СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В последние годы индустрия видеоигр привлекает большое количество человек. Игры стали одним из обычных способов проведения досуга. Одним из ключевых аспектов, делающих игры такими привлекательными, является наличие живого, динамично меняющегося мира, населенного неигровыми персонажами, каждый из которых имеет свои уникальные черты и поведение. Разработчики игр стремятся увеличить количество неигровых персонажей (нпс), чтобы сделать свои миры еще более реалистичными и увлекательными. Однако с ростом числа нпс увеличиваются и вычислительные затраты на их симуляцию, что ставит перед разработчиками задачу поиска баланса между детализацией мира и производительностью.

Создание убедительных и реалистичных нпс требует сложных систем искусственного интеллекта, способных моделировать разнообразное и естественное поведение. Традиционные подходы, такие как деревья поведения, хорошо зарекомендовали себя для управления поведением нпс в непосредственной близости от игрока. Однако они становятся слишком ресурсоемкими при необходимости моделировать поведение больших групп нпс, особенно в играх с открытым миром, где игрок может встретить тысячи персонажей. Это вызывает необходимость исследования альтернативных подходов, которые могли бы уменьшить вычислительные затраты без ущерба для качества и реалистичности игрового мира.

В данной магистерской диссертации предлагается разработка и анализ двухуровневой модели симуляции нпс, которая стремится сбалансировать детализацию поведения и вычислительные затраты. Исследуется подход, при котором нпс, находящиеся в непосредственной близости от игрока, симулируются с использованием деревьев поведения для обеспечения высокого уровня детализации и реалистичности. В то же время для симуляции нпс, находящихся на значительном расстоянии от игрока, предлагается использовать гидродинамическую модель, позволяющую эффективно

управлять большими группами персонажей. Этот подход позволяет значительно снизить вычислительные затраты, сохраняя при этом уровень детализации и реализм игрового мира, что делает его особенно актуальным для разработки современных видеоигр с открытым миром.

Целью данной работы является разработка многоуровневой модели симуляции неигровых персонажей на основе гидродинамического моделирования.

Для достижения этой цели были поставлены следующие задачи:

1) рассмотреть имеющиеся подходы к моделированию нпс в играх;

2) рассмотреть имеющиеся подходы к моделированию пешеходных и транспортных потоков;

3) рассмотреть применение изученных моделей в играх;

4) предложить собственное решение и разработать его

5) провести тестирование предложенного решения и проверить на соответствие заявленным требованиям;

6) сравнить полученные результаты с аналогами.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Существующие подходы к моделированию неигрового персонажа в играх

Разработка нпс для видеоигр претерпела значительное развитие с момента своего появления. От простых заранее заданных скриптов до сложных систем, способных к обучению и адаптации [1]. Подходы к программированию нпс в играх становятся всё более комплексными. Сегодня на вооружении разработчиков нпс есть множество методов и техник, каждая из которых имеет свои уникальные характеристики и области применения.

Далее представлен обзор ключевых подходов к программированию нпс в играх. Анализируя достоинства и недостатки каждого из этих подходов, можно получить представление о текущем состоянии искусственного интеллекта в игровой индустрии и о том, какие методы наиболее эффективны для создания убедительных, динамичных и интерактивных игровых миров.

1.1.1 Скрипты

Это традиционный подход, где поведение нпс задается заранее написанными скриптами. Данный метод считается достаточно примитивным, однако он хорошо работает в играх с линейным сюжетом и предсказуемым игровым процессом. Однако нпс на основе скриптов может ограничивать динамичность игрового мира, так как нпс не способны адаптироваться к неожиданным действиям игрока или изменениям в окружающей среде [2].

Скрипты обладают следующими преимуществами:

1) Позволяют создавать схемы с простым поведением при малых трудозатратах.
2) Не требуют дополнительных знаний для разработки.

Скрипты обладают следующими недостатки:

1) Не подходят для создания сложного поведения.
2) Для каждого нпс приходится разрабатывать отдельный скрипт.

1.1.2 Деревья поведений

Деревья поведений являются популярным методом создания искусственного интеллекта (ии) для неигровых персонажей. Деревья поведений представляют собой иерархическую структуру, состоящую из узлов, представляющих проверки условий или действия. Исполнение дерева поведения начинается с корневого узла, который передаёт управление одному или нескольким дочерним узлам, в зависимости от их типа. Исполнение узлов может быть последовательным, выборочным или параллельным, что обеспечивает гибкость в управлении поведением. Каждый узел возвращает статус после выполнения — успех или неудача.

Этот подход позволяет создавать сложное поведение нпс, разбивая его на более мелкие и управляемые части. Деревья поведений обеспечивают хороший баланс между гибкостью и контролем, делая их популярным выбором среди разработчиков игр [3, 4].

Деревья поведения обладают следующими преимуществами:

1) Благодаря большой гибкости и модульности данного метода, деревья поведений легко модифицировать и расширять. Это позволяет создавать сложных нпс с разнообразными задачами.

2) Относительно других методов, деревья поведений обладают понятной визуальной структурой, которая упрощает разработку и отладку.

3) Деревья поведений обладают широкой популярностью. Многие игровые движки и инструменты предлагают встроенную поддержку.

Деревья поведений обладают следующими недостатки:

1) С ростом сложности нпс структура дерева может стать громоздкой и трудной для управления.

2) Может быть сложно управлять общим состоянием и памятью для сложных задач.

1.1.3 Goal-Oriented Action Planning

Goal-Oriented Action Planning (GOAP) представляет собой систему планирования, которая позволяет нпс определять последовательность действий для достижения своих целей. GOAP работает на основе заданных целей и доступных действий, выбирая наиболее подходящий план в зависимости от текущего состояния мира и заданных условий.

Основная идея метода заключается в том, чтобы разбить сложную задачу или цель на ряд более мелких, простых действий, которые может выполнить агент. Затем эти действия организуются в последовательность действий, которые приведут агента к желаемой цели.

Ключевым элементом GOAP являются цели, к достижению которых стремится нпс. Каждая цель достигается через серию действий, которые нпс может выполнить. Каждое действие имеет условия для его начала и эффекты, которые изменяют состояние мира после выполнения действия.

Этот метод обеспечивает высокую степень адаптивности и реализма поведения [5, 6].

GOAP обладает следующими преимуществами:

1) Данный метод автоматически генерирует последовательности действий для достижения целей, учитывая текущее состояние мира.

2) GOAP позволяет нпс адаптироваться к изменениям в игровом мире, выбирая различные стратегии для достижения целей.

3) Отсутствие четко прописанного поведения на каждую ситуацию способствует созданию более убедительного и непредсказуемого нпс.

GOAP обладает следующими недостатками:

1) Планирование может быть ресурсоемким, особенно в сложных средах с множеством возможных действий.

2) Требует тщательного проектирования системы действий и условий.

3) Разработка эффективного планировщика и системы действий может быть трудоемкой.

1.1.4 Utility Systems

Это подход в программировании игровых ии, который позволяет создать гибкое и адаптивное поведение, где каждое действие оценивается по ряду факторов, таких как близость к цели, риск и потребности персонажа.

Utility systems оценивают каждое возможное действие или выбор на основе набора критериев или параметров, таких как расстояние до цели, здоровье персонажа, близость к врагу, и так далее. Каждый параметр имеет функцию полезности, которая преобразует его в числовое значение полезности. Затем, эти значения комбинируются для каждого действия, чтобы определить его общую полезность. Действие с наивысшей полезностью выбирается для исполнения.

Utility systems находят применение в различных жанрах игр, от стратегических и ролевых игр до симуляторов и экшенов. Они особенно полезны в играх, где важно создать иллюзию интеллектуального поведения нпс, способных к адаптации и стратегическому планированию [7].

Utility Systems обладают следующими преимуществами:

1) Данный подход предоставляет гибкость принятия решений. нпс, спроектированный при помощи utility systems, оценивает действия на основе их «полезности», что позволяет выбирать наилучшие действия в различных ситуациях.

2) Utility system является высоко адаптивным подходом. Ии, спроектированный с помощью utility system, может легко адаптироваться к изменениям в игровой среде, изменяя веса и функции полезности.

Utility systems обладают следующими недостатками:

1) Данный метод может обладать высокой сложностью настройки. Нахождение правильных функций и весов полезности может быть сложным и требовать много итераций тестирования.

2) Без должной настройки нпс может вести себя слишком предсказуемо или неоптимально.

1.1.5 Hierarchical Task Network Planning

Это метод, который базируется на принципе, декомпозиции задач на подзадачи до тех пор, пока не будут достигнуты простые действия, которые могут быть выполнены. В контексте видеоигр это означает, что поведение нпс может быть разработано путем определения серии задач, каждая из которых ведет к выполнению конкретного набора действий.

Hierarchical Task Network Planning (HTNP) поддерживает создание динамичных и адаптируемых систем поведения, которые могут реагировать на изменения в игровом мире и действия игроков, обеспечивая более глубокое взаимодействие [8].

HTNP обладает следующими преимуществами:

1) Данный метод позволяет создавать более реалистичное и сложное поведение нпс в разных игровых ситуациях благодаря декомпозиции задач на подзадачи.

2) Использование иерархического подхода к планированию задач позволяет разработчикам повторно использовать задачи и подзадачи в различных контекстах.

3) Данный метод предоставляет возможность легко вносить изменения в поведение персонажей, добавляя новые задачи или модифицируя существующие иерархии задач без необходимости переработки всей системы.

HTNP обладает следующими недостатками:

1) Первоначальная настройка и разработка иерархий задач является времязатратной и может требовать значительных усилий.

2) В некоторых случаях, строго определённые иерархии задач, могут ограничить способность ии адаптироваться к неожиданным ситуациям в игре.

3) Вычисление оптимального плана в большой и сложной иерархии задач может потребовать значительных вычислительных ресурсов.

1.1.6 Goal-Oriented Behavior

Это подход, позволяющий создавать искусственный интеллект с акцентом на достижение конкретных целей. Центральной концепцией Goal-Oriented Behavior (GOB) является ориентация AI на достижение заданных целей через выбор наиболее подходящих действий. Эти цели могут быть статическими (например, охрана определённой точки) или динамическими (адаптация к действиям игрока). Разработчики задают набор целей и возможных действий, а AI выбирает, какие действия предпринять для достижения этих целей на основе текущего состояния игровой среды. GOB особенно полезен в стратегических играх, симуляторах и RPG, где поведение нпс должно адаптироваться к динамическим ситуациям и выборам игрока. Этот метод позволяет создавать сложные сценарии взаимодействия, в которых нпс не просто реагируют на действия игрока, но и самостоятельно преследуют свои цели, делая игровой мир более живым и увлекательным [9].

GOB обладает следующими преимуществами:

1) GOB позволяет создавать гибкое поведение. AI, созданный с помощью этого подхода, может адаптироваться к изменениям в игровом мире, выбирая различные стратегии для достижения своих целей;

2) Различные способы достижения целей могут привести к разнообразному поведению нпс.

3) Целенаправленное поведение делает нпс более «живыми», так как их действия кажутся мотивированными и осмысленными.

GOB обладает следующими недостатками:

1) Настройка целей, действий и правил оценки может быть времязатратной и требовать значительных усилий для обеспечения нужного поведения нпс.

2) Вычислительные затраты на оценку возможных действий и их влияние на достижение целей могут быть значительными, особенно в сложных игровых мирах.

3) Без должного внимания к разнообразию возможных действий поведение нпс может стать предсказуемым и менее интересным.

1.2 Анализ существующих подходов к моделированию пешеходных и транспортных потоков

В современном мире широко применяются разнообразные методы моделирования для изучения и анализа различных аспектов жизни, таких как поведения больших групп людей [10, 11] или транспортных потоков [12]. Эти методы находят своё применение в самых разных областях – от урбанистики и транспортного планирования до обеспечения безопасности на массовых мероприятиях и моделирования распространения информации среди граждан. Способность точно симулировать поведение людей и движение транспортных средств при различных обстоятельствах позволяет принимать обоснованные решения в планировании, а также способствует оптимизации существующих систем и процессов [13].

Так, например, методы моделирования пешеходных потоков применяются для анализа и предсказания поведения людей в различных ситуациях. Это может включать в себя симуляцию движения через узкие проходы, взаимодействие в больших толпах людей, а также реакцию на чрезвычайные ситуации [18]. Моделирование пешеходных потоков помогает проектировать более безопасные и комфортабельные общественные пространства, а также оптимизировать потоки людей в различных условиях, например, в торговых центрах, на стадионах, или в транспортных узлах [14, 15].

Методы моделирования транспорта применяются для анализа транспортных потоков, планирования транспортных сетей и оптимизации маршрутов. Они позволяют оценить изменения в дорожных сетях, вклад новых дорожных развязок, а также введения новых видов транспорта на общую эффективность транспортной системы [16]. Моделирование транспортных потоков позволяет предсказывать загрузку дорог, определять

необходимость строительства новых транспортных артерий, и улучшения работы общественного транспорта [17].

Хотя эти методы изначально не разрабатывались специально для интерактивных развлечений, они могут быть адаптированы и успешно применены в разработке игр и виртуальных сред. Рассмотрим некоторые из них.

1.2.1 Модель притягивающихся сил

Модель притягивающихся сил была создана в конце 1970-х годов профессором С. Оказаки и С. Мацусита. Эта микроскопическая и непрерывная модель. В ее основе лежит предположение, что пешеходы и препятствия обладают положительным электрическим зарядом, в то время как отрицательный заряд локализован в пункте назначения. Пешеходы подвергаются действию двух основных сил, стремясь достигнуть пункта назначения и избегая столкновений [19]. Первая сила, основанная на законе Кулона, зависит от размера заряда пешехода и расстояния до его цели. Вторая сила направлена на предотвращение столкновений с другими пешеходами или объектами, действуя через механизм ускорения. На примере, представленном на рисунке 1, ускорение a влияет на пешехода A, чтобы скорректировать его траекторию движения к направлению, параллельному линии AC, которая является касательной к окружности вокруг пешехода B. Эта окружность символизирует «личное пространство» человека, границы которого не принято нарушать. С. Оказаки определяет ее радиусом 60 сантиметров [20].

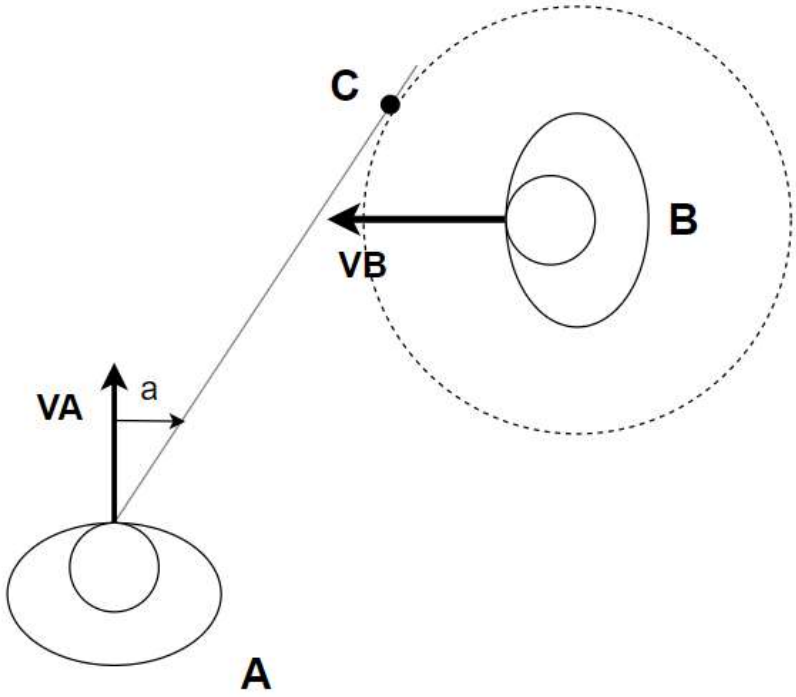

Рисунок 1 – Пример сил влияющих на пешехода

Достоинства модели:

– высокая скорость работы модели;

– простота реализации.

Недостатки модели:

– отсутствие возможности для задания уникальных характеристик каждого пешехода, таких как телосложение и другие личные данные;

– невозможность учитывать психологических и экстренных ситуаций, которые могут существенно изменить поведение человека;

– ограничение вариативности поведенческих паттернов пешеходов кроме двух основных: достижение цели и избегание столкновений;

– поскольку модель является микроскопической, симуляция выполняется для каждого отдельного пешехода, что в свою очередь является затратным по вычислительным ресурсам.

1.2.2 Модели обслуживания очередей

Модель обслуживания очередей – математический инструмент в теории организации очередей, позволяющий анализировать пешеходные потоки и их взаимодействие с сервисами [21]. Движение пешеходов в этой модели описывается через функции плотности вероятности. Пешеходы прибывают в сервис с определённой вероятностью, где их обслуживание происходит согласно вероятностному распределению. Если все каналы обслуживания заняты, пешеходы встают в очередь.

Основные элементы в системе организации очередей включают прибытие, обслуживающий механизм и саму организацию очереди, обычно реализуемую по принципу FIFO (первым пришёл – первым ушёл). Ласло Ловас предложил стохастическую модель для моделирования пешеходных потоков в зданиях как процесс организации сети очередей. В этой модели каждый пешеход рассматривается как отдельный элемент потока, взаимодействующий с другими элементами. Узлы в сети очередей соответствуют комнатам, а связи – дверям или переходам.

В процессе движения люди выбирают следующий узел согласно определённой вероятности, что приводит к формированию уравнения, описывающего изменения в количестве пешеходов от времени t до времени $t+h$. Из этого уравнения можно оценить, например, эффективность эвакуации людей из здания, рассчитав ожидаемое количество людей, оставшихся в узле, и ожидаемое число эвакуированных людей.

Модель обслуживания очередей позволяет учёным и инженерам изучать и оптимизировать потоки пешеходов, предотвращая перегрузку и минимизируя время ожидания, что особенно важно в местах с высокой плотностью движения людей, таких как транспортные узлы, стадионы, театры и торговые центры [22].

Достоинства модели обслуживания очередей:

– модель предоставляет количественные оценки, такие как среднее время ожидания в очереди и вероятность возникновения задержек, что помогает в планировании и улучшении потоков;

– модель может быть легко адаптирована к различным условиям и параметрам, таким как изменение интенсивности потока пешеходов или различных стратегий обслуживания;

– модель позволяет эффективно смоделировать и оптимизировать эвакуационные маршруты и процедуры, что улучшает безопасность в случае чрезвычайных ситуаций.

Недостатки модели обслуживания очередей:

– модель может не учитывать индивидуальное поведение пешеходов и их взаимодействие, например, социальное дистанцирование или предпочтения в выборе очереди;

– модель основывается на предположении о случайности интервалов времени прихода и обслуживания, что может не соответствовать некоторым реальным сценариям;

– простые модели очередей могут не отражать сложность реальных ситуаций, когда, например, возможны внезапные изменения в потоке людей или поведении;

– стохастическая природа модели требует использования сложных математических и компьютерных методов для получения точных результатов, что может затруднить понимание и применение модели без специальных знаний.

1.2.3 Клеточные автоматы

Клеточный автомат применяется как простой и эффективный метод моделирования пешеходного потока [11]. Весь интересующий нас пространственный участок представляется в виде сетки из клеток, каждая клетка может быть занята только одним пешеходом.

Движение пешеходов в модели клеточных автоматов смоделировано через изменение состояния клеток по определённым правилам, которые могут включать следующие пункты:

– если соседняя клетка свободна, пешеход может в неё переместиться;

– если клетка занята, пешеход должен подождать или изменить свой маршрут;

– пешеходы могут следовать за другими пешеходами, учитывая определённые правила следования на оптимальном расстоянии.

Правила, применяемые к клеткам, определяются в зависимости от цели моделирования и могут быть настроены для имитации различного поведения пешеходов и разных условий окружающей среды. Эти правила различаются в разных реализациях модели клеточных автоматов [23, 24], позволяя имитировать от простейших до сложных и реалистичных сценариев движения.

Достоинства модели клеточного автомата:

– модель легко визуализируется и понятна благодаря дискретному представлению пространства и времени;

– из-за простоты правил и дискретности, модели клеточных автоматов требуют относительно меньше вычислительных ресурсов;

– модель хорошо масштабируется и поддается распараллеливанию, что позволяет эффективно симулировать большие системы.

Недостатки модели клеточного автомата:

– модель имеет ограниченную реалистичность: она может не улавливать всю сложность пешеходного поведения в реальном мире;

– дискретность пространства и времени может привести к неестественности движения, так как в реальности пешеходы двигаются непрерывно;

– при слишком мелкой сетке количество клеток и вычислительная сложность быстро возрастают;

– модель очень зависима от начальных условий и правил, при их неправильном выборе результаты могут быть неверными и непредсказуемыми.

1.2.4 Газокинетическая и гидродинамическая модели

Газокинетическая модель в моделировании пешеходных потоков рассматривает пешеходов аналогично частицам газа, движущимся и взаимодействующими друг с другом в пространстве. Этот подход, вдохновлённый сходствами между динамикой газов и жидкостей и поведением пешеходов, обеспечивает фундамент для анализа потоков на макроскопическом уровне [25].

Ранние модели пешеходных потоков, взявшие вдохновение от гидродинамики и газокинетики, подчёркивают общие черты пешеходных потоков с жидкостями и газами. Одними из таких черт являются движение вокруг препятствий и динамику в условиях промежуточной плотности. Хендерсон, исследуя движение больших скоплений людей и находя аналогии между этими движениями и классическим газом, обнаружил соответствие функций распределения скорости в толпе с распределением Максвелла-Больцмана при условиях низкой плотности [26].

Эти наблюдения привели к разработке жидкодинамической теории пешеходного потока, где взаимодействия между пешеходами рассматриваются через аналогию со столкновениями частиц, в которых происходит обмен импульсами и энергией [27]. В этой модели, гомогенная толпа может быть описана с помощью кинетической теории газов, где такие понятия, как давление и температура, находят аналоги в контексте пешеходного движения. Температура в этой аналогии может соответствовать скоростным различием, связанным с распределением требуемых скоростей пешеходов, а давление – требованию двигаться против силы в определённом направлении.

Позднее жидкодинамическая теория потока эволюционировала в гидродинамические модели. В таких моделях пешеходный поток моделируется как поток одномерной сжимаемой жидкости [28]. Основные параметры этих моделей включают поток – количество людей, пересекающих конкретный участок за единицу времени; плотность потока – число людей на единицу площади в данный момент времени; скорость потока – средняя скорость движения людей в данной области. Между этими параметрами устанавливаются закономерности, которые описывают текущее состояние пешеходного потока. Учитывая сохранение потока и зависимость между скоростью и плотностью потока, утверждается, что при увеличении плотности потока (с увеличением числа людей на участке) средняя скорость движения снижается.

Достоинства газокинетической и гидродинамической моделей:

– подход позволяет анализировать большие группы людей как единую систему, что упрощает понимание общих тенденций и паттернов;

– модель особенно полезна для анализа потоков в местах с большой концентрацией людей;

– модель фокусируется на общих потоках и макроскопических характеристиках, что позволяет упростить вычисления.

Недостатки газокинетической и гидродинамической моделей:

– модель не учитывает индивидуальные стратегии, предпочтения и поведение пешеходов, что может снижать точность прогнозов в определённых контекстах;

– модель может быть менее эффективной при анализе потоков низкой плотности, где важно учитывать детальное взаимодействие между отдельными пешеходами;

– модель фокусируется на макроскопических параметрах, из-за чего детальное моделирование микроскопических взаимодействий и столкновений может быть недостаточно точным.

1.3 Анализ существующих подходов к моделированию неигрового персонажа в играх

Перед выбором глобальных моделей для симуляции нпс, важно проанализировать существующие методы симуляции, связанные с ними игровые механики и ограничения, которые могут влиять на процесс симуляции.

В ходе анализа было получено, что глобальная симуляция нпс чаще всего применяется в играх с открытом миром. Далее представлен примеры таких игр и используемые ими особенности симуляции нпс.

1.3.1 Marvel’s Spider-Man

Spider-Man – видеоигра в жанре экшн-приключения от третьего лица, разработанная компанией Insomniac Games и изданная Sony Interactive Entertainment. Действия игры происходят в открытым мире, схематично представляющим город Нью-Йорк.

Разработка реалистичного виртуального мегаполиса потребовала от создателей игры внедрения сложной системы для наполнения города жителями. Тротуары и пешеходные дорожки в игре размечены таким образом, чтобы задавать движение по определённым маршрутам, включая пешеходные переходы, а также чтобы настраивать и другие правила передвижения. При приближении игрока, вокруг дорог активируются специальные зоны, где начинают генерироваться нпс. Город поделён на квадраты со стороной 128 метров, что помогает определить дистанцию генерации нпс относительно игрока. Нпс, появляющиеся на тротуарах, двигаются в соответствии с заранее заданными направлениями [29]. При достижении конца специальной области, нпс сам решает куда ему пойти: повернуть на перекрестке и пойти по другой улице либо, если пешеход стоит перед светофором, дождаться зеленого света и перейти проезжую часть.

Это указывает на то, что поведение нпс моделируется с использованием микроскопической модели Такое моделирование применяется только в

пределах видимости игрока – за пределами этой зоны моделирование не выполняется.

1.3.2 The Sims 4

The Sims 4 – однопользовательская компьютерная игра в жанре симулятора жизни, четвёртая по счёту из серии игр The Sims, разработанная компанией Maxis и издаваемая Electronic Arts для Windows. В The Sims 4 нет чётко выраженного сюжета, а игровой процесс нелинеен и не имеет заданной конечной цели. Игрок контролирует до восьми собственноручно созданных или выбранных персонажей, направляя их на выполнение различных видов деятельности, таких, как удовлетворение собственных потребностей, зарабатывание денег и создание взаимоотношений с другими персонажами.

Для того чтобы иметь возможность выстраивать взаимоотношения с другими персонажами, управляемыми игрой, разработчикам пришлось создать систему, которая бы позволила моделировать жизнедеятельность нпс вне зоны видимости игрока. В [30] Rez Graham рассказал, что для решения такой задачи применялся разный уровень детализации: для нпс вне видимости игрока симуляция существенно упрощалась, симулируя лишь некоторые аспекты его жизни. Когда нпс появлялся у поле видимости игрока, симуляция его поведения обретала прежний вид. При переходе из низкого уровня детализации к высокому не симулированные аспекты жизнедеятельности нпс восстанавливались при помощи симулированных данных и различных графиков соответствий.

Такое решение позволило симулировать большое количество нпс как в соседних районах города, так и в соседних городах.

1.3.3 Middle-earth: Shadow of War

Middle-earth: Shadow of War – компьютерная игра в жанре Action/RPG с открытым миром, разработанная компанией Monolith Productions и выпущенная 27 сентября 2017. В Middle-earth: Shadow of War одной из ключевых особенностей является система «Nemesis», которая служит

инновационным способом макросимуляции в игровом мире. Система «Nemesis» применяется к оркам и урукам, являющимися врагами главного персонажа.

«Nemesis» создает диническую иерархию среди вражеских нпс в игре. У каждого орка есть свой ранг, амбиции и враги внутри своего общества. Орки помнят и реагируют на встречи с главным персонажем. Если орк выживает после столкновения с игроком, он может получить повышение или стать личным неприятелем (немезидой) главного персонажа. В течение игры орки могут сражаться между собой за власть и продвижение по служебной лестнице, что создает в игре постоянно меняющую социальную структуру. Действия игрока влияют на структуру власти внутри мира орков. Игрок может специально вмешаться в эти вражды, спроворовать орков друг на друга или помочь определённым оркам подняться в иерархии. Система «Nemesis» симулируется особыми «шагами», которые выполняются после того, как игрок выполняет действия, меняющие состояние этой системы [31].

Помимо симуляции социальной структуры нпс и различных реакций на события, система Nemesis не выполняют какую-либо другую симуляцию во время игры. Симулируемые нпс появляются в заданных заранее точках интереса. Перемещение между точками интереса и изменение состояния нпс происходят дескретно во время выполнения «шага» итерации.

Стоит также отметить, что помимо системы «Nemesis», при близком расстоянии к игроку, нпс симулируются при помощи микроскопической модели. Вероятно, микроскопической моделью является метод GOAP, который был использован в предыдущей части игры [32].

1.3.4 S.T.A.L.K.E.R.

Серия игр S.T.A.L.K.E.R. – это франшиза компьютерных игр в жанре шутера от первого лица с элементами RPG (ролевой игры) и survival horror, которая разработана украинской компанией GSC Game World. Франшиза насчитывает три выпущенные игры: S.T.A.L.K.E.R.: Shadow of Chernobyl

(2007), S.T.A.L.K.E.R.: Clear Sky (2008), S.T.A.L.K.E.R.: Call of Pripyat (2009), а также находящуюся в разработке S.T.A.L.K.E.R. 2: Heart of Chernobyl. Названная серия игр использует систему симуляции A-Life, которая позволяет симулировать различные аспекты жизни нпс в зоне отчуждения, такие как перемещение, бой с другими нпс, зарабатывание рейтинга и получения новых внутриигровых вещей.

По словам главного AI программиста Дмитрия Ясенева [33], реализация A-life довольно проста. В ней существуют два термина, которые характеризуют две модели поведения персонажа, отличающиеся деталями реализации: оффлайн и онлайн. Офлайн-поведение очень простое: персонаж не воспроизводит анимации или звуки, не управляет активно своим инвентарем, не строит подробные и плавные пути (хотя он строит пути в соответствии с глобальным навигационным графом,) и т.д. В отличие от этого, онлайн-поведение полностью детализировано. Таким образом, оффлайновое поведение можно считать LoD (уровнем детализации) онлайнового. Пока игрок действует на каком-то уровне, нпс на других уровнях находятся в оффлайне, то есть используют оффлайновое поведение. Более того, учитывая высокую плотность персонажей на уровне, не все персонажи на уровне игрока имеют онлайн-поведение, а только те, которые находятся в заданном радиусе от игрока (может варьироваться от уровня к уровню, но обычно это около 150 метров), или то, что задает дизайнер игры. Для этого система A-life следит за перемещениями игрока и оффлайн-персонажей и при необходимости переключает последних в онлайн/оффлайн.

Навигация для нпс в онлайн и оффлайн режиме отличается. В игре есть уровни, для каждого из которых строится отдельный навигационный граф, используемый персонажами для передвижения в онлайн-режиме. Он также называется подробным графом. Для каждого графа создается менее подробная версия, вершины которой могут быть соединены с вершинами графов других

уровней. Именно этот граф используется персонажами для навигации в оффлайн режиме.

У каждого неигрового персонажа всегда есть какая-либо цель. Для достижения этой цели нпс выполняет сгенерированные квесты. Во время выполнения квеста он может встретить врагов и сразиться с ними. В оффлайн-режиме это симулировалось как пошаговая стратегия: противники по очереди делали ходы, результат рассчитывался по формуле и случайному фактору. В результате один противник мог убежать от другого, иногда персонажей убивали, иногда они даже не замечали друг друга или решали избежать столкновения. Если персонажи были дружественны друг к другу, они могли торговать, руководствуясь разработанной схемой торговли. Все это работало как в оффлайн, так и в онлайн-режимах.

1.3.5 Assassin's Creed Unity

Assassin’s Creed Unity – компьютерная игра из серии Assassin’s Creed в жанре action-adventure, которую разработала компания Ubisoft в 2014 году. Действие игры происходит во время Французской революции. Разработчики из Ubisoft представили один из самых живых и населенных виртуальных миров на тот момент. На улицах Парижа игроки сталкиваются с массовыми толпами людей, которые реагируют на события в мире игры, например, на прибытие революционеров или на акты насилия, совершенные игроками.

Для отображения больших толп людей была разработана специальная система под названием «Bulk», которая является своеобразным менеджером уровня детализации для нпс [34]. Помимо изменения детализации полигональной сетки данный менеджер также меняет систему симуляции персонажа. Если персонаж далеко от игрока, то система заменит этого нпс на «куклу», которая проигрывает некоторые анимации и передвигается в заданном направлении, избегая коллизии с другими «куклами» и нпс. При приближении «куклы» к персонажу она подменяется менеджером на настоящего нпс, который симулируется и может давать реакцию на действия

игрока. При отдалении от игрока персонаж подменяется обратно на «куклу». Подмена выполняется достаточно быстро благодаря заранее созданному пулу с нпс.

В определенной дальности от игрока система уничтожает куклы для экономии ресурсов. То есть описанный подход исключает макросимуляцию нпс.

1.3.6 Вывод

Можно заметить, что в некоторых играх используются макромодели, которые симулируют определенную часть игрового процесса. Все эти модели занимаются упрощенной симуляцией нпс и являются своего рода отдельным упрощенным уровнем детализации. Более того описанные модели разработаны с учетом игровых проектов, в которых они применяются. Также были рассмотрены игры, в которых макромоделированние не применяется в угоду сохранения ресурсов. В них разработчики стараются не симулировать нпс, которых игрок не видит или которые не смогут повлиять на игровой опыт.

Оба подхода возможны в играх от третьего и первого лица, где отсутствует возможность посмотреть на весь город сразу. Однако данные подходы делают невозможным моделирование настоящего перемещения нпс по городу, исключается возможность встретить одного и того же персонажа, если он уйдет достаточно далеко.

В случае, когда нпс все же симулируются, разработчики стараются подобрать оптимальное количество персонажей чтобы избежать чрезмерной нагрузки.

2 АЛГОРИТМ ПРЕДЛОЖЕННОЙ МОДЕЛИ

2.1 Формализация критериев разрабатываемой модели

Благодаря проведенному ранее исследованию можно составить ряд требований к модели, предназначенной для симуляции неигровых персонажей в видеоиграх:

1) нпс, находящиеся в непосредственной близости от игрока, должны симулироваться каждый индивидуально;

2) для создания реалистичного эффекта «жизни» в игровом мире следует симулировать всех нпс;

3) модель должна иметь производительность, позволяющую получать результат симуляции не реже чем раз в 0.5 секунд.

Первый критерий предполагает использование микроскопической модели для симуляции нпс в непосредственной близости к игроку. Однако, для моделирования всего игрового мира микроскопические модели могут не справиться с требованиями по производительности, указанными в третьем критерии.

Это обстоятельство подталкивает к созданию комбинированной модели. В такой модели в зоне близкой к игроку будет использоваться детальная микроскопическая модель, тогда как за её пределами — более быстрая и менее детализированная модель для улучшения общей производительности.

2.2 Сопоставление и выбор микроскопической и макроскопической модели

В качестве микроскопической модели было решено выбрать метод проектирования искусственного интеллекта на основе деревьев поведений. Это решение продиктовано оптимальным соотношением достоинств и недостатков рассмотренного метода: он хорошо подходит для проектирования простых нпс и имеет встроенную поддержку в игровой движок Unreal Engine. Также на выбор оказало влияние наличие опыта проектирования нпс с использованием деревьев поведений у исследователя.

Также ранее были рассмотрены модели, с помощью которых можно выполнять симулирование пешеходных и транспортных потоков. Было выявлено что из-за совокупностей достоинств и недостатков газокинетические макроскопические модели и микроскопические модели на основе клеточных автоматов наиболее подходят для моделирования поведения неигровых персонажей в игровых мирах. Необходимо провести анализ этих подходов, чтобы определить их преимущества в контексте применения в видеоиграх, где ключевым аспектом является производительность модели. Это обусловлено требованиями к игровым приложениям, которые должны поддерживать определённую частоту кадров в секунду, что является критическим для обеспечения плавности и комфорта игрового процесса.

Газокинетические и другие макроскопические модели обобщают поведение отдельных нпс, преобразуя это в общие параметры, такие как поток, плотность и средняя скорость. Это снижает требования к вычислительным ресурсам, однако устраняет возможность моделирования индивидуального поведения каждого нпс. Это может быть ограничением, учитывая современные требования к искусственному интеллекту в играх, где часто используется поведение нпс по расписанию для создания более реалистичного и живого игрового мира. Такой подход давно применяется в игровой индустрии [35, 36].

При использовании макроскопической модели возникает вопрос о распределении потока нпс при разветвлениях дорог. Простейшие модели могут использовать заранее заданные пропорции для распределения потоков, тогда как более сложные методы могут учитывать изменения в зависимости от времени суток или текущих игровых событий.

С другой стороны, микроскопические модели, включая те, что основаны на клеточных автоматах, рассматривают каждого нпс как независимую единицу с уникальными целями и маршрутами. Это исключает необходимость дополнительного распределения потоков на узлах и позволяет каждому нпс

самостоятельно определять свой путь. Однако такой подход требует значительных вычислительных ресурсов, что может стать проблемой при ограниченных ресурсах. Этот недостаток можно компенсировать, использовав графические процессоры для выполнения симуляции модели, поскольку вычисление клеточных автоматов поддаются распараллеливанию. Однако это в свою очередь может снизить ресурсы, необходимые для системы рендеринга. Сегодня максимальная утилизация ресурсов графического процессора особенно важна, учитывая растущее использование экранов высокого разрешения [37, 38].

Таким образом, в контексте интерактивных развлечений газокинетические и гидродинамические макроскопические модели могут предложить оптимальный баланс между производительностью и реализмом моделирования поведения нпс, учитывая все технологические ограничения и требования к частоте кадров для плавного игрового процесса.

2.3 Структура и вычисление гидродинамической модели

Существует большое количество макроскопических моделей, использующихся для моделирования пешеходного потока. Все они имеют свои уникальные характеристики. В контексте видеоигр, где производительность является ключевым критерием, предпочтение следует отдавать моделям, которые требуют минимальных вычислительных ресурсов. Из всех гидродинамических моделей такой является LWR модель.

Использование гидродинамической модели включает применение уравнения непрерывности, являющимся дифференциальным уравнением в частных производных. Оно включает в себя следующие макроскопические параметры:

- скорость (V);
- плотность (ρ);
- поток (Q).

В основе любой гидродинамической модели действует следующее уравнение (1), которое связывает поток с плотностью и скоростью [39]:

$$Q = p * V. \quad (1)$$

Однако, для комплексности модели необходимо ввести дополнительные уравнения. Лайтхилл и Уитхэм, а также Ричардс предложили статические соотношения (2) и (3):

$$V(x,t) = V_e(p(x,t)). \quad (2)$$

Или

$$Q(x,t) = Q_e(p(x,t)). \quad (3)$$

Эти соотношения предполагают, что скорость мгновенно адаптируется к изменениям плотности потока, как в стационарных условиях, так и в динамичных. Аналогичное утверждение справедливо и для потока.

При анализе участка маршрута как секции с однородным движением, предполагается, что приток и отток пешеходов равномерно распределены, что приводит к следующему выражению (4):

$$\frac{\partial p}{\partial t} + \frac{\partial Q}{\partial x} = 0. \quad (4)$$

Вставляя ранее упомянутое соотношение, получаем каноническую форму (5) модели LWR, [39]:

$$\frac{\partial p}{\partial t} + \frac{\partial Q_e(p)}{\partial p} * \frac{\partial p}{\partial x} = 0. \quad (5)$$

Зависимость потока пешеходов от плотности потока обычно изображается на фундаментальной диаграмме. Наиболее простой в моделировании является обратная диаграмма V-формы, представленная на рисунке 2. В этой диаграмме существует прямая линейная зависимость между плотностью и потоком до достижения оптимальной плотности p_{opt} (режим свободного движения), после чего наблюдается обратная линейная зависимость, характеризующая режим затора.

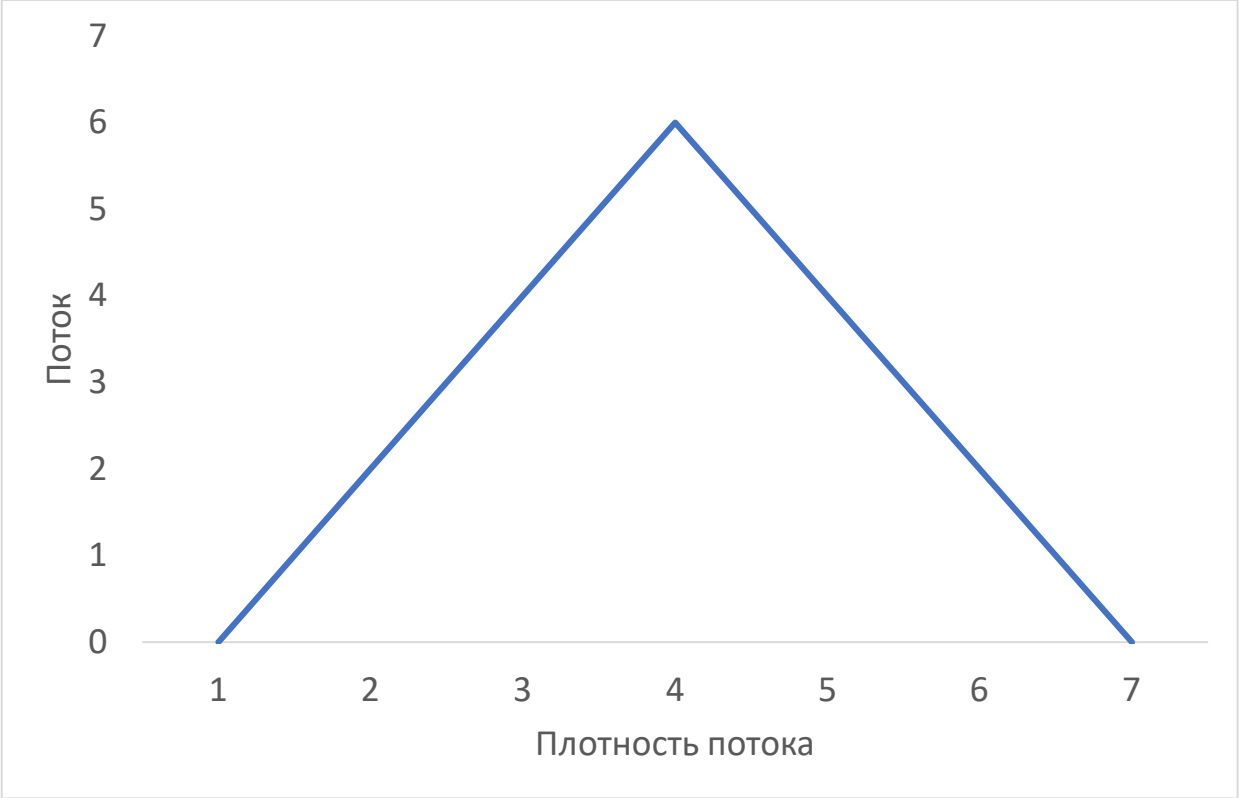

Рисунок 2 – Фундаментальная обратная диаграмма V-вида

Следствием этого является необходимость определения значения оптимальной плотности p_{opt} и значения потока при данной плотности Q_{opt}, которое может быть получено, имея значение скорости при данной плотности V_{opt}, используя уравнение потока для гидродинамических моделей.

Таким образом, для описания участка, по которому движутся пешеходы, требуются параметры, аналогичные тем, что используются для моделирования дорожного трафика, но адаптированные под специфику пешеходных потоков.

Таким образом, для описания участка пути, по которому движутся пешеходы, требуется установление параметров, подобных тем, что используются в моделировании дорожного трафика, но адаптированных под специфику пешеходных потоков. В контексте данной работы, под участком пути понимается ребро между узлами путевого графа игрового мира с однонаправленным движением пешеходов, на протяжении которого поток равномерный. Участок пути ограничен узлом путевого графа или участком пути с другими значениями параметров p_{opt} или V_{opt}. Для равномерного распределения потока между смежными участками пути предлагается использовать коэффициент пропорциональности k_{i-j} для каждой пары смежных пешеходных участков i, j. При этом сумма всех коэффициентов, исходящих из участка пути i должна быть равна единице.

Таким образом, участок пути описывается следующими параметрами:

– количество персонажей, начинающих своё движение за единицу времени на единицу протяжённости пути N_{in};

– количество персонажей, заканчивающих своё движение за единицу времени на единицу протяжённости пути N_{out};

– текущая плотность потока p;

– оптимальная плотность потока p_{opt};

– текущий поток Q;

– оптимальная скорость движения V_{opt};

– длина участка пути $size$;

– множество соседних участков пути, из которых входят персонажи на данный участок U_{in};

– множество соседних участков пути, на которые персонажи покидают данный участок U_{out};

– множество коэффициентов пропорциональности для соседних участков пути U_{kj};

Схема с представленным параметрами изображена на рисунке 3.

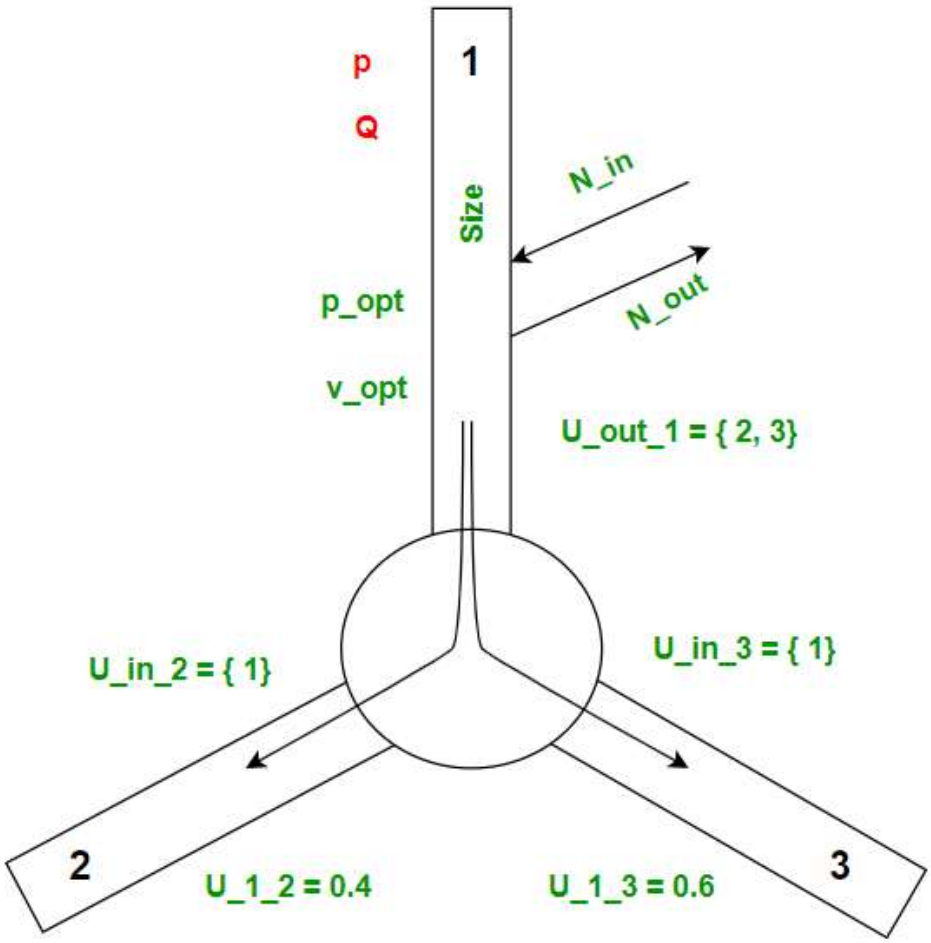

Рисунок 3 – Пример системы из трех участков, соединённых узлом. Зеленые значения – вводимые параметры, красные – рассчитанные

Поскольку каждый участок пути может разветвляться, то на каждом шаге моделирования i текущий поток Qi можно описать следующим уравнением (6):

$$Q_i = \min\left(leave(p_i), enter(U_{out})\right). \quad (6)$$

А текущую плотность потока можно выразить уравнением (7):

$$p_i = p_{i-1} + \frac{dt}{size} * (flow_{sum}(U_{in}) - Q_i) + N_{in} * dt - N_{out} * dt \quad (7)$$

где *dt* – шаг моделирования времени; *leave* – функция, которая определяет максимальный поток пешеходов, который может выйти из текущего участка пути; *enter* – функция, которая определяет какой поток пешеходов способны принять соседние участки пути; $flow_{sum}$– функция, определяющая поток пешеходов, поступающий из соседних участков пути.

Когда плотность потока на определённом участке пути превышает заданное оптимальное значение, этот участок способен поддерживать максимально возможный поток Q_{opt}. В случаях, когда плотность потока ниже этого порога, пропускная способность участка будет определяться по коэффициенту, который связывает текущий поток p с его оптимальным значением p_{opt}, что отражает прямую линейную зависимость между этими параметрами. Эта зависимость формулируется в уравнении (8).

$$leave(p_{i-1}) = \begin{cases} Q_{opt}, p_{i-1} > p_{opt} \\ Q_{opt} * \frac{p_{i-1}}{p_{opt}}, p_{i-1} \leq p_{opt} \end{cases}. \quad (8)$$

Для нахождения размера потока, который могут принять смежные участки пути, необходимо сначала оценить пропускную способность каждого участка. Если плотность потока на участке ниже оптимальной, участок может принять максимально возможный поток Q_{opt}. Иначе, при превышении p_{opt} поток, принимаемый участком пути, будет уменьшаться пропорционально разнице между текущей плотностью p и p_{opt}. Этот обусловлено прямой линейной зависимостью между плотностью и потоком. Описанная зависимость представлена в уравнении (9).

$$flow_j = \begin{cases} Q_{opt}, p_{i-1} \leq p_{opt} \\ Q_{opt} - Q_{opt} * \dfrac{p_{i-1} - p_{opt}}{p_{opt}}, p_{i-1} > p_{opt} \end{cases}. \quad (9)$$

Так как каждый участок пути получает пешеходные потоки из нескольких смежных участков, предлагается учитывать сумму потоков, исходящих из смежных участков, умноженных на соответствующие коэффициенты пропорциональности. Если суммарный поток равен или меньше входящего потока, то потоки учитываются в полном объёме. В противном случае потоки распределяются пропорционально их доле в общей сумме. Этот метод требует создания дополнительного поля с массивом потоков, готовых к передаче между смежными участками $U_{flo\ \ j}$.

Схема с представленным новыми параметрами изображена на рисунке 4.

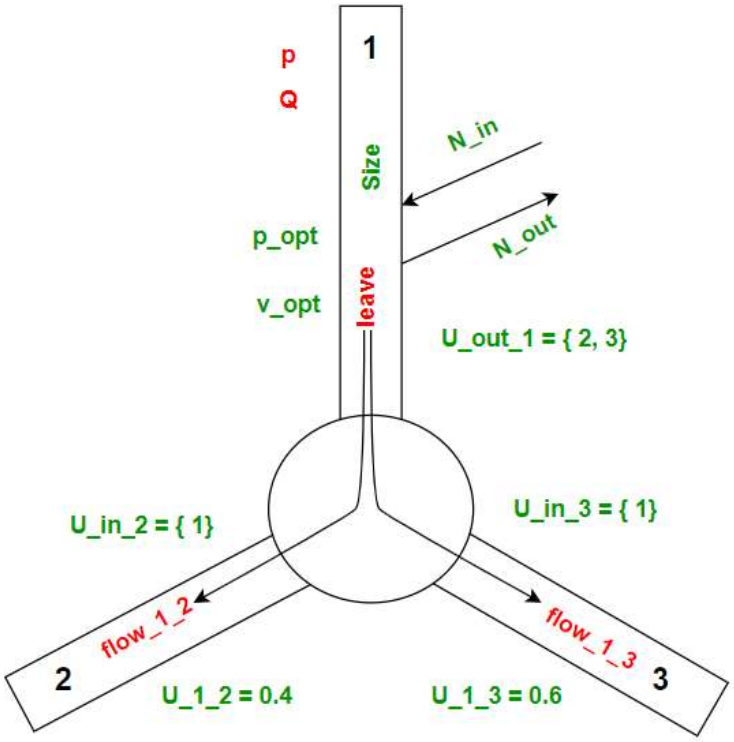

Рисунок 4 – Пример системы из трех участков, соединённых узлом с отображенными исходящими потоками. Зеленые значения – вводимые параметры, красные – рассчитанные

Поток пешеходов, который могут принять смежные участки пути *enter* равен сумме потоков, готовых к передаче этими смежными участками. Описанная зависимость представлена в уравнением (10).

$$enter = \sum_{i=0}^{n} U_{flow_i}. \quad (10)$$

Поскольку для вычисления $flow_{su}$ и U_{fl} $_j$ необходимо знать значения *leave* и p_{i-1} на всех участках пути, и для расчета *enter* необходимо иметь значение U_{flow_j} для всех участков, то последовательность вычислений выглядит следующим образом:

- рассчитать p_{i-1} и *leave* для каждого участка пути;
- рассчитать U_{flow_j} и $flow_{sum}$ для каждого участка пути;
- рассчитать Q, p, *enter* для каждого участка пути.

3 РЕАЛИЗАЦИЯ ПРЕДЛОЖЕННОЙ МОДЕЛИ

В рамках предыдущего этапа исследований был разработан алгоритм для вычисления всех параметров макроскопической модели симуляции нпс на каждом шаге моделирования. В этой главе представлены детали реализации макроскопической и микроскопической моделей, а также их интеграция.

3.1 Разметка игрового мира

Для работы гидродинамической модели и микроскопической модели необходима особая разметка игрового мира. Основой такой разметки служит навигационный граф, который представляет собой сеть узлов, расставленных по игровому пространству и соединённых рёбрами, символизирующими участки пути макромодели с заданными параметрами. Эта разметка напоминает навигационный граф, используемый в системе Alife [33] для упрощённого перемещения нпс по миру.

Каждый узел навигационного графа соответствует определённой точке в игровом мире, которая может быть частью пути, зоной интереса или перекрестком. Рёбра графа определяют возможные пути перемещения между этими узлами.

Для представления узла графа был создан класс Node, который содержит настраиваемый массив объектов класса Edge и состоит из кубического коллайдера. Также Node является наследником класса Actor чтобы его можно было разместить на игровой сцене. Это позволяет определять является ли узел в области микромоделирования. Класс Edge представляет собой ребро навигационного графа. Он содержит редактируемые поля, необходимые для симуляции в макромодели и является наследником класса UStruct. Для разметки игрового мира необходимо расставить объекты Node на игровой сцене и в каждом поставленном объекте добавить необходимое количество экземпляров Edge с нужными параметрами. Рёбра должны соответствовать реальным или логическим путям перемещения нпс в игровом мире.

3.2 Детали реализации микромодели и макромодели

В процессе моделирования, когда нпс находятся зоне микромоделирования, они будут перемещаться по навигационному мешу игрового мира от узла, из которого они появились к узлу, который назначен им следующей точкой назначения. При достижении точки назначения, которая находится внутри области микромоделирования, нпс получает следующую точку назначения. Выбор следующей точки назначения происходит случайно с распределением, заданным предварительно. Заданные коэффициенты соответствуют параметру «множество коэффициентов пропорциональности для смежных участков пути» которые обозначены на рисунке 2 как U_1_2 и U_1_3. Такое поведение позволит микромодели поддерживать распределение нпс подобно тому, как это делает макромодель. Появляясь нпс наследует скорость и другие параметры от соответствующего участка пути.

В случае, когда нпс покидает область микромоделирования, он уничтожается и агрегируется в соответствующий участок пути макромодели.

Для реализации разработанного в пункте 2.4 алгоритма был создан класс Macromodel, который выполняет шаги макромоделирования каждую итерацию игрового цикла приложения. Код создания объекта Macromodel перебирает все объекты класса Node и запоминает ссылки на объекты класса Edge. Это позволяет изменять параметры участков пути во время выполнения программы.

3.3 Интеграция моделей

Поскольку микиромоделирование необходимо проводить в определенной области вокруг игрока, то для этой цели была создана компонента NodeDetector, состоящая из кубического коллайдера, которая прикрепляется к персонажу и помогает определять находящиеся рядом узлы. Размеры коллайдера являются изменяемым параметром и могут настраиваться в зависимости от потребностей.

При попадании узла в область микромоделирования все участки пути, связанные с этим узлом, переключаются в режим микромоделирования. В этом режиме участки пути моделируют перемещение нпс путем их создания и задания им точки назначения. Также компонент NodeDetector необходим для для того, чтобы определять покидающих область микроскопического моделирования нпс. При выходе нпс за эту область происходит уничтожение этого нпс, вычисление следующего участка пути и увеличения его параметра концентрации на единицу.

При старте приложения, в стартовых узлах участков путей, которые симулируются микроскопическим образом, конструируются нпс число которых равно концентрации соответствующего участка пути. При этом концентрация этих участков пути обнуляется и как следствие исходящий поток из этих участков пути тоже обнуляется. При последующей симуляции при превышении в участке пути концентрации единицы необходимо конструировать нового нпс и уменьшать концентрацию на единицу.

Макроскопические параметры участков путей, которые пересекаются с микроскопической областью необходимо изменить специальным образом. Из-за того, что перемещение нпс в микроскопической области выполняется передвижением от узла к узлу, то необходимо считать значение параметра leave равным нулю. Это приведет к обнулению значения потока и как следствие исключит из симуляции уменьшение плотности потока. Исключать участки пути, которые моделируются микроскопическим образом из макромоделирования полностью нельзя, поскольку прекращение изменения плотности потока приведет к тому, что в микроскопической области моделирования перестанут появляться новые нпс.

Поскольку конструирование нпс происходит в определенном радиусе вокруг стартового узла участка пути, то может произойти ситуация, когда свободного места в этой области не будет, например в случае, когда нпс слишком медленно идут или если на дороге появилась преграда. В таком

случае конструирование нового нпс откладывается до момента, когда появится свободное место. Такое решение позволяет накапливать значение макроскопического параметра и влиять на соседние участки пути.

Все описанные выше классы были реализованы на языке программирования C++ и Blueprints в игровом движке Unreal Engine.

3.4 Отслеживание и симуляция важных нпс

В играх зачастую нпс должны обладать некоторым состоянием, которое они сохраняют и как-то изменяют даже если игрок находится далеко от них. Например, это могут быть странствующие торговцы, положение которых меняется даже если игрок находится слишком далеко. В дальнейшем игрок может встретить такого странствующего торговца в другой части игрового мира. При этом есть системы, где помимо положения также меняется и ассортимент этих торговцев. Таких нпс принято называть важными поскольку они важны для игрового опыта.

Поскольку использование макроскопической гидродинамической модели не позволяет отслеживать отдельных нпс и выполнять их дополнительную логику, то было решено дополнить макроскопическую модель, добавив в нее модуль отслеживания и симуляции важных нпс.

Поскольку мы знаем скорость передвижения по участку пути и длину этого участка пути, то вычисление положения нпс на этом участке пути не составляет сложности. При превышении положения нпс длины участка пути происходит выбор

В макромодели важные нпс симулируются с использованием алгоритма, который вычисляет их новое положение на основе заданных параметров скорости и размера участка пути, на котором эти нпс находятся. Это позволяет отслеживать движение нпс по игровому миру в макромодели. Когда нпс достигает конца участка пути, выполняется подбор следующего участка пути согласно предопределённому распределению в макромодели. Такой подход обеспечивает непрерывность движения нпс по игровому пространству.

На каждом шаге симуляции макромодели для каждого важного нпс, который симулируется макроскопическим образом, вызывается событие, подобное TickEvent. Это выполнить какую-либо заданную логику. Такой подход позволяет не просто выполнять симуляцию передвижения нпс, но и создавать некоторые упрощенные поведения в макромодели, привнося тем самым в игровой мир больше динамики и жизни.

Появление и уничтожение важных нпс на границе макроскопического и микроскопического моделирования происходит также, как и для обычных нпс, за некоторым исключением. При уничтожении важный нпс перестает быть активным, при этом у него выключается все системы: от коллизии до ai контроллера. При создании важного нпс происходит активация всех его систем, далее нпс переносится в позицию узла, где необходимо его создать. При создании нпс в узле навигационного графа создаются сначала важные нпс, а затем обычные. В случае если важных нпс было больше, чем концентрация в данном участке пути, то происходит создание всех важных нпс с переходом параметра концентрации в отрицательное значение. При последующей симуляции прирост концентрации скомпенсирует это отрицательное значение. Такое решение позволяет сохранять суммарную концентрацию в макроскопической модели, что особенно важно в случае настройки графа при которой прирост и убытие концентрации не используются.

4 ТЕСТИРОВАНИЕ РАЗРАБОТАННОЙ МОДЕЛИ

4.1 Анализ области применения модели

Прежде чем приступить к выполнению тестирования разработанной модели необходимо установить условия, при которых ее можно применять.

Прежде всего стоит отметить ряд преимуществ, которыми обладает разработанная модель. Основным является возможность симулировать нпс на всей игровой сцене с относительно небольшими вычислительными затратами для работы макроскопической модели. Также применение двухуровневой модели позволяет управлять потоком нпс в зависимости от игрового времени или геймплейных особенностей.

Поскольку микроскопическое моделирование нпс выполняется в определенной области вокруг игрока, а во вне этой области нпс отсутствуют, то одним из важнейших ограничений является ограниченный обзор игровой сцены камерой персонажа. Например, недопустимым является подход, в котором необходимо показывать игроку большой объем пространства. Часто такой подход встречается в играх стратегиях с видом сверху.

Наилучшим образом модель можно применить в играх с видом от первого или третьего лица. Пример таких игр представлен на рисунке 5 и рисунке 6.

Рисунок 5 – Игра, в которой можно эффективно применить разработанную модель. Watch Dogs: Legion

Рисунок 6 – Игра, в которой можно эффективно применить разработанную модель. Grand Theft Auto VI

Также модель можно успешно применить в игре с видом сверху при условии, что камера направлена вниз и видит небольшое количество узлов. В таком случае область микроскопического моделирования должна покрывать видимое камерой пространство. Если в игре отсутствует персонаж игрока, то область микроскопической симуляции должна определяться теми узлами, которые попадают в область видимости камеры. Микроскопическое моделирование в таком случае выполняется только для видимых узлов, остальные узлы симулируются макроскопической моделью. Пример таких игр представлен на рисунке 6 и рисунке 7.

Рисунок 7 – Игра с видом от третьего лица, в которой можно эффективно применить разработанную модель. Diablo IV

Рисунок 8 – Игра с видом от третьего лица без персонажа игрока, в которой можно эффективно применить разработанную модель. Stellaris

Еще одним ограничением разработанной модели является невозможность полноценной симуляции большого количества важных нпс в макроскопической модели. Это происходит из-за того, что выполнение логики, сравнимой по сложности с логикой выполнения в микроскопической области, может нивелировать работу макроскопической модели. Особенно сильно это может быть заметно в больших игровых мирах с большим количеством нпс. Однако наличие упрощенной логики симуляции все еще позволяет нпс из макроскопической модели участвовать в игровой логике. Например, использование упрощенной логики симуляции активно используется в рассмотренной ранее игре S.T.A.L.K.E.R. Также эта особенность играет большую роль в геймплее игры Mount & Blade II: Bannerlord, представленной на рисунке 9.

Рисунок 9 – Игра, в которой невозможно использование разработанной модели. Mount & Blade II: Bannerlord

Таким образом можно вывести следующие ограничения для использования разработанной модели:

– камера игрока всегда видит ограниченную часть игровой сцены;

– при большом количестве нпс в игровом мире, используется упрощенная логика макроскопической симуляции.

4.2 Требования к игровой сцене

Так как модель предусматривает тестирование с большим количеством нпс, то необходимо обеспечить достаточно пространства для их размещения. Большой размер сцены позволяет оценить производительность и эффективность модели в условиях, приближенных к реальным.

Кроме того, игровая сцена должен иметь навигационный меш, который является критически важным элементом для функционирования микроскопической модели симуляции. Навигационный меш позволяет нпс, которые симулируются микроскопической моделью, строить маршруты и перемещаться по ним. Кроме того, навигационный меш должен учитывать размещенные на сцене препятствия.

Наилучшей топологией для игровой сцены при тестировании данной модели является городская среда. Городские сцены предоставляют четко

определенные улицы и перекрестки, что упрощает создание и настройку навигационного графа модели. Помимо этого, симуляция большого количества нпс в городской среде является достаточно реалистичным сценарием, который многие разработчики воссоздают в игровых проектах.

4.3 Выбор ассетов для игровой сцены

Для успешного тестирования разработанной модели необходимо найти или создать сцену какого-либо реально существующего города на основе ГИС данных. Для этого была найдена модель района города Лондон, Англия [40].

Модель состоит из мешей, схематично моделирующих здания и объектов инфраструктуры, таких как пешеходные тротуары, дороги и мосты. На рисунке 10 представлена найденная модель. Общая площадь сцены составляет 1 километр квадратный. Также игровая сцена обладает уже построенным навигационным мешем, что является большим достоинством поскольку разметка на таких больших площадях требует больших трудозатрат.

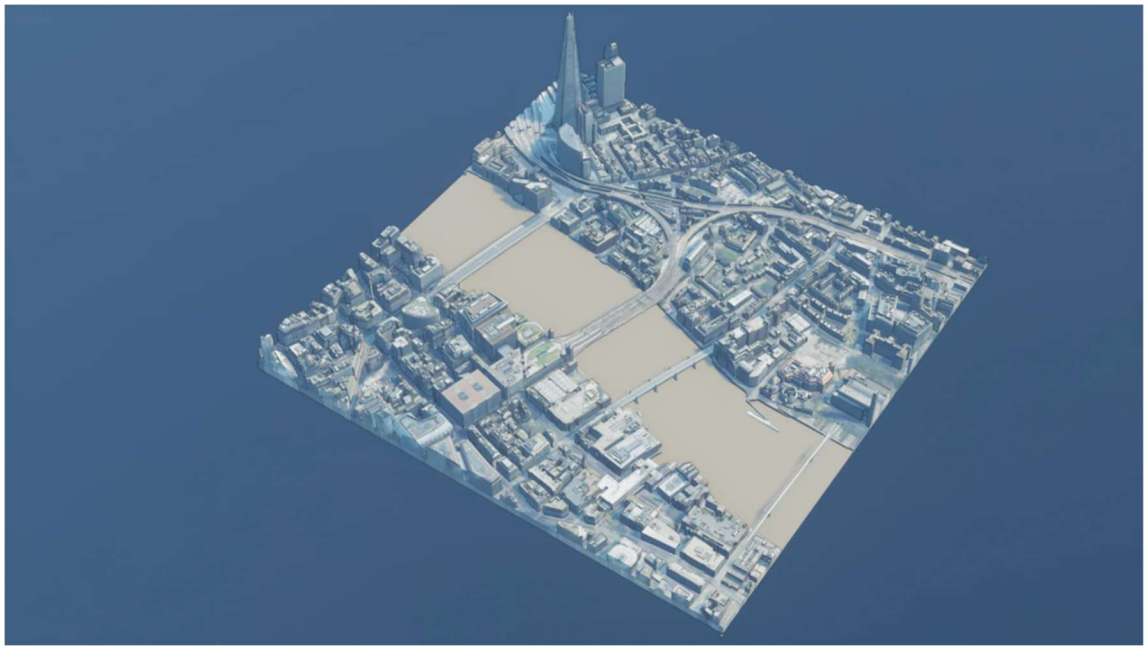

Рисунок 10 – Модель района Лондона

4.4 Разметка игровой сцены

Разметка путей на этой сцене позволит смоделировать реальный сценарий использования модели. На рисунке 11 показан пример разметки игровой сцены.

Рисунок 11 – Размеченный участок игровой сцены. Зеленым показаны границы навигационного меша, желтым выделены узлы модели

В ходе выполнения разметки игровой сцены был выведен ряд рекомендаций, которые могут помочь сократить трудозатраты при разметке и нивелировать недостатки, вызванные устройством модели. Далее представлен ряд этих рекомендаций:

1) Расставлять узлы необходимо как можно дальше от мест, в которых нежелательно нахождение нпс. В данном случае узлы расставлялись как можно ближе к зданиям чтобы исключить выход нпс на проезжую часть дороги. Также этот недостаток можно решить при помощи отделения навигационного меша для нпс от общего навигационного меша, но такой подход является трудозатратным при большой площади игровой сцены.

2) В текущей реализации нпс может выбрать путь, обратный которому он следовал до этого. Если такое поведение является

нежелательным, то необходимо построить две цепочки путей, ребра которых будут направлены в разные стороны.

3) Поскольку дорога может идти достаточно далеко по прямой, то узлы необходимо расставлять за какими-либо визуальными преградами чтобы исключить появление нпс и его пропажу прямо на глазах. Этот недостаток также можно устранить, применяя особые настройки отрисовки на дальних расстояниях и визуальные эффекты.

4.5 Методика оценивания разработанной модели

На предыдущем этапе работы была выполнена разметка игровой сцены, удовлетворяющей заданные требованиям. Теперь предстоит выполнить сравнение разработанной модели с уже аналогами.

Для сравнения на размеченной игровой сцене будут функционировать разработанная двухуровневая модель и одноуровневая модель, аналогичная используемым в играх. Обе модели необходимо сравнить по заданным критериям:

1) потребление вычислительных ресурсов при низкой плотности нпс (менее 1 нпс на 500 м);

2) потребление вычислительных ресурсов при средней плотности нпс (1-4 нпс на 500 м);

3) потребление вычислительных ресурсов при высокой плотности нпс (5 и более нпс на 500 м);

Как показатель производительности будет использоваться частота кадров приложения. Для минимизации влияния внешних факторов измерения будут повторены несколько раз.

Для проведения сравнения полученной модели игровая сцена функционировала сначала только с одноуровневой микроскопической моделью на основе деревьев поведений, а затем с полученной двухуровневой. В случае одноуровневой модели, необходимое количество нпс будет рассчитываться путем умножения заданной плотности на общую длину всех

путей (21112.82 м), и последующим равномерным размещением нпс по всей сцене. Для двухуровневой модели, заданная плотность нпс будет устанавливаться как макроскопический параметр для всех участков пути.

4.6 Результаты эксперимента

Каждый эксперимент был проведен пять раз, и для анализа данных использовалось медианное значение. Погрешность измерений составила 1 мс. Полученные результаты представлены в таблицах 1, 2 и 3.

Таблица 1 – Результаты эксперимента при низкой плотности

Плотность нпс	Одноуровневая модель	Разработанная модель
1 нпс на 3200 м.	7.42 мс	7.42 мс
1 нпс на 1500 м.	7.42 мс	7.42 мс
1 нпс на 750 м.	9.33 мс	7.42 мс

Таблица 2 – Результаты эксперимента при средней плотности

Плотность нпс	Одноуровневая модель	Разработанная модель
1 нпс на 400 м.	10.4 мс	7.47 мс
1 нпс на 250 м.	13.9 мс	7.48 мс
1 нпс на 200 м.	16.6 мс	7.5 мс
1 нпс на 150 м.	20 мс	8.5 мс
1 нпс на 120 м.	24.2 мс	8.6 мс

Таблица 3 – Результаты эксперимента при высокой плотности

Плотность нпс	Одноуровневая модель	Разработанная модель
1 нпс на 120 м.	27.7 мс	11.1 мс
1 нпс на 80 м.	32.3 мс	12.9 мс

На основе представленных данных можно сделать следующие выводы:

1) При низкой плотности потока нпс различия в производительности практически отсутствуют. Это объясняется тем, что общее количество нпс остается невысоким и не оказывает значительной нагрузки на вычислительные ресурсы, обеспечивая хорошую производительность обеих моделей.

2) При средней плотности нпс производительность одноуровневой модели линейно ухудшается с увеличением количества нпс, видно существенное снижение кадровой частоты. В то же время, производительность двухуровневой модели остается стабильной благодаря минимальному влиянию микроскопического моделирования на общую производительность системы.

3) При высокой плотности нпс обе модели демонстрируют снижение производительности, однако двухуровневая модель показывает меньшее падение производительности и более эффективное использование вычислительных ресурсов даже в самых сложных условиях.

4.7 Оценка симуляции важных нпс

Поскольку модель предоставляет возможность симуляции важных нпс в области макроскопической симуляции, то также необходимо оценить какое количество вычислительных ресурсов потребляется для симуляции разного количества важных нпс.

Для оценки были произведены замеры времени выполнения симуляции важных нпс в макроскопической области при разном количестве важных нпс.

Замеры проводились на персональном компьютере с процессором Intel Core i5-7200U с тактовой частотой 2.5 гигагерц. Результат представлен в таблице 4.

Таблица 4 – Время выполнения макросимуляции важных нпс.

Количество нпс	Время выполнения
100	0.06 мс
200	0.1 мс
300	0.13 мс

По результатам замеров можно сделать вывод что с увеличением количества нпс пропорционально растет время симуляции важных нпс. Этот факт необходимо учитывать применяя данную модель.

4.8 Потенциальные улучшения разработанной модели

В ходе выполнения разметки игровой сцены и проведения эксперимента был выявлен ряд недостатков, которые планируется исправить в будущем.

Первым из недостатков является усложненная настройка разметки. Усложнена разметка тем, что необходимо настраивать каждый участок пути по отдельности, вводя значения каждого макроскопического параметра и множества соседних участков путей. На сцене с большим количеством узлов можно легко допустить ошибку в настройке. Улучшение интерфейса настройки участка пути с возможностью явно заполнять множество соседних участков пути выбирая их на игровой сцене в режиме редактора существенно поможет сократить количество совершаемых ошибок при разметке игровой сцены.

Еще одним недостатком является тот факт, что макроскопические параметры не являются интуитивно понятными для настройки. Использование более простых для понимания параметров с дальнейшим вычислением необходимых поможет устранить этот недостаток.

Также возникают сложности при отладке работы макроскопической модели. Это можно исправить добавлением инструментов отладки, которые позволят отображать участки пути, отрисовывая сплайн по кратчайшему маршруту от узла к узлу и также отображать необходимые параметры участков путей на поверхности сплайнов. Помимо улучшения процесса отладки это также поможет в разметке игровой сцены, поскольку позволит легко находить ошибочно построенные участки путей.

ЗАКЛЮЧЕНИЕ

В ходе данной работы был выполнен анализ существующих подходов к моделированию нпс в играх. Анализ показал, что в играх игр для симуляции большого количества нпс используют специальные подходы для сокращения требований к вычислительным ресурсам, симулируя только определенную деятельность нпс. Использование таких подходов достаточно редкое явление, поскольку в большинстве игр симуляция большого количества нпс не происходит вовсе – нпс создаются в определенной области рядом с игроком и исчезают после того, как вышли из нее. Обобщение и использование этих подходов в других проектах в полной мере невозможно, поскольку они достаточно сильно зависят от игровых механик, для которых были разработаны.

В ходе работы было решено объединить два метода моделирования для создания двухуровневой модели, которая бы позволила симулировать большое количество нпс с относительно низкими вычислительными затратами. При таком подходе нпс рядом с игроком симулируются микроскопической моделью, а все остальные макроскопической. Для этого был проведен анализ существующих подходов к моделированию пешеходных и транспортных потоков, а также были рассмотрены подходы к моделированию нпс в играх.

Для создания двухуровневой модели симуляции нпс в качестве микроскопической модели был выбран подход симуляции с использованием деревьев поведений, а для макроскопической модели была выбрана гидродинамическая модель LWR. Для макроскопической модели был разработан алгоритм вычисления параметров на шаге симуляции.

В рамках работы разработанный алгоритм макроскопической модели был имплементирован в игровой движок Unreal Engine с использованием языка программирования C++. Была создана микроскопическая модель симуляции на основе дерева поведения, которое находит следующий узел

навигационного графа и перемещает нпс к нему. Также был разработан ряд классов, с помощью которых выполняется объединение двух моделей в одну при помощи создания области микроскопического моделирования вокруг персонажа.

Полученная двухуровневая модель была протестирована. Для этого была найдена модель района Лондона, удовлетворяющая все поставленные требования. Игровая сцена, на основе найденной модели Лондона, была размечена для многоуровневого моделирования, а также простого одноуровневого моделирования. В ходе разметки были сформулированы рекомендации к разметке игровых сцен.

В результате тестирования было выявлено что полученная двухуровневая модель потребляет меньшее количество вычислительных ресурсов по сравнению с прямым симулированием нпс. Разница в использовании вычислительных ресурсов незаметна при низкой плотности нпс на игровой сцене, а при средней и высокой вырастает в пользу разработанной двухуровневой модели.

Также был выполнен анализ области применения разработанной модели. В результате анализа было установлено, что для эффективного использования двухуровневой модели в игре камера игрока всегда должна видеть ограниченную область игровой сцены для уменьшения области микроскопического моделирования. Также игрок теряет возможность встретить нпс, увиденных ранее в другой части сцены из-за агрегирования нпс в параметры макроскопической модели.

В конце работы был сформирован список недостатков, которые были выявлены в ходе тестирования модели и предложены способы их решения.

Реализация модели представлена в репозитории https://github.com/stastmn/MultiLevelSimulationModel.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ahlquist J. B., Novak J. Game Artificial Intelligence. – Thomson, 2008.
2. McNaughton M. et al. Patttern-Based AI Scripting Using ScriptEase //Advances in Artificial Intelligence: 16th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2003, Halifax, Canada, June 11–13, 2003, Proceedings 16. – Springer Berlin Heidelberg, 2003. – С.35 – 49.
3. Colledanchise M., Ögren P. Behavior trees in robotics and AI: An introduction. – CRC Press, 2018.
4. Hilburn D. Simulating behavior trees: A behavior tree/planner hybrid approach //Game AI Pro 360: Guide to Architecture. – CRC Press, 2019. – С.53 – 66.
5. Hall T., Magnusson M. Adaptive Goal Oriented Action Planning for RTS Games. – 2010.
6. de Buy Wenniger G. M., Houtkooper A. GOAP. – 2008.
7. AI Made Easy with Utility AI [Электронный ресурс]. – 2021. – URL: https://medium.com/@morganwalkupdev/ai-made-easy-with-utility-ai-fef94cd36161 (дата обращения 21.12.2023).
8. Humphreys T. Exploring HTN planners through example //Game AI Pro 360: Guide to Architecture. – CRC Press, 2019. – С.103 – 122.
9. Design Unpredictable AI in Games. [Электронный ресурс]. – 2022. – URL: https://medium.com/@stannotes/design-unpredictable-ai-in-games-part-1-architecture-3752a618db6#:~:text=GOB%2C%20or%20Goal-Oriented%20Behavior,is%20the%20lack%20of%20planning (дата обращения 12.12.2023).
10. Якимов М. Р. Основные подходы к моделированию движения пешеходных потоков //Мир транспорта. – 2015. – Т. 13. – №. 4. – С.166 – 173.
11. Scorohodova T., Gonta L. Модели движения толпы //Microelectronics and Computer Science. – 2017. – С.322 – 323.

12. Потапова И. А., Бояршинова И. Н., Исмагилов Т. Р. Методы моделирования транспортного потока //Фундаментальные исследования. – 2016. – №. 10-2. – С.338 – 342.
13. Кутузов В. В. Организация, обеспечение безопасности и управления автомобильными дорогами и транспортной инфраструктурой. – 2020.
14. Yang J. Safety risk analysis and countermeasures study on regular mass passenger flow of China's urban subway //Procedia Engineering. – 2016. – Т. 135. – С. 175-179.
15. Искаков Т. А. Организация транспортного обеспечения стадиона «Лужники» на примере футбольного матча 11. 11. 2017 //Автоматика на транспорте. – 2018. – Т. 4. – №. 2. – С.279 – 296.
16. Козулина Ю. В., Емельянович В. В. Анализ и оптимизация движения транспортного потока на улично-дорожной сети г. Читы //Вестник науки и образования. – 2020. – №. 14-1 (92). – С.16 – 22.
17. Кенжаева Б. О. Методы моделирования процесса работы городского общественного транспорта //Экономика и социум. – 2023. – №. 12 (115)-1. – С.1154 – 1157.
18. Мельников Р. В. Моделирование пешеходных потоков при подготовке к проведению мега-событий //Инженерный вестник Дона. – 2017. – Т. 45. – №. 2 (45). – С.72.
19. Helbing D., Molnar P. Social force model for pedestrian dynamics //Physical review E. – 1995. – Т. 51. – №. 5. – С.4282.
20. Okazaki S., Matsushita S. A study of simulation model for pedestrian movement with evacuation and queuing //International Conference on Engineering for Crowd Safety. – 1993. – Т. 271. – С.2.
21. Elbaum Y., Novoselsky A., Kagan E. A Queueing Model for Traffic Flow Control in the Road Intersection //Mathematics. – 2022. – Т. 10. – №. 21. – С.3997.

22. Antoine G. et al. Real-time traffic flow-based traffic signal scheduling: A queuing theory approach //World Review of Intermodal Transportation Research. – 2021. – Т. 10. – №. 4. – С.325 – 343.
23. Dijkstra J., Jessurun A. J., Timmermans H. J. P. A multi-agent cellular automata model of pedestrian movement //Pedestrian and evacuation dynamics. – Springer, 2001. – С.173 – 181.
24. Yang L. et al. Simulation of evacuation process based on cellular automaton model //Chin Sci Bull. – 2002. – Т. 47. – С.896 – 901.
25. Hoogendoorn S., Bovy P. H. L. Gas-kinetic modeling and simulation of pedestrian flows //Transportation Research Record. – 2000. – Т. 1710. – №. 1. – С.28 – 36.
26. Henderson L. F. On the fluid mechanics of human crowd motion //Transportation research. – 1974. – Т. 8. – №. 6. – С.509 – 515.
27. Helbing D. A fluid dynamic model for the movement of pedestrians //arXiv preprint cond-mat/9805213. – 1998.
28. Kadanoff L. P. Simulating hydrodynamics: a pedestrian model //Journal of statistical physics. – 1985. – Т. 39. – С. 267-283.
29. Procedurally Crafting Manhattan for ’Marvel’s Spider-Man’ [Электронный ресурс]. – 2019. – URL: https://www.gdcvault.com/play/1026415/Procedurally-Crafting - Manhattan-for-Marvel (дата обращения 11.01.2024).
30. Stop Fighting! Systems for Non-Combat AI [Электронный ресурс]. – 2019. – URL: https ://www.gdcvault.com/play/1025833/Stop-Fighting-Systems-for-Non (дата обращения 05.02.2024).
31. How the Nemesis System Creates Stories [Электронный ресурс]. – 2021. – URL: https://www.youtube.com/watch?v=Lm_AzK27mZY&t=0s (дата обращения 09.02.2024).
32. Goal-Oriented Action Planning: Ten Years of AI Programming [Электронный ресурс]. – 2018. – URL:

https://www.youtube.com/watch?v=gm7K68663rA (дата обращения 09.02.2024).

33. A-Life, Emergent AI and S.T.A.L.K.E.R.: An Interview with Dmitriy Iassenev [Электронный ресурс]. – 2022. – URL: https://janjilecek.medium.com/a-life-emergent-ai-and-s-t-a-l-k-e-r-70a9cdde3fac (дата обращения 19.02.2024).
34. Massive Crowd on Assassin's Creed Unity: AI Recycling [Электронный ресурс]. – 2015. –URL: https://gdcvault.com/play/1022141/Massive-Crowd-on-Assassin-s (дата обращения 04.02.2024).
35. Living Worlds: The Joy Of NPC Schedules [Электронный ресурс]. – 2016. – URL: https://www.rockpapershotgun.com/the-joy-of-npc-schedules (дата обращения 12.01.2024).
36. Ultima Game Developer: NPC Schedules [Электронный ресурс]. – 2014. – URL: https://lycaeum.ultimacodex.com/npc-schedules/comment-page-1/ (дата обращения 15.01.2024).
37. 4K Benefiting from Traditional TV Buying Cycle [Электронный ресурс]. – 2019. – URL: https://www.tvtechnology.com/news/4k-benefiting-from-traditional-tv-buying-cycle (дата обращеения 19.01.2024).
38. Common Screen Resolutions | What are they & How to Test? [Электронный ресурс]. – 2023. – URL: https://testsigma.com/blog/common-screen-resolutions (дата обращения 19.01.2024).
39. Martin T., Arne K. Traffic Flow Dynamics. Data, Models and Simulation. Translated by Martin Treiber and Christian Thiemann. – New York : Springer, 2013. – 504 с
40. AccuCities Textured Sample [Электронный ресурс]. – 2022. – URL: https://www.unrealengine.com/marketplace/en-US/product/accucities-textured-sample (дата обращения 20.04.2024).