单播查各节点的路由表。

ETX：本例选两跳；要比较时延，还需每跳速率、重传时间、接入竞争与队列统计。

本节小结

用机制解释结果，再用实验检验假设

消息编码：读清源ID、消息ID、载荷长度和校验；序列化不等于可靠交付。

数据分发：域、主题、类型和QoS共同决定匹配；完整到达不等于足够新鲜。

多跳路由：HELLO→MPR→TC→路由表；基础跳数目标与ETX扩展分开。

实验承接：固定通信任务，记录发送、到达、数据年龄和运动响应，解释性能差异。