

I. MARCO TEÓRICO

A continuación, se observarán las plataformas que contendrán los trabajos realizados. Estas son: *Second Life* y *Unity*. La mayor parte de los proyectos para la primera se realizaron dentro de la misma, mas en el caso de *Unity*, se recurrió a software externo para desarrollar el contenido.

I.1. SECOND LIFE

Second Life es un mundo virtual desarrollado por *Linden Lab*, una compañía privada de Internet en Estados Unidos, cuyas oficinas centrales se encuentran en San Francisco. Fue lanzado el 23 de junio de 2003. Es un programa que les permite a sus usuarios interactuar entre sí mediante *avatares*. Las diversas locaciones en Second Life son conocidas como islas, mismas que pueden ser compradas por usuarios para la creación de contenido exclusivo.

Lo que empezó como una herramienta recreativa y de interacción social con el paso de los años terminó por llamar la atención de instituciones educativas, convirtiéndola en un medio socorrido por universidades y corporaciones.

La UNAM decidió incursionar en este campo, colocando dentro de Second Life un compendio de cursos educativos a distancia, demostraciones virtuales, tutoriales e incluso prácticas para materias que son impartidas en la universidad actualmente, en el Anexo de Ciencias Básicas, como lo son las prácticas para la materia de Geometría Analítica, disponiendo de varias islas para estas tareas.

A pesar de que la herramienta ha caído en desuso y no se espera que su popularidad continúe en el futuro, fue de gran utilidad en su momento, sin mencionar que sentó bases sólidas para proseguir con la virtualización de demostraciones, tutoriales y otras actividades interactivas.

I.1.1. Herramientas de trabajo

I.1.1.1. Modelado geométrico

La mayor parte del trabajo realizado en Second Life fue referente al modelado geométrico. El entorno cuenta con su propia herramienta para esta tarea, misma que, puede decirse, es muy básica.

El modelado geométrico interno en Second Life consta en añadir figuras geométricas básicas – *prims* – e irlas modificando mediante la ventana interactiva que el entorno presenta.

Las figuras que se pueden añadir son: cubo, cilindro, esfera y toroide. Sin embargo, dichas figuras, como se ha mencionado, se pueden modificar.

La forma más directa y sencilla de alterar estas formas es mediante las transformaciones geométricas: traslación, escalamiento y rotación. Sin embargo, a menos que nos remitamos a un

objeto en extremo simple, no bastará con aplicar estas transformaciones solamente para obtener el efecto visual deseado.

Alterando los parámetros de la figura, puede adquirir una forma muy distinta. A continuación, presentamos una imagen de la ventana que nos permite alterar los parámetros:

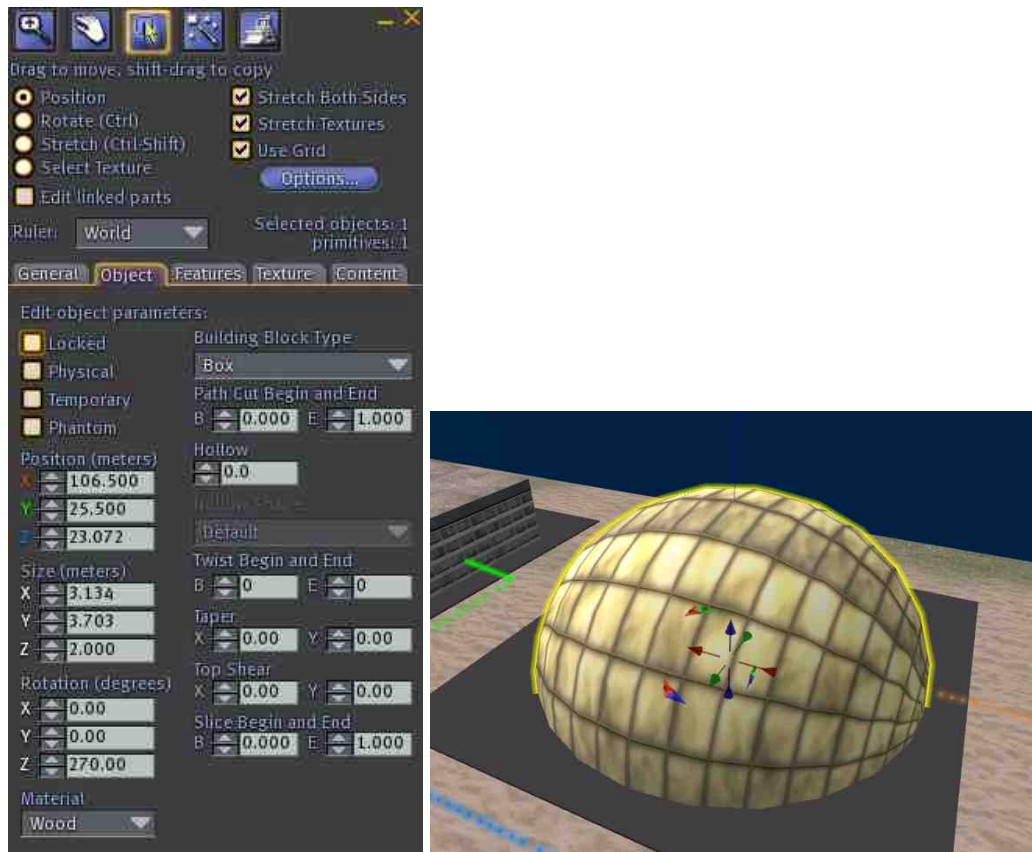


Fig. 1.1 y 1.2: parámetros alterables y ejemplo de prim.

Como podemos apreciar por la imagen, la posición, tamaño y rotación del elemento pueden ser cambiados de la forma más exacta con estos parámetros en las tres dimensiones; ciertamente más todavía que mover nuestros objetos con las flechas auxiliares que aparecen en el mundo virtual.

Los parámetros que se pueden ver en el menú desplegable (que en la imagen se muestra está activo el parámetro “Box” – Caja o Cubo) modifican directamente la figura para que tome una forma distinta, es decir, podemos cambiar nuestro cubo por una esfera o un cilindro.

A continuación, vemos los parámetros “Path Cut Begin and End”. Éstos sirven para “recortar” el objeto en una medida deseada, ya sea desde su inicio o fin.

Para ilustrar mejor esto, imaginemos un toroide el cual se forma a partir de un punto inicial y termina de formarse en un punto final. Ambos puntos se encuentran en el mismo sitio, como si se

tratase de un sólido de revolución. Al alterar pues, los parámetros de inicio y fin, podemos indicar a partir de dónde comienza a dibujarse nuestro toroide y dónde finaliza. Esto hace de dicha herramienta una de las más socorridas durante el modelado de nuevos objetos.

Asimismo, también contamos con el parámetro “Hollow” (Hueco), el cual nos permite crear un agujero en la figura y modificar su tamaño. De este modo, podemos crear objetos similares a marcos de puertas y paredes con un hueco para una ventana, entre muchas otras cosas.

El resto de los parámetros que podemos encontrar más abajo modifican la forma del objeto directamente, inclinándolo más hacia un eje determinado o torciéndolo en cierto sentido, por ejemplo.

Adicionalmente, podemos añadirle texturas a nuestra figura, para darle un mejor aspecto. A continuación, otra imagen que muestra la interfaz de adición de texturas:

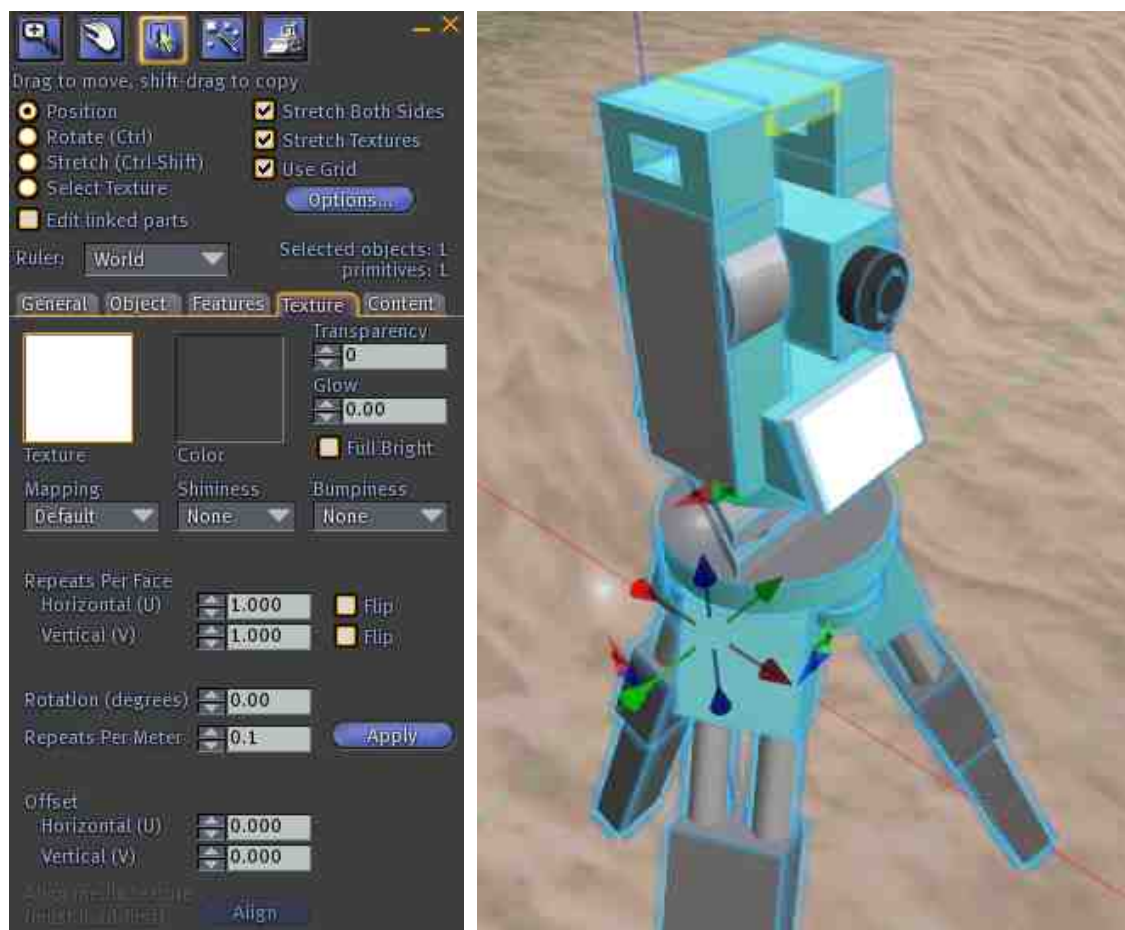


Fig. 1.3 y 1.4: adición de una textura y prims con color alterado.

Second Life tiene una biblioteca inicial de imágenes que podemos utilizar como texturas. La variedad es sin duda alguna buena para iniciar, pero con el tiempo, será necesario subir imágenes

para crear nuestra propia biblioteca. También puede aplicársele un color a nuestra textura, para darle una tonalidad diferente.

El modo de mapeo de texturas es uno planar muy básico, el cual funciona perfectamente para figuras cúbicas. No obstante, puede representar problemático al aplicarlo a figuras como esferas y cilindros, por lo cual hay que tener en cuenta los parámetros que podemos utilizar para acomodar nuestras imágenes para que den el efecto deseado.

Los parámetros son muy claros en cuanto a su efecto: “Repeats per face” – repeticiones por cara – da un efecto de mosaico mientras mayor sea el número ingresado en las coordenadas U o V (horizontal o vertical). Si el número es menor a 1, sólo se mapeará un fragmento de la imagen. De igual forma, podemos también añadir un efecto de “espejo” a la textura para que ésta se mapee al revés; “Rotation”, rotación, nos permite girar la imagen en cierto número de grados y el parámetro “Offset” representa el desplazamiento que queremos la imagen tenga en las coordenadas U o V.

Existen otros parámetros modificables que podemos ver de menús desplegables. El tipo de mapeado nos permite utilizar un mapeado planar distinto que no se amolda a la figura y que nos puede servir para objetos que hemos alterado. “Shininess”, brillo, nos permite darle un efecto metálico a la figura y “Bumpiness” da un efecto de rugosidad con mapeados predeterminados.

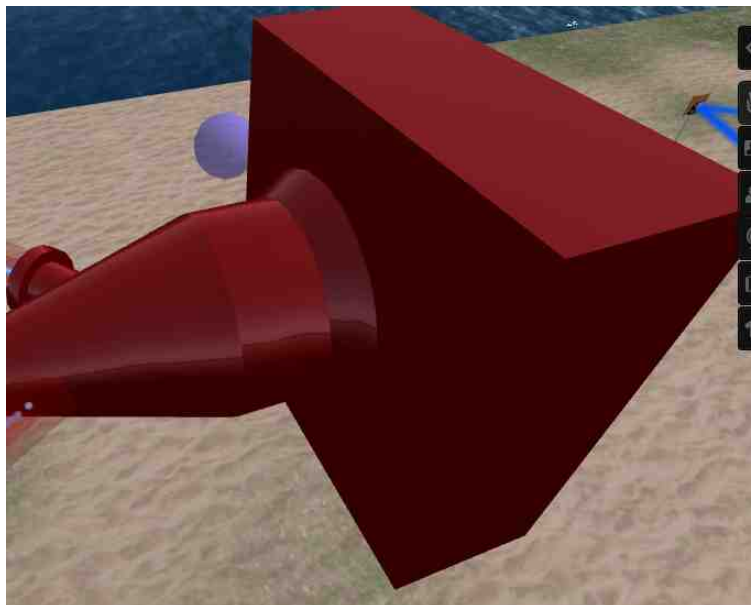


Fig. 1.5: ejemplo de prim con brillo.

Por último, podemos controlar el grado de transparencia que deseamos tenga nuestra figura y el brillo, aunque desde este menú nos es imposible dar un efecto de transparencia total. Más adelante aprenderemos cómo es que se resuelve este problema.

Así pues, tenemos varias herramientas para modificar nuestro objeto y darle el aspecto que deseemos. Podemos utilizar varias figuras alteradas para componer un modelo convincente y llamativo.

Lamentablemente, cada isla tiene un límite de *prims* que pueden utilizarse. Por ello, para algunos modelos, es posible recurrir a herramientas externas de modelado geométrico, como lo son 3D Max, Blender y Maya, por nombrar algunas.

Existen *plug-ins* para estos programas que nos permiten guardar los modelos realizados como imágenes *.tga*, mismas que Second Life puede interpretar como “*sculpties*”. Así como podemos seleccionar del menú desplegable la figura que deseamos trabajar – cubo, cilindro, esfera, etc. – también podemos elegir usar un sculptie y asignarle la imagen creada a partir del modelo hecho en el software externo, misma que Second Life interpretará como un nuevo objeto y le dará la forma aproximada del modelo creado.

Desafortunadamente, no es tan cómodo como aparenta ser. Second Life interpreta muy pobremente modelos demasiado complejos, por lo que si se decide hacer una figura muy complicada, como un modelo humano, será necesario utilizar varios sculpties para diversas partes de su cuerpo.

Pese a este inconveniente, se puede apreciar un efecto más convincente que si sólo usásemos *prims*. Adicionalmente, podemos emplear texturas personalizadas en nuestro modelo, tomando ventaja de que Second Life respeta el esquema de mapeado que le hayamos asignado a nuestro sculptie en el software externo, sin importar lo complejo que sea.

1.1.1.2. Programación

El proceso de modelado en Second Life por sí solo puede rendir entornos virtuales convincentes. Sin embargo, para poder interactuar con ellos, es necesario recurrir a la programación.

Second Life funciona mediante scripts Linden (LSL scripts), que es un lenguaje orientado a eventos. Usualmente se añade un script a una figura para que ésta siga las instrucciones establecidas en el script.

La siguiente es una imagen de la ventana que ya conocemos para modificar figuras, mas ésta nos muestra la pestaña para poder añadir elementos externos (que pueden ser animaciones, otros objetos y, por supuesto, scripts):



Fig. 1.3: adición de elementos a la figura.

Debido a la orientación del lenguaje de programación, usualmente encontraremos que un objeto llevará a cabo lo que dicte su script hasta que se den las condiciones para que esto suceda. Ejemplos de eventos que activan los scripts son que un avatar “toque” dicho objeto, se siente sobre él o ingrese un comando mediante el chat de Second Life.

A lo largo de este informe podremos ver diversos proyectos. Algunos de ellos requirieron de la adición de un programa muy sencillo, ya que su finalidad era ser demostrativos o formar parte del entorno, mientras que en otros el peso de la programación fue muy substancial.

I.2. UNITY

Unity es una herramienta para desarrollo de videojuegos en 3D. Sin embargo, puede utilizarse para otros fines, como bien se ha demostrado en la DECD.

Unity es una herramienta similar a *Director*, el motor de juegos de *Blender*, *Virtools* o *Torque Game Builder* en lo que se refiere a poseer un ambiente gráfico integrado, mismo que utiliza como el método primario de desarrollo. No obstante y como se demostrará posteriormente, es posible utilizarlo en conjunto con otras herramientas de software. Primordialmente, se recurrió a software de modelado geométrico externo para el desarrollo de los ambientes tridimensionales.

El editor funciona en Windows y en Mac OS X y es capaz de producir juegos para ambos sistemas operativos, el iPad, el iPhone y la plataforma de Nintendo, Wii. Más recientemente, se ha añadido el soporte para las consolas de videojuegos Xbox 360 y PlayStation 3.

Debido al declive que presentó Second Life, se optó por comenzar a utilizar Unity para desarrollar los nuevos proyectos de la DECD. Incluso se han migrado algunos ambientes de Second Life a esta plataforma de desarrollo.

I.2.1. Herramientas de trabajo

Durante la experiencia laboral, no se utilizó Unity directamente en la creación de contenido, sino como destino del mismo.

Para el modelado geométrico, se empleó *Autodesk 3ds Max*: este software de paga fue desarrollado por Autodesk y es utilizado por modeladores, comerciales de televisión y estudios de visualización arquitectónica, aunque es mayormente utilizado en la industria de los videojuegos y diseño. Es un software flexible, capaz de soportar varios formatos de archivos y técnicas de animación y modelaje, y cuenta con su propio lenguaje de scripting.

A pesar de que tiene algunos aspectos en los que no es tan completo, hay una vasta cantidad de plug-ins que resuelven esta falla y debido a su versatilidad, la adición de dichos elementos es sencilla.

Para la edición y creación de imágenes, se utilizó *Photoshop*, que es quizás el editor de paga de imágenes más conocido en el mercado. Desarrollado por Adobe, es generalmente utilizado para retocar, eliminar o añadir algunos detalles de fotografías. Sin embargo, tiene las herramientas necesarias para poder realizar texturas de alta calidad. Adicionalmente, es compatible con varios plug-ins que aumentan su repertorio de herramientas; por ejemplo, la capacidad de hacer un mapa de normales de las imágenes (aunque cabe mencionar que no se utilizó dicha herramienta en la creación de mapas de normales).

I.3. OTRAS HERRAMIENTAS DE TRABAJO

En menor medida y en ocasiones para tanto Unity como Second Life, las siguientes son las herramientas de trabajo que se utilizaron para modelado geométrico adicional y la creación de mapas de normales de las imágenes hechas con Photoshop.

En primer lugar, tenemos el software libre *Blender*, que fue utilizado para la creación de modelos geométricos sencillos. Éste es el software de modelado geométrico más conocido. Es utilizado mayormente en diseño y aplicaciones de realidad virtual. Cuenta, además, con un game-engine y con scripting en Python.

Sus desventajas incluyen un motor de render que no es grandioso, una interfaz poco intuitiva, errores que ocasionen que el programa se cierre inesperadamente, dificultad para exportar a otros

formatos (aunque esto puede arreglarse con algunos plug-ins que la comunidad de usuarios de Blender ha desarrollado) y algoritmos de modelado en malla que pueden parecer inferiores, cuando se compara con programas licenciados como 3ds Max. Sin embargo, cuenta con técnicas para hacer UV Unwrap que resultan mucho mejores que las de muchos otros productos, es posible hacer scripts complejos y funcionales, incluye su propio editor de video y, por supuesto, es una herramienta gratuita. Todo esto convierte a Blender en un programa ideal para iniciar a incursionar en el terreno del software de modelado geométrico y para usuarios avanzados enfocados en desarrollo de realidad virtual.

Para la creación de mapas de normales a partir de imágenes dadas, se utilizó el programa *PixPlant*. Éste es un software de paga permite la generación de patrones de una imagen dada; es decir, es útil para el llamado efecto mosaico, en el cual una imagen se repite indefinidamente. En cuanto a la creación de mapas de normales, PixPlant ofrece al usuario varios índices para personalizar el mapa resultante, como la intensidad general de la imagen, el nivel de detalles chicos, medianos y grandes y su ‘rugosidad’.

Por último, pese a contar con PixPlant para la implementación del efecto de mosaico en las imágenes, para este fin, se utilizó el programa de paga *Corel Paint Shop Pro X2*, el cual es otro editor de imágenes y la principal competencia de Photoshop. Sus ventajas incluyen un costo más accesible, el hecho de que ocupa un menor espacio en el disco duro y un mejor desempeño, ya que no demanda tantos recursos de la computadora, a diferencia de Photoshop.