Добрый день, уважаемая комиссия. Тема моей выпускной работы – разработка метода интеграции больших языковых моделей средствами REST API для управления виртуальными агентами в Unreal Engine 5. (версия движка 5.2.1)

**[#2. ]**

Требуется разработать симуляцию с виртуальными агентами, которые имеют различные внутренние параметры. За управление агентов отвечает большая языковая модель, общение с которой происходит через стороннее приложение по REST API (Ollama). Также необходимо в пошаговом режиме синхронизировать действия агентов между собой и ответами от LLM.

**[#3. ]**

Актуальность работы. Большие языковые модели появились в 2018 году, и с тех пор всё лучше справляются с широким спектром задач без предварительного обучения. В Epic Game MarketPlace – основном рынке приложений на Unreal Engine – в настоящий момент представлено небольшое количество решений для интеграции LLM в движок. Также большинство проектов c LLM используют их в основном для творческих задач.

**[#4. ]**

Первое важное определение – компьютерный или виртуальный агент. Виртуальный агент - это автономная сущность, которая действует в окружающей среде и наблюдает за ней. Агенты по внешним и внутренним параметрам проводят анализ и принимают своё решение.

**[#5. ]**

Второе важное определение – большая языковая модель или LLM. Это нейросеть с огромным числом параметров (весовыми коэффициентами), которую обучили на большом количестве данных. Модели работают с естественным языком. Несмотря на огромное возможности, внутренняя работа LLM остается загадкой. Они оперируют миллиардами параметров и их взаимодействие между собой до конца непонятно. (обучение без учителя, на неразмеченном тексте).   
  
Стоимость одной итерации обучения LLM примерно 1 млн. рублей, Алексей Фетисов, генеральный директор Т1, выпускник рк6, был в мгту в конце апреля, 26 апреля.  
  
LLM можно дообучать (fine-tuning). Большинство открытых моделей – pretrained, они не специализированные. даю

**[#6. ]**

Принцип работы LLM. Для начала введём понятие токен. Токен – минимальная единица текста, например, слово, словосочетание, знак препинания. Принцип генерации текста состоит в следующем: модель получает на вход текст или «промпт», разбивает его на токены; затем она предсказывает один токен и полученный текст снова загружается на вход модели. Так происходит до тех пор, пока не будет сгенерирован ответ.

**[#7. ]**

Механизм взаимодействия с LLM следующий: модель получает системный запрос с правилами симуляции, после старта приложения каждый раунд собирается информация со всех виртуальных агентов. Затем формируется большой POST запрос с текущим состоянием симуляции. Также создаются 2 события-обработчика, которые принимают успешный и неуспешный ответ, соответственно. При неудачном ответе формируется новый запрос с прежней информацией о симуляции. При успешном ответе симуляция продолжается.

**[#8. ] xd [#9. ]**

Поле состоит из Instanced Static Mesh Component кубов, символизирующих природные зоны. Каждый блок – наследник от класса базового куба. К некоторым блокам дополнительно генерируется инстансы доп. объекта (растение или камень). Каждый регион имеет свою температуру и максимальную скорость передвижения по нему. Генерируется поле на основе двумерного массива строк, где за определённым символов закреплён тип блока – если встречается неизвестный символ, куб не появится. Такой подход позволяет создавать карты разнообразных форм.   
  
Инстансинг геометрии (дублирование геометрии) – подход, позволяющий отрисовывать множество копий одного и того же 3d-объекта за один проход.

* Основная цель инстансинга – понижение количества вызовов отрисовки.
* Каждая копия объекта называется *инстансом*.

**[#10. ]**

В моем проекте есть два агента: крокодил и зебра. У каждого из них есть параметры, такие как очки здоровья (health point), сытость (satiety), энергия (energy), температура и максимальная скорость передвижения. Зебра дополнительно имеет характеристику жажды, а крокодил может таскать с собой ветку из леса. Над моделями агентов расположен 3D-виджет с информацией в виде progressbar, круглых слайдеров и текстовых полей. Также у них есть параллелепипеды под ногами, которые определяют в каком регионе сейчас находится агент. Дополнительно, персонажи имеют сферы разных радиусов для расчета коллизий между ними.

**[#11. ]**

В течение жизнедеятельности агента его параметры медленно убывают каждый тик внутреннего таймера. Если множество характеристик не в норме, здоровье агента будет уменьшаться. При здоровья меньше 0, агент умрёт, его объект будет уничтожен.  
Зебра может восстановить жажду на блоке воду, отдыхать на любом блоке, кроме воды; и есть, находясь в регионе равнин.   
Крокодил способен отдыхать только в своём гнезде, питаться он может только при успешной охоте – если сфера крокодила пересекла сферу зебры – при этом объект зебры уничтожается в мире, а крокодил пополняет сытость. Любой агент может отправиться в случайное место на карте или в определённую зону. Только крокодил может начать следовать за ближайшей зеброй. Зебры быстрее крокодилов, на них не действуют ограничения максимальной скорости от регионов. Однако крокодил может улучшить гнездо и получить бонусы к скорости с помощью веток из леса.

[**#12. ]**

При успешном ответе каждый агент фильтрует ответ от модели, получая свои команды. Если команды нет в списке разрешённых действий, агент идёт в случайную область. После каждого действия для агента может быть сформирована подсказка для следующего запроса. Например, если модель неправильно задала действие для агента – в подсказку запишутся все возможные команды для данного агента. Или, если модель пытается восстановить агенту параметр, значение которого максимум, подсказка предложит модели выбрать другое действие.

[**#13. ]**

Проект тестировался на разных ПК. Результаты представлены на слайде. Офлайн модели: Gemma7B, LLama3-7B. Быстрее всех llama3, но у Gemma большая часть ответов затрагивала все имена агентов (ллама3 иногда ходила только для зебр или крокодилов). При этом, если использовать онлайн LLM – то среднее время ответа будет 4-6 секунд. GPT 3.5 сразу понял особенность крокодилов и единственный, кто смог улучшить гнездо.

[**#14. ]**

Итогом выпускной квалификационной работы стал проект, симулирующий деятельность виртуальных агентов в разнообразном мире под контролем LLM модели. Выполненные задачи – проектирование трёхмерной сцены из (Instanced Static Mesh Component) кубов и других mesh-ей. Созданы различные виртуальные агенты с анимациями. Проведены тесты на разных ПК. Проанализированы локальные, онлайн LLM и качество ответов от изменения их параметров. Выяснено, что техника «промпт-инжиниринга» очень важна, как и система подсказок.

В заключение хотелось бы сказать, что разработанный инструмент можно использовать для тестирования новых LLM на предмет их развития и возможности генерации разумных решений.

Промпт-инжиниринг - это относительно новая дисциплина разработки и оптимизации промптов для эффективного использования языковых моделей в широком спектре приложений и исследовательских тем. Навыки промпт-инжиниринга помогают лучше понять возможности и ограничения больших языковых моделей. Например, Self-Consistency - Идея заключается в выборе нескольких различных путей рассуждения. В запросе много примеров вопросов/ответов, после них идет вопрос, на который llm должна ответить похожим образом.

Агенты – слепые, в проекте не используются система восприятия (perception).

Любые аналитические модели и конечные автоматы делают поведение детерменированным, в них есть шаблонность. Есть деревья поведений агентов, но при росте числа параметров – деревья сложно модернизировать, у них появляется большое ветвление. Причем деревья нельзя менять динамически.  
Самое лучшее - система подсчёта очков на основе выгодности действий. Система выявляет доступные для агента действия и начисляет им очки в зависимости от складывающихся обстоятельств, тем самым выбирая наиболее выгодное. Для каждого параметра строится кривая приоритета на основе текущего значения. Но данные для кривых нужно брать из перечисления всех возможных игровых ситуаций. Нужно грамотно настроить начисление очков приоритета. А при использовании LLM, не надо считать мат модель.

Особенность проекта в том, что в unreal используются блюпринты и с++. А с моделями в основном общаются через высокоуровневые языки.