UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ingeniería

Tema de Tesis:

Animación y Reconstrucción sobre la Zona Prehispánica de Monte Albán, Oaxaca

Director de Tesis:

M.I. Noé de Jesús Romero Serrano

Alumnos:

Juan Manuel Jiménez Gutiérrez

Luis Fernando Mendoza Félix

ÍNDICE

RESUMEN

La presente tesis, en términos generales, se basa en el desarrollo de una aplicación 3D interactiva, con la cual será posible realizar un recorrido virtual por la ciudad prehispánica de Monte Albán, cuyas ruinas se ubican en el estado de Oaxaca, México. En conjunto podremos apreciar parte del entorno de ésta ciudad prehispánica, datos históricos y su arquitectura.

En este proyecto se muestra el prototipo, de lo que podría servir como una nueva herramienta turística – educativa, con el objetivo de incentivar y promover los paseos o viajes virtuales por los monumentos históricos, edificaciones e iconos de gran relevancia de nuestro país para lograr un mejor aprendizaje de nuestros jóvenes sobre cuestiones históricas relacionadas con nuestro país.

Capítulo I

Introducción

Actualmente la tecnología ha ayudado a apreciar la arquitectura de una manera más sencilla y práctica, basándose en plataformas de diseño tridimensional, aunado a la manipulación e interactividad que permite dicha tecnología creando lo que conocemos comúnmente como realidad virtual. Realidad virtual: "el nombre mismo en sí encierra mucha contradicción. Algo que es y a la vez no es. Sin embargo no tenemos que complicarnos la vida tratando de explicar la paradoja. La realidad virtual es la representación de las cosas a través de medios electrónicos, que nos da la sensación de estar en una situación real en la que podemos interactuar con lo que nos rodea"[1].

¿Por qué se concibe este medio? El hombre siempre ha buscado la manera de llegar a todos aquellos lugares en los cuales solo la imaginación puede, por lo cual, se han generado herramientas tecnológicas para poder recrear todo aquello que el hombre imagina. Por consecuente, podemos crear paisajes y mundos que queremos dar a conocer a todas aquellas personas que quieran sumergirse en ellos; gracias a esto, podemos ofrecer paisajes verdaderos que se encuentran al otro lado del mundo, trayendo en si al paisaje mismo.

Por ende, el hombre comenzó a interesarse mas en expresar lo que el mundo nos ofrece y todo lo que lo rodea, gracias al cual podía ya no solamente plasmar sus ideas y/o representar su mundo de una forma burda, sino además transmitir su sentir y recrear para los demás las cosas tal cual las imaginaba en su mente. Surge así la realidad virtual.

Este logro sin embargo, solo pudo concretarse plenamente hasta que se inventaron las computadoras, y hoy en día con la evolución de las tarjetas gráficas y la expansión de los medios de comunicación, como el internet, podemos decir que la realidad virtual es parte de nuestras vidas.

La presente tesis se dedica a la creación de un entorno interactivo 3D sencillo en el que se puede hallar algunos elementos de espacios virtuales como texturas interactivas, cámaras virtuales, cinemática, física en tiempo real y personajes controlables.

Las dos directrices fundamentales de la tesis son la creación de interactividad y de una sensación presencial en los espacios virtuales, en cuyo diseño desempeña un papel clave el lugar del espectador. Este se encuentra en movimiento dentro de un espacio virtual, donde puede explorar y visualizar la arquitectura original de las ruinas de Monte Albán, auxiliando de pequeñas narraciones e interacciones con personajes autómatas dentro del medio.

DEBATE SOBRE LOS ESPACIOS VIRTUALES

¿Qué buscas al entrar en un espacio virtual?

"Mi primer objetivo al entrar en un espacio virtual es aprender los métodos de manipulación del espacio… En el momento que dura esta breve orientación, busco información visual o auditiva que me permita saber que acciones provocan algún tipo de reacción… Cuando ya me encuentro cómodo comunicándome con el espacio, puedo comenzar a descubrir cuál es el propósito del mundo. Al igual que esa primera información inmediata fue esencial para ayudarme a orientarme con los controles, lo que busco ahora son puntos de referencia o pistas que me digan a donde ir o que hacer. ¿Hay algún lugar que deba descubrir o una misión que cumplir? ¿Soy un simple observador o un agente activo? Llegado a este punto, estoy en manos de los desarrolladores para guiarme hacia el propósito que hayan dado al espacio."[2] (**Zach Rosen**, *Diseñador y desarrollador, e investigador en el ITP de la New York University*)

"Crear un mundo posible, crearlo con objetos, definir las relaciones entre ellos y la naturaleza de las interacciones entre los mismos. Poder presenciar un objeto o estar dentro de él, es decir penetrar en ese mundo que solo existirá en la memoria del observador un corto plazo (mientras lo observe) y en la memoria de la computadora. Que varias personas interactuen en entornos que no existen en la realidad sino que han sido creados para distintos fines. Hoy en día existen muchas aplicaciones de entornos de realidad virtual con éxito en muchos de los casos. En estos entornos el individuo solo debe preocuparse por actuar, ya que el espacio que antes se debía imaginar, es facilitado por medios tecnológicos."[3] (Jorge Alvarez, *Diseñador y desarrollador*)

"El principal objetivo que busco en un espacio virtual es emular mundos artificiales, que deben cumplir con la mayor cantidad de propiedades del mundo real o una física y lógica bien entendidas por el usuario. De esta manera es posible generar en el usuario la sensación de inmersión, para que se sienta parte de ese mundo."[4]

Un espacio virtual es la construcción de entornos visuales, sonoros y táctiles, que sintetizan fenómenos y objetos artificiales, los cuales tienen un propósito u objetivo, que pueden ser un modelo del mundo real, un fenómeno o una idea. El espacio virtual es una interface para usar, explorar y comprender el mundo artificial, a través de una representación visual-sonora-táctil, sin la exigencia de conocer u operar los detalles, cuyo acceso es posible con la implementación de niveles de detalles. Es una conexión del usuario con el mundo físico o artificial, a través del mundo sintético o virtual.

Las principales características de un espacio virtual, son:

- La presencia. Se refiere a sentirse parte del mundo artificial.
- La telepresencia. Tiene que ver con que personas geográficamente distribuidas tengan presencia simultánea en el mundo.
- La inmersión. Se relaciona con que el espectador se sienta rodeado del entorno.
- Amplio ancho de banda visual. Es contar con despliegues panorámicos, que permitan colocar gran cantidad de información visual en el escenario.
- Estereografía. Contempla la percepción de profundidad y la ubicación en el espacio tridimensional de los objetos.
- Interactividad. Incluye la realimentación que se tiene con otros espectadores a través del medio de la realidad virtual, así como la respuesta en tiempo real del sistema computacional (y el mundo virtual) a las acciones tomadas por los usuarios.

Estos elementos se han explotado exitosamente en la disciplina. Ahora, hay que aprovecharlos para crear mundos artificiales para la exploración de datos, mundos en los que habiten esos datos y las ideas de grupos de investigadores, que participen en la generación de nuevos paradigmas."(Lizbeth Heras Lara y José Luis Villarreal Benítez)

¿Qué importancia tiene el espectador en el diseño de espacios virtuales?

"El espectador es lo más importante a la hora de diseñar espacios virtuales, como lo es el lector en el diseño de una novela." [5](**Ken Perlin**, Profesor del Departamento de Ciencias Computacionales en la New York University)

"En los sistemas de realidad virtual actuales, el espectador es casi siempre representado por un avatar, que puede ser desde sólo una "mano" hasta un "cuerpo" virtual completo. De acuerdo con Javier Echeverría, esta técnica da al espectador mayor cabida para apreciar el ambiente digital: «… hay otro procedimiento para lograr el efecto de inmersión, a saber: estar a la vez dentro y fuera del mundo virtual, siendo actor y espectador de cuanto allí sucede, y todo ello en un entorno para múltiples usuarios."[6] (**Francisco Estrada R.,** Revista Digital Universitaria)

"El espectador es de gran importancia en el diseño de espacios virtuales ya que todo lo creado en estos va dirigido a ellos."[7] (**Huges,** *Universidad de Kansas*)

¿Puedes describir una experiencia de espectador (situación o emoción) que encuentres específica de los espacios virtuales? ¿Puedes comparar la experiencia de un espectador en un espacio virtual con una experiencia interactiva de un espectador de cine?

"Podemos tener la tentación de contemplar la realidad virtual del mismo modo en que vemos el cine. El espectador, en ambos casos, es conducido a un sistema inmersivo de visión y sonido, sumando el ratón o el teclado en el caso de un entorno de realidad virtual. Pero podemos apreciar diferencias más importantes entre la realidad virtual y las películas si atendemos al modo en que los sistemas en tiempo real cambian la forma de transmitir contenido y la estructura del contenido que está siendo transmitido. Las diferencias pueden ser incluso mayores que las que existen entre el teatro y el cine. En la realidad virtual la experiencia del espectador va más allá de la narración secuencial… En un caso, los espectadores de cine siguen un punto de vista que no puede cambiarse; en el otro caso, los espectadores son libres para elegir su propio punto de vista y la localización de la cámara y para crear su propia edición, etc…

El modo en que se transmite el contenido en las películas y en la realidad virtual es tan distinto que ni siquiera puedo estar seguro que puedan ser comparados. Quizás haya un modo de reutilizar algunas experiencias formales tomadas en préstamo de las películas (por ejemplo, encuadre, movimiento de cámara y dominio del tiempo de una escena)."[8] (**Florent Aziosmanoff**, Psicosociólogo, Director de Le Cube, taller para artistas y autores dedicados a la creación de proyectos de realidad virtual de Francia)

"El objetivo de la Realidad Virtual es crear una experiencia que haga sentir al usuario que se encuentra inmerso en un mundo virtual, aparentemente real; para ello, se sirve de gráficos 3D así como del sonido que envuelve las escenas mostradas. La realidad virtual utiliza la visión de un observador, el usuario, quien se mueve dentro del mundo virtual utilizando dispositivos adecuados, como anteojos o guantes electrónicos.

La Realidad Virtual explota todas las técnicas de reproducción de imágenes y las extiende, usándolas dentro del entorno en el que el usuario puede examinar, manipular e interactuar con los objetos expuestos. Un mundo virtual es un modelo matemático que describe un «espacio tridimensional», dentro de este «espacio» están contenidos objetos que pueden representar cualquier cosa, desde una simple entidad geométrica, por ejemplo un cubo o una esfera, hasta una forma compleja, como puede ser un desarrollo arquitectónico, un nuevo estado físico de la materia

ó el modelo de una estructura genética. Se trata, en definitiva, de un paso mas allá de lo que sería la simulación por computadora, tratándose realmente de la simulación interactiva, dinámica y en tiempo real de un sistema."9 (Walter Palermo, Universidad de Palermo)

"La realidad virtual tiene tales características que, al espectador, la experiencia le parece totalmente real, es decir que, durante "el viaje", percibe lo que percibe, con sus cinco sentidos y es libre de actuar en todo momento como quiera, de modo que todo resulta para él como en la "vida real". Ya hay en algunos lugares gente en tratamiento psiquiátrico porque no logran distinguir entre su personalidad "real" y su personalidad "virtual". Tan afines, tan idénticas son las experiencias de ambas "realidades". Es, pues, el momento de preguntarse: ¿Es que la vida es más "real"?

¿Qué es, en realidad, lo que llamamos vida? Lo mismo: Un percibir, a través de nuestros cinco sentidos, una serie de estímulos que interpretamos de distinta manera y un decidir libremente lo que vamos a hacer y un hacerlo y, luego, un experimentar los efectos de nuestra actuación.

En ambos casos, en ambas "realidades", prácticamente iguales, sin embargo, tras recibir e interpretar los estímulos sensitivos, antes de actuar, antes de decidir, pensamos. Pero en ambas, si bien las experiencias, las vivencias pueden ser distintas, el Pensador es el mismo. Y su pensamiento es etéreo, inaprehensible, superior a lo material de ambas realidades y, por tanto, ajeno a las dos. Y por ser superior y ser siempre lo mental causa de lo material y ser siempre el mismo, es más "real" que aquéllas."[10](Francisco-Manuel Nácher)

¿Cómo se puede aprender de los espacios virtuales?

"Veamos, los espacios virtuales pueden mejorar el modo en que vemos cosas y memorizamos información valiosa sobre la realidad de las cosas. Aún vemos el mundo como una tabla. Sólo en un lugar de la costa, mirando hacia el mar, podemos ver que el mundo se redondea en el horizonte y aún así son nuestros conocimientos (y no nuestra experiencia) los que nos hacen creer que es redondo. En este contexto seríamos capaces de experimentar distintas formas de perspectivas de representación con la ayuda de la tecnología. Por ejemplo, podríamos percibir una brizna de hierba desde el punto de vista de un insecto."[11] (**Miró Kirov**, Escultor y artista 3D, colaborador en la creación de un cuerpo humano virtual para el *Advanced Educacional Systems de la New York University*)

"Una de las principales aplicaciones de la realidad virtual en el ámbito académico es la formación en facultades de medicina, especialmente en las asignaturas de anatomía y cirugía. En la Universidad de Washington se está experimentando con clases demostrativas de cirugía virtual. En

esta universidad se ha creado un «cadáver virtual», donde los estudiantes pueden empuñar un bisturí virtual y practicar. En este sentido es fácil imaginar un mundo virtual creado con VRML que represente un completo quirófano virtual internacional, en el que se recogieran las mejores técnicas quirúrgicas de distintos médicos de cualquier parte del mundo; esta información podría servir de aprendizaje para los estudiantes de medicina y también para otros médicos."[12] (**Judkins** , *Universidad de Washington*)

"Una de las aplicaciones educativas más notorias de la Realidad Virtual es el entrenamiento técnico, especialmente el de pilotos de aeronaves. En este caso, con esta tecnología se evitan riesgos que se presentan en el entrenamiento real, tales como tormentas o vientos fuertes que pueden causar accidentes al avión real si el piloto no tiene la suficiente pericia para salir adelante en estas situaciones.

Pilotos de aerolíneas y del ejército utilizan simuladores de realidad virtual para medir sus reacciones en medio de circunstancias virtuales peligrosas, además de su utilización en estos y otros campos del conocimiento, siempre existe la posibilidad de aplicar la realidad virtual para la creación de los propios centros de enseñanza. En este sentido, ya se está experimentando con universidades, campus, bibliotecas, laboratorios y aulas virtuales."[13] (García Ruiz)

1.1 Planteamiento del Problema.

Actualmente existe gran parte de la población que no conoce la zona prehispánica de Monte Albán, por lo cual hemos hecho la pregunta ¿Cómo hacerles llegar este sitio maravilloso sin tener que salir de la comodidad de su casa y se pudiera aprender de él? Así surgió la idea de crear un diseño virtual de las ruinas de Monte Albán.

Actualmente, en la sala destinada para la zona prehispánica de Monte Albán en el Museo de Antropología e Historia no se encuentra ninguna ayuda virtual interactiva para la visualización de la misma. Se elaborará una herramienta basada en la reconstrucción y animación de la zona arqueológica de Monte Albán, Oaxaca, así como ayudas auditivas sobre estructuras arqueológicas importantes, trayendo como beneficio al visitante una mejor visualización de la zona gracias al recorrido que se puede efectuar en ella.

1.2 Objetivo general.

El desarrollo de una interfaz o aplicación usuario-máquina basada en métodos de inteligencia artificial y autómatas finitos para la exploración y reconstrucción de la arquitectura de la Gran Plaza de Monte Albán para el Museo de Antropología e Historia de la Ciudad de México, de modo que pueda utilizarse como herramienta visual y de apoyo educativo, haciendo uso de las técnicas de programación como:

- Programación basada en modelaje de polígonos.
- Programación estructurada.
- Programación orientada a objetos.

1.3 Objetivos específicos.

- Diseñar nuestro paseo virtual como un instrumento de gran valor educativo, en base a información documentada haciendo de este un proyecto sin fines de lucro y abierto a todo público.
- La implantación de autómatas determinísticos para simular Interactividad entre personajes del paseo virtual.

1.4 Justificación.

Actualmente tenemos a nuestro alcance toda una gama de software de diseño 3D con los cuales se pueden recrear espacios virtuales que sirven de apoyo a ciertas disciplinas específicas como la arquitectura, la publicidad o la medicina, y que funcionan adecuadamente satisfaciendo las necesidades para las que son creados.

Con la presente tesis se pretende mostrar al público una parte de la historia de nuestro país desde una perspectiva distinta a como la hemos visto, mediante un paseo virtual. Uno de los objetivos primordiales es lograr una representación óptima con un gran nivel de creatividad y realismo de la

zona arqueológica en cuestión, sin incurrir en gastos excesivos de recursos de cómputo y/o económicos.

1.5 Alcances del proyecto.

El método a utilizar será la elaboración y animación virtuales a través de Software de modelado de tercera dimensión, con el fin de obtener una recreación virtual aproximada de lo que eran las ruinas de Monte Albán en el lapso donde se encontraban erguidas y habitadas.

Dentro del Museo, la maqueta que se encuentra dentro de la sala de Monte Albán es solo una representación monocromática de la zona prehispánica, perdiendo fácilmente el interés sobre la misma, presentándose la oportunidad de generar una manera más gráfica e interactiva de la zona.

El software a utilizar será 3D Studio Max, Quest 3D así como el FaceGen. El software cumple con los parámetros de diseño gráfico necesarios, como la manipulación, modificación y texturización de polígonos editables, así como el requisito del motor de gráficos que se necesita para la visualización de las animaciones de tiempo real gracias a su plataforma basada en APIs de DirectX. El software escogido es muy versátil para poder manipular de una forma clara la implantación de autómatas e inteligencia artificial sobre el mismo.

Se tenían dos propuestas de software para usar en la parte de modelado y simulación, una de ellas era Maya de Autodesk y Virtools de Dassault Systèmes. La razón por la cual se usó 3D Studio Max en vez de Maya, fue la parte de asignación de Bípedos y animación de personajes, ya que al momento de asignación de vértices de la maya con el hueso, es más sencilla, y sobre todo que la licencia de este software es prestada temporalmente por la entidad privada en la cual trabaja uno de los integrantes de esta tesis. La razón por la cual se usó Quest3D en vez de Virtools, es porque Quest3D puede renderizar más fácilmente zonas muy grandes de paisajes con iluminación y personajes en ellas; también se escogió este programa por la facilidad que nos dieron los fabricantes al brindarnos una licencia de prueba de su versión completa.

El proyecto se enfocará exclusivamente en el desarrollo del paseo virtual de la Gran Plaza de Monte Albán, mismo que se mostrará al espectador, utilizando el modelaje y la animación en 3D.

La interacción entre el espacio virtual y el usuario será únicamente mediante el teclado y ratón, no se podrá hacer uso de otros dispositivos periféricos.

La tecnología utilizada para este proyecto solo garantiza su correcto funcionamiento en aquellos equipos de cómputo que cumplan con los siguientes requerimientos mínimos:

- GB en RAM .
- 500 MB Mínimo en disco duro.
- Sistema Operativo XP o superior.
- Procesador Intel Core duo.
- Tarjeta de video de 128 MB, compatible con DirectX9.0 y Shader Model 1.0.

1.6 Estructura.

Este proyecto se encuentra estructurado en 8 Capítulos, de los cuales se da una breve descripción a continuación.

Capítulo I: Introducción. En este capítulo se describen las características generales del proyecto tales como: el planteamiento, los objetivos generales y específicos, la justificación que le da el fundamento y los alcances que se esperan lograr con esta tesis.

Capítulo II: Metodología de desarrollo orientado a prototipos. Capítulo en el cuál veremos cuál fue la metodología que se siguió para la planeación y elaboración de esta tesis.

Capítulo III: Gráficos por computadora. En este capítulo se da una explicación breve sobre los procesos que lleva a cabo una computadora para mostrar información legible al usuario, pero principalmente se adentra en la forma en cómo procesa los modelos tridimensionales, y los recursos de hardware que se requieren para ejecutar estas tareas en específico.

Capítulo IV: Tecnologías de ambientes virtuales. Aquí se da un repaso a los principales tipos de software de modelado tridimensional y motores gráficos que existen en la actualidad, destacando por su importancia aquellos que han revolucionado la técnica de optimización de recursos de la PC ofreciendo un alto desempeño de gráficos tridimensionales en tiempo real.

Capítulo V: Desarrollo del proyecto: "Animación y reconstrucción sobre la zona prehispánica de Monte Albán, Oaxaca". En este capítulo se describen los pasos que se siguieron para la creación del paseo virtual, la metodología que se implementó, las técnicas que se utilizaron para las distintas tareas, y en general la forma en cómo se trabajó con el software que hizo posible el desarrollo del mismo.

Capítulo VI: Autómatas finitos determinísticos aplicados. Se podrá observar en este capítulo la elaboración de los autómatas finitos determinísticos para el manejo del comportamiento de un personaje secundario dentro del paseo, así como la implantación de estos mismos para la elaboración del Menú del recorrido.

Capítulo VII: Evaluación y pruebas del proyecto. Este apartado trata sobre las pruebas que se realizaron al paseo virtual unas vez concluido para detectar posibles fallos en el aspecto funcional, así como la puesta en marcha del mismo. Además de los lineamientos que se establecieron para su evaluación.

Capítulo VIII: Resultados y conclusiones. Aquí se especifican los resultados que se observaron a partir de las pruebas aplicadas al proyecto, junto con las conclusiones derivadas del análisis de los mismos.

Capítulo II

Metodología de Desarrollo Orientado a Prototipos

2. Metodología de Desarrollo Orientado a Prototipos

El Ciclo de vida de un proyecto es parte fundamental de las metodologías con las que se afrontarán los proyectos, ya que decidirá el nivel de satisfacción del cliente en base a la velocidad y asertividad del proyecto.

Hoy en día existen varios Modelos de Ciclo de Vida (MCV), y cada uno de ellos tienen sus propios métodos, herramientas y procedimientos con los cuales afrontar el proyecto.

En nuestro caso, se optó por poner en práctica la **"Metodología de desarrollo orientado a prototipos"**, por ser esta la que mejor se adapta a los objetivos que se pretenden lograr.

En la metodología de prototipos, un prototipo consiste en dar una "vista general" de la implementación que se generará en base a los requerimientos del cliente, obteniendo de esta manera una retroalimentación temprana por parte del cliente.

A continuación se muestra en la figura 1 un diagrama de las etapas de desarrollo usando la metodología de desarrollo orientado a prototipos:

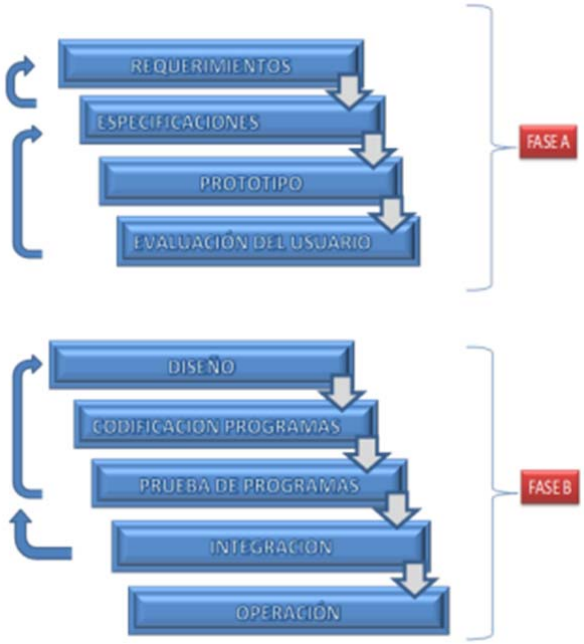


figura 1.

Este modelo aplicado a nuestro proyecto en específico derivó en la siguiente forma de desarrollo:

FASE A: Las principales actividades que se realizaron en esta primera fase fueron: Análisis de los requerimientos del sistema, integración de los requerimientos, selección del entorno y creación de la geometría, al finalizar esta fase el proyecto prototipo estaba listo para ser evaluado por el cliente.

FASE B: En la segunda fase se inició con la conversión de la geometría, se trabajó con la implementación de tecnologías de entorno, se detallaron aspectos visuales y se integraron las distintas etapas hasta dejarlo listo para su puesta en operación.

2.1 Requerimientos.

Antes de sentar los requerimientos, quisiéramos hacer hincapié en un artículo que el Periódico Mexicano "El Universal" publicó el día Jueves 16 de diciembre de 2010 escrito por Yanet Aguilar y Sonia Sierra, quienes nos dan un reporte de las condiciones en las que el mexicano se ubica actualmente ante la lectura y visita de los Museos, factores que también forman parte de nuestra Tesis.

"El problema con el que nos encontramos es que en los últimos 12 meses en México, sólo 27% de la población ha leído un libro; 86% nunca ha ido a una exposición; 43% no conoce una biblioteca. El lugar preferido para hacer turismo cultural es Teotihuacán y el 24% no tiene ningún libro en casa…

…La población está dividida entre quienes tienen interés por la cultura y quienes sienten una mínima afinidad por el tema. Sólo 44% de la gente ha ido alguna vez a un sitio arqueológico ya sea porque le gusta, por razones académicas o algún otro motivo; 66% nunca ha ido a una presentación de danza y 43% no conoce una biblioteca.

..En un país donde existen 42 mil 614 vestigios arqueológicos, de los cuales 176 son zonas arqueológicas abiertas al público nos damos cuenta que es un problema muy grave al que nos estamos enfrentando ya que a medida que pasa el tiempo observamos como la población mexicana sabe cada vez menos sobre su cultura y esto no es bueno ya que la cultura de México es la suma de lo mejor del pasado y del presente…"

Claro con esto no decimos que toda la culpa la tiene la población ya que en muchas ocasiones no se puede asistir a este tipo de lugares aunque sean gratuitos no por falta de interés sino porque en algunas ocasiones se encuentran a distancias que la mayoría de la población por razones económicas, falta de tiempo, etc. no tiene la oportunidad de visitar.

Por lo que proponemos una tarea para ayudar a fomentar más el interés de la población en aprender sobre nuestras culturas en especial la cultura Mesoamericana localizada en la Zona Arqueológica de Monte Albán en Oaxaca.

Esto se puede lograr mediante el diseño de un software que de manera virtual contenga un gran nivel de detalle en las estructuras arquitectónicas así como realismo con respecto a la dimensión espacial. Esto es, que cada edifico que se construya virtualmente se encuentre en la misma posición física y que presente las mismas dimensiones en una escala virtual que los que están situados físicamente en la Zona arqueológica de Monte Albán.

De la misma manera el software debe ser interactivo para que sea más atractivo al usuario además de contener la recreación del paisaje que rodea la Zona arqueológica, para que así el usuario pueda aprender mas sobre este maravilloso espacio cultural que nos brinda la cultura mesoamericana en México y así incrementar su interés por ella.

2.2.- Especificaciones.

El software debe contar con buen manejo de luces y sombras, buenos efectos de sonido, con gran calidad en las texturas, que maneje una interacción con el entorno y que tenga un excelente desempeño en computadoras personales, por lo cual se necesitaran varias herramientas como son: Quest3D, 3D Studio Max, Sketch Up, PhotoShop y FaceGen[14], herramientas que en el tema 1 se explica la razón del uso de las mismas, para poder cubrir con todas las necesidades antes mencionadas.

Otros requerimientos para el software serian:

- La Programación basada en modelaje de polígonos.
- La programación estructurada orientada a objetos
- La programación basada en modelos matemáticos
- La programación del diseño arquitectónico tridimensional
- Programación de la animación en tiempo real

Los cuales serán mencionadas a detalle más adelante.

2.3.- Diseño.

Para el diseño del paseo virtual se tiene contemplado usar herramientas de modelado tridimensional ya antes mencionadas 3D Studio Max, de la compañía Autodesk. 3D Studio Max es un programa de modelado y animación en 3D . Ofrece gran capacidad para crear cualquier cosa existente en la vida real. Su potencia y posibilidades de uso son inmensas.

Para la etapa del Diseño se contemplaron los puntos de Modelado e iluminación por computadora.

Modelado.

El modelo hablando de Geometría poligonal, consiste en la creación de polígonos tridimensionales que son parte básica del modelado tridimensional de los objetos, los cuales están constituidos por varios componentes como son caras, vértices, aristas, planos, coordenadas de textura y normales; siendo esto lo que constituye la red poligonal.

Representación de polígonos.

La red poligonal puede ser manipulada por las siguientes operaciones para poder representar el objeto en 3D de la manera más apropiada:

- Encontrar todos los lados incidentes en un vértice y todos los polígonos que comparten un lado o un vértice.
- Encontrar los vértices conectados por un lado así como los lados de un polígono.
- Mostrar la red.

La mejor manera de modelar un polígono es creando uno que consista en 4 vértices, ya que el motor del render siempre hace operaciones con el polígono más básico, que es el de 3 vértices. El

motor del render segmentará el polígono, de tal manera, que queden triángulos, y segmentar un cuadrado en triángulo es más fácil que segmentar un polígono triangular en triángulos.

Es muy importante que cada vértice de los polígonos sea el único en unir al polígono circundante, ya que si se traslapan los vértices se consumirán más procesos y operaciones en motor del render.

Para evitar el problema anterior, 3dMax representa al polígono en base a las coordenadas tridimensionales del vértice que contiene, haciendo más fáciles los cálculos de estos.

Adicionalmente de los vértices y las caras que usemos para modelar el objeto, también vamos a necesitar guardar las normales, ya que estas se tendrán en cuenta cuando se calcule la iluminación de los objetos.

Los objetos tridimensionales pueden ser manipulados en base a sus componentes que son los vértices, aristas, caras o el elemento en sí.

2.4.-Pruebas y Evaluación.

En esta sección se tratará de externar de manera general los resultados obtenidos. Las pruebas basadas en ingeniería de software se explican de manera más detallada en el capítulo 7.

A este software se le realizaron varias pruebas como son:

- Con respecto a la cámara se realizaron pruebas haciendo un recorrido por toda las Zona para ver que funcionara correctamente, verificando que no tuviera algún tipo de comportamiento inusual, que se viera correctamente, que no tuviera un mal ángulo de visión, que no se atorara en algún punto, que no le permitiera hacer el recorrido deseado o que no atravesara las edificaciones, además de comprobar que el giro de esta fuera el adecuado para permitirnos tener una extraordinaria visión de toda la Zona Arqueológica.

Después de todas las pruebas antes mencionas obtuvimos resultados muy favorables ya que logramos tener una visión muy buena de la Zona, la cual nos permitió observar tanto el personaje como todo el entorno donde este se desarrolla y así poder disfrutar de cada una de las edificaciones que nos ofrece este paseo así como de un maravilloso paisaje.

- En el Paseo Virtual también se diseñaron dos personajes a los cuales se les realizaron varia pruebas como son: que ninguno de los dos se atravesara el uno al otro como si fueran un fantasma al chocar entres si o que de la misma forma atravesaran paredes o se viera que flotan en lugar de caminar sobre la superficie de la Zona, también se les hizo pruebas de texturización, verificando que a ninguno de los personajes les faltara parte del cuerpo por texturizar así como que cuando se movieran la textura no estuviera bien adherida a los personajes y se observara como si se jalara.

  Con respecto a los controles de los personajes, al personaje principal es decir el que vamos controlando y observando durante nuestro recorrido en la Zona, tuvimos que corroborar que el personaje hiciera los movimientos deseados dependiendo de la instrucción que se le diera, esto quiere decir que si quisiéramos hacer correr al personaje,este se observara haciendo ese movimiento tanto en brazos y piernas, de la misma manera si solo quisiéramos ver caminando verificando que no se vea esto pero de manera inversa.

  Además se verifico que el personaje no se moviera sin tener movimientos en brazos y piernas o que se moviera al presionar cualquier tecla que no fuera las que se determinaron como los controles.

  Respecto al personaje secundario este no puede ser manipulado por el usuario pero al igual que al personaje principal se le realizaron pruebas para que hiciera los movimientos en piernas y brazos y para el caso de este personaje se definió una ruta que seguirá de forma automática y el movimiento de sus brazos será manipulado por un autómata finito determinístico.

- Respecto al diseño del entorno las pruebas realizadas fueron parecidas a la de la cámara, ya que también se hizo un recorrido completo de la zona para observar cada punto de esta y ver que ninguna parte quedara sin texturizarse para al final poder hablar de que el entorno se asemeja mucho a la zona montañosa donde se encuentra localizada la Zona Arqueológica de Monte Albán y así observar que la texturas son muy buenas aunque se podría alcanzar más realismo si en lugar de colocarle una textura que simulara varios árboles, a esta se le colocaran figuras de árboles virtuales independientes, pero por motivos de renderización no se pudo llevar acabo lo antes mencionado, sin embargo la texturización de la Zona es tan buena que nos permite darnos cuenta de cuál era el entorno de Monte Albán.

Acercándonos más a lo que es la Zona podemos ver que las texturas en los edificios son muy buenas ya que cuentan con un realismo bastante bueno, ya que incluso en algunas partes podemos observar como parece que crece pasto sobre las rocas.

Aquí también se revisó que todas las edificaciones estuvieran bien asentadas sobre la plataforma de la Zona, es decir que ninguno de los edificios se vieran como si estuvieran flotando o en su contraparte que ninguno se viera como si estuviera enterrado.

- En este proyecto también se diseñó un pequeño mapa que aparece en la parte superior izquierda, el cual nos ayuda a saber dónde nos encontramos ubicados en el Paseo virtual mediante un pequeño recuadro en color rojo.

Al mapa también se le realizaron pruebas las cuales consistían en ver que el movimiento del personaje y la cámara coincidieran con el recuadro en color rojo dentro del mapa esto es que si el personaje esta frente a la plataforma norte el recuadro en rojo también se encuentre frente a la plataforma norte para así tener una ubicación precisa de donde nos encontramos dentro del paseo virtual.

- Asimismo se realizaron pruebas de manipulación a los controles pero antes de mencionarlas necesitamos decir cuáles fueron las teclas que el usuario debe oprimir para lograr la manipulación del personaje, esas teclas fueron:

  a) la tecla con la letra W es la que permite caminar hacia adelante.

  b) la tecla con la letra S es la que permite caminar hacia caminar hacia atrás.

  c) la tecla con la letra A es la que permite hacer el movimiento lateral izquierdo

  d) la tecla con la letra D es la que permite hacer el movimiento lateral derecho

  e) la tecla con la letra Q es la que permite correr. Esta tecla deberé ser apretada junto con cualquiera de las teclas anteriores

  f) Mouse → Rotación del personaje sobre su propio eje

En resumen la manipulación de los controles son fáciles de reconocer como son la letra W del teclado es la que permite el avance del personaje y si presionamos al mismo tiempo la letra W y Q nos permite que el avance sea más rápido, otra cosa que es buena y practica para la manipulación es que los giros del personaje para movernos en la dirección que deseamos son manipulados por el

Mouse, el cual hace que su control sea bueno y fácil de manejar para que cualquier usuario tan solo con saber estas simples instrucciones de manejo pueda controlar sin problema alguno al personaje dentro del paseo virtual.

Y de la misma manera que las anteriores las pruebas realizadas a los controles fueron corroborar que las teclas antes mencionadas funcionen correctamente y realicen los movimientos que se les asignaron no importando en que parte del paseo virtual nos encontremos así como que al presionar cualquier otra tecla que no sean las antes mencionadas no suceda nada en el paseo, es decir, que nos lleve al menú, que se mueva el personaje, que corra alguna grabación con la explicación de algún edificio de la Zona, que se trabe el personaje o que definitivamente nos saque del paseo.

- También se diseño un Menú, ya que el paseo virtual debe de tener un menú de opciones que permita visualizar y manipular atributos que éste mismo tiene. La creación del menú está basada en técnicas de renderización por prioridad como lo es el manejo del Buffer-Z aunado al manejo de la Interfaz de Usuario Gráfica.

El menú de nuestro paseo virtual está estructurado de la siguiente manera:

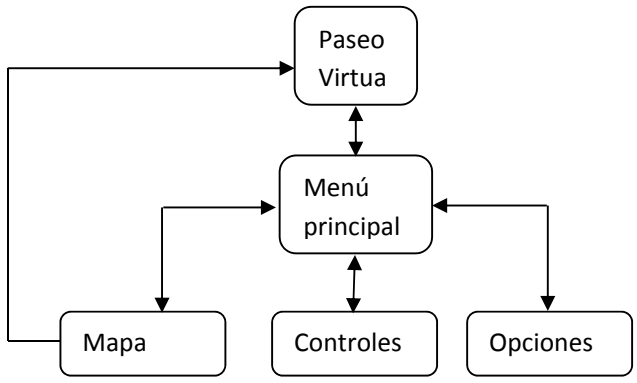


Diagrama de las fases del menú figura 2.

De la figura 2 podemos observar que del paseo virtual pasamos al menú, esto se puede lograr si presionamos el botón derecho del mouse. Una vez adentro del menú observamos que tres atributos que podemos manipular, el primero es el MAPA, el cual contiene accesos directos seleccionables a

las posiciones estratégicas dentro del paseo virtual; el segundo es los CONTROLES, donde podemos observar las teclas y periféricos que controlan al personaje principal dentro del paseo virtual; y por último podemos observar a las OPCIONES, donde nos muestran los atributos modificables de rapidez del mouse y del muñeco, y también los atributos modificables de volumen del sonido ambiental e informativo de los edificios. Para todo lo anterior se llevaron acabo pruebas de manipulación para verificar que no importando en que lugar del Paseo virtual nos encontremos siempre que deseemos realizar estas acciones se lleven acabo sin ningún problema.

- La ultima prueba que se llevó a cabo fue la prueba de sonido la cual consiste que conforme se vaya haciendo el recorrido al Paseo Virtual y nos acerquemos a cada edificio nos de una explicación de cada uno de ellos de una forma muy clara y precisa corroborando que la explicación dada corresponda a la edificación en la que nos encontramos.

Todo lo anterior se explicará con más detalle en capítulos posteriores.

Evaluación.

Para le evaluación del sistema se realizaron varias pruebas en diferentes computadoras y nos percatamos que para obtener un buen rendimiento el equipo debe de contar con las siguientes especificaciones mínimas recomendadas:

a) Procesador Intel i2.

b) 3 GB en RAM.

c) 100 MB Mínimo de espacio disponible en disco duro.

d) Sistema Operativo XP o superior.

e) Tarjeta de video de 512 MB compartida. Se recomienda para mejores resultados GeForce 200 Series o ATI Radeon HD 2000 Series

Ya que si el equipo no cumple con estas características al realizar el paseo nos encontraremos con varias dificultades como que el personaje es más lento, las texturas se ven de una manera muy caricaturesca y pierden su resolución, las explicaciones al acercarse a cada edificio se traban o en el peor de los casos ni siquiera se escuchan, la iluminación de la zona es sumamente mala incluso el paisaje luce opaco y si queremos movernos más rápido con la ayuda del mapa no podremos hacerlo ya que ni siquiera nos deja ver la interfaz.

Capítulo III

Gráficos por Computadora

3.1 Breve historia de los gráficos por computadora.

La utilización apropiada y conveniente de la tecnología ha hecho de la computadora un dispositivo perfecto para realizar imágenes de forma económica y rápida.

La graficación por computadora es el campo encargado de proyectar de manera tridimensional los modelos 2D con ayuda del procesamiento de una computadora, bajo técnicas y métodos de desarrollo gráfico.

En las últimas tres décadas, las técnicas de modelado han evolucionado significativamente, entre las cuales se han empleado modelos basados en líneas, polígonos, puntos y superficies, sin embargo estos no han sido suficientes para simbolizar las características complicadas de los objetos y fenómenos naturales ya que los modelos matemáticos utilizados suelen ser en ocasiones poco controlables.

Por lo que se optó por desarrollar técnicas más avanzadas de modelado, con la finalidad de suministrar mecanismos eficientes, concisos, controlables y flexibles para animar y determinar los objetos naturales.

En el mundo real los objetos tienen una definición perfecta y una gran cantidad de detalles en su composición siendo algunos invisibles a simple vista, por lo que al desarrollar un entorno virtual, el primordial reto es proporcionar a los objetos en cuestión de todas esas características para así aportar el mayor realismo posible. Para llevar acabo lo anterior se necesita guardar estos detalles de una manera digital, pero que nos permita su manipulación en tiempo real sin atraso.

3.2 La geometría de los gráficos por computadora.

La manera de poder representar un objeto a través de un medio electrónico es a través de su geometría representada en un plano de 2 dimensiones o, en su geometría en un plano de 3 dimensiones.

- Gráficos 2D

Hablaremos de dos tipos de gráficos bidimensionales, los gráficos de vector y los gráficos de Mapas de Bits:

En los gráficos de vector se guardan datos geométricos precisos, topología y estilo como posiciones de coordenada de puntos, uniones entre puntos, color y grosor. Los sistemas de vectores gráficos también pueden usar primitivas geométricas de forma estándar como son: conos, círculos, cilindros, esferas y rectángulos. Las ventajas de los gráficos vectoriales es lograr estirar, retorcer, mover y ampliar el tamaño de una imagen, así como permitir describir el aspecto de un documento aparte de la resolución del dispositivo de salida. Los formatos más conocidos en gráficos vectoriales son: PostScript y PDF.

En los gráficos de tramas o mapa de bits cada pixel tiene un valor específico como brillo, transparencia en color o una combinación de ambos. Una imagen de mapa de bits tiene una resolución finita de un número específico de filas y columnas. Las computadoras estándares muestran una imagen de mapa de bits de resoluciones como 1280 (columnas) x 1024 (filas) pixeles.

Una matriz de bits que especifica el color de cada píxel de una matriz rectangular de píxeles se denomina mapa de bits. El número de bits dado a un píxel particular determina el número de colores que se pueden establecer a dicho píxel. Por ejemplo, si cada píxel se representa con 4 bits, a un píxel determinado se le podrá asignar uno entre los 16 colores diferentes (2^4 = 16).

A lo anterior se le conoce como profundidad de color o bits por pixel (bpp), que es la cantidad de bits de información necesarios para representar el color de un píxel en una imagen digital. Esto es que una profundidad de bits de n implica que cada píxel de la imagen puede tener 2^n posibles valores y por lo tanto, representar 2^n colores distintos. Los valores de profundidad de color comúnmente son múltiplos o divisores de 8, como son: 1, 2, 4, 8, 16, 24 y 32.

Así que mientras el número de bits aumenta, el número de colores posible se convierte en un gran mapa de color. Así que en mayores profundidades de color, el valor codifica directamente la intensidad de rojo, verde y azul para especificar un color en el modelo de color RGB.

Las profundidades de color más usadas son:

- **8-bits:** Esta constituido de tres bits para el verde (G) y tres para el rojo(R), en cambio para el azul (B) sólo 2 bits ya que el ojo humano normal es menos sensible a la componente azul

que al rojo o verde, por lo que se le asigna un valor menor que a los demás. El resultado es de 256 colores diferentes.

- **16-bits:** Utiliza 5 bits para el color rojo, así como otros 5 para representar el azul y 6 bits para el verde. Éstos, por consiguiente, pueden combinarse para dar 65.536 colores mezclados. Al color de 16-bits se le conoce como "miles de colores" o highcolor.

- **24-bits:** También llamado color verdadero, puede imitar mucho más de los colores que se encuentran en el mundo real, produciendo 16,8 millones de colores distintos.

  Esto se acerca al nivel en el que los monitores con megapixeles pueden mostrar diferentes colores para la mayoría de las imágenes fotográficas. Usa 8 bits para representar el color rojo, 8 bits para representar el azul y 8 bits para representar de color verde, por consiguiente pueden combinarse para dar un total de 16,8 colores mezclados. Al color de 24-bits se le conoce como "millones de colores" o truecolor.

En la tabla 1 de abajo se muestran unos cuantos ejemplos del número de colores que se le pueden asignar a un píxel que se representa con un número de bits determinado.

Bits Por Pixel	Numero de Colores que se pueden asignar a un pixel.
1	2^1 = 2
2	2^2 = 4
4	2^4 = 16
8	2^8 = 256
16	2^16 = 65,536
24	2^24 = 16,777,216

tabla 1.

Archivos de imagen.

Los archivos que guardan mapas de bits tienen, por lo general, uno o varios bloques de información que guardan datos como, número de píxeles de cada fila, el número de bits por píxel y número de filas de la matriz. Un archivo de este tipo podría contener una tabla que asigna números a colores específicos en el mapa de bits denominada tabla de colores.

Algunos mapas de bits no requieren de una tabla de colores. Por ejemplo, si un mapa de bits utiliza 24 bits por píxel, el propio mapa de bits puede guardar los colores en lugar de los índices de una tabla de colores.

A continuación se nombraran formatos para guardar mapas de bits en archivos de disco:

GIF (Graphics Interchange Format, Formato de Intercambio de Gráficos).

El formato GIF es común de las imágenes que aparecen en páginas Web. Los archivos GIF trabajan bien para dibujar líneas, imágenes con bloques de color sólido e imágenes con límites definidos entre colores, estos también se comprimen, sin que se deje información durante el proceso de compresión.

En un archivo GIF se puede almacenar como máximo 8 bits por píxel, por lo que se limitan a 256 colores. En este formato también se puede especificar un color como transparente, de manera que la imagen tenga el color de fondo de cualquier página Web en la que se muestre.

BMP (Bit MaP).

BMP es un formato que Windows emplea para guardar imágenes independientes del dispositivo e independientes de la aplicación. El número de bits por píxel (1, 4, 8, 15, 24, 32 o 64) de un archivo BMP definido se especifica en un encabezado de archivo. Los archivos BMP con 24 bits por píxel son muy usuales, no suelen comprimirse y, por tanto, no son muy apropiados para su transferencia a través de Internet.

EXIF (Exchangeable Image File, Archivo de Imagen Intercambiable).

EXIF es un formato empleado para las fotografías que se capturan con cámaras digitales. Un archivo EXIF contiene una imagen comprimida acorde a la especificación JPEG, así como información acerca de la fotografía (fecha de toma, tiempo de exposición, etc.) e información acerca de la cámara (fabricante, modelo, etc.).

JPEG (Joint Photographic Experts Group, Grupo Conjunto de Expertos en Fotografía).

JPEG es un formato que funciona bien para escenas naturales como fotografías escaneadas donde en el proceso de compresión se pierde algo de información, pero la pérdida suele ser invisible para el ojo humano. El nivel de compresión de las imágenes JPEG puede configurarse, pero mientras mayor sea el nivel de compresión (archivos más pequeños), mayor será la pérdida de información. Los archivos JPEG almacenan 24 bits por píxel, por lo que son capaces de mostrar más de 16 millones de colores, y estos no permiten transparencia ni animación.

El formato de intercambio de archivos JPEG (JFIF) es un formato de archivos comúnmente utilizado para almacenar y transferir imágenes que se han comprimido conforme al esquema JPEG. Los archivos JFIF que presentan los exploradores Web emplean la extensión .jpg.

TIFF (Tag Image File Format, Formato de Archivo de Imágenes con Etiquetas).

Los archivos TIFF pueden guardar imágenes con un número arbitrario de bits por píxel así como diferentes imágenes en un único archivo TIFF de varias páginas. Estos archivos son un formato flexible y extensible, compatible con una amplia variedad de plataformas y aplicaciones de procesamiento de imágenes.

La información relacionada con la imagen (orientación, marca del escáner, muestras por píxel, tipo de compresión, etc.) puede guardarse en el archivo y organizarse mediante el uso de etiquetas. El formato TIFF puede extenderse cuando se precise con la adición y aceptación de nuevas etiquetas, como se muestra en la figura 3.

Detalle de una imagen rasterizada. Si hacemos zoom sobre esta imagen, podemos ver los cuadros (pixeles) que la conforman.

figura 3.

PNG (Portable Network Graphics, *Gráficos de Red Portátiles*).

El formato PNG mantiene muchas de las ventajas del formato GIF pero también contribuye con más funciones. Al igual que los archivos GIF, los archivos PNG se comprimen sin que se pierda información y pueden almacenar colores con 8, 24 o 48 bits por píxel y escalas de grises con 1, 2, 4, 8 o 16 bits por píxel, mientras que los archivos GIF sólo pueden utilizar 1, 2, 4 u 8 bits por píxel.

El formato PNG tiene una mejora con respecto al formato GIF la cual consiste en mostrar aproximaciones cada vez mejores de la imagen a medida que ésta llega a través de una conexión de red. Los archivos PNG pueden contener información sobre la corrección de color para que las imágenes puedan mostrarse con precisión en varios dispositivos de presentación.

- Gráficos 3D

Los entornos virtuales se forman por objetos con características específicas que se pueden construir con una gran variedad de técnicas de software. Cada objeto se arma de primitivas
y está representado por:

- Su geometría
- Sus atributos

Los diferentes tipos de primitivas de las que se componen los objetos son: Vectores, Pixeles, Polígonos (triángulos, cuadriláteros), Primitivas de volumen (esferas, conos, etc.).

Los objetos representados en los entornos virtuales están creados con polígonos en forma de triángulos, por su facilidad para formar otras figuras, así como por ser la superficie más simple que se puede representar con pocos vértices.

Los Polígonos convexos son figuras donde todos sus ángulos interiores miden menos de 180 grados y todas sus diagonales son interiores. Cualquier recta que pase por un lado de un polígono convexo deja a todo el polígono completamente en uno de los semiplanos definidos por la recta. Un polígono es convexo solo si el segmento no corta los lados y todos los vértices "apuntan" hacia el exterior del polígono. Todos los triángulos son polígonos convexos. Un ejemplo de polígono convexo es el octágono que se muestra en la figura 4.

figura 4.

Los Polígonos cóncavos son figuras en las que al menos uno de sus ángulos interiores mide más de 180 grados. En un polígono cóncavo al menos una de sus diagonales es exterior al polígono. En todo polígono cóncavo hay al menos dos vértices que al ser unidos por un segmento, este corta uno o más lados. Los triángulos son los únicos polígonos que no pueden ser cóncavos, dado que ninguno de sus tres ángulos puede superar los 180 grados. Un ejemplo de polígono cóncavo es la figura 5.

figura 5.

Los polígonos tridimensionales son la unidad básica de todos los gráficos 3D realizados en computadora. Por lo que la mayoría de los motores gráficos de 3D están basados en el almacenaje de puntos por medio de tres simples coordenadas dimensionales (X,Y,Z), así como en líneas que

conectan aquellos grupos de puntos, donde las caras son definidas por las líneas y luego una secuencia de caras crean los polígonos tridimensionales. Como se observa en la figura 6.

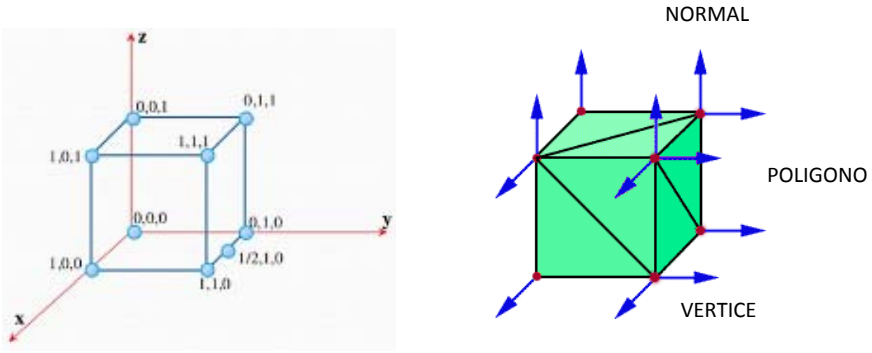


figura 6. Una escena 3D se define por los puntos, líneas y planos que la componen.

Sin embargo, los programas actuales para la generación de gráficos va más lejos de sólo el almacenaje de polígonos en la memoria de la computadora puesto que los gráficos de hoy no solo son el producto de colecciones masivas de polígonos en formas reconocibles, sino también resultan de técnicas como el empleo de texturing (texturizado), shading (sombreadores), y la rasterización (en referencia a mapas de bits).

A continuación se muestran algunos formatos de archivos de los principales programas de modelado y animación.

Formato Archivo 3D	***Extensión***	***Mat***	***Jer***	***U/V***	***CV***	***LC***	***NURBS***	***Anim***	***Hue***	***Pi***
3D Studio R1-R4	.3ds	X	X	X		X		X		
3ds Max	.max	X	X	X	X	X		X	X	X
DirectX	.x	X	X	X	X			X	X	X
FBX	.fbx	X	X	X		X	X	X	X	X
Lightwave	.lwo, .lws	X	X	X	X	X		X		
OpenGL C Code	.c	X		X	X					
X3D	.x3d	X	X	X	X	X	X	X		

dónde:

Mat: Material
Jer: Jerarquía de escena
U/V: Mapeado UV de texturas
CV: Color de vértices de las mallas

L/C: Luces y cámaras
NURBS: Superficies tipo NURBS
Anim: Animación de cuadros
Hue: Esqueletos definidos por huesos o uniones
Pi: Malla manipulada por los huesos.

3.3 Los objetos geométricos.

Típicamente pensamos en un modelo como un objeto en sí, por ejemplo, un árbol, un escritorio, una mesa, una silla etc., y en la escena como el ensamblado de estas partes en un ambiente 3D completo en el que se incluyen las luces y la cámara.

Las maneras de crear un objeto gráfico computacional 3D son tan diversas como los tipos de objetos.

Éstas pueden ser, por ejemplo:

- Modelos poligonales, donde el modelado empieza con una de las primitivas 3ds max, y el uso de herramientas tales como biselado y extrusión.
- Construir un objeto mediante una interfaz 3D, como, por ejemplo, con 3D Studio.
- Tomar datos de un dispositivo como un láser o un digitalizador 3D.
- Nurbs los cuales establecen el grado de influencia de cada vértice de control en la superficie o curva.
- Primitivas estándar son: cono, cajas, esfera, cilindro, tetera, toro, tubo, pirámide y plano.
- Primitivas predefinidas son: prisma, toro nudo, cápsula, esfera, la geosfera, la manguera, etc.

La representación de objetos en una PC, es la interpretación de la matriz de coordenadas del objeto dentro de un espacio virtual, lo cual forma una "figura" tridimensional que el usuario ve en pantalla.

El manejo de objetos se refiere a la forma en que se permitirá manipular la forma de un objeto, lo cual depende de la representación.

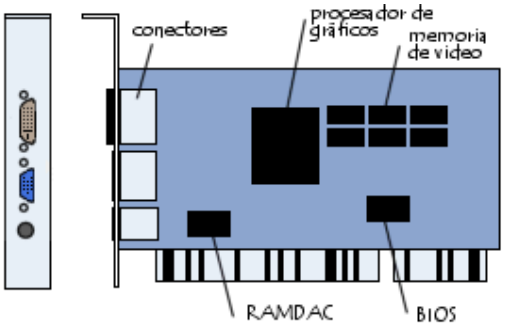
3.4 Procesamiento gráfico 3D.

Una tarjeta de video o tarjeta gráfica es un componente de la computadora que permite transformar los datos digitales en un formato gráfico que puede ser visualizado en una pantalla.

Al inicio la principal tarea de las tarjetas gráficas fue la de enviar píxeles a la pantalla, así como una variedad de manipulaciones gráficas simples como:

- Trazado de polígonos.
- Trazado de rayos.
- Mover bloques.

Las tarjetas gráficas de última generación tienen procesadores fabricados para manejar gráficos complejos en 3D como se muestra en la figura 7.



Componentes de una tarjeta de video.

figura 7.

COMPONENTES DE LA TARJETA DE VIDEO	FUNCIÓN
Unidad de procesamiento Gráfico (GPU)	Es el corazón de la tarjeta de gráficos y procesa las imágenes de acuerdo a la codificación utilizada. La GPU es un procesador especializado con funciones relativamente avanzadas de procesamiento de imágenes, en especial para gráficos 3D.

Memoria de video	Su función es almacenar las imágenes procesadas por el GPU antes de mostrarlas en la pantalla (búfer de trama). A mayor cantidad de memoria de video, mayor será la cantidad de texturas que la tarjeta gráfica podrá controlar cuando muestre gráficos 3D. Las tarjetas de gráficos presentan una dependencia importante del tipo de memoria que utiliza la tarjeta. Su tiempo de respuesta es fundamental en lo que respecta a la rapidez con la que se desea mostrar las imágenes. La capacidad de la memoria también es importante porque afecta el número y la resolución de imágenes que puede almacenarse en el búfer de trama.
Convertidor digital-analógico de RAM (RAMDAC)	Se utiliza a la hora de convertir las imágenes digitales almacenadas en el *búfer de trama* en señales analógicas que son enviadas a la pantalla. La frecuencia del **RAMDAC** determina a su vez la frecuencia de actualización (el número de imágenes por segundo, expresado en Hercios: Hz) que la tarjeta gráfica puede soportar.
BIOS de video	Contiene la configuración de tarjeta gráfica, en especial, los modos gráficos que puede soportar el adaptador.
La interfaz	Este es el tipo de bus que se utiliza para conectar la tarjeta gráfica en la placa madre. El bus AGP está especialmente diseñado para controlar grandes flujos de datos, algo absolutamente necesario para mostrar un video o secuencias en 3D. El bus PCI Express presenta un mejor rendimiento que el bus AGP y en la actualidad, casi puede decirse que lo ha remplazado.

tabla 2.

El desarrollo 3D es bastante nuevo, y cada vez más importante, esto lo podemos observar en algunas PC's, ya que estas ya cuentan con más poder de procesamiento que algunas estaciones de trabajo.

En términos generales, el procesamiento de gráficos en 3D es un proceso que se divide en cuatro etapas:

- Secuencia de comandos: presentación de elementos.
- Geometría: Creación de objetos simples.
- Configuración: transformación de los objetos a triángulos 2D.
- Renderizado: aplicación de textura a los triángulos.

Mientras más rápido la tarjeta aceleradora 3D pueda procesar estos pasos por sí sola, la velocidad que se muestre en pantalla será mayor. En un inicio, los chips sólo podían renderizar y le dejaban el resto de la tarea al procesador. Por este motivo las tarjetas gráficas suelen incluir un "setup engine", el cual permite controlar la configuración y renderizado.

Por ejemplo, un procesador Pentium II de 266 MHz que procesa la secuencia de comandos, geometría y configuración, procesa 350.000 polígonos por segundo; cuando procesa tan sólo dos, puede llegar a procesar hasta 750.000 polígonos por segundo. Esto demuestra cuánta es la carga que las tarjetas gráficas aminoran en los procesadores.

El tipo de bus también es un factor importante. Aunque el bus AGP no mejora las imágenes 2D, las tarjetas que emplean ese bus tiene un mejor rendimiento. Esto se debe a que el bus AGP está conectado directamente a la memoria RAM, lo que le da a su vez un ancho de banda mayor al del bus PCI.

PCI express o PCIe es un bus que trabaja en serie, donde PCIe 1.1 para cada carril lleva 250 MB/s en cada dirección, PCIe 2.0 dobla esta tasa a 500 MB/s y PCIe 3.0 la dobla de nuevo. Cada ranura de expansión lleva uno, dos, cuatro, ocho o dieciséis carriles de datos entre la placa base y las tarjetas conectadas donde el número de carriles se escribe con una x de prefijo x1 para un carril simple y x16 para una tarjeta con dieciséis carriles.

En comparación con otros buses, un carril simple es aproximadamente el doble de rápido que el PCI normal. PCI-Express está pensado para ser usado sólo como bus local ya que su velocidad superior permitirá reemplazar casi todos los demás buses, AGP y PCI incluidos.

En el presente, estos productos de alta tecnología necesitan ser fabricados con la misma calidad que los procesadores, como un ancho de canal de entre 0.25 µm y 0.35 µm.

Capítulo IV

Tecnologías de ambientes virtuales

4.1 Software de diseño 3D.

Existe una gran variedad de software de modelado que el diseñador puede usar para la creación de los elementos del entorno, entre los más importantes podemos nombrar por ejemplo: Maya, Autocad, Lightware, 3D Studio Max, etc. Para el desarrollo del paseo virtual se usó la herramienta de modelado tridimensional conocida como 3D Studio Max, de la compañía Autodesk. 3D Studio Max es el programa de render, modelado y animación en 3D preferido por los creadores de videojuegos y los desarrolladores de todo tipo de proyectos de animación, publicidad, efectos especiales y arquitectura. Ofrece gran capacidad para crear cualquier cosa existente en la vida real. Su potencia y posibilidades de uso son inmensas.

4.2 Modelado e iluminación por computadora.

Modelado de escenas 3d.

Geometría de los Objetos 3D.

La representación poligonal es el modelado dominante para la generación de gráficos en 3D. Un objeto se representa por una ***red poligonal***, que es la representación aproximada de una superficie de un objeto. Como se muestra en la figura 8.

Modelo a base de polígonos.

figura 8.

Una red poligonal tiene varias componentes:

- Caras.
- Aristas.

- Vértices.
- Normales.
- El elemento en sí mismo.

Algunos componentes pueden ser determinados, por ejemplo, se pueden determinar los lados dados los vértices y las aristas:

Teorema de Euler:

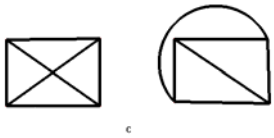


figura 9.

En todo grafo conexo y plano que esté apropiadamente representado se verifica que el número de caras más el de vértices menos el de aristas vale 2. Es decir

Un conjunto de polígonos se denominan redes o *meshes* (por su nombre en idioma Inglés). Estas caras pueden formar objetos poligonales abiertos o cerrados. Objetos cerrados son redes que encierran un espacio dentro del objeto en sí mismo. Objetos abiertos son redes que no encierran ningún espacio por lo que son solo una superficie. Como se muestra en la figura 10.

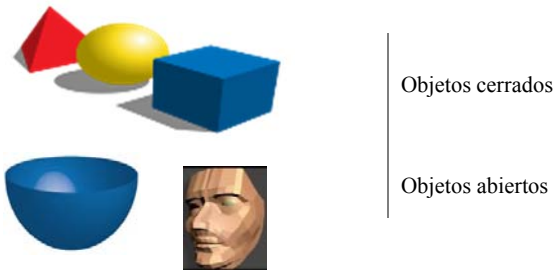


figura 10.

Representación de polígonos.

La representación de la red poligonal se realiza mediante operaciones típicas como:

- Encontrar todos los lados incidentes en un vértice.
- Encontrar todos los polígonos que comparten un lado o un vértice.
- Encontrar los vértices conectados por un lado.
- Encontrar los lados de un polígono.
- Mostrar la red.

Si se mantiene una estructura simple y muy definida, se hará más rápida la representación de la misma, pero se aumenta el tamaño de la misma.

Una representación explicita de la estructura se da por:

$$P = ((x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_n, y_n, z_n))$$

Los vértices son almacenados conforme al sentido en que se haga la detección de los mismos.

(3,-2,5), (3,6,2), (-6,2,4)
(2,2,4), (0,-1,-2), (9,4,0), (4,2,9)
(1,2,-2), (8,8,7), (-4,-5,1)
(-8,2,7), (-2,3,9), (1,2,-7)

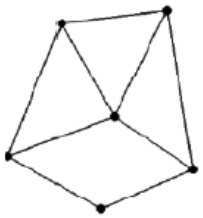


figura 11.

Pero, para una red poligonal es muy complejo hacer este tipo de detección, ya que el procesamiento es muy exhaustivo y no se puede asegurar que los vértices estén conectados entre sí o traslapados.

La solución a este problema es poner todos los vértices en una tabla única de cada cara de la red poligonal, logrando hacer una lista de vértices, donde cada vértice se almacena solo una vez.

$$V = ((x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_n, y_n, z_n))$$

Cada polígono se define como una lista de índices a la lista de vértices.

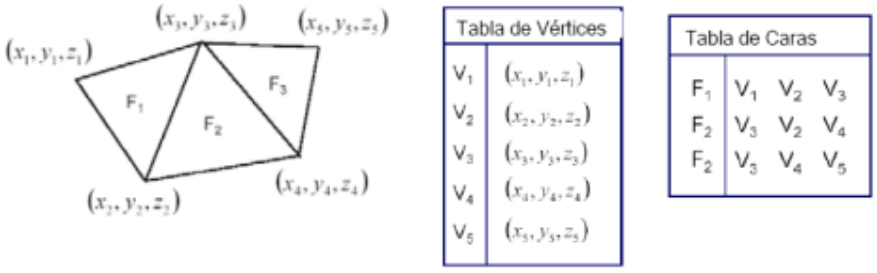
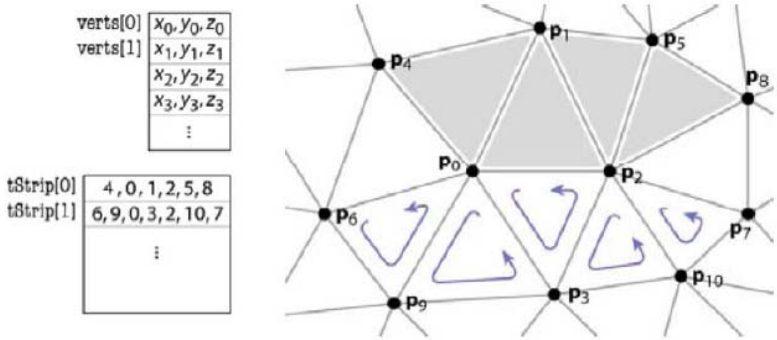


figura 12.

La única desventaja de esta representación es que no soporta el acceso directo a los triángulos vecinos, para esto se utilizan otros mecanismos de representación denominados Strips de triángulos y Fans de triángulos, en estas nuevas representaciones cada triángulo es usualmente adyacente al previo, cada nuevo vértice (excepto los primeros) crea un triángulo reusando el segundo y tercer vértice del triángulo previo y cada secuencia de tres vértices produce un triángulo. Como se muestra en la figura 13.



Strips de triángulos.

figura 13.

Las Normales.

Las normales asociadas a cada vértice, son parte importante de los objetos, ya que con ellas será

posible el cálculo de la iluminación. Como se muestra en la figura 14.

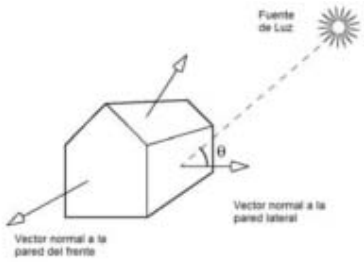


figura 14.

Los vértices *V*1, *V*2, *V*3 y *V*4 definen la pared lateral y tienen la misma normal **n**1. Vértices de la pared frontal, tal como *V*5, usarán la normal **n**2. (*V*1 y *V*5 están ubicados en el mismo punto en el espacio, pero usan normales diferentes). Como se muestra en la figura 15.

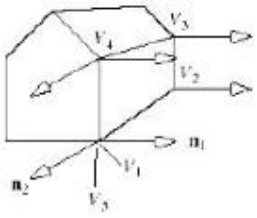


figura 15.

Para la superficie cilíndrica, tanto el vértice *V1* de la cara *F1* como el *V2* de la cara *F2* usan la misma normal ***n***, que es el vector perpendicular a la superficie suave subyacente. Como se muestra en la figura 16.

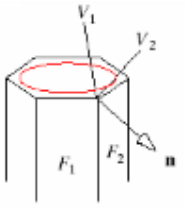


figura 16.

Iluminación en escenas 3d.

La iluminación es una parte importante de la red poligonal ya que con ella nos ayuda a dar profundidad y perspectiva al objeto, logrando así, darle realismo al objeto o la escena en sí misma.

La figura 17 es un ejemplo del manejo de la luz dentro del ambiente 3D puede dar un sentido de realismo, profundidad y perspectiva del mismo entorno.

Ejemplo del libro de iluminación en Strata 3D

figura 17.

Las fuentes de iluminación que tenemos son tres:

- Desde un punto: Este tipo de fuente emite rayos de luz en todas direcciones. Lo que quiere decir es que la información que contiene cada rayo es la de dirección y posición.
- Direccional: En este tipo de fuente se simula la iluminación solar, donde la información que contiene cada rayo solo es de dirección, haciendo de esto que los rayos sean paralelos entre sí.

- Spot: Este tipo de fuente funciona como el foco de una lámpara, donde el ángulo de la luz sigue un patrón en cono.

Para propósitos de nuestra tesis, se empleó la luz desde un punto, ya que ocupa menos recursos al momento de hacer el Render y es mejor para generar las sombras que necesitamos.

Cada tipo de fuente de iluminación consta de tres componentes:

- Ambiental: Determina la reflexión que la Luz tendrá sobre las superficies que tengan propiedades difusas. Establece el nivel de iluminación de la escena ante cualquier tipo de luz.

- Difusa: Cuando incide la luz sobre el objeto, ésta se refleja de igual forma en todas direcciones, sirviendo para calcular el nivel de brillo del objeto. Esta tipo de iluminación no depende de la posición del espectador. Cuando el objeto está en movimiento respecto a la fuente de la luz, ésta se mueve a través de los objetos.

- Especular: Esta depende de la posición del espectador, la dirección de la luz y la orientación del polígono en el que se refleja. Este tipo de iluminación nos permite aplicar efectos en la reflexión de la luz. El efecto de movimiento solo es posible cuando se aplica desplazamiento y reflejo para dar apariencia a los distintos objetos. Como se muestra en las figuras 18 y 19.

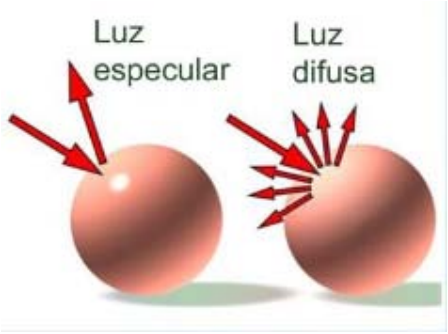


figura 18.

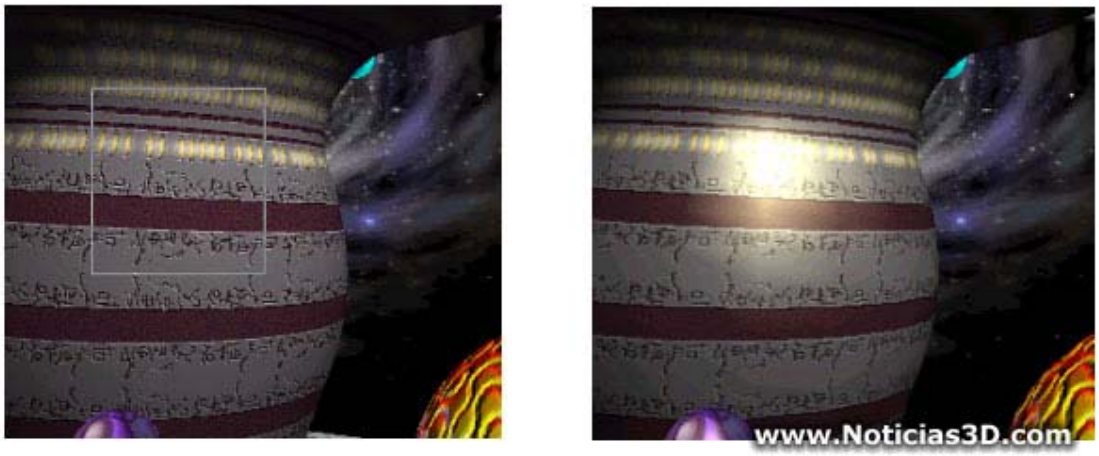


figura 19. Ejemplos de luz difusa (Arriba Izq.) y especular (Arriba Der.)

El procesamiento para la iluminación de objetos es muy demandante para la Unidad Procesamiento Gráfica (GPU por sus siglas en Inglés), y ésta la hace a través de operaciones de vectores a través de modelos de iluminación.

Uno de los modelos de iluminación que ayuda a dar esta perspectiva, y que es más común usarlo para escenarios grandes, el cual fue nuestro caso, es el Modelo de Blinn-Phong.

Este modelo procesa la intensidad de reflexión como una función de la normal de la superficie local $\vec{n}$, la dirección hacia el punto de luz $\vec{l}$, la dirección de reflexión $\vec{h}$, la intensidad del punto de luz I_L , el coeficiente de brillo ambiental k_a , el coeficiente de brillo difuso k_d , el coeficiente de brillo especular k_s y n:

$$I = k_a + I_L k_d(\vec{l} \cdot \vec{n}) + I_L k_s(\vec{h} \cdot \vec{n})^n$$

Sombras en escenas 3D.

Hoy en día, el realismo que se tiene que poner en la realidad virtual es muy extensa, ya que una buena iluminación te llevará a la generación de un buen sombreado de los elementos tridimensionales, dándole una perspectiva estratégica que jamás se hubiera experimentado en la vida real, dándole muchos créditos a los efectos cinematográficos.

Las sombras no son objetos renderizados; en vez de eso, son áreas de la pantalla que reciben menos luz debido al procesamiento de la luz dentro de la escena.
Una de las técnicas de sombrado consiste en dos partes, se representa el objeto tridimensional en un polígono de superficie plana y son renderizados en un buffer plano. Esto se repite para cada fuente de luz, y las imágenes resultantes son adheridas entre ellas para crear un imagen final, este proceso es llamada Renderizado Multipase.

Para generar la sombra del personaje principal, se elaboró una elipsoide de pocos polígonos que cubre en parte al personaje, y se le aplicó la técnica de sombreado al elipsoide y no al personaje principal.
La característica de esta elipsoide, es que su canal alfa es 0 y su matriz de movimiento es igual a la matriz de movimiento de personaje. Como se muestra en la figura 20.

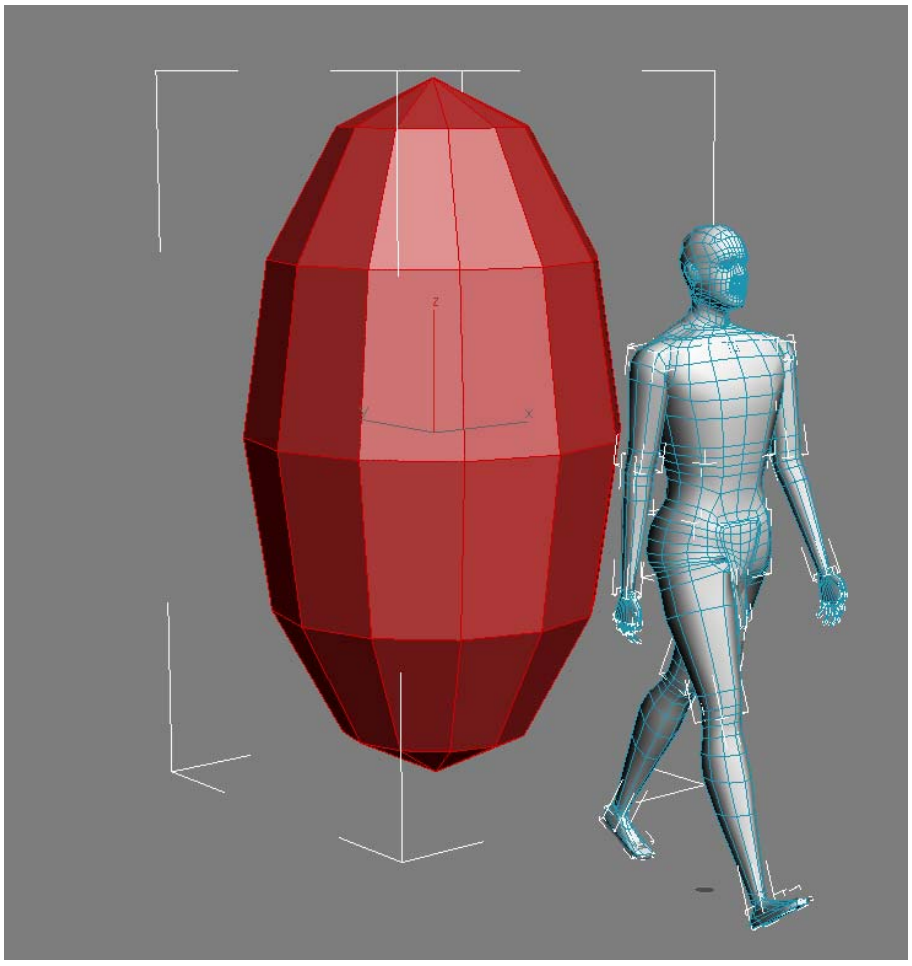


figura 20.

4.3 Texturas.

El objeto 3D cuando tienen la necesidad de representar algún objeto del mundo real, le es necesario crearle imágenes con textura que simule el objeto en cuestión.

La textura es una imagen que se genera para simular el objeto que deseamos representar en el mundo 3D. Está textura se aplica a diferentes polígonos de la red poligonal de una manera conveniente. La técnica de aplicación es similar a la proyección de mapas, por lo que se denomina "mapeo".

La técnica usada para fines de esta tesis fue la de Mapeo UV (UV Mapping), la cual se explicará de forma concisa a continuación.

UV Mapping

Se usan *U* y *V* para definir la posición del objeto transformado, pero una tercer coordenada *W* se añade cuando se trata de representar un textura procedural. Las tres coordenadas U, V y W representan lo siguiente, U es el ancho de la textura, V es el largo y, W es la profundidad.

Cuando aplicamos mapas UV, las coordenadas son medibles en una escala del 0.0 al 0.1, con 0.0 y 0.1 en lugares opuestos de la textura. Números grandes que 1.0 harán que la textura se repita, y números negativos harán que a la textura se le aplique un espe jo. Como se muestra en la figura 21.

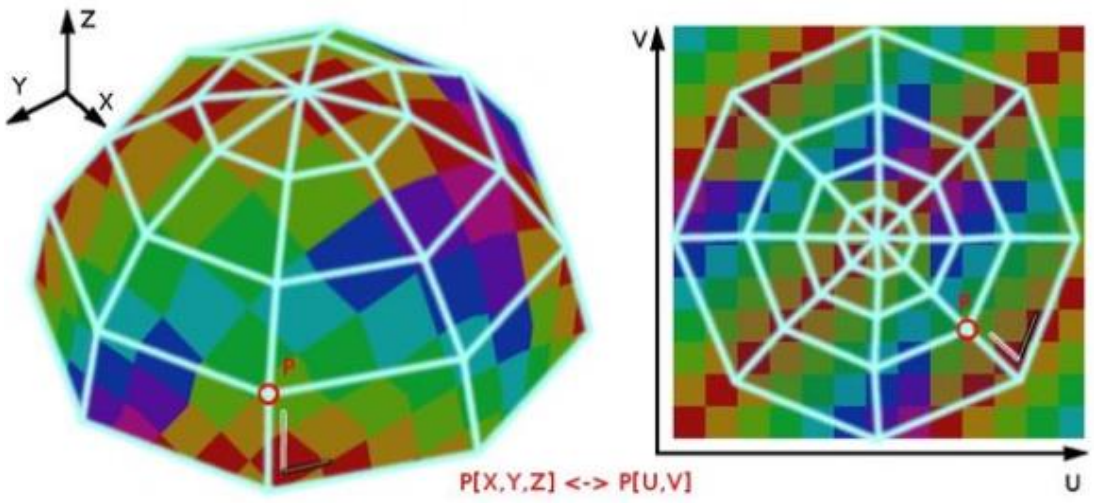


figura 21. Es un ejemplo de cómo se puede asignar un punto de un objeto 3D en una imagen 2D.

Cuando un modelo es creado como un polígono editable, las coordenadas UV pueden ser generadas para cada vértice en el polígono, logrando con ello extender el triángulo de cada polígono en un mapa plano.

Las coordenadas UV son aplicadas por caras, no por vértices. Esto significa que un vértice compartido puede tener diferentes coordenadas UV en cada uno de sus triángulos, así los triángulos adyacentes puede ser cortados aparte y posicionados en diferentes áreas de la textura.

El método UV mapping usa el método Least Squares Conformal Map (LSCM – Mapa conformal de mínimos cuadrados), el cuál crea una representación en 2D del objeto en 3D. Este método nos permite obtener un mapa en 2D con coordenadas UV del objeto que se encuentra en el espacio 3D, haciendo que éste último se desenvuelva de tal forma que genere un mapa el cual podrá ser texturizado.

Capítulo V

Desarrollo del proyecto:

"ANIMACIÓN Y RECONSTRUCCIÓN SOBRE LA ZONA PREHISPÁNICA DE MONTE ALBÁN, OAXACA"

5.1 Introducción

En este apartado se explica cómo se diseñó, modeló y desarrolló el entorno virtual arquitectónico de la zona arqueológica, los pasos que se siguieron desde la selección del sitio hasta el modelado de las estructuras. La selección de las herramientas que se utilizaron y las consideraciones del hardware necesario para lograr un buen desempeño del sistema.

5.2 Análisis de requerimientos del sistema

En vista de que se desea crear un entorno virtual que realce la grandeza de una ciudad precolombina en pleno apogeo, se torna una exigencia trabajar con cierto nivel de detalle en los exteriores, específicamente en fachadas, pisos y terrazas, sin dejar de lado los elementos ambientales que conforman el mundo exterior como lo son el paisaje y la atmósfera. Por lo tanto, y considerando que los resultados obtenidos deben satisfacer las expectativas del público, el proyecto debe cumplir con las siguientes características:

- Alto nivel de detalle en las estructuras arquitectónicas.
- Alto nivel de realismo en cuanto a dimensión espacial.
- Buena calidad en las texturas.
- Buen manejo de luces y sombras.
- Buenos efectos de sonido.
- Interacción con el entorno.
- Excelente desempeño en computadoras personales.

Está por demás decir que no existe en el mercado una herramienta con la cual podamos cubrir todas estas necesidades al mismo tiempo, por lo tanto se utilizará una serie de ellas, las cuales fueron seleccionas en base a su buen desempeño, específico a cada etapa de desarrollo del proyecto, estas son: 3D Studio Max, Photoshop, FaceGen, Total Texturas, Sketch Up y Quest 3D.

5.3 Reconstrucción virtual

5.3.1 Selección de la zona arqueológica

En esta etapa se selecciona el entorno virtual a crear. Para esta tesis, desde un principio la idea central fue trabajar con una zona arqueológica del país, en vista de que uno de los objetivos clave del proyecto es apoyar mediante este tipo de tecnología virtual los campos tanto educativo, de investigación y turístico.

Por lo tanto, después de analizar detenidamente varios lugares arqueológicos nacionales, optamos por la zona de Monte Albán, ubicada en el estado sureño de Oaxaca. Los factores que se consideraron para la selección fueron: importancia del lugar, viabilidad del proyecto, accesibilidad, que fuese un territorio inexplorado aún en el tema y sobre todo, que tuviese construcciones de gran belleza y relevancia que mostrar.

Todos estos requisitos los reunió Monte Albán:

Fue la ciudad capital del imperio zapoteca que floreció en el sureste mexicano. Eso sin duda le da una importancia indiscutible.

Al existir una maqueta muy bien planificada en el Museo Nacional de Antropología e Historia de la Ciudad de México se supuso que debía existir información muy bien documentada sobre el sitio, inclusive planos arquitectónicos de escala real, por lo tanto gracias a un excelente trabajo de investigación se llegó a la conclusión de que el proyecto era viable de desarrollarse.

Actualmente la ciudad de Oaxaca, capital del estado del mismo nombre cuenta con una autopista que la conecta directamente con la ciudad de Puebla y, por consiguiente, con la Ciudad de México, la zona arqueológica de Monte Albán se halla a tan solo 20 minutos de la capital oaxaqueña, por lo tanto es un lugar muy accesible. Esto fue tomado en cuenta para las tareas de campo.

Al realizar un recuento sobre los proyectos similares que se habían realizado o que se desarrollaban a la par se descubrió que la mayoría de ellos estaban enfocados a lugares de civilizaciones más conocidas como los aztecas o los mayas, por lo tanto se vio como una buena oportunidad

incursionar en el imperio de los zapotecas, tanto por su importancia como para ampliar la gama de proyectos de este tipo.

Finalmente el último requisito, el de la belleza urbana, es innegable en Monte Albán, según lo describe el siguiente texto:

> *"Los benizáa (señores guerreros), habían planeado hacer una ciudad majestuosa, comparada sólo con Teotihuacán, la gran ciudad de las tierras altas desde donde llegaban las influencias religiosas, políticas y artísticas.*
>
> *Así, sus templos más grandes y bellos fueron diseñados sobre grandes plataformas piramidales, teniendo todas al centro una gran escalinata para el uso exclusivo de los sacerdotes y de aquellos personajes que conducirían las ceremonias. Dichos templos eran edificios de piedra que se componían de un patio cuadrado y cerrado al centro, rodeado por tres o cuatro habitaciones.*
>
> *Los templos estaban dedicados a las diferentes deidades y en ellos se llevaban a cabo ceremonias muy concurridas para honrar a un dios, o muy exclusivas en que los sacerdotes decidían los destinos de la ciudad solos, con la única presencia de sus dioses. Otros templos estaban dedicados a los ritos de iniciación y a los matrimonios.*
>
> *Algunos de estos edificios eran muy complejos, pues en realidad se trataba de conjuntos arquitectónicos integrados por una gran plataforma con un templo en su parte superior, un patio cerrado en la base y, adosado a éste, un adoratorio donde se colocaban las ofrendas.*
>
> *Para ornamentar la arquitectura se adoptó el diseño del tablero teotihuacano, que consistía en un marco alargado que iba a los lados de las escalinatas centrales, al límite de los muros en talud que las enmarcaban; pero luego los artistas zapotecos concibieron un tablero más complejo que el teotihuacano: lo hicieron doble, sobreponiendo dos cornisas al mismo muro vertical. (**Fuente:** Pasar a Bibliografía. Cita Num. 1)*

5.3.2 **Creación de la zona arqueológica**

Esta segunda etapa es crucial, ya que consiste en la creación de todos los objetos y elementos que conforman el paseo virtual y que en su mayoría son los componentes de las construcciones tales como paredes, pisos, escalinatas, fachadas, etc., y elementos del paisaje como la geografía del

terreno y partes de la maleza. Es en esta fase donde el proyecto empieza a tener forma, las tareas a desarrollar principalmente son modelado y texturización.

Existe una gran variedad de software de modelado que el diseñador puede usar para la creación de los elementos del entorno, entre los más importantes podemos nombrar por ejemplo: Maya, Autocad, Lightware, 3D Studio Max, etc. Para el desarrollo del paseo virtual se usó la herramienta de modelado tridimensional conocida como 3D Studio Max, de la compañía Autodesk. 3D Studio Max es el programa de render, modelado y animación en 3D preferido por los creadores de videojuegos y los desarrolladores de todo tipo de proyectos de animación, publicidad, efectos especiales y arquitectura. Ofrece gran capacidad para crear cualquier cosa existente en la vida real. Su potencia y posibilidades de uso son inmensas.

Trabajando con 3ds Max

- *Modelado*

La Gran Plaza de Monte Albán fue diseñada en base al plano que se muestra en la figura 22, el cual registra 12 edificaciones menores o independientes, un altar o adoratorio y dos grandes conjuntos urbanos denominados Plataformas Norte y Sur respectivamente.

aymond Ostertag

Figura. 22. *Izquierda*: Plano del conjunto arquitectónico de la Gran Plaza de Monte Albán. *Derecha*: Panorámica de la zona arqueológica en la actualidad.

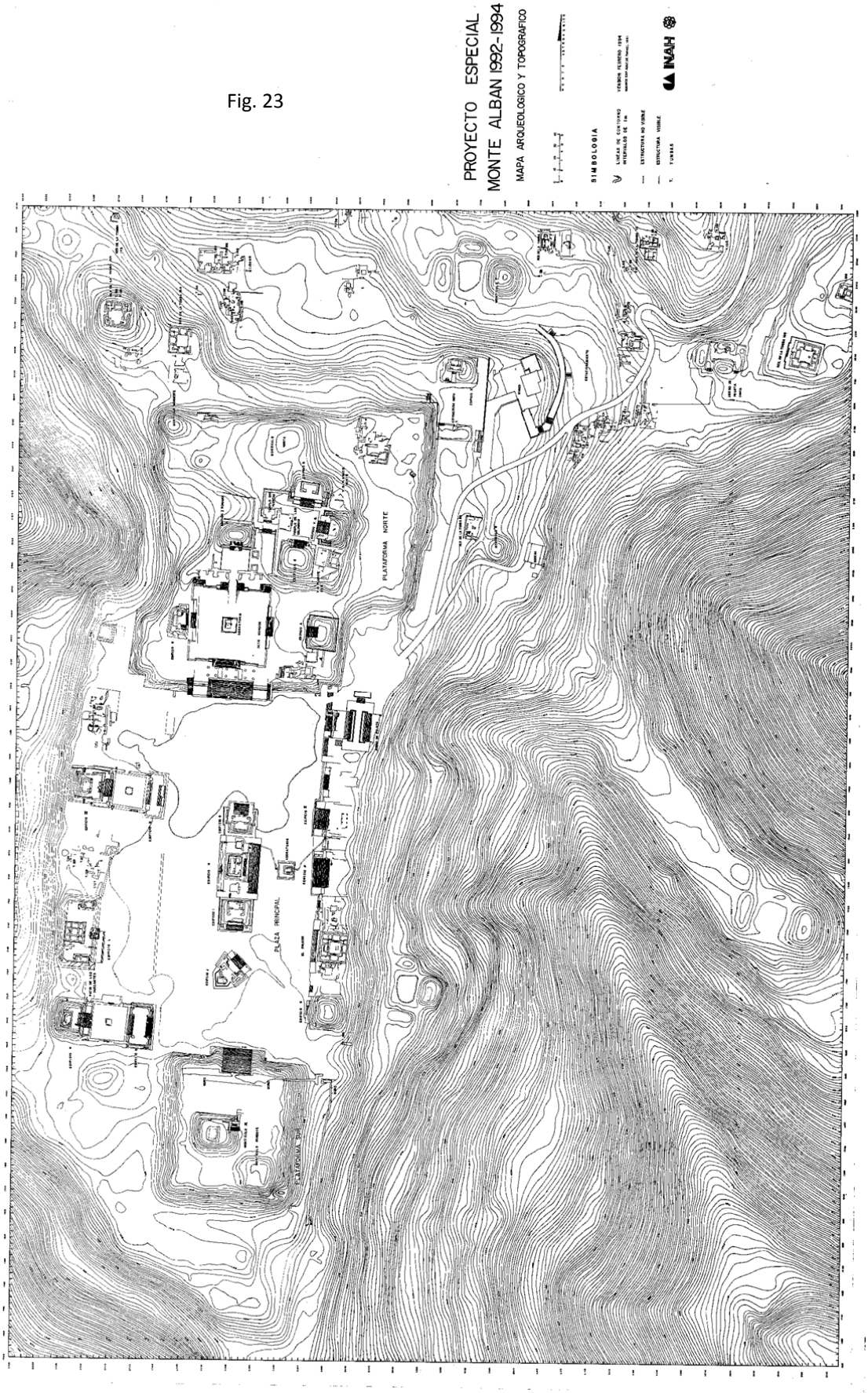
Todas las construcciones presentes en el paseo virtual fueron creadas en escala real, con la ayuda de planos arqueológico – arquitectónicos,

Estos planos fueron obtenidos directamente del Museo de Antropología e Historia, con la finalidad de hacer un proyecto con dimensiones lo más aproximadas posibles.

Dentro de la serie de los planos, podemos encontrar la siguiente imagen la cual muestra la magnitud y escala de la zona que se quiere presentar en un entorno virtual tridimensional. El plano que contiene dicho archivo es de 9314x5758 píxeles, y cada unidad mínima de regla del plano equivale a 5 metros en la vida real, lo cual lo podemos traducir que cada 40 píxeles de la imagen del plano equivale a 5 metros en la realidad. Por lo tanto podemos decir que cada píxel de la imagen equivale a 125 milímetros en la realidad; para fines de esta tesis, la escala será 125:1

La imagen que se muestra a continuación tiene un formato muy grande como para caber en esta hoja, por tal motivo, se mostrará en la siguiente hoja:

Fig. 23


PROYECTO ESPECIAL
MONTE ALBAN 1992-1994
MAPA ARQUEOLOGICO Y TOPOGRAFICO
SIMBOLOGIA
INAH
PLAZA PRINCIPAL
PLATAFORMA NORTE

Para obtener las medias tanto de alto ancho y largo de cada uno de los edificios, y la escala del plano anterior, primero utilizábamos los planos que tenemos y que están a escala, de los cuales cada uno de ellos los exportábamos a Microsoft Paint y ahí tomábamos las medidas en píxeles como se muestra en las siguientes figuras.

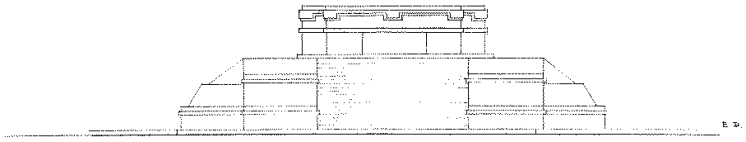


Figura. 24

Aquí después de importar el plano a cada una de sus líneas obteníamos sus medidas en píxeles. La obtención de estos últimos se elaboraban revisando los atributos de imagen dentro de Microsoft Paint:

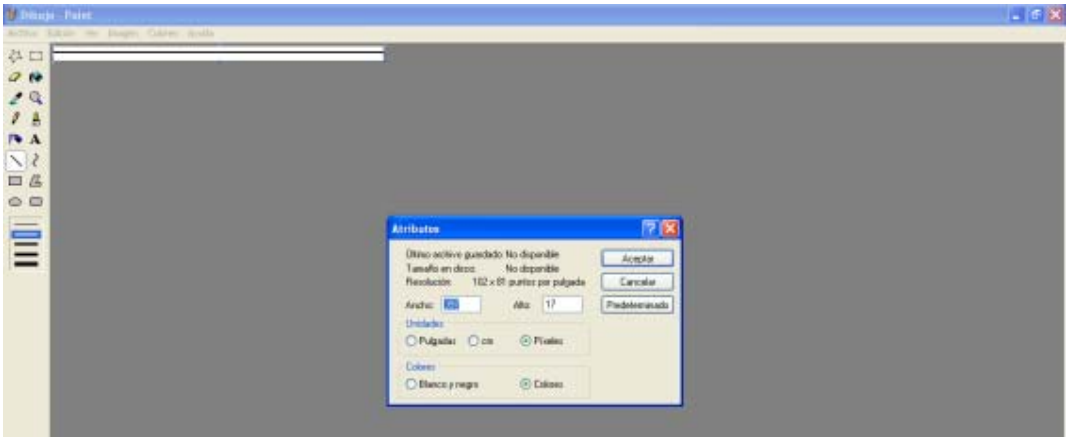


Figura. 25

Para posteriormente hacer el mismo procedimiento con todos los planos y así obtener todas las medidas en píxeles que necesitábamos para crear las estructuras en 3DMax con medidas en pixeles

El modelado consiste en ir dando forma a objetos individuales que luego serán usados en la escena virtual. En esta etapa podemos modelar a través de diferentes técnicas que existen dentro del software 3D Studio Max, las cuales son:

- Geometría Solida Constructiva: Modelo de representación de objetos 3D en forma de árbol basado en dos elementos: a) Primitivas gráficas (esfera, cubo, cilindro, etc.) b) Operaciones booleanas: Operaciones de conjuntos (unión, intersección y diferencia son las más importantes)

- Patch Grids: Una patch es un tipo de objeto deformable. Un objeto patch es útil para crear delicadas superficies curveadas, y provee de un muy detallado control par manipular geometría compleja.

- Editable Mesh: Una malla editable es un tipo de objeto deformable. Una malla editable es una trimalla: eso es, que usa polígonos triangulares. Las mallas editables son usadas para objetos de pocos polígonos o para una malla suavizada.

- Editable Poly: Un polígono editable es un tipo de objeto deformable. Un polígono editable es una malla poligonal; eso es, diferente a una malla editable, ya que usa polígonos de más de tres lados. Un polígono editable es un objeto con cinco niveles de sub-objetos: vértices, aristas, bordes, polígonos y elemento.

- NURBS (Non-Uniform Rational B-Splines): Son creados específicamente para modelado en 3D asistido por computadora. Son especialmente aplicadas a superficies con complicadas curvas. NURBS son populares porque son muy fáciles de manipular y, porque el algoritmo para crearlos son ambos eficientes y numéricamente estables. Específicamente:

    a) Non-Uniform significa que el alcance del control de influencias de vértices puede variar. Esto es útil cuando se modela superficies irregulares.

    b) Rational: Significa que la ecuación usada para representar la curva o superficie es expresada como un radio de dos polinomios, más que una simple suma de polinomios. c) A B-spline (basis spline): Es una manera de construir una curva que es interpolada entre tres o más.

- Subdivision Surfaces: Se crea una subdivisión de superficie aplicando un modificador a un objeto. Dos tipos de subdivisión de superficies son soportadas:

    a) El modificador HSDS provee una subdivisión de superficies jerárquica

    b) El modificador MeshSmooth provee suavizado.

- Low-Polygon Modeling: En el Modelado de baja poligonización, se cuida especialmente la economía en cuanto a número de polígonos del personaje para aliviar al procesador gráfico y poder representar una acción más fluida sin que se sature o ralentice el sistema.

La técnica que se emplerá para la esquematización de la geometría será la de polígonos editables y la de modelado de baja poligonización, ya que para crear un paseo virtual en tiempo real, necesitamos hacer nuestros modelos tridimensionales lo más bajos en polígonos, para que el procesador puede calcular de una forma rápida las ecuaciones que forman parte de nuestro entorno virtual.

Para poder manipular nuestro polígono editable en cada uno de sus niveles de sub-objetos, tendremos que hacer uso de transformaciones geométricas en cada eje coordenado del objeto, las cuales son trasladar, rotar y escalar. Estas tres transformaciones geométricas se explican a continuación de una manera rápida y concisa.

1. *Traslación*: es el movimiento de un punto desde la posición $P_1=(x_1,y_1,z_1)$ a la posición $P_2=(x_2,y_2,z_2)$. La operación se representa mediante la siguiente matriz:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

donde t_x, t_y, t_z corresponden a la distancia de traslación respecto de los ejes x, y, z; podemos decir también que:

$x_2 = x_1 + t_x$ $\qquad$ $y_2 = y_1 + t_y$ $\qquad$ $z_2 = z_1 + t_z$

Esta forma de traslación es válida para todos los puntos del objeto en cuestión. Por lo tanto, en un ambiente tridimensional, un objeto se traslada al trasladar cada uno de los puntos del objeto.

Para un objeto que se representa como un conjunto de superficies poligonales, trasladamos cada vértice de cada superficie y volvemos a trazar las facetas del polígono en la nueva posición.

2. *Rotación*: Para girar un objeto se debe designar un eje de rotación y el valor de la rotación. Si se gira con respecto al eje z (el cual vemos como un punto en el origen de los sistemas coordenados X_1, Y_1; X_2, Y_2), el plano de giro será el plano XY, tal y como se muestra en la siguiente figura.

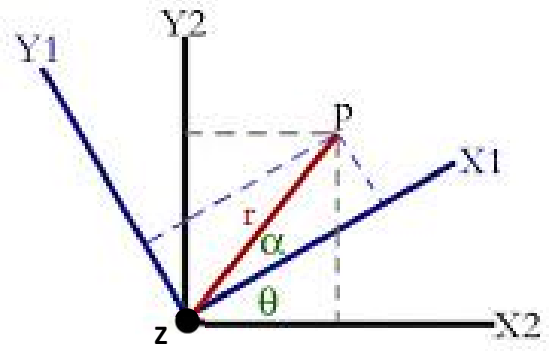


Figura 26. En esta figura se muestra el angulo total de giro, que es la suma de los ángulos Θ y α.

De la imagen:

$$Y_1 = r \text{ sen } \alpha$$

$$X_1 = r \cos \alpha$$

$$Y_2 = r \operatorname{sen}(\alpha+\Theta)$$

$$X_2 = r \cos(\alpha+\Theta)$$

Pero:

$$\operatorname{Sen}(\alpha+\Theta) = \operatorname{sen}\alpha \cos\theta + \operatorname{sen}\theta \cos\alpha \quad .....(1)$$

$$\operatorname{Cos}(\alpha+\Theta) = \cos\alpha \cos\theta - \operatorname{sen}\alpha \operatorname{sen}\theta \quad .....(2)$$

Sustituyendo las ecuaciones 1 y 2 y los valores de X1 y Y1 en las ecuaciones X2, Y2, tenemos:

$$X_2 = X_1 \cos\Theta - Y_1 \operatorname{sen}\Theta$$

$$Y_2 = Y_1 \cos\Theta + X_1 \operatorname{sen}\Theta$$

$$Z_2 = Z_2$$

Escribiendo las ecuaciones anteriores en forma matricial, obtenemos:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -sen\theta & 0 & 0 \\ sen\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

donde la rotación, con respecto al eje z, es en Θ grados.

La matriz se puede adaptar para los giros en los otros ejes X, Y . A continuación se muestra las Matrices de rotación de los ejes X, Y y Z, partiendo del procedimeinto anterior:
Matriz operacional de rotación sobre el eje de las X:

$$X_2 = X_1$$

$$Y_2 = Y_1 \cos\Theta - Z_1 \operatorname{sen}\Theta$$

$$Z_2 = Y_1 \operatorname{sen}\Theta + Z_1 \cos\Theta$$

En forma matricial, obtenemos:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -sen\theta & 0 \\ 0 & sen\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

Matriz operacional de rotación sobre el eje de las Y:

$$X_2 = Z_1 sen\Theta + X_1 cos\Theta$$

$$Y_2 = Y_1$$

$$Z_2 = Z_1 cos\Theta - X_1 sen\Theta$$

En forma matricial, obtenemos:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & sen\theta & 0 \\ 0 & 1 & 0 & 0 \\ -sen\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

Matriz operacional de rotación sobre el eje de las Z:

$$X_2 = X_1 cos\Theta - Y_1 sen\Theta$$

$$Y_2 = Y_1 cos\Theta + X_1 sen\Theta$$

$$Z_2 = Z_2$$

En forma matricial, obtenemos:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -sen\theta & 0 & 0 \\ sen\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

3. *Escalación*: Es el cambio de tamaño del objeto en sus tres ejes coordenados respecto al origen del eje de coordenadas y lo posiciona nuevamente respecto a él. Los parámetros de escalación son valores positivos cualesquiera. Los parámetros de escalamiento S_X , S_Y , S_Z se muestran a continuación:

$$X_2 = X_1 * S_X$$

$$Y_2 = Y_1 * S_Y$$

$$Z_2 = Z_1 * S_Z$$

La matriz de escalación de una posición P(X1 , Y1 , Z1) respecto a un eje coordenado se repereseta por:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} S_X & 0 & 0 & 0 \\ 0 & S_Y & 0 & 0 \\ 0 & 0 & S_Z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

Con ayuda de las técnicas de modelado y las transformaciones geométricas podremos empezar a hacer nuestros personajes que serán empleados en el paseo virtual de Monte Albán.

Cabe resaltar que cuando la computadora hace operaciones con los polígonos del objeto tridimensional, los hace con el polígono más bajo, el cual es el triángulo, ya que el triángulo es el polígono más sencillo con el cual puede efectuar todas las operaciones. Cuando se está modelando, para nosotros es muy sencillo manejar polígonos cuadrados, pero a la par, la computadora efectúa operaciones sobre los dos triángulos que conforman al cuadrado.

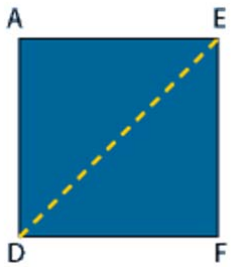


Figura 27. Se muestra como se divide el cuadrado formado por los vértices ADEF en el polígono más sencillo, que es los triángulos formados por los vértices ADE y DEF

5.3.3 **Diseño y Creación del personaje guía**

Para poder crear un personaje en 3D, hemos hecho 4 pasos para poder lograr esta tarea:

- *Blueprints:* Creación de ayudas visuales del personaje de cuerpo completo, incluyendo diferentes vistas del mismo, como vista frontal, lateral, encima, ortogonal, etc.
- *Modelado:* Generación del personaje en 3D a partir de los blueprints con ayuda de herramientas de modelado en 3D.
- *Texturización:* Creación de imágenes que simulará la ropa del cuerpo del personaje, y serán mapeadas en el modelo hecho en 3D.
- *Animación:* Generación de los movimientos que ejecutará el personaje en un ambiente virtual.

Estos cuatro pasos son lo que a continuación se presentarán de una manera clara y concisa respecto a su forma de desarrollo.

GENERACIÓN DE BLUEPRINTS

El blueprint la representación gráfica de un objeto, donde se muestra de forma detallada su diseño en diferentes perspectivas, como lo son las vistas frontal, lateral, isométrica, etc.

Para la elaboración del personaje principal se han generado dos perspectivas del mismo, una vista frontal y una vista lateral de su cuerpo completo, pero se incluyó una vista desde arriba de su brazo, como se muestra en la siguiente figura.

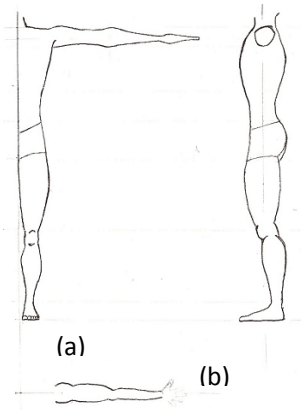


Figura 28. Perspectivas del personaje, (a) frontal, (b) lateral, (c) vista desde arriba del brazo.

Para la elaboración de los personajes secundarios, se usó la figura 28 para poder crearlos, pero para la cabeza de estos personajes, se creó a partir del blueprint que muestra la siguiente figura.

Figura 29. Blueprint de la cabeza en vistas frontal y lateral

Las manos del personaje principal y los personajes secundarios, se hicieron a partir del blueprint mostrado a continuación.

Figura 30. (a) Manos del personaje secundario. (b) Manos del personaje principal

Modelado personaje principal

Todos los personajes fueron hechos dentro del software 3D Studio Max, ya que su interfaz es más amigable y tenemos más herramientas para generarlos.

Para seguir la técnica de modelado de baja poligonización, se empieza a crear una caja, la cual nos servirá como molde deformable virtual para darle forma humana al personaje.

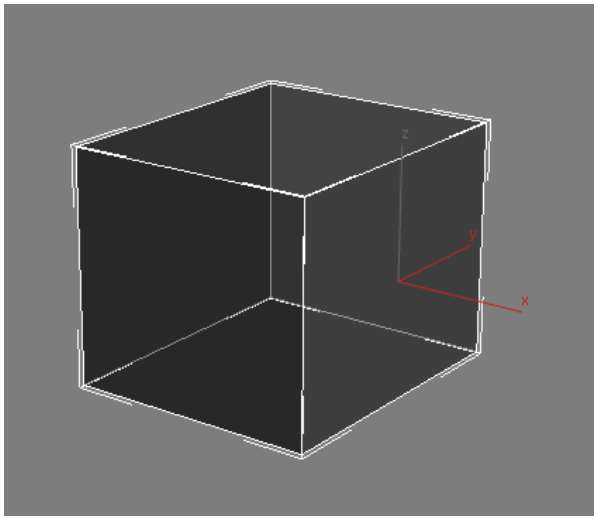


Figura 31. Caja que va a ser moldeada

Ahora la caja, siguiendo la técnica de polígonos editables, deberá ser convertida en un polígono editable, ya que con esto tendremos la manera más sencilla de poder deformarla y acoplarla a nuestras necesidades. En este paso lo que hacemos es indicarle al software que queremos editar las propiedades de los polígonos que forman a la caja, como lo son sus aristas, vértices, polígonos o caras y el objeto en sí. Con esto podremos crearle a la caja los polígonos necesarios para formar al personaje.

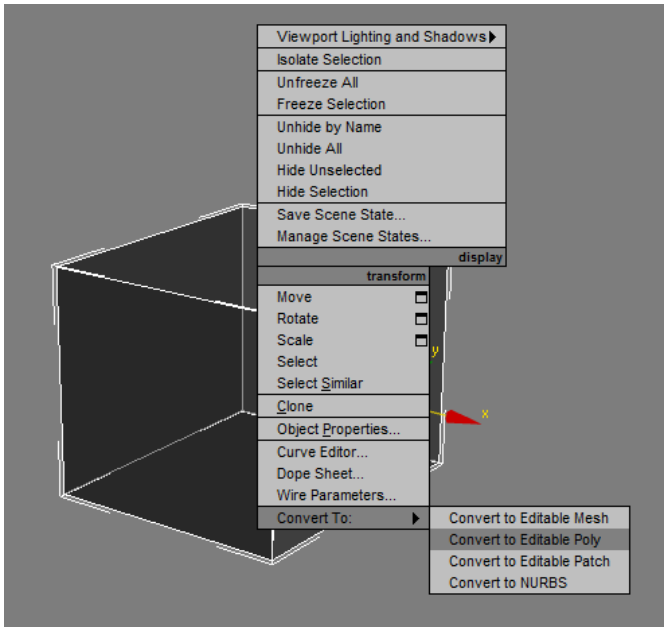


Figura 32. Aquí se observa que la caja es convertida a un polígono editable

Para inciar el proceso de modelar la caja, necesitamos unas guías, las cuales no servirán para tener una referencia de cómo debe quedar el personaje. Para eso necesitamos los blueprints que hemos hecho con anterioridad.

Los blueprints deberán de ir colocados de tal forma que nos den una vista frontal y lateral de cómo debe quedar el personaje, y estos los podemos poner sobre planos que alberguen a éstos.

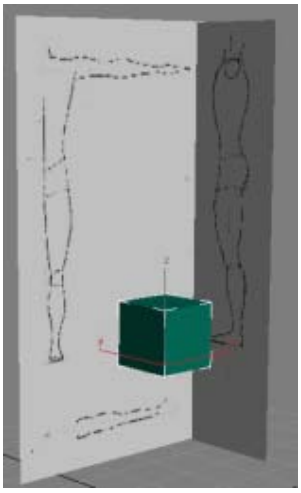


Figura 33. Se observa que por la parte trasera y de costado del polígono editable, existen las guías que son los blueprints

El modelado del personaje, después de haber efectuado las transformaciones geométricas al polígono editable, que es la caja en este caso, el personaje principal es hecho como se muestra a continuación:

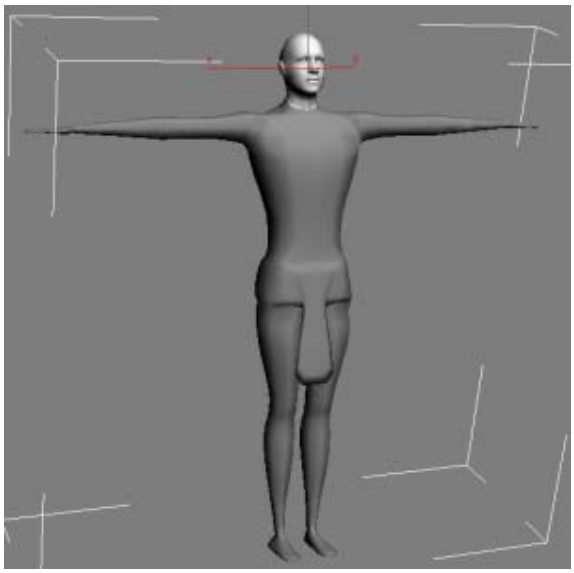


Figura 34. Después de una larga serie de transformaciones geométricas, nuestro polígono editable se convierte en nuestro personaje principal

Modelado personaje secundario

Para los personajes secundarios, se utilizaron las mismas técnicas anteriores de modelado. Los cuales son ilustrados a continuación:

Figura 35. Este es el personaje secundario. Imita a un guerrero leopardo

Texturización personaje principal

Para nuestro personaje principal, se ha efectuado la texturización en partes, es decir, se texturiza primero la parte de las extremidades, el torso, el taparrabo y la cabeza.

Para explicar el procedimiento que se siguió para la aplicación de las texturas, se explicará en forma general, presentando ilustraciones de todas las partes que se texturizaron.

Se empieza seleccionando los polígonos de la parte a aplicarle las texturas.

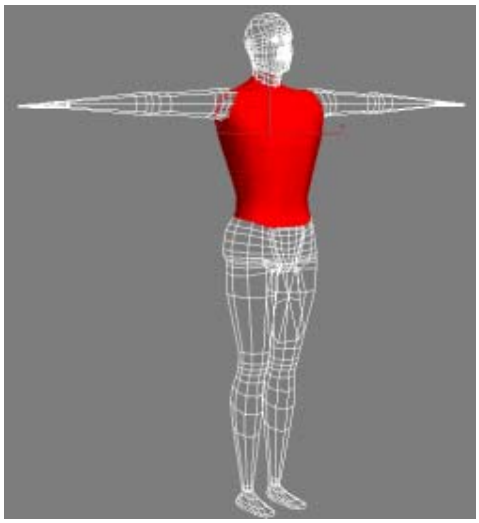


Figura 36. Se muestra la selección del torso del personaje. Esta es la manera de hacerlo para todos los polígonos del personaje

Después se le asigna a cada una de las partes un identificador único, ya que en el editor de materiales se hará la relación con los identificadores del multimaterial.

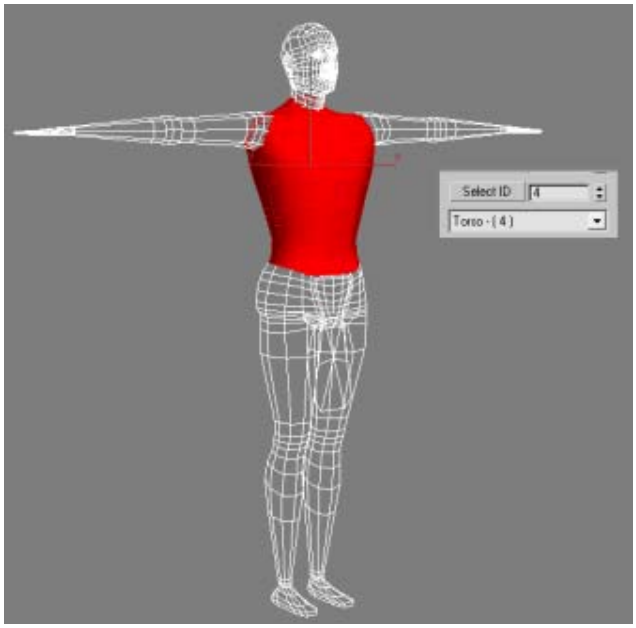


Figura 37. Se muestra el identificador que le corresponde al torso, el cual es el número 4.

Por cada identificador de los polígonos, aplicaremos el UV Mapping, el cual quedará como sigue

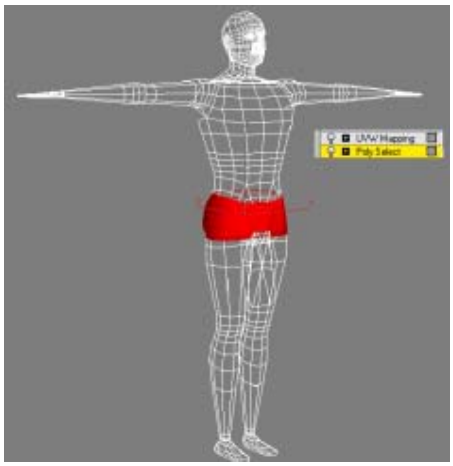


Figura 38. Se muestra en esta imagen la aplicación del UV Mapping a una parte del taparrabo del personaje

Ahora para cada UV mapping, les asignaremos el identificador correspondientes de las texturas almacenadas en el multimaterial.

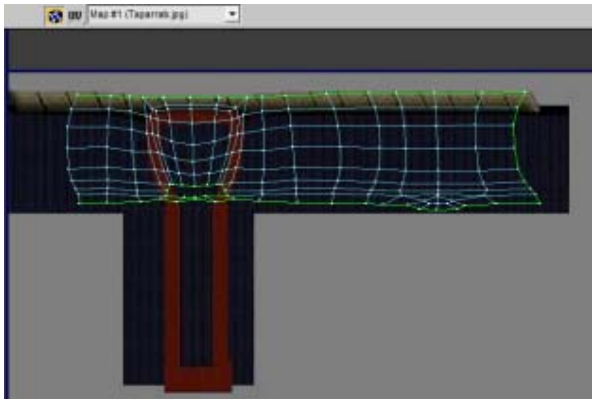


Figura 39. Aquí se observa el desenrrollamiento de la figura 11, el cual es la malla con contornos verdes. En el fondo se observa la textura que le corresponde a esa parte

Una vez efectuándose el UV mapping se regresa al polígono editable para hacer una colisión de las texturas hacia los polígonos del personaje principal.

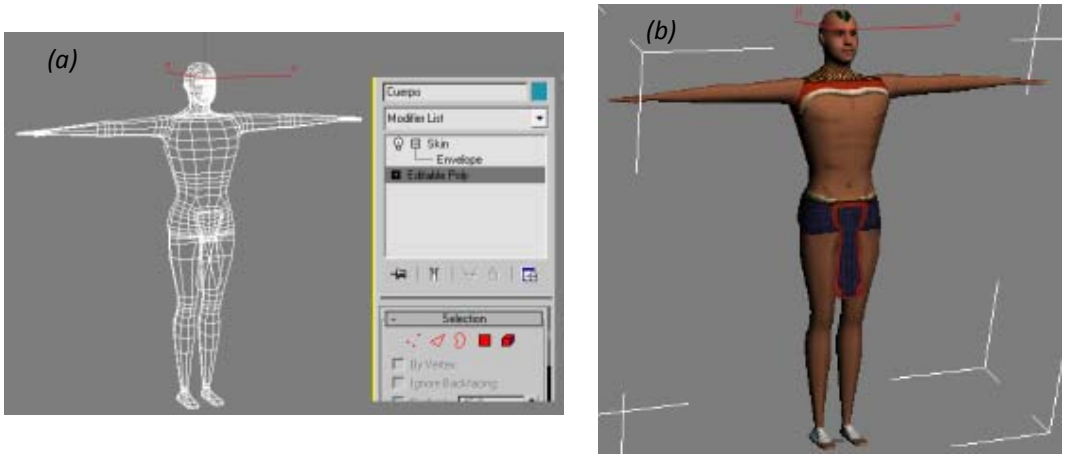


Figura 40. La colisión en la pila que se muestra en la imagen (a) es para que desaparezca los UV mapping y se colisionen a nuestro personaje en la imagen (b).

Texturización personaje secundario

El procedimiento de texturización para este personaje secundario sigue los mismos pasos que el personaje principal. Ya colapsadas las texturas hacia los polígonos del personaje secundario, éste queda como sigue

Figura 41. Este es el personaje secundario después del proceso de texturización.

ANIMACIÓN

Antes de empezar a hablar de la animación de los personajes, necesitamos hablar de cuál es el sistema que impulsa al personaje para que se mueva. Para que el personaje empiece a moverse, necesitamos hablar de la etapa del "skinning" o asignación del esqueleto al personaje, para que éste último lo podamos articular a nuestro antojo y poder formar movimientos suaves y asemejarlos a los movimientos reales. El esqueleto tiene la bondad de que sus junturas, ya tienen por defecto, límite de movimiento dependiendo de que hueso se esté manejando. Así el personaje, no podrá hacer un movimiento que no sea semejante al del cuerpo humano.

Nuestro polígono editable cuando tenemos que animarlo, tenemos que empezar a pensar en él, como si fuera la pura piel de un humaniode, es ahí cuando el skinning toma lugar. El skinning es básicamente el procedimiento se asignación de valores de peso a los vértices de la piel o "skin" de nuestro personaje a través de los huesos de un esqueleto, el cual está prefabricado en cualquier software de modelado en 3D. A cada hueso se le asignan un grupo de vértices; esta asignación es dada por el Árbol Euclidiano de Steiner, donde el radio de Steiner es $2/\sqrt{3}$. Estos valores de peso para cada vértice del skin son albergados en una tabla, los cuales nos indican el nivel de influencia que tiene ese hueso sobre las transformaciones geométricas que sufre el vértice en el espacio debido al movimiento. La asignación de valores de peso son dadas por el método de Deformación Sub-espacial del Esqueleto (SSD por sus siglas en inglés)

Deformación Sub-espacial del Esqueleto (SSD)

Es el método más simple y usado para calcular las deformaciones de la piel en tiempo real. Para explicarlo hay que tener en cuenta las siguientes anotaciones: la posición v de un vértice en la posición de descanso del personaje está escrito como $\hat{v}$. Los huesos están indexados desde 1 a b. La matriz de transformación asociada con el hueso i en su posición actual es llamada como T_i y la transformación del mismo hueso en la posición de descanso esta escrita como $\hat{T}_i$.

La posición del vértice v cuando se mueve rígidamente con un hueso en particular puede ser escrito como: para cada hueso i la posición del vértice en la posición de descanso es primero transformado desde las coordenadas del personaje ($\hat{v}$) a las coordenadas del hueso ($\hat{v}_i$) aplicando la inversa a la transformación del hueso en su posición de descanso

$$\hat{v}_i = \hat{T}_i^{-1}\hat{v}$$

El vértice $\hat{v}_i$ en las coordenadas del hueso, es transformado de regreso en las coordenadas del personaje aplicando la transformación del hueso en su nueva configuración del esqueleto

$$v_i = T_i\hat{v}_i = T_i\hat{T}_i^{-1}\hat{v}$$

Esto da la posición del vértice cuando se mueve rígidamente con el hueso *i* manteniéndose estático relativamente a él.

SSD determina la nueva posición de un vértice por la combinación lineal de resultados por la rígida transformación del vértice con cada hueso. Un peso escalar w_i está dado a cada hueso influenciado y la suma de las influencias da la posición del vértice *v* en la nueva posición, como se muestra

$$v = \sum_{i=1}^{b} w_i T_i \hat{T}_i^{-1} \hat{v}$$

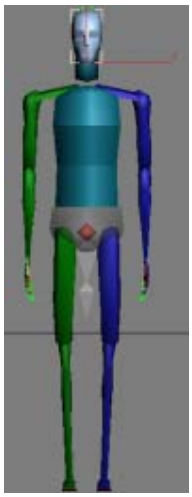
Para los huesos que no tienen influencia en el movimiento del vértice, el valor del peso asociado será de 0 . Los valores por defecto de los pesos están asignados como

$$\sum_{i=1}^{b} w_i = 1$$

Por la anterior podemos observar que, cuando se le asigna a un hueso un vértice del skin, éste último tiene un valor de peso de 1, y todos los demás vértices que no están asignados a este hueso, su valor de peso es de 0.

Para lograr el skinning en el software de modelado de 3D, primero tenemos que asignar el esqueleto al personaje. El esqueleto debe de encajar los más que se pueda dentro del personaje, para lograr movimientos más naturales.

(a)



(b)

Figura 42. En la imagen (a) se muestra al esqueleto fuera del personaje. En la imagen (b) se muestra al esqueleto dentro del personaje

Una vez que hayamos encajado el esqueleto al personaje, tenemos que ejecutar el comando que nos permite indicarle al software de modelado que nuestro personaje será asociado a un esqueleto, el comando es SKIN. Cuando se ejecuta el Skin, hay que agregar a una lista los huesos que deseamos se les asignen los vértices del personaje, así el Skin asiganará automáticamente los vértices que se encuentren alrededor de cada hueso.

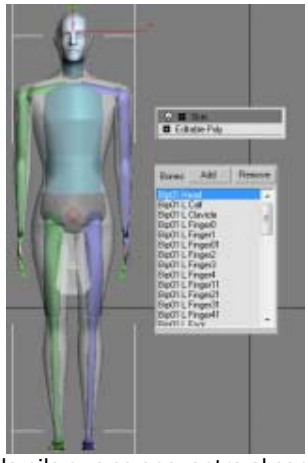


Figura 43. Se muestra en la pila que se encuentra al comando Skin, y también la lista de los huesos a asignarles los vértices

En este punto, la geometría del personaje está influenciada por los huesos por medio de SSD. Se genera una envoltura alrededor de cada hueso basado en su tamaño usando el árbol de Steiner. Los vértices que caigan dentro de la envoltura de un hueso, le serán asignados valores de peso asociado a eso hueso, por ende ya podemos mover nuestro personaje por medio de los huesos. Cabe mencionar que cada vértice únicamente podrá ser influenciado por tres huesos como máximo, lo cual quiere decir que ese vértice se mueve cada vez que cualquiera de esos huesos sean movidos. Las envolturas de cada hueso deberán ser manipuladas por sus transformaciones geométricas, ya que deben de encajar solo con el área del hueso.

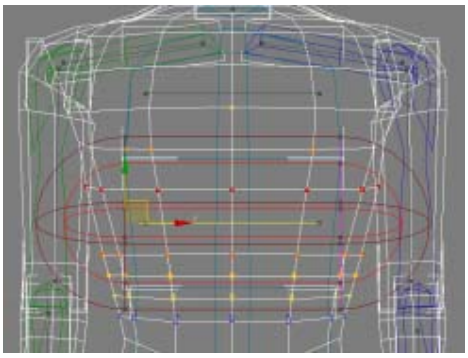


Figura 44. En esta imagen se muestra la envoltura de una de las vértebras del esqueleto. Los vértices del personaje se ponen de colores cuando están influenciados por el hueso

Mientras se le asignan los valores de peso automáticamente a los vértices, uno podrá observar que el movimiento de los vértices no es el adecuado, por ejemplo, cuando se dobla el brazo, puede haber unos vértices que hagan que la piel se contraiga a sí misma, lo cual quiere decir que ese vértice tiene un valor de peso muy grande (valor de 1) asignado por la influencia del hueso sobre éste.

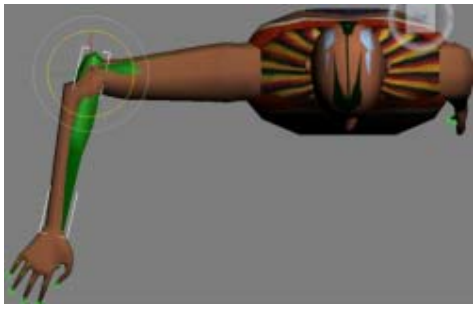


Figura 45. Podemos observar que los valores de peso de los vértices del antebrazo están, ya el hueso de éste se sale de la piel, y se dobla de una manera muy peculiar el codo

Para prevenir este problema de valores de peso de los vértices, tenemos que modificarlos individualmente. Para esto, el comando SKIN genera en paralelo una tabla con todos los valores de peso de los vértices y con que huesos están influenciados. Nosotros podemos manipular esos valores dentro de la tabla para que el movimiento de los vértices sea lo más parecido a la realidad. Solo recordaremos que cuando un vértice tiene un valor de peso de 1, éste vértice sigue al hueso sin ningún retraso; pero si un vértice tiene un valor de peso de 0, éste no sigue al hueso, lo cual indica que no tienen influencia el hueso sobre él. Por eso, cuando se modifiquen los valores de peso del vértice, tenemos que tomar en cuenta que, para tener un movimiento suave del vértice, éste deberá tener un valor de peso entre 0 y 1.

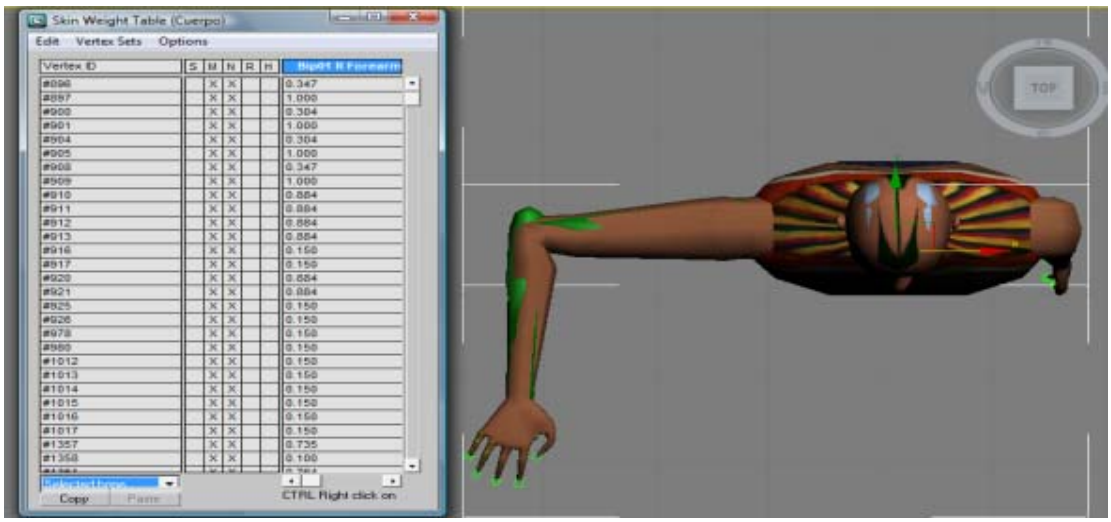


Figura 46. Aquí se muestra la tabla de valores de peso (lado derecho) para el antebrazo. También se observa que el dobles del codo es más cercano al movimiento real de un codo humano

Una vez hecho el skinning, se procede a efectuar la animación del personaje. La animación es los movimientos que efectuará el personaje a través de un determinado lapso del tiempo. Existen diferentes técnicas de animación, las cuales a continuación explicaremos brevemente.

1. *Interpolación entre llaves (keyframe interpolation):* El proceso de creación de los fotogramas intermedios para rellenar la acción entre dos posiciones clave se denomina interpolación (en inglés, in-betweening). Se han desarrollado técnicas que permiten que la computadora cree estos fotogramas mediante el cálculo de los puntos comunes entre un fotograma clave y otro. En el caso más sencillo, la computadora dibuja el movimiento intermedio de dos puntos correspondientes calculando la distancia al punto medio. La repetición de cálculos del punto medio puede generar la ilusión de un movimiento fluido y continuo.

2. *Cinemática inversa (IK, inverse kinematics):* Usa un método de metas directas donde el animador posiciona un objeto hijo y el programa calcula la posición y orientación del objeto padre. Se puede tener una cadena cinemática, la cual es un conjunto de elementos sujetos entre sí.

3. *Deformaciones (Morphing):* Morphing es un término derivado de metamorfosis, lo cual quiere decir que cambia el modelo físico o la forma. Morphing es comúnmente usado para movimiento de labios y expresión facial en caracteres de 3D, pero también es usado para confiar la forma de cualquier modelo en 3D. el proceso del Morphing es la de trasladar la posición de los vértices desde su arreglo en un objeto al arreglo de otro objeto; ambos deben de tener el mismo número de vértices.

4. *Sistemas de partículas como agua, fuego, humo, etc (particle systems):* Es un sistema de objetos que generan sub-objetos no editables llamados partículas.

5. *Multitudes (Flock, crowd):* Te permite similar el comportamiento de un grupo de personas, animales y otros entes paramétricos. Usando un procedimiento procedural se pueden crear caracteres autónomos que se pueden animar a sí mismos basados en condiciones elegibles.

6. *Telas (cloth):* Es una composición de mallas de triángulos que tienen propiedades físicas. Se puede usar para simular ropa, trampolines, láminas de metal y otros artículos bidimensionales

7. *Captura de movimiento*: Se utilizan trajes ópticos, magnéticos y mecánicos para la captura del movimiento con el uso de sensores conocidos como "Trackers".

Para la elaboración de la animación del personaje, haremos uso de las técnicas de interpolación de llaves y cinemática inversa.

Cinemática inversa

El objetivo de la cinemática inversa consiste en encontrar el gesto que deben adoptar las diferentes articulaciones para que el final del sistema articulado llegue a una posición concreta, la cual se conoce y se impone dentro del espacio tridimensional.

Para su principio básico, lo haremos explicándolo de forma sencilla a través de un ejemplo que estudiamos y hayamos.

Tenemos un sistema de piernas de nuestro esqueleto, las piernas surgen de la cadera y a continuación estaría el fémur, la rodilla, la tibia y el pie.

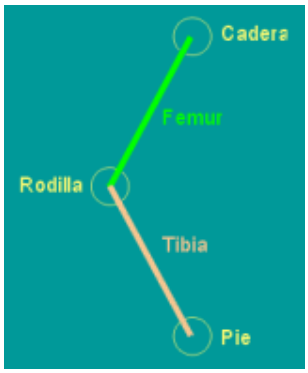


Figura 47. Sistema de piernas desde la cadera hasta el pie

En esta figura podemos observar el sistema que pretende imitar una pierna, como es lógico la posición de la cadera es fija al esqueleto.

Lo que queremos hacer a continuación es posicionar el pie del sistema a 10cm en vertical y a 5cm desde la cadera. Aquí es donde entra el método geométrico de la cinemática inversa, ya que esto soluciona el ángulo que debe de tener el fémur respecto a la cadera, el ángulo que debe de adoptar la tibia respecto a la rodilla y la posición que debe de adoptar la rodilla para que el pie pueda estar en la posición que queremos que esté.

El origen del sistema cartesiano será la cadera, tomando en cuenta que estamos trabajando sobre el plano XY. Los vectores del fémur y tibia tienen longitud de 10cm cada uno.

Ahora ponemos las coordenadas de la posición del pie (-5,-10). De acuerdo al sistema de referencia, la coordenada en X es de -5, porque nuestro pie lo queremos articular hacia en frente de la cadera, y el dobles de la rodilla solo nos permite ese movimiento en sentido negativo de las abcisas y de las ordenadas. Para fines de este ejemplo, tomaremos solo el valor absoluto de las coordenadas deseadas para el pie.

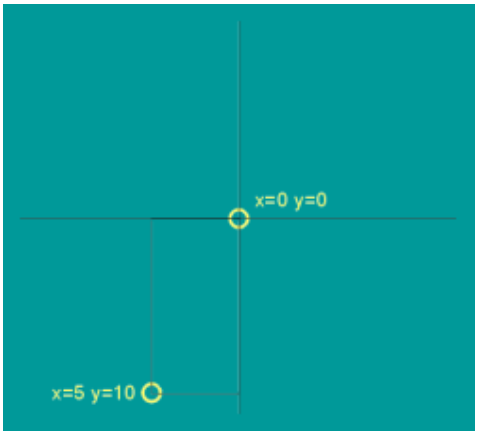


Figura 48. Sistema de referencia con la cadera como punto en el origen

Se observa que entre los puntos de la cadera y el pie, se hace un triángulo rectángulo. Así podremos calcular la hipotenusa (h) de éste último.

$$h = \sqrt{X^2 + Y^2} = \sqrt{5^2 + 10^2} = 11.80 \text{ cm}$$

Como podemos observar de la fórmula anterior, el pie debe de estar posicionado a una distacia de 11.80 cm desde la cadera.

Ahora debemos de calcular el ángulo que forma la hipotenusa de la siguiente forma:

$$\theta = \arctan\left(\frac{X}{Y}\right) = \arctan\left(\frac{5}{10}\right) = 26.56^{\circ}$$

Ya tenemos definido en coordenadas polares el punto donde queremos poner el pie del robot con respecto al punto de origen formado por la cadera que es de donde parte el fémur.

Ahora vamos a calcular que ángulo tendrán que tomar el fémur y la tibia para que el pie apunte justo a esa posición.

Como el fémur y la tibia tienen las mismas longitudes, podemos decir que el punto donde convergen siempre es el punto medio de la distancia entre cadera y pie, que es la línea punteada de la siguiente figura, así que, la rodilla siempre estará sobre esa línea punteada.

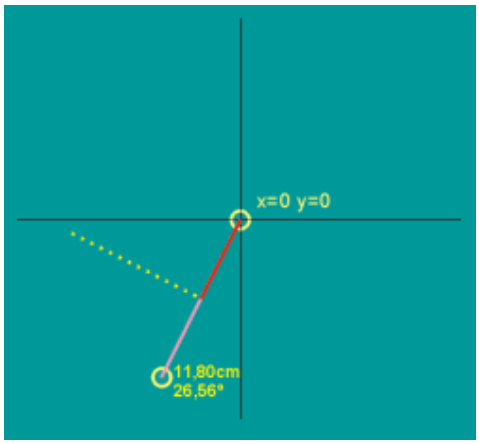


Figura 49. Longitud y ángulo del pie desde el origen, junto con el punto medio que es la rodilla.

Observamos que la línea roja y al línea punteada amarilla forman un ángulo de 90º y como en el caso anterior, tendremos un triángulo rectángulo solo que ahora la hipotenusa es el propio fémur y contamos con la ventaja de saber su longitud, con lo que solo tendemos que calcular su ángulo, y tenemos que tener en cuenta que este ángulo que obtendremos será en relación a la línea entre cadera y pie por lo tanto para posicionar en el sistema cartesiano tendremos que sumarlos como veremos a continuación. Ahora vamos a calcular el ángulo del fémur de la siguiente manera.

La longitud del cateto opuesto a la rodilla (línea roja) es de 5.9cm (recordemos que es la mitad de la longitud entre cadera y pie) y la hipotenusa (fémur) mide exactamente 10cm con lo cual podemos calcular el seno de este triángulo rectángulo como sigue:

$$sen\,\theta_1 = \frac{cateto\quad opuesto}{hipotenusa} = \frac{5.9}{10} = 0.59$$

Para saber el valor del ángulo θ_1 tenemos que calcular su arcsen de la función anterior.

$$\theta_1 = arcsen(0.59) = 36.16^{o}$$

Este ángulo es el ángulo agudo del triángulo rectángulo así que hay que restarle 90º para obtener el ángulo que nos interesa, y esto sumado al ángulo formado por la línea entre cadera y pie nos da el ángulo total referenciado al sistema de coordenadas cartesiano o lo que es lo mismo referenciado al anclaje de la pierna o cuerpo del robot.

θ_2 = Ángulo del fémur = (90-36.16)+26.56 = 80.4º

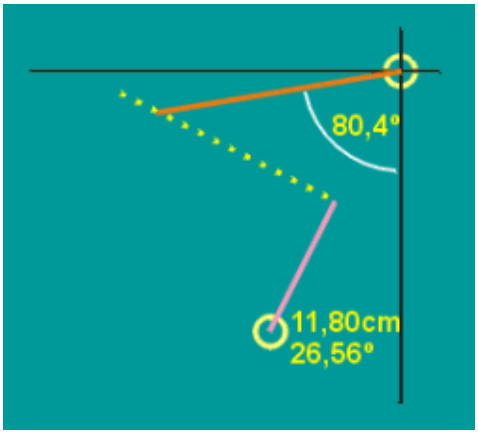


Figura 49. Ángulo del fémur para posicionarlo respecto al pie.

Así ya tenemos el fémur en su sitio y solo nos falta calcular el ángulo de la tibia, pero este cálculo es mucho más simple ya que tal ángulo es el doble del ángulo agudo calculado anteriormente, con lo cuál:

θ_3 = Angulo Tibia = 36,16º x 2 =72,32º

Este ángulo es en relación al ángulo del fémur como puede observarse en la siguiente figura.

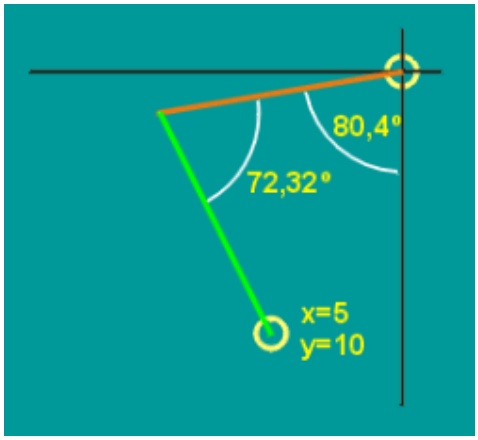


Figura 50. Ángulos del fémur y tibia para posicionarlos de acuerdo al pie

Llegados a este punto ya hemos conseguido el objetivo de saber que ángulo deben adoptar las extremidades de una supuesta pierna, para alcanzar un punto concreto.

Interpolación de llaves

La interpolación es un proceso que crea puntos medios entre dos puntos dados. Esto trasladado a la animación, se puede describir como encontrar los puntos medios entre una posición inicial de un objeto y su punto final.

La interpolación de llaves es la técnica para crear llaves intermedias entre dos llaves clave. Cada clave es un fotograma con una medida establecida dentro de una línea del tiempo; cada fotograma clave se le asigna la posición inicial del objeto en cuestión, y a otro fotograma clave se le asigna la posición final del objeto.

Cabe mencionar que cuando se hace una interpolación de llaves, lo que se interpola en realidad son cada uno de los vértices de la geometría a animar.

Existen varios métodos de interpolación, como lo son la interpolación lineal, interpolación sostenida, interpolación de Bezier, etc. Pero fines de esta explicación, se explicará la interpolación lineal, ya que es la más sencilla y con la que se trabajó para animar al personaje.

Interpolación lineal.

Consiste en crear un punto desconocido entre dos puntos dados. Este punto deberá contenerse dentro de la recta formada por los dos puntos dados. Como se muestra a continuación:

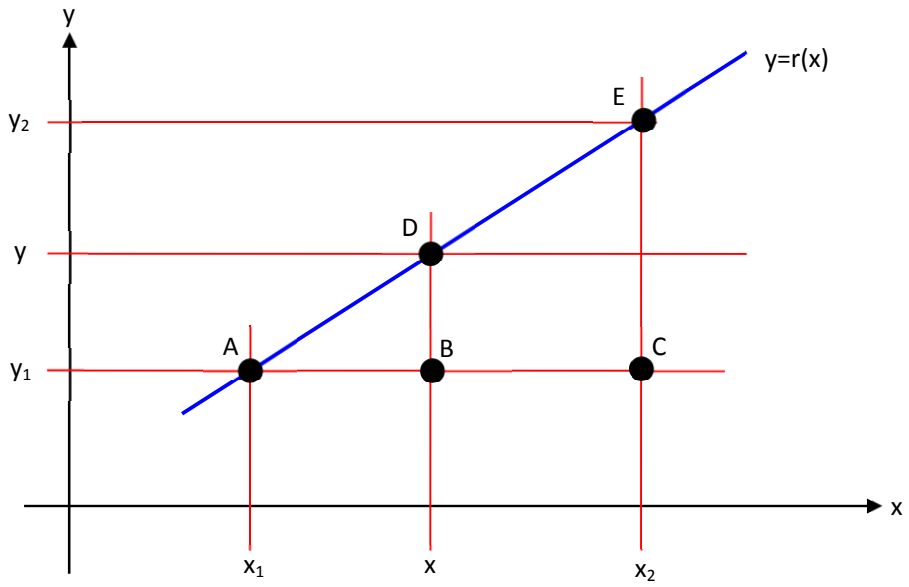


Figura 51

Para encontrar el punto (x, y) que contiene la recta y = r(x) formada por los puntos (x1,y1) y (x2,y2), tenemos que adentrarnos en el concepto de triángulos semejantes. Dos triángulos son semenjantes cuando sus ángulos de los dos son iguales.

En base a triángulos semejantes, podemos decir que los triángulos formados por los puntos ABD y ACE, son semejantes. Entonces:

$$\frac{\overline{AC}}{\overline{AB}} = \frac{\overline{CE}}{\overline{BD}}$$

Despejando el lado del triángulo BD del triángulo ABD obtenemos

$$\overline{BD} = \frac{\overline{AB}}{\overline{AC}}\,\overline{CE} \quad \text{..... (1)}$$

De acuerdo a los valores dados por la figura anterior, debemos de hallar el valor de la función $y = f(x)$ la cual contiene al punto D de la recta $y = r(x)$. Sustituyendo los valores de la gráfica en la ecuación (1), obtenemos:

$$(y - y_1) = \frac{(x - x_1)}{(x_2 - x_1)} (y_2 - y_1) \quad \text{.....(2)}$$

Despejando el valor de y de la ecuación (2) obtenemos:

$$y = \frac{(x - x_1)}{(x_2 - x_1)} (y_2 - y_1) + y_1$$

Este es valor de los puntos que se generan cuando interpolamos dos puntos dados en el espacio.

Personaje principal

Se empezará explicando la animación del movimiento de correr del personaje. Para la animación del personaje principal, se tuvo que emplear, en primer lugar la cinemática inversa de uno de los vértices de cada una de las extremidades, ya que moviendo el pie, el fémur y la tibia se van a mover con él, así para el caso de las manos, se mueve el brazo y el antebrazo con ellas. Una vez posicionados todos los huesos en el lugar que queremos, creamos la llave clave de la posición inicial. Después de dos fotogramas se genera el siguiente movimiento que le sigue al movimiento inicial y se le genera su llave clave. Esta operación se repite cuatro veces más para generar todas las llaves claves del movimiento de correr del personaje. Cuando se generar todas las llaves claves del movimiento, se ejecuta la interpolación entre llaves, lo cual quiere decir que, entre las llaves claves, se generan llaves intermedias con todas las interpolaciones de cada uno de los vértices del personaje, generando así la ilusión del movimiento del personaje. Este proceso se sigue para cada uno de los movimientos del personaje (caminar y parado).

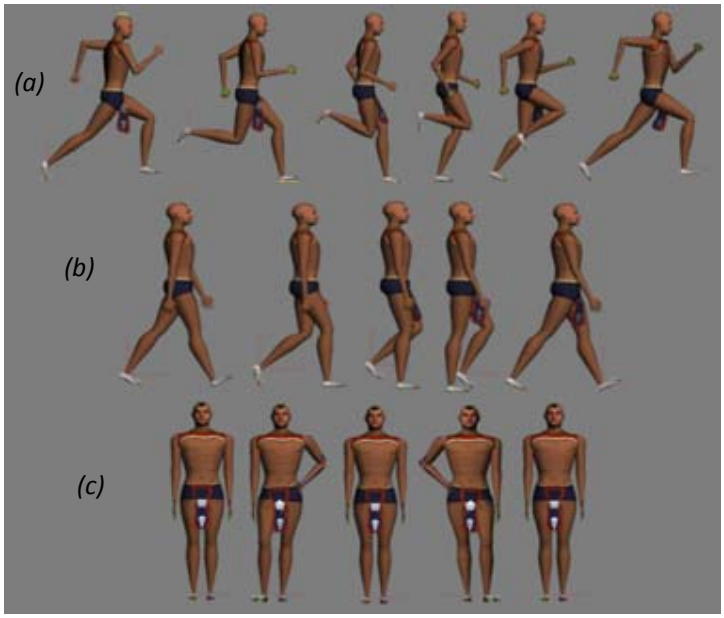


Figura 52. (a) Correr. (b) Caminar. (c) Parado

Personaje secundario

De la misma manera se anima al personaje secundario. La siguiente figura muestra los movimientos de éste.

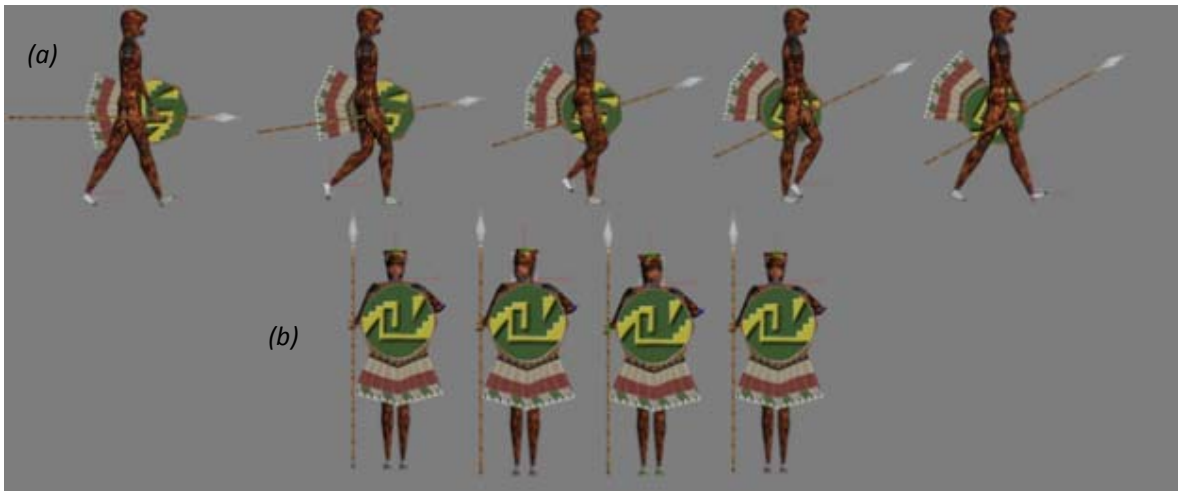


Figura 53. (a) Caminar. (b) Parado

5.3.4.- Trabajando en Quest3D

El software que se empleo para generar visualizar nuestras geometrías en tiempo real e interactuar con ellas, fue Quest3D. Es un software de diseño en 3D, que permite la generación de entretenimiento digital en tiempo real, como es el caso de los videojuegos.

Quest3D tiene un interfaz de programación a base de un sistema de CANALES, lo cual permite evitar a gran medida los errores de sintáxis y compilado. Pero también tiene la flexibilidad de hacer una reprogramación a esos canales para que compilen de la manera que uno quiere. También puedes crear tus propios canales haciendo uso de la progrmación orientada a objetos.

Se escogió usar Quest3D en vez de Virtools por tres razones muy importantes; la primera es que su licencia es más barata que Virtools; la segunda es porque Quest3D está orientado a simulaciones, diseño web en 3D, arquitectura y videojuegos, lo que Virtools no posee, ya que está más oreintado a videojuegos; y por último por que sus requerimientos que necesita su motor de renderizado para manejarlo es muy bajo, como lo son:

• Windows 2000, Windows XP, Vista o mejor (64 o 32 bit)

• 256 MB memoria RAM

• 1Ghz en procesador

• Tarjeta de gráficos compatible con DirectX

• 32 MB en memoria de tarjeta de gráficos

• 400MB libres en disco duro

Virtools necesita mínimo de Windows XP, 1 Gb en memoria RAM y 256 Mb en memoria de tarjetas de gráficos.

Daremos una breve introducción de cómo es que se programa a través de canales en el interfaz de Quest3D. Esta breve explicación será dada a través del resumen del manual del software.

El gráfico de canales es la ventana donde todas las estructuras de canales son construidas. La programación en Quest3D está basada en canales y cada canal tiene una función propia. Estos canales tienen entradas y salidas. Sus entradas con los rectángulos pequeños debajo de cada canal, y sus salidas son los rectángulos pequeños arriba de cada canal. La imagen de un canal se muestra abajo

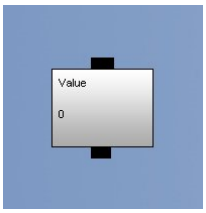


Figura 54. Canal en Quest 3D.

Un grupo de canales es una estructura donde se interconectan varios canales como se muestra a continuación.

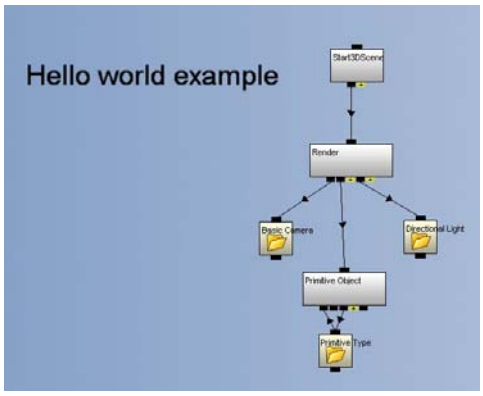


Figura 55. Grupo de Canales en Quest 3D.

Los canales pueden interconectarse entre sí, y cuando se conectan, el canal que queda arriba es llamado Padre, y el canal que queda abajo es llamado Hijo. Los hijos son usados como entradas de datos para los padres o como salidas de datos de éstos. La imagen que se muestra a continuación ilustra mejor lo anteriormente dicho.

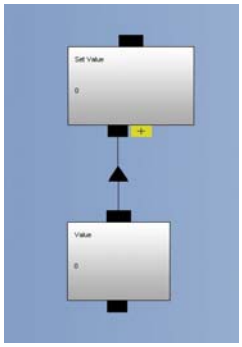


Figura 56. Canales de Entradas o "canales hijos" de un "canal padre".

Cada uno de los canales tiene sus propias entradas, la cuales sirven como variables de entrada para el objeto que maneja ese canal, y dependiendo del dato de entrada del canal, será el resultado que éste genere. Por ejemplo, cuando se aplica el canal de ClearScreen a nuestro proyecto, lo que éste hará es cambiar el color de fondo del proyecto. Este canal maneja valores RGB, los cuales son los valores de Rojo, Verde y Azul de nuestro display, donde el valor por defecto de esos valores es 0, ya que si le asignamos un valor de 1 a una de sus entradas, el color que maneja esa entrada prevalecerá. La siguiente imagen se le asigna un valor de 1 a la entrada que controla en color rojo, esta asignación del valor se hace a través de un canal que se convierte en hijo. Esto se ilustra en la siguiente imagen, donde el valor de 0 se asigna en la entrada que controla al color rojo (parte de arriba de la imagen), y después le asignamos un valor de 1 a esa misma entrada(parte de abajo de la imagen).

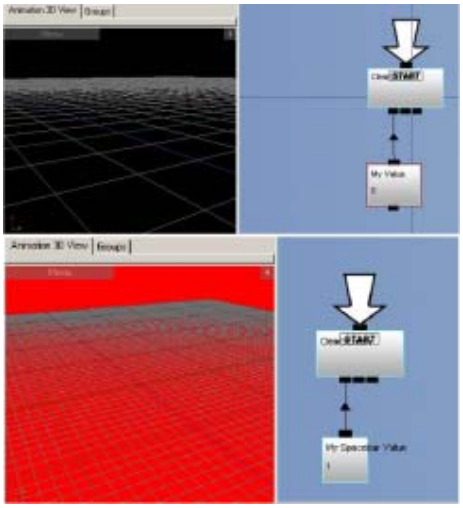


Figura 57. Canales de Entradas o "canales hijos" de un "canal padre".

Este el funcionamiento a grandes resgos de todos los canales que maneja Quest3D, y así es como se les asignan los valores de entrada dependiendo del tipo de canal que se esté manejando.

Cabe mencionar que el canal Padre de cualquier grupo de canales, siempre manda llamar en primer lugar al canal hijo que se encuentre más a la izquierda de sus canales hijos, hasta llegar al último canal hijo que se encuentra más a su derecha de sus canales hijos.

La intención de este breve resumen de compartamiento de los canales, es puramente informativo, ya que queremos que se conozca como funcionan los canales, y sobre todo, para que al momento de explicar como se hizo la programación de los elementos del paseo, se pueda explicar de manera más sencilla.

5.3.4.1.- Importación de la geometría.

Cuando tratamos de exportar nuestros trabajos de un software a otro, lo que tratamos de hacer es crear un archivo que contenga toda la información, para que éste sea asimilado por el software al cual queremos introducir dicha información. El archivo debe de ir escrito en un lenguaje el cual, tanto el software emisor como el software receptor, puedan procesar y calcular todos sus atributos que éste contenga y podamos visualizarlos en todo momento.

El exportador que utilizamos para poder pasar nuestro proyecto de 3DStudio a Quest3D, fue el llamado Panda DirectX Exporter Tool (PDE). La razón por la cual usamos PDE fue porque no hay una exportación nativa directa entre Quest3D y 3DMax Este exportador no tiene muchas documentación acerca de como fue hecho o alguna estructura, pero trataremos de explicarlo.

Panda DirectX Exporter Tool (PDE) fue hecho por Andy Tather. Te permite exportar todos tus objetos de tu escena y las animaciones hechas para este en un solo archivo .X , lo cual lo hace un software de licencia libre. La característica de este archivo .X es que describe los datos de la geometría, la jerarquía en los fotogramas y la animación. PDE fue hecho como una alternativa al exportador DirectX de Microsoft. Algunas diferencias entre PDE y el exportador de Microsoft son que genera matrices de los movimientos de cada uno de los fotogramas de la animación hecha en 3DStudio, lo cual el exportador de Microsoft necesita de otro proceso para generar esas matrices de perspetivas. Los colores de vértices son soportados por PDE, lo cual no hace el exportador de Microsoft al momento de generar el archivo. Tiene opción para manipular los vértices y normales con referencia a la textura de la geometría, logrando así la reducción de los vértices.

Se puede hablar un poco de cómo es un archivo .X a grandes rasgos. Éste tipo de archivo almacena toda la información de un método jerárquico, es decir, que respecta el nodo padre de una ramificación con sus respectivo hijos. Cada nivel contiene tiene objetos con sus respectivos datos, pero generalmente se divide lo suficiente como para que sólo unos pocos objetos estén en el mismo nivel. El objeto principal de la jerarquía del modelo geométrico es un hueso. Para la animación, el objeto principal de la jerarquía es una colección de movimientos. El archivo .X puede ser guardado como datos individuales de rotación, transformación o escalamiento, o como una matriz que haga todo lo anterior. El tamaño de la información que soporta el archivo .X no tiene límite, pero hay que tener en cuenta que el cuello de botella es de cuantos polígonos está formado tu estructura y cómo va a impactar en el renderizado en tiempo real.

Dada la explicación superficial de lo que hace el PDE, daremos algunas imágenes de lo que contiene dicho exportador y una breve explicación de cada una de ellas.

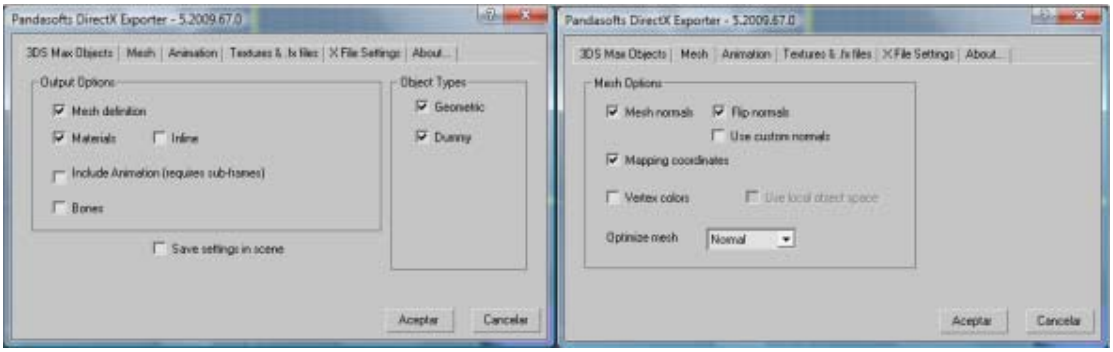


Figura 58. En la pestaña "3DS Max Objects" podemos elegir qué queremos exportar de nuestro personaje y todos sus atributos. En la pestaña "Mesh" seleccionamos que propiedades de nuestra geometría queremos exportar referentes al mapeo de texturas.

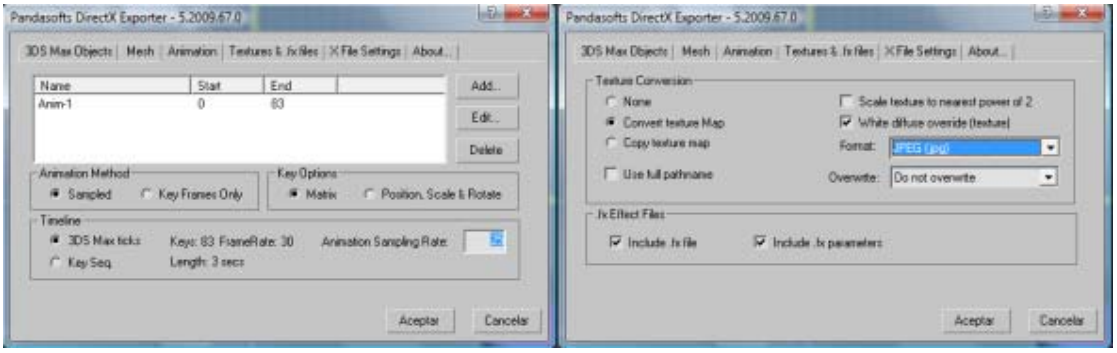


Figura 59. En la pestaña "Animation" escogemos las propiedades con las cuales la animación del personaje será exportada. En la pestaña "Textures & .fx files" escogemos las propiedades con las cuales las texturas de nuestro personaje serán exportadas.

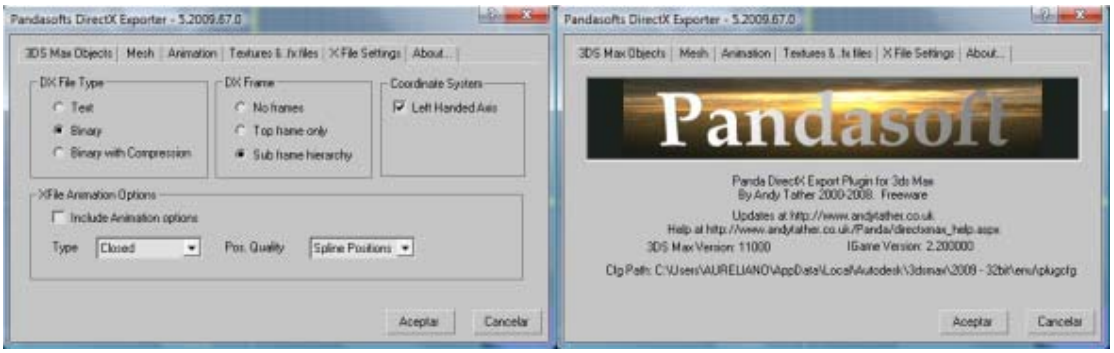


Figura 59. En la pestaña "X Files Settings" manipulamos las propiedades del archivo .X para poder exportarlo a nuestro gusto. En la pestaña "About" es la información acerca del plug in PDE.

5.3.4.2.-Animación, gravedad y colisiones.

En esta sección presentaremos el cómo fue posible controlar a nuestro personaje principal dentro de Quest3D, junto con lo que se necesito para efectuar dicha tarea. Cabe mencionar que la explicación de los canales que maneja Quest3D, será en base a la información que se haya recopilado al respecto, ya que ese tipo de información casi no existe, debido que el fabricante no otorga el tipo de información específica por cada uno de sus diseños.

ANIMACIÓN

Cuando se exporta la geometría con todas las características inherentes a ella hacia Quest3D, se generan los canales que se muestran en la siguiente imagen.

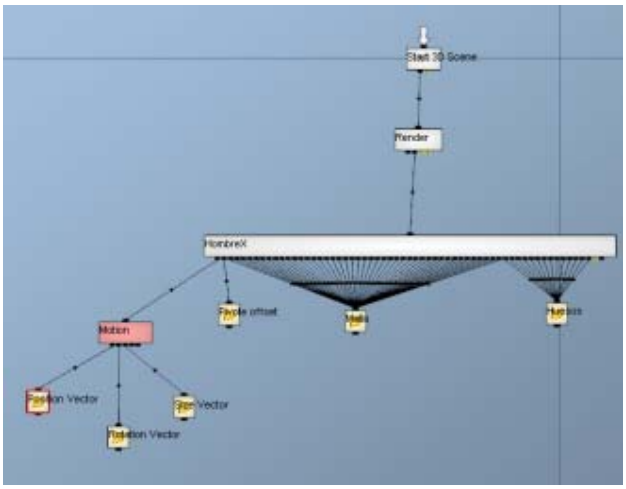


Figura 60. Canales que se generan cuando se importa a Quest3D nuestro personaje desde 3DStudio Max.

La figura anterior nos muestra cuales son los canales que se generan automáticamente dentro de Quest3D. El canal de "Start 3D Scene", indica a nuestra tarjeta de gráficos que se generará un renderizado en tiempo real de nuestro proyecto. El canal de "Render", muestra o renderiza nuestras geometrías, luces y todas las peculiaridades que se alojan en el proyecto hacia la cámara que nosotros estemos manejando en cualquier momento. El canal de "HombreX" es una canal que aloja todas las propiedades de nuestro personaje, como lo son la matriz de movimiento dentro del espacio, la textura, los huesos, animación, etc. El canal "Motion" es la matriz de 4x4 de todas las tranformaciones geométricas que sufre nuestra geometría a través del videojuego; ésta matriz está conformada por los vectores de posición, rotación, escalamiento y un vector que haga homogénea la matriz. La carpeta "Pivote offset" es una matriz de movimiento de 4x4 que sirve para posicionar al personaje en un lugar diferente al origen del sistema. La carpeta de "Malla" contiene a cada uno de los huesos del personaje, junto con los vértices que éstos mueven y sus respectivas texturas. La carpeta de "Huesos" almacena toda la información pertinente a los IDs de los polígonos de la geometría, como lo es la textura que debe ir en cada polígono,

lo huesos que deforman la geometría de esos grupos de polígonos y la animación que siguen esos polígonos.

Enfocaremos nuestra atención a la carpeta “Huesos”, ya que ahí tendremos que modificar las propiedades de animación del personaje.

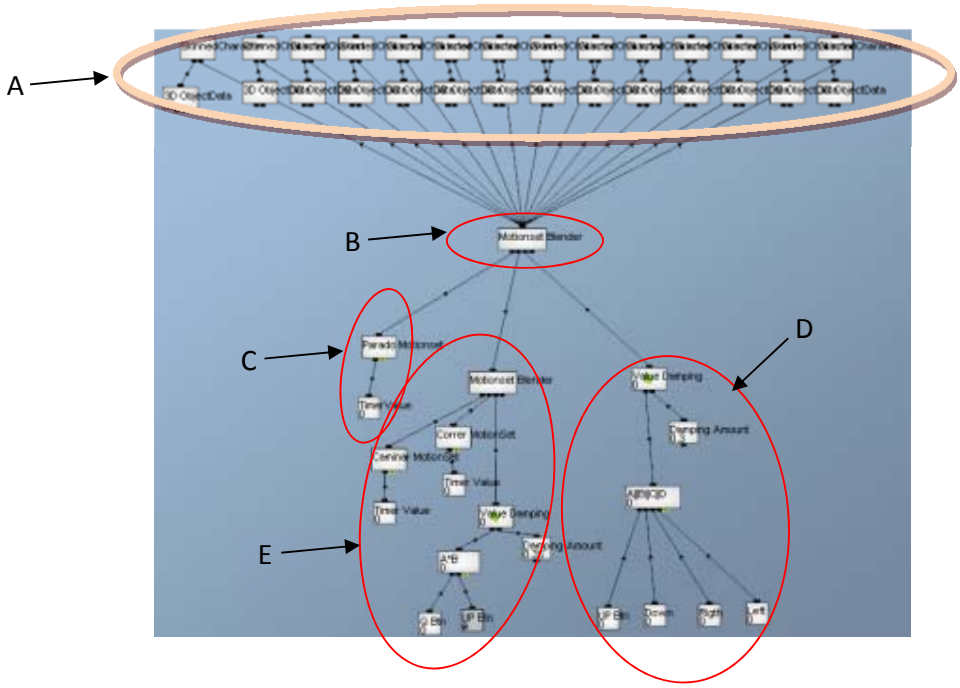


Figura 61. Canales que se alojan dentro de la carpeta “Huesos”

En el circulo A encontramos el grupo de IDs de los polígonos de nuestra geometría y los huesos que influyen es esos IDs. El círculo C aloja el canal “Parado Motionset”, el cual aloja toda la animación del personaje, y esa animación la usa para cada hueso de nuestro personaje. Como éste canal tiene toda la animación que hace el personaje cuando está parado, necesita que se le asigne un lapso en el tiempo, en el cuál esa animación solo podrá ser ejecutado. Para esto, se le asigna un canal ligado a él, el cual define el tiempo que tiene esa animación de parado para ser ejecutado, y ese canal es el “Timer Value”; este canal contiene el lapso de tiempo que tiene la animación de parado para ser ejecutado, este lapso de tiempo es único y solo será utilizado para la animación de parado; este lapso de tiempo será llamado “Parado”. En el círculo E podemos observar que hay la misma estructura de canales que en el círculo C; el canal “Caminar Motionset” y el canal “Correr Motionset”, siguen la misma programación que los canales que se encuentran en el círculo C, con la diferencia de que sus canales “Timer Value” hacen referencia a los lapsos de tiempo únicos de “Caminar” y “Correr” respectivamente. Observamos también que en el círculo E se encuentra el canal “Motionset Blender”, el cual sirve para hacer una transición de una animación a otra cuando se lo indiquen; para indicarle la transición de la animación, lo que necesita es una señal con un valor entre 0 y 1, y este valor lo que hará es que cuando sea 0, la animación de nuestro personaje será el de caminar, y si el valor cambia a 1, la animación de nuestro personaje es el de correr; la peculiaridad de este canal es que la transición de la animación puede hacerse de una manera suave dependiendo de qué tan gradual sea el valor de la señal de entrada, por ejemplo, si el valor de entrada

pasa de 0 a 1 de manera inmediata, el cambio de la animación es abrupto, pero si va gradualmente de 0 a 1, el cambio de animación será muy suave. Para mandarle a este canal la señal con el valor gradual que indique el cambio de animación, se necesita un canal llamado "Value Damping", este canal pasa del valor 0 al 1 en un incremento que nosotros le debemos, y en el círuclo E obsevamos que su incremento es de 0.35. En este mismo círculo observamos el canal que se llama "A*B" el cual es un canal para hacer expresiones matemáticas, y este canal tiene dos canales de entrada llamados "Q Btn" y "UP Btn"; el canal "Q Btn" es la entrada A del canal "A*B" que da el usuario cuando aprieta la tecla de la letra Q, la cual sirve para que el muñeco pueda correr, así cuando la tecla Q esté sin apretar arrojará el valor de 0, y en caso de que sea apretada arrojará el valor de 1; el canal "UP Btn" es la entrada B del canal "A*B" que da el usuario cuando aprieta la tecla de la letra W, la cual sirve para que el muñeco pueda caminar, así cuando la tecla W esté sin apretar arrojará el valor de 0, y en caso de que sea apretada arrojará el valor de 1. La expresión matemática que hace el canal "A*B", es la de multiplicación, como su nombre lo indica; esta expresión nos dice lo siguiente, cuando las teclas W y Q son apretadas al mismo tiempo, se hace la multiplicación de sus valores cuando están apretadas, el cual es 1, y ese valor indica que se hará la transición de la animación de caminar a la animación de correr del personaje. El círculo D contiene el canal de "Value Damping", el cual es el mismo que el del círculo E, pero ahora tiene un canal de expresión matemática ligado a él llamado "A||B||C||D"; tal y como su nombre indica, este último canal hace la expresión booleana de OR, lo cual indica que éste canal adquiere el valor de cualquiera de sus canales hijos(canales unidos a él por debajo de éste); las variables de entrada A,B,C y D del canal "A||B||C||D", son los valores arrojados cuando se aprietan las teclas W, S, D y A respectivamente.En el círculo B observamos que se encuentra otro canal de "Motionset Blender", el cual tiene la misma función que su igual del círculo E, sólo que ahora junto con los círculos C, E y D, hace la siguiente función: Cuando el personaje no tiene ningún estimulo por parte del usuario, éste solo se queda haciendo la animación de parado, pero si el usuario aprieta cualquiera de las teclas W, S, D, o A, se activa la animación de caminar, pero si el usuario usa la tecla W y Q al mismo tiempo, se activa la animación de correr.

Ahora veremos como activar esos lapsos de tiempo del canal "Timer Value" de las animaciones de parado, caminar y correr, para eso necesitamos ver la siguiente figura.

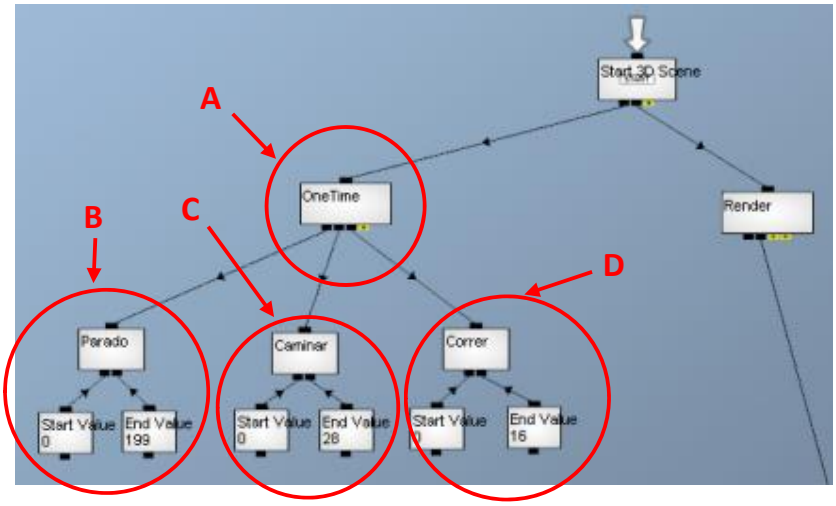


Figura 62. El arreglo de canales con canal padre llamado "One time", es el lugar donde iniciaremos los lapsos de tiempos.

Cuando un lapso de tiempo es creado, necesitamos inicializarlo para que la animación de nuestro personaje sea ejecutada. Para esto necesitamos un canal que mando los valores de inicio y final del lapso de tiempo a cada una de las animaciones que tiene nuestro personaje; estos canales serán los de nombre "Parado", "Caminar" y "Correr" de los círculos B, C y D respectivamente. El círculo B, tiene el canal de "Parado" el cual hace referencia al canal "Timer Value" que contiene el lapso de tiempo llamado "Parado" en la figura 61 en el círculo C; el canal "Parado" tiene dos canales hijos, el canal de de "Start Value" que sirve para indicarle al lapso del tiempo que empiece la animación desde el fotograma 0, y el canal "End Value" que le indica al lapso de tiempo que termine la animación hasta su fotograma 199; el canal "Parado" también tiene la propiedad de que podemos indicarle qué hacer cuando termine el lapso de tiempo, por lo tanto, se le da la propiedad de que comience la animación y que vuelva a repetirse el lapso de tiempo que la contiene indefinidamente. El círculo C tiene el canal de "Caminar" que hace referencia al canal hijo "Timer Value" del canal padre "Caminar" de la figura 61 en el círculo E; el canal "Caminar" se programa de la misma manera que el canal "Parado" del círculo B, solo que los valores de inicio y final de los fotogramas son 0 y 20 respectivamente. El círculo D tiene el canal de "Correr" que hace referencia al canal hijo "Timer Value" del canal padre "Correr" de la figura 61 en el círculo E; el canal "Correr" se programa de la misma manera que el canal "Parado" del círculo B, solo que los valores de inicio y final de los fotogramas son 0 y 16 respectivamente. El círculo A contiene el canal "One time", el cual su funcionamiento es el de ejecutar solo una vez sus canales hijos cuando éste canal es llamado, haciendo así que los lapsos del tiempo se ejecuten una vez y por sí solos se repitan indefinidamente. Con esto podremos hacer que nuestro personaje, al momento de iniciar el videojuego, se anime automáticamente sin ningún problema.

GRAVEDAD

Para poder simular la fuerza de atracción hacia el centro de nuestro mundo virtual, refiriéndonos a la fuerza de gravedad, se empleo un vector de posición con solo componente en Y, ya que en nuestro mundo virtual, el eje de las Y está perpendicular al plano ZX donde el personaje se para.

Para lograr este efecto de gravedad se genera el grupo de canales que muestra la siguiente figura.

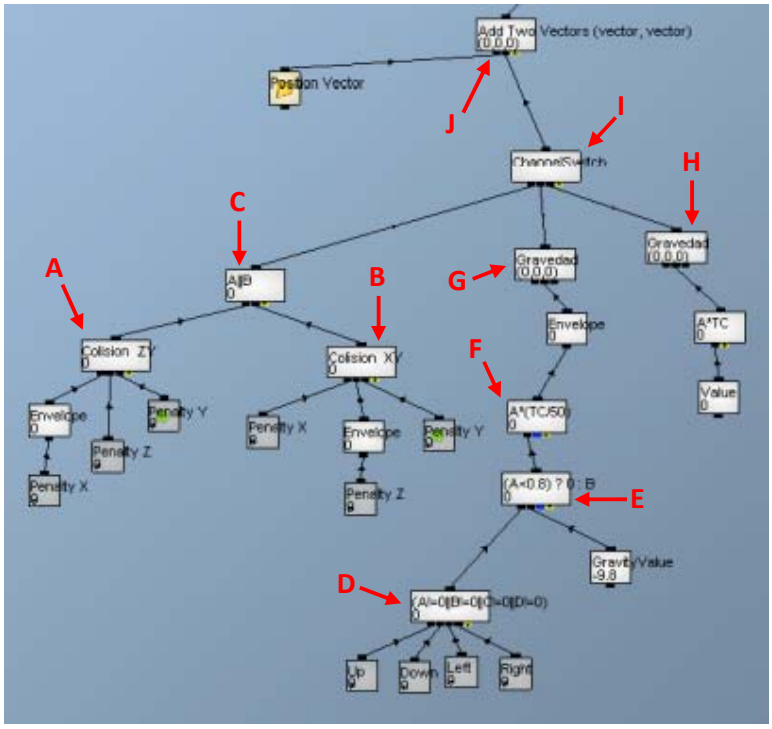


Figura 63. Grupo de canales hechos para la simulación de la gravedad.

En la figura 63 con el indicador A, es el canal "Colision ZY" que es un canal de expresión matemática, el cual efectúa el siguiente cálculo A=0&&B<>0&&C<>0, esto quiere decir que se efectúa una operación booleana de AND, y cuando A sea igual a 0, B sea diferente a 0 y C sea diferente a 0, este canal arrojará el valor de 1. Las entradas A, B, y C son canales de detección de colisiones sobre los ejes X, Z y Y respectivamente. Por lo cual podremos decir que este canal sirve para detectar una colisión sobre el plano ZY, ya que las entradas diferentes a 0 son las B y C, que corresponden a las colisiones en los ejes Z y Y.

En la figura 63:

- Con el indicador B, es el canal "Colision XY" que es un canal de expresión matemática, el cual efectúa el siguiente cálculo A<>0&&B=0&&C<>0, esto quiere decir que se efectúa una operación booleana de AND, y cuando A sea diferente a 0, B sea gual a 0 y C sea diferente a 0, este canal arrojará el valor de 1. Las entradas A, B, y C son canales de detección de colisiones sobre los ejes X, Z y Y respectivamente. Por lo cual podremos decir que este canal sirve para detectar una colisión sobre el plano XY, ya que las entradas diferentes a 0 son las A y C, que corresponden a las colisiones en los ejes X y Y.

- Con el indicador C, es el canal "A||B" que es un canal que hace la operación booleana de OR entre sus entradas A y B. Lo que nos quiere decir este canal es que su valor es 1 cuando cualquiera de sus entradas sea 1, o las dos juntas sean 1. Este canal sirve para indicarnos que ocurrió una colisión en los planos ZY o XY o en los dos a la vez.

- Con el indicador D, es el canal "(A!=0||B!=0||C!=0||D!=0)" que efectúa la operación matemática que su mismo nombre indica. Es una operación booleana de OR, donde la condición es cierta cuando cualquiera de sus cuatro entradas sea diferente a 0. Sus entradas A, B, C y D son canales que capturan las teclas W, S, A y D respectivamente, las cuales son teclas de control del personaje.

- Con el indicador E, es el canal "(A<0.8) ? 0 : B" que hace la siguiente operación, cuando su entrada A es menor que 0.8 toma el valor de 0, pero si su entrada A es mayor a 0.8 toma el valor de su entrada B, entonces podremos decir que cuando el usuario aprieta cualquier tecla de control del personaje, el valor del canal es de -9.81(entrada B), pero si el usuario no aprieta ninguna tecla el valor es de 0.

- Con el indicador F, es el canal "A*(TC/50)" que hace la operación matemática que su nombre indica. La finalidad de este canal es que en cada fotograma que pase durante el paseo virtual, se les apliquen los valores de la Entrada A.

- Con el indicador G, es el canal "Gravedad", el cual es un vector con solo componente en Y. Este vector en su componente en Y, tiene el valor de su canal hijo "Envelope", donde este canal atenúa el valor de su canal hijo multiplicándolo por un factor para que siempre sea -0.06 o 0. Ya con esto, el valor en Y del vector de gravedad es de -0.06 o 0. Este vector servirá para que el personaje cuando esté subiendo en las escaleras y se tenga que detener en ellas, no siga deslizándose hacia debajo de ellas, se mantenga en su lugar.

- Con el indicador H, es el canal de "Gravedad", que es otro vector con solo valor en su componente en Y, y este valor es de 0. Este vector sirve para simular que no hay gravedad, y poder subir las escaleras.

- Con el indicador I, es el canal "Channel Switch", el cual hace la función de ejecutar sus entradas B o C, dependiendo del valor de su entrada A. Si su entrada A es igual a 0, se ejecuta su entrada B, pero si A es diferente a 0, se ejecuta su entrada C. Por lo anterior podremos decir que, cuando no hay colisiones sobre los planos ZY o XY, el vector de gravedad en su componente en Y toma el valor de -0.06; pero si existe una colisión entre los planos ZY o XY, el vector de gravedad en su componente en Y es de 0, eso quiere decir que está subiendo las escaleras nuestro personaje.

- Con el indicador J, es el canal "Add Two Vector…" que es el canal que suma los vectores de posición del personaje junto con el vector de gravedad, así con esto podremos tener siempre un vector que se suma en la componente en Y del vector de posición del personaje para que éste siempre permanezca en el suelo.

COLISIONES

Las colisiones dentro del entorno virtual son parte fundamental de éste, ya que con ellas podemos evitar que nuestro personaje principal sea capaz de moverse sobre el suelo, pueda subir escaleras, pueda toparse con las paredes para que no las atraviese. Gracias a esto podemos sentir que en realidad las estructuras

existen. Para lograr este objetivo, Quest3D diferentes técnicas de colisiones, pero solo nos enfocaremos en una sola, que fue la que usamos. La explicación que daremos será muy superficial.

El canal (Fast Collision Response) que maneja Quest3D para generar las colisiones utiliza dos técnicas para lograr colisiones, una de ellas es la de Binary Space Partitioning Trees (Árboles Binarios de Partición Espacial) son sus siglas en Inglés BSP Trees. Los árboles BSP son usados para representar de manera sencilla los cuerpos sólidos dentro del espacio, donde cada nodo del árbol le corresponde un plano del sólido. La otra técnica que usa es la del Esferoide, la cual sirve para crear un esferoide alrededor de nuestro personaje con un radio dado, y así servirá para detectar la distancia que existe entre el esferoide y el plano a colisionar con él.

El funcionamiento de este canal es la de sustituir el vector de posición del personaje haciéndolo propio del esferoide, así este esferoide tendrá la misma posición que el personaje en todo momento. Los semiejes del esferoide se vuelven el radio de colisión del personaje. Cuando se va a generar una colisión, el canal hace una búsqueda rápida en los árboles BSP para ver cual es el plano con el cual colisionará, y dependiendo de la distancia entre el punto de origen del esferoide y el plano de colisión (con el sentido de su vector normal hacia el esferoide) será que el personaje detenga el incremento en su vector de posición ya sea sobre el eje X, Y o Z dependiendo sobre qué eje se de la colisión. La ventaja que ofrece usar esferas como una herramienta envolvente, es que las colisiones siempre serán calculadas respecta su centro y radio, y no como pasa con un cubo, que se tiene que calcular siempre la alineación de sus bordes con los ejes coordenados; adicionalmente, cuando una esfera rota, siguen habiendo cálculos de las colisiones respecto a su centro, mientras que en el cubo, se tiene que calcular de nuevo su rotación en sus 6 vértices mínimos y máximos.

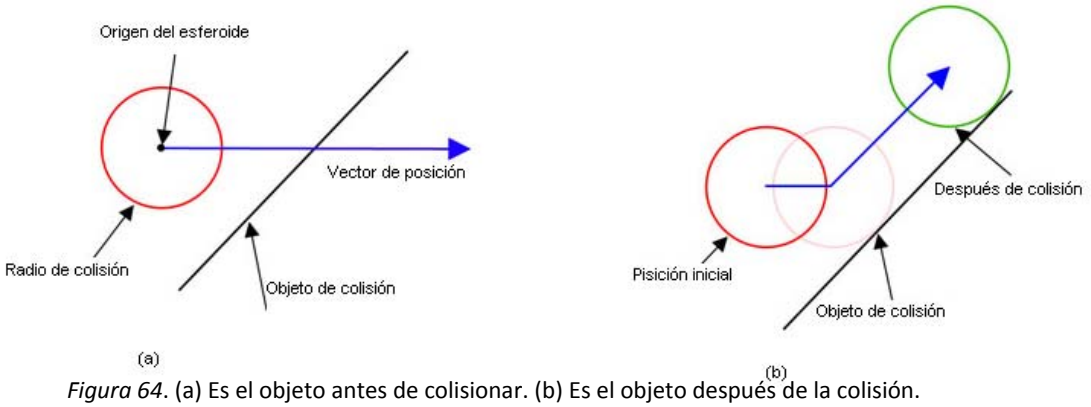


Figura 64. (a) Es el objeto antes de colisionar. (b) Es el objeto después de la colisión.

Árboles BSP

Cada nodo interno del árbol BSP esta asociado a un plano y tiene dos apuntadores a los hijos, uno para cada lado del plano. Asumiendo que las normales se encuentran apuntando fuera del objeto, el hijo izquierdo esta dentro o detrás del plano y el derecho se encuentra afuera o delante del plano. Si el espacio se vuelve a subdividir entonces ese hijo se convierte en la raíz de un subárbol, si el subespacio es homogéneo, entonces este hijo es una hoja, representando una región completamente dentro del sólido o

completamente fuera del sólido. Estas regiones homogéneas son llamadas celdas “adentro” y celdas “afuera”. Cada cara del sólido reside en un único plano, y en dos dimensiones un una única línea.

Para la construcción de los árboles se siguen los siguientes pasos. Suponiendo que las normales apuntan afuera del plano y viendo estos planos desde su vista horizontal (se ven como líneas):

Figura 65. Normales de los planos

Nos movemos a través de los planos en orden numérico. El plano 1 se vuelve el nodo padre; el plano 2 está adentro del plano 1. En consecuencia, después del plano 2, obtenemos el siguiente árbol BSP.

Figura 66. Árbol BSP

El plano 3 está parcialmente adentro del plano 1 y parcialmente afuera. Dividimos el plano 3 a lo largo de una línea contenida en el plano 1. Los planos resultantes son 3a y 3b. Ellos heredan el sentido de sus normales del plano 3.

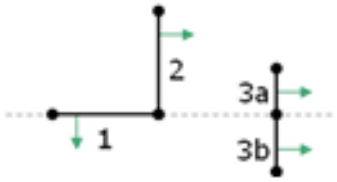


Figura 67. Sentido de las normales heredadas.

Colocamos en nuestro árbol a los planos 3a y 3b separadamente. El plano 3a está adentro del plano 1 y afuera del plano 2. El plano 3b está afuera del plano 1. Y el árbol BSP queda como:

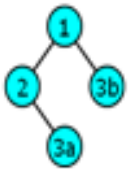
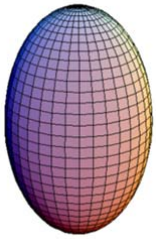


Figura 68. Arbol BSP final.

Esferoide

Está técnica se usa para crear una envoltura alrededor de nuestro personaje con forma de esferoide, y sirve para que se pueda detectar la distancia a la cual el personaje hará una colisión con el objeto en cuestión. El principio básico de funcionamiento es que la envoltura se genere en el punto medio del personaje, dando así el origen del esferoide, y dependiendo del tamaño de los ejes en X, Y y Z del esferoide, será el radio de colisión que tendrá el personaje.

Podemos empezar a dar un poco de teoría la respecto. Empezamos dar las ecuaciones del esferoide en coordenadas cartesianas centrado en el origen.



$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 + \left(\frac{z}{c}\right)^2 = 1 \quad \ldots\ldots\ldots\ldots\ldots (A)$$

Figura 69. Sentido de las normales heredadas

siendo a, b y c los semiejes. Estos semiejes pueden ser manipulados en Quest3D para que nosotros podamos de tal manera envolver a nuestro personaje. Estos semiejes se convierten en el radio de colisión en cada eje coordenado que el personaje necesita, y sobre todo, el centro del esferoide está colocado en el centro de nuestro personaje.

Para detectar la colisión entre el esferoide y el plano de colisión, se efectúa el cálculo de la distancia entre el origen del esferoide y el plano de colisión. La distancia se efectúa en el origen del esferoide ya que el radio de colisión es uno de sus semiejes.

Para calcular la distancia entre el punto P(x0,y0,z0) y el plano π, es la distancia menor que existe entre el punto y los puntos contenidos en el plano. La distancia corresponde a la perpendicular hecha desde el punto al plano.

$$P(x_0, y_0, z_0) \qquad \pi: Ax + By + Cz + D = 0$$

$$d(P,\pi) = \frac{|Ax_0 + By_0 + Cz_0 + D|}{\sqrt{A^2 + B^2 + C^2}} \quad \ldots\ldots\ldots\ldots\ldots (B)$$

La ecuación (B) es la distancia más corta entre el punto P y el plano π, entonces podremos decir que de la ecuación (A) y (B)

$a \geq d(P,\pi)$	*Existe colisión*	$b \geq d(P,\pi)$	*Existe colisión*	$c \geq d(P,\pi)$	*Existe colisión*
$a < d(P,\pi)$	*No existe colisión*	$b < d(P,\pi)$	*Existe colisión*	$c < d(P,\pi)$	*Existe colisión*

Estas relaciones entre los semiejes (radio de colisión) del esferoide y la distancia entre el centro del esferoide y el plano de colisión, sirve para todos los planos XY, YZ del ZX del espacio virtual.Una vez dado un poco de teoría procederemos a explicar el grupo de canales que se elaboró para efectuar las colisiones.

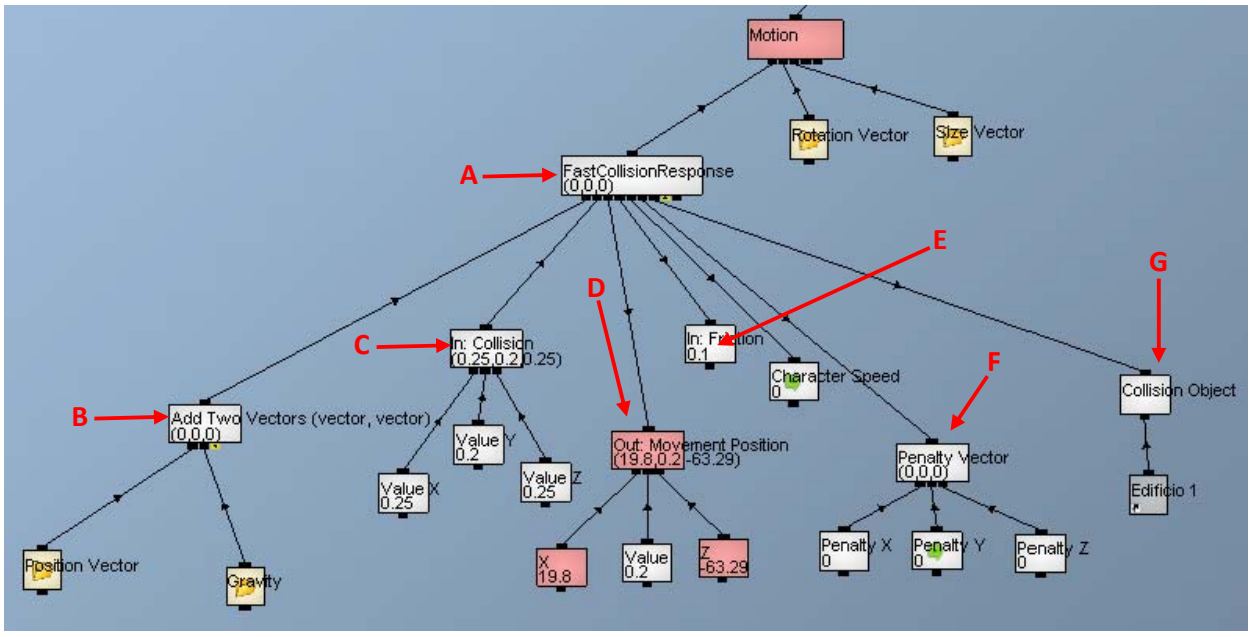


Figura 70. Canales hechos para simular las colisiones entre la estructura y el personaje

De la figura 70:

- Con el indicador A, se muestra el canal “Fast Collision Reponse”, el cual efectúa, junto con las técnicas de árboles BSP y esferoide, las colisiones de nuestro personaje con su entorno virtual. Este canal va conectado a la entrada de vector de movimiento de la matriz de movimiento del personaje, y es porque las colisiones se efectuarán sobre las posiciones que tenga el personaje en todo el paseo.

- Con el indicador B, se puede observar el canal que efectúa la adición de el vectore de posición del personaje y el vector de la gravedad, y este canal es hijo del canal de colisiones.

- Con el indicador C, se puede observar el canal “In: Collision”, que es un canal que alberga un vector el cual servirá para determinar la longitud de los semiejes del esferoide en sus ejes X, Y y Z, en otras palabras, para determinar el radio de colisión del esferoide. Los hijos de este canal, los cuales son “Value X”, “Value Y” y “Value Z”, son los valores del radio de colisión sobre los ejes X, Y y Z del personaje respectivamente.

- Con el indicador D, se muestra un canal que maneja la posición en la que se encuentra nuestro personaje dentro del espacio virtual, esta posición la maneja como un vector.

- Con el indicador E, es un canal que indica la fricción con la se maneja las colisiones, esto es, cuando existe una colisión entre el personaje y un objeto, qué tan rápido se mueve el personaje sobre el objeto.

- Con el indicador F, se muestra un canal que contiene el vector que se genera para saber cuánto fue desplazado el personaje durante una colisión, es quiere decir que cuando hay una colisión, sobre cualquier eje coordenado, este canal registra en un vector la cantidad de desplazamiento que se generó cuando el personaje chocó con un objeto.

- Con el indicador G, se encuentra el canal “Collision object”. Este canal genera los árboles BSP del objeto en 3D que tiene como hijo. Gracias a este característica que tiene el canal de colisión, es más rápido, ya que el canal de “Collision object” hace más rápida la búsqueda de los planos de colisión. Se pueden generar más de estos canales, ya que cada canal de “Collision object” deberá de tener como hijo a cada una de las estructuras de nuestra arquitectura virtual, así que, se tendrá un canal por cada estructura sólida que exista en nuestro paseo virtual, lo cual hace que se generen muchísimos canales unidos al canal de colisión. Por cuestiones didácticas, solo se explico con una sola estructura, pero en realidad hay más de cien canales de este tipo unidos al canal de colisiones.

5.3.4.3.- Controles y manejo de cámara

CONTROLES

Ya tenemos nuestra animación de nuestro personaje, pero ahora lo que queremos es asignarle los controles para que nuestro personaje principal pueda pasear a través de toda la zona arqueológica, y también hacer el grupo de canales que nos permitan hacer mover el personaje.

Antes de comenzar a explicar los grupos de canales que se efectuaron para elaborando para hacer moverse al personaje dentro del espacio virtual, necesitamos decir cuáles fueron las teclas que el usuario debe oprimir para lograr ese cometido.

Tecla "W" → Caminar hacia adelante.

Tecla "S" → Caminar hacia atrás.

Tecla "A" → Movimiento lateral izquierdo.

Tecla "D" → Movimiento lateral derecho.

Tecla "Q" → Correr. Esta tecla deberé ser apretada junto con cualquiera de las teclas anteriores.

Mouse → Rotación del personaje sobre su propio eje.

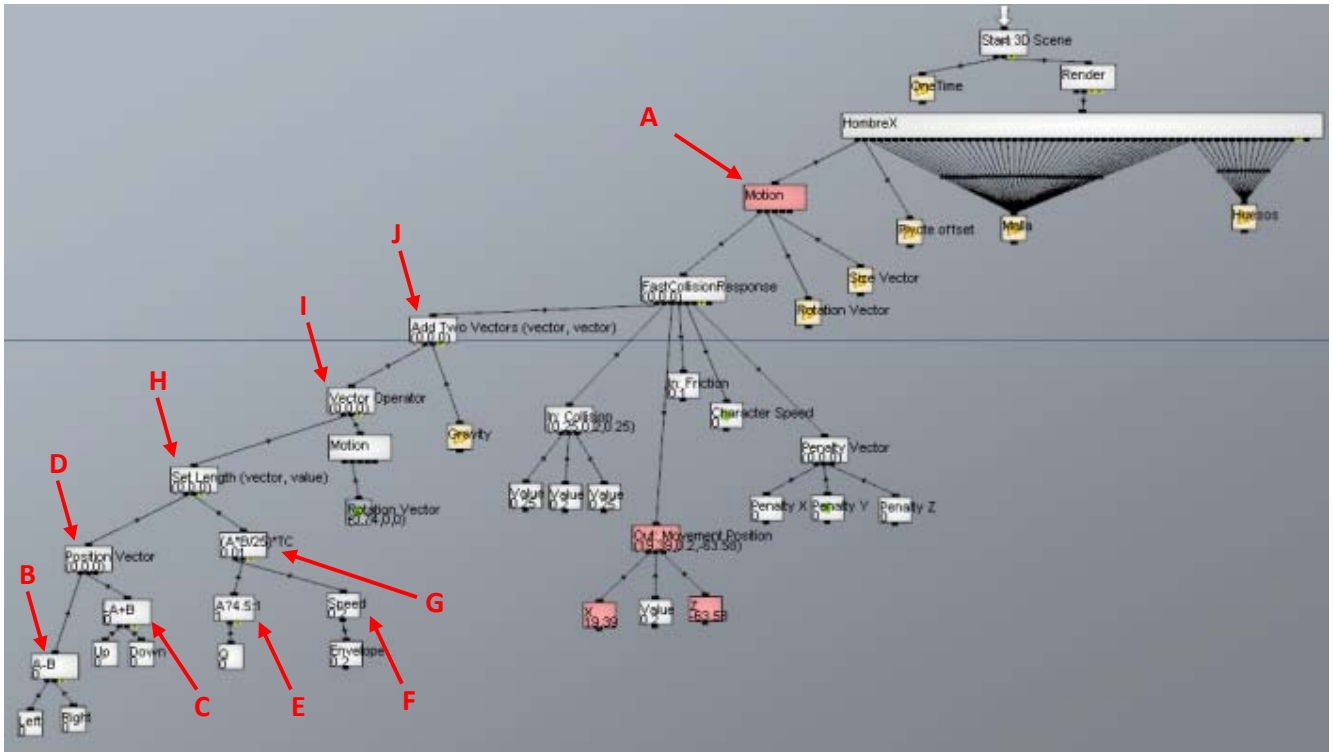


Figura 71. Grupo de canales hechos para generar el cambio de posición de nuestro personaje dentro del entorno virtual.

Para comenzar a trasladar a un stro personaje principal dentro de nuestro entorno virtual, necesitaremos modificar sus transformaciones geométricas de posición y de rotación de la matriz de movimiento (canal "Motion") de nuestro personaje, la cual se encuentra dentro de la figura FF en la indicación A. Podremos observar que uno de los canales hijos de nuestro canal "Motion" tiene el nombre de "FastCollisionResponse" que ocupa el lugar del vector de posición del personaje, y esto es porque ese canal sirve para controlar las colisiones de nuestro personaje a través del entorno virtual; ese canal se explicará más adelante, por el momento solo nos enfocaremos a las operaciones sobre el vector de posición de nuestro personaje.

Modificación del vector de posición del personaje

De la figura 71:

Con el indicador B, se encuentra un canal de expresión matemática "A-B", el cual hace la operación de resta de sus dos nodos hijos "Left" y "Right". Cada nodo hijo es un canal que captura la entrada del usuario en el teclado, lo cual quiere decir que cuando apretamos la tecla "A", el canal "Left" adquiere el valor de 1, y si apretamos la tecla "D" el canal "Right" adquiere el valor de 1. El canal "A-B" lo que hace es que cuando apretamos la tecla "A", éste adquiere el valor de 1, lo cual quiere decir que sobre el eje de las X de nuestro sistema de coordenadas del paseo virtual, avanzará una unidad; ahora si es apretada la tecla "D", el canal "A-B" adquiere el valor de -1, lo cual quiere decir que nuestro personaje retrocederá una unidad sobre el eje de las X.

Con el indicador C, se encuentra un canal de expresión matemática "-A+B", el cual hace la operación de resta de sus dos nodos hijos "Up" y "Down". Cada nodo hijo es un canal que captura la entrada del usuario en el teclado, lo cual quiere decir que cuando apretamos la tecla "W", el canal "Up" adquiere el valor de 1, y si apretamos la tecla "S" el canal "Down" adquiere el valor de 1. El canal "-A+B" lo que hace es que cuando apretamos la tecla "W", éste adquiere el valor de -1, lo cual quiere decir que sobre el eje de las Y de nuestro sistema de coordenadas del paseo virtual, retrocederá una unidad; ahora si es apretada la tecla "S", el canal "-A+B" adquiere el valor de 1, lo cual quiere decir que nuestro personaje avanzará una unidad sobre el eje de las Y.

Con el indicador D, se encuentra el vector de posición de nuestro personaje, donde en sus entradas de las componentes X y Y, se encuentran los canales hijos "A+B" y "-A+B" respectivamente; esto quiere decir que, el personaje principal se desplazará sobre el plano XY las unidades que se le indiquen por estos dos canales hijos.

- Con el indicador E, se encuentra el canal de expresiones matemáticas "A?4.5:1" y en uno de sus hijos se encuentra el canal "Q", el cual es un canal de captura de entrada del usuario en el teclado, esto es que cuando apretamos la tecla "Q", el canal "Q" adquiere el valor de 1. El canal "A?4.5:1" es una expresión que nos indica que, cuando la tecla "Q" es apretada (entrada A del canal "A?4.5:1" con valor 1), el canal "A?4.5:1" adquiere el valor de 4.5, y si la tecla "Q" no es apretada, el valor del canal "A?4.5:1" es de 1. Con este canal podemos aumentar la rapidez del personaje para hacer el efecto que ya va corriendo.

- Con el indicador F, es el canal "Speed" con el cual generamos un valor que se usará como factor para controlar la rapidez con la cual camina o corre nuestro personaje.

- En la figura FF con el indicador G, es un canal de expresión matemática llamado "(A*B/25)*TC" y sus hijos los canales "Speed" y "A?4.5:1" en las entradas A y B respectivamente. La operación matemática del canal"(A*B/25)*TC" es la misma que su nombre, en el cual la variable "TC" es el contador del actual fotograma; los fotogramas son ejecutados a la razón de 25 por segundo. La variable B está dividida por 25, lo cual quiere decir que a cada fotograma que se ejecute en un segundo, se le asignará una veinticincoava parte de la rapidez total que se le asigna a un 1 seg. El canal "(A*B/25)*TC" quiere decir que, cuando la tecla "Q" sea presionada, el valor de B incrementará, multiplicándose por el factor de rapidez que se le asigna al fotograma que se ejecute en ese momento, y así para todos los fotogramas que transcurran en todo el tiempo del paseo virtual.

- Con el indicador H. se observa el canal "Set Length…". Este canal sirve para establecer la magnitud de un vector. En este caso es para incrementar el vector de posición del personaje multiplicándolo por el escalar que nos da el canal "(A*B/25)*TC" que es la magnitud de la rapidez para cada fotograma. Con esto podremos decir que ya podemos pasar de un movimiento de caminata a un movimiento de correr.

- Con el indicador I, se observa el canal "Vector Operator" el cual efectúa la multiplicación del vector de movimiento ya modificado del personaje, con una matriz de movimiento que tiene solamente componentes en su vector de rotación, donde ese vector de rotación corresponde al del personaje que servirá para hacer girarlo sobre su propio eje. En el canal "Vector Operator" se efectúa la transformación del vector de posición del personaje cuando el personaje gira en cualquier momento, esto es, cuando el personaje vaya caminando, éste pueda girar hacia donde lo indiquemos.

- Con el indicador J, corresponde al canal "Add Two Vector…", el cual sirve para sumar dos vectores, uno de ellos es el vector de posición del personaje que ya contiene la rotación del mismo, y el otro vector es el empleamos para simular que nuestro personaje está bajo las leyes de gravedad de nuestro entorno virtual. El vector que simula la gravedad se verá más adelante.

Modificación del vector de rotación del personaje

Teniendo el desplazamiento del personaje, tendremos ahora que hacer la rotación del personaje con la finalidad de que se efectúe el movimiento de giro de éste.

Para lograr esto, tendremos que modificar el vector de rotación de la matriz de movimiento del personaje. El grupo de canales para la rotación queda como se muestra a continuación.

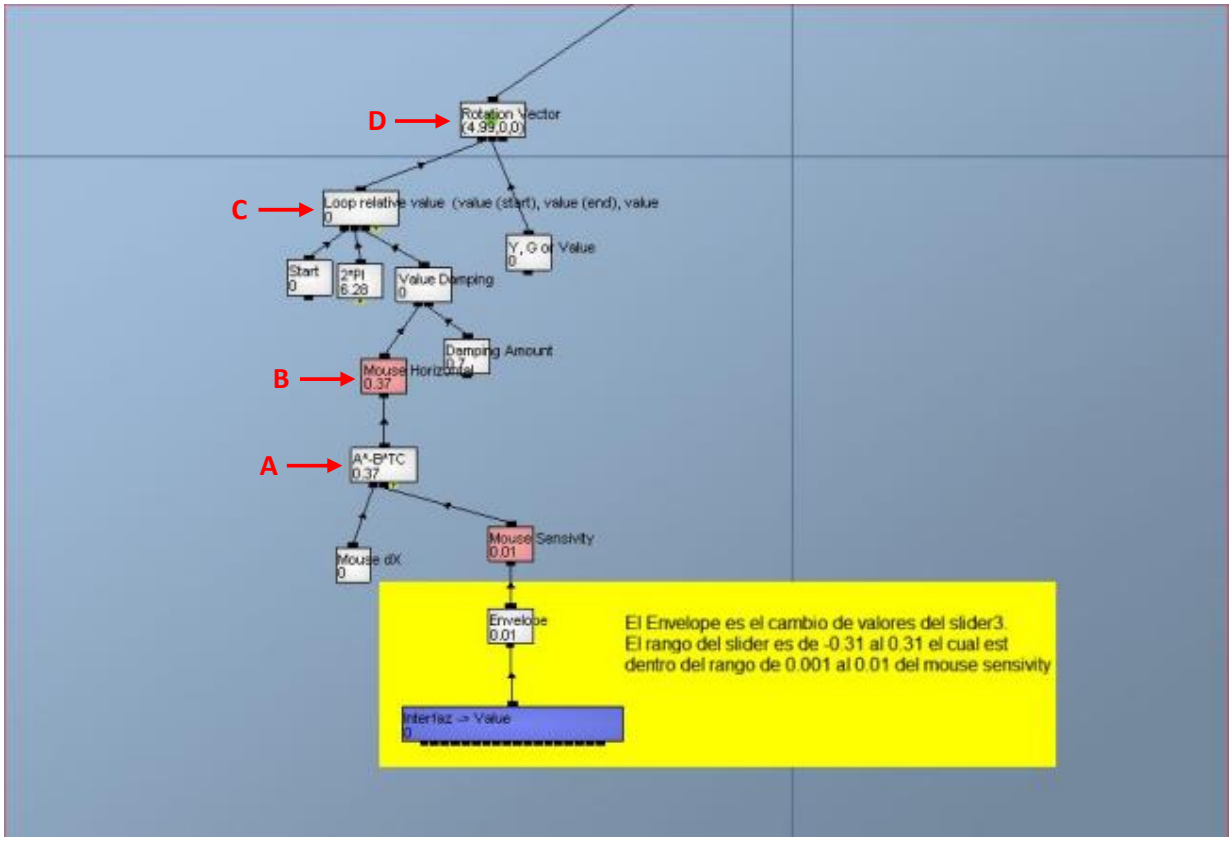


Figura 72. Grupo de canales hechos para generar la rotación de nuestro personaje.

Da la figura 72:

- Podremos apreciar visualmente que existe un rectángulo color amarillo, donde se encuentran dos canales dentro de él, por el momento esos dos canales no nos interesan ya que forman parte de otro subtema.

- Con el indicador A, se encuentra un canal de expresión matemática llamado "A*-B*TC" y la operación matemática que efectúa es la que su nombre indica. Éste canal tiene dos canales hijos, el primero es el canal "Mouse dX", el cual adquiere el valor de 1 cuando existe movimiento del Mouse sobre su eje de las X; y el segundo canal hijo es el de "Mouse Sensivity", el cual es solo un valor constante y contiene el factor que afectará la magnitud de giro del personaje. El canal "A*-B*TC" nos indica que cuando hay un movimiento (Entrada A) por parte del Mouse sobre su eje de las X, éste será multiplicado por una constante (nótese que es de valor negativa, ya que el sentido de movimiento horizontal del Mouse es inverso al sentido de nuestro sistema de coordenadas. Es la entrada B), la cual sirve para regular la magnitud con la que dará la rotación nuestro personaje para todos los fotogramas (variable TC) que pasen durante nuestro paseo.

- Con el indicador B, es el canal donde solo guardamos el valor de la magnitud con la cual dará el giro nuestro personaje, por eso le decidimos llamar "Mouse Horizontal".

- Con el indicador C, tenemos el canal "Loop Relative Value…" el cual efectúa el incremento o decremento de un valor que está limitado por dos valores. Este canal servirá para controlar la rapidez con la cual dará el giro nuestro personaje y funciona de la siguiente manera. El primer canal hijo es el valor de inicio del giro, el cual se puede decir que empieza en 0º; el segundo canal hijo es el valor final del giro, el cual tiene un valor de 2*π que indica que su giro termina en 360º; y por último su tercer canal hijo es un valor que dicta el tamaño del incremento desde el valor inicial hasta el valor final. Con estos tres valores, podemos decir que la rotación del personaje sobre su eje será continuo y podrá dar una vuelta completa de 360º y seguir haciéndolo.

- Con el indicador D, tenemos al vector de rotación del personaje, donde el canal "Loop Relative Value…" está unido a la componente en X de este vector, lo cual quiere decir que el personaje dará su rotación sobre su eje de las X.

CÁMARA

En esta sección veremos cómo es que asignamos una cámara al personaje. Esta cámara tiene las características que sigue al personaje a una distancia considerable para que se pueda ver tanto el personaje como el paisaje, como sigue al personaje en todo momento, también la cámara hace su rotación respecto a él, ya sea de modo horizontal como de forma vertical. Como esta cámara sigue en todo momento al personaje, es una cámara "tonta", lo cual quiere decir que no tiene ninguna instrucción de reacción ante su entorno.

Para incorporar una cámara hacia el personaje se hace el grupo de canales que se muestran en la siguiente figura.

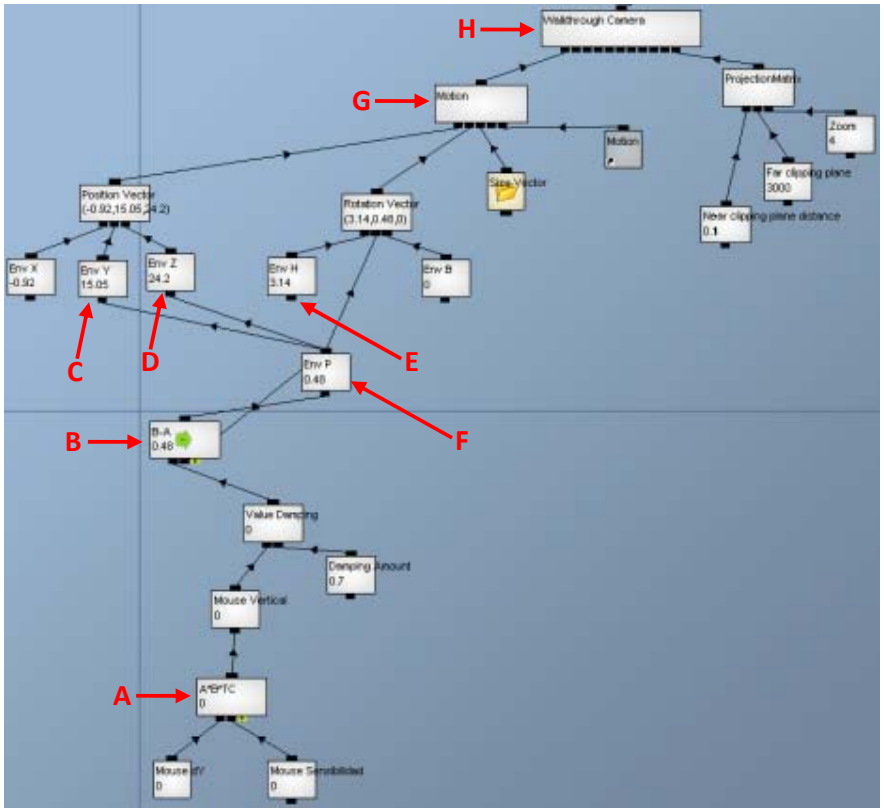


Figura 73. Grupo de canales hechos para asignarle una cámara a nuestro personaje.

De la figura 73:

- Con el indicador A, es el canal "A*B*TC" el cual hace la operación matemática que su nombre indica. La finalidad de este canal es que cuando haya un movimiento por parte del mouse sobre su eje Y (entrada A), se multiplique por un factor (entrada B) que atenúe la magnitud del movimiento vertical del mouse para todos los fotogramas que transcurren en la duración del paseo.

- Con el indicador B, es el canal "B-A" que hace la operación matemática que su mismo nombre indica. La finalidad de este canal es que se permita hacer un movimiento suavizado tanto vertical como horizontal de la cámara en todo momento, y así conseguir un movimiento completo. Su entrada A es el aumento suavizado del valor del movimiento vertical del mouse; su entrada B es la retroalimentación del valor de salida de este mismo canal, lo cual permite que se genere un movimiento completo de la cámara.

- Con el indicador C, se muestra la componente Y del vector de posición de la cámara. Esta componente en Y, controlará la rotación de la cámara respecto al personaje sobre el eje Y del sistema de coordenadas del paseo virtual; este control dará la magnitud del radio de rotación de la cámara y hasta cuantos grados podrá hacerlo.

- Con el indicador D, se muestra la componente Z del vector de posición de la cámara. Esta componente en Z, controlará la rotación de la cámara respecto a su propio eje en Z del sistema de coordenadas del paseo virtual; este control dará los grados que puede rotar sobre su propio eje.

- Con el indicador E, se muestra la componente X del vector de rotación de la cámara. Esta componente en X tiene el valor de π que es 3.1416 aproximadamente; este valor se lo dimos porque necesitábamos darle un giro sobre su eje de las X de 180°.

- Con el indicador F, se muestra la componente Y del vector de rotación de la cámara. Esta componente en Y contiene los valores que se encuentran entre -1 y 1, lo cual hace que la cámara rote sobre su propio eje Y.

- Con el indicador G, se muestra el canal "Motion" el cual contiene la matriz de movimiento de la cámara, esto incluye los vectores de posición, rotación y escalamiento de ella; pero también, observamos que tiene un canal hijo llamado "Motion", el cual es un canal que hace referencia a la matriz de movimiento del personaje. Lo que sucede cuando se le adjunta la matriz de movimiento del personaje a la matriz de movimiento de la cámara, lo que hace es que se genera una animación jerárquica, lo que significa que los movimientos de la cámara heredan los movimientos del personaje, esto quiere decir que los movimientos de la cámara serán respecto a los movimientos del personaje.

- Con el indicador H, se muestra el canal "Walkthrough camara" el cual es el canal que genera en el espacio una cámara. En este canal podremos hacer un grupo de canales que le digan que hacer en su entorno virtual.

5.3.4.4.-Desarrollo del mapa y sonido

Generación de mapa con localización en tiempo real.

Es un hecho, que en muchos juegos de videos que implican navegaciones en grandes extensiones, requieren de una referencia en tiempo real de su posición dentro del mundo virtual que contiene al personaje principal.

Para fines específicos de esta tesis, se genera una ecuación lineal que implica la relación de unidades de longitud del mundo virtual que contiene al personaje principal, con las unidades de medida de una imagen que contendrá al mapa.

Antes de empezar a hacer la ecuación de relación, cabe mencionar lo siguiente. La unidad de medida de una imagen que usaremos es el *pixel*, ya que para poder visualizar la posición del personaje principal, necesitaremos un desplazamiento sobre la imagen. Respecto a las unidades de longitud del mundo virtual, serán tomadas del sistema coordenado que por default viene en el software (Quest3D) que se maneja para elaborar el paseo virtual; éstas unidades de longitud serán tomadas como referencia absoluta.

Para poder hacer la relación entre el mundo real y el mapa (entiéndase como 'mapa' a la imagen que lo contendrá), se necesita crear un plano en la base de toda la zona arqueológica. Con la longitud del plano haremos la relación de los píxeles del mapa.

Para hacer esta relación entre el mapa y el plano del mundo virtual, empezaremos haciendo una regla de tres, como a continuación se presenta

Si U es la longitud máxima de uno de sus lados del plano del mundo virtual, y W es la longitud máxima en pixeles del mapa de uno de sus lados, y V es la longitud avanzada del personaje dentro del mundo real, entonces Q serán los pixeles avanzados dentro del mapa:

$$Q = V\frac{W}{U} \qquad \text{..... (A)}$$

La ecuación (A) es la relación píxeles del mapa respecto a las unidades avanzadas en el mundo virtual. Esta ecuación la podemos adaptar para relacionar las unidades avanzadas en un eje coordenado del mapa, con las unidades avanzadas en un eje coordenado del plano.

$$X_M = X_P\frac{W_X}{U_X} \qquad \text{..... (B)}$$

$$Y_M = Y_P\frac{W_Y}{U_Y} \qquad \text{..... (C)}$$

Donde:

X_M , Y_M = Unidades a avanzar en píxeles dentro del mapa en sus ejes coordenados X y Y respectivamente.

X_P , Y_P = Unidades avanzadas dentro del plano en sus ejes coordenados X y Y respectivamente.

W_X , W_Y = Longitud máxima en píxeles del mapa en sus ejes coordenados X y Y respectivamente.

U_X , U_Y = Longitud máxima del plano en sus ejes coordenados X y Y respectivamente.

Las ecuaciones anteriores (B) y (C) están representadas gráficamente en las figuras 74 y 75 respectivamente, como se muestra a continuación:

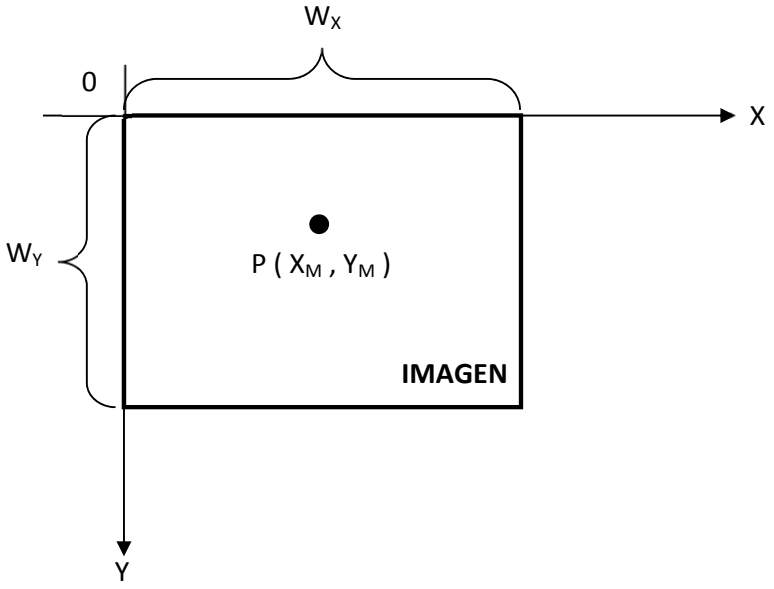


Figura 74. Sistema de ejes coordenados para el mapa.

En la figura 74 podemos observar que el sistema coordenado para una imagen, en este caso el mapa, los ejes de las abscisas se encuentra en sentido inverso al convencional, por lo cual los valores positivos del eje Y serán en sentido contrario al convencional.

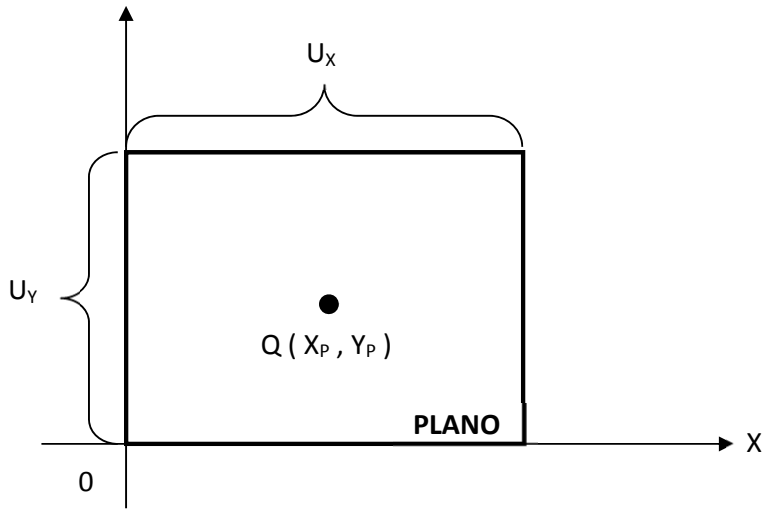


Figura 75. Sistema de ejes coordenados para el plano.

En la figura 75 podemos observar el plano que se hizo en la base de la zona arqueológica en el mundo virtual. Sus sistema coordenado es el convencional.

La generación de un objetivo dentro del mapa es muy importante, ya que éste nos dará la ubicación exacta del personaje dentro de la zona virtual en tiempo real y dentro del mapa.

Para cada pixel dentro del mapa, será representado por el siguiente vector que muestra la siguiente figura:

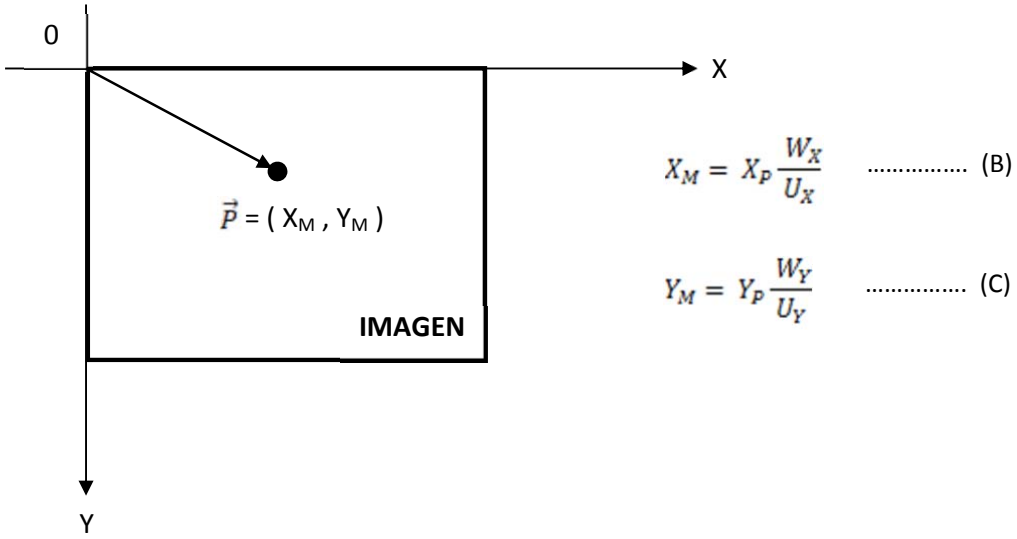


Figura 76. Representación de un pixel a través de un vector dentro del mapa.

Como el objetivo es el que nos indicará el lugar donde se encuentra el personaje dentro del mapa, éste será hecho por una imagen similar a la del mapa.

Para fines de esta tesis, cuando el personaje se encuentre en el centro del plano del mundo virtual, el objetivo tendrá que iniciar desde el centro del mapa. El centro del plano será ubicado en el origen del sistema de coordenadas del mundo virtual Q (0,0).

Para poder ubicar el objetivo en el centro del mapa, necesito ver las fórmulas (B) y (C), las cuales nos indican que, si el personaje se ubica en el origen del mundo virtual, el cual será el centro del plano, el objetivo que representa al personaje en el mapa estará en el origen de éste. Por ejemplo:

$$X_P = 0$$

$$Y_P = 0$$

Este ejemplo, gráficamente nos indica:

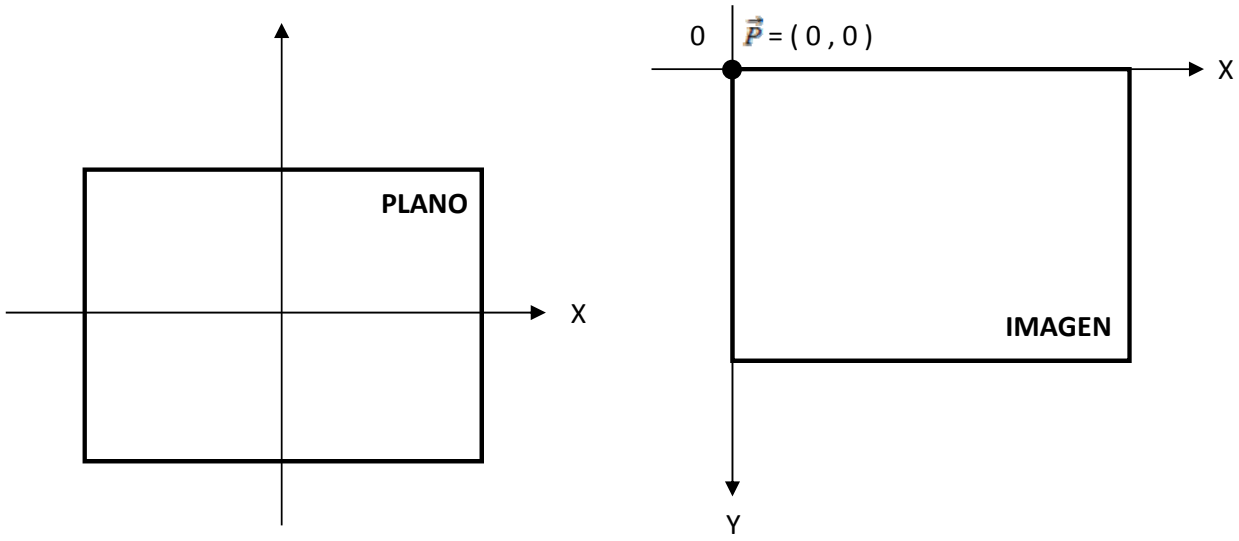


Figura 77. El objetivo se encuentra en el origen del mapa.

Como podemos apreciar en la figura 77, el objetivo será ubicado en el origen del mapa cuando el personaje se encuentre en el centro del plano del mundo virtual.

Para poder ubicar al objetivo en el centro del mapa cuando el personaje se encuentre en el origen del plano, debemos de sumar un vector que contenga el punto central del mapa con un vector que contenga la posición inicial del objetivo en el origen del mapa. Entonces tenemos lo siguiente:

Sea **F** un vector que contiene el punto central del mapa F (X_F , Y_F), y sea **P** el vector que contiene la posición original del objetivo P (X_M , Y_M), entonces la suma de ambos vectores dará como resultado el vector **G** de posición inicial del objetivo dentro del mapa G (G_X , G_Y). Entonces:

$$\vec{G} = \vec{P} + \vec{F}$$

$$\vec{G} = (X_M + X_F , Y_M + Y_F) \quad \text{..... (D)}$$

Sustituyendo (B) y (C) en (D):

$$\vec{G} = (X_p \frac{W_X}{U_X} + X_F , Y_p \frac{W_Y}{U_Y} + Y_F) \quad \text{..... (E)}$$

$$G_X = X_p \frac{W_X}{U_X} + X_F \quad \text{..... (F)}$$

$$G_Y = Y_p \frac{W_Y}{U_Y} + Y_F \quad \text{..... (G)}$$

Donde $\mathbf{G_X}$ y $\mathbf{G_Y}$ son las coordenadas en los ejes X y Y respectivamente de la nueva posición inicial del objetivo desde el centro del mapa.

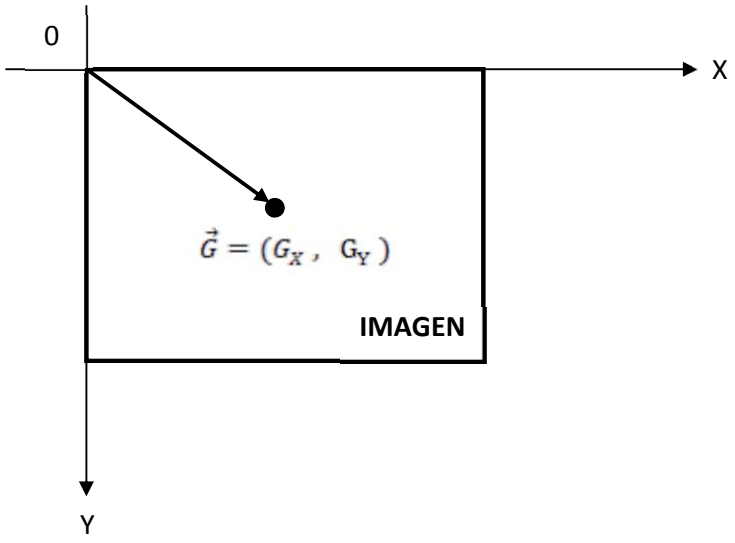


Figura 78. El objetivo se encuentra en el centro del mapa cuando el personaje se encuentra en el origen del plano.

Cuando el objetivo se posiciona en el centro del mapa, ya en la práctica, podemos observar que solo el origen del objetivo es el que se ha posicionado en el centro del mapa.

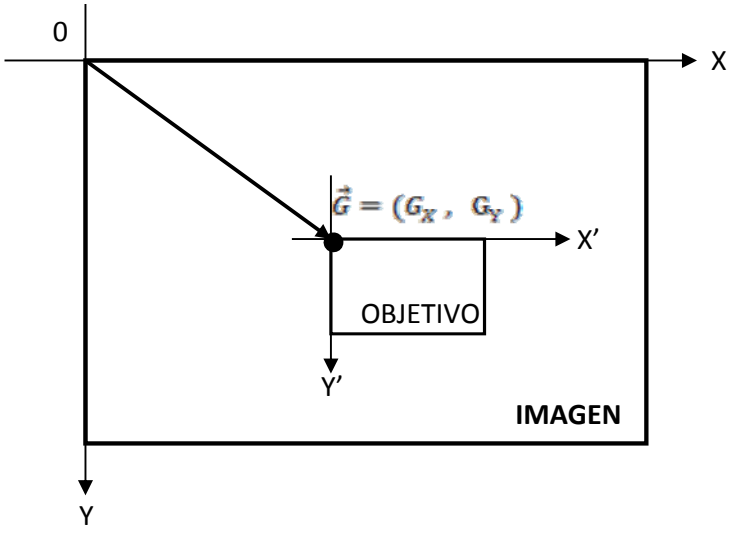


Figura 79. La posición del origen del objetivo, se encuentra en el centro del mapa

Por efectos de estética, el centro del objetivo debe de coincidir con el centro del mapa, pero como se muestra en la figura 79, la ecuación no empata los centros del mapa y el objetivo. Por tal motivo, se ha tomado un vector que contenga el centro del objetivo y que inicie desde su origen para poder realizar el empate de los centros.

Sea **L** un vector que parte desde el origen del objetivo y que contiene su punto central L (L_X, L_Y), y sea **G** el vector que contiene el punto central del mapa G (G_X , G_Y), entonces la resta de ambos vectores dará como resultado el vector **H** (H_X , H_Y) resultado del empate del centro del mapa y el centro del mapa. Entonces:

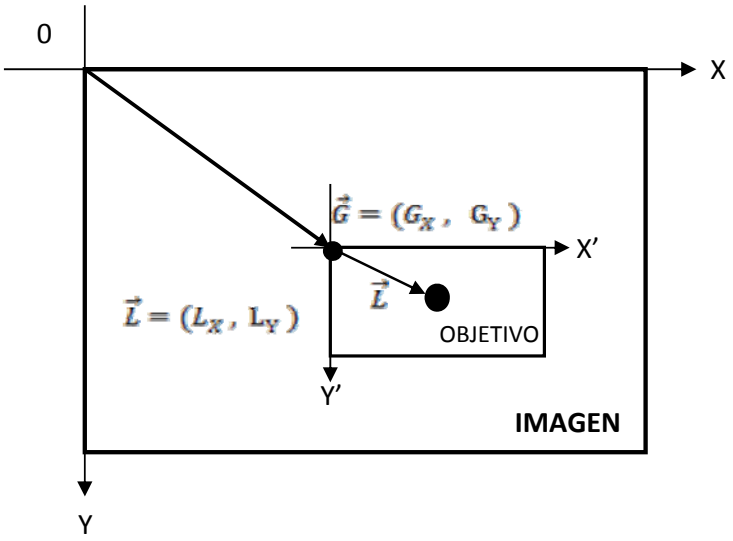


Figura 80. Se muestra el vector L que contiene origen y centro del objetivo

$$\vec{H} = \vec{G} - \vec{L}$$

$$\vec{H} = (G_X - L_X, G_Y - L_Y) \quad \text{..... (I)}$$

$$\vec{H} = \left(X_P \frac{W_X}{U_X} + X_F - L_X, Y_P \frac{W_Y}{U_Y} + Y_F - L_Y\right) \quad \text{..... (J)}$$

$$H_X = X_P \frac{W_X}{U_X} + X_F - L_X \quad \text{..... (K)}$$

$$H_Y = Y_P \frac{W_Y}{U_Y} + Y_F - L_Y \quad \text{..... (M)}$$

El empate del centro del objetivo con el centro del mapa en los ejes coordenados X y Y de este último, está dado por las ecuaciones (K) y (M) respectivamente. ***El vector resultante H inicia desde el origen del mapa terminando hasta el origen del objetivo***. Con estas ecuaciones podemos obtener la regla de correspondencia entre el mundo virtual y el objetivo que se encuentra en el mapa

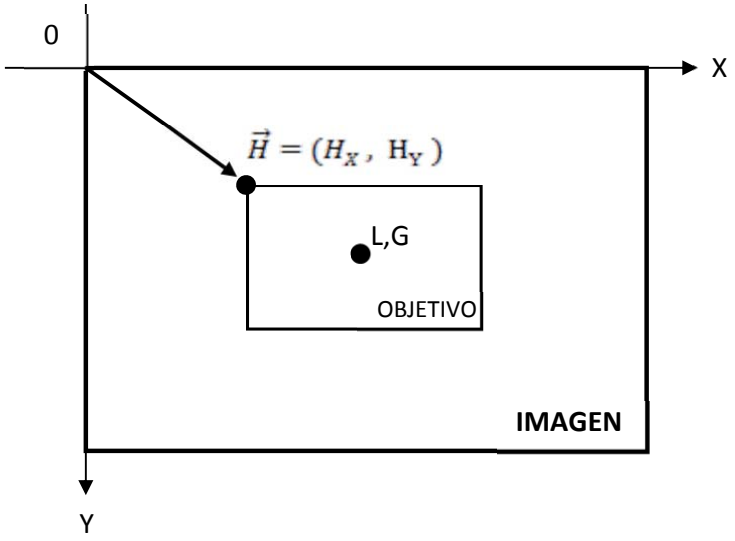


Figura 81. El vector H muestra el correcto empate del centro del objetivo y el centro de la imagen.

Las ecuaciones de correspondencia que te posicionan el centro del objetivo dentro del mapa relacionando los ejes coordenados de este último con el del mundo virtual son las siguientes, ***y solo se puede usar directamente cuando el centro del plano esta en el origen del sistema del mundo virtual y la posición del personaje se encuentre en este mismo origen:***

$$H_X = X_P \frac{W_X}{U_X} + X_F - L_X \qquad \text{..... (K)}$$

$$H_Y = Y_P \frac{W_Y}{U_Y} + Y_F - L_Y \qquad \text{..... (M)}$$

Dónde:

H_X , H_Y = Desplazamiento del origen del objetivo dentro del mapa en los ejes X y Y respectivamente.

X_P, Y_P = Cantidad de desplazamiento del personaje dentro del mundo real sobre sus ejes coordenados X y Y respectivamente.

W_X , W_y = Longitud máxima de los lados del mapa en los ejes coordenados X y Y respectivamente.

U_X , U_Y = Longitud máxima de los lados del plano que contiene a toda la zona arqueológica en los ejes coordenados X y Y respectivamente.

X_F , Y_F = Posición del centro del mapa.

L_X , L_Y = Posición del centro del objetivo desde el origen de éste mismo.

W_X / U_X , W_Y / U_Y = Es el factor de relación entre la unidades del plano y los pixeles del mapa

Creación de nuestro mapa

Una vez obtenida la ecuación de la regla de correspondencia, debemos de crear nuestro propio mapa.

El primer paso para la creación del mapa, debemos de hacer un plano en la base de la zona de edificios del mundo virtual, sacar la longitud de los lados del plano, obtener el vector de su centro y posicionar el personaje en este mismo.

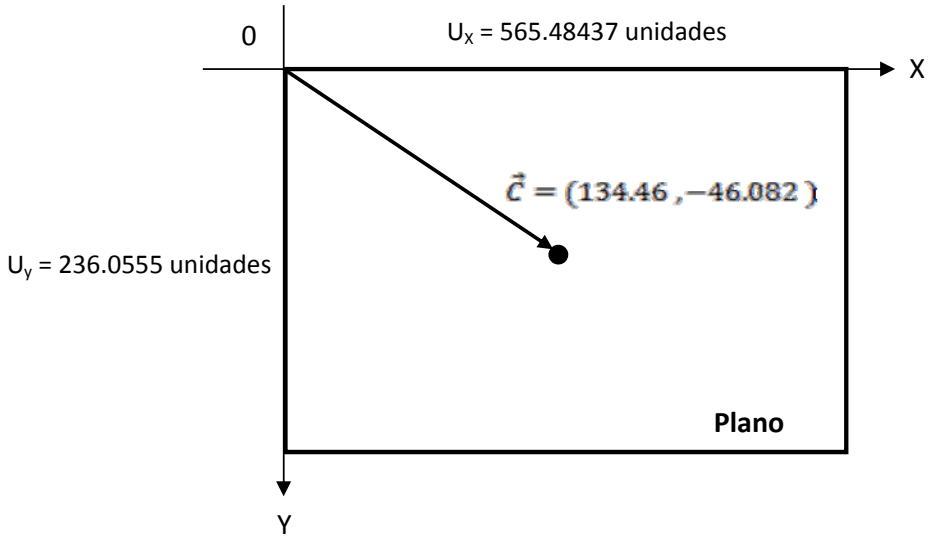


Figura 82. Plano en la base de la zona arqueológica

Ahora tenemos que hacer la imagen del mapa, la cual es una imagen de 256 x 256 píxeles

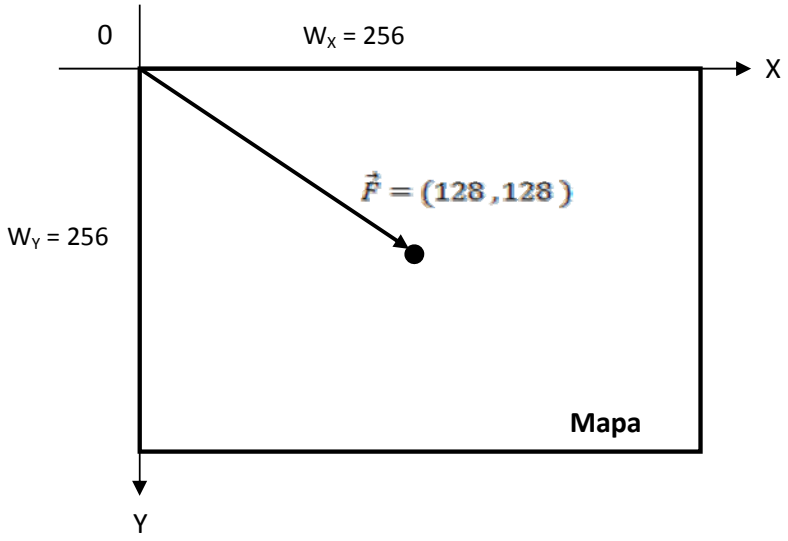


Figura 83. Imagen del mapa

Hay que crear en el siguiente paso el objetivo que representará el movimiento del personaje dentro del mapa. El objetivo es una imagen de 9 x 9 píxeles

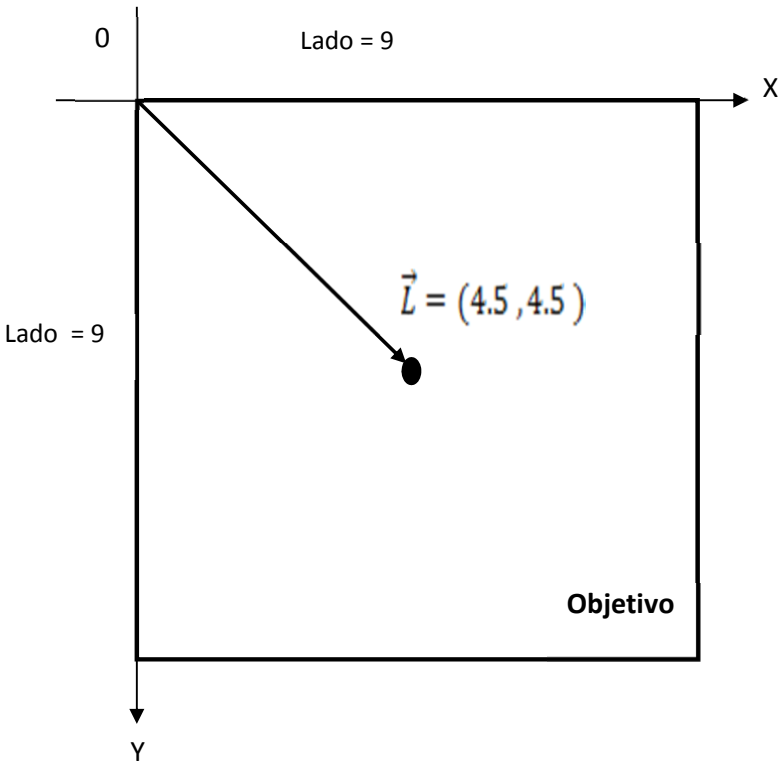


Figura 84. Imagen del Objetivo

Teniendo los datos de las figuras 9, 10 y 11 podremos sacar las ecuaciones de correspondencia entre el plano del mundo virtual y el mapa.

Sustituimos los datos de las figuras 9, 10 ,11 en las ecuaciones (K) y (M):

$$H_X = X_p \frac{256}{565.49} + 128 - 4.5 \quad \text{..... (N)}$$

$$H_Y = Y_p \frac{256}{236.055} + 128 - 4.5 \quad \text{..... (O)}$$

$$H_X = 0.4527 X_p + 123.5 \quad \text{..... (Q)}$$

$$H_Y = 1.0845\, Y_p + 123.5 \quad \text{..... (R)}$$

Pero tenemos un problema con esta ecuación, la posición que directamente te da esta ecuación , posiciona el objetivo fuera del centro del mapa, y esto es porque la ecuación funciona directamente cuando el centro del plano esté en el origen del mundo virtual, y en este caso se encuentra en el punto C (134.46 , -46.082).

La siguiente figura ilustrará lo anterior:

Cuando $X_P = C_X = 134.46$

$Y_P = C_Y = -46.082$

De (Q) y (R):

$H_X = 184.370042$

$H_Y = 73.524071$

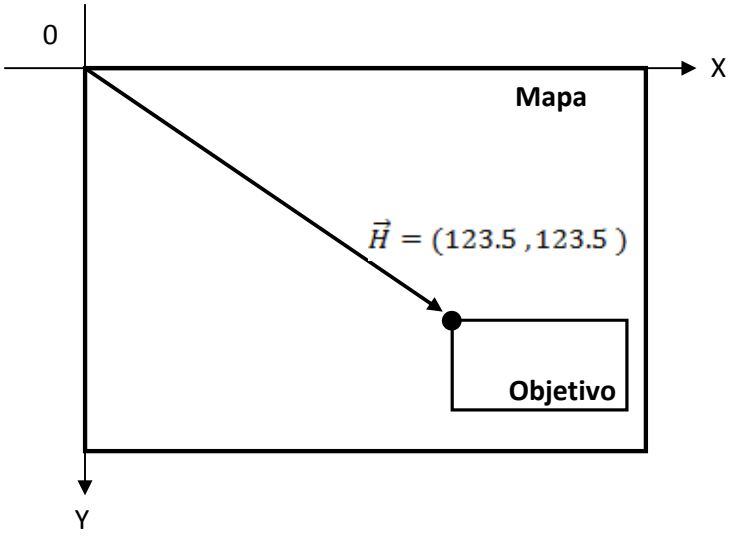


Figura 85. El Objetivo se encuentra desplazado por la posición del centro del plano

Para centrar el objetivo en el mapa lo único que tenemos que hacer es simular en la ecuación que el centro del plano, y por ende la posición del personaje, se encuentra en el origen del mundo virtual. Para esto, lo único que hay que hacer es restar las componentes del vector C (fig.9) en las ecuaciones (Q) y (R) que son las componentes del vector H. Para poder restar el vector C, hay que multiplicarlo primero por el factor de relación de las ecuaciones mencionadas anteriormente

$S_X = 0.4527 * C_X = 0.4527 * 134.46 = 60.870042$

$S_Y = 1.0845 * C_Y = 1.0845 * 46.082 = 49.975929$ → El valor negativo en C_Y no se toma en cuenta porque solo necesitamos su valor absoluto para sacar una relación.

$$\vec{R} = \vec{H} - \vec{S} = (H_X - S_X , H_Y - S_Y)$$

$$R_X = 0.4527X_p + 123.5 - 60.870042$$

$$R_Y = 1.0845Y_p + 123.5 - 49.975929$$

$$R_X = 0.4527X_p + 62.63 \quad \text{.....(S)}$$

$$R_Y = 1.0845Y_p + 73.524071 \quad \text{.....(T)}$$

Las nuevas ecuaciones de correspondencia para acoplar el centro del objetivo con el centro del mapa, están dadas por la ecuaciones (S) y (T) para cada eje coordenado dentro del mapa.

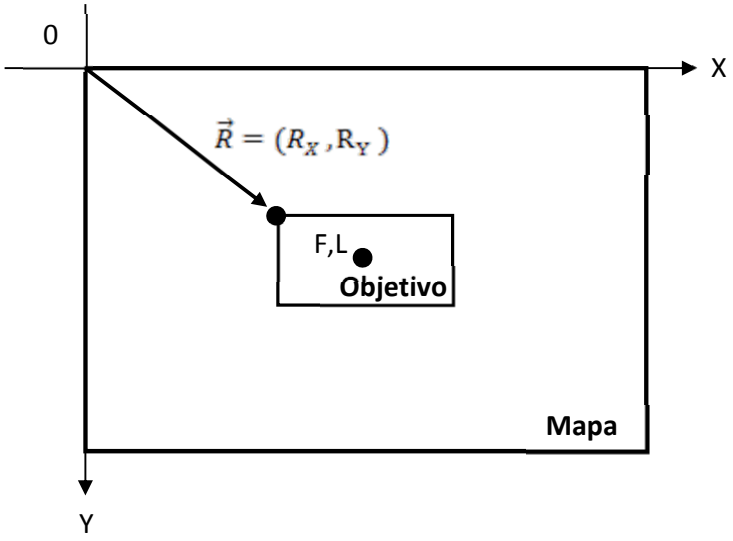


Figura 86. El vector R acopla los centros del objetivo y el mapa

Ya con el vector R en cada uno de sus componentes, la posición del objetivo dentro del mapa, seguirá a la posición del personaje dentro del mundo virtual, y esto nos indicará a cada momento en donde se ubica el personaje dentro de la zona arqueológica.

$$R_X = 0.4527X_P + 62.63 \quad \text{.....(S)}$$

$$R_Y = 1.0845Y_P + 73.524071 \quad \text{.....(T)}$$

Ya implantada esta solución, se tuvo que desplazar la imagen del mapa 10 píxeles desde su origen, por lo cual, en las ecuaciones (S) y (T) tuvimos que sumar ese desplazamiento de 10 píxeles.

$$R_X = 0.4527X_P + 62.63 + 10 \qquad R_Y = 1.0845Y_P + 73.524071 + 10$$

$$\boldsymbol{R_X = 0.4527X_P + 72.63} \quad \text{.....(V)}$$

$$\boldsymbol{R_Y = 1.0845Y_P + 83.524071} \quad \text{.....(W)}$$

MAPA

Para la elaboración del mapa dentro de nuestro entorno virtual, se tuvo que generar texturas que simularán las características de nuestro entorno virtual y exponerlas en todo momento en nuestro paseo.

Las texturas son, la primera de ellas, el contorno de nuestros edificios que conforman la zona arqueológica de Monte Albán; la segunda textura se usa para generar la transparencia que necesita la primer textura para solo presentar los contornos de los edificios, y la tercer textura es un color sólido que representará el objetivo del mapa que a su vez representa la posición actual del personaje dentro de la zona arqueológica.

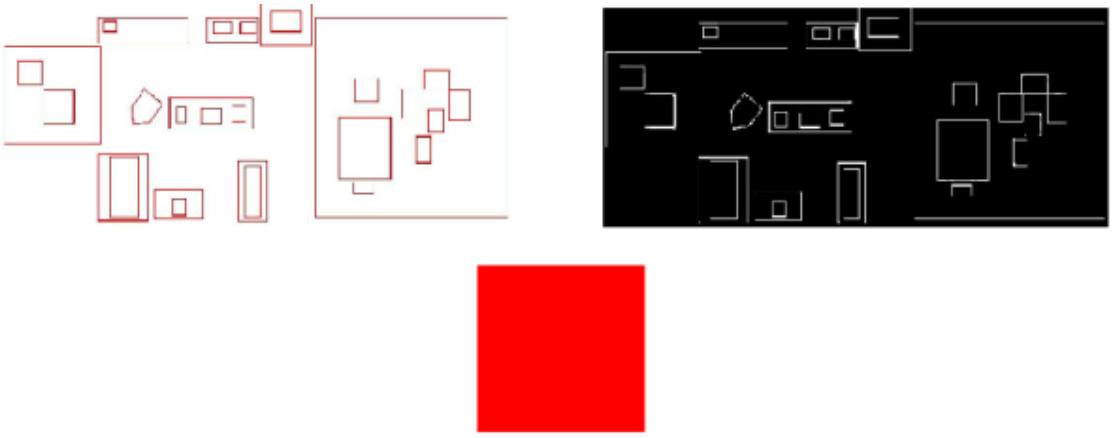


Figura 87. (a) Textura de los contornos de los edificios de la zona, (b) Textura para crear la transparencia y solo se vean los contornos de los edificios y (c) Textura empleada para el objetivo que marca a cada momento la posición del personaje.

Ahora veremos el grupo de canales que se hicieron para generar el mapa en base a las ecuaciones (V) y (W) que se obtuvieron en la explicación matemática anterior.

$$R_X = 0.4527X_P + 72.63 \quad \text{.....(V)}$$

$$R_Y = 1.0845Y_P + 83.524071 \quad \text{.....(W)}$$

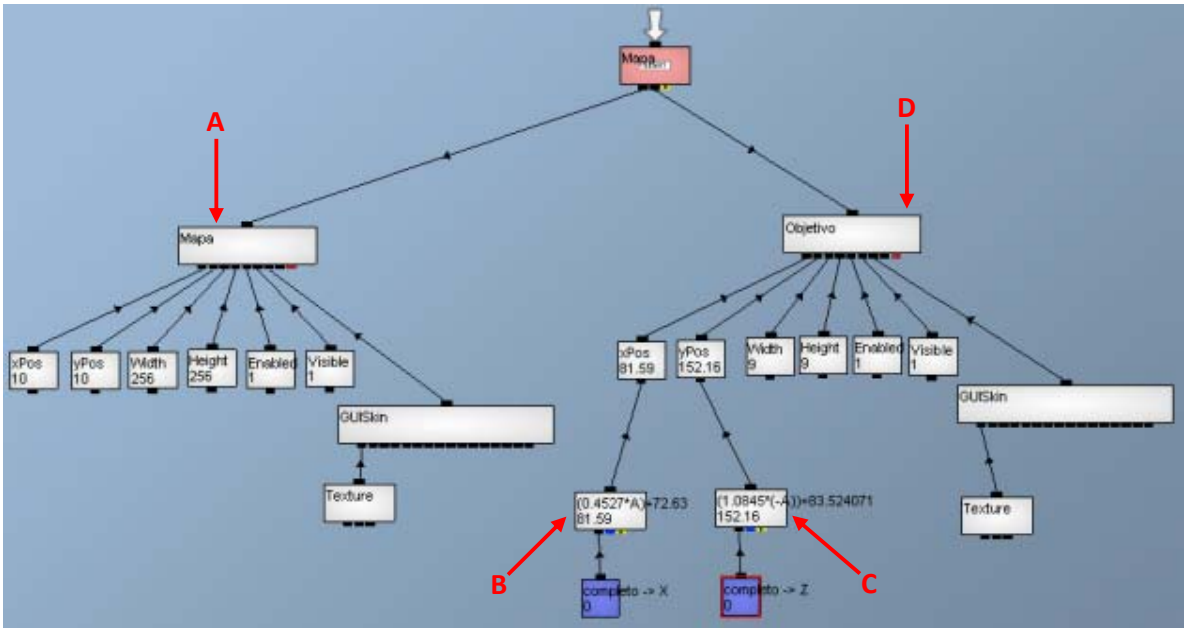


Figura 88. Grupo de canales hechos para mostrar en todo momento la posición del personaje dentro del paseo virtual.

De la figura 88:

- Con el indicador A se encuentra el canal "Mapa" el genera la proyección del mapa en la cámara todo el tiempo. Éste canal tiene varios atributos, pero explicaremos los atributos que nos interesa de él. Daremos una breve reseña de cada uno de sus canales hijos que tiene adjunto de izquierda a derecha. Los canales "xPos" y "yPos" son las coordenadas del punto origen del mapa dentro del sistema de coordenadas de la pantalla donde se proyecta la cámara. Los canales "Width" y "Height" representan el tamaño de la imagen en pixeles a lo ancho y a lo largo respectivamente cuando se muestra en la cámara. Los canales "Enable" y "Visible" sirven para habilitar la imagen y mostrarla en la cámara respectivamente. El canal "GUISkin" es el canal que sirve para agregar las texturas de las imágenes, como se puede observar, este canal tiene un canal hijo llamado "Texture" el cual contiene la imagen de los contornos de los edificios.

- Con el indicador B se encuentra el canal "(0.4527*A)+72.63" el cual hace la operación matemática que su propio nombre indica. Este canal sirve para integrar la ecuación de relación (V) para la posición en X del objetivo. Este canal tiene como única entrada la componente en X del vector de posición del personaje.

- Con el indicador C se encuentra el canal "(1.0845*(-A))+83.524071" el cual hace la operación matemática que su propio nombre indica. Este canal sirve para integrar la ecuación de relación (W) para la posición en Y del objetivo. Este canal tiene como única entrada la componente en Z del vector de posición del personaje; el porque tiene asignada la componente en Z es porque el

personaje se mueve en el plano XZ y el plano de la imagen solo tiene el plano XY, es por eso que se relacionó el eje Z del personaje con el eje Y de la imagen.

- Con el indicador D se encuentra el canal "Objetivo" el cual contiene la imagen del objetivo y todos sus atributos los cuales explicaremos a continuación. Los canales "xPos" y "yPos" son las coordenadas del objetivo en el sistema de coordenadas de la pantalla donde se proyecta la cámara. Los canales "Width" y "Height" representan el tamaño del objetivo en pixeles a lo ancho y a lo largo respectivamente cuando se muestra en la cámara. Los canales "Enable" y "Visible" sirven para habilitar la imagen y mostrarla en la cámara respectivamente. El canal "GUISkin" es el canal que sirve para agregar las texturas de las imágenes, como se puede observar, este canal tiene un canal hijo llamado "Texture" el cual contiene la imagen del objetivo.

Sonido.

Para explicar el funcionamiento del sonido tendremos que apoyarnos en la siguiente imagen, la cual nos indica el desarrollo de cómo es usado el sonido dentro de Quest3D ya aplicado a nuestros fines.

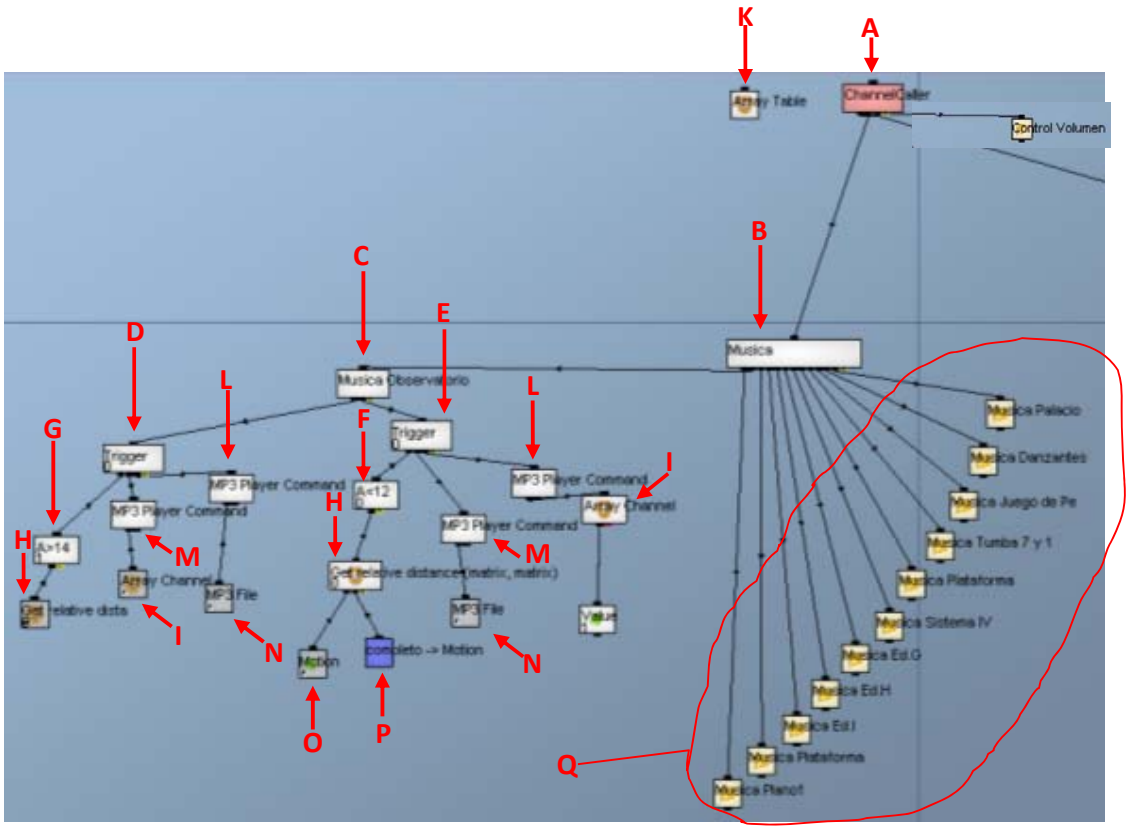


Figura 89. Grupo de canales hechos para generar el sonido.

En el Caller Channel (Indicador A, figura 89) se conectó un canal llamado Música (Indicador B, figura 89) el cual es un channel caller que es un canal el cual va a llamar a todos los canales hijos vinculados de uno en uno, de izquierda a derecha a menudo, este canal se utiliza para crear una estructura gráfica del canal que se llama de otra estructura. Posteriormente a el canal Música conectamos otro cannel caller, (Indicador C, figura 89) esto por que se necesitan varios channels caller ya que requerimos tener varios sonidos para cada una de las estructuras de la zona por lo cual colocamos el canal música como fuente y los demás channels caller para controlar los canales hijos de cada una de las estructuras.

En cada uno de los channels caller colocamos 2 triggers (Indicador D y E, figura 89) que tienen como función llamar a otro canal cuando se pasa un valor de un cierto punto, es decir que se puede configurar el trigger para llamar al canal cuando el valor pasa en cualquier dirección o se puede especificar la dirección en nuestro caso lo que hacen los trigger es que cuando se acerque el personaje al edificio suene el sonido que le asignamos y cuando se aleje se detenga.

Cada trigger tiene conectado una expression value el cual es un canal de expresión es decir es un valor basado en una expresión. Entre otras cosas, que le permite crear fórmulas, como A + B, A * B, etc. las cuales puede utilizar el canal de expresión como un canal de valor normal. En nuestro caso suponemos que para el trigger de la derecha A<12 (Indicador F, figura 89) donde A<12 es una expression value que tiene la finalidad de que cuando la distancia entre el personaje y el edificio sea menor a 1 arroje un valor de 1, lo que hace es enviar una señal al trigger y accionar los canales de la derecha en este caso activa el sonido.

Para el trigger de la izquierda suponemos que A>14 (Indicador G, figura 89) le mando el comando de parar al sonido, cuando el personaje esta alejada del edificio a una distancia de 2 unidades

Para lo anterior necesitamos calcular la distancia entre 2 matrices una del personaje y la otra del edificio, si suponemos que el edificio esta alejado 4.5 unidades, esta cantidad es la distancia entre 2 matrices, la matriz del personaje y la matriz del edificio y esto lo podemos realizar con el canal llamado Get relative distance (Indicador H, figura 89) el cual es un value operaitor, que queremos decir con esto, que es un canal que contiene un valor que es un punto flotante de lo que puede tener un número entero o fraccionado, y este canal es utilizado como canal hijo, como Vector, Sobre, Luces, Cámara y muchos más, y que aquí lo conectamos a la expression value llamada A<12 y A>14, las cuales a su vez están conectadas al trigger que les corresponde.

Con respecto a lo anterior, si la distancia entre el personaje y el edificio es menor a 12 la grabación se enciende y si es mayor a 14 se para la grabación, para esto nos sirve el Array Channel (Indicador I, figura 89) ya que para jalar un dato que se aloja en nuestra tabla en este caso una grabación, cuando seleccionamos el Array Channel, aparece un select table donde esta nos hace referencia a una tabla donde introducimos la información que requerimos.

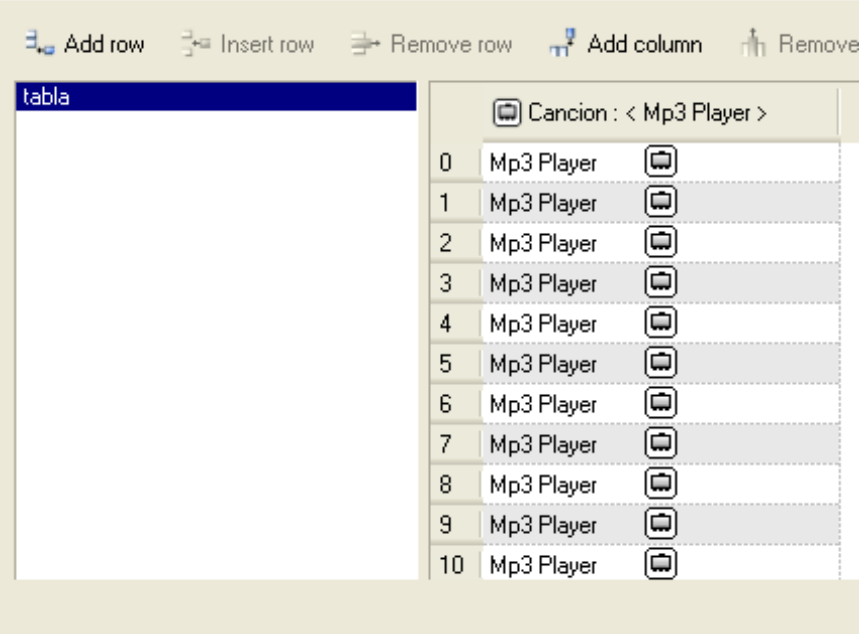


Figura 90. Arreglo de archivos de sonidos

En la tabla anterior que es llamada mediante el canal Array Table (Indicador K, figura 89) donde en cada renglón hay un Mp3 Player donde puedes usar este canal para controlar el canal de archivos MP3 y obtener información sobre el archivo MP3 que se está reproduciendo, en nuestro caso cada una de las grabaciones que contienen la información de cada uno de los edificios de la zona.

También para cada uno de los trigger colocamos un canal llamado MP3 Player Command (Indicador L, figura 89) y aquí colocamos un Play File From Star para que sea desde el principio dándole doble click a MP3 Player Command y esa señal es enviada al trigger. Lo que hicimos fue que cuando el personaje esta a menor de 12 unidades de distancia del edificio se activa el trigger, el cual envía una señal al MP3 Command que reproduce la grabación de la tabla. Los otros 2 MP3 Player Command (Indicador M, figura 89), el de la izquierda sirve para detener la grabación y el de la derecha para encender la grabación.

Los MP3 File son los canales que guardan la grabación que deseamos (Indicador N, figura 89), mientras que el canal Motion de la izquierda guarda la matriz de movimiento del edificio (Indicador O, figura 89) y el canal Motion de la derecha guarda la matriz de movimiento del personaje (Indicador P, figura 89).

Para cada uno de los canales encerrados en el indicador Q, figura 89, se hace el mismos procedimiento solo se varia la distancia de las expression value.

Para el sonido ambiental se conectaron otros canales, que para su mejor explicación nos apoyaremos con la siguiente imagen:

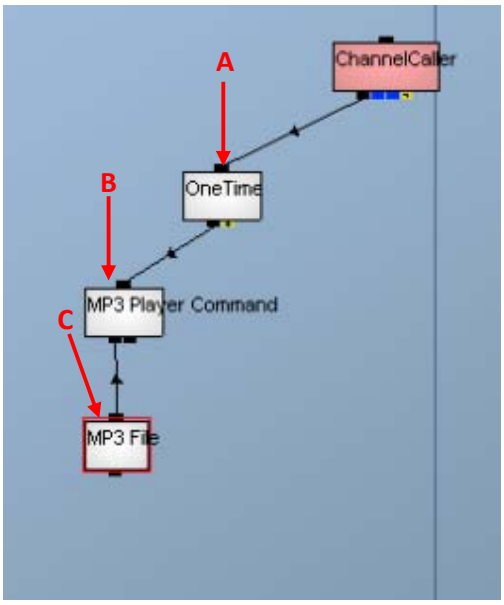


Figura 91. Grupo de canales hechos para generar el sonido ambiental.

Para el sonido ambiental conectamos al Channel Caller un One Time (Indicador A, figura 91), cuando se llama a este canal, se llamará en primer lugar, "Un canal hijo" sólo una vez, aun cuando este canal sea llamado varias veces y si se desea restablecer este canal se requiere una vez que el canal hijo puede usar el canal OneTimeReset o vincular un valor en el "Reset" la posición de enlace y establecer canal hijo menor a 1.Cuando este canal es uno volverá a llamar al canal hijo. En nuestro caso se llama al canal MP3 Player Command (Indicador B, figura 91) el cual a su vez tiene conectado a un MP3 File(Indicador C, figura 91).

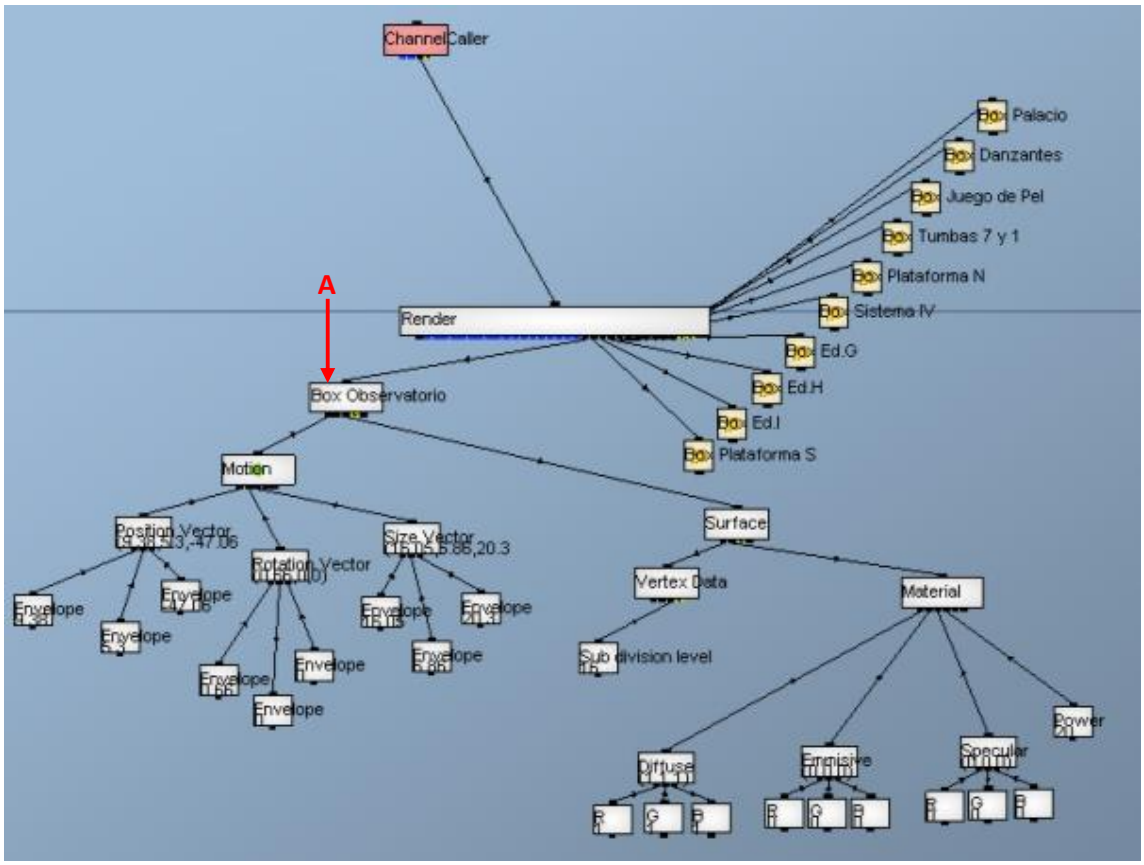


Figura 92. Grupo de canales hechos para generar el sonido.

Para poder censar la distancia entre el personaje y cada uno de los edificios tuvimos que recurrir a un canal que genera la instancia de una caja dentro de un espacio virtual. El canal "Box observatorio", tiene la capacidad de cambiar su posición, rotación, tamaño y su texturización, y este canal esta siendo ubicado por el indicador A de la figura 92.

Para el control del volumen utilizamos los canales que se muestran en la figura W. El Set Volume MP3 (Indicador A, figura 93) es el canal encargado del control del rango del volumen, mientras que el canal Interfaz Value 04 (Indicador C, figura 93) hace la llamada al canal publico que contiene el menú de opciones y este valor lo da el rango de desplazamiento de la barra deslizante que controla el volumen ambiental mientras que el canal de Interfaz Value 03 es el que controla la barra deslizante del volumen de las grabaciones (Indicador D, figura 93) , el rango de la barra deslizante es de -0.31 a 0.31 unidades, pero como este rango de valores no nos sirve ocupamos un canal Envelope (Indicador B, figura 93) el cual nos ayuda a modificar el rango de salida el cual debe de ser de 0.75 a 1.

Los demás canales de Set Volume MP3 contienen los mismos canales con la excepción de que estos contienen la Interfaz Value 03, ya que como ya mencionamos no colocamos en estos la Interfaz Value 04 por que esta solo la ocupamos para el sonido ambiental no para las grabaciones.

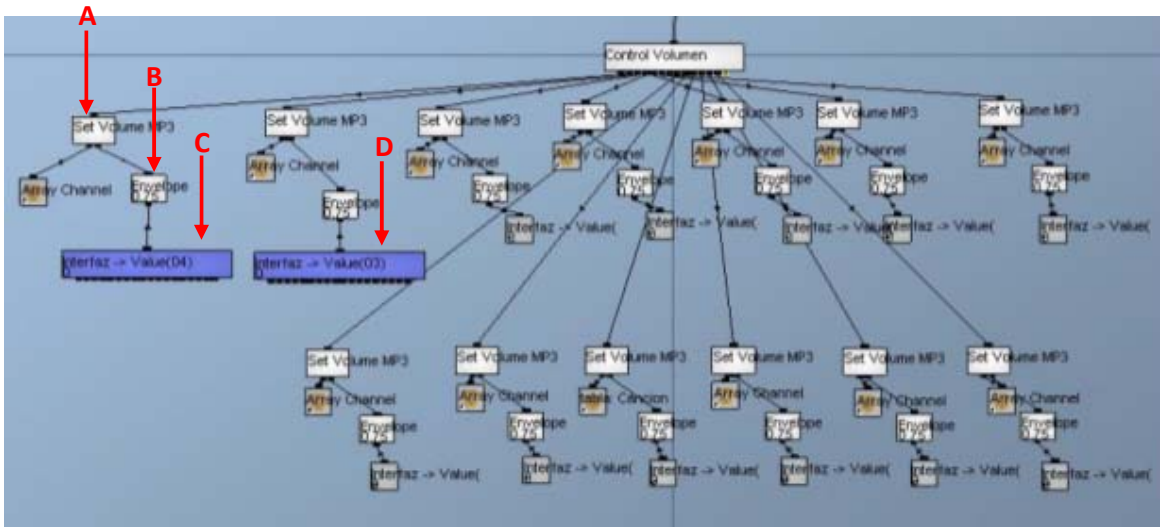


Figura 93. Grupo de canales hechos para generar el sonido.

Capítulo VI

Autómatas Finitos Determinísticos Aplicados

6.1 Introducción a los Autómatas Finitos

Un autómata de estado finito es un diagrama el cual indica el comportamiento de sus elementos cuando estos son sometidos a condiciones de una cierta lógica. Se puede decir que nos muestran como fluye la información de un punto a otro gracias a condiciones de transiciones que tiene.

En otras palabras, si estas en un juego de futbol, y quieres meter gol al equipo contrario, los elementos serían los jugadores que se pasen el balón con la condición que sean del mismo equipo y que uno de sus compañeros le grite el pase, y por último tu estado final sería la anotación del gol o no.

La definición formal de un autómata de estado finito está dada por los elementos

$$M = (Q, \sum, \delta, q_0, F)$$

dónde:

Q : Es el conjunto de estados

$\sum$: Alfabeto que define las palabras de entrada

δ : Funciones de transición donde $\delta: (Q \times \sum) \rightarrow Q$

q_0 : Estado inicial donde $q_0 \in Q$

F : Conjunto de estados finales donde $F \subseteq Q$

6.2 Desarrollo de ventana secundaria (tipo Menú) para el manejo de atributos

MENU

El paseo virtual debe de tener un menú de opciones que te permita visualizar y manipular atributos que éste mismo tiene. La creación del menú está basada en técnicas de renderización por prioridad como lo es el manejo del Buffer-Z aunado al manejo de la Interfaz de Usuario Gráfica.

El menú de nuestro paseo virtual está estructurado de la siguiente manera

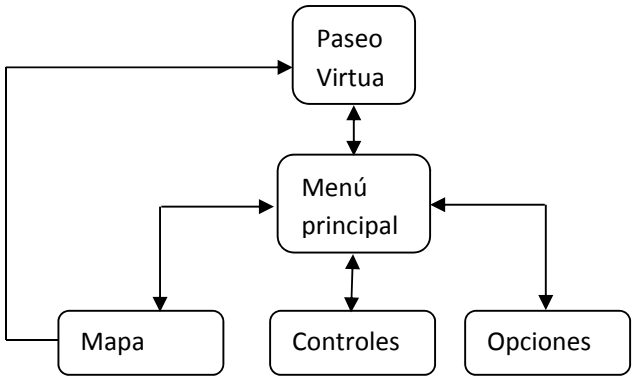


Figura 94. Diagrama de las fases del menú

De la figura 94 observamos que del paseo virtual pasamos al menú, para hacer esa transición tenemos que oprimir el botón derecho del mouse. Una vez adentro del menú observamos que tres atributos que podemos manipular, el primero es el MAPA, el cual contiene accesos directos seleccionables a las posiciones estratégicas dentro del paseo virtual; el segundo es los CONTROLES, donde podemos observar las teclas y periféricos que controlan al personaje principal dentro del paseo virtual; y por último podemos observar a las OPCIONES, donde nos muestran los atributos modificables de rapidez del mouse y del muñeco, y también los atributos modificables de volumen del sonido ambiental e informativo de los edificios.

Para empezar a hablar de cómo fue creado el menú, necesitamos dar un poco de teoría acerca de la técnica de Z-buffer y autómatas finitos, que es lo que se empleo para hacer el menú.

Técnica Z-buffer

Esta técnica lo que hace es que para cada pixel de la imagen que se proyecta en la cámara del espectador, se genera un par de coordenadas (x_s, y_s) más un valor de profundidad Z que se grabará en una matriz en el buffer (Z-buffer), cabe mencionar que este buffer se aloja en una parte de la memoria de la tarjeta de gráficos que está reservada para ello. Está técnica se emplea para solucionar el problema de decidir qué elementos del renderizado son visible u ocultos.

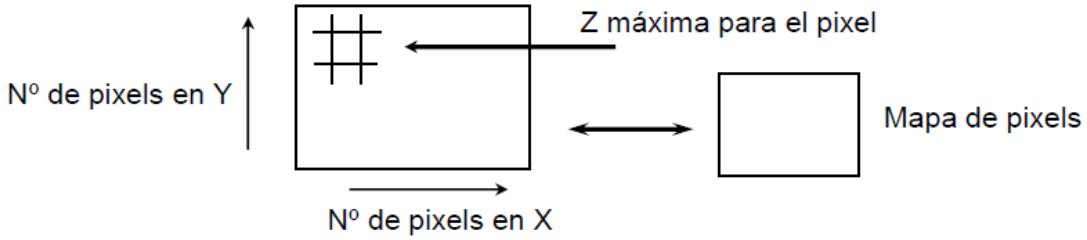


Figura 95. Matriz (Z-buffer) para almacenar la profundidad (valor Z) para cada pixel

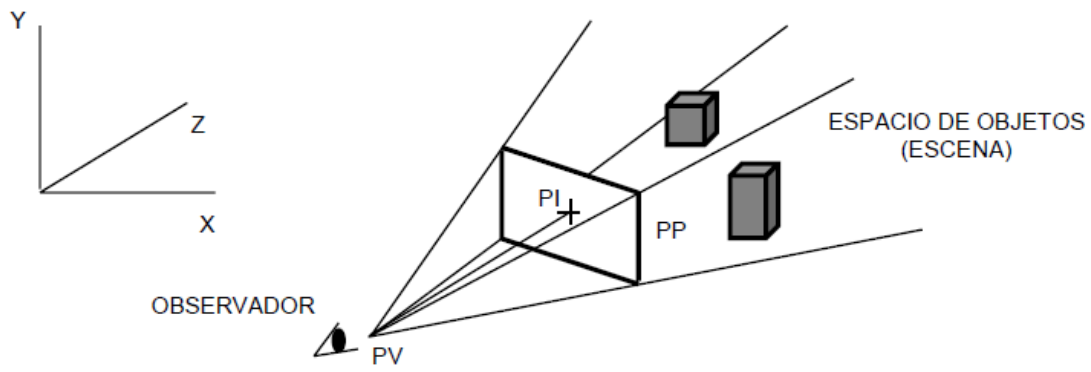


Figura 96. Vista del observador desde un cámara que se encuentra en un espacio virtual

De la figura 96 observamos que el punto de vista (PV) fija la mirada en un punto de interés (PI), y cuando nos fijamos en el PI podemos observar que cuando se hace un render se genera un plano de proyección (PP) el cual contiene todo el recurrido virtual, y es sobre el cual el observador ve su PI. Sobre este plano de proyección podemos ejecutar la técnica de Z-buffer ya que ese plano contiene todos los objetos visibles u ocultos de nuestra escena.

Para ilustrar como trabaja la técnica de Z-buffer podemos observa la siguiente figura

Figura 97. Se observa que con la técnica ya podemos elegir qué pixeles de la imagen se pueden mostrar u ocultar

El algoritmo de la técnica de Z-buffer funciona de la siguiente manera, se busca todos los polígonos que se contienen en el plano de proyección que se intersecan con un rayo que parte del ojo del observador y pasa por el píxel S. Si interseca, comparamos su profundidad (valor z del polígono) con la profundidad que hay en el z-buffer (xs,ys). Si el valor es menor al del z-buffer, el polígono es más cercano al

observador que el que tenemos actualmente en el buffer, reemplazamos el valor del z-buffer por el valor z del polígono y pintamos el píxel con el color del polígono.

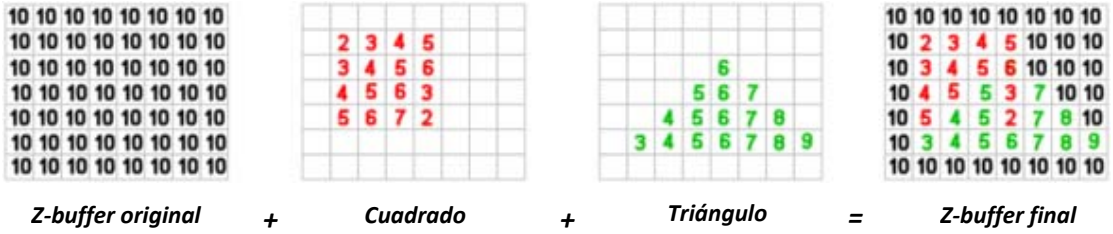


Figura 98. El resultado de emplear la Z-buffer es que cuando se solapan dos imágenes, se muestra parte de ellas si su valor de profundidad (valor Z) es menor en el pixel en cuestión

El modelo matemático que genera la asignación de los valores de profundidad (valor Z) para cada pixel en el plano de perspectiva a ser renderizado está normalmente definido entre un valor cercano (*N*) y otro lejano (*F*) de z. Después de una transformación de perspectiva, el nuevo valor de z, o z', está definido por:

$$z' = \frac{F+N}{F-N} + \frac{1}{z}\left(\frac{-2*F*N}{F-N}\right)$$

Donde z es el valor antiguo de z en el espacio de la cámara.

Los valores resultantes de z' están normalizados entre los valores -1 y 1, donde el plano cercano (*N*) está en -1 y el lejano (*F*) en 1. Los valores fuera de este rango corresponden a puntos que no están en la visión y no serán mostrados o renderizados.

Autómatas finito no determinístico

Para nuestro menú, se generó el siguiente diagrama de transiciones que se muestra en la siguiente figura, y del cual sacaremos su definición formal del autómata de estado finito.

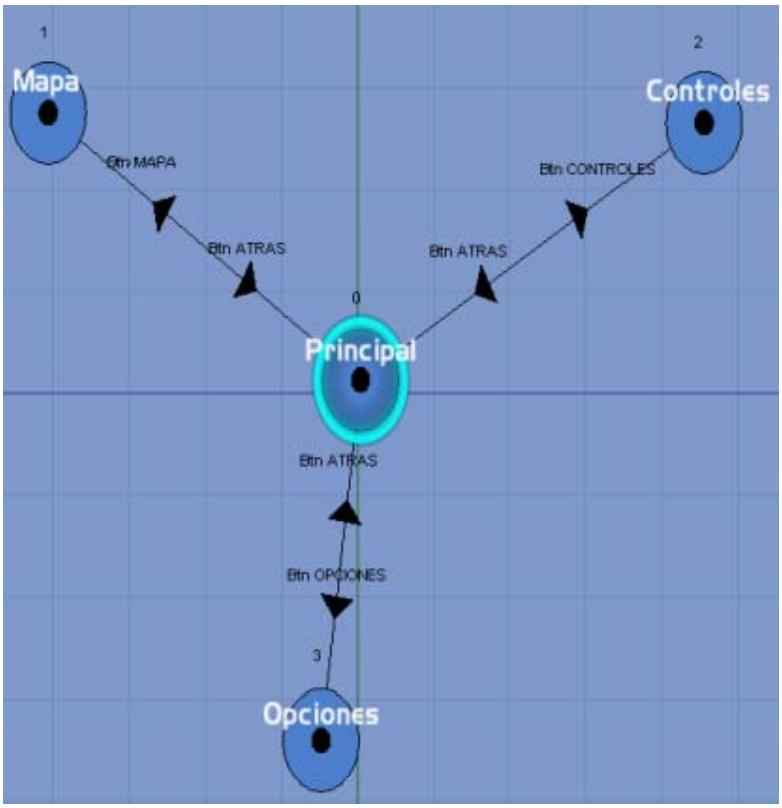
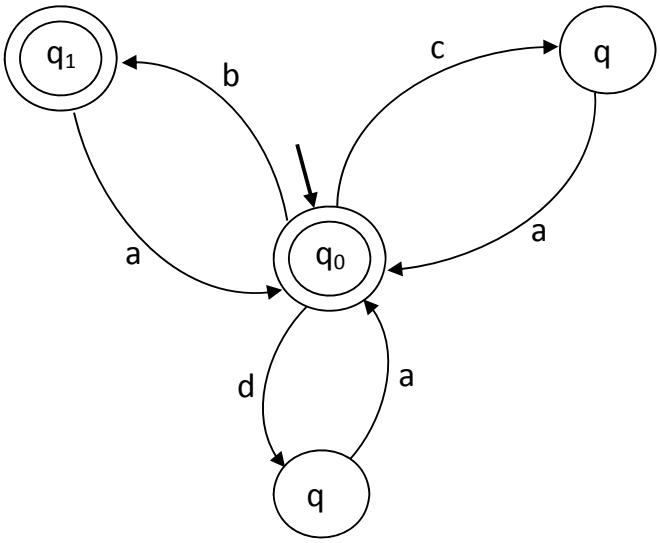


Figura 99. El resultado de emplear la Z-buffer es que cuando se solapan dos imágenes, se muestra parte de ellas si su valor de profundidad (valor Z) es menor en el pixel en cuestión

Para hacer notaciones más familiares necesitaremos emplear otra nomenclatura para identificar perfectamente el autómata de estado finito; de la figura 99 se puede cambiar las nomenclaturas como a continuación se muestra:

q_0 = Principal	a = Btn ATRAS
q_1 = Mapa	b = Btn MAPA
q_2 = Controles	c = Btn CONTROLES
q_3 = Opciones	d = Btn OPCIONES

Una vez ya obtenida esta nueva nomenclatura podemos proceder a sacar su Autómata Finito y un nuevo diagrama de transiciones tal y como se muestra a continuación:



$\mathbf{M = (Q, \sum, \delta, q_0, F)}$

$\mathbf{Q} = \{ q_0 , q_1, q_2, q_3 \}$

$\boldsymbol{\delta} = (q_0, b) \rightarrow q_1$

$(q_0, c) \rightarrow q_2$

$(q_0, d) \rightarrow q_3$

$(q_1, a) \rightarrow q_0$

$(q_2, a) \rightarrow q_0$

$(q_3, a) \rightarrow q_0$

$\sum = \{a, b, c, d\}$

$\mathbf{q_0}$ = Estado inicial

$\mathbf{F} = \{ q_0, q_1 \}$

Figura 100. Autómata Finito Determinístico definido

La peculiaridad que tiene este Autómata Finito de nuestro Menú, es que tiene dos estados finales q_0 y q_1 , lo que significa que cuando estamos en el menú principal podemos irnos al paseo virtual, y cuando estamos en menú de mapa, podemos pasarnos directamente al paseo virtual.

Para empezar a dar la explicación de cómo se hizo nuestro menú en base al autómata finito, empezaremos a explicar como se hizo cada parte del menú, o sea, qué se hizo en cada estado del autómata finito, para que al final podamos explicar como se hizo todo el autómata en si.

Menú Principal

De la figura 119, observamos que ahí se encuentran 4 botones y el fondo que los contiene, y también observamos el título del menú donde nos encontramos.

Comencemos explicando fondo del menú, para eso tendremos que ver la siguiente imagen.

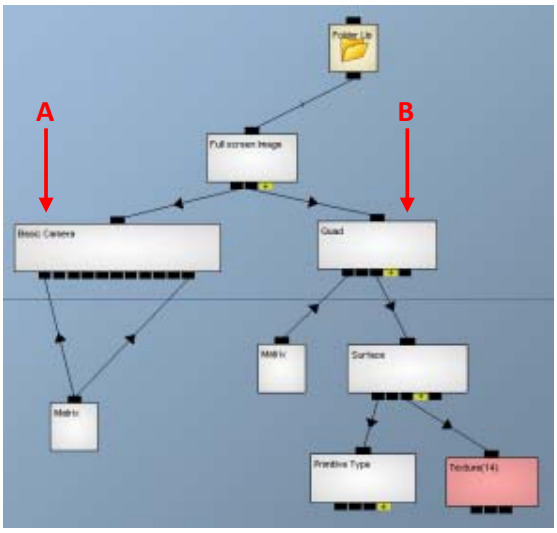


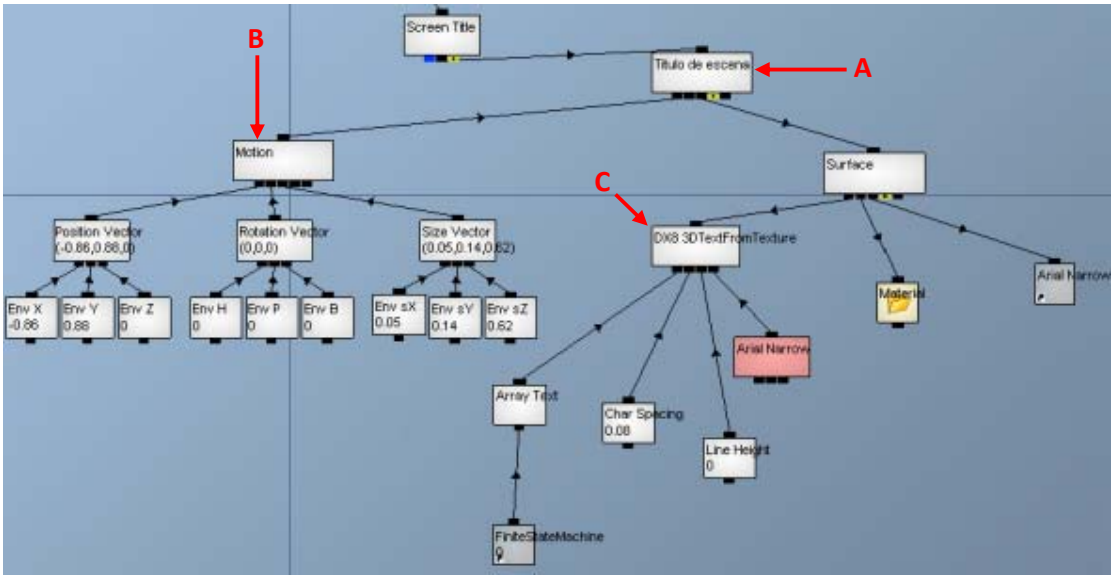
Figura 101. Grupo de canales hechos para generar el fondo del menú y en cada estado de él.

De la figura 101:

- Con el indicador A, se encuentra el canal “Basic Camera” el cual crea una cámara para visualizar nuestro entorno virtual, es una cámara muy sencilla ya que en su matriz de movimiento solo existen los valores de escalamiento, ya que esa cámara la necesitamos en el origen del sistema; también otra peculiaridad es que se le conecta la misma matriz a su entrada de matriz de proyección, y esto es porque necesitamos que muestre el origen del sistema en el plano XY.

- Con el indicador B, se encuentra el canal “Quad” que crea la instancia del objeto dentro de nuestro espacio virtual, y eso objeto va a ser un plano cuadrado, este plano lo determina uno de sus canales hijos llamado “Primitive Type”. Ahora ese plano deberá tener la textura que vemos en el fondo, la cual se lo dará el canal hijo “Texture(14)” la cual contiene esa imagen. A la imagen se le ha asignado a su valor del Z-buffer de 1, ya que va a ser la imagen más alejada, lo cual quiere decir que va a ser la imagen de fondo

Ahora explicaremos el título de cada parte del menú, el cual podemos apreciar en la esquina superior izquierda de la figura 119, la cual dice “Paseo en pausa”.



De la figura 102:

- Con el indicador A, se encuentra el canal "Título de escena" el cual crea la instancia del elemento que contendrá el título dentro de nuestro espacio virtual.

- Con el indicador B, se encuentra el canal "Motion" el cual tiene la matriz de movimiento del título del menú. Aquí solo se ajusta su vector de posición y su vector de escalamiento para moverlo a la esquina superior izquierda y modificar su tamaño de letra.

- Con el indicador C, se encuentra el canal "DX8 3DTextFrom Texture" el cual contiene los datos los atributos del elemento que se empleará para generar el título, como las texturas de las letras, el texto a mostrar, espacio entre letras, alto de letras, etc. Su canal hijo "Array Text" es un canal que apunta a una tabla que contiene los títulos del menú, y dependiendo del índice que se le dé a este canal, obtendrá el título que le corresponde; éste índice lo da su canal hijo "FiniteStateMachine", que es una acceso directo al canal que genera el autómata finito del menú. La tabla mencionada es la que sigue:

	Titulo : <Text>
0	Paseo en Pausa
1	Mapa
2	Controles
3	Opciones

Figura 103. Leyenda de botones del menú en arreglo.

Cuando el canal “FiniteStateMachine” arroja un valor entero entre 0 y 3, éste le indica al canal “Array Text” que obtenga el valor del renglón de la tabla hacia la cual apunta, y en este caso apunta hacia el renglón 0, lo cual significa que arrojará la cadena de caracteres “Paseo en Pausa” hacia el canal “DX8 3DTextFrom Texture”, el cuál a su vez buscará los caracteres en la textura que está contenida en su canal hijo “Arial Narrow”. Cabe mencionar que la textura de la letras debe de tener un valor menor a 1 en el Z-buffer para que esté encima de la imagen de fondo.

Ahora explicaremos el botón de “MAPA” de la figura 119. Cabe mencionar que explicando el grupo de canales de botón, se darán por explicados la elaboración de los botones “Controles” y “Opciones” ya que fueron hechos bajo el mismo procedimiento.

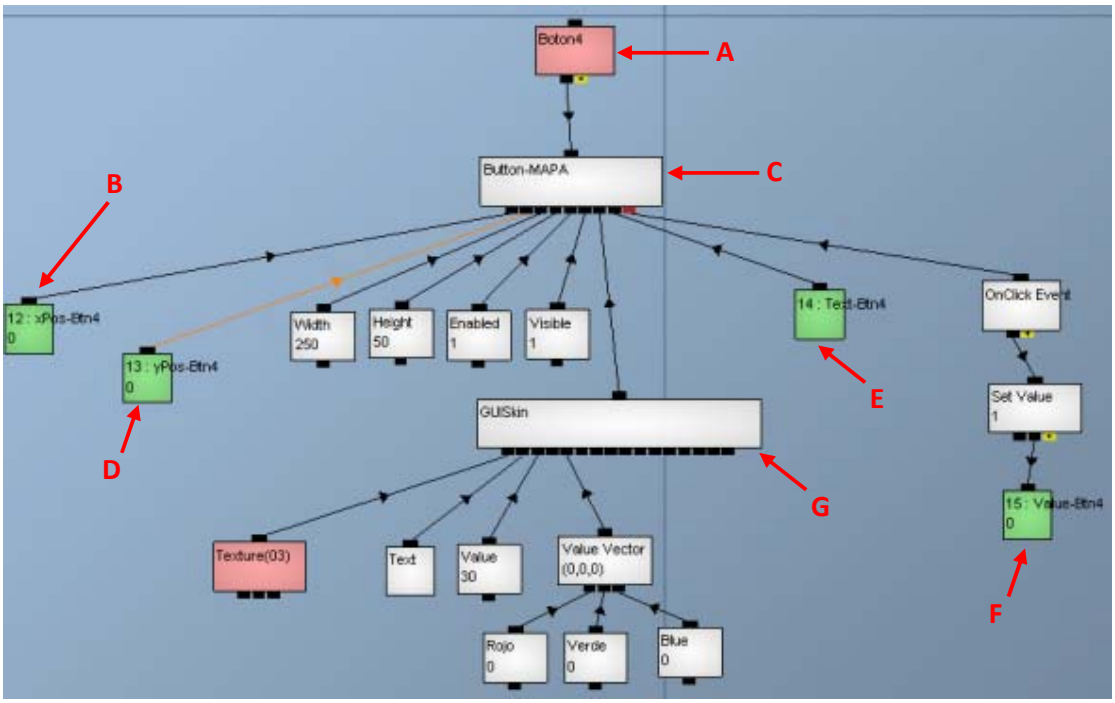


Figura 104. Grupo de canales hechos para generar el botón de mapa.

Antes de empezar a explicar el grupo de canales de la figura 104, tenemos que explicar primero que existen canales privados, canales públicos, canales con llamadas a canales públicos y parámetros de canales públicos.

Los canales públicos son canales que pueden ser llamados desde un grupo de canales externo, y estos canales son coloreados de color rojo, como el canal con el indicador A de la figura 104. Los canales privados son los canales en color blanco, y son canales que solo sirven para el grupo de canales que los contienen. Los canales que hacen la llamada a canales públicos son coloreados en azul, y son usados para recurrir a las funciones que efectúa el grupo de canales que contiene al canal público.

Figura 105. Canal que hace llamada a un canal público

Los parámetros de canales públicos, son canales que se usan como parámetros de entrada de los canales públicos, y estos tienen un color verde como el canal que se muestra con el indicador B de la figura 104. Cabe mencionar que estos parámetros tanto pueden ser de entrada, como pueden ser de salida.

Cuando es llamado un canal público que tiene canales hijos de parámetros en cualquier nivel del árbol, éste canal de llamado es generado con entradas, las cuales serán los parámetros de entrada del canal público.

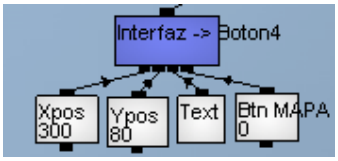


Figura 106. Aquí se muestra el canal de llamado "Interfaz -> Boton4", el cual hace referencia al canal "Boton4" de la figura A1. El canal de llamado es generado con 4 entradas, las cuales corresponden a los parámetros (canales de color verde) que existen en los canales hijos del canal público "Boton4".

De la figura 104:

- Con el indicador A, se encuentra el canal "Boton4" el cual es un canal público que contiene el grupo de canales que genera en la pantalla el botón de MAPA.

- Con el indicador C, se encuentra el canal "Button-MAPA" el cual genera una imagen con valor en el Z.buffer de -1 para que siempre sea visible en la cámara. Este canal genera la imagen que simula un botón sobre la pantalla.

- Con el indicador B, D, E, y F, se encuentran los canales "12: xPos-Btn4", "13: yPos-Btn4", "14: Text-Btn4" y "15: Value-Btn4" respectivamente. El primer canal sirve como parámetro de entrada para ajustarla coordenada en X del botón. El segundo canal sirve como parámetro de entrada para ajustarla coordenada en Y del botón. El tercer canal sirve como parámetro de entrada para asignarle el nombre al botón, en este caso MAPA. El cuarto canal sirve como parámetro de salida, el cual almacena el valor de 1 cuando se le hace click al botón "MAPA", esto gracias a que el canal "OneClick Event" se acciona cuando se presiona el botón y le manda una señal a su canal hijo "Set Value" que le asigan el valor de 1 a su canal hijo.

- Con el indicador G, se encuentra el canal "GUISkin" el cual contiene información de la textura del botón, el tipo y tamaño de letra de su nombre y el color de la misma.

Una vez explicado el botón de “MAPA” de la figura 119, ya no es necesario explicar los botones “Controles” y “OPCIONES”, ya que fueron hechos idénticamente a éste.

Ahora explicaremos el botón de “ATRAS” del menú principal, el cuál es el mismo en cada parte del menú.

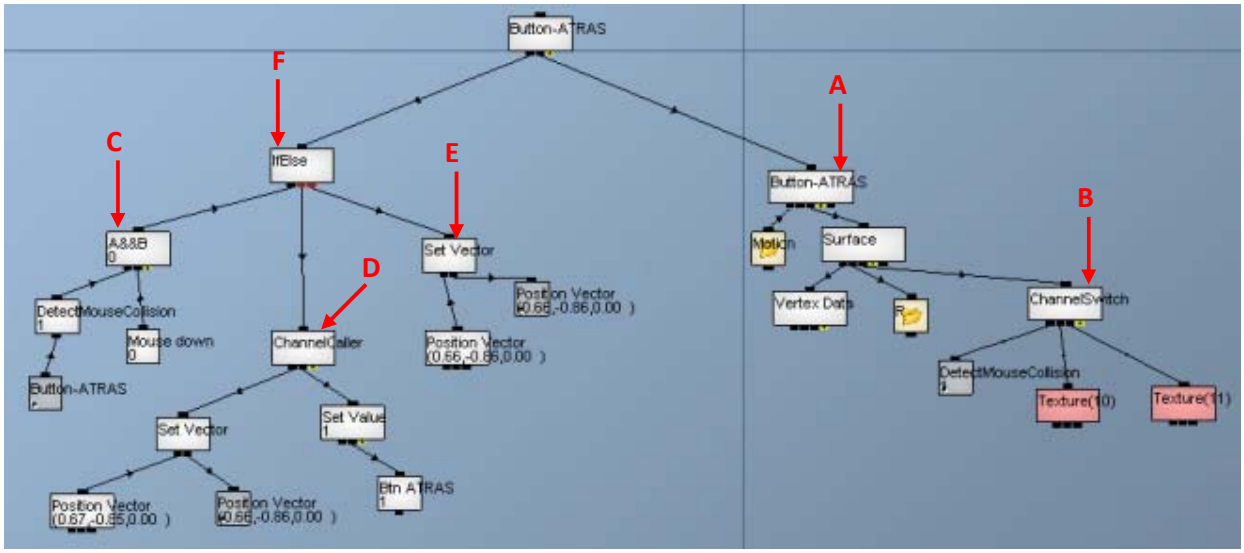


Figura 107. Grupo de canales para hacer el botón “ATRAS”

De la figura 107:

- Con el indicador A, se encuentra el canal “Button-ATRAS”, el crea la instancia del botón sobre nuestro menú. Las texturas de este botón tiene un valor ene. Z-buffer de -1, ya que este botón debe ser visible sobre todas las imágenes en todo momento.

- Con el indicador B, se encuentra el canal “ChannelSwitch”, este canal hace el cambio de canales hijos de uno a otro dependiendo de qué valores de entrada le demos. El canal de entrada que tiene es su canal hijo “DetectMouseCollision”, que es un canal que entrega el valor de 1 cuando detecta que el puntero del Mouse pasa sobre el objeto al cual hace referencia, en este caso, es el botón de ATRÁS; cuando el canal hijo “DetectMouseCollision” le envía el valor de 1 a su canal padre “ChannelSwitch”, este último cambia del canal hijo “Textura(10)” al canal “Textura(11)”, lo cual significa que cambia de textura al botón cuando el puntero del Mouse pasa sobre él.

- Con el indicador C, se encuentra el canal “A&&B” el cual es un canal que hace una operación matemática que su propio nombre indica. La operación es booleana y es un AND, esto significa que cuando el puntero del Mouse esté posicionado sobre el botón de ATRÁS, la entrada A del canal “A&&B” será 1, ahora cuando se haga un clic con el botón izquierdo del Mouse su entrada B del canal “A&&B” será 1, con la entrada A y B del canal “A&&B” son positivas, este canal tomará el

valor de 1. De manera gráfica, se puede decir que este canal detecta cuando se le hace clic al botón de ATRÁS.

- Con el indicador D, se encuentra el canal “ChannelCaller”, el cual hace la función de llamar a todos sus canales hijos. El canal hijo de su derecha es el “Set Vector” el cual copia los valores de las componentes de un vector a otro, esto significa que cuando es llamado este canal, copia un vector dado al vector de posición del botón ATRÁS, lo cual significa que la posición original del botón ATRÁS será afectada de tal manera que sufrirá un pequeño desplazamiento. Otro canal hijo del canal “ChannelCaller” es el “Set Value”, el cual pega el valor de 1 a su canal hijo “Btn ATRAS”, este último canal será usado para las transiciones del Autómata que maneja a todo el menú, por lo cual será explicado más adelante.

- Con el indicador E, se encuentra el canal “Set Vector” el cual copia los valores del vector de posición original del botón ATRÁS, al vector de posición que actualmente tiene el botón de ATRÁS, esto quiere decir que este canal mueve a su posición original al botón ATRÁS después de ser desplazado.

- Con el indicador F, se encuentra el canal “IfElse”, el cual contiene la sentencia lógica condicional que tiene como función lo siguiente, cuando no se encuentra el Mouse sobre el botón de ATRÁS, significa que su entrada de este canal es falsa (0), por lo cual se ejecuta el grupo de canales hijos que posiciona al botón ATRÁS en su lugar original. Cuando el Mouse se encuentra sobre el botón ATRÁS y se le da clic sobre él, el canal ejecuta el desplazamiento del botón a una posición nueva y le da un valor de 1 al canal que maneja las transiciones en el autómata del menú.

Menú Mapa

De la figura QQ, observamos que ahí se encuentran dos elementos distintos al menú principal, que son los botones de posición verdes que hacen la función de posicionar al personaje principal en el lugar que representan, y también observamos que existe una leyenda y un mapa real de la zona virtual en perspectiva.

Empezaremos explicando el grupo de canales que se hizo para la generación de los botones verdes, pero solo daremos la explicación de uno solo ya que el resto de los botones de posición fueron hechos idénticamente.

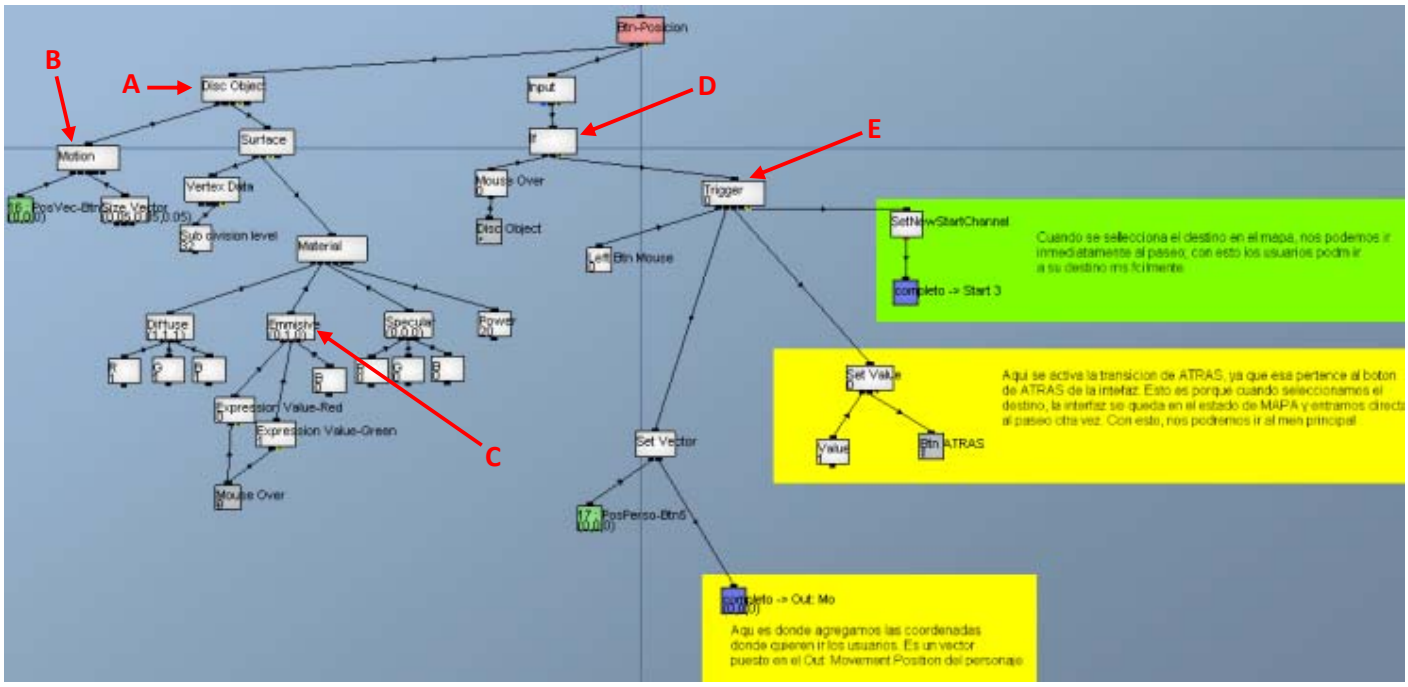


Figura 108. Grupo de canales para hacer el botón de posición del menú de mapa (botones verdes)

De la figura 108:

- Con el indicador A, se encuentra el canal “Disc Object” el cual hace la instancia del botón de posición sobre el mapa en perspectiva. Estos botones tiene un valor en Z-buffer de -1, ya que son los que van a estar por encima de todas las imágenes.

- Con el indicador B, se encuentra el canal “Motion” el cual contiene la matriz de movimiento del botón de posición. Este canal tiene un canal hijo en su vector de posición, el cual es un parámetro de entrada que sirve para posicionar el botón de posición en el mapa. Este parámetro fue creado porque se generaron muchas instancias de este mismo botón, pero cada instancia tiene una posición diferente, por lo cual se generó ese parámetro.

- Con el indicador C, se encuentra el canal “Emmisive”. Este canal maneja el canal emisivo de color del botón de posición, y a su vez tiene dos canales hijos “Expression Value-Red” y “Expression Value-Green” que controlan el nivel de color rojo y verde respectivamente. Estos canales hijos a su vez tiene un canal hijo en común, el cual es “Mouse Over” el cual cambia de valor cuando el puntero del Mouse está sobre el botón de posición, lo cual significa que cuando el Mouse no está sobre ningún botón de posición, éste tiene un color verde, pero cuando el Mouse se pone encima de cualquier botón de posición, éste cambia a color rojo.

- Con el indicador D, se encuentra el canal “If”. Este canal ejecuta una sentencia lógica, la cual es verdadera cuando el Mouse se encuentra encima de cualquier botón de posición, esto lo hace su

canal hijo “Mouse Over” el cual manda la señal de verdadero para el canal “If”. Cuando el canal “If” es verdadero, ejecuta el grupo de canales que se encuentran en su canal hijo “Trigger”.

- Con el indicador E, se encuentra el canal “Trigger”. Este canal se activa cuando su canal hijo “Left Btn Mouse” detecta que se ha hecho un clic con el botón izquierdo del Mouse sobre algún botón de posición. Cuando se activa el canal “Trigger” a través de su canal hijo “Left Btn Mouse”, su primer canal hijo que se activa es “Set Vector”, este canal lo que hace es que copia las coordenadas de posición del parámetro de entrada (canal hijo color verde) hacia el vector de posición del personaje, así cada botón de posición del mapa contendrá coordenadas específicas para ubicar al personaje un la posición deseada. El segundo canal hijo de “Trigger” que se activa es “Set Value”, el cual le asigna el valor de 1 a su canal hijo “Btn ATRAS” el cual es usado para hacer la transición del autómata hacia el menú principal. El tercer canal hijo de “Trigger” que se activa es “SetNewStartChannel”, este canal lo que hace es poner como canal de inicio del proyecto, el canal de inicio que contiene al paseo virtual, ya que el menú debe de terminar.

Para la imagen de la leyenda y el mapa en perspectiva, son instancias que se crearon con un canal de Objeto de 3D, lo único que se hizo para visualizar ambas imágenes, se les dieron valores en Z-buffer de -1 a la imagen de la leyenda, y un valor de 0 a la imagen del mapa en perspectiva, ya que este último debía de estar entre la imagen de fondo y los botones de posición.

Menú Controles

De la figura 121, observamos que se encuentran imágenes estáticas las cuales solo hacen referencias a los controles que se emplean para controlar al personaje. Para hacer esta parte del menú, sólo manejamos valores de profundidad en el Z-buffer de cada imagen para que no se traslapen las imágenes. A continuación se mostrará la figura que contiene a todas las carpetas donde son guardados los canales de texturas donde se almacenan las texturas.

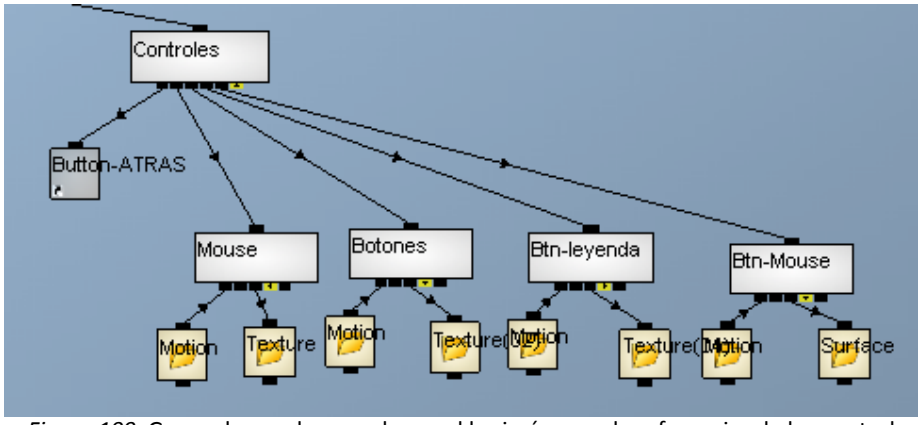


Figura 109. Grupo de canales para hacer el las imágenes de referencias de los controles del personaje dentro del paseo virtual.

De la figura 109, podemos observar que los canales “Mouse”, “Botones”, “Btn-leyenda” y “Btn-Mouse” tienen folders de Motion y Textura, lo cual implica que cada uno de los canales mencionados son canales que generar instancias de imágenes en nuestro espacio virtual que podemos manipular su posición y su textura que llevarán.

El canal de “Mouse” contiene la imagen de la figura del Mouse que se aprecia en la figura 121, y está imagen tiene un valor en el Z-buffer de 0, ya que es una imagen que debe estar entre dos imágenes, una es la imagen de fondo y la otra la descripción de sus botones y sus movimientos.

El canal de “Botones” tiene almacenada la imagen que simula el área de las teclas que se deben de usar para manejar al personaje. Ésta imagen tiene un valor de profundidad en el Z-buffer de -1, ya que es una imagen que siempre estará al frente de la imagen de fondo y contiene ninguna imagen frente a ella.

El canal de “Btn-leyenda” almacena la imagen de descripción del movimiento del personaje que genera cada tecla al ser apretada. Esta imagen tiene un valor de profundidad en el Z-buffer de -1 ya que es una imagen que está hasta el frente de la pantalla.

El canal de “Btn-Mouse” contiene la imagen que describe los botones del Mouse y sus movimientos verticales y horizontales respecto al control del personaje. Esta imagen tiene un valor de profundidad en el Z-buffer de -1, ya que es una imagen está hasta el frente de las imágenes del fondo y del Mouse.

Menú Opciones

De la figura 122, observamos que se encuentran imágenes estáticas que hacen referencia a cada parte del paseo virtual que el usuario puede cambiar, como lo son el volumen del ambiente, volumen de narraciones, rapidez con la que gira el personaje y la rapidez con la que camina y corre; estas referencias están contenidas en una sola imagen. Las imágenes que tiene las leyendas de Normal y Max, son imágenes que están estáticas y que sirven como descripciones de los límites de cada atributo que pueden modificar los usuarios.

Las barras deslizantes que se ubican en el menú de opciones, son imágenes que están compuestas por una barra azul larga que es estática y una imagen de un cuadrito rojo que puede deslizarse a través de su barra azul. Las cuatro barras fueron hechas de la misma manera, así que solo daremos la explicación de cómo se hizo una de ellas, y de las barras restantes solo explicaremos sus canales que sirven para manipular atributos del paseo virtual.

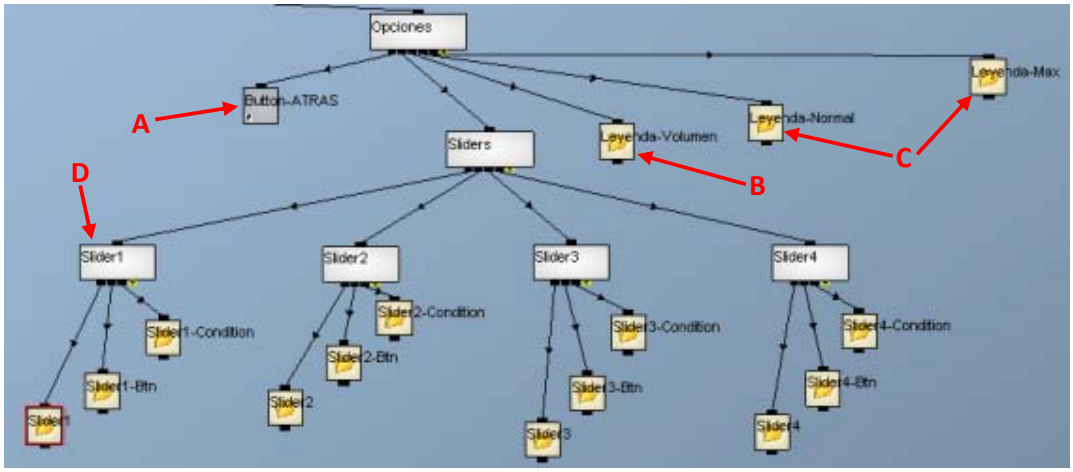


Figura 110. Grupo de canales para hacer el menú de opciones

De la figura 110:

- Con el indicativo A, se encuentra el canal “Button-ATRAS” que es el canal que maneja una de las transiciones que utiliza el autómata del menú. Este canal es un acceso directo hacia otro que se encuentra fuera de este grupo de canales, pero se explicará más adelante.

- Con el indicativo B, se encuentra el fólder “Leyenda-Volumen”, que es el fólder que contiene al canal que genera la instancia de la imagen de las descripciones de volumen del ambiente, volumen de narraciones, rapidez con la que gira el personaje y la rapidez con la que camina y corre el personaje. Este canal contiene la textura que contiene las palabras de descripción y a esta textura se le modifico su valor de profundidad en el Z-buffer a -1 ya que debe de verse todo el tiempo por encima del fondo del menú.

- Con el indicativo C, se encuentran los fólder “Leyenda-Normal” y “Leyenda Max”, que son los fólder que contienen a los canales que generan las instancias de las imágenes Normal y Max que hacen referencia a los límites inferior y superior respectivamente de los valores que puede manipular el usuario. Las texturas que manejan estas imágenes tienen un valor de profundidad en el Z-buffer de -1, ya que deben de verse por encima del fondo todo el tiempo.

- Con el indicativo D, se encuentra el canal “Slider 1” que contiene los grupos de canales que hacen que podamos manipular la barra deslizante del Volumen Ambiental. Este canal tiene tres fólder como canales hijos, el primero es “Slider1” que contiene la imagen de la barra azul, el segundo es “Slider1-Btn” que contiene el cuadrito rojo que podemos deslizar a través de la barra y el tercero es “Slider1-Condition” que tiene el grupo de canales que manipula la reacción del cuadrito rojo cuando éste sea afectado por el puntero del Mouse.

Cabe señalar que el canal “Slider1” modifica el volumen del sonido ambiental, el canal “Slider2” modifica el volumen del sonido de las narraciones, el canal “Slider3” modifica el factor de la rapidez de rotación del personaje y el canal “Slider4” modifica el factor de rapidez del caminado y el correr del personaje.

El resto de los canales son canales basados en el canal “Slider 1”, por lo cual procederemos a explicar los fólder que contiene este canal.

Fólder "Slider1"

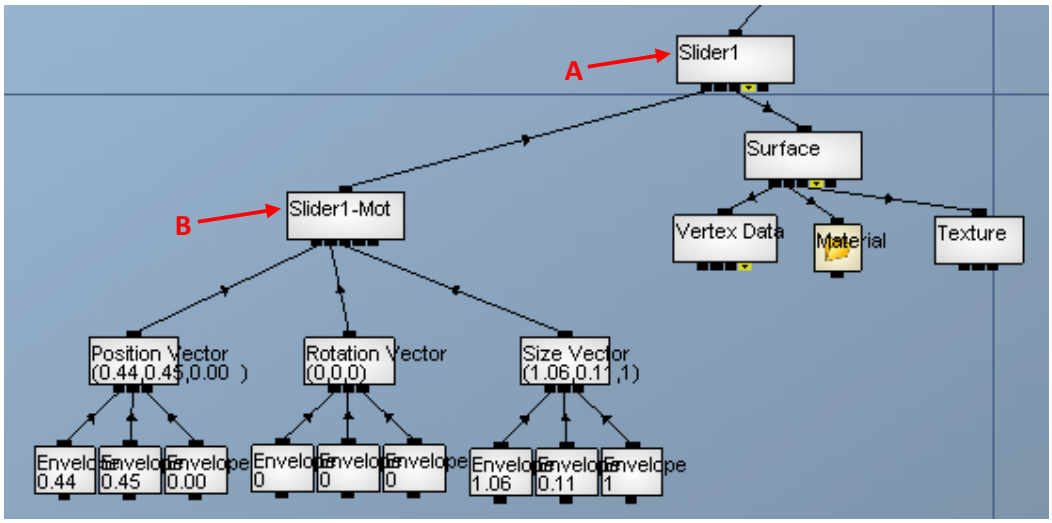


Figura 111. Grupo de canales para hacer la barra azul de la imagen de la barra deslizable

De la figura 111:

- Con el indicativo A, se encuentra el canal "Slider1" que es el canal que genera la instancia de la barra dentro de nuestro espacio virtual. Este canal contiene la textura de la imagen de la barra, a la cual le dimos en valor de profundidad dentro del Z-buffer de 0, ya que debe de estar entre el fondo del menú y el cuadrito rojo.

- Con el indicativo B, se encuentra el canal "Slider1-Mot", que es el canal de la matriz de movimiento de la barra. Este canal solo tiene modificaciones en su vector de posición ya que la barra solo está estática en un lugar específico del menú de opciones.

Fólder "Slider1-Btn"

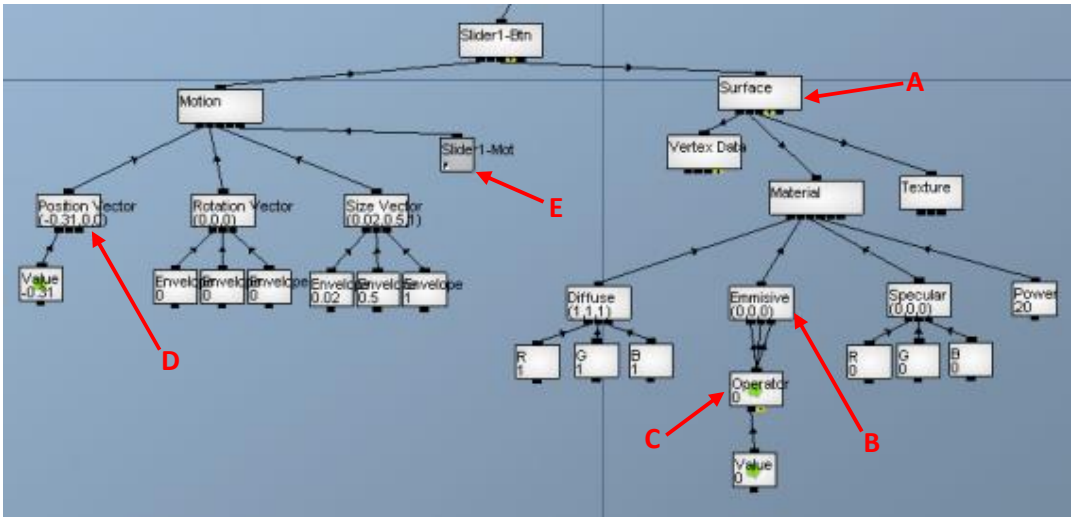


Figura 112. Grupo de canales para hacer el cuadrito rojo que contiene cada barra deslizable

De la figura 112:

Con el indicativo A, se encuentra el canal "Surface" que es el canal que genera la instancia del cuadrito rojo dentro de nuestro espacio virtual. Este canal se le modificó su canal hijo "Emmisive" con el indicador B, el cual maneja la componente emisiva de su color para que al momento de seleccionar el cuadrito rojo, éste sufra un sombreado en su textura. A la imagen de cuadrito rojo le dimos en valor de profundidad dentro del Z-buffer de -1, ya que debe de estar delante de la barra azul.

- Con el indicativo C, se encuentra el canal "Operator" el cual contiene la operación matemática de A*0.1, esto quiere decir que cuando le demos un valor, éste será disminuido al 90% de su valor. Con este valor disminuido se manejará la componente emisiva de la textura del cuadrito rojo, para que al momento de ser seleccionado, se sombree.

- Con el indicativo D, se encuentra el canal "Position Vector" que es el canal que contiene la posición del cuadrito en todo momento. Podemos observar que este vector solo tiene valor en su componente en X, y es porque el cuadrito rojo solo podrá moverse sobre su eje en X, o sea que tendrá un movimiento horizontal únicamente.

- Con el indicativo E, se encuentra el canal "Slider1-Mot"el cual es un acceso directo de la matriz de movimiento de la barra azul sobre la cual descansa el cuadrito rojo. El motivo de esta matriz es que la matriz de movimiento del cuadrito rojo herede la posición de la barra y con esto el cuadrito no se pueda mover a otro lado que no fuera la posición de la barra.

Fólder "Slider1-Condition"

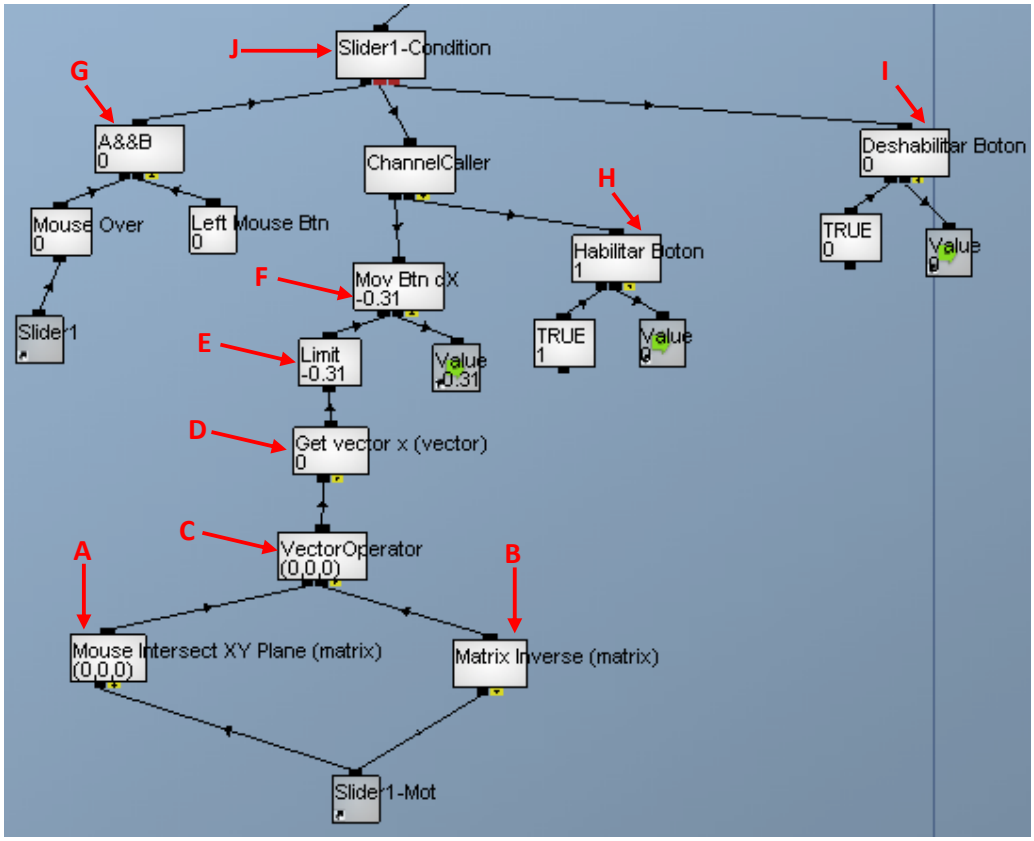


Figura 113. Grupo de canales para hacer la condición de movimiento del cuadrito rojo sobre su eje de las X.

De la figura 113:

- Con el indicativo A, se encuentra el canal "Mouse Intersect XY Plane…" el cual hace una operación matemática entre una matriz y un vector. La finalidad de este canal es que obtenga el punto sobre el plano XY donde interseca el puntero del mouse con la matriz de movimiento de la barra azul que contiene al cuadrito rojo en cuestión. En otras palabras saca el punto sobre el plano XY que se encuentra más cerca del punto de posición del puntero, y lo que hace este canal es que calcula la distancia entre un punto en el espacio y el plano XY, ya que la distancia entre un plano y un punto es la distancia que existe entre el punto más cercano del plano al punto en el espacio en cuestión.

- Con el indicativo B, se encuentra el canal "Matrix Inverse…" el cual calcula la matriz inversa de la matriz de movimiento de la barra azul que contiene al cuadrito rojo en cuestión.

- Con el indicativo C, se encuentra el canal "Vector Operator" el cual hace una multiplicación de un vector con una matriz con la finalidad de transformar el movimiento del vector con respecto a la matriz con la que se multiplica. Su entrada A es el vector de la posición del punto de

intersección entre el mouse y la barra azul sobre el plano XY, y la entrada B es la matriz inversa de la matriz de movimiento de la barra azul, con esta operación lo que estamos haciendo es que el vector se maneje sobre nuevas coordenadas locales espaciales, las cuales están referenciadas hacia la matriz de movimiento de la barra azul, ya con esto, la posición del objeto al que pertenece el vector, su coordenada local siempre será 0, esto quiere decir que, el vector de posición de la barra azul, se vuelve el origen en coordenadas locales de vector de posición del cuadrito rojo.

- Con el indicativo D, se encuentra el canal "Get vector X…" el cual obtiene la componente en X del vector de posición que se conecte a su canal hijo, esto quiere decir que estamos obteniendo la traslación sobre el eje de la X del mouse en las coordenadas locales de la barra azul

- Con el indicativo E, se encuentra el canal "Limit". Este canal transforma el rango de valores de sus entradas por unos rangos de valores nuevos en su salida. Esto quiere decir que, si nosotros le damos los límites inferior y superior de un rango de entrada, podremos decirle a este canal que muestre como rango de salida uno más pequeño o más grande que los rangos de entrada; este canal hará una interpolación lineal de los valores que se encuentren entre el rango de entrada y los asignará a los valores que se generen por la interpolación lineal de los rangos de salida. La finalidad de este canal, es la delimitar el movimiento del cuadrito sobre el eje de las X de las coordenadas locales de la barra azul, esto quiere decir que el cuadrito rojo solo se podrá deslizar entre el rango de valores de -0.31 y 0.31 desde el centro de la barra azul sobre su eje X, no importando que el mouse se pase de esa posición aún estando sobre la barra azul.

- Con el indicativo F, se encuentra el canal "Mov Btn dX". Este canal pega el valor que contiene el canal "Limit" sobre el acceso directo del valor de la componente en X del vector de posición del cuadrito rojo, lo que implica que el cuadrito rojo se moverá sobre la barra azul bajo la posición que ocupe el mouse sobre este último.

- Con el indicativo G, se encuentra el canal "A&&B" el cual hace la operación booleana que su propio nombre indica. Este canal tomará el valor de 1 si sus entradas son verdaderas únicamente. El canal de entrada en A es "Mouse Over" el cual detecta si el mouse está sobre la barra azul, y su entrada B es "Left Mouse Btn" el cual detecta si el botón izquierdo del mouse ha sido apretado. El canal "A&&B" es verdadero (tiene el valor de 1) únicamente si el mouse está sobre la barra azul y ha sido apretado el botón izquierdo del mouse.

- Con el indicativo H, se encuentra el canal "Habilitar Boton" el cual cuando es llamado, pega el valor de 1 al acceso directo que hace referencia al valor de entrada del canal "Operator" de la figura 112 con el indicativo C. Esto quiere decir que cuando es llamado el canal el color del cuadrito rojo se sombrea y se visualiza como si éste último estuviera seleccionado.

- Con el indicativo I, se encuentra el canal "Deshabilitar Boton" el cual cuando es llamado, pega el valor de 0 al acceso directo que hace referencia al valor de entrada del canal "Operator" de la figura 112 con el indicativo C. Esto quiere decir que cuando es llamado el canal, el color del cuadrito rojo se le quita el sombreado y vuelve a su color original.

- Con el indicativo J, se encuentra el canal “Slider1-Condition”. Este canal es una expresión de condición que se asemeja al IF-ELSE. La función de este canal es la siguiente, cuando no se encuentra posicionado el Mouse dentro de la barra azul o se haya hecho un clic con el botón izquierdo del Mouse fuera de la barra azul, el cuadrito rojo se mantendrá en su color original (grupo de canales donde el canal padre es “Deshabilitar Botón”); pero, si el Mouse se posiciona dentro de la barra azul y se hace clic o se mantiene abajo el botón izquierdo del mouse, el color del cuadrito rojo se sombrea y se activa el desplazamiento limitado que tiene éste para moverse sobre el eje de las X dentro de las coordenadas locales de la barra azul.

Menú Completo

Para explicar el menú completo, se debe de explicar el como funciona el canal que genera el Autómata Finito que manipula a nuestro menú.

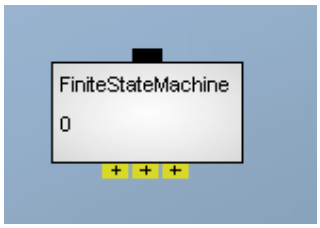
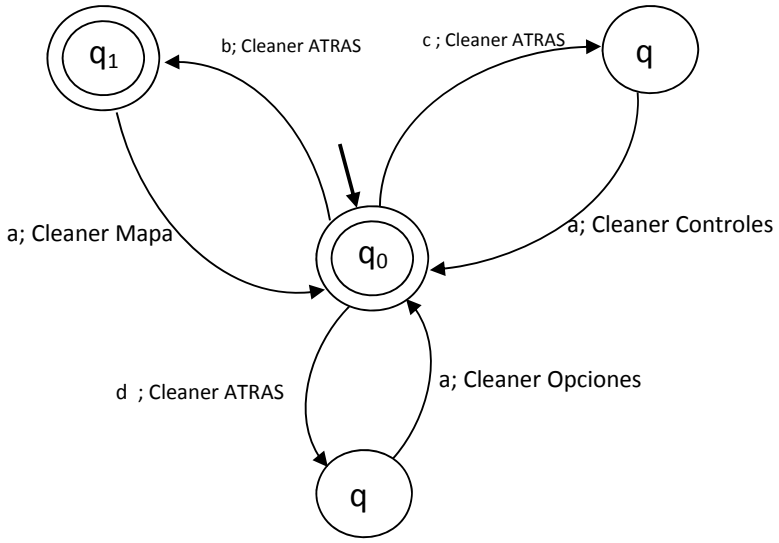


Figura 114. Canal diseñado para elaborar el diagrama de transiciones del Autómata Finito que se necesita para controlar el Menú

El canal de Finite State Machine sirve para generar auxiliarte a elaborar el diagrama de transiciones del Autómata Finito que quieras. Este canal tiene 2 entradas y 1 salida. La primera entrada sirve para que nosotros conectemos canales con valores flotantes que sirven como disparadores para que se efectúe una transición entre estados, en otras palabras, es el alfabeto de palabras que acepta nuestro AF (Autómata Finito). La segunda entrada son grupos de canales que contienen todas las funciones que deberán ejecutarse cuando nos encontremos en determinado estado del diagrama de transiciones. La salida son grupos de canales que ejecutan funciones cuando se está efectuando la transición de un estado a otro.

Recordando, el diagrama de transiciones que se elaboró para poder hacer la manipulación del menú, es necesario acoplarlo al canal Finite State Machine de la siguiente manera



donde

q_0 = Menú Principal	a = Btn ATRÁS
q_1 = Menú Mapa	b = Btn MAPA
q_2 = Menú Controles	c = Btn CONTROLES
q_3 = Menú Opciones	d = Btn OPCIONES

Los valores de 0 y 1 que necesita nuestro canal para generar las transiciones son a, b, c y d.

Del diagrama de transiciones anterior podemos decir lo siguiente, cuando es llamado el Menú dentro del paseo virtual, el estado q_0 es el estado inicial y nos muestra el Menú principal, pero éste estado también es un estado final, lo cual quiere decir que estando en el Menú Principal podremos regresarnos al paseo virtual apretando el botón ATRÁS (a). Si queremos ingresar al Menú Mapa, tenemos que apretar el botón MAPA (b) y toma el valor de 1 para hacer la transición al estado q_1 , y mientras se efectúa la transición se ejecuta la función Cleanear ATRÁS, lo cual quiere decir que el botón ATRÁS(a) tomará el valor de 0 para que no provoque errores. Cuando estamos en q_1 observamos que también es un estado final, esto significa que cuando seleccionamos un lugar de interés dentro del mapa, inmediatamente se termina el Menú y pasamos al paseo virtual. Cuando estamos en el Menú de Mapa y queremos regresarnos al Menú principal, tenemos que apretar el botón ATRÁS(a) para hacerlo, pero mientras se hace la transición se ejecuta la función Cleaner Mapa, lo cual quiere decir que el valor del Botón MAPA (b) toma el valor de 0

y el botón de ATRÁS(a) toma el valor de 1. Este mismo procedimiento lo siguen los dos restantes Menús, lo cual implica que los botones que acciones las transiciones, sólo uno de ellos (a, b, c, d) debe de tener el valor de 1 para poder hacer la transición a la parte del Menú que deseamos, y cuando se ejecuta esa transición, las funciones de Cleaners harán que el valor del botón de la parte del Menú que se sale, le dará el valor de 0, limpiando su valor.

Ahora daremos el grupo de canales que conforman al Autómata Finito.

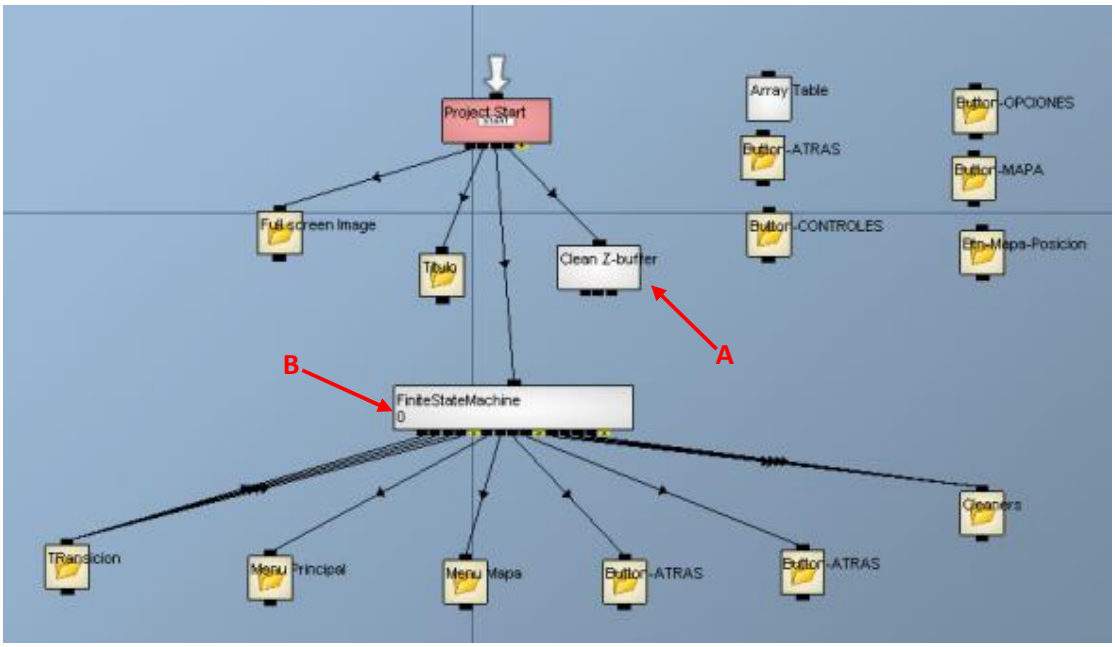


Figura 115. Grupo de canales hecho para elaborar el Menú completo

De la figura 115:

Con el indicador A, se encuentra el canal “Clean Z-buffer” el cual pone todos los valores de la matriz del Z-buffer en 0. Esto sirve para que empecemos a asignarle valores de profundidad a nuestras imágenes y minimizar el traslape de imágenes.

Con el indicador B, se encuentra el canal “FiniteStateMachine” el cual alberga al Autómata Finito del diagrama de transiciones que nosotros hicimos. De este canal solo nos interesa los folders hijos “TRansicion”, “Menu Principal”, “Menu Mapa” y “Cleaners”, ya que los folders hijos restantes se explicaron con detalle anteriormente, y lo mismo aplica para las demás carpetas que están dentro de la figura.

Folder "TRansicion"

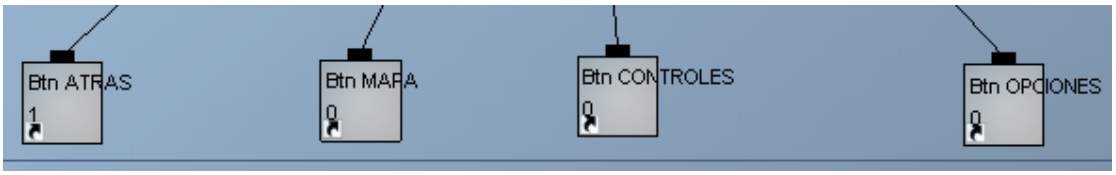


Figura 116. Canales que contiene el valor para autorizar la transición entre estados del diagrama de transiciones

Estos canales contiene el valor de 0 o 1 que autoriza la transición entre estados; sólo uno de ellos puede estar con el valor 1 a la vez, y eso significa que una transición fue hecha por el usuarios al momento de apretar algún botón de Menú.

Folder "Cleaners"

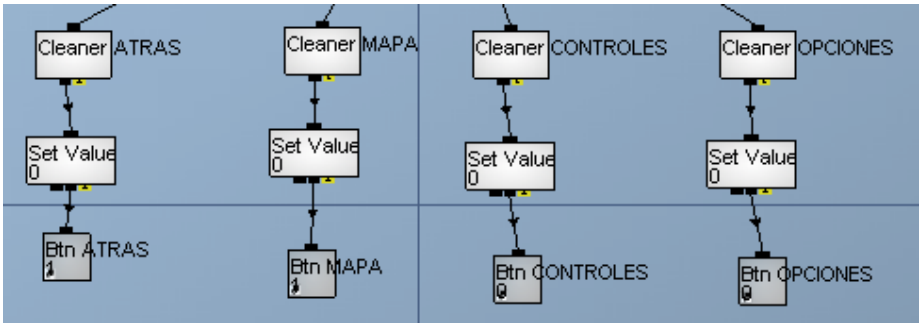


Figura 117. Grupo de canales que contiene el valor de 0 para limpiar el valor de los canales de transición

Podemos observar de la figura 117, que existen cuatro grupos de canales con solo tres canales en cada uno. Estos grupos de canales hacen lo mismo, por lo cual solo explicaremos uno de ellos. Se observa un canal llamado "Cleaner ATRAS" el cual contiene a un canal hijo llamado "Set Value", este último canal es para pegar el valor 0 en el botón de ATRÁS, y eso significa que cuando una transición se está realizando, las función de Cleaner ATRÁS se ejecuta.

Folder "Menu principal"

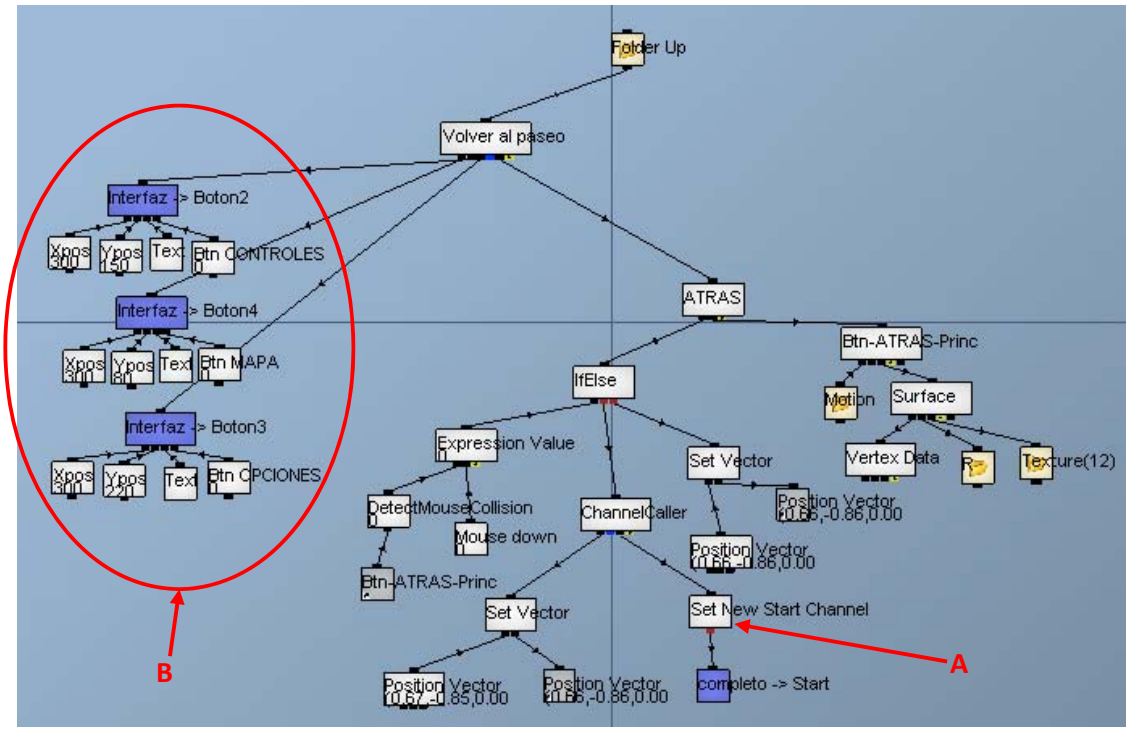


Figura 118. Grupo de canales hechos para elaborar el Menú Principal

Este es el grupo de canales que se emplearon para hacer el Menú Principal, sólo que aquí el botón atrás se modificó porque tenía que ejecutar otra función, y también se agregan las instancias de los Botones Mapa, Controles y Opciones.

De la figura 118:

- Con el indicador A, se encuentra el canal "Set New Start Channel". Este canal es la modificación que se le hizo al botón de ATRÁS original, y sirve para apuntar hacia el padre de otro grupo de canales que debe de ser el canal de inicio del proyecto.

- Con el indicador B, se encuentran los canales "Interfaz -> Boton2", "Interfaz -> Boton4" y "Interfaz -> Boton4". Estos canales son llamados a canales públicos, los cuales son los botones de Controles, Mapa y Opciones respectivamente. Como llaman a los canales públicos que contiene a los canales hechos para los botones, generan una instancia de ellos en nuestra pantalla, y a la vez como estos canales tienen parámetros de entrada (véase figura 104), en el canal de llamada se generan las entradas correspondientes al canal público. Con esto nosotros podemos modificar su posición en la pantalla, el nombre del botón y darle valores a los canales que permiten se genere una transición entre estados.

6.3.- Vista final del paseo virtual

En este subcapítulo, se pretende mostrar parte del trabajo final sobre los autómatas ya implantados en este recorrido.

Cuando estamos dentro del recorrido virtual, hay ocasiones en la que queremos que se mueva o gire más rápido el personaje, bajar el volumen del audio de las narraciones que vienen dentro del recorrido, etc; por lo cual, queremos mostrar lo que ocurre cuando presionamos el botón derecho del mouse dentro del recorrido virtual:

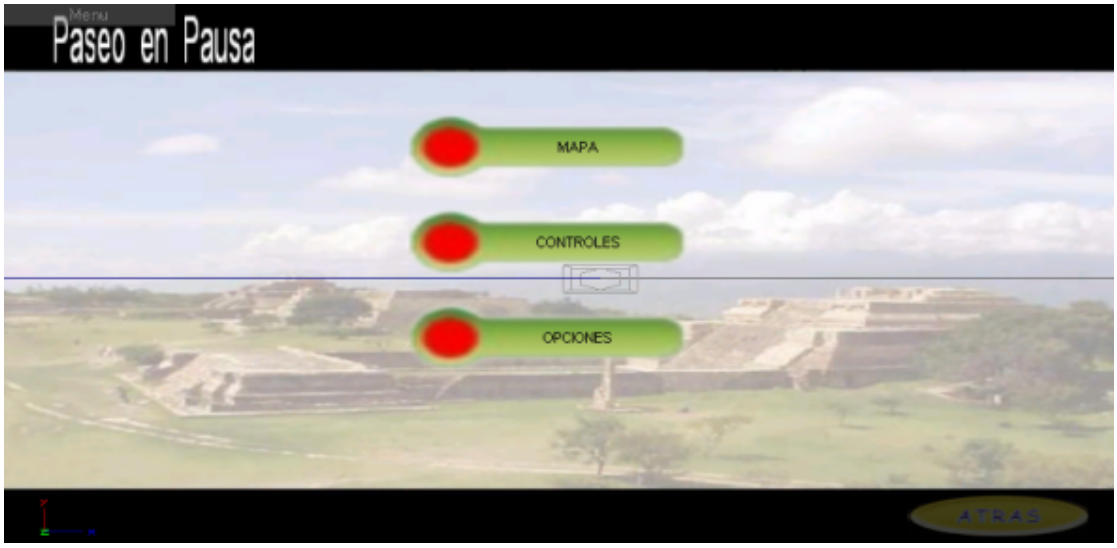


Figura 119. Menú principal. Aquí podemos seleccionar las opciones que queremos modificar, seleccionar u observar. También podemos regresarnos al paseo seleccionando el botón atrás.

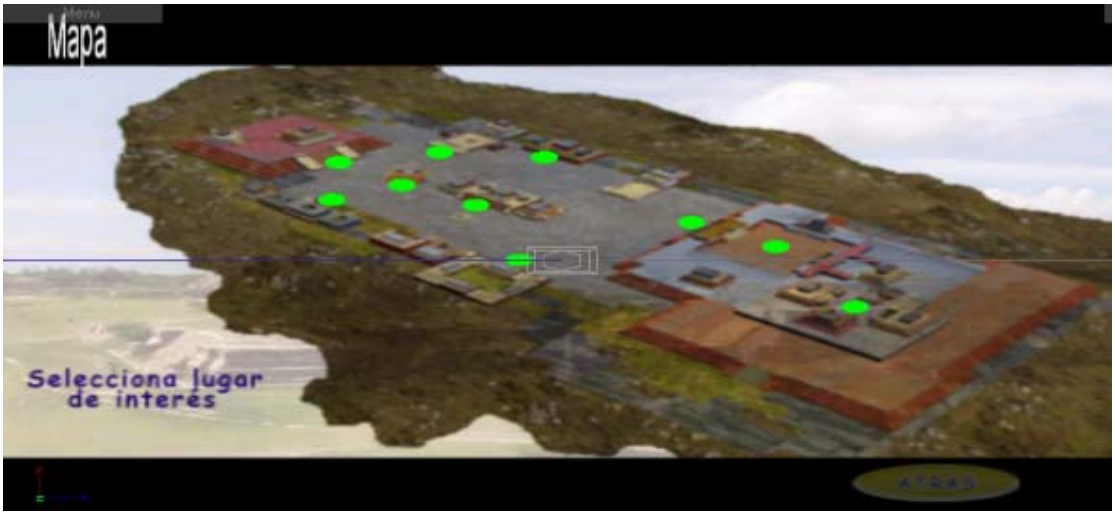


Figura 120. Opción de Mapa. Aquí podemos seleccionar los puntos de interés a los cuales podremos ir rápidamente, y están marcados por un punto de color verde vivo.

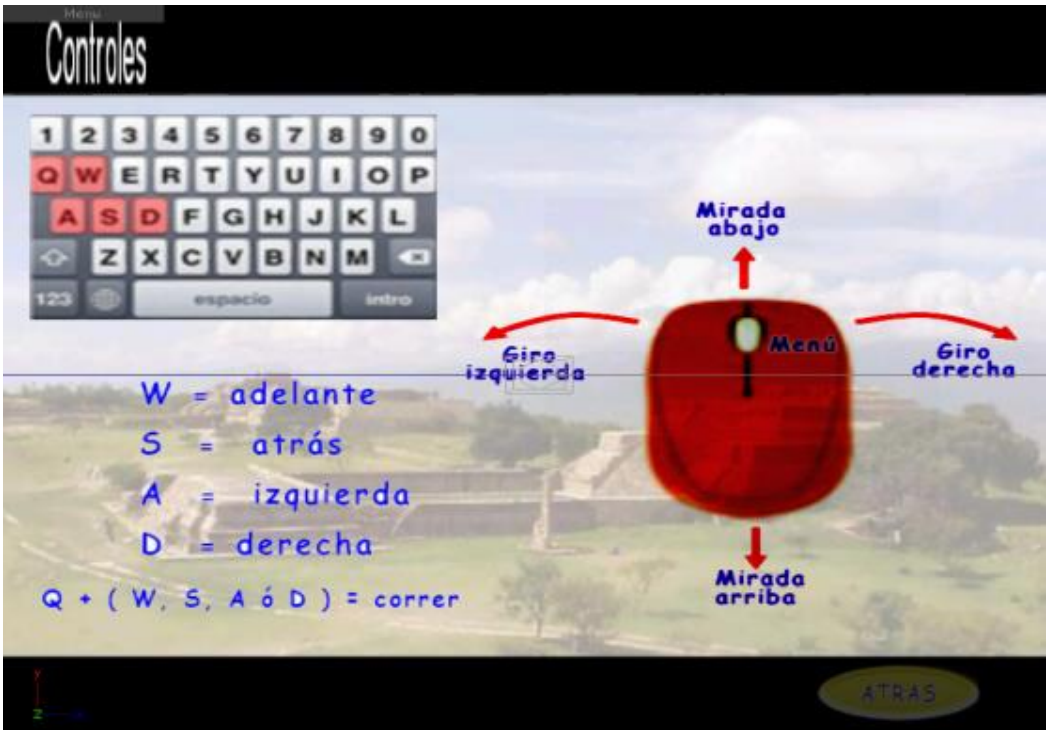


Figura 121. Opción de Controles. Aquí podemos observar las teclas funcionales de control del personaje y también las asignaciones de botones y movimientos del mouse para el personaje.

Figura 122. Opción de Opciones. Aquí podemos observar los "sliders" que manipulan las opciones dadas a la izquierda de cada uno de ellos. Es el control que tenemos sobre algunos atributos del paseo.

Capítulo VII

Evaluación y Pruebas del proyecto

7.1.- Procedimiento

El procedimiento a utilizar está basado en el tipo de cajas blancas o unitarias, pero con una condicionante, que los resultados se basarán en módulos que comprenden un todo de cada aspecto de las entradas; por ejemplo, que los únicos módulos de entrada solo sean los botones del teclado, el mouse en sí y su botón izquierdo.
También se definirán los módulos en base a las entradas que afecto solo al personaje, ya que el personaje es la parte con la que el usuario interactúa, y es una manera resumida de mostrar las pruebas que se realizaron

Primero se designan los datos de entrada:

• Tecla Q	• Tecla W	• Tecla A	• Tecla S
• Tecla D	• Movimiento del Mouse en dirección Horizontal (Dx)	• Botón izquierdo del Mouse (Mi)	• Barra de desplazamiento del menú para rapidez de giro del personaje (Bg)
• Barra de desplazamiento del menú para rapidez de movimiento del personaje (Bd)			

Ahora, designamos los módulos:

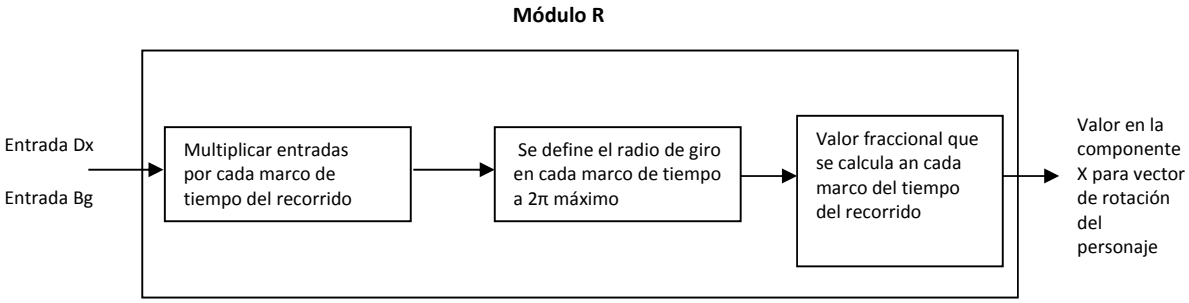
- Módulo R: Vector de Rotación personaje principal
- Módulo S: Vector de Tamaño del personaje principal
- Módulo M: Vector de movimiento del personaje
- Módulo Mx: Matriz de Movimiento del personaje

7.2.- Pruebas

Los niveles de pruebas a las cuales nos sujetaremos son los que a continuación se mencionan y se darán los resultados obtenidos junto con ellas.

Pruebas unitarias

En esta fase de pruebas, lo que se pretende es analizar y descubrir fallas en cada uno de los módulos que se establecieron dentro del procedimiento. Para cada uno de los módulos se analizó las entradas y salidas de cada función o canal que los componen.
Se mostrará a continuación el tipo de prueba unitaria con el Módulo R (Vector de rotación del personaje)



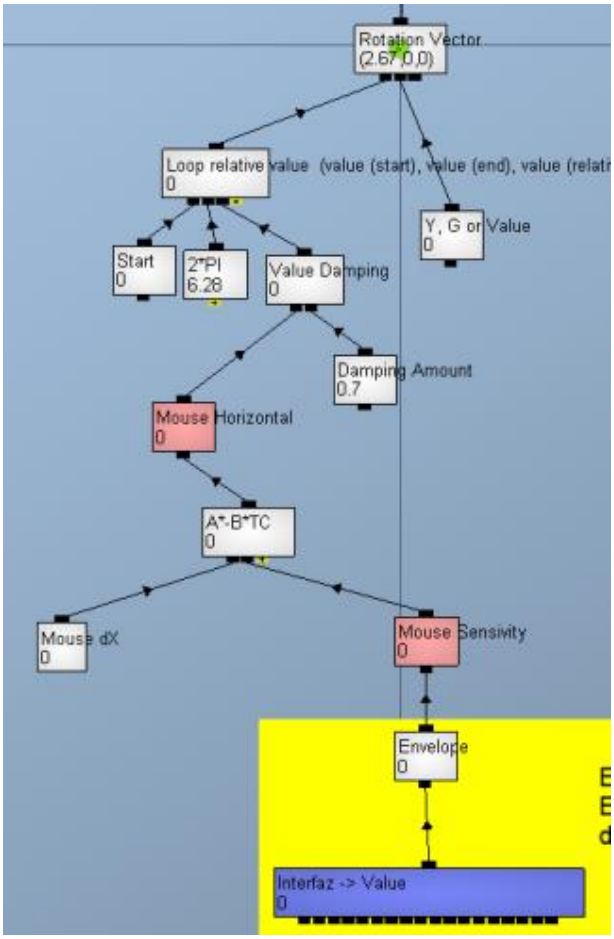
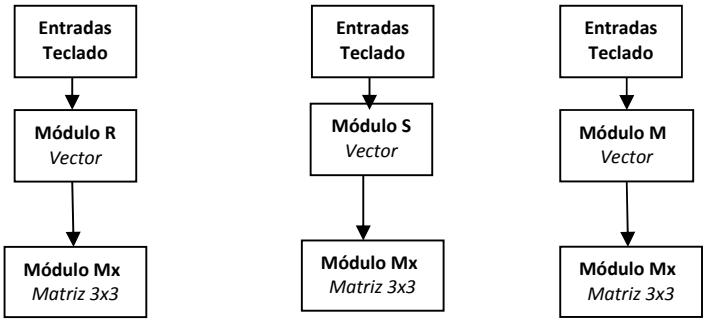


Figura 123. Módulo R (Vector de Rotación del personaje)

Pruebas de integración

Para las pruebas de integración, nos basaremos en solo como se hacen los llamados entre los módulos que modifican la posición del personaje. Las pruebas se hicieron bajo la integración ascendente, las cuales implican los llamados desde los módulos que afectan la movilidad del personaje



En esta integración se observaron los resultados de como los módulos de movimiento afectan la matiz de movimiento del personaje, teniendo como disparadores los dispositivos de entrada, como son el teclado y el mouse.

Una vez que se terminaron de hacer las pruebas bajo un ambiente de pruebas, se hizo lo mismo bajo los módulos del ambiente productivo o real, donde los mismos resultados que se esperaban.

Pruebas de implantación

En la parte de implantación del recorrido, se empezó bajo las siguientes características en cuanto a hardware:

- Procesador Intel Core 2 Duo
- Tarjeta de gráficos Intel Express.
- Memoria RAM de 2GB

Y por la parte de Software, se probó un sistema operativo Windows XP 32 bits.

Bajo estas especificaciones, al momento de ejecutar nuestro recorrido, hubo muchas complicaciones, como son :

- Se tardaba mucho en abrir el archivo.
- Había un retraso al momento de empezar a tocar el sonido ambiental y las narraciones.

- Había un retraso en el movimiento del mouse y presión de las teclas de dirección con la respuesta del personaje en el ambiente virtual.
- El tiempo respuesta de apertura del Menú, era bastante considerable.

En base a los resultados obtenidos de la primera prueba de implantación, se optó buscar Hardware y Software más apropiados para el recorrido, resultando en las siguientes especificaciones mínimas requeridas:

f) Procesador Intel i2.

g) 3 GB en RAM.

h) 100 MB Mínimo de espacio disponible en disco duro.

i) Sistema Operativo XP o superior de 64 bits.

j) Tarjeta de video de 512 MB. Recomendación mínima para mejores resultados GeForce 200 Series o ATI Radeon HD 2000 Series

Pruebas de aceptación

En la fase de las pruebas de aceptación, éstas se establecieron dentro de esta tesis en sí.

Se propuso hacer un recorrido virtual bajo las siguientes características:

- Recorrido y reconstrucción virtual de la Plaza de Monte Albán, Oaxaca.
- Las edificaciones deberán de ser hechas a escala, lo más aproximado posible.
- Se tendrá que contar con información durante el recorrido.
- Se deberá de usar metodologías vistas dentro de la carrera de Ingeniería en Computación.
- Se deberá de elaborar toda la documentación en cuanto el desarrollo y herramientas utilizadas, así como su justificación.

Estos son los puntos importantes que se tuvieron en cuenta para la elaboración de este proyecto de tesis. Los cuales, al final, solo los profesores de la carrera tendrán la última palabra para la aceptación de éste mismo.

7.3.- Evaluación

De acuerdo a las pruebas realizadas en varias computadoras nos percatamos que para obtener un buen rendimiento el equipo debe de contar con las siguientes características:

Características mínimas de la PC
Procesador Intel Core i2
3 GB en RAM
100 MB Mínimo en disco duro
Sistema Operativo XP o superior de 64 bits
Procesador Intel Core duo
Tarjeta de video de 512 MB compartida. Recomendación mínima para mejores resultados GeForce 200 Series o ATI Radeon HD 2000 Series

Figura 124. Características mínimas de la PC.

Ya que si el equipo no cumple con estas características al realizar el paseo nos encontraremos con varias dificultades como que el personaje es más lento, las texturas se ven de una manera muy caricaturesca y pierden su resolución, las explicaciones al acercarse a cada edificio se traban o en el peor de los casos ni siquiera se escuchan, la iluminación de la zona es sumamente mala incluso el paisaje luce opaco y si queremos movernos más rápido con la ayuda del mapa no podremos hacerlo ya que ni siquiera nos deja ver la interfaz.

Capítulo VIII

Resultados y Conclusiones

La Manipulación de los controles, los cuales son fáciles de reconocer como son la letra W del teclado es la que permite el avance del personaje y si presionamos al mismo tiempo la letra W y Q nos permite que el avance sea más rápido, otra cosa que es buena y practica para la manipulación es que los giros del personaje para movernos en la dirección que deseamos son manipulados por el Mouse lo cual hace que su control sea bueno y fácil de manejar para que cualquier usuario tan solo con saber estas simples instrucciones de manejo pueda controlar sin problema alguno al personaje dentro de la Zona Virtual.

Respecto al entorno podemos hablar de que es algo muy parecido a la zona montañosa donde se encontraba la Zona Arqueológica de Monte Alban y podemos observar que la textura de este es buena aunque se podría alcanzar más realismo si en lugar de colocarle una textura que simulara varios árboles, a esta se le colocaran figuras de árboles virtuales independientes, pero en si esta texturización nos permite darnos cuenta de cuál era el entrono de Monte Albán, así como se muestra en la siguiente figura.

Figura 125. Vista isométrica de la Plaza de Monte Albán

Acercándonos más a lo que es la Zona podemos darnos cuenta que las texturas en los edificios son muy buenas ya que cuentan con un realismo bastante bueno ya que incluso en algunas partes podemos observar como parece que crece pasto sobre las rocas como se muestra en la siguiente figura.

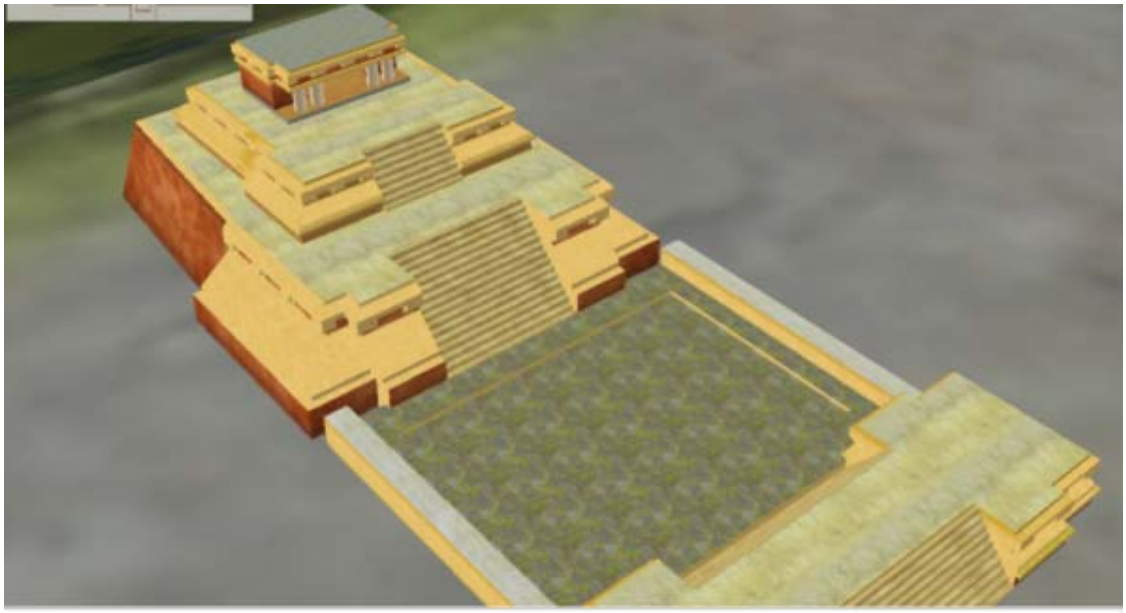


Figura 126. Perspectiva del Edificio J

El giro de la cámara nos permite tener una visión sumamente buena ya que podemos observar tanto el personaje como todo el entrono donde este se desarrolla y así podemos disfrutar viendo el paisaje, los edificios incluso el cielo como lo muestran las siguientes figuras.

Figura 127. Vista de la cámara.

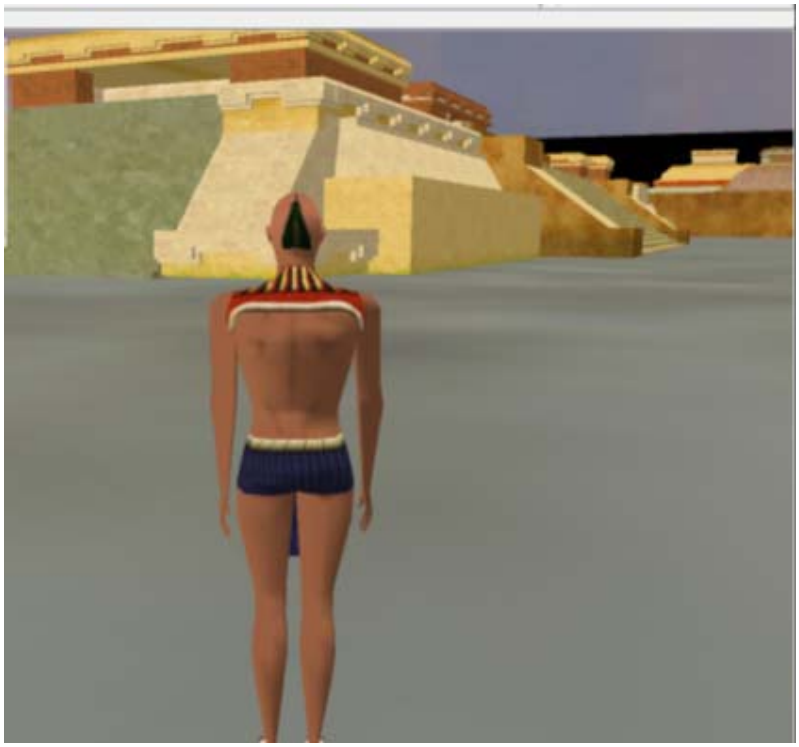


Figura 128. Vista de la cámara desde otro ángulo.

Otra cosa que es muy buena en este proyecto es el pequeño mapa que aparece en la parte superior izquierda, ya que este nos ayuda a saber donde nos encontramos y al igual si no queremos recorrer toda la Zona, podemos seleccionar el lugar a donde queremos dirigirnos de una manera mas rápida, el mapa al que nos referimos podemos observarlo en la siguiente figura.

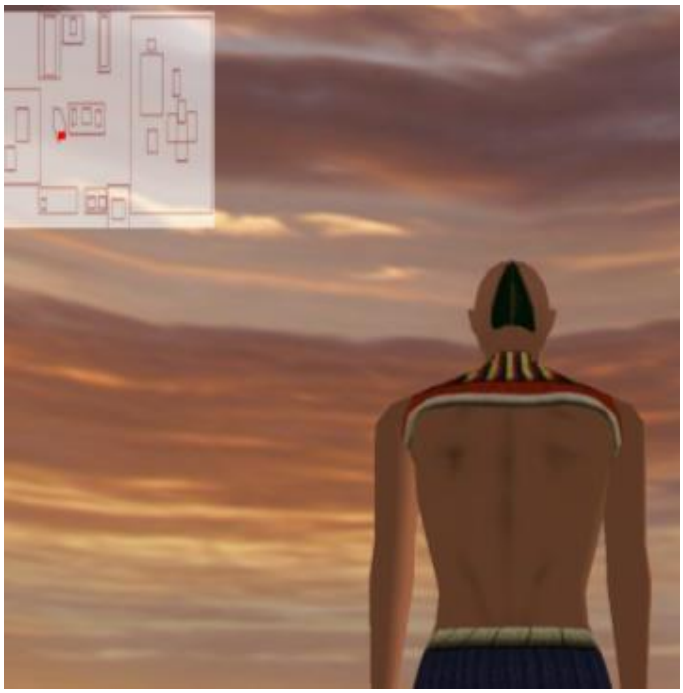


Figura 129. Vista isométrica de la Plaza de Monte Albán

El sonido de este proyecto nos ayuda a entender mas sobre la Zona Arqueológica de Monte Alban ya que conforme vamos haciendo el recorrido y al acercarnos a cada uno de los edificios nos da una explicación de estos de una forma muy clara.

Conclusiones

La elaboración de la tesis presenta de una virtual los gloriosos momentos de la edificación de la Gran Plaza de Monte Albán, haciendo que podamos presenciar y magnificarnos sobre el gran tamaño que ésta tiene, gracias a que su elaboración en el mundo virtual fue hecha en escala.

La implantación de temas vistos en la carrera fueron aplicados, sirviendo como un gran cimiento para la elaboración y ejecución de esta misma tesis; los conocimientos sobre temas del campo de Ingeniería en Computación, como, Ingeniería de Software, Álgebra, Trigonometría, Lenguajes formales y autómatas y otros, fueron los más socorridos y ejecutados en base a las necesidades de nuestra Tesis.

La representación tridimensional de la zona Arqueológica, así como, la elaboración de una interfaz entre el usuario y el mundo virtual, fueron alcanzados, gracias a las herramientas que existen en el mercado para el diseño y realización de un entorno virtual.

El diseño gráfico fue todo un desafío, ya que el poco conocimiento que se tiene al respecto dio como consecuencia el desaprovechamiento del personaje secundario, el cuál iba a tener el propósito de darle más vista al paseo. Dicho personaje no se pudo integrar debido al gran número de polígonos que contiene su malla, y sumados los polígonos del personaje principal excedían el límite de polígonos por escena ideal (10000 polígonos) para tener una óptima respuesta por parte de la tarjeta de video. En otras palabras, no tener pausas durante el recorrido debido a la gran carga de procesamiento que le hubiéramos dado a la GPU (Graphic Processor Unit) con los dos personajes dentro de una misma escena.

Estamos muy contentos con los resultados obtenidos, ya que es recorrido que no se enfoca tanto en la parte de diseño visual, sino en parte de Ingeniería que pudimos implantar en él.

Apéndice de figuras.

figura 4. http://www.escolares.net/matematicas/clasifiquemos-los-poligonos/

figura 5. http://www.escolares.net/matematicas/clasifiquemos-los-poligonos/

figura 6. Cubo transparete: http://blog.utp.edu.co/metalografia/2012/07/30/3-cristalografia/

Cubo verde: http://sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/Tutoriales/tutorial

%20directx%209/html/Luces.htm

figura 7. Fuente: http//es.kioskea.net

figura 8. http://sabia.tic.udc.es/gc/trabajos%202011-12/tbdr/normal.html

figura 9:http://uniruizgutierrez.wordpress.com/2011/09/28/grafos-de-euler-y-hamilton/

figura 10. Objetos cerrados http://commons.wikimedia.org/wiki/File:Basic_shapes.svg

Objeto abiertos: tazon. http://aytuto.blogspot.mx/2013/03/videotutorial-objeto-3d-partir-del-giro.html

Cara:https://www.google.com.mx/search?q=primitivas+geometricas+ricas&source=lnms&tbm=isch&sa=X&ei=WY1TUoymGIrm9ASCq4DICg&ved=0CAcQ_AUoAQ&biw=1422&bih=685&dpr=1#q=rostros+en+3d+max&tbm=isch&facrc=_&imgdii=_&imgrc=LsiJB4VMu06YzM%3A%3B8Yy_ZsdSZ2rv0M%3Bhttp%253A%252F%252Fi1.ytimg.com%252Fvi%252F2hkriK2DI1Q%252Fhqdefault.jpg%3Bhttp%253A%252F%252Fwww.youtube.com%252Fwatch%253Fv%253D2hkriK2DI1Q%3B480%3B360

figura 17. http://janton3.wordpress.com/page/2/

figura 18. http://yaneliekmartin.blogspot.mx/2009/11/que-es-una-reflexion-difusa-y-especular.html

figura 19. http://www.Noticias3D.com

figura 21. http://wiki.blender.org/index.php/Doc:2.6/Manual/Textures/Mapping/UV

Fig.23 .- Museo de Antropología e Historia. Coordinación Monte Albán. Proyección Vertical Zona completa para maqueta central de Sala Monte Albán

Fig.24 .- Museo de Antropología e Historia. Coordinación Monte Albán. Proyección frontal edificio Zona B

Bibliografía

1.- Monte Albán y los Zapotecos. "Pasajes de la Historia". Tomo #3. Editorial Conaculta-Mexico Desconocido, México, 2000. 1ra Edición.

2.- Revista Creatividad y Sociedad, Francisco Javier Pérez Martínez: Presente y Futuro de la Tecnología de la Realidad Virtual.

3.- http://www.lanacion.com.ar/1619280-jorge-alvarez-me-hubiera-gustado-ser-un-mafioso

4.-http://www.injuve.es/sites/default/files/RJ92-16.pdf

5.- Revista New york university, Ken Perlin , mrl.

6.-Revista digital universitaria,Biblioteca de Edicion Digital,Articulos digitales,Jesus R. Burguete,Mari Carmen González.

7.-http://www.diversity.ku.edu/sites/diversity.drupal.ku.edu/files/docs/outlook-2009-11-alt.pdf

8.-Living Art,Aziosmanoff, Florent,libreria Canoppe.

9.-http://www.uladech.edu.pe/documentos/la-educacion-a-distancia-en-el-peru.pdf

10.- http://www.fraternidadrosacruzmadrid.com/fmnacher/libros/Que%20pasa%20cuando%20nos%20morimos.pdf

11.- Criterios de Intervención en el Patrimonio Arquitectónico del Siglo XX. Calameo.

12.- http://www.google.com.mx/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=6&ved=0CFYQFjAF&url=http%3A%2F%2Fwww.elistas.net%2Fcgi-bin%2FeGruposDMime.cgi%3FK9D9K9Q8L8xumopxCkdqz%3FdnuwqlludnqCTQVRCtjogkmCnoqdy-qlhhyCUSggb7&ei=Ci5wUpP9B-rO2gXM3IGgBA&usg=AFQjCNEHYNwIlTwPScOm8_0F-bZdyJaKKA&sig2=NTK3snttpFq822GFFwRiwA

13.-http://www.slideshare.net/gladysmargie/realidad-virtual-15822449

14. –Para Quest 3d:Bisschop, J., (2007), *AIMMS Optimization Modeling,* Paragon Decision Technology B.V. (USA).

Para 3DStudioMax: ANIMACIÓN CON 3DS MAX, Mccarthy, M. ; Bousquet, Michele

Para SketchUP: SketchUp Pro 7 paso por paso en español,Joao Gaspar. VectorPro.

Para PhotoShop: Photoshop CS4 Avanzado – Ben Willmore – Anaya.

Para FaceGen: http://en.wikipedia.org/wiki/FaceGen.

15.-http://www.grupodircom.com/archivos/ebook-libro-docencia-y-comunicacion-en-latinoamerica-grupo-dircom.pdf

Glosario

AGP: El bus AGP (Accelerated Graphics Port) o Puerto de Gráficos Acelerado se ha creado con el único propósito de conectarle una tarjeta de video, funciona al seleccionar en la tarjeta gráfica un canal de acceso directo a la memoria, evitado así el uso del controlador de entradas/salidas. Las tarjetas que emplean este bus de gráficos requieren de menos memoria integrada ya que cuentan con un acceso directo a la información gráfica.

API: Conocido por sus siglas en ingles Application Programming Interface (Interfaz de programación de aplicaciones), es el conjunto de funciones y métodos, en la programación orientada a objetos que brinda cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas.

Blueprint: Es la representación gráfica de un objeto, donde se muestra de forma detallada su diseño en diferentes perspectivas, como lo son las vistas frontal, lateral, isométrica, etc.

Buffer-Z: En los gráficos por computadora, el Buffer Z es la parte de la memoria de un adaptador de video encargada de administrar las coordenadas de profundidad de las imágenes en los gráficos en tres dimensiones. Este Buffer es una de las soluciones al problema de determinar qué elementos de una escena renderizada son visibles y cuales ocultos. Al Buffer Z también se le conoce como buffering de profundidad.

GPU (Graphic Processor Unit): La unidad de Procesamiento Gráfico, mejor conocida por sus siglas en inglés (GPU), es la unidad encarda de hacer los cálculos matemáticos para la ejecutar y simular un entorno virtual de 3D o 2D.

PCI: Un PCI o Interconexión de Componentes Periféricos (Peripheral Component Interconnect) es un bus estándar para conectar dispositivos periféricos directamente a su placa base. Estos dispositivos pueden ser circuitos integrados o tarjetas de expansión que se ajustan en conectores.

PCIe 1.1: Una ranura PCIe 1.1 es un nuevo desarrollo del bus PCI que usa los conceptos de programación y los estándares de comunicación existentes, pero se basa en un sistema de comunicación en serie mucho más rápido, en una PCIe 1.1 cada carril transporta 250 MB/s en cada dirección.

PCIe 3.0: Una ranura PCIe 3.0 es una evolución de la ranura PCIe 1.1 y PCIe 2.0, donde cada carril transporta 1GB/s en cada dirección.

Renderizado: (render en inglés) es un término usado para referirse al proceso de crear una imagen o vídeo mediante el cálculo de iluminación partiendo de un modelo en 3D . Este término es utilizado en programas de diseño en 3D como por ejemplo 3DMax, Maya, Blender, etc.

Texturización: Las áreas poligonales pueden contener datos correspondientes de más de un color, pero en el software más avanzado, pueden ser una imagen rasterizada (una imagen rasterizada es un fichero de datos que representa puntos de color, denominada matriz, que se puede visualizar en un monitor), donde la imagen es colocada en una cara, o en una serie de caras y es llamada Textura.