Beijing Forest Studio

北京理工大学信息系统及安全对抗实验中心

面向无人机系统的漏洞挖掘

硕士研究生 张耕源

2026年05月24日

问题回溯

- 问题回溯
  - 报告时长较短，时间把控不到位
  - 读ppt现象较为严重

- 相关内容
  - 2024.05.26 邵思源《面向网络应用程序的模糊测试》
  - 2025.01.19 杨语航《面向操作系统的模糊测试》
  - 2025.06.30 张浩然《软件灰盒定向模糊测试技术》
  - 2026.02.08 徐菊彬《协议模糊测试方法》

内容提要

- 预期收获
- 案例引入
- 题目内涵解析
- 研究背景与意义
- 研究历史与现状
- 知识基础
- 算法原理
  - DPFuzzer
  - RouthSearch
- 特点总结与工作展望
- 参考文献

预期收获

- 预期收获
  - 1. 了解无人机系统的基本结构
  - 2. 了解无人机系统的漏洞类型
  - 3. 掌握无人机系统漏洞挖掘的新的视角

案例引入 无人机灯光秀集体坠落事故

- 2024年12月21日美国奥兰多节日无人机灯光秀集体坠落
- 500架全自主编队无人机进行节日灯光秀表演
- 无人机群在升空变换队列时，未能按照预定轨迹同步飞行，在空中发生大规模密集互撞，随后如雨点般散落坠入湖中及观众席
- 地理围栏参数错误(1->5)、坐标轴软件错位、路径规划文件丢失

题目内涵解析与研究目标

- 内涵解析（面向无人机系统的漏洞挖掘）
  - 无人机系统：由无人驾驶航空器、通信链路、控制站及全套自主导航组件构成的高协同网络物理系统
  - 漏洞挖掘：用自动化或半自动化方法（如模糊测试），在系统部署前主动识别并捕获程序或固件中潜伏的安全边界缺陷与非预期逻辑漏洞的技术
  - 无人机系统的漏洞挖掘：通过向其控制程序和通信协议发送随机或畸形的输入数据，以检测并发现其中软件缺陷与安全漏洞的过程

- 研究目标
  - 发现并修复机载控制程序中的软件缺陷与逻辑漏洞
  - 验证无人机系统在复杂与对抗性场景下的鲁棒性与可靠性

研究背景与意义

- 研究背景
  - 无人机系统的广泛应用
    - 无人机(UAV)在农业、气象监测、搜索救援以及军事行动等众多领域得到了前所未有的广泛应用
  - 传统模糊测试存在物理行为感知盲区
    - 现有模糊测试工具依赖内存崩溃等判定标准，无法捕获“程序正常但飞机失控”等物理行为缺陷
  - 飞控软件输入空间庞大且高维
    - 无人机飞行控制软件包含数千个参数和控制指令，传统测试方法难以穷尽所有组合
- 研究意义
  - 对无人机固件的飞行控制、导航、避障等特定功能，定制特殊的漏洞挖掘方法

研究历史与现状 无人机系统漏洞挖掘

PGFuzz:通过预处理提取MTL(度量时序逻辑)策略缩减输入空间；在模拟器中执行策略引导式模糊测试，实时计算当前状态与违反策略之间的距离并消除环境噪声干扰，通过最小化全局距离来诱发违规；利用后处理对发现的漏洞序列进行最小化简化，以便根因定位

ICSEARCHER：引入了基于深度学习（LSTM）的预测器来替代昂贵的物理模拟或实机执行，极大地提高了模糊测试的效率 。同时提出了片段偏差评估机制，通过累积一段时间内的预测误差来降低单次预测偏差或传感器噪声的干扰，提高了检测的鲁棒性

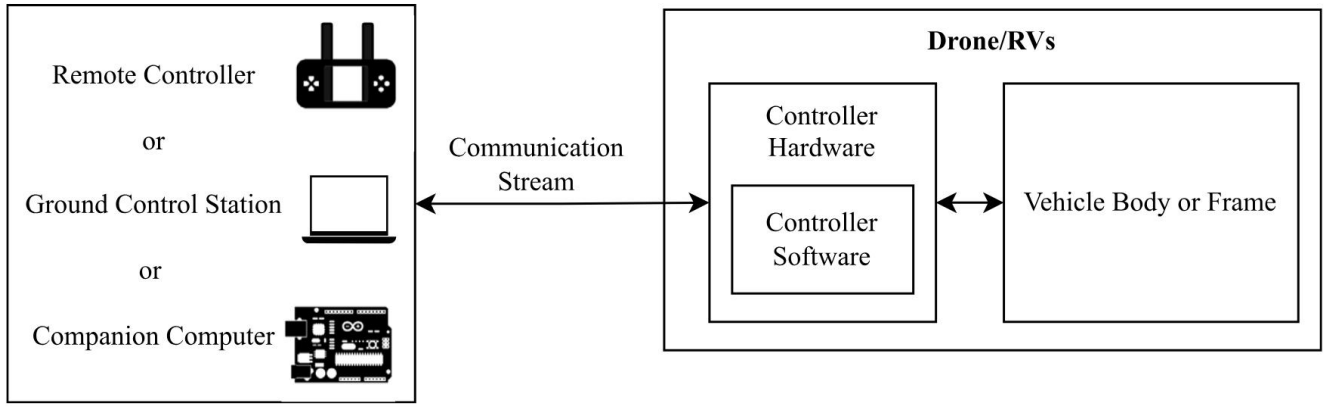
RouthSearch：通过经典控制理论劳斯-赫尔维茨稳定性判据联合高效坐标搜索，从飞控系统配置规范与动态仿真行为中推断多维PID参数安全约束区域，通过空间降维与边界连续性复用大幅精简测试路径，并利用离线时序逻辑预言机进行全轨迹异常行为校验与漏洞发掘

DroneID：逆向大疆固件和无线电物理层，破解了专有的DroneID和DUML协议结构,利用廉价的通用硬件（SDR）构建解码器，证实了DroneID协议未加密并会泄露操作员位置,最后开发了一个针对DUML协议的黑盒模糊测试器，通过自动监测手机App界面的异常变化来捕捉传统方法难以发现的逻辑漏洞

DPFuzzer：通过基于进化算法的场景生成联合环境风险因子（ERF）指标引导，从三维物理空间与无人机高频运行状态中推断隐性安全威胁，并利用碰撞、静止和超时等在线测试预言机拦截并捕获逻辑缺陷 。该方法能够实现超过37%至116%的关键场景生成数量提升，同时保持了较低的时间开销

知识基础 无人机系统基础架构

- 指令与决策源
  - 遥控器、地面控制站、机载电脑
- 通信数据流
  - 无线电数传/总线传输
  - MAVLink协议
- 无人机实体
  - 控制器
    - 控制器硬件：飞控的物理电路板（单片机），里面集成了加速度、姿态感知硬件等
    - 控制器软件：刷写在飞控芯片里的底层固件（如PX4或Ardupilot），包含闭环控制算法（如PID控制器等）
  - 物理机体与结构
    - 飞控软件计算出的控制信号，会转变为PWM电信号发送给电调（ESC），电调进而控制电机和螺旋桨（Actuators）改变转速



知识基础 无人机系统软件缺陷和漏洞类型

- 输入验证缺陷
  - 无人机机载控制程序中未能正确检查、或完全缺失了对输入有效值的校验
  - 当系统接收到外部非预期或畸形的数据时，由于没有做好边界和格式限制，会导致程序崩溃或行为异常
  - 缓冲区溢出、参数范围规范错误、浮点异常、并发安全问题
- 控制语义缺陷
  - 控制逻辑在实现层面出现错误或系统规范本身存在设计缺陷
  - 即使输入的数据格式完全合法，由于控制算法或业务逻辑本身的漏洞，无人机仍会做出错误的物理决策（例如无法保持正确的下降角度导致断翼）
  - 异常事件处置失败、交通/安全规则违背问题
- 通信协议与流缺陷
  - 协议在设计或实现时缺乏必要的安全防护，导致数据在传输过程中容易被窃听、篡改或伪造

知识基础 度量时序逻辑

- 度量时序逻辑(MTL)
  - 经典时序逻辑(LTL)的延展，引入了时间度量约束（如时段、步长等）
  - 将“一个时间段的飞行安全规范”精确表达为数学公式
    - $\{(ALT_t < RTL\_ALT) \wedge (Mode_t = RTL) \rightarrow (ALT_{t-1} < ALT_t)\}$
    - 返航模式且当前高度小于设定阈值必须处于爬升状态

- 离线测试预言机
  - 在飞行仿真结束后，通过静态扫描完整的历史轨迹日志进行“全局事后审计”，精准判定无人机在整个生命周期中是否违反了长周期安全规范
- 在线测试预言机
  - 在无人机飞行过程中，通过滑动窗口等机制实时监控当前的状态数据流进行“即时动态检测”，在运行阶段对异常物理行为进行捕捉与拦截

DPFuzzer: Discovering Safety Critical Vulnerabilities for Drone Path Planners

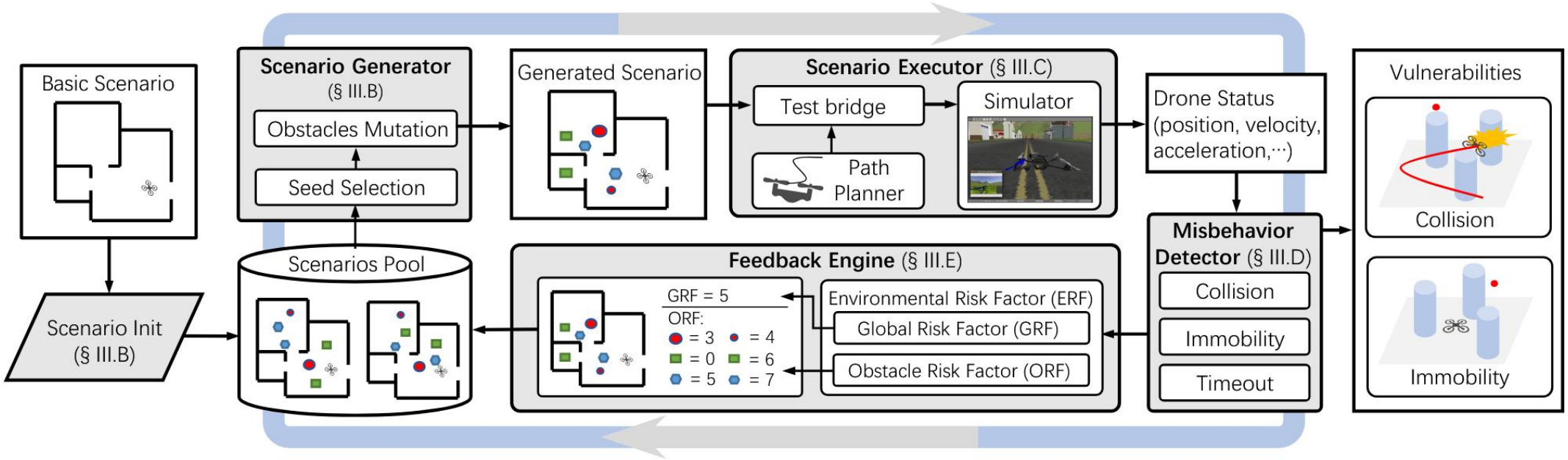
算法原理 TIPO

T	目标	自动化发现无人机路径规划器的潜在的安全漏洞
I	输入	初始障碍物集合、待测路径规划器*3
P	处理	1.场景初始化，场景生成器采用环境风险因子指标生成场景 2.场景执行器将生成的场景在仿真器中实现并执行路径规划器 3.异常行为检测器实时监控无人机状态，借助在线测试预言机检测异常行为 4.根据检测结果迭代生成场景，最终生成触发漏洞的安全关键场景
O	输出	成功触发漏洞的安全关键测试场景集合、安全漏洞类型

P	问题	现有方法在生成多样化障碍场景以及测试路径规划器方面能力有限
C	条件	仿真环境能够解析物理场景描述，实时提供无人机状态数据
D	难点	如何高效生成能够触发多样化安全漏洞的场景
L	水平	2025 CCFA ICSE

算法原理 创新说明

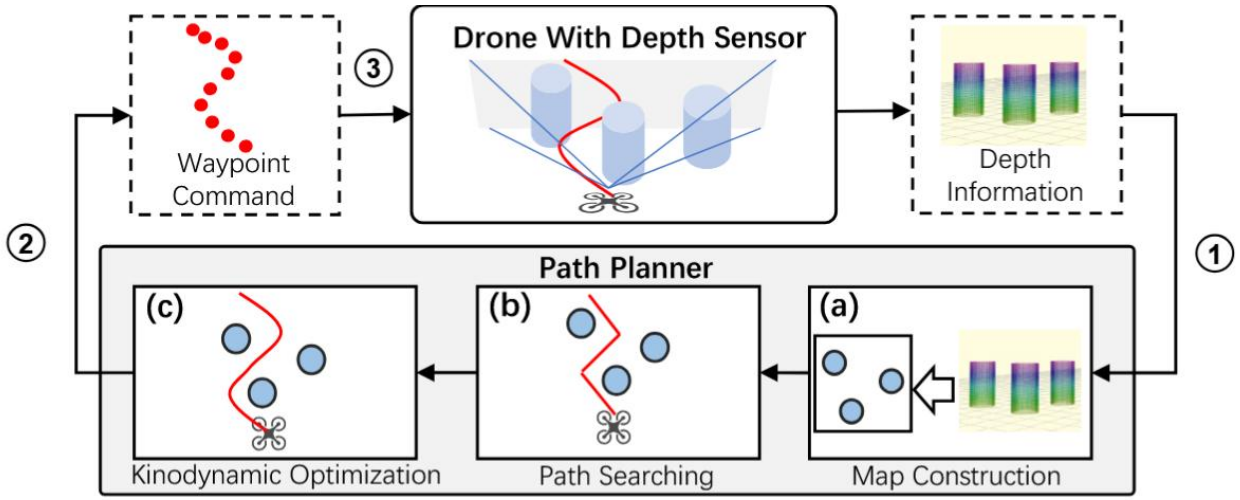
- **DPFuzzer**
  - 首次提出一个新型的测试框架，聚焦于无人机路径规划器中此前未被重视的安全关键漏洞
  - 提出环境风险因子(ERF)指标用以抽象场景的潜在威胁，高效指导关键场景生成
  - 发现了多种类型的漏洞并在真实场景验证了漏洞



知识基础 无人机路径规划器

- 地图构建
  - 将传感器（如激光雷达、深度相机）采集到的原始深度信息，实时转换为三维网格地图
- 路径搜索
  - 利用图搜索或空间采样算法，在宏观地图中快速寻找一条连接起点与终点的通用拓扑路径
- 动力学优化
  - 引入无人机的物理特性约束（如最大速度、加速度、转弯半径和电机推力极限），将上述粗略路径转化为一条连续、平滑且不与障碍物重叠的物理轨迹，转换航点

地图构建误差、路径搜索失败、动力学优化

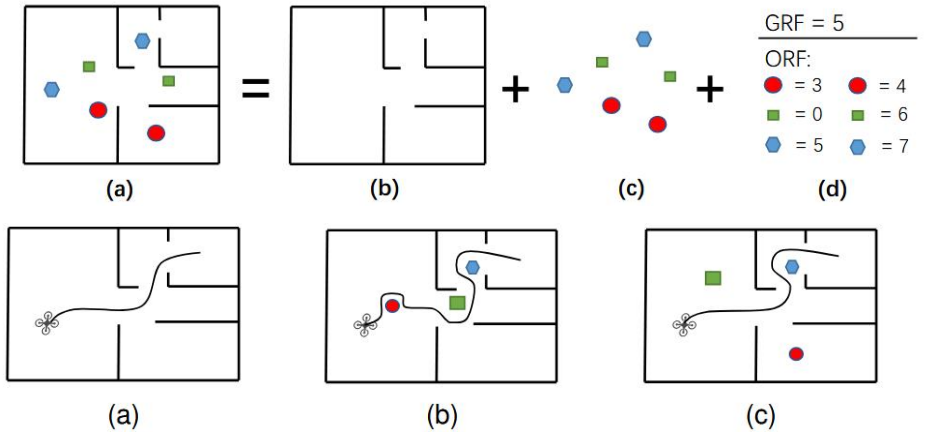


知识基础 四类风险指标

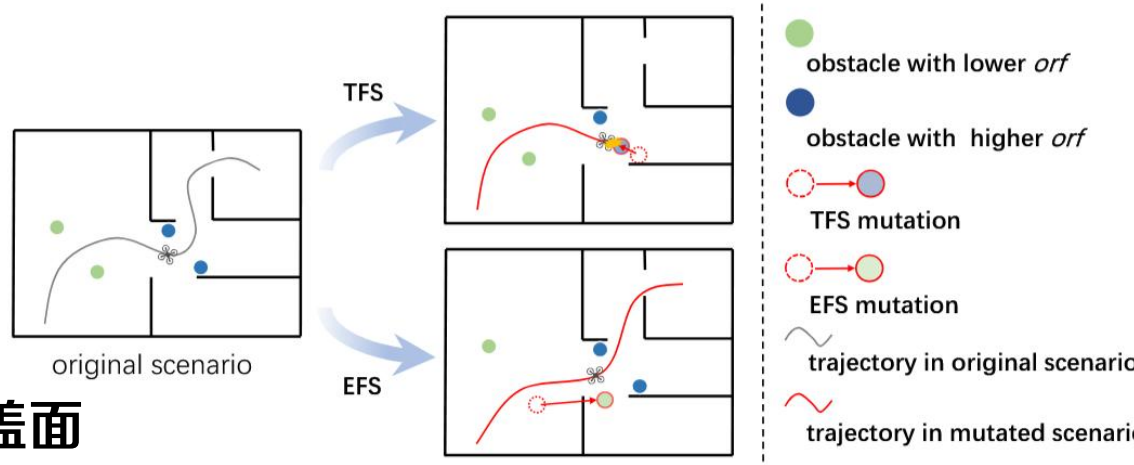
- 反应风险:衡量与速度矢量延伸方向上直接存在的障碍物发生碰撞的概率
  - $R_{reac,i} = \frac{\|\vec{v_i}\|}{d_{vel,i}}$
- 硬转向风险:无人机沿规划轨迹移动时，速度过快时无法响应突然的横向加速度
  - $R_{turn,i} = \|\vec{v_i}\| \cdot \left\| \vec{a_i} - \vec{v_i} \cdot \frac{\vec{a_i} \cdot \vec{v_i}}{\|\vec{v_i}\|^2} \right\|$
- 急刹车风险:路径规划器尝试急刹车时,往往会导致无人机停滞
  - $R_{brake,i} = \|\vec{v_i}\| \cdot \left| \frac{\vec{a_i} \cdot \vec{v_i}}{\vec{v_i} \cdot \vec{v_i}} \right|, a^i \cdot v^i < 0$
- 距离风险：衡量至最近障碍物的最小距离
  - $R_{dist,i} = \frac{1}{d_{min,i}}$

算法原理 场景生成器

- 场景初始化
  - 构建初始测试场景池
    - 获取无障碍轨迹
    - 围绕轨迹随机布置可变障碍物

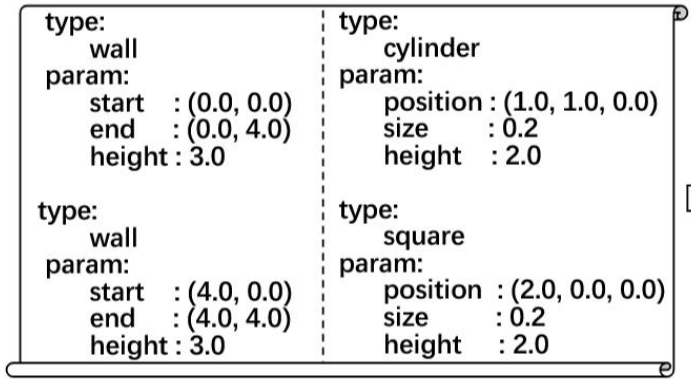


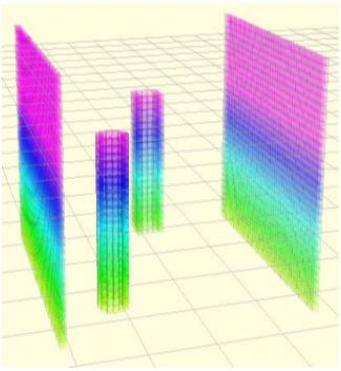
- 种子筛选
  - 依据全局风险GRF将所有场景降序排列，挑选得分最高的M个场景进入种子池
- 双策略障碍物变异
  - 触发优先策略
    - 微调高ORF障碍物的位置，触发临界状态
  - 探索优先策略
    - 剧烈随机变异低风险障碍物，扩大测试覆盖面

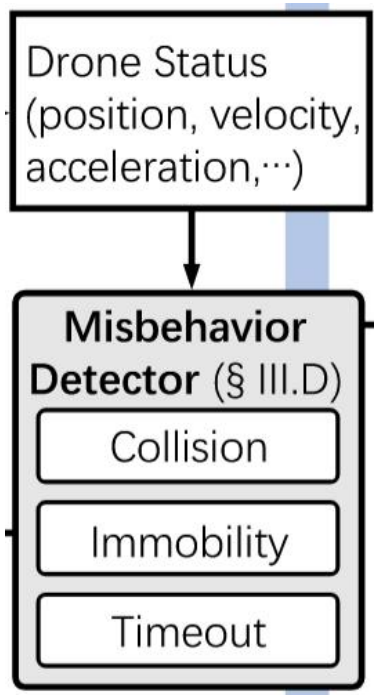


算法原理 场景执行

- 场景仿真与执行
  - 环境解析
    - 将变异后的障碍物信息加载到仿真器
  - 轨迹飞行与实时日志记录（P、V、A）
    - 位置序列、速度序列、加速度序列
- 异常行为监控（在线测试预言机）
  - 碰撞监控
    - 无人机中心与最近障碍物距离<0.15米，撞击/坠毁
  - 静止监控
    - 无人机在某处停止并悬停时间>10秒，规划器卡死/电池耗尽坠机
  - 超时监控
    - 无人机未能在合理时间内达到指定终点，迷路/原地转圈

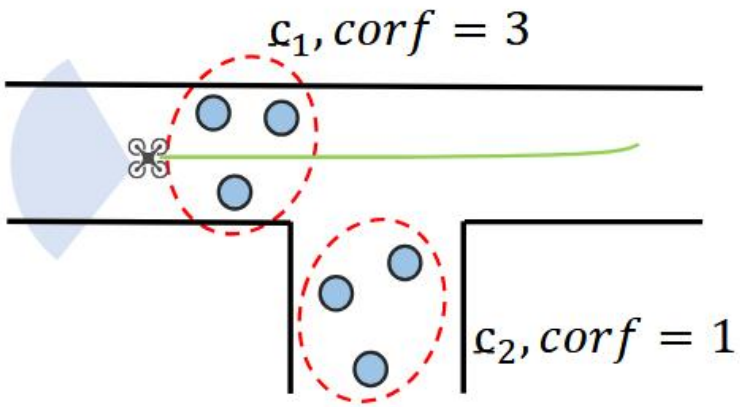






算法原理 风险反馈

- 风险反馈计算
  - 计算每个位置的潜在风险（归一化、欧几里得范数）
    - 反应风险、急转弯风险、急刹车风险、距离风险
    - $PR_i = \sqrt{NR_{reac,i}^2 + NR_{turn,i}^2 + NR_{brake,i}^2 + NR_{dist,i}^2}$



  - 计算每个障碍物风险因子ORF
    - $orf_j = max(\{\frac{PR_i}{d_{j,i}} | i = 1,2,3 \cdots n\})$
  - 计算场景的全局风险因子GRF
    - 计算每个聚类的障碍物簇的平均ORF（记作corf）
    - $GRF = max(\{corf_i | i = 1,2,3 \cdots k\})$
  - 更新后ERF（包含各个orf_j与GRF）的场景会被重新放回场景池，开启新一轮的优胜劣汰和变异测试，优先变异GRF高的场景，触发优先策略、探索优先策略

实验设计 数据资源

- 数据资源
  - 三种开源路径规划器：Ego-Planner、Ego-Planner-Swarm（多选一）、FUEL
  - 测试场景：仿真场景（2个室内场景、小巷/胡同场景）、物理场景
- 评估标准
  - 生成的关键场景数量
  - 漏洞类型的覆盖率
  - 时间开销
- 对比方法
  - 随机测试法
    - 每一轮场景生成中，完全随机变异和摆放可变障碍物的位置与形状
  - 基于距离的遗传算法
    - 仅依赖“无人机与障碍物之间的绝对距离”单一指标引导场景变异

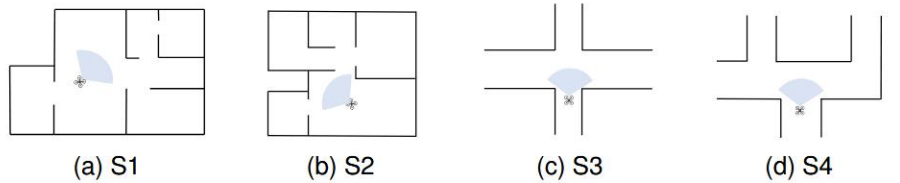


Fig. 9. Plan view of basic scenarios for testing.

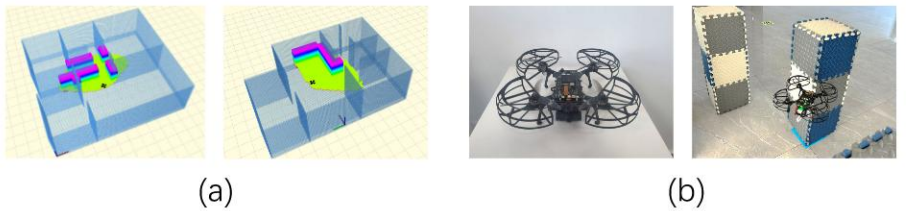


Fig. 10. (a)Simulator. (b)Real-world drone.

实验设计 漏洞类型

- 漏洞类型与数量
  - DPFuzzer共发现969个导致异常行为的严重场景

漏洞描述		影响*	能力不足	路径规划器		
				EP*	ES*	燃料
#1	未能避免侧向碰撞	C	障碍物规避	✓	✓	✓
#2	未能避免追尾碰撞	C	避障功能	✓	✓	✓
#3	找到有效轨迹但在障碍物附近停止移动	C,I	避障功能	✓		✓
#4	未能找到有效的航线到达目的地，正在原地盘旋	U	路径搜索	✓	✓	
#5	未能找到有效的航线到达目的地，保持悬停状态	I	路径搜索	✓		✓
#6	优先选择更短轨迹而非安全，导致碰撞或无法移动	C,I	决策过程	✓	✓	✓
#7	优先考虑平滑轨迹而非安全性，导致碰撞或无法移动	C,I	决策过程	✓		✓
#8	地图构建不精确，导致无法移动或原地打转	I,U	感知	✓	✓	✓

* **[影响]** C：碰撞。I：无人机失去移动能力。U：无人机进入不稳定状态。

* **[路径规划器]** EP：自主规划器。ES：自主规划器群集。✓：路径规划器存在漏洞。

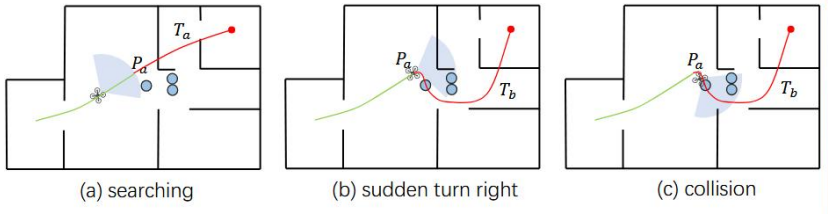


Fig. 11. Schematic diagram about vulnerability #1.

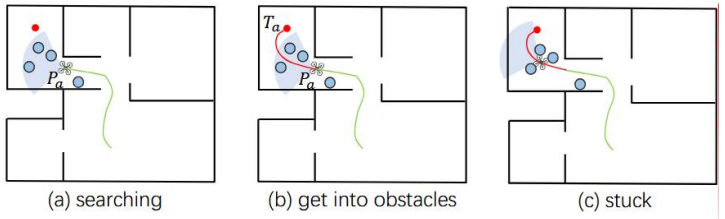


Fig. 12. Schematic diagram about vulnerability #3.

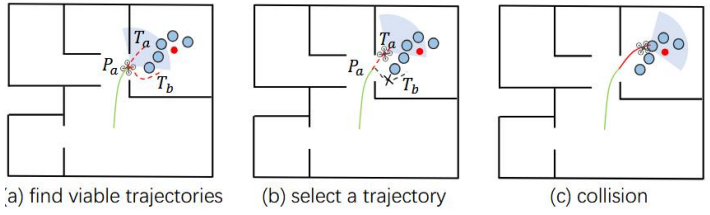


Fig. 14. Schematic diagram about vulnerability #7.

实验设计 对比实验

- 对比实验结果
  - DPFuzzer在关键场景生成数量与漏洞类型覆盖率上均显著优于随机测试（Rand）和基于距离的遗传算法（D-GA）

Scenario	Path Planner	Alg.	The number of critical scenarios							The number of covered types						
			R1	R2	R3	R4	R5	Avg.	Δ	R1	R2	R3	R4	R5	Avg.	Δ
S1	Ego-Planner	Ours	11	12	8	12	9	10.4	+2.0(23.81%)/	5	5	4	6	6	5.2	+2.6(100.00%)/
		D/R	7/2	9/3	11/7	10/3	5/4	8.4/3.8	+6.6(173.68%)	3/2	3/3	3/3	2/2	2/2	2.6/2.4	+2.8(116.67%)
	Ego-Planner-Swarm	Ours	9	6	6	4	5	6.0	+1.8(42.86%)/	4	4	4	3	3	3.6	+1.2(50.00%)/
		D/R	3/2	2/5	4/4	4/1	8/4	4.2/3.2	+2.8(87.50%)	2/2	2/3	3/2	3/1	2/2	2.4/2.0	+1.6(80.00%)
	FUEL	Ours	13	10	11	14	11	11.8	+2.4(25.53%)/	6	5	5	6	5	5.4	+1.2(28.57%)/
		D/R	8/7	10/6	10/5	11/5	8/6	9.4/5.8	+6.0(103.45%)	4/4	4/3	5/4	4/3	4/4	4.2/3.6	+1.8(50.00%)
S2	Ego-Planner	Ours	10	8	12	14	7	10.2	+0.4(4.08%)/	6	5	6	5	6	5.6	+2.0(55.56%)/
		D/R	7/6	10/7	13/6	10/7	9/6	9.8/6.4	+3.8(59.38%)	4/3	4/3	3/3	3/2	4/3	3.6/2.8	+2.8(100.00%)
	Ego-Planner-Swarm	Ours	9	8	10	5	11	8.6	+0.2(2.38%)/	5	3	5	4	5	4.4	+1.4(46.67%)/
		D/R	8/7	6/5	7/10	9/2	12/8	8.4/6.4	+2.2(34.38%)	3/3	2/2	3/3	3/2	4/3	3.0/2.6	+1.8(69.23%)
	FUEL	Ours	9	13	10	10	13	11.0	+5.6(103.70%)/	5	4	6	5	6	5.2	+2.4(85.71%)/
		D/R	5/5	6/4	6/3	6/5	4/4	5.4/4.2	+6.8(161.90%)	2/4	3/3	3/2	3/3	3/3	2.8/3.0	+2.2(73.33%)
S3	Ego-Planner	Ours	2	2	3	2	4	2.6	+0.8(44.44%)/	2	2	2	2	3	2.2	+0.6(37.50%)/
		D/R	2/1	3/0	1/2	1/0	2/1	1.8/0.8	+1.8(225.00%)	2/1	2/0	1/2	1/0	2/1	1.6/0.8	+1.4(175.00%)
	Ego-Planner-Swarm	Ours	3	0	8	0	2	2.6	+0.6(30.00%)/	2	0	4	0	2	1.6	+0.2(14.29%)/
		D/R	1/0	3/0	3/0	1/0	2/3	2.0/0.6	+2.0(333.33%)	1/0	2/0	1/0	1/0	2/2	1.4/0.4	+1.2(300.00%)
	FUEL	Ours	9	5	6	6	9	7.0	+4.0(133.33%)/	6	5	5	5	5	5.2	+3.0(136.36%)/
		D/R	3/1	3/2	2/3	3/3	4/2	3.0/2.2	+4.8(218.18%)	3/1	2/2	1/2	2/3	3/2	2.2/2.0	+3.2(160.00%)
S4	Ego-Planner	Ours	9	8	4	8	7	7.2	+1.4(24.14%)/	5	3	4	5	2	3.8	+1.0(37.71%)/
		D/R	4/5	8/3	9/1	6/3	2/4	5.8/3.2	+4.0(125.00%)	3/2	3/1	4/1	3/2	1/2	2.8/1.6	+2.2(137.50%)
	Ego-Planner-Swarm	Ours	4	6	1	3	4	3.6	+0.6(20.00%)/	3	3	1	3	2	2.4	+0.4(20.00%)/
		D/R	2/0	3/2	2/3	3/2	5/0	3.0/1.4	+2.2(157.14%)	1/0	3/1	1/2	2/1	3/0	2/0.8	+1.6(200.00%)
	FUEL	Ours	7	5	8	9	5	6.8	+4.0(142.86%)/	5	3	4	5	4	4.2	+1.8(75.00%)/
		D/R	4/3	2/5	3/2	4/0	1/3	2.8/2.6	+4.2(161.54%)	4/2	2/3	2/2	3/0	1/3	2.4/2.0	+2.2(110.00%)

* **D/R:** D-GA/Rand, the result of the Distance-based GA method and the Random method, respectively.
* **Δ:** The result *DPFuzzer* compared to the Distance-based GA method and the Random method, respectively.

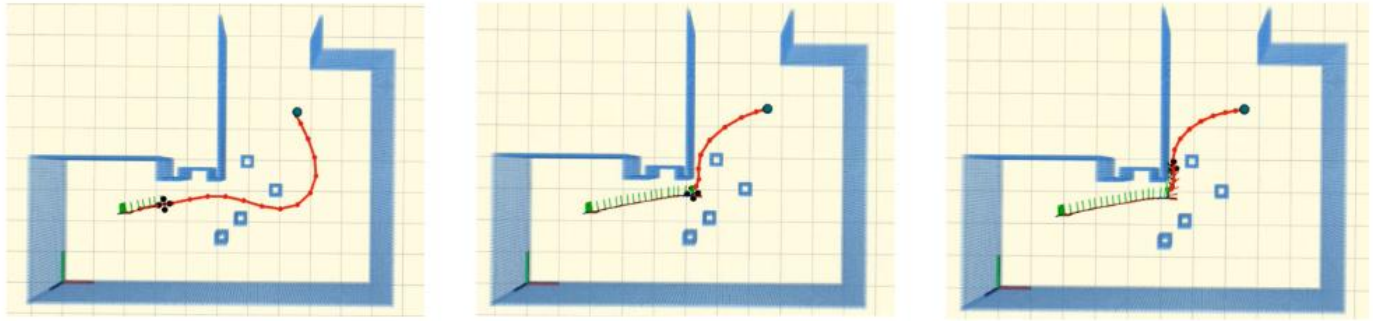
实验设计 方法效率

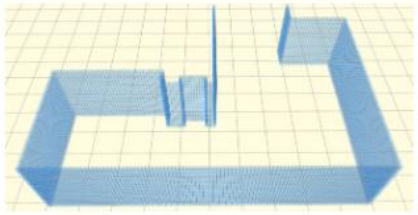
- 时间开销
  - DPFuzzer在生成关键场景的数量(平均提升116.26%)和漏洞类型覆盖率(平均提升103.33%)方面均优于随机生成法
  - 相较于D-GA，DPFuzzer在生成的关键场景数量上平均提升37.19%,在漏洞类型覆盖率上平均提升57.42%
  - 相较随机生成方法，反馈模块额外耗时1.59秒，耗时增加6.54%

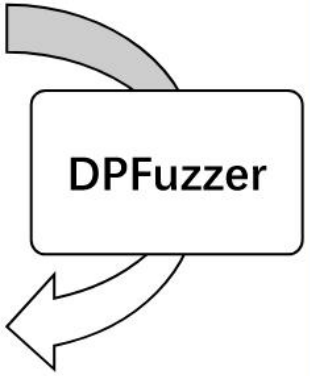
Fig. 15. Breakdown of the time consumption.

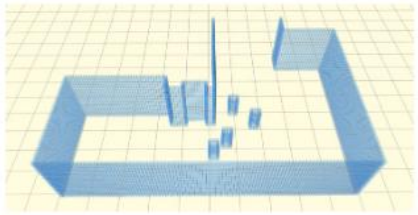
实验设计 真实场景验证

- 硬件设备
  - 搭载 Ego-Planner 路径规划器的商用无人机、水平视场角为120度、感知范围为 4 米的深度相机（侧向碰撞、临近障碍物卡死、无法搜路悬停、感知地图不精准）

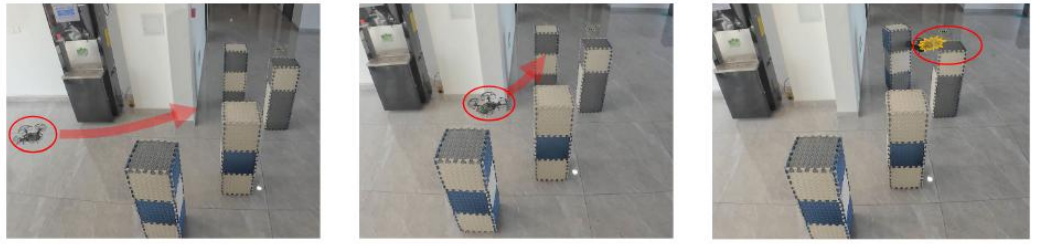








(a) An obstacle-free trajectory.



算法总结

- 算法贡献
  - 提出一个新型的测试框架
    - 聚焦于无人机路径规划器中此前未被重视的安全关键漏洞
  - 提出环境风险因子(ERF)指标
    - 抽象场景的潜在威胁，高效指导关键场景生成
  - 发现了多种类型的漏洞并在真实场景验证了漏洞
- 算法局限
  - 环境噪声干扰
    - 无人机在现实中的物理轨迹和运动细节不可能与纯理想化的仿真完全一模一样
  - 机载算力瓶颈
    - 算力的限制会导致现实中的无人机比仿真更容易因为“反应不及时”而发生惨剧
  - 飞控系统延迟与协调
    - 飞控系统在实际控制中存在物理滞后和不精准

RouthSearch: Inferring PID Parameter Specification for Flight Control Program by Coordinate Search

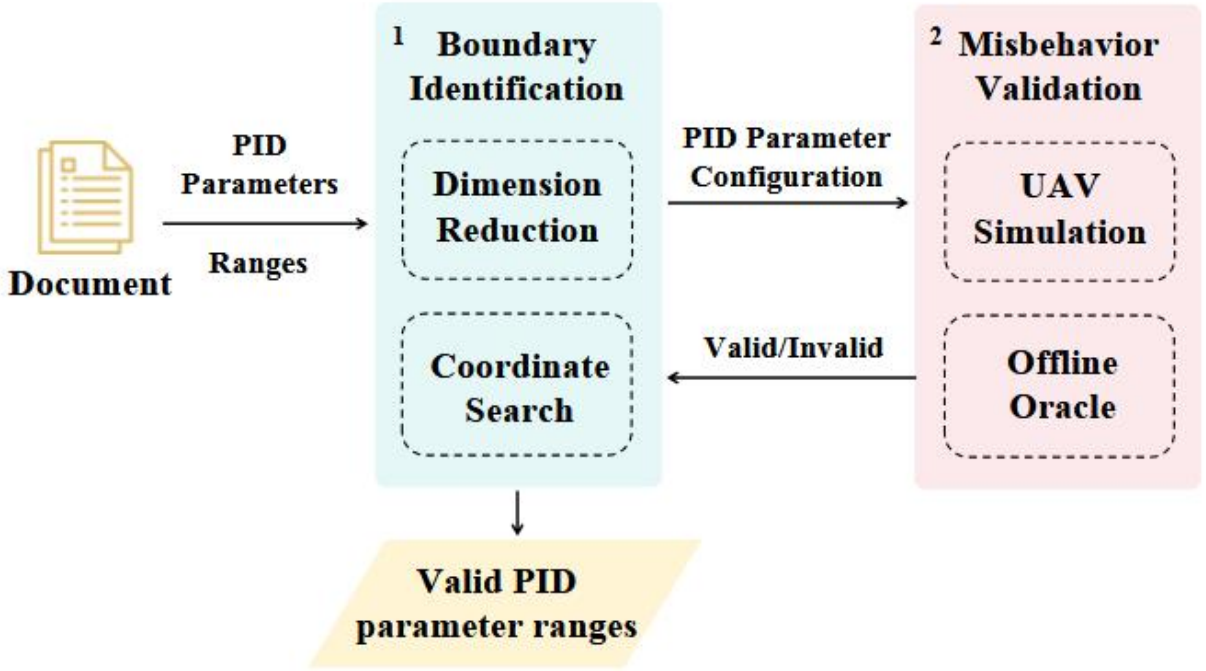
算法原理 TIPO

T	目标	解决无人机飞行控制程序中因PID参数配置错误导致的安全问题
I	输入	PID参数范围与步长、飞行任务
P	处理	1.从文档获取PID取值范围，并将三维搜索空间简化为多个二维平面 2.结合劳斯稳定性判据与坐标搜索，动态识别PID参数的安全分界线 3.从二维平面的边缘出发，结合前述理论边界，快速刻画边界偏移 4.通过无人机模拟与逻辑校验拦截误配置并挖掘代码漏洞
O	输出	分类边界线、逻辑漏洞、误配置实例

P	问题	现有的测试方法在三维空间中搜索效率极低且每次验证PID参数都需要动态飞行模拟，时间成本较高
C	条件	黑盒测试、仿真环境依赖、先验知识
D	难点	在受外部噪声和飞行模式差异影响的巨大三维空间内，高效且精准地定位相互关联的PID参数安全边界
L	水平	2025 CCFA ISSTA

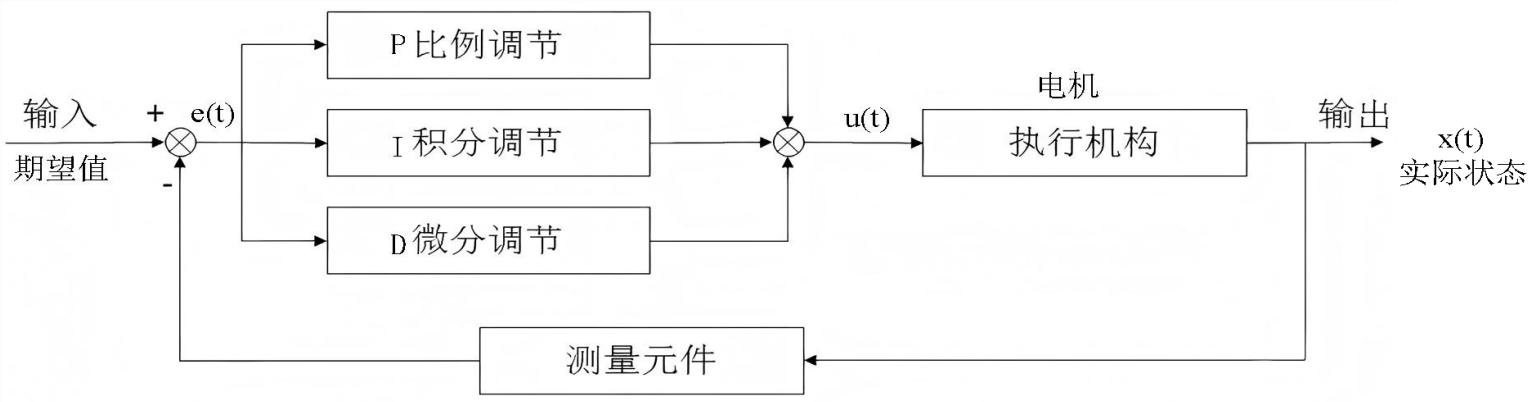
算法原理 创新说明

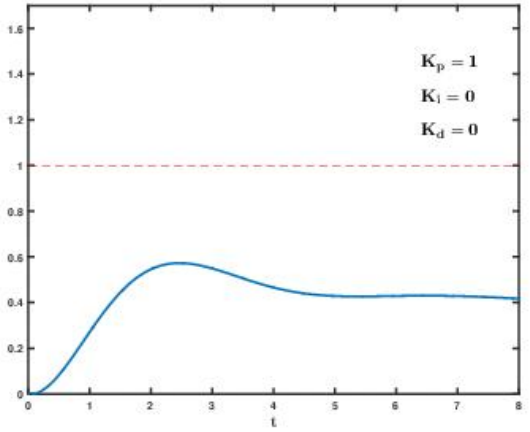
- RouthSearch
  - 提出一种基于劳斯-赫尔维茨稳定性判据的方法来确定三维PID参数的有效范围的方法。该有效范围可帮助用户避免在飞行过程中错误配置PID参数
  - 采用高效坐标搜索技术,针对无人机飞行控制程序中的单个PID参数错误配置进行定位



知识基础 PID参数调节

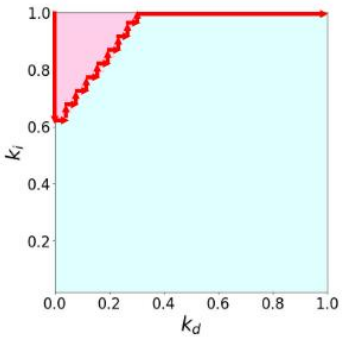
- 控制理论
  - 通过 PID 动态调节，使系统误差$e(t)$在时间趋于无穷大时逼近于零
  - PID公式
    - $u(t) = k_p e(t) + k_i \int_0^t e(\tau)d\tau + k_d \frac{de(t)}{dt}$
    - 比例参数P:成比例对抗当前误差 。过大会引发系统剧烈震荡，过小导致响应迟钝
    - 积分参数I:随时间累加历史误差，消除系统静差。过大易引发严重超调与持续绕圈
    - 微分参数D:根据误差变化率提供阻尼，起到提前刹车作用 。配置不当会放大高频物理噪声。



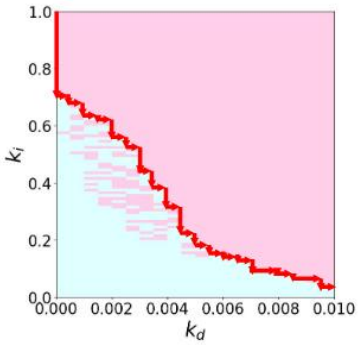


知识基础 劳斯-赫尔维茨稳定性判据

- 劳斯判据
  - 代数控制理论中判定系统稳定性的经典方法
  - 闭环受控物理方程
    - $\frac{d^2x(t)}{dt^2} + a_2\frac{dx(t)}{dt} + a_1x(t) = u(t)$
  - PID控制输入项
    - $u(t) = k_pe(t) + k_i\int_0^t e(\tau)d\tau + k_d\frac{de(t)}{dt}$
  - 理论安全边界公式（三阶常微分方程、劳斯-赫尔维茨判据）
    - $k_p + a_1 > 0, k_d + a_2 > 0, k_i > 0, (k_p + a_1)(k_d + a_2) > k_i$
- 实际边界和理论边界偏差
  - 未知的边界偏移：飞行模式约束、高频随机噪声干扰
- 坐标搜索：通过坐标逐点搜索来最小化指定的目标函数或其他搜索目标



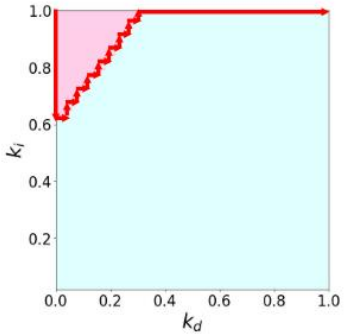
(a) Linear Boundary Line



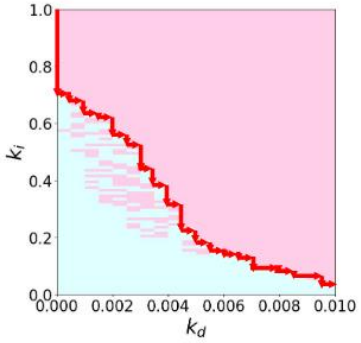
(b) Noisy Boundary Line

算法原理 空间降维与起点确定

- 输入与初始化
  - 从无人机飞控官方文档提取PID参数的初始可配置取值范围、递增搜索步长以及设定的飞行任务
- 空间降维
  - 固定k_p为特定常量p_{const}，将三维空间切分为n个二维平面(k_i, k_d)
- 确定边界
  - Routh-Hurwitz判据表明存在一条将稳定PID参数值与不稳定PID参数值分隔开的分类边界。在平面(ki ,kd)上,当$k_d = p_{const}$时,可得到如下线性边界:
  - $(p_{const} + a_1)(k_d + a_2) = k_i$
  - a_1和a_2是由物理系统和具体飞行模式决定的常数
  - 有效参数和无效参数区域保持近似连续性



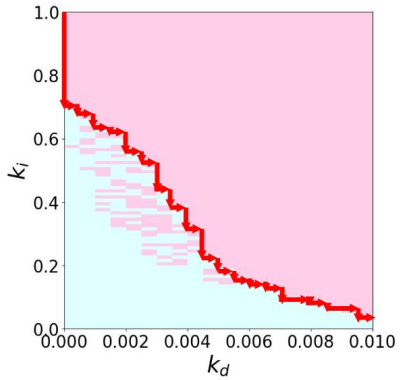
(a) Linear Boundary Line

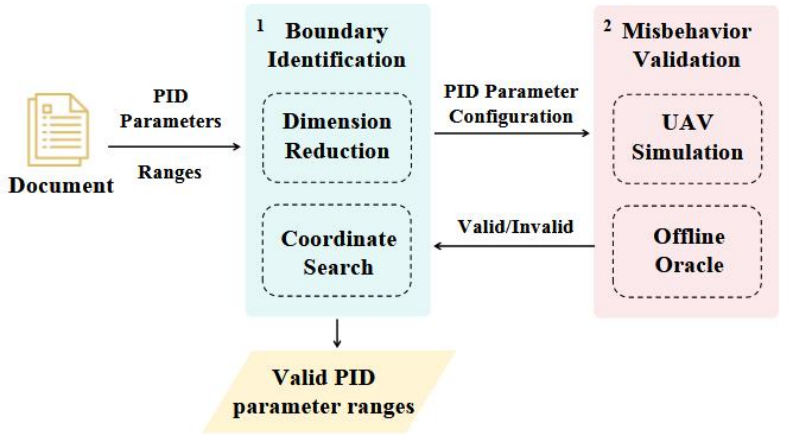


(b) Noisy Boundary Line

算法原理 基于坐标的高效搜索算法

- 高效搜索算法
  - 初始化层搜索起点
    - 积分参数k_i轴初始化为最大值i_{max}，沿k_d轴横向推进，按固定步长从下限逐步递增到上限
  - 自适应边界判定与追踪
    - 每一个k_d位置上调用offline oracle验证当前配置安全性
      - Oracle == False，当前k_i过高，以微小步长执行向下搜索直到True
      - Oracle == True，以微小步长执行向上搜索直到False，退回最后安全点
  - 记忆化复用
    - 成功锁定当前k_d步长下临界安全点i_{save}
      - 将该点(p, i_{save}, d)插入到分类边界线集合BL中
      - 将i_{save}作为新一轮搜索的基准起点





算法原理 异常行为验证模块

- 异常行为验证
  - 基于时序逻辑的安全规范
    - 评估无人机仿真运行的完整物理轨迹ω是否符合MTL公式Φ
    - $\{(ALT_t < RTL\_ALT) \wedge (Mode_t = RTL) \rightarrow (ALT_{t-1} < ALT_t)\}$
    - 返航模式且当前高度小于设定阈值必须处于爬升状态
  - 验证模块
    - 无人机仿真器
      - 接收当前待测试的三维PID参数配置和预设的飞行命令在SITL或jMAVSim等轻量级虚拟环境中动态运行无人机，全程记录无人机各项物理状态的飞行日志
    - 离线预言机（offline oracle）
      - 仿真结束后，该组件会读取并扫描解析生成的完整飞行日志，利用时序逻辑规则库，逐时间步检验整条飞行轨迹是否合规

实验设计 数据资源

- 数据资源
  - 自构建包含有效和无效（误配置）标签的无人机PID参数基准数据集
- 实验环境
  - 无人机飞控程序：两款开源的ArduPilot V4.4.1和 PX4 V1.15.0 alcce7e
  - 动态验证环境：ArduPilot自带的SITL模拟器以及PX4的jMAVSim仿真器
  - 8种无人机飞行模式：
    - ArduPilot： RTL、Zigzag（步进/锯齿）、Circle、Brake模式
    - PX4： Orbit（轨道）、Return（回航）、Land（着陆）、Hold（悬停）模式
- 评估标准
  - MR(Miss Rate,漏报率)、HR(Hit Rate,命中率) $MR = \frac{|GT - RS|}{|GT|}$ $HR = \frac{|RS \cap GT|}{|RS|}$
- 对比方法
  - PGFuzz：带有简单引导的随机搜索策略来发现飞控参数误配置

实验设计 对比实验

- 对比实验结果
  - RouthSearch在48小时内平均挖掘出3853组引发无人机异常的PID误配置实例，基于随机引导搜索的PGFuzz平均仅挖出449组
  - ArduPilot的Brake（刹车）模式差距更加明显
  - 依靠理论边界引导的系统化搜索在效率上远超盲目的随机突变

Mode	RouthSearch	$PGFuzz_1$	$PGFuzz_2$	$PGFuzz_3$
AP:Zigzag	3,390	557	617	518
AP:Brake	11,820	406	289	262
AP:RTL	4,969	1,319	1,325	1,286
AP:Circle	3,533	93	78	76
PX4:Orbit	1,909	290	236	207
PX4:Return	1,092	299	278	256
PX4:Land	2,037	635	495	576
PX4:Hold	2,070	305	193	171
Average	3,853	488	439	419

实验设计 搜索算法比较

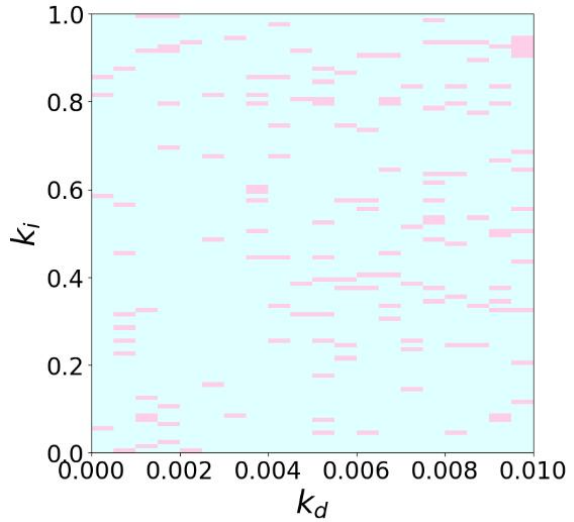
- 搜索算法横向比较
  - 爬山(HC)算法缺乏定量反馈,退化为产生狭窄且不精确边界的局部随机搜索
  - 遗传(GA)算法缺乏适应度引导的优化,仍陷于无方向的随机探索,导致持续低命中率和次优的故障配置检测

Mode	RouthSearch-CS			RouthSearch-HC			RouthSearch-GA		
	MR	HR	number	MR	HR	number	MR	HR	number
AP:Zigzag	31.2%	91.8%	3390	98.3%	25.1%	481	93.0%	20.2%	1928
AP:Brake	10.1%	98.6%	11,820	93.5%	49.3%	2373	89.9%	35.5%	3688
AP:RTL	2.6%	99.7%	4,969	95.0%	37.9%	1379	88.4%	30.3%	3204
AP:Circle	16.9%	74.0%	3,533	95.4%	45.6%	934	90.2%	19.1%	1874
PX4:Orbit	26.2%	81.8%	1,909	98.0%	22.8%	173	96.4%	42.2%	319
PX4:Return	12.1%	98.0%	1,092	98.3%	37.6%	162	96.4%	40.3%	345
PX4:Land	7.5%	97.4%	2,037	96.3%	40.8%	358	95.8%	39.3%	401
PX4:Hold	15.5%	94.8%	2,070	96.8%	21.9%	158	94.5%	29.8%	270
Average	13.0%	92.0%	3853	96.5%	35.1%	752	93.1%	32.1%	1504

实验设计 边界识别准确率分析

- 边界识别准确率分析
  - 在各飞行模式下的平均命中率达到92.0%
    平均漏报率仅为15.3%
  - AP:Zigzag模式：漏报率（MR）偏高
    - 控制系统的调节时间问题
    - 劳斯判据只能保证时间趋于无穷大时的稳定性，未考虑调节时间 。只管‘能不能到’，不管‘多久能到’,当P和I值同时过小时，系统响应过慢导致任务失败
  - AP:Circle模式/PX4:Orbit模式:命中率（HR）偏低
    - 异常行为验证器的过度敏感性
    - 长周期的绕圈任务中，传感器引入的轻微物理噪声会导致物理轨迹产生实际上无害的小偏差，敏感的验证其会将其识别为“异常行为”

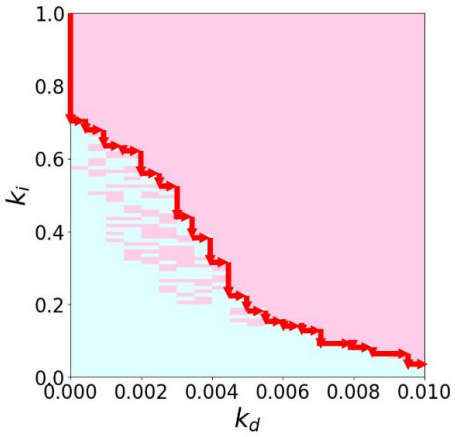
Mode	Total	Identified	Accurate	MR	HR
AP:Zigzag	28,162	21,107	19,367	31.2%	91.8%
AP:Brake	36,584	33,357	32,902	10.1%	98.6%
AP:RTL	27,267	26,624	26,545	2.6%	99.7%
AP:Circle	20,477	23,016	17,022	16.9%	74.0%
PX4:Orbit	8,979	8,103	6,630	26.2%	81.8%
PX4:Return	9,592	8,604	8,429	12.1%	98.0%
PX4:Land	9,610	9,123	8,885	7.5%	97.4%
PX4:Hold	4,930	4,394	4,165	15.5%	94.8%
Average	18,200	16,791	15,493	15.3%	92.0%



实验设计 消融实验

- 向下搜索机制必要性
  - RouthSearch-DSOff的命中率略有提升,因其避免探索无效配置与有效配置混杂的噪声区域
  - 向下搜索策略在识别配置边界方面效果显著

Mode	Total	RouthSearch-DSOff			RouthSearch		
		Number	MR	HR	Number	MR	HR
AP:Zigzag	28,162	531	98.1%	97.3%	19,367	31.2%	91.8%
AP:Brake	36,584	32,515	11.1%	98.2%	32,902	10.1%	98.6%
AP:RTL	27,267	26,531	2.7%	99.7%	26,545	2.6%	99.7%
AP:Circle	20,477	15,678	23.4%	93.1%	17,022	16.9%	74.0%
PX4:Orbit	8,979	3,779	57.9%	99.9%	6,630	26.2%	81.8%
PX4:Return	9,592	5,340	44.3%	99.7%	8,429	12.1%	98.0%
PX4:Land	9,610	6,737	29.9%	98.7%	8,885	7.5%	97.4%
PX4:Hold	4,930	3,175	35.6%	99.5%	4,165	15.5%	94.8%
Average	18,200	11,786	37.9%	98.3%	15,493	15.3%	92.0%



实验设计 消融实验和超参数实验

- 离线验证预言机准确性分析
  - 短周期任务两者专家一致率表现相当
  - 长周期模式差距明显
    - 在线滑动窗口无法容纳长轨迹上下文，容易造成严重误判

Mode	Online	Offline	Mode	Online	Offline
AP:Brake	96.5%	99.4%	PX4:Hold	100%	100%
AP:Circle	72.7%	96.5%	PX4:Land	100%	100%
AP:RTL	99.7%	99.7%	PX4:Return	100%	100%
AP:Zigzag	97.1%	97.1%	PX4:Orbit	28.5%	97.3%

- 抽样评估准确性与效率分析
  - 抽样评估仅需907个CPU小时，仅为穷举评估（9,067小时）的十分之一

Mode	Experimental Setting	MR	HR	CPU hours
AP: RTL	Exhaustive	6.7%	98.9%	9,067
	Sampling-based	2.6%	99.7%	907

- k_p步长的影响
  - RouthSearch算法对步长的敏感度较低，具有极高的鲁棒性

Step Size	Total	Identified	Accurate	MR	HR
1x	374,578	330,727	325,300	13.2%	98.4%
10x	36,584	33,357	32,902	10.1%	98.6%
100x	4,012	3,812	3,673	8.4%	96.4%

论文总结

- 贡献
  - 首次将经典控制理论中的劳斯-赫尔维茨稳定性判据与计算机科学的自适应坐标搜索相结合，提出了一种系统化的PID参数规范推断方法
  - 将三维搜索简化为平面搜索，利用边界连续性进行“沿线追踪”，将计算复杂度从$O(n^3)$彻底降至$O(n^2)$，显著提升了参数推断的效率
  - 平均边界识别率高达92.0%，误配置发现数量是现有方法8.58倍

- 局限性
  - 长周期跟踪模式下验证器对细微的传感器偏差过度敏感，会误判有效的PID配置
  - 无法捕获由“长调节时间”引起的误配置
  - 离线预言机需要记录并分析完整的飞行轨迹，反馈具有滞后性，无法实现即时的在线拦截

特点总结与未来展望

特点总结与未来展望

- 特点总结
  - PGFuzz
    - 首次提出一个新型的测试框架，聚焦于无人机路径规划器中此前未被重视的安全关键漏洞
    - 提出环境风险因子(ERF)指标用以抽象场景的潜在威胁，高效指导关键场景生成
  - RouthSearch
    - 将劳斯-赫尔维茨稳定性判据结合高效坐标搜索技术，大大提高了PID控制参数错误配置发现的效率

- 未来展望
  - 从“人工定义规范”向“大模型自主语义感知”演进，自动化提取安全规范
  - 引入高精度物理引擎，深度模拟真实物理噪声与环境干扰，缩小虚实差距
  - 从后期的“离线检测”转向运行时的“在线即时拦截与自愈修复”

参考文献

[1] Wang Y, Yang C, Zhang X D, et al. DPFuzzer: Discovering Safety Critical Vulnerabilities for Drone Path Planners[C]. 2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE), 2025.

[2] Wang S, Dong Z, Li H, et al. RouthSearch: Inferring PID Parameter Specification for Flight Control Program by Coordinate Search[C]. Proceedings of the International Symposium on Software Testing and Analysis (ISSTA), 2025.

知人者智，自知者明。胜人者有力，自胜者强。知足者富。强行者有志。不失其所者久。死而不亡者，寿。

谢谢！