УДК 004.89

ИСПОЛЬЗОВАНИЕ ИИ ДЛЯ МОДЕЛИРОВАНИЯ ПОВЕДЕНИЯ NPC В ИГРАХ

Акулова Е. О., студент гр. ИТб-211, IV курс
Протодьяконов А. В., к.т.н., доцент
Кузбасский государственный технический университет имени Т. Ф. Горбачева
г. Кемерово

Игровая индустрия в наши дни переживает впечатляющий скачек в развитии. Современный игровой процесс стал чем-то большим, чем просто развлечение и досуг. Практически во всех сферах — от обучения до экономики — видеоигры находят свое применение. Это создаёт необходимость в разработке игр, которые могут максимально увлечь игрока и удерживать его внимание на протяжении всего игрового процесса.

Одной из наиболее актуальных задач для разработчиков при создании видеоигр является создание глубокого погружения игрока в игру. Погружение — это своего рода ощущения при взаимодействии с игрой, когда пользователь чувствует себя частью виртуального мира, который отличается от повседневной реальности, а также сосредоточивает и полностью захватывает внимание. Это ключевые факторы, которые определяют, насколько игра интригует и передает задуманные автором идеи и концепции. Именно для этого разработчики используют Искусственный интеллект при создании своих игровых миров.

Искусственный интеллект в играх — это набор алгоритмов, который определяет действия неигровых персонажей NPC[1]. Прогрессивная технология, использующая машинное обучение для анализа данных в виртуальной среде, приводит к созданию поведения, все более схожего с нашим. Еще несколько десятилетий назад подобные технологии представлялись исключительно невероятными, но в наше время их применение стало чем-то обыденным. Искусственный интеллект способен создать персонажа, который ведет себя как человек, управлять всеми NPC и разнообразными аспектами игровой механики, что делает игру более интерактивной и увлекательной.

Целью данного доклада является ознакомление с современными технологиями и методами искусственного интеллекта, применяемыми для создания и моделирования поведения неигровых персонажей (NPC) в видеоиграх. Мы рассмотрим пути улучшения реалистичности и адаптивности персонажей, а также обсудим алгоритмы и подходы, используемые в этой области.

[1] NPC (от англ. Non-player character) — неигровой персонаж в играх, который не находится под контролем игрока.

Благодаря интеллектуальным системам, персонажи не просто воспроизводят фиксированные действия, но и способны адекватно реагировать на действия игроков. Это усиливает реализм игрового процесса и создает иллюзию разумности NPC. Основная цель ИИ остается неизменной: не просто выигрывать соревнования, а приносить удовольствие игроку.

Ниже приведены несколько примеров, как ИИ может быть использован для улучшения игрового процесса:

- Улучшенный ИИ: Нейросети могут способствовать созданию более умных NPC, которые умеют адаптироваться к действиям игрока и принимать разумные решения.

- Эмоциональные реакции: Персонажи могут выражать эмоции в ответ на действия игроков, будь то радость, страх или гнев, что обогащает взаимодействие и делает его более глубоким.

- Обучение в режиме реального времени: ИИ может обучать персонажей в процессе игры, а не заранее, что делает их поведение более динамичным.

- Генерация контента: Нейросети могут создавать контент, такой как диалоги, квесты и уровни, что позволяет разнообразить игровой процесс.

Это лишь некоторые примеры того, как ИИ может быть применен в игре. С развивающимися технологиями и искусственным интеллектом, горизонты применения нейросетей будут становиться только шире и более захватывающими для разработчиков и игроков.

Алгоритмы, управляющие поведением неигровых персонажей, разрабатываются с применением различных методов и инструментов, среди которых можно выделить несколько ключевых подходов:

- Rule-based ИИ:

  - Этот алгоритм основывается на наборе заранее определённых правил и условий, которые создаются разработчиками. Он отлично подходит для реализации простых моделей поведения. Например, можно установить правило, что «если игрок приближается к животному ближе определенной дистанции, то оно начинает убегать». Однако значительный недостаток данного метода заключается в том, что поведения персонажа остаются фиксированными, что делает игровой процесс предсказуемым и снижает интерес к игре. Поэтому важно найти пути для улучшения искусственного интеллекта, чтобы поведение NPC стало более многообразным и разумным.

- Конечные автоматы:

  - Этот подход позволяет NPC плавно переходить между разными состояниями. К примеру, персонаж может патрулировать по заданной траектории, но, встретив игрока, переключается на новое состояние, начинает проявлять активность. Конечные автоматы управляют этими переходами: они обрабатывают информацию о предыдущем состоянии и переходят в новое. В рамках модели конечных автоматов (FSM) разработчик анализирует

все возможные сценарии, с которыми может столкнуться ИИ, и затем программирует соответствующую реакцию для каждого из них. Например, в шутерах[2] искусственный интеллект может начинать атаку, когда замечает игрока, а затем отступать, если его здоровье падает ниже определённого уровня. В рамках алгоритма FSM NPC может выполнять четыре главных действия в ответ на различные ситуации: поиск помощи, уклонение, блуждание и нападение.

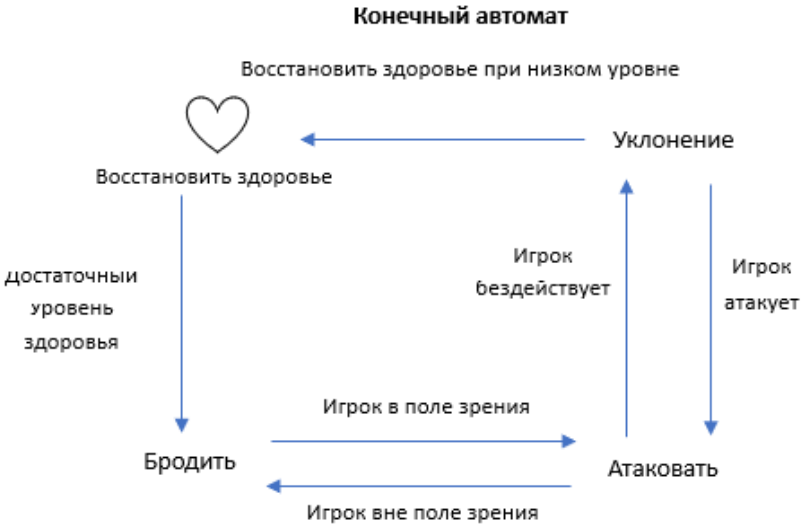

Рисунок 1 – пример алгоритма «Конечный автомат»

В примере алгоритма FSM NPC может выполнять основные действия в ответ на возможные ситуации: восстановление здоровья, уклонение, бродить и атаковать.

- Дерево поиска:

- Алгоритм дерева поиска Монте-Карло (MCTS - Monte Carlo Tree Search). Этот подход более продвинутый, способен обрабатывать реакции с использованием множества различных методов. Он позволяет расставлять приоритеты в реагировании на внешние стимулы и внедрять новые модели поведения на определенных уровнях дерева. Разработка алгоритма MCTS была направлена на устранение проблемы повторяемости, характерной для алгоритмов конечных автоматов (FSM). В начале работы MCTS анализирует все потенциальные ходы, доступные NPC в текущий момент. Затем для каждого из этих возможных действий исследуются реакции, которые может предпринять игрок. В заключение алгоритм снова возвращается к оценке действий NPC, основываясь на собранной информации о поведении игрока.

[2] **Шутер** — **жанр компьютерных игр**, основу игрового процесса которой составляет взаимодействие с внутриигровыми объектами посредством стрельбы в них.

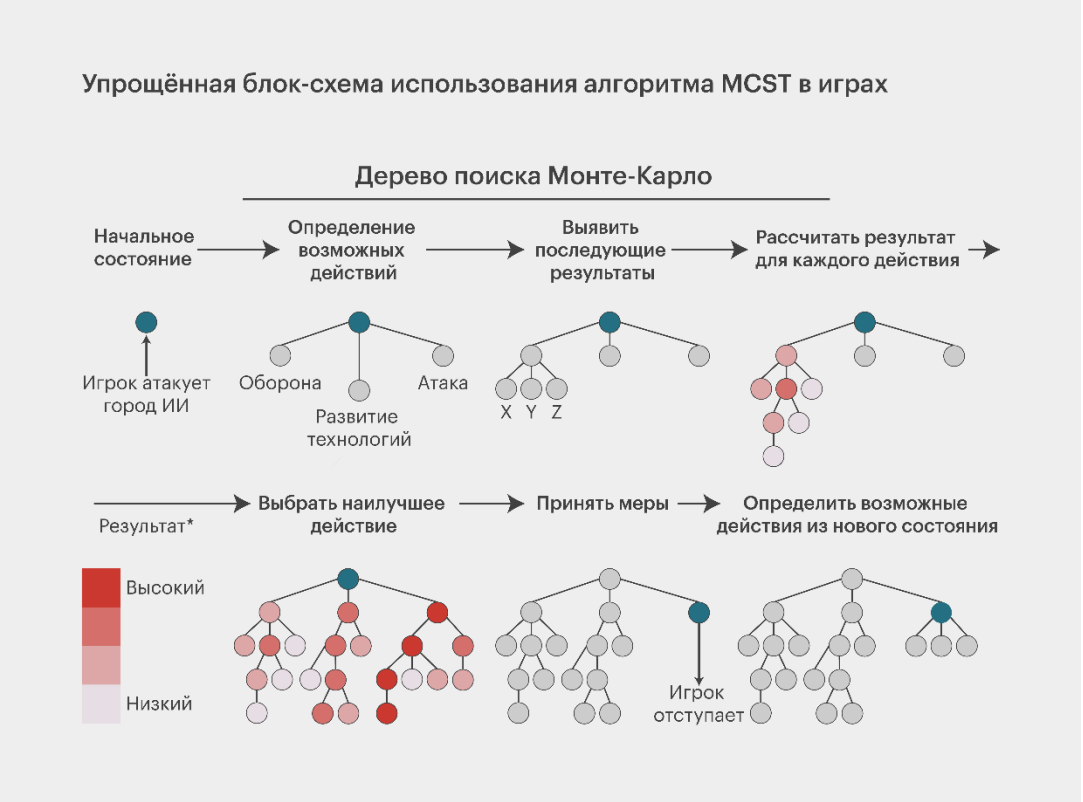

Рисунок 2 – Пример алгоритма «Дерево поиска»

- Автоматизированные планировщики:

- К числу таких методов относят GOAP (планирование действий, ориентированное на цели) и HTN (иерархическая сеть задач). GOAP применяет динамическую оценку целей NPC, предоставляя им возможность самостоятельно определять выбор действий в зависимости от актуальных условий и приоритетов. Каждое действие сопровождается уникальными условиями и последствиями, что позволяет принимать более гибкие решения. В свою очередь, HTN использует иерархическую структуру, позволяя делить сложные задачи на более простые подзадачи. Это делает процесс планирования более структурированным и продуктивным.

Такой подход особенно актуален в случаях, когда необходимо выполнять многоуровневые действия для достижения конечной цели. В совокупности эти техники формируют реалистичное и адаптивное поведение NPC, что значительно обогащает игровой опыт и взаимодействие с игроком. Хотя эти методы обеспечивают масштабируемость, их реакции могут быть менее предсказуемыми, поскольку алгоритмические решения зависят от текущего состояния игровой среды и целей NPC.

- Навигационная сетка:

- Это специализированная карта, указывающая области, по которым могут перемещаться NPC. На навигационной сетке может быть наложена система узлов, помогающая NPC правильно интерпретировать сценарии поведения. Навигационная сетка разбивается на сегменты многоугольников,

составляющих игровую карту. Персонажи используют алгоритмы поиска, чтобы выяснить кратчайший путь между двумя точками, перемещаясь через узлы сетки. Преимущества данного подхода включают в себя высокую скорость вычислений и предсказуемое поведение NPC, так как они следуют заданным маршрутам. Тем не менее, требуется тщательная настройка сетки, чтобы избежать ситуаций, когда NPC застревает в труднопроходимых местах или не может добраться до цели из-за неправильно спроектированной структуры.

- Алгоритмы машинного обучения:

- Это технология дает персонажам возможность «принимать решения» относительно своих дальнейших действий. Основой данного подхода является алгоритм оптимизации, который учится на наборах данных, базирующихся на состоянии персонажа в процессе обучения. Типы алгоритмов включают:

1. Обучение с учителем — алгоритмы, которые обучаются на размеченных данных, где каждая входящая информация соответствует известному выходному значению. Основная цель — создать модель, способную предсказывать выход на новых данных. К примерам относятся линейная регрессия, решающие деревья, нейронные сети.
2. Обучение без учителя — применяется к неразмеченным данным, где алгоритмы пытаются выявить структуры и закономерности. Примеры включают кластеризацию (например, K-средних), методы понижения размерности (например, PCA).
3. Обучение с подкреплением — NPC учится определять, как действовать в окружении, получая вознаграждения или штрафы за свои действия. Алгоритмы, такие как Q-обучение и Deep Q-Networks, позволяют персонажам адаптироваться к меняющимся условиям.

Рассмотрим технологии, необходимые для внедрения ИИ, ее реализацию и тестирование.

Существует множество фреймворков машинного обучения, ниже представлены некоторые из наиболее популярных, используемых в разработке игр:

TensorFlow : платформа с открытым исходным кодом для машинного обучения.

Unity ML-Agents : отличный набор инструментов, если вы используете игровой движок Unity.

PyTorch : еще одна библиотека машинного обучения с открытым исходным кодом.

Основные шаги для подключения ИИ к NPC:

1. Сначала следует установить критерии и параметры, которые будут определять решения NPC. Например, в играх с врагами такими параметрами

могут быть расстояние до игрока, его текущее здоровье и уровень развития, а также ряд других относящихся характеристик.

2. Собрать данные и разработать обучающую выборку, которая будет использоваться для тренировки. Это можно осуществить через анализ игрового процесса, чтобы выявить взаимосвязь между различными решениями, принимаемыми NPC.

3. Разработайте и обучите нейросеть, основываясь на собранном наборе данных. Для этого подойдут специализированные инструменты для создания и обучения нейросетей.

4. Внедрите обученную нейросеть в игровую платформу или систему NPC. Это потребует разработки кода, который будет собирать игровые данные и делать выводы на основе работы нейросети.

5. Протестируйте и отладьте системы NPC с интегрированной нейросетью. Важно удостовериться, что NPC принимает корректные и логичные решения, основываясь на результатах работы нейросети и заданных критериях.

6. При необходимости выполните дополнительное обучение и оптимизацию нейросети. В процессе игрового взаимодействия могут возникать новые сценарии или изменяться условия, требующие обновления нейросети или добавления новых обучающих данных.

7. Обеспечьте развитие и поддержку системы NPC с подключённой нейросетью. В процессе эксплуатации могут возникнуть проблемы или появиться потребность в усовершенствовании, что требует постоянного контроля и корректировки системы.

Процесс обучения нейросети на тренировочных данных включает в себя несколько шагов:

- Подготовка данных: тренировочные данные должны быть представлены и структурированы в формате, доступном и понятном для нейросети. Это может включать в себя преобразование данных в числовой формат, нормализацию значений и т.д.

- Определение архитектуры нейросети: Перед началом обучения необходимо выбрать тип нейросети и ее структуру. Можно использовать различные виды нейросетей, такие как перцептрон, сверточные, глубокие или рекуррентные нейросети. Важно выбрать наиболее подходящую архитектуру для решения конкретной задачи.

- Инициализация параметров: прежде чем начинать обучение, необходимо установить начальные значения параметров нейросети, включая в себя инициализацию весов, выбор функции активации и др.

- Прямое распространение: на этом этапе тренировочные данные подаются на вход нейросети, и начинается процесс прямого распространения сигнала. Информация проходит через различные уровни сети, где

каждый из них выполняет свою роль в обработке.

- Обратное распространение ошибки: после прямого распространения вычисляется ошибка, отражающая разницу между предсказанным выходом нейросети и реальным результатом. Эта ошибка затем возвращается назад через нейросеть для корректировки весов с целью минимизации ошибки.

- Повторение обучения: Процесс прямого и обратного распространения продолжается до тех пор, пока нейросеть не достигнет требуемой точности или не завершит обучение на всех тренировочных данных.

После завершения обучения нейросети на тренировочных данных, она будет готова к интеграции в систему игровых NPC. Итоговые результаты процесса обучения обеспечат нейросети способность принимать обоснованные решения, руководствуясь логикой и целями, заданными разработчиками.

Таким образом, создание модели нейросети в программе является важным этапом, который предшествует подключению нейросети к NPC в игре. Логически выверенная и тщательно обученная модель может существенно повысить уровень искусственного интеллекта NPC, сделав игровой мир более живым и привлекательным для пользователей.

Это лишь некоторые примеры того, как нейросети могут быть применены в разработке игр. Искусственный интеллект играет ключевую роль в создании увлекательного и динамичного игрового опыта. Моделирование поведения NPC с использованием алгоритмов ИИ открывает новые горизонты для разработчиков и позволяет создавать более реалистичных и адаптивных персонажей. В будущем, с развитием технологий и искусственного интеллекта, возможности применения нейросетей станут еще более широкими и интригующими для разработчиков и игроков.

Список литературы

1. С. Хайкин, Нейронные сети: Полный курс. - 2-е изд. - Москва: Издательский дом "Вильямс", 2006. - 1104 с.

2. Создание реалистичного поведения NPC с помощью машинного обучения: руководство 2024 года [Электронный ресурс] URL: https://toxigon.com/creating-realistic-npc-behaviors-with-machine-learning (дата обращения: 30.03.2025).

3. Не совсем человек: искусственный интеллект в играх [Электронный ресурс] URL: https://skillbox.ru/media/gamedev/iskusstvennyy-intellekt-v-igrakh/ (дата обращения: 29.03.2025).

4. Игровая революция - разблокируй огромный потенциал с подключением нейросетей к НПС в игре! [Электронный ресурс] URL: https://madelephant.ru/b/podklyuchenie-nejroseti-k-nps-v-igre-instrukciya-i-primery (дата обращения: 31.03.2025).