



# 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 Key Challenges . . . . . 2

1.3 Scope and Boundaries . . . . . 3

1.4 Design Principles & System Preview . . . . . 3

1.5 Project Objectives & Expected Deliverables . . . . . 4

1.6 Glossary & Terminology . . . . . 4

References 6

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

## 1.1 Problem Setting & Motivation

The demand for **interactive and companion-like experiences** in real-time gaming environments is rapidly growing, with players seeking more than just simple automation or static overlays[1], [2]. As gaming experiences evolve, the focus is shifting towards AI assistants that offer genuine engagement and companionship, providing players with dynamic, responsive partners who enhance the overall gameplay experience. This trend is being validated by both indie games and major industry players, who are working to create AI partners that do more than just automate tasks—they engage, interact, and form meaningful connections with players.

On the one hand, games like AI2U: “With You ’Til The End” are already capitalizing on this trend, drawing players in with the novelty of engaging with generative AI characters [3]. At the same time, larger companies are investing in foundational technologies like NVIDIA’s ACE (Avatar Cloud Engine) and Ubisoft’s “NEO NPCs”, aiming to create autonomous agents that enhance gameplay rather than just drive conversation [4], [5]. These shifts in the gaming industry are validating the commercial viability of AI-driven experiences, as evidenced by increased player engagement and media interest [6].

The rise of AI-driven virtual streamers, particularly the Neuro-sama phenomenon, highlights a significant shift in both technology and community-driven commercialization. Neuro-sama, an AI-powered virtual streamer (VTuber), engages in real-time conversations and dynamic gameplay, capturing the attention of audiences [7]. Although Neuro-sama remains closed-source, its success has inspired a vibrant open-source ecosystem, with developers aiming to replicate or expand upon its capabilities [8]–[10]. This technological innovation has been accompanied by strong commercial traction on platforms like Twitch, indicating a growing market demand for AI that offers both utility and companionship [11], [12]. Through persistent, low-latency voice loops, these AI systems create engaging player experiences, driving both user engagement and monetization.

This project is timely. While the market demand is clear, the technological feasibility for a **reproducible, non-API-dependent** assistant has only recently emerged. Foundational research is now converging on the key methodologies required for productization. This includes the development of **unified evaluation protocols** and **modular ablation frameworks**, which are essential for ensuring that experiments are reproducible and comparable [13], [14]. Furthermore, recent work has demonstrated the viability of **General Computer Control (GCC)** pathways (i.e., screen-in, keyboard/mouse-out), confirming that capable agents can operate without relying on game-specific APIs [15], [16].

Inspired by this clear convergence of market demand and emerging academic feasibility, this project aims to bridge the gap. The core objective is to define and build a prototype for a

**companion-style assistant.** This assistant is envisioned as a persistent, in-game partner that delivers **event spotting** and **tactical guidance**, leveraging these new, reproducible methodologies to enhance the player’s experience without interrupting gameplay.

## 1.2 Key Challenges

In real player scenarios, the challenge is not just the individual model score, but the overall experience that is **stable, accurate, fast, and controllable**. Below, the challenges and possible solutions are listed based on recent literature:

**Long-Horizon Stability.** When using only GUI (GCC), errors and misalignments can accumulate along the interaction chain, leading to the problem of “drifting off track.” This can be measured with *pass@k* (firstly pass the test at k-th attempt) and *APO* (attempts per opportunity). The combination of “planning + skills (macro) + reflection + memory” can alleviate drift, but it is not a panacea [15].

**Vision-Centric Grounding & Memory.** In purely visual or continuous space setups, localization/tracking/counting, timing control, and long-term visual memory are current model limitations. These can be statistically measured by *OAS* (opportunity-normalized success), broken down by opportunity types such as “pickable items/time points/path nodes” [17].

**Invalid Actions & Think-Action Mismatch.** Converting free text into actions often results in “thinking correctly but clicking the wrong place.” Using the combination of structured output and legal move constraints can reduce invalid predictions, which can be used to evaluate calibration by *Brier MAE*. On the training side, rewards based on four granularities, “format/type-/coordinates/content”, align execution details [16], [18]. At the implementation level, output strategies such as “*semantic to allowed actions*” mapping or *probability modeling over the set of allowed actions* can be used to suppress out-of-bounds and misaligned actions from the start [19].

**Out-of-Distribution & Protocol Consistency.** When the environment or version changes, comparison results become difficult. *Procedural generation* is needed for out-of-distribution (OOD) evaluation, with fixed *post-processing* and prompt scaffolding to control variables, and reporting *macro/micro* metrics under a *unified protocol* and *leaderboard/battle arena* [13], [14], [18].

**Latency & User Experience (UX).** An assistant must “respond instantly.” The key metrics are *RT* (reaction time per opportunity) and *voice RTT* (round-trip voice latency). Engineering strategies to reduce latency include *quantization/pruning/KV caching/streaming decoding* and *on-device/edge-cloud collaboration*, along with single-flight and barge-in strategies [20].

**Safety & Robustness.** High-risk actions must be “confirmable, rollbackable, and traceable.” The approach includes *permission whitelists*, *double confirmation*, *shadow execution*, and *rollback/emergency stop*, with *logging/auditing* to locate *think-action mismatch* [21]–[23].

### 1.3 Scope and Boundaries

To address the issue of inconsistent action interfaces and the challenge of reproducibility caused by reliance on scene-specific APIs, as mentioned in previous sections, this project makes the following clear definition of its technical scope:

- **Unified action interface via GUI (GCC):** This refers to a human-homomorphic channel with screen-in, keyboard/mouse-out.
- **Structured output as the default path:** The default action generation and execution path is defined as **structured output + constrained decoding**.

This scope is defined based on its demonstrated feasibility. Existing studies have shown that, without relying on application-specific APIs, long-horizon tasks can be completed through a pipeline consisting of "planning, macro/skill organization, self-reflection, and memory". Additionally, on real platforms, the "screenshot to structured action" end-to-end navigation path has been proven to be reproducible, and **legal move constraints** significantly reduce invalid actions [15], [16], [18].

**Out-of-scope.** This project does not involve **adapting application/game-specific APIs** on a per-application basis. It does not commit to **large-scale end-to-end training and data collection** in this report. It will not rely on platform-level enhancement privileges (such as A11y/private DOM hooks). **VLA direct-action** will be used only as a benchmark comparison and not as the default path [18].

### 1.4 Design Principles & System Preview

**Design principles.** This project follows four principles: **Unified Input** (ensures portability across applications), **Structured Output** (reduces invalid actions and is easy to audit), **Protocol Consistency** (ensures reproducibility and ablative evaluation), **Low-Coupling Orchestration** (MCP-style, facilitates modular insertion/removal of skills/tools).

**System preview.** The system flow is as follows: **screen/audio** capture, lightweight **VLM** perception, **agentic** (planning/memory/reflection) modules, **MCP-style** skill/tool routing (including OCR/retrieval/computation **tool use**), **GUI execution** (keyboard and mouse), and finally passing through the **safety** module (confirmation/rollback/emergency stop). To reduce end-to-end latency, the deployment strategy will combine the **tool-augmented MLLM** approach with **on-device inference** quantization/caching strategies as key engineering tactics [20], [24].

## 1.5 Project Objectives & Expected Deliverables

### Project Objectives.

- (i) Develop a **GCC-based assistant** real-time prototype that covers event prompting, strategy suggestions, and voice loops.
- (ii) Organize **skills/macros, planning, memory, reflection** using **MCP-style orchestration**, enabling plug-and-play and auditability without relying on application-specific APIs.
- (iii) Define a set of **small and reproducible** evaluation elements (task scripts and metric families), focusing on **pass@k/TTC/Invalid%/macro-micro** and **OAS/RT/APO** experience-related metrics [13], [14].

### Expected Deliverables.

- (a) **System Prototype:** Screen capture and lightweight perception, agentic modules, MCP-style skill bus, GUI executor, and basic safety safeguards.
- (b) **Evaluation Scripts and Configuration:** Reproducible task scripts, metric calculation, and logging/audit tools (including module switches for comparison).
- (c) **User Documentation and Demos:** Installation/operation instructions, configuration templates, and demo videos.

## 1.6 Glossary & Terminology

**TODO:** Add more ...

Since this research area is relatively new, the terminology and naming conventions across different works are not yet unified. Therefore, before entering the literature review, this project aligns key terms and definitions:

**GCC (General Computer Control):** A human-homomorphic action interface defined as screen-in + keyboard/mouse-out; this is the default execution channel in this project [21].

**LAM (Large Action Models):** A family of models where structured actions are treated as first-class outputs; referenced as a comparative paradigm in this project [21].

**VLM vs VLA:** Text/JSON output mapped into action space vs direct action vectors/distributions; evaluation will consistently use legal move mapping + constrained decoding approach [22].

**Scaffold vs Orchestration (MCP-style):** The former refers to the stable interaction "scaffolding" during evaluation, while the latter refers to module/tool registration and routing; both are complementary [23].

**Metric Definitions:** **pass@k**, **TTC**, **Invalid%**, and **macro/micro** will be reported together; opportunity-driven **OAS/RT/APO** serve as core supplements in companion-style scenarios [25].

**Memory–Reasoning–I/O (M-R-I/O):** The internal working division and terminology anchor of this project. Here, planning and reflection align with reasoning, skills/macros represent action output forms on the I/O side, and memory remains independent. The output side will default to a “semantic-to-allowed action” mapping or a compliance strategy that models probabilities over the set of allowed actions [19].



## References

- [1] StreamElements. “State of the stream: Twitch 2024 year in review.” Accessed: 2025-10-11. [Online]. Available: <https://blog.streamelements.com/state-of-the-stream-twitch-2024-year-in-review-ef4d739e9be9>.
- [2] “Q4 2024 global live streaming landscape.” Accessed: 2025-10-11, Streams Charts. [Online]. Available: <https://streamscharts.com/news/q4-2024-global-livestreaming-landscape>.
- [3] AlterStaff, *Ai2u: With you 'til the end*, [https://store.steampowered.com/app/2880730/AI2U\\_With\\_You\\_Til\\_The\\_End/](https://store.steampowered.com/app/2880730/AI2U_With_You_Til_The_End/), Accessed: 2025-10-10, 2025.
- [4] NVIDIA. “Nvidia ace for games - autonomous game characters.” Accessed: 2025-11-05, NVIDIA Developer. [Online]. Available: <https://developer.nvidia.com/ace-for-games>.
- [5] Ubisoft. “How ubisoft’s new generative ai prototype ‘neo npcs’ changes the narrative.” Accessed: 2025-11-05, Ubisoft News. [Online]. Available: <https://news.ubisoft.com/en-gb/article/5qXdxhshJBXoanFZApdG3L/how-ubisofts-new-generative-ai-prototype-changes-the-narrative-for-npcs>.
- [6] J. Kim. “Bringing personality to pixels, inworld levels up game characters using generative ai,” NVIDIA Blog. [Online]. Available: <https://blogs.nvidia.com/blog/generative-ai-npcs/>.
- [7] Vedal and Neuro-sama, *Neuro-sama official youtube channel*, <https://www.youtube.com/@Neurosama>, Accessed: 2025-10-10, 2022.
- [8] 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.
- [9] moeru-ai, *Airi: Ai waifu / virtual character container inspired by neuro-sama*, <https://github.com/moeru-ai/airi>, Accessed: 2025-10-10, 2025.
- [10] kimjammer, *Neuro: A local-model recreation of neuro-sama*, <https://github.com/kimjammer/Neuro>, Accessed: 2025-10-10, 2025.
- [11] “Vedal’s ai vtuber neuro-sama sets new twitch hype train world record.” Accessed: 2025-10-11, Streams Charts. [Online]. Available: <https://streamscharts.com/news/vedals-ai-vtuber-neuro-twitch-hype-train-record>.
- [12] “Vedal987 — streamer overview & stats.” Accessed: 2025-10-11, TwitchTracker. [Online]. Available: <https://twitchtracker.com/vedal987>.
- [13] 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](https://arxiv.org/abs/2506.03610).

- [14] L. Hu *et al.*, “Lmgame-bench: How good are llms at playing games?, 2025a,” *URL* <https://arxiv.org/abs/2505.15146>,
- [15] W. Tan *et al.*, “Cradle: Empowering foundation agents towards general computer control,” *arXiv preprint arXiv:2403.03186*, 2024.
- [16] Z. Gu *et al.*, “Ui-venus technical report: Building high-performance ui agents with rft,” *arXiv preprint arXiv:2508.10833*, 2025.
- [17] X. Zheng *et al.*, “V-mage: A game evaluation framework for assessing vision-centric capabilities in multimodal large language models,” *arXiv preprint arXiv:2504.06148*, 2025.
- [18] P. Guruprasad, Y. Wang, S. Chowdhury, H. Sikka, and P. P. Liang, “Benchmarking vision, language, & action models in procedurally generated, open ended action environments,” *arXiv preprint arXiv:2505.05540*, 2025.
- [19] S. Hu *et al.*, “A survey on large language model-based game agents,” *arXiv preprint arXiv:2404.02039*, 2024.
- [20] J. Xu *et al.*, “On-device language models: A comprehensive review,” *arXiv preprint arXiv:2409.00088*, 2024.
- [21] C. Zhang *et al.*, “Large language model-brained gui agents: A survey,” *arXiv preprint arXiv:2411.18279*, 2024.
- [22] F. Tang *et al.*, “A survey on (m) llm-based gui agents,” *arXiv preprint arXiv:2504.13865*, 2025.
- [23] X. Hu *et al.*, *Os agents: A survey on mllm-based agents for computer, phone and browser use*, 2024.
- [24] W. An, J. Nie, Y. Wu, F. Tian, S. Lu, and Q. Zheng, “Empowering multimodal llms with external tools: A comprehensive survey,” *arXiv preprint arXiv:2508.10955*, 2025.
- [25] Z. Durante *et al.*, “Agent ai: Surveying the horizons of multimodal interaction,” *arXiv preprint arXiv:2401.03568*, 2024.

## **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.