ТЕХНОЛОГИЧЕСКИЕ ТРЕБОВАНИЯ 3D-МОДЕЛИРОВАНИЯ ПРОЕКТА К2

Версия 3.5-К2

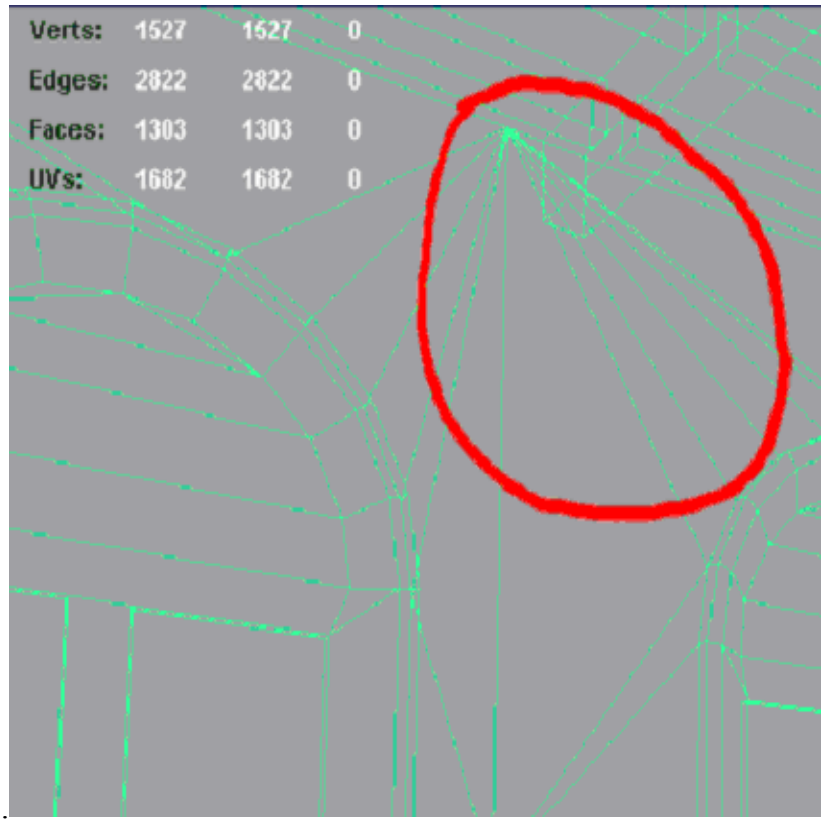
1. Начальные требования

1. Войти в меню Window > Settings / Preferences > Preferences… и в категории Settings установить систему измерения в метрах, углы в градусах, а количество кадров анимации в 15 fps.

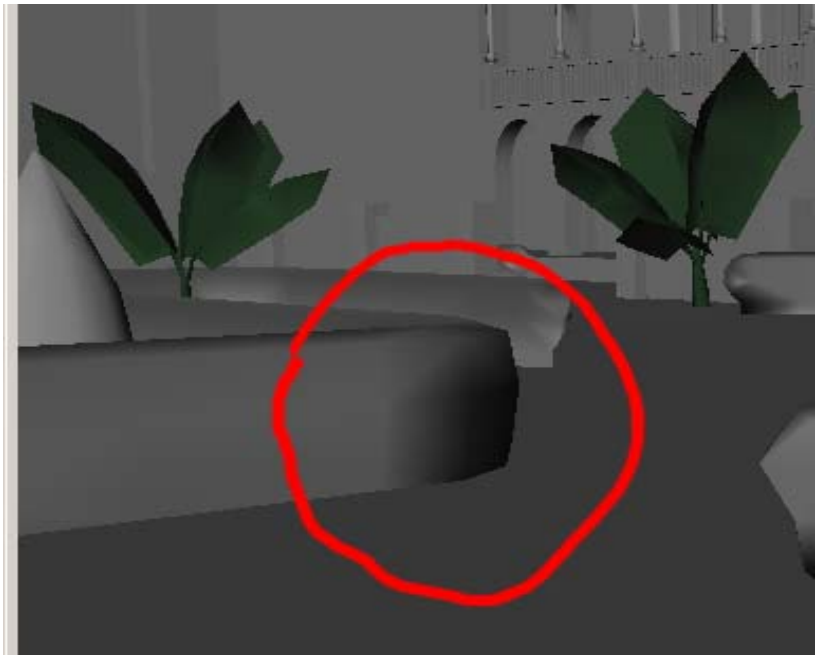
2. Подготовка элементов сцен

Этап полигонального моделирования

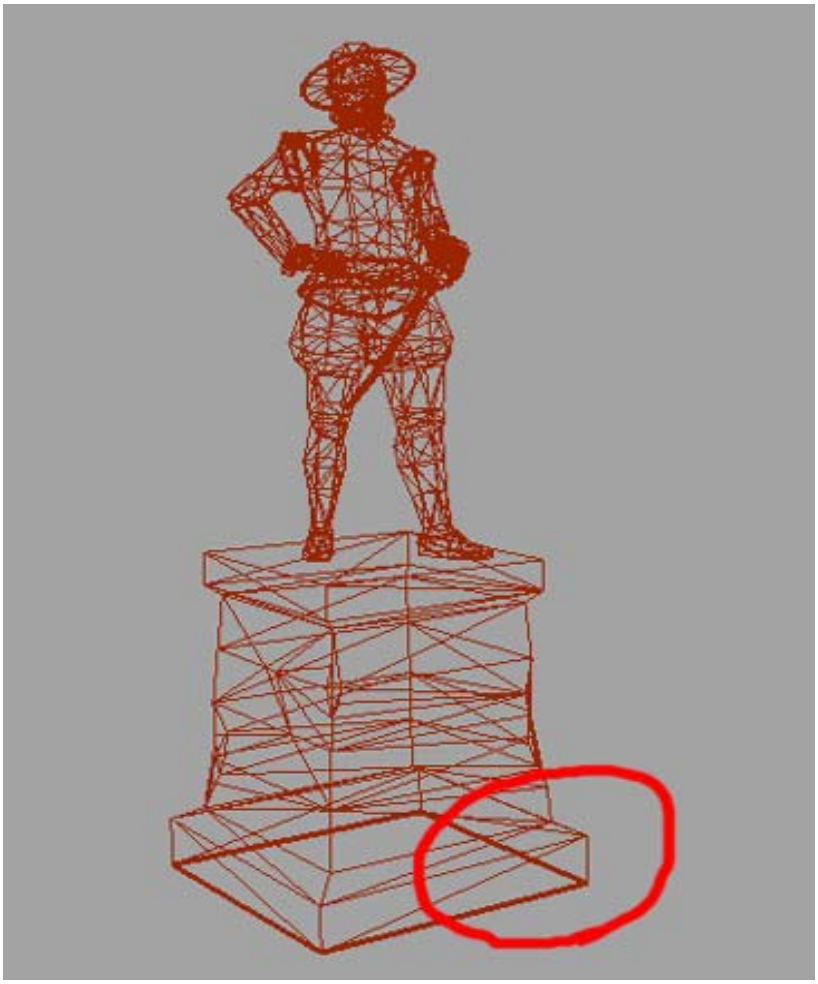
1. Для полного контроля над пропорциями при изготовлении моделей, пользоваться временным объектом, описывающем виртуальный размер игрового персонажа.
2. Должно быть **полностью** исключено использование double side полигонов и полигонов с функцией outside. Для изготовления тонкостенных предметов (листья, ткани, посуда и т.д.) пользоваться дублированием объекта с выворачиванием нормалей. Эти объекты после процедуры Combine не должны подвергаться Merge Vertices во избежание появления двусторонних полигонов с общими вершинами.
3. Конечные модели должны быть **цельными**, состоящими из одного объекта, не разбитыми на компоненты. Не должны иметь несшитых между собой компонентов. Допускается вертексы краёв мелких деталей объекта припускать (прятать) под плоскости соседних полигонов.
4. Полигоны моделей с открытыми плоскостями (стены, полы, и т. д.) не должны иметь треугольники с углами менее 20 градусов



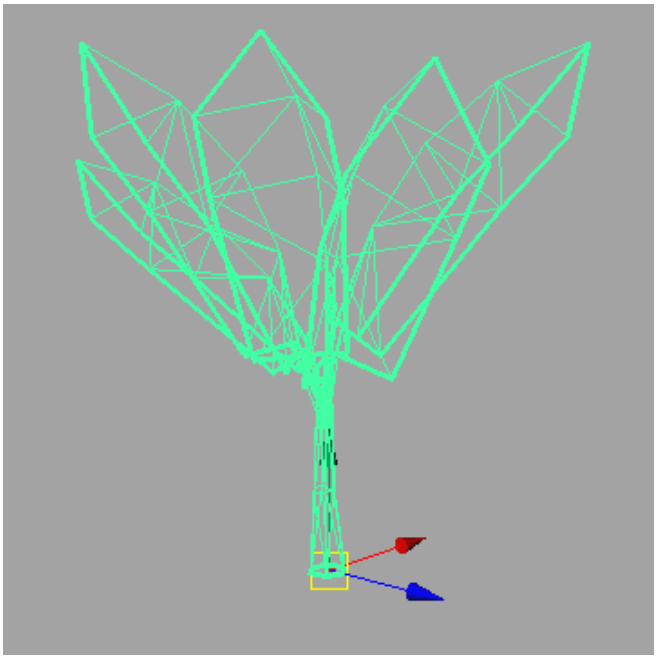
5. Угол сглаживания edges должен отражать художественный замысел модели. Не должно быть так, чтобы вся модель была или с полностью сглаженными edges, или с полностью не сглаженными edges.
6. Нормали вершин объектов должны иметь правильное направление и не приводить к появлению черных или затененных объектов.



7. Конечные модели должны иметь включенное отображение Border Edges с толщиной 3.0, что позволит контролировать несшитые edges.



8. Цельные модели сцены, логически объединенные в один объект (дом, дерево, скала и т. д.), должны быть подвергнуты процедуре Combine, с последующим Merge Vertices, если это целесообразно (**избегать** появления **двусторонних** полигонов с общими вершинами). Желательно объединение родственных объектов в группы и layers.
9. Конечные модели объектов должны иметь функцию Freeze Transformations, их Pivot должен располагаться или в геометрическом центре модели, или в центре основания модели, если она предназначена к расположению на поверхности других объектов при сборке сцен.



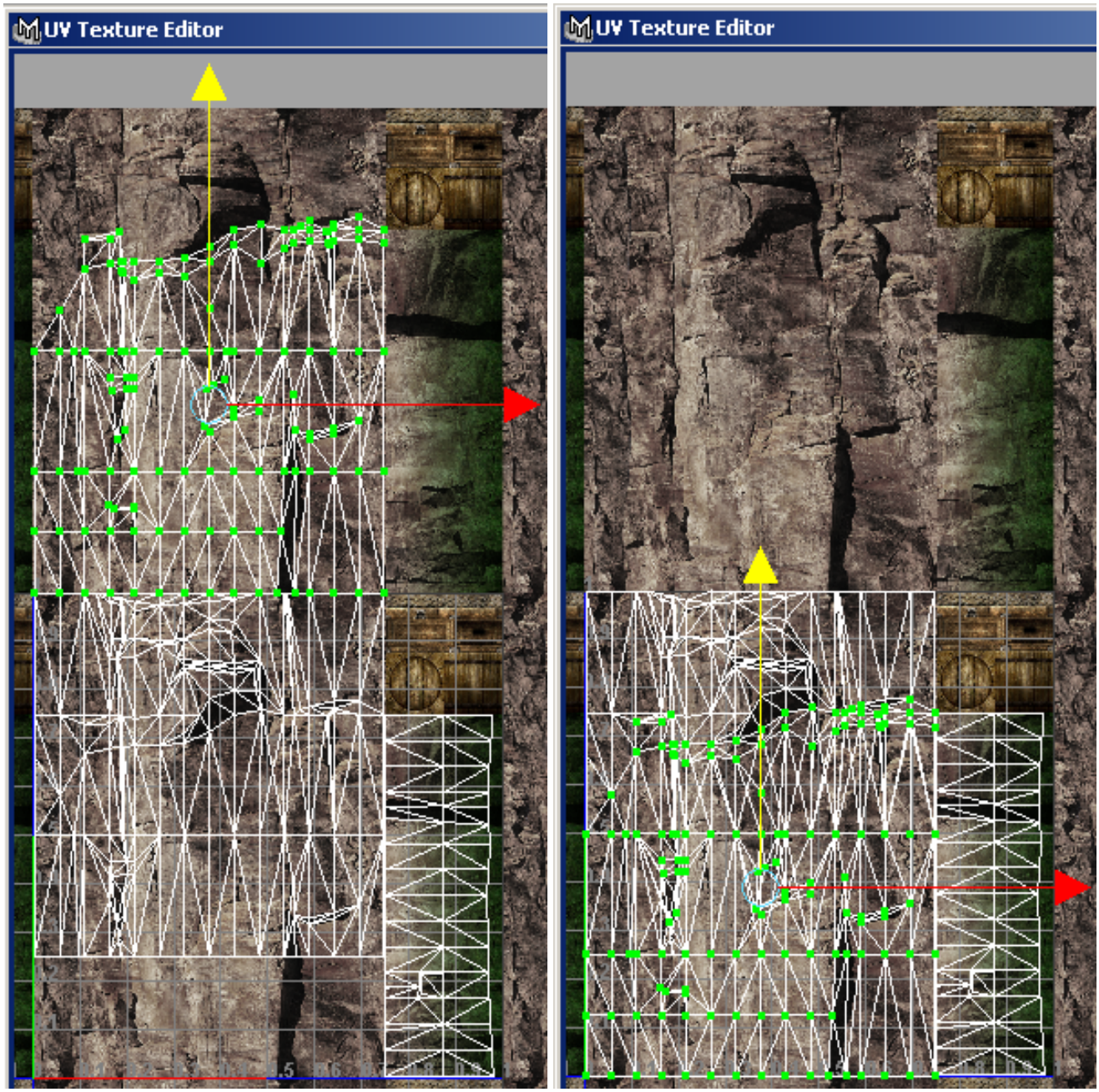
10. При изготовлении ступеней учитывать размеры: высота ступени **0,2** м, глубина **0,3** м.
11. При моделировании учитывать последующее вертексное освещение сцены в игровом движке. Для этого должна поддерживаться определенная плотность полигональной сетки в местах затенения или зонах контрастного освещения. **Важно:** освещение просчитывается путем просчёта попадания фотонов света непосредственно на полигон, а не на вертекс. Поэтому допускается вертексы краёв мелких деталей объекта припускать (прятать) под плоскости соседних полигонов, но только при условии, если они не будут переосвещены с другой стороны (Например: все вертексы локаций внутренних помещений не должны выступать за пределы полигонов стен, так как полигоны, принадлежащие им, будут освещены как и внутренними источниками света, так и внешним).
12. Строго придерживаться оптимизации моделей, даже за счет нескольких полигонов. Учитывать невидимость полигонов, находящихся на обратных сторонах объектов и полигонов, невидимых игроком снизу, с поверхности игрового мира.
13. Имена моделей, групп и layers внутри сцены должны носить понятные имена на английском языке начинающиеся со строчной буквы, или быть сокращениями от английских слов, в конце которых должна быть добавлена определенная цифра, если объекты повторяются. Например: *treebig1*, *treebig2*. Не допускается оставление дефолтных имен типа *pCube1* или *polySurface1*.
14. Из файла модели должны быть удалены все временные и вспомогательные объекты и Nodes.
15. Конечные модели должны быть в обязательном порядке подвергнуты процедурам Delete by Type History.

16. Названия файлов моделей или сцен также должны носить понятные имена на английском языке, или быть сокращениями от английских слов.

Текстурирование моделей

17. Все стандартные модели должны обладать материалами типа *Lambert* с дефолтными настройками, где на атрибут *Color* должна быть назначена 2D текстура *File* (формата TGA), или текстура типа *Layered Texture*, в которой не более чем на два слоя (канал 0 и канал 1), должны быть назначены текстуры типа *File* (также формата TGA). Канал 0 должен содержать уточняющую или грязевую текстуру, канал 1 должен содержать текстуру, предназначенную для *Color*. Режим *Blend mode* канала 0 должен быть включен на *Multiply*. При необходимости использования уточняющей текстуры допускается назначать файл в формате TGA на атрибут *Bump Mapping*. В данном случае UVSet уточняющей текстуры будет считываться с UVSet`а текстуры *Color*. **Важно:** при использовании уточняющей текстуры, или текстуры, изображающей грязевые потёки, сколы, трещины и т.д., которая кладется в канал 0 *Layered Texture* или на атрибут *Bump Mapping,* цвет текстуры в игровом движке организуется по следующей схеме: *Color* + серый цвет RGB 127-127-127 = оригинальный *Color, Color* + белый цвет = ярко-светлый *Color*, *Color* + тёмно-серый цвет = затемнённый *Color*.
18. Имена файлов текстур (изображений) должны носить понятные имена на английском языке, или быть сокращениями от английских слов, иметь в длину желательно не более восьми символов. Имена нодов текстур материалов (шейдеров) **в строгом порядке** должны иметь то же имя, что и файл изображения, используемый в ней, к которому должен быть добавлен суффикс *_1*. Например: *rockligh_1, bumpland_1*. Имена нодов типа Layered Texture должны состоять из имен файлов изображений, входящих в него, с использованием нижней черты между ними и суффикса *_1: имя текстуры, используемой на канал 0 + _имя текстуры, используемой на канал 1 + _1. Например: bumpland_rockligh _1*. Имена нодов самих материалов (шейдеров) должны иметь то же имя, что и файл изображения, используемый в нем, как атрибут Color. В случае использования Layered Texture как атрибут Color, имя материала должно совпадать с именем Layered Texture, но без суффикса *_1*.
19. Модель в обязательном порядке на всех своих полигонах должна содержать *UVSet*'ы (не более двух), и к каждому UVSet'у должна быть прилинкована своя текстура. Имена UVSet'ов должны быть в строгом порядке иметь следующую структуру: *map1* (обязательно) - базовая текстура, *map2* (опционально). UVSet *map2* добавляется только при использовании Layered Texture для составляющей Color. В случае накладывания уточняющей текстуры в поле *Bump Mapping*`а один из UVSetов будет работать с двумя текстурами. Мэппинг учитывает текстурную утилиту *place2dTexture* - трансформацию текстурных координат: *Repeat UV* - тайлинг текстурных координат, *Offset* - смещение мэппинга, *Rotate UV* - поворот текстурных координат. Все остальные опции в place2dTexture не учитываются.
20. Размеры сторон текстур в пикселях должны быть кратны к степени двойки. Наименьший размер текстуры должен быть 16 x 16. Отдельные элементы текстурных листов должны быть собраны из текстур, размеры сторон которых в пикселях также должны быть кратны к степени двойки. Собирать текстурные листы следует из элементов, тождественных по размеру друг к другу. **Не допускается** оставление пустых, неиспользуемых в текстурном листе мест.
21. При изготовлении и выборе текстур соблюдать кратность и масштабность текстур друг к другу, относительно виртуальных единиц измерения.
22. При изготовлении PSD-оригинала текстуры желательно производить логическое наименование слоев словами на английском языке, или сокращениями от английских слов. Не допускать создания лишних слоев. Имя PSD-файла должно совпадать с именем текстуры, в которую он будет сконвертирован.
23. При использовании тайлинга пользоваться **отдельными текстурами**.

24. При работе с UV Texture Editor`ом обязательно отключать фильтрацию текстур и обязательно использовать функцию Pixel Snap для более точного маппинга.
25. При расположении Mapping`а в поле UV Texture Editor`а желательно не допускать пересечения Shell`ов, а располагать их по всей свободной площади окна редактора, по возможности не разбрасывая их в удаленные углы.



Желательно *Нежелательно*

3. Сборка сцен (локаций, уровней) и сдача в игровой движок

1. При подготовке крупных элементов сцен учитывать пункты 1-16, перечисленные выше для этапа полигонального моделирования отдельных объектов, предназначенных к сборке и пункты, касающиеся текстурирования моделей.
2. Руководствоваться логикой и здравым смыслом при сборке сцен.
3. Учитывать, что игровая локация, это не мир с полной свободой, поэтому предусматривать логические ограничения свободы передвижения игровых

персонажей за пределы «патча движения» (см. ниже) путем расстановки преграждений (складки местности, растительность, камни, заборы, ящики и т.д.).
4. При изготовлении ландшафтов избегать чрезмерной детализации мест, недоступных игроку.
5. **При стыковке различных объектов сцены строго учитывать факторы, приводящие к мерцанию швов** (При геометрическом стыке объектов **обязательно** топология края одного объекта должна совпадать с топологией края другого объекта повертексно с функцией Snap to vertex). Не допускать пересечения полигонов, находящихся в одинаковых координатах. (Например: полигональный ковер, лежащий на полигональном полу. Надо: в полу должно быть вырезано отверстие под размер и форму ковра). Разрешается притопление края одного объекта под плоскоть другого объекта, во избежание чрезмерной детализации стыков. (Например: нижний край ствола пальмы может быть спрятан под плоскость поверхности почвы).
6. При сборке сцен строго придерживаться лимита по полигонам, не допуская максимального их числа, исключающего дальнейшую свободу по изменению игрового сюжета в процессе создания игры.
7. Готовая сцена должна содержать ограниченное количество слоев Layers, имеющих логическое наполнение, правильные короткие имена на английском языке и различную цветовую корреляцию. Не допускается использование нескольких слоев для однотипных элементов; принадлежность, через группы, одних объектов к различным слоям; использование дефолтных имен для слоев; наличие пустых слоев.
8. Обязательно должны быть удалены все временные объекты и Nodes из сцены.
9. Вся основная полигональная геометрия должна иметь полигональная
10. Имя конечного файла должно быть утверждено руководителем проекта или лицом, ответственным за соблюдением дизайн-документа.

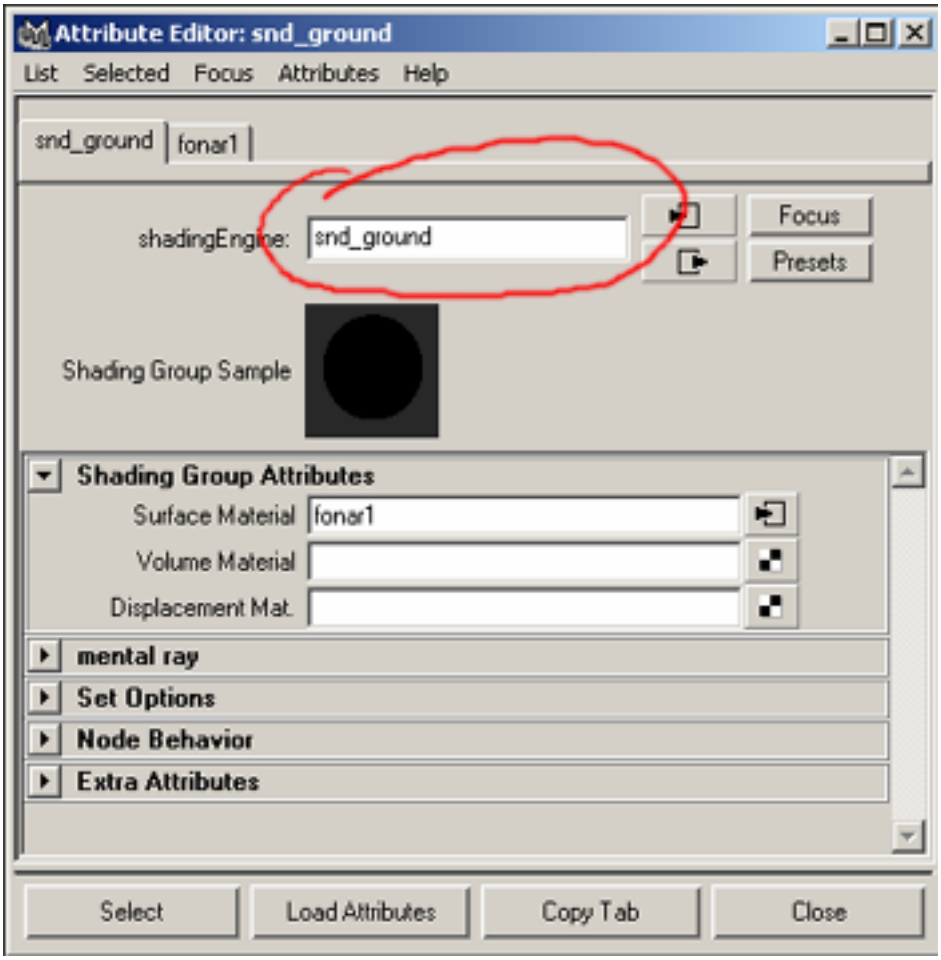
4. Требования к патчам локации

Сцены (локации) имеют определенные технологические элементы, называемые «патчами». Разделяют «патчи движения», «патчи травы» и «прыжковый патч».
Патч движения представляет собой полигональный объект, по которому отслеживается свобода передвижения всех персонажей игры. Причем в некоторых локациях возможно разделение на два патча – «Дневной» и «Ночной», которые могут попеременно подключаться в различное время игровых суток, если при смене времени происходит изменение расположения объектов сцены. (Например: ночью могут исчезать торговцы со своими товарами, освобождая ранее загороженные проходы).
Патч прыжковый определяет место на патче движения, где можно спрыгивать вниз с определенной высоты в сторону от основного патча.
Патч травы – это полигональный объект, по которому происходит генерация спрайтов анимированной травы.

Патч движения

1. Должен повторять (отслеживать) топологию (тесселяцию) поверхности ландшафта или полов, во избежание проникновения ног персонажа под поверхность или продвижения по воздуху над ней. Исключения: а) патчи через ступени лестниц представляют собой плоскости, проходящие через верхние кромки ступенек; б) тесселяция патчей плоских полов не обязательно должна повторять плотность полигонов соответствующего пола.
2. Должен логически ограничивать свободу передвижения персонажей по игровой локации, поэтому он обязан не допускать игрока в те места, откуда могут быть видны места, не предназначенные для глаз игрового персонажа и должен не допускать проникновения персонажа через различные объекты сцены.

3. Патч должен быть цельным неразрывным объектом без резких перепадов высот. Невысокие камни, коряги и др. объекты, через которые персонажу разрешается переступать должны быть покрыты патчем с уклоном не более 45 градусов.
4. Полигоны патчей не должны иметь double side и функцию outside. Их нормали всегда должны иметь направление к персонажу.
5. Патч не должен близко проходить к объектам, ограничивающим пространство сцены, так как персонаж тогда может проникать внутрь этих объектов. Желателен зазор не менее 0,4 м.
6. Патч не должен по краям иметь углы менее 120 градусов. Поэтому обязательно надо закруглять все несоответствующие этому требованию места.
7. Патч должен иметь один UVSet с именем map1, назначенный как Planar Mapping по оси Y.
8. На разные полигоны патча движения обязательно должны быть назначены специальные отдельные материалы (шейдеры), по которым происходит идентификация звуков ходьбы, различных в зависимости от материалов почвы и полов под ногами игрового персонажа. Это должны быть материалы типа *Lambert* с дефолтными настройками, где на атрибут *Color* должны быть назначены любые 2D текстуры *File* (формата TGA) из набора текстур, присутствующих в **обязательном порядке** в этой сцене, но со специальными именами, прописанными в поле s*hadingEngine*, атрибута *Shading Group Sample* материала:



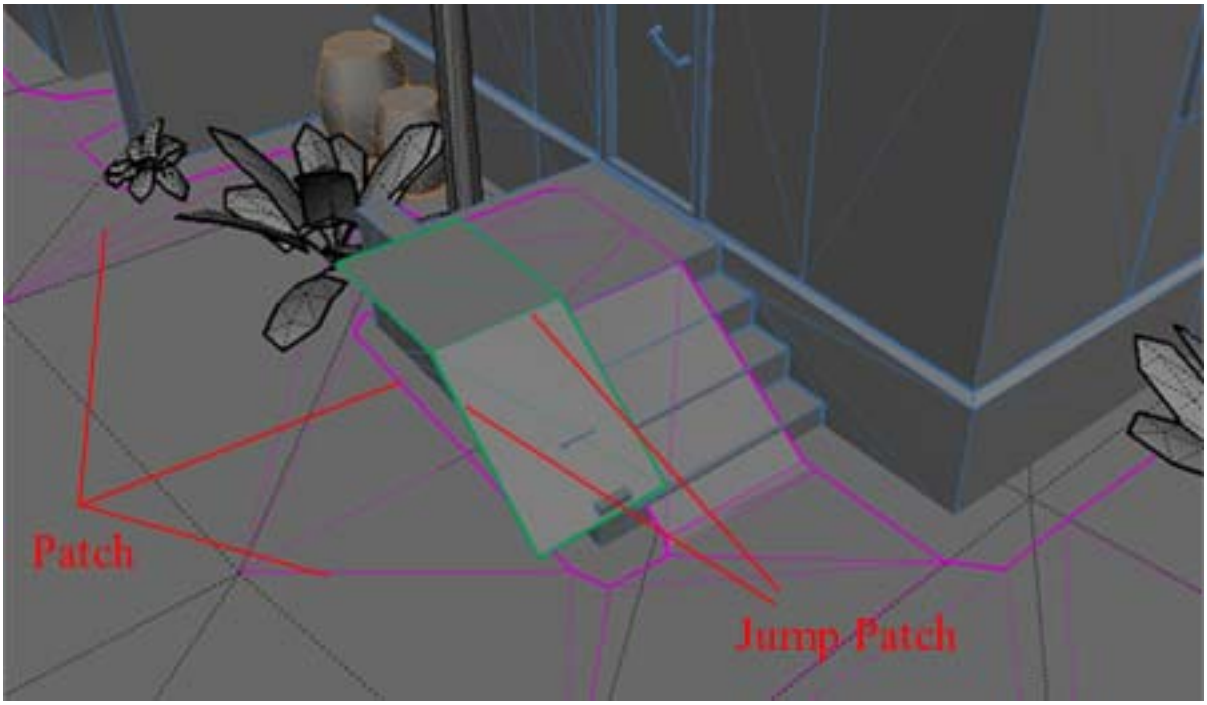
Звук шагов по материалу:	Имя s*hadingEngine*
Деревянный настил, пол	snd_wood
Каменная мостовая	snd_stone
Земляная дорога	snd_ground
Газоны, джунгли	snd_grass
Песчаный пляж	snd_sand
Пол пещеры, подвала	snd_echo
Ковровое покрытие	snd_carpet

Для более естественного соответствия отображения звука шагов относительно текстур поверхности, по которой ходит игровой персонаж, требуется дополнительная тесселяция патча.

9. Конечный патч должен быть в обязательном порядке подвергнут процедурам Triangulate и Delete by Type History.

Патч прыжковый

10. Строится у края патча движения, перекрывая его при виде сверху на небольшое расстояние и на высоте около 0,1 м над ним. Разрешается его устанавливать только в логически обоснованных местах, и там, где прыжок персонажа приземлит его на тот же патч движения, находящийся ниже.

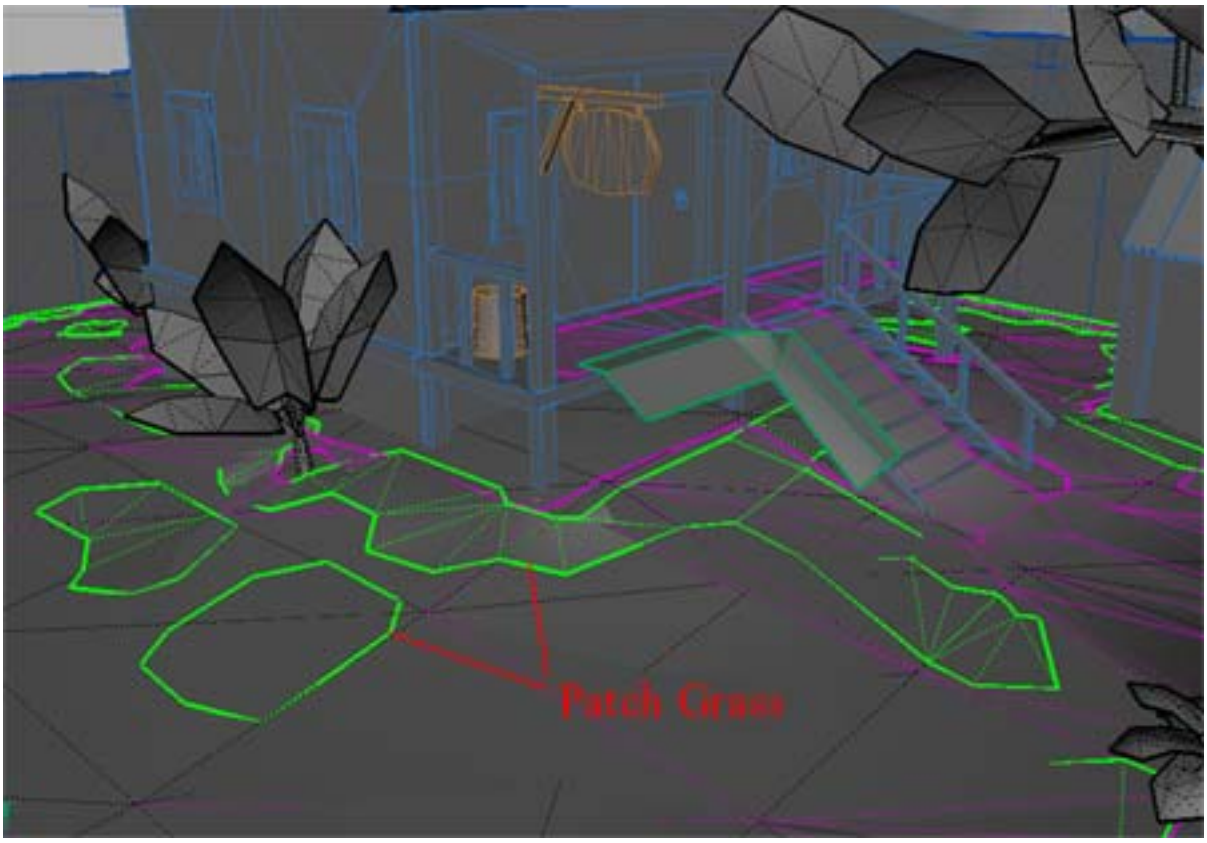


11. В связи с особой спецификой строения патча он не должен быть единым неразрывным объектом, хотя и объединённым в один mesh.

12. Патч должен иметь один UVSet с именем map1, назначенный как Planar Mapping по оси Y.
13. На прыжковый патч назначается любой материал сцены с одной текстурой на канале *Color*.
14. Конечный патч должен быть в обязательном порядке подвергнут процедурам Triangulate и Delete by Type History.

Патч травы

15. Строится по геометрии ландшафта, повторяя собой его топологию. Должен выглядеть в виде неоднородных по форме кусков поверхности. Не допускается его подвисание над поверхностью или глубокое провисание под ней.
16. Нежелательно чрезмерное увеличение площади патча травы относительно площади основного ландшафта, так как генерация травы отнимает ресурсы компьютера.



17. В связи с особой спецификой строения патча, он не должен быть единым неразрывным объектом, хотя и объединённым в один mesh.
18. Патч должен иметь один UVSet с именем map1, назначенный как Planar Mapping по оси Y.
19. На патч травы назначается любой материал сцены с одной текстурой на канале *Color*.
20. Конечный патч должен быть в обязательном порядке подвергнут процедуре Delete by Type History.

5. Иерархия сцены

Собранная для выгрузки в игровой движок локация должна иметь определенную иерархию и определенные технологические требования.
Вверху иерархии стоят локаторы, к которым припарентчены (Parenting) соответствующие полигональные объекты или группы объектов, в том числе группы локаторов.
Полная схема локации со всеми компонентами выглядит следующим образом:

root

Группы полигональных объектов или сами объекты, отвечающие за сцену, которая и является собственно геометрией локации (кроме фонарей со стеклом, которые подгружаются отдельно, исчезающих по ночам рынков и полигональных задников, являющихся ширмами заднего вида, на которые кладется текстура джунглей, гор или перспектив горизонта).

fonar_day

Группы полигональных объектов или сами объекты, включающие модели фонарей со стеклом. В игровой движок подгружается отдельной датой.

fonar_night

Номинально является копией предыдущей иерархии fonar_day. Единственное отличие – на эти объекты может быть назначена другая текстура, более соответствующая ночным фонарям. Если фонарь ночью не работает, и является лишь декоративным элементом, на нем должна быть оставлена текстура дневного фонаря. В игровой движок подгружается отдельной датой.

markets

Группы полигональных объектов или сами объекты, включающие модели рынков и разнообразных товаров, которые будут убираться на ночь. В игровой движок подгружается отдельной датой.

back 1

Полигональные объекты (задники), являющихся ширмами заднего вида, на которые кладется текстура джунглей, гор или перспектив горизонта, расположенные к нам на **первом** уровне. Строятся в виде колец или полу-колец, окружающих игровую локацию на определенном расстоянии. В игровой движок подгружается отдельной датой.

back 2

Полигональные объекты (задники), расположенные к нам на **втором** уровне. В игровой движок подгружается отдельной датой.

back 3

Полигональные объекты (задники), расположенные к нам на **третьем** уровне. В игровой движок подгружается отдельной датой.

patch_day

Патч движения персонажей, описывающий зону свободы передвижения **днём**.

patch_night

Патч движения персонажей, описывающий зону свободы передвижения **ночью**. При отсутствии *markets* не требуется.

jump-patch

Патч, описывающий места возможности спрыгивать вниз игровому персонажу.

patch-grass

Патч, описывающий места генерации процедурной травы.

locators

Группы локаторов, которые описывают места определенных событий и процедур игрового мира. Эта иерархия **обязательно** должна содержать в себе кроме групп локаторов особый объект в один полигон с именем *poly*, на который назначается любой материал сцены с одной текстурой на канале *Color*. Он должен быть опущен ниже уровня локации и не видим игровым персонажем.