Дипломка:

1. Оптимизация производительности 3D-игр: подходы к снижению использования ресурсов на мобильных устройствах.
2. Разработка системы генерации процедурных уровней для рогаликов
3. Адаптивная сложность в играх: создание системы, подстраивающейся под уровень игрока.

**Тема** "Разработка системы генерации процедурных уровней для рогаликов"

Алгоритмы генерации:

Можно сравнить эффективность и качество различных алгоритмов, таких как:

1. Псевдослучайные генераторы чисел (PRNG): исследовать влияние различных PRNG на разнообразие генерируемых уровней.
2. Подземелья на основе клеточных автоматов: Анализировать, как различные правила клеточных автоматов влияют на структуру уровней.
3. BSP-деревья (Binary Space Partitioning): изучить влияние параметров BSP-деревьев на сложность и играбельность уровней.
4. Триангуляция Делоне: Исследовать возможности использования триангуляции для создания более органических и естественных уровней.
5. Методы на основе графов: Применить алгоритмы поиска пути и генерации графов для создания взаимосвязанных и логичных уровней.
6. Генерация на основе ограничений: Разработать систему, которая генерирует уровни, удовлетворяющие определенным ограничениям, например, наличие определенных комнат или сложность прохождения.

Метрики качества: Определить и использовать метрики для оценки качества генерируемых уровней. Это могут быть:

Сложность: Как сложно пройти уровень? Можно использовать метрики, основанные на расстоянии до цели, количестве врагов или наличии ловушек.

Интересность: Насколько интересен уровень для игрока? Это более субъективная метрика, но можно использовать такие показатели, как разнообразие комнат, наличие секретных областей или эстетическая привлекательность.

Играбельность: Насколько хорошо сгенерированный уровень подходит для игры? Это может включать в себя такие факторы, как отсутствие тупиков, сбалансированность сложности и наличие достаточного пространства для маневра.

Оптимизация: исследовать методы оптимизации генерации уровней для достижения высокой производительности, особенно при генерации больших и сложных уровней.

Влияние параметров: Провести анализ влияния различных параметров алгоритмов генерации на характеристики получаемых уровней. Например, как изменение плотности комнат влияет на сложность прохождения?

**Комбинирование методов:** Исследовать возможности комбинирования различных алгоритмов генерации для достижения более интересных и разнообразных результатов. Например, использовать клеточные автоматы для создания общей структуры уровня, а затем BSP-деревья для разделения больших комнат на более мелкие.

Тема: Оптимизация производительности 3D-игр: подходы к снижению использования ресурсов на ПК

Введение:

Актуальность проблемы оптимизации производительности игр на ПК.

Рост требований к ресурсам современных игр и необходимость оптимизации.

Цели и задачи исследования.

Обзор существующих методов оптимизации.

Глава 1. Анализ производительности игр на ПК:

1.1. Факторы, влияющие на производительность:

Аппаратные ограничения (CPU, GPU, RAM, хранилище).

Программные ограничения (движок игры, API, драйверы).

Игровые факторы (сложность сцены, количество объектов, AI).

1.2. Инструменты анализа производительности:

Профилировщики (GPUView, Intel VTune Amplifier).

Бенчмарки (3DMark, Unigine Heaven).

Встроенные инструменты разработчика в игровых движках.

1.3. Типичные проблемы производительности на ПК:

Низкая частота кадров (FPS).

Задержки ввода (input lag).

Высокая загрузка CPU/GPU.

Перегрев компонентов.

Глава 2. Методы оптимизации производительности:

2.1. Оптимизация рендеринга:

Уменьшение количества draw calls.

Использование Level of Detail (LOD).

Occlusion Culling.

Оптимизация шейдеров.

Асинхронный рендеринг.

2.2. Оптимизация ресурсов:

Сжатие текстур и моделей.

Управление памятью.

Стриминг данных.

2.3. Оптимизация CPU:

Многопоточность.

Оптимизация алгоритмов ИИ и физики.

Уменьшение количества вычислений на CPU.

2.4. Оптимизация для конкретных аппаратных конфигураций:

Масштабирование настроек графики.

Использование API, оптимизированных под конкретное оборудование (DirectX 12, Vulkan).

Глава 3. Практическое применение методов оптимизации:

3.1. Выбор тестовой игры и платформы.

3.2. Анализ производительности тестовой игры.

3.3. Внедрение выбранных методов оптимизации.

3.4. Анализ результатов оптимизации.

3.5. Сравнение производительности до и после оптимизации.

Глава 4. Разработка инструмента или метода оптимизации:

4.1. Описание разрабатываемого инструмента/метода.

4.2. Архитектура и реализация инструмента/метода.

4.3. Тестирование и оценка эффективности инструмента/метода.

Заключение:

Выводы по результатам исследования.

Перспективы дальнейшего развития методов оптимизации.

Практическая значимость работы.

Список литературы:

Приложения:

Код программы (если разрабатывался инструмент/метод).

Результаты тестирования.

Варианты более конкретных тем:

"Оптимизация производительности 3D-игр на ПК с использованием асинхронного рендеринга"

"Разработка инструмента автоматической оптимизации шейдеров для 3D-игр на ПК"

"Сравнительный анализ эффективности API DirectX 12 и Vulkan в современных играх на ПК"