УДК 004.942

Разработка конструктора для построения имитационной модели городского парка

Пономарев И.В., Якимов Б.Б.
Алтайский государственный университет, г. Барнаул
igorpon@mail.ru, berrek17@gmail.com

Аннотация

Статья посвящена разработке программы-конструктора для моделирования городской зоны отдыха. Программа позволяет полностью визуализировать инфраструктуру городского парка и снабжена инструментами для моделирования поведения посетителей. Основной упор сделан на создание "умных" объектов парковой среды, разнообразие которых увеличивает вариативность возводимой среды и интерактивные возможности агентов.

Ключевые слова: Имитационное моделирование, дерево состояний, умные объекты, utility AI

1. Введение

В работе [1] рассматривалась компьютерная модель, имитирующая поведение посетителей парка. Данная модель помогла выявила проблемные участки парка и предложить рекомендации по улучшению парковой инфраструктуры.

При дальнейшем улучшении модели были выявлены проблемы с использованием стандартных решений, предоставляемых средой разработки Unreal Engine, из-за их низкой гибкости и сложности расширения функционала. Также были отмечены недостатки в моделировании поведения агентов, делающее поведение однообразным. В результате было принято решение о внедрении в программу ряда изменение.

2. Структура модели

В обновленной версии модели в основе также лежит Utility AI [2, 3]. Но теперь изменение кривых полезности уникально для каждого агента за счет набора индивидуальных характеристик, назначаемых случайным образом при инициализации агента [4]. Например, возрастной посетитель будет быстрее уставать. Также, выполняется учет состояния среды: в жаркий день за счет изменения кривых "голод" и "жажда" смещается спрос посетителей в сторону напитков. В холодную погону наоборот.

Объекты, с которыми взаимодействуют посетители представлены собственной реализацией умных объектов. В отличии от стандартных [5], они поддерживают дополнительное промежуточное состояние между "свободно" и "занято", динамические очереди любой длинны, одновременное использование слотов, улучшенный механизм их занятия. За счет открытой реализации упрощен поиск и опрос слотов. Информация об объектах задается не напрямую, а в таблицах данных, и затем во время выполнения извлекается из соответствующих таблиц. Это ускоряет процесс настройки множества объектов за счет пропуска этапа компиляции.

В таблице 1 приведено сравнение стандартных умных объектов с собственной реализацией.

Таблица 1

Сравнение возможностей стандартных умных объектов с собственной реализацией

Особенность	**Smart Objects**	**Собственная реализация**
Gameplay Tags хранятся в:	Слот	Объект
Слот представлен	Компонентом	Дочерним актером
Состояния слота	Свободен, Занят	Свободен, Pending, Занят
Поиск выполняется среди	Свободных слотов	Свободных и Pending слотов
Слот считается занятым	Сразу, как был найден	n метров от объекта
Поиск **ближайшего** слота/объекта	Нетривиальная задача	С помощью EQS
Одновременное использование слота	Не поддерживается	Любое количество NPC
Очереди **переменной** длинны	Не поддерживается	Любая длина

В стандартной реализации умных объектов слоты поддерживают два состояния: свободен и занят. Когда агент выбирает слот, он (слот) сразу переходит в состояние занят. При этом другой агент, даже если он был ближе, вынужден искать другой слот. Промежуточное состояние означает “на слот кто-то претендует”, а занятие происходит только при приближении агента на определенное расстояние, проверки свободен слот или нет. Так, агенты могут конкурировать за слоты.

Слоты с очередями дополнительно реализовывают логику обработки очередей, и перегружают методы обработки состояния слотов (рисунок 1).

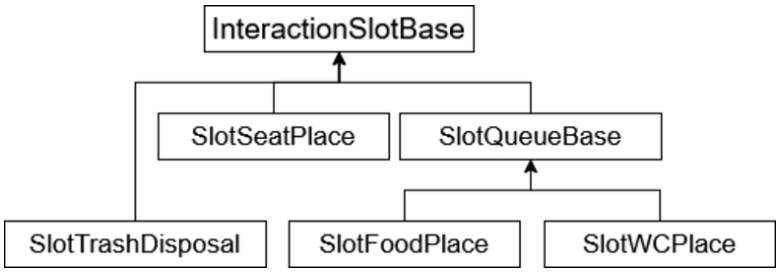

Рисунок 1. Фрагмент иерархии интерактивных слотов

Поведение агентов перенесено в дерево состояния [6]. В предыдущей реализации дерево поведения [7] становилось похожим на машину состояний. Для поддержки большего количества состояний агентов дерево поведения было заменено на дерево состояний, объединяющие в себе лучшие решения машины состояния и дерева поведения. Машина состояний легко может превратиться в полносвязный граф. Деревья поведений очень быстро становятся громоздкими и ограничены тем, что выбор выполняемого поведения всегда ведется слева на право, что увеличивает стоимость его обхода. Дерево состояний поддерживает разные порядки обхода, безусловные и условные переходы в любое состояние в дереве, улучшенный механизм опроса состояния объекта, возможность учитывать глобальное состояние с помощью Evaluator‘s, вызывать и обрабатывать события внутри дерева. При этом вместе с событиями можно передавать полезную нагрузку.

Дерево состояния посетителя парка изображено на рисунке 2. Фактически состоит из двух деревьев: Основного (Root), в второстепенного, отвечающего за обработку поведения в очереди (InQueue).

Основная ветка состоит из состояний:

- Удовлетворения потребностей – поиска умного объекта (Найти слот), попытки его занять, обработки поведения нахождения в очереди в случае, если слот её поддерживает, выполнение поведения, связанного со слотом;
- Свободного перемещения по случайным точкам: прогулка;
- Выход из парка – дойти до точки спавна и удалить агента со сцены.

Ветка обработки очередей выполняет в случае поддержки их слотом, а успешным данное состояние становится при достижение агентом первой позиции в очереди, при которой выполнение возвращается в основную ветку. До тех пор, пока агент не первый в очереди, он будет получать локацию последнего человека в ней, и занимать позицию рядом с ним.

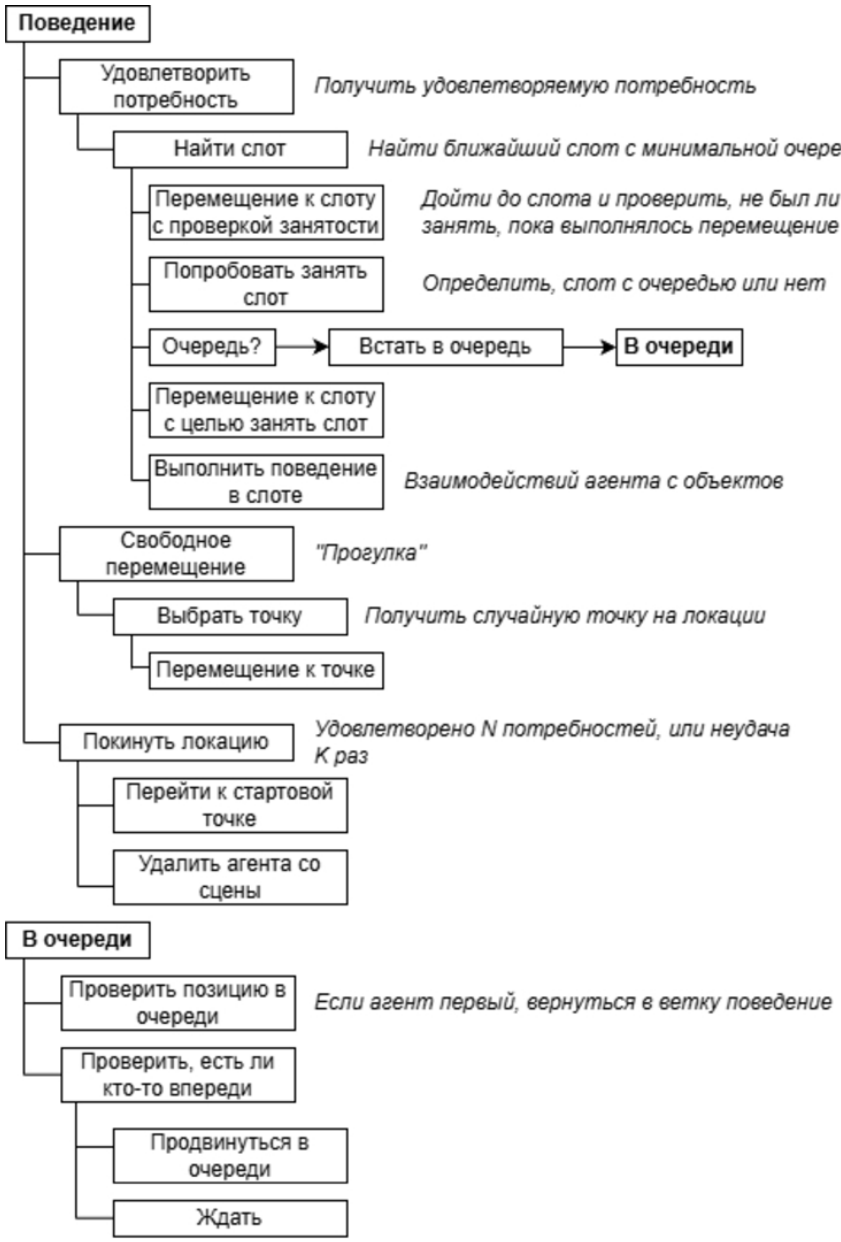

Рисунок 2. Дерево состояние агента посетителя парка

Для перемещения агентов используется динамическая навигационная сетка [8]. Размещение дороги приводит к её обновлению и расширяет возможности агентов в перемещении. Для строительства дорог необходимо три основных сегмента (рисунок 3). Они

выбираются в зависимости от направления прокладываемой дороги и сегментов, соседних к размещаемому.

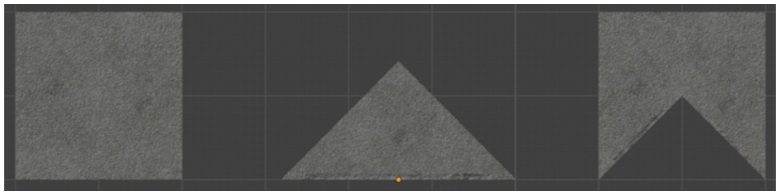

Рисунок 3. Сегменты для прокладки дорог

Строительство возможно по осям X, Y, и диагоналям. Тип сегмента определяется с помощью определения направления движения по осям (рисунки 4,5).

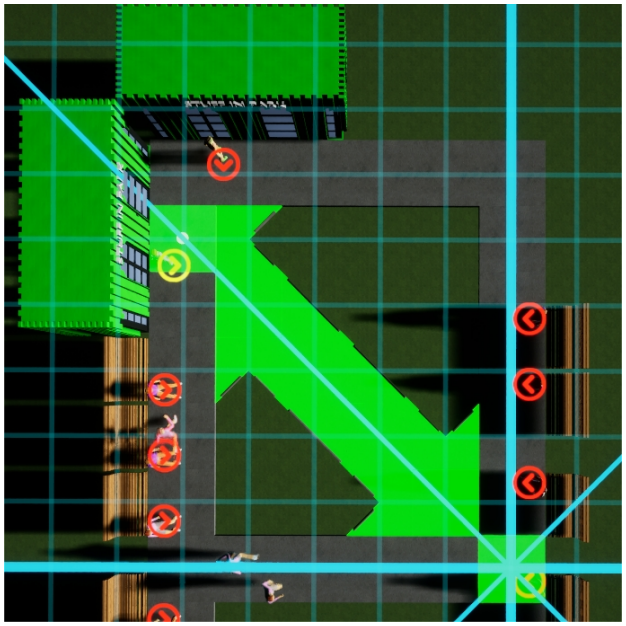

Рисунок 4. Размещение диагональной дороги. Зеленое – предпросмотр сегментов

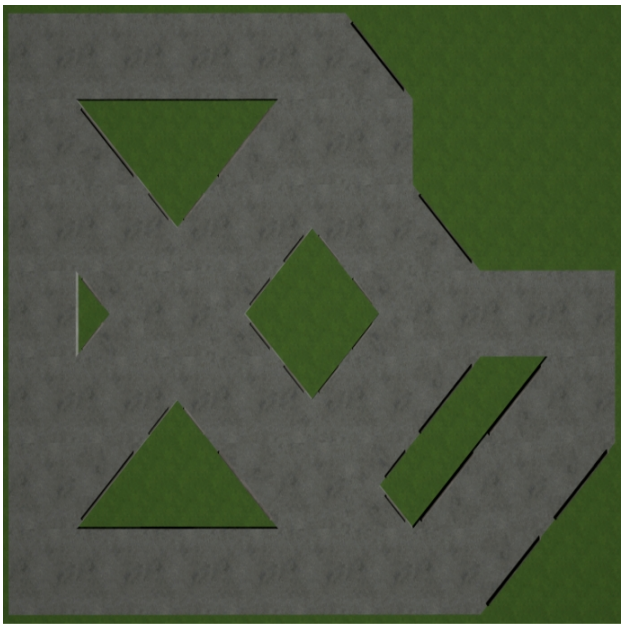

Рисунок 5. Обработка различный вариантов пересечения

Для отображения информации о среде, агентах, объектах реализованы элементы пользовательского интерфейса (рисунки 6,7). В данной реализации ненужно делать выгрузку данных и делать визуализацию отдельно. Информацию доступна в реальном времени.

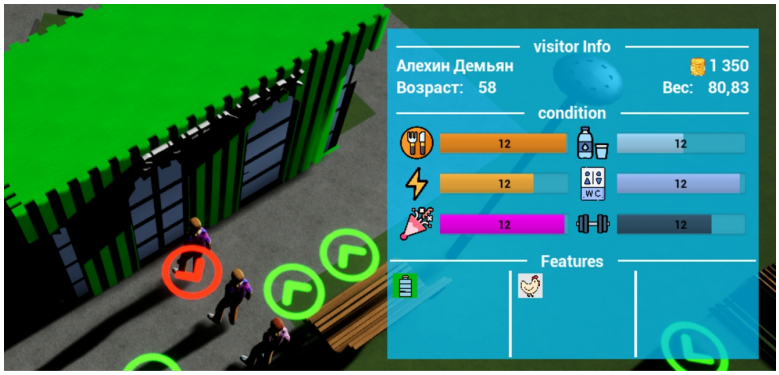

Рисунок 6. Просмотр информации об агенте: общая информация, состояние, индивидуальные особенности

Рисунок 7. Умные объекты со слотами в разном состоянии: красный – занят, зеленый - свободен, желтый – промежуточное состояние “может быть занят”

3. Заключение

Внедрение в программу разработанных объектов позволит значительно улучшить имитационное поведение агентов, сделает модель более универсальной и релевантной. Условно свободное размещение дорог и тропинок позволяет возводить парковую инфраструктуру любого вида. Собственная реализация умных объектов позволяет добиться большей гибкости: поддержка динамических очередей любой длинны, использование объектов несколькими агентами сразу, включение/отключение слотов, более интеллектуальный механизм занятия слотов агентами за счет промежуточного состояния.

Список литературы

1. Пономарев И.В., Якимов Б.Б. Имитационная модель поведенческих предпочтений посетителей городского парка // Известия Алтайского государственного университета. — 2025. — № 1(141). — С. 123–128.

2. Донских А.К., Барабанов В.Ф., Гребенникова Н.И., Белых М.А. Обзор архитектуры систем управления интеллектом на основе полезности и дерева поведения // Вестник Воронежского государственного технического университета. — 2021. — Т. 17, № 2. — С. 36–41.

3. Millington I. Artificial Intelligence for Games. — London : Morgan Kaufmann Publishers, 2006. — 856 p.

4. Gameplay Tags / Epic Games Unreal Engine Documentation. — URL: dev.epicgames.com/documentation/en-us/unreal-engine/using-gameplay-tags-in-unreal-engine?application{_}version=5.3. Дата обращения 14.06.2025.

5. Smart Objects Quick Start / Unreal Engine 5.5 Documentation. — URL: dev.epicgames.com/documentation/en-us/unreal-engine/smart-objects-in-unreal-engine---quick-start. Дата обращения 15.11.2024.

6. StateTree / Epic Games Unreal Engine Documentation. — URL: dev.epicgames.com/documentation/en-us/unreal-engine/state-tree-in-unreal-engine. Дата обращения 14.06.2025.

7. Behavior Trees / Epic Games Unreal Engine Documentation. — URL: dev.epicgames.com/documentation/en-us/unreal-engine/behavior-trees-in-unreal-engine. Дата обращения 14.06.2025.

8. Navigation System / Epic Games Unreal Engine Documentation. — URL: dev.epicgames.com/documentation/ru-ru/unreal-engine/navigation-system-in-unreal-engine?application{_}version=5.3. Дата обращения 15.11.2024.