ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

***«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

**ИНСТИТУТ КОМПЬЮТЕРНЫХ НАУК И ТЕХНОЛОГИЙ**

**ВЫСШАЯ ШКОЛА ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ И СУПЕРКОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**

**Отчет о прохождении   
стационарной производственной практики (научно-исследовательской работы на тему:** «Создание прототипа системы процедурной генерации и визуализации 3D моделей зданий»**)**

Ларионова Алексея Сергеевича, гр. 3530203/60101

**Направление подготовки:** 02.03.03 Математическое обеспечение и администрирование информационных систем

|  |
| --- |
| **Место прохождения практики:** СПбПУ, ИКНТ, ВШИСиСТ |

*(указывается наименование профильной организации или наименование структурного подразделения*

|  |
| --- |
|  |

*ФГАОУ ВО «СПбПУ», фактический адрес)*

**Сроки практики:** с 02.09.19 по 23.12.19

**Руководитель практики от ФГАОУ ВО «СПбПУ»:**

|  |
| --- |
| Кудрявцев Д.В., доцент ВШИСиСТ, к.т.н. |

*(Ф.И.О., уч. степень, должность)*

|  |
| --- |
| **Консультант от ФГАОУ ВО «СПбПУ»:**  Туральчук К.А., ст. преп. ВШИСиСТ |

*(Ф.И.О., должность)*

|  |
| --- |
|  |

**Оценка:**

Руководитель практики

от ФГАОУ ВО «СПбПУ» Кудрявцев Д.В.

Консультант от ФГАОУ ВО «СПбПУ» Туральчук К.А.

Обучающийся Ларионов А.С.

Дата:

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc28021182)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 4](#_Toc28021183)

[1.1. Основные понятия предметной области 4](#_Toc28021184)

[1.2. Определение существующей проблемы 5](#_Toc28021185)

[2. МЕТОДЫ И ИНСТРУМЕНТЫ РАЗРАБОТКИ 7](#_Toc28021186)

[2.1. Методы процедурной генерация трехмерных моделей 7](#_Toc28021187)

[2.1.1. Формальные грамматики 7](#_Toc28021188)

[2.1.2. Псевдослучайные распределения 7](#_Toc28021189)

[2.1.3. Методы машинного обучения 8](#_Toc28021190)

[2.2. Низкоуровневые API для визуализации на компьютерном экране 9](#_Toc28021191)

[2.3. Системы автоматизированного проектирования (САПР) 9](#_Toc28021192)

[2.4. Другие программные средства 11](#_Toc28021193)

[2.4.1. Языки программирования 11](#_Toc28021194)

[2.4.1. Библиотеки пользовательского интерфейса 12](#_Toc28021195)

[2.5. Резюме применимых для генерации моделей зданий технологий 12](#_Toc28021196)

[3. ПОСТАНОВКА ЗАДАЧИ И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ 14](#_Toc28021197)

[3.1. Требования к системе процедурной генерации и визуализации моделей зданий 14](#_Toc28021198)

[3.2. Постановка цели и пути ее достижения 15](#_Toc28021199)

[ЗАКЛЮЧЕНИЕ 16](#_Toc28021200)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 17](#_Toc28021201)

ВВЕДЕНИЕ

В течение всей эпохи развития компьютерных технологий перед программными продуктами стояла задача автоматизации работы человека, и как следствие, ускорение и удешевление человеческой деятельности в целом. Изначально компьютеры позволяли делать математические расчеты по сложным алгоритмам, затем они смогли перевести оборот документов в электронный вид, что ускорило работу множества прикладных отраслей и сократило расход бумаги.

На данный момент все еще существуют целые отрасли, активно использующие компьютерную графику, например, кинопроизводство, архитектура, разработка компьютерных игр, инженерное проектирование. Для них вопрос автоматизации все так же актуален – необходимо подготавливать большое число трехмерных моделей, причем обычно для этого привлекаются специальные художники или инженеры, владеющие соответствующими системами автоматизированного проектирования. Полный цикл разработки одной модели (включая создание геометрии, наложение текстур, создание метаинформации для визуальных эффектов и интеграция в целевое приложение) может занимать несколько рабочих дней одного художника.

Отсюда возникает потребность в автоматизации этого процесса, или хотя бы ускорения его составных частей. Одним вариантом реализации решения является процедурное генерирование трехмерных моделей, т.е. использование специальных алгоритмов, которые по набору параметров образуют геометрию модели в специальном формате. В таком случае пользователю остается либо принять модель и использовать ее в других этапах своей работы, либо скорректировать параметры, пока результат не будет удовлетворительным. Примером подобной системы является система процедурной генерации моделей зданий, которая может использоваться как для трехмерной визуализации городов, так и для набросков городских архитектурных решений без необходимости кропотливого создания моделей.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Основные понятия предметной области

Компьютерная графика – совокупность программного обеспечения и аппаратных устройств, которая используется для создания и обработки визуальной информации (изображений). Визуальная информация может иметь художественный, информативный, иллюстративный, научный характер и может быть использована как для временной визуализации (в пользовательском интерфейсе или симуляциях, в режиме реального времени), так и для фиксации на устройстве хранения (в специальном формате файла, для последующего воспроизведения).

Компьютерную графику классифицируют по типу представления на двухмерную и трехмерную, по времени воспроизведения на статичную (изображение) и динамичную (анимации), по типу цифрового формата на растровую (сохранение дискретной информации о цвете отдельных групп пикселов) и векторную (в виде параметризованных математических формул).

Трехмерная модель – математическое представление трехмерной сущности. На практике определяется способом цифрового представления этой сущности. Примеры способов: массив геометрических примитивов с метаинформацией (точки, треугольники, воксели и т.п.), параметры известных математических объектов (кубических кривых, поверхностей, кватернионов и т.п.)[1]. Модель нередко сопровождается дополнительными элементами (например, карты текстур и визуальных эффектов, а также развертка модели на эти карты).

Визуализация трехмерных моделей – отображение их на дискретном двухмерном экране, с помощью ортогональной или перспективной проекции (первая сохраняет параллельность линий на изображении, а вторая учитывает положение точки обзора по отношению к модели). Визуализацию в компьютерных системах зачастую выполняют графические ускорители (т.к. аппаратно считают операции линейной алгебры и чисел с плавающей точкой).

1.2. Определение существующей проблемы

Трехмерная компьютерная графика – одно из наиболее динамично развивающихся направлений информационных технологий, что связано с вычислительной сложностью обработки большого количества объектов, которые необходимы для создания реалистичного изображения. Она находит применение в инженерных отраслях (создание макетов и планов), в индустрии развлечений и художеств (кино, компьютерные игры, иллюстрация, реклама), а также в системах симуляции (в том числе с применением виртуальной или дополненной реальности)[2; 3]. Качество и «правдоподобие» трехмерных изображений значительно улучшилось за последние два десятка лет, что связано с многократным увеличением аппаратного быстродействия вычислительных систем (в частности графические ускорители пользовательского сегмента ныне достигают производительности в 5-10 терафлопс и, зачастую, они в 5-6 раз производительнее процессора). Реалистичные компьютерные изображения использовались еще и в начале века (например, трилогия кинокартин «Властелин Колец» 2001-2003 гг.), однако процесс визуализации занимал дни. Теперь же компьютерные системы способны создавать гораздо более сложные изображения в реальном времени.

На данный момент, узким местом реалистичной компьютерной графики является создание моделей, которые складываются в трехмерную сцену. Художник или инженер, который специально обучен принципам работы в системах автоматизированного проектирования (САПР), должен разработать для каждого объекта его трехмерную модель, которая сочетает качества достаточной детализации (т.е. набор математических примитивов отражает форму объекта довольно точно без потери реалистичности) и при этом ограниченной детализации (т.е. количество таких примитивов не превышает разумные пределы, чтобы обработка модели занимала как можно меньшее время). Помимо этого, еще необходимо подготовить карты текстур и эффектов освещения и встроить модель в целевое приложение (сцену фильма или локацию в симуляции). При особых потребностях приложения, модель может иметь не только представление с внешней стороны, но и с внутренней (например, для моделей зданий, помимо фасада может потребоваться модель внутренней отделки комнат).

Как итог, из-за того, что реальный мир состоит из большого количества объектов, а для его компьютерного представления необходимо создавать множество сложных моделей, именно производство трехмерных сцен занимает большую часть времени выпуска фильмов, анимаций, компьютерных игр и симуляций (несмотря на то, что многие из них могут и не быть замечены зрителем или пользователем, в силу того, что зрение человека имеет некоторый фокус). В связи с этим возникает актуальная проблема ускорения этого процесса.

Создание моделей, анимаций и текстур, как и многие другие человеческие задачи могут быть запрограммированы на автоматическое выполнение, но в данном случае конечным наблюдателем модели является человек, а значит важно, чтобы эти объекты выглядели органично (по некоторым субъективным человеческим критериям). Поэтому генерация таких моделей должна лишь способствовать ускорению работы, путем сокращения монотонных действий, нежели подразумевать полное исключение человека из процесса.

Решение обозначенной проблемы не является тривиальным, и на данный момент исследованы некоторые абстрактные математические конструкции, которые позволяют организовать генерацию, а также реализованы несколько процедурных генераторов, которые используются только для конкретного приложения (конкретной системы симуляции или компьютерной игры) исходя из конкретной небольшой задачи, однако многофункциональная система генерации разносортного контента не будет реализована еще несколько лет, в силу отсутствия математического аппарата для учета человеческого понятия «красоты». Тем не менее задача, очевидно, является решаемой, т.к. генераторы для конкретных приложений существуют и показывают достаточно удовлетворительные результаты.

2. МЕТОДЫ И ИНСТРУМЕНТЫ РАЗРАБОТКИ

Рассмотрим существующие технологии, методы, программные средства, которые направлены на процедурную генерацию трехмерных моделей в целом и на генерацию конкретно моделей зданий в частности.

2.1. Методы процедурной генерация трехмерных моделей

2.1.1. Формальные грамматики

Данный метод основывается на использовании математического объекта «грамматика», который состоит из примитивных объектов, составляющих модель, для каждого из которых определяется контекст, в котором он применяется (грамматические правила). Целевое приложение решает задачу с использованием грамматики, присвоив каждому абстрактному примитиву некоторое представление. Метод является хоть и ограниченным способом построения, но также в достаточной мере он гибок и отражает конструкции реального мира, из-за чего он применим в том числе для процедурно генерируемых моделей зданий, где каждому абстрактному примитиву ассоциируется параметризованная форма модели (параллелепипед, пирамида, сфера и т.п.).

Различные вариации и расширения классических формальных грамматик были реализованы на примере L-системы для генерации ландшафта городов [4], грамматики форм (shape grammar) для архитектурного дизайна [5] и генерации карт реальных городов по GIS данным [6], разделенные грамматики (split grammar) в [7].

2.1.2. Псевдослучайные распределения

Метод основывается на построении множества точек по определенным правилам или известным функциям. Множество может быть бесконечно (или очень далеко расширено), и должно напоминать при визуализации встречающиеся в природе паттерны. Например, диаграмма Вороного для случайных точек выглядит как случайная мозаика (применяют для генерации текстур кирпича, древесной коры), а шум Перлина, в разных конфигурациях позволяет симулировать случайные колебания значений (например, перепад высоты ландшафта [8]). Используются и другие способы определения множества точек, которые не имеют известного математического подтекста, но напоминают реальные конструкции, например, разбиение на сетки (tiling) с элементами случайности выпуклой фигуры применялось для генерации модели городских дорог [9]. В остальном, данная категория методов ограниченно применима конкретно к генерации трехмерных моделей зданий, в силу того, что они определяют случайное изменение в точке пространства по отношению к соседним точкам, что редко встречается в архитектуре, где элементы либо организованы, либо имеют лишь небольшую вариацию в относительном положении и ориентации.

2.1.3. Методы машинного обучения

Данный набор технологий имеет множество форм применения при генерации моделей зданий, из-за своей гибкости при решении очень большого класса задач в информатике. Например, реализован прототип системы, который по двухмерному наброску (буквально несколько карандашных линий) формирует элемент здания, например, крышу, пристройку, окно (тип этого элемента также распознается автоматически). Система использует для генерации свёрточную нейронную сеть, обученную на размеченных изображениях зданий[10]. Другие системы используют машинное обучение для создания полноценной трехмерной модели из всего одной фотографии реального здания (соответственно максимально точно воспроизводится показанная сторона, а задняя «достраивается» некоторым несложным образом)[11].

Применение искусственного интеллекта может найти место в любой части процесса генерации модели, т.к. он может быть обучен для репликации практически любых шаблонов (а трехмерные модели, требующие креативности, неминуемо отдаются на разработку человеку-художнику).

2.2. Низкоуровневые API для визуализации на компьютерном экране

Данные программные библиотеки представляют собой минимальный уровень абстракции от аппаратного обеспечения при выводе изображения на экран. С их помощью можно управлять перемещением данных в памяти и исполнением программ на графическом ускорителе. Некоторые функции из программного интерфейса реализуют известные математические концепты вычислительной геометрии, линейной алгебры (операции над матрицами в однородных координатах, методы проекции на плоскость).

a. OpenGL ([Khronos Group](https://en.wikipedia.org/wiki/Khronos_Group)) – кроссплатформенная библиотека, актуальные версии которой поддерживаются большинством графических ускорителей последних 10 лет. Не спроектирована для многопоточного исполнения, но может быть ограниченно использована с ним. Активная разработка проекта замедлилась с развитием библиотеки Vulkan.

б. Vulkan ([Khronos Group](https://en.wikipedia.org/wiki/Khronos_Group)) – высокопроизводительный API, наследник OpenGL. Находится в активной разработке с 2016 (из-за чего не все графические процессоры поддерживают его). Направлен на балансировку работы CPU/GPU, а также многопоточное исполнение. Имеет перспективы становления стандартом индустрии компьютерной графики [12]. Поддержка этой библиотеки и OpenGL в разрабатываемой системе покрывает абсолютное большинство аппаратных и программных платформ.

в. Direct3D (Microsoft, часть пакета DirectX) – последние версии этой библиотеки аналогичны по функционалу с Vulkan, и несмотря на то, что аппаратная поддержка намного больше чем у последнего, поддерживается DirectX только в операционной системе Windows.

2.3. Системы автоматизированного проектирования (САПР)

На данный момент существуют САПР, предлагающие cреду разработки для полного цикла создания файлов компьютерной графики (большинства типов, которые упомянуты в классификации в главе 1), в связи с чем могут использоваться в том числе и для системы процедурной генерации моделей. Чаще всего в них поддерживается продвинутый редактор вершин, ребер, поверхностей в сцене, а также система анимаций скелета, тканей, система смешивания текстур, система визуального или скриптового программирования и прочее.

a) Maya (Autodesk) – многофункциональная, кроссплатформенная САПР, нацеленная скорее на анимацию моделей, нежели на их создание. Последние версии продукта включают среду визуального моделирования, физически-корректные симуляции тканей, кожи, волос. Продукт распространяется по дорогостоящей годовой подписке.

b) Houdini (SideFX) – мощная профессиональная САПР, представляющая все объекты и модификации над ними в виде вершин графа. Это позволяет создавать и модели, и анимации, и эффекты унифицированно, быстро и настраиваемо. Предлагает широкий набор известных алгоритмов для всех областей компьютерной графики, особенно визуальных эффектов и модификаторов. Перечисленные компоненты доступны только при приобретении дорогостоящей лицензии.

c) Blender – свободная и открытая профессиональная кроссплатформенная САПР, предлагающая аналогичный функционал для вышеупомянутых технологий, а также других проприетарных САПР. Программа заслужила большую поддержку и является одним из самых мощных открытых продуктов в мире. Включает возможности автоматизации на скриптовом языке Python.

d) Unreal Engine 4 (Epic Games) – прежде всего это игровой движок, нежели САПР, содержит полноценную программируемую среду (на С++) для управления объектами в памяти и на трехмерной сцене, создания пользовательского интерфейса, работы с видео и звуком, управления физикой реального времени и прочее. В отличие от остальных наиболее хорошо подходит для разработки отдельного программного обеспечения, используя имеющийся функционал. Имеет доступный вариант лицензирования для небольших проектов.

2.4. Другие программные средства

2.4.1. Языки программирования

Системы генерации контента (и трехмерных моделей зданий в частности) могут быть реализованы на любом языке программирования общего назначения. В таком случае выбор конкретного языка для создания системы основывается на спектре приложений, которые принято создавать в индустрии с его помощью.

* С++ − стандартный язык системного программирования, пригодный для создания программ с точным контролем выделяемой памяти в компьютере и предсказуемым быстродействием [13]. Большинство низкоуровневых библиотек имеют интерфейс для C++. Тем не менее у языка нет общего способа включения и конфигурации этих библиотек, отсутствуют встроенные средства сборки и автоматического тестирования.
* Java – кроссплатформенный язык, направленный на парадигму объектно-ориентированного программирования, предоставляет средства для введения многоуровневых абстракций, позволяет уменьшить сложность использования оперативной памяти, однако интерфейсы библиотек являются намного менее производительными, по сравнению с C++.
* Python – скриптовый интерпретируемый язык, традиционно используемый для модулей машинного обучения, из-за наличия сильно оптимизированных интерфейсов к коду на языке C. Не сильно пригоден для других приложений.
* Rust – наследник идей системных языков программирования C/C++, стремящийся сохранить их производительность, при этом решая множество проблем неопределенного поведения (undefined behavior), которые допускают эти языки. Достигается это строгой типизацией и контролем компилятора правильности всех ссылок на участки памяти на момент компиляции, в том числе для многопоточных программ [14]. Язык имеет интегрированную систему сборки с библиотеками, систему автоматических тестов, совместимость с программами на языке C. Разрабатывается Mozilla Research, применяется в одноименном браузере. Является предпочтительным в силу совмещения скорости и корректности программ.

2.4.1. Библиотеки пользовательского интерфейса

Эти библиотеки абстрагируют от операционной системы и способа визуализации оконного интерфейса в ней. Альтернативой использованию таких библиотек, можно применить встроенные конфигураторы интерфейсов других программных пакетов (например, Unreal Engine 4)

* Qt Framework (C++, Rust, cross platform) – мощная платформа для создания сложных настраиваемых программных интерфейсов, из-за чего программирование с этим инструментов требует написание множества дополнительного программного кода. Требует покупки дорогостоящей лицензии, однако предлагает тестовые сборки и несколько утилит с открытым исходным кодом.
* GTK (C/C++, Rust, cross platform) – имеет открытый исходный код, предлагает достаточный функционал для привычных оконных элементов управления. Предпочтителен, т.к. бесплатен для использования, при этом являясь сильно развитым инструментом
* Azul (Rust, cross platform) – является библиотекой только к одному языку, но использует подход визуализации пользовательского интерфейса как веб-страниц, предоставляет большие настройки для дизайна интерфейса.

2.5. Резюме применимых для генерации моделей зданий технологий

Исходя из вышеизложенного описания методов и технологий, особо применимым для генерации трехмерных моделей является использование формальных грамматик (т.к. метод широко исследован, и по сравнению с машинным обучением дает более контролируемые результаты, хотя использование последнего тоже возможно).

Языком программирования имеет смысл взять Rust из-за совмещения производительности и эффективной защиты от логических ошибок. Отдельные модули возможных алгоритмов машинного обучения могут быть реализованы на Python из-за широкого спектра библиотек-абстракций для построения интеллектуальных моделей. При необходимости использования С/C++ интерфейса к дополнительным библиотекам или программным пакетам, можно также применить язык C++.

Для упрощения системы, в качестве пользовательского интерфейса может быть взята надстройка над движком Unreal Engine 4 (из-за уже реализованного визуализатора и возможности интеграции в другие приложения). Однако для опциональной самостоятельной реализации визуализатора следует использовать низкоуровневый API Vulkan (с возможной поддержкой OpenGL), в силу его кроссплатформенности и современности (соответственно направленности на современную графическую аппаратуру), а для пользовательского интерфейса взять библиотеку GTK из-за ее открытой доступности для разработки и многофункциональности.

3. ПОСТАНОВКА ЗАДАЧИ И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ

3.1. Требования к системе процедурной генерации и визуализации моделей зданий

Разработанная система должна, в первую очередь, иметь возможность генерировать трехмерные модели зданий (т.е. формировать набор точек и геометрических примитивов, который определяет геометрию зданий). Генерация происходит случайно, с использованием заданных параметров. Крайне желательным является также формирование «развертки» модели на двумерное изображение, что позволит применить текстуры и специальные эффекты над моделью (возможно даже применение каких-либо распространенных текстур). Пользователь имеет возможность настраивать параметры генерации (численные характеристики, например, высота, или геометрические, например, форма основания). Небольшое изменение параметров ведет к предсказуемым изменениям очередной модели. Система не должна жестко привязываться к одному «стилю» генерируемых зданий, при этом и не должна генерировать абсолютно случайные и сюрреалистичные модели (если только это не то, чего хочет пользователь). Одну и ту же модель можно получить еще раз, при вводе одинаковых параметров и такой же инициализации системы.

Разработанная система должна иметь графический интерфейс, с возможностью визуализации полученных моделей, желательно чтобы изменяемые параметры без задержек применялись при перегенерации для обеспечения простоты настройки. Пользователь имеет средства навигации в окне визуализации (изменение масштаба, осмотр под другим углом, из другой позиции). Опциональным является добавление «пробного» окружения, для оценки реалистичности модели, например, небо с определенным положением солнца (которое меняет угол освещения здания), простая поверхность земли и травы и т.п.

Полученная модель может быть сохранена в некотором распространенном цифровом формате файла (например, OBJ или STL).

3.2. Постановка цели и пути ее достижения

Цель данной работы – реализация прототипа системы для процедурной генерации трехмерных моделей зданий (внешней отделки, фасада) по заданным параметрам, а также визуализации полученной модели. Данный прототип должен удовлетворять большинству требований, обозначенных выше, подчеркивая направленность системы на помощь художникам в сокращении рутинной работы, нежели исключение их из процесса создания модели.

Для достижения данной цели, необходимо решить следующие задачи:

* реализовать модуль процедурной генерации трехмерных моделей зданий, который предоставляет ряд конфигурационных параметров. Генерация должна занимать разумное время и быть контролируемой для пользователя системы,
* реализовать модуль визуализации трехмерных сцен с полученной моделью, а также систему навигации обзора по сцене
* создать дополнительную функциональность для сохранения моделей в специальном формате файла, а также захвата управления с клавиатуры и мыши и пр.,
* интегрировать другие модули в графический интерфейс, в котором пользователь сможет работать с полученным прототипом системы (например, настраивать параметры генерации);

ЗАКЛЮЧЕНИЕ

В ходе данной работы был проведен анализ предметной области и постановка задачи перед реализацией прототипа системы процедурной генерации трехмерных моделей зданий и визуализации этих моделей. Из обзора существующих решений были выведены требования, которые должна выполнять данная система.

На основе сравнительного разбора актуальных методов и технологий, описанных в литературе и применяемых в сфере компьютерной графики и процедурной генерации, были определены инструменты разработки данной системы, а также примерные программные компоненты, решающие поставленные задачи.

Таким образом, результатом данной работы являются сформированная цель и задачи выпускной работы, сформированные критерии к разрабатываемой системе, а также проанализированные основные технологии и методы разработки.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Божко А., Жук Д.М., Маничев В.Б. Компьютерная графика. Гриф УМО ВУЗов России. − М.: Издательство «МГТУ им. Баумана», 2007. − 392 с.
2. Procedural Content Generation: Goals, Challenges and Actionable Steps // Artificial and Computational Intelligence in Games / Togelius J., Champandard A.J., Lanzi P.L., et al., – Dagstuhl: 2013. – Vol. 6 – P. 61-75.
3. Sony Pictures Imageworks Arnold / Kulla C. D., Conty A., Stein C., et al. // ACM Trans. Graph. 37 – 2018.: P. 29:1-29:18.
4. A Survey of Procedural Content Generation Techniques Suitable to Game Development / Carli D., Bevilacqua F., Pozzer C., et al. // Brazilian Symposium on Games and Digital Entertainment, SBGAMES. – P. 26-35.
5. Shape Grammars for Architectural Design / Pauwels P., Strobbe T., Eloy S. et al. // CAAD Futures 2015 – Springer etc., 2015 – Vol. 527. – P. 507-526.
6. Procedural modeling of buildings / Müller P., Wonka P., Haegler S., et al. // ACM SIGGRAPH 2006 Papers, 2006. – P.614-623.
7. Instant Architecture / Wonka P., Wimmer M., Sillion F.X., et al. // ACM Transactions on Graphics, Association for Computing Machinery, 2003. – P. 669–677.
8. Медведева О.А., Гордеев К.И. Генерация карт высот с использованием шума Перлина для построения ландшафтов / Сборник статей V Междунар. научно-практической конференции "Фундаментальные и прикладные научные исследования: актуальные вопросы, достижения и инновации", Пенза, 15 мая 2017 г. – Пенза: МЦНС "Наука и Просвещение", 2017. –Т. 4. – С. 105-108.
9. Kelly G., McCabe H. A Survey of Procedural Techniques for City Generation // The ITB Journal – 2006. – Vol. 7 – Iss. 2 – A. 5.
10. Interactive Sketching of Urban Procedural Models / Nishida G., Garcia-Dorado I., Aliaga D., et al. // ACM Transactions on Graphics (SIGGRAPH Conference Proceedings) – 2016. – Vol. 35 – A. 130 – P. 130:1-130:11.
11. Gutierrez D., Sheffer A. Procedural Modeling of a Building from a Single Image // Computer Graphics Forum (Eurographics) – 2018. – Vol. 37 – N. 2.
12. Shiraef J. An exploratory study of high performance graphics application programming interfaces: thes... mast. computer sci. – Chattanooga, 2016. – 99 p.
13. Якушев Д. М. "Философия" программирования на языке С++ / Якушев Д. М. – 2-е изд., доп. и испр. – М.: Новый издательский дом, 2004. – 319 с.
14. Blandy J., Orendorff J. Programming Rust – Sebastopol: O’Reilly Media, 2017. – 985 p.