UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Centre de la Imatge i la Tecnologia Multimèdia

Creación de un Entorno 3D Estilizado

Borja Gómez García

Director: Ismael Navales Farreras

Grau en Multimedia

Curs 2022-23

Universitat Politècnica de Catalunya (CITM)

Índice

Resumen

El proyecto consiste en la creación de un entorno 3D de estética estilizada. Se presentará como una *build* jugable de Unreal Engine 5, en la que el espectador podrá recorrer el espacio, observando cómo se comportan los elementos orgánicos y los detalles de los espacios y localizaciones construidas.

El mismo proyecto engloba varios procesos, partiendo desde el germen de una idea, y su conceptualización, a partir de un estudio y búsqueda de referentes en videojuegos, piezas audiovisuales, y en general, arte estilizado.

Seguido de la creación del propio entorno y todo aquello que lo forma, además del modelado, esculpido, retopología, desplegado de UVs, texturizado e inserción de objetos 3D en el entorno.

El estudio y aprendizaje de recursos y sobre todo del funcionamiento de software desconocido, la programación y creación de materiales y elementos que doten al entorno de personalidad y vida.

Hasta obtener un resultado en el que se apliquen varios de los conocimientos adquiridos durante la estancia del autor en el CITM, además de aquellos obtenidos por su propia cuenta, para generar una pieza que sirva cómo herramienta para acceder al sector laboral en el mundo del 3D y que muestre las capacidades del autor.

La realización del proyecto ha sido significativa en cuanto a la realización de los límites que se pueden alcanzar, la motivación por saber más y más sobre algo que realmente resulta interesante y la tosquedad con la que continuar a pesar de encuentros fortuitos con obstáculos y problemas, de este proyecto nace una ilusión y una pasión por el aprendizaje y continuar creando universos que beben del folklore personal y mundos oníricos.

Palabras Clave

Modelado 3D, Conceptualización, Entornos digitales, Unreal Engine, Texturizado, Optimización de recursos, Aprendizaje, Ambientación.

Enlaces

Build del proyecto

Índice de tablas

Índice de figuras

Glosario

Se pueden encontrar las palabras definidas en el glosario señaladas entre comillas, en cursiva y subrayadas.

Hard surface modeling: Técnica de modelado 3D usada para crear vehículos, maquinaria, armas o cualquier objeto no orgánico con superficies duras y estáticas.

Stylized art: El arte estilizado se centra en la libertad artística por encima de la representación objetiva, normalmente tiende a distorsionar, simplificar o exagerar ciertos elementos o crear representaciones únicas y llamativas.

Workflow: El workflow o flujo de trabajo, en un ámbito creativo, es el conjunto de procesos a seguir para la obtención de un resultado concreto.

Rigging: Es una técnica usada para la animación digital o para el stop motion, a través del cual se crea una estructura que permite deformar y animar personajes de manera sencilla y efectiva.

Retopologia: Es una técnica digital basada en el control y la simplificación de figuras 3D, mediante el calco o el redibujado de una figura ya existente.

Modelos Low y High Poly: Son modelos con un número elevado o reducido de polígonos, pueden referirse a un mismo modelo realizado con menor o mayor detalle.

Primitivas: Las primitivas son formas 3D básicas: prismas, conos, cilindros, esferas, cuñas, pirámides y toroides.

Loops o Edge loops: Se trata de cortes realizados en superficies 3D, que forman un anillo, en el que el primer vértice del corte, suele encontrarse con el último.

Pintado de pesos: Es el proceso por el cual se le asignan pesos a cada uno de los vértices de una malla, indicando cuánta influencia tiene cada hueso del esqueleto sobre los vértices.

Splines: Es una curva en un espacio 3D definida por lo menos, por dos puntos.

Curvas Bézier: Es una curva paramétrica que define una curva suavizada y contínua.

Box modeling: Es la técnica de crear objetos 3D, dándole forma a primitivas existentes.

Booleanas: Son varios tipos de operaciones realizadas con objetos (*suma, sustracción, inserción* y *división*).

Bevel: Un *bevel*, conecta dos objetos o caras mediante una o más caras anguladas.

3D sculpting: Es el uso de software que ofrece herramientas para empujar, tirar, alisar, agarrar, pellizcar o manipular un objeto digital como si estuviera hecho de arcilla.

Pixols: Una tecnología que almacena información de iluminación, color, material, orientación y profundidad de los puntos que componen todos los objetos en la pantalla.

Polycount: El *polycount* es un término para definir cuántos polígonos tiene un modelo.

Mapas de UVs: Los UV maps son imágenes 2D proyectadas sobre una superficie 3D.

Bake: Es el proceso de almacenar información de una malla 3D en una textura 2D.

Blueprints: Es un *asset* que permite a los creadores de contenido agregar fácilmente funcionalidades además de las clases de juego existentes.

Concept arts: Son una representación visual que cuenta una historia o transmite una determinada estética.

Moodboard: Un conjunto de imágenes, materiales o referentes que pretenden evocar un estilo o concepto particular.

Master Material: Es un material que utiliza parámetros customizables con una mayor utilidad que un material estándar.

Nanite: El nuevo sistema de renderizado de geometría introducido en Unreal Engine 5, que renderiza la malla con mayor o menor detalle, en función de la distancia la que dichos detalles se pueden percibir.

Seamless materials: Es una imagen que se puede colocar junto a sí misma (arriba, debajo o al lado) sin que se perciba un límite obvio entre ambas imágenes.

Procedural: Algo generado de forma procedural, se refiere a un método de creación de contenidos a través de algoritmos, en oposición a un método de creación manual.

Jaggies: Es un término para referirse a anomalías en imágenes de pantalla. El resultado muestra bordes de forma de pequeños cuadrados o "escalones" en lugar de una línea suave.

World Position Offset (WPO)**:** Un *input* que permite que los vértices de una malla sean manipulados en el espacio del mundo por el material. Esto es útil para hacer que los objetos se muevan, cambien de forma, roten, y una variedad de otros efectos.

1. Introducción

1.1 Motivación

Mi interés por el 3D viene desde antes de empezar la carrera. Tras los años cursados en la carrera y sobre todo, por interés propio, me he familiarizado con el software para el modelado, texturizado, esculpido y animación 3D.

En este último año, además, he adquirido el conocimiento necesario para el modelado de formas y personajes orgánicos, algo que siempre había intentado evitar, centrándome únicamente en el "*hard surface modeling*". Además, he podido adquirir la soltura suficiente para modelar y texturizar modelos con un estilo "*stylized*", en el que me siento muy cómodo.

Este proyecto nace de la idea de unificar todos estos conceptos por los que siento verdadera pasión. Crear algo que hable y muestre mi trabajo y habilidad en este sector al que me gustaría apuntar en mi futuro profesional.

La creación de personajes, entornos, "*assets*", sus respectivas animaciones, texturas y la inserción de los mismos en un motor gráfico en el que programar sus interacciones me parece un reto y la mejor forma de terminar mi estancia en la universidad.

1.2 Formulación del problema

El problema a resolver en este proyecto es la obtención de un entorno interactivo, pulido y acabado, que represente y defina un estilo y "*workflow*" profesional, con el objetivo de asumir una estética y ambientación de un videojuego estilizado actual.

Necesitaré poner en práctica todos los conocimientos adquiridos y aprender otros muchos, usar un amplio abanico de software para las distintas fases de la producción y preproducción del proceso, conceptualizar de forma coherente todos los elementos del entorno y sus personajes y gestionar el tiempo para realizar de forma óptima todo este volumen de trabajo.

1.3 Objetivos Generales

El principal objetivo del proyecto es la obtención de una versión jugable que contenga el entorno creado, en la que poder visualizarlo en primera persona, desplazarnos por él e interactuar con el mismo y sus personajes.

Para ello me centraré en el ámbito del modelado, esculpido, texturizado, "*rigging*", animación, "*retopología*", iluminación y programación. Aprendiendo y poniendo en práctica los conocimientos de dichos ámbitos.

1.4 Objetivos específicos

- Investigar y hacer acopio de varios referentes y ejemplos de entornos interactivos.
- Gestionar de forma correcta y planificar los horarios y el tiempo para un proyecto de estas dimensiones, teniendo en cuenta que es un trabajo individual.
- Conceptualizar el entorno y todos los elementos que lo componen:
    - Estética y ambientación, la narrativa visual del entorno
    - Vegetación, deformaciones de terreno y formaciones geológicas
    - Personajes y sus características físicas y rasgos de personalidad
    - Los espacios en los que encontramos a los personajes y lo que dicen de ellos
- Modelar los *assets*, el terreno y los personajes.
- Esculpir un modelo "*high poly*" de aquellos elementos que lo necesiten.
- Hacer una retopología de los modelos esculpidos.
- Texturizar dichos assets, el terreno y personajes.
- Crear *riggs* y su correspondiente pintado de peso para los personajes y aquella vegetación o las partes del entorno que vayan a ser animadas.
- Adquirir más conocimientos en motores gráficos y cómo insertar los modelos consiguiendo un buen resultado visual sin que afecte al rendimiento.
- Iluminar el entorno con las herramientas que ofrecen dichos motores gráficos.
- Crear shaders y partículas para aquellas texturas que necesitan una animación como el agua o el fuego.
- Programar las interacciones entre el jugador y el entorno y sus personajes.
- Conseguir una sensación de inmersión en primera persona al desplazarse por el entorno, jugando con las cámaras y animaciones.
- Presentar de forma correcta el proyecto y las posibilidades que ofrece el entorno al espectador.

1.5 Alcance del proyecto

Las limitaciones del proyecto vienen dadas, entre otras cosas, por el volumen de trabajo, el cual, si no está bien definido desde un inicio, puede ir escalando durante el proyecto. A la hora de conceptualizar el entorno, los personajes y el resto de elementos debo de ser realista con el tiempo que tendré para hacerlos y no incluir todo lo que personalmente me gustaría, sin que esto llegue afectar al acabado visual qué deseo alcanzar.

A esto se le suma el nivel de conocimientos que poseo sobre el tema, que aunque no son nulos y es un campo que me motiva mucho, están lejos de llegar a un nivel profesional. Por otra parte, mis competencias en cuanto a motores gráficos como *Unreal Engine* o *Unity*, son algo escasas y seguramente sean difíciles de adquirir durante la realización del proyecto, teniendo en cuenta el tiempo que debo dedicar a las otras partes.

Aun contando con estas limitaciones y muchas otras con las que me encontraré, el alcance del proyecto comprende la creación y puesta en escena de todos los elementos que formarán un entorno digital, y la generación de una versión jugable que permita al usuario visualizar dicho entorno e interactuar con él.

El proyecto va orientado a varios grupos de personas. El primero de ellos, un público más genérico, que no tiene por qué tener conocimientos sobre el 3D y que pueden disfrutar de una experiencia gráfica e interactiva.

El segundo grupo de personas, son aquellos usuarios con conocimientos del 3D y que consumen contenido de este tipo, como fuente de inspiración o que también comparten su trabajo y valoran el del resto de usuarios. Existen comunidades de este tipo en plataformas cómo *Sketchfab*, *Reddit* o *ArtStation*, la idea del proyecto, una vez esté terminado, es la de compartirlo por dichas plataformas para atraer a dichos usuarios.

Por último, el proyecto pretende ser una carta de presentación de mi trabajo, así como formar parte de mi portfolio, por lo que también pretendo enfocarlo a un entorno laboral, siendo el último grupo de personas, posibles clientes futuros interesados en mi trabajo.

Por lo que el principal benefactor del proyecto voy a ser yo, no solamente mi perfil laboral, sino que pretendo aprender nuevas técnicas y ganar soltura en las que ya conozco.

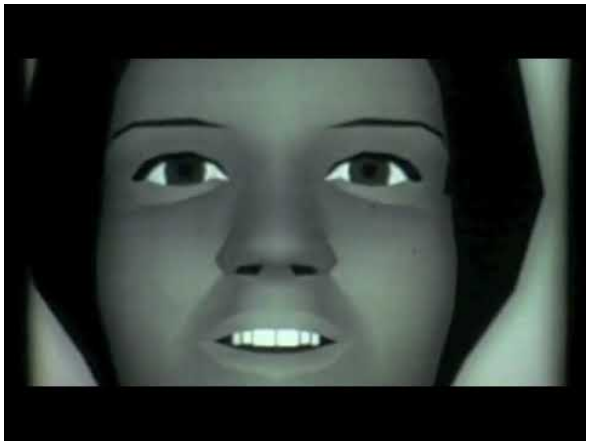
Aunque varios artistas también pueden beneficiarse de mi trabajo, tomando inspiración del mismo. Además, al confeccionar la memoria del proyecto, voy a hacer acopio de todo el proceso de producción del mismo, las técnicas y herramientas usadas, por lo que puede servir como punto de partida o guía para proyectos similares.

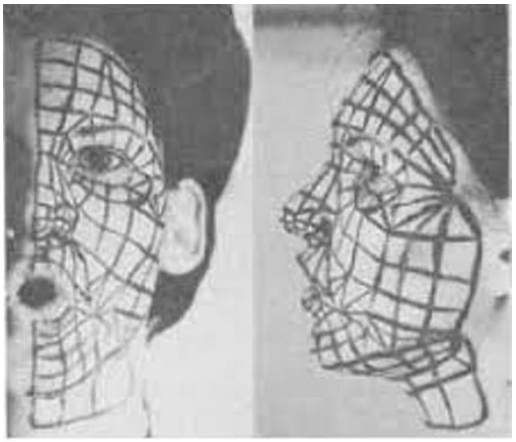
2. Estado del Arte

Introducción

El modelado 3D consiste en formar figuras con volumen, puesto que añadimos una dimensión más. Esto es útil para una infinidad de campos, desde dar vida a los proyectos y planos bidimensionales de los arquitectos, hasta la representación del funcionamiento de maquinaria de los ingenieros.

Los primeros modelos 3D empezaron a aparecer en los años 70, ya que aunque se había intentado trabajar con ellos anteriormente, era necesario un sistema para el renderizado de los mismos. Es gracias a *Sketchpad*, el primer software capaz de manipular objetos en un entorno 3D, que personas como Edwin Catmull, fundador de Pixar, empiezan a interesarse por las aplicaciones de este campo. Gracias a ellos, tenemos registro de obras como "*A Computer Animated Hand*" de 1972, que muestran los primeros modelos en movimiento, más allá de "*primitivas*" como cubos o cilindros.





[1] Fotograma del cortometraje "*Computed Animated Hand*" (1972).
[2] Imagen de la mujer de Fred Parke, usada como modelo para la misma película.

Más tarde, Edwin Catmull toma la presidencia de Pixar, empiezan creando los primeros cortometrajes, como "*Luxo Jr.*" de 1986. Hoy en día son considerados iconos de la animación 3D. Finalmente, en 1995, se estrena"*Toy Story*", el primer largometraje enteramente generado por computadora. La película es un éxito y el interés por la animación 3D aumenta inconmensurablemente.

Hoy en día, el modelado 3D está presente en una enorme variedad de industrias y sus usos se han diversificado, funcionando de formas distintas en función del campo. Existe una infinidad de software, destinado a aplicaciones específicas, programas como *Solidworks*, son usados por ingenieros para el diseño de piezas, engranajes, válvulas… *SketchUp*, es muy popular entre los arquitectos, ya que permite mostrar los entornos a partir de sus planos y tiene una biblioteca con un elenco de vegetación, personas y texturas que permiten visualizar de forma realista sus ideas.

Este proyecto gira, en una enorme parte, en torno al 3D, a diferencia de los casos anteriores donde se suele premiar la practicidad del modelado, sobre la integridad de la malla de los modelos. El campo o sector al que va orientado el proyecto, se tiene especial detalle en la topología de los modelos, optimizando el número de polígonos y su disposición.

Esto es aún más notorio para los modelos que van a ser animados, ya que para articular un movimiento que pretenda doblar o deformar la malla de un modelo, sus polígonos tienen que estar dispuestos de forma que lo permitan sin que el modelo se "rompa".

Se puede ver la presencia de "*loops*" cerca de las articulaciones.

[3] Modelo 3D de Miguel, personaje de la película de Pixar *COCO* (2017), por Fede FRS.

Los modelos "orgánicos", se modelan de forma distinta que el resto, especialmente si van destinados a ser animados, para animarlos es necesario un *rigging* previo, en este proceso se crea un esqueleto para el modelo. Mediante un "*pintado de pesos*", se vinculan los polígonos al esqueleto del modelo, esto es esencial para evitar "romper" los modelos al moverlos, ya que permite indicar la influencia que tiene un determinado hueso sobre una zona del modelo. Los polígonos cercanos a la rodilla se verán afectados tanto por el hueso del fémur como el de la tibia, de la misma forma que los polígonos de la cabeza se verán influenciados por los huesos del cuello, en menor o mayor medida dependiendo de la distancia a la que se encuentren de los mismos.

Hoy en día, en el 3D orientado a la industria del cine,los videojuegos y la publicidad, existen programas "todoterreno" que ofrecen al usuario un gran abanico de herramientas para realizar cada uno de los procesos para el modelado, *rigging* y animación de los modelos y en la mayoría de los casos, hasta para su texturización o esculpido.

Aunque la mayoría de programas "estándar" pueden proporcionar todo lo necesario al usuario, éstos se especializan en función del producto final que quiere obtenerse, y se usan para determinadas cosas. Se pueden distinguir los modelos orgánicos, mencionados anteriormente, y los "modelos *hard surface*", a continuación se hará una explicación del *workflow* o proceso que se suele seguir en la actualidad para obtener un resultado acorde con los estándares de la industria.

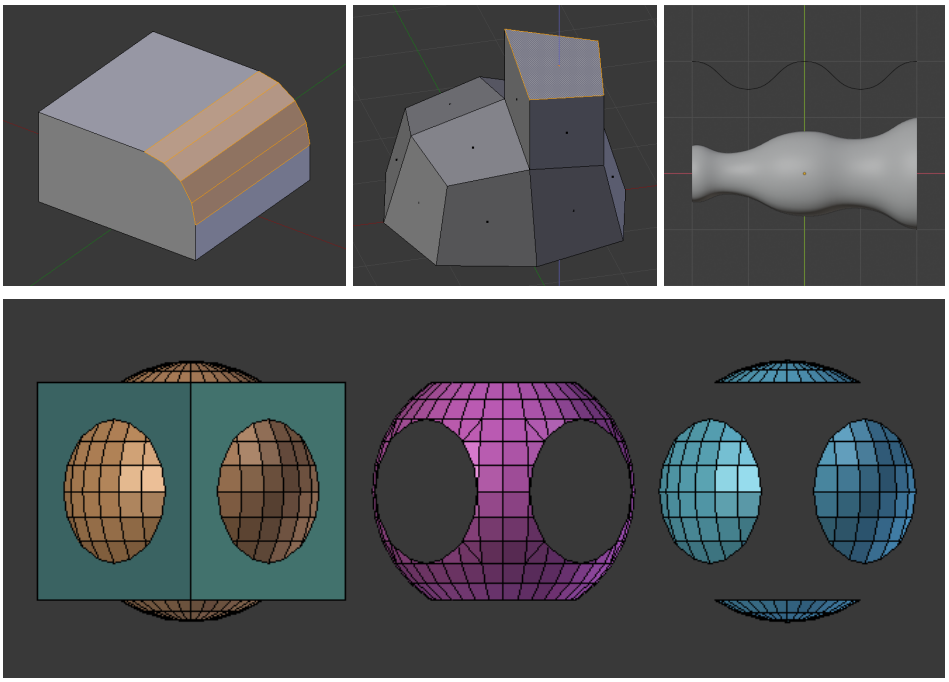
Modelado *Hard Surface*

El proceso del *Hard surface modeling* pretende llegar siempre al mismo punto: la obtención de modelos dotados de una superficie rígida (como coches, armaduras, mobiliario, armas…), pero existen distintas formas para llegar al mismo punto.

El más extendido de todos, parte de objetos simples o *primitivas*, que se editan y modifican para llegar a la forma deseada, todas estas modificaciones son muy diversas, en ellas se extruyen polígonos, se biselan aristas, se sueldan vértices o se añaden *loops*, simulando cortar los polígonos de los objetos. Este método es el más usado porque ofrece un mayor control en la manipulación de las caras, aristas y vértices del modelo, que garantizan una buena topología y una malla generalmente más pulcra.

Aunque existen otros métodos muy útiles en determinadas ocasiones, como el modelado mediante "*Splines*", estas son unas líneas que podemos trazar en un espacio tridimensional, definiendo la posición de varios puntos por los que pasará la línea. Podemos editar dichos puntos, haciendo que formen ángulos rectos o formar "*curvas Bézier*", las *splines* con este tipo de curvas se llaman *B-splines*, muy útiles para modelar cuerdas o cables, por ejemplo. Mediante modificadores que permiten envolver el recorrido de los *splines*, podemos obtener resultados muy distintos.

También es importante mencionar el "*Box modeling*" y el modelado mediante "*Booleanas*", estos métodos se centran en el uso de primitivas, que no suelen editarse demasiado. Las *booleanas*, permiten la obtención de figuras tridimensionales mediante la sustracción, unión o intersección de otras dos figuras distintas. Esto es muy práctico y se obtienen resultados de forma más rápida en comparación al resto de métodos, pero se debe de tener cuidado con la integridad de la malla, ya que este proceso suele crear estragos en la topología del objeto.



[4] Imagen que ilustra el biselado o "*bevel*" de la arista de un cubo, del Manual de Blender.
[5] Imagen que muestra la extrusión de una de las caras de una semiesfera, del Manual de Blender.
[6] Imagen del *Taper Mode*, herramienta para crear topología mediante *splines*, del Manual de Blender.
[7] Imagen que muestra los distintos tipos de modelado mediante booleanas, del Manual de Blender.

La gran mayoría de software para el modelado 3D está dotado para ofrecernos todas las herramientas mencionadas anteriormente, pero por encima del resto, y en el sector al que va orientado este proyecto, destacan:

- Maya: Desarrollado por Autodesk, es el programa más usado y extendido en el sector, es capaz de realizar casi todas las tareas, es personalizable y además destaca por su gran capacidad para riggear, animar y en general, trabajar con personajes, ya que han trabajado mucho la configuración de los fotogramas clave y la animación por curvas, entre otras cosas.

- Blender: Es una alternativa gratuita al resto de software, no está tan popularizado en la industria ya que es relativamente nuevo, pero tiene todas las herramientas para realizar cualquier tipo de proyecto. Además cuenta con una comunidad, que aunque no puede ofrecer el apoyo de un servicio técnico de un programa de pago, hacen "*plugins*", herramientas y tutoriales gratuitos. Se actualiza constantemente, ofreciendo cada vez más herramientas, recientemente han implementado la posibilidad de realizar animaciones en 2D y combinarlas con escenarios en 3D, todo desde Blender.

- Cinema 4D: Cinema 4D, está desarrollado por MAXON y es el estándar para "los *motion graphics*", la generación de infografías, rótulos, textos, animaciones más abstractas o surrealistas, etc en general, está bastante ligado al diseño. Destaca por la facilidad con la que los usuarios pueden familiarizarse con las herramientas y desplazarse por el entorno 3D, ya que cuenta con una interfaz muy simplificada e intuitiva.

- 3Ds MAX: El otro software desarrollado por Autodesk, destaca más en sectores como la arquitectura o la ingeniería y lleva más años funcionando en la industria y permite interactuar directamente con el resto de software de Autodesk. Además posee una configuración de materiales y ajustes de render únicas, su rendimiento está muy bien optimizado y funciona en la mayoría de hardware.

Modelado Orgánico

El modelado orgánico pretende crear objetos 3D que imitan materia orgánica, esto puede ser complejo de conseguir usando las mismas herramientas que en el *hard surface modeling*, ya que en la naturaleza encontramos formas más complejas.

Para ello, existe algo llamado "*3D Sculpting*", un método que puede parecerse al esculpido tradicional con arcilla, pero este ofrece una infinidad de herramientas y posibilidades, pudiendo exportar el resultado a medios digitales.

Todos los programas mencionados anteriormente, a excepción de 3Ds Max, que cuenta con algo parecido, llamado "*Paint Deformation*", poseen herramientas propias para el esculpido digital, siendo la de Blender la mejor valorada por los usuarios. Aun así, hay un líder indiscutible en cuanto a software para el *3D sculpting*, la aplicación desarrollada por Pixologic: ZBrush.

ZBrush ofrece herramientas para modelar, texturizar, pintar y renderizar modelos, pero por encima de todo, destaca por su increíble potencial para el esculpido digital. No solamente tiene una gran variedad de pinceles, deformadores y maneras para pulir, modificar y optimizar la malla, sino que es capaz de trabajar con millones de polígonos en pantalla sin que su rendimiento se vea afectado. Esto es gracias a que usa una tecnología propia llamada "*Pixols*", estos funcionan cómo píxeles 3D, que contienen información de la profundidad, orientación y el material de la malla, lo que permite trabajar con altas cargas poligonales.

Retopology

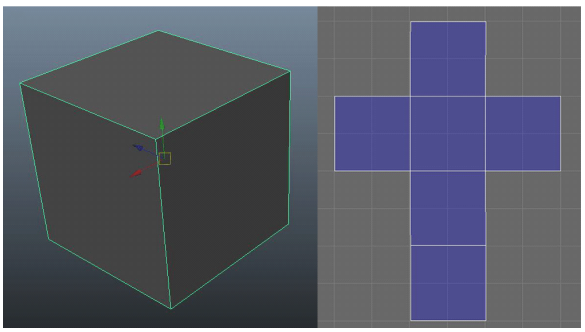
Mientras que ZBrush permite trabajar con un "*polycount*" elevado, para el resto de software, esto puede suponer un problema, sobre todo teniendo en cuenta que un motor gráfico tendrá que cargar y renderizar todos los modelos y texturas del proyecto.

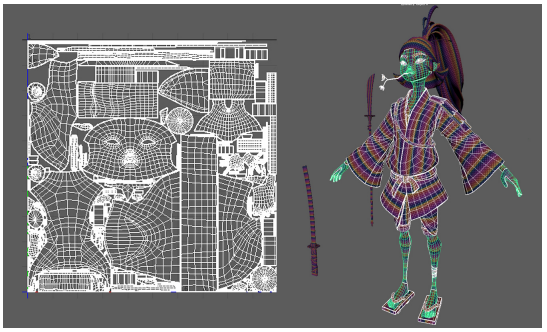
Para ello, se usa la "*Retopología*", esta técnica pretende reducir el número de polígonos del modelo, intentando perder la mínima cantidad de información posible. El resultado será un objeto *low poly*, llamado así ya que contiene una cantidad de polígonos más baja que el objeto obtenido del *3D sculpting*, llamado *high poly*.

Con tal de imitar la forma lo mejor posible, se suele modelar por encima del *high poly*. Programas como Maya, hacen este proceso mucho más cómodo, ya que cuentan con una propiedad de "*snap*" que hace que las caras, aristas o vértices se adhieran a la superficie del *high poly*. Además del "*Quad Draw*", una especie de lápiz de polígonos, que permite dibujar "*quads*" sobre la superficie del modelo.

Mapeado de UVs

Con tal de darle color al modelo, y texturizarlo de forma correcta, es necesario sacar los "*mapas de UVs*" del modelo. Las UVs son una representación del modelo 3D en un espacio 2D, para hacerlo se "despliega" el modelo. Este proceso es el equivalente a desplegar todas las caras de un cubo sobre un plano, aunque es más complejo, el principio es el mismo.





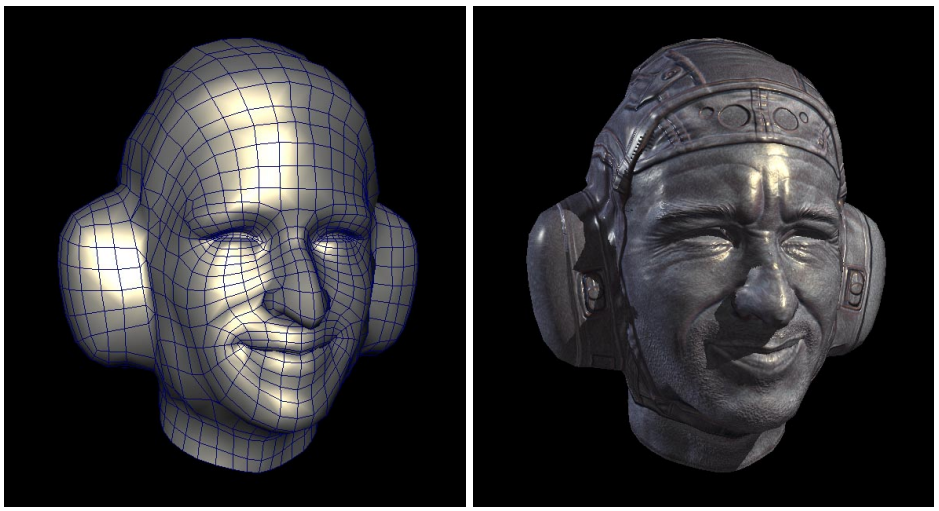
[8] Imagen tomada en Maya, mostrando como hacer un "*unfold*" de las UVs, por The App Guruz.
[9] Modelo y UV map de Vertex School, del curso "*Stylized Characters in 3D*".

Si es conveniente, se pueden sacar más de un mapa de UVs para un mismo objeto, pero al trabajar con varios objetos a la vez, es recomendable no excederse con el número de mapas, incluso usar un mismo mapa para distintos objetos. Ya que posteriormente cada mapa de UVs tendrá su *color map*, *height map*, *normal map*, *AO map*, etc. correspondiente, influyendo considerablemente en el rendimiento.

La inmensa mayoría del software 3D, tiene herramientas para sacar los mapas de UVs, existen también programas específicamente creados para ello. Aunque, Blender y Maya son muy populares, con varias herramientas para separar, proyectar y reorganizar las UVs, haciendo muy cómodo el proceso.

Texturización y *Baking*

Una vez hemos obtenido una versión con menor carga poligonal de nuestro modelo original, y hemos sacado las UVs del modelo, podemos realizar un "*Bake*". Este proceso permite almacenar la información de nuestro modelo *high poly*, en nuestro modelo *low poly*. Mediante *mapas de texturas*, que contienen información específica de nuestro modelo (cómo el *color map*, el *position map*, los *normal maps* o los *height maps*).



[10, 11] Modelo *low poly* antes y después del *bake*, por Sebastien Legrain.

Al *bakear* el *high poly* sobre el *low poly*, se cargan los mapas del primer modelo sobre el segundo, permitiendo conservar gran parte de la apariencia y el detalle del modelo original, pero con una menor cantidad de polígonos. Esto resulta muy útil, puesto que un solo polígono puede, por ejemplo, almacenar hendiduras o protuberancias gracias a los mapas de normales o *height maps*.

En este punto, se suele texturizar el modelo, cada software tiene editores de materiales distintos, con sus ventajas y desventajas. Hay programas dedicados exclusivamente al texturizado de modelos y creación de materiales, dependiendo del resultado deseado es preferible escoger uno u otro, pero si hablamos de *Stylized art*, por encima del resto destacan los programas de Adobe: Substance Painter, para texturizar y pintar a mano los modelos 3D, ya que ofrece un enorme abanico de posibilidades y formas de modificar casi todos los aspectos de los mapas de texturas. Y Substance Designer, para crear cualquier tipo de material, con infinidad de maneras y herramientas para customizarlos.

Una vez se han creado los distintos *assets* y personajes, y se han exportado las texturas creadas para cada uno de ellos. Es necesario importarlos a un motor gráfico, que servirá para renderizar nuestros objetos 3D y será en el que programaremos la interacción con el usuario, o desarrollaremos la experiencia de juego deseada.

Programación y Motores Gráficos

Las primeras computadoras construidas en la década del 1940 empiezan a utilizarse para programar aplicaciones de ajedrez, a medida que mejoraba la potencia de las computadoras los programas se volvían más complejos, dejaron de usar la "fuerza bruta" de la máquina y empezaron a programar inteligencias artificiales hasta que el 11 de mayo de 1997, *Deep Blue*, la supercomputadora de Intel vence a Garri Kaspárov, campeón mundial de ajedrez.

La potencia de los ordenadores nunca ha dejado de mejorar de la misma forma que tampoco se han dejado de desarrollar videojuegos, actualmente resulta imposible ganar a una computadora jugando al ajedrez, ya que conoce todos los posibles movimientos del jugador, y la industria de los videojuegos se encuentra en un punto álgido, en el que es comparable a la del cine u otros medios de entretenimiento convencionales.

Hoy en día, la industria del videojuego se distinguen dos grupos, aquellos videojuegos desarrollados por grandes compañías, conocidos como "*juegos triple A*" o "*AAA*", y los juegos hechos por desarrolladoras independientes o "*juegos indies*".

Éstos no sólo se diferencian por el presupuesto y personal destinado a cada uno de ellos, sino porque los *AAA*, acostumbran a estar hechos con un motor gráfico propio, mientras que los indies tienden a usar motores gráficos "de terceros". Esto les permite centrarse en el diseñar el juego, en lugar de crear un motor propio, aunque más tarde la compañía propietaria del motor gráfico se lleve un porcentaje de los beneficios obtenidos con el videojuego.

Este proyecto no profundiza mucho en la programación, por lo que no se ha contemplado la idea de crear un motor gráfico para desarrollar el juego, en lugar de eso se usará uno existente.

Actualmente, los dos motores gráficos más usados son Unity y Unreal Engine, existen otros motores gráficos de terceros como Frostbite de EA o el Creation Engine de Bethesda, pero pertenecen a sus respectivas compañías y para usarlos es necesario asociarse con ellas.

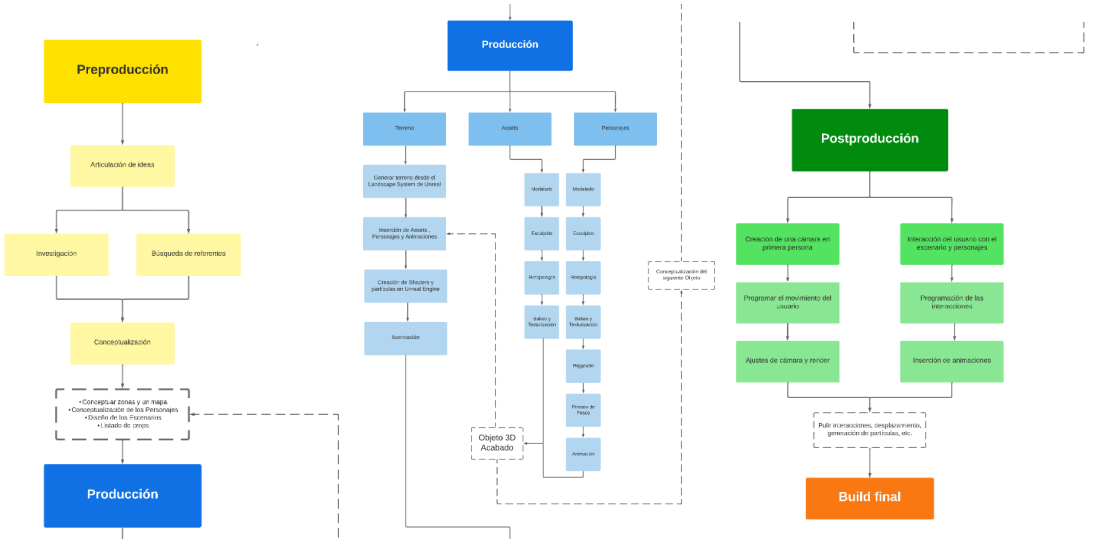
Si los comparamos, Unity es relativamente más fácil de aprender que Unreal Engine, ya que puede programarse en *C#*, por lo que alguien con conocimientos de programación, podrá familiarizarse rápidamente con el programa. Por otra parte, Unreal Engine es más complejo que Unity y aunque puede ser programado en *C++*, destaca por el uso de "*Blueprints*" un sistema de programación propio, mediante un sistema de nodos.

El motor gráfico de Unreal, ofrece resultados gráficos más realistas y un mejor rendimiento que Unity al desplegar todo su potencial gráfico. Sin embargo, Unity da muchas posibilidades para la creación de videojuegos en 2D, siendo raro encontrar un videojuego en dos dimensiones hecho en Unreal Engine.

Recientemente, las aplicaciones del motor gráfico de Unreal Engine 5, han dejado de ser exclusivamente para el desarrollo de videojuegos, ya que muchas compañías que trabajan en el mundo del 3D, se han decantado por usar Unreal para renderizar sus productos, ya que la iluminación del mismo, es muy potente, pudiendo alcanzar resultados hiper-realistas.

3. Gestión del proyecto

Se ha dividido el proyecto en 3 partes, la preproducción, la producción y la postproducción. Esta decisión se ha tomado tras realizar un diagrama del flujo de trabajo, se ha creído conveniente partir de dicho diagrama, ya que la manera de abordar la realización del proyecto, no sigue un orden descendente (metodología cascada), sino que hay actividades que se desarrollan paralelamente y otras que siguen un orden descendente, para luego volver a puntos anteriores:



[12] Capturas de pantalla tomadas del diagrama realizado, consultar en este enlace.

Durante la fase de preproducción, es donde se comienzan a formar las ideas, tras buscar información y referentes, el proceso termina con su correspondiente conceptualización, ya sea de un personaje, un *asset*, o un entorno. Durante esta fase, es difícil contabilizar el tiempo, ya que la mayoría de ideas y su conceptualización, llegan de formas inesperadas, o forman parte de un imaginario propio, que bebe de referentes profundamente arraigados.

Al terminar su conceptualización, se puede seguir conceptualizando elementos del mismo tipo; si por ejemplo estamos realizando "*concept arts*" para la vegetación del entorno, tras conceptualizar un tipo de palmera, pasaremos a bocetar otro tipo de palmera o planta/árbol/arbusto. Pero en vez de realizar el proceso de conceptualización de TODOS los elementos, se ha creído conveniente saltar a la producción, para generar elementos similares una vez conceptualizados.

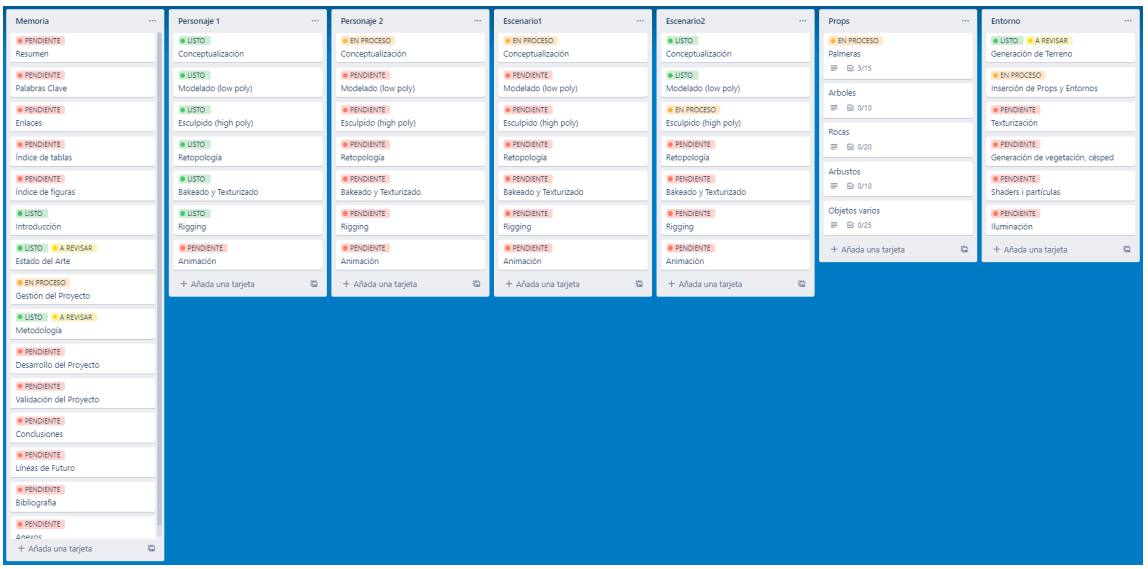
Esta decisión se ha tomado para detectar posibles problemas que pueden surgir durante la producción de forma preventiva, además para evitar estancarse en las primeras fases del proyecto, puesto que son fases muy creativas y repetirlas con continuidad puede llevar a la frustración o pérdida de la motivación.

Durante la fase de producción se generarán de forma paralela los *assets* y personajes además del propio escenario, esto resulta conveniente, ya que si construimos el escenario una vez generados todos los elementos que lo componen, podemos encontrarnos con problemas en las medias de los objetos 3D o con que el escenario se encuentra demasiado vacío o demasiado lleno de elementos.

Además, al generar los elementos, resulta muy fácil perderse en los detalles, por lo que es importante ir dando saltos de una fase a otra, ya que con una versión rudimentaria de un modelo se pueden probar dimensiones y su cohesión estética, para cambiarla a tiempo si fuese necesario.

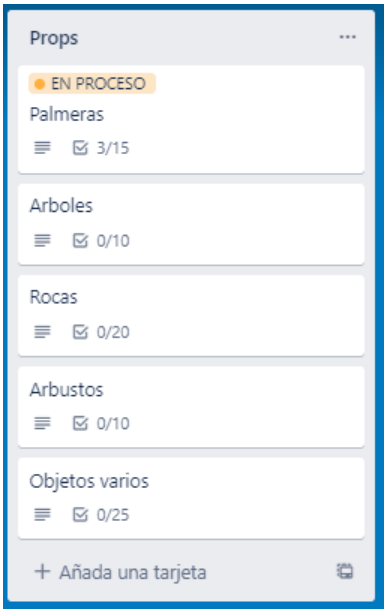
La postproducción es la única fase del proyecto que sigue un orden más convencional, puesto que se centra en la parte de la interacción y parte desde un escenario ya construido. Durante esta parte del proyecto, se generará una cámara en primera persona, que podrá controlar el usuario para desplazarse por el entorno y experimentar la inmersión en el mismo. También se programará la interacción con el escenario y sus personajes, añadiendo las animaciones realizadas, cuadros de diálogo, etc.

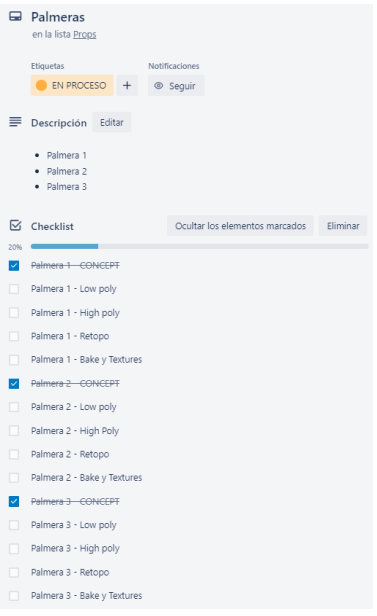
Una vez establecidas las distintas fases del proyecto y como pretenden ejecutarse, se ha planteado una forma para mantener un orden y registro de las tareas a seguir. Mediante la herramienta de los tableros de Trello, se han realizado varias listas con los elementos necesarios para el proyecto.



En estas listas podemos encontrarnos con varios procesos, cada uno de los cuales tiene un estado, para indicar si ya han sido realizados, si lo están siendo o si están pendientes de serlo.

Dentro de estas listas podemos encontrarnos más elementos, dentro de ellos, encontramos más procesos con una checklist para indicar su estado.





[13, 14, 15] Capturas de pantalla tomadas del tablero de Trello creado para la gestión del proyecto.

3.1. DAFO

Siendo consciente de alguna de las fortalezas y debilidades, se ha elaborado un análisis DAFO, para conocer de forma preventiva aquellos puntos es posible encontrarse con puntos débiles durante la realización del proyecto, además de dejar claras las oportunidades que ofrece la realización del mismo.

	Positivos	**Negativos**
Origen Interno	**Fortalezas** • Interés y pasión por el sector en el que se desarrolla el proyecto • Conocimiento de gran cantidad de las técnicas que se usarán para realizar el proyecto • Una idea ya arraigada del entorno a construir durante el proyecto	**Debilidades** • Gran desconocimiento sobre Unreal Engine y su funcionamiento • Una extensión y volumen de trabajo considerablemente grande para una sola persona • Posibilidad de fustigarse cuando algo no sale de la forma deseada
Origen Externo	**Oportunidades** • La obtención de una pieza que sirva de portfolio • Adquisición de experiencia, trabajando con técnicas actuales y que se usan a nivel profesional	**Amenazas** • Problemas de rendimiento en la build • Problemas de falta de tiempo para conseguir el resultado deseado

3.2. Riesgos y plan de contingencias

Se trata de un proyecto de grandes dimensiones, y aunque solamente se centra en el ámbito digital, se usan una gran cantidad de software, por lo que es posible encontrarse frente a una gran cantidad de problemas. Por ello, se ha evaluado los posibles riesgos y se ha elaborado un plan de contingencia frente a dichos problemas, aportando varias soluciones.

Riesgos	**Soluciones**
Problemas con el software	1. Cuento con distintas versiones de los programas más usados en el proyecto. 2. Uso programas con un servicio técnico al que recurrir. 3. Resido cerca de la universidad, donde puedo encontrar casi todo el software usado en el proyecto.

Problemas con el Hardware	1. Almaceno los documentos de memoria y los archivos generados durante el proyecto, en la nube. 2. Poseo otro ordenador, en el que cuento con todos los programas necesarios para seguir trabajando
Verdadera falta de tiempo	1. Durante la conceptualización de los escenarios, se ha previsto dicha situación y se han generado variantes menos complejas de los elementos del proyecto. 2. Se ha establecido un grado de importancia para los elementos que forman el proyecto en caso de tener que prescindir de alguno de ellos por falta de tiempo
Problemas de rendimiento en la Build	1. Se probará de forma preventiva los límites del motor gráfico para calcular razonablemente hasta dónde puede llegar. 2. Pueden generarse distintas versiones de la build en función al equipo con el que se ejecutará, disminuyendo la calidad de las texturas, su número de mapas de texturas o incluso la densidad de polígonos de los modelos.

3.3. Herramientas de validación

A medida que se vayan generando elementos, ya sean *assets*, personajes, *props* del escenario (rocas, árboles, plantas, etc.) o entornos, se comparará su cohesión visual entre ellos mismos, y entre los referentes recogidos, para lograr la ambientación e identidad visual deseada.

El público objetivo del proyecto es un grupo muy extenso de perfiles, por lo que comprobar si cumple con todos los requisitos para cada uno de los grupos de usuarios, puede ser complicado.

Para analizar la aceptación y el agrado o rechazo del público ante mi proyecto, se hará uso de la difusión por redes sociales y el boca a boca, en círculos más cercanos, con tal de apelar al público más "*casual*", que simplemente se encontrará frente a una experiencia interactiva.

Con tal de llegar a los usuarios más entendidos en el sector, se consultará con la docencia del centro especializada en el tema, además de la difusión mediante redes laborales personales y foros y espacios interesados en el mundo del 3D y la creación de entornos digitales.

Junto a la build del juego se adjuntará un documento para que los usuarios puedan proporcionar "*feedback*" acerca del proyecto. El documento variará en función del público al que quiere ir dirigido.

Finalmente, como se mencionó previamente en el Alcance del proyecto, el último grupo de personas al que va dirigido este proyecto, pertenecen al sector laboral del que se pretende formar parte en un futuro. La manera con la que determinar si se trata de un proyecto válido, desde el ámbito laboral es bastante obvia, pero esto tendrá que ser comprobado más adelante.

3.4. Análisis inicial de los costes

Se ha buscado información acerca del salario medio en España, para la realización de la memoria y gestión del proyecto, también se ha realizado una búsqueda del salario medio de los campos en los que se trabaja en la realización del proyecto, concept artist, modelado, texturización, rigging y animación de los modelos, además de la programación en el ámbito de los videojuegos. Por último he contado las horas de la iluminación de la escena y la creación de shaders y partículas como "*3D Artist*", un generalista del 3D.

Se ha tenido en cuenta también el coste del software utilizado y el precio del equipo con el que se trabaja, y lo que costaría amortizarlo, haciendo una estimación del tiempo que dure el proyecto.

Tarea	Precio	€/h	€/dia	€/mes	h/mes	Meses	Costes
Preproducción							
Concept artist		13,29 €/h			9	2	239,22
Total Preproducción:							**239,22 €**
Producción							
Modelado 3D		7,59 €/h			25	4	759,30 €
Retopología		7,59 €/h			8	2	121,44 €
Texturizado		11,31 €/h			20	3	407,16 €
Riggeado		11,18 €/h			10	1	111,80 €
Animación		8,84 €/h			15	3	397 €
Iluminación, shaders y partículas		14 €/h			10	1	140 €
Total Producción:							**1.936,7 €**
Postproducción							
Programación		10,87 €/h			20	2	434,80 €
Total Postproducción:							**434,80 €**
Software							
Cinema 4D				61,39 €		6	368,34 €
ZBrush				32,69 €		6	196,14 €
Substance Painter				48,39 €		6	290,34 €
Maya				279 €		6	1.620 €
Total Software:							**2.474,82 €**

Hardware							
Ordenador Sobremesa	1500 €		0,595 €	17,85 €		8	142,80 €
Portátil	950 €		0,527 €	15,83 €		8	126,64 €
Monitores	340 €		0,165 €	4,96		8	39,68 €
Total Hardware							**309,12 €**
Gestión del Proyecto							
Gestión		22,22 €/h			15	6	2000 €
Total Gestión de Proyecto							**2000 €**
TOTAL PROYECTO: 7.394,66 €							

Análisis de los costes final

Durante la producción del proyecto ha surgido la necesidad de usar más programas de los que se esperaba inicialmente:

TreeIt, usado para crear el follaje de los árboles y la vegetación y que no requiere de una licencia de pago para su uso. También se suma a la plantilla de programas *Substance Designer*, que sí requiere de una suscripción de pago mensual. Este software permite crear materiales y patrones con infinitas variaciones mediante el uso de nodos. Será usado para la creación de materiales y pinceles que darán color al terreno del entorno.

De todas formas, aunque la inclusión de nuevo software podría influir negativamente a los costes, hay un nuevo plan de suscripción que ofrece Adobe, a tener en cuenta. Ahora, podemos contar con *Substance Painter*, *Designer* y *Sampler* por 19’35 € al mes, gracias al plan de “Substance 3D”

Software							
Cinema 4D				61,39 €		6	368,34 €
ZBrush				32,69 €		6	196,14 €
Substance Painter				19,35 €		6	116,1 €
Substance Designer				~		6	~
Maya				279 €		6	1.620 €
Total Software:							**2.300,58 €**

Este cambio supone un ahorro de 171,24 € en total, por lo que el coste del proyecto sería de:

TOTAL PROYECTO: 7.223,42 €

4. Metodología

Como se ha mencionado anteriormente en Gestión del proyecto, se ha dividido el proyecto en 3 partes, preproducción, producción y postproducción:

Preproducción

La primera de las partes comprende el proceso de la articulación de la idea y su conceptualización, esto incluye la realización de bocetos y *concept arts*. En paralelo, se hará una búsqueda de referentes, con la creación de un "*moodboard*" o tablero de referentes visuales y la investigación de métodos y técnicas más apropiados para llevar a cabo dicha idea. Esta fase es de gran importancia, ya que pretende atar cabos y encaminar la fase de producción, de forma que surjan las menores dudas o problemas posibles, que pueden llevar a la pérdida de motivación.

Producción

La fase de producción es la que comprende el mayor volumen de trabajo, en ella se pretenden obtener todos los componentes, assets, personajes y entornos que construirán el proyecto, junto a sus respectivas texturas o animaciones.

La etapa de modelado, es la más extensa, ya que engloba la creación de un modelo inicial, el esculpido de una versión *high poly* más detallada y la retopología del modelo una vez terminado el esculpido. A continuación, se realizará el *bakeado* de los mapas de información del modelo y la creación de texturas. En caso de que fuese necesario animar el modelo, primero se *riggearía*, creando un esqueleto para el mismo y vinculando cada uno de los huesos con un pintado de pesos. Finalmente, se procedería a realizar la animación o animaciones.

En paralelo, se construirá el entorno usando las herramientas de generación de terrenos de Unreal Engine, a continuación se irán añadiendo los modelos y escenarios generados para ir completando el entorno, una vez construido, se procederá a realizar *shaders y* partículas y finalmente la iluminación del mismo.

Postproducción

La postproducción comprende los últimos puntos del proyecto, aunque es la responsable de que todo funcione, en ella se construirá la funcionalidad de una cámara en primera persona, para que el usuario pueda desplazarse y visualizar el entorno, este punto comprende la programación de un movimiento fluido y los propios ajustes de cámara y de render, para que la experiencia de visionado sea lo más óptima posible.

De forma paralela a esto, se programará la interacción entre el usuario y los personajes y el entorno, este proceso va de la mano de la inserción de las animaciones, que variarán en función de la interacción. Finalmente, una vez pulidas las partes interactivas del proyecto, solo quedaría generar una build *jugable* del proyecto.

Metodología: Segunda Rúbrica

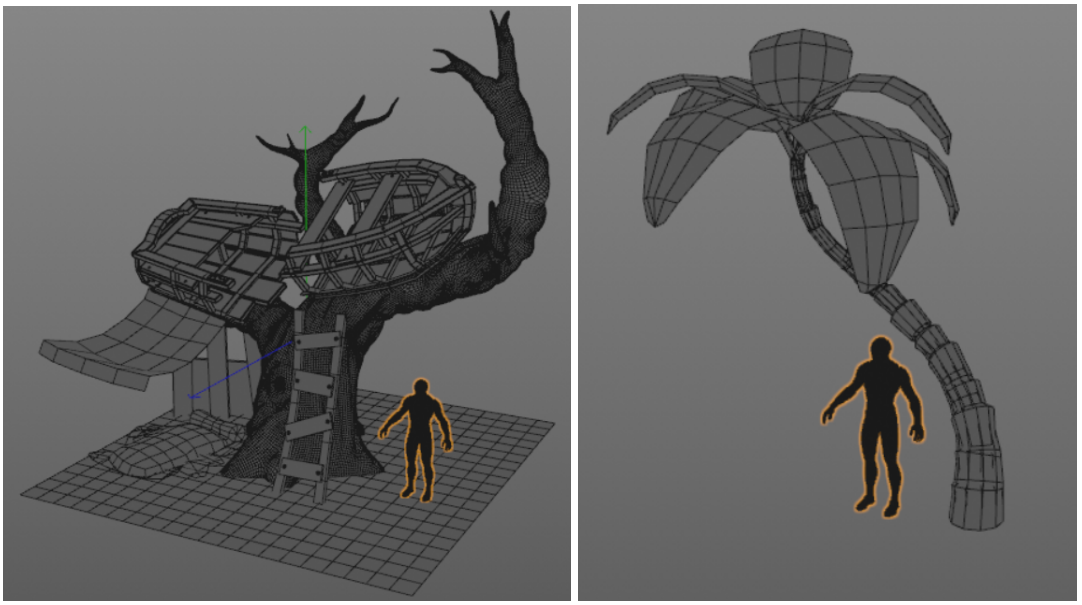
A medida que se ha ido avanzando en el desarrollo del proyecto, se ha creído conveniente realizar algunos cambios en la metodología de trabajo.

Si bien es cierto que se conserva la estructura inicial, presentada en la primera rúbrica, que divide el proyecto en 3 partes, y se mantiene el flujo de trabajo entre la Preproducción y Postproducción. Han habido problemas que han requerido reformular la fase de Producción.

En la metodología de la rúbrica anterior, se plantea el modelado de todos los assets y elementos 3D en paralelo a la construcción de la isla y el mismo entorno. Tras intentar insertar varios assets en el entorno, surgieron problemas de escalas y proporciones debido a trabajar paralelamente entre los softwares de modelado 3D y el de construcción de entornos Unreal Engine.

Si bien se consideró esta posibilidad, y por ello mismo, se decidió trabajar en paralelo, estableciendo una comunicación entre ambos softwares. Resulta mucho más conveniente construir primero el entorno y posteriormente llenarlo y ambientarlo con los modelos creados.

Éstos, serán modelados teniendo en cuenta la proporción con las dimensiones del entorno y además importando en la escena, una copia del modelo del personaje usado en la build del juego, de esta manera se tendrá en cuenta la escala del jugador y como verá los elementos modelados.



[16,17] Capturas de pantalla tomadas durante la fase de modelado.

En este momento el proyecto sigue en fase de producción, ya que es la etapa con un mayor volumen y carga de trabajo, además de ser aquella en la que se había predicho encontrar más errores y más cantidad de conocimientos habría que asimilar y aprender.

5. Desarrollo del proyecto

Para el desarrollo del proyecto se han considerado previamente las opciones más óptimas para la producción del contenido. Además, se han creado herramientas o métodos muy convenientes frente a situaciones que han surgido durante la realización del proyecto.

5.1. Preproducción

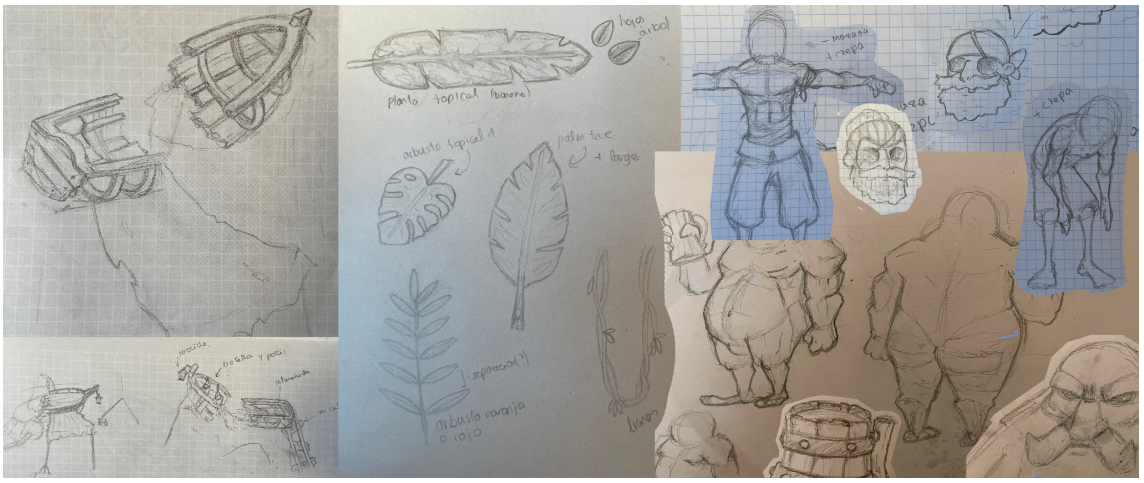
Búsqueda de referentes

La preproducción ha sido hasta ahora, la fase más visual de todas, en ella se ha realizado un trabajo de investigación para conocer las limitaciones técnicas y de tiempo del proyecto. Una vez establecido este límite hubo un gran trabajo de búsqueda de referentes, con el que se estableció el estilo visual del proyecto, la búsqueda de referentes estéticos ha sido constante en todas las fases del desarrollo, pero tiene una gran importancia en la preproducción.

[18] Captura de pantalla del videojuego *Sea Of Thieves* (2018) por el usuario BansheeRob.
[19] Imagen tomada en la creación de entornos del videojuego *Sea Of Thieves* (2018) por Emilis Baltrusaitis

Conceptualización

A continuación, partiendo de unos referentes visuales claros, hubo un gran trabajo de conceptualización, en la que se plasmaron las ideas de los personajes, entornos y su ambientación.

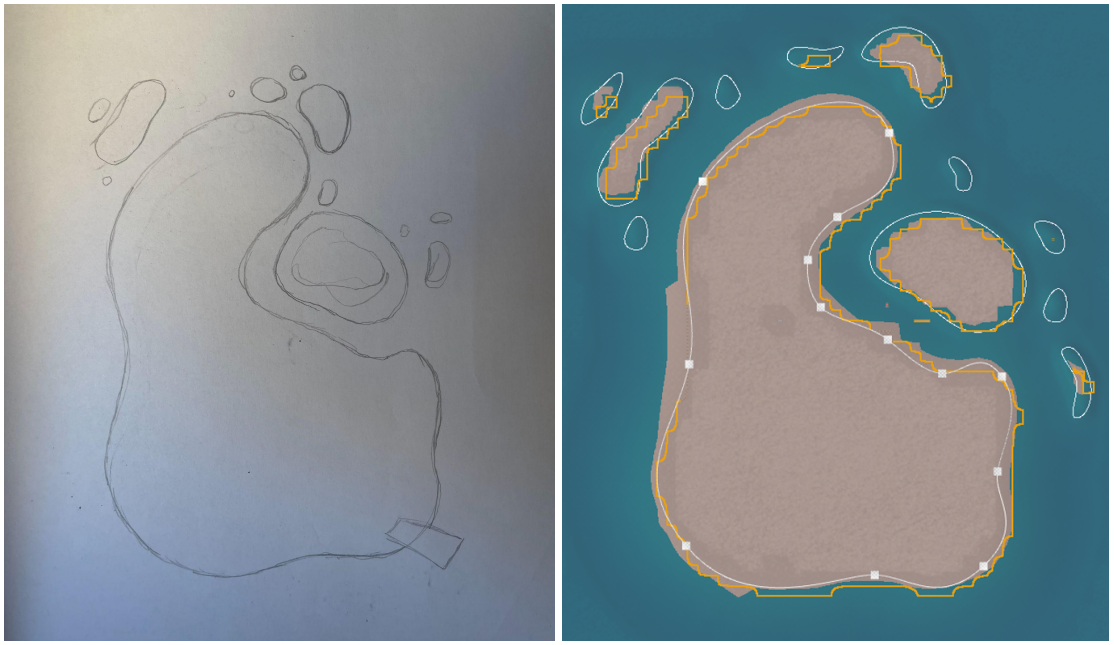


[20] Conjunto de fotografías tomadas de algunos conceptos realizados para el proyecto.

5.2. Producción

Creación del escenario

Se empezó creando la isla a partir de los *concepts art* establecidos, para ello se utilizó la *Landscape Tool* de Unreal Engine, que ofrece una creación dinámica, permitiendo trabajar con pinceles o la herramienta *lasso*, facilitando mucho acercarnos a la idea del *concept*.

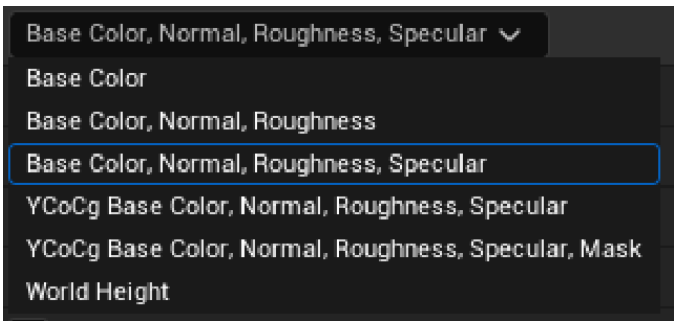


[21] Fotografía tomada del concepto original de la forma de la isla.
[22] Captura de pantalla tomada desde la build de Unreal Engine, durante la creación del escenario.

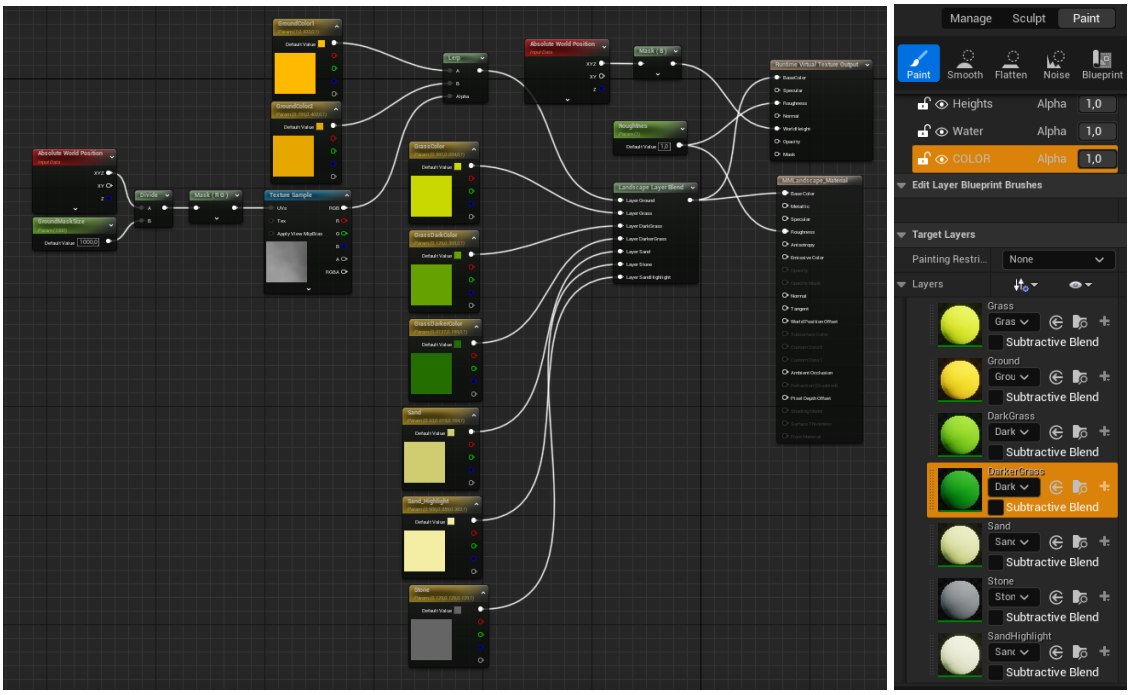
La herramienta de creación de terrenos de Unreal incluye, mediante el uso de plugins, la opción para trabajar con agua, creando océanos, lagos o ríos y permitiendo modificar la apariencia del agua y su comportamiento con el resto de superficies.

Además, permite pintar el propio terreno usando pinceles personalizables y con materiales creados por el usuario, sin la necesidad de realizar un desplegado de UVs y con la ventaja de que el motor gráfico, es capaz de detectar los terrenos creados, como un área jugable, sobre la que poder disponer automáticamente vegetación o assets, sin necesidad de establecer manualmente las zonas de colisión.

Es importante hablar sobre las Real *Runtime Virtual Textures* o RTVs, que al crear un *Runtime Virtual Texture Asset*, permiten almacenar diferentes tipos de información, a la que se puede acceder después para customizar materiales e insertar modelos en el entorno.



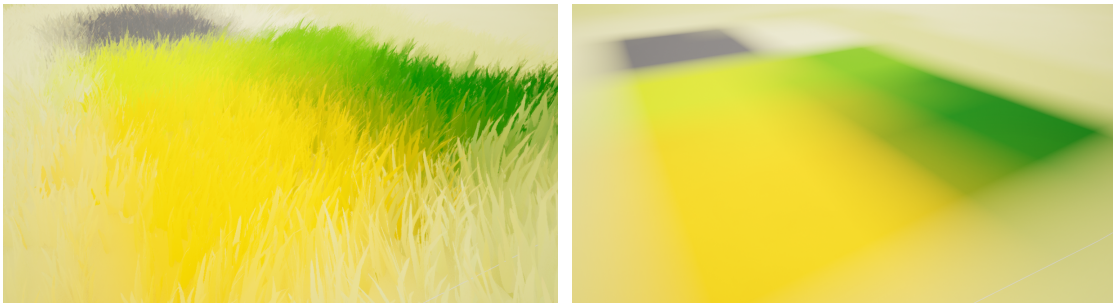
[23] Captura de pantalla tomada desde las opciones de las RTV en Unreal Engine.



[24] Captura de pantalla del *Master Material* creado para el *Landscape* del proyecto.
[25] Captura de pantalla tomada desde el *Landscape Mode* en Unreal Engine.

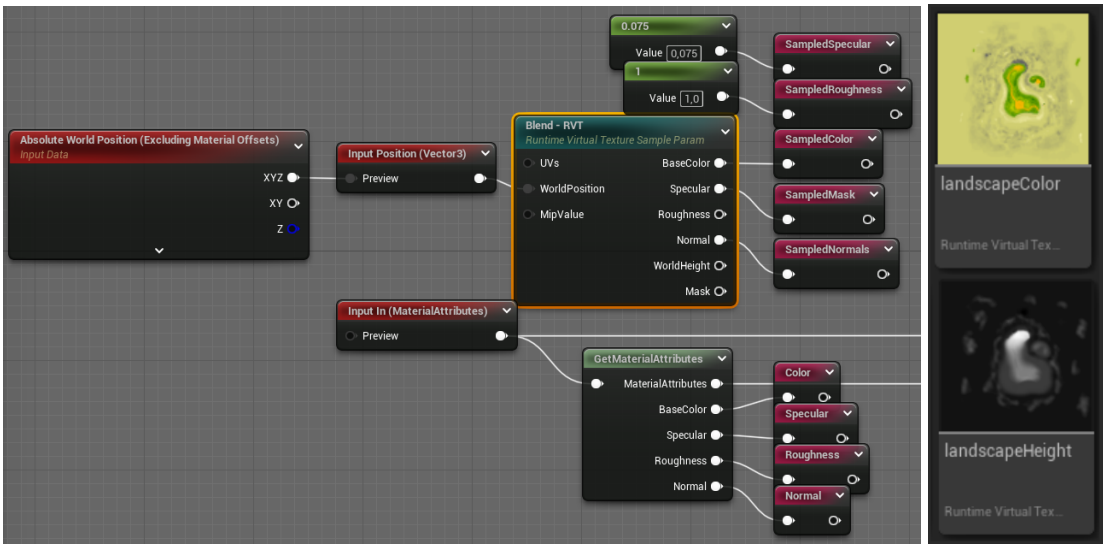
Como se ha mencionado anteriormente, el *Landscape Mode* permite pintar y esculpir sobre el terreno, esto funciona mediante capas, que pueden combinar ambos tipos de información si es preciso. Sobre las capas puede se puede pintar con distintos materiales, que se combinan en un "*Master Material*".

Si se vinculan las RTVs y el *Master Material* al *Landscape*, a partir de este momento, tras pintar o esculpir sobre el terreno, sus RTVs correspondientes cambiarán al editarse el entorno. Esto es muy útil, ya que abre las posibilidades a varias cosas; se puede llamar a las RTVs, desde otros materiales para, por ejemplo, darle color al césped o vegetación en función al terreno, crear coloración para los modelos, o realizar gradientes entre el terreno y los objetos.



[26 y 27] Capturas de pantalla del proyecto que muestran el *Landscape*, con y sin el *Foliage* visible.

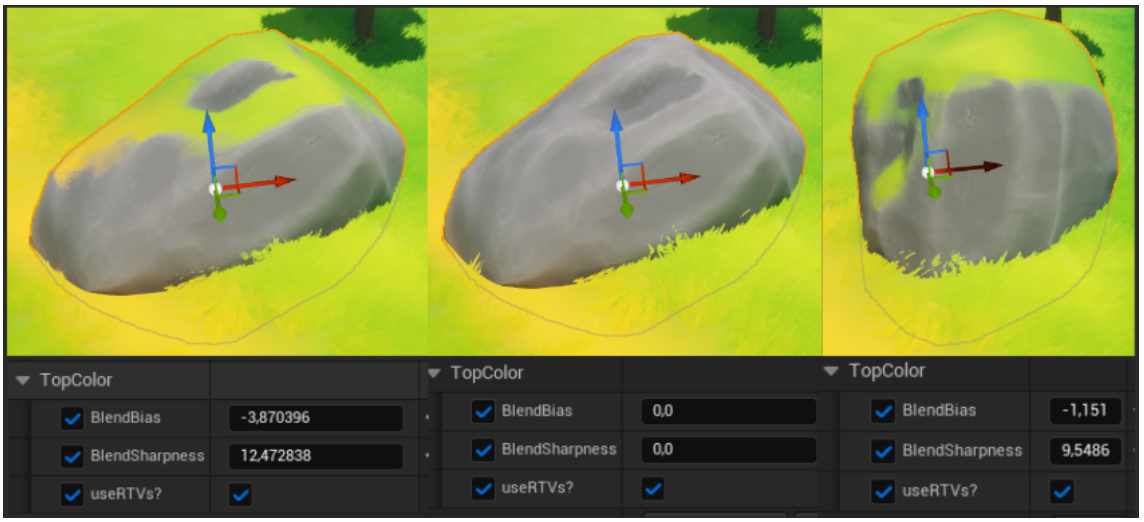
En el ejemplo de la *figura 28* (la imagen que se encuentra al inicio de la siguiente página), se muestra cómo puede referenciarse y extraer la información de ambas RTVs, en este caso, para generar el *Blending* entre el césped y el *Landscape*.



[28] Captura de pantalla de la *Material Function Foliage Blend*, utilizada en el proyecto.
[29] Captura de pantalla de los mapas de texturas obtenidos al usar las RTVs.

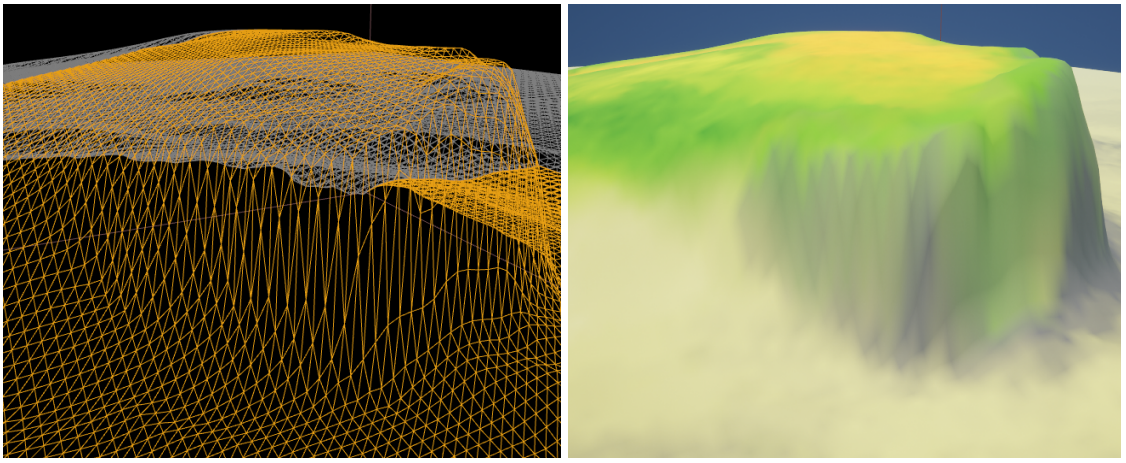
También puede usarse los heightmaps del *Landscape*, obtenidos desde las RTVs para crear materiales complejos, con los que mediante la *Absolute World Position* del proyecto, puede indicarse mediante un vector de interpolación lineal o *Lerp*, que el material del objeto se funda con el RTV del color del terreno. También se ha programado la medida en la que ambos materiales se funden y el contraste o *sharpness* con el que se produce la mezcla.

Esto es posible gracias a la opción de convertir parámetros del material en expresiones, que podrán ser modificadas en tiempo real si se instancia dicho material.



[30] Conjunto de capturas de un material inteligente del proyecto tomadas en Unreal Engine.

Aunque también hay inconvenientes, puesto que la resolución de polígonos de los *Landscapes* tiene un límite de 512x512 quads, por lo que si no se es cuidadoso, puede notarse la falta de polígonos con facilidad. El autor ha investigado sobre el tema tendidamente y aunque sabe que ha de ser posible, no ha encontrado una solución, ni en la documentación de Unreal Engine 5 (puesto que sí se ha encontrado información respecto al tema en Unreal Engine 4), ni en foros ni tutoriales.

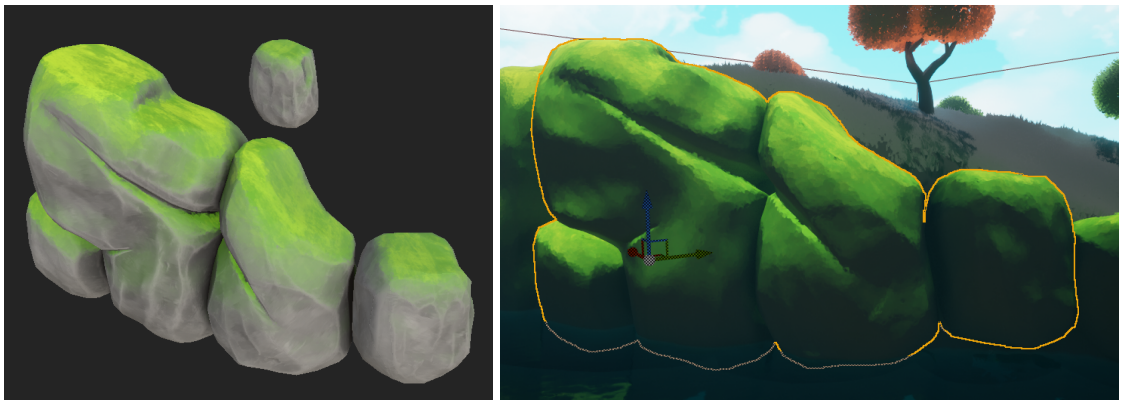


[31 y 32] Capturas de pantalla de la malla del *Landscape* del proyecto tomadas en Unreal Engine.

Aunque recientemente, se ha añadido la compatibilidad de "*Nanite*" con el *Landscape Mode*, por lo que se explorará la posibilidad de su inserción a líneas de futuro.

Materiales para la texturización del entorno

Se ha mencionado anteriormente, la posibilidad de usar materiales inteligentes con la integración de las RTVs, y si bien se ha hecho uso de ellos, el autor ha creído conveniente usar las herramientas proporcionadas por Substance Painter y Designer en la mayoría del texturizado para la creación de materiales que se acerquen más a la estética deseada.



[33] Captura de pantalla de unas rocas usadas en el proyecto desde Substance Painter.
[34] Captura de pantalla que muestra el canal emisivo de las rocas, tomada desde Unreal Engine.

La mayoría de los materiales creados de forma "convencional", con el software de Substance, ha sido retocado posteriormente en Unreal Engine. Se han seguido usando, aunque en menor medida, las *material expressions*, desde las que pueden crearse parámetros con los que modificar, por ejemplo, la intensidad de la emisión de un material, presente en casi todos los *assets*, con tal de conseguir una paleta de colores vivos, y saturados.

Posteriormente se hablará en mayor profundidad acerca del texturizado de los objetos 3D en Substance Painter, y de la creación de "*seamless materials*" en Substance Designer.

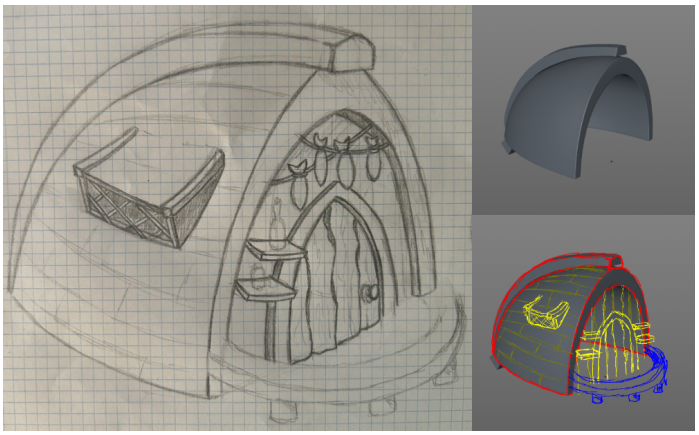
Creación de Assets

Este proceso es uno de los más costosos a nivel de tiempo y engloba una gran cantidad de pasos que serán repetidos durante la fase de producción del proyecto, por lo que se tomará cómo ejemplo la creación de uno de los objetos 3D para explicar el *workflow* que se sigue para el resto de objetos.

Se ha escogido una localización importante, en vez un objeto sencillo como podría haber sido un barril, una botella o una caja, ya que así podrá verse más a fondo cada uno de los procesos. Cómo todo, empieza con una idea, ese boceto mental se plasma con formas muy básicas en el entorno, para tener presente el volumen de dicho objeto al construir el entorno.

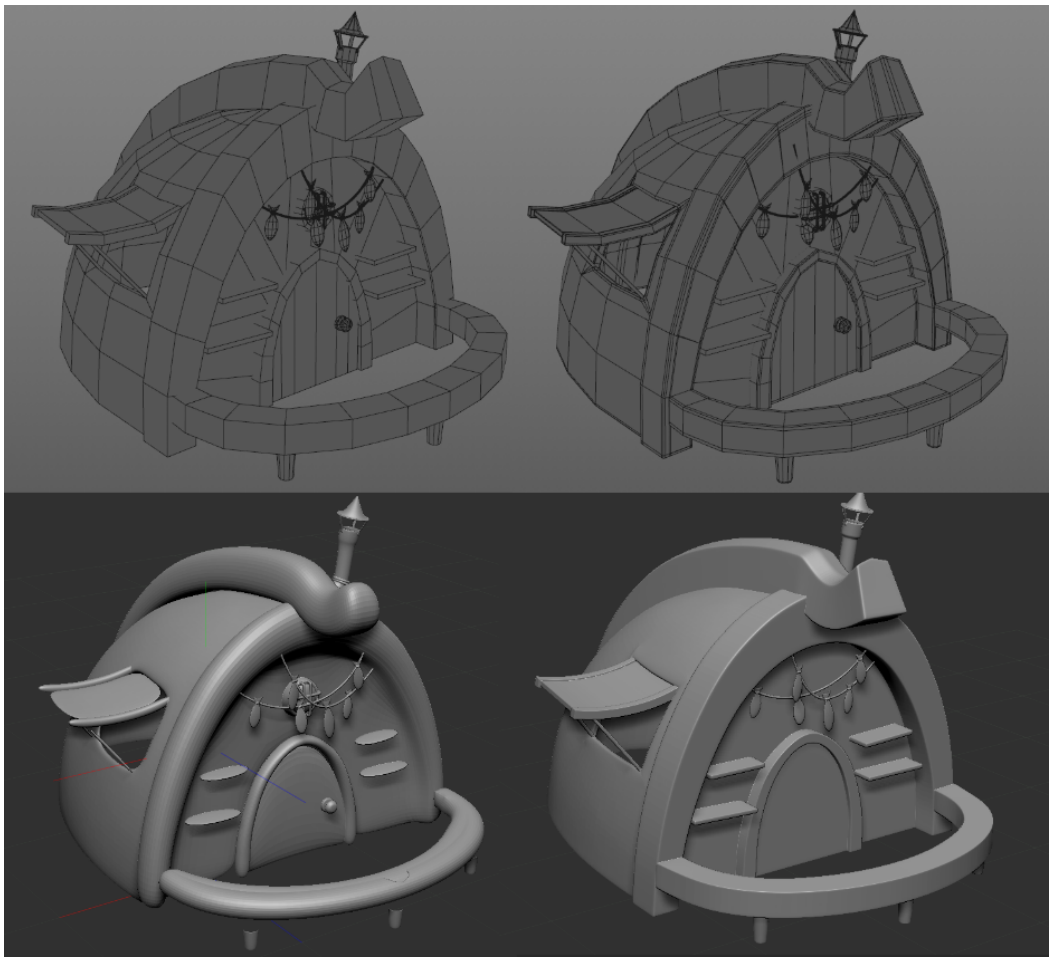
[35] Conjunto de capturas de los modelos finales texturizados en la parte superior, tomadas en Substance Painter y de los modelos temporales, tomadas en Unreal Engine.

Una vez se ha construido el entorno, se retoma esa idea y se pasa a la conceptualización de la misma. Se busca sobre todo entender las formas básicas que constituyen el objeto, posteriormente si es necesario se añaden detalles, pero se pone especial atención en los volúmenes. Una vez se ha plasmado la idea, en papel o de forma digital, y pasa a ser algo tangible, que no puede marchar de la cabeza, se procede al modelado.



El modelado inicial es más bien un *block out*, para intentar replicar esos volúmenes creados en la conceptualización. Se perfilan y detallan las formas y se exporta el resultado ya que será de ayuda en la creación del modelo *low poly*. A continuación se le añaden *loops* al modelo cerca de las aristas, esto evita que el modelo pierda su forma al subdividir su geometría.

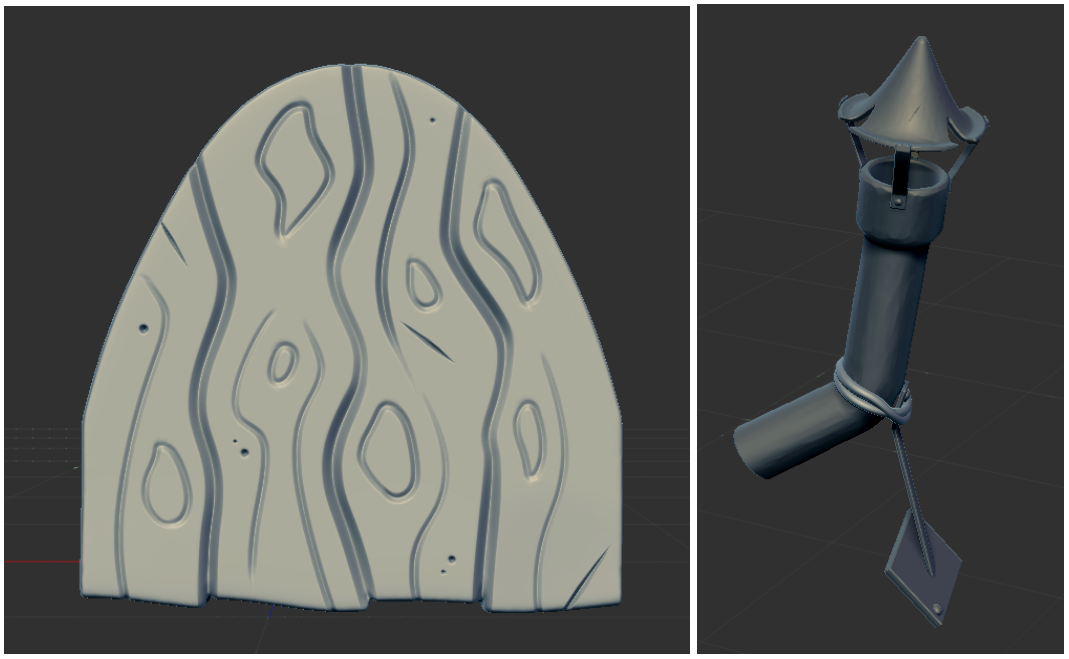
[36 y 37] Fotografía tomada de un concept art dibujado a mano y conjunto de capturas recreando las formas del concepto, tomadas en Cinema4D.



[38] Conjunto de capturas de pantalla del modelo 3D, tomadas en Cinema4D y ZBrush.

Como puede verse en la *fiugra 38*, añadir *loops* cerca de las aristas facilita el proceso del esculpido, ya que para ello, son necesarios miles de polígonos con tal de poder deformar la malla sin que se rompa, por lo que es necesario subdividir la carga poligonal de los objetos. Este proceso tiende a esferizar la malla, debido a que la subdivisión "*Catmull-Clark*", es un algoritmo que pretende suavizar el modelo, creando nuevos vértices, la posición de los cuales es determinada por otros vértices cercanos.

Una vez importado y subdividido el modelo de forma satisfactoria a ZBrush, se procede a detallar el modelo, esto se lleva a cabo mediante el uso de pinceles y transformaciones. Para este objeto y en general para el resto del proyecto, se han usado repetidamente los pinceles *Orb*, brindados por el artista Michael Vicente de forma gratuita, con los que resulta mucho más fácil recrear formas estilizadas, sobre todo en madera, metal y piedra.



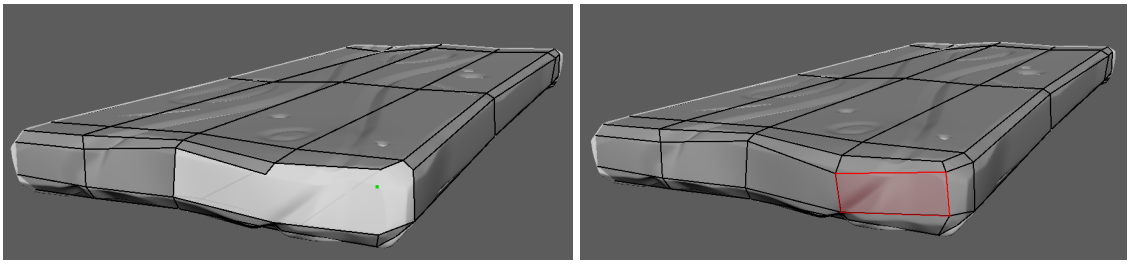
[39 y 40] Capturas de pantalla tomadas durante la fase de esculpido del modelo en ZBrush.

Durante este proceso, se intentan deformar los bordes de la malla, creando imperfecciones, se hacen hendiduras, para simular las vetas de la madera y se aplica presión mediante alfas con los que se pretende recrear cortes o pequeños agujeros.

Una vez completado el esculpido, se obtendrá como resultado un modelo detallado con miles o millones de polígonos, este modelo sería muy costoso de renderizar en el motor gráfico, y en general en cualquier software que no sea ZBrush, que en vez de usar polígonos utiliza *Pixols*, que representan el modelo no en 3D, sino en 2.5D, haciendo posible llegar hasta más de los 40 millones de polígonos.

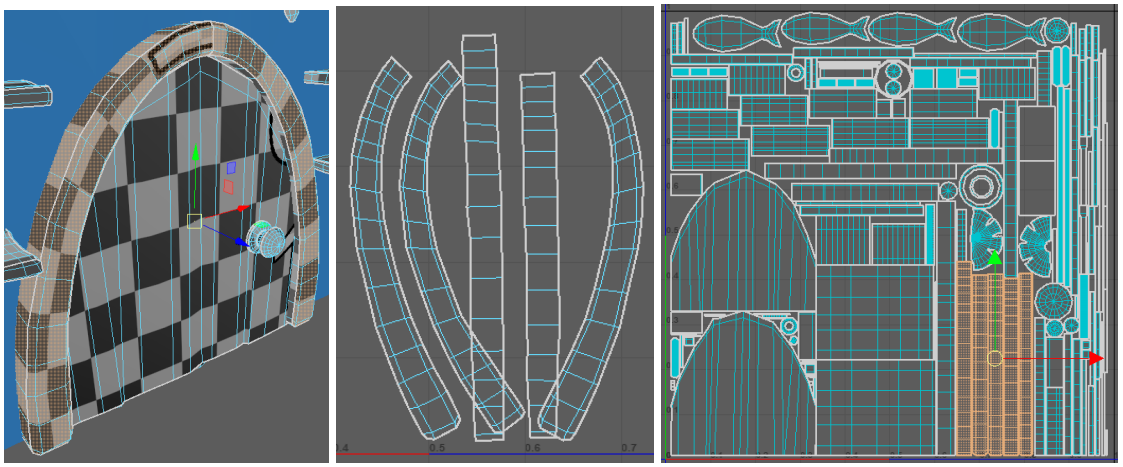
Por esto mismo es necesario hacer una *retopología* del modelo, para esta tarea se ha escogido el software de Maya, que como se ha explicado previamente en el estado del arte, cuenta con herramientas muy útiles para el proceso. Se ha de llevar cuidado con la distribución de los polígonos y el tamaño de sus caras, puesto que posteriormente serán usados para los mapas de UVs del modelo.

Para la retopología, no se partirá desde 0, se usará el modelo *low poly* que se generó en los primeros pasos del modelado, esto resulta mucho más conveniente que redibujar todo el modelo con la herramienta *Quad Draw*, en vez de eso se reposicionarán las caras, aristas y vértices del modelo y se añadirán loops en las partes que sea necesario para ajustarse a la forma global del objeto, si es preciso también se eliminará parte de la maya para reconstruirla manualmente. Esto es especialmente importante para el futuro bake, que requiere que ambas mallas se asemejen, todo lo posible en volumen y forma.



[41 y 42] Capturas de pantalla usando la *Quad Draw Tool*, tomadas durante la retopología en Maya.

Una vez realizada la retopología de todos los componentes que forman el modelo, se procederá al desplegado de UVS, para ahorrar recursos, es conveniente usar el menor número de mapas de texturas posibles, por lo que varios componentes compartirán un mismo mapa de texturas. En este caso, como el objeto 3D con el que se está trabajando es considerablemente grande, se compondrá de dos mapas de texturas, en caso de realizar un despliegue de UVs de un objeto más pequeño como un barril o una caja, se trataría de juntar el mayor número posible de objetos en un mismo mapa, aunque estos pertenezcan a entornos distintos.



[43, 44 y 45] Capturas de pantalla del desplegado de UVs y los mapas de texturas, tomadas en Maya.

Es importante optimizar el espacio de los mapas de texturas, para ello se intentará hacer rectas todas aquellas formas curvas, esto no solamente reduce el espacio que ocupan dentro del mapa, sino que es muy conveniente para el posterior texturizado, ya que las formas que se proyecten sobre las UVs no se verán deformadas.

Hay que añadir, que con tal de ahorrar espacio se pueden *stackear* las UVs, como se hizo previamente para la corteza de los árboles o sus hojas, esto se ha realizado únicamente en elementos como los tornillos, que tienen la misma forma y son elementos pequeños, o bien en los peces. Con tal de ahorrar horas de trabajo al pintar los modelos a mano, se han *stackeado* sus UVS, de tal manera que solo se pintarán 2 peces en vez de 6.

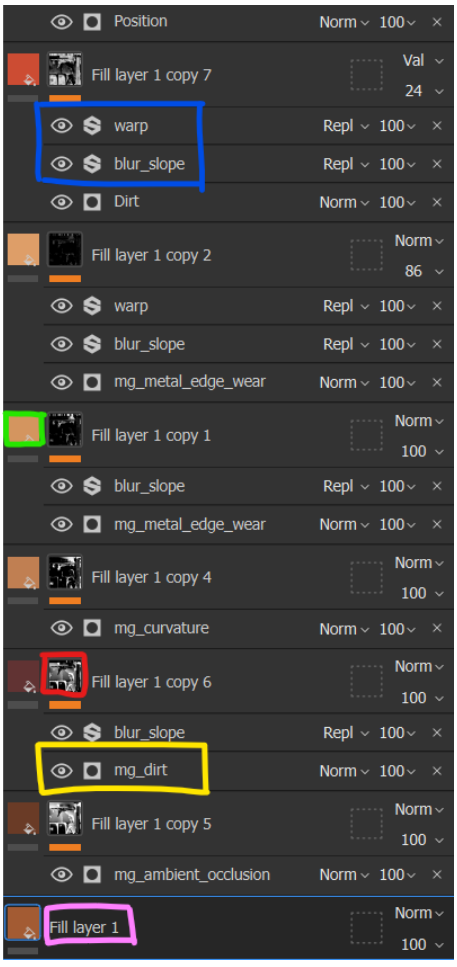
Una vez desplegadas todas las UVs, se separarán los mapas de texturas en 2 materiales y se exportarán los objetos junto con sus respectivos materiales a Substance Painter, desde el que se realizará un *bake*, del modelo que acaba de ser exportado junto con la versión obtenida del esculpido en ZBrush, es en este proceso donde se obtienen los mapas de *Position*, *World space normal*, Ambient Occlusion, Curvature y Thickness del mismo *high poly*.

El proceso de texturizado en Substance Painter que el autor sigue, suele ser similar para casi todos los objetos. Se ayudará de la *figura 46* para la explicación del proceso.

Substance Painter funciona mediante capas, estas capas se superponen una encima de otras, podemos encontrar capas de relleno o *Fill Layers*, que cubren todo el mapa de texturas, las *Fill Layers*, pueden contener información sobre el color, las normales, la oclusión ambiental, etc. además de poder añadirle canales como el emisivo, o el de opacidad si fuese necesario.

También podemos encontrar *Paint Layers*, en las que el usuario podrá dibujar libremente sobre el modelo, afectando a los canales deseados y usando pinceles customizados. Hay también *Effects*, que comprenden una gran variedad de capas de las que se hablarán a continuación. Se pueden añadir también, *Presset Layers*, que vienen siendo materiales ya creados, aunque estos, no se usarán en el proyecto, más allá de los *Smart Materials* creados por el propio autor.

En todos los objetos, se suele empezar desde una **capa base** marcada en rosa, esta es una *Fill Layer*, que contendrá el **color de la capa**, marcado en verde y será, además, el color principal del modelo, además, en este caso, de la rugosidad del mismo.



Se le siguen añadiendo más *Fill Layers*, con colores distintos, aunque esta vez no cubrirán todo el mapa de texturas, sino que se distribuirán mediante el uso de **máscaras**, marcadas en rojo, estas funcionan de la misma forma que en Photoshop.

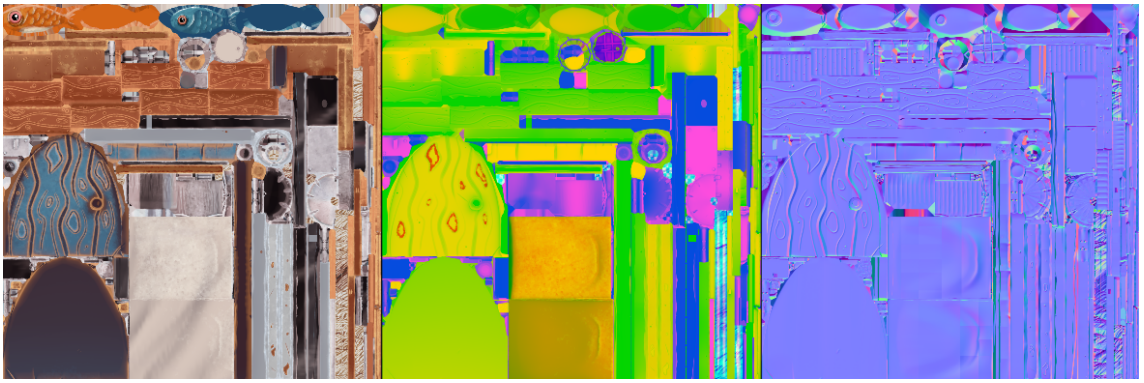
Para controlar como actuan las máscaras, se le añaden ***Generators***, marcados en amarillo. Éstos comprenden una gran variedad de efectos, pero se basan de los mapas obtenidos del *bake* para enmascarar las capas, como por ejemplo, mediante el *Curvature Map*, se pueden enmascarar únicamente las aristas más marcadas del modelo y mediante el "*Dirt generator*", añadir ruido a las mismas, u otros valores que podremos controlar gracias a *sliders* del mismo programa.

Por último, Substance Painter también cuenta con una biblioteca de **filtros**, marcados en azul, con efectos varios, en el ejemplo, en la *figura 46*, se pueden ver los filtros "*blur scope*" y "*wrap*", estos filtros, aplicados con la intensidad adecuada, a diferencia de un simple *blur*, consiguen un desenfoque de la máscara con un efecto *handpainted*, que se acerca al estilo que se pretende lograr en la estética del proyecto.

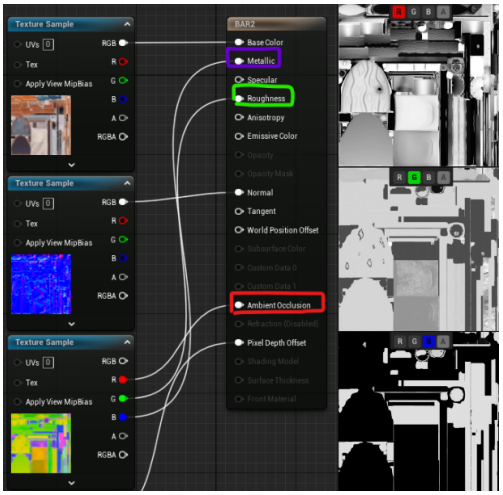
[46] Captura de pantalla de las capas de un material usado en el objeto 3D de la explicación, tomada en Substance Painter.

En el ejemplo de la página anterior, se pretende conseguir un material que recree el aspecto de la madera, para otro tipo de materiales, el proceso sigue siendo el mismo, pero cambiando los valores de los *generators*, los colores y el orden de las capas. En algunos casos, también se llega a usar *Baked Lighting*, una capa que proyecta luz sobre el material y que se almacenará en los mapas de texturas, por lo que es independiente a la iluminación de la escena en la que se integre el modelo.

Cómo se usan los mismos mapas de texturas para distintos objetos, es común encontrarnos con varios materiales que intentan recrear madera, metal, roca o tela en un mismo mapa, para que no se superpongan unos encima de otros, se suelen almacenar en carpetas, y de la misma manera que en programas cómo Photoshop, estas carpetas enmascaran, en este caso polígonos, que delimitan qué parte del mapa de texturas corresponde a un material u otro.



[47] Mapas de texturas generados en Substance Painter para el material del que se habla en la explicación.



Una vez texturizados todos los componentes del objeto 3D, debemos exportarlos. El paquete de software de Substance, ofrece una gran variedad de *presets* para exportar las texturas a distintos programas, incluso, se puede encontrar un plugin en Unreal Engine llamado "*Substance in UE5*", que permite vincular directamente los materiales desde los que se está trabajando en Substance Designer y actualizarlos a tiempo real desde el programa de texturizado sin haver de realizar el proceso de exportación e importación a Unreal.

[48] Captura del material en el editor de Unreal Engine.

En este caso, se exportarán las texturas mediante el mismo *preset* que ofrece Substance Painter para el motor gráfico "*Unreal Engine 4 (packed)*", por defecto, con los ajustes preestablecidos que ofrece está opción, se obtendrán 3 mapas de texturas diferentes, uno para el *Base Color*, otro para el *Normal* y otro conocido como *ORM*, que son las siglas de *Oclussion Ambiental, Roughtness* y *Metalic*, de esta forma, se consigue ahorrar espacio y recursos, ya que el mismo mapa ORM (la imagen de en medio que se muestra en la *figura 47*), almacena en cada uno de los canales RGB, la información para los 3 mapas de texturas distintos, correspondiendo el *Red* para la Oclusión Ambiental, el *Green* para la rugosidad y el *Blue* para el canal Metálico del material.

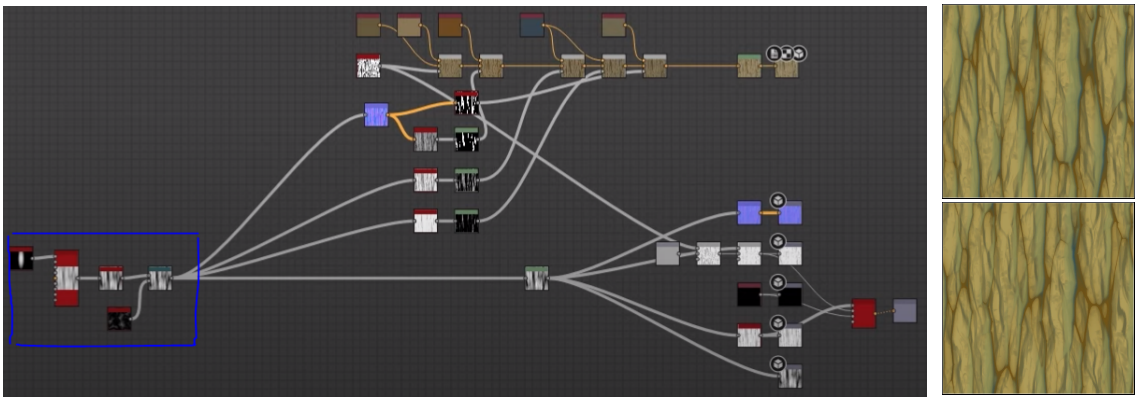
Árboles y vegetación

Para la creación de la flora de la isla, han habido distintos métodos. Elementos duros, cómo los troncos de los árboles y palmeras, se han modelado de forma convencional, usando *splines* para trazar la forma de los mismos y se generando variantes, modificando la forma del tronco.

Con el fin de detallar los modelos, se han seguido distintos enfoques:

La corteza de los árboles se ha realizado mediante texturas complejas generadas "*proceduralmente*" en Substance Designer, ya que como se creía necesario obtener unas cuantas variantes, el proceso de esculpir un modelo *high poly* en Zbrush iba a ser más detallado pero mucho más costoso en cuanto a tiempo se refiere.

Substance Designer trabaja mediante nodos, normalmente se parte desde una forma básica, cómo un círculo, un rectángulo, una cápsula o algún tipo del ruido al que se le aplican deformaciones o filtros, de esta forma, se obtienen texturas muy detalladas y que pueden cambiarse únicamente modificando la semilla de los primeros nodos.



[49] Captura de pantalla del conjunto de nodos que forman un material, tomada en Substance Designer.
[50] Texturas obtenidas directamente al exportar el "*Substance Graph*" de la figura anterior.

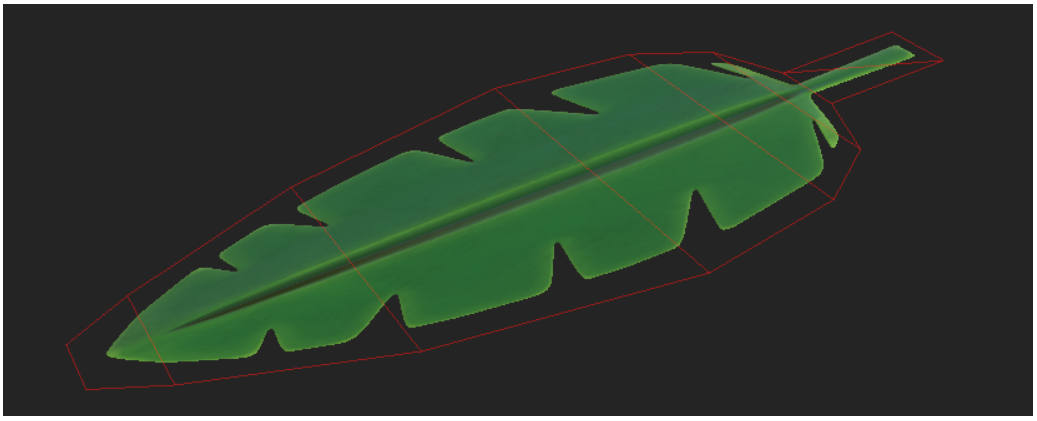
Lo más importante de todo es que las texturas son *tileables*, esto resulta muy práctico ya que, permite que las texturas se repitan horizontal y verticalmente sin que se vean cortadas.

Para hacerlo, cuando se realice el mapeado de UVs, se pondrá especial atención en que todas las caras que construyen el modelo del tronco se desplieguen de forma que tengan el mismo tamaño y se encuentren en la misma posición, *stackeadas* unas encima de otras. El proceso resulta bastante sencillo, ya que el modelo viene a ser un tubo acabado en punta.

Con esto se consigue la posibilidad de generar texturas distintas para las diversas variantes de modelos con relativa rapidez.

En contraposición, para detallar el tronco de las palmeras, se pensó en un estilo *cartoon* el cual no requería de mucho detalle, aún así se generó una versión *high poly* con la que se realizaría un bake en Substance Painter y se texturizaría en el mismo programa.

Las hojas de las palmeras y las lianas siguieron un proceso diferente a la de los árboles y el resto de la vegetación, ya que se partió de una versión *high poly*, sobre la que se modeló un plano que cubriera toda la hoja y que además tuviera las suficientes divisiones como para poder doblarse y recrear las formas de las hojas de una palmera o interactuar con el viento.



[51] Captura de pantalla de una hoja de palmera tomada en Substance Painter.

A continuación, se texturizó en Substance Painter, con el que gracias a los mapas de opacidad se pudo ajustar la forma de los planos poco detallados al contorno detallado del modelo *high poly*. Finalmente, tras *bakear* y texturizar todas las hojas, se reimportaron a Maya, dónde se construyeron todas las variantes de palmeras, se corrigió su *pivote* y se eliminó su historial.

A partir de aquí es posible realizar un *bake* y almacenar casi todo ese nivel de detalle en el modelo con menor carga poligonal, y mediante el uso de mapas de opacidad, ahorrar una cantidad significativa de polígonos.

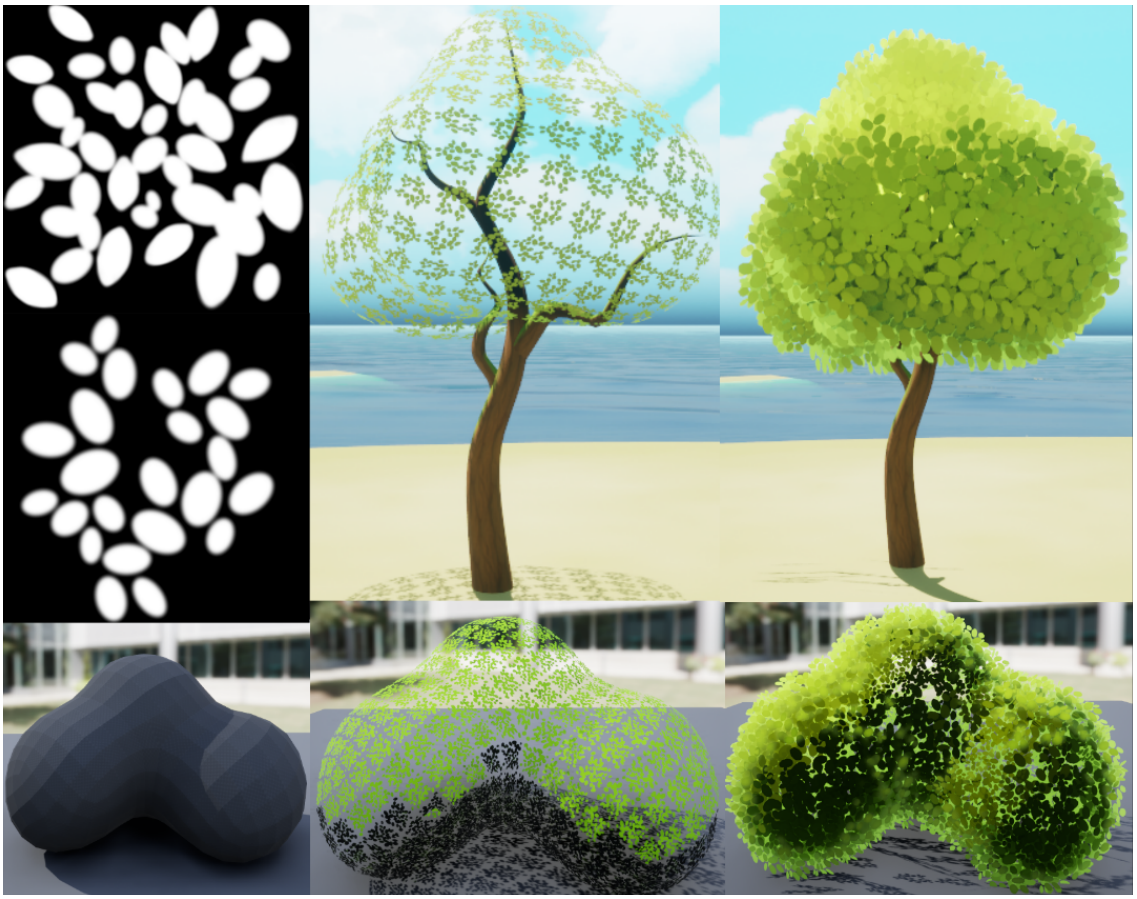
Para la creación de las hojas de los árboles se estudiaron varias alternativas, inicialmente se pensó en usar el programa *TreeIt*. En el que se quiso recrear la forma del árbol y posteriormente añadir los grupos de hojas mediante las funcionalidades del programa.

[52 y 53] Captura de pantalla de *TreeIt* y texturas usadas para la creación de las hojas.

El programa funciona mediante un mapa de texturas con un alfa, con el que pueden crearse y distribuir las hojas de forma orgánica, además, cuenta con un montón de posibilidades para modificar tanto las ramas del árbol y su tronco, como las mismas hojas.

Aunque el autor trabajó tendidamente con el software, hasta sentirse cómodo con el mismo, se exploraron otras opciones para acercarse a la estética deseada y de manera que fuesen poco costosas a nivel de recursos. De la misma manera que en el caso anterior, el autor partió generando varios *alfas* en Photoshop, a los que se le aplicó un blur ya que servirían como máscaras de opacidad para generar conjuntos de hojas mediante los materiales hechos por nodos en Unreal Engine y de esta forma evitar "*jaggies*".

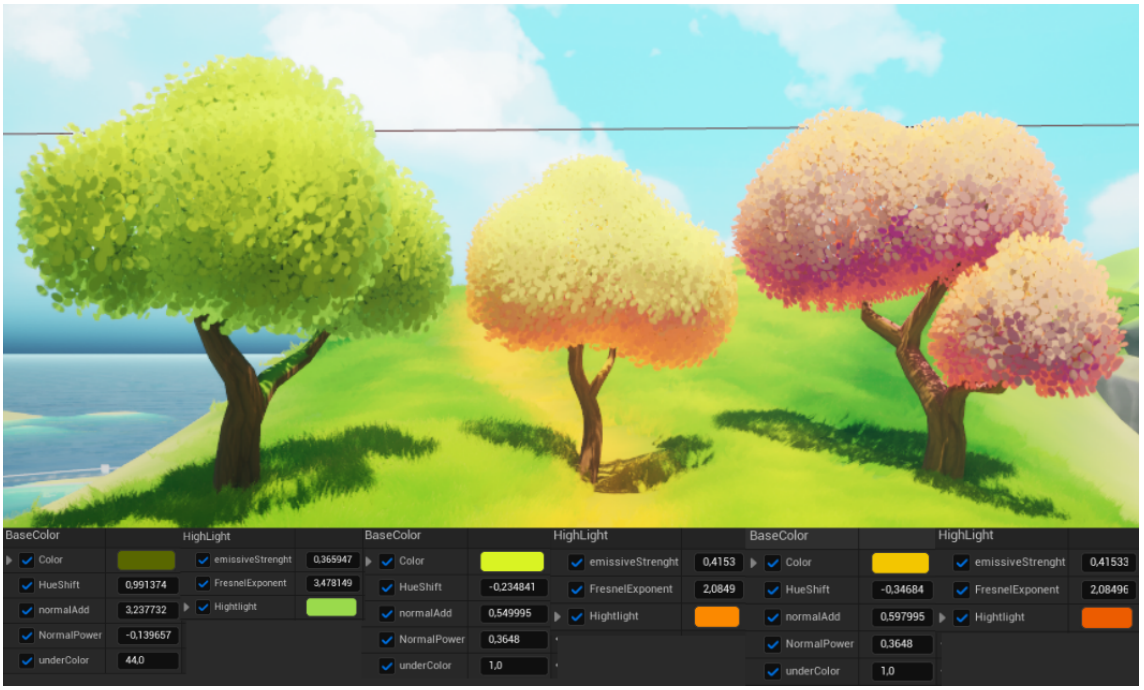
En vez de proyectar las hojas sobre planos individuales que colgasen de ramas, como se pensó hacer inicialmente, se decidió esculpir las copas de los árboles en Zbrush, partiendo de una esfera. Una vez el autor estuvo contento con el resultado, disminuyó considerablemente el número de polígonos de las copas y desplegó cada una de las caras del modelo una encima de otra, de la misma manera que se hizo con los troncos de los árboles. De esta manera se pretende conseguir que las hojas se proyecten sobre cada una de las caras del modelo de la copa, inicialmente puede sonar extraño ya que se obtendría un resultado similar al siguiente:



[54] Conjunto de capturas de pantalla de los alfas, las copas de los árboles y los árboles ya integrados, con y sin modificar su *World Position Offset*, tomadas desde el editor de Unreal Engine.

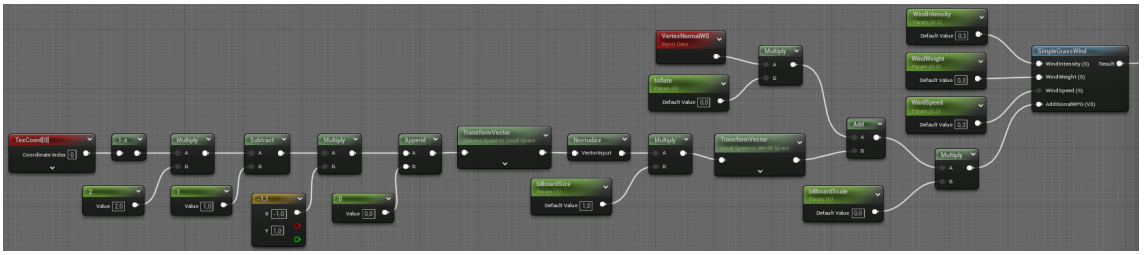
En la *figura 41*, se muestra el proceso de cómo las alfas generadas en photoshop se proyectan sobre cada cara del modelo de la copa, sin embargo, en el árbol que se muestra más a la derecha de la imagen puede verse como las hojas se distribuyen, se orientan y se escalan de forma distinta por todo el modelo. Esto es debido al "*World Position Offset*" o WPO, que permite manipular los vértices de una malla a partir de un material, de esta forma se le pueden aplicar transformaciones mediante vectores, para afectar a su posición, rotación, escala y a sus normales.

En la misma *figura 41*, también se puede observar zonas sombreadas y una especie de contorno iluminado que crea cierta frondosidad y una falsa profundidad, esto es gracias a que los *billboards* no se comportan de manera individual, sino como las caras del modelo sombreando la malla de forma global. Sin embargo el color emisivo que se le ha aplicado, funciona usando el WPO, por lo que crea contornos de colores vivos.



[55] Conjunto de capturas de pantalla del proyecto, que muestran la coloración de varios árboles y el resultado de regular algunas de las expresiones de sus materiales.

Se han programado también expresiones para cambiar el color base de sus hojas, el tono, color emisivo, la potencia del mismo y la del efecto de sus normales, es decir la capacidad de dibujar profundidad mediante el sombreado. De esta forma es posible crear varias combinaciones de colores para repartirlos por todo el mapa. Todas estas expresiones podrán ser modificadas en tiempo real siempre y cuando se cree una instancia del material y se aplique al modelo.



[56] Captura de pantalla tomada del material de las hojas creadas en el editor de Unreal Engine.

Los nodos de la *figura 42*, son los responsables de que los alfas usados como hojas se comporten como algo conocido como "*Billboards*", estos son planos o texturas 2D que siempre apuntan hacia donde mire la cámara. Se han programado también, mediante *material expressions*, el tamaño de cada uno de los planos, la escala de la copa del árbol y además se ha añadido un efecto *inflate*, que escala de forma global todos los planos, separándolos unos de otros como si se estuviera inflando la copa.

Cómo la sucesión de todos estos nodos se conecta al *output* del material *World Position Offset*, se ha aprovechado para añadir también el movimiento de viento que afecta a las hojas, se trata del nodo de color azul que se encuentra al final de la *figura 42*, y los 3 parámetros verdes que salen de él. Pero se hablará con más detalle sobre el viento y el *foliage*, en la creación del césped.

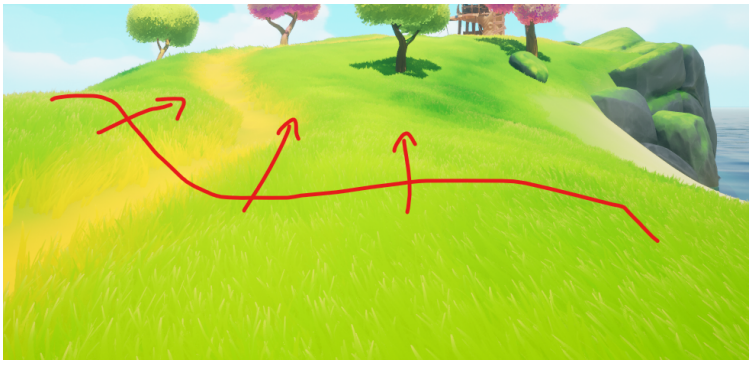
Foliage

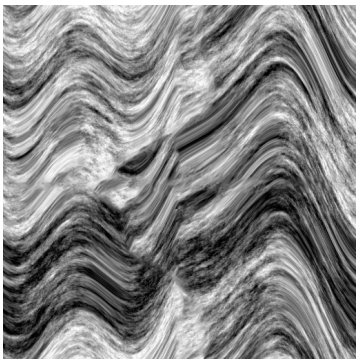
Una vez creados los modelos de toda la vegetación, se usará la herramienta *Foliage* de Unreal Engine, que permite pintar césped, arbustos o árboles directamente sobre la superficie del terreno creado. Con la opción para customizar los pinceles, se puede lograr una mayor densidad en función del modelo a pintar.

Unreal Engine también cuenta con un seguido de funciones *Wind*, con los que se puede asignar viento a las hojas desde los mismos materiales, y de esta forma, meter todos aquellos elementos que actuarán como follaje, dentro del mapa de UVs.

Si bien se ha usado esta herramienta de una forma más sencilla para las hojas y el resto de la vegetación, una función que viene predefinida por el propio motor gráfico conocida cómo "*SimpleGrassWind*", ésta es algo básica ya que únicamente podemos controlar la intensidad, velocidad y peso del viento sobre la malla. Si se conecta al canal de WPO del material resultará en un movimiento sencillo pero conseguido, en el que la malla se mecerá de forma más o menos violenta en función de los valores de las 3 expresiones mencionadas.

Para el césped se ha querido seguir el ejemplo visual de videojuegos estilizados como "The Legend of *Zelda: Breath of the Wild*" (2017) o bien "*Genshin Impact*" (2020), en los que el césped no sólo se mece con el viento, sino que también le da color, tiñendo las puntas del mismo, siendo visibles las rachas de viento.





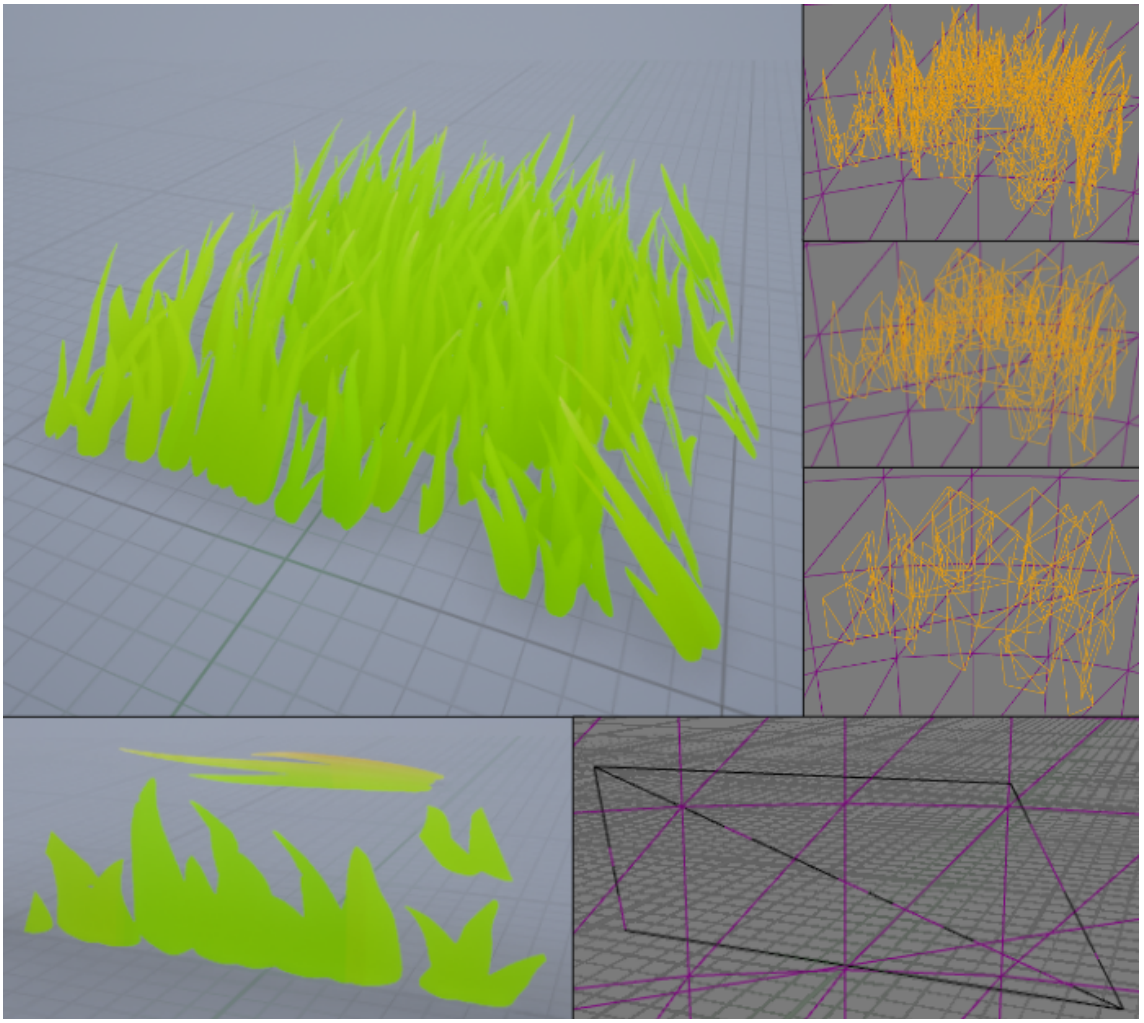
[57] Captura del movimiento del césped provocado por el "viento", tomada en Unreal Engine.
[58] Imagen del ruido *tileable*, generado en Photoshop para simular la fuerza del viento.

Para lograrlo, se genera un movimiento siguiendo un patrón que viene dado por un ruido en escala de grises y *tileable* hecho en Photoshop. El ruido es una imagen de 512x512px que crea ondulaciones. Con el uso del nodo "*Panning*", la imagen del ruido se desplaza en un eje, haciendo que el césped y las flores se doblen en una dirección, con mayor intensidad en función del valor comprendido entre el blanco y el negro, siendo el blanco más claro un 100% de intensidad y el negro más oscuro un 0%.

Para que el césped pueda doblarse, debe contarse con una malla con las suficientes subdivisiones, con tal de que el césped pueda doblarse con la influencia del viento. Si bien el césped se mueve de forma muy fluida, tener tantos vértices en pantalla puede resultar muy costoso a nivel de recursos, ya que al usar la herramienta *Foliage*, tienden a generarse miles y miles de instancias.

Y aunque Unreal Engine proporciona herramientas como *Nanite*, que permite reducir de forma automática el número de polígonos de un modelo a medida que la cámara se aleja del mismo, poco puede hacer cuando se trata de un objeto con tan pequeño y con tan pocos polígonos cómo el césped.

Es aquí dónde entran en juego los *LODs* o *Level Of Detail*, éste es un recurso que se usa con frecuencia en los videojuegos. Los *LODS* son versiones menos detalladas del modelo original que reemplazan la versión con mayor carga poligonal cuando el jugador se encuentra a cierta distancia. Se suelen realizar 2 o 3 *LODS*, además de una versión todavía menos detallada para cuando el jugador se encuentra muy lejos, cada una de estas versiones proyecta el nivel de detalle de la versión original y se comporta de la misma manera frente al viento, ya que se le aplican instancias del mismo material a cada uno de los *LODs*.



[59] Capturas de pantalla tomadas de los diferentes LODs que conforman el césped del proyecto.

Creación de personajes

A partir de una conceptualización previa se realizó un *block out*, para definir las formas básicas del personaje. A continuación se fusionaron dichas formas y se trabajaron en ZBrush para unificarlas y crear una apariencia más orgánica.

[60 y 61] Capturas de pantalla que muestran el efecto de los *Lightning Channels* sobre los objetos 3D.

Una vez realizado el modelo *high poly*, se hará una retopología en Maya, usando la *Quad Draw Tool*, una herramienta que permite dibujar polígonos, directamente sobre la superficie del modelo *high poly*, adhiriéndose a él gracias al *snap* o fijación.

En este proceso ha de tenerse en cuenta la disposición de los polígonos, puesto que en las zonas cercanas a las articulaciones o aquellas que requieran más movilidad, tendrán que contar con una mayor densidad de polígonos para poder doblarse con facilidad en la animación.

Antes de texturizar el modelo será necesario realizar un despliegue de UVs, distribuyendo las partes del personaje por el mapa de texturas, dando prioridad a aquellas que requieran de un nivel mayor de detalle, cómo el rostro del personaje, dándole más espacio y una escala mayor en el mapa de texturas.

Finalmente se realizará un *bake* para almacenar toda la información del modelo con mayor densidad de polígonos, en el modelo *low poly*. Mediante mapas de texturas puede obtenerse una apariencia visual similar a la del *high poly* con un menor número de polígonos y para posteriormente proceder al texturizado.

Agua y *Causatics*

El agua generada en el proyecto viene, en parte, dada a partir del plugin "*water*", de Unreal Engine. El funcionamiento de la misma es algo muy complejo, sobre lo que se ha investigado a fondo y se ha comprendido gran parte de su construcción pese a los conocimientos limitados del software. En conclusión, el agua viene a ser una malla de la misma resolución que el *Landscape* (512x512), cuyo movimiento se controla a partir de un *Master Material* y varias

funciones del mismo. El movimiento de las olas viene dado a partir de un conjunto de nodos que simulan el resultado de la *ecuación de Euler*, para un movimiento periódico, lo que resulta en la creación de onda *trocoidal* u *onda de Gerstner*.

Si bien no se ha terminado de comprender la resolución de los procesos matemáticos, hay una lógica detrás de la apariencia del material del agua, el cual se aprovecha de las propiedades de *Scattering* y *Absortion* del agua para los reflejos, el color de la misma y cómo se comporta al acercarse a la orilla.

De igual manera, se utilizan los mismos recursos para generar los *Causatics*, que viene a ser la apariencia del agua al verse desde dentro de la misma, aplicando efectos de postprocesado como "*Cámera Aberration*", una especie de niebla o *fog* generada a partir de la *Absortion* y el *Scattering* del material . Mediante la WPO y una lógica booleana que funciona mediante "*Water Body Data*", que recoge la altura del terreno en el eje Z, la profundidad del agua, su velocidad y posición, se limita cuándo deben aplicarse los *causatics*.

Aunque se ha investigado mucho sobre el tema y se ha podido comprender parte del funcionamiento del agua, el autor sólo se ha limitado a modificar los valores de los nodos y parámetros ya establecidos. Aún así se han ido adquiriendo conocimientos respecto al tema, para en un futuro, poder incluir la generación de partículas en forma de alfas y conseguir realizar simulaciones de espuma desde cero.

5.3. Postproducción

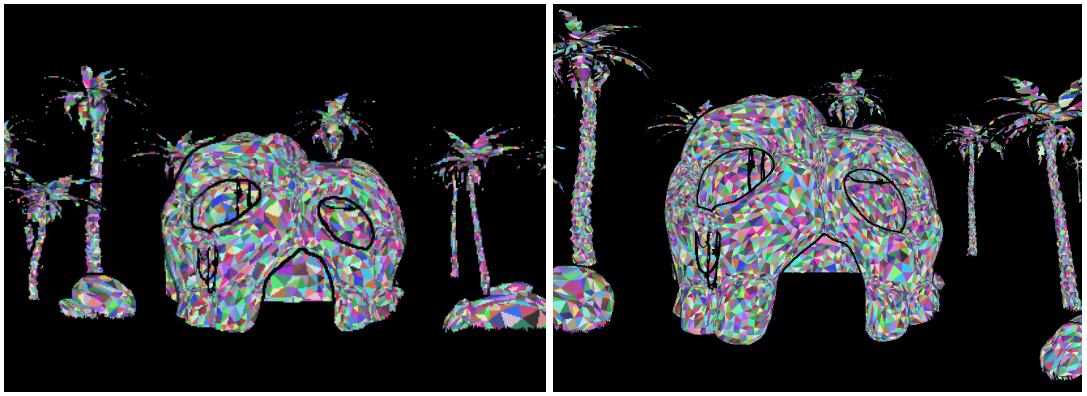
Rendimiento

Si bien es cierto que debido a las limitaciones de tiempo y constantes problemas técnicos, no se ha podido implementar la funcionalidad interactiva con la que iba a contar inicialmente el proyecto, esto ha supuesto la exploración de otros campos.

Se ha investigado ampliamente acerca de la optimización de recursos y el rendimiento, debido a que el equipo con el que se ha realizado el proyecto, ha llegado a su límite en varias ocasiones, llegando incluso a dañarse de forma permanente.

Se ha buscado información acerca del sistema de renderizado de geometría añadido recientemente a Unreal Engine, llamado *Nanite*, del que se ha hablado en algunas ocasiones durante la memoria. Esta herramienta permite redibujar la malla de forma inteligente, mostrando una versión más detallada y con una mayor carga poligonal cuando el jugador se acerca al objeto 3D y reducir el número de polígonos de forma automática y sin que su aspecto se vea afectado a medida que el jugador se va alejando.

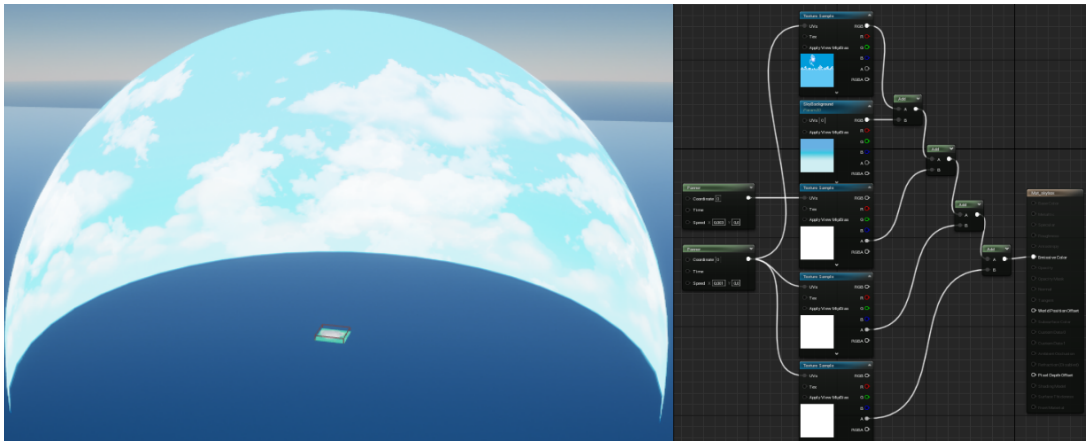
La herramienta se ha introducido en el proyecto y se ha podido experimentar una mejora del rendimiento, aunque para objetos con mallas de reducidos polígonos, no es de gran utilidad, ya que el jugador debe de alejarse demasiado para que el software redibuje su malla. Es por eso que también se ha investigado el funcionamiento de los LODs, explicados en el proceso de producción. Y se ha tenido en cuenta el número de polígonos al realizar las versiones *low poly* de los objetos que iban a cargarse en la escena y el número de mapas de texturas y materiales que se iban a renderizar.



[62 y 63] Capturas de pantalla tomadas de la *Skybox* vista desde lejos y los nodos que forman su material.

Skybox e Iluminación

El cielo del del proyecto, es un *Skybox*, una esfera que ilumina la escena y que puede contener objetos en movimiento, en este caso, nubes. Esto se ha conseguido mediante la creación de nubes pintadas a mano en Photoshop y exportadas en grupo por capas. A continuación se crea un material que debe aplicarse a la esfera, el material se compone de un fondo azul sobre el que se superponen las nubes.



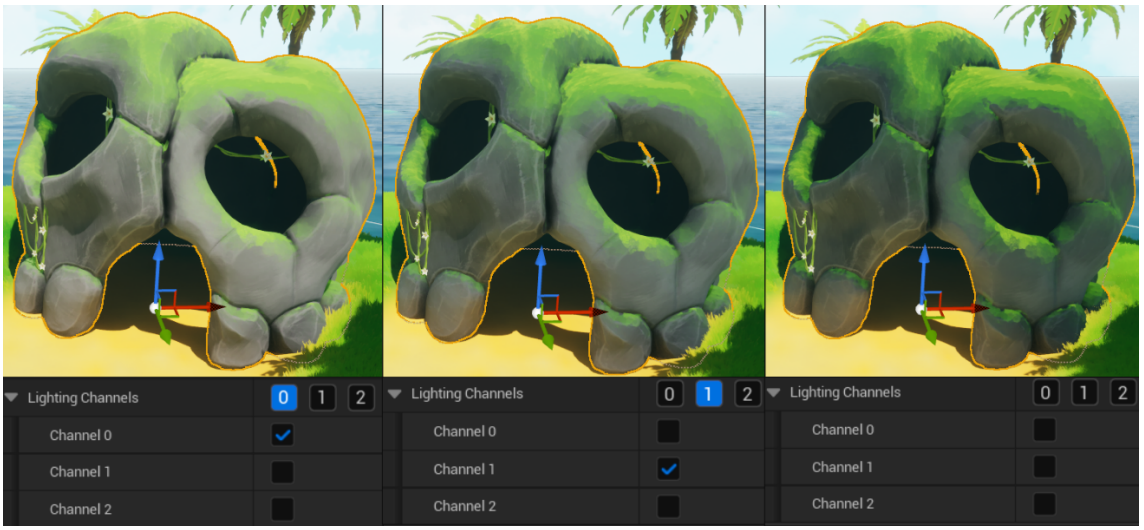
[64] Capturas de pantalla tomadas de la *Skybox* vista desde lejos y los nodos que forman su material.

Cómo se muestra en la *figura 60*, mediante dos nodos *Panning*, con valores diferentes, se consigue que las nubes avancen de forma horizontal, recorriendo la esfera a distintas velocidades y creando un efecto de falsa profundidad conocido como *Parallax*.

Para conseguir ese aspecto más *cartoon*, se ha usado únicamente el canal de emisión y se ha cambiado el modo de visualización de *Lit*, a *Unlit*, por lo que el material, no se ve afectado por la iluminación global.

Siguiendo con el tema de la iluminación, la escena está iluminada a partir de 3 *Directional Lights* distintas, que funcionan como si fuesen un sol. Cada una de las luces, tiene valores distintos y se usan para cosas distintas. Para controlar el efecto de las mismas sobre los distintos elementos del escenario, cada una de las luces se almacena en un *Lightning Channel* propio. De esta manera, podemos indicar en los ajustes de renderizado de cada objeto, que canal o canales, son los responsables de su iluminación.

El primer canal es responsable de la iluminación de casi toda la escena, crea contrastes marcados y una ambientación brillante, el segundo canal está pensado para objetos con una superficie muy lisa o que contienen información emisiva y que expuestos a una luz muy intensa pierden gran parte de esa información. Por último, el tercer canal se usa para evitar que se proyecten sombras en zonas no deseadas.



[65] Capturas de pantalla que muestran el efecto de los *Lightning Channels* sobre los objetos 3D.

También es necesario hablar sobre un sútil efecto de niebla, programada mediante un material en el que mediante el valor de *Scene Depth*, del propio Unreal, se establece una lógica generar la niebla a partir de una cierta distancia, se ha convertido este valor en *material expression*, además de crear otras expresiones para el color y la intensidad de la niebla, que pueden ser controladas en tiempo real al instanciar el material original.

[66 y 67] Capturas de pantalla que muestran el efecto del material de la niebla, tomadas en Unreal Engine.

Postprocesado

Se ha buscado información sobre los efectos de postprocesado y se han modificado la cámara en primera persona, sustituyéndola por una *Cinema Camera*, que ofrece un mayor número de opciones de personalización. De esta manera, se ha creado un sistema de auto enfocado, en el que el jugador traza unos rayos en la dirección donde está mirando, los rayos chocan contra aquello que tenga el jugador, midiendo la distancia. A partir de esta distancia, se puede regular la profundidad de campo de la cámara, creando un efecto cinematográfico.

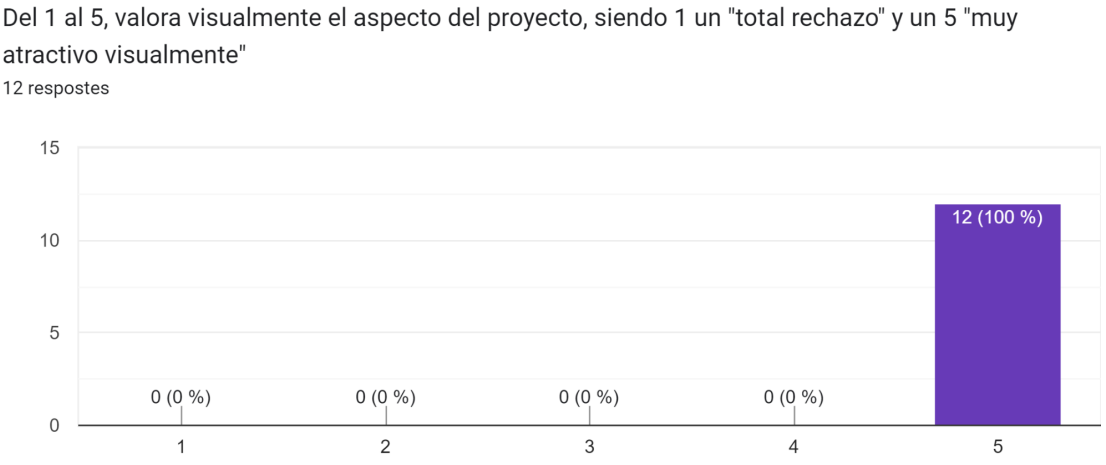
6. Validación del proyecto

Para validar el proyecto, se ha intentado llegar a tres tipos de público distinto, un público más casual, que no tenga conocimientos acerca de la creación de entornos digitales, modelado 3D y las demás técnicas usadas para realizar el proyecto.

Un público que conozca o tenga nociones de cómo se ha podido construir el entorno y los objetos del proyecto. Y un público profesional, con conocimientos extensos en el sector.

Si bien se ha podido llegar a los dos primeros grupos, mediante conocidos y compañeros de clase, se ha intentado dar a conocer el proyecto en foros y espacios donde otros artistas compartan su trabajo, pero no se ha conseguido que respondan a las preguntas formuladas, ni tampoco ha captado demasiada atención.

Aún así se ha difundido una build del proyecto, de la mano de un cuestionario, para obtener información acerca de la validación del mismo. De todas formas, puede que las respuestas no sean demasiado objetivas, puesto que la mayoría de la muestra que participa en el cuestionario, forma parte del entorno del autor.



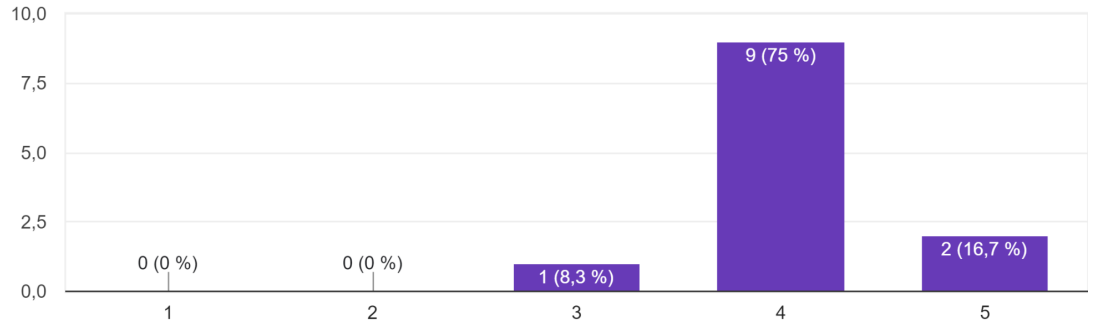
En caso afirmativo, menciona a cual o cuales

9 respostes

- Zelda
- Zelda Totk
- Zelda Breath of the Wild
- El zelda
- un poco zelda y animal crossing
- Sea of Thieves
- Me recuerda un poco a zelda o animal crossing
- Jack and dexter si fuera un remake, también a ciertos niveles de crash bandicoot

Tras probar el proyecto, valora del 1 al 5 el rendimiento del mismo en tu equipo, siendo 1 "injugable" y 5 "fluido y sin problemas de rendimiento".

12 respostes



Especifica si las conoces, las especificaciones del equipo donde has probado el proyecto

4 respostes

- 32gb de ram, gugabyte raedon rx 580, intel i7-8700
- I7-7700, nvidia grx 3080 ti
- Processor: Intel(R) Core(TM) i5-9600K CPU @ 3.70GHz, 3696 Mhz, 6 Core(s), 6 Logical Processor(s)

  RTX 1660 TI

  16GB RAM
- rtx 3070, i7 penultima generación

¿Te ha resultado compleja la exploración del entorno? Si es as explica porqué

12 respostes

- No
- No me ha resultado dificil
- Ha estado perfecta
- Para nada
- Todo ha sido muy intuitivo y facil de aprender, incluso para mi que soy una persona con poca experiencia en videojuegos.
- Me ha resultado muy atractiva tanto visualmente como en la jugabilidad.
- No mucho, el entorno esta bien construido
- Nada compleja la verdad
- no

¿Has echado en falta algún elemento en el proyecto o crees que debería contar con algo que no tiene?

12 respostes

- No
- Encuentro que el proyecto es muy complejo a la par que completo y me ha sorprendido mucho a nivel vidual el resultado obtenido, mis aplausos
- Todo genial
- Me parece todo perfecto
- Es incrible
- Creo que seria interesante incorporar algun animalito para darle más realismo a la escena.
- No creo que le falte nada al proyecto. Esta 10/10. OLEEEEEEE ESE BORJA
- Quizas con algun elemento mas por la playa o por la isla, en algun arbol escondido o algo
- Nada, me parece un proyecto increíble

7. Conclusiones y líneas de futuro.

El proceso de realizar este proyecto, ha cambiado un poco mi punto de vista, empecé muy motivado, llevaba tiempo queriendo crear un entorno en 3D, en el ámbito del arte estilizado, algo que encuentro muy atractivo visualmente y que resuena conmigo a nivel personal.

Tenía conocimientos previos de modelado y texturizado, no estaba muy familiarizado con el modelado de formas orgánicas, pero no pensaba que la creación de un entorno, en un motor gráfico fuese una tarea demasiado laboriosa, quería centrarme en el modelado de las localizaciones y en una sencilla trama que se desarrollaba al interactuar con los personajes.

Si bien estoy contento con mi trabajo, esperaba más, no es que haya perdido el tiempo, ni mucho menos, no he parado de aprender cosas nuevas, me he empapado de conocimiento. Es precisamente por eso que esperaba más. He aprendido a utilizar softwares desde cero, he aprendido e investigado acerca de la creación de materiales procedurales, sistemas de partículas, simulaciones y un montón de funcionalidades de Unreal Engine que no he podido implementar, ya que debía de ser realista con el tiempo, pero que me muero de ganas por probar y continuar con el desarrollo del proyecto.

He perfeccionado mi *workflow* a la hora de crear modelos 3D, unos años atrás veía videos y piezas creadas por otros artistas, las admiraba con una envidia sana, estaba motivado y quería crear cosas como esas, pero no sabía ni por dónde empezar.

A día de hoy puedo decir que me muevo con soltura por varios programas de modelado como Cinema 4D y Maya, que he aprendido a desplegar UVs de forma correcta, sin usar el despliegue automático. Que me siento extremadamente cómodo esculpiendo en ZBrush, ya sean formas orgánicas o superficies duras. Que veo el texturizado con otros ojos, gracias a las ganas y horas que le he metido a Substance Painter y que no se me quita de la cabeza el potencial y todas las cosas que quiero hacer en Substance Designer. Además, de que he descubierto una nueva pasión, me he enamorado de Unreal Engine, con el que antes no soportaba trabajar y que ahora estoy deseando incluir en mi biblioteca de software de confianza.

Puede que me encuentre un poco decepcionado por no haber llegado donde me propuse, pero he encontrado una motivación y unas ganas de crear que hacía tiempo que no tenía y que estoy deseando aplicar.

Definitivamente, voy a continuar con el proyecto y voy a seguir creando universos, aunque igual, de menores dimensiones.

Quiero pulir el resultado obtenido y sacar piezas de él, que pueda mostrar, con la esperanza de que me abran alguna puerta en un futuro laboral, esperemos, no muy lejano.

8. Bibliografía

Documentación de Unreal Engine, consultada constantemente a lo largo del proyecto

- https://docs.unrealengine.com/5.2/

Foros de Unreal Engine, ***Water Causatics tutorial***, por el usuario *papptimus* [Consulta: 20 de junio, 2023]

- https://forums.unrealengine.com/t/water-caustics/

Conjunto de **vídeos acerca de la *First Person Cámera* en Unreal Engine 5**, por Matt Aspland [Consulta: 25-28 de julio, 2023]

- https://www.youtube.com/watch?v=TvXJdDSHzro
- https://www.youtube.com/watch?v=YDOAo_4L2a8

"***Substance Academy Series: Getting Started with Substance Designer***", por Adobe Substance 3D [Consulta: 8-12 de mayo, 2023]

- https://www.youtube.com/watch?v=i_q_JaCg7hk&list=PLB0wXHrWAmCwWfVVurGIQO_tMVWCFhnqE&index=1

Workflow for creating Game Envoirments in Unreal Engine, por Math Roodhuizen [Consulta: 20 de Marzo, 2023]

- https://youtu.be/7nxrUMQzDIk?si=0E8Sr3vfLQaUKINX

"***The Beginner's Guide to Niagara***", por SARKAMARI [Consulta 20 de julio, 2023]

- https://www.youtube.com/watch?v=SAAZRWBry_I

Explicación de cómo generar efectos animados a partir de *Billboards*, por Ghislain Giradot [Consulta: 22 de julio, 2023]

- https://www.youtube.com/watch?v=hLqrU4JRmxc&t=220s

"***Sea of Thieves***" (2018), por Rare Limited [Consultado durante toda marzo y abril]

- https://store.steampowered.com/app/1172620/Sea_of_Thieves_2023_Edition/

CGI - E*stado del arte* [Consulta: 10-15 de febrero, 2023]

- https://www.hooksounds.com/es/blog/cgi-imagenes-generadas-por-computadora/
- https://es.wikipedia.org/wiki/Imagen_generada_por_computadora
- https://www.nfi.edu/what-is-cgi/
- https://www.vectary.com/3d-modeling-blog/level-up-your-3d-design-flow-with-booleans/

3D Modeling - *Estado del arte* [Consulta: 10-14 de febrero, 2023]

- https://grupo-ae.com/historia-y-valor-de-la-tecnologia-3d-en-el-aula/4899007260/

- https://www.easyrender.com/a/a-brief-history-of-3d-visualizations-the-ins-and-outs#:~:text=The%20'70s%20%E2%80%93%20The%20arrival%20of,available%20for%20the%20personal%20computer.

3D Sculpting - *Estado del arte* [Consulta: 12 de febrero, 2023]

- https://www.sculpteo.com/en/3d-learning-hub/3d-printing-software/best-sculpting-software/

3D Rigging and Animation - *Estado del arte* [Consulta: 16 de febrero, 2023]

- https://infocusfilmschool.com/history-of-3d-animation/
- https://www.youtube.com/watch?v=SPMFhcC4SvQ

Motores Gráficos y la Actualidad de los videojuegos - *Estado del arte* [Consulta: 28 de febrero, 2023]

- https://www.xataka.com/videojuegos/indies-tenian-razon-unity-motores-terceros-le-han-ganado-partida-a-motores-propios-a-hora-crear-juegos-1
- https://www.incredibuild.com/blog/unity-vs-unreal-what-kind-of-game-dev-are-you#:~:text=Unreal%20may%20be%20more%20complex,known%20for%20their%20photorealistic%20quality

9. Anexos

En el enlace al Drive, que sirve cómo repositorio, además de la build del proyecto, se puede encontrar una copia del proyecto en la versión 5.2 de Unreal Engine, llamada "FPIsland", en la carpeta, dentro de "Content", se encuentran ordenados por carpetas: los modelos 3D usados, sus texturas y todos los *Blueprints* y materiales creados durante el proyecto.

Me resulta imposible facilitar una versión descargable de los modelos detallados en ZBrush o de los archivos del distinto software usado en la creación, texturización o retopología de los mismos, debido a su tamaño.