

УДК 004

## ПРОЦЕДУРНАЯ ГЕНЕРАЦИЯ ГОРОДОВ: ОБЗОР МЕТОДОВ И АЛГОРИТМОВ

Степанов С.А., студент гр. 4410, IV курс

Научный руководитель: Валитова Н.Л., к.т.н., доцент кафедры ПМИ  
Казанский Национальный Исследовательский  
Технический Университет им. А. Н. Туполева – КАИ  
г. Казань

### Введение

Виртуальные города находят применение в играх, кино, градостроительстве и архитектурной визуализации, однако их ручное моделирование требует значительных ресурсов. Процедурная генерация предлагает решение, автоматически создавая городские пространства с помощью алгоритмов, что ускоряет разработку и снижает затраты.

Существует множество подходов к процедурной генерации: от простых методов построения дорог до сложных систем, генерирующих здания и инфраструктуру. Среди них — L-системы, грамматика форм, Wave Function Collapse, Flood Fill и распределение Пуассона. В данной работе рассматриваются эти методы, их преимущества и ограничения.

### Анализ алгоритмов

#### L-системы (L-Systems)

L-системы используют набор правил для генерации сложных структур путем переписывания строк символов [1]. Этот метод часто применяется для создания сетей дорог и городских кварталов. Например, городская дорога может быть описана строкой символов "F+F-F-F", где каждый символ обозначает действие («F» — движение вперёд, «+» — поворот направо, «-» — поворот налево). Применение правил к строке позволяет создавать сложные сети дорог и кварталов.

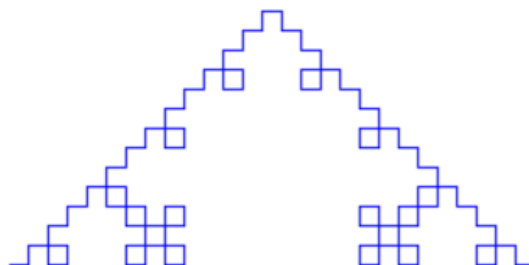


Рис.1 Пример кривой, созданной L-системой

### Преимущества:

- автоматизированность генерации;

- возможность создания сложных структур.

**Недостатки:**

- низкая производительность на больших территориях;
- сложность управления получаемыми результатами.

**Грамматики форм (Shape Grammars)**

Грамматики форм представляют собой параметризованные правила, которые описывают преобразование простых форм в сложные архитектурные объекты, такие как здания [2]. Например, правило «разделить фасад на три равные части и вставить окна в каждую часть» может быть многократно применено, создавая разнообразные здания.

Простое правило деления прямоугольника на фасаде позволяет варьировать число окон и дверей, создавая уникальный внешний вид.

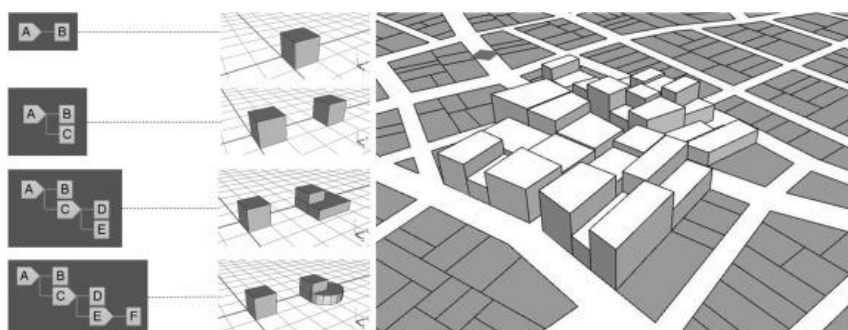


Рис.2 Пример использования грамматики форм в урбанистике.

**Преимущества:**

- обеспечивают реалистичное и разнообразное оформление зданий и архитектурных элементов;
- позволяют легко масштабировать и варьировать детали, меняя простые параметры правил.

**Недостатки:**

- ограничены по применению, больше подходят для генерации отдельных объектов (например, зданий);
- необходимы тщательная подготовка и настройка исходных правил, что может требовать значительных затрат времени.

**Алгоритм Wave Function Collapse (WFC)**

WFC использует тайлы — заранее подготовленные элементы (квадратные/прямоугольные), которые комбинируются по заданным правилам. Тайлы содержат дороги, здания, ландшафтные элементы и другие объекты. Правила совместимости между соседними тайлами позволяют алгоритму создавать структурированные и реалистичные пространства.

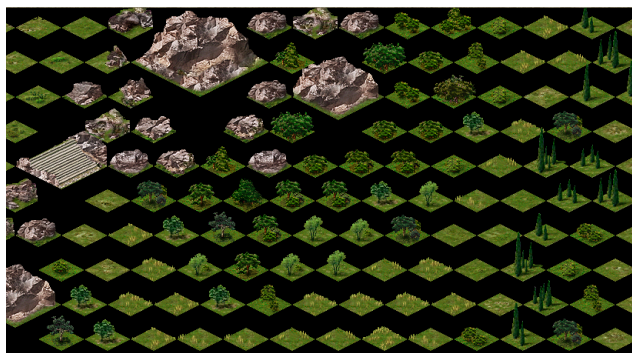


Рис.3 Пример тайлов для построения ландшафта.

Вот как выглядит алгоритм процедурной генерации, основанный на коллапсе волновой функции:

1. Пространство разбито на одинаковые ячейки. Каждая ячейка содержит набор возможных модулей (состояний). Изначально доступны все возможные состояния [3]:

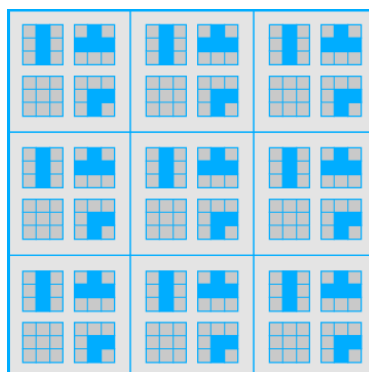


Рис.4 WFC шаг 1.

2. Выбирается ячейка с минимальным числом состояний ( $\geq 2$ ). Все её варианты, кроме одного случайно выбранного, удаляются. Этот оставшийся модуль фиксируется как окончательное состояние ячейки [3]:

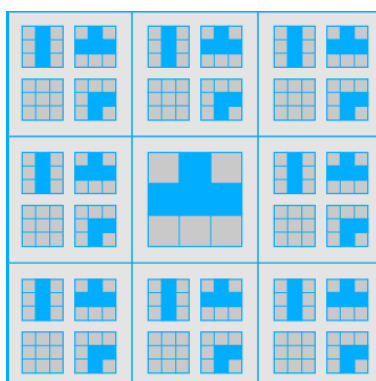


Рис.4 WFC шаг 2: центральная ячейка определена.

3. Соседние ячейки обновляются: удаляются все состояния, несовместимые с выбранным модулем. Например, если модуль содержит

дорогу, поворачивающую влево, в левой ячейке остаются только состояния с продолжением этой дороги [3]:

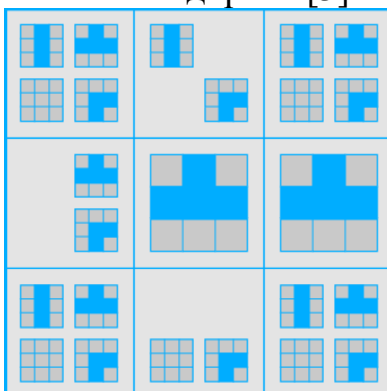


Рис.4 WFC шаг 3: обновляем соседние ячейки.

4. Если при обновлении были удалены состояния, процесс повторяется для следующих ячеек — убираются все несовместимые с новыми условиями модули. Это "волновое" распространение изменений продолжается, пока не стабилизируются все состояния [3].

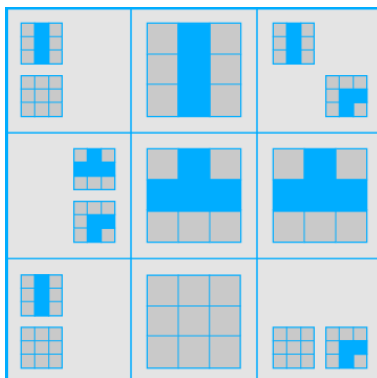


Рис.5 WFC шаг 4: обновляем следующие соседние ячейки.

5. Шаги 2-4 повторяются до тех пор, пока в ячейках больше одного состояния:

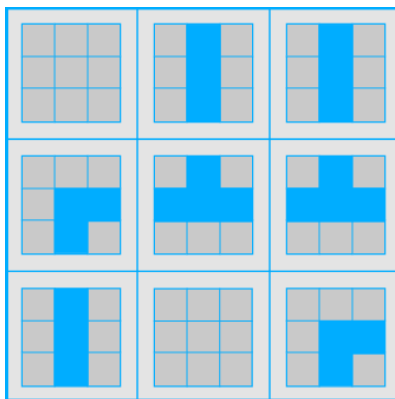


Рис.5 Повторяем шаги.

6. После шага 5 в каждой ячейке должно остаться одно состояние. Если вариантов не остается, алгоритм откатывается на предыдущие шаги и пробует другие комбинации. В завершение модули соединяются в единую структуру [3]:

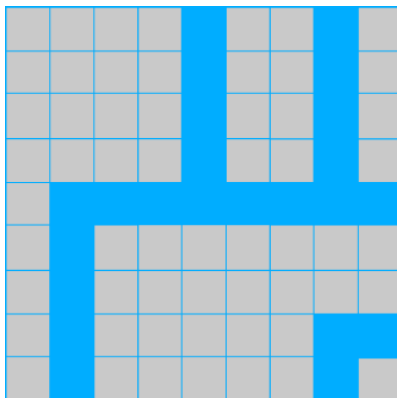


Рис.5 Результат работы алгоритма.

#### **Преимущества:**

- Высокая скорость и эффективность генерации, подходящая для больших виртуальных пространств.
- Простота реализации, ясная логика работы алгоритма и возможность быстрого расширения набора тайлов.

#### **Недостатки:**

- Сложность в достижении высокого уровня реалистичности без качественно подобранных тайлов и правил соседства.
- Возможность возникновения противоречий, что требует дополнительных мер для обработки ошибок.

#### **Flood Fill (заливка областей)**

Flood Fill — это алгоритм заливки связанных областей, используемый как в графических редакторах для закрашивания одноцветных зон, так и в процедурной генерации городов для распределения функциональных зон (жилых, коммерческих, парковых) путём заполнения областей соответствующими тайлами [4].

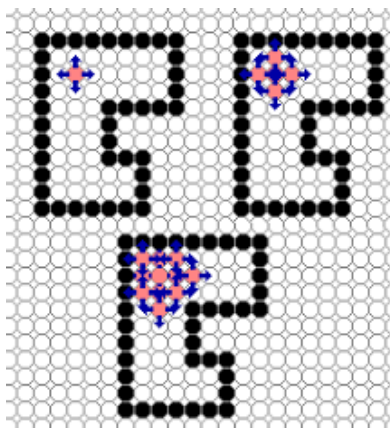


Рис.6 Пример алгоритма Flood Fill.

**Преимущества:**

- Простота и высокая скорость выполнения, что делает алгоритм подходящим для зонирования больших пространств.
- Легко контролировать размер и форму зон при помощи условий заполнения.

**Недостатки:**

- Результат зависит от начальной точки и может быть недостаточно разнообразным.
- Ограниченная функциональность, подходит только для простых задач (например, определения районов и зон).

**Алгоритм выборки по диску Пуассона**

Алгоритм выборки по диску Пуассона обеспечивает равномерное случайное размещение объектов (деревьев, фонарей и др.) в городской среде, исключая их скученность за счёт проверки минимальных расстояний между элементами [5].

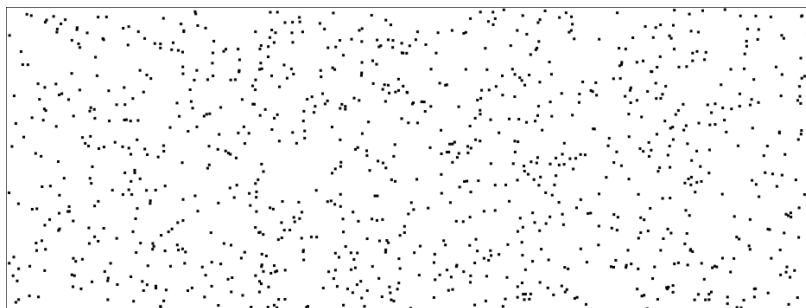


Рис.7 Пример размещения точек с помощью Math.random()

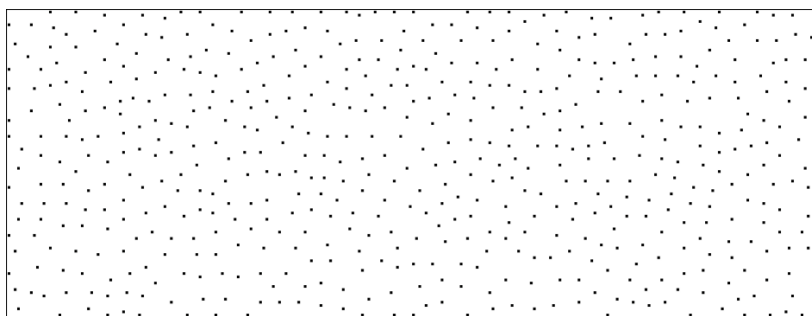


Рис.8 Пример размещения точек по диску Пуассона

**Преимущества:**

- Обеспечивает естественное и равномерное размещение объектов в пространстве.
- Предотвращает перекрытие и скученность элементов, создавая реалистичную визуальную картину.

**Недостатки:**

- Может быть ресурсоёмким при большом количестве объектов.
- Требуется точной настройки минимального расстояния и количества итераций для достижения оптимального результата.

## Сравнение методов

Составим таблицу, позволяющую наглядно увидеть различия между рассмотренными алгоритмами для процедурной генерации городов.

Таблица 1. Сравнение методов

Критерий	L-системы	Грамматик и форм	WFC	Flood Fill	Диск Пуассона
Производительность	Низкая	Высокая	Высокая	Высокая	Высокая
Реалистичность	Средняя	Высокая	Высокая	Средняя	Средняя
Сложность реализации	Высокая	Средняя	Средняя	Низкая	Средняя
Гибкость и расширяемость	Средняя	Высокая	Высокая	Средняя	Средняя
Контроль результата	Средний	Высокий	Средний	Низкий	Средний

Из таблицы видно, что Wave Function Collapse — оптимальный метод для генерации городов, сочетающий производительность, реалистичность и гибкость. L-системы эффективны для сложных структур, но требуют много ресурсов. Shape Grammars подходят для детализации зданий, а Flood Fill и диск Пуассона — для зонирования и размещения объектов.

## Заключение

Процедурная генерация городов представляет собой мощный инструмент для автоматизированного создания виртуальных городских пространств, значительно сокращающий временные и финансовые затраты по сравнению с ручным моделированием. В работе были рассмотрены ключевые алгоритмы: L-системы, грамматики форм, Wave Function Collapse, Flood Fill и распределение Пуассона. Каждый из них обладает уникальными преимуществами и ограничениями, что делает их применимыми для разных этапов генерации — от проектирования дорожных сетей до детализации зданий и размещения объектов.

Наиболее универсальным методом оказался алгоритм Wave Function Collapse, сочетающий высокую производительность, гибкость и относительную простоту реализации. Однако для достижения максимальной реалистичности и эффективности рекомендуется комбинировать несколько подходов.

## Список литературы:

1. Rozenberg G., Salomaa A. The mathematical theory of L systems. – New York: Academic Press, 1980. – 352 p.

2. **Gu N., Wang X.** Slow Computing: Teaching Generative Design with Shape Grammars // Computational Design Methods and Technologies. – Hershey: IGI Global, 2012. – P. 78–95.
3. **Moskalev A.** Wave Function Collapse for procedural generation in Unity [Электронный ресурс]. – 2023. – URL: <https://pvs-studio.com/en/blog/posts/csharp/1027/> (дата обращения: 12.03.2025).
4. **Henrich D.** Space-efficient region filling in raster graphics // The Visual Computer. – 1994. – Vol. 10, № 4. – P. 205–215.
5. **Cook R. L.** Stochastic sampling in computer graphics // ACM Transactions on Graphics. – 1986. – Vol. 5, № 1. – P. 51–72.