DTS311TC FINAL YEAR PROJECT

Player-Aware Intelligent Monitoring and Operations Navigator

Proposal Report

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Engineering

Student Name	:	Taimingwang Liu
Student ID	:	2037690
Supervisor	:	Xihan Bian

School of AI and Advanced Computing
Xi'an Jiaotong-Liverpool University

November 2025

Abstract

Apply the font of Times New Roman to the paragraphs of the abstract using font size of 12. An abstract is usually one to three paragraphs long with a length of 150 to 350 words.

Contents

1 Introduction **1**
1.1 Problem Setting & Motivation . . . 1
1.2 Scope & Working Definitions . . . 1
1.3 Key Challenges . . . 2
1.4 Our Positioning & Contributions . . . 2
1.5 Design Principles & System Preview . . . 2
1.6 Summary of Findings (Optional) . . . 2

2 Literature Review **3**
2.1 Perception: Modalities & Grounding . . . 3
2.2 Action Interfaces: GUI-only (GCC) vs API/MCP . . . 3
2.3 Agentic Modules: Planning, Memory, Reflection, Skills . . . 4
2.4 Learning Paradigms: Zero-shot, RAG, Finetune, IL/RL, Distillation . . . 4
2.5 Benchmarks & Datasets (OS-like, Games, Desktop) . . . 4
2.6 Evaluation Protocols & Metrics . . . 4
2.7 Deployment & Real-time Considerations . . . 5
2.8 Safety, Permissions & Robustness . . . 5
2.9 Synthesis: Trends, Gaps & Our Niche . . . 5

3 Project Plan **6**
3.1 3.1 Proposed Solution / Methodology . . . 6
3.2 Experimental Design . . . 6
3.3 Expected Results . . . 6
3.4 Progress Analysis and Gantt Chart . . . 6

4 Conclusion **7**

References **8**

Appendix A. Title of Appendix A **I**
A.1 Appendix Heading 1 . . . I
A.2 Appendix Heading 2 . . . I
A.3 Appendix Table and Figure Captions . . . I

Appendix B. Title of Appendix B **I**

1 Introduction

> 说明：交代问题空间（游戏伴随式助手，companion-style game agent）、范围（scope）、研究缺口（gap）、本文立场与贡献（positioning & contributions）。涉及了2.1-2.4和2.6-2.7。

1.1 Problem Setting & Motivation

TODO: 背景：实时游戏场景（real-time gaming），玩家需要事件提示（event spotting）、策略建议（tactical guidance）、低延迟语音交互（voice loop）。动机：现有VLM/VLA在桌面/游戏的落地与稳定性存在鸿沟。

近年来，面向玩家的智能交互快速涌现：从*AI*游戏主播/虚拟角色到*LLM*驱动的*NPC*/插件，社区与产业侧案例表明"大模型+游戏交互"具备显著关注度与潜在影响（game changer potential）。然而，这些案例多为定制工程，缺乏统一接口与可复现实验协议。本文聚焦伴随式（*companion-style*）实时助手，围绕统一动作接口与低延迟体验提供可复现的方法与评测。

Industry/Community Signals 除学术工作外，社区与产业侧的"AI× 游戏/直播"案例为本研究提供了现实动机。例如*Neuro-sama* 及其开源复刻框架[1]–[4]，以及叙事解谜作品*AI2U: With You 'Til The End*[5] 展示了"对话即操作"（dialogue-as-action）与高交互度（LLM-controlled NPCs）的设计可能性。我们据此聚焦于"游戏+ 大模型交互"的可复现路径，但在本文中不将这些案例视为方法有效性的学术证据，而是将研究问题落在统一动作接口（action interface：GUI-only/GCC vs. API/MCP）、低延迟体验（low-latency）与评测协议（evaluation protocols）上。

1.2 Scope & Working Definitions

TODO: 给出工作定义（working definitions）：多模态（multimodal）、动作接口（action interface: GUI-only vs API/MCP），伴随式助手（companion-style），短时托管（autopilot/macros），评测术语（success rate, latency, advice adoption）。

我们将动作接口（action interface）划分为*GUI-only*与*API/MCP*两类。

GUI-only以**General Computer Control (GCC)**为代表：*screen-in, keyboard/mouse-out*的人类同态接口（human-homomorphic interface）。代表作*Cradle*展示了在不依赖应用API的前提下完成长链路桌面/游戏任务的可行性；本文在此基础上采取*GUI-first*并机会主义接入*API/MCP*以提升确定性与效率（determinism & efficiency）。[6]

作为与本文评测设置相关的代表性工作，*Orak* 提供了覆盖12 款真实电子游戏、跨6 大类型的训练与评测基准，并以*Model Context Protocol (MCP)* 实现*plug-and-play*的代理与环境对接；其*Leaderboard/Battle Arena*与*agentic modules*消融，为比较不同模块与输入模态（text/vision）提供了统一框架. [7]我们在本文中参考其"统一评测—模块消融—可复现配置"的思路，结合本场景的实时性需求构建更贴合"伴随式助手

（companion-style）”的评测协议。

作为开放式环境下的系统评测参考，我们采用基于*procedural generation*的统一框架来度量*VLA/VLM*在多步轨迹与OOD设定中的表现，并将架构/数据/输出后处理作为可控变量纳入对比**guruprasad2025benchmarkingvisionlanguage**.

1.3 Key Challenges

TODO: 长链路稳定性（long-horizon stability）、UI变化鲁棒（robustness）、延迟预算（latency budget）、权限安全（permissions/rollback）、跨游戏迁移（generality/portability）。

1.4 Our Positioning & Contributions

TODO: 工程立场：GUI-first + 机会主义API/MCP；引入skills/macros、planning/memory/reflection、语音链路；提出面向伴随式助手的评测协议（advice adoption, voice RTT, macro success）。可列1-3条要点。

1.5 Design Principles & System Preview

TODO: 一句话系统图预告：screen/audio→VLM→LLM/agentic modules→(GUI kb/-mouse | API/MCP)→safety guard（permissions, rollback, kill-switch）。把详图留到方法章节。

1.6 Summary of Findings (Optional)

TODO: 一句话总结文献趋势：从GUI-only通用性到API/MCP确定性，从对话式感知到任务化（taskification）与技能化（skills）。可选，若版面紧张可删。

2 Literature Review

> 说明：采用taxonomy组织而非按时间；每小节末给1-2句“与本文关系”。Cradle放到2.2（GUI-only/GCC）而非开头。涉及了2.1-2.9。

2.1 Perception: Modalities & Grounding

> 说明：输入模态与定位。涉及2.3-2.4。**TODO:** 视觉为主（screen/video）+可选音频（audio）；VLM能力：检测/描述/grounding；提一嘴VLA（如果动作权重内生）与传统VLM+tool的差别。与本文：我们选轻量VLM，优先本地（on-device）与流式ASR/TTS。

2.2 Action Interfaces: GUI-only (GCC) vs API/MCP

> 说明：动作接口对比的核心小节，放Cradle。涉及2.6-2.7。**TODO:** 定义GCC：screen-in, keyboard/mouse-out。代表作：**Cradle**（GUI-only，skill curation/registry, reflection/memory）。优点：通用性（generality）、可迁移性（portability）；缺点：确定性/延迟。API/MCP：确定性高、速度快、但依赖适配。本文策略：GUI-first + API/MCP作为加速通道，GUI兜底。附一句对比表指引。

GUI-only via GCC: *Cradle* 接口范式：提出统一的*General Computer Control (GCC)*设定（*screen-in, keyboard/mouse-out*），避免应用专用API，强调通用性（generality）与可迁移性（portability）。系统机制：规划（planning）→技能整理与注册（skill curation/registry）→自我反思（self-reflection）→记忆（memory）的管线，降低长链路任务的错误累积。作者声称（**claimed novelty**）：首次系统化提出*GCC*作为统一接口，并展示了商业游戏与桌面软件上的长时序任务可行性（长时间连续操作）。优势/局限：*GUI-only*具通用性强，但在确定性（determinism）、延迟（latency）与复杂UI鲁棒性上受限。与本文关系：我们采纳其*skills/macros + reflection + memory*思想，以*GUI-first*保证覆盖面，同时在可用时接入*API/MCP*通道以提速与稳态（保持GUI兜底）。

API/MCP tool-use（简述以对比） API/MCP路径通过结构化接口获得高确定性与较低延迟，但需要适配成本（porting cost）与接口可得性（availability）。本文选择*GUI-first + API/MCP as accelerator*的折中策略。

在动作接口上，*GUI-only* 的*General Computer Control (GCC)* 强调以“screen-in, keyboard/mouse-out”的人类同态通道获取最大通用性（代表作如*Cradle*），而*API/MCP* 路径则以结构化接口换取更强的确定性与更低的延迟（例如MCP 的*plug-and-play*对接与模块化评测）。两者形成“通用性—确定性”的互补光谱：本文采取*GUI-first*以保证覆盖面，并在可用时以*API/MCP*加速关键路径，同时保持GUI 兜底以应对接口缺失与版本漂移。

Takeaway GUI-only（GCC）与API/MCP形成通用性–确定性（generality–determinism）的互补；面向实时伴随式助手（companion-style），折中方案更契合我们的延迟与稳定性目标。

2.3 Agentic Modules: Planning, Memory, Reflection, Skills

> 说明：机制视角。涉及2.1, 2.2, 2.4。**TODO:** 规划（planning）、记忆（memory, 用户偏好/历史）、反思（self-reflection, 纠错/风格一致）、技能库（skills/macros, 原子→复合）。说明这些机制如何提升长链路成功率与体验一致性。与本文：直接采纳skills+reflection+memory组合。

2.4 Learning Paradigms: Zero-shot, RAG, Finetune, IL/RL, Distillation

> 说明：训练与推理范式。涉及2.1, 2.3。**TODO:** 列常见范式及成本/收益：零样本与提示工程、检索增强（RAG for UI schema/FAQ）、轻量微调（LoRA）、模仿/强化（IL/RL）、蒸馏到小模型。与本文：优先零样本+RAG，必要时小规模LoRA以稳UI。

2.5 Benchmarks & Datasets (OS-like, Games, Desktop)

> 说明：基准版图。涉及2.6。**TODO:** 按类型分：桌面/操作系统类（如OSWorld系）、游戏/模拟器类（如ALE等）与自建任务脚本。指出覆盖能力与缺口：缺少"伴随式建议/语音互动"的评测。与本文：定义我们的小型、可复现实验设置与演示脚本。

面向真实游戏场景的评测正在从"单一玩法/小游戏"转向"跨类型、可扩展"的统一框架：*Orak* 以*Model Context Protocol (MCP)*提供*plug-and-play*式对接，使代理（agent）与环境的连接解耦，并在统一配置下考察*planning / reflection / memory / tool-use*等*agentic modules*对性能的边际贡献（ablation）。这类基准不仅有助于横向比较（不同模型/模态），也便于纵向分析（同一模型的模块策略差异），从而把"机制—性能—可复现配置（reproducibility）"串到一起。本文在伴随式场景（companion-style）中借鉴其"统一评测维度+ 模块消融"的体例，但更强调低延迟与语音互动的用户体验指标（例如*advice adoption*与*voice RTT*）。[7]

Placement note（放置说明） 将Orak 置于"Benchmarks & Datasets"主位；在"Action Interfaces: GUI vs. API/MCP"小节中一句话指出其采用*MCP* 的*plug-and-play* 思路以支撑模块化评测。

2.6 Evaluation Protocols & Metrics

> 说明：强烈关联本文贡献。涉及2.2, 2.7。**TODO:** 客观：success rate, time-to-completion, no-misclick/rollback rate, latency（voice RTT, frame→hint时间）；主观：advice adoption, user satisfaction。与本文：将新增advice adoption与macro success作核心指标。

2.7 Deployment & Real-time Considerations

> 说明：工程现实。涉及2.2, 2.6。**TODO:** 本地/云混合、量化（INT4/FP8）、流式解码、语音中断（barge-in）、资源占用与帧率影响。与本文：给出延迟预算（如≤500ms提示、≤ 1.5s语音回路）。

2.8 Safety, Permissions & Robustness

> 说明：安全边界。涉及2.2。**TODO:** 权限模型（whitelist, scope）、操作确认、影子模式（shadow mode）先预测后执行、回滚/急停。与本文：作为系统必要模块。

2.9 Synthesis: Trends, Gaps & Our Niche

> 说明：综述收束到本文位置。关联全篇。**TODO:** 趋势：GUI-only通用→API/MCP混合确定性；机制：从对话到任务化/技能化；缺口：缺少“伴随式建议+语音”的统一评测与低延迟实现。本文niche：针对实时游戏的companion-style助手，提供可复现小型协议与演示。

3 Project Plan

3.1 3.1 Proposed Solution / Methodology

This section contains the methodology, technical design for the project.

3.2 Experimental Design

This section contains the methodology, technical and experimental design for the project.

3.3 Expected Results

This section contains the expected results.

3.4 Progress Analysis and Gantt Chart

This section contains the progress analysis and Gantt chart.

4 Conclusion

References

[1] Vedal and Neuro-sama, *Neuro-sama official youtube channel*, https://www.youtube.com/@Neurosama, Accessed: 2025-10-10, 2022.

[2] O.-L.-V. contributors, *Open-llm-vtuber: An open-source ai vtuber framework*, https://github.com/Open-LLM-VTuber/Open-LLM-VTuber, Accessed: 2025-10-10, 2025.

[3] kimjammer, *Neuro: A local-model recreation of neuro-sama*, https://github.com/kimjammer/Neuro, Accessed: 2025-10-10, 2025.

[4] moeru-ai, *Airi: Ai waifu / virtual character container inspired by neuro-sama*, https://github.com/moeru-ai/airi, Accessed: 2025-10-10, 2025.

[5] AlterStaff, *Ai2u: With you 'til the end*, https://store.steampowered.com/app/2880730/AI2U_With_You_Til_The_End/, Accessed: 2025-10-10, 2025.

[6] W. Tan *et al.*, "Cradle: Empowering foundation agents towards general computer control," *arXiv preprint arXiv:2403.03186*, 2024.

[7] D. Park *et al.*, "Orak: A foundational benchmark for training and evaluating llm agents on diverse video games," 2025, arXiv:2506.03610. arXiv: 2506.03610.

Appendix A. Title of Appendix A

A.1 Appendix Heading 1

Text of the appendix goes here

A.2 Appendix Heading 2

Text of the appendix goes here

A.3 Appendix Table and Figure Captions

In appendices, table and figure caption labels and numbers are typed in manually (e.g., Table A1, Table A2, etc.). These do not get generated into the lists that appear after the Table of Contents.

Appendix B. Title of Appendix B

Text of the appendix goes here if there is only a single heading.