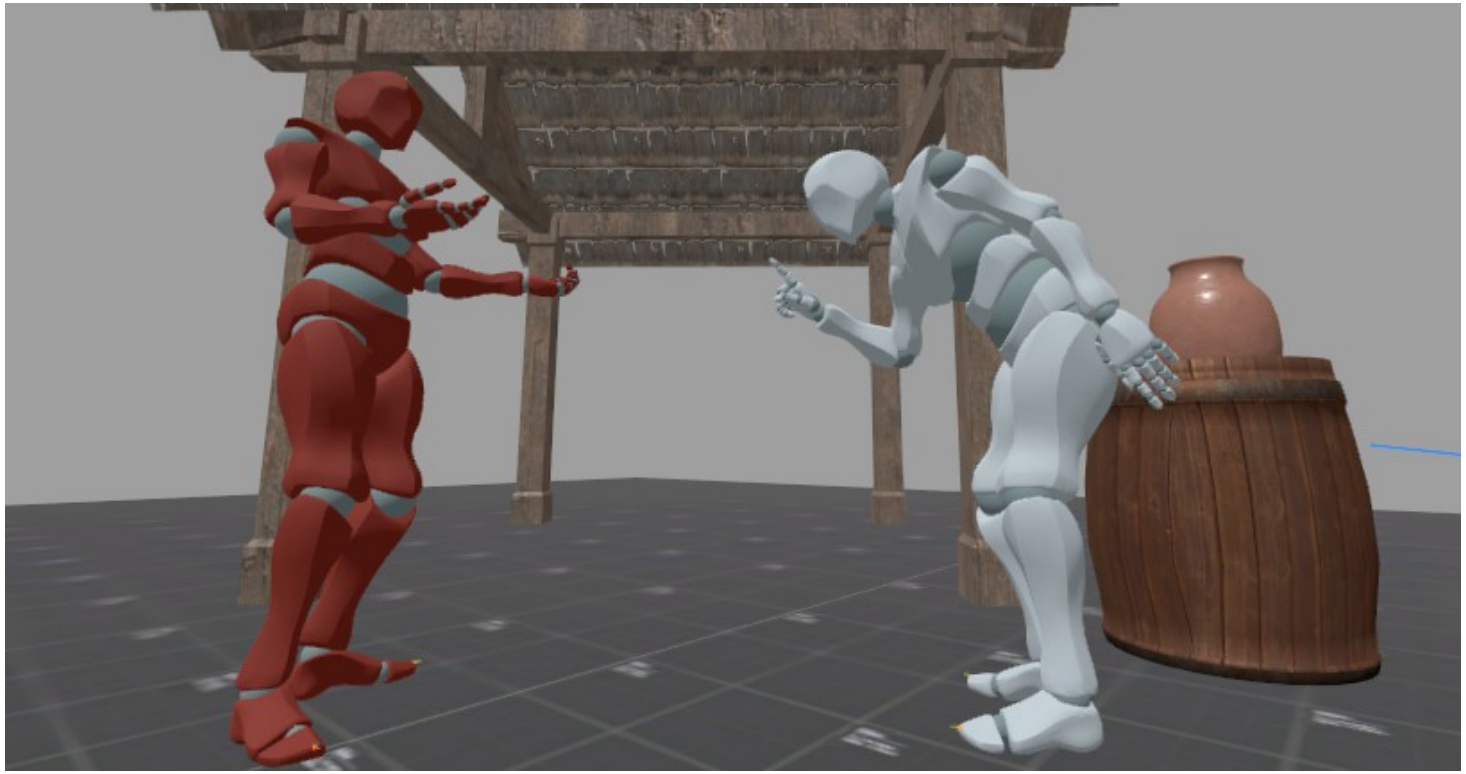
01 Інтелектуальний когнітивний агент NPC у тривимірному ігровому просторі із застосуванням LLM

Магістерська кваліфікаційна робота

Студент: Овсюк В.М.
Керівник: Ільїн О.О.

02 Об'єкт дослідження: NPC

NPC (Non-Player Character) - це персонаж віртуального середовища, керування яким здійснює не гравець, а комп'ютерна програма.



03 Актуальність теми

Еволюція віртуальних світів

Ігрова індустрія досягла фотореалізму, але інтелект персонажів залишився на рівні скриптів 2000-х років. Це створює дисонанс: світ виглядає живим, але поводиться як декорація.

Технологічний прорив

Поява Великих Мовних Моделей (LLM) дозволяє генерувати поведінку в реальному часі. Проблема полягає у відсутності архітектурних рішень для поєднання повільних, текстових LLM зі швидким, фізичним 3D-середовищем.

04 Проблематика: Розрив контекстів

«МОЗОК» (ШІ)

- Оперує текстом.
- Не має поняття часу.
- Не знає про фізику та простір.
- Велика затримка.

ЯК ЇХ З'ЄДНАТИ?

«ТІЛО» (Game Engine)

- Оперує векторами.
- Працює в реальному часі.
- Вимагає чітких команд.
- Синхронне виконання.

Тема диплому - побудова цього мосту (архітектури), щоб думки ШІ перетворювалися на дії.

05 Мета роботи

Розробити програмну архітектуру автономного агента,
здатного до адаптивного планування дій
шляхом інтеграції LLM у 3D-середовище Godot.

06 Етапи дослідження

1

1. Дослідження та Аналіз

Обґрунтування вибору технологій та архітектурних патернів.

2

2. Проєктування та Розробка

Створення архітектури «Мозок–Тіло» та реалізація компонентів системи.

3

3. Експерименти та Валідація

Перевірка адаптивності агента на складних сценаріях.

07 Архітектурні принципи системи

Для побудови системи було обрано наступні підходи:

Модульність

Чітке розділення на компоненти:
- Сенсори (Vision)
- Логіка (Brain)
- Виконавці (Body)

Асинхронність

Використання HTTPRequest для неблокуючих запитів до API.
Це дозволяє рушію працювати плавно, поки ШІ думає.

Подієва модель

Використання патерну Publisher-Subscriber (EventBus).
Компоненти спілкуються через сигнали, не знаючи один про одного.

08 Обґрунтування вибору технологій

Godot Engine 4.5 (GDScript)

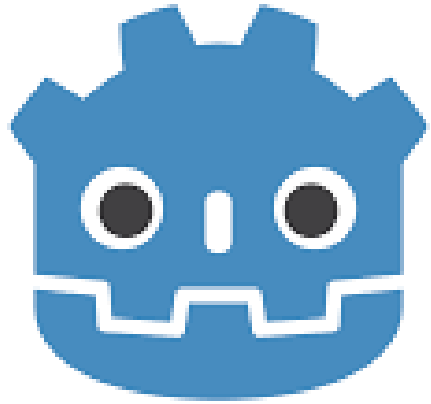
Легковагий рушій з гнучкою системою вузлів. GDScript має нативну підтримку JSON та HTTP.

Формат JSON

Єдиний протокол обміну даними. Забезпечує сувору структуру команд для ШІ.

Метод ReAct

Reasoning + Acting. Дозволяє моделі не просто говорити, а виконувати дії.



GODOT
Game engine

09 Аналіз LLM API: Швидкість або Інтелект

Було проаналізовано два ключові провайдери для різних задач:

1. Groq API (LPU)

- Технологія: Language Processing Unit.
- Перевага: Наднизька затримка (< 0.5 с).
- Роль: Реактивні діалоги та швидкі дії.

2. Google Gemini API

- Технологія: Мультимодальні моделі.
- Перевага: Глибокий контекст та логіка.
- Роль: Стратегічне планування та аналіз помилок.

10 Вибір конкретних моделей

GPT-OSS-120b

Легка модель. Ідеальна для швидкої генерації.

Pro 2.5

Флагманська модель. Використовується для побудови складних планів.

Використано гібридний підхід. Система має можливість динамічно обирати модель залежно від складності задачі.

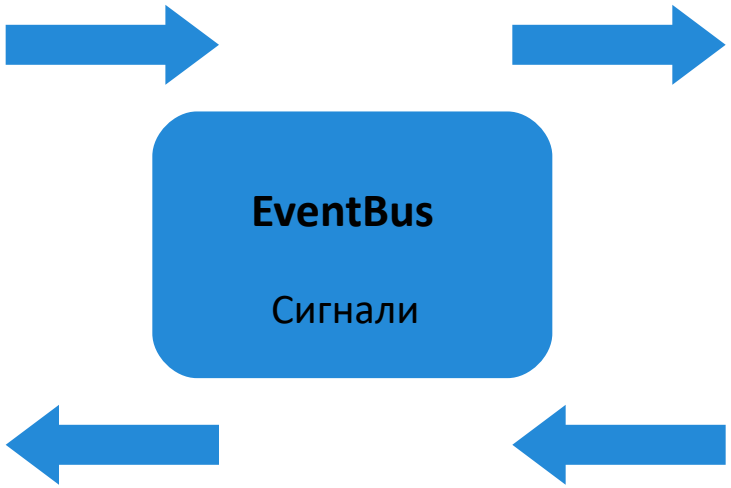
11 Схема взаємодії «Мозок–Тіло»

Як відбувається розподіл відповідальності:

МОЗОК (Logic)

AI_Component.gd

1. Отримує контекст.
2. Формує промпт.
3. Парсить JSON.
4. Обробляє помилки.



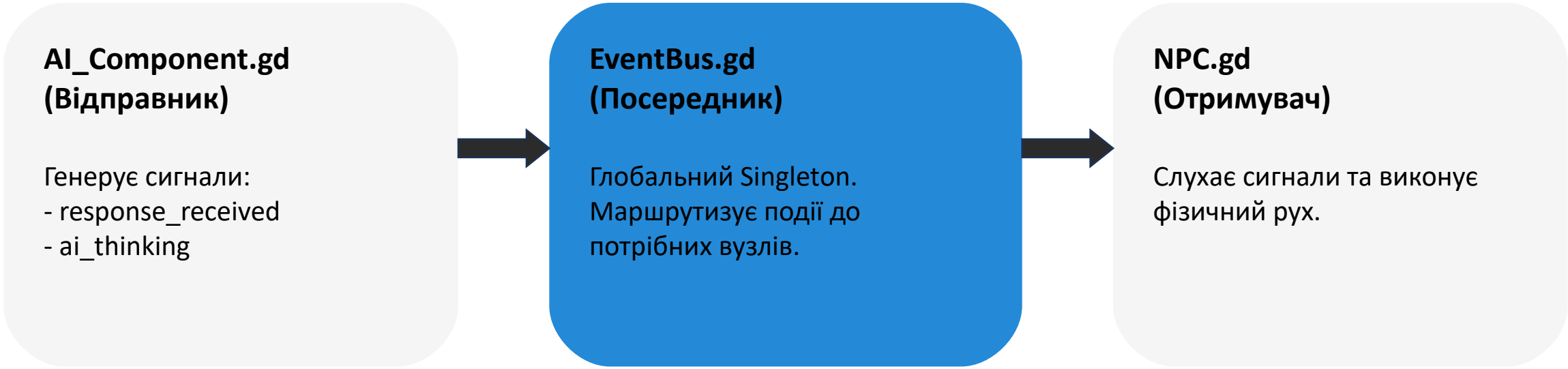
ТІЛО (Physics)

NPC.gd + VisionArea

1. Сканує простір (Vision).
2. Розраховує шлях (NavMesh).
3. Програє анімації.
4. Звітує про завершення.

12 Реалізація зв'язку: Глобальна шина

Як «Мозок» керує «Тілом», не знаючи про нього прямо?



13 Як уникнути переривання?

Конфлікт

ШІ думає 1-3 секунди. Ігровий рушій вимагає оновлення 60 разів на секунду. Синхронний запит заморозив би поток.

Рішення (Async)

Використання HTTPRequest у фоновому потоці. Поки приходить JSON, агент наприклад програє анімацію роздумів.

14 Парадигма ReAct: Як мислить агент?

ReAct (Reasoning + Acting) - це метод, що дозволяє LLM поєднувати логічні роздуми з виконанням дій.

Reasoning (Мислення)

- Аналіз поточної ситуації.
- Розбиття складної задачі на підзадачі.
- Формулювання стратегії.

Acting (Дія)

- Вибір конкретного інструменту.
- Формування параметрів для функції.
- Взаємодія із зовнішнім світом.

15 Процес генерації плану

Від абстрактного бажання до конкретного JSON.

1 **Вхідні дані**

Контекст: «Я бачу зачинені двері. У мене є ключ».

2 **Ланцюжок думок**

«Щоб відкрити двері, мені треба використати ключ».

3 **Формалізація**

JSON: [{tool: interact, args: {obj: door, action: unlock}}]

Агент самостійно будує цей ланцюжок, без жорсткого коду.

16 Інструментарій агента (Tools API)

«Тіло» надає «Мозку» 13 базових функцій для взаємодії зі світом:

1. Навігація та Зір

- go_to_target(id)
- go_to_position(x, y, z)
- follow_player(id)
- stop_following()
- search_environment()
- describe_object(id)

2. Фізична Взаємодія

- take_item(id)
- interact_with_object(id, action)
- list_inventory()

3. Соціум та Навчання

- speak_to_nearby_agents(msg)
- propose_trade(...)
- request_item_from_npc(...)
- learn_action_template(...)

17 Мова спілкування (JSON Protocol)

Приклад реальної команди, яку генерує «Мозок»:

```
{
  "thought": "Скриня закрита, мені потрібен ключ.",
  "plan": [
    { "tool_call": "search_environment", "args": {} },
    { "tool_call": "go_to_target", "args": { "object_name":
"Key_Chest" } },
    { "tool_call": "take_item", "args": { "object_name":
"Key_Chest" } }
  ]
}
```

Система парсить цей JSON і викликає відповідні функції Godot.

18 Ключова перевага: Адаптивність

Що відбувається, коли план провалюється?

1. Дія

Агент намагається взяти ключ.
Виклик: take_item()

2. Помилка

Фізичний рушій повертає:
{status: error,
reason: too_far}

3. Адаптація

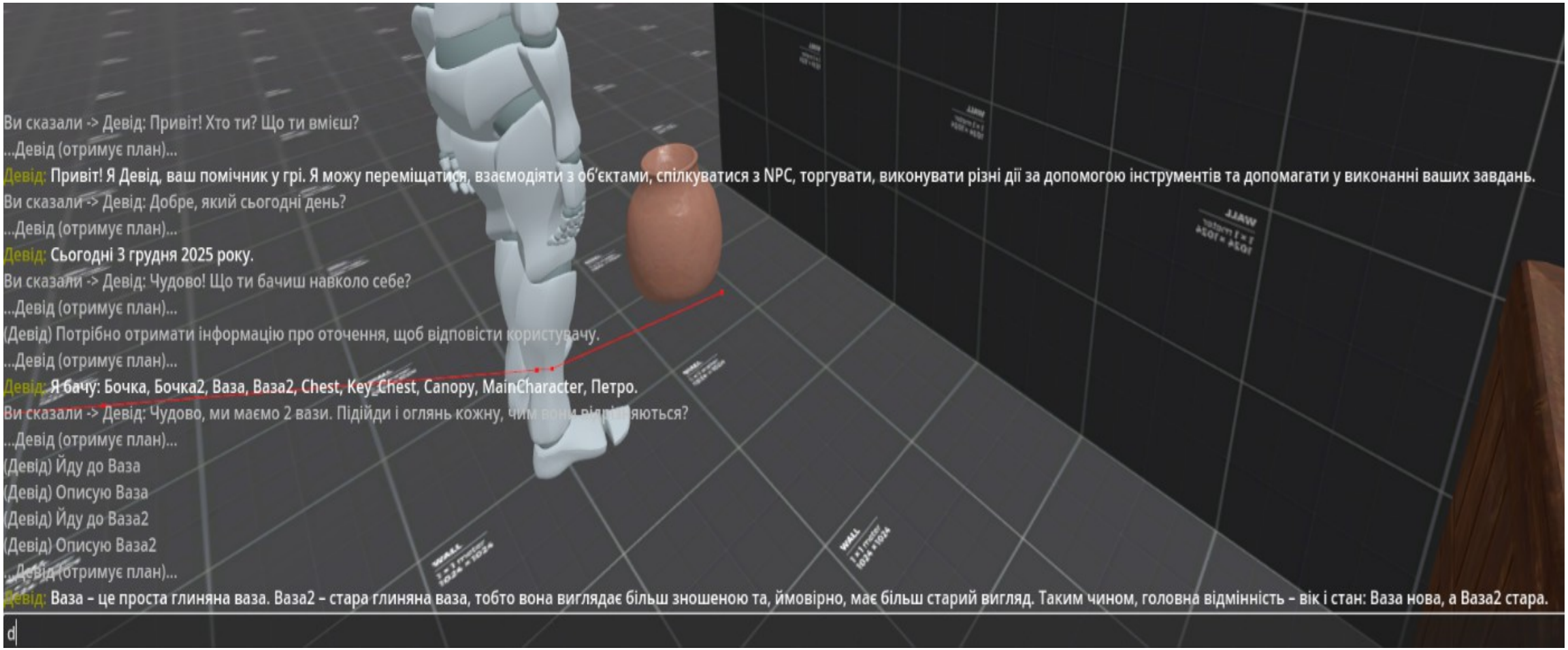
«Мозок» отримує помилку і створює новий план:
[go_to -> take]

19 Результати експериментального тестування

Перевірка відповідності розробленої системи поставленим вимогам:

Сценарій	Завдання	Фактичний результат	Оцінка
1. Базовий	Аналіз оточення	Точна ідентифікація об'єктів.	Успіх
2. Середній	Навігація	Успішне розпізнавання та рух до цілі з обходом перешкод.	Успіх
3. Високий	Адаптація	Автоматичне виправлення помилок.	Успіх
4. Експертний	Дедукція	Самостійний пошук ключа без підказки.	Успіх

20 Приклад сценарій №1



21 Приклад сценарій №2

22 Когнітивний агент або традиційний NPC?

Критерій	Традиційний NPC	Когнітивний Агент
Реакція на невідоме	Відсутня (ігнорує або ламається).	Адаптивна (шукає логічний вихід).
Масштабованість	Низька (потрібен новий код).	Висока (універсальні інструменти).
Вирішення проблем	Тільки за інструкцією.	Емерджентна (спонтанна дедукція).

23 Перспектива: Sim-to-Real Transfer

Розроблена архітектура дозволяє переносити інтелект з віртуального середовища на фізичні носії без зміни когнітивного ядра.

Godot

- Віртуальне тіло NPC.
- Симуляція фізики.
- Безпечне навчання без ризиків.

Physical Deployment

- Фізичне тіло (Android).
- ROS (Robot Operating System).
- Керування сервоприводами.

24 Загальні висновки

- 1. Спроєктовано архітектуру «Мозок–Тіло», що інтегрує ШІ у фізичний простір.
- 2. Реалізовано механізм ReAct для автономного планування та виправлення помилок.
- 3. Експериментально доведено здатність агента до вирішення нестандартних задач.

ДЯКУЮ ЗА УВАГУ!

Готовий відповісти на ваші запитання