PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

SEDE AMBATO

ESCUELA DE INGENIERÍA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS

"APLICACIÓN DE REALIDAD VIRTUAL INMERSIVA ORIENTADA A LA ARQUITECTURA"

ENRIQUE XAVIER GARCÉS FREIRE
MAURICIO ALEJANDRO SALAZAR MORALES

DIRECTOR DE LA DISERTACIÓN:

ING. SANTIAGO ACURIO

AMBATO, 2004

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

SEDE AMBATO

ESCUELA DE INGENIERÍA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TíTULO DE INGENIERÍA DE SISTEMAS

"APLICACIÓN DE REALIDAD VIRTUAL INMERSIVA ORIENTADA A LA ARQUITECTURA"

DIRECTOR:

Ing. Santiago Acurio

ENRIQUE XAVIER GARCÉS FREIRE

MAURICIO ALEJANDRO SALAZAR MORALES

DEDICATORIA

Todo trabajo, todo proyecto, sueño o anhelo tiene un motor que apoya y guía por el camino a la cristalización y materialización de las ideas planteadas; es mi propósito entonces hacer un pequeño y justo homenaje al motor que me ha llevado a culminar esta etapa de mi vida.

En primer Lugar dedicarlo a Dios, pues todos necesitamos algo o alguien en quien creer, y Dios es mi Padre y Señor.

Lo dedico a mis padres, hermanos, sobrinos y tíos que me han acompañado de corazón, y me han guiado para que hoy esté donde estoy; haciendo cuanto esfuerzo sea necesario por mi porvenir.

Espero no defraudarlos y recompensar su esfuerzo.

Enrique Garcés

DEDICATORIA

En la vida todo lo que comienza tiene que terminarse, mi esfuerzos y sacrificio lo dedico especialmente a Dios, sin su ayuda esta dedicatoria no existiría, a mis padres William y Eulalia, por apoyarme incondicionalmente en todos los momentos de mi etapa estudiantil y de este proyecto, a mi esposa Carmi por darme la fuerza y el apoyo para sacar adelante los momentos difíciles del proyecto, muchas gracias por todo los AMO.

Mauri

AGRADECIMIENTO

Al alcanzar un nivel o una meta, existe siempre un grupo de personas que han compartido los momentos gratos y desventurados, y es justo darles un agradecimiento, claro esta que el mencionarlos a todos podría llevarme a redactar un libro entero.

Agradezco pues a Mauricio Salazar, por formar el 50% del grupo de trabajo que finalizo este sueño, a Zlaly por ser la persona que me apoyó siempre y me dió fuerzas cuando la montaña se veía difícil de escalarla; a nuestro director de tesis Ing. Santiago Acurio por guiarnos y darnos una mano para adentrarnos en este mundo. Al Ing. Telmo Viteri por creer en el sueño idealista nuestro y permitirnos hacerlo público.

A todos esos amigos que han compartido escenarios, conciertos y repasos de música que me han ayudado a descargar energía.

A mis compañeros, profesores, amigos y amigas que de una u otra forma me han acompañado y apoyado.
Un agradecimiento final a todos los que son parte del juego experimental y a Buffy por pasar junto a nosotros.

Enrique Garcés

AGRADECIMIENTO

Este proyecto no sería lo mismo sin su ayuda, agradezco en primer lugar de Dios, a mis padres William y Eulalia son los mejores, mi esposa Carmi gracias por su amor, mi hermano Andrés por la ayuda en las pruebas, mi compañero de tesis Enrique por toda la comprensión y apoyo gracias amigo, a Buffy por acompañarnos en todo el desarrollo de la tesis.

No me puedo olvidar del Ing. Santiago Acurio por la guía y ayuda en el desarrollo del proyecto, al Ing. Telmo Viteri por creer en nosotros, a mis maestros por sus enseñanzas, mis compañeros por brindarme su amistad, y a todos los que nunca creyeron en nosotros, para ellos un agradecimiento especial.

Muchas gracias de corazón.

Mauri

ÍNDICE

INTRODUCCIÓN

A finales de los años 80, los gráficos por computadora entraron en una nueva época, los gráficos tridimensionales (3D) salieron a la luz, esto no significó que comenzaron a reemplazar los enfoques bidimensionales o de dibujos lineales (2D), sino que existía la necesidad de dotar a los usuarios un espacio de trabajo totalmente interactivo generado a través de la tecnología, esta nueva herramienta de dibujo y modelado por computadora fue adoptada por los profesionales de la arquitectura pues les brinda una mejor presentación de sus proyectos y facilita una recreación tridimensional de la edificación.

Los avances tecnológicos no han llegado a la cúspide, la Realidad Virtual, hija joven de la informática y tal vez la más controvertida, y desconocida de esta ciencia, es poco explotada inclusive en países de primer mundo. Se podría describir como "La simulación de medios ambientes y de los mecanismos sensoriales del hombre por computadora, de tal manera que se busca proporcionar al usuario la sensación de inmersión y la capacidad de interacción con medios ambientes artificiales" (Universidad J.F. KENNEDY, 1998). Puede ser aplicada en un sinnúmero de campos de la ciencia humana, tales como: Medicina, Psicología, Arquitectura, Educación, Adiestramiento Profesional, Simulaciones y Entretenimiento, entre otras.

El presente trabajo investigativo pretende mostrar esta puerta de entrada a ese misterioso y fascinante mundo donde las cosas no existen pero se ven; donde el movimiento de los objetos, su apariencia, su forma; son creados a nuestra voluntad y sólo la imaginación del programador o del diseñador limita la creación del espacio virtual.

Las características de la Realidad Virtual que cubre este trabajo son las siguientes:

- Inmersión.- Podrá ingresar a mundos virtuales tomados desde un Archivo de Intercambio de Dibujo 3D (3DS), mediante unas gafas especiales para Realidad Virtual.
- Existencia de un punto de observación o referencia.- El usuario tendrá un control de su ubicación en el mundo virtual.
- Navegación.- El usuario podrá moverse a placer por el mundo virtual mediante un teclado estándar de computadora y un mouse.

La manipulación no será cubierta por está investigación por razones de tiempo y costo en los equipos que son necesarios.

LO REAL SON IMPULSOS ELÉCTRICOS QUE TU CEREBRO INTERPRETA

CAPÍTULO I

1. PROYECTO DE INVESTIGACIÓN.

1.1. PROBLEMA Y PROBLEMATIZACIÓN.

Para el presente Proyecto se ha planteado el siguiente problema general:

¿La no disponibilidad de un software que utilice Realidad Virtual en el área del diseño arquitectónico, ha causado que en el medio los proyectos no sean percibidos con claridad por parte del cliente, provocando la demora en su aceptación y la desmotivación del profesional involucrado en ese tipo de trabajo. Y el no contar con un software de este tipo en la Escuela de Ingeniería de Sistemas de la Pontificia Universidad Católica del Ecuador Sede Ambato?

Problematizando lo expuesto anteriormente:

¿Escaso conocimiento en manejo del Formato de Archivos de 3D Studio (3DS) con herramientas de programación?

¿Ausencia de ambientes virtuales inmersivos en la Escuela de Sistemas de la PUCESA?

¿Desinterés sobre Realidad Virtual inmersiva en la Escuela de Sistemas de la PUCESA?

1.2. DELIMITACIÓN.

La investigación será desarrollada en la Escuela de Ingeniería de Sistemas de la Pontificia Universidad Católica del Ecuador Sede Ambato, en el período de tiempo entre marzo del 2003 y julio del 2004.

Entre las características que son parte de la Realidad Virtual nuestra aplicación cubre:

- Inmersión.- Podrá ingresar a mundos virtuales tomados desde un Archivo de Dibujo (3DS), mediante un I-Glasses especial para Realidad Virtual.
- Existencia de un punto de observación o referencia.- El usuario tendrá un control de su ubicación en el mundo virtual.
- Navegación.- El usuario podrá moverse por el mundo virtual mediante un teclado y mouse de computadora.

La manipulación no será cubierta por está investigación por razones de tiempo y costo en los equipos que son necesarios.

Para cargar los archivos la aplicación partirá de ficheros en formato 3DS, que permitan la exportación a ficheros X nativos de DirectX.

Por lo tanto el tema de la investigación será:

"APLICACIÓN DE REALIDAD VIRTUAL INMERSIVA ORIENTADA A LA ARQUITECTURA"

1.3. OBJETIVOS.

1.3.1. OBJETIVO GENERAL.

- Generar una aplicación 3D inmersiva utilizando técnicas de Realidad Virtual para usos arquitectónicos.

1.3.2. OBJETIVOS ESPECÍFICOS.

- Aplicar librerías Microsoft Direct X 8.0 SDK con Microsoft Visual Basic 6.0 y manejo de Archivos de Dibujo (3DS) transformados a formato X.
- Cargar Ambientes Virtuales Inmersivos.
- Sembrar bases de estudio sobre 3D y Realidad Virtual Inmersiva en la Escuela de Sistemas de la PUCESA.

1.4. HIPÓTESIS.

- La utilización de un software inmersivo en tres dimensiones facilitará la toma de decisiones oportunas en la realización de proyectos arquitectónicos a los profesionales de la arquitectura y el cliente.

1.5. METODOLOGÍAS DE INVESTIGACIÓN.

La metodología nos permitirá ordenar las actividades del proceso investigativo, la misma que tiene:

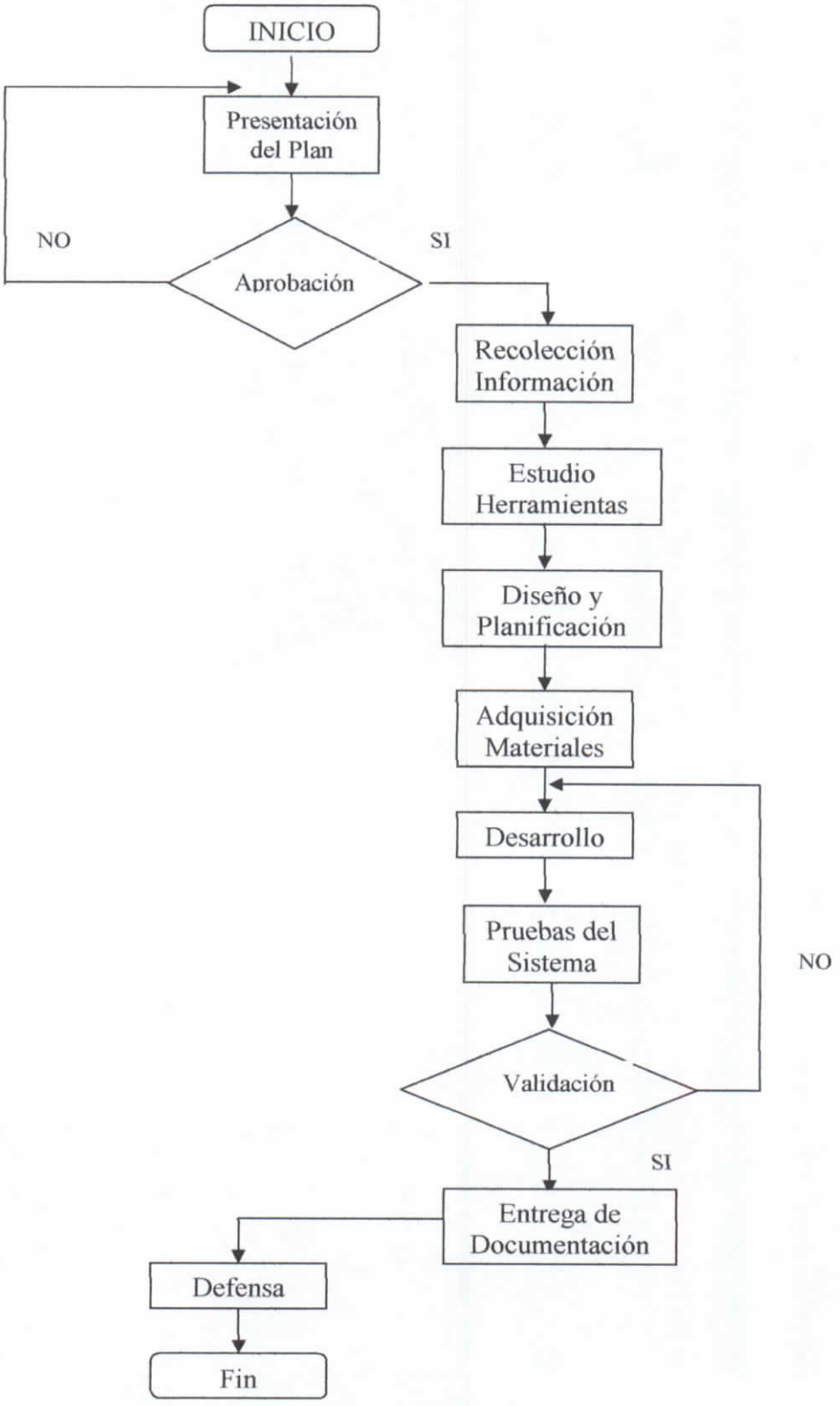
PARADIGMA: Racionalista y Pragmática.- Racionalista ya que se parte de la idea de lograr inmersión mediante Realidad Virtual sin basarse en trabajos anteriormente hechos; y Pragmática porque el proyecto está estrechamente relacionado entre la teoría y la práctica.

TIPO DE INVESTIGACIÓN: Aplicada.- Porque se va a realizar en un lugar adecuado para el desarrollo de la aplicación como son Laboratorios Informáticos y Bibliotecas.

NIVEL DE INVESTIGACIÓN: Diagnóstico, Descriptivo y Explicativo.- Debido a que se diagnostica la manera en que se realizan los procesos, además se describe los mismos y se explican desglosados en una documentación.

TÉCNICAS DE LA INVESTIGACIÓN: Fichaje y Entrevistas.- Porque es una técnica empleada en la Investigación Científica, que permite ir documentando datos de la información, obtenida en el transcurso de la investigación; y entrevistas no formales porque eventualmente se realizarán conversaciones con profesionales tanto de Informática como en el área de la Arquitectura.

1.6. DIAGRAMA DE PROCEDIMIENTOS.



CAPÍTULO II

2. MARCO TEÓRICO.

2.1 ARQUITECTURA EN ESPACIOS 3D.

2.1.1. HISTORIA.

Hace tan sólo 25 años, todos los dibujos se ejecutaban utilizando lápiz y papel. Cuando se precisaba realizar cambios, era necesario borrar y volver a dibujar. Si el cambio era importante, se repetía el dibujo por completo. Si un cambio afectaba a otros documentos se tenía que buscar a mano en cada uno de ellos y modificarlos.

El Diseño Asistido por Computadora cuyas siglas en inglés son CAD (Computer Aided Design), modifica por completo el método tradicional de dibujo por uno más ágil y sofisticado, mejorando todos los aspectos del diseño arquitectónico. Al principio esta nueva herramienta trabajaba bajo dibujo 2D (2 dimensiones, ancho y alto), para evolucionar luego en un sistema capaz de representar el diseño en 3D (3 dimensiones, ancho, alto y profundidad), pudiendo de esta manera comprender el diseño terminado sin necesidad de armar maquetas y con igual resultado.

A principios de la década de los 70, varias compañías empiezan a desarrollar sistemas de dibujo automatizados, y muchas empresas especialmente del sector automotriz y aeroespacial adoptan estos sistemas, muchos de los productos que manejamos en la actualidad tuvieron sus comienzos en este período, y comienzan a mostrar algunas capacidades 3D.

En los años 80, se crea la compañía Autodesk, lanzando al mercado el primer AutoCad que corre sobre una PC, convirtiéndose en el sistema gráfico más popular, sobre otras compañías que empiezan a seguir los pasos de esta, y empiezan a desarrollarse los primeros sistemas GIS (Geographical Information Systems).

Los años 90, marcan el inicio y se generalizan las visualizaciones en 3D, se comercializan gran variedad de programas para diferentes aplicaciones en diferentes campos, sobresaliendo el Autocad 12.

A finales de esta década Autodesk crea el sistema 3D Studio, programa de diseño netamente en 3D, y alcanza cifras de venta de 100.000 usuarios y 1.000.000 en su sistema Autocad.

2.1.2. TENDENCIAS.

Las tecnologías de sistemas gráficos se encuentran ya en una fase de madurez. Su utilidad es indiscutible y han abierto posibilidades para el rediseño y fabricación de objetos o piezas, construcciones arquitectónicas, impensables sin estas herramientas.

La visualización realista en estas aplicaciones destacan los defectos como zonas de sombras, tal como si se observara el objeto real. Incluso se pueden mostrar las superficies coloreándolas según la curvatura, con lo que las posibles discontinuidades aparecen de inmediato. Estos métodos expuestos corrigen alrededor del 90% de los defectos que puedan existir.

2.2. GEOMETRÍA ESPACIAL O GEOMETRÍA DESCRIPTIVA.

2.2.1. INTRODUCCIÓN A LA GEOMETRÍA ESPACIAL.

2.2.1.1. Historia.

La Geometría Espacial o conocida también como Geometría Descriptiva existía antes de ser inventada. La complejidad de los cortes de la piedra o la madera ha requerido siempre el uso de proyecciones ortogonales, y sin embargo el sistema diédrico es relativamente moderno. La perspectiva cónica nació de un proceso artístico lento.

Gaspar Monge (1746-1818 Francia) fué el fundador de esta rama de las matemáticas, la expresión escogida para designar a esta materia fué Geometría Descriptiva, perseguía aprovechar el prestigio de la llamada Geometría Analítica, contrastando con ella.

2.2.1.2. Concepto.

La Geometría Descriptiva estudia la forma de representar los cuerpos de tres dimensiones sobre el plano.

2.2.2. PROYECCIONES.

2.2.2.1. Introducción.

El dibujado en 3D resulta más complejo que el 2D debido a las siguientes razones:

- Se tiene una dimensión más (eje z).
- Los medios (dispositivos) de visualización son 2D.
- Se introduce una fase más en el proceso de dibujo, que es la de proyección

El paso de proyección es necesario para traducir de coordenadas 3D a 2D, de forma que la fase de dibujo sea como la bidimensional. En la Fig. 2.1. se muestra los componentes de una proyección.

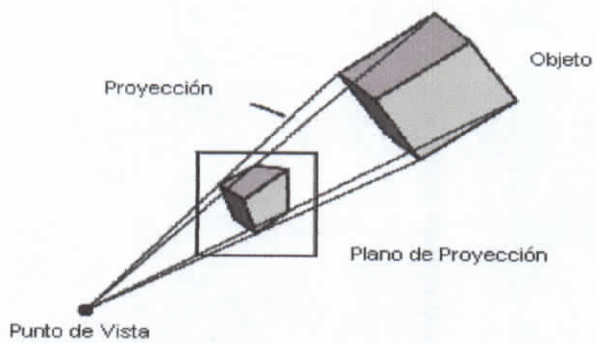


Fig. 2.1.

2.2.2.2 Concepto.

Una proyección es una transformación de unos puntos que están en un sistema de coordenadas de dimensión **n** a un sistema de coordenadas de dimensión menor que **n**.

El centro de proyección es el punto de vista desde donde se mira la figura. El plano de proyección es un plano determinado por un punto en él llamado punto de referencia, y una perpendicular a él.

2.2.2.3 Tipos de Proyecciones.

Tenemos dos tipos de proyecciones: Perspectivas y Paralelas. La diferencia entre ellas nos la da el centro de proyección.

En las perspectivas el centro de proyección está a una distancia finita del plano de proyección, mientras que en las paralelas está a una distancia infinita de dicho plano. Debido a esto mientras las proyecciones en perspectiva se especifican o definen por su centro de proyección, las paralelas lo hacen por la dirección de proyección.

La proyección de perspectiva parece más realista y queda mejor en pantalla que la paralela, pero no es buena para medir distancias, no se mantienen los ángulos en las figuras y las líneas paralelas de los objetos seguramente no se mantendrán paralelas, y esto último si lo hacen las proyecciones paralelas.

2.2.3. GEOMETRÍA 3D.

2.2.3.1. Modelos de Superficies.

Existen varias razones para querer representar un objeto mediante un modelo de superficie:

Cuando el objeto mismo es una superficie que podemos suponer sin grosor. Este tipo de representación nos permite visualizar superficies abiertas, mientras que los sólidos se caracterizarán por tener su superficie necesariamente cerrada sobre sí misma.

Cuando tan sólo nos interesa visualizar su aspecto visual externo, sin detalles sobre su estructura interna, aunque el objeto ocupe un cierto volumen.

Cuando se desea realizar una visualización en tiempo real, y para ello utilizamos hardware o software gráfico que está sólo preparado para visualizar polígonos.

En cualquiera de estos casos es conveniente utilizar una representación de la superficie del objeto. En principio la información sobre una superficie debe dar cuenta de su geometría, de sus propiedades visuales (cómo se comporta frente a la luz y quizás también de alguna propiedad física, como la elasticidad) si se va a efectuar una simulación física sobre el objeto.

2.2.3.2. Mallas Poligonales (Primitivas Poligonales).

Las primitivas o mallas poligonales son grupos de polígonos que se describen de forma conjunta para ahorrar espacio de almacenamiento y coste de visualización en tiempo real, razón por la cual son ampliamente utilizadas.

Algunas de las primitivas más utilizadas son:

- Tira de cuadriláteros: Los primeros cuatro vértices definen el primer cuadrilátero, y cada nuevo par define otro cuadrilátero formado por éste par de vértices y el anterior (Ver Fig. 2.2.).

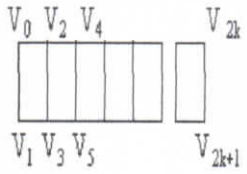


Fig. 2.2.

- Tira de triángulos: Con los tres primeros puntos se construye un triángulo y los demás se forman añadiendo sucesivos puntos. Cada nuevo triángulo se forma por los tres últimos vértices añadidos, de tal forma que con **n** puntos se obtienen **n-2** triángulos. (Ver Fig. 2.3.).

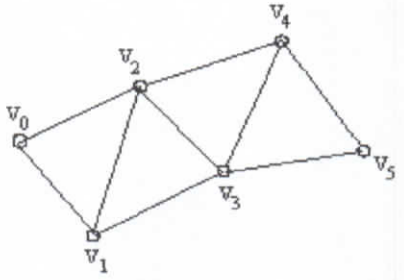


Fig. 2.3.

- Abanico (Fan): Se da un primer punto y luego el resto siguiendo un abanico. Aparecen **n-2** triángulos formados por el primer vértice y los vértices **i**, **i+1** (i diferente de 1). Se emplea para conseguir formas imposibles de lograr con la tira de triángulos. Esta primitiva, al igual que la anterior, tiene la ventaja de que no es necesario comprobar la coplanaridad que es el que dos triángulos compartan en lado en común (Ver Fig. 2.4.).

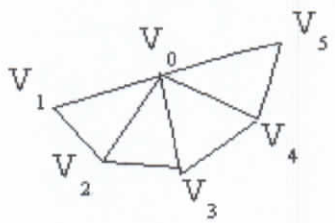


Fig. 2.4.

En realidad, la mayor parte de aplicaciones y librerías gráficas descomponen posteriormente todas estas primitivas en triángulos durante el proceso de visualización.

2.3. REALIDAD VIRTUAL.

2.3.1. HISTORIA Y DESCRIPCIÓN.

2.3.1.1. Introducción.

La idea de Realidad Virtual lleva rondando hace unos 39 años, pero es ahora cuando la expresión está aflorando en todas partes, en revistas de moda, en cadenas de televisión por cable, en centros universitarios y ahora en la Universidad Católica como parte de esta ola tecnológica de mundos virtuales; es preciso dar un poco de historia de lo que han sido los sucesos más relevantes en estos años de vida de la Realidad Virtual se detallan los eventos por medio de una cronología.

2.3.1.2. Historia.

La exploración en el campo de la Realidad Virtual se inició con los experimentos de un grupo de investigación de la Universidad de Harvard encabezado por Ivan Sutherland. Este grupo de investigadores diseñó el primer casco, conocido como Incredible Helmet (casco increíble), un dispositivo bastante rudimentario con dos tubos de rayos catódicos, por su construcción el dispositivo presentaba numerosos problemas de movilidad y comodidad.

Las limitaciones del hardware existentes en aquella época ocasionaron que las primeras tecnologías resultaran poco convincentes. Los ambientes se creaban usando el sistema de generación de gráficos vectoriales más avanzado del momento sin embargo, solo se logró producir la sensación de estar en un mundo de objetos que parecían estar hechos de alambre, y la ilusión de inmersión era insuficiente.

Si bien las primeras exploraciones en el campo de la Realidad Virtual no fueron exitosas, sirvieron para demostrar que era posible llegar a una mayor evolución a futuro. Como consecuencia, numerosas empresas y centros de investigaciones civiles y militares mostraron interés en su desarrollo.

• 1965.- Surge el concepto de Realidad Virtual, cuando Ivan Sutherland (hoy miembro de Sun Microsystems Laboratories) publicó un artículo titulado "The Ultimate Display" (El Ultimo Visualizador), en el cual describía el concepto básico de la Realidad Virtual

• 1966.- Sutherland creó el primer casco visor de Realidad Virtual (Incredible Helmet)

• 1968.- Ivan Sutherland y David Evans crean el primer generador de escenarios con imágenes tridimensionales.

La primera adaptación de un dispositivo que permitiera simular reacciones táctiles de fuerza fue lograda en 1968 por un grupo de investigadores de la Universidad de Carolina del Norte. Para esto se sirvieron de un dispositivo robótico similar a los que se usan para la manipulación remota de materiales radiactivos.

• 1971.- Redifon Ldt en el Reino Unido comienza a fabricar simuladores de vuelo con displays gráficos. Henry Gouraud presenta su tesis de doctorado "Despliegue por computadora de Superficies Curvas."

• 1972.- General Electric, bajo comisión de la Armada norteamericana, desarrolla el primer simulador computarizado de vuelo.

• 1977.- Dan Sandin y Richard Sayre inventan un guante sensitivo a la flexión.

• 1981.- En el Reino Unido el desarrollo de ambientes sintéticos de inmersión fue protagonizado por Jonathan Waldern fundador de W Industries y de Virtuality, quien lanzó al mercado el primer producto basado en el uso del casco.

• 1982.- Ocurre uno de los acontecimientos históricos en el desarrollo de los simuladores de vuelo, cuando Thomas Furness presentó el simulador más avanzado que existe, contenido en su totalidad en un casco parecido al del personaje Darth Vader y creado para la Fuerza Aérea de Estados Unidos .

Thomas Zimmerman patentó un guante para introducir datos, basado en sensores ópticos, de modo que la refracción interna puede ser correlacionada con la flexión y extensión de un dedo.

• 1984.- El primer sistema que se proponía atravesar el umbral de las dos dimensiones y recrear una verdadera "telepresencia en un espacio de datos," fue concebido en los laboratorios Ames de la NASA en California.

• 1985.- Los investigadores del laboratorio Ames de la NASA construyen el primer sistema práctico de visores estereoscopios. Mike Mc Greevy y Jim Humphries construyen un HMD (Head Mounted Display) con un LCD (Liquid Crystal Display) monocromo del tamaño de una televisión de bolsillo.

• 1986.- En el centro de investigaciones de Schlumberger, en Palo Alto, California, Michael Deering (científico en computación) y Howard Davidson (físico) trabajaron en estrecha relación con Sun Microsystems para desarrollar el primer visor de color basado en una estación de trabajo.

• 1987.- La NASA utilizando algunos productos comerciales, perfecciona la primera realidad sintetizada por computadora mediante la combinación de imágenes estéreo, sonido 3-D, guantes, etc.

• 1991.- En julio apareció Dactyl Nightmare el primer juego en el que varios usuarios pueden interactuar en un mismo espacio.

• 1992.- Sun Microsystems hace la primera demostración de su Portal Visual, el ambiente RV de mayor resolución hasta el momento.

Al Gore, vicepresidente de Estados Unidos en esta época y promotor de la Realidad Virtual, dictó seminarios sobre la importancia de esta tecnología para la competitividad norteamericana.

• 1994.- La Sociedad de Realidad Virtual fue fundada.

Durante la Primera Conferencia Internacional de World Wide Web, en Génova, realizada en mayo de ese año; se organiza una reunión para discutir las interfaces de la Realidad Virtual al Web. Resultando que muchos participantes estaban trabajando en proyectos para conseguir herramientas de visualización tridimensional que ínter operaran con el Web; se llegó a un consenso sobre los requerimientos para conseguir un lenguaje modelado de Realidad Virtual, y así discutieron las tecnologías existentes antes de pronunciarse por la tecnología Open Inventor, de Silicon Graphics.

Nace VRML (Virtual Reality Modelling Language)

A partir de ese momento se propuso un ciberespacio consistente y definido por el uso de VRML para mejorar la navegación en la red sin embargo, la discusión y la actividad que siguieron resultaron en la especificación de un lenguaje común para definir las escenas tridimensionales más que en la generación de una interfaz.

2.3.1.3. Concepto.

Existe mucha controversia sobre un concepto definitivo de lo que es la Realidad Virtual, hay muchas definiciones sobre ella, para poder orientarse un poco en lo que es el tema es preciso definir qué es la Realidad como tal y luego el significado de Virtual, y combinarlos obteniendo una posible definición de Realidad Virtual o RV:

Virtual = Un efecto de ser sin ser en realidad o que existe o resulta en esencia o efecto pero no como forma, nombre o hecho real.

Realidad = La propiedad de ser real y "real", es la propiedad de existir de forma correcta.

Por lo tanto se podría decir que la Realidad Virtual es la manipulación de los sentidos humanos, por medio de entornos tridimensionales generados por computadora que actúan de manera rápida e intuitiva que hacen desaparecer el PC de la mente del usuario dando como real el entorno virtual.

Existen muchos conceptos vertidos por científicos expertos en RV, se nombra algunos de estos a continuación:

"Realidad Virtual: Es una simulación que satisface los requerimientos que le pide nuestro cerebro a nuestros sentidos para sentir una situación verdadera." Steve Bryson de la NASA.

"Realidad Virtual: Un sistema computacional usado para crear un mundo artificial en el cual el usuario tiene la impresión de estar en ese mundo y con la habilidad de navegar a través de él y manipular los objetos que lo forman." C. Manetta and R. Blade in "Glossary of Virtual Reality Terminology" in the International Journal of Virtual Reality, Vol.1 Nr.2 1995.

2.3.1.4. Descripción.

Para poder explicar de mejor manera lo que se puede sentir con la Realidad Virtual podemos compararla con la sensación de estar soñando, del mismo modo que en sueños convive lo que tiene sentido y lo que no lo tiene, en la Realidad Virtual se mezclan libremente lo lógico y lo ilógico.

Lo dicho anteriormente trae a escena algo que tiene que ser aclarado para no tergiversar los conceptos e ideas de RV y es: "LA REALIDAD VIRTUAL Y REALIDAD ARTIFICIAL NO SON SINÓNIMOS." La Realidad Virtual tiende a destacar la posibilidad de simular el mundo real con una finalidad cognoscitiva, basándose en modelos que se demuestren instrumentalmente adecuados para ese objetivo. La realidad artificial simula entornos y escenas inexistentes, o imposibles porque incumplen leyes físicas a fin de explorar las potencialidades expresivas del medio más allá de sus capacidades reproductivas en relación con lo real.

La tecnología de la RV asume todo lo que puede ofrecer la realidad artificial e incluso todo lo que permite a la imaginación, mezclando lo artificial con lo real.

2.3.1.5. Características de un Sistema de Realidad Virtual.

La Realidad Virtual tiene las siguientes características:

Un sistema de Realidad Virtual tiene que cumplir con algunas características, no es imprescindible que las cumpla todas pero si una parte de ellas, dichas características nos ayudan a clasificar posteriormente a los sistemas de RV entre las características básicas se pueden anotar las siguientes:

- Interacción.- Rasgos que permiten al usuario manipular el curso de la acción dentro de una aplicación de Realidad Virtual, permitiendo que el sistema responda a los estímulos de la persona que lo utiliza; creando interdependencia entre ellos.

  Existen dos aspectos únicos de interacción en un mundo virtual. El primero de ellos es la navegación, que es la habilidad del usuario para moverse independientemente alrededor del mundo. Las restricciones para este aspecto las coloca el inventor del software, que permite varios grados de libertad, si se puede volar o no, caminar, nadar, etc.

El otro punto importante de la navegación es el posicionamiento del punto de vista del usuario. El usuario se puede mirar a sí mismo (a través de los ojos de alguien más), o puede moverse a través de cualquier aplicación observando desde varios puntos de vista.

El otro aspecto de la interacción es la dinámica del ambiente, que no es más que las reglas de cómo los componentes del mundo virtual interactúan con el usuario para intercambiar energía o información.

- Inmersión.- Esta palabra significa bloquear toda distracción y enfocarse selectivamente sólo en la información u operación sobre la cual se trabaja.

  Posee dos atributos importantes, el primero de ellos es su habilidad para enfocar la atención del usuario, y el segundo es que convierte una base de datos en experiencias, estimulando de esta manera el sistema natural de aprendizaje humano (las experiencias personales).

  Puede ser también descrita como propiedad mediante la cual el usuario tiene la sensación de encontrarse dentro de un mundo tridimensional.

- Tridimensionalidad.- Esta es una característica básica para cualquier sistema llamado de Realidad Virtual, tiene que ver directamente con la manipulación de los sentidos del usuario, principalmente la visión, para dar forma al espacio virtual; los componentes del mundo virtual se muestran al usuario en las tres dimensiones del mundo real, en el sentido del espacio que ocupan, y los sonidos tienen efectos estereofónicos.

- Existencia de un Punto de Observación o Referencia.- Permite determinar ubicación y posición de observación del usuario dentro del mundo virtual en el que se encuentra.

- Navegación.- Propiedad que permite al usuario cambiar su posición de observación, esto lo puede hacer mediante un dispositivo de entrada como puede ser un joystick, un mouse o el teclado.

2.3.2. TÉCNICAS DE REALIDAD VIRTUAL.

Existen una variedad de diferentes concepciones y sistemas llamados Realidad Virtual. Un sistema de este tipo debe tener las característica básicas definidas en el punto anterior y como ya fue mencionado no es suficiente si se tienen una o dos de ellas, deben estar como mínimo las tres presentes.

Al tomar en cuenta las características básicas, se podría hacer una clasificación de técnicas de Realidad Virtual con los que se cuentan en el mundo virtual:

- Cabina de Simulación.- El ejemplo más común de este tipo de simulador es la cabina para el entrenamiento de aviadores.

  Generalmente la cabina recrea el interior del dispositivo o máquina que se desea simular. (carro, avión, tanque etc.), las ventanas de la misma se reemplazan por pantallas de computadoras de alta resolución, además existen bocinas estereofónicas que brindan el sonido ambiental y puede estar colocada fija o sobre ejes móviles.

  El programa está diseñado para responder en tiempo real a los estímulos que el usuario le envía por medio de los controles dentro de las cabinas.

  Se puede decir que este tipo es una Realidad Virtual semi inmersiva pues las pantallas son generadas en monitores o ventanas de los simuladores

- Realidad Proyectada.- En este tipo de Realidad Virtual una imagen en movimiento del usuario es proyectada junto con otras imágenes en una extensa pantalla donde el usuario puede verse a sí mismo como si estuviese en la escena.

  En esencia los usuarios se miran ellos mismos como proyectados hacia el mundo virtual.

  Un ejemplo actual de este tipo de Realidad Virtual son los escenarios virtuales que se utilizan en ciertos programas de televisión.

- Realidad Aumentada.- Este tipo de Realidad Virtual se logra cuando una persona escoge fiarse del mundo real como línea de referencia, pero utiliza visores de cristal transparentes u otros medios inmersivos para aumentar la realidad, superponiendo esquemas, diagramas, textos, referencias, etc. Como ejemplo la Boeing está explorando la posibilidad de utilizar este sistema en la ingeniería de los aeroplanos, de tal suerte que sus técnicos e ingenieros no tengan que irse a ver un manual para resolver un problema.

- Telepresencia.- Término creado por Marvin Minsky que significa presencia remota, es un medio que proporciona a la persona la sensación de estar físicamente en otro lugar por medio de una escena creada por computadora.

  Es una experiencia sicológica que ocurre cuando la tecnología de simulación funciona lo suficientemente bien como para convencer al usuario de que está en un mundo virtual.

- Realidad Virtual de Escritorio (Desktop): Es una subinstalación del tradicional sistema de Realidad Virtual. En lugar de utilizar cascos para mostrar la información visual utiliza un monitor grande de computadora o un sistema de proyección.

- Ventanas Acopladas Visualmente.- Esta es la clase de sistema de inmersión que se asocia más a menudo con Realidad Virtual.

  Este sistema se basa en colocar las muestras directamente en frente del usuario, y conectando los movimientos de la cabeza con la imagen mostrada. Para lograr mayor acople la inmersión se logra con un casco estereofónico (HMD Head Mounted Display), que posee sensores de posición y orientación que informan a la máquina la posición del usuario en todo momento, además de indicarle hacia donde está mirando. Un ejemplo de este tipo de Realidad Virtual son los juegos que hicieron popular esta técnica de computación los últimos años.

2.3.3. MUNDOS VIRTUALES.

Una síntesis de los tipo de Realidad Virtual anteriormente mencionados da una clasificación un poco más adentrada en los tipos de mundos que se están recreando, basándonos en una de las características de la Realidad Virtual como es la interacción del usuario con el mundo virtual y los detalles modelados, se puede decir que existen tres tipos básicos de mundos virtuales que pueden existir por separados como también mezclados entre sí:

- Mundo Muerto.- Es aquel en el que no hay objetos en movimiento ni partes interactivas, por lo cual sólo se permite su exploración.

  Suele ser el que vemos en las animaciones tradicionales, en las cuales las imágenes están precalculadas y producen una experiencia pasiva.

- Mundo Real.- Es aquel en el cual los elementos tienen sus atributos reales, de tal manera que si miramos un reloj, marca la hora. Si pulsamos las teclas de una calculadora, se visualizan las operaciones que esta realiza y así sucesivamente.

- Mundo Fantástico.- Es el que nos permite realizar tareas irreales, como volar o atravesar paredes.

  Es el típico entorno que visualizamos en los videojuegos, pero también proporcionan situaciones interesantes para aplicaciones serias, como puede ser observar un edificio volando a su alrededor o introducirnos dentro de un volcán.

En conclusión nuestro sistema cumple con las siguientes características: inmersión, tridimensionalidad, punto de referencia, navegación.

Utiliza las técnicas: desktop e inmersiva.

Los tipos de mundo: muerto y fantástico.

2.3.4. TENDENCIAS.

La tecnología de la RV permite conocer cosas que de otro modo no estarían a nuestro alcance: estructura del átomo, hélice del ADN, futura estación espacial Alpha o la programación de un recorrido en Marte. También permite a los cirujanos operar a pacientes inexistentes, sin riesgo de matarlos, a químicos intentar la creación de nuevas moléculas, a diseñadores de automóviles probar los vehículos en carreteras ficticias. Aquí es posible mezclar los conocimientos adquiridos con la fantasía y poner a prueba hipótesis sin el riesgo de destruir objetos o entornos delicados.

También se puede reproducir en la memoria de una computadora una pirámide egipcia o cavernas prehistóricas cuya visita constituye un riesgo para su conservación, cualquier persona podría realizar la visita caminando por dicho monumento.

Dentro del campo de la educación y de la ciencia en general será una herramienta de gran valía y tal vez indispensable en los años por venir. Por ejemplo como será el aula este próximo siglo: el aula podrá ser nuestra propia sala o una propia terminal dentro de un campus universitario. Complementada con un par de lentes o cascos con audífonos integrados, así como un par de guantes especiales y traje ajustado de cuerpo completo.

Todo lleva consigo un riesgo, por ejemplo.: fomento a la violencia, escape de la realidad, pornografía, proliferación de grupos extremistas, juegos enajenables. Pero el ser humano deberá adaptarse y basarse en su capacidad para minimizar o desaparecer esas influencias negativas para bien de todos.

Algunos científicos como Ray Kurzwelly y Hill Joy jefes de Sun Mycrosistems pronostican que en 20 años la tecnología Matrix será verdad.

Por su parte, Glem Yeffeth hace una compilación de información llamada "taking the red pill =science, phylosophy and religión in the matriz" habla sobre el cruce de entes biológicos y no biológicos en Biorobótica. Antes se hablaba de la tecnología Nuclear - Química - Biológica y ahora los estudios se enfocan a la tecnología Genética - Nanotecnología – Robótica.

2.3.5. DISPOSITIVOS.

2.3.5.1. Introducción.

Para empezar con los requerimientos técnicos necesarios para la Realidad Virtual se ha dividido en 4 puntos con los cuales trabaja un sistema virtual:

- Hardware.- Debe tener una capacidad de renderizar gráficos 3D en tiempo real con una alta calidad.
- Dispositivos de Entrada.- Dispositivos que permitan al usuario interactuar con el mundo.
- Dispositivos de Salida.- Dispositivos capaces de remplazar la información visual del usuario del mundo físico con los generados por el computador.
- Software.- Debe tener capacidad de procesar la entrada/salida de información, renderizar, acceder a la base de datos del mundo virtual en la cual se define el ambiente virtual en tiempo real.

2.3.5.2. Tipos de Dispositivos.

2.3.5.2.1 Sonido.

Importante para la percepción especial de una persona y es más efectivo cuando las ayudas visuales son mínimas.

Equipos empleados para audición:

Los audífonos son el equipo básico empleado para escuchar los sonidos propios de un ambiente virtual. A continuación se presentan algunas variantes de estos equipos:

Audífonos convencionales: son los audífonos de uso más corriente, a través de estos se escucha el sonido simulado de los objetos sin identificar auditivamente el punto de ubicación de los mismos.

Convolvotrón: estos audífonos además de simular el sonido propio de los objetos, simulan la ubicación de los mismos dentro del ambiente virtual.

2.3.5.2.2. Rastreadores.

Un dispositivo es sujetado al objeto o al usuario para que los movimientos de la cabeza o manos puedan ser detectados, esto se lleva a cabo por medios electromagnéticos u ópticos.

Los movimientos son expresados en coordenadas de posición y orientación que son descifradas por la computadora. Es entonces, cuando las imágenes correspondientes a ese punto de vista son presentadas.

2.3.5.2.3 Guante.

El dispositivo más omnipresente para el control y entrada instrumentado con fibras ópticas flexibles que recorren cada articulación de la mano conocidos en inglés como Data Glove.

Los guantes son dispositivos separados que pueden ser acoplados a cualquier sistema de computadora y a las aplicaciones más recientes, desde videojuegos hasta proyectos de investigación.

2.3.5.2.4 Dispositivos de visualización estereoscópica.

Los equipos de visión estereoscópica se dividen en dos tipos:

- HMD.- Son denominados unidades de presentación montadas sobre la cabeza. En estos los dispositivos de visión quedan suspendidos enfrente de los ojos del usuario.

  Los dispositivos visuales estereoscópicos de gran ángulo y en color hacen ligeramente diferentes las imágenes que cada ojo recibe, de tal forma que las imágenes producidas (es decir, el entorno virtual) parecen tener profundidad. Algunas unidades están equipadas con auriculares para un acompañamiento auditivo. Eventualmente, los medios para alimentar con estímulos los otros sentidos estarán construidos también dentro del casco.

  Hay buenos dispositivos para la cabeza comercialmente asequibles, pero algunos grupos de investigación prefieren desarrollarlos. Los investigadores japoneses están intentando conseguir una tecnología capaz de reemplazar o minimizar estos dispositivos.

  El objetivo de algunos HMDs es mantener lejos de la cabeza las presentaciones generadas por computadora, transmitiendo las imágenes de alta resolución por medio de fibra óptica para aligerar las lentes suspendidas cerca de los ojos.

  La desventaja de este dispositivo radica en su alto costo, se encuentran cascos desde los $2000 en adelante.

- I-Glasses (Gafas Estereoscópicas).- Las gafas son utilizadas en aplicaciones 3D para computadoras y películas en televisión. Las imágenes aparecen en 3D porque cada ojo recibe un dibujo único (cuando se mira sin las gafas, el dibujo tiembla o se observan dos imágenes montadas).

Las grandes pantallas son apropiadas para entornos virtuales de gran distancia o gran área, como un recinto arquitectónico o rutas para bicicletas. Para áreas pequeñas son suficientes pequeñas pantallas o terminales (por ejemplo, en la construcción de moléculas).

Este tipo de dispositivo funciona mandando el 50% del rendering al ojo de la izquierda, y el otro 50% al ojo de la derecha, alternando el despliegue de imágenes entre izquierda y derecha cada vez que se refresca la pantalla. Estas gafas están compuestas por dos LCDs, una al frente de cada ojo, cada LCD tiene la capacidad de bloquear la imagen cuando la parte de la izquierda recibe la imagen, el de la derecha se bloquea y viceversa, generando así la profundidad, esta secuencia es tan rápida que no se puede percibir.

Esta tecnología es más accesible al usuario normal debido a su relativo bajo costo en comparación con los cascos.

2.3.5.2.6. Traje.

El traje es básicamente como un guante de datos específico para todo el cuerpo. Está instrumentado con el mismo tipo de cable de fibra óptica que recorre un guante de datos.

Al moverse, curvarse o hacer señas el usuario, el sistema toma coordenadas espaciales para cada parte del traje, rastreando dinámicamente una extensa serie de acciones. Actualmente, 20 o más sensores recogen la mayoría de las articulaciones del cuerpo.

2.3.5.2.7. Varas.

Como una varita mágica con un sensor en su extremo y un sensor de seis grados de libertad en su base, este dispositivo es probablemente el dispositivo de control más sencillo utilizado en entornos virtuales. Algunas varas tienen interruptores o botones de selección adheridos. Al mover una vara, un participante de un mundo virtual puede utilizarla para pintar, interaccionarse con otros objetos o emitir un conjunto limitado de órdenes. Algunas versiones emiten un rayo láser desde su parte frontal. Cuando se apunta un objeto en la escena virtual, lo selecciona o recupera.

2.4. GENERACIÓN DE ESPACIOS 3D EN PLATAFORMAS WINDOWS.

2.4.1. LIBRERÍAS GRÁFICAS.

2.4.1.1. Concepto.

Librería Gráfica es un conjunto de instrucciones diseñado para generar imágenes en base a modelos matemáticos y patrones de iluminación, texturas, sombras, cámaras entre otras, y permiten acceder a características de hardware especializado sin tener que escribir código específico de él.

Lo que se busca con estas librerías es independencia de hardware tanto de entrada como de salida, además de brindar independencia de la aplicación generada, y la opción de escoger el lenguaje de desarrollo.

2.4.1.2. Tipos de Librerías Gráficas.

En 1982 nació en la Universidad de Standford el concepto de "máquina gráfica" (graphics machine) y este fué utilizado por Silicon Graphics Corporation en su propia estación Silicon IRIS para crear un renderizador. Así nació la librería **IRIS GL**, en respuesta a esta librería otras empresas empezaron a desarrollar sus estándares de librerías gráficas como Hewlett Packard y Tektronix con sus **Starbase** y **IGL-Plot 10** respectivamente, sin llegar a ser trascendentes en el mundo de graficación por computadora.

A raíz de esto, allá por 1992 muchas empresas del hardware y software se pusieron de acuerdo para desarrollar conjuntamente una librería gráfica libre cuyo nombre final fué **OpenGL** (Open Graphics Library). Entre estas empresas destacaban Silicon Graphics Inc., Microsoft, IBM Corporation, Sun Microsystems, Digital Equipment Corporation, Hewlett-Packard Corporation, Intel e Intergraph Corporation, convirtiéndose en una de las más difundidas por su característica de ser abierta (La característica de ser "Abierta" significa que un programa escrito para una plataforma puede ser fácilmente convertible prácticamente para cualquier tipo de plataforma, obteniendo los mismo resultados).

Al cabo de unos años Microsoft comienza sus estudios por separado creando así su librería **DirectX 1.0,** la cual por su extremada dificultad no fué acogida, asociándose así con Sun Microsystems para crear el proyecto **Fahrenheit,** que era un híbrido entre el OpenGL y DirectX en sus primeras versiones, el cual por diferencias políticas entre estas dos empresas nunca vio la luz.

Microsoft continúa su investigación por separado de su librería gráfica sacando DirectX 3.0 la cual tuvo mejor acogida que sus antecesoras, llegando actualmente a la versión 9.0 y convirtiéndose en la librería más usada en aplicaciones multimedia y entretenimiento en plataformas Windows.

Existieron otras librerías sin mayor relevancia como son:

- OpenInventor: es un conjunto de bibliotecas de rutinas de desarrollo para aplicaciones de visualización de objetos 3D Inventor. Inventor es una herramienta orientada a objetos para desarrolladores que simplifica dramáticamente la programación de gráficos 3D.

- GKS (Graphical Kernel System) Sistema de núcleo gráfico: Lenguaje de gráficos independiente de dispositivos para imágenes gráficas bidimensionales, tridimensionales. El GKS permite que las aplicaciones gráficas sean desarrolladas en un sistema y fácilmente llevadas a otro sistema con mínimo o nada de cambio.

- Glide: es el estándar gráfico que promovió la empresa 3dfx cuando lanzó su gama de tarjetas gráficas Voodoo. Cuando apareció Glide, DirectX no era un sistema muy extendido ni muy fiable, por lo que muchos juegos funcionaban con DirectX de forma limitada y con todas las opciones si empleaban Glide. Hoy en día está en desuso.

2.4.1.3. Librerías Gráficas más Difundidas.

2.4.1.3.1. OpenGL.

Desde el punto de vista del programador OpenGL es una API (Aplication Programing Interfase) para interactuar con dispositivos gráficos y aceleradoras 3D. Contiene cerca de 150 comandos que nos ayudan a definir objetos, aplicar transformaciones a esos objetos, cambiar sus propiedades color, textura, luz, posición de la cámara, entre otros. También hay que tener claro que OpenGL es una librería gráfica, no posee funciones para el control de Audio, Red o Control de Entrada.

Principales características.

- Primitivas geométricas y raster: Nos permite utilizar todas las primitivas geométricas básicas: puntos, líneas, polígonos. Y el raster usar un bitmap.
- B-splines: Son usadas para dibujar líneas curvas.
- Transformaciones de vista y modelo: Gracias a estas transformaciones podemos fácilmente mover, rotar y escalar los objetos dentro de la escena y a su vez mover la cámara.
- Trabajar con el color: OpenGL nos permite operar con colores en modo RGBA (red-green-blue-alpha) o usando Modo Indexado, donde los colores se seleccionan desde una paleta.
- Eliminación de líneas y superficies ocultas: Por medio del algoritmo Z-Buffer.
- Doble buffer: OpenGL nos permite utilizar un buffer o dos. El buffer doble es usado para eliminar el parpadeo de las animaciones. Cuando se está mostrando un frame en el buffer primario el siguiente se dibuja en el doble buffer y cuando está terminado se copia al buffer primario, así se eliminan esos parpadeos.
- Mapeado de textura: Algo vital en cualquier API gráfica 3D
- Antialiasing: Nos permite suavizar los bordes de polígonos y líneas. Este suavizado se realiza cambiando la intensidad de los píxeles adyacentes a la línea que procesamos consiguiendo un efecto de "difuminación" con la consiguiente eliminación de esos zig-zag.
- Luces: Nos permite establecer la fuente de la luz, su posición, su intensidad, color.
- Efectos atmosféricos: Por ejemplo niebla o humo.
- Transparencia.

Librerías adicionales de OpenGL.

La librería principal de OpenGL nos suministra todas las funciones necesarias para mostrar un entorno 3D aunque hay algunas operaciones que son algo tediosas de realizar utilizando sólo esta librería. Estas son por ejemplo calcular la matriz de proyección. Para esto se han desarrollado también librerías auxiliares:

- GLU: Esta librería acompaña a la OpenGL principal. Incluye funciones más complejas que GL por ejemplo definir un cilindro o un disco con un solo comando, también contiene funciones para trabajar con splines y operaciones con matrices.
- GLUT: Esta librería es independiente de la librería OpenGL de cada plataforma, esta librería no incluye funciones adicionales para OpenGL, pero nos permite utilizar funciones para el tratamiento de ventanas, teclado y ratón. Para trabajar con OpenGL en cualquier plataforma.
- GLAUX: Esta librería, muy parecida a GLUT, es la que Microsoft ha desarrollado para Windows. Mantiene prácticamente la misma estructura que GLUT con el defecto de que sólo sirve para Windows, mientras que GLUT sirve para cualquier plataforma.
- GLX: Esta es la librería utilizada para trabajar en un sistema de X-Windows (Linux), permite no sólo renderizar en la máquina local, sino también a través de una red.

2.4.1.3.2. DirectX.

DirectX es una API creada por Microsoft, para permitir a los desarrolladores de juegos programar eficientemente bajo Windows, lo cual era improbable antes de la llegada de él, ya que tecnologías como GDI (Graphics Device Interfase) no permitían acceder al hardware tan directamente como lo hace DirectX. Existen dos versiones de DirectX, una para usuarios, que permite correr la mayoría de los juegos, aplicaciones multimedia o gráficas y otra para desarrolladores llamada DirectX SDK.

Principales características.

- Manipulación de múltiples buffers de vídeo.
- Acceso directo a la memoria de vídeo.
- Mezclado de canales de audio.
- Aceleración hardware.
- Acceso directo al dispositivo de sonido.
- Captura de Audio.
- Simplifica el acceso de las aplicaciones a los servicios de comunicación.
- Otorga independencia para la creación de servidores de juegos.
- Comunicaciones punto a punto y Cliente/Servidor.
- Interfaz gráfica para hardware 3D.
- Permitir gráficos tridimensionales interactivos en aplicaciones de Windows.
- API 3D de bajo nivel.
- Independencia del dispositivo.
- Sistemas de coordenadas 3-D Left-handed (mano izquierda) y podemos simular el right-handed (mano derecha).
- Transformaciones 3-D traslación, rotación, escalado.
- Representación de polígonos normales de cara y vértice, modos de sombreado, interpolaciones de triángulos.
- Reglas de rasterización de triángulos.

Componentes de DirectX.

- DirectGraphics: Proporciona librerías que permite optimizar la renderización de objetos 2D y 3D tomando las ventajas del hardware existente, o mediante software si no se encontrara el hardware adecuado, compuesto por las librerías Direct3D (manejo de funciones 3D) y DirectDraw (manejo de funciones 2D) anteriormente estaban separadas las dos pero desde la versión 8 forman un solo componente.

- DirectAudio: Como su nombre lo índica, la librería de DirectAudio maneja todo lo relacionado con el sonido. Proporcionado tecnologías de mixing, sonido estéreo y 3D, aprovechando al máximo las capacidades del hardware, compuesto por DirectMusic (para crear y reproducir gran variedad de ficheros de audio) y DirectSound (para reproducir y capturar sonidos complejos como sonidos 3D) que de igual manera están unidas desde la versión 8.

- DirectPlay: Se encarga de las características multijugador de los juegos.

- DirectInput: Proporciona una interfaz para el manejo de entrada y salida. Como el teclado, el mouse, joystick, etc.

2.4.2. FORMATOS DE INTERCAMBIO DE DIBUJO.

2.4.2.1 Introducción.

Con la evolución y difusión de los gráficos por computadora, muchas empresas desarrolladoras de software para este campo sacaron al mercado sus aplicaciones con sus respectivos formatos, algunos pasaron desadvertidos mientras otros se convirtieron en estándares de archivos utilizados hasta la actualidad, el tener diferentes formatos de gráficos dificultaba el intercambiar archivos entre programas, pues no todos las aplicaciones disponían de las suficientes herramientas que necesitaban los graficadores, teniendo así que usar varios programas para un solo proyecto.

Por lo que las compañías optaron por incluir en sus aplicaciones el soporte de formatos estándar, que brindaron una compatibilidad entre los programas.

2.4.2.2 Tipos de Ficheros Gráficos más Difundidos.

Tenemos dos tipos de ficheros gráficos uno en 3D y los otros de 2D, tales como:

Ficheros 2D.

- BMP: Mapa de bits que contiene gráficos digitales.
- JPG: Gráficos vectoriales comprimidos.
- GIF: Gráficos vectoriales comprimidos que pueden o no tener animación.
- DWG: Archivos de Autocad.
- DXF: Draw Interchange Files (Archivo de intercambio de dibujo).
- Entre otros.

Ficheros 3D.

- 3DS: Archivo perteneciente a la empresa Discreet para su programa 3D Studio, siendo uno de los primeros modeladores 3D este fichero es uno de los más difundidos.
- LWO: Archivo perteneciente al programa de modelado 3D Lightwave de gran difusión en la actualidad.
- OBJ: Para el programa Wavefront que pertenece a la compañía Acrobat.
- WRL: Formato que es usado en VRML.
- DXF: Draw Interchange Files (Archivo de intercambio de dibujo).
- X: Formato de archivo 3D nativo de DirectX.

2.4.2.3. 3DS.

3DS es posiblemente el formato más extenso de intercambio de datos de los programas de modelado 3D incluido algunos sistemas CAD para sistemas informáticos.

El formato 3DS posibilita el intercambio de escenas entre programas de diseño y modelado de 3D. Los ficheros 3DS se encuentran en formato binario.

Un archivo 3DS puede contener un objeto o una escena completa en 3D, la cantidad de información para un archivo 3D es muy compleja para describir la geometría de los modelos 3D.

2.4.2.4. Ficheros X.

Los ficheros X, son los archivos de 3D de DirectX, su formato puede ser tanto binario como ASCII.

Puede contener información de la malla, textura, movimiento, materiales; no todos los modeladores 3D permiten exportar directamente a este formato, por lo cual Microsoft provee un conversor (CONV3DS.EXE) incluido en las librerías SDK, para transformar los ficheros 3DS a X.

2.4.3. TÉCNICAS DE VISUALIZACIÓN DE GRÁFICOS 3D.

Desde la aparición de los gráficos por ordenador hasta la actualidad, se han producido una gran cantidad de innovaciones, debidas al desarrollo de las técnicas de los programas y, en mayor medida, a la evolución de los ordenadores. Haciendo un breve repaso, la cronografía puede ser más o menos la siguiente, que además corresponde a los distintos tipos de visualización:

- Modo Wireframe o Modelos de Alambre: como su nombre indica, sólo se representan líneas (tanto rectas como curvas), ver Fig 2.5.

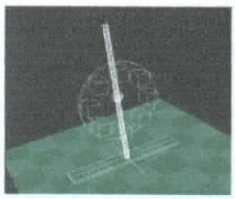


Fig. 2.5.

- Modo Filling o de Rellenado Continuo: Se representan objetos con caras de colores totalmente uniformes, ver Fig 2.6.

Fig. 2.6.

- Modo Rendering: Ahora, no sólo se representan las caras con colores, si no que, además estos contienen degradados, sombras, etc. dentro de las técnicas de rendering, podemos destacar dos grupos:

    - Volume Rendering: Cuando los datos se obtienen de hardware de adquisición de datos en forma de puntos 3D. Muy utilizado en medicina para RMN (Resonancia Magnética Nuclear) y TAC (Tomografía Axial Computerizada).

    - Object Rendering: cuando los datos se obtienen de modelos matemáticos 3D. Este es el tipo de rendering más utilizado, ver Fig 2.7.

Fig. 2.7.

2.4.3.1. Rendering e Iluminación.

2.4.3.1.1. Rendering.

Rendering o Rénder se llama al proceso final de generar la imagen 2D o animación a partir de la escena creada. Esto puede ser comparado con tomar una foto o filmar la escena en la vida real, después que se terminó de armar. Generalmente se buscan imágenes de calidad foto realista.

El rénder puede simular efectos cinematográficos, y de origen natural, como la interacción de la luz con la atmósfera o el humo. Ejemplos de estas técnicas incluyen simular lluvia, humo o fuego, así como también niebla, polvo entre otros efectos atmosféricos, y el efecto de la luz al atravesar superficies refractantes.

El proceso de rénder requiere gran procesamiento, ya que requiere simular gran cantidad de procesos físicos complejos. La capacidad de procesamiento se ha incrementado rápidamente a través de los años, permitiendo un grado superior de realismo en los rénders.

Dependiendo del algoritmo de cálculo en el proceso de rendering, se tienen los siguientes tipos:

- Método Scanline.- Tienen en cuenta sólo la iluminación de los objetos, y no las propiedades físicas de éstos. Son métodos muy rápidos, pero los resultados que se obtienen son los menos realistas.
- Método Radiosity.- Calculan las ecuaciones de radiosidad para cada objeto, en función de la energía que reciben, emiten y propagan. Son métodos independientes del punto de vista del observador, muy lentos, pero que producen los mejores efectos de sombreado/iluminación. Tienen en cuenta las propiedades físicas relacionadas con la reflexión, pero no la refracción.

Fig. 2.8.

Fig. 2.9.

Fig. 2.10.

Fig. 2.11.

Fig. 2.12.

Fig. 2.13.

Fig. 2.14.

Fig. 2.15.

La Fig. 2.8 está muy poco iluminada y apenas se puede percibir el reflejo de la fuente de luz.

La Fig. 2.9 aún podría beneficiarse de una fuente de luz más potente, ahora se percibe una esfera reflejando la fuente de luz.

La Fig. 2.10 muestra la mayoría de los objetos en la escena, pero no da una fuerte impresión. Las texturas comienzan a emerger.

La Fig. 2.11 muestra casi todos los objetos en la escena, nótese que ya no se distingue el reflejo de la fuente de luz y que la textura en un tercio de la esfera se ha perdido.

La Fig. 2.12 saca a luz todos los objetos en la escena de una forma clara. Los colores de la esfera están ahora sobresaturados por causa de la mayor potencia de la luz (comparada con la de la Fig. 2.10)

La Fig. 2.13 muestra que los objetos que rodean a la esfera central comienzan a presentar una sobresaturación de luz y sus colores comienzan a saturarse.

La Fig. 2.14 simplemente intensifica los efectos mencionados en la descripción de la Fig. 2.13.

La Fig. 2.15 muestra unos colores sobresaturados y objetos sobre expuestos con una considerable proporción de texturas y detalle perdido debido a la sobre exposición.

- Dirección de la Luz.- Imaginemos una escena donde existe luz de intensidad y color equivalente, incidiendo sobre un rostro humano en todas direcciones por igual y sobre un fondo negro. Veríamos solamente una silueta bidimensional del rostro recortada sobre el fondo. Porque los rayos de igual color e intensidad pintarán todos los lados del rostro con el mismo color y la misma intensidad. Si una sombra tendiera a formarse, sería instantáneamente borrada por los rayos que inciden sobre esa región.

  La razón por la cual somos capaces de reconocer la forma de un objeto, es que los rayos de luz de distintas intensidades que golpean al objeto desde distintas direcciones, pintan al objeto con brillos y sombras.

Fig. 2.16.

Fig. 2.17.

La Fig. 2.16 muestra al objeto iluminado desde una única fuente de luz ubicada a la izquierda de la cámara. Es posible apreciar claramente los pliegues, las protuberancias y depresiones de la superficie. También es posible ver claramente la base del objeto en contacto con el piso y arrojando una sombra.

La Fig. 2.17 representa al mismo objeto, pero la fuente de luz se encuentra directamente tras la cámara. Los detalles frontales se han perdido casi en su totalidad, debido a que las sombras arrojadas en Fig. 2.16 se han desvanecido por la acción de la luz directa. Sin embargo, aún es apreciable algún detalle sobre los bordes.

- Color de la Luz.- La razón por la que somos capaces de reconocer la forma de un objeto es que rayos de luz de distintas intensidades, provenientes de distintas direcciones, impactan en el modelo, pintándolo con brillos y sombras.

  El color de una luz directa depende de su fuente de irradiación. La luz blanca está compuesta por todos los posibles colores existentes. Un rayo de luz blanca cambiará de color si encuentra un obstáculo que no sea ni blanco ni negro. Si impacta a un objeto blanco, el mismo rayo es reflejado. Si el objeto es de color negro, el objeto absorbe toda la luz, sin importar de qué color era originalmente y nada es reflejado.

  De manera que básicamente al observar un objeto totalmente negro, se ve de ese color porque no hay luz que ingrese al ojo proveniente de esa dirección.

  En la Fig. 2.18, es posible ver un rayo blanco de luz directa, que es reflejado por un piso azul. El piso absorbe todos los colores del rayo que lo impacta, a excepción del azul que es reflejado. Nótese que la luz es reflejada con un ángulo idéntico al ángulo de impacto con relación al piso.

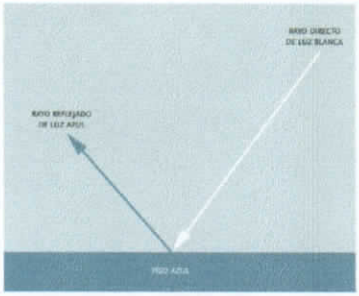


Fig. 2.18.

- Tamaño de la fuente de Luz.- El tamaño de la fuente tiene un efecto preponderante en la sensibilidad general de la escena. Una fuente de luz de reducido tamaño arroja sombras muy definidas y apreciables, incorporando un elemento de tensión a la imagen. Un ejemplo de una fuente de luz de tamaño reducido podría ser la lamparita de una linterna, la cual en efecto arroja sombras muy definidas.

  Una fuente de luz de un área mayor arroja sombras mucho más suaves (menos apreciables).

  Si posamos los ojos por un momento sobre la Fig. 2.19 de abajo. Nos daremos cuenta que su rostro está iluminado por una fuente de luz muy pequeña y concentrada que arroja unas sombras muy apreciables, especialmente bajo la nariz y las cejas. Ambas figuras están iluminadas por fuentes de luz ubicadas por encima, pero la fuente de luz en la Fig. 2.20 es mucho más grande (evidenciado por las sombras suaves) que la de la Fig. 2.19.

Fig. 2.19.

Fig. 2.20.

2.4.4. MANEJO DE DISPOSITIVOS.

Cuando un programador desea emplear aceleración grafica, puede emplear DirectX o también OpenGL. No es que una sea más veloz que la otra, porque ambas se encargan de acceder directamente al chip grafico de la tarjeta de video. En todo caso una tiene ciertas prestaciones que la otra no y viceversa.

En la actualidad las tarjetas de video tienen un procesador grafico que trabaja independiente del procesador central, con su propia memoria y buffer de transmisión, ya que las aplicaciones multimedia de hoy en día requieren mucha velocidad de procesamiento, en estos chips gráficos vienen impresas instrucciones y directivas tanto de DirectX como de OpenGL. En los tiempos de DOS e inicios de Windows las tarjetas gráficas solo venían con el paquete de directivas OpenGL, pero en la actualidad son más difundidas las librerías DirectX por los convenios que la empresa Microsoft tiene con los fabricantes y desarrolladores de software multimedia para plataformas Windows.

2.4.5. CONCLUSIÓN DE HERRAMIENTAS A USARSE.

2.4.5.1. DirectX u OpenGL.

Las 2 API's cumplen con los requerimientos de nuestra aplicación, pero por el medio en que nos desenvolvemos y algunas ventajas que tiene DirectX sobre OpenGL como:

- Rapidez: El rendimiento en el desarrollo de una aplicación multimedia es fundamental, DirectX permite obtener toda la capacidad posible del hardware, inclusive, aprovechando mejoras en el Hardware que podrían aparecer, después de construir la aplicación.

- Facilidad: Sin dudas, tareas como configurar el tipo de video, la profundidad de colores, la reproducción de sonido, la compatibilidad con el hardware existente, resultan tarea más sencillas trabajando con DirectX en Windows.

- Compatibilidad: Al ser de la misma casa Microsoft posee mas compatibilidad con el sistema operativo Windows, incluyendo funciones que interactúan directamente con la librería WIN32 incluida en todas las plataformas Windows y mejorada según su versión, con compatibilidad de diversos dispositivos externos.

Por tales motivos decidimos utilizar como librería gráfica DirectX 8.

2.4.5.2. Lenguajes de Programación.

Los lenguajes de programación más extendidos bajo plataforma Windows y que pueden utilizar DirectX 8.0 son Visual Basic, Visual C++ y Delphi.

Las diferencias residen en la facilidad de programación mientras Visual C++ es un lenguaje más complejo pero a la vez considerado más potente por muchos programadores multimedia tradicionalmente, el Visual Basic y Delphi presentan un entorno más amigable para un programador, y poco difundido por programadores multimedia. Aunque en la actualidad están teniendo mejor acogida especialmente por los programadores que recién empiezan en el campo multimedia.

La poca documentación existente en la actualidad de Delphi y librerías gráficas sobre Visual Basic 6.0 y librerías gráficas, además la familiaridad con éste último nos lleva a la siguiente conclusión:

Utilizar como lenguaje de programación a Microsoft Visual Basic 6.0.

2.4.5.3. Ficheros Gráficos.

Entre los tipos de ficheros mencionados anteriormente se ha escogido el formato de dibujo 3D a los ficheros 3DS, por su documentación y por ser el más extendido en todas las aplicaciones de modelado de entornos 3D y por contar con el conversor a ficheros X, con lo cual se facilita la manipulación de modelos en DirectX.

CAPÍTULO III

3. REALIZACIÓN DEL PROYECTO.

3.1. LIBRERÍAS MICROSOFT DIRECTX 8.0 SDK.

3.1.1. INTRODUCCIÓN.

Anteriormente se usaba el DOS, el sistema operativo más popular entre los PCs, y era donde se desarrollaban los juegos por muchos años, luego surgió Windows, el cual no tenia un buen soporte para el desarrollo de juegos y aplicaciones multimedia, es mas el rendimiento era pésimo y por esto los desarrolladores seguían usando la plataforma DOS para seguir desarrollando sus aplicaciones. En los noventa Microsoft lanzó DirectX, un conjunto de librerías que permitía a los programadores manejar gráficos y sonidos con el mismo rendimiento o mejor que en DOS, y con la ventaja de poder despreocuparse de los drivers del sistema (puesto que cada fabricante provee sus propios drivers a Windows).

DirectX es un conjunto de APIs para programar juegos y aplicaciones 2D/3D en general. Se compone de:

- DirectPlay: Soporte para redes: Internet, modem, serial, etc.
- DirectGraphics: Gráficos en 2D y 3D.
- DirectAudio: Sonido y música.
- DirectInput: Control del ratón, teclado, joystick, etc.

Lo principal de DirectX es que va a correr en cualquier PC aunque no se tenga lo último en hardware, puesto que DirectX automáticamente emula esas características mediante software de forma que al menos la aplicación funcione.

Otra característica es que cada versión nueva incluye nuevas prestaciones que no tenían versiones anteriores, Microsoft es muy cuidadosa en preservar la funcionalidad de las versiones previas, con esto se asegura que una aplicación hecha con DirectX 5 correrá muy bien en DirectX 6, 7, 8 y posteriores.

Microsoft DirectX SDK (Software Development Kit).

Es una compilación de librerías, archivos de cabecera, herramientas de diagnóstico, utilitarios, ejemplos en Visual Basic y en C/C++, y una extensa documentación que nos permitirá el desarrollo de aplicaciones que utilicen DirectX. También, incluye a DirectX propiamente dicho. DirectX SDK es totalmente gratuito y se puede bajar desde el sitio de Microsoft. (154 Megas).

DirectX y COM (Component Object Model).

El estándar COM es una técnica usada por Microsoft para proveer a los programadores de Windows un modelo de objetos reusables. Los componentes COM típicamente residen en un archivo DLL, el cual puede acceder a diferentes aplicaciones, muchas funciones del sistema operativo, y desde la versión DirectX 7 hace posible que Visual Basic 6.0 y DirectX sean compatibles.

3.1.2. COMPONENTES DIRECTX 8.

Todos los componentes de DirectX 8 poseen nuevas e importantes características que mejoran significativamente las posibilidades de DirectX. Se han reescrito casi completamente tres componentes: los gráficos (DirectGraphics), el sonido (DirectAudio) y las redes (DirectPlay). DirectInput incluye nuevas características de configuración, asignación de acciones y compatibilidad internacional mejorada. Además, por primera vez, DirectShow forma parte de DirectX.

En la Fig 3.1. se muestra los componentes de DirectX.

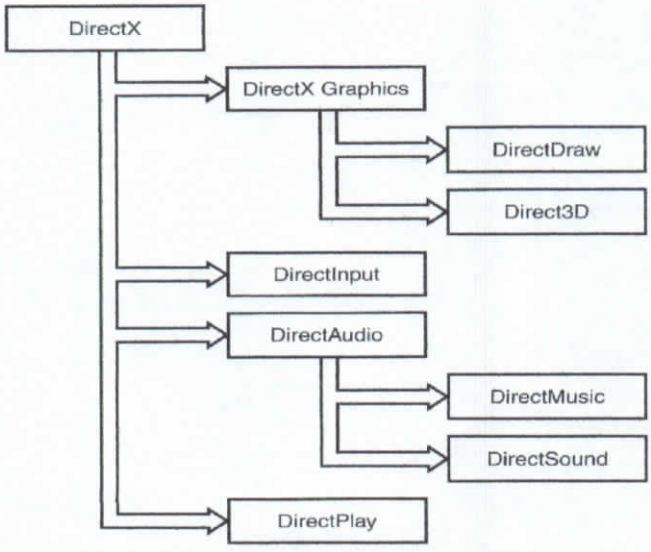


Fig. 3.1.

3.1.2.1. DirectGraphics.

Microsoft DirectDraw y Microsoft Direct3D se han combinado en un único componente DirectGraphics. La API se ha actualizado completamente para que resulte más fácil de utilizar y para que sea compatible con el último hardware de gráficos.

Son muchas las características y mejoras de rendimiento que se han agregado a las interfaces API de DirectGraphics.

- Lenguaje de procesamiento de vértices programable.
- Lenguaje de procesamiento de píxeles programable.
- Compatibilidad con el procesamiento de muestreo múltiple.
- Sprites de punto.
- Texturas volumétricas 3D.

- Compatibilidad para primitivas avanzadas.
- Compatibilidad de herramientas de creación de contenido 3D.
- Combinación de vértices indexados.
- Expansión de la biblioteca de utilidades de Direct3D.

Uno de los cambios más notables en DirectGraphics es la fusión de DirectDraw (para gráficos 2D) y Direct3D (para gráficos 3D) en una sola en común.

Esta nueva integración simplifica la inicialización y el control del núcleo de Direct3D y realiza esas operaciones de forma más sencilla. Además de simplificar la inicialización de aplicaciones, este cambio mejora la asignación de datos y el rendimiento de la administración, lo que reduce la gran cantidad de memoria utilizada. Una de las razones para fusionar las interfaces es el hecho de que DirectDraw no deja de desarrollarse y que todos los cambios efectuados, tanto en DirectX 6.0 como en DirectX 7.0, son de naturaleza superficial. La figura 3.2 muestra la nueva arquitectura de Direct3D.

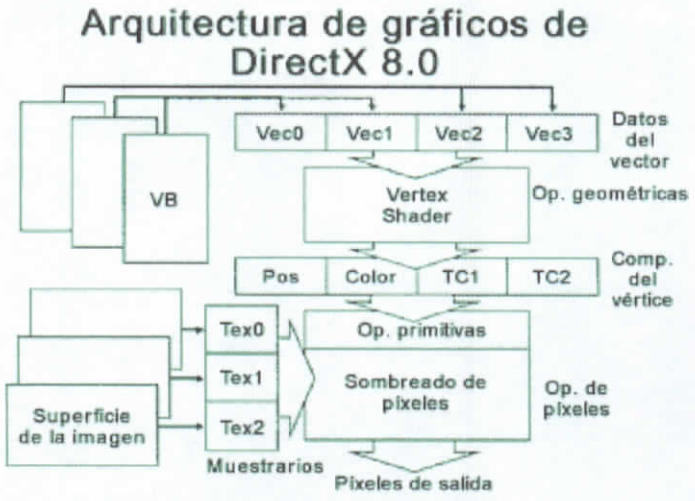


Fig. 3.2.

La incorporación de sombreados programables es otra de las características más importantes de DirectX 8 introducida en la actualización de Direct3D. Los lenguajes pueden escalarse sin problemas. Las técnicas de sombreado se suministran tanto para el procesamiento de píxeles como para el de vértices.

El lenguaje de procesamiento de vértices programable permite a los programadores escribir sombreados de hardware que se pueden aplicar a:

- Animación de transformación/interpolación.
- Modelos de luz definidos por el usuario.
- Asignación del entorno general.
- Geometría de procedimientos.
- Cualquier otro algoritmo definido por el desarrollador.

El lenguaje de procesamiento de píxeles programable permite a los programadores escribir sombreados de hardware que se pueden aplicar a:

- Expresiones de combinación de texturas generales.
- Iluminación por píxel.
- Asignación del entorno por píxel (para efectos foto realistas espectaculares).
- Algoritmos definidos por otros desarrolladores.

Hay que tener en cuenta que la incorporación de técnicas de sombreado para las operaciones de píxeles y de vértices coexisten con los métodos "antiguos" de transformación, iluminación y multitextura utilizados en la canalización de vértices. La figura 3.3 ilustra la canalización de los gráficos de DirectX 8.

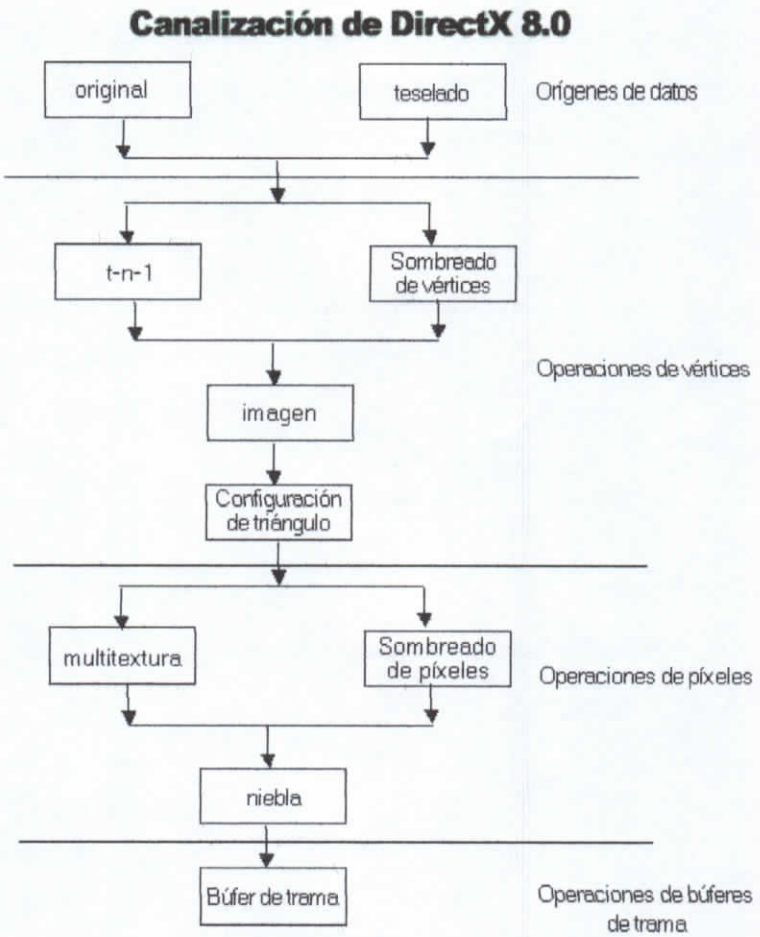


Fig. 3.3.

Algunas otras características facilitan la implementación de una variedad de efectos especiales. La compatibilidad para el procesamiento de muestreo múltiple permite el alisado de los bordes de escena completa y efectos de muestreo múltiple, como la difuminación de movimiento y la profundidad de campo. La compatibilidad con el sprite de punto permite el procesamiento de alto rendimiento de sistemas de partículas para chispas, explosiones, lluvia, nieve, etc. Las texturas de volumen permiten la atenuación de alcance en la iluminación por píxel y los efectos atmosféricos volumétricos y se pueden aplicar a una geometría más compleja.

La compatibilidad de primitivas avanzadas mejora el aspecto del contenido 3D y facilita su asignación desde importantes herramientas de edición 3D. Además, los gráficos de DirectX 8 incluyen plug-ins de herramientas de creación de contenido 3D para llevar a cabo la exportación de mallas con máscara en archivos X para su posterior uso en Direct3D (mediante una variedad de técnicas de Direct3D).

3.1.2.2. DirectAudio.

Microsoft DirectAudio ofrece una nueva arquitectura para la reproducción de música integrada y efectos de sonido. Aunque se siguen utilizando los nombres Microsoft DirectSound y Microsoft DirectMusic, no existe una distinción clara entre ambos y se espera que las APIs de DirectMusic sean las elegidas para la creación de efectos de sonido interactivos.

Algunas de las nuevas características de DirectAudio incluyen:

- Integración de archivos wav y sonidos basados en mensajes en un único mecanismo de reproducción.
- Síntesis DLS2, que incluye efectos especiales.
- Secuencias de comandos de sonido.
- Objetos contenedores que incluyen todos los componentes de un proyecto DirectMusic Producer en un único archivo.
- Mayor control sobre los rendimientos, segmentos y pistas.

La nueva arquitectura de sonido trata el sintetizador de DirectMusic como el generador de sonido principal de DirectAudio. Este sintetizador Downloadable Sounds Level 2 (DLS2), enormemente optimizado, crea todos los sonidos, los mezcla y envía la salida a los búferes de DirectAudio para continuar con el procesamiento. El sintetizador de DirectMusic puede también mezclar varias voces individuales antes de producir la salida. De esta manera, se pueden procesar diversos sonidos individuales mediante los mismos efectos de sonido y colocarlos en la misma posición en el espacio 3D, utilizando tan sólo un búfer de DirectSound3D y reduciendo así el uso del CPU y los requisitos del hardware 3D. Consulte la figura 3.4 para ver la nueva arquitectura de sonido.

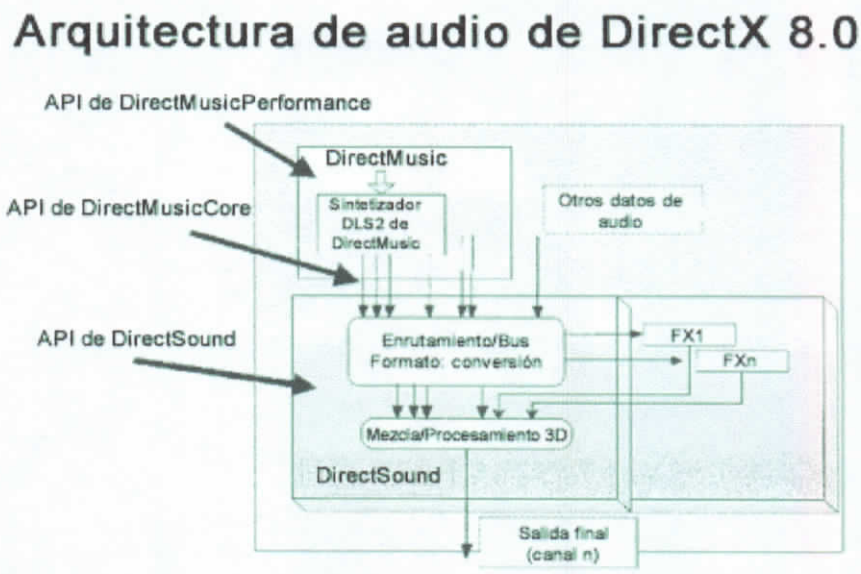


Fig. 3.4.

3.1.2.3. DirectPlay.

DirectPlay es una interfaz de software de alto nivel entre aplicaciones y servicios de comunicación que facilita la conexión de aplicaciones a través de Internet, de un módem o de una red. DirectPlay ofrece tanto un grado avanzado de nivel de transporte (por ejemplo, entrega garantizada y no garantizada, límite de tráfico para los vínculos lentos, detección de conexiones perdidas). La figura 3.5 ilustra la arquitectura de DirectPlay y la forma en la que ofrece independencia de los proveedores de servicios de comunicaciones.

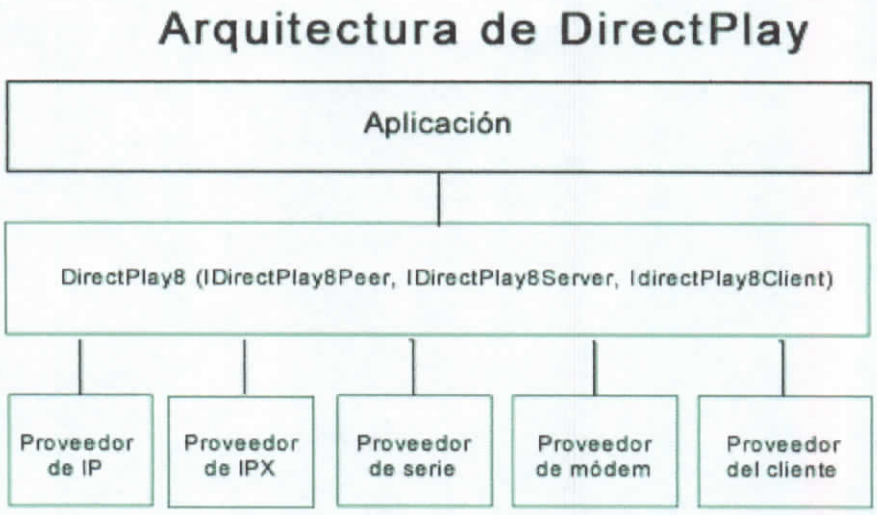


Fig. 3.5.

A continuación se presenta una lista con las nuevas características de la API DirectPlay de Microsoft para DirectX 8:

- Se han reescrito completamente las interfaces.
- La recepción es ahora independiente del resto de DirectPlay.
- Se ha agregado la transmisión por voz.
- La información de direccionamiento ha pasado de datos basados en GUID a un formato de datos basado en URL.

- Se ha agregado una mayor escalabilidad y una mejor administración de la memoria.
- Se ha agregado una compatibilidad mejorada con los servidores de seguridad y los traductores de dirección de red.

3.1.2.4. DirectInput.

DirectInput ofrece una interfaz de última generación para una variedad de dispositivos de entrada (como joysticks, Mouse multi-botón, etc.) y para dispositivos de fuerza de respuesta. Al trabajar directamente con controladores de dispositivos, DirectInput evita el sistema de mensajería de Windows para mejorar el rendimiento.

A continuación se muestra una lista con las nuevas características de la API Microsoft DirectInput:

- Asignación de acciones.
- Mayor compatibilidad con aplicaciones internacionales.
- Nueva compatibilidad para la creación de interfaces.
- Cambios en los datos del deslizador del joystick.

La asignación de acciones es un paso importante para solucionar el problema de compatibilidad de los dispositivos de entrada. La asignación de acciones simplifica el bucle de entrada y reduce la necesidad de controladores de juego personalizados, de perfiles de dispositivos personalizados y de interfaces de usuario para configuraciones personalizadas en los juegos.

La asignación de acciones también incluye una interfaz de usuario predeterminada que permite a los usuarios configurar los dispositivos de forma más rápida y sencilla. Esta API estándar se complementa con una interfaz de usuario de bajo nivel que permite a las aplicaciones tener un acceso directo a las imágenes de los dispositivos para utilizarlas en sus interfaces de usuario personalizadas.

3.1.3. GENERACIÓN DE ESPACIOS.

La librería encargada de la generación de espacios en DirectX como se dijo anteriormente es Direct3D la cual forma parte de DirectGraphics.

Programar aplicaciones con Direct3D requiere tener conocimientos con principios de geometría descriptiva, entre los temas a tomarse en cuenta para el diseño en 3D tenemos:

- Tubería de renderizado.
- Sistemas de coordenadas 3D.
- Primitivas 3D.
- Reglas para el rasterizado de triángulos.
- Técnicas de Sombreado.
- Transformaciones 3D y Matrices.
- Intercambio de Página y Back Buffering.

Estos temas posibilitan un alto nivel de entendimiento de los conceptos básicos empleados por una aplicación Direct3D.

3.1.3.1. Tubería de Renderizado.

La tubería de renderizado de Microsoft Direct3D consiste en una serie de etapas de procesamiento que especifican el comportamiento de la iluminación y transformación de vértices, y el mezclado de textura y píxel.

Cuando se diseña una aplicación 3D, se puede especificar el mundo en cualquier unidad que se crea conveniente, desde micrones a parsecs. La aplicación pasa una descripción de ese mundo a Direct3D. Esa descripción consta del tamaño, las posiciones relativas de todos los objetos en el mundo, la posición y orientación del observador. Direct3D transforma esas descripciones (vértices) en series de píxeles en la pantalla. El primer paso de este proceso es la transformación de la geometría suministrada en una imagen bidimensional que es la tubería de geometría, también llamada la tubería de transformación. Después del procesamiento de los vértices, los datos del píxel se usan para realizar multitexturado y mezclado de niebla. Luego se realizan operaciones de alpha, plantilla y pruebas de profundidad. Finalmente, el rasterizador realiza el mezclado del frame buffer.

3.1.3.2. Sistemas de Coordenadas 3D.

Usualmente, las aplicaciones 3D usan dos tipos de sistemas de coordenadas cartesianas. Con la regla de la mano derecha o la mano izquierda. En ambos, el eje X positivo apunta hacia la derecha, y el eje Y positivo apunta hacia arriba. Se puede recordar la dirección positiva del eje Z apuntando los dedos de la mano derecha o izquierda en la dirección X positiva, y moviéndolos hacia la dirección Y positiva. La dirección del pulgar, es la dirección en la que apunta el eje Z positivo para ese sistema de coordenadas. La figura 3.6 muestra los dos sistemas de coordenadas.

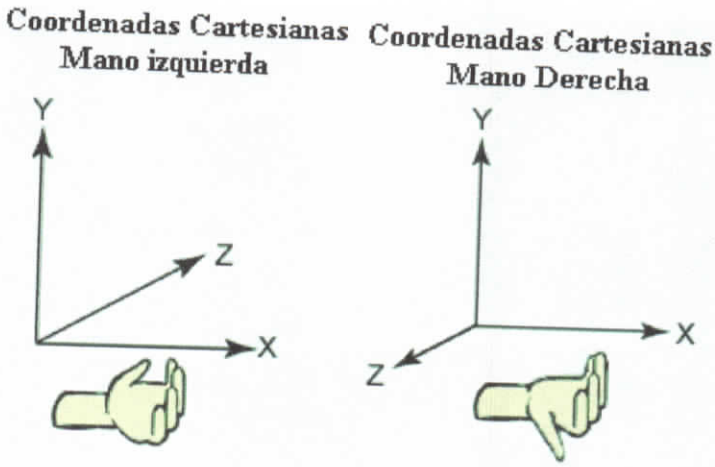


Fig. 3.6.

Microsoft Direct3D utiliza el sistema de coordenadas cartesianas con la regla de la mano izquierda. Si se está transportando una aplicación que está basada en un sistema mano derecha, debe hacer cambios a los datos pasados a Direct3D.

Aunque los sistemas de coordenadas mano derecha y mano izquierda son los sistemas más comunes, hay una gran variedad de otros sistemas de coordenadas usados en programas 3D. Por ejemplo, no es inusual para aplicaciones de modelado 3D usar un sistema de coordenadas en el cual el eje Y apunta hacia o desde el observador, y el eje Z apunta hacia arriba. En este caso, son mano derecha los sistemas de coordenadas que tienen cualquier eje positivo (X, Y ó Z) apuntando hacia el observador. Son mano izquierda los sistemas de coordenadas que tienen un eje positivo (X, Y ó Z) alejándose del observador. Si está convirtiendo una aplicación de modelado de un sistema de mano izquierda donde el eje Z positivo apunta hacia arriba, debe hacerse una rotación en todos los datos del vértice.

Las operaciones esenciales aplicadas en objetos definidos en sistemas de coordenadas 3D son traslación, rotación y escalado. Se pueden combinar esas transformaciones básicas para crear una matriz de transformación.

Cuando combina esas operaciones, los resultados no son conmutativos, el orden en el cual multiplica las matrices es importante.

3.1.3.3. Primitivas 3D.

Una primitiva 3D es una colección de vértices que forman una única entidad 3D. La primitiva más simple es una colección de puntos en un sistema de coordenadas 3D, que se llama lista de puntos.

En general, las primitivas 3D son polígonos. Un polígono es una figura 3D cerrada delimitada por tres vértices como mínimo. El polígono más simple es un triángulo.

Microsoft Direct3D usa triángulos para componer la mayoría de sus polígonos porque los tres vértices de un triángulo siempre son coplanares (para definir un plano se necesita como mínimo tres puntos). Dibujar vértices no coplanares es ineficiente. Se pueden combinar triángulos para formar polígonos y figuras grandes y complejas.

La figura 3.7 muestra un cubo. Dos triángulos forman cada cara del cubo. El conjunto de triángulos completo forma una primitiva cúbica. Se pueden asignar texturas y materiales a las superficies de las primitivas para hacerlas parecer como una sola forma sólida.

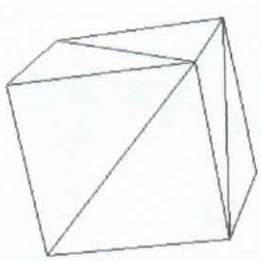


Fig. 3.7.

También se pueden usar triángulos para crear primitivas cuyas superficies parezcan ser curvas suaves. La figura 3.8 muestra cómo una esfera puede ser simulada con triángulos. Luego de que se le asigna un material, la esfera se ve curva cuando se la dibuja.

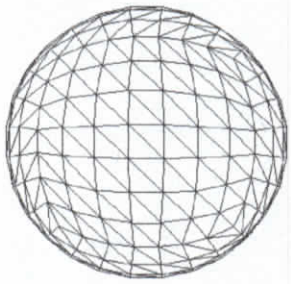


Fig. 3.8.

3.1.3.4. Reglas para la Rasterización de Triángulos.

En general, los puntos especificados por los vértices no coinciden con los píxeles en la pantalla. Cuando esto pasa, Microsoft Direct3D aplica las reglas para la rasterización de triángulos para decidir qué píxeles están involucrados en un triángulo.

Direct3D usa una llamada al método de llenado arriba-izquierda para llenar triángulos. Es la misma llamada usada para rectángulos en los gráficos GDI (Graphics Device Interfase) y OpenGL. En Direct3D, el centro del píxel es el punto decisivo. Si el centro está dentro de un triángulo, el píxel es parte del triángulo. Los centros de los píxeles se encuentran en coordenadas enteras.

La descripción de las reglas para la rasterización de triángulos usada por Direct3D no se ajusta necesariamente a todo el hardware disponible. Las pruebas pueden descubrir variaciones menores en las implementaciones de esas reglas.

La figura 3.9 muestra un rectángulo cuya esquina superior izquierda se encuentra en (0, 0) y cuya esquina inferior derecha está en (5, 5). Este rectángulo llena 25 píxeles, como se esperaba. El ancho del rectángulo se define como derecha menos izquierda. El alto se define como abajo menos arriba.

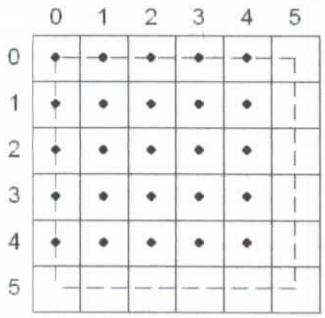


Fig. 3.9.

En la llamada al método de llenado arriba-izquierda, arriba se refiere a la posición vertical de longitudes horizontales, e izquierda se refiere a la posición horizontal de los píxeles dentro de esa longitud. Un eje no puede ser un eje superior a menos que sea horizontal en general, la mayoría de los triángulos sólo tienen ejes izquierdo y derecho.

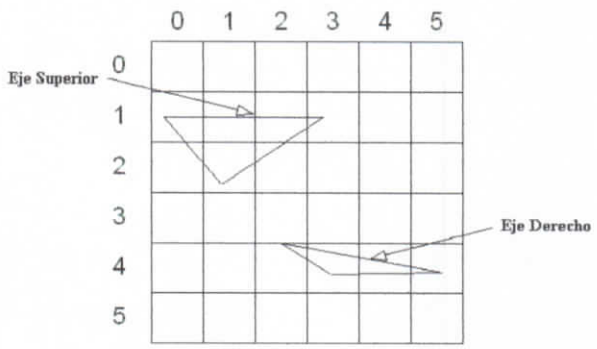


Fig. 3.10.

La llamada al método de llenado arriba-izquierda determina la acción tomada por Direct3D cuando un triángulo pasa por el centro de un píxel. La figura 3.11 muestra dos triángulos, uno en (0, 0), (5, 0) y (5, 5) y el otro en (0, 5) (0, 0) y (5, 5).

El primer triángulo en este caso tiene 15 píxeles (mostrados en negro), mientras que el segundo tiene solamente 10 píxeles (mostrados en gris) porque el eje compartido es el eje izquierdo del primer triángulo.

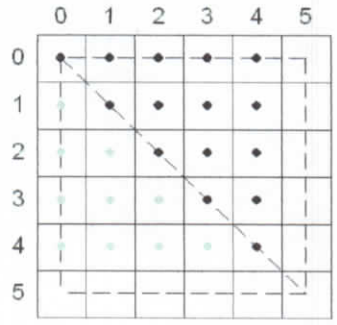


Fig. 3.11.

3.1.3.5. Técnicas de Sombreado.

El modo de sombreado usado para renderizar un polígono tiene un profundo efecto en su apariencia. Los modos de sombreado determinan la intensidad del color y luminosidad de todo punto de la superficie de un polígono. Microsoft Direct3D usa dos modos de sombreado:

3.1.3.5.1. Sombreado Plano.

En el modo de sombreado plano, la tubería de renderizado de Direct3D, usa el color del material del polígono en su primer vértice como el color del polígono entero. Los objetos 3D que se renderizan con sombreado plano tienen bordes visiblemente afilados entre polígonos si no son coplanares.

La figura 3.12 muestra una tetera renderizada con sombreado plano. El contorno de cada polígono es claramente visible. El sombreado plano es el método de sombreado menos costoso computacionalmente.

Fig. 3.12.

3.1.3.5.2. Sombreado Gouraud.

Cuando Microsoft Direct3D renderiza un polígono usando sombreado Gouraud, calcula el color para cada vértice usando la normal del vértice y los parámetros de luminosidad. Entonces, interpola el color a través de la superficie del polígono. La interpolación se efectúa linealmente. Por ejemplo, si la componente roja del color del vértice 1 es 0.8 y la componente del vértice 2 es 0.4, usando el modo de sombreado Gouraud y el modelo de color RGB, el módulo de luminosidad de Direct3D asigna una componente roja de 0.6 al píxel en el punto medio de la línea entre esos vértices.

La figura 3.13 muestra el sombreado Gouraud. Esta tetera está compuesta de muchos polígonos triangulares planos. Sin embargo, el sombreado Gouraud hace que la superficie del objeto parezca curva y lisa.

Fig. 3.13.

3.1.3.6. Transformaciones 3D y Matrices.

Microsoft Direct3D usa matrices para ejecutar transformaciones 3D. En aplicaciones que trabajan con gráficos 3D, se pueden usar transformaciones geométricas para hacer lo siguiente:

- Expresar la ubicación relativa de un objeto con respecto a otro objeto.
- Rotar y dimensionar objetos.
- Cambiar puntos de vista, direcciones, y perspectivas.

Se puede transformar cualquier punto en otro usando una matriz de 4×4. En el siguiente ejemplo, una matriz reinterpreta el punto (x, y, z), produciendo el nuevo punto (x', y', z').

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix}$$

Se ejecutan las siguientes operaciones sobre (x, y, z) y la matriz para producir el punto (x', y', z').

$$x' = (x \times M_{11}) + (y \times M_{21}) + (z \times M_{31}) + (1 \times M_{41})$$
$$y' = (x \times M_{12}) + (y \times M_{22}) + (z \times M_{32}) + (1 \times M_{42})$$
$$z' = (x \times M_{13}) + (y \times M_{23}) + (z \times M_{33}) + (1 \times M_{43})$$

Las transformaciones más comunes son traslación, rotación, y escalado. Se pueden concatenar las matrices que producen estos efectos en una matriz simple para calcular varias transformaciones de una vez. Por ejemplo, se puede construir una matriz simple para trasladar y rotar una serie de puntos. Las matrices se escriben en el siguiente orden fila-columna.

3.1.3.6.1. Traslación.

La siguiente transformación traslada el punto (x, y, z) a un nuevo punto (x', y', z').

$$[x'y'z'1] = [x\ y\ z\ 1]\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{bmatrix}$$

3.1.3.6.2. Rotación.

La siguiente transformación rota el punto (x, y, z) sobre el eje x, produciendo un nuevo punto (x', y', z').

$$[x'y'z'1] = [x\ y\ z\ 1]\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La siguiente transformación rota el punto sobre el eje y.

$$[x'y'z'1] = [x\ y\ z\ 1]\begin{bmatrix} \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La siguiente transformación rota el punto sobre el eje z.

$$[x'y'z'1] = [x\ y\ z\ 1]\begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

En estas matrices de ejemplo, la letra griega theta (θ) simboliza el ángulo de rotación, en radianes. Los ángulos se miden en el sentido horario cuando se mira sobre el eje de rotación hacia el origen.

3.1.3.6.3. Escalado.

La siguiente transformación escala el punto (x, y, z) por medio de valores arbitrarios en las direcciones x, y, z a un nuevo punto (x', y', z').

$$[x'y'z'1] = [x\ y\ z\ 1]\begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.1.3.6.4. Concatenación de Matrices.

Una ventaja de usar matrices es que se pueden combinar los efectos de dos o más matrices multiplicándolas. Esto significa que, para rotar un modelo y luego trasladarlo a alguna ubicación, no se necesita aplicar dos matrices. En lugar de eso, se multiplican las matrices de rotación y traslación para producir una matriz compuesta que contiene todos sus efectos. Este proceso, llamado concatenación de matrices, se puede escribir con la siguiente fórmula.

$$C = M_1 \cdot M_2 \cdot M_{n-1} \cdot M_n$$

En esta fórmula, C es la matriz compuesta que se está creando, y M1 a Mn son las transformaciones individuales que contiene la matriz C. En la mayoría de los casos, sólo se concatenan dos o tres matrices, pero no hay límite.

3.1.3.7. Intercambio de Página y Back Buffering.

El intercambio de página es una técnica clave en programación de aplicaciones multimedia, de animación y juegos. Esta técnica es análoga a la utilizada en las animaciones con papel. En cada página el artista modifica las figuras un poco, para que cuando se cambia rápidamente las hojas, el dibujo parece moverse.

La técnica de intercambio de página es muy similar a este proceso. Direct3D implementa el intercambio de páginas mediante una cadena de intercambio. Inicialmente se designa una serie de buffers que están destinados a intercambiarse con la pantalla de la misma manera que un artista intercambia hojas de papel. El primer buffer se denomina buffer de color frontal, y los buffers detrás de éste son buffers traseros (back buffers). Su aplicación dibuja en un buffer trasero, y luego lo intercambia con el buffer de color frontal, para que aparezca en la pantalla. Mientras el sistema muestra la imagen, el programa vuelve a escribir en un buffer trasero. Este proceso continúa mientras esté animando, permitiéndole animar rápida y eficientemente.

Direct3D facilita este proceso, utilizando un simple buffer doble (un buffer frontal con un buffer trasero), hay esquemas más sofisticados que utilizan buffers traseros adicionales.

3.2. SOLUCIÓN.

3.2.1. ELEMENTOS DE UNA APLICACIÓN 3D EN GENERAL.

Una Aplicación 3D esta compuesta de varios elementos que hacen posible simular la realidad en un entorno informático, estos elementos son:

3.2.1.1. Espacio.

El Espacio es la manera de reservar sitios de memoria para que la aplicación 3D pueda trabajar, tanto en video como en RAM. Mediante las funciones respectivas de inicialización dependiendo la API que se ocupe (OpenGL, DirectX).

Además configura los dispositivos (video, audio, red, etc.), para ser accedidos directamente según las necesidades de la aplicación, entre estas configuraciones de pantalla, profundidad de color, sincronización de refresco de monitor, generación del z-buffer (profundidad), abrir canales de sonido, abrir puertos de red, etc. Generando una ventana lista para contener un mundo 3D.

3.2.1.2. Cámara.

La Cámara simula los ojos de una persona en el mundo 3D, permitiendo observar el mundo, sin una cámara en la aplicación no se podría observar éste. Para esto se trabajan con matrices que permiten hacer operaciones de traslación, rotación, proyección, mediante operaciones de matrices.

3.2.1.3. Luces.

Las Luces deben ser programadas para poder distinguir de mejor manera los objetos, simulando diversas fuentes de luz como el sol, focos, linternas, lámparas etc. Para lo cual hay que asignar la posición, el color en formato RGB, intensidad, y si se las prende o apaga, mediante vectores.

3.2.1.4. Objetos.

Los Objetos forman el mundo 3D que se visualiza en la aplicación mediante la cámara, estos están formados por triángulos, y son modelados en diferentes aplicaciones especializadas, están en formato ASCII o Binario, pueden contener información de texturas, normales de caras, materiales, animación, luces. Se los posiciona en el espacio mediante coordenadas (X, Y, Z), y se los puede trasladar, rotar, escalar mediante operaciones de matrices.

3.2.1.4.1. Texturas y Materiales.

- Texturas son fotos que se aplican a las caras que forman el objeto para hacerlo más real, por ejemplo aplicar una textura de ladrillos a la pared de una casa modelada.

  El tamaño esta dado en píxeles que son potencias de dos, ejemplo 32*32, 64*64, 128*128, 256*256, dependiendo del tamaño la aplicación es más rápida o lenta. Los objetos pueden o no tenerlas.

- Materiales describen la forma como los polígonos reflejan o emiten luz, simulando la forma como en el mundo real se perciben los colores, esta información siempre tiene que estar presente en el objeto.

La figura 3.14. muestra los elementos que forman la aplicación 3D

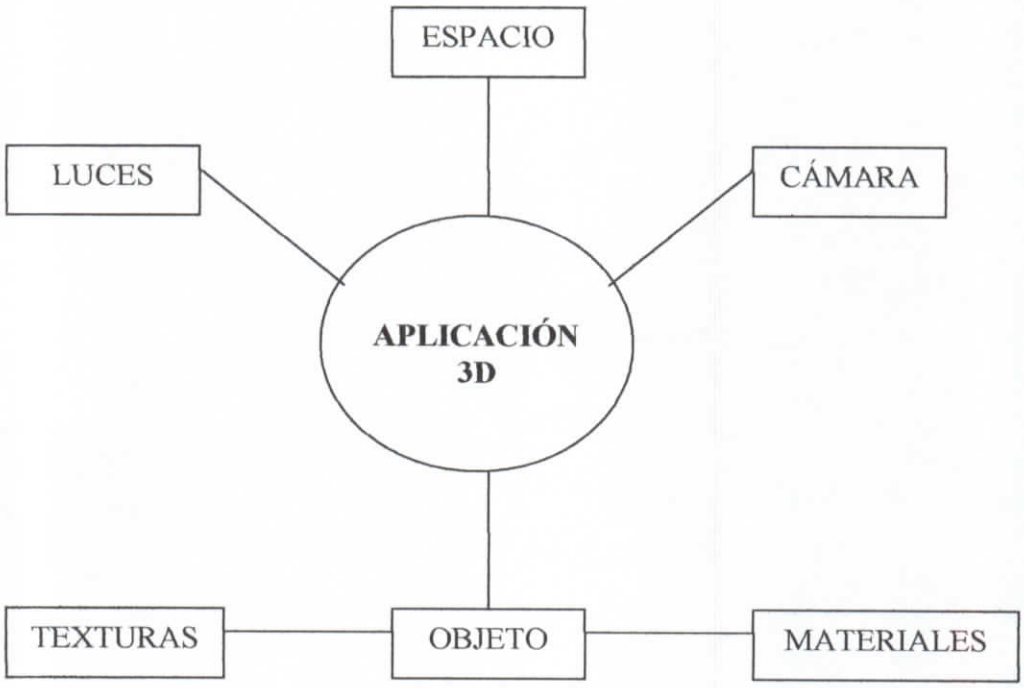


Fig. 3.14.

3.2.2. ELEMENTOS DE APLICACIÓN GENERADA EN VISUAL BASIC 6.0 Y DIRECTX 8.0; APLICACIÓN DESKTOP.

3.2.2.1. Funciones de inicialización de aplicación.

Funciones que inicializan las variables globales a utilizarse en los procedimientos necesarios para correrlos en la aplicación; al igual que los dispositivos a usarse, además inicialización de la geometría y de su ubicación en el mundo mediante matrices.

3.2.2.1.1. Form Load.

En este procedimiento las variables globales son inicializadas con valores para su posterior uso en los procedimientos descritos a continuación.

```
Private Sub Form_Load()
Me.Show
aux = False
tiporender = D3DFILL_SOLID
Angle = g_360d
TURN_SPEED = g_90d / 18
MOVE_SPEED = 0.8
r = 0.2371: g = 0.2241: b = 0.1291
r1 = 0.2371: g1 = 0.2241: b1 = 0.1291
r2 = 0.2371: g2 = 0.2241: b2 = 0.1291
r3 = 0.2371: g3 = 0.2241: b3 = 0.1291
Check1.Enabled = False: Check2.Enabled = False
Check3.Enabled = False: Check4.Enabled = False
Check5.Enabled = False: Check6.Enabled = False
Check7.Enabled = False:Check8.Enabled = False
colorfondo = D3DColorARGB(0, 40, 80, 120)
mouseabajo = False
mousemueve = False
caras = nMaterials - 1
initdirectinput
initd3d
** Activa el bucle principal **
Timer1.Enabled = True
End Sub
```

3.2.2.1.2. Inicialización DirectX 3D.

Aquí se inicializan los parámetros necesarios para generar el espacio de trabajo de la aplicación 3D, tales como: configuración de tarjeta de video y funciones de ventana, crear z-buffer, parámetros generales de las luces, tipo de procesamiento de vértices, manejo de errores.

Sub initd3d()

```
** Seteo de variables Direct3D **
Set v_md3d8 = v_mdx8.Direct3DCreate()
Set v_d3dx8 = New D3DX8
** Manejo de errores **
On Error GoTo manejoerror
** Seteo del adaptador de tarjeta de video **
v_md3d8.GetAdapterDisplayMode D3DADAPTER_DEFAULT, dmode
v_presentparad3d8.BackBufferFormat = dmode.Format
v_presentparad3d8.Windowed = 1
v_presentparad3d8.SwapEffect = D3DSWAPEFFECT_FLIP
v_presentparad3d8.hDeviceWindow = Picture1.hWnd
v_presentparad3d8.EnableAutoDepthStencil = 1
v_presentparad3d8.AutoDepthStencilFormat = D3DFMT_D16
** Modulo que setea el procesamiento de vertices por software o por hardware **
rendertipo = D3DDEVTYPE_HAL
rendertrabajo
** Activación de luces en el dispositivo **
v_dispd3d8.SetRenderState D3DRS_LIGHTING, 1
** Activacion del Z-Buffer (Buffer de profundidad) **
v_dispd3d8.SetRenderState D3DRS_ZENABLE, 1
** Llama al Procedimiento de seteo del mundo **
 initmatrices
```

```
** Carga geometrías y luces y controla errores **
If InitialiseGeometry = False And aux = True Then
    MsgBox "no geometria"
    GoTo manejoerror
End If
If SetupLights = False Then
  GoTo manejoerror
End If
If v_dispd3d8 Is Nothing Then
  manejoerror:
 MsgBox "Error de inicialización: no se pudo crear el dispositivo Directx8"
** Limpia los dispositivos y finaliza la aplicación **
  limpiado
End If
End Sub
```

3.2.2.1.3. Inicialización DirectXInput.

En este procedimiento se inicializa los parámetros referentes a dispositivos de entrada como el teclado que es usado para el movimiento de la cámara.

```
Sub initdirectinput()
** Se crea el dispositivo DirectX Input **
 Dim devprop As DIPROPLONG
  Dim devinfo As DirectInputDeviceInstance8
  Set v_di = v_mdx8.DirectInputCreate
** Captura el dispositivo de entrada (teclado) **
Set v_dispositivo_di = v_di.CreateDevice("GUID_SysKeyboard")
  v_dispositivo_di.SetCommonDataFormat DIFORMAT_KEYBOARD
```

*** Activa el dispositivo ***

v_dispositivo_di.Acquire

End Sub

3.2.2.1.4. Configuración del Mundo y Cámara.

Lugar del código donde se inicializan los valores para posicionar la cámara frente al modelo 3D, operaciones que limpian e inicializan las matrices usadas por la cámara.

Sub initmatrices()

camx = boundingvector.X

camy = boundingvector.Y

camz = boundingvector.Z + 10

Angle = g_180d

pitch = 0

*** matriz del mundo ***

D3DXMatrixIdentity matWorld

v_dispd3d8.SetTransform D3DTS_WORLD, matWorld

*** matriz de cámara ***

D3DXMatrixLookAtLH matView, MakeVector(0, 50, 90), MakeVector(0, 0, 0), MakeVector(0, 1, 0)

v_dispd3d8.SetTransform D3DTS_VIEW, matView

*** matriz de proyeccion ***

D3DXMatrixPerspectiveFovLH matProj, pi / 4, 1, 0.1, 1000

v_dispd3d8.SetTransform D3DTS_PROJECTION, matProj

Picture1.SetFocus

End Sub

3.2.2.1.5. Inicialización de Geometría.

Función que carga el objeto 3D, crea el buffer de vértices, normaliza caras para reflejo de luz, crea buffer de materiales y optimiza la malla quitando vértices innecesarios para aumentar la velocidad de procesamiento.

Private Function InitialiseGeometry() As Boolean

*** Manejo de errores ***

On Error GoTo BailOut:

*** Declaración de Variables ***

Dim mtrlBuffer As D3DXBuffer

Dim i As Long

Dim TextureFile As String

Dim adiacenbuffer As D3DXBuffer

Dim meshtemp As D3DXMesh

*** Obtener el archivo ***

Set Mesh = v_d3dx8.LoadMeshFromX(ubicacion, D3DXMESH_MANAGED, v_dispd3d8, adiacenbuffer, mtrlBuffer, nMaterials)

If Mesh Is Nothing Then GoTo BailOut:

*** Redimensiona la malla ***

ReDim MeshMaterials(nMaterials) As D3DMATERIAL8

ReDim MeshTextures(nMaterials) As Direct3DTexture8

*** Llenado del vector de materiales ***

For i = 0 To nMaterials - 1

v_d3dx8.BufferGetMaterial mtrlBuffer, i, MeshMaterials(i)

MeshMaterials(i).Ambient = MeshMaterials(i).diffuse

TextureFile = v_d3dx8.BufferGetTextureName(mtrlBuffer, i)

If TextureFile <> "" Then

```
    Set MeshTextures(i) = v_d3dx8.CreateTextureFromFileEx(v_dispd3d8, App.Path
& "\" & TextureFile, 128, 128, D3DX_DEFAULT, 0, D3DFMT_UNKNOWN,
D3DPOOL_MANAGED, D3DX_FILTER_LINEAR, D3DX_FILTER_LINEAR, 0, ByVal
0, ByVal 0)

    End If

Next i

** Crea un bounding sphere para centrar la malla **

Call v_d3dx8.ComputeBoundingSphereFromMesh(Mesh, boundingvector,
radiobounding)

** Cambia el formato de la malla **

If (Mesh.GetFVF <> (D3DFVF_XYZ Or D3DFVF_NORMAL Or D3DFVF_TEX1))
Then

    Set mesh1 = Mesh.CloneMeshFVF(D3DXMESH_SYSTEMMEM,
D3DFVF_VERTEX, v_dispd3d8)

  If mesh1 Is Nothing Then GoTo BailOut:

    If Mesh.GetFVF <> D3DFVF_NORMAL Then

      v_d3dx8.ComputeNormals mesh1

    End If

    Set Mesh = mesh1

  End If

** Optimiza la malla **

  Dim s As Long:   Dim adjBuf1() As Long:   Dim adjBuf2() As Long

  Dim facemap() As Long:   Dim newmesh As D3DXMesh

  Dim vertexMap As D3DXBuffer

  s = adiacenbuffer.GetBufferSize

  ReDim adjBuf1(s / 4):   ReDim adjBuf2(s / 4)

  s = Mesh.GetNumFaces

  ReDim facemap(s)

  v_d3dx8.BufferGetData adiacenbuffer, 0, 4, s * 3, adjBuf1(0)

  Set newmesh = Mesh.Optimize(D3DXMESHOPT_COMPACT Or
D3DXMESHOPT_ATTRSORT Or D3DXMESHOPT_VERTEXCACHE, adjBuf1(0),
adjBuf2(0), facemap(0), vertexMap)
```

Set Mesh = newmesh

caras = nMaterials - 1

initmatrices

*** Activa los menús ***

posoriginal.Enabled = True: barraSolido.Enabled = True

barraAlambre.Enabled = True: botonaumentacaras.Enabled = True

botondisminuyecaras.Enabled = True: prendeluces.Enabled = True

apagaluces.Enabled = True: botoncolorluz.Enabled = True

Check1.Enabled = True: Check2.Enabled = True

Check3.Enabled = True: Check4.Enabled = True

*** Manejo de errores ***

InitialiseGeometry = True

Exit Function

BailOut:

InitialiseGeometry = False

End Function

3.2.2.1.6. Módulo de Configuración de Dispositivo.

Módulo que se emplea en la configuración del tipo de procesamiento de vértices de la geometría, utilizada en el procedimiento Init3D.

Sub rendertrabajo()

Set v_dispd3d8 = v_md3d8.CreateDevice(0, rendertipo, Principal.Picture1.hWnd, D3DCREATE_HARDWARE_VERTEXPROCESSING, v_presentparad3d8)

End Sub

3.2.2.2. Funciones de Luces.

Inicialización de las cuatro luces (superior, inferior, posterior, frontal), cada una de ellas con valores de color, intensidad, posición, rango y activación.

Private Function SetupLights() As Boolean

*** MAnejo de errores ***

On Error GoTo BailOut:

*** Seteo de reflejo de luz ***

Dim Mtrl As D3DMATERIAL8, Col As D3DCOLORVALUE

Col.a = 0.2: Col.r = 1: Col.g = 1: Col.b = 1

Mtrl.Ambient = Col : Mtrl.diffuse = Col

v_dispd3d8.SetMaterial Mtrl

*** Luz inferior ***

Lights(0).Type = D3DLIGHT_POINT

Lights(0).Position = MakeVector(boundingvector.X, boundingvector.Y - 200, boundingvector.Z)

Lights(0).Range = 500: Lights(0).Attenuation1 = 0.001

Lights(0).diffuse.r = r: Lights(0).diffuse.g = g

Lights(0).diffuse.b = b

v_dispd3d8.SetLight 0, Lights(0)

*** Luz superior ***

Lights(1).Type = D3DLIGHT_POINT

Lights(1).Position = MakeVector(boundingvector.X, boundingvector.Y + 200, boundingvector.Z)

Lights(1).Range = 50000: Lights(1).Attenuation1 = 0.001

Lights(1).diffuse.r = r1: Lights(1).diffuse.g = g1

Lights(1).diffuse.b = b1

v_dispd3d8.SetLight 1, Lights(1)

*** Luz frontal ***

Lights(2).Type = D3DLIGHT_POINT

Lights(2).Position = MakeVector(boundingvector.X, boundingvector.Y, boundingvector.Z + 200)

Lights(2).Range = 500: Lights(2).Attenuation1 = 0.001

Lights(2).diffuse.r = r2: Lights(2).diffuse.g = g2

Lights(2).diffuse.b = b2

v_dispd3d8.SetLight 2, Lights(2)

*** Luz posterior ***

Lights(3).Type = D3DLIGHT_POINT

Lights(3).Position = MakeVector(boundingvector.X, boundingvector.Y + 200, boundingvector.Z - 200)

Lights(3).Range = 50000: Lights(3).Attenuation1 = 0.001

Lights(3).diffuse.r = r3: Lights(3).diffuse.g = g3

Lights(3).diffuse.b = b3

v_dispd3d8.SetLight 3, Lights(3)

*** Manejo de error ***

SetupLights = True

Exit Function

BailOut:

Debug.Print "Error en el SetupLights"

SetupLights = False

End Function

3.2.2.3. Funciones de Renderizado.

Funciones que sirven para dibujar la geometría en pantalla interpretando las posiciones de vértices, parámetros de luces y cámara, tanto del buffer frontal como del buffer trasero.

3.2.2.3.1. Ciclo Timer.

Ciclo principal de dibujado en pantalla, controla la frecuencia de dibujado, y comprueba el movimiento de la cámara dentro del mundo mediante el teclado.

```
Private Sub Timer1_Timer()
** Llamada a la función de limpiado y dibujado **
render
** Llamada al movimiento **
mover_matrices
** Número de frames por segundo **
 If GetTickCount - ultimofps >= 1000 Then
    ultimofps = GetTickCount:     frameps = dibujoframe
   dibujoframe = 0
  Me.Caption = "PUCESA - Frames por Segundo  {" & frameps & "fps}"
** Control de velocidad de movimiento **
Select Case frameps
    Case 0 To 10
        TURN_SPEED = g_90d / 4: MOVE_SPEED = 15
    Case 11 To 30
         TURN_SPEED = g_90d / 9: MOVE_SPEED = 1.2
    Case 31 To 70
      TURN_SPEED = g_90d / 18: MOVE_SPEED = 0.4
   End Select
End If
```

```
dibujoframe = dibujoframe + 1
DoEvents
End Sub
```

3.2.2.3.2. Render.

Procedimiento que limpia la pantalla y dibuja en ella al igual que en el buffer trasero.

```
Sub render()
** Limpiado de pantalla o del Z-buffer **
Dim i As Single
v_dispd3d8.Clear 0, ByVal 0, D3DCLEAR_TARGET Or D3DCLEAR_ZBUFFER, colorfondo, 1#, 0
** Dibujado **
v_dispd3d8.BeginScene
For i = 0 To caras 'nMaterials - 1
        v_dispd3d8.SetMaterial MeshMaterials(i)
        v_dispd3d8.SetTexture 0, MeshTextures(i)
      Mesh.DrawSubset i
   Next i
v_dispd3d8.EndScene
** Presenta en pantalla lo dibujado **
v_dispd3d8.Present ByVal 0, ByVal 0, Picture1.hWnd, ByVal 0
End Sub
```

3.2.2.4. Funciones de movimiento.

En este apartado se describe el código que permite la navegación por el mundo virtual, mediante operaciones con matrices como se explicó anteriormente.

3.2.2.4.1. Movimiento de Cámara.

Movimiento de la cámara con DirectInput (teclado), simulando el movimiento del usuario dentro del mundo, por medio de operaciones de traslación y rotación de matrices.

Sub mover_matrices()

```
Dim matRotation As D3DMATRIX: Dim matPitch As D3DMATRIX
Dim matLook As D3DMATRIX: Dim matPos As D3DMATRIX
Dim AngleConv As Single
** Obtener el estado del teclado **
v_dispositivo_di.GetDeviceStateKeyboard KeyState
** Movimiento **
If KeyState.Key(DIK_RIGHT) <> 0 Then  'derecha
   Angle = Angle - TURN_SPEED
   If Angle < 0 Then
      Angle = g_360d - (-Angle)
   End If
End If
If KeyState.Key(DIK_LEFT) <> 0 Then 'izquierda
   Angle = Angle + TURN_SPEED
   If Angle > g_360d Then
        Angle = 0 + (Angle - g_360d)
   End If
End If
```

```
AngleConv = g_360d – Angle
If KeyState.Key(DIK_UP) <> 0 Then 'adelante
        camx = camx + (Sin(AngleConv) * MOVE_SPEED)
        camz = camz + (Cos(AngleConv) * MOVE_SPEED)
End If
If KeyState.Key(DIK_DOWN) <> 0 Then 'atras
        camx = camx - (Sin(AngleConv) * MOVE_SPEED)
        camz = camz - (Cos(AngleConv) * MOVE_SPEED)
End If
** Gira cámara **
If KeyState.Key(DIK_A) <> 0 Then 'camara arriba
   pitch = pitch + PITCH_SPEED
End If
If KeyState.Key(DIK_Z) <> 0 Then 'camara abajo
   pitch = pitch - PITCH_SPEED
End If
** Mover Cámara **
If KeyState.Key(DIK_NUMPAD8) <> 0 Then 'cámara arriba
    camy = camy + 2
End If
  If KeyState.Key(DIK_NUMPAD5) <> 0 Then 'cámara abajo
    camy = camy - 2
End If
SetupLights
** Operaciones de matrices **
D3DXMatrixIdentity matView: D3DXMatrixIdentity matPos
D3DXMatrixIdentity matRotation: D3DXMatrixIdentity matLook
D3DXMatrixRotationY matRotation, Angle
```

```
D3DXMatrixRotationX matPitch, pitch
D3DXMatrixMultiply matLook, matRotation, matPitch
D3DXMatrixTranslation matPos, -camx, -camy, -camz
D3DXMatrixMultiply matView, matPos, matLook
v_dispd3d8.SetTransform D3DTS_VIEW, matView
v_dispd3d8.GetTransform D3DTS_VIEW, matView
DoEvents
D3DXMatrixPerspectiveFovLH matProj, pi / 4, 1, 0.1, 1000
v_dispd3d8.SetTransform D3DTS_PROJECTION, matProj
```

End Sub

3.2.2.4.2. Movimiento del Objeto.

Movimiento del objeto mediante el mouse, operaciones de rotación de matrices del mundo.

Sub mover()

```
** Rota el m undo **
If rotateangle >= 360 Then rotateangle = rotateangle - 360
** Operaciones con matrices **
D3DXMatrixIdentity matWorld: D3DXMatrixIdentity MATTEMP
D3DXMatrixRotationY MATTEMP, rotateangle * (pi / 180)
D3DXMatrixMultiply matWorld, matWorld, MATTEMP
v_dispd3d8.SetTransform D3DTS_WORLD, matWorld
```

End Sub

3.2.2.5. Limpiado.

Procedimiento de limpieza de dispositivos y cerrado de aplicación.

Sub limpiado()

*** Limpieza de dispositivo de video ***

Set v_dispd3d8 = Nothing: Set v_md3d8 = Nothing

*** Limpieza de Direct Input ***

v_dispositivo_di.Unacquire: Set v_dispositivo_di = Nothing

Set v_di = Nothing

*** Limpieza de mallas ***

Set Mesh = Nothing: Set mesh1 = Nothing

End

End Sub

3.2.3. ELEMENTOS Y DISPOSITIVOS DE APLICACIÓN GENERADA EN VISUAL BASIC 6.0 Y DIRECTX 8.0; APLICACIÓN INMERSIVA.

3.2.3.1. I-Glasses.

Las I-Glasses conocidas también como Shutter-glasses son usadas en conjunto con monitores de tubo de rayos catódicos (monitores estándar de computador RCT), estos dispositivos no sirven con monitores de cristal líquido (LCD).

Las dos imágenes que se necesitan para formar la visión estereoscópica, son mostradas en el monitor por turnos. Esto es, por una fracción de segundo la imagen dedicada al ojo izquierdo es mostrada en el monitor, después se muestra la imagen del ojo derecho por el mismo intervalo de tiempo. La función entonces de las I-Glasses es impedir que el ojo derecho vea la imagen del ojo izquierdo y viceversa.

Para lograr esto, la luz es bloqueada por un LCD "Shutter". Hay dos formas de hacer esto: La primera es colocar un Shutter en el monitor y observarlo a través de unas gafas polarizadas. La segunda es colocar el Shutter en las gafas y polarizarlas directamente a ellas, esto es lo que se conoce como Shutter-Glasses, (ver Fig. 3.15.).

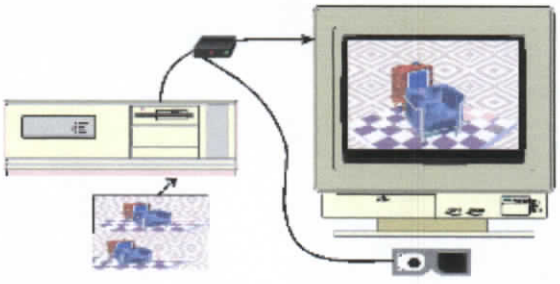


Fig. 3.15.

LAS GAFAS A USARSE SON: "EDIMENSIONAL 3D WIRED GLASSES"

3.2.3.2. eDimensional 3D Wired Glasses.

3.2.3.2.1. Requerimientos Técnicos Mínimos.

- Sistema Operativo Windows 98/ME/2000/XP.
- Memoria RAM 64 Mb.
- Procesador Intel o AMD de 400Mhz.
- Especio en Disco Duro 10 Mb.
- Monitor CRT (No Laptop o LCD).

Está diseñada para tarjetas de video NVIDIA y ATI, incluye sus respectivos controladores, en tarjetas de video de otras marcas se ocupa el controlador ATI.

La figura 3.16. muestra las i-glasses con sus partes.

Fig.3.16.

3.2.3.2.2. Configuración.

3.2.3.2.2.1. Hardware.

- Apagar la computadora y desconectar el cable del monitor.
- Conectar el sincronizador de video a la tarjeta de video.
- Conectar el cable del monitor en el sincronizador de video.
- Conectar las gafas al sincronizador de video.

Nota: En caso de no recibir señal el monitor se necesita un cable de alimentación que es distribuido en los Estados Unidos por la compañía eDimensional.

3.2.3.2.2.2. Software.

Tarjetas NVIDIA.

- Instalar el último controlador de video versión 45.23.
- Instalar el controlador 3D 45.23 (los dos incluidos en el CD).
- Resetear la computadora.

- Abrir el Panel de Control, doble clic en Pantalla, Opciones Avanzadas, clic en la barra GeForce, buscar las nuevas propiedades al lado izquierdo, clic en Stereo Properties, y buscar Enabled, aplicar y OK.

Tarjetas ATI y otras Marcas.

- Instalar controladores eDimensional para ATI en el CD.
- Abrir el directorio de eDimensional en Archivos de Programa, hacer doble clic en EDController.exe, activando la aplicación en la barra del reloj.
- Instalar Power Strip para el refresco del monitor.

La figura 3.17. muestra los íconos que se añaden a la barra de tareas.

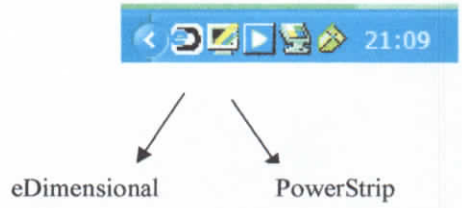


Fig. 3.17.

3.2.3.2.3. Uso de las Teclas de Activación.

- **F5** Activa el efecto de las Gafas.
- **F4** Desactiva el efecto de las Gafas.
- **F7** Une las dos Imágenes en sentido Horizontal.
- **Shift+F7** Desune las dos Imágenes en sentido Horizontal.
- **F6** Incrementa el Efecto (profundidad).
- **Shift+F6** Decrementa el Efecto (profundidad).
- **F8** Une las dos Imágenes en sentido Vertical.

- **Shift+F8** Desune las dos Imágenes en sentido Vertical.

3.2.3.2.4. Interfaz del Software de las Gafas.

3.2.3.2.4.1. Propiedades.

Para configurar las gafas se debe hacer clic derecho en el icono de eDimensional para desplegar el menú (ver Fig. 3.18.).

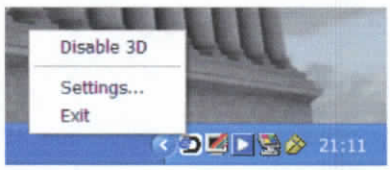


Fig. 3.18.

- Disable 3D. Permite activar o desactivar el software de las gafas.

- Settings. Despliega la ventana de propiedades (ver Fig. 3.19.).

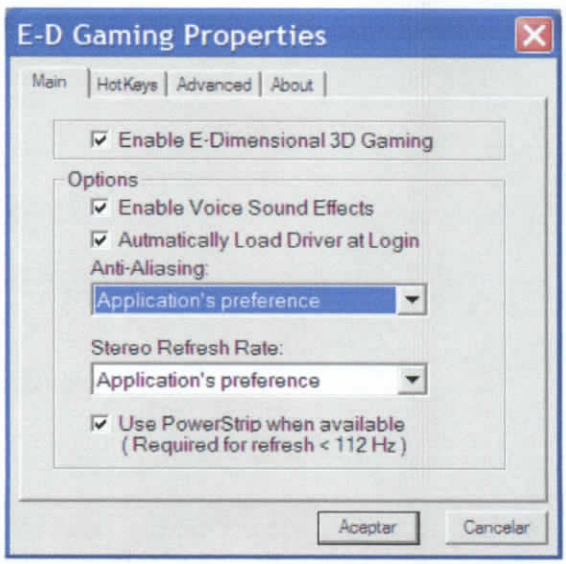


Fig. 3.19.

Nota: Para configurar diríjase al manual de usuario provisto en el CD.

3.2.3.3. Modificaciones a la Aplicación Desktop para Correrla en Fullscreen.

Para que la aplicación pueda funcionar con las gafas debe correr en pantalla completa (Fullscreen), para lo cual hay que hacer modificaciones en el código del procedimiento Init3D. A continuación se listan los cambios:

***** Se aumentan estas líneas*****

*** Cambiar el ancho y el alto de pantalla ***

dmode.Width = 1024: dmode.Height = 768: dmode.Format = D3DFMT_R5G6B5

v_presentparad3d8.SwapEffect = D3DSWAPEFFECT_COPY_VSYNC

v_presentparad3d8.BackBufferCount = 1

v_presentparad3d8.BackBufferFormat = dmode.Format

*** Cambiar el ancho y el alto del back buffer ***

v_presentparad3d8.BackBufferHeight = 768

v_presentparad3d8.BackBufferWidth = 1024

v_presentparad3d8.hDeviceWindow = Principal.hWnd

v_md3d8.GetAdapterDisplayMode D3DADAPTER_DEFAULT, dmode

*** El valor 0 para fullscreen***

v_presentparad3d8.Windowed = 0

***** Se substituyen estas líneas por las anteriores *****

v_presentparad3d8.BackBufferFormat = dmode.Format

*** El valor 1 para ventana***

v_presentparad3d8.Windowed = 1

v_presentparad3d8.SwapEffect = D3DSWAPEFFECT_FLIP

v_presentparad3d8.hDeviceWindow = Picture1.hWnd

v_presentparad3d8.EnableAutoDepthStencil = 1

v_presentparad3d8.AutoDepthStencilFormat = D3DFMT_D16

NOTA: Para referencia de las funciones usadas en el código revisar la ayuda de Microsoft DirectX SDK.

CAPÍTULO IV

4. VALIDACIÓN Y VERIFICACIÓN DE LOS RESULTADOS.

4.1. VALIDACIÓN.

Para consultar las hojas que certifican la validación de las aplicaciones ver los Anexos 1 y 2.

4.2. CONCLUSIONES.

- Es factible la utilización de Visual Basic 6.0 en el desarrollo óptimo de aplicaciones 3D, Desktop e Inmersivas.
- Se logró el desarrollo de una aplicación Inmersiva con bajo costo.
- La utilización de la aplicación en proyectos arquitectónicos y construcción de objetos en general permite tener una perspectiva mejorada del producto final.
- Se reducen posibles suposiciones u omisiones de constructores y propietarios de construcciones arquitectónicas.
- La utilización de texturas en el modelado mejora los conceptos de Realidad Virtual.
- El principal inconveniente en el desarrollo de aplicaciones 3D recae en la falta de documentación adecuada.
- La programación de aplicaciones 3D, es una de las ramas más complicadas de la programación informática.
- El desarrollo de la aplicación despertó el interés en estudiantes que cursan actualmente la carrera.
- Las I-glasses proporcionan una Inmersión de bajo costo y calidad aceptable.
- La aplicación ayudó a visualizar un proyecto arquitectónico a futuros clientes.
- Nuestro sistema cumple con las características de: inmersión, tridimensionalidad, punto de referencia y navegación; utilizando técnicas de Realidad Virtual de tipo desktop e inmersiva, y se usan dos tipos de mundos: muertos y fantásticos.

4.3. RECOMENDACIONES.

- Se recomienda programar en una tarjeta de video con chip de aceleración gráfica con memoria interna propia y recursos computacionales altos en general.
- Se recomienda tener conocimientos básicos de Geometría Descriptiva.
- Se recomienda para futuras investigaciones incluir texturizados de polígonos.
- Se recomienda la utilización del proyecto de tesis sobre prototipos 3D, antes que sobre planos finales de dibujo arquitectónico 3D.
- Se recomienda a los profesionales de arquitectura diseñar anteproyectos con herramientas 3D para que la inmersión sea óptima.
- Se debe usar preferentemente programas que generen archivos de intercambio estandarizados de dibujo 3D.
- Recomendamos que para futuras investigaciones se busque financiamiento externo con el fin de utilizar dispositivos de alta tecnología que permitan una inmersión más real.
- Recomendamos para futuras investigaciones en la PUCESA utilizar recursos ya existentes y de amplia difusión para obtener mejores resultados y disminución de tiempo de implantación.
- Se recomienda para desarrollar aplicaciones de este tipo tener conocimientos de nivel medio de inglés por la documentación.

BIBLIOGRAFÍA.

- **Libros.**

WINDOWS GAME PROGRAMING WITH VISUAL BASIC AND DIRECTX; Freeze Wayne; Tomo único; Editorial QUE; Indianapolis, 2002; Segunda Edición.

THE VIRTUAL REALITY; Gradecki Joe; Tomo único, Editorial PCVR Magazine; 1994; Segunda Edición.

3D STUDIO MAX 4; Matossian Michele, Tomo Unico; Editorial Prentice may; 2000; Primera Edición en Español.

- **Internet.**

Programación:

- www.vbexplorer.com/directx4vb/

  Página de programación de DirectX con Visual Basic.
- http://www.opengl.org/

  Página oficial de programación de OpenGL.
- http://www.microsoft.com/windows/directx/

  Página oficial de DirectX.
- http://msdn.microsoft.com/vbasic/

  Página oficial de Visual Basic.
- http://www.directxfaq.com/kbase/D3D/dxvsopengl.htm

  Foro de discusión sobre librerías gráficas.
- http://www.ladei.com.ar/D3D/Intro.html

  Página de traducción de la ayuda de DirectX.

Realidad Virtual

- http://www.vrealities.com/hmd.html

  Página de dispositivos de Realidad Virtual.
- http://www.ia.hiof.no/~michaell/home/vr/vrhiof98/index.html

  Página de información de Realidad Virtual.
- http://worldspace.berlios.de/fase1/

  Página de técnicas y conceptos de Realidad Virtual.
- http://www.e-d.com

  Página oficial de las gafas eDimensional.

Diseño 3D

- http://www.discreet.com

  Página oficial de 3D Studio Max y su Fichero 3DS.
- http://www.3dcafe.com

  Sitio web dedicado al 3D contiene artículos, modelos, texturas.

RESULTADOS
PRIMER CONGRESO NACIONAL DE INGENIERÌA DE SISTEMAS DE LA PUCE

No.	TITULO PONENCIA	SEDE	CALIFICACIONES	TOTAL
1	Creaciòn de una comunidad virtual aplicada a la educaciòn mediante nuevas tecnologìas	Ambato	8,1 - 5,9 - 8,7	22,7
2	Realidad virtual desktop utilizando visual basic y las librerìas Direc 8	Ambato	9,1 - 7,0 - 9,2	25,3
3	Procesamiento automàtico mediante correo electrònico	Ibarra	8,9 - 9,7 - 9,9	28,5
4	Intrenet para gestiòn acadèmica y administrativa de la PUCESE	Esmeraldas	5,5 - 3,1 - 9,3	17,9
5	Programaciòn por aspectos Aspect J	Quito	8,0 - 8,6 - 10	26,6

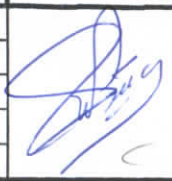
JURADO CALIFICADOR: ESMERALDAS: KLEVER VERA
IBARRA: JUAN CARLOS ARMAS
AMBATO: PATRICIO MEDINA

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO
DIRECCION
ESCUELA DE INGENIERIA
DE SISTEMAS

NOMBRE	CEDULA	OCUPACION	INSTITUCION	COMENTARIO	FIRMA
OLINA BENITEZ	0801688788-0	ESTUDIANTE	PUCESE DE ESMERALDAS	ESTUVO bonita e interesante Fue la mejor DE TODAS, YO LES RECOMENDARIA Q' TRABAJEN CON VIDEO JUEGOS FELICITACIONES SIGAN ADELANT	
RIO COSNEROS	1802/1022-9	ESTUDIANTE	UNIANDES	FELICITACIONES!, ME PARECE UN EXCELENTE TRABAJO, CON EL TIEMPO HAY SEGUIR MEJORANDO ESTE PROGRAMA PARA SU PROVECHO. ESPECIALMENTE MEJORAR IMAGENES TIPO CANASTA.	
Rivadeneira	0502273672	Estudiante	PUCESA	Hagan juegos por la red	
sé Castillo	1104147444-0	Estudiante	Pucesa	Sinceramente el proyecto es muy pequeño, tienen que mejorarlo. Hagan un proyecto que sea de calidad y entretenimiento.	
STINA PINTO R.	180360048-3	ESTUDIANTE	PUCESA	SU trabajo es muy bueno y le veo muchas utilidades, les recomiendo aumentarle más opciones	CRISTINA PINTO R
ego Suárez	180366048-7	Estudiante	PUCESA	Les recomendaría que trabajen en juegos de Videos y realizar juegos por la Red se les felicita sigan adelante	
an Suárez y Klin Gavilanes	180366729-2 180359228-4	Estudiantes	PUCESA	Primeramente felicitarles por su proyecto q' es muy importante e interesante y de ayuda ya q' con el tiempo lo pueden mejorar con más calidad y eficacia.	

NOMBRE	CEDULA	OCUPACIÓN	INSTITUCIÓN	COMENTARIO	FIRMA
er Villacrés V.	050186322-9	ESTUDIANTE	PUCESA	El proyecto esta muy interesante y sería bueno q' lo implemente con cosco virtual o algun aparato de realidad Virtual ¡SIGAN ADELANTE!	
ilson Ojeda Morales	1802881141-5	Soporte Técnico Empresarial	Dishard	El proyecto es muy bueno y sería fantastico q' culminen con todos los dispositivos q' crean convenientes para lo que es realidad Virtual	
é Yovanny Carrillo	1103764500-0	Ingeniería de sistemas y control de riesgos	Sistek. U.E. El "sauce"	Bueno en realidad este proyecto tiene finalidad, es fiable, pero pensemos que existen sistemas, incluso programas ya establecidos que podemos usarlos.	
ana Mantilla Ramirez	1803598208	Estudiante	UTA	La propuesta de su proyecto es muy bueno y lo q' tienen desarrollado es exelente. Seria bueno q' lo implementen con dispositivos externos.	
Eduardo Ricaurte Carranza	1809240799-3	Estudiante	UTA	Mi sugerencia es q' sigan investigando Mas Sobre el desarrollo de figuras 3D con Visual Basic 6.0.	
n Andrade R.	1715626303	Estudiante	PUCE	El proyecto está bien sustentado y alienta a seguir en la investigación y desarrollo en 3D.	
ulino Sánchez	1803365129	Estudiante	PUCESA	El proyecto es muy interesante ya que en nuestra Universidad no un proyecto nuevo en base a 3D.	

NOMBRE	CEDULA	OCUPACIÓN	INSTITUCIÓN	COMENTARIO	FIRMA
wiN Viteri	0802805507	Estudiante	PUCESE	FUE EL MEJOR DE TODOS FELICITACIONES	
BONES	080285151-7	Estudiante	PUCESE	Excelente Me gustaria que colaguen todos tutorial en la web de la Universidad y la descarga sea solo con # de Matricula	
IO ROMERO	1303983990-2	ESTUDIANTE	PUCESE	ME GUSTO MUCHO Y CREO Q' FUE EL MEJOR TEMA SIGAN ASI	
SSICA ORTIZ C.	080294697-0	ESTUDIANTE	PUCESE	Me parecio una muy buena exposición espero que sigan adelante investigando que desarrollen sus conocimientos al máximo para bien del país	
celo Gallardo	180297601-7	Estudiante	PUCESA	Muy Bueno, tienen lo mas importante que es el inicio de un proyecto el cual les dara mucho exito Sigan perfeccionando y me gustaria q' el movimiento de las imagenes sea automatica asi como el	
o Guaico	180245742-7	Estudiante	PUCESA	Cambio de imagenes y su visualizacion en cuanto al acercamiento y Caumento y disminucion de tamaño sea visualizada como si fuera cinema pero sin Casco Virtual.	
				La exposición tiene muy buenas perspectivas y me parecio un proyecto que tiene un Campo muy amplio de acciones debe Seguir adelante	

NOMBRE	CEDULA	OCUPACION	INSTITUCION	COMENTARIO	FIRMA
DIGBERTO SANCHEZ	180318471-0	ING. SIST.	PUCESA	BIEN	
Paúl Illingworth	180266 374-3	Estudiante	PUCES A	Súper bien un excelente proyecto.	
ALO AGUIRRE	180340431-6	ESTUDIANTE	PUCE-SA	Muy buena preparación e investigación hasta el punto donde están. Sigan trabajando hasta que logren sus propósitos en este proyecto.	
ergüer Proaño	180333270-7	Estudiante	PUCESA	Felicitaciones... pocos se introducen en la magia de la realidad virtual. Espero que su trabajo sea el inicio de un constante crecimiento de aplicaciones 3D realizadas en nuestro país	
ictor Zurita	180290557-8	Estudiante	PUCESA	Felicitaciones me parece un excelente proyecto	
RODRIGO MARTINEZ M	180027367	INGENIERO	PUCESA	Felicitaciones sigan adelante	

NOMBRE	CEDULA	OCUPACION	INSTITUCION	COMENTARIO	FIRMA
[illegible] Medinas	180233327-6	Docente Sistemas	Puce	- Quisiera la aplicación a una Realidad Cercana - Interesante la Conexión Diseño x - Se Debería Cerrar Ventanas Previas	
[illegible] Jodan	180715266-8	Docente Sist.	PUCESD	- Muy buen trabajo ya que para lograr modelos o animación se requiere tiempo, paciencia y buen gusto, con su trabajo se nota que han conjugado todo esto	
[illegible] Quiñonez	080190169-5	Docente Sistemas	PUCESE	- Muy bueno pintura en el campo del diseño y modelación de casas porque se pueden ver antes de construir o pintar dinero	
[illegible] Altamirano	180284325-8	Profesora	PUCESD	Es un excelente trabajo ya que ustedes empezaron todo de cero de todo corazón que Dios los Bendiga y los felicito son buenos. JOU XVAX	

NOMBRE	CEDULA	OCUPACION	INSTITUCION	COMENTARIO	FIRMA
Juan Fernando Acosta	100244751-2	Tecnico Sistemas de Hardware	Processum & Cia.	El sistema es bueno pero esta orientado a objetos almacenados mas no captados, los objetos deberían ser capturados por algun medio fisico externo.	
Wilian León	100258227-6	TECNICO EN REDES ELECTRICOS Y SOPORTE EN REDES LAN	SINFOTECNIA	QUE SIMPLEMENTE DEBERIAN ENFOCARCE MAS EN LO QUE SE REFIERE A LA INFORMATICA Y NO A LO DE DISEÑO EN AUTOCAD O OTROS PROGRAMAS	
Carlos Gudiño	100293154-7	ESTUDIANTE	PUCE-I	Me parecio un proyecto muy bueno, pero si debbian investigar un poco mas y cumplir los objetivos que se han propuesto Y FELICITACIONES	
Leber Vera	1306246065	~~Prof~~. SISTEMAS	POCESE	PROYECTO MUY INTERESANTE Y CON MUCHAS PERSPECTIVAS EN EL MEDIO ECUATORIANO. COMPLEMENTENLO CON ANIMACIONES INTERACTIVAS Y TENDRA MAYORES REFERENCIAS	

ANEXOS 2: VALIDACIÓN DE APLICACIÓN INMERSIVA.

Ambato agosto 23, 2004
PRO-115

Ingeniero
Telmo Viteri
DIRECTOR ESCUELA DE SISTEMAS
Pontificia Universidad Católica de Ambato
Presente

Ingeniero Viteri:

Por la presente me permito indicar que luego de haber examinado el trabajo realizado por los señores Mauricio Alejandro Salazar Morales y Enrique Xavier Garcés Freire, alumnos de la Escuela de Sistemas referente a Realidad Virtual Inmersiva aplicada a la Arquitectura, considero que es aceptable y útil este software, por lo que se puede dar paso a la defensa de este proyecto.

Por la gentil atención, reciba mi agradecimiento.

Atentamente,

P. Dr. César González Loor
PRORRECTOR PUCE SEDE AMBATO

cc.: archivo

PONTIFI
UNIVERS
CATOLIC
DEL ECU
SEDE A
PRO-RE
Av. Manuelita Sáen
Sector El Tropezón
Apartado Postal No
Telf: 593 3 411 868
Fax: 593 3 411 868
pucesedeambato@h
Ambato - Ecuador
www.pucesa.edu

CARLOS RIBADENEIRA A.
ARQUITECTO
LP. 213 T. RM. 1140

Ambato, Agosto 3 del 2004

CERTIFICADO

A petición verbal de los interesados tengo a bien certificar que he revisado el programa elaborado por los señores: Enrique Garcés y Mauricio Salazar, y considero que se puede dar utilidad en el campo de la Arquitectura para facilitar la expresión de los espacios en 3 dimensiones, y así el cliente pueda visualizar con mayor claridad el proyecto arquitectónico que desea obtener.

Es lo que puedo certificar en honor a la verdad, y autorizo a los interesados hacer uso del presente como a bien convenga a sus interesados.

Att.

Carlos Ribadeneira A.
Arquitecto
LP: 213 T.

Carlos Ribadeneira A.
ARQUITECTO
LP. 213 T

Ambato, 24 de Agosto de 2004

Ing. Telmo Viteri
DIRECTOR DE LA ESCUELA DE INGENIERIA DE SISTEMAS
PUCESA

Por medio de la presente tengo a bien entregarle el informe de la Validación de la Disertación de Grado titulada "APLICACIÓN DE REALIDAD VIRTUAL INMERSIVA ORIENTADA A LA ARQUITECTURA", realizada por los estudiantes, **Enrique Xavier Garcés Freire** y **Mauricio Alejandro Salazar Morales.**

Una vez que he revisado la aplicación realizada por los estudiantes debo indicar que cumple con sus objetivos planteados, al construir una herramienta que visualice mundos virtuales inmersivos aplicados a la arquitectura, por lo cual certifico que cumple con el proceso de validación.

Por la atención que se de a la presente, le anticipo mis agradecimientos.

Atentamente

Ing. Janio Jadán
PROFESOR

ANEXOS 3: MANUAL DE USUARIO.

INTRODUCCIÓN.

La aplicación está desarrollada con Visual Basic 6.0 y DirectX 8.0, el software está orientado a la arquitectura, para poder visualizar y navegar por modelos suministrados por el usuario tanto en formato 3DS como en X.

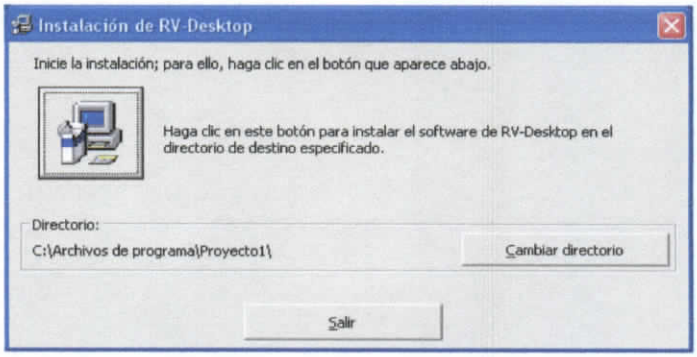
GUÍA DE INSTALACIÓN.

Para usuarios que posean I-Glasses se debe correr el instalador de la carpeta Fullscreen del CD. Para el resto de usuarios se debe correr el instalador de la carpeta Desktop que esta dentro del CD.

PASOS DE INSTALACIÓN.

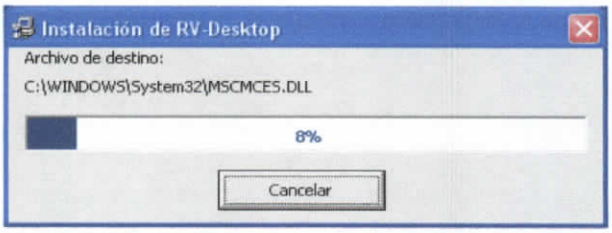
- Correr el Setup.exe
- Presionamos Aceptar en la ventana de Bienvenidos.

- Escoger el directorio de instalación o dejar el que sale por defecto y presionar el botón en la esquina superior derecha para continuar.



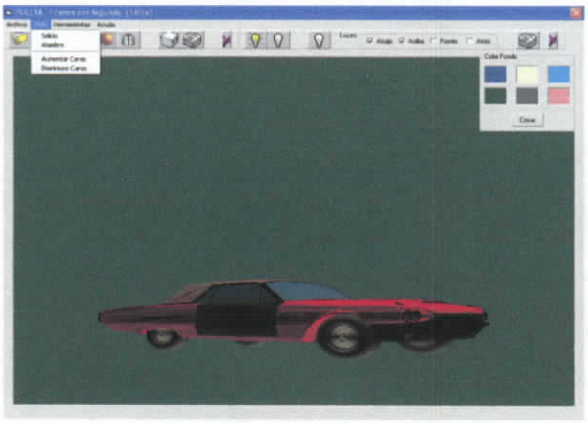
- Escoger el nombre del grupo de programas en el menú inicio o dejar el que sale por defecto y presionar el botón continuar.

- Ventana con barra de progreso de la instalación.

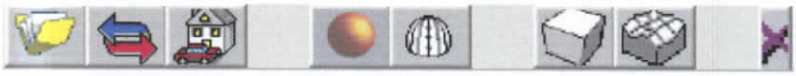


- Para finalizar clic en Aceptar de la ventana de instalación satisfactoria.

GUÍA DE USUARIO APLICACIÓN DESKTOP.



Barra Principal.



A continuación una descripción de cada uno de los botones de la barra.

Botón que carga ficheros X.

Botón que transforma ficheros en formato 3DS a X.

Botón que reinicia a la posición original de la cámara.

Botón para presentar la malla en forma sólida.

Botón para presentar la malla en forma de alambre (wireframe).

Botón para aumentar caras en la malla.

Botón para quitar caras en la malla.

Botón que oculta la barra.

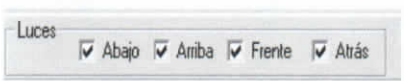
Barra de Luces.

A continuación una descripción de cada uno de los botones de la barra.

Botón que prende todas las luces.

Botón que apaga todas las luces

Botón que muestra la ventana para cambiar el color de las luces.



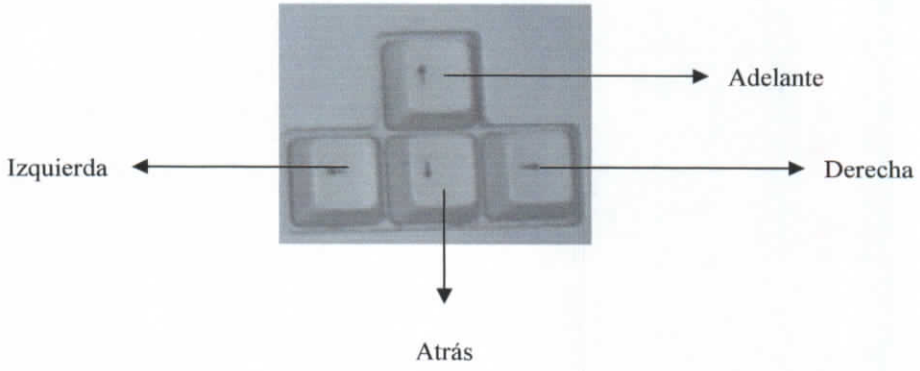
Prende y apaga las luces individualmente.

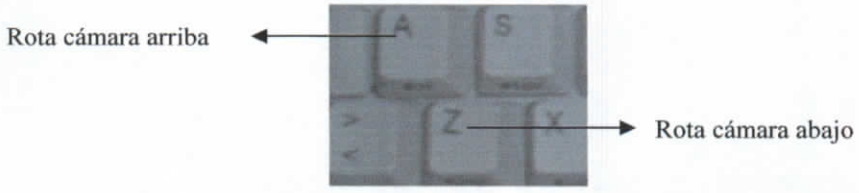
Botón que muestra la ventana para cambiar el color del fondo.

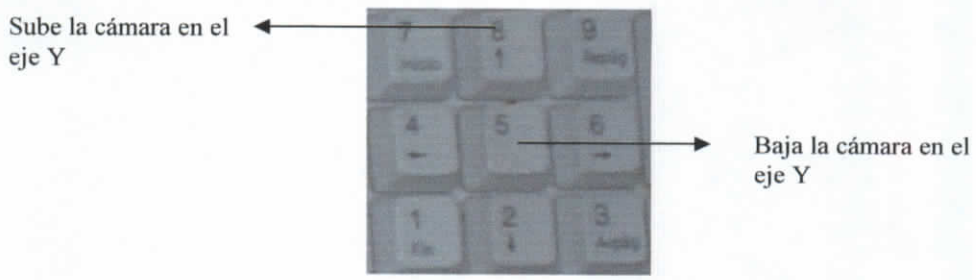
Botón que oculta la barra.

DESCRIPCIÓN DEL MOVIMIENTO EN LA APLICACIÓN.

Teclado







Mouse

Algunos de los movimientos que se hacen con el teclado se los puede hacer con los botones del mouse:

- Botón Izquierdo: Gira el objeto en el mundo en el eje Y.
- Botón Derecho: Simula el movimiento de las teclas 8 y 5 del teclado numérico.
- Botón Central o del Scroll (presionado): Simula el movimiento de las teclas A y Z.