|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_***Робототехника и комплексная автоматизация***\_\_\_\_\_\_

КАФЕДРА \_***Системы автоматизированного проектирования (РК-6)***\_

**ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

Студент\_Больных Андрей Сергеевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*фамилия, имя, отчество*

Группа\_\_***РК6-84Б***\_\_\_

Тип практики \_\_\_\_\_\_\_\_\_\_\_***Преддипломная***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Название предприятия**\_\_\_\_*Компания «ИФК Тренинг»*\_\_\_**

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_*А.С. Больных*\_\_\_\_\_**

*подпись, дата фамилия, и.о.*

Руководитель практики

от кафедры **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_ *Д.И. Оглоблин*\_\_\_\_**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***2024 г.***

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой *РК6*

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_ *А.П. Карпенко* \_

« \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

**ЗАДАНИЕ**

**на прохождение производственной практики**

**\_\_\_\_\_\_*Преддипломная*\_\_\_\_\_\_**

Тип практики

Студент

\_\_\_Больных Андрей Сергеевич\_\_\_\_\_\_\_\_\_\_\_\_ \_\_***4***\_\_ курса группы \_***РК6-84Б***\_

Фамилия Имя Отчество № курса индекс группы

в период с \_***13 мая 2024***\_\_\_ г. по ***26 мая 2024*** г.

*Предприятие:* **\_\_\_\_\_\_\_\_\_\_\_*Компания «ИФК Тренинг»*\_\_\_\_\_\_\_\_\_\_**

*Подразделение:* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(отдел/сектор/цех)

*Руководитель практики от предприятия (наставник):*

Витюков Фёдор Андреевич, преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия Имя Отчество полностью, должность)

*Руководитель практики от кафедры:*

Оглоблин Дмитрий Игоревич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия Имя Отчество полностью, должность)

*Задание:*

*Модифицировать систему общения LLM модели с виртуальными агентами в Unreal Engine 5. Провести программную реализацию взаимодействия генеративных виртуальных агентов с внешним миром. Создать интерфейс, показывающий лог событий и данные о выбранных объектах, для конечного пользователя.*

Дата выдачи задания ***14 мая 2024***г.

Руководитель практики от предприятия  **\_\_\_\_\_\_\_\_\_\_ /\_\_*Ф.А. Витюков*\_\_/**

Руководитель практики от кафедры  **\_\_\_\_\_\_\_\_\_\_ /\_\_ *Д.И. Оглоблин*\_/**

Студент **\_\_\_\_\_\_\_\_\_\_ /\_\_ *А.С. Больных\_*\_/**

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc167366275)

[1. ПОСТАНОВКА ЗАДАЧИ 4](#_Toc167366276)

[2. КРАТКИЙ ОТЧЕТ О ВЫПОЛНЕННЫХ РАБОТАХ 5](#_Toc167366277)

[2.1. Разработка виртуального агента «Зебра» 5](#_Toc167366278)

[2.2. Разработка интерфейса виртуальных агентов. 6](#_Toc167366279)

[2.3. Модификация механизма взаимодействия с LLM моделью. 8](#_Toc167366280)

[3. АНАЛИЗ РЕЗУЛЬТАТОВ 11](#_Toc167366281)

[ЗАКЛЮЧЕНИЕ 12](#_Toc167366282)

[СПИСОК ЛИТЕРАТУРЫ 13](#_Toc167366283)

**ВВЕДЕНИЕ**

LLM – это мощные модели искусственного интеллекта, которые используют возможности обработки естественного языка для эффективного понимания, создания и управления человеческим языком. Эти модели обучаются на огромных объемах текстовых данных, что позволяет им преуспеть в различных языковых задачах, таких как контекстно-зависимое понимание естественного языка, генерация текста, перевод и обобщение.

Данный проект является прототипом внедрения LLM модели внутри трёхмерного движка UE 5. В контексте компьютерных игр можно отметить, что создание ботов - модели программных компонентов для игровых агентов – использует разные подходы и методы. Например, конечно-автоматная модель, модель поведения с применением машинного обучения, структура с расписанием действий. В данной работе, в качестве модели поведения игрового агента используются текстовые запросы и ответы LLM модели.

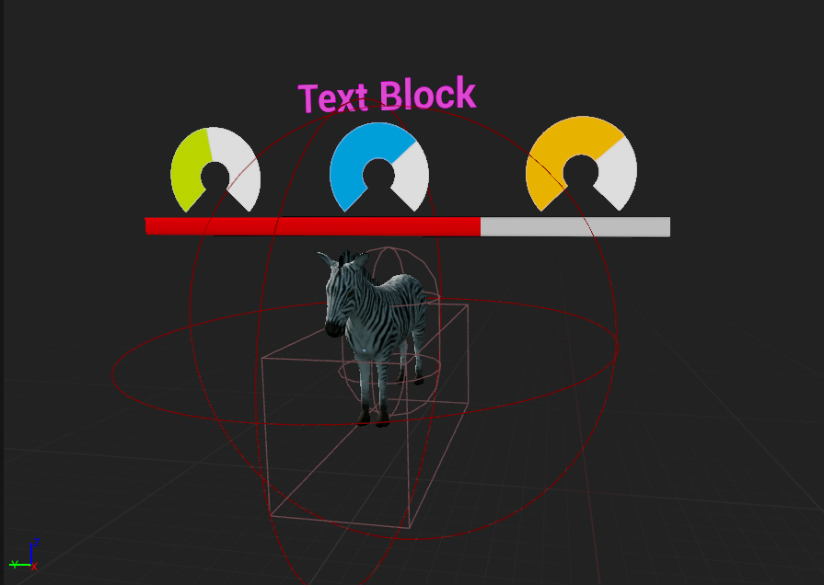
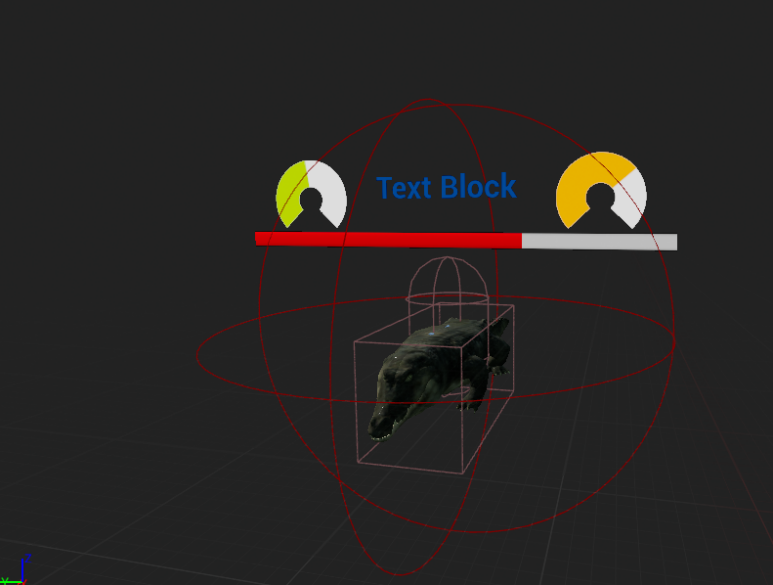
Для реализации эффективного взаимодействия с LLM моделью необходимо выбрать нужную модель, правильно формировать вопрос и ответ к ней, реагировать на некорректное поведение, продумывать различные сценарии использования модели, исследовать предметную область и рынок, анализировать качество уже придуманных решений.

Цель практики: расширить имеющуюся систему для внедрения множества виртуальных агентов под начальством LLM модели.

1. **ПОСТАНОВКА ЗАДАЧИ**

Требуется разработать систему общения LLM модели с несколькими игровыми агентами. Скоординировать их действия в пошаговом режиме. Реализовать взаимодействие генеративных игровых агентов с внешним миром. Создать интерфейс, показывающий лог событий и данные о выбранных объектах, для конечного пользователя.

1. **КРАТКИЙ ОТЧЕТ О ВЫПОЛНЕННЫХ РАБОТАХ**
   1. Разработка виртуального агента «Зебра»

За основу данного агента был взят ранее разработанный виртуальный агент «крокодил» (рис. 1).

а) б)

Рисунок 1. Трёхмерная модель агента зебры (а) и крокодила (б).

В дополнение к обеим объектам типа Character был добавлен компонент Sphere Collision с разными радиусами для двух агентов. Он будет использован для реализации механики охоты для крокодила.

Изменена логика расчёта жизненных показателей для агентов. Теперь за это отвечает бесконечный таймер с периодом в 1 секунду, который запускается в функции BeginPlay(). Показатели всё также изменяются, если состояние агента в дереве поведений (объект BehaviorTree) отлично от состояния idle (бездействие) (рис. 2).

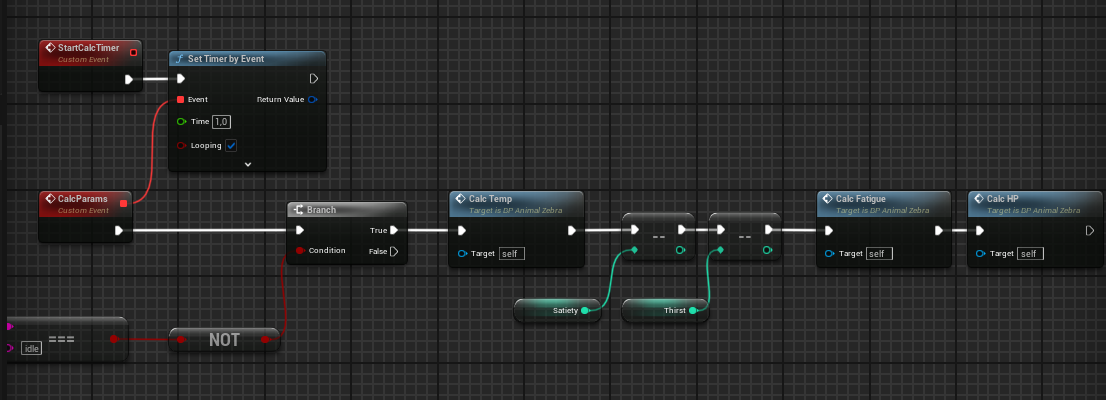


Рисунок 2. Таймер и событие для расчёта жизненных показателей виртуального агента.

Для виртуального агента зебры была добавленная переменная, отвечающая за жажду (Thirst). Данная характеристика может быть пополнена на 50 единиц, если агент находится на блоке воды (объект WaterBlock) и выполняет действие «drink». В отличии от сытости (Satiety), параметр не может быть больше 100 (рис. 3).

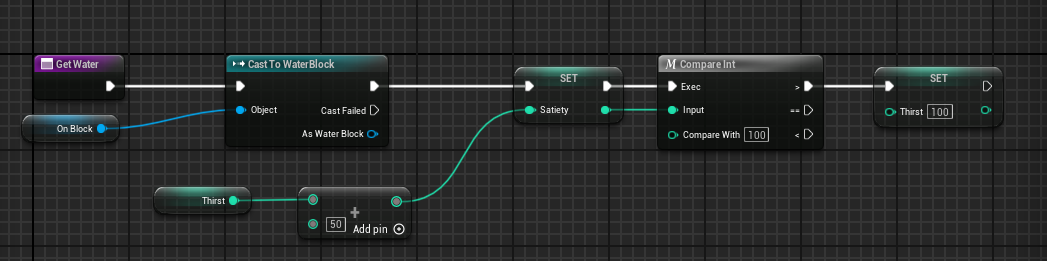


Рисунок 3. Функция для расчёта параметра жажды (Thirst) агента зебры.

* 1. Разработка интерфейса виртуальных агентов.

Для удобного мониторинга параметров виртуальных агентов, необходимого конечному пользователю, принято решения заменить ранее разработанные ToolTip-ы на 3D Widget-ы. Данные компоненты входят в состав моделей игровых агентов. Они состоят из объекта ProgressBar для отображения параметра здоровья, а также из объектов RadialSlider для прочих параметров. Объект TextBox нужен для отображения имени виртуального агента.

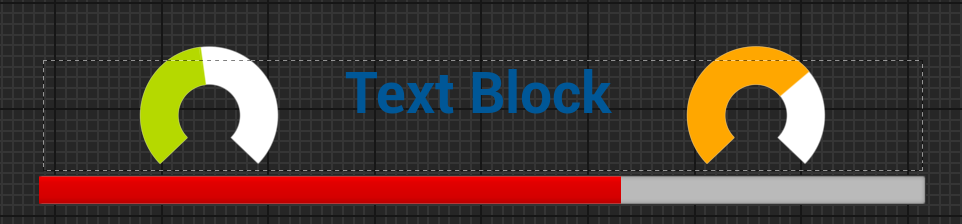
****

Рисунок 4. Объект BarWidget для агента «Крокодил».

Архитектура интерфейсов для крокодила и зебры представлена на рисунках 5,6.

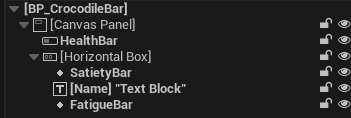
****

Рисунок 5. Архитектура BarWidget для агента «Крокодил».

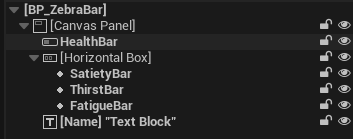
****

Рисунок 6. Архитектура BarWidget для агента «Зебра».

Параметры визуальных компонентов изменяются в функции Tick(). Там же происходит расчёт положения Widget-а относительно экрана.

Стоит упомянуть, что существует два типа отрисовки Widget-а: World и Screen. При значении World объект помещается на сцену, в соответствии с настройками модели. При значении Screen объект отрисовывается непосредственно на экране у игрока (в данном случае игрок представляет собой подвижную камеру сверху), также он масштабируется в зависимости от настроек экрана (рис. 7). Опция Screen работает корректно в исключительных случаях, поэтому для объекта установлен режим рендера World с последующей программной реализации поворота, в соответствии с камерой игрока (рис. 8). Результат показан на рисунке 9.

Рисунок 7. Отображение 3D Widget в режиме Screen.

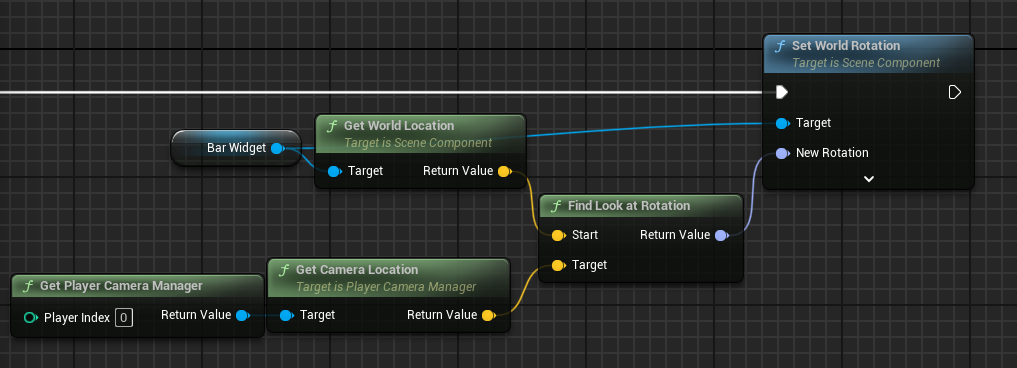


Рисунок 8. Поворот объекта 3D Widget относительно местоположения камеры игрока.

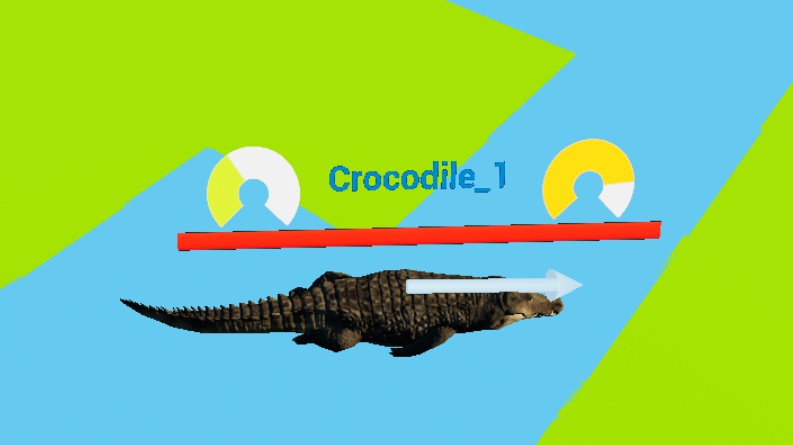


Рисунок 9. Отображение 3D Widget в режиме World с программной реализацией поворота.

* 1. Модификация механизма взаимодействия с LLM моделью.

Так как система должна быть расширяемой и поддерживать множество одновременно существующих виртуальных агентов, механизм взаимодействия с LLM моделью нужно модифицировать.

В предыдущей версии проекта агент сам отправлял и получал запросы от LLM модели. Теперь эту функцию перенял объект BP\_Spawner – наследник C++ класса ASpawner. Он принимает запросы от LLM модели, расформировывает их и отправляет виртуальным агентам с нужными именами. При этом агенты сами фильтруют запросы, преобразовывая их в действия. После получения действия агент выставляет свою готовность (переменная IsRequestDone типа Boolean). Стоит отметить, что BP\_Spawner не проверяет повторно эту переменную у всех агентов. Этот процесс трудозатратный для большого числа агентов и не несёт в себе функции синхронизации действий агентов; синхронизация происходит при отправлении ответа LLM всех агентам единожды. Функции фильтрации текста у агентов не сильно отличаются, задержки работоспособности не наблюдаются.

После окончания действия агенты переходят в состояние idle (бездействие) и отправляют объекту BP\_Spawner свои запросы. Пример запроса с указанием жизненных параметров агента и подсказок для LLM модели представлен на рисунке 10.

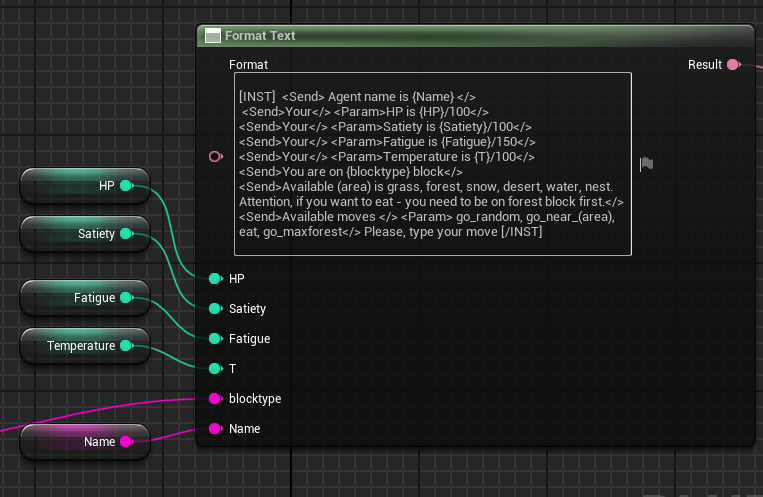


Рисунок 10. Текстовый запрос от виртуального агента.

Получив все запросы от агентов BP\_Spawner формирует составной запрос для LLM модели с просьбой расписать последующие действия для каждого агента. Стоит отметить, что получаемый текст получается большим даже с учётом формата JSON. Для корректной обработки текста LLM моделью значения «памяти» модели в виде числа обрабатываемых токенов было увеличено. Эта величина формируется в самом первом запросе к модели.

1. **АНАЛИЗ РЕЗУЛЬТАТОВ**

Результат практической работы показал хорошие перспективы использования LLM модели для управления множеством виртуальных агентов. Планируется реализация игровых механик взаимодействия агентов, такие как охота, размножение. Предусматривается возможность указания пользователем начального количества игровых агентов (крокодилы и зебры) на сцене. В планах также есть система оценки состояния игры в виде бальной системы в зависимости от числа живых агентов и их параметров. Предполагается тестирование других LLM моделей для формирования управляющих команд виртуальных агентов.

Что касается разработки интерфейса, – проведён анализ режимов рендеринга объекта 3D widget у моделей агентов, реализован поворот компонентов от местоположения камеры.

**ЗАКЛЮЧЕНИЕ**

В результате выполнения работы были получены навыки работы с Unreal Engine 5, получен опыт разработки на C++. Освоены навыки 3D-моделирования и анимирования. Изучены способы взаимодействия, настройки и интеграции модели LLM в движок Unreal Engine 5.

В процессе работы выполнены следующие задачи:

1. Создание виртуального агента «Зебра»;
2. Создание 3D интерфейса для виртуальных агентов;
3. Программная модификация взаимодействия с LLM моделью и генеративными агентами.

**СПИСОК ЛИТЕРАТУРЫ**

1. Unreal Engine 5 Documentation [электронный ресурс] // Unreal Engine Documentation URL: <https://docs.unrealengine.com/>. Дата обращения: 16.05.2024.
2. Основа геймплея игры на C++ для Unreal Engine [электронный ресурс] // URL: https://habr.com/ru/articles/357976/. Дата обращения: 11.03.2024.
3. Настройка параметров БЯМ [электронный ресурс] // URL: <https://obnimorda.ru/guides/prompt/introduction/settings/>. Дата обращения: 16.05.2024.
4. Display aspect ratio [электронный ресурс] // Wikipedia, the free encyclopedia URL: <https://en.wikipedia.org/wiki/Display_aspect_ratio/>. Дата обращения: 16.05.2024.
5. Geometry instancing [электронный ресурс] // Wikipedia, the free encyclopedia URL: <https://en.wikipedia.org/wiki/Geometry_instancing>. Дата обращения: 16.05.2024.
6. Using Large-Language Models (LLM) in Game Development [электронный ресурс] // URL: <https://gamedevacademy.org/using-large-language-models-llm-in-game-development-tutorial-list/#How_Can_Large_Language_Models_Be_Used_for_Games> Дата обращения 16.05.2024.
7. Generative Agents: Interactive Simulacra of Human Behavior [электронный ресурс] // URL: <https://arxiv.org/abs/2304.03442> Дата обращения: 16.05.2024.
8. Инстансинг [электронный ресурс] // Habr: интернет-портал. URL: <https://habr.com/ru/post/352962/>. Дата обращения: 16.05.2024.