DOI 10.21672/2074-1707.2020.52.4.066-074 УДК 004.94

## ОСОБЕННОСТИ ПОДГОТОВКИ 3D-ОБЪЕКТОВ, СМОДЕЛИРОВАННЫХ В BLENDER, ДЛЯ ИМПОРТА В UNITY 3D

Статья поступила в редакцию 15.09.2020, в окончательном варианте — 18.09.2020.

**Гараева Энже Ринатовна**, Казанский национальный исследовательский технический университет им. А.Н. Туполева – КАИ, 420111, Российская Федерация, г. Казань, ул. К. Маркса, 10, магистрант, e-mail: ange0112@yandex.ru

**Бикмуллина Ильсияр Ильдаровна**, Казанский национальный исследовательский технический университет им. А.Н. Туполева – КАИ, 420111, Российская Федерация, г. Казань, ул. К. Маркса, 10, кандидат технических наук, доцент, ORCID: 0000-0002-0194-2687, e-mail: elsiyar-b@yandex.ru **Барков Игорь Александрович**, Казанский национальный исследовательский технический университет им. А.Н. Туполева – КАИ, 420111, Российская Федерация, г. Казань, ул. К. Маркса, 10, доктор технических наук, профессор

При разработке игры в Unity с использованием 3D-моделей возникла необходимость создания собственных 3D-моделей, поэтому были проанализированы собственные возможности Unity на предмет создания 3D-моделей. Так как возможности Unity как среды для создания 3D-моделей ограничены, были рассмотрены сторонние программные продукты, предназначенные для моделирования 3D-объектов. В результате анализа для разработки объектов был выбран 3D-редактор Blender, в связи с чем возникла необходимость изучения особенностей импорта готовых объектов из Blender в Unity. Решение использования в Unity 3D-моделей, созданных в Blender, исходит из ограниченности возможностей Unity в сфере 3D-моделирования. Из-за чего было сформулировано руководство, в котором описано, на что необходимо обращать внимание для подготовки объектов из Blender в Unity. В данной статье приводится краткое описание профессионального инструмента для разработки 3D-моделей и игрового движка, в которое предполагается экспортирование полученных 3D-моделей, включая краткую историческую справку о них, их основные возможности, особенности, благодаря которым они выигрывают среди прочих программных продуктов, предназначенных для 3D-моделирования и создания игр, а также сведения о том, в каких областях и насколько широко данные продукты используются. Рассмотрены основные ошибки, возникающие при импорте готовых объектов, а также описаны особенности подготовки готовых моделей из Blender в Unity. Кроме того, рассмотрены причины, по которым требуется использование нескольких программных продуктов для создания полноценного продукта. Представлено описание процесса экспорта в рассматриваемом инструменте. Подробно разобран поэтапный процесс подготовки объекта к экспорту в Blender. Порядок и особенности подготовки к экспорту объектов из Blender в Unity рассмотрены на примере подготовки и экспорта смоделированной в Blender модели игрушечного медведя. Для этого был проведён детальный анализ предметной области, результаты которого представлены в данной статье в виде поэтапного описания процесса подготовки объекта к экспорту.

Ключевые слова: 3D-объекты, импорт, экспорт, 3D-модели, моделирование, Unity, Blender

# FEATURES OF PREPARING 3D OBJECTS MODELED IN BLENDER FOR IMPORT INTO UNITY 3D

The article was received by the editorial board on 15.09.2020, in the final version -18.10.2020.

*Garayeva Enzhe R.*, Kazan National Research Technical University named after A.N. Tupolev – KAI, 10 K. Marx St., Kazan, 420111, Russian Federation,

master's student, e-mail: ange0112@yandex.ru

*Bikmullina Ilsiyar I.*, Kazan National Research Technical University named after A.N. Tupolev – KAI, 10 K. Marx St., Kazan, 420111, Russian Federation,

Cand. Sci. (Engineering), Associate Professor, e-mail: elsiyar-b@yandex.ru

**Barkov Igor A.**, Kazan National Research Technical University named after A.N. Tupolev – KAI, 10 K. Marx St., Kazan, 420111, Russian Federation,

Doct. Sci. (Engineering), Professor

When developing a game in Unity using 3D models, it became necessary to create your own 3D models, so Unity's own capabilities were analyzed for creating 3D models. Since Unity's capabilities as a 3D modeling environment are limited, third-party software products designed to model 3D objects were considered. As a result of the analysis, the 3D blender editor was chosen for the development of objects, and therefore there was a need to study the features of importing finished objects from Blender to Unity. Unity's decision to use 3D models created in Blender is based on Unity's limited 3D modeling capabilities. Because of this, a guide was formulated, which describes what you need to pay attention to in order to prepare objects from Blender to Unity. This article provides a brief description of the professional tool for developing 3D models and game engine, which is supposed to export the resulting

3D models, including a brief historical background about them, their main features, features, thanks to which they win among other software products designed for 3D modeling and game creation, as well as information about what areas and how widely these products are used. The main errors that occur when importing ready-made objects are considered, as well as the features of preparing ready-made models from Blender to Unity are described. In addition, the reasons why it is necessary to use several software products to create a full-fledged product are considered. The description of the export process in the considered tool is presented. The step-by-step process of preparing an object for export to Blender is analyzed in detail. The order and features of preparation for exporting objects from Blender to Unity are considered on the example of preparation and export of a model of a toy bear modeled in Blender. For this purpose, a detailed analysis of the subject area was carried out, the results of which are presented in this article in the form of a step-by-step description of the process of preparing the object for export.

Keywords: 3D objects, import, export, 3D models, modeling, Unity, Blender

# Технических рекомендаций подготовки Технологии создания 3D объектов с объекта к импорту в игровой движок использованием высокополигонального Проверить размеры созданного 3D-объекта моделирования Проверить направления нормалей. Проверить направления локальных осей 3D – редактора и игро Трансформации 3D модели, должны завершаться сбросом углов поворота и на значения по умолчанию. Проверить местоположение точки Origin. Почистить историю, чтобы при экспорте не увеличивался размер файла пой информацией. рођу на основе насоконолиговальной Создание UV резвертки для low-poly Проверить правильность имен импортируемых объектов. Почистить сцену от лишних объектов. ее на low-poly Проверить созданные зацикленные анимации – первый и последний кадр ы совпазать. Технических рекомендаций подготовки объекта к импорту в игровой движок Технологии создания 3D объектов с использованием высокополигонального Трансформации 3D модели, должны инвершиться обросом углов поворота мод умолчанию. моделирования шъность пыен импортируемых объектов ee sa low-poly

### Graphical annotation (Графическая аннотация)

Введение. Сегодня уровень развития информационных технологий предоставляет широкие возможности и богатый арсенал программных продуктов, языков программирования и технологий для успешного решения задач по 3D-моделированию и созданию 3D-игр и видео с использованием 3D-объектов. Одним из получивших широкое распространение программных продуктов, позволяющим легко моделировать и редактировать 3D-модели, предназначенные для использования в играх, видео, рекламе, является Blender [1].

Blender – программный продукт, предназначенный для создания 3D-визуализаций: от статичных объектов до редактирования видео, разработанный анимационной студией NeoGeo [2]. В 1995 году компания начала переписывание имеющихся у них 3D-инструментов, что стало началом создания Blender, последняя же выпущенная версия Blender 2.80 вышла на рынок в июле 2019 года. Одной из основных особенностей Blender является то, что он имеет открытый исходный код и хорошо подходит не только частным лицам, но и небольшим студиям, которые могут создавать 3D-видеоролики, имея в своем арсенале один лишь Blender. Данный продукт поддерживает такие инструменты, как рендеринг, моделирование, анимацию и перенастройку созданных объектов, композинг, текстурирование и другие типы симуляций. Также Blender предоставляет пользователю случай расширять возможности созданных моделей при помощи языка Python.

Если говорить о создании игр с использованием 3D-моделей, то хочется выделить для этих целей Unity 3D. Blender имеет свой игровой движок, однако возможности, предоставляемые Unity, являются более широкими в этом плане.

Unity – мощный межплатформенный продукт для создания компьютерных игр [3]. Он поддерживает создание игр для более чем 20 операционных систем. Первый выпуск продукта состоялся в 2005 году, последняя версия датируется апрелем 2019 года. Основным достоинством продукта является наличие визуальной среды разработки, межплатформеная поддержка и модульная система компонентов. Unity позволяет создавать отличные игры и даже имеет свой редактор 3D-объектов, однако на сегодняшний день возможности встроенного 3D-редактора достаточно ограничены [4], поэтому требуют разработки сложных 3D-объектов в сторонних программных продуктах, таких как Blender. Unity является коммерческим продуктом для бизнеса, однако предоставляет бесплатную версию продукта для обучения. Для создания 3D-игры в Unity необязательно хорошо знать программирование, а необходимый программный код пишется на С#, при необходимости их можно скачать с AccetStore.

С учётом вышеизложенного выбор программных продуктов Blender и Unity 3D для моделирования 3D-объектов и создания компьютерной игры представляется очевидным.

Разработка технологии создания 3D-объектов с использованием высокополигонального моделирования. Если при создании объектов с помощью примитивов или экструдированием количество полигонов мы примерно можем контролировать сразу, то, как говорилось ранее, при скульптурировании количество полигонов может быть огромным, и после окончания моделирования формы объекта необходимо их количество сокращать. Таким образом, при создании объектов с использованием высокополигональной модели должен быть продуманный технологический процесс, что позволит существенно сократить сроки разработки путем уменьшения ошибок и, соответственно, временем на их исправление. Кроме того, надо понимать, что, как правило, высокополигональное моделирование используется для создания более сложных объектов, таких как персонажи, поэтому для анимирования таких объектов требуется создание скелета, по которому можно будет управлять моделью при создании анимации.

Ниже на рисунке 1 представлена схема этапов создания 3D-объекта в редакторе.

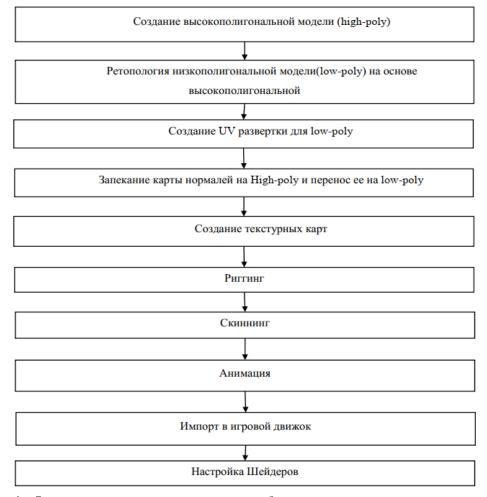


Рисунок 1 – Схема технологического процесса создания объекта

Представленная схема применима и при создании 3D-объектов с использованием только низкополигональной модели при исключении или смене местами некоторых этапов. Таким образом, местами могут быть сменены даже первые 2 пункта, этому может способствовать ряд различных факторов, например:

- 1) создание ряда 3D-объектов, имеющих сходное строение тела:
- 2) необходимость увеличения детализации и реалистичности уже имеющихся объектов.

Описание метода подготовки импорта в игровой движок. Завершающим шагом при создании 3D-объекта под проект является его импорт в среду использования. Как правило, направления осей координат, масштабы и правильная ориентация объектов в пространстве различаются в 3D-редакторе и игровом движке. Таким образом, для корректного импорта объекта в игровой движок необходимо выполнить его подготовку к экспорту из 3D-редактора в самом Blender.

Описание метода настройки шейдеров. Наложение созданных текстурных карт на объект и настройка материалов осуществляется с помощью редактора шейдеров. В Blender есть возможность создания встроенных шейдеров, в Unity для этих целей можно воспользоваться встроенными в игровой движок шейдерами или создать свои с помощью редактора шейдеров Shader Forge, который является плагином Unity. Так как Unity не поддерживает созданные процедурно текстуры из Blender, то настройку шейдером будем делать в Unity, используя заранее подготовленные текстурные карты [5].

Совместимость форматов Blender и Unity. Blender предоставляет широкие возможности для моделирования, одним из преимуществ этого программного продукта является то, что в нем собраны возможности для моделирования различными методами: это и инструменты для скульптурирования, и возможности работы со сплайнами, кривыми Безье, и полигональное моделирование. Каждый из методов имеет свои преимущества и недостатки. В результате моделирования мы получаем модель, которую можем в этом же редакторе оттекстурировать, а также задать анимацию. Полученную модель можно использовать как для создания видео или статичной картинки в самом Blender, так и экспортировать модель в другие программные продукты, где использовать для конечных целей. Например, в Unity для использования модели при создании компьютерной игры.

Unity поддерживает различные форматы для импорта 3D-моделей, включая такие частные форматы программ для моделирования, как .blend (расширение Blender), .mb (расширение Maya), .c4d (расширение Cinema 4D). Кроме того, из Blender можно экспортировать не только отдельные объекты, но и целые сцены.

Если говорить о выборе формата для экспорта, то он зависит в первую очередь от личных предпочтений моделлера. Например, форматы .FBX, .dae (Collada), .3DS, .dxf и .obj являются общими экспортированными 3D-форматами. Преимуществами их использования является следующее:

- такие файлы имеют меньший вес относительно файлов собственного формата 3D-редактора;
- такие файлы позволяют использовать в Unity модели, которые были созданы в программных продуктах, собственные форматы которых не поддерживаются в Unity,

Но недостатком выбора таких форматов может являться то, что при необходимости изменения модели, необходимо возвращаться к исходному файлу в собственном формате 3D-редактора.

Если использовать в Unity модели в формате частного формата (.blend), то это определенно облегчает процесс редактирования объектов, а также снижает риск потери исходного файла объекта, так как импортированный в Unity файл можно будет выгрузить в редактор. Однако такие файлы могут иметь больший вес и содержать лишние компоненты.

Но все же при экспорте объектов важным является не расширение импортируемого файла, а предварительная подготовка созданной модели.

В руководстве Unity 3D есть следующие подсказки по подготовке объектов к экспорту:

- для мешей необходимо почистить историю их построения, удалить неоднородные рациональные безьесплайны, неоднородные рационально сглаживаемые сетки, а также проверить, чтобы все полигоны являлись трех- или четырехугольными;
- текстуры должны загружаться либо из папки проекта Unity, либо из папки \textures проекта;
- также необходимо проверить необходимость групп сглаживании и/или сглаживаний мешей.

Кроме этого, в руководстве Unity есть предупреждение, что камеры и источники света в Unity импортируются некорректно.

Если говорить об экспорте объектов из Blender в Unity, то в первую очередь стоит сказать, что в Unity можно импортировать:

- все узлы с положением, вращением и масштабом. Центры вращения и имена тоже импортируются;
  - меши с вершинами, полигонами, треугольниками, UV и нормалями;
  - кости;

- skinned меши (меши с привязкой к костям);
- анимании

Разработка технических рекомендаций подготовки объекта к импорту в игровой движок. Вlender предоставляет широкие возможности для моделирования, одним из преимуществ этого программного продукта является то, что в нем собраны возможности для моделирования различными методами: это и инструменты для скульптурирования, и возможности работы со сплайнами, кривыми Безье, и полигональное моделирование. Каждый из методов имеет свои преимущества и недостатки. В результате моделирования мы получаем модель, которую можем в этом же редакторе оттекстурировать, а также задать анимацию.

Перед экспортом объекта необходимо провести некоторые подготовительные работы:

- 1) проверить размеры созданного 3D-объекта. Необходимо учитывать, что в Blender единицы измерения по умолчанию одни, а в Unity другие. Несомненно, размеры объекта можно подгонять и в игровом движке, однако стоит учитывать, что при уменьшении или увеличении размеров объекта могут вылезти недостатки, которые при изначальных размерах не были заметны;
- 2) проверить направления нормалей. Как правило, в Blender в процессе создания видны полигоны с неправильным направлением нормалей, однако их можно не заметить, если не просмотреть объект со всех сторон. Данную ошибку можно исправить в игровом движке, однако подобные методы будут ресурсозатратными, поэтому лучше проводить пересчет нормалей в 3D-редакторе;
- 3) проверить направления локальных осей 3D-редактора и игрового движка. Одной из главных проблем при импорте моделей является путаница осей, в которых экспортируется готовый объект. В Blender осью, направленной вверх, является ось z, а осью, определяющей направление «лица» модели (forward, перед), является ось –у. Это можно увидеть на рисунке 1, а в Unity вертикальной осью является ось y, а «передом» модели ось z. Таким образом, для правильного экспорта в Unity 3D-модели в Blender модель должна смотреть лицом в направлении оси –у (минус у), а верхушкой в направлении положительной оси z;
- 4) трансформации 3D-модели должны завершаться сбросом углов поворота модели на значения по умолчанию. Для выполнения сброса углов поворота и совмещения локальных осей 3D-модели (или оси сразу нескольких 3D-моделей) с глобальными осями координат необходимо выбрать 3D-модель в режиме объекта и нажать сочетание клавиш Ctrl+A, вызвав в открывшемся меню Apply. В этом меню необходимо выбрать пункт «Rotation». В меню Applay, помимо сброса поворота модели и всего остального, присутствует ещё и пункт сброса масштаба модели. Это связано с тем, что после масштабирования 3D-модели на сцене в режиме объекта значение параметров масштабирования также необходимо сбросить на значения по умолчанию, выбрав в упомянутом выше меню «Scale»;
- 5) проверить местоположение точки Origin. Для объектов, не имеющих скелета, скорее всего, точка Origin была пересчитана перед созданием им анимации, однако если речь о персонажах, то в их анимации точка Origin не принимала участия, а для правильной работы персонажа в игровом движке необходимо точку переместить на уровень ступней персонажа, таким образом персонажи будут ходить по плоскости, а не проваливаться под нее;
  - 6) почистить историю, чтобы при экспорте не увеличивался размер файла ненужной информацией;
- 7) проверить правильность имен импортируемых объектов. Правильное наименование экспортированных объектов из 3D-редактора необходимо для ускорения процесса настройки объектов в игровом движке. Названия объектов следует писать на английском языке без дополнительных символов в начале (например, тире), так как списки объектов в игровых движках располагаются, как правило, в алфавитном порядке;
- 8) следует соблюдать чистоту сцены. Пустые или лишние скрытые объекты могут импортироваться в файл FBX вместе с объектом. В результате чего размер файла увеличится и возникнет неудобство в работе с ним в игровом движке;
- 9) следует сохранять текстурные карты в правильном формате. Предпочтительным форматом текстурных карт является PNG. Данный формат прост и удобен в редактировании, корректно сохраняет картинку и имеет альфа-канал. Формат PSD также поддерживается игровыми движками (автоматически сливает слои), удобен, когда текстура находится в процессе изготовления, но для использования в игровом движке следует заменить этот тяжелый многослойный формат, так как он сильно увеличивает размер проекта игры. Более подробно о чтении форматов текстур игровыми движками можно прочитать в документации к ним;
- 10) проверить созданные зацикленные анимации первый и последний кадр должны совпадать. Малейшее отклонение по оси, направленной вверх, (как правило, оси Y) может привести к постепенному проваливанию или поднятию персонажа относительно плоскости в процессе игры;
  - 11) проверить объекты на наличие полигонов, просматриваемых с двух сторон;

12) проверить растровую карту на соответствие текстур UV развертке в местах близкого расположения частей с разными текстурами;

13) проверить в анимации корректность перемещения полигонов.

Ошибки при экспорте. Ошибки в переопределении осей координат. Одной из главных проблем при импорте моделей является путаница осей, в которых экспортируется готовый объект.

В Blender осью, направленной вверх, является ось z, а осью, определяющей направление «лица» модели (forward, перед), является ось -у. Это можно увидеть на рисунке 2, а в Unity вертикальной осью является ось у, а «передом» модели – ось z (рис. 3).

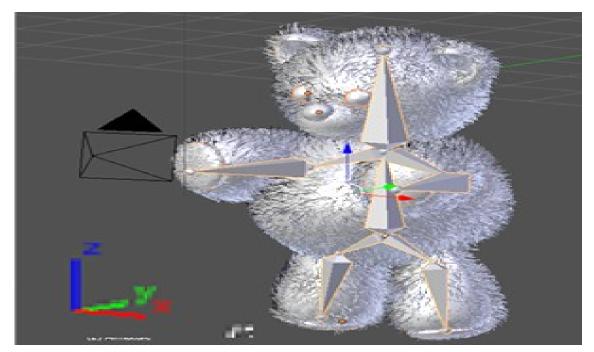


Рисунок 2 – Направления осей в Blender

Таким образом, для правильного экспорта в Unity 3D-модели в Blender модель должна смотреть лицом в направлении оси -у (минус у), а верхушкой - в направлении положительной оси z.

Но и здесь есть свои нюансы, когда объект поворачивается в объектовом режиме, то изменяются направления его локальных осей. Поэтому подобные трансформации 3D-модели должны завершаться сбросом углов поворота модели на значения по умолчанию.

Для выполнения сброса углов поворота и совмещения локальных осей 3D-модели (или оси сразу нескольких 3D-моделей) с глобальными осями координат необходимо выбрать 3D-модель в режиме объекта и нажать сочетание клавиш Ctrl+A, вызвав в открывшемся меню Apply. В этом меню необходимо выбрать пункт «Rotation». В меню Applay, помимо сброса поворота модели и всего остального, присутствует ещё и пункт сброса масштаба модели. Это связано с тем, что после масштабирования 3D-модели на сцене в режиме объекта значение параметров масштабирования также необходимо сбросить на значения по умолчанию, выбрав в упомянутом выше меню «Scale».

Однако Blender, несмотря на подобные подготовки объекта, при экспорте может иметь проблемы с поворотом модели, то есть, даже проделав сброс углов и совмещение осей, может оказаться так, что в результате экспорта объекта в Unity модель будет смотреть лицом в сторону отрицательной оси у, а «верх» – в сторону z. Это связано с тем, что в настройках осей для экспортируемой модели в Blender по умолчанию стоит Forward: -y, Тор: z, то есть оси перевернуты, и Blender предлагает в качестве «лица» модели использовать отрицательное направление оси у, а в качестве «верха» – ось z. Поэтому необходимо изменять их на систему координат Unity, выставив в настройках для осей нужные параметры, и только после этого экспортировать модель.

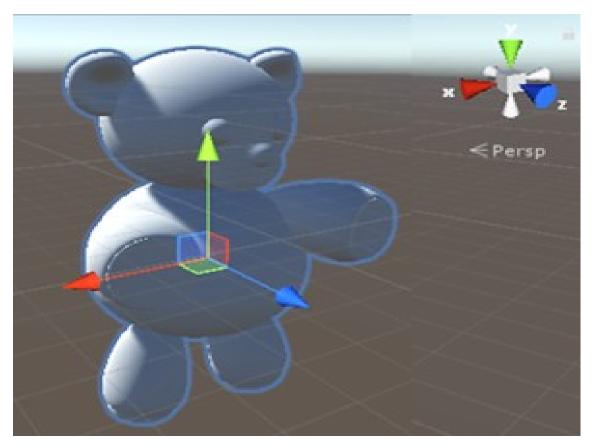


Рисунок 3 – Направления осей в Blender

Но и это может не помочь, поэтому следует повернуть 3D-модель следующим образом:

- 1 шаг. z верх, –z низ модели, лицо в направлении оси –у.
- 2 шаг. После чего выбрать 3D-модель вида «справа»

3 шаг. Необходимо повернуть ее вокруг оси х по часовой стрелке на 90 градусов. Это выполняется нажатием на клавишу «R», затем нажимается клавиша «X» для выбора оси, затем вводится значение 90 – угол поворота, подтверждение выполняется нажатием клавиши Enter.

4 шаг. Теперь необходимо сбросить значения поворота и масштаба модели на значения по умолчанию и повернуть модель в исходное положение (на минус 90 градусов вокруг оси х), и только после этого можно смело экспортировать 3D-модель для использования в Unity3D.

Ошибки при экспорте. Ошибки, связанные с точкой Origin. Кроме осей, при экспорте объектов важным является точка Origin — в Unity она является центральной или нулевой точкой модели, относительно которой производится поворот 3D-модели. Если импортировать в Unity одну и ту же модель 2 раза, но в первый раз точку Origin оставить в середине нижней границы объекта по умолчанию, а во второй раз объект сместить относительно этой точки, тогда при вращении объекта в Unity модели будут вести себя по-разному. Первый будет вращаться на месте, а второй — вокруг невидимого объекта. На самом деле оба объекта будут вращаться вокруг точки Origin, однако для первого объекта она будет являться осью самого объекта, а для второго — точкой, вокруг которого ходит объект. Такому объекту очень сложно описать функции движения. Поэтому для устранения проблемы точку Origin необходимо будет сместить, что можно сделать, как вернув модель в Blender, так и в Unity при описании скриптов.

Кроме прямых способов редактирования точки Origin, можно создать «костыль» для таких объектов. Для этого на сцене создается пустышка, и пустышку назначают объектом-родителем такой неправильно экспортированной 3D-модели, а саму 3D-модель, соответственно, дочкой объекта-пустышки. Но такой костыль может оказаться весьма громоздким, поэтому проще всего исправлять такую ошибку на этапе моделирования и подготовки к экспорту.

Также существует ошибка, когда точка Origin находится не по нижнему краю объекта, а в середине объема фигуры, тогда после экспорта в Unity модель будет проваливаться под землю. В таких случаях также необходимо исправлять расположение точки Origin.

Особенности текстурирования. Как правило, задание материалов и текстурирование – это последний шаг при создании 3D-модели, и при экспорте/импорте 3D-модели из Blender в Unity3D хочется, чтобы материалы и карты нормалей сразу назначались для модели.

Как говорилось ранее, текстуры должны загружаться либо из папки проекта Unity, либо из папки «\textures» проекта, поэтому как бы мы ни старались, сохранить модель и текстуры единым файлом не получится. Однако это не значит, что процесс подготовки текстур слишком сложный.

Для объекта в свойствах в разделе материала необходимо добавить новый материал.

После этого необходимо создать текстуру материалу и выбрать изображение, которое следует использовать для нее. В Blender есть существенное ограничение, за один раз можно запечь только 1 тип материала: прозрачность, глянец, свечение и т.д. Поэтому если объект сложный – требует разных текстур, то необходимо будет воспользоваться дополнительным программным обеспечением с более широкими возможностями для текстурирования объекта.

После того как предыдущие действия будут выполнены, необходимо будет создать еще одну текстуру и назначить ей карту нормалей [6]. Процесс добавления карты нормалей почти ничем не отличается от добавления диффузной карты.

В результате создания карты нормалей может получиться не то, что вы хотели увидеть. Это связано с тем, что текстура будет теперь отображаться на поверхности 3D-модели как обычная диффузная карта. Для того чтобы карта нормалей отображалась как карта нормалей в Blender, необходимо обозначить назначенную 3D-модели текстуру как карту нормалей в меню текстуры во вкладках «Image Sampling» и «Марріng».

Возвращаясь к тому, что в Blender экспорт текстуры не получится сохранить вместе с 3D-моделью, решение очень простое: 3D-модель можно экспортировать в формате FBX, а текстуры приложите к ней во время импорта 3D-модели в Unity3D.

В итоге получится, что при импорте в Unity необходимо выбрать файл FBX и две карты: карту нормалей и диффузную карту (текстуру) и перетащить их все вместе в одну из папок «Assets» в Unity3D. Если все было сделано правильно, то сразу после импорта Unity3D предложит пометить карту нормалей как карту нормалей. Данное предложение можно принять либо отказаться и настроить атрибуты для изображения в меню настройки изображения в «Inspector» либо в меню материала.

Теперь при добавлении импортированной 3D-модели на сцену модель будет добавляться вместе с материалами и текстурой, которые были созданы в Blender.

Что касается объектов, содержащих волосы или ворсинки, созданные с помощью системы частиц в Blender, такие текстуры импортировать в Unity не получится. Для создания объектов, имеющих шерсть или волосы, необходимо либо систему частиц с помощью ряда модификаторов перевести в меши, в результате чего получится «highpoly» объект, что совершенно не подходит для создания игр. Либо необходимо создание имитации волос путем наложения соответствующих текстур.

Заключение. Для работы в сфере 3D-моделирования в зависимости от целей может быть достаточно и одного программного продукта [7, 8], однако, как правило, возможностей одной программы недостаточно для реализации задуманного проекта. Unity - не только профессиональный мощный инструмент для создания игр и иных приложений с использованием 3D-моделирования, но и неплохо оснащённая платформа для создания простых 3D-объектов, недостающих для визуализации каких-либо сцен. Однако более сложные объекты удобнее всего создавать в специальном редакторе Blender, после чего полученный объект импортировать в Unity, хоть и процесс импортирования имеет свои нюансы.

В статье были рассмотрены главные ошибки, возникающие в процессе экспорта объектов и способы их устранения, что значительно облегчит результат переноса объектов из Blender в Unity, так как возможные проблемы будут устраняться на этапе подготовки объектов к экспорту.

Таким образом были сделаны следующие выводы:

- основные сложности экспорта объектов из Blender в Unity связаны с различием ориентации осей в этих программах;
- для объектов, которые в Unity будут использоваться как динамические, важно правильное определение точки Origin;
  - материал и текстуры для объекта нельзя сохранить единым файлом с объектом;
- при создании объекта в Blender можно предусмотреть только единую текстуру для всего объекта;
  - импорт в Unity системы частиц, созданных в Blender, невозможен напрямую;
  - выбор формата для экспорта из Blender в Unity зависит от предпочтений моделлера.

#### Библиографический список

- 1. Blender. Режим доступа: https://www.blender.org/about/license/, свободный. Заглавие с экрана. Яз. англ.
- 2. Blender 2.80 Руководство. Режим доступа: https://docs.blender.org, свободный. Заглавие с экрана. Яз. англ.
- 3. Хокинг Джозеф. Unity в действии. Мультиплатформенная разработка на С# / Хокинг Джозеф. Санкт-Петербург: Питер, 2016. 336 с. ISBN 978-1617292323.
  - 4. Unity. ProBilder. Режим доступа: https://unity3d.com/, свободный. Заглавие с экрана. Яз. англ.
- 5. Кенни Ламмерс. Шейдеры и эффекты в Unity. Книга рецептов / Кенни Ламмерс. Москва : ДМК-пресс, 2016. 274 с.
- 6. Запекаем карты нормалей в текстуру и ленивое текстурирование 3D-модели через авторазвертку и запекание текстуры в Blender для игрового движка Unity3D. Режим доступа: http://www.3dbuffer.com/articles/zapekaem-kartyi-normaley-v-teksturu-i-lenivoe-teksturirovanie-3d-modeli-cherez-avtorazvertku-i-zapekanie-teksturyi-v-blender-dlya-igrovogo-dvizhka-unity3d/#.Xc\_ByVczbIU, свободный. Заглавие с экрана. Яз. рус.
- 7. Бикмуллина И. И. Разработка методики автоматизированного синтеза информационных систем на основе семантических отношений предметной области / И. И. Бикмуллина, И. А. Барков, А. П. Кирпичников // Вестник Казанского технологического университета. 2015. Т. 18, № 7. С. 236–242.
- 8. Шаехов И. М. Организация визуального представления данных / И. М. Шаехов, И. И. Бикмуллина, А. П. Кирпичников // Вестник Казанского технологического университета. 2019. Т. 22, № 6. С. 167–169.

#### References

- 1. Blender. Available at: https://www.blender.org/about/license/
- 2. Blender 2.80 Manual. Available at: https://docs.blender.org
- 3. Hawking Joseph. Unity in action. Multiplatform development in C#. St. Petersburg, Piter Publ., 2016. 336 p. ISBN 978-1617292323.
  - 4. Unity. ProBilder. Available at: https://unity3d.com/
  - 5. Kenny Lammers. Shaders and effects in Unity. A book of recipes. Moscow, DMK-press Publ., 2016. 274 p.
- 6. Zapekaem karty normaley v teksturu i lenivoe teksturirovanie 3D-modeli cherez avtorazvertku i zapekanie tekstury v Blender dlya igrovogo dvizhka Unity3D [Bake normal maps into a texture and lazy texturing of the 3D model through auto-expansion and baking of the texture in Blender for the Unity3D game engine]. Available at: http://www.3dbuffer.com/articles/ zapekaem-kartyi-normaley-v-teksturu-i-lenivoe-teksturirovanie-3d-modeli-cherez-avtorazvertku-i-zapekanie-teksturyi-v-blender-dlya-igrovogo-dvizhka-unity3d/#.Xc ByVczbIU
- 7. Bikmullina I. I., Barkov I. A., Kirpichnikov A. P. Razrabotka metodiki avtomatizirovannogo sinteza informacionnykh sistem na osnove semanticheskikh otnosheniy predmetnoy oblasti [Development of methods for automated synthesis of information systems based on semantic relations of the subject area]. *Vestnik Kazanskogo Tekhnologicheskogo universiteta* [Bulletin of the Kazan technological University], 2015, vol. 18, no. 7, pp. 236–242.
- 8. Shaekhov I. M., Bikmullina I. I., Kirpichnikov A. P. Organizatsiya vizualnogo predstavleniya dannykh [Organization of visual representation of data]. *Vestnik Kazanskogo Tekhnologicheskogo universiteta* [Bulletin of the Kazan technological University], 2019, vol. 22, no. 6, pp. 167–169.