Исследование и применение различных методов и алгоритмов для реализации искусственного интеллекта в игровых персонажах

У. А. Гаврилова, А. И. Степанчикова
Белорусский государственный университет информатики и радиоэлектроники, Минск, Республика Беларусь

В статье представлена информация о различных техниках и алгоритмах реализации искусственного интеллекта для игровых приложений. Проведён анализ и выявлены на практике преимущества и недостатки таких алгоритмов, как искусственный интеллект, построенный на правилах, конечные автоматы и дерево поведений.

Ключевые слова: Искусственный интеллект, ИИ построенный на правилах, Конечный автомат, Дерево поведений, Нейронные сети.

Введение

На данный момент искусственный интеллект (ИИ) стал важным элементом в самых разных областях, включая игровую индустрию. Персонажи, обладающие высокой степенью автономности и способности адаптироваться к изменяющимся условиям игрового мира, представляют собой ключевой элемент для создания увлекательного и непредсказуемого игрового опыта. Цель данной работы заключается в обзоре и анализе существующих методов и подходов к созданию искусственного интеллекта для игровых персонажей, а также их применении в игровой индустрии. Понимание и освоение этих методов позволит разработчикам игр создавать более реалистичных персонажей, что в свою очередь приведет к улучшению качества игрового контента.

Теоретическая часть

Искусственный интеллект в играх — это набор алгоритмов, которые диктуют поведение неигровых персонажей в разных ситуациях [1]. Игровой ИИ не обладает способностью к мышлению или творчеству, так как его действия подчинены правилам разработчиков. Несмотря на такие ограничения, грамотно созданный ИИ может адаптироваться к условиям работы и изменять свои действия в зависимости от ситуации.

Благодаря искусственному интеллекту персонажи не просто повторяют заученные действия, а действительно могут реагировать на поведение игрока. Это позволяет сделать игру более реалистичной и создает иллюзию разумности NPC. Основная задача ИИ, по-прежнему, заключается не в том, чтобы победить игрока, а помочь пользователю получить удовольствие от игры.

В основе искусственного интеллекта лежит общий принцип: получение информации, анализ, действие [1].

После того как ИИ получил информацию о действиях игрока, он принимает решение о дальнейшей тактике. В этом ему помогают различные алгоритмы, такие как ИИ построенный на правилах, конечные автоматы, дерево поведения, иерархические конечные автоматы, нейронные сети.

Самая простая форма ИИ — это система, построенная на наборе правил. Можно даже поспорить, является ли данная система ИИ. В любом случае она далека от традиционного представления об искусственном интеллекте. Поведение игровых объектов происходит за счёт уже установленных алгоритмов, учитывающими определённые факторы игры [2]. Такой подход можно эффективно использовать для создания простого поведения. Например, «если

противник подходит к игроку на расстояние меньше заданного значения, он начинает атаковать игрока».

Конечные автоматы – самый распространенный алгоритм в видеоиграх. Разработчик заранее прописывает все ситуации, которые могут произойти с NPC, и его реакцию. Конечный автомат состоит из состояний, событий и таблицы переходов [3]. Автомат начинает работать с заданного стартового состояния. Если происходит передача данных во время выхода из события, то на входе в другом состоянии необходимо принять и обработать эти данные. Отправка события — действие конечного автомата, а не конкретного состояния.

Преимущество этого подхода в том, что персонаж всегда будет находиться в каком-то состоянии и не зависнет где-то между ними. Так как разработчик должен прописать все переходы, он точно знает, в каких состояниях может находиться игровой объект. Минус этого алгоритма в том, что чем больше и детализированнее становится игра, тем больше ситуаций нужно предусмотреть, а это в свою очередь увеличивает риск возникновения ошибок.

Более усовершенствованный метод, который используют разработчики для повышения персонализированного игрового опыта, — алгоритм дерева поиска Монте-Карло или дерево поведений. Его уникальная черта заключается в том, что все состояния персонажа организованы в виде ветвистой структуры с понятной иерархией. Данный алгоритм был создан для предотвращения аспекта повторяемости, который присутствует в алгоритме конечных автоматов [4].

Главным преимуществом деревьев поведения является формальность. При помощи набора инструментов, шаблонов и структур могут быть реализованы очень интересные и сложные состояния, даже связанные друг с другом. В этом кроется одна из причин того, что именно деревья поведений стали частым выбором для реализации искусственного интеллекта в компьютерных играх и создании небольших роботов.

Иерархические конечные автоматы объединяют особенности конечных автоматов и дерева поведения. Особенность такого подхода в том, что разные графы внутри логики могут отсылаться друг к другу [1]. Иерархические конечные автоматы являются расширением конечных автоматов, в которых состояния могут иметь подсостояния. Это обеспечивает более структурированное и модульное представление поведения системы, что упрощает ее проектирование и обслуживание.

Создание искусственного интеллекта игроков может быть проблематично. Есть второстепенные персонажи, которые могут сопровождать игрока или быть очень важной частью игрового процесса. И отдав управление над таким персонажем обученной нейронной сети можно добиться более универсального поведения с меньшим количеством затрат времени и ресурсов. А также хороший партнер может значительно улучшить восприятие игры [5].

Использовать нейронные сети можно не только для интеллекта дружественных персонажей, но и для интеллекта противника. Искусственные боты среди врагов очень часто не могут приспособится к экстраординарным действиям игрока, так как они слишком ограничены, и они не способны на реакции, не заданные их программой. Но нейронные сети могут улучшить это поведение во много раз. Их можно научить реагировать на окружающую ситуацию, например, прятаться и адаптироваться, чего практически невозможно достичь только с помощью скриптов.

Практическая часть

Искусственный интеллект разрабатывался для мобильной казуальной игры. По задумке игрок сражается с ИИ-противником. Задача игрока — вытолкнуть другого снежками с платформы либо убить его, потратив все жизни. На уровне генерируются различные бонусы, такие как лечение, ускорение игрока и модификации атаки.

Изначально ИИ в игре слишком примитивный: враг просто преследует игрока и стреляет, когда находится вблизи. Поэтому было принято решение усложнить поведение противника. Были реализованы следующие модели: ИИ построенный на правилах, конечные автоматы и дерево поведений.

Rule-based AI не предполагает слишком сложную логику поведения. Такой вариант подошёл бы для NPC, который не должен реализовывать слишком сложное поведение. Так, например, для проекта подошла бы схема, изображённая на рис. 1.

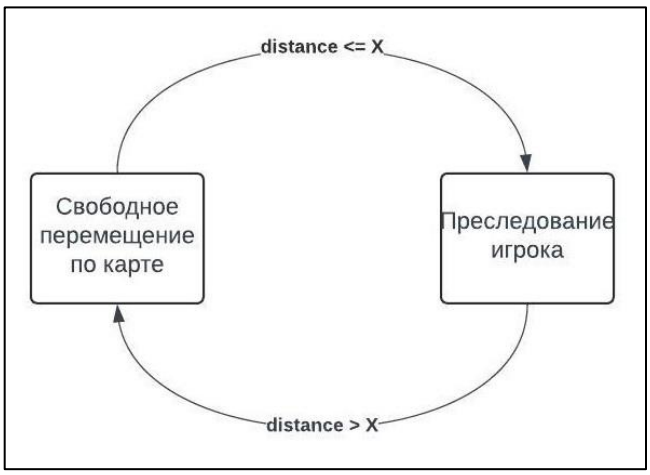

Рис. 1. ИИ построенный на правилах

Так, пока расстояние между игроком и NPC больше некой переменной, NPC хаотично перемещается по платформе. Однако, если же расстояние между ними сокращается до требуемой величины, NPC начинает преследовать игрока. Реализация данной ИИ-модели требует создания специального класса, например, EnemyBehavior, который бы отвечал за поведение NPC. Но если количество игровых механик растет, растет и размер класса. Связи между поведениями запутываются, поддерживать такой код становится сложно.

Конечные автоматы решают проблемы, обозначенные выше. Они хороши, когда необходимо ориентироваться на конкретные события, а модель поведения должна оставаться прежней, пока не изменятся условия [6].

Каждое состояние — это сценарий MonoBehaviour, который выполняет или не выполняет переход к следующему состоянию. Это позволяет усложнить поведение ИИ и при этом не запутаться в коде. На рис. 2 показана усложненная схема поведения врага для проекта в виде конечных автоматов.

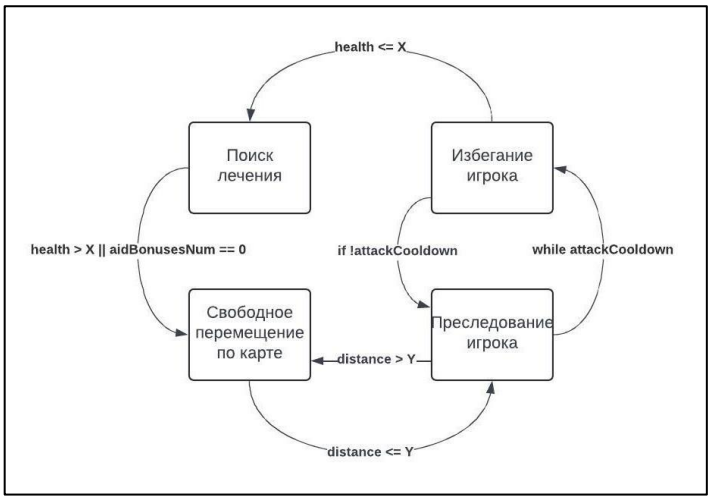

Рис. 2. Конечные автоматы

Теперь два состояния врага можно расширить до, например, четырех: поиск лечения, избегание игрока, свободное перемещение по карте и преследование игрока. Однако отсутствие разделения состояний на категории может в последующем усложнить жизнь разработчику. Например, если необходимо добавить новое состояние «поиск бонусов», схема сильно усложнится.

В целях оптимизации можно использовать дерево поведения. Данная модель позволит создать более чёткую архитектуру приложения. Схема для разрабатываемого проекта изображена на рис. 3.

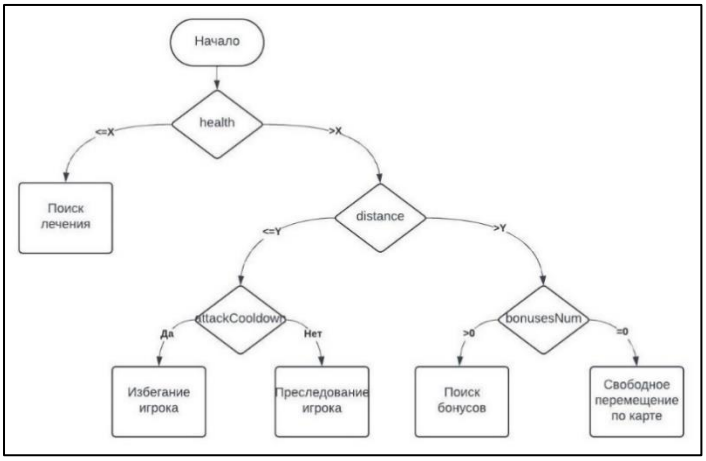

Рис. 3. Дерево поведений

В отличие от конечных автоматов, деревья имеют более формальную структуру, поэтому с их помощью проще программировать поведение ИИ.

Для данного проекта такая модель искусственного интеллекта является самой оптимальной, так как в полной мере реализует требуемый функционал и в то же время для его реализации не требуются углубленные знания. Однако в случае дальнейшего развития проекта и увеличения требований к ИИ данная модель может исчерпать себя, поэтому может потребоваться её усовершенствование и использование более продвинутых алгоритмов, таких как иерархические конечные автоматы или даже нейронные сети.

Заключение

В результате работы были рассмотрены следующие алгоритмы для реализации искусственного интеллекта для игровых персонажей: искусственный интеллект, построенный на правилах, конечные автоматы, дерево поведений, иерархические конечные автоматы и нейронные сети. На практике были выявлены преимущества и недостатки выбранных моделей. Подводя итог, важно отметить, что каждый алгоритм реализации искусственного интеллекта должен определяться исходя из требований конкретного проекта.

Список источников

[1] Имитация разума: как устроен искусственный интеллект в играх / URL: https://habr.com/ru/companies/netologyru/articles/598489/
[2] ИИ в играх: кто управляет NPC и как обыграть моба / URL: https://blog.skillfactory.ru/iskusstvennyy-intellekt-v-igrakh/
[3] Что такое конечные автоматы и как их использовать в разработке игр / URL: https://dtf.ru/gamedev/31493-chto-takoe-konechnye-avtomaty-i-kak-ih-ispolzovat-v-razrabotke-igr
[4] Что такое деревья поведения и как они используются / URL: https://habr.com/ru/companies/cloud_mts/articles/306214/
[5] Анализ применения искусственного интеллекта в видеоиграх / URL: https://sci-article.ru/stat.php?i=1669122954
[6] **Паласиос, Х.** Unity 5.x. Программирование искусственного интеллекта в играх/ Х. Паласиос; пер. Р. Н. Рагимова — М.: ДМК Пресс, 2017. — 272 с.

Научный руководитель — Рябычина Ольга Петровна, кандидат технических наук, доцент, доцент кафедры ИРТ БГУИР, Минск, Беларусь

Research and application of various methods and algorithms for the implementation of artificial intelligence in game characters

U. A. Haurylava, A. I. Stepanchikova
Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Annotation

The article presents information on various techniques and algorithms for implementing artificial intelligence in gaming applications. An analysis has been conducted, identifying the practical advantages and disadvantages of algorithms such as rule-based AI, finite state machines, and behavior trees.

Keywords: Artificial intelligence, Rule-based AI, Finite state machine, Behavior Tree, Neural networks.