Simulated Profiling Environment for Embodied Intelligence (SPEEN)

Darroll Saddi[1] **Ken Lin**[1] **Ryan Li**[1] **Matthew Fulde**[2] **Jon Lagasca**[1]

University of California, Davis
[1]Computer Science [2]Computer Science and Engineering

{dwsaddi, kemlin, ryjli, mpfulde, jonlagasca}@ucdavis.edu

Abstract

The Simulated Profiling Environment for Embodied Intelligence (SPEEN) is an open-source platform for evaluating embodied Large Language Model agents in a simulated game environment. As LLMs are increasingly integrated into robotics and embodied systems, SPEEN addresses the need for standardized evaluation frameworks by providing a well-documented and modifiable system for utilizing and benchmarking embodied LLMs using the game development engine, Godot. We implement a proof-of-concept Minecraft-like environment that utilizes our systems to demonstrate structured quantitative benchmarking through diverse scenarios as well as an open-world sandbox for more qualitative assessment of decision-making behaviors over extended periods. Our approach enables researchers and enthusiasts to test the implementation of embodied LLMs in a simulated environment,

1 Problem Identification

1.1 Background

This project originated as a Senior Design Project at UC Davis, where our team collaborated with Justin Jia (affiliated with Apple) to address practical needs in AI testing. The overaching goal of the project at this stage was to develop a sandbox environment specifically for testing AI programs. Initial discussions led us to focus on Large Language Models (LLMs) as a primary technology of interest.

1.2 Exploratory Research

1.2.1 Definitions

We define agentic AI as systems capable of autonomous decision-making and environmental interaction. Embodied AI, our primary focus, represents a subset specifically concerned with agents that interact with physical or simulated physical worlds. In the following sections, we justify our choice of LLMs as the core technology for our agents.

1.2.2 Focusing on Large Language Models

Our research identified significant gaps in environmental design for evaluating advanced AI systems. We examined existing benchmarking platforms, particularly NeuralMMO, which provides an open-source environment for measuring **reinforcement learning** algorithm performance. NeuralMMO is a game environment with high discrete input complexity, requiring agents to navigate a 2D grid world

with tasks including resource management, exploration, and agent interaction. One crucial insight from Joseph Suarez, NeuralMMO's creator, significantly influenced our approach:

> "It is very easy to create an interesting looking simulator. It is very hard, under the constraints of making useful AI research [to create an environment meant for testing and training AI]. . . it is not just a game, it is an AI simulation."**[TODO]**cite his thesis defense

However, despite continuous added complexity to the environment over the past few years, we observed that advanced algorithms like Proximal Policy Optimization (PPO), given sufficient compute, could effectively solve most presented tasks.

This observation shifted our focus from reinforcement learning toward Large Language Models, which represent more recent advancements in artificial intelligence. We were particularly interested in the growing integration of LLMs, which have proven highly capable in natural language understanding and code generation, into robotics and embodied systems.

1.2.3 Environment Design

Several proposed embodied LLM architectures use Minecraft as their testing environment. Projects including NVIDIA's Voyager project (extending from the MineDojo project) and Project Sid by Altera attempt to provide prompting and training architectures for agents to exist in Minecraft. However, there are identifiable problems between these projects:

- **Lack of Open-Source Availability:** Project Sid articulates a way for many-agents to exist within Minecraft, proposing supposedly powerful prompting architectures, but is not open-source. It is further unclear how to reproduce the results from their methodology, representing potential conflict of interest
- **Limited Flexibility:** While MiniDojo projects like NVIDIA's Voyager have yielded impressive results and are open-source, we believe there is an increasing need to develop benchmarks that generalize to real-world use cases and to metrics specific to agentic AI rather than Minecraft. The domain-specific metrics of area explored & percentage of available items collected are not necessarily useful for evaluating LLMs in a more general sense.

Additional barriers exist in the choice of environment. While Minecraft offers inherent complexity and extensive documentation that may aid in implementation of the agents, it presents significant limitations as a benchmark:

1. It requires a commercial license, creating accessibility barriers
2. No existing standardized context provisioning or testing metrics

Although environments that more closely emulate the real world would better map to real-world robotic applications, we determined that focusing on standardizing an accessible contextualization and prompting systems would be more beneficial. Thus, our focus was on the development of the prompting architecture, e.g. the interaction between the LLM and the game environment, and the collection of evaluation metrics, as opposed to purely on the environment itself. With this in mind, our environment is kept less realistic and still similar to Minecraft, but we hope that future work will be able to build upon our work and expand to more realistic environments.

[TODO]Mention Google Deepmind Creating Multimodal Interactive Agents with Imitation and Self-Supervised Learning

1.2.4 Open-Source Requirements

Our research identified that many successful LLM evaluation solutions are not open-source, limiting their utility for broader research purposes. By building our system to allow for transparent and reproducible integration of new LLMs and prompting architectures, we aim to address this limitation. This constraint guided our decision to use the Godot game engine, which is open-source and allows for easy modification and expansion, and we believe is heading in a positive directions for potential use in environment design and research.

1.2.5 Research Use Case

Since the system is intended for benchmarking, we wanted to ensure the backend Python websocket, which connects to the game environment, supported both cloud-hosted and locally-hosted LLMs. This flexibility allows researchers and enthusiasts to test the performance of various LLMs, especially as new models are released. Using abstraction layers, new models can be integrated into the system without requiring significant changes to the codebase on the development side. For higher-level access, a configuration file would be used to streamline the selection of models and their parameters, including locally-hosted LLMs.

In terms of measurement and validation strategy, we will implement a series of scenarios that test the capabilities both high and low-performing LLMs, and tracking success rate (adherence to the goal of the scenario). These scenarios will serve as our metrics capture meaningful dimensions of embodied intelligence like planning, adaptability, and environmental reasoning.

In the current product, we support the OpenAI API for GPT models, Google Gemini API, and have tested the integration of locally hosted LLMs using Ollama such as Gemma3, Deepseek-r1, and other available models. These are also largely the models we used for testing our implementation of the prompting architecture, agentic pipeline, and evaluation metrics.

1.3 Design Requirements

Based on our research findings, we had the sufficient justification to develop SPEEN as an open-source benchmarking environment specifically for evaluating embodied AI (LLM) performance. Listed below were our comprehensive design goals and requirements for the system:

1. **Standardized Evaluation Framework:**
    - Implement quantitative metrics and qualitative assessment framework for embodied LLM agents in Godot.
    - Provide standardized system for prompting and game state context provisioning.
    - Streamline integration of any LLM, prompting architectures, or evaluation methods.
2. **LLM Integration Capabilities:**
    - Support both cloud-hosted and locally-hosted LLM connections for research/evaluation purposes.
    - Provide abstraction layers for integration of LLMs when cloud hosting.
    - Include high-level configuration options for connection parameters and API settings.
3. **Open-Source Compliance:**
    - As a research tool, ensure the system is accessible and modifiable for researchers and enthusiasts.
    - Ensure appropriate open-source licensing.
    - Provide comprehensive documentation of system architecture and project structure.

These are the requirements of the system we plan to achieve. To prove its usability, we shall also implement the testing environment and prompting architecture as a proof of concept.

1. **Accessible Game Environment:**
    - Develop a Minecraft-like 3D environment using the open-source Godot engine.
    - Create a world with sufficient complexity for testing embodied AI behaviors.
    - Design intuitive interfaces for human researchers to monitor and evaluate agent performance.
2. **Structured Scenarios:**
    - Implement diverse testing scenarios targeting specific embodied capabilities.
    - Create both directed tasks and open-ended challenges.
    - Support automated performance scoring where appropriate.
3. **Prompting Architecture:**

- Implement a prompting architecture emulating chain-of-thought and goal-based prompting systems.
- Allowing for easy modification and expansion of the prompting system.

2 Development

2.1 Environment Design

A majority of early development was spent on agent's environment for the qualitative evaluation use case. Features included:

- Procedural world generation with day/night cycle (qualitative)
- Navigation and A* pathfinding via Godot's Navigation3D system
- Suite of API functions for the agent to call in order to interact with the environment
- NPC AI to act as objectives and adversaries for benchmarking (animals, zombies)
- An item and inventory system to allow for item collection and manipulation

At the end of the first 10 weeks, we had a functional environment that was sufficiently complex for our proof of concept. This led directly into the development of the prompting architecture and the agent-to-environment pipeline.

2.2 Agent-Environment Pipeline

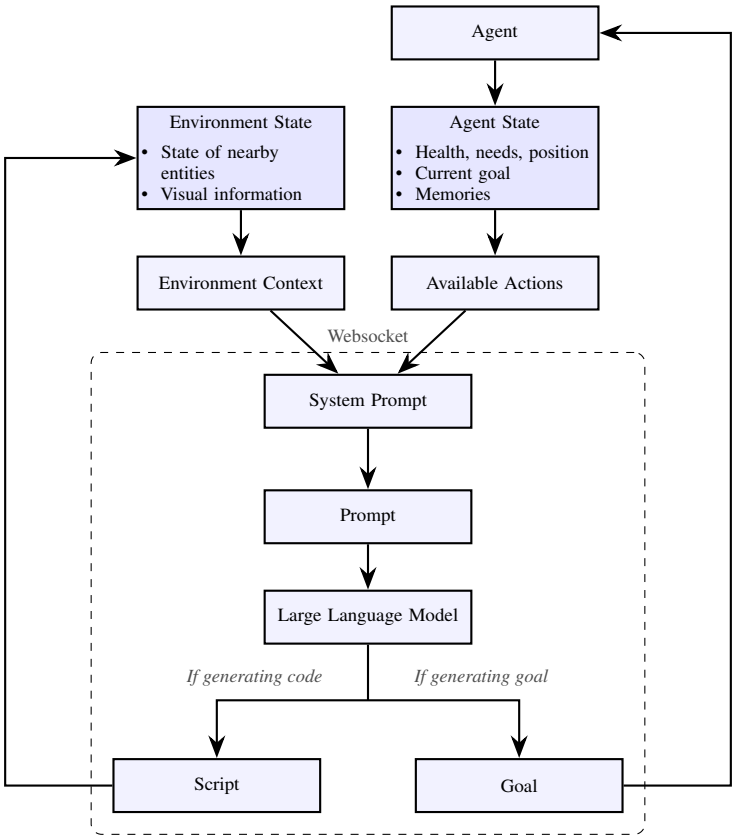

Figure 1: Agent-Environment Pipeline Architecture

Figure 1, depicts our agent-environment pipeline architecture, whic also serves as our prompting architecture. Environmental information is fed directly into the LLM, which generates either a script or a goal if no goal is set for the agent.

If the LLM generated a script, the agent will execute the script, the outcome of which may be logged in the "Memories" of the agent. This allows for chain-of-thought reasoning, a common technique for emulating LLM decision making.

The available actions are the function calls available to the agent, which abstract away the low-level interaction logic of the environment. This technique focuses the LLM on reasoning and decision making rather than on code adherence, although it is possible for broken code to be generated. In this event, we catch broken code and are able to either reprompt with the error for self-fixing behavior, or alternatively count the iteration as a failure.

Once the script is ready to be ran, the agent's "body" will execute the input actions from the LLM. With our pipelines complete, we moved on to designing scenarios that will measure the LLMs' reasoning and correctness.

3 Testing and Evaluation

3.1 Benchmarking Scenarios

We implemented a variety of scenarios to test different aspects of agent capabilities:

1. Basic navigation: Testing the agent's ability to navigate to specific locations
2. Item manipulation: Testing picking up and giving items
3. Entity Interaction: Testing communication and combat with other entities
4. Sequential Tasks: Testing the agent's ability to perform multi-step tasks

Each scenario includes success criteria and automatic performance tracking.

3.2 Evaluation Metric

The system tracks several key metrics:

- Success Rate: Percentage of tasks completed successfully
- Failure Rate: Percentage of tasks that failed to complete
  - Counting errors in code compilation and logical errors as failures

4 Connecting to Sandia Mission Areas and NAE Grand Challenges

4.1 Relevance to Sandia Mission Areas

Our project aligns with several Sandia National Laboratories' mission areas:

1. Computing and Information Sciences: SPEEN provides a platform for evaluating and improving computational intelligence systems, directly supporting Sandia's work in advanced computing and software engineering. Our system enables systematic testing of AI capabilities.
2. National Security Technologies: By developing evaluation frameworks for embodied AI, we contribute to technologies that could enhance decision-making systems relevant to nation security applications. Autonomous AI systems require rigorous evaluation before deployment, which SPEEN facilitates
3. Engineering Sciences: The design and implementation of our evaluation environment draws on principles from systems engineering, human-machine interfaces, and simulation technology, all core engineering areas within Sandia's scope of research

4.2 Connection to NAE Grand Challenges

Our work addresses aspects of multiple National Academy of Engineering Grand Challenges

1. Engineer the Tools of Scientific Discovery: SPEEN serves as a tool for scientific discovery in the field of artificial intelligence, allowing researchers to systematically evaluate and improve embodied AI systems. This directly supports NAE's challenge to develop better tools for scientific understanding
2. Enhanced Virtual Reality: Our platform advances techniques for creating interactive, responsive virtual environments with intelligent agents. These environments can simulate and test what AI systems that could operate in the physical world.
3. Advanced AI for Improved Quality of Life: By developing better evaluation methods for embodied AI, we contribute to the broader goal of creating AI systems that can effectively assist humans in daily life and critical operations. Reliable AI requires thorough testing like those we've developed.

5 Conclusion and future developments

LLMs are a relatively new technology, meaning there are not many tools for measuring LLMs' competency in decision-making and environment interaction. There are projects such as Mindcraft and Voyager that do integrate LLMs into a virtual world but lack benchmarking features for researchers to record their LLMs' performance and are built for Minecraft itself, requiring researchers to obtain paid licenses for Minecraft. Our project aims to be the solution to the problems of benchmarking tools and accessibility.