14. Петров В.М., Муравей Л.А., Копылова Т.В. «Задачи моделирования и оптимизации некоторых технологических процессов в микроэлектронике» – Дубна, Вестник Международного университета природы, общества и человек «Дубна», №2(17), декабрь, 2007.

15. Muravey L.A., Petrov V.M., Romanenkov A.M. Modeling and optimization of ion-beam etching process. Proceedings. III International conference on optimization methods and applications (OPTIMA-2012). Costa da Caparica, Portugal, September 23–30, 2012. Edited by VI Zubov., vol. 47, p. 181. 2006.

Ссылка для цитирования:
Гусев С. С., Анисимов Е. Ф. Численная реализация метода оптимизации для нахождения оптимального управления процессом нагрева диска // Инженерно-строительный вестник Прикаспия : научно-технический журнал / Астраханский государственный архитектурно-строительный университет. Астрахань : ГАОУ АО ВО «АГАСУ», 2021. № 4 (38). С.107–113.

УДК 004.94+711.4+721
DOI 10.52684/2312-3702-2021-38-4-113-117

МОДЕЛИРОВАНИЕ ЗАСТРОЙКИ ПРОИЗВОЛЬНОЙ ФОРМЫ С ИСПОЛЬЗОВАНИЕМ PYTHON В СРЕДЕ 3DS MAX

В. И. Жигулин, К. А. Шумилов, А. А. Семенов
Санкт-Петербургский государственный архитектурно-строительный университет, г. Санкт-Петербург, Россия

В данной работе представлены результаты разработки алгоритма создания и редактирования трехмерной графики в Autodesk 3ds Max с использованием возможностей высокоуровневого языка программирования Python. На основе полученных результатов появляется возможность моделировать различные варианты квартальной застройки, которые можно использовать как основу BIM-модели при решении задач обучения и проектирования. Распределяются парковые зоны, жилые комплексы и элементы инфраструктуры. По завершению работы алгоритма формируется карта области застройки и ее 3D-модель. В процессе генерации модели с помощью языка Python формируется текстовый документ, содержащий информацию обо всех объектах, что исключает необходимость сохранения самой 3D-модели при каждой случайной генерации. Предлагаемый алгоритм можно использовать для тестовых и учебных заданий, а также возможны различные модификации для использования алгоритма в реальных проектах.

Ключевые сова: *3Ds MAX, Python, генерация застройки, алгоритм, градостроительство, проектирование, BIM.*

MODELING AN ARBITRARY BUILDING WITH PYTHON IN THE 3DS MAX ENVIRONMENT

V. I. Zhigulin, K. A. Shumilov, A. A. Semenov
Saint Petersburg State University of Architecture and Civil Engineering, Saint Petersburg, Russia

This paper presents the results of the development of an algorithm for creating and editing three-dimensional graphics in Autodesk 3ds Max using the capabilities of the high-level programming language Python. Based on the results obtained, it becomes possible to simulate various options for quarterly development, which can be used as the basis of a BIM model when solving training and design problems. Park areas, residential complexes and infrastructure elements are distributed. Upon completion of the algorithm, a map of the building area and its 3D model are generated. In the process of generating a model using the Python language, a text document is generated containing information about all objects, which eliminates the need to save the 3D model itself with each random generation. The proposed algorithm can be used for test and educational tasks, and various modifications are also possible for using the algorithm in real projects.

Keywords: *3Ds MAX, Python, building generation, algorithm, urban planning, design, BIM.*

Введение

Современный уровень развития технологий информационного моделирования можно найти, например, в работах [1–5]. С каждым годом программное обеспечение, позволяющее создать информационную модель объекта строительства, охватывает все больше стадий жизненного цикла и отдельных их разделов. Например, увеличивается количество работ, посвященных вопросам стадии эксплуатации [6, 7], что до недавнего времени было большой редкостью.

Помимо «классических» вопросов создания информационных моделей, в настоящее время появляются работы, посвященные генеративному дизайну [8–11].

Так, в работе [8] авторами рассмотрено применение технологии генеративного дизайна для решения задач строительства. Генеративный дизайн рассматривается, как цифровая технология, работающая с другими сквозными цифровыми технологиями, в том числе с аддитивным производством и искусственный интеллектом. Обсуждаются проблемы широкого внедрения технологии, в том числе в процесс информационного моделирования объектов строительства (BIM).

Предложены подходы к решению некоторых задач проектирования, строительства и эксплуатации с применением технологии генеративного дизайна. Сделан вывод, что технологию генеративного дизайна можно использовать в процессе информационного моделирования объекта строительства на всем его жизненном цикле.

Генеративный дизайн является новым подходом к проектированию, при котором человек делегирует часть процессов подбора параметров объекта компьютерным технологиям.

Фактически, это позволяет в автоматическом режиме и с учетом указанных ограничений создавать набор вариантов какого-либо объекта (или группы объектов) в информационной модели. В

частности, это могут быть и объекты городской застройки – их распределение на каком-либо участке территории с определенными условиями.

Генерации такого поля объектов и посвящена данная работа.

Теория и методы

Разработана схема генерации области застройки случайной формы и размеров. Для этого создан алгоритм, которой произвольно срезает часть площади по периметру прямоугольной формы с учетом количества единичных кварталов, заданных вдоль осей (рис. 1).

В данной работе единичными кварталами являются прямоугольные объекты, имитирующие здания с минимальными размерами по длине и ширине. Логика данного алгоритма строится на прямоугольной квартальной застройке, в которой необходимо задать количество кварталов вдоль осей *X* и *Y*. В дополнение к этому можно задать процент срезаемой по краям части.

Сама форма, представляющая собой прямоугольную форму города, строится через двумерный массив в Python [12–14]. Этот массив изначально заполняется нулями, обозначающими свободное место под единичные кварталы.

Далее, случайным образом в цикле, выбирается одна из четырех крайних сторон сформированной прямоугольной формы. После этого, на выбранной стороне случайно выбирается один из элементов и, если этот элемент равен нулю, он заменяется на единицу, показывая, что в дальнейшем в данном месте нельзя будет разместить застройку или часть застройки. Если же данный элемент уже равен единице, происходит случайный выбор оси, по которой будет происходить смещение выбора элемента. После того, как ось выбрана, случайным образом осуществляется смещение выбора, т. е. либо увеличение, либо уменьшение на единицу.

Если алгоритм вышел за границу формы или же, другими словами, происходит выход за пределы массива и будет превышено допустимое значение, то осуществляется возврат к предыдущему элементу и повторение предыдущих шагов. В противном случае, если перемещение к новому элементу прошло успешно, вновь будет проверено значение этого элемента и, если оно окажется равным нулю, будет изменено на единицу и начнется новая итерация цикла. Если же значение нового элемента равно единице, то будут повторены предыдущие шаги.

Алгоритм закончит работу сразу, как только количество заполненных единицами элементов станет равно заданному проценту срезаемой по краям части в начале алгоритма.

После получения формы области застройки, генерируется карта, на которой обозначено, какой участок и каких размеров выделяется под три разных локации, а именно жилые комплексы, инфраструктуру и парковые зоны.

Вначале распределяются парковые зоны. В скрипте помимо процента срезаемой по краям части формы задается процент парковых зон на всю итоговую территорию без учета уже срезанной части. В представленном алгоритме полученное значение общей территории, которая была выделена под парки, делится на четыре для каждой четверти итогового города с целью более равномерного распределения зеленых зон по всей территории.

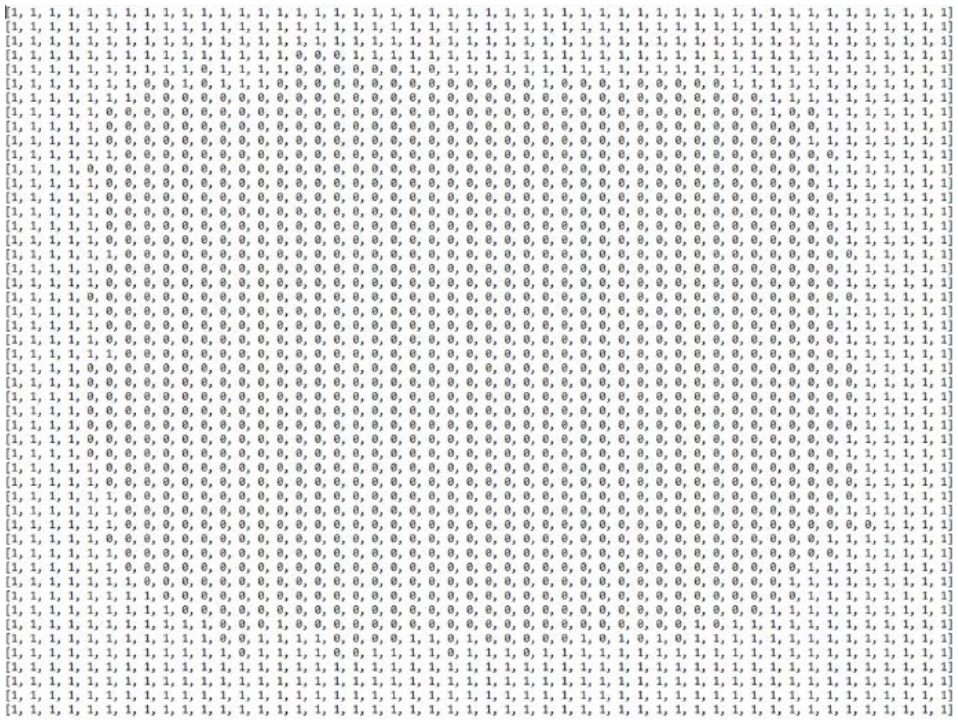


Рис. 1. Смоделированное очертание области застройки

После этого случайным образом задаются размеры сторон парковых зон и генерируется позиция на территории, с которой в данный момент происходит работа (одна из четвертей формы). Далее происходит проверка всех элементов массива, на которые попадает по размерам парковая зона, а также дополнительно по одному элементу во все стороны от зеленой зоны, чтобы в итоговой модели парки не прилегали вплотную друг к другу, и, если все элементы под эту зону равны нулям, на эти места в массиве записывается парковая зона. В противном случае происходит возврат к началу итерации и производится новая попытка разместить зеленую зону.

При успешном размещении генерируется модель парка зеленого цвета и парковой зоне присваивается номер, который записывается в «карту города», что продемонстрировано на рисунке 2. Этот номер состоит из двух букв, первая из которых это латинская буква *P* (означает парковую зону), затем идет буква из латинского алфавита и последним идет цифра от 01 до 99, обозначающая номер зоны.

После успешного размещения зеленой зоны подсчитывается количество элементов массива, которые уже были заняты парками для данной четверти и, если данное число достигает значения выделенной территории для одной четверти всей формы, происходит переход на следующую четверть. После завершения работы со всей территорией работа с парковыми зонами заканчивается и начинается следующий этап.

Далее генерируются жилые комплексы и элементы инфраструктуры. Для них в алгоритме прописывается минимальное и максимальное количество этажей здания, а также высота одного этажа, чтобы при генерации модели случайно выставлялась высота дома в заданном диапазоне. Логика построения у этих объектов следующая.

Алгоритм в цикле перебирает все нули (свободные места под строительство объектов) и случайным образом выбирает какой объект будет установлен, а именно жилой комплекс или инфраструктурный элемент. Случайным образом выбираются размеры квартала под этот объект. Моделируется единичный квартал по оси *X*, протяженный от одного до шести по оси *Y*, либо от одного до шести по оси *X* на один по оси *Y*. Далее происходит проверка, возможно ли разместить сгенерированную форму в данной позиции. Если по оси *X* расположен один элемент, то движение осуществляется по оси *Y* и проверяются значения элементов. Если все они равны нулям, то на эти позиции записывается данный объект и генерируется модель, в противном случае происходит возврат к началу итерации цикла. Аналогично и в ситуации, когда по оси *Y* располагается один элемент, а по оси *X* несколько.

При успешном размещении жилых комплексов генерируются модели серого цвета, а для элементов инфраструктуры – оранжевого. В «карту города» жилые комплексы заносятся с номером, который формируется таким же образом, как и для парковых зон, но с отличием, заключающимся в первой букве – она меняется на латинскую букву *H*. Для сгенерированного элемента инфраструктуры номер начинается с латинской буквы *I*.

По завершению работы данного алгоритма формируется карта города (рис. 2) и его 3D-модель.

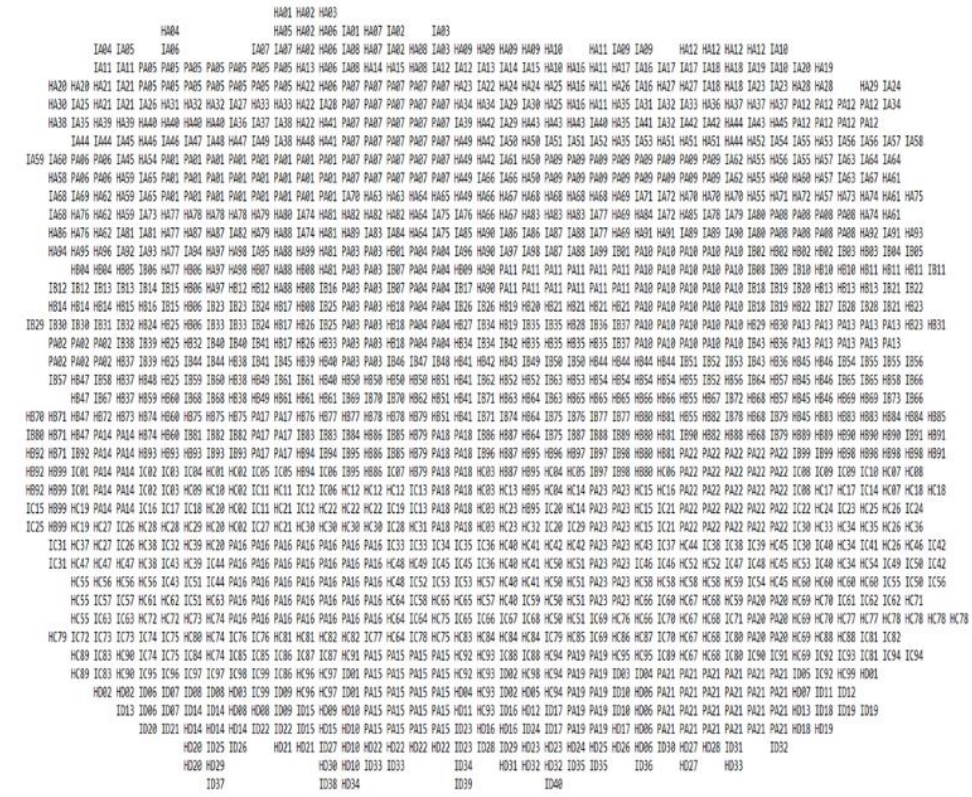


Рис. 2. Смоделированная карта города

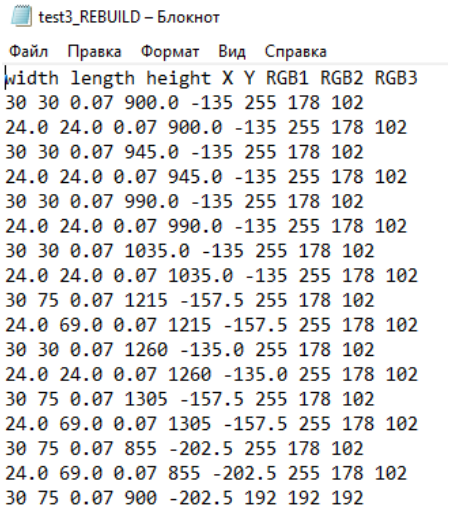
В процессе генерации модели города с помощью языка Python и его возможностей создается и формируется текстовый документ, продемонстрированный на рисунке 3.

В данный документ заносятся все характеристики каждой генерируемой трехмерной модели в процессе выполнения скрипта из 3Ds Max [15, 16], а именно длина, ширина и высота объекта, координаты центра модели и три числовых значения цветовой модели RGB. С помощью данного документа появляется возможность в дальнейшем, используя данные из файла и возможности языка Python по работе с текстовыми документами, заново построить сгенерированную модель в 3Ds Max.

В данном случае, построение копии ранее сгенерированной модели происходит по тем же алгоритмам, за исключением того, что вместо случайных (высота) или вычисленных (координаты цента объекта) значений берутся значения из текстового документа.

Данное дополнение к работе исключает необходимость сохранения самой 3D-модели при каждой случайной генерации. Таким образом, появляется возможность сохранять данные о каждом сгенерированном городе в трех текстовых документах, что значительно экономит выделяемую память ЭВМ под набор генерируемых моделей. Особенно заметно это преимущество при генерации моделей больших размеров.

На рисунке 4 продемонстрирован вид сверху на сгенерированную модель города для демонстрации идентичности карты города, показанной на рисунке 2, и его 3D-модели.


test3_REBUILD – Блокнот

Файл Правка Формат Вид Справка

```
width length height X Y RGB1 RGB2 RGB3
30 30 0.07 900.0 -135 255 178 102
24.0 24.0 0.07 900.0 -135 255 178 102
30 30 0.07 945.0 -135 255 178 102
24.0 24.0 0.07 945.0 -135 255 178 102
30 30 0.07 990.0 -135 255 178 102
24.0 24.0 0.07 990.0 -135 255 178 102
30 30 0.07 1035.0 -135 255 178 102
24.0 24.0 0.07 1035.0 -135 255 178 102
30 75 0.07 1215 -157.5 255 178 102
24.0 69.0 0.07 1215 -157.5 255 178 102
30 30 0.07 1260 -135.0 255 178 102
24.0 24.0 0.07 1260 -135.0 255 178 102
30 75 0.07 1305 -157.5 255 178 102
24.0 69.0 0.07 1305 -157.5 255 178 102
30 75 0.07 855 -202.5 255 178 102
24.0 69.0 0.07 855 -202.5 255 178 102
30 75 0.07 900 -202.5 192 192 192
```

Рис. 3. Содержимое текстового файла для генерации модели

Для итогового варианта программы также был создан графический пользовательский интерфейс с использованием библиотеки Pyside2. В данном пользовательском интерфейсе есть возможность задавать количество кварталов вдоль оси *X* и оси *Y*, задавать процент срезаемой по краям области застройки, задавать процент парковых зон от итоговой площади произвольной области застройки. Присутствует возможность задать длину и ширину единичных кварталов, в зависимости от которых формируются кварталы больших размеров, а также минимальную и максимальную высоту зданий. Также возможен контроль ширины проезжей части.

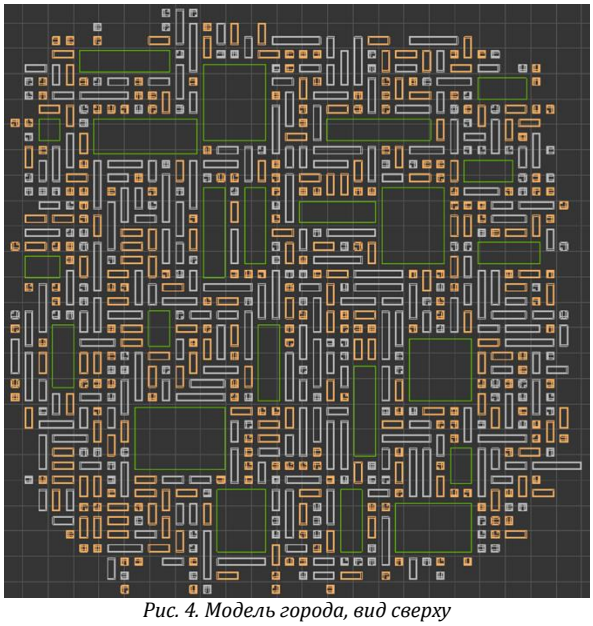


Рис. 4. Модель города, вид сверху

На рисунке 5 продемонстрирована сгенерированная модель города при обзоре с перспективы.

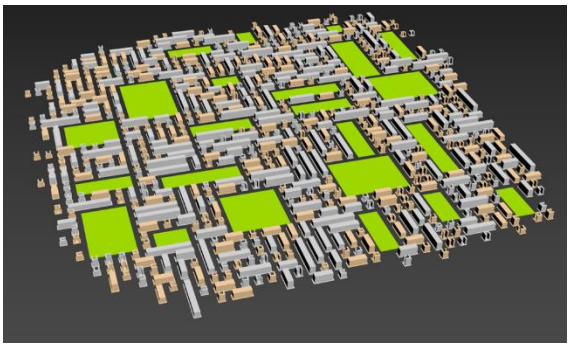


Рис. 5. Модель города, перспектива

В пользовательском интерфейсе помимо построения модели по веденым данным, присутствует возможность построения ранее сгенерированной модели на основе имеющегося файла с данными.

Заключение

Полученный алгоритм и модели можно использовать для решения задач благоустройства территорий, обучения архитекторов, градостроителей и ландшафтных дизайнеров, а также как кейсы для проведения чемпионатов по BIM-проектированию.

Благодарности

Исследование проведено в рамках проекта «BIM-ICE – BIM Integration in Higher and Continuing Education» Программы приграничного сотрудничества поддержки совместных проектов по внешним границам ЕС «Юго-Восточная Финляндия – Россия 2014–2020».

Список литературы

1. Pezeshki Z., Ivari S. A. S. Applications of BIM: A Brief Review and Future Outline // Archives of Computational Methods in Engineering. 2016. DOI: 10.1007/s11831-016-9204-1.
2. Шеина С. Г., Петров К. С., Федоров А. А. Исследование этапов развития BIM-технологий в мировой практике и России // Строительство и техногенная безопасность. 2019. № 14(66). С. 7–14.
3. Вербицкий В. А. Анализ программных комплексов и опыта внедрения BIM-технологий // International Journal of Advanced Studies. 2019. Т. 9, № 2. С. 14–28. DOI: 10.12731/2227-930X-2019-1-14-28.
4. Захарова Г. Б. Как BIM перерастает в CIM и в цифровой двойник города // BIM-моделирование в задачах строительства и архитектуры: материалы IV Международной научно-практической конференции / под общ. ред. А. А. Семенова. СПб.: СПбГАСУ, 2021. С. 27-36. DOI: 10.23968/BIMAC.2021.003.
5. Згода Ю. Н., Семенов А. А. Перспективы развития программного и аппаратного обеспечения BIM-моделирования // Новые информационные технологии в архитектуре и строительстве: материалы научно-практической конференции с международным участием. Екатеринбург: УрГАХУ, 2020. С. 43.
6. Гиря Л. В., Трофимов Г. П. Применение BIM-технологий в практике эксплуатации зданий и сооружений // BIM-моделирование в задачах строительства и архитектуры: материалы IV Международной научно-практической конференции / под общ. ред. А. А. Семенова. СПб.: СПбГАСУ, 2021. С. 113-119. DOI: 10.23968/BIMAC.2021.003.
7. Толстолуцкая А. А. Информационное моделирование и применение BIM-технологий на этапе эксплуатации зданий // Сборник докладов IX международной научно-практической конференции студентов, аспирантов и молодых ученых. БГТУ им. В.Г. Шухова, 2018. С. 251–254.
8. Игнатова Е. В., Предеина В. П. Состояние и перспективы применения технологии генеративного дизайна в строительстве // Строительство и архитектура. 2021. Т. 9, № 1. С. 71–75. DOI: 10.29039/2308-0191-2021-9-1-71-75.
9. Федчун Д. О., Тлустый Р. Е. Сравнительный анализ методов параметрического, информационного и генеративного архитектурного проектирования // Вестник Инженерной Школы Дальневосточного Федерального Университета. 2018. № 1(34). С. 103–115. DOI: 10.5281/zenodo.1196721.
10. Бжахов М. И., Ефимова М. М., Журтов А. В. Алгоритмическое проектирование в архитектуре // Инженерный вестник Дона. 2018. № 2 (49). С. 166.
11. Кривенко А. А., Моор В. К., Гаврилов А. Г. Генеративное проектирование как средство формирования архитектурных объектов // Архитектура и дизайн: история, теория, инновации. 2017. № 2. С. 203–206.
12. Bronshteyn I.E. Study of defects in a program code in Python // Programming and Computer Software. 2013. Vol. 39. P. 279–284. DOI: 10.1134/S0361768813060017.
13. Корныхин Е.В., Хорошилов А.В. Использование языка программирования Python для описания ограничений на архитектурные модели // Труды ИСП РАН. 2015. Т. 27, № 5. С. 143–156. DOI: 10.15514/ISPRAS-2015-27(5)-8.
14. Lvov M., Kruglyk V. Teaching algorithmization and programming using Python language // Education and Information Technologies. 2014. No. 20. P. 13–23. DOI: DOI:10.14308/ite000493.
15. Документация: Autodesk 3ds Max 2021. URL: http://help.autodesk.com/view/3DSMAX/2021/ENU/ (дата обращения: 11.01.2021).
16. Документация: qtmax [Электронный ресурс]. – Режим доступа: https://help.autodesk.com/view/MAXDEV/2021/ENU/?guid=Max_Python_API_qtmax_module_html (дата обращения: 23.10.2020).

Ссылка для цитирования:
Жигулин В. И., Шумилов К. А., Семенов А. А. Моделирование застройки произвольной формы с использованием Python в среде 3DS MAX // Инженерно-строительный вестник Прикаспия : научно-технический журнал / Астраханский государственный архитектурно-строительный университет. Астрахань : ГАОУ АО ВО «АГАСУ», 2021. № 4 (38). С. 113–117.

УДК 721.02+004.42
DOI 10.52684/2312-3702-2021-38-4-117-123

ВИЗУАЛЬНОЕ ПРОГРАММИРОВАНИЕ В ЗАДАЧАХ МОДЕЛИРОВАНИЯ СТРОИТЕЛЬНЫХ КОНСТРУКЦИЙ

Н. Г. Георгиев, К. А. Шумилов, А. А. Семенов
Санкт-Петербургский государственный архитектурно-строительный университет, г. Санкт-Петербург, Россия

Технологии информационного моделирования зданий (BIM) или построения цифровой информационной модели (ЦИМ) являются наиболее перспективным и востребованным направлением в современном проектировании и строительстве. В данной работе описываются созданные авторами алгоритмы визуального программирования в связках Dynamo – Revit и Grasshopper – Rhinoceros. С помощью предлагаемых алгоритмов возможно создавать строительные конструкции различной формы с вариативной настройкой исходных данных. В качестве примера реализации такого алгоритма создан скрипт, с помощью которого пользователь может моделировать башни различной формы и с различными исходными данными, такими как: тип геометрии основания, число этажей, длина стороны этажа или радиуса окружности (в зависимости от основания), высота этажа. Сгенерированные модели анализируются на надежность и прочность в вычислительном комплексе SCAD Office.

Ключевые слова: *BIM, визуальное программирование, скрипт, Revit, Dynamo, Rhinoceros, Grasshopper.*

VISUAL PROGRAMMING IN THE PROBLEMS OF MODELING BUILDING STRUCTURES

N. G. Georgiev, K. A. Shumilov, A. A. Semenov
Saint Petersburg State University of Architecture and Civil Engineering, Saint Petersburg, Russia

Building information modeling (BIM) or digital information model (CIM) technologies are already the most promising and demanded direction in modern design and construction. This article describes the visual programming algorithms created by the authors in the