Автоматизация создания инженерных учебных WebGL приложений

А.В. Власенко [1], С.С. Александрова [1], Ф.Г. Садреев [1], П.А. Воронин [1]

[1] Московский авиационный институт (МАИ), Оршанская улица дом 3, г.Москва, 121359, Россия

Аннотация

В статье рассматривается дополнение (аддон) для Blender 3D 2.9x, написанное на языке Python 3.9.2, с использованием библиотек Pandas, Jinja2, math и др., которое позволяет автоматизировать операции по созданию анимации болтов, шайб, гаек. Позволяет вставлять из внешних файлов дополнительные инструменты (отвертки, ключи, стрелки и т.п.), участвующие в процессе анимации. Наносить материалы из внешней библиотеки, которая представляет собой *.blend файл с набором материалов, исходя из названий объектов в сцене. Экспортировать полученную сцену в WebGL 2.0, с использованием фреймворка Babylon.js, с возможностью проигрывания полученной анимации, которая включает в себя традиционные кнопки управления анимацией, задания скорости воспроизведения, возможности перемещения по кадрам анимации, а также доступ к виду из камеры Blender 3D, там, где имеет место ее анимации. В html странице создается интерактивный список объектов сцены, который был автоматически сгенерирован из имен объектов сцены Blender 3D при помощи Jinja2, при нажатии на галочку элементов списка объект скрывается/отображается. В качестве примеров оценки работы аддона использовались машиностроительные сборки редукторов, импортированные из T-FLEX CAD 3D 17 в формате *.stl. Разбираются некоторые особенности создания аддонов для Blender 3D. Приводится сравнение библиотеки Three.js и фреймворка Babylon.js с целью их использования для создания инженерных интерактивных приложений. Рассмотренный в работе аддон для Blender 3D может быть использован не только при разработке учебных инженерных приложений, но и как элемент подготовки при создании, например ИЭТР.

Ключевые слова:

WebGL, T-FLEX CAD 3D, Blender 3D, Python, Three.js, Babylon.js, 3D графика.

1. Введение

Традиционно программное обеспечение, специализирующееся на создании интерактивной 3D анимации промышленных изделий и конструкций является частью экосистемы компаний разработчиков САПР (система автоматизированного проектирования) или компаний занимающихся разработкой ИЭТР (интерактивных электронных технических руководств). В этой области можно отметить такие программные продукты как Cortona3D, SolidWorks Composer, Solid Edge 3D Publishing, SimLab Composer и др.

Для создания учебного контента в этой области, данные программы довольно дорогостоящи (за исключением возможно SimLab Composer), требовательны к ресурсам компьютера, обладают избыточным функционалом, например помимо 3D анимации также позволяют делать 2D иллюстрации, и в большинстве своем не имеют учебных версий, не так гибки в настройках по 3D анимации, и нет открытого доступа к API продуктов. Некоторые продукты, например, Cortona3D требуют в процессе работы для достижения определенного функционала интеграции с CAD, PDM (Product Data Management) системами.

ГрафиКон 2021: 31-я Международная конференция по компьютерной графике и машинному зрению, 27-30 сентября 2021 г., Нижний Новгород, Россия

EMAIL: moluk@yandex.ru (А.Н. Власенко); sweta.sergeeva@gmail.com (С.С. Александрова); sadreevfg@yandex.ru (Ф.Г. Садреев); heyholyrampage@yandex.ru (П.А. Воронин)

ORCID: 0000-0002-8820-9954 (А.Н. Власенко)

С другой стороны, мы имеем традиционные программы полигонального моделирования и анимации, такие как 3Ds Max, Blender 3D, Maya, Cinema4D и др., которые в первую очередь ориентированы на создание рендера сцены в виде одиночного растрового изображения или набора изображений для последующей записи в видео.

В тоже время, для создания интерактивного 3D контента в дополнение к программам моделирования и анимации используются чаще всего так называемые игровые движки, например такие как, Unity3D, Unreal Engine, Godot Engine, Unigine и др., направленные прежде всего на создание игр различных жанров. Игровые движки поддерживают компиляцию приложения под многие существующие платформы (ПК – Windows, Linux, игровые консоли, мобильные системы – Android, iOS и др.), а для создания логики приложения использую в основном такие языки программирования как C++ и C#.

Во время пандемии и после нее возросла роль в создании интерактивного учебного контента, ориентированного прежде всего на web платформы. Для создания интерактивных 3D приложений в браузере в настоящее время используется кроссплатформенный API – WebGL (Web-based Graphics Library) версий 1.0 и 2.0, а также язык программирования низкого уровня WebAssembly для стековой виртуальной машины, спроектированный как портативная цель компиляции для высокоуровневых языков, таких как Си, C++, C# и др. Второй вариант используют игровые движки, в том числе и для портирования в web старых приложений (игр), разработанных на C++, OpenGL и т.п. [1, 2].

2. Общее описание проекта

Для создания промышленных изделий традиционно используют твердотельные CAD (Computer-aided design) программы. В качестве примеров были взяты сборки планетарного и червячного редукторов, спроектированных в T-FLEX CAD 3D. Так как Blender 3D корректно не поддерживает импорт CAD форматов, в отличие от 3Ds MAX, то сборки были экспортированы в виде набора из отдельных *.stl файлов при помощи T-FLEX CAD 3D.

После импорта набора тел в единую сборку в Blender 3D, модели, как правило, требуют сглаживания. В Blender для этих целей есть быстрая команда, которая называется «Гладкое затенение» (Shade Smooth), но она не всегда работает корректно из – за неправильного задания угла наклона нормалей граней, производимого автоматически. Это можно исправить, задав вручную для каждого тела угол автосглаживания, который по умолчанию равен 30^0. Однако, если в сборке более 100 деталей, то для каждого объекта нужно выполнить предыдущих два действия, что занимает довольно продолжительное время. Как раз для подобных однотипных, повторяющихся операций и существует возможность автоматизации процесса с использованием «оболочки» (wrap) для API Blender 3D, которая использует язык программирования Python.

Программа Blender 3D поставляется с полноценной версией Python (3.7.7 в Blender 2.92 и 3.9.2 для будущих версий Blender 2.93 и 3.x), на который можно устанавливать дополнительно практически любые библиотеки, не относящиеся к самому Blender, например Pandas, Jinja2 и др.

Основным модулем для программирования в Blender, является bpy (акроним от blender python). Написание скриптов можно делать в Blender при помощи встроенного редактора текста, но в нем нет автозаполнения и подсветки синтаксиса кода в случае ошибок, как это сделано в большинстве IDE. Для восполнения данного недостатка можно вручную установить библиотеку автодополнения bpy (англ. autocomplete) с помощью штатного установщика pip install fake-bpy-module-2.91 в зависимости от версии Blender, и использовать привычную IDE, например PyCharm, VS Code и др.

Об ошибках программы можно только узнать, вызвав окно системной консоли Blender. Консоль по умолчанию использует кодировку cp1251 (если мы рассматриваем работу в Blender под MS Windows), из-за этого кириллические тексты отображаются в виде непонятных символов. Устранить эту проблему можно, указав кодовую страницу UTF-8 в консоли cmd до запуска самого Blender или создать *.bat файл сценария запуска Blender.

Сам модуль bpy не имеет ни свойств, ни методов, а служит неким пространством имен, отделяющим модули Blender от модулей Python, т.е. например, обращение к объектам (mesh) сцены blend-файла происходит так: **bpy.data.objects**. Чтобы, например, получить список

выделенных объектов обращаемся так: **bpy.context.selected_objects**, т.е. мы получаем доступ к свойствам объектов в контексте выполнения, а сам контекст зависит от того, над какой областью находится курсор мыши, или какой(ие) объект(ы) выделен(ы) мышью. Посмотреть, как называются другие подмодули bpy можно, обратившись к официальной справке по Python API Blender [3, 4] или посмотреть в журнале операций (в Blender это окно называется «Информация»), который отображает в виде кода практически все наши действия в Blender. Также можно поднести курсор мыши к любому элементу в меню и во всплывающей подсказке помимо описания, также будет написан код этого элемента.

Далее рассмотрим некоторые отдельные элементы меню, созданного в правой боковой панели свойств элементов (так называемой N-панели, по быстрой клавише, которая ее вызывает) сцены Blender.

2.1. Модуль анимации

В рассмотренных в качестве примера сборках планетарного и червячного редукторов наибольший интерес представляет автоматическое выделение однотипных объектов и их последующая автоматическая анимация при процессе сборки и разборки, а также добавление в сцену инструментов, которые необходимы для сборки и разборки (ключи, отвертки и прочее). Рассмотрим, например, крепежные элементы (болты, гайки, шайбы).

Внешний вид интерфейса модулей панели инструментов и анимации приведен на рисунке 1:

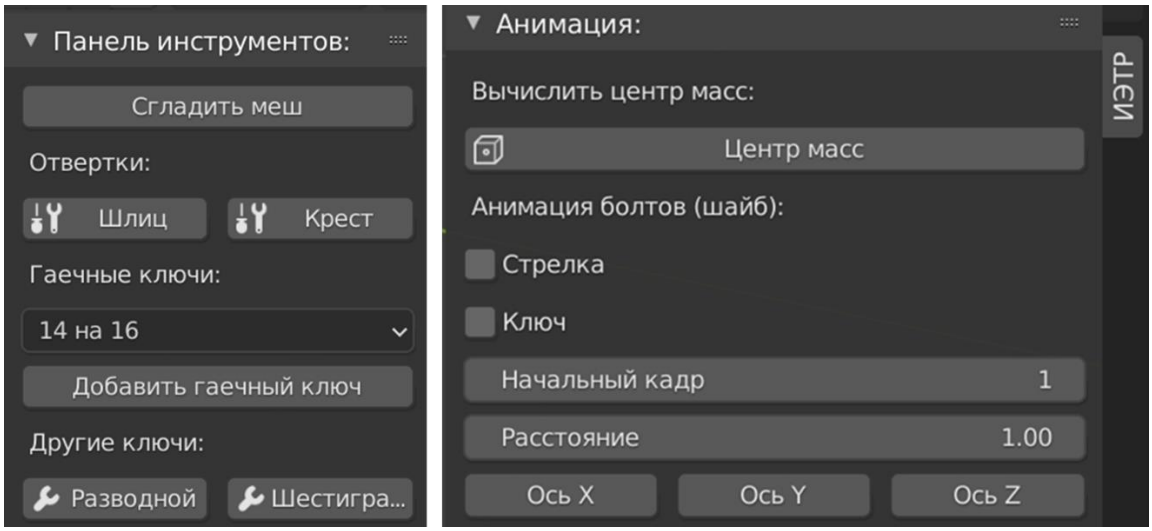

Рисунок 1: Пример интерфейсной формы модулей панели инструментов и анимации

Перед началом анимации осуществляется расчет центра масс объектов (болтов, гаек, шайб) сцены с использованием штатной команды Blender: **bpy.ops.object.origin_set(type='ORIGIN_CENTER_OF_VOLUME', center='MEDIAN')**. Координаты центра масс объектов сохраняются в отдельный датафрейм. После расчета центра масс:

- выделяем мышкой нужные для анимации болты;
- задаем начальный кадр анимации, сама анимация занимает 60 кадров (2.5 сек);
- задаем расстояние (в Blender расстояния выражаются в так называемых unit единицах, один unit по умолчанию равен 1 м, это можно изменить в настройка Blender), на которое будут разноситься объекты, причем если расстояние положительное или отрицательное, то это будет дополнительно указывать направление оси координат, например X или -X;
- нажимаем на нужную ось координат (в сцене Blender красная ось – это X, зеленая – Y) относительно которой будет происходит разбор элементов.

Автоматически в процессе анимации будут выделены относящиеся к болтам их шайбы и гайки, также в зависимости от направления оси будет анимирован угол выкручивания болта (гайки). Дополнительно также можно вставить в сцену вспомогательные элементы в виде гаечного ключа и стрелки, которая указывает направление выкручивания. Эти инструменты будут автоматически размещены напротив первого выделенного болта.

Автоматический выбор болтов и гаек происходит на основе запросов с использованием библиотеки Pandas к датафрейму с координатами центра масс объектов и выделении близких по значениям объектов. Близость определяется нахождением функции минимума координат шайб и гаек по отношению к координатам выделенных болтов.

2.2. Модуль нанесения материалов

Помимо анимации, для более реалистичного представления элементов сборки, необходимо нанести соответствующие материалу объектов текстуры (например, металл, окрашенный металл, металл с признаками старения и пр.)

Внешний вид панели по нанесению материалов приведен на рисунке 2:

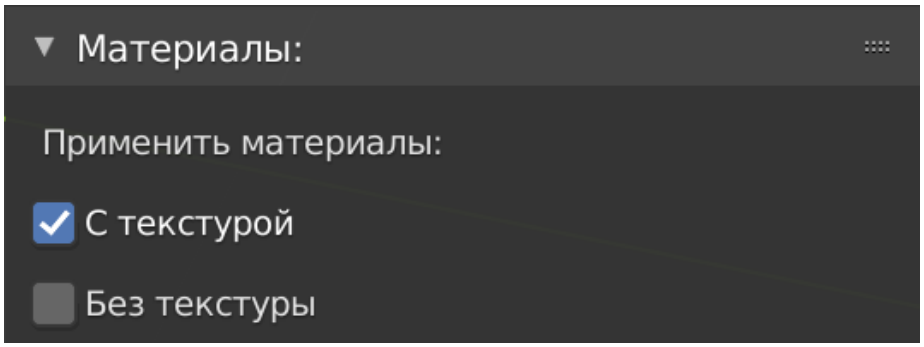

Рисунок 2: Пример интерфейсной формы модуля нанесения материалов

Сами материалы содержатся во внешнем blend-файле и по умолчанию наносятся на объекты исходя из их названия. Названия объектов и соответствующих им материалов хранятся в таком типе данных Python как словари. В словарях есть возможность обращаться к конкретному имени объекта, и они обладают достаточным быстродействием. При нанесении материала также автоматически создается UV-развертка объектов там, где она необходима. Конечно, автоматически сгенерированная UV развертка уступает по качеству ручной или сделанной в специальных программных продуктах типа UVLayout Pro и др., но, например для создания шероховатости на металлических поверхностях вполне подходит.

Так как модели в дальнейшем будут экспортироваться в WebGL, то материалы в зависимости от размера модели и требуемого уровня детализации сцены выбраны двух типов: с текстурой и без текстуры, с нанесенным обычным цветом. Материалы, которые с текстурой, представляют собой материалы на основе PBR (Physically based rendering) рендеринга. В PBR материалы включены следующие параметры (текстуры):

- Основной цвет (Base Color);
- Металличность (Metalness) для металлических поверхностей;
- Шероховатость (Roughness) имитация на поверхности неровностей и шероховатостей;
- Карта нормалей (Normal map) карта неровностей.

2.3. Модуль экспорта в WebGL

Внешний вид панели экспорта в WebGL приведен на рисунке 3:

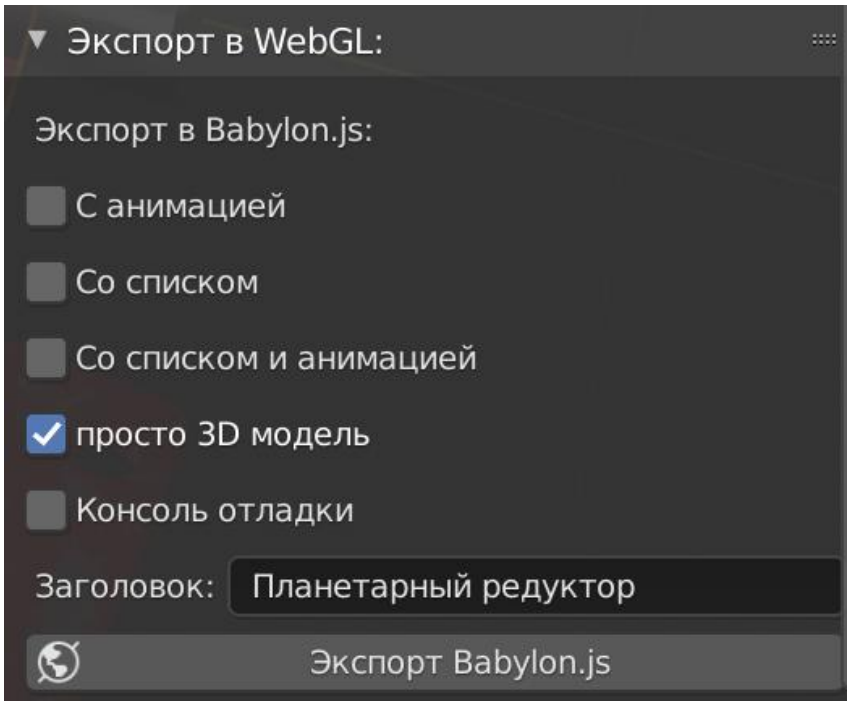

Рисунок 3: Пример интерфейсной формы модуля экспорта в WebGL

Для экспорта в WebGL использовался фреймворк Babylon.js. В таблице 1 приведено сравнение Babylon.js с наиболее популярной библиотекой Three.js.

Таблица1

Сравнение библиотеки Three.js и фреймворка Babylon.js

Параметр сравнения	Babylon.js	Three.js
Консоль разработчика (для отладки приложения)	Есть	Нет
Наличие своего GUI (graphical user interface)	Есть	Используется внешняя библиотека dat.gui
Названия объектов в сцене после экспорта в формате GlTF из Blender	Названия такие же, как и в программе	Пробелы заменяются нижним подчеркиванием, точки убираются совсем
Формат карт окружения (environment map)	Рекомендуется CubeMap *.dds или *.env	Любой формат
Среда для разработки приложений, без использования кода	Babylon.JS Editor (offline на базе Electron.js)	Three.JS Editor (on-line, offline)
Экспорт из Blender	Свой аддон или штатный экспорт в gltf	штатный экспорт в gltf
Экспорт из 3Ds Max	Свой плагин (формат *.Babylon или gltf)	Нет
Экспорт из Maya	Свой плагин (формат *.Babylon или gltf)	Нет
Экспорт из Cinema4D	Свой плагин (формат *.Babylon или gltf)	Нет
Возможность создания нативного приложения	Есть	Нет
Нодовый редактор материалов	Есть	Нет

Разработчики	MS Microsoft + сообщество	Сообщество
Основные разработчики	Дэвид Катуш (David Catuhe), Дэвид Руссе (David Rousset) и др. сотрудники Microsoft	Рикардо Мигель Мистер Дуб Кабелло (Ricardo Mr.Doob Cabello)
Дата выхода (выпуска)	2015	2013
Дополнительные особенности	Имеется своя on-line среда разработки, интеграция с Visual Studio, есть свой официальный канал на YouTube и форум	Самая популярная WebGL библиотека в мире, больше всего реализованных примеров и учебных материалов

Как видим из таблицы, фреймворк Babylon.js [5,6], в отличие от библиотеки Three.js обладает возможностью экспорта сцены в формате GlTF (Graphics Language Transmission Format or GL Transmission Format) из всех наиболее популярных пакетов по 3D моделированию и анимации. Необходимо также обязательно помнить и о некоторых различиях, отраженных в таблице, между библиотеками при экспорте в данном формате. Помимо стандартизованного формата GlTF можно использовать и специальный формат фреймворка *.Babylon, который отличается возможностью экспорта дополнительных свойств (материалов, освещения, анимации, возможности по группировке объектов и др.) присущих каждому из пакетов в отдельности, из-за этого размер файла по сравнению с GlTF может вырасти в несколько раз. Также фреймворк предоставляет очень удобную консоль разработчика для наглядного управления многими свойствами объектов, которые довольно проблематично подобрать при помощи кода, например настройка положения и свойств камеры, настройки параметров материалов, освещения и т.п.

У фреймворка Babylon.js есть и своя библиотека по созданию как традиционного 2D интерфейса, которая представляет собой слой canvas поверх WebGL, так и 3D GUI (англ. graphical user interface) внутри сцены, который может быть использован в том числе и в VR (англ. virtual reality) проектах.

Что касается библиотеки Three.js [7,8], то это одна из самых популярных на сегодняшний день WebGL библиотек в мире по данным GitHub. На ней реализовано наибольшее количество проектов, а также огромное количество разнообразных учебных материалов, отчасти это обусловлено и более ранним появлением библиотеки по сравнению с Babylon.js. Библиотека более легковесная, если сравнивать размеры файлов, по сравнению с Babylon.js. Синтаксис ближе к нативному WebGL, из-за меньшего количества абстракций в некоторых случаях пишется гораздо большее количество кода, по сравнению с Babylon.js. К недостаткам или особенностям библиотеки следует отнести выход новой версии буквально через каждый месяц-два, помимо добавления нового функционала часто меняется синтаксис уже традиционных функций, что не всегда полезно для длительных проектов. Из-за этого возникают разного рода проблемы по воспроизведении кода, например из разных учебных материалов. В Babylon.js эта проблема решена путем создания специальной интерактивной on-line среды (или песочницы, https://playground.babylonjs.com/), с подсветкой синтаксиса кода и автозаполнения которая поддерживает разные версии фреймворка и может быть использована бесплатно в том числе и для учебных целей.

При экспорте сцены используется шаблонизатор кода Jinja2, на базе него было создано несколько шаблонов: по управлению анимацией и создания интерактивного списка объектов сцены. На рисунке 4 приведен пример сгенерированного приложения с меню анимации и со списком, а также включенной консолью разработчика Babylon.js.

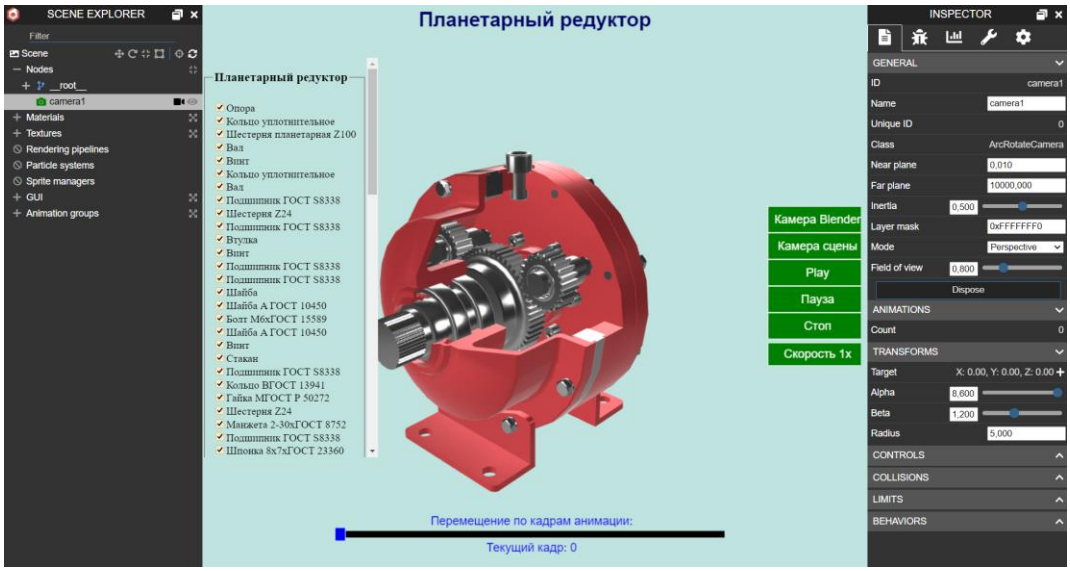

Рисунок 4: Пример сгенерированного приложения с меню анимации и со списком, а также включенной консолью разработчика

Список объектов сцены в Babylon.js был автоматически сгенерирован из имен объектов сцены Blender при помощи Jinja2, при нажатии на галочку объект скрывается/отображается. Меню анимации включает в себя традиционные кнопки управления анимацией, задания скорости воспроизведения, возможности перемещения по кадрам анимации, а также доступ к виду из камеры Blender, там, где имеет место ее анимация.

3. Заключение

Рассмотренный в работе аддон для Blender может быть также использован не только при разработке учебных инженерных приложений, но и как элемент подготовки при создании, например ИЭТР. В дальнейшем планируется развивать данное приложение с целью создания полностью автоматизированного процесса анимации промышленных изделий, включающей такие элементы конструкции как кабели, гибкие шланги и т.п.

4. Литература

[1] А.Н. Власенко, В.В. Степанов Использование WebGL в образовательном процессе при подготовке специалистов в области сварочного производства. Дистанционное и виртуальное обучение. 2016. № 12 (114). С. 115-121.
[2] Е.В. Тетюев, Ю.А. Жук Использование WebGL для визуализации учебных материалов в образовательном процессе. Труды 30-й Международной конференции по компьютерной графике и машинному зрению (Санкт-Петербург 22–25 сентября 2020 г.). Том 1 / Институт прикладной математики имени М. В. Келдыша РАН. – Санкт-Петербург, 2020. С. 59 – 66.
[3] Blender Python API, документация разработчика, URL: https://docs.blender.org/api/current/index.html
[4] Witold Jaworski Programming Add-Ons for Blender 2.8 - version 2.0 URL: http://airplanes3d.net/downloads/pydev2/pydev-blender-en.pdf
[5] Официальная документация по фреймворку Babylon.js URL: https://doc.babylonjs.com/
[6] Julien Moreau-Mathis Babylon.js Essentials. Packt Publishing Ltd, 2016, 200p.
[7] Официальная документация по библиотеке Three.js URL: https://threejs.org/docs/
[8] Jos Dirksen Learn Three.js: Programming 3D animations and visualizations for the web with HTML5 and WebGL, 3rd Edition. Packt Publishing Ltd, 2018, 528 p.