UAgentOS：真实世界智能体系统架构与双赛题验证学术报告

报告摘要

现有 Agent 框架擅长调用网页、数据库和云端 API，却较少处理真实设备带来的连接状态、传感器有效性、空间来源、物理动作确认和故障恢复问题。本报告提出 UAgentOS 真实设备协调范式：把每台现实设备表示为一个边缘节点，把设备内部可感知或可执行的功能表示为能力，并让观测证据和执行结果持续回到 Agent。Agent 因此不只“发出命令”，还能够知道能力来自哪台设备、当前是否可用、动作是否真正发生。

为验证该架构，本项目完成同一套现实空间管家原型，并从两项赛题的不同视角提交：XiUOS/K210 节点持续采集摄像头画面，在端侧运行 YOLOv2/KPU 并回传检测事件与对应触发帧；Gemini-S1 节点运行 OpenVela Agent，读取板载温湿度、光照和接近数据，接收 XiUOS 视觉事件，再调用 QQ 和语音能力向用户反馈。两项实践共同形成“感知—理解—行动—确认”的闭环。

关键词： 真实世界智能体；边缘节点；设备能力；XiUOS；OpenVela；Gemini-S1；YOLOv2

1. 研究问题：虚拟工具调用不等于现实设备调度

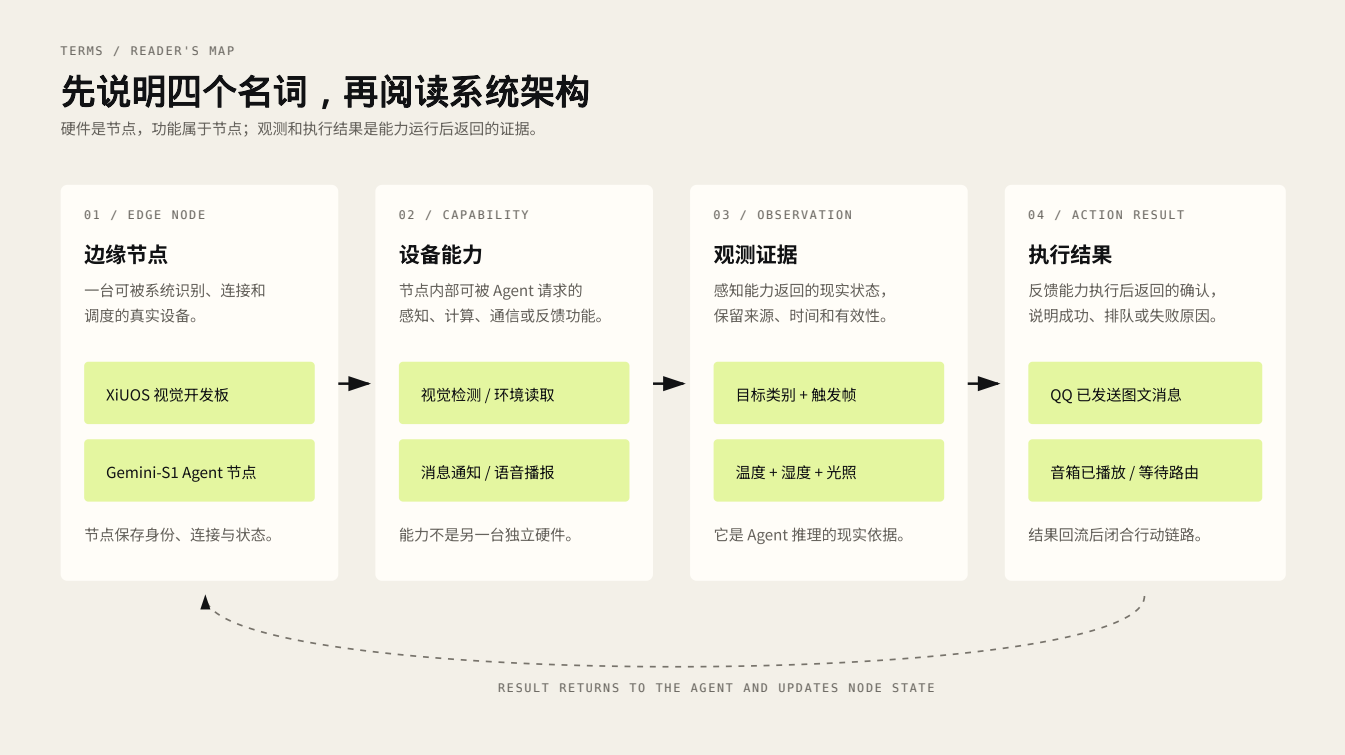
真实设备与软件 API 有三点根本差异。第一，设备具有空间位置、连接方式和持续变化的运行状态；第二，传感器读数可能过期、缺失或来自错误节点；第三，物理动作在“请求发出”之后仍可能因为网络、协议或外设状态而失败。因此，一个面向真实世界的 Agent 不能只保存工具名称，还必须管理节点、能力、证据和执行确认。

本项目研究的问题可以概括为：

> 如何让不同操作系统、不同协议和不同资源规模的真实设备，以可理解、可调度、可验证的方式成为 Agent 的感知与行动面？

UAgentOS 的回答不是重新发明一个唯一 Agent 核心，而是在 Agent 与设备协议之间建立稳定的设备能力层。上层可以是简单 ReAct、OpenVela Agent 或其他推理运行时；下层可以是摄像头开发板、传感器节点、通信端点或音频设备。

2. 概念定义：先区分节点、能力、证据和结果



2.1 边缘节点

边缘节点（Endpoint / Edge Node）是一台可被系统识别、连接和调度的真实设备。节点保存身份、空间位置、连接状态和能力集合。本项目中的 XiUOS 视觉板与 Gemini-S1 都是计算节点；QQ 和音箱则是用户反馈节点。

2.2 设备能力

设备能力（Capability）是节点内部可被 Agent 请求的感知、计算、通信或反馈功能。视觉检测属于 XiUOS 节点；环境读取、设备状态、QQ 通知与音频播报属于 Gemini-S1 节点或其可达反馈端点。能力不是与节点并列的独立硬件。

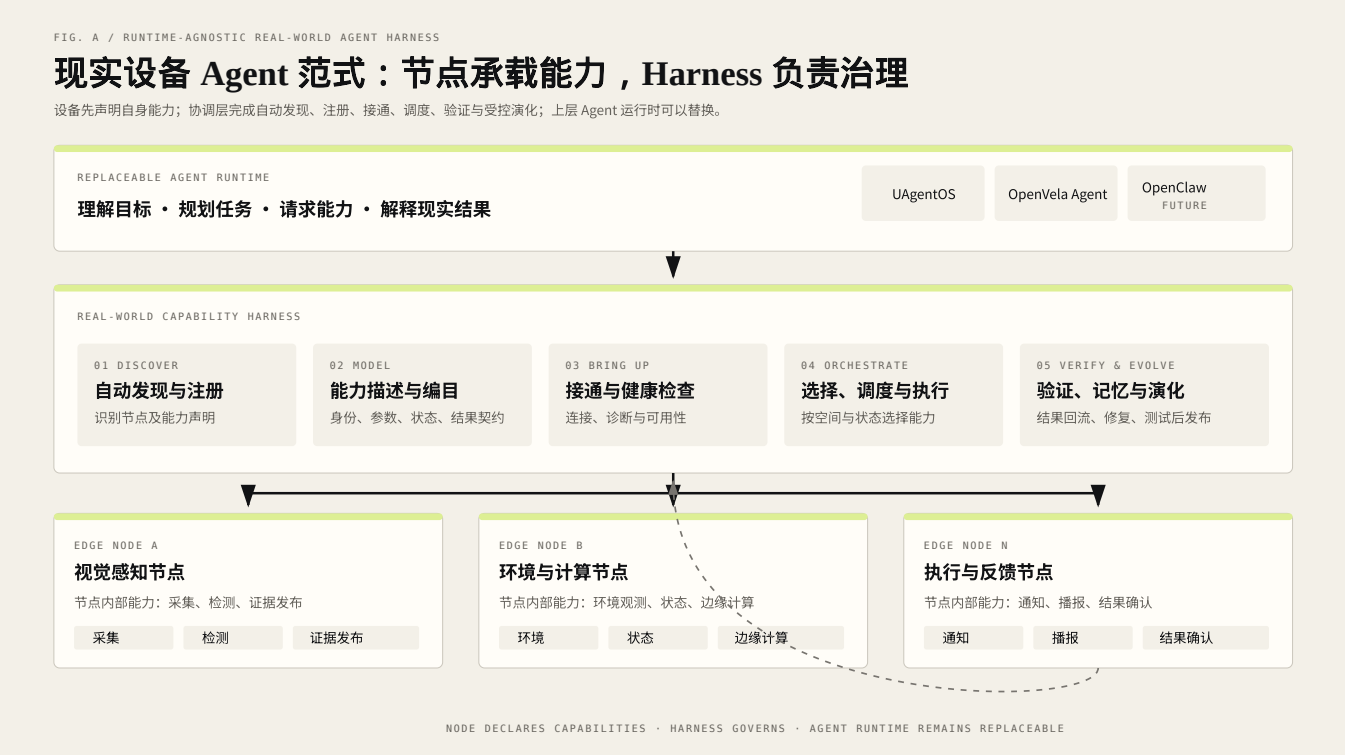
2.3 观测证据与执行结果

观测证据（Observation）是感知能力返回的现实状态，必须携带来源、时间和有效性。执行结果（Action Result）说明动作是否已送达、仍在排队或失败。两者都是 Agent 下一步推理的依据。

2.4 工具与技能

在 OpenVela 中，Tool 是 Agent 调用设备能力的结构化接口；Skill 是用文本描述的多步骤场景和约束。Tool 解决“如何调用”，Skill 解决“何时组合调用”。

3. UAgentOS 系统架构



UAgentOS 将设备接入分为六个连续阶段：

1. **发现与注册：** 确认有哪些设备节点及其身份；
2. **描述与编目：** 说明每个节点具备哪些能力、参数和返回值；
3. **接通与健康：** 验证连接、进程、消息新鲜度和下游服务；
4. **调度与执行：** 根据任务、位置、状态和策略选择合适节点；
5. **验证与记忆：** 将观测、动作和结果组织为可追溯事件；
6. **有界修复：** 链路失效时执行范围明确的重连或服务恢复，并再次验证。

这套生命周期解决了“设备注册了但并不能用”的常见问题。当前实现已经覆盖节点接入、能力注册、策略选择、事件记录、健康检查和有界修复；自动生成新适配器或修改工具代码属于后续受控演化方向，需要隔离测试、权限确认和版本回滚。

上图只描述公开的 UAgentOS 系统模块，不使用内部迭代编号。设备接入层接收节点声明和事件；能力目录把设备功能转成稳定的工具契约；调度层选择节点并执行；证据与健康层记录结果、发现故障并进行有限恢复。

4. 已完成的综合原型

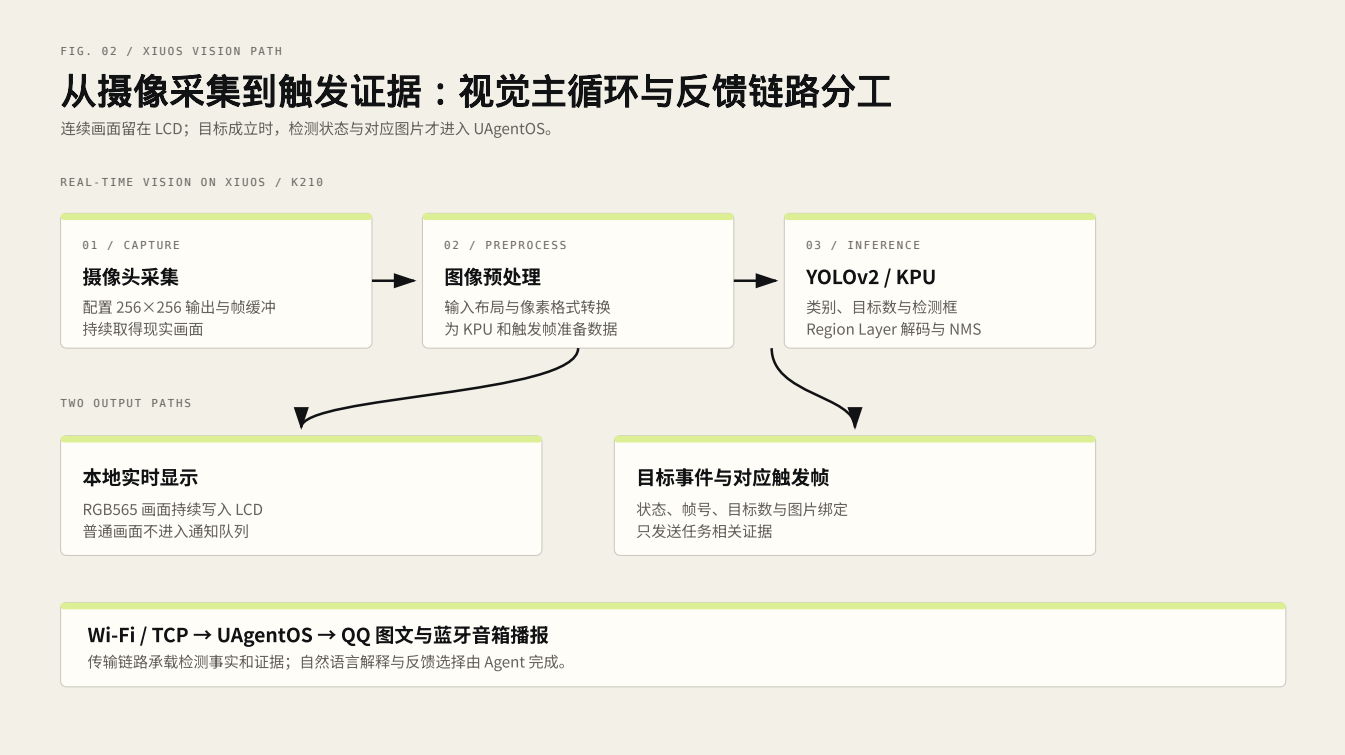
综合原型的现实故事非常明确：XiUOS 摄像头看到目标并在 LCD 出框；同一时刻的检测状态与触发帧进入 Gemini-S1 上的 OpenVela Agent；Agent 可结合板载环境数据理解当前空间状态，并通过 QQ 发送自然语言与图片，通过蓝牙音箱提供现场播报。

系统中每个硬件都有独立职责：XiUOS 保证实时视觉，Gemini-S1 保证推理与工具调用，QQ 和音箱负责不同形式的人机反馈。网关只在设备协议尚未原生支持时做传输转换，不替代 Gemini-S1 上的 Agent 决策。

5. XiUOS 实例：现实观测如何进入 Agent

5.1 对赛题任务的直接回答

XiUOS 赛题任务	已完成实现
摄像头采集与预处理	XiUOS 驱动取流；256×256 输入；LCD 显示；目标裁剪、缩放和灰度证据帧
轻量模型部署与推理	约 2.71 MB YOLOv2 KModel；K210 KPU；Region Layer 与 NMS；head/helmet 状态输出
结果反馈与联动	LCD 画框；独立 Wi-Fi 线程；事件与同帧证据；Gemini-S1 Agent；QQ 图文与音频反馈



5.2 端侧职责

XiUOS 主循环持续完成采集、KPU 推理和 LCD 显示。只有目标成立时才生成网络证据，未触发画面继续本地显示。视觉路径与发送路径分离，避免把外部通知延迟直接施加在摄像头刷新上。

5.3 同帧证据

检测结果与触发帧共享帧号。接收端只有在找到对应证据后才发布视觉事件，避免旧图解释新事件。该设计把模型标签提升为可审计的现实观测。

6. OpenVela 实例：Agent 如何理解并作用于现实

6.1 对初赛要求的直接回答

OpenVela 初赛要求	已完成实现
至少两个自定义工具	环境读取、设备状态、QQ 通知、音频播报、通用能力事件、状态记录共六类工具
自然语言闭环	普通聊天、Gemini-S1 温度读取/播报、综合视觉监控三类演示
可复现与异常处理	官方包补丁脚本、Skill、实机运行路径、传感器双读取路径、结构化错误结果

6.2 Gemini-S1 原生工具

环境工具读取 SHTC3 温湿度与 LTR553 光照/接近传感器；设备状态工具读取运行时间和内存；QQ 工具通过板端 HTTPS 路径发出通知；音频工具返回已播放、已排队或失败状态；通

用能力事件用于接入 XiUOS 等外部节点。工具通过 OpenVela 官方 Registry 进入 `ai_agent`。

6.3 Skill 的作用

Skill 规定 Agent 如何组合工具并避免错误推断：仅有接近数据时不声称“看见了人”；只有视觉节点确认目标时才生成视觉提醒；工具未确认动作成功时，不向用户宣称已经完成。

7. 两项比赛为何是同一个研究命题

XiUOS 比赛从感知侧验证：资源受限 RTOS 能否稳定产生带证据的现实事件。OpenVela 比赛从决策侧验证：嵌入式 Agent 能否理解自然语言、读取真实传感器并调用反馈工具。前者解决“现实如何进入 Agent”，后者解决“Agent 如何作用于现实”。

它们不是两套互不相关的演示，而是同一系统的两个观察角度。XiUOS 材料重点解释摄像头、预处理、YOLOv2/KPU 和结果联动；OpenVela 材料重点解释工具注册、ReAct 闭环、Skill、异常处理与真实开发板适配。

8. 实验结果与可核验证据

证据类别	已获得结果	解释边界
XiUOS 本地视觉	摄像头画面、YOLOv2 检测框、LCD 状态	关键帧清晰度仍可继续优化
视觉事件接入	同帧号事件与证据绑定、结构化视觉事件	无证据时不使用邻近旧图替代
Gemini-S1 Agent	`ai_agent` 运行路径、原生工具与 Skill	实机状态受当前连接环境影响
环境观测	温湿度、光照、接近的真实读取路径	单项不可用时显式返回失败
QQ 通知	自然语言与图片反馈链路	受机器人配置与网络白名单影响
音频反馈	板端语音请求、事件 outbox、网关适配	排队不等于音箱确认播放

9. 竞赛评分映射

9.1 XiUOS

- **技术完整性：** 驱动、采集、预处理、KPU、LCD、网络、Agent 与反馈完整。

- **开源合规性：** 基于 XiUOS 开源工程开发，代码、依赖和隐私配置分离。
- **场景适配性：** 端侧实时推理与事件驱动传输适合资源受限监护场景。
- **创新性：** 两个嵌入式节点协同，视觉证据进入端侧 Agent 的现实上下文。

9.2 OpenVela

- **自定义工具：** 六类结构化工具，超过初赛最低数量。
- **自然语言闭环：** 普通聊天验证基础交互，温度读取/播报与综合视觉监控验证现实工具调用。
- **可复现与异常：** 官方包集成、实机路径、双读取策略、明确失败状态。
- **加分方向：** Skill、多设备协作、真实开发板适配和外部反馈端点。

10. 限制与后续研究

当前原型不是生产级监控系统。XiUOS 端仍需优化触发帧清晰度、网络时延和长时间运行；Gemini-S1 到特定蓝牙音箱的原生 A2DP 兼容性仍在完善；多 LLM 对比、MCP/OpenClaw Node 和空间权限策略尚属后续工作。

下一阶段将进一步研究设备能力的受控演化：Agent 发现某节点缺少适配器或工具时，可以提出增量实现，但必须经过隔离构建、测试、安全策略、人工授权和版本化发布。其目标不是让 Agent 任意修改硬件代码，而是使真实设备能力能够被安全地补全和更新。

11. 结论

UAgentOS 通过 XiUOS 与 OpenVela 两项实践证明：真实设备可以被组织为带节点身份、内部能力、观测证据和执行结果的可治理行动面。XiUOS 负责可靠地产生视觉事实，Gemini-S1 上的 OpenVela Agent 负责理解、调度与反馈。该架构把“调用一个 API”扩展为“协调并验证真实世界中的设备状态变化”，为面向空间的 Agent 系统提供了可复现的工程基础。

参考资料

1. XiUOS 开源操作系统、赛事指定代码仓与开发文档。
2. OpenVela 官方文档、OpenVela Agent 框架与赛事指定代码仓。
3. 本项目提交包中的源代码、构建脚本、运行说明、工具 schema、Skill 与日志证据。