①关于奖励，每个节点可以以离目标点的距离作为奖励函数，这样的绝对奖励是没问题的。

②关于节点数目，初步设定为5个·节点，**信道建模＋代数连通度作为信道奖励**

③设计对应的奖励函数为：信道连通奖励+离目的地的奖励

④查看MADDPG代码，以及最新的TD3 、MAAC 框架，思考实现。

⑤信道选择 + 轨迹规划 为 （ 离散 ＋ 连续 ）怎么考虑实现。

3-18

①改为连续空间，输出为加速度，速度范围限制（具体是多少限制需要查资料）（环境代码修改·完成，更改模型输出以及模型的代码，已经完成）。去除信道离散选择，后续需要可以加上

②新增代数连通度计算，无向图计算函数作为奖励（未完成）

③归一化奖励和惩罚（未完成），非常难啊

④暂时先去除信道选择和功率控制，只研究对应的连续动作输出（已完成）

3-19

①加上batch\_normalize,状态还需要加上当前速度！

②编写代数连通度奖励

③测试现有设置条件下，两种信道模型下，无人机功率在0-1w，无干扰的情况下，其满足sinr大于10所提供的距离为多少。以便于设置场景放缩尺寸因子（重要）

经验：

1.**由于是绝对值作为奖励，到达目的地后如果不持续给奖励，那么其实是在持续的给惩罚，因为没到目的地会持续获取更多奖励，所以需要1.持续给奖励（会导致critic训练结果不准确）或者 2.使用差值作为奖励，这样很完美（1.这是导致我之前一直训练不会向目的地移动的原因）LOSS无法下降，为什么**

**2. 现在奖励由于两部组成 连通度奖励 + 离目的地的差值奖励**

**但是加了连通度奖励之后，计算特别特别慢，直接变慢十分之一，可能是求特征值的时候很慢**

**3.训练效果很差，是否加上离建筑物距离增加的惩罚，而不是直接稀疏惩罚**

**探索的时候如何平衡前期和后期算法。探索非常失败,前期随机性不足**

**①必须增加优先回放经验，不然没办法取得进步：又说效果不好，暂时先不加**

**②增加靠近建筑物的非线性惩罚**

**③在碰撞时不更新位置，但是更新碰撞状态，给与对应惩罚（这个效果很好，只是节点可能会显得很呆）**

**查看了一下需要一轮探索结束后再训练，以及探索噪声设置**

**使用TD3**

**代数连通度的 上下界限为 0 – n, 当图不连通时，其代数连通度为0**

**今晚任务：**

**实现ppo算法和mappo算法，并跑一轮验证，因为这两个据说最好**

**MADDPG的部分还没更新无人机碰撞部分**

**COMA和MADDPG和MAAC：**

**DDPG：不考虑参数共享，随智能体增加复杂度提高，扩展性差。  
COMA假设智能体同构，actor共享参数，通过为每个智能体设置优势函数来判断并优化每个智能体的贡献。（都是用在离散空间上的）  
MAAC在SAC的基础上增加了注意力机制等：注意力机制用于忽略不重要的信息：用不上，输入全是最简单的重要信息。（都是用在离线空间上的）**

**COMADDPG：**

**通过添加每个智能体的critic网络，能够直接利用每个智能体的动作预测每个智能体的奖励。  
这里可能考虑到共享参数的问题，因为每个智能体同构，所以可以共享参数  
全局critic网络用于判断当前决策的全局奖励，从而提升单个critic网络的参数**

**分层强化学习：**

**先上层控制层生成对应的中间目标移动点然后，实际下层决策层生成具体策略，降低训练难度。**

**但是感觉还是很难**

**基于多LOSS平衡的comaddpg架构**

**由于任务场景主要有两个任务：到达目标点 以及 保持网络连通，适合多任务学习的奖励分解，因此通过改造maddpg，通过设置每个无人机的单critic负责任务1，而全局的critic通过全局信息负责任务2的优化，再通过gradnorm实现二者的联合优化，从而同时优化两项任务。**

**通信保持能力设置为1理论上比较合适（通信保持奖励应该持续计算，并且需要一个函数计算整个游戏过程中的到达奖励和保持奖励的函数总数）**

**直接使用MADDOG非常有问题，不知道为什么无人机就是会乱飞，然后卡死在某个位置**

**Reward50质量非常差**

**TODO:**

**①编写一个函数用于更新保存reward最大的episode，后期test的次数应该要增加（只要能保存到一个最优的模型就可以了）**

**②完善comaddpg中critic网络，查看梯度变化，（观察随机噪声幅度的影响）**

**③重新设置reward：  
拓扑保持应该按照增量式的，大小应该比全速前进的奖励大：设置为1比较合适**

**前进奖励应该归一化为1，也就是大约是 / 5，其绕路，设置为- 0.5比较好**

**④同样参数下，设置200 – 30000 观察结果，如果上不去，再加一层网络深度，这个很重要**

**对于36节点研究：**

**①单体奖励为靠近奖励归一化  
②全局奖励为碰撞奖励  
③一样可以使用cotd3**

**④是否可以使用共享参数**

**还要统计一个保持连通度奖励情况的函数，连通保持则统计其加一（）**

**路径规划和轨迹规划**

**使用CBS+RRT\*算法实现路径规划，传统的多智能体路径规划只是单智能体路径规划 + 冲突避免算法，其过程中不会考虑到无人机路径之间相对的通信联系方式。因此无法保障无人机之间通信链路的保持效果，无法满足抗干扰的需求  
本研究提出的方法通过DDPG将链路保持度作为奖励，能够保证路径规划在路径短的同时，保障移动过程中的通信抗干扰能力。**

**这种算法还无法解决：动态障碍  
RRT不能收敛到最优，1998**

**RRT\*能 2011**

**CBS：2015**

**CCBS: 2019   
还有一些改进延续到2024年  
使用这种方法**

**1.原问题加层数：不改变 reward还是1**

**2.判断是不是概率问题，高斯噪声10，增加探索，使得能够随机探索更多区域**

**3.判断是不是步长问题导致复杂度太高，速度变为2，加速度变为2**

**4.使用累计reward不连通则减1**

**这些都跑完后  
需要进行计算，①根据对应的判断**

**接下来：**

**①整理环境场景设置：信道建模，代数连通度计算**

**②整理网络模型设计：模型架构，动作空间状态空间，奖励函数**

**③整理仿真结果：5节点结果，36节点结果，**

**④传统单路径规划算法综述：路径规划＋轨迹优化，基于搜索的，基于采样的，基于智能的  
⑤多智能体路径规划算法：CBS + RRT\* 还有很多多智能体路径规划，除了避障主要解决的是全局最优的问题：路径总和最小，时间最短，最晚到达的时间。**

**1）A\* 必须沿着最优路径扩展所有的节点，因此A\* 算法在解决大规模智能体的MAPF 问题时效率和质量都不高。**

**2）基于冲突搜索算法是目前最常用的算法，目前求解MAPF 速度和质量最好的算法大都是在基于冲突搜索算法上进行改进和优化。CBS(120-200个智能体)、CCBS（非连续时间）、EECBS（搜索算法）**

**3）代价增长树搜索（increasing cost tree search，ICTS）算法将MAPF 问题分解为两个问题：找到每个智能体的代价和找到这些代价的有效解决问题方案：2012-2017，不主动处理冲突，适合智能体更少，复杂度更低的情况**

**人工智能的MAPF主要分为以下三类：**

**专家演示型（IL+RL）：由专家生成大量优质路径供学习：PRIMAL（2018）**

**任务分解型：  
 改进通信型：**

**为什么传统的算法是这样的：**

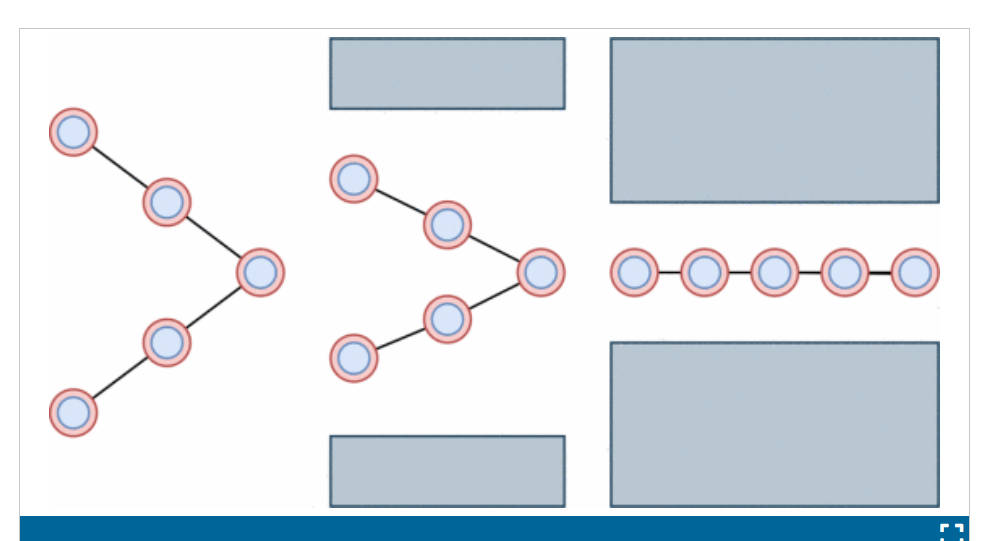
**现有的你说的这些问题的已有算法，和**[**marl**](https://zhida.zhihu.com/search?content_id=508487090&content_type=Answer&match_order=1&q=marl&zhida_source=entity)**相比计算代价小，效果也不错，一定是收敛的，而且基本都能找到最优解。marl当智能体个数超过10计算代价很大，收敛也难。但传统算法智能体个数几十个是没问题的。**

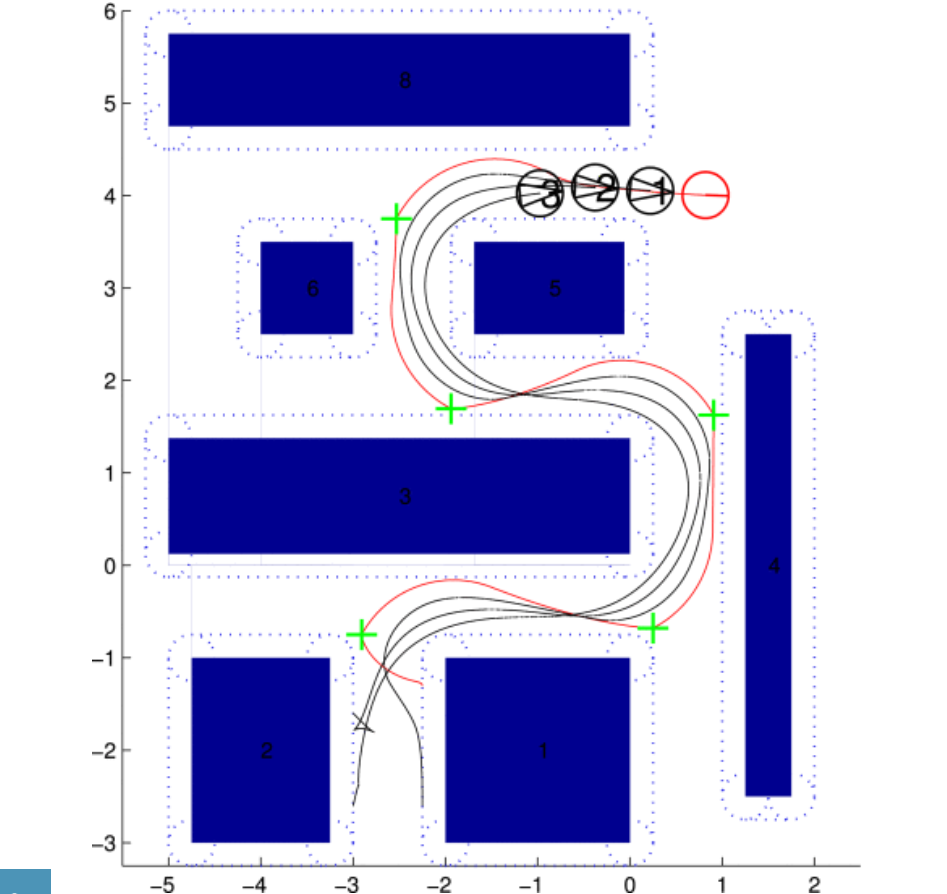
**回答老师的问题：**

**①为什么不使用编队变换加整体路径规划的形式：**

**编队行进主要有两种形式，一种是编队变换（V形/线形变换），一种是lead-follower方式  
1）队形自适应变换在复杂密集的连续建筑物场景下难以实现，目前只看到简单的避障变换，论文1。**

**2）lead-follower算法不考虑时变场景，难以适应动态干扰场景，论文【2】。**

****

**D. N. Bui, M. D. Phung and H. P. Duy, "Self-Reconfigurable V-Shape Formation of Multiple UAVs in Narrow Space Environments," *2024 IEEE/SICE International Symposium on System Integration (SII)*, Ha Long, Vietnam, 2024**

D. Panagou and V. Kumar, "Cooperative Visibility Maintenance for Leader–Follower Formations in Obstacle Environments," in *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 831-844, Aug. 2014

**②为什么不能实时探索：**

**实时探索算法（如动态窗口法DWA）只适合局部避障规划，只能解决短期的路径选择问题（容易陷入局部最优和死路），一般结合全局规划器（一样要通过全局搜索）使用，无法单独规划出全局路径**

**设计实验：**

**5节点场景下：**

**①干扰信号匀速移动：设计干扰信号匀速移动模型 （实现）**

**②干扰信号强度增大： （相应的连接惩罚应该增加，这样节点才会绕路）**

**③无人机数量，最多设计5，7， 10三组作为对比**

**设计实验：**

**①移动场景下可能会发生不移动也导致断联的情况出现：每次更新位置前检测是否连通，不连通则直接结束游戏并给与惩罚，**

**奖励设置：2,3,5。**

**直接结束的惩罚，设置为-10。（待设计实验）**

**优化场景：**

**①针对静止场景，如果回溯了全部节点的位置，则渲染的时候不渲染该帧，统计帧数的时候也不统计 （已完成）**

**②针对最后保存的效果最佳的数据，如果res相差不超过1，则保存step最小的那一个作为结果。（已完成）**

**③运动场景下，每次更新节点位置前检查一下当前是否连通，不连通的话则直接终止训练，给与一个大的惩罚，同时对应的链接惩罚应该增大，这样的话节点可能就会进行一个绕路到达中终点的操作。（已完成）。**

**Test-8：训练的clr为1e-3的情况下，网络不容易陷入局部最优，不会直接收敛到60左右**

**结论：1e-3的时候网络探索性更强，明显更容易达到最后的最优效果（fuck其实我根本就没改成）**

**Todolist  
1.整理文献解决两个问题**

**2.完善回溯函数，从而在回溯时能够同时回溯干扰位置，从而使得永远保持连接，绘图的时候去除回溯帧，很完美 （已完成）**

**3.36节点时调整clr改为1e03,同时解决对应的position函数 （√）**

**4.实现简单的跟随算法，在动态干扰移动场景下无法保证对应的能力。（已实现）**

**主要利用到编队控制加单智能体轨迹规划算法，由于编队控制在密集障碍物下难以实现集群自适应转换避障，所以主要采用分阶段时间延迟轨迹复制的方式解决该场景下的问题。**

**简单的leader-follower在开阔环境很好用，但在城市里要叠加很多队形变化、智能决策和规划技术，才能安全、有效地编队前进。实验设计：**

**1.使用离散程度度量（中心度量点，距离均方根，最大距离）指标，限制离散决策的范围。**

**2.600个step还是太多了**

**设计300step + guided**

**目前有两种手段增加探索速度**

**①设置离散度指标衡量，防止节点之间距离太远导致无法处理干扰突然出现的情况**

**②离散策略生成，引导至目的方向**

**传统路径规划：**

**OCRA：计算周围节点的速度障碍区域，限制自身的速度规划**

**TEB: 用于在全局规划后通过人工势场微调中间路径**

**DWA：评估单个无人机的每个决策（离目标点+离障碍物），选择最优的。**

**抗干扰：  
①把干扰看做障碍物：这种做法当然可以，障碍物的大小应该怎么确定呢：**

**通信连接不仅和干扰的距离有关，和无人机之间的距离也有关，不能简单地将干扰当做障碍物，将其大小限制在固定的范围，因为不同的无人机之间的距离也会影响这个障碍物的大小。**

**②干扰死角绕路问题：实时评估其通过难度，然后选择范围内的绕路点进行绕行规划。这是一个决策问题，目前应该没有算法能够这么做。**

**优化点**

**①连续回溯太多次则直接进行一个终止，连续回溯太多次不施加惩罚了还是说自动生成**

**②优先经验回放td-error**

**定义 1 (概率完备性): 给定路径规划问题，概率完备性意味着如果存在一条可行路径，算法最终能够找到它。**

**定义 2 (渐进近似最优性): 给定一个路径规划问题，轨迹的代价函数为 c (σ)。渐进近似最优性意味着，如果样本数量趋于无穷大，则 c (σ) 收敛到最优值。**

**对于DWA + SLAM + OCRA+D\* LITE + CBS**

**当前的算法应该属于基于采样的DWA + 虚拟导航点 + 判断法**

**层次化路径规划：**

**怎么做**

**①DWA先筛选对应的可行解的集合，然后根据DWA评估对应的优势，选择当前最优的执行一步（DWA）**

**②虚拟引力点：进入干扰威胁范围内，评估是否存在无法前进的干扰死区，不存在则启用虚拟引力点法穿越该高干扰区域**

**③绕路点决策：存在干扰死区则启动绕行节点引力决策进行绕路节点设置为中间规划节点。**

**有思路了：**

**①D\*LITE 规划大粒度的路径，此时只输入建筑物位置作为已知信息**

**②传感器实时感知一定范围内局部干扰，设置·较小的障碍范围作为禁区，然后作为动态信息补充，D\* lite进行重新规划**

**③大粒度路径作为中心目标节点引力方向，使用DWA （方向，离散度指标，通信链路保持）作为底层的规划指引**

**D\*liTE保障能够顶层路径能够脱离干扰禁区**

**DWA保证底层链路保持**

**从而实现通信抗干扰轨迹规划。**

**我现在的rt-rrt算法是通过采样的方式实现多个无人机的智能规划，现在我要编写一段代码，使用RRT作为动态规划的上层规划，DWA作为底层的细粒度规划方式。RRT \* 主要作用是用于根据当前地图信息规划中间无人机的目标点，步长可选，其需要维护自己的一个地图信息。初始的地图信息只有地图的基本建筑物信息，然后每架无人机会感知自身位置周围固定范围的干扰位置，一旦地图信息刷新，RRT\*需要判断自己之前规划的路径是否需要更新。 需要注意的是，所有的地图信息来自于env中，env通过step更新干扰的出现和移动，所以RRT\*需要在每次step后都需要刷新。**

**明天要实现，①调试RRT\* ②使用DWA实现局部规划 ③建立数据库 ④看结果**

**DWA放在外面这是可以的，因为我的就是根据这个采样，只是人工势场法的引力要一直改成对应的节点，每次step之后都需向其指向的方向移动，（重规划路径触发条件包括到达下一目标点的附近）。**

**已完成：**

**接下来的优化：**

**①DWA优化法，选择的下一位置应该是从可选中评估后的，设计评估函数**

**②研究中间过程中人工势场虚拟引力点的增加效果**

**③绘制RRT导航路线以及可视化的迷雾效果**

**测试性能：**

**①绕行能力，将干扰产生的范围和对应的干扰功率结合，设计大干扰死路时的绕行能力**

**②抗干扰能力，**

**③步数实现：**

**我有思路了，使用强化学习替换对应的局部DWA规划，输入为无人机的位置和无人机的目标点，reward就是通信保持以及离目标点的离散程度，这样就结合和RRT和RL,并且由于只需要分段采集经验，每一段更容易学习，所以训练的速度会提升，而不是从头到尾都进行学习导致效果很差无法前进。**

**需要补充的可视化**

**①干扰范围随功率变化。**

**②局部导航点绘图，刷新路径**

**③战争迷雾，未知和已知的范围**

**添加功能**

**①RRT优化取最优（未做，较没意义）**

**②DWA取这么多测试里的最优，已实现**

**周报，展示两种情况下的规划方法。**

**①穿越**

**②绕行**

**③建筑密度增大**

**下周计划：五节点局部规划的DWA用强化学习替换**

**36节点也一样使用替换。**

**①五节点：在原来的环境进行更改，节点指引的奖励为由rrt的下一时刻距离或者目标点（当rrt大于③时）计算，**

**模型输入和输出都不变，其他基本保持不变**

**1.将RRT规划模块搬过来，添加new\_jammer**

**2.将奖励函数进行重写**

**3.render添加迷雾和全局规划**

**②36节点：先实现①，再修改为5节点环境，然后再运行。**

**①编写函数转化位置坐标，转化cbs或这d\*实现**

**②编写业务坐标**

**③设置信道和网络**

**④跑完获得结果后分析对比**

**⑤分析对应的36节点的结果**

**⑥编写实现对应的场景**

**36节点复杂度降低算法**

**①分簇跟随控制，簇内队形变化，避障控制**

**②分簇轨迹重复算法**

**连续密集障碍环境（如城市建筑群、森林、仓库货架）中，传统编队算法确实面临巨大挑战。这些场景的复杂性主要体现在：**

1. **障碍物连续性：障碍物间距小、分布无规律，队形需频繁变形**
2. **通信遮挡：建筑物导致信号中断，分布式决策失效**
3. **路径歧义：多重绕行路径需全局优化，非局部避障能解决**

**编队队形： 菱形、v形、一字形**

**36节点：分4队，每队9个节点，3\*3为一队。  
碰撞避免：原先是单节点判定，现在是集群碰撞判定，需要修改判定规则，包括集群间和建筑物。**

**局部规划：相当于四个虚拟节点，需要判断是否连通，直接用原来的函数。**

**队形相当只生成leader节点决策，然后其他同编队节点按照菱形编队控制实现跟随。通过矩阵运算获取。**

**①仿真5节点对比结果**

**②整理对应的场景目前结果，以及还需要的东西**

**③对应的36节点的结果图**

**④对应的5节点训练结果图**

**场景：第一个场景：添加分群业务，查询ITM是怎么计算地形的pathloss的。**

**①第一个场景完善（已经实现第一个场景，后续还在等待结果）**

**②分节点编队完善（实现转弯放缩队形变换）**

**③保存五节点数据，大于108就保存，（修完bug，是因为reversed\_flag）**

**④改变地形尺寸和功率跑出不同的数据进行验证(代跑)**

**无人机：25dBm，干扰机：25:2:35  
看下要不要写脚本自动实现 (实现)**

**⑤36节点优化（有问题，应该不同）**