

Lej



UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

CREACION DE UN SISTEMA DE REALIDAD VIRTUAL

T E S I S
QUE PRESENTA
CLAUDIA CANCHE RODRIGUEZ
PARA OBTENER EL TITULO DE
INGENIERA EN COMPUTACION

DIRECTOR DE TESIS: ING. GERMAN SANTOS JAIMES



MEXICO, D. F.

1999

TESIS CON FALLA DE ORIGEN

214153



Universidad Nacional
Autónoma de México

Dirección General de Bibliotecas de la UNAM

Biblioteca Central



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

A Dios, Uno y Trino, de Quien he recibido todo.

AGRADECIMIENTO

A todas aquellas personas
que hicieron posible la realización
del presente trabajo.

Mis papás

Lic. Alejandro Rodríguez Rodríguez
Leonardo Fabio Garay Medina

Ing. Germán Santos Jaimes

Dr. Jesús Savage Carmona
Ing. Mauricio Jacobo Romero
Ing. Esteban
Ing. Alberto González Guizar

Mis hermanos

SUMARIO

INTRODUCCIÓN	1
CAPÍTULO 1	
Definición de Conceptos	3
CAPÍTULO 2	
HITL South (Human Interface Laboratory, sucursal South)	34
CAPÍTULO 3	
Creación de un Sistema de Realidad Virtual	47
CONCLUSIONES GENERALES	81
ANEXO A	
Resultados de añadir Propiedades a los Objetos	86
ANEXO B	
Archivo VDI final y su representación en el Control Panel	90
BIBLIOGRAFÍA	117
ÍNDICE	120

INTRODUCCIÓN

El presente trabajo representa la documentación del proyecto de tesis realizado en el HITL South o Laboratorio de Interfaces Inteligentes de la Facultad de Ingeniería de la Universidad Nacional Autónoma de México, que consiste en la creación del Sistema de Realidad Virtual que representa al Laboratorio mencionado, con el fin de que, en un futuro sea utilizado como Espacio Virtual de pruebas para los proyectos ahí realizados.

Este trabajo está formado por tres capítulos y dos anexos. En el primer capítulo: Definición de Conceptos, nos dedicamos a definir los términos básicos para la comprensión del tema Realidad Virtual, puntualizamos las características de los Sistemas de Realidad Virtual y hablamos en general de los dispositivos propios de Realidad Virtual. Esto, con el objetivo de sintonizar a todo aquel que sea principiante en el tema y que no se pierda en el desarrollo del proyecto.

En el segundo capítulo: HITL South, comenzamos enunciando el proyecto a realizar y nos dedicamos a enlistar el hardware y software con que contamos en el Laboratorio para la realización del proyecto, para finalmente elegir lo que nos conviene emplear para la creación de nuestro Sistema de Realidad Virtual.

En el tercer capítulo profundizamos un poco en el software dVISE con el que convertimos un mundo tridimensional a un mundo virtual y presentamos, paso a paso, el desarrollo necesario para existosamente crear un completo Sistema de Realidad Virtual. Todo el capítulo es acompañado con fotografías de las pantallas donde se agregan las propiedades a cada objeto del Mundo Virtual.

En el anexo A se enlistan los resultados de agregar propiedades a los objetos a través de pantallas, en el archivo VDI. Y en el anexo B se presenta el listado del archivo VDI de todo el Mundo Virtual creado y su representación en el Control Panel de dVISE.

CAPÍTULO 1

DEFINICIÓN DE CONCEPTOS

Parfraseando a Santo Tomás de Aquino, quien al inicio de su opúsculo *De ente et essentia*, hace la siguiente aseveración tomada de Aristóteles: “ un pequeño error al principio es grande al fin ”¹; comenzaremos nuestro trabajo con la enunciación y definición, hasta donde sea posible, de los conceptos propios de Realidad Virtual.

1.1 Realidad Virtual

Existen múltiples y diversas definiciones de Realidad Virtual² (RV), si bien es cierto que es un clon o una metáfora del mundo real; podríamos decir que RV es proporcionarle a una persona la ilusión mas convincente posible de una realidad o de una fantasía.³

Afirmar lo anterior nos remite a aceptar que la RV es utilizada para simular que viajamos a lugares inalcanzables, que hacemos cosas imposibles e inclusive se utiliza para vagar por mundos utópicos.

Lo anterior, sin embargo, no significa que la RV es en su mayoría un asunto de pasatiempos, sino que científicamente hablando, lugares inalcanzables son los demás planetas y el interior de las moléculas, entre otros y cosas imposibles son desenrollar una cadena de DNA, tocar y levantar una piedra de Marte, dialogar con nuestros antepasados, etc.

¹ AQUINO DE, TOMAS, *De ente et essentia*, Proemio A.

² “Virtual. Cerca de ... casi como... algo como...”: HAYWARD, TOM, *Adventures in Virtual Reality*, p.1

³ Cfr. HARRISON, DAVID; JAQUES, MARK, *Experiments in Virtual Reality*, p.2; CHORAFAS, DIMITRIS N.; STEINMANN, HEINRICH, *Realidad Virtual. Aplicaciones prácticas en los Negocios y la Industria*, p.27

1.1.1 Mundo Virtual

Es todo un universo virtual, la unidad más grande que exista en ese momento.⁴

1.1.2 Espacio Virtual

Es una porción del mundo virtual; área específica que pertenece a un mundo virtual. También llamado Entorno Virtual. Tenemos pues, que un conjunto de entornos virtuales constituyen un mundo virtual.⁵

1.2 Sistema de RV

Un sistema⁶ de RV es la combinación de todo el potencial de una sofisticada computadora con gráficos tridimensionales enriquecidos con sensaciones del mundo real a través de dispositivos visuales, auditivos y de otro tipo para sintetizar un entorno ficticio interactivo en el cual el usuario se sienta inmerso.⁷

Tridimensionales, sintetizar, interactivo, inmerso: importante juego de palabras que dan sentido a nuestro siguiente apartado.

1.3 Características de los Sistemas de RV

Para que se cumpla con la definición arriba expuesta, un sistema de RV debe satisfacer determinadas características; pero en cuanto tal, para identificar y apropiadamente llamar un sistema de RV necesitamos que estas características

⁴ Cfr. STAMPE, DAVE; ROEHL, BERNIE; EAGAN, JOHN, Realidad Virtual. Creaciones y Desarrollo, p. 138

⁵ Ibid, p.138

⁶ "Un sistema es un ente formado por un conjunto de entradas, un conjunto de salidas y una relación bien definida entre ambos conjuntos", RODRÍGUEZ R., FRANCISCO J., Dinámica de Sistemas, p.2

⁷ Cfr. LARIJANI, CASEY L., Realidad Virtual, p.11; CHORAFAS, DIMITRIS N.; STEINMANN, HEINRICH, op. cit., p.21 ; DEL PINO GONZALEZ, L. M., Realidad Virtual, p.19

sucedan al mismo tiempo. Ante ello hay quien se atreve a afirmar que al sistema le puede hacer falta una de estas características, pero si le hacen falta más, ya no se puede considerar un sistema de RV.⁸

1.3.1 Capacidad Sintética o Respuesta en Tiempo Real

Las imágenes que se presentan al usuario al ir recorriendo el espacio virtual no son reproducidas, sino sintetizadas. Es decir, todos los objetos que conforman el espacio virtual están almacenados en forma abstracta, de tal forma que cada vista del usuario se genera en el momento. Lo anterior indica que en una base de datos se almacena el color, tamaño, posición y orientación de todos y cada uno de los objetos y la computadora calcula, de acuerdo a la posición en la que el usuario se encuentre, cómo los vería desplegados en pantalla.⁹

1.3.2 Tridimensionalidad

El hombre vive en un mundo tridimensional y todo lo percibe en tres dimensiones, aún lo que se le presenta plasmado en un papel su mente lo transforma en algo tridimensional; no porque un mapa carezca de volumen o de sombreado piensa en dos dimensiones, sino que asume que es una vista superior y volvemos a la percepción en tres dimensiones. En consecuencia, si buscamos una ilusión de la realidad, tenemos que presentar el mundo virtual en tres dimensiones y para lograr esto entran en juego modelos matemáticos que agregan esa tercera

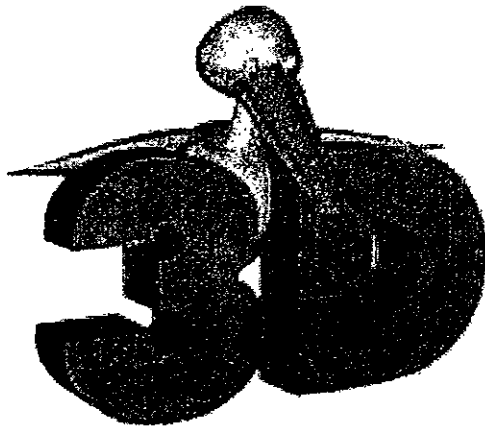


Figura 1. Objeto en tres dimensiones

⁸ Cfr. DEL PINO GONZALEZ, L. M., *op. cit.*, p.33

⁹ *Ibid* p.20-21

dimensión a los objetos desplegados en una pantalla que solo cuenta con dos dimensiones.

1.3.3 Estereoscopia¹⁰

Debido a la posición que guardan los ojos en el cuerpo humano, cada ojo percibe una imagen ligeramente distinta de una misma escena. De esta manera, el hombre percibe la profundidad de su entorno al comparar el par de imágenes que le proporcionan al cerebro sus dos ojos.

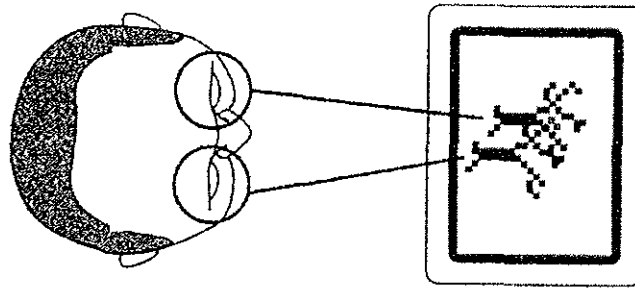


Figura 2. Cada ojo percibe una imagen ligeramente distinta

Los sistemas de RV deben ser capaces de calcular la imagen para cada ojo o en su defecto deben agregar claves de profundidad psicológicas a la imagen generada.

1.3.3.1 Claves de Profundidad

De manera breve podemos decir que existen claves de profundidad físicas y psicológicas.¹¹ Dentro de las claves de profundidad físicas se encuentra la misma estereoscopia, el ángulo de convergencia que define una rotación que es aplicada a la imagen del ojo derecho y a la imagen del izquierdo para establecer el punto

¹⁰ “Estereoscópico. Sistema que provee dos imágenes con información de paralaje, lo que proporciona una ilusión de profundidad o relieve.”: VINCE, JOHN, *Virtual Reality Systems*, p.371

¹¹ Cfr. DEL PINO GONZALEZ, L. M., *op. cit.*, p.41

focal en el espacio tridimensional y el paralaje es la distancia horizontal entre dos puntos cualesquiera de las imágenes que estamos viendo con cada ojo.¹²

Las claves de profundidad psicológicas son la perspectiva lineal, la interposición, los sombreados y las sombras, el nivel de detalle y la difuminación.

La perspectiva lineal define el tamaño de los objetos de acuerdo a la distancia a la que éstos se encuentran; siendo los objetos de mayor tamaño los más cercanos al usuario.¹³

La interposición define cuáles objetos están a la vista del usuario parcial o totalmente; si un objeto esta parcialmente tapado por otro objeto el cerebro humano puede asumir que el uno se encuentra a una distancia mayor que el otro.¹⁴

El sombreado define la apariencia de cada superficie a razón de su capacidad de reflejo, su posición y su orientación con respecto a las fuentes de luz, y a otras superficies. Las sombras dan realismo y añaden sensación de profundidad a una escena; si un objeto A proyecta una sombra sobre la superficie de B, entonces sabemos que el A está entre B y una fuente de luz directa.¹⁵

El nivel de detalle es añadir o quitar detalles de los objetos al incrementar o decrementar la distancia del observador. Se crean múltiples niveles de detalle para determinados rangos de distancia y se intercambian conforme sea conveniente; esto además de producir sensación de profundidad, aumenta el desempeño del sistema.¹⁶

¹² Cfr. Geometry Tools for UNIX Workstations. User Guide, c.2, p.36, GRADECKI, JOE, Realidad Virtual. Construcción de proyectos, p.108

¹³ Cfr. HAYWARD, TOM, op. cit., p.153

¹⁴ Ibid, p.153

¹⁵ Cfr. FOLEY, JAMES D; VAN DAM, ANDREIS; FEINER, STEVEN K.; HUGHES, JOHN F., Computer Graphics. Principles and Practice, p.612, 614

¹⁶ Cfr. Geometry Tools, c.2, p.14, 15

La difuminación consiste en desvanecer los colores de los objetos conforme aumente la distancia del observador.

1.3.4 Interactividad o Manipulación

Interactividad¹⁷ se refiere al grado en que una persona puede tomar decisiones dentro del ambiente; estas decisiones se basan en las reglas y comportamientos del ambiente. Es darle al usuario la posibilidad de tener el control sobre el mundo. La comunicación resulta más efectiva cuando puedes alcanzar, tocar y mover, es decir, manipular los objetos a tu alrededor, se dice que el punto clave no es que los objetos posean apariencia real, sino que es cómo podemos interactuar con los objetos mientras recorremos el espacio virtual. Un objeto puede llegar a ser real en el contexto de nuestra absorción en el entorno. Es por ello que un sistema de RV debe ser capaz de actualizar todos los parámetros del modelo del espacio virtual que sea necesario.¹⁸

En un sistema de RV el usuario debe actuar en Primera Persona¹⁹.

Cuando se promueve un intercambio de factores que afectan al espacio virtual se habla de una interactividad dinámica. Cuando la interactividad no es dinámica se llama Navegación.

¹⁷ "Interactivo. Con capacidad de responder inmediatamente a elecciones y comandos realizados por el usuario.": JACOBSON, LINDA, *Garage Virtual Reality*, p.409

¹⁸ Cfr. <http://www.oz.net/~avi/gloss.htm>, DEL PINO GONZALEZ, L. M., *op. cit.*, p.22; HEIM, MICHAEL, *The Metaphysics of Virtual Reality*, p.110; HARRISON, DAVID; JAQUES, MARK, *op. cit.*, p.8; CHORAFAS, DIMITRIS N. & STEINMANN, HEINRICH, *op. cit.*, p.39

¹⁹ Primera Persona. Literariamente se define como la *voz*, cuando el narrador y el autor son el mismo. Interactivamente hablando el autor no es otro que el mismo usuario quien es el que maneja la experiencia y la narración es la presentación visual, auditiva y de texto. En consecuencia, la Primera Persona generalmente requiere una *vista a través de los ojos*. Cfr. <http://www.oz.net/~avi/gloss.htm>

1.3.5 Navegación²⁰

La navegación es poder moverse como si estuviéramos dentro del mundo virtual, ver todo lo que estaría a nuestro alrededor y explorarlo. Haciendo una analogía con el mundo real, es entrar a un museo, observar todo lo que nos presenta y visitar todas las salas que nos resulten atractivas, pero está prohibido tocar.

1.3.6 Ilusión de Realidad o Simulación de Comportamiento

Las computadoras no son suficientemente poderosas como para crear mundos tan parecidos a la realidad. Pero el hombre posee una enorme imaginación que fácilmente puede hacer realidad algo que tan sólo se parece a la realidad. Basta que cumpla reglas comprensibles por el hombre, es decir, que su comportamiento sea creíble aunque se trate de un mundo de fantasía.²¹

1.3.7 Inmersión

Inmersión es sentirse dentro del mundo virtual y sentir ser parte de dicho mundo. También se define como la ilusión de estar sumergido en el mundo virtual. Podríamos decir que es producir un estado de presencia o un sinónimo de *presencia*^{22, 23}.

Como sabemos, el cuerpo humano se guía por los cambios en su entorno percibidos a través de sus sentidos. En consecuencia, para lograr esa sensación de estar ahí, es necesario estimular nuestros sentidos hasta conseguir un efectivo

²⁰ Cfr. LARIJANI, CASEY L., *op. cit.*, p.XIII; DEL PINO GONZALEZ, L. M., *op. cit.*, p.23

²¹ Cfr. HARRISON, DAVID; JAQUES, MARK, *op. cit.*, p.3; DEL PINO GONZALEZ, L. M., *op. cit.*, p.30

²² "Presencia. La sensación de realismo creada por una experiencia virtual.": VINCE, JOHN, *op. cit.*, p.368

²³ Cfr. HEIM, MICHAEL, *op. cit.*, p.110; HARRISON, DAVID; JAQUES, MARK, *op. cit.*, p.4

ambiente para el espacio virtual. Aunque hay quien asegura, que con sólo estimular el sentido de la vista, es suficiente.²⁴

Por lo antes expuesto, vale la pena mencionar dos conceptos más, en lo que a inmersión se refiere.

1.3.7.1 Inmersión de bajo nivel²⁵

Es cuando el usuario, sentado frente a la computadora percibe todo el mundo a través de la pantalla. Definíamos en párrafos anteriores que inmersión es sumergirse, pues aún y cuando esta situación sea llamada de bajo nivel es mejor no llamarle inmersiva, sino como es más comúnmente llamada *De Escritorio* o *Ventana en el Mundo* (Window on World, WoW).²⁶

1.3.7.2 Inmersión de alto nivel²⁷

Toda la información del mundo virtual es desplegada al usuario a través de un dispositivo llamado Head Mounted Display (HMD). El HMD o también llamado casco de visualización, es, como su nombre lo indica, un casco que contiene audífonos y un par de pantallas colocadas frente a los ojos, consiguiendo de este modo aislar al usuario del mundo exterior para aumentar la ilusión de presencia al no tener punto de comparación con el entorno real, además consta de un dispositivo de localización que permite a la computadora saber la posición y orientación de la cabeza del usuario y poder calcular la escena correspondiente.

²⁴ Cfr. HARRISON, DAVID; JAQUES, MARK, *op. cit.*, p.4; CHORAFAS, DIMITRIS N.; STEINMANN, HEINRICH, *op. cit.*, p.42

²⁵ Cfr. <http://www.oz.net/~avi/gloss.htm>

²⁶ Cfr. <http://www.oz.net/~avi/gloss.htm>, <http://www.cms.dmu.ac.uk/~cph/VR/whatisvr.html>, c. I.1

²⁷ Cfr. <http://www.oz.net/~avi/gloss.htm>

1.3.8 Punto de Vista

Lo anterior indica que para presentar la imagen adecuada al usuario, necesitamos conocer su punto de vista, es decir, lo que él estaría viendo dentro del mundo virtual de acuerdo a su posición y a la orientación de sus ojos.²⁸

1.4 Hardware de un Sistema de RV

En consecuencia, haciendo una revisión de todo lo que un sistema debe ofrecer para ser considerado de RV, llegamos a la conclusión de que requerimos ciertos dispositivos específicos de entrada y salida. Y observando los hechos, una adecuada computadora no puede faltar.

1.4.1 Dispositivos de Entrada

Para que un sistema de RV soporte la comunicación hombre-máquina basada en la natural forma de expresión del hombre, necesita incorporar algunos dispositivos especiales para comunicarse con la computadora.²⁹

1.4.1.1 Dispositivo de Localización

Definitivamente el teclado es el más común dispositivo de entrada, pero como mencionábamos en párrafos anteriores un sistema de RV necesita conocer, de manera automática, la posición del usuario en un espacio tridimensional, así como la orientación de su punto de vista para dar esa sensación de inmersión en el espacio virtual.

²⁸ Cfr. HAYWARD, TOM, op. cit., p.13

²⁹ Cfr. JACOBSON, LINDA, op. cit., p.117

Tenemos pues, que estos dispositivos de localización, por lo general, son colocados en el casco de visualización o HMD y los hay de varios tipos.

- *Dispositivos de localización electromecánicos, consisten en una serie de barras conectadas del casco a una base, con suficientes uniones para permitir el movimiento del usuario; en cada unión se encuentra un sensor que mide el ángulo y con la lectura de los ángulos y algunas operaciones se conoce la posición y orientación del casco, éste es un sistema basado en el brazo mecánico.*³⁰

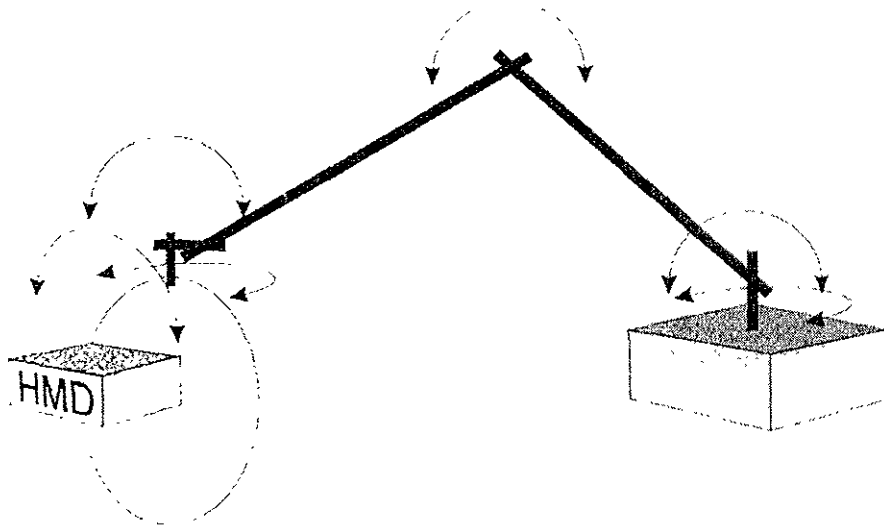


Figura 3. Dispositivo de Localización Electromecánico

- *Dispositivos de localización electromagnéticos, no existe conexión física entre la base y el casco, sino que una base emite un campo magnético a través de tres bobinas ortogonales. El receptor también contiene tres bobinas ortogonales y cada una genera una corriente de acuerdo a la fuerza y ángulo del campo magnético de la base, al procesar esta información se conoce la posición y orientación del casco. Cabe señalar*

³⁰ Cfr. HOLLANDS, ROBIN, *The Virtual Reality Homebrewer's Handbook*, p.146. Los tipos de sistemas de seguimiento mecánicos se pueden consultar en GRADECKI, JOE, *op. cit.*, p.212, 213

que a pesar de que su campo de transmisión es reducido, es el sistema más popularmente comercial.³¹

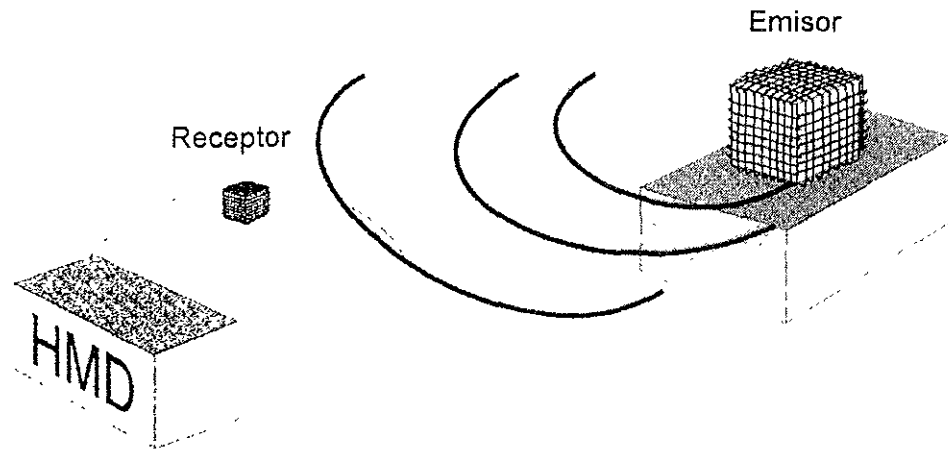


Figura 4. Dispositivo de Localización Electromagnético

➤ Dispositivos de localización ultrasónicos, la única conexión entre la base y el casco son pulsos ultrasónicos. Para medir la posición necesitamos un emisor y tres receptores colocados triangularmente, el pulso ultrasónico se envía al mismo tiempo que se disparan tres timers, cuando el pulso llega a cada receptor detiene su timer; cada tiempo representa distancia con la que por triangulación se calcula la posición. Para medir la orientación se añaden otros dos emisores.³²

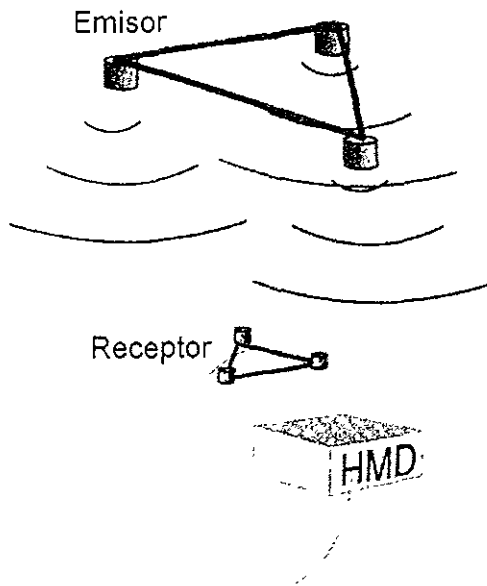


Figura 5. Dispositivo de Localización Ultrasónico

³¹ Cfr. HOLLANDS, ROBIN, *op. cit.*, p.147; <http://www.cms.dmu.ac.uk/~cph/VR/whatisvr.html>, c.I.2

³² *Ibid* p.148

- Dispositivos de localización ópticos, tampoco existe una conexión física sino que se localiza al usuario mediante dos métodos, el primero consiste en un conjunto de LED's colocados en el techo y sensores o cámaras colocadas en el casco; la computadora activa un LED y checa si algún sensor lo detectó, esto se repite para cada LED; toda esta información se envía a la computadora para compararla con la información anterior y así detectar cambios en la posición del usuario. El segundo método consiste en observar al usuario mediante una cámara y digitalizar esa imagen para que mediante procedimientos de análisis como detección de bordes o uso de fuentes visibles determine su posición.³³

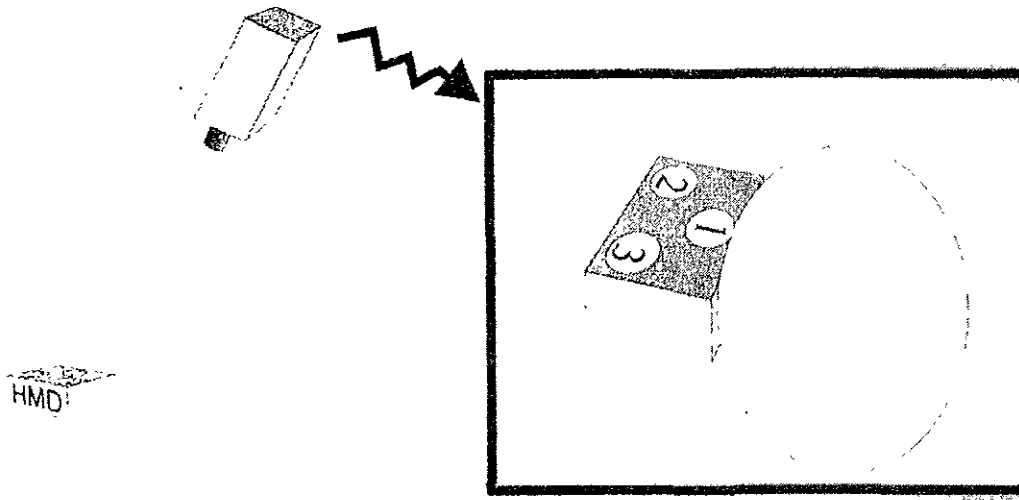


Figura 6. Dispositivo de Localización Óptico (Segundo Método)

La lista de dispositivos de localización continua, sin embargo no es el caso de ello el presente trabajo, por tanto nos remitimos sólo a mencionarlos.

³³ Cfr. GRADECKI, JOE, *op. cit.*, p.214, 217; <http://www.cms.dmu.ac.uk/~cph/VR/whatisvr.html>, c.I.2

Dispositivos de Localización Inerciales³⁴, Dispositivos de Localización de Brújula e Inclinación y Dispositivos de Localización Giroscópicos³⁵.

1.4.1.1.1 Latencia

Es el mayor problema en los sistemas de localización de la posición. Es el tiempo que tarda el sistema de RV desde que el usuario realiza un movimiento, el dispositivo de localización lleva a cabo las mediciones y la computadora realiza los cálculos, hasta que la nueva imagen es presentada en el dispositivo de salida. Sin embargo en forma general, para cualquier acción que realice el usuario es el tiempo requerido para realizar procesos, antes de actualizar la información de salida.³⁶

1.4.1.2 Dispositivo de Control

Para que se dé la interacción en el sistema se ofrece al usuario el control del espacio que le permite modificar su punto de vista, manipular objetos y realizar determinadas acciones.³⁷

Entre los dispositivos de control se encuentra el mouse tridimensional, el joystick y el forceball, este último es empleado en los sistemas de escritorio ya que consiste en un dispositivo que en la parte de arriba cuenta con una esfera la cual se empuja, se jala y se gira en cualquier dirección. El mouse tridimensional y el joystick son especiales para sistemas inmersivos, cuentan con un dispositivo de

³⁴ <http://www.cms.dmu.ac.uk/~cph/VR/whatisvr.html>, c.I.2

³⁵ Se puede consultar sobre estos dos dispositivos en HOLLANDS, ROBIN, *op. cit.*, p.151-154

³⁶ Cfr. <http://www.cms.dmu.ac.uk/~cph/VR/whatisvr.html>, c. I.2; DEL PINO GONZALEZ, L. M., *op. cit.*, p.23

³⁷ Cfr. STAMPE, DAVE; ROEHL, BERNIE; EAGAN, JOHN, *op. cit.*, p.84; CHORAFAS, DIMITRIS N.; STEINMANN, HEINRICH, *op. cit.*, p.88

localización en su interior para calcular su posición y desplegar la imagen de una mano dentro del espacio virtual.³⁸



Figura 7. Mouse Tridimensional a la izquierda, Joystick al centro y Power Glove a la derecha

Pero en cuanto tal, el Data Glove o Guante es el dispositivo adecuado para verdaderamente manipular los objetos debido a que los tocamos con nuestra propia mano, ante ello surge la necesidad de sentir los objetos por lo que también es considerado un dispositivo de salida.

Generalmente el desplazamiento se realiza con el dispositivo de control y el dispositivo de localización se encarga de la orientación o dirección de la vista.

1.4.1.2.1 Mareo de Inmersión

Sucede cuando el usuario dentro de un sistema de RV inmersivo, experimenta desplazamientos a través de escenas virtuales pero él físicamente no se mueve, el problema consiste en el descontrol de sus neuronas al estar engañando sus sentidos. También sucede cuando se presentan retardos detectables entre los movimientos de la cabeza y las imágenes desplegadas en el HMD.³⁹

³⁸ Cfr. HOLLANDS, ROBIN, *op. cit.*, p.244-246

³⁹ Cfr. JACOBSON, LINDA, *op. cit.*, p.102

1.4.2 Dispositivos de Salida

Recordando que pretendemos darle al usuario una ilusión convincente y proporcionarle la sensación de estar dentro del mundo virtual, tenemos pues, que estimular sus sentidos.

1.4.2.1 Dispositivo de Presentación

Así como para el perro el sentido del olfato es su suprasentido, el sentido de la vista y el sentido del oído son los suprasentidos del hombre, debido ello a que mediante estos dos sentidos se guía principalmente.

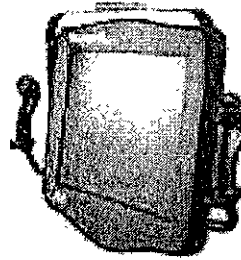


Figura 8. Pantalla de Despliegue (WoW)

Así tenemos, que el esmero porque el dispositivo a través del cual se despliegan las imágenes sea de alta calidad, ha sido el elemento más importante para muchos diseñadores. De esta manera, hoy en día contamos con poderosos cascos de visualización en su mayoría y se siguen diseñando novedosos dispositivos de presentación para tareas específicas dentro de los sistemas de RV.

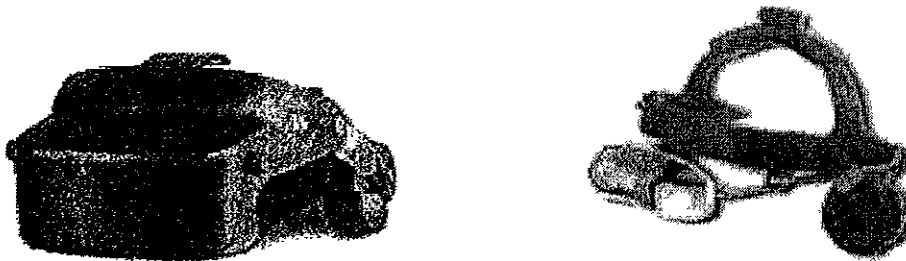


Figura 9. Casco de Visualización V6 y Casco de Lentes Glasstron

1.4.2.2 Dispositivo de Audio

Argumentábamos párrafos atrás, que el hombre vive en un mundo tridimensional y todo lo percibe en tres dimensiones, incluyendo el sonido. Debido a la posición de nuestros oídos podemos identificar la procedencia de un sonido, el cerebro compara lo que cada uno de nuestros oídos percibe y la diferencia⁴⁰ nos proporciona la dirección precisa de donde se produjo un sonido; la distancia es cuestión de volumen. Claro que si queremos ser más precisos necesitamos filtrar el sonido de acuerdo a la ecuación matemática que determina la forma de nuestras orejas.⁴¹

La computadora debe ser capaz de calcular los sonidos del mundo virtual en forma tridimensional y presentarlos adecuadamente en el dispositivo de sonido, generalmente, un par de audífonos.

1.4.2.3 Dispositivos de Realimentación Táctil

Los avances de la ciencia han permitido aumentar la sensación de presencia al agregar información táctil a los sistemas de RV y aunque no es considerado un suprasentido, estamos de acuerdo que en lugares faltos de buena visibilidad, el sentido del tacto es nuestra alternativa.

Para estimular el sentido del tacto de todo el cuerpo necesitaríamos una especie de traje de buzo forrado de unos diez millones de sensores, ante ello, lo actualmente disponible es un Data Glove o guante que estimula la mano del usuario con sensaciones lo más cercanas posibles al espacio virtual que se esté

⁴⁰ Las diferencias que dan la capacidad de localizar la fuente de un sonido se pueden consultar en GRADECKI, JOE, *op. cit.*, p.166-177

⁴¹ HAYWARD, TOM, *op. cit.*, p.56

procesando, además de ser un dispositivo de manipulación y control disponible para el usuario.⁴²

Los otros dos sentidos son más difíciles de estimular, cálculos químicos extremadamente complejos se necesitan para saber la composición de una molécula de olor, a cada olor corresponde una molécula distinta, para su generación se necesitaría todo un sofisticado laboratorio. Y la opción de depositar los olores para ser utilizados en su momento tampoco es muy factible. En cuanto al sentido del gusto, no sería agradable conectar una computadora a la lengua, pero independientemente de la conexión, ¿a quién se le antoja probar un objeto virtual?⁴³

1.4.2.4 Dispositivos Móviles

Una variante de la inmersión de alto nivel es el uso de Cabina, una pequeña bóveda en la que se encierra al usuario. En las paredes se proyectan las imágenes del mundo virtual lo que da la sensación de viajar por el mundo dentro de algún medio de transporte. Esta cabina esta montada en una plataforma móvil que realiza los movimientos acordes con la situación de la cabina en el espacio virtual y dependiendo del mundo virtual se eligen los dispositivos de control.⁴⁴

Los dispositivos móviles son empleados en la simulación de vuelos, simulación de la conducción de un barco, un tractor, un submarino o una cápsula que viaja al fondo del mar, o bien una nave espacial que explora el espacio, entre otros.

⁴² Cfr. GATES, BILL, Camino al Futuro, p.130, 131, JACOBSON, LINDA, op. cit., p.127

⁴³ Cfr. GATES, BILL, op. cit., p.130

⁴⁴ Cfr. <http://www.cms.dmu.ac.uk/~cph/VR/whatisvr.html>, c.I.3

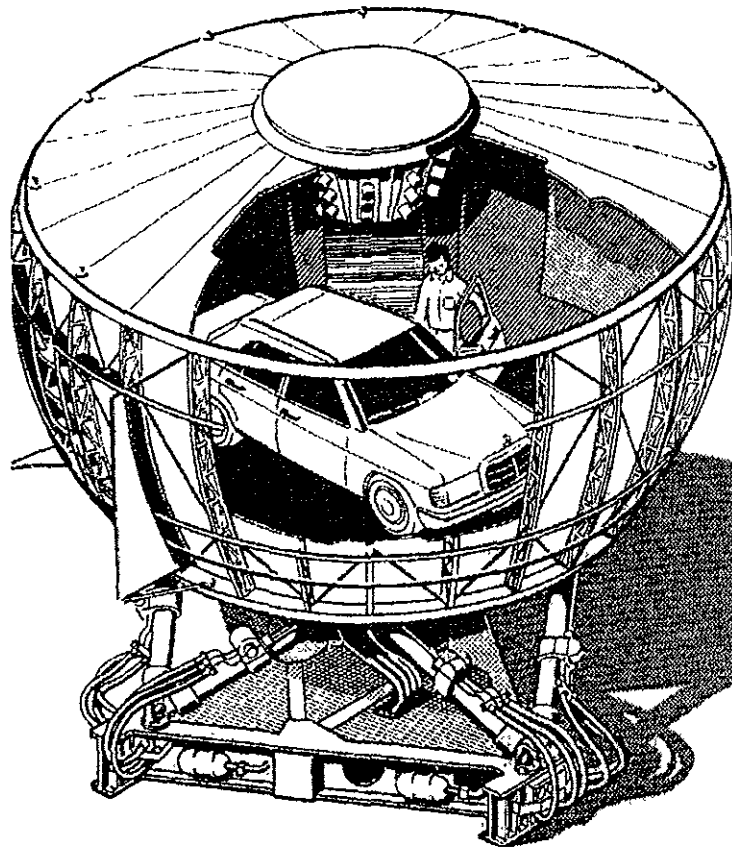


Figura 10. Simulador de conducción de un automóvil

1.4.3 Computadora

Cuando observamos todo lo que se requiere para un sistema de RV, nos damos cuenta que el equipo de cómputo no será tan ordinario.

La computadora procesa y despliega simulaciones interactivas en tercera dimensión, es decir, mundos virtuales; sin embargo no olvidemos que estos mundos, para la computadora, son una enorme base de datos de código binario almacenados en su memoria. Esta, tiene que hacer cálculos de despliegue para los modelos a una velocidad adecuada para proporcionar un despliegue constante, tiene que controlar la navegación; para lo cual necesita mucha memoria donde

tenga almacenados todos los detalles del mundo virtual y además actualizar todo lo que el usuario realice para cumplir con la interactividad pero cuidando convencer al usuario de que está interactuando con un espacio virtual y no con una computadora.

Avanzados estudios sobre los patrones de uso de las computadoras han revelado que las computadoras de uso general pasan más del 80% del tiempo ejecutando instrucciones simples como: cargar, guardar y tomar decisiones. Las instrucciones complejas se ejecutan con poca frecuencia y desafortunadamente no solo agregan funcionalidad, sino que también producen retraso; debido a la adicional decodificación y los largos ciclos de tiempo. Este retraso adicional reduce el performance de la ejecución de instrucciones simples y no representa ninguna ventaja en la implementación de instrucciones complejas.⁴⁵

Los procesadores RISC (Reduce Instruction-Set Computers) ejecutan instrucciones sencillas a gran velocidad; a pesar de emplear el doble de sencillas instrucciones que las complejas instrucciones de los procesadores CISC (Complex Instruction-Set Computers) para llevar a cabo una misma tarea. Como la arquitectura RISC saca los datos de la memoria, los coloca en los registros para manipularlos y después los vuelve a guardar en memoria, el número de accesos a memoria por instrucción es la mitad que el de los procesadores CISC que manipulan los datos directamente en la memoria. Además el Tiempo de Reloj por instrucción es menor en los procesadores RISC.⁴⁶

El conjunto de instrucciones reducidas está diseñado para ser una excelente opción para compiladores optimizados al usar instrucciones sencillas que son ejecutadas en un ciclo de CPU.⁴⁷

⁴⁵ Cfr. <http://www.hp.com/nsa/pa1.1/html/acd-4.html#HEADING4-0>

⁴⁶ SIEWIOREK, DANIEL P.; KOOPMAN, PHILIP JOHN, JR., *The Architecture of supercomputers*, p.77

⁴⁷ Cfr. <http://www.hp.com/nsa/pa1.1/html/acd-4.html#HEADING4-0>

Esto nos lleva a elegir una poderosa WorkStation; estación de trabajo con procesador RISC, que en tiempo real calcule cada frame, es decir cada imagen de acuerdo a la ubicación del usuario y despliegue toda la secuencia de frames de la manera más rápida. Y por si fuera poco una estación de trabajo tiene la capacidad de realizar cálculos matemáticos extremadamente complejos y de estar conectada a una red de área local para compartir archivos y recursos.⁴⁸

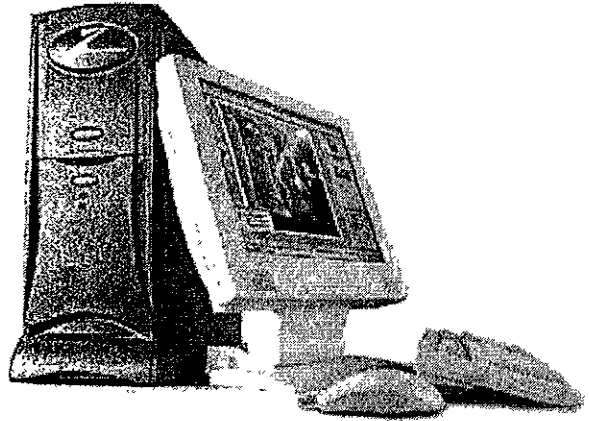


Figura 11. Estación de Trabajo Intergraph

1.5 Software de un Sistema de RV

Conociendo lo que se tiene que hacer y teniendo el hardware para hacerlo, ahora hay que programarlo.

Todos los programas de presentación siguen un ciclo de seis pasos: primero obtener la entrada del usuario, con esto transformar y proyectar vértices, ahora ordenar los objetos, después eliminar facetas ocultas, dar color y por último dibujar en pantalla.⁴⁹

El usuario proporciona la entrada a través de los dispositivos de entrada, el software debe ser capaz de comunicarse con la electrónica de estos dispositivos para utilizar la información suministrada. Con esta información obtiene las coordenadas del mundo real, las transforma a coordenadas de visualización y después a coordenadas de pantalla, aquí es donde se hacen los cálculos de todo lo

⁴⁸ Cfr. JACOBSON, LINDA, *op. cit.*, p.51-54.

⁴⁹ Cfr. GRADECKI, JOE, *op. cit.*, p.242

que el usuario haya modificado, para esto se usan matrices de transformación que trabajan sobre los vértices del objeto, cabe señalar que son cálculos muy costosos en tiempo real. Para ordenar los objetos se hace uso de su coordenada z para identificar qué objetos están delante de cuáles. La eliminación de superficies ocultas es con el fin de ahorrar tiempo de representación y se determina a partir del vector normal de los polígonos, si el vector normal apunta hacia el usuario entonces la superficie es visible, si apunta hacia la parte opuesta del usuario entonces ese polígono se elimina. El color es el que dota de realismo a nuestro mundo, cada luz posee dirección y color, cada luz afecta a cada objeto y se emplea algún procedimiento de sombreado. Finalmente el procedimiento de representación es dibujar en la pantalla de la computadora.⁵⁰

Pero además de desplegar nuestro mundo virtual es necesario aplicarle leyes físicas que le den realismo y simular el comportamiento de nuestros elementos para poder interactuar con ellos de la manera más real.

Existen diversos programas de software para crear mundos virtuales, pues aún y cuando la elección depende de nuestras necesidades, entran en juego las capacidades de un software de RV.

1.5.2 Capacidades de un Software de RV.

Polígonos por segundo. Sabiendo el número de polígonos que puede procesar en un segundo, nos proporciona una aproximación de la rapidez con que se calcularán los frames del espacio virtual.⁵¹

Sombreado. La manera en que la luz es modelada en la superficie de los objetos puede ser Lambert, Gouraud y Phong. En el sombreado Lambert, también llamado plano, se colorea uniformemente cada cara del objeto de acuerdo a su

⁵⁰ Ibid, p.242-247

⁵¹ Cfr. HOLLANDS, ROBIN, *op. cit.*, p.50

orientación con la luz. En el sombreado Gouraud, llamado suave, cada vértice del objeto se colorea de acuerdo a su orientación con la luz y toda el área intermedia es una combinación de los colores a su alrededor. El sombreado Phong es también llamado suave con *toques de luz*⁵², como su nombre lo indica es un sombreado Gouraud que añade una impresión de brillo a la superficie del objeto.⁵³

Múltiples luces. La iluminación del mundo virtual es un factor importante que podemos programar contando con luces ambientales, luces tipo reflector, luces direccionales y luces omni-direccionales.⁵⁴

Mapeado de texturas. Es permitir mapear imágenes en las caras de los objetos con una correcta perspectiva para que el mapeado se vea bien desde cualquier ángulo.⁵⁵

Escalable. Se refiere al cálculo de los objetos tomando en cuenta su perspectiva lineal y su nivel de detalle.⁵⁶

Resolución de Pantalla. Tener en cuenta la resolución de despliegue que el software soporta. De nada nos sirve tener el mejor dispositivo de presentación si el software no cuenta con tanta capacidad.

Compatibilidad para exportar e importar objetos. El formato en que almacene los objetos tridimensionales le dará la posibilidad de intercambiar objetos con AutoCAD (dxf⁵⁷) y 3Dstudio (3ds).⁵⁸

⁵² Toque de luz es, por ejemplo el brillo en una bola de billar. HOLLANDS, ROBIN, *op. cit.*, p.50

⁵³ Cfr. *Geometry Tools*, c.2, p.22 y 23.

⁵⁴ Cfr. *Ibid*, c.2, p.21

⁵⁵ Cfr. HOLLANDS, ROBIN, *op. cit.*, p.51

⁵⁶ *Ibid* p.52

⁵⁷ dxf=Data eXchange Format

⁵⁸ Cfr. HOLLANDS, ROBIN, *op. cit.*, p.53

Jerarquía en los objetos. Esto se refiere a la posibilidad de ligar objetos con el propósito de heredar características y crear una relación entre ellos, por ejemplo al momento de moverlos o modificarlos.⁵⁹

Nivel de detalle. El software debe ser capaz de modelar el nivel de detalle de los objetos.

Detección de Colisiones. Es la capacidad que tiene el software de saber cuando un objeto choca con otro o intenta atravesar otros objetos y cuando la mano virtual está tocando los objetos para poder manipularlos.

Lenguaje de Programación. Lenguaje que usa el software de RV para programar acciones más específicas.⁶⁰

Múltiples puntos de vista. Cantidad de puntos de vista permitidos para el mundo virtual.

Plataforma. Tipos de plataformas en las que el software puede correr.

1.6 Otros

Existen algunos conceptos básicos que aún falta definir.

1.6.2 Luces

Para la iluminación del mundo virtual se emplean distintos tipos de luces:

➤ Point

Es la fuente de luz más simple para modelar, radia la misma intensidad de luz hacia todas direcciones. Para ser modelada solo requiere su posición y su intensidad.

⁵⁹ Cfr. dVISE, c.7, p.30

⁶⁰ Cfr. HOLLANDS, ROBIN, op.cit., p.54

➤ Directional

Este tipo de luz simplemente irradia luz hacia una dirección específica y su intensidad incide sobre la superficie en esa dirección.

➤ Spot

El nombre de esta luz significa lunar o mancha y efectivamente la luz que proyecta incide sobre una dirección específica y sobre un área específica simulando una mancha de iluminación. Para ser modela requiere intensidad, posición, dirección y ángulo de iluminación.

➤ Ambient

Este tipo de luz es una muy buena opción para esas superficies que no van a recibir ningún tipo de iluminación por ninguna parte. Es como un nivel de luz de fondo.

1.6.3 Ejes de Coordenadas

Un término importante para especificar la orientación del usuario y determinar su punto de vista es la rotación de los planos comprendidos sobre dos ejes:

- Yaw. Rotación sobre el eje y.
- Pitch. Rotación sobre el eje x.
- Roll. Rotación sobre el eje z.

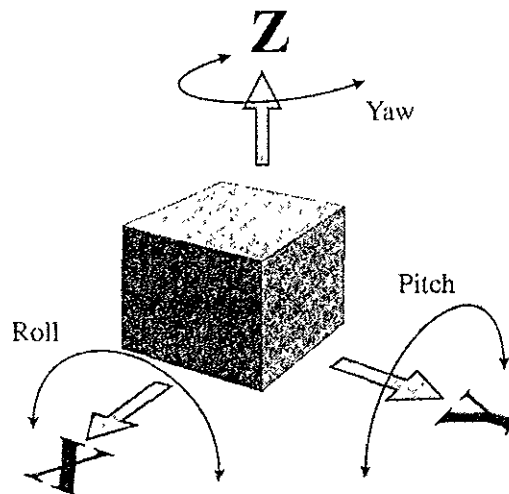


Figura 12. Ejes de Coordenadas

Para mayor facilidad se ilustra en las siguientes imágenes:

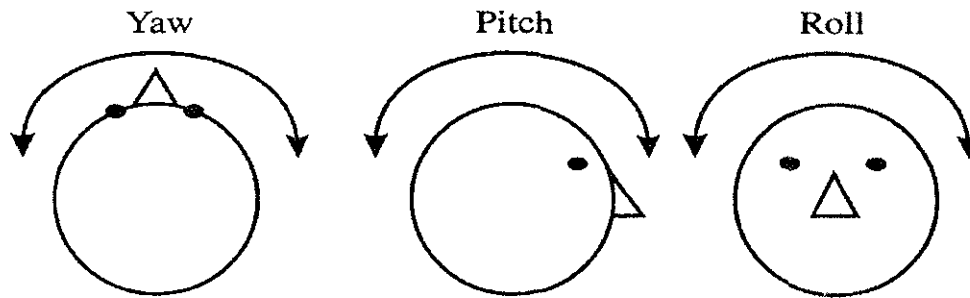


Figura 13. Movimientos de Yaw, Pitch y Roll

1.7 Tomas Furness y HITL

Entre los pioneros de la RV se encuentran Morton Heilig, Ivan Sutherland, Dr. Frederick Brooks, Jr., Tomas Furness III y Myron Krueger.

Morton Heilig construyó la primera máquina inmersiva llamada Sensorama, patentada en 1961. Era una especie de bicicleta cubierta por una caja, Sensorama

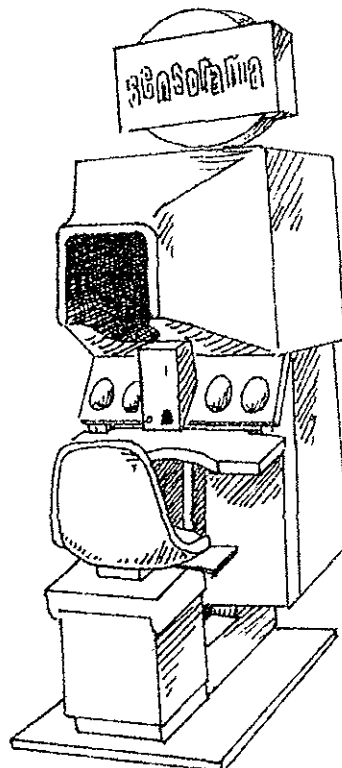


Figura 14. Sensorama

colocaba al observador en un teatro para una persona y le permitía experimentar una de cinco películas tridimensionales a todo color con sensaciones auxiliares de sonido, movimiento, viento en la cara y olor.⁶¹

Las películas eran de poca calidad al igual que el sonido, la presentación de movimiento y olor era irregular; sin embargo con los ojos contra los goggles estereoscópicos y una película a todo color se lograba la sensación de “ estar ahí ”: Lo que significa RV sin computadoras.⁶²

Ivan Sutherland, el padre de los Gráficos por Computadora, en 1960 cumplió



Figura 15. Fotografía del casco de Sutherland

por primera vez con la promesa de la RV al imaginar un casco de visualización con visión estereoscópica controlado por computadora. Para 1966, Ivan Sutherland construyó el primer HMD que sería el principio del concepto de inmersión y comenzó una serie de experimentos con pantallas tridimensionales. Este primer casco de visualización era demasiado voluminoso, pesado e inestable.⁶³

Cabe señalar que sus contribuciones, el HMD y los gráficos por computadora hicieron posible la RV.⁶⁴

⁶¹ Cfr. HARRISON, DAVID; JAQUES, MARK, op. cit., p.5; HAMIT, FRANCIS, Virtual Reality and the Exploration of Cyberspace, p.55

⁶² Cfr. HAMIT, FRANCIS, op. cit., p.56

⁶³ Cfr. HAMIT, FRANCIS, op. cit., p.58; HARRISON, DAVID; JAQUES, MARK, op. cit., p.37; CHORAFAS, DIMITRIS N.; STEINMANN, HEINRICH, op. cit., p.27

⁶⁴ Cfr. HAMIT, FRANCIS, op. cit., p.59

Dr. Frederick Brooks, Jr. junto con su equipo de la Universidad de Carolina del Norte creó en 1971, una aplicación para trabajar con los modelos tridimensionales de las moléculas de proteína. Además construyó otra aplicación médica que consistía en pantallas transparentes que permitían ver al paciente y la imagen de la computadora al mismo tiempo, lo que permitía a los médicos aprender a operar en tercera dimensión.⁶⁵

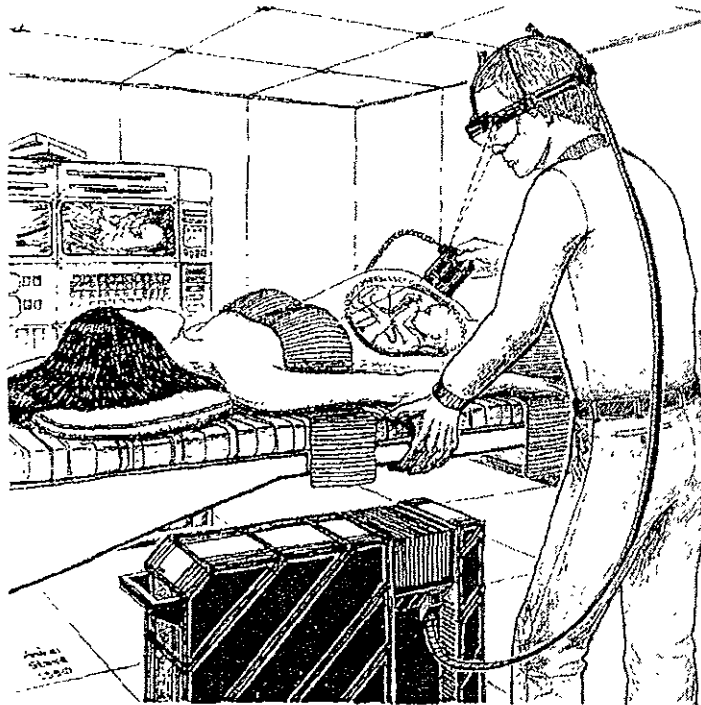


Figura 16. Aplicación de la Realidad Virtual a la Medicina

Myron Krueger en 1969, comenzó su investigación en la Universidad de Wisconsin con respecto a la interacción hombre-máquina, cuya parte esencial, desde un punto de vista filosófico, son los factores humanos de percepción. En 1970 inventó el término *Realidad Artificial*, título del libro que en 1972 terminó de escribir.

⁶⁵ Cfr. HARRISON, DAVID; JAQUES, MARK, *op. cit.*, p. 38-40; HAMIT, FRANCIS, *op. cit.*, p. 60

Tomas Furness III, jefe de la Subdivisión de Sistemas de Despliegue Visual de la División de Ingeniería Humana del Laboratorio de Investigaciones Aeroespaciales Armstrong de la Base de la Fuerza Aérea Wright-Patterson en Ohio fue la primer persona que hizo un sistema de RV totalmente completo, el SuperCockpit. Este, permitía a los pilotos viajar a muy altas velocidades usando sólo su cabeza, sus ojos y movimientos de la mano. La idea de un casco montado en la cabeza con *vista infrarroja para guiar los misiles hacia el blanco*, llevó al concepto del SuperCockpit.⁶⁶

Se dice que gracias a las posibilidades económicas de la Armada se consiguió construir este primer sistema de RV totalmente completo, ya que los patrocinadores de la RV eran la Fuerza Aérea, la Marina de USA y la NASA; lo anterior, sin embargo, no resta créditos a Tomas Furness.⁶⁷

En 1989 Tomas Furness fundó el HITL después de dejar la Fuerza Aérea, HITL son las siglas de Human Interface Technology Laboratory o Laboratorio de Tecnología en Interfaces Hombre-Máquina de la Universidad de Washington en Seattle.⁶⁸

El objetivo del laboratorio no era solamente la creación de nueva tecnología de RV para la industria, educación, diseño, manufactura y entretenimiento; sino además la investigación de la relación entre el hombre y la computadora, comprender la forma en que el hombre percibe el mundo que le rodea, es decir, profundizar en la capacidad cognositiva, perceptiva, sensitiva y motora del hombre.⁶⁹

⁶⁶ Cfr. HAMIT, FRANCIS, *op. cit.*, p.61-62

⁶⁷ Cfr. HARRISON, DAVID; JAQUES, MARK, *op. cit.*, p.43; CHORAFAS, DIMITRIS N; STEINMANN, HEINRICH, *op. cit.*, p.27

⁶⁸ Cfr. HAMIT, FRANCIS, *op. cit.*, p.110

⁶⁹ Ibid

Al principio contaba con el equipo que la Fuerza Aérea le prestó y con el patrocinio de la compañía Boeing, sin embargo por alguna razón todo esto terminó, perdió a mucha gente así como muchos proyectos pero siguió trabajando en dos proyectos: VEOS, Sistema Operativo de Ambientes Virtuales y VRS, dispositivo de despliegue de imágenes directamente en la retina a través de láser, llamado Scanner Virtual en la Retina. En 1992 salió adelante con 18 proyectos de hardware, software y factores humanos y ha conseguido satisfacer su ambición de ser el mayor centro de investigación en RV.⁷⁰

En Febrero de 1991 patrocinó el primer Simposium Industrial Anual en Tecnología de Mundos Virtuales, en este evento participaron 500 corporaciones y la compañía Boeing.⁷¹

En Noviembre de 1997 al termino del evento La Realidad Virtual en México, organizada por la Facultad de Ingeniería de la UNAM, en el cual participaron 5 compañías y 6 Universidades, incluyendo la Universidad de Washington; fue inaugurado por Tomas Furness, en la Facultad de Ingeniería de la UNAM, el HITL South, llamado por el mismo Tomas Furness el retoño del HITL de la Universidad de Washington.

⁷⁰ Ibid p.111-113

⁷¹ Ibid p.110

ENTONCES ...

Un completo Sistema de Realidad Virtual involucra poderosos y modernos recursos que solo trabajando coordinadamente logran producir en el usuario ese efecto de estar presente dentro de un espacio en el cual puede tomar parte, al grado de modificarlo, de acuerdo a sus acciones.

Repasemos las características necesarias y suficientes de un Sistema de Realidad Virtual:

- ◆ Capacidad Sintética
- ◆ Tridimensionalidad
- ◆ Estereoscopia
 - Claves de Profundidad
- ◆ Interactividad o Manipulación
- ◆ Navegación
- ◆ Inmersión
- ◆ Punto de Vista

CAPÍTULO 2

HITL South

(Human Interface Laboratory, sucursal South)

Laboratorio de Interfaces Inteligentes

La Universidad Nacional Autónoma de México en la Facultad de Ingeniería cuenta con un laboratorio que se especializa en la realización de proyectos para la interfaz hombre-máquina de una forma óptima e inteligente. Por varios años; maestros y estudiantes de licenciatura y de maestría de la Facultad han trabajado en estos proyectos.

A este laboratorio se incorporó un nuevo proyecto al llegar un equipo de Realidad Virtual en marzo de 1997, desde ese momento la Facultad de Ingeniería de la UNAM tuvo la oportunidad de tomar parte en ese tema tan desconocido hasta entonces.

En Noviembre de 1997 oficialmente Tomas Furness III inauguró el laboratorio como HITL South (Human Interface Technology Laboratory, sucursal South).

En el HITL South se desarrollan proyectos con varios robots que han sido diseñados y/o programados por académicos del laboratorio, entre estos, se encuentra el robot TX8, que cuenta con un procesador PENTIUM, tarjeta de red para poder tener acceso a él de forma remota, 2 microcontroladores HC11 para el control de los sensores; los sensores con los que TX8 cuenta son sonares, infrarrojos y sensores de choque, también tiene una tarjeta de distribución para el control de estos sensores. Todo el software desarrollado para el robot TX8 es producto de muchas horas de trabajo; el objetivo de TX8 es la búsqueda de ruta y planificación de movimientos haciendo uso de una red neuronal y un algoritmo; esto lo hace tener un inmenso valor académico.

Obviamente resulta necesario realizar repetidas pruebas de los algoritmos que se están desarrollando, además de que alumnos de maestría necesitan aprender a programar al robot y por desgracia aprenden a base de prueba y error.

Es por ello que surge la necesidad de contar con un ambiente virtual en el cual se lleven a cabo todas las pruebas preliminares, de manera tal, que solo la prueba final o definitiva se lleve a cabo directamente al robot TX8, además de dar una solución para que los alumnos puedan aprender a programarlo sin estarlo dañando.

En el laboratorio se tienen diagramas en 2 dimensiones del espacio de prueba del robot, sin embargo, como humanos que somos no solemos conformarnos con eso y queremos ver que realmente funcione la prueba realizada y ante ello se ejecuta directamente en el robot; ahora bien, si en vez de probar directamente en el robot se ofrece la alternativa de hacer la prueba virtualmente, introduciendo al programador dentro del espacio virtual donde el robot realiza sus pruebas al ejecutar el programa, cumpliríamos con el objetivo de que el programador "vea" virtualmente la ejecución como si en realidad estuviéramos dentro del espacio de prueba.

Por otro lado, hay ocasiones en que tanta interferencia e imprecisión en los datos de entrada nos impiden determinar si lo que está fallando es el algoritmo programado, la red neuronal empleada o en realidad es error en los datos de entrada que el ambiente arroja.

Para lograr este objetivo, es necesario diseñar dicho espacio virtual de pruebas y contar con la interfaz que nos permita ejecutar los algoritmos que se están programando para el robot directamente dentro del espacio virtual.

2.1 El Diseñador

Es tarea del diseñador, cubrir todas las características necesarias para proporcionarle al usuario la oportunidad de interactuar con un Sistema de Realidad Virtual.

El usuario lo que espera obtener, es la oportunidad de: alcanzar, tocar y mover los objetos virtuales, es decir, manipular el espacio virtual; desplazarse y explorar el espacio virtual, lo que llamamos navegar; y experimentar la sensación de inmersión.⁷²

Ante este punto de vista del usuario, el diseñador es el encargado de: conocer, en todo momento, la posición del usuario, en particular la posición y orientación de su cabeza y de su mano; con el objeto de proporcionar la sensación de un ambiente real; además debe controlar la adquisición de los datos de entrada y las características y comportamientos de los objetos virtuales para proporcionar una sensación de control en el ambiente virtual; también debe preocuparse por el despliegue y la retroalimentación de los dispositivos de salida visuales, auditivos y táctiles.⁷³

Dentro del HITL South, el diseñador puede cubrir en su totalidad sus actividades, con ayuda del equipo de RV y el software con que cuenta el Laboratorio.

2.2 Hardware para un Sistema de RV

Como mencionábamos en párrafos anteriores en el HITL South existe equipo de Realidad Virtual.

2.1.1 Dispositivos de Entrada

Cuando se interactúa con un espacio virtual, varios sensores y transductores se necesitan para monitorear las acciones del usuario y las señales de

⁷² Cfr. HARRISON, DAVID; JAQUES, MARK, *op. cit.*, p.8

⁷³ *Ibid*, p.9

retroalimentación que reflejan el estatus del ambiente. Por ejemplo, un joystick interactivo debe ser detectado en tres dimensiones. Para monitorear la posición del joystick se necesita un sensor que nos proporcione las coordenadas x,y,z relativas a una referencia, las cuales se complementan con ángulos de orientación.⁷⁴

2.2.1.1 Dispositivo de localización

Por la necesidad obvia de proporcionar información de referencia espacial y angular a cada instante, el sistema detector debe operar en tiempo real y por lo menos proporcionar 50 actualizaciones por segundo.

El detector de posición con que el HITL South cuenta es un dispositivo electromagnético de la serie 3SPACE de la empresa Polhemus: Polhemus Fastrack, el cual, proporciona actualizaciones a una velocidad de 120 Hz. Como ya se mencionó este tipo de dispositivos de localización son los más comerciales, su costo es relativamente bajo y permite cierta libertad de movimiento, esta empresa fue la primera en sacarlo al mercado y la mayoría de los paquetes de SW de desarrollo de RV tiene drivers para los sensores de Polhemus. Cabe señalar sus desventajas: alcance limitado por la potencia del emisor, su nivel de error es proporcional a la distancia así como a la presencia de objetos metálicos grandes, tiene una latencia de 10 ms, lo que hace que el movimiento del usuario y de la imagen no estén sincronizados.⁷⁵

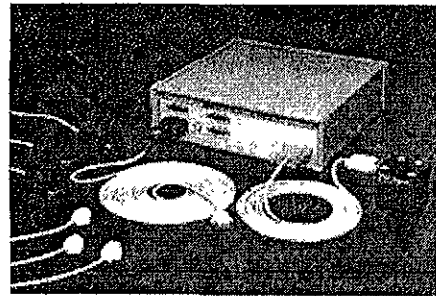


Figura 17. Polhemus Fastrack

⁷⁴ Cfr. VINCE, JOHN, opt. cit., p.283

⁷⁵ Cfr. DEL PINO GONZALEZ, L. M., op. cit., p.74-75

2.2.1.2 Dispositivo de control

Existen varios productos para *sen*sar en tiempo real los movimientos de la mano y retroalimentar la información proporcionada al espacio virtual. Un simple toque puede generarse con un pequeño y ligero transductor de baja frecuencia electromagnética, pero fuerzas más complicadas involucran un diseño cuidadoso para no ser un transductor demasiado grande y pesado que interfiera con los movimientos naturales del usuario.⁷⁶

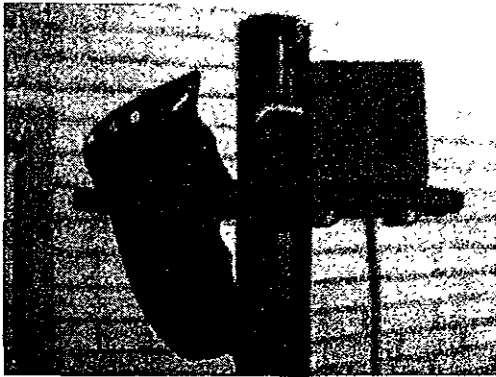


Figura 18. Joystick de Division

Entre los dispositivos de control tenemos un Division 3D Mouse, joystick con el cual es posible ofrecer al usuario un control de su punto de vista en el entorno además de la posibilidad de manipular los objetos y provocar algunas acciones al disparar eventos. Este dispositivo también tiene integrado el sistema de localización electromecánico de la compañía Polhemus.⁷⁷

Una unidad de control es la encargada de retroalimentar las entradas y resultados del sistema detector de posición y del Division 3D Mouse a la estación de trabajo a través de puertos seriales.⁷⁸

2.2.2 Dispositivos de Salida

En el HITL South, el diseñador cuenta con los siguientes dispositivos de salida:

⁷⁶ Cfr. VINCE, JOHN, *opt. cit.*, p.283

⁷⁷ Cfr. STAMPE, DAVE; ROEHL, BERNIE; EAGAN, JOHN, *op. cit.*, p.84-85

⁷⁸ Cfr. *dVISE* c.2, p.2

2.2.2.1 Dispositivo de Presentación

El casco de visualización es modelo VR4 de la compañía Virtual Research, es un ligero casco de 0.935 kg que presenta la imagen en dos pantallas de cristal



Figura 19. Casco de Visualización VR4

liquido mientras bloquea el contacto visual con el mundo exterior lo que produce un buen efecto de inmersión. Usando el casco de visualización la navegación por el ambiente es una operación natural.⁷⁹

No es estereoscópico porque no contamos con un equipo de computo que soporte dos tarjetas de video, ni dos equipos de computo; uno para cada display. Ciertamente no se puede prescindir de la visión

periférica pero se dice que con un campo de visión de 80 a 85 grados se obtiene una buena inmersión.⁸⁰

2.2.2.2 Dispositivo de Audio

El casco de visualización incluye un par de audífonos, lo que significa un punto a favor en vistas a conseguir que el usuario se sienta inmerso.

⁷⁹ Ibid, c.2, p.3

⁸⁰ Cfr. IOVINE, JOHN, Step into Virtual Reality, p.26, 35, 45; dVISE, c.2, p.3

2.2.3 Computadora

En el HITL South contamos con una estación de trabajo O2 de Silicon

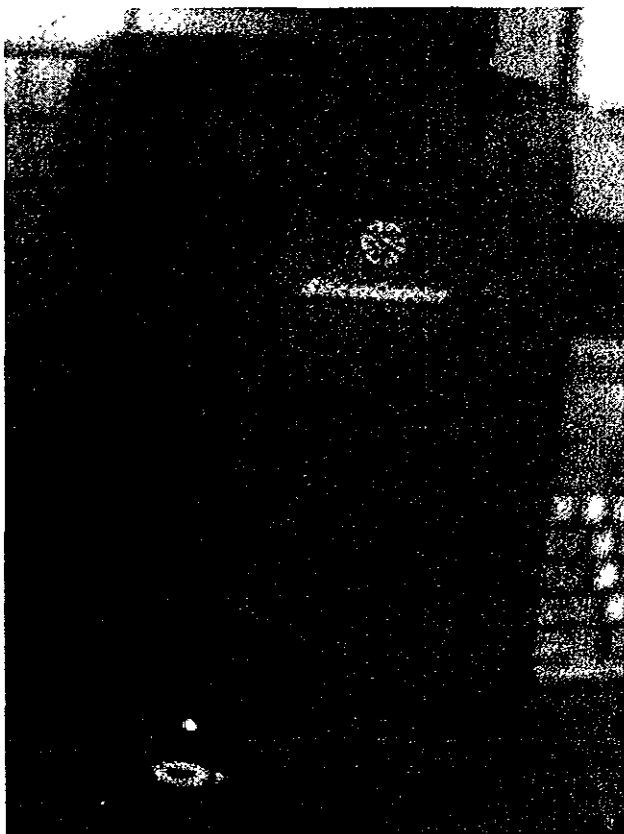


Figura 20. Estación de Trabajo O2 Silicon Graphics

Graphics. O2 es ideal para diseño tridimensional y animación al contar con un sistema que combina el performance de Silicon Graphics, poderoso procesamiento de imágenes, hardware integrado para la texturización y un procesador MIPS R5000 RISC, este procesador ofrece un alto desempeño en operaciones de punto flotante y enteros, cálculos directos de gráficos tridimensionales con múltiples instrucciones.

Tiene una arquitectura de 64 bits, su velocidad de procesamiento es de 180 MHz, su velocidad de rendero aproximada es de 380 triángulos por segundo y su arquitectura base tiene integrada la compresión.

O2 ofrece un increíble ancho de banda para acelerar los gráficos y las aplicaciones intensivas de cómputo.⁸¹

⁸¹ Cfr. <http://www.sgi.com/Products/hardware/desktop/products/index.html#O2> ,
<http://www.sgi.com/Octane/vrm!/LivingO2-PC/TechDocs/cpu.html> ,
<http://www.sgi.com/Chat/Live/faqo2b.html>

2.3 Software para un Sistema de RV

Existen tres métodos principalmente empleados para diseñar mundos virtuales:⁸²

- Uno es usando un lenguaje de programación para crear rutinas propias que conectadas definen el mundo virtual. Este es el método más difícil pues se requieren virtuosas habilidades de programación, mucha paciencia y constancia para los años de trabajo que lleva su construcción.
- Otra alternativa es comprar una herramienta de construcción de mundos que incluya rutinas prediseñadas creadas con un lenguaje de programación, de tal manera que se puedan combinar, modificar y conectar con gran variedad. Estas rutinas comunmente usadas son llamadas librerías y eliminan gran parte del arduo trabajo.
- Así como las computadoras y las capacidades de Realidad Virtual han progresado también los programadores han conseguido realizar este tercer método llamado herramienta de diseño o programa editor de mundos que, si es empleado de manera apropiada arroja efectivos resultados. Al igual que un programa procesador de texto, incluye funciones seleccionables que producen tareas que normalmente requieren un trabajo inmenso. Una herramienta de diseño de mundos virtuales posee un conjunto de herramientas representadas gráficamente y en tres dimensiones; al seleccionarlás y proporcionar la información necesaria, automáticamente se está elaborando el código de programación que define el mundo virtual.

⁸² Clasificación tomada de: HAYWARD, TOM, op. cit., p.181-182

2.3.1 VRML⁸³

VRML es la primer posibilidad de importar un modelo tridimensional para usarse dentro de Internet, por eso es llamado el estandar emergente de los mundos tridimensionales en la red; pero la verdadera RV no existe en Internet pues tan solo ofrece la creación de mundos tridimensionales y la posibilidad de entrar y navegar dentro de estos mundos sin inmersión. (Recordemos todas las caracterísiticas de un Sistema de RV)

VRML es el acrónimo de Virtual Reality Modeling Language aunque originalmente era Virtual Reality Markup Language. VRML no fue, ni es diseñado para crear mundos virtuales perfectos, además de que la programación a nivel escritorio no es una tarea fácil, ya que se necesita tomar en consideración distancias, relaciones espaciales, iluminación y ángulo de las camaras, entre otras cosas.

Su ventaja es ser independiente de la plataforma del sistema, su principal problema es el bajo ancho de banda en la red. Para ver los mundos virtuales en Internet se necesita un browser, un browser simplemente es un programa que permite ver e interactuar con el archivo de Internet; los browsers son sencillos de obtener de la red.

⁸³ Información de este apartado de: GORALSKI, POLI, VOGEL, VRML: Exploring Virtual Worlds on the Internet, p.102-104; WODASKI, RON, VR Madness 1996, p.802-804

2.3.2 SENSE8⁸⁴

Una popular herramienta de construcción de mundos es el llamado WorldToolKit de Sense8, constituye una librería de funciones en lenguaje C que hace posible la programación de mundos virtuales mediante la programación directa a los mismos, segundo método para diseñar mundos virtuales mencionado al principio de este apartado.

Se necesita ser un hábil programador en C para hacer que el mundo funcione. WorldToolKit hace posible el uso de objetos importados en los mundos virtuales además del mapeado de texturas y su naturaleza como librería de funciones la convierte en una herramienta que puede utilizarse con diversas plataformas e incluye drivers para varios periféricos. Basta comprar el paquete y las licencias adecuadas a las necesidades presentes.

2.3.3 DIVISION

Un sistema que permite editar o cambiar el mundo de manera fácil con funciones diseñadas para ahorrar tiempo y esfuerzo, es más accesible y más interactivo. Esto significa que se ahorra tiempo y energía para poder aprovechar este ahorro en diseños más productivos o para avanzar en el profundo y basto trabajo del diseño de mundos virtuales aún más especializados.

Sin embargo no por usar un sistema editor, cualquiera puede crear mundos virtuales, se necesitan amplios conocimientos del tema y no significa que con solo dar click al mouse se van a obtener mundos que posean todas las características que uno puede imaginar. Existen muchas ocasiones en que la función deseada no

⁸⁴ Cfr. HAYWARD, TOM, op. cit., p.92 y 182; STAMPE, DAVE; ROEHL, BERNIE; EAGAN, JOHN, op. cit., p.10-13; PIMENTEL, KEN; TEIXEIRA, KEVIN, Virtual Reality Through the new looking glass, p.61

está incluida en las herramientas proporcionadas y es cuando el diseñador tiene que programar los comandos que le hacen falta para completar su diseño.⁸⁵

dVISE de DIVISION es un editor de mundos virtuales que de manera sencilla ofrece la posibilidad de importar mundos tridimensionales, mapear texturas, es de plataforma independiente, reconoce diversidad de periféricos especializados para RV, su único inconveniente es el precio.

En el HITL South contamos con estos tres tipos de software gracias a la reciente donación de Sense8, a la compra hecha a DIVISION y VRML se puede obtener de la red. Por lo que ya vimos para la creación de un completo Sistema de RV solo podemos elegir entre los dos últimos tipos de software; WorldToolKit de Sense8 es nuevo para nosotros y aún estamos aprendiendo a usarlo, dVISE de DIVISION desde hace año y medio se ha estado usando y este es el software que utilizamos para la realización del proyecto de tesis aquí tratado.

dVISE es una muy buena opción donde se lleve a cabo la primer parte de nuestro objetivo: el diseño del espacio virtual de pruebas; ya que de manera óptima se logra su diseño, para la segunda parte de nuestro objetivo: contar con la interfaz que nos permite ejecutar los algoritmos del robot dentro del espacio virtual; dVISE posee una herramienta adicional que proporciona esta interfaz y estamos intentando conseguirlo.

De cualquier manera se sigue trabajando en el proyecto y hay personas encargadas se aprender Sense8 con el propósito de, mientras se consigue la interfaz de dVISE, elaborar el espacio virtual aquí diseñado en WorldToolKit de Sense8 para incorporarle la interfaz que nos comunice con los algoritmos de TX8. Pero eso es un proyecto aparte.

⁸⁵ Cfr. HAYWARD, TOM, *op. cit.*, p.182-183

ENTONCES ...

En el HITL South disponemos de poderosos recursos de cómputo y dispositivos de Realidad Virtual aptos para construir un completo Sistema de Realidad Virtual.

De entre los distintos programas de Software que tenemos elegimos dVISE por ser el SW compatible con los dispositivos de RV, por su ambiente amigable y por las ventajas presentadas al ser sujeto a pruebas de performance.⁸⁶

⁸⁶ Cfr. JACOBO, MAURICIO, Análisis Comparativo de Ejecución de Software de Realidad Virtual, p.55-57

CAPÍTULO 3

CREACIÓN DE UN SISTEMA DE REALIDAD VIRTUAL

Como mencionábamos al inicio de este trabajo, un Sistema de Realidad Virtual es la combinación de todo el potencial de una veloz computadora con gráficos tridimensionales enriquecidos con sensaciones del mundo real a través de dispositivos visuales, auditivos y de otro tipo para sintetizar un entorno ficticio interactivo en el cual el usuario se sienta inmerso.

Además cubre características como:

- ◆ Capacidad Sintética
- ◆ Tridimensionalidad
- ◆ Estereoscopía
 - ❖ Claves de Profundidad
- ◆ Interactividad o Manipulación
- ◆ Navegación
- ◆ Inmersión
- ◆ Punto de Vista

Si pretendemos construir un Sistema de Realidad Virtual, lo primero que necesitamos es crear todos y cada uno de los objetos virtuales que van a tomar parte en el espacio virtual, por supuesto también hay que crear dicho espacio virtual. Cada objeto virtual debe ser modelado con sus propiedades físicas necesarias; una vez creados los objetos virtuales se pueden situar en el espacio virtual para supervisar su comportamiento si existieran, ya aceptados; al espacio virtual se le sujeta a condiciones físicas modeladas matemáticamente para predecir su comportamiento general.

Una vez creado el Espacio Virtual, lo siguiente es integrar los dispositivos de RV; para conseguir cubrir las características de un Sistema de Realidad Virtual, como se mencionaba en párrafos anteriores.

3.1 DIVISION

Además del equipo de RV, en marzo de 1997 al HITL South llegó un software de Realidad Virtual de la marca DIVISION: dVS con su aplicación dVISE.⁸⁷

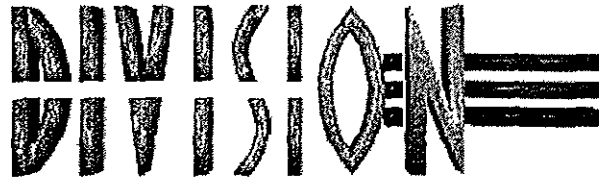


Figura 21. Logotipo de la Compañía DIVISION

3.1.1 dVS

dVS es la arquitectura distribuida de RV de DIVISION.

dVS provee y hace accesibles un conjunto de servicios de RV, que requieren todas las aplicaciones de RV como: dVISE, la herramienta de simulación y control de sistemas de RV.

Los servicios proporcionados por dVS están implementados por un conjunto de *servidores*. Un *servidor* es un programa responsable de una actividad específica del sistema, cada *servidor* está altamente optimizado para una tarea en particular con el fin de obtener el máximo performance del HW. La flexibilidad de dVS proviene de su arquitectura modular; porque está estructurado como una colección de servidores cooperativos de proceso, los llamados *servidores*; también conocidos como Actores. Éstos corren en paralelo, en paralelo entre ellos y en paralelo a la aplicación, lo que significa una ventaja en performance en una

⁸⁷ La información de todo este apartado es una recopilación de: VINCE, JOHN, *opt. cit.*, p.180, 283, 299, 307, 309; dVS, c.2 p.1-5; dVISE, c.2 p.5-7; http://indy1.adetti.iscte.pt/~visinet/aplicacoes/dVISE_i.html y <http://dval.larc.nasa.gov/DVAL/Whatsnew/dvise/>

máquina multiproceso; entre las tareas de los *servidores* está aislar al diseñador de todos los detalles de HW; lo que significa una gran ventaja al poder despreocuparse de los especializados y diversos periféricos que requieren los sistemas de RV.

Sabemos que las aplicaciones de RV tienen una significativa necesidad de poder de cómputo y procesamiento gráfico; esto por la necesidad de simplificar el proceso de definición de espacios virtuales complejos y por la necesidad de optimizar el performance de la presentación final. Por lo antes mencionado el objetivo principal en el diseño de dVS es crear un modelo distribuido para simulación de RV.

El modelo distribuido es aquel en el cual diferentes servidores pueden controlar diferentes elementos de todo el sistema. Se dice que las funciones de los *servidores* pueden dividirse en 3 grupos (como se muestra en la figura 22):

- ◆ Sensado del mundo real, por ejemplo la detección de posiciones
- ◆ Control del mundo virtual, como todo movimiento de los objetos
- ◆ Despliegue del mundo virtual, que es la presentación a través de los dispositivos.

Dado este modelo en el cual diferentes elementos de un todo se proveen por servidores separados, es necesario crear una infraestructura eficiente a través de la cual los servidores se puedan comunicar. En arquitecturas tradicionales de cliente/servidor, un conjunto de mensajes son definidos para encapsular todas las posibles funciones, entonces el cliente direcciona al servidor mediante llamadas locales o remotas. Sin embargo, esto no es escalable a sistemas multiservicio en los cuales diferentes partes de la interfaz son manejados por diferentes servidores; *tampoco las direcciones de memoria cubren las necesidades de un sistema*

multiusuario en el cual múltiples servidores necesitan arbitrariamente hacer cambios y mantener la consistencia al mismo tiempo.

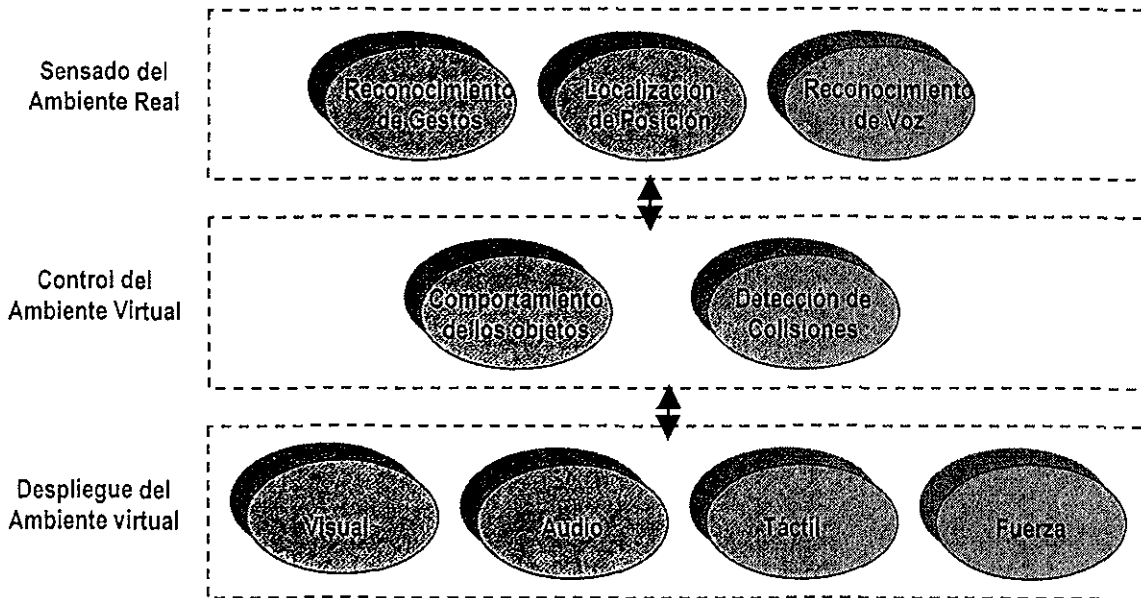


Figura 22. Los tres tipos de funciones de los *servidores*

Ante ello, dVS adopta un modelo más abstracto. En lugar de que cada servidor establezca una comunicación directa con los demás servidores, es creado un espacio de datos compartido en el cual cada servidor coloca los objetos compartidos llamados elementos. Diferentes servidores monitorean estos elementos y responden a cambios globales en el ambiente. Los servidores esencialmente se comunican a través de estos objetos compartidos y la estructura de estos objetos define la funcionalidad del sistema.

dVS provee un flexible ambiente de escritorio a base de ventanas para las aplicaciones de RV, dVISE. dVISE es una poderosa herramienta interactiva de control que permite a los diseñadores crear y a los usuarios interactuar con sistemas de RV complejos, sin necesidad de estar programando a nivel escritorio.

3.1.2 dVISE

dVISE está construido en la parte superior del ambiente dVS y está basado en un modelo simple donde el mundo virtual se compone de un número de objetos, cada uno con atributos⁸⁸ definidos por el usuario.



Figura 23. Logotipo de dVISE

dVISE es una herramienta de visualización y creación de sistemas de RV ampliamente utilizada, es un sistema de software que permite crear una representación virtual de una escena basada en el diseño CAD⁸⁹. Geometrías tridimensionales pueden ser importadas a dVISE para interactuar con ellas en un ambiente virtual. Es decir, los objetos pueden ser diseñados con modeladores CAD y después animados con dVISE. dVISE convierte los formatos tridimensionales CAD a su propio formato lo que proporciona la posibilidad de realizar paseos virtuales en escenarios tridimensionales diseñados con sistemas CAD.

dVISE soporta las relaciones de herencia entre objetos virtuales para trabajar con grupos de objetos que hereden propiedades.

Los formatos de los que se puede importar a dVISE son: AutoCAD archivos dxf, Autodesk 3D Studio archivos 3ds, MultiGen y MultiGen Flight archivos fit, Wavefront archivos obj y mtl.

Es a través de dVISE que el diseñador añade características auditivas, visuales y de comportamiento a dicha representación, es decir, dVISE modela la interacción entre los componentes, así como entre la escena y los componentes.

⁸⁸ Atributos son propiedades asociadas a los objetos tridimensionales, como color, restricciones de movimiento, comportamiento y características de audio. *dVISE*, c.2 p.5.

⁸⁹ CAD = Computer Aid Design, Diseño Asistido por Computadora

Dentro de los ambientes virtuales de dVISE un conjunto de herramientas pueden usarse para cambiar las propiedades de geometría como materiales, cambiar las condiciones de iluminación y para asociar eventos específicos como colisiones con fuentes de sonido.

Division, emplea un formato abierto para describir los objetos virtuales y sus atributos como: restricciones, propiedades físicas, comportamiento, geometría, materiales y sus relaciones de herencia. Esto es llamado Virtual Data Interchange format, es decir formato de intercambio virtual de datos, VDI.

dVISE provee dos modos de controlar o editar un mundo. Existe una interfaz tridimensional que permite al usuario modificar los objetos mientras se encuentra inmerso en el mundo virtual. La otra interfaz es una ventana de dos dimensiones donde los objetos están representados por iconos y son controlados usando el mouse y el teclado.

Los ambientes virtuales pueden ser desplegados en monitores convencionales y son manipulados con el mouse tradicional, pero en estaciones de trabajo Silicon Graphics, dVISE es capaz de generar secuencias en tiempo real con características como: texturas mapeadas e iluminación Phong, entre otras.

En dVISE todos los cálculos emplean unidades métricas del sistema inglés: longitud (metros), tiempo (segundos), masa (kilogramos), fuerza (newtons), torca (newton metros) y ángulos (radianes).

Al igual que dVS, dVISE emplea el concepto de *servidores* que son los responsables de las actividades del sistema:

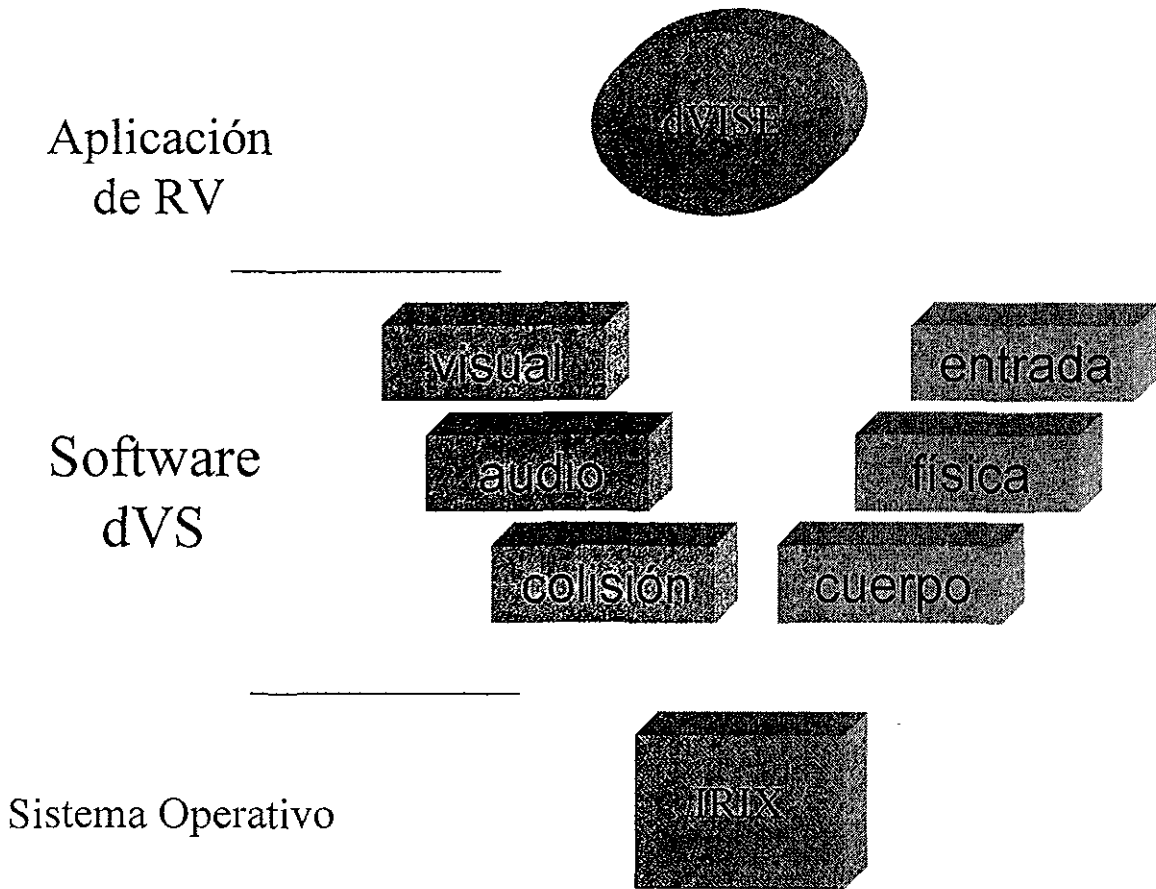


Figura 24. Distribución de los servidores de dVS que maneja dVISE

El servidor visual realiza la función rendering en todos los objetos virtuales del espacio virtual, decide cuales de los atributos de los objetos desplegar y puede realizar rendering a vistas estero con texturización en tiempo real.

El servidor de entrada lee las acciones del usuario y de los detectores de posición, de los cuales soporta un amplio rango.

El servidor físico usa la segunda ley de Newton⁹⁰ para modelar las fuerzas y el efecto de gravedad, monitorea las propiedades físicas de los objetos como inercia, fricción y masa gravitacional.

El servidor de colisiones monitorea todos los objetos virtuales del espacio virtual, define los bordes para detectar colisiones y los estados de colisión, además de que los actualiza después de cada colisión

El servidor de audio atiende todas las peticiones de audio asignadas a algún comando, monitorea la posición del usuario para proporcionar los detalles al HW que calcula el sonido estéreo.

El servidor de cuerpo interpreta las acciones del usuario, proporciona una interfaz entre el usuario y el espacio virtual, crea una entidad llamada cuerpo para representar al usuario, monitorea la posición del usuario actualizando los cambios de acuerdo a lo reportado por el servidor de entradas, también monitorea los bordes de la entidad cuerpo con ayuda del servidor de colisiones para saber cuando el cuerpo tuvo contacto con algún otro objeto virtual.

Los requerimientos de plataforma de dVISE son:

Plataforma	Sistema Operativo, Versión
SGI	IRIX 5.3(R3000) IRIX 6.2, 6.3, 6.4
HP	HPUX 9.05, 9.07, 10.01, 10.10 HPUX 10.20
Sun	Solaris 2.5.1
NT	NT 3.51, NT 4.0

⁹⁰ $F = ma$

Mas específicamente, los requerimientos de plataforma Silicon Graphics son:

Hardware

Mínimo: O2, 64 MB RAM

Recomendado: 128MB, Indigo Impact o Octane

Software

Mínimo: IRIX 5.3 o posteriores

Recomendado: IRIX 6.2 o posteriores.⁹¹

Nosotros cubrimos los requerimientos mínimos de Hardware, de acuerdo al equipo con que el HITL South cuenta y que enlistamos en el capítulo anterior.

Ahora nos dedicaremos a la creación del espacio virtual con ayuda de dVISE. El espacio virtual a construir es el HITL South, o también conocido como Laboratorio de Interfaces Inteligentes; este laboratorio está físicamente ubicado en la parte izquierda de la planta baja del edificio Baldés Vallejo.

Los pasos a seguir para crear un sistema de RV a partir de un espacio tridimensional son:

- Importar las geometrías conservando todos sus atributos visuales, sus nombres y sus herencias.
- Construir una representación que sea de plataforma independiente y que optimice la navegación en tiempo real, así como la interacción.
- Añadir propiedades de sonido, comportamiento y animación, a cada objeto; para darle vida al diseño.⁹²

⁹¹ Todos estos requerimientos se pueden consultar en:

http://www.ima.hk-r.se/ima_web/kursmtr/dVISE_instr/VERSION4/share/platforms.html

⁹² http://www.ima.hk-r.se/ima_web/kursmtr/dVISE_instr/VERSION4/dvreview/concepts/4rev_concepts.html

3.2 De Espacio Tridimensional a Espacio Virtual

Como mencionábamos párrafos atrás, una de las ventajas de dVISE es su compatibilidad con paquetes de diseño, lo que permite modelar los espacios tridimensionales de manera óptima para después importarlos a dVISE.

El modelo del laboratorio fue realizado en Alias/Wave Front⁹³ y se exportó a formato DXF⁹⁴. Después se importó a 3Dstudio para definir texturas y materiales, fue necesario agrupar todas las geometrías en objetos individuales para darles nombre y dejarlos preparados para la conversión de formato.

3.2.1 Convertidores⁹⁵

Ya mencionamos los formatos soportados por dVISE, sus correspondientes convertidores son:

- ◆ 3ds2vdi convierte de 3Dstudio los archivos con extensión .3ds
- ◆ dxf2vdi convierte de AutoCAD los archivos con extensión .dxf
- ◆ flt2vdi convierte de MultiGen y ModelGen los archivos con extensión .flt
- ◆ wf2vdi convierte de Wavefront los archivos con extensión .obj y .mtl

Todos los convertidores devuelven 3 tipos de archivos:

- Un archivo VDI (Virtual Data Information) para dVISE. VDI es un archivo ASCII que define el mundo virtual, con todos los objetos y todos sus atributos como apariencia y comportamiento, así como todas sus posiciones y orientaciones.

⁹³ Alias/Wavefront se encuentra corriendo en una computadora Silicon Graphics modelo Onyx Real Engine 2 del laboratorio de Visualización de Departamento de Supercómputo de DGSCA.

⁹⁴ Data eXchange Format utilizado por AutoCAD

⁹⁵ La información de este apartado es extracción de: Geometry Tools c.4 p.1-17 y c.5 p.9

- Para cada objeto existe un archivo en formato BGF (Binary Geometry File). BGF es un archivo en representación binaria, compacto y rápido de cargar en memoria, BGF define la apariencia visual de cada objeto. Su equivalente VGF es un archivo ASCII posible de editar.
- Un archivo de formato BMF (Binary Material File) para dVISE define el color y textura de la superficie de cada objeto, este archivo es una librería de todos los materiales. Su equivalente ASCII es llamado VMF y es editable.

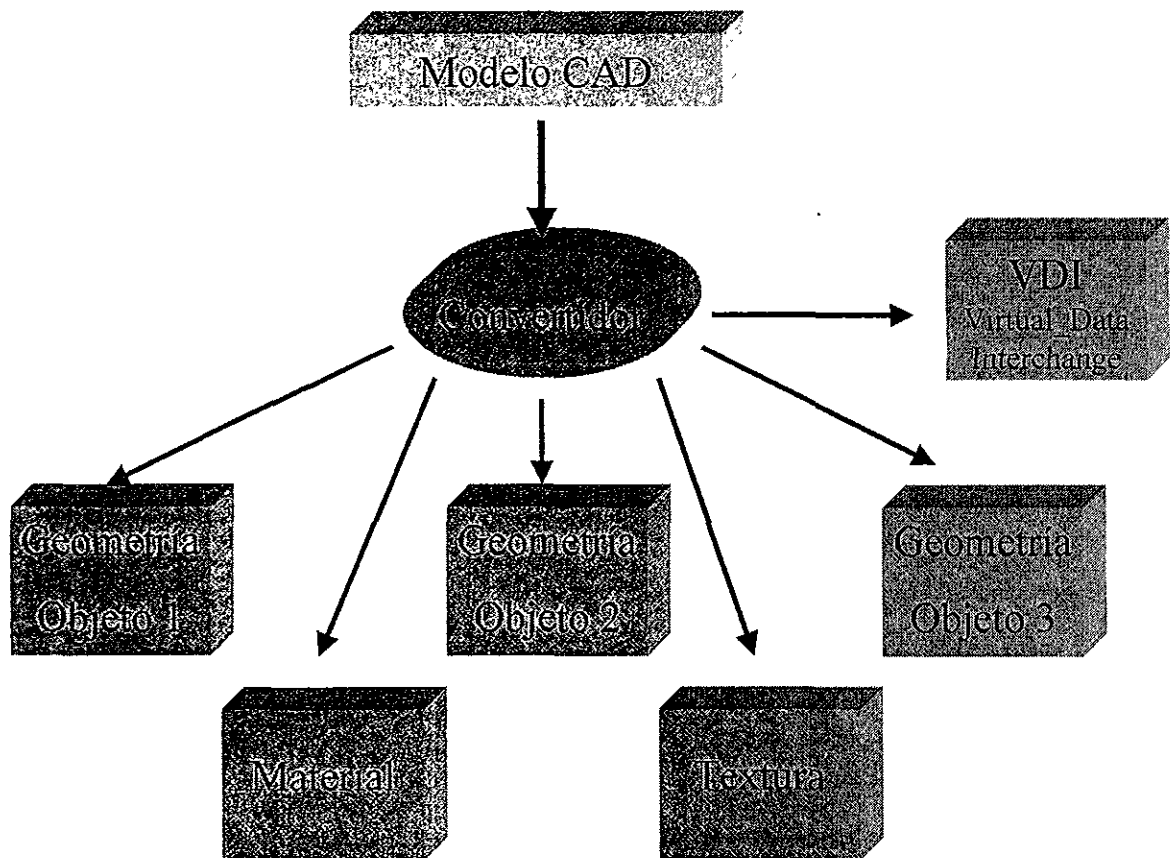


Figura 25. Resultados arrojados por el convertidor

Los formatos CAD describen varios objetos con un mismo archivo; los convertidores se encargan de colocar cada uno de estos objetos en un archivo por separado.

Para el Sistema de Coordenadas, dVS usa la regla de la mano derecha osea contrareloj, al igual que la mayoría de los sistema CAD, sin embargo, su conversión para el eje Z es de altura y no de profundidad. Los convertidores se encargan de hacer la rotación de -90 grados en X para estabilizar.

3.2.1.1 3ds2vdi

Los objetos de 3Dstudio son una colección de primitivas, a las cuales se asignan propiedades materiales como color y textura. A cada objeto se le asignó un nombre; éste nombre se conserva en el archivo VDI para poder ser utilizado en dVISE. Por cada objeto de 3DStudio se genera un archivo BGF. Cabe señalar que en AutoCAD se genera un archivo por cada capa o por cada color.⁹⁶

Bajo la plataforma dVS se teclea, en nuestro caso:

```
3ds2vdi hitlab.3ds ↵
```

y como resultado obtenemos, bajo el directorio actual cuatro subdirectorios con sus correspondientes archivos:

```
geometry
  hitlab1.bgf
  hitlab2.bgf
  □
  hitlabn.bgf; donde n es el número de objetos del espacio tridimensional

material
  hitlab.bmf
```

⁹⁶ Para mas detalle al respecto de cada convertidor consultar [Geometry Tools c.4](#)

texture
 nombre de la textura.vtx

vdifiles
 hitlab.vdi

3.2.1.1.1 img2vtx

Sabemos que las texturas añaden realismo a cada objeto y por ello son ampliamente utilizadas. Para mapear texturas a los objetos cada vértice debe incluir un conjunto de coordenadas en dos dimensiones (u,v). 3DStudio automáticamente genera estas coordenadas.

dVS soporta texturas del formato:

- .int SGI formato de intensidad
- .inta SGI formato de intensidad con alpha
- .rgb SGI color de 24 bits
- .rgba SGI color de 24 bits con alpha
- .tga TARGA RGB de 24 bits y alpha de 8 bits
- .vtx multi formato Division

El convertidor de texturas puede convertir archivos del formato⁹⁷:

- .bmp (Microsoft's Bitmap Format) mapa de bits de Microsoft Windows
- .gif (Graphics Interchange Format) formato gráfico de imágenes de CompuServer.
- .tiff (Tagged Image File Format) creado por Aldus y Microsoft, diseñado para importar imágenes en programas de escritorio.
- .jpg o JPEG (Joint Photographic Experts Group) formato de archivo que hace referencia al más popular método de compresión.

⁹⁷ Información de los formatos de: MORRISON, MIKE, The Magic of IMAGE Processing, p. 122-125

Las texturas aplicadas en 3DStudio es necesario convertirlas a formato .vtx, para ello, simplemente cada textura empleada se copia al subdirectorio texture y se teclea:

```
img2vtx -u nombre_de_la_textura.jpg
```

esto nos arroja como resultado el archivo nombre_de_la_textura.vtx cuadrado.

dVS respeta las deformaciones aplicadas a cada textura en 3DStudio, pero lo indica al momento de realizar la transformación enviando mensajes de Warning.

La resultante es un mundo tridimensional de formato Division, en el que ya se puede navegar; ahora resta darle vida a cada objeto y poder interactuar con el espacio.

Es dVISE la aplicación en la que completaremos los pasos para obtener un Sistema de Realidad Virtual, pasos que iremos realizando en el resto de los apartados de este capítulo.

3.3 Instrucciones Básicas de dVISE

Con el objeto de ejemplificar de manera clara las instrucciones de este apartado, presentamos las rutas de directorios de nuestro caso.

Directorio Raíz	/usr/people/acanche/
Directorio de Trabajo	/usr/people/acanche/tesis/
Dir. de ubicación del Mundo	/usr/people/acanche/tesis/tesis/

Partimos del conocimiento que el software ha sido instalado y configurado y que nuestra cuenta tiene los permisos de ejecución necesarios.

3.3.1 Cargar Mundo y Abrir dVISE

Para empezar a trabajar con el mundo tridimensional es necesario copiar el archivo `dvisedemos` a nuestro directorio raíz y editar la línea:

```
set demo_dir = "tesis";
```

donde `tesis` es el directorio bajo el cual se encuentran todos nuestros mundos virtuales.

También es necesario copiar el archivo `.dvsdemo` al directorio bajo el cual se creo el árbol del futuro espacio virtual. Las líneas a editar son:

```
set description = "Tesis dVISE: HITLab"  
set demo_dir_name = "tesis"  
set vdi_file = "hitlabf.vdi"
```

Como resultado obtenemos:

- i) el archivo modificado `.dvsdemo` bajo la ruta `/usr/people/acanche/tesis/tesis/`, este archivo crea la opción de utilizar nuestro mundo dentro del menú en el que activamos dVISE;
- ii) la copia del script `dvisedemos` bajo la ruta `/usr/people/acanche/`.

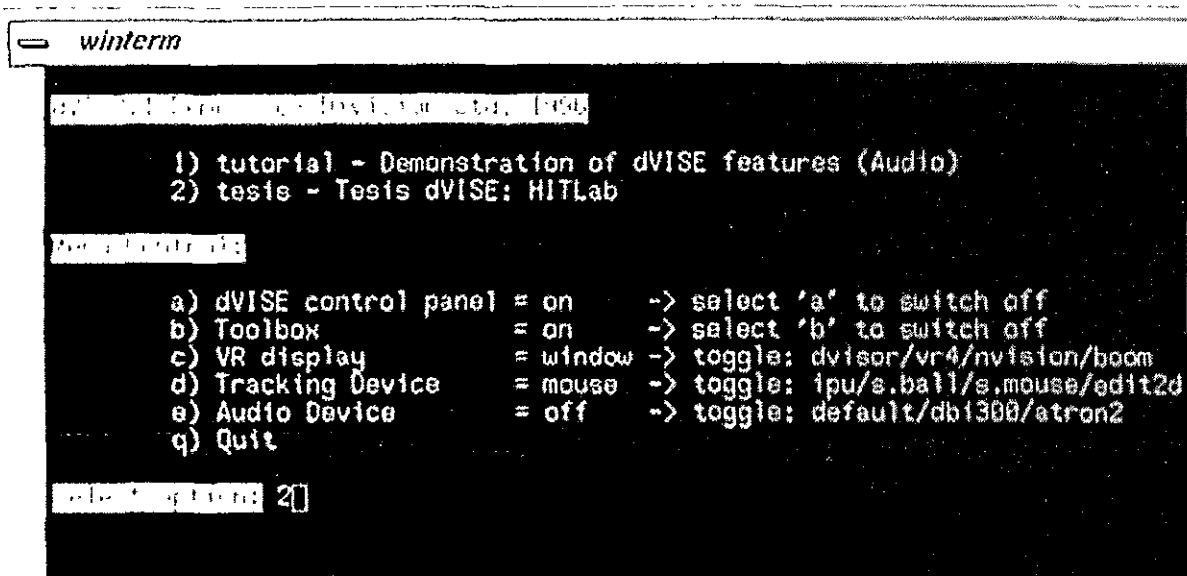
Estos pasos solo se necesitan llevar a cabo la primera vez que cargamos nuestro mundo.

Para desplegar el menú en el que abrimos dVISE basta con ejecutar el script copiado `dvisedemos` en la raíz de nuestra cuenta. La ventana resultante se muestra

en la Figura 26, se intercambian las opciones de configuración con ayuda del teclado hasta seleccionar:

- para dVISE control panel, on;
- para VR display, window y
- para Tracking Device, mouse.

Finalmente seleccionar el número correspondiente a nuestro espacio y teclear Enter.



```
winterm
dVISE v.1.0.0 (c) 2011, HITLab, HITLab

1) tutorial - Demonstration of dVISE features (Audio)
2) tesis - Tesis dVISE: HITLab

Personalización:

a) dVISE control panel = on    -> select 'a' to switch off
b) Toolbox              = on    -> select 'b' to switch off
c) VR display          = window -> toggle: dvisor/vr4/nvision/boom
d) Tracking Device     = mouse  -> toggle: ipu/s.ball/s.mouse/edit2d
e) Audio Device        = off    -> toggle: default/db1300/atron2
q) Quit

Welcome to dVISE: 2
```

Figura 26. Pantalla resultante del script dvisedemos

A partir de este momento comienzan a cargarse todas las texturas utilizadas y después de unos segundos aparece nuestro mundo virtual y nuestra interfaz de dVISE.⁹⁸

⁹⁸ En caso de tener problemas con el control básico del mundo consultar: [dVISE c.4](#)

3.3.2 Salir de dVISE

Para salir de esta aplicación basta con seleccionar la opción Exit del menú de opciones File que se encuentra dentro de la Barra de Menús del Control Panel⁹⁹ de dVISE.

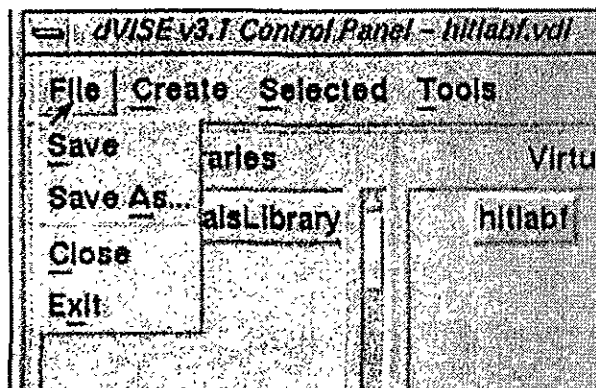


Figura 27. Ubicación de la opción Exit dentro de la Barra de Menús del Control Panel

3.4 Propiedades a cada Objeto

Continuando con los pasos necesarios para la creación del Sistema de RV, pasamos a darle vida a nuestros objetos tridimensionales importados. Para ello el siguiente apartado hablará de la interfaz a través de la cual vamos a controlar y a editar el mundo virtual.

3.4.1 Control Panel de dVISE

En párrafos anteriores hablábamos de una ventana de dos dimensiones donde los objetos son representados por iconos y se controlan usando el mouse y el teclado.

⁹⁹ La descripción del Control Panel de dVISE está en el apartado 3.4.1 de este trabajo.

El Control Panel es una ventana para crear y modificar ambientes virtuales, proporciona una interfaz de programación visual a base de ventanas, mediante la cual se pueden construir ambientes virtuales; se controla con los estándares menús, botones y cajas de diálogo. Tan pronto se realizan cambios se pueden observar sus efectos en el mundo virtual, los cambios realizados se graban solo cuando explícitamente se selecciona la opción Save del menú de opciones File que se encuentra dentro de la Barra de Menús del Control Panel de dVISE, lo que significa que podemos estar experimentando con los cambios sin alterar nuestro mundo original.

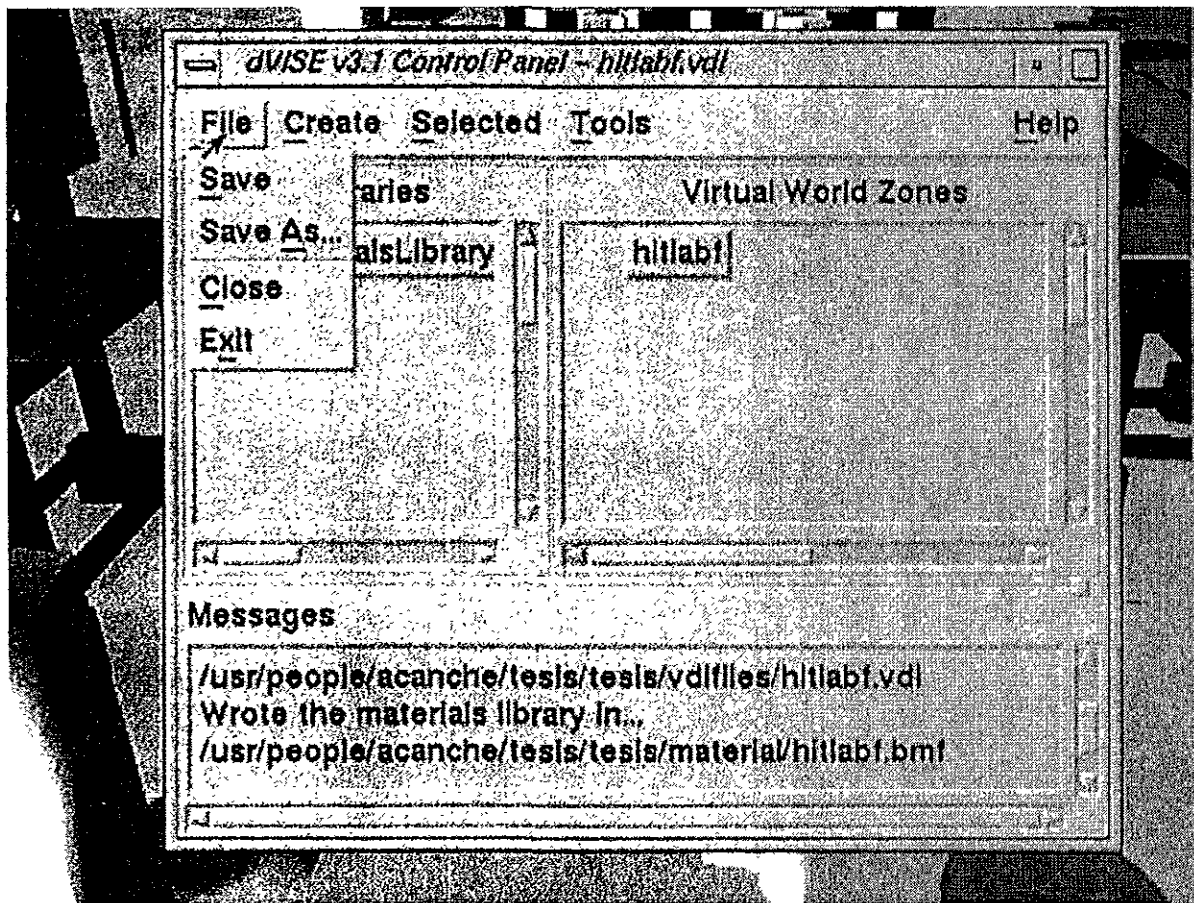


Figura 28. Control Panel de dVISE al frente y mundo virtual al fondo

El Control Panel está dividido en tres partes:

- En la parte derecha las Librerías
- En la parte izquierda las Zonas del Mundo Virtual
- En la parte inferior los Mensajes

El área de las Librerías contiene un icono por cada librería, cada icono despliega el nombre de la librería VDI que representa.

El área de las Zonas del Mundo Virtual también contiene un icono por cada zona del ambiente virtual, cada icono está etiquetado con el nombre de la zona que representa.

El área de Mensajes, como su nombre lo indica, es donde dVISE despliega información.

3.4.1.1 Zona

Para editar cada zona del ambiente virtual basta con seleccionar de la Zona del Mundo Virtual el icono que representa la zona deseada dando doble click con el mouse. La ventana resultante se muestra en la Figura 29.

La ventana de Zona muestra todos los objetos virtuales dentro de esta zona. Cada icono representa un objeto virtual. Esta Ventana de Zona posee como todas las ventanas su Barra de Menús, adicionalmente contiene una Barra de Herramientas donde se encuentran los botones que representan el Menú de Propiedades y es aquí donde vamos a agregar las propiedades a los objetos.

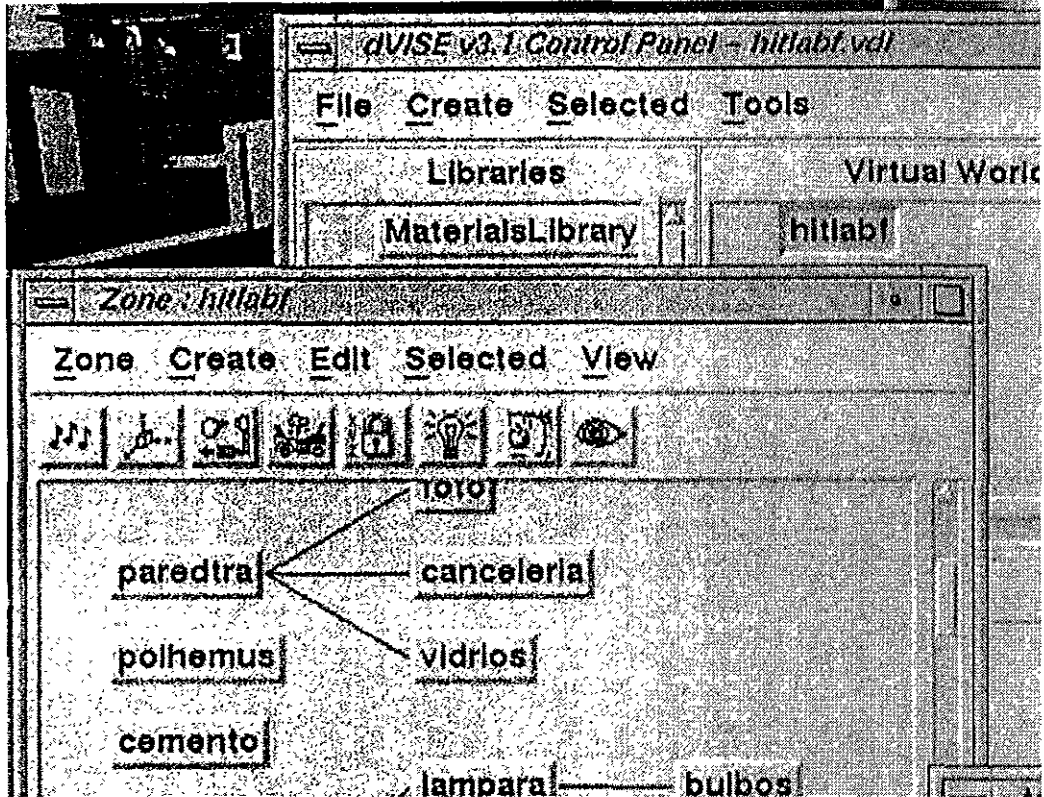


Figura 29. Ventana de Zona frente al Control Panel

Los botones de propiedades son ocho, para:

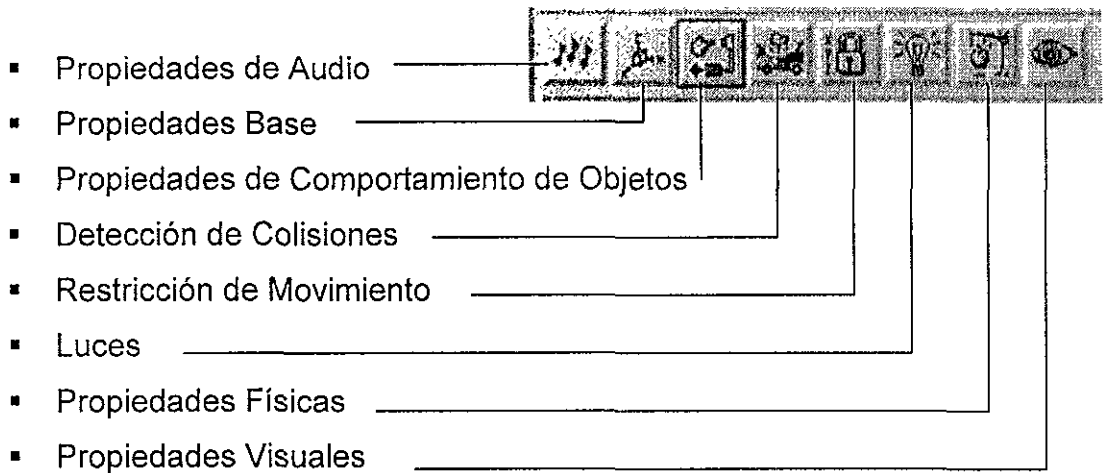


Figura 30. Botones de Propiedades

El mundo virtual consiste de una colección de objetos y fuentes de luz que son manipulados por procedimientos de simulación física, en paralelo con algoritmos de monitoreo de colisiones entre objetos específicos. El estado del mundo virtual está dado por las señales de entrada provenientes de cada dispositivo.¹⁰⁰

Para darle vida a nuestros objetos tridimensionales y convertirlos en objetos virtuales, es necesario agregarles propiedades como: herencia, restricciones, colisiones, masa y comportamiento; así mismo al mundo se le aplican propiedades como: iluminación, gravedad y restricciones. El orden en que se aplican estas propiedades no es un método definido, sin embargo, la práctica nos llevó a sugerir este orden.

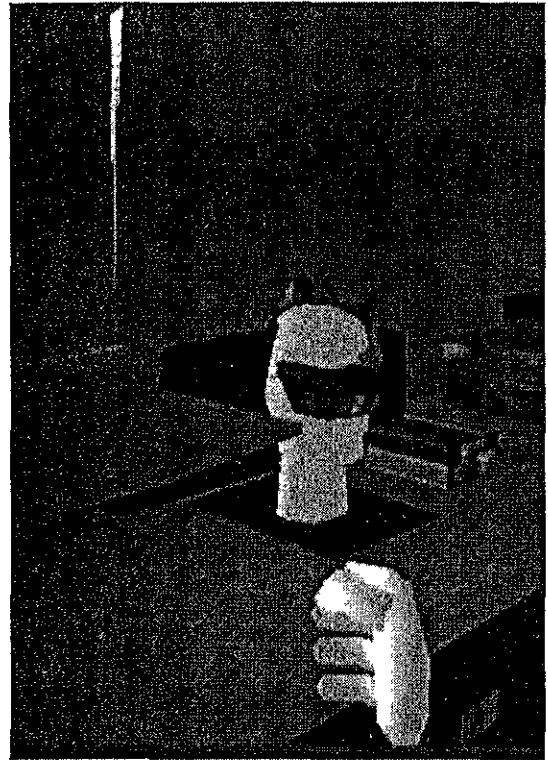


Figura 31. Vista del mundo virtual

3.4.2 Ligar

Las ligas se emplean por la necesidad de agrupar objetos y por la facilidad de heredar propiedades. Al ligar un objeto a otro se crean jerarquías entre ellos, lo que permite obtener resultados de una manera implícita, por ejemplo, al mover una mesa si los objetos que se encuentran encima de esta mesa están ligados a ella, al momento de desplazarla, los objetos se van a desplazar junto con la mesa, como sucede en el mundo real. Además se obtiene la facilidad de ahorrar asignación de propiedades a objetos puesto que el objeto padre ya las contiene.

¹⁰⁰ Cfr. VINCE, JONH, *opt. cit.*, p.168

Para lograr esto, en dVISE, existe una sencilla instrucción que nos da como resultado una liga entre dos objetos y como su función lo indica, recibe el nombre de *Link*. Esta instrucción se encuentra dentro de las opciones del menú Selected del Control Panel. Su método de empleo es:

a) Seleccionar el objeto a ligar, la selección de un objeto se hace al dar un click con el mouse sobre el icono que representa el objeto.

b) Activar la instrucción Link

c) Seleccionar el objeto al cual se quiere ligar el objeto del inciso a).

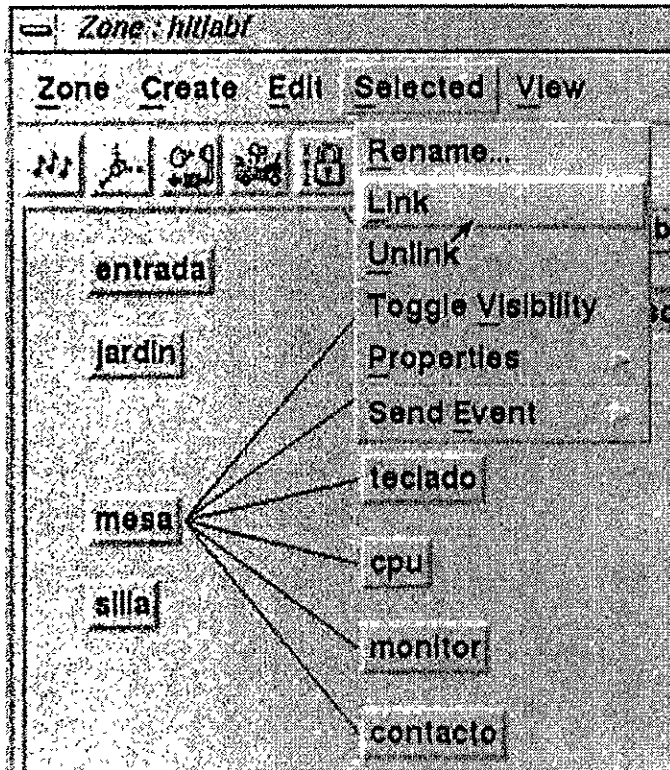


Figura 32. Ubicación de la instrucción Link

Por ejemplo, para ligar los bulbos con la lámpara y a su vez, la lámpara con el techo; en primera instancia vamos a ligar los bulbos a la lámpara, para ello seleccionamos los bulbos, activamos la instrucción Link y al seleccionar la lámpara automáticamente se crea la liga entre ellos dos. Posteriormente vamos a ligarlos al techo de manera tal, que seleccionamos la lámpara aunque dicha lámpara ya tiene un objeto ligado, activamos la instrucción Link y seleccionamos el techo para crear la liga entre la lámpara y el techo. Como resultado obtenemos los bulbos ligados a la lámpara y la lámpara ligada al techo.



Figura 33. Objetos Ligados

En el Anexo A se presenta el resultado de estas ligas en el archivo VDI.

3.4.3 Luces

La iluminación en el mundo virtual puede realizarse tomando en cuenta que en el mundo real las fuentes de luz no se mueven. Así solo se ilumina lo necesario agregando fuentes de luz y con intensidad, por ejemplo, se simulan diferentes momentos del día.

Los objetos importados ya no necesitan mas luces.

Para modificar la luz ambiental del mundo, en dVISE:

a) Se selecciona el icono que la representa.

b) Se presiona el botón de Luces.

c) Se seleccionan las opciones: State: On

Type: Ambient¹⁰¹

Color (RGB): 0.5 0.5 0.5

d) Al aceptar estas opciones automáticamente se modifica la luz Ambiental del Mundo

En el Anexo A se presenta el resultado de modificar la Luz Ambiental en el archivo VDI.

Si se requiere aplicar otro tipo de luces, el procedimiento es básicamente el mismo.

¹⁰¹ Los tipos de luces son tema del apartado 1.6.1

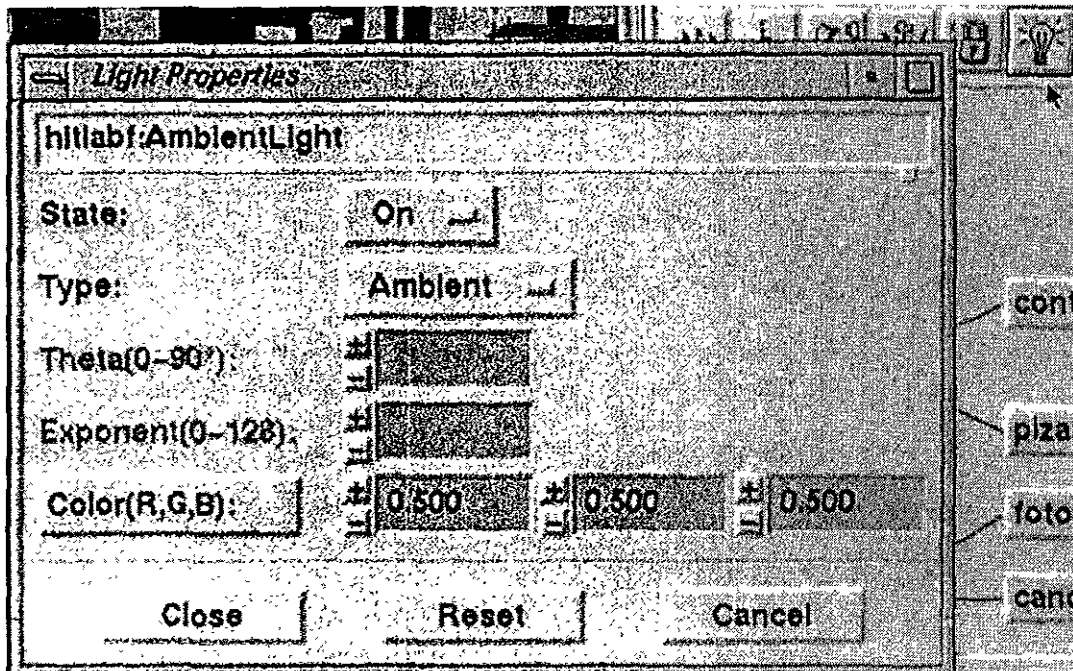


Figura 34. Ventana para editar las luces

3.4.4 Restricciones

En general, los objetos dentro de un Espacio Virtual se dividen en dos grupos: estáticos y dinámicos. Los objetos estáticos son pisos, paredes, techos, escaleras, etc. Cuando se construye un espacio, se debe indicar cuales objetos pueden moverse y cuales no, para evitar anomalías geométricas. Además algunos objetos dinámicos necesitan restricciones físicas de movimiento con el objeto de ponerles límites de translación y/o rotación sobre los ejes locales de referencia.¹⁰²

Para aplicar restricciones en dVISE:

- a) Se selecciona el objeto al cual se van a aplicar las restricciones.
- b) Se presiona el botón de restricciones.

¹⁰² Cfr. VINCE, JONH, *opt. cit.*, p.169-170

c) Se seleccionan las opciones: State: On

Constrain Type: Position & Orientation.

Y las necesarias para cada caso.

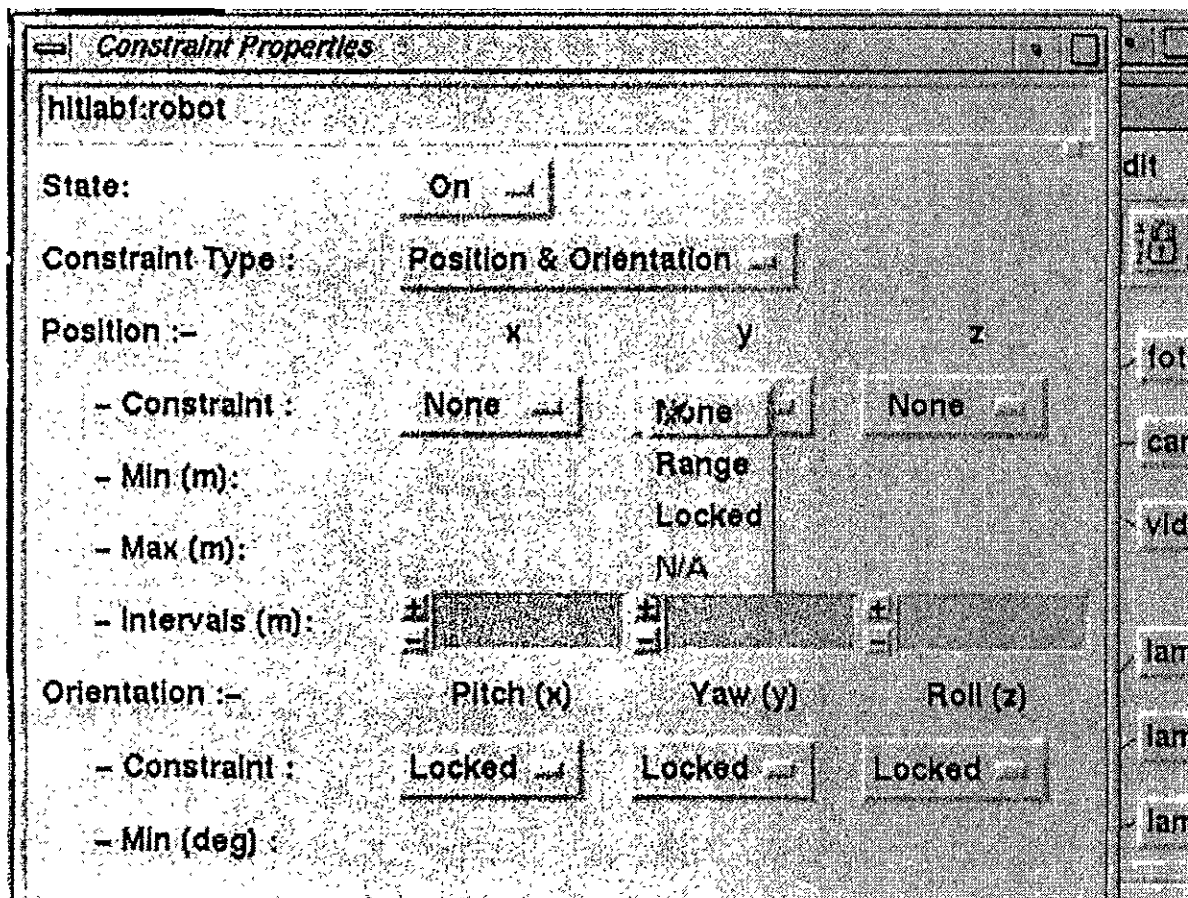


Figura 35. Ventana para aplicar Restricciones a los objetos

Por ejemplo, para evitar que las sillas sufran rotación y forzar que solo se deslicen sobre el piso al moverlas; aplicaremos las restricciones a Yaw, Pitch, Roll¹⁰³ y a los ejes X y Z, esto se hace seleccionando Locked en ellos y al eje Y lo dejaremos sin restricciones, None.

¹⁰³ Cualquier duda consultar el apartado 1.6.2

En el Anexo A se presenta el resultado de aplicar estas restricciones a la silla en el archivo VDI.

3.4.5 Colisiones

La detección de colisiones entre objetos es necesaria para conseguir operaciones básicas como tomar o arrastrar objetos.

Para activar las colisiones a un objeto, en dVISE:

- a) Se selecciona el objeto.
- b) Se presiona el botón de Colisiones.
- c) Se enciende la opción State.
- d) Al aceptar esta selección presionando Accept, automáticamente ese objeto es detectable a otros objetos.

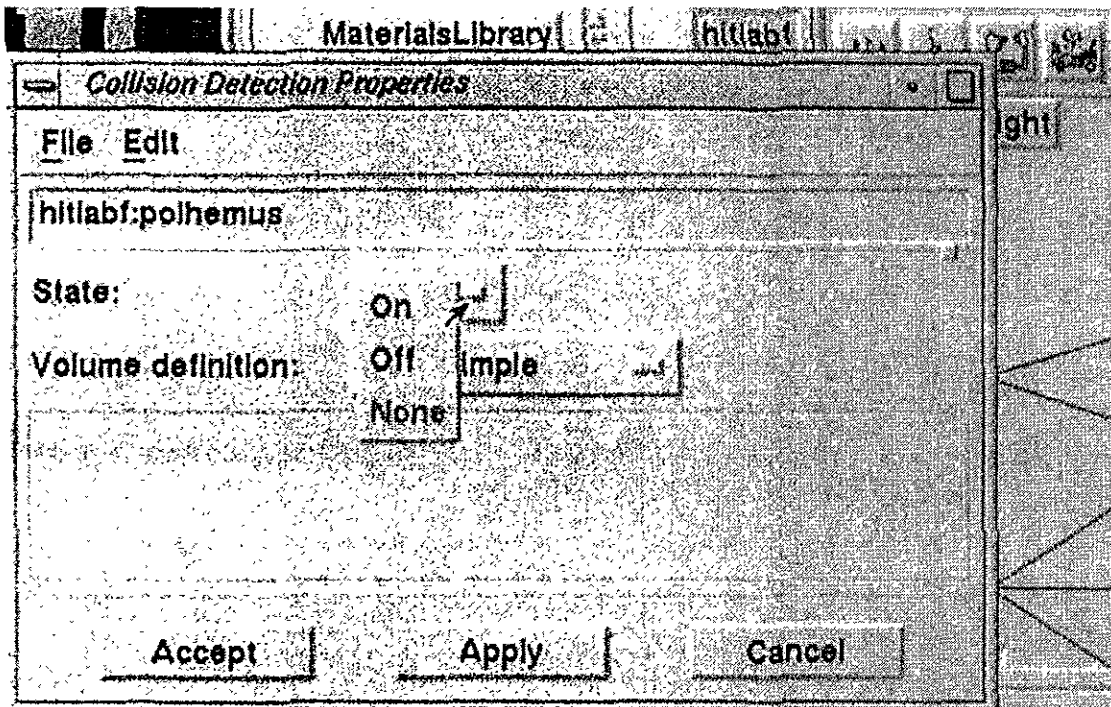


Figura 36. Ventana para activar las Colisiones

En la Figura 36 se activan las colisiones del objeto Polhemus para que en adelante, pueda tomarlo y moverlo el usuario. Resultado en Anexo A.

3.4.6 Comportamiento

Es en este momento cuando vamos a activar la gravedad en el mundo, vamos a asociarle a cada elemento una cantidad de masa para que caigan y además vamos a programar eventos que disparen determinadas acciones.

Para activar la gravedad en el Espacio Virtual se selecciona la opción Properties del menú de opciones Zone que se encuentra dentro de la Barra de Menús de la ventana de Zona. Y en la ventana resultante se coloca ON en la propiedad Gravity y se teclea 9.81 en la propiedad Gravity (m/s^2).

Para asociarle una cantidad de masa a cada objeto, tan fácil como al activar las colisiones:

- a) Después de seleccionar el objeto
- b) Presionar el botón de Gravedad
- c) Indicar la masa para el objeto y por default se asigna velocidad de 9.81 (las unidades m/s^2 ya están definidas en la zona). Al aceptar estas opciones automáticamente el objeto está sujeto a la Segunda Ley de Newton.

Para programar eventos, en dVISE:

- a) Como siempre, se selecciona el objeto.
- b) Se presiona el botón de Comportamiento.
- c) Se selecciona el evento deseado. La lista de eventos aparece al dar un click sobre el botón Añadir de los Eventos.
- d) Una vez que se introdujo el evento se procede a asignarle las acciones a realizar. Estas acciones aparecen enlistadas al dar un click sobre el botón Añadir de las Acciones. Cada acción necesita información adicional, ésta puede ser

añadida de manera sencilla y específica cuando al dar doble click sobre la acción aparece una ventana que presenta toda la información necesaria y los espacios para teclear o seleccionar dicha información.

e) Al aceptar toda esta información la acción ya está programada. Podemos asociar varias acciones a un mismo evento al repetir el inciso d).

Inclusive a un mismo objeto se le pueden programar varios eventos y esto se consigue repitiendo los incisos c) y d).

f) Al aceptar los eventos automáticamente están asociados al objeto.

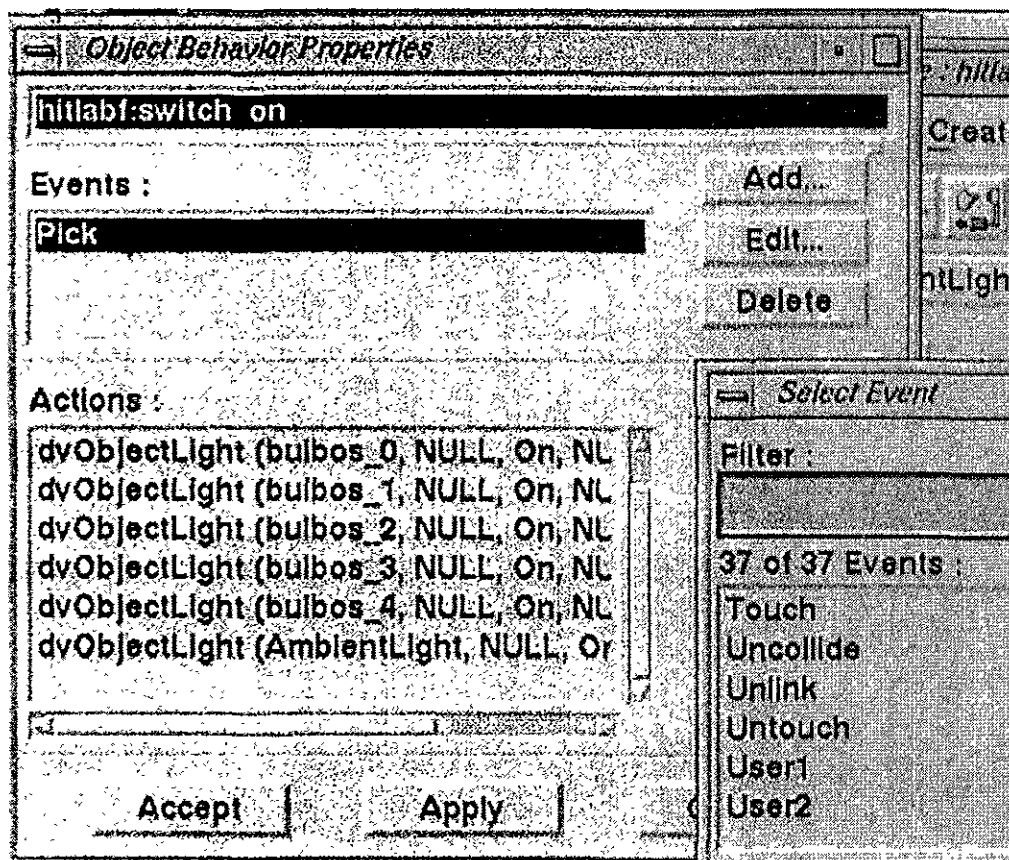


Figura 37. Ventanas para programar Eventos a los objetos

El resultado de activar la gravedad en el Mundo Virtual, asociar masa a un objeto y programar eventos a, por ejemplo, el switch para encender la luz en el Laboratorio Virtual se enlista al final del Anexo A.

El diseñador; de acuerdo al Mundo Virtual que desee programar va a decidir que propiedades debe llevar cada objeto. En nuestro caso particular, para el diseño del HITL South el resultado final en el archivo VDI, del Mundo Virtual programado se enlista en el Anexo B y tres imágenes del Espacio Virtual conseguido se presentan a continuación, de acuerdo al esquema:

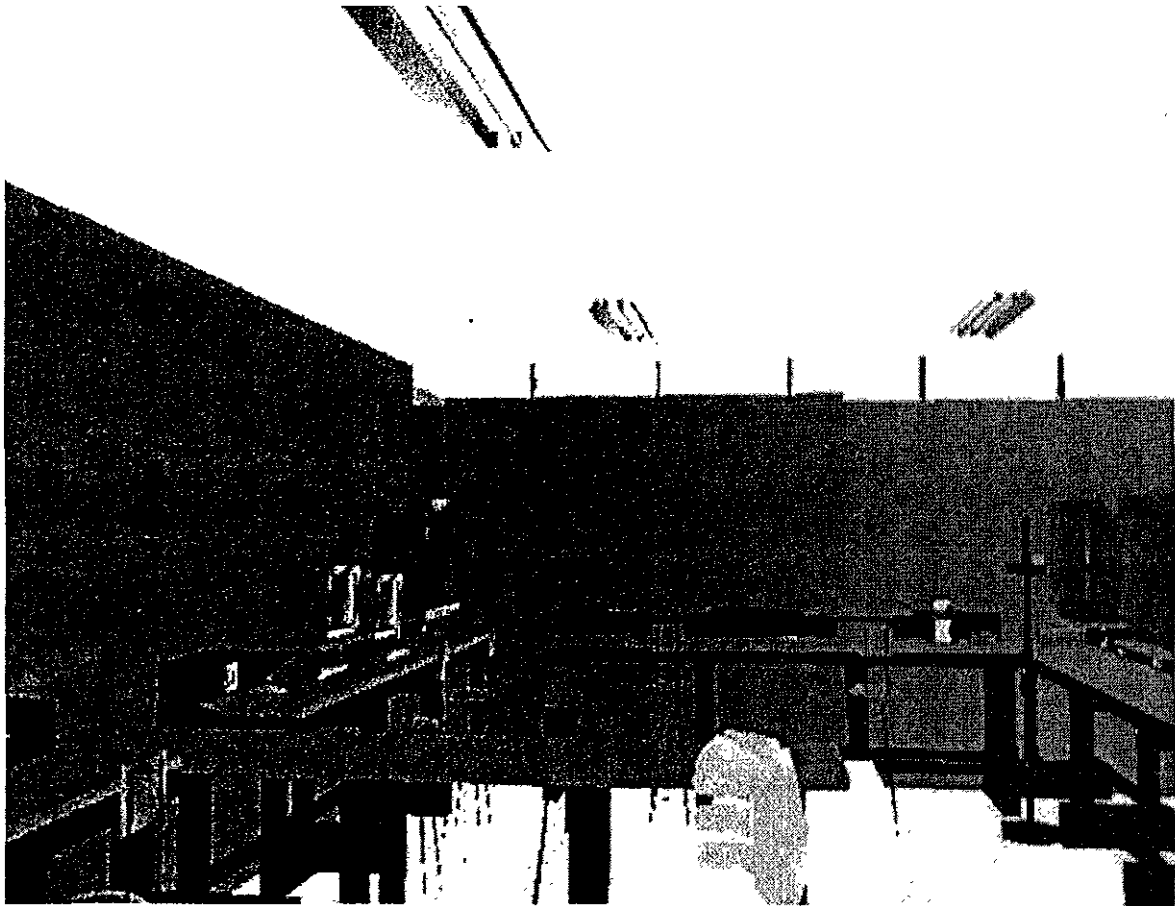
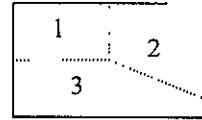


Figura 38. Imagen 1 del mundo virtual donde observamos las mesas y sillas de trabajo del HITL South con dos computadoras a la izquierda, tres televisiones, dos a la derecha y una a la izquierda; al frente la estación de trabajo, el casco de visualización y el dispositivo de localización.



Figura 39. Imagen 2 del mundo virtual del HITL South donde observamos además de los dispositivos de RV, la salida, el apagador, el pizarrón y otra computadora.

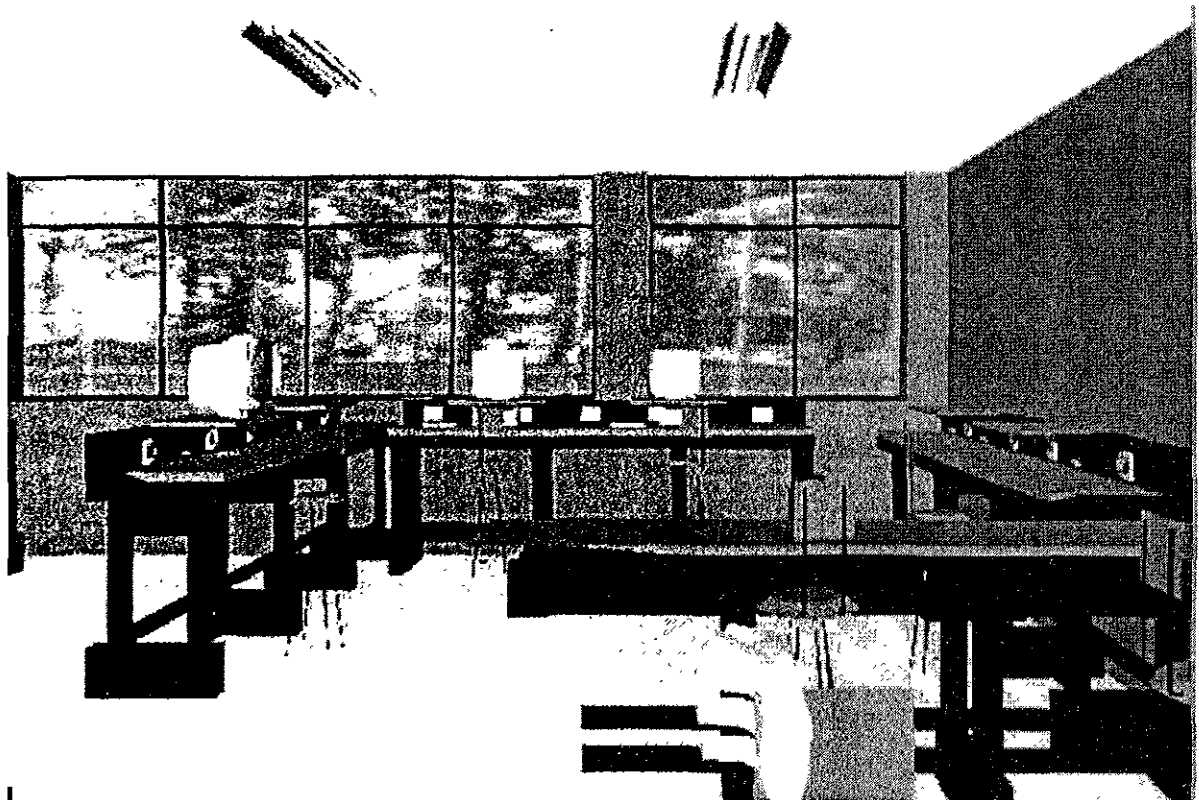


Figura 40. Imagen 3 del HITL South virtual donde observamos el resto de las mesas, sillas y computadoras del laboratorio y al fondo el ventanal.

3.5 Activar Periféricos

Para activar los periféricos propios del Sistema de RV, en nuestro caso, como dispositivos de entrada tenemos: el dispositivos de localización, Polhemus Fastrack y como dispositivos de control, Joystick de Division. Como dispositivo de salida: el dispositivo de presentación y de audio, el casco de visualización VR4. Una vez encendidos, basta con ejecutar el script dvisedemos en la raíz de nuestra cuenta y en la ventana resultante (ver Figura 26) intercambiar las opciones de configuración con ayuda del teclado. Seleccionar para dVISE control panel: on; para VR display: vr4; para Tracking Device: edit2d y finalmente activar el mundo virtual.

De esta manera una vez que se cargo totalmente el mundo virtual, nos sumergimos al colocarnos el casco de visualización y estamos aislados del mundo real, lo que permite crear una ilusión de realidad; el control para manipular el mundo virtual lo tiene la mano virtual que aparece en la pantalla, esta mano es controlada por nosotros mismos a través del joystick. Cada movimiento es captado por el dispositivo de localización y el resultado es presentado en el casco de visualización en tiempo real para poder navegar e interactuar con libertad.

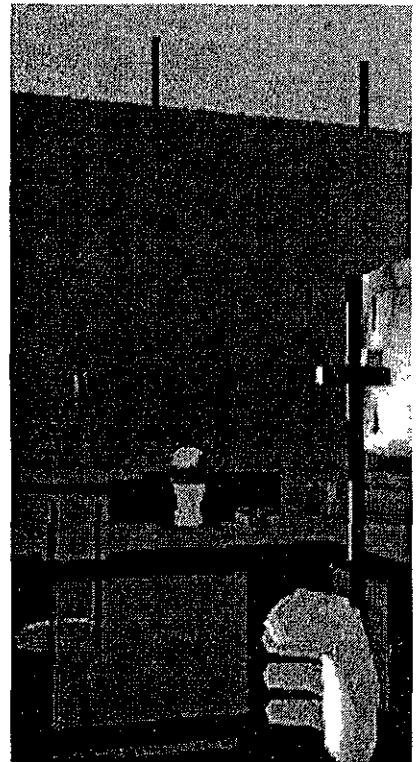


Figura 41. Mano con la que se controla el mundo virtual

3.5.1 Tool Box

Además del Control Panel de dVISE, existe una interfaz tridimensional llamada Tool Box, a través de la cual se permite, como con el Control Panel, modificar en tiempo real, cada uno de los objetos así como el mismo espacio

virtual, pero con la diferencia de que todas estas ediciones se realizan estando inmersos en el mundo.

El Tool Box, se activa desde el script `dvisedemos` con el cual abrimos `dVISE` con la opción de configuración `Toolbox` al ponerla en `on`, y junto con las demás opciones del apartado 3.5, ya que debemos estar inmersos para poder controlar el Tool Box con el joystick.

No es tema de este trabajo profundizar en el uso de esta interfaz tridimensional de edición por la naturaleza del Sistema de RV; sin embargo para Sistemas que pretendan colaborar con la creación de prototipos o con el diseño de vistosos espacios es una muy eficaz herramienta de trabajo.¹⁰⁴

¹⁰⁴ Para conocer la función de los botones del Tool Box consultar: [dVISE](#), c.5 y c.7

ENTONCES ...

El objetivo planteado ya se consiguió, el Mundo Virtual diseñado, desplegado a través del casco de visualización y manipulado a través del joystick, es un completo Sistema de Realidad Virtual.

El éxito de este trabajo, es conseguir que cada parte trabaje en conjunto, comunicándose para devolver al final, lo que el usuario espera y como vimos, esto se consiguió de una manera óptima al crear este Sistema de RV con la herramienta de diseño o programa editor de mundos virtuales, dVISE.

CONCLUSIONES GENERALES

Ya indicamos las características que un Sistema de Realidad Virtual debe cumplir, a manera de repaso y con el propósito de verificar una a una su cumplimiento, las volvemos a mencionar:

- Capacidad Sintética. Nuestro Espacio Virtual se genera en tiempo real, no es una simple proyección o animación previamente elaborada. ✓
- Tridimensionalidad. Nuestro Espacio Virtual está hecho de objetos diseñados en tres dimensiones, en su totalidad. ✓
- Estereoscopia. Desgraciadamente no contamos con la capacidad de presentar una imagen por pantalla en el Casco de Visualización. ✗
 - Sin embargo se cumplen todas las Claves de Profundidad. ✓
- Navegación. Nuestro Espacio Virtual es navegable en su totalidad, basta desplazarse dentro de él con ayuda del mouse o al estar inmerso con el joystick. ✓
- Interactividad. El estado de este Espacio Virtual está dado por las entradas que proporciona el usuario, ya que los objetos se pueden manipular al tomarlos con el joystick. Esto quiere decir que entre el usuario y el Espacio Virtual existe interacción mutua, de acuerdo a lo que el usuario hace a través de los dispositivos de entrada, la máquina procesa estas entradas, genera la resultante y lo presenta a través de los dispositivos de salida. ✓

- Inmersión. Esta se consigue con ayuda del Casco de Visualización, el Espacio Virtual se presenta a través de éste lo que permite aislar al usuario del Mundo Real para que pueda, después de unos instantes, experimentar la sensación de estar dentro del Espacio Virtual. ✓

- Punto de Vista. Gracias al sistema de detección de posición en todo momento conocemos la posición y orientación del usuario para generar las salidas correspondientes. ✓

Este trabajo posee un valor académico inmenso puesto que es el resultado del esfuerzo conjunto del Dr. Jesús Savage y de varios estudiantes, ahora Ingenieros que consiguieron obtener el equipo y la aplicación de RV, instalaron el software, conectaron y comunicaron el equipo de RV con la aplicación, en este caso, dVISE y finalmente me apoyaron para la realización del presente trabajo, el cual, es el punto de partida de todo un proyecto de RV que se está desarrollando en el HITL South, siendo éste el siguiente paso que daremos como Equipo de RV en el laboratorio.

Para la realización del presente trabajo conté con mucho apoyo; se me facilitaron las herramientas necesarias para su culminación, se me asesoró para avanzar sin dificultades mayores y conseguir el resultado ya expuesto. En contra parte, sin llegar a llamarles impedimentos, las necesidades que se presentaron y que persisten hasta el día de hoy, fueron la falta de un equipo capaz de presentar una imagen distinta para cada lente del casco de visualización y así presentar una visión estereoscópica, otra fue la falta del software complementario de dVISE por medio del cual se comunican los Mundos Virtuales a rutinas del Lenguaje de Programación C y con lo cual podríamos controlar el robot virtual que se presenta dentro del Sistema.

Yo recomiendo esta herramienta de diseño de mundos virtuales, dVISE, para aplicaciones que requieran muy buena presentación, su método es sencillo y hasta motivante para quienes realizan su primer Sistema de Realidad Virtual.

Queda entendido que este trabajo representa el primer Sistema de RV, completo, de la Facultad de Ingeniería; sin embargo esto es solo el inicio del camino. La construcción de este primer Sistema de RV se consiguió empleando los recursos con los que hasta ahora cuenta el Laboratorio; pero se obtendrían todavía mejores y más productivos resultados añadiendo algunas herramientas: si conseguimos que los procesos corran en la SuperComputadora de la UNAM, Berenice 32, estaríamos optimizando el esquema de servidores que emplea dVS y dVISE, que como ya vimos es un sistema distribuido, con lo que la O₂ del Laboratorio funcionaría solo como consola de RV y esto restaría toda la latencia que hasta ahora se presenta y que como definimos en el primer capítulo se traduce en mayor Ilusión de Realidad y mayor Inmersión para el usuario. Para esto lo que se necesita es una cuenta en Berenice 32, suficiente espacio para nuestros Sistemas de RV, pero sobre todo, una conexión de red con un ancho de banda que no se vea afectado por factores del medio o de demanda excesiva; porque de lo contrario la latencia reducida se transforma en grandes tiempos de espera de respuesta de la red. Por otra parte, existe una herramienta de dVISE que nos permite ejecutar rutinas escritas en C dentro del mundo virtual construido, esto significa que se amplía el número de propiedades a asignar al mundo y a cada objeto, por existir la posibilidad de programar propiedades nuevas y diversas y esto se traduce en Sistemas de RV con aplicaciones no solo de Diseño y Presentación sino aplicaciones de Medicina e Ingeniería Avanzadas, entre otras. Claro que actualizar las versiones del software de RV con el que contamos también representa una ventaja para mejores Sistemas de RV. Como vemos, hablando de Sistemas de RV aún existe mucho que abarcar, la Facultad de Ingeniería puede realizar grandes proyectos de RV con unas cuantas herramientas más.

En lo personal, la Realidad Virtual es una poderosa herramienta que nos proporciona la posibilidad de explorar y experimentar mundos y sensaciones ficticios, lo que representa un gran avance en la ciencia; y esto significa responsabilidad del diseñador; hacer buen uso de la ciencia, puesto que la tecnología está para servir, no para esclavizar al hombre, ser racional que posee inteligencia y voluntad para trabajar por el bien de sí mismo y de sus hermanos. Por ningún motivo debemos permitir que la tecnología, en este caso un Sistema de Realidad Virtual, sustituya la capacidad de reflexión del hombre; el éxito de un avance tecnológico no es su deslumbrante presentación si para conseguirlo se valió de sensacionalismos, deformaciones o desprestigios; ante todo debe reinar el espíritu ético y humanista del diseñador basado en una profunda racionalización para ser transparente y honesto. Como ingenieros diseñadores, es lógico que utilizaremos todos los elementos a nuestro alcance, pero siempre en beneficio de nuestro prójimo, respetando los valores humanos, valores con que todavía cuenta nuestro país. Y si, al ser veraces y justos nos van a atacar, precisamente va a ser nuestro reto, porque con un Sistema de Realidad Virtual podemos hacer mucho bien a la humanidad.

ANEXO A

Resultado de ligar los bulbos con la lámpara y la lámpara con el techo:

```
Object (Name=techo) {
  Position {-2.25994, 1.27022, -2.51659}
  Visual {
    Geometry {"hitlabf21"}
  }
  Object (Name=lampara) {
    Position {2.578, 0.095, -1.131}
    Visual {
      Geometry {"hitlabf27"}
    }
    Object (Name=bulbos) {
      Position {0.00173008, -0.00171995, 0}
      Visual {
        Geometry {"hitlabf26"}
      }
    }
  }
}
```

Resultado de asignar luz ambiental:

```
Object (Name=AmbientLight) {
  Light {
    Type {Ambient }
    State {On }
    Colour {1, 1, 1}
  }
}
```

Resultado de aplicar restricciones a la silla:

```
Object (Name=silla) {
  Position {-0.383208, -0.400694, -3.83859}
  Visual {
    Geometry {"hitlabf32"}
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
}
```

Resultado de activar las colisiones del objeto polhemus:

```
Object (Name=polhemus) {
  Position {-1.97188, 0.105214, -1.06705}
  Visual {
    Geometry {"hitlabf11"}
  }
  Collision {
    State {On }
  }
}
```

Resultado de activar la gravedad al mundo virtual:

```
Zone (Name=hitlabf){
  Physical {
    Gravity {0.81, 0, -1, 0}
    State {On }
  }
}
```

Resultado de asignar masa al objeto robot:

```
Object (Name=robot) {
  Position {0.99383, -0.95003, -6.6508}
  Visual {
    Geometry {"hitlabf18"}
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
}
```

Resultado de programar el evento de encendido de los bulbos de la lámpara al switch:

```
Object (Name=switch_on) {
  Position {1.24, 0.001, 3.526}
  Visual {
    Geometry {"hitlabf28"}
  }
  Event {
    Pick {
      dvObjectLight(bulbos, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_0, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_1, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_2, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_3, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_4, NULL, On, NULL, NULL, NULL);
    }
  }
}
```

ANEXO B

Archivo VDI que representa todo el mundo virtual, así como todos los objetos que lo conforman y su comportamiento:

```
DIV-VDI1
Header (Version=2:1; Date=2:12:98; Time=11:47; Unit=M ){}

UseLibrary (File="hitlabf"; Type=MaterialsFile ){
}

Zone (Name=hitlabf){
  Physical {
    Gravity {0.81, 0, -1, 0}
    State {On }
  }
  Object (Name=AmbientLight) {
    Light {
      Type {Ambient }
      State {On }
      Colour {1, 1, 1}
    }
  }
  Object (Name=mesita) {
    Position {0.74, -0.401, -6.899}
    Scale {0.8, 1, 1}
    Visual {
      Geometry {"hitlabf2"}
    }
    Constraints {
      LockY
      LockRoll
      LockPitch
    }
    Collision {
      State {On }
    }
  }
  Object (Name=caja) {
    Position {-0.546, -0.885, -2.906}
    Visual {
      Geometry {"hitlabf3"}
    }
    Constraints {
      LockRoll
      LockPitch
      LockYaw
    }
    Collision {
      State {On }
    }
  }
}
```



```

Physical {
  Mass {0.01}
  State {Off }
}
Audio {
  State {Off }
  Voice {"glass"}
  Velocity {63}
  Gain {70}
}
Event {
  Drop { dvObjectPhysicalState("*", On);}
  Collide {
    dvObjectPhysicalState("*", Off);
    dvObjectAudio("*", NULL, On, 1, NULL, NULL);
  }
}
}
Object (Name=paredlat) {
  Position {-1.52986, 0, -2.50993}
  Visual {
    Geometry {"hitlabf8"}
  }
Object (Name=contac) {
  Position {-0.19, -1, 2}
  Orientation {0, -90, 90}
  Visual {
    Geometry {"hitlabf29"}
  }
Object (Name=switch_on) {
  Position {1.24, 0.001, 3.526}
  Visual {
    Geometry {"hitlabf28"}
  }
  Constraints {
    LockX
    LockY
    LockZ
    LockRoll
    LockPitch
    LockYaw
  }
  Collision {
    State {On }
  }
  Event {
    Pick {
      dvObjectLight(bulbos, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_0, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_1, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_2, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_3, NULL, On, NULL, NULL, NULL);
      dvObjectLight(bulbos_4, NULL, On, NULL, NULL, NULL);
    }
  }
}

```

```

    }
  }
}
Object (Name=switch_off) {
  Position {1.19, 0.001, 3.526}
  Visual {
    Geometry {"hitlabf28"}
  }
  Constraints {
    LockX
    LockY
    LockZ
    LockRoll
    LockPitch
    LockYaw
  }
  Collision {
    State {On }
  }
  Event {
    Pick {
      dvObjectLight(bulbos, NULL, Off, NULL, NULL, NULL);
      dvObjectLight(bulbos_0, NULL, Off, NULL, NULL, NULL);
      dvObjectLight(bulbos_1, NULL, Off, NULL, NULL, NULL);
      dvObjectLight(bulbos_2, NULL, Off, NULL, NULL, NULL);
      dvObjectLight(bulbos_3, NULL, Off, NULL, NULL, NULL);
      dvObjectLight(bulbos_4, NULL, Off, NULL, NULL, NULL);
    }
  }
}
}
}
Object (Name=pizarron) {
  Position {-3.69922, 0.372809, -0.90533}
  Visual {
    Geometry {"hitlabf31"}
  }
}
Object (Name=borrador) {
  Position {0.0412302, -0.548405, 1.60256}
  Visual {
    Geometry {"hitlabf15"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {

```

```

        Drop { dvObjectPhysicalState("*", On);}
        Collide {
            dvObjectPhysicalState("*", Off);
            dvObjectAudio("*", NULL, On, 1, NULL, NULL);
        }
    }
}
}
Object (Name=paredtra) {
    Position {-2.23484, 0, 0.885318}
    Visual {
        Geometry {"hitlabf9"}
    }
    Object (Name=foto) {
        Position {-1.42416, 0.33, -0.985318}
        Visual {
            Geometry {"hitlabf12"}
        }
    }
    Object (Name=canceleria) {
        Position {0.80858, 0.51494, -4.79052}
        Visual {
            Geometry {"hitlabf13"}
        }
    }
    Object (Name=vidrios) {
        Position {0.8223, 1.10443, -4.79039}
        Visual {
            Geometry {"hitlabf14"}
        }
    }
}
Object (Name=polhemus) {
    Position {-1.97188, 0.105214, -1.06705}
    Orientation {-90, 0, 0}
    Visual {
        Geometry {"hitlabf11"}
    }
    Constraints {
        LockRoll
        LockPitch
        LockYaw
    }
    Collision {
        State {On }
    }
    Physical {
        Mass {0.01}
        State {Off }
    }
    Audio {
        State {Off }
        Voice {"glass"}
    }
}

```

```

    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("...", On);}
    Collide {
      dvObjectPhysicalState("...", Off);
      dvObjectAudio("...", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=cemento) {
  Position {-1.90812, -2.92063e-05, -2.90225}
  Visual {
    Geometry {"hitlabf16"}
  }
}
Object (Name=robot) {
  Position {0.99383, -0.95003, -6.6508}
  Visual {
    Geometry {"hitlabf18"}
  }
  Collision {
    State {On }
  }
  Constraints {
    LockRoll
    LockPitch
    LockYaw
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("...", On);}
    Collide {
      dvObjectPhysicalState("...", Off);
      dvObjectAudio("...", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=piso) {
  Position {-2.25994, -1.27022, -2.51659}
  Visual {
    Geometry {"hitlabf20"}
  }
  Collision {

```

```

    State {On }
  }
  Constraints {
    LockX
    LockY
    LockZ
    LockRoll
    LockPitch
    LockYaw
  }
}
Object (Name=techo) {
  Position {-2.25994, 1.27022, -2.51659}
  Visual {
    Geometry {"hitlabf21"}
  }
Object (Name=lampara) {
  Position {2.578, 0.095, -1.131}
  Visual {
    Geometry {"hitlabf27"}
  }
Object (Name=bulbos) {
  Position {0.00173008, -0.00171995, 0}
  Visual {
    Geometry {"hitlabf26"}
  }
  Light {
    Type {Directional }
    State {On }
    Colour {1, 1, 1}
  }
}
}
Object (Name=lampara_0) {
  Position {2.578, 0.095, -4.262}
  Visual {
    Geometry {"hitlabf27"}
  }
Object (Name=bulbos_0) {
  Position {0.00173008, -0.00171995, 0}
  Visual {
    Geometry {"hitlabf26"}
  }
  Light {
    Type {Directional }
    State {On }
    Colour {1, 1, 1}
  }
}
}
Object (Name=lampara_1) {
  Position {0.578, 0.095, -1.131}
  Visual {
    Geometry {"hitlabf27"}
  }
}

```

```

    }
    Object (Name=bulbos_1) {
      Position {0.00173008, -0.00171995, 0}
      Visual {
        Geometry {"hitlabf26"}
      }
      Light {
        Type {Directional }
        State {On }
        Colour {1, 1, 1}
      }
    }
  }
  Object (Name=lampara_2) {
    Position {0.578, 0.095, -4.262}
    Visual {
      Geometry {"hitlabf27"}
    }
    Object (Name=bulbos_2) {
      Position {0.00173008, -0.00171995, 0}
      Visual {
        Geometry {"hitlabf26"}
      }
      Light {
        Type {Directional }
        State {On }
        Colour {1, 1, 1}
      }
    }
  }
  Object (Name=lampara_3) {
    Position {-1.578, 0.095, -1.131}
    Visual {
      Geometry {"hitlabf27"}
    }
    Object (Name=bulbos_3) {
      Position {0.00173008, -0.00171995, 0}
      Visual {
        Geometry {"hitlabf26"}
      }
      Light {
        Type {Directional }
        State {On }
        Colour {1, 1, 1}
      }
    }
  }
  Object (Name=lampara_4) {
    Position {-1.578, 0.095, -4.262}
    Visual {
      Geometry {"hitlabf27"}
    }
    Object (Name=bulbos_4) {
      Position {0.00173008, -0.00171995, 0}

```

```

    Visual {
      Geometry {"hitlabf26"}
    }
    Light {
      Type {Directional }
      State {On }
      Colour {1, 1, 1}
    }
  }
}
Object (Name=multimedia) {
  Position {2.54305, 0, -2.51659}
  Visual {
    Geometry {"hitlabf22"}
  }
}
Object (Name=cubiculos) {
  Position {-7.06294, 0, -2.51659}
  Visual {
    Geometry {"hitlabf23"}
  }
}
Object (Name=entrada) {
  Position {-2.25994, 0, 2.77092}
  Visual {
    Geometry {"hitlabf24"}
  }
}
Object (Name=jardin) {
  Position {-2.25994, 0, -7.8041}
  Visual {
    Geometry {"hitlabf25"}
  }
}
Object (Name=mesa) {
  Position {-0.383208, -0.400694, -3.83859}
  Visual {
    Geometry {"hitlabf30"}
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
  Collision {
    State {On }
  }
}
Object (Name=base) {
  Position {-1.16115, 0.138096, 3.38887}
  Visual {
    Geometry {"hitlabf33"}
  }
}
Object (Name=casco) {

```

```

    Position {0.000200033, 0.067986, -0.023419}
    Visual {
        Geometry {"hitlabf10"}
    }
    Collision {
        State {On }
    }
    Constraints {
        LockRoll
        LockPitch
        LockYaw
    }
    Physical {
        Mass {0.01}
        State {Off }
    }
    Audio {
        State {Off }
        Voice {"glass"}
        Velocity {63}
        Gain {70}
    }
    Event {
        Drop { dvObjectPhysicalState("*", On);}
        Collide {
            dvObjectPhysicalState("*", Off);
            dvObjectAudio("*", NULL, On, 1, NULL, NULL);
        }
    }
}
Object (Name=video) {
    Position {-1.41628, 0.046612, 3.38454}
    Visual {
        Geometry {"hitlabf17"}
    }
    Collision {
        State {On }
    }
    Physical {
        Mass {0.01}
        State {Off }
    }
    Audio {
        State {Off }
        Voice {"glass"}
        Velocity {63}
        Gain {70}
    }
    Event {
        Drop { dvObjectPhysicalState("*", On);}
        Collide {
            dvObjectPhysicalState("*", Off);
            dvObjectAudio("*", NULL, On, 1, NULL, NULL);
        }
    }
}

```



```

    }
  }
}
Object (Name=teclado) {
  Position {-0.018273, 0.107276, 0.00226998}
  Visual {
    Geometry {"hitlabf4"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("*", On);}
    Collide {
      dvObjectPhysicalState("*", Off);
      dvObjectAudio("*", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=cpu) {
  Position {-0.018273, 0.107276, 0.00226998}
  Visual {
    Geometry {"hitlabf6"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("*", On);}
    Collide {
      dvObjectPhysicalState("*", Off);
      dvObjectAudio("*", NULL, On, 1, NULL, NULL);
    }
  }
}

```

```

}
Object (Name=monitor) {
  Position {0.442, 0.376, -1.158}
  Scale {1.6, 1.6, 1.3}
  Visual {
    Geometry ("hitlabf5")
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("", On);}
    Collide {
      dvObjectPhysicalState("", Off);
      dvObjectAudio("", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=contacto) {
  Position {-1.99676e-06, 0.080203, 0.0157599}
  Visual {
    Geometry ("hitlabf7")
  }
}
Object (Name=silla) {
  Position {-0.383208, -0.400694, -3.83859}
  Visual {
    Geometry ("hitlabf32")
  }
  Collision {
    State {On }
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
}
Object (Name=mesa_0) {
  Position {1.097, -0.401, -1.599}
  Orientation {0, 270, 0}
  Visual {
    Geometry ("hitlabf30")
  }
}

```

```

}
Constraints {
  LockY
  LockRoll
  LockPitch
}
Collision {
  State {On }
}
Object (Name=tv_0) {
  Position {2.395, 0.476, -1.76}
  Scale {2, 2, 1.1}
  Visual {
    Geometry {"hitlabf1"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("*", On);}
    Collide {
      dvObjectPhysicalState("*", Off);
      dvObjectAudio("*", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=contacto_0) {
  Position {-1.99676e-06, 0.080203, 0.0157599}
  Visual {
    Geometry {"hitlabf7"}
  }
}
Object (Name=ipu) {
  Position {1.061, 0.095, 3.292}
  Orientation {0, 90, 0}
  Visual {
    Geometry {"hitlabf19"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
}

```

```

    }
    Audio {
        State {Off }
        Voice {"glass"}
        Velocity {63}
        Gain {70}
    }
    Event {
        Drop { dvObjectPhysicalState("", On);}
        Collide {
            dvObjectPhysicalState("", Off);
            dvObjectAudio("", NULL, On, 1, NULL, NULL);
        }
    }
}
Object (Name=tv_1) {
    Position {1.195, 0.476, -1.76}
    Scale {2, 2, 1.1}
    Visual {
        Geometry {"hitlabf1"}
    }
    Collision {
        State {On }
    }
    Physical {
        Mass {0.01}
        State {Off }
    }
    Audio {
        State {Off }
        Voice {"glass"}
        Velocity {63}
        Gain {70}
    }
    Event {
        Drop { dvObjectPhysicalState("", On);}
        Collide {
            dvObjectPhysicalState("", Off);
            dvObjectAudio("", NULL, On, 1, NULL, NULL);
        }
    }
}
Object (Name=mesa_1) {
    Position {-1.583, -0.401, -1.679}
    Orientation {0, 90, 0}
    Visual {
        Geometry {"hitlabf30"}
    }
    Constraints {
        LockY
        LockRoll
        LockPitch
    }
}

```

```

Collision {
  State {On }
}
Object (Name=teclado_0) {
  Position {1.242, 0.107, 0.002}
  Visual {
    Geometry {"hitlabf4"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("", On);}
    Collide {
      dvObjectPhysicalState("", Off);
      dvObjectAudio("", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=cpu_0) {
  Position {1.342, 0.107, 0.002}
  Visual {
    Geometry {"hitlabf6"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("", On);}
    Collide {
      dvObjectPhysicalState("", Off);
      dvObjectAudio("", NULL, On, 1, NULL, NULL);
    }
  }
}

```

```

}
Object (Name=monitor_0) {
  Position {1.522, 0.336, -0.438}
  Scale {1.3, 1.3, 1.1}
  Visual {
    Geometry {"hitlabf5"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("*", On);}
    Collide {
      dvObjectPhysicalState("*", Off);
      dvObjectAudio("*", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=contacto_1) {
  Position {-1.99676e-06, 0.080203, 0.0157599}
  Visual {
    Geometry {"hitlabf7"}
  }
}
Object (Name=tv) {
  Position {1.115, 0.476, -1.88}
  Scale {2, 2, 1.1}
  Visual {
    Geometry {"hitlabf1"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {

```

```

    Drop { dvObjectPhysicalState("*", On);}
    Collide {
        dvObjectPhysicalState("*", Off);
        dvObjectAudio("*", NULL, On, 1, NULL, NULL);
    }
}
Object (Name=cpu_3) {
    Position {0.702, 0.107, -0.078}
    Visual {
        Geometry {"hitlabf6"}
    }
    Collision {
        State {On }
    }
    Physical {
        Mass {0.01}
        State {Off }
    }
    Audio {
        State {Off }
        Voice {"glass"}
        Velocity {63}
        Gain {70}
    }
    Event {
        Drop { dvObjectPhysicalState("*", On);}
        Collide {
            dvObjectPhysicalState("*", Off);
            dvObjectAudio("*", NULL, On, 1, NULL, NULL);
        }
    }
}
Object (Name=monitor_3) {
    Position {0.942, 0.276, -0.438}
    Scale {1.3, 1.3, 1.1}
    Visual {
        Geometry {"hitlabf5"}
    }
    Collision {
        State {On }
    }
    Physical {
        Mass {0.01}
        State {Off }
    }
    Audio {
        State {Off }
        Voice {"glass"}
        Velocity {63}
        Gain {70}
    }
    Event {
        Drop { dvObjectPhysicalState("*", On);}

```

```

        Collide {
            dvObjectPhysicalState("", Off);
            dvObjectAudio("", NULL, On, 1, NULL, NULL);
        }
    }
}
Object (Name=mesa_2) {
    Position {-1.583, -0.401, -5.519}
    Orientation {0, 90, 0}
    Visual {
        Geometry {"hitlabf30"}
    }
    Constraints {
        LockY
        LockRoll
        LockPitch
    }
    Collision {
        State {On }
    }
    Object (Name=contacto_2) {
        Position {-1.99676e-06, 0.080203, 0.0157599}
        Visual {
            Geometry {"hitlabf7"}
        }
    }
}
Object (Name=mesa_3) {
    Position {1.097, -0.401, -5.999}
    Orientation {0, 270, 0}
    Visual {
        Geometry {"hitlabf30"}
    }
    Constraints {
        LockY
        LockRoll
        LockPitch
    }
    Collision {
        State {On }
    }
    Object (Name=teclado_1) {
        Position {0.702, 0.107, 0.002}
        Visual {
            Geometry {"hitlabf4"}
        }
        Collision {
            State {On }
        }
        Physical {
            Mass {0.01}
            State {Off }
        }
    }
}

```



```

Audio {
  State {Off }
  Voice {"glass"}
  Velocity {63}
  Gain {70}
}
Event {
  Drop { dvObjectPhysicalState("", On);}
  Collide {
    dvObjectPhysicalState("", Off);
    dvObjectAudio("", NULL, On, 1, NULL, NULL);
  }
}
}
Object (Name=monitor_1) {
  Position {1.682, 0.376, -1.138}
  Scale {1.6, 1.6, 1.3}
  Visual {
    Geometry {"hitlabf5"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("", On);}
    Collide {
      dvObjectPhysicalState("", Off);
      dvObjectAudio("", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=contacto_3) {
  Position {-1.99676e-06, 0.080203, 0.0157599}
  Visual {
    Geometry {"hitlabf7"}
  }
}
}
Object (Name=mesa_4) {
  Position {-0.483, -0.401, -3.839}
  Orientation {0, 180, 0}
  Visual {
    Geometry {"hitlabf30"}
  }
}

```

```

Constraints {
  LockY
  LockRoll
  LockPitch
}
Collision {
  State {On }
}
Object (Name=teclado_2) {
  Position {-0.178, 0.107, 0.002}
  Visual {
    Geometry {"hitlabf4"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("*", On);}
    Collide {
      dvObjectPhysicalState("*", Off);
      dvObjectAudio("*", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=cpu_2) {
  Position {0.002, 0.107, 0.002}
  Visual {
    Geometry {"hitlabf6"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("*", On);}
  }
}

```

```

    Collide {
        dvObjectPhysicalState("", Off);
        dvObjectAudio("", NULL, On, 1, NULL, NULL);
    }
}
Object (Name=monitor_2) {
    Position {0.222, 0.336, -0.438}
    Scale {1.3, 1.3, 1.1}
    Visual {
        Geometry {"hitlabf5"}
    }
    Collision {
        State {On }
    }
    Physical {
        Mass {0.01}
        State {Off }
    }
    Audio {
        State {Off }
        Voice {"glass"}
        Velocity {63}
        Gain {70}
    }
    Event {
        Drop { dvObjectPhysicalState("", On);}
        Collide {
            dvObjectPhysicalState("", Off);
            dvObjectAudio("", NULL, On, 1, NULL, NULL);
        }
    }
}
Object (Name=contacto_4) {
    Position {-1.99676e-06, 0.080203, 0.0157599}
    Visual {
        Geometry {"hitlabf7"}
    }
}
Object (Name=teclado_4) {
    Position {1.2, 0.107, 0.002}
    Visual {
        Geometry {"hitlabf4"}
    }
    Collision {
        State {On }
    }
    Physical {
        Mass {0.01}
        State {Off }
    }
    Audio {
        State {Off }
        Voice {"glass"}
    }
}

```

```

    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("", On);}
    Collide {
      dvObjectPhysicalState("", Off);
      dvObjectAudio("", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=cpu_4) {
  Position {1.342, 0.107, 0.002}
  Visual {
    Geometry {"hitlabf6"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("", On);}
    Collide {
      dvObjectPhysicalState("", Off);
      dvObjectAudio("", NULL, On, 1, NULL, NULL);
    }
  }
}
Object (Name=monitor_4) {
  Position {1.502, 0.336, -0.438}
  Scale {1.3, 1.3, 1.1}
  Visual {
    Geometry {"hitlabf5"}
  }
  Collision {
    State {On }
  }
  Physical {
    Mass {0.01}
    State {Off }
  }
  Audio {
    State {Off }
    Voice {"glass"}
    Velocity {63}
  }
}

```

```

    Gain {70}
  }
  Event {
    Drop { dvObjectPhysicalState("", On);}
    Collide {
      dvObjectPhysicalState("", Off);
      dvObjectAudio("", NULL, On, 1, NULL, NULL);
    }
  }
}
}
Object (Name=silla_0) {
  Position {0.817, -0.401, -3.839}
  Visual {
    Geometry {"hitlabf32"}
  }
  Collision {
    State {On }
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
}
Object (Name=silla_1) {
  Position {-1.443, -0.401, -2.179}
  Orientation {0, 90, 0}
  Visual {
    Geometry {"hitlabf32"}
  }
  Collision {
    State {On }
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
}
Object (Name=silla_2) {
  Position {-1.443, -0.401, -3.059}
  Orientation {0, 90, 0}
  Visual {
    Geometry {"hitlabf32"}
  }
  Collision {
    State {On }
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
}

```

```

}
Object (Name=silla_3) {
  Position {0.28, -0.401, -0.339}
  Orientation {0, 190, 0}
  Visual {
    Geometry {"hitlabf32"}
  }
  Collision {
    State {On }
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
}
Object (Name=silla_4) {
  Position {-1.443, -0.401, -5.339}
  Orientation {0, 90, 0}
  Visual {
    Geometry {"hitlabf32"}
  }
  Collision {
    State {On }
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
}
Object (Name=silla_5) {
  Position {0.04, -0.401, -4}
  Orientation {0, 190, 0}
  Visual {
    Geometry {"hitlabf32"}
  }
  Collision {
    State {On }
  }
  Constraints {
    LockY
    LockRoll
    LockPitch
  }
}
Object (Name=silla_6) {
  Position {-1.34, -0.401, -4}
  Orientation {0, 190, 0}
  Visual {
    Geometry {"hitlabf32"}
  }
  Collision {
    State {On }
  }

```

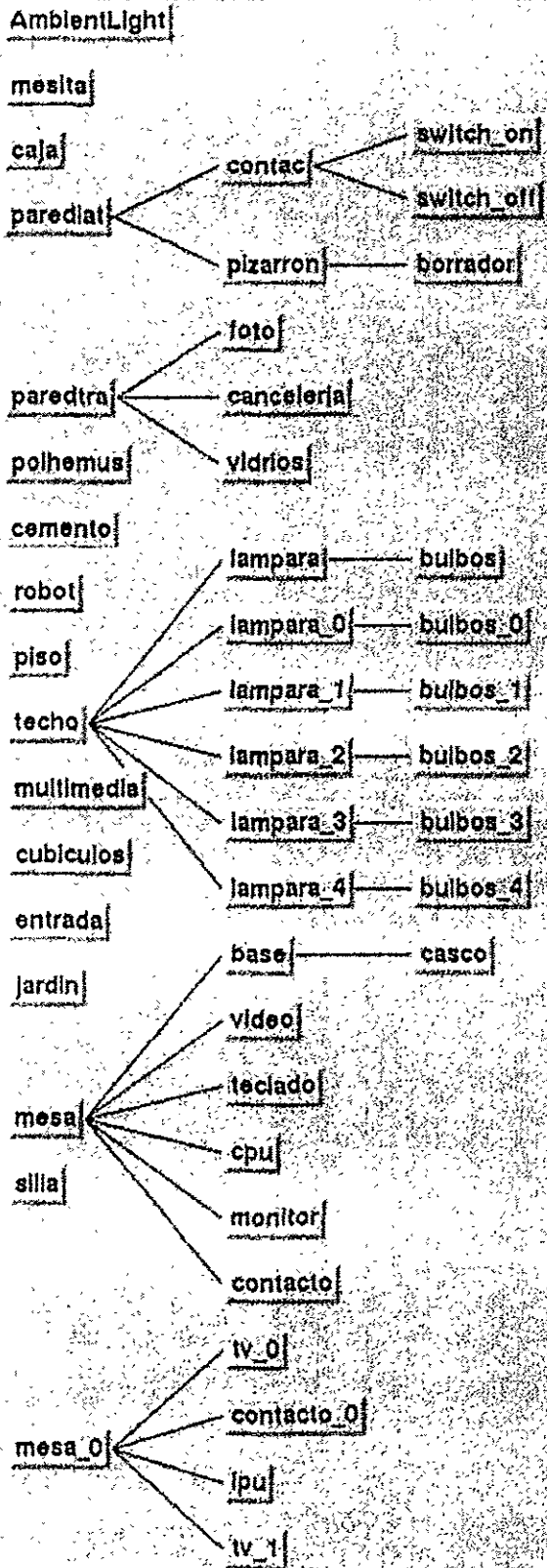
```

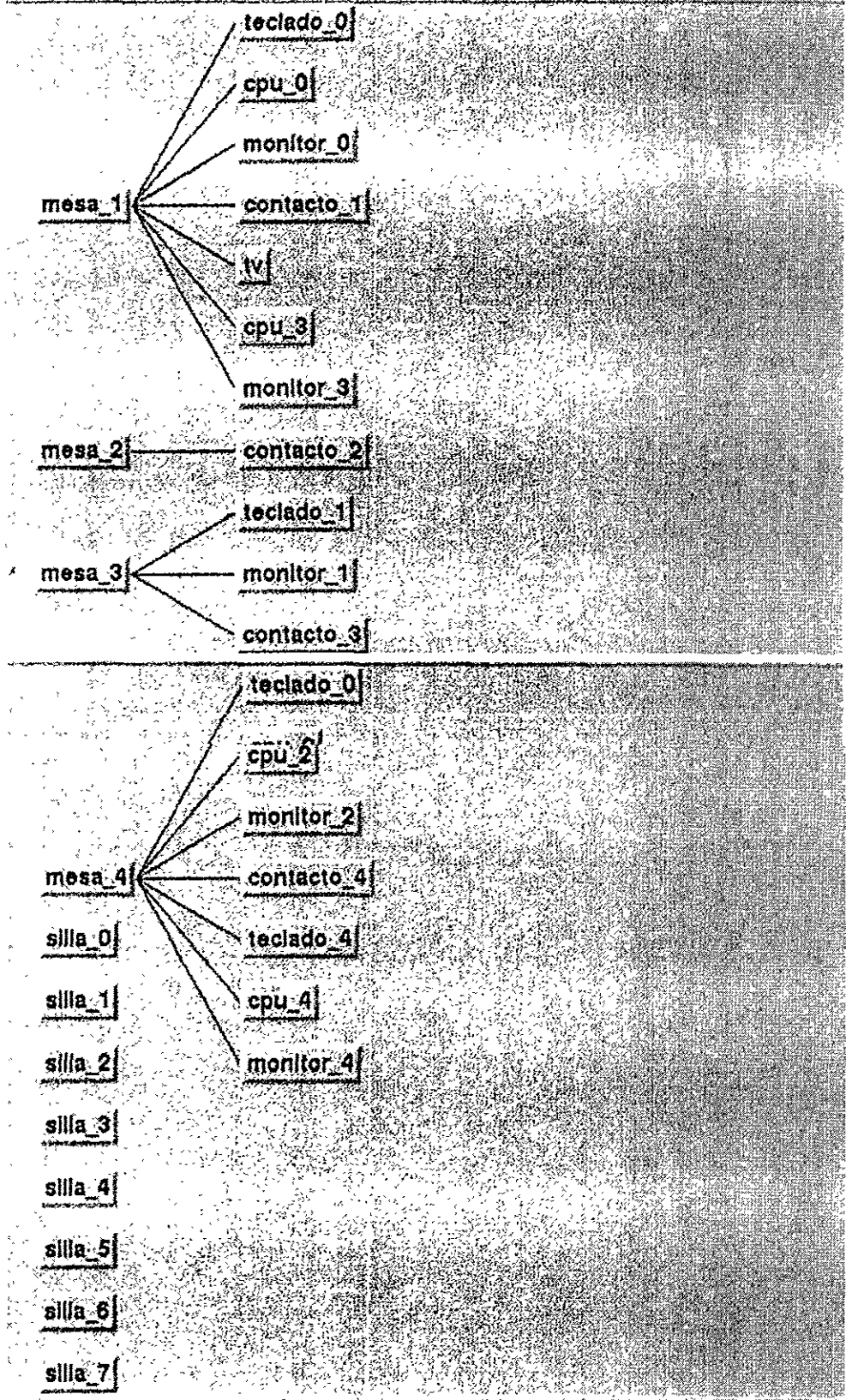
    }
    Constraints {
        LockY
        LockRoll
        LockPitch
    }
}
Object (Name=silla_7) {
    Position {0.92, -0.401, -4.58}
    Orientation {0, 270, 0}
    Visual {
        Geometry ("hitlabf32")
    }
    Collision {
        State {On }
    }
    Constraints {
        LockY
        LockRoll
        LockPitch
    }
}
}
}

```

Fin del listado del archivo VDI

Las siguientes imágenes ilustran la forma en que el Control Panel de dVISE representa cada uno de los objetos que conforman el mundo virtual creado. Esta representación se hace de acuerdo al archivo VDI que se enlista en la parte superior. Una manera sencilla y gráfica de comprobarlo es identificar los objetos ligados y buscarlos en el archivo VDI. Las características de cada objeto se comprueban, estando en el Control Panel, con las ventanas correspondiente a cada propiedad asignada, de acuerdo a los apartados 3.4.3 a 3.4.6 de este trabajo.





BIBLIOGRAFÍA

FUENTES PRIMARIAS

CHORAFAS, DIMITRIS N.; STEINMANN, HEINRICH, Realidad Virtual. Aplicaciones prácticas en los negocios y la industria, Prentice Hall, Hispanoamericana, S. A., México 1996

dVISE for UNIX Workstations. User Guide, Division, 1996

dVS for UNIX Workstations. User Guide, Division, 1996

Geometry Tools for UNIX Workstations. User Guide, Division, 1996

GRADECKI, JOE, Realidad Virtual. Construcción de proyectos, Alfaomega Grupo Editor, S. A. De C. V., México, 1997

HAMIT, FRANCIS, Virtual Reality and The Exploration of Cyberspace, Sams Publishing, USA, 1993

HARRISON, DAVID; JAQUES, MARK, Experiments in Virtual Reality, Ed. Butterworth Heinemann, Gran Bretaña, 1996

HAYWARD, TOM, Adventures in Virtual Reality, Que Corporation, USA, 1996

HOLLANDS, ROBIN, The Virtual Reality Homebrewer's Handbook, Ed. J. Wiley, New York, 1996

JACOBSON, LINDA, Garage Virtual Reality, Sams Publishing, USA, 1994

LARIJANI, CASEY L., Realidad Virtual, Mc Graw-Hill, España, 1994

PIMENTEL, KEN; TEIXEIRA, KEVIN, Virtual Reality Through the new looking glass, Ed. Intel/Windcrest/McGraw-Hill, Inc., 1993, USA

PINO GONZALEZ, L. M. DEL, Realidad Virtual, Ed. Paraninfo, España, 1995

STAMPE, DAVE; ROEHL, BERNIE; EAGAN, JOHN, Realidad Virtual. Creaciones y Desarrollo, Ed. Anaya Multimedia, S. A., Madrid, 1995

VINCE, JOHN, Virtual Reality Systems, Ed. Addison-Wesley Publishing Company, Gran Bretaña, 1995

FUENTES SECUNDARIAS

AQUINO DE, TOMAS, De ente et essentia,

FOLEY, JAMES D; VAN DAM, ANDREIS; FEINER, STEVEN K.; HUGHES, JOHN F., Computer Graphics. Principles and Practice, Ed. Addison-Wesley Publishing Company, USA, 1990

GATES, BILL, Camino al Futuro, Mc Graw-Hill/Interamericana de México, S. A. De C. V., México, 1995

GORALSKI, POLI, VOGEL, VRML: Exploring Virtual Worlds on the Internet, Ed. Prentice Hall, New Jersey, 1997

IOVINE, JOHN, Step into Virtual Reality, Ed. Windcrest/ Mc Graw-Hill, Inc., USA, 1995

MORRIS, MIKE, The Magic of IMAGE Processing, Sams Publishing, USA, 1993

RODRIGUEZ R., FRANCISCO J., Dinámica de Sistemas,

SIEWIOREK, DANIEL P.; KOOPMAN, PHILIP JOHN, JR., The Architecture of supercomputers, Ed. Academic, Boston, 1991

WODASKI, RON, Virtual Reality Madness 1996, Sams Publishing, USA, 1996

PÁGINAS WEB

<http://www.oz.net/~avi/gloss.htm>

<http://www.sgi.com/Products/hardware/desktop/products/index.html#O2>

<http://www.sgi.com/Octane/vrml/LivingO2-PC/TechDocs/cpu.html>

<http://www.sgi.com/Chat/Live/faqo2b.html>

<http://www.oz.net/~avi/gloss.htm>, <http://www.cms.dmu.ac.uk/~cph/VR/whatisvr.html>

<http://www.hp.com/nsa/pa1.1/html/acd-4.html#HEADING4-0>

http://indy1.adetti.iscte.pt/~visinet/aplicacoes/dVISE_i.html

<http://dval.larc.nasa.gov/DVAL/Whatsnew/dvise/>

<http://www.ima.hk->

[r.se/ima_web/kursmtrl/dVISE_instr/VERSION4/share/platforms.html](http://www.ima.hk-r.se/ima_web/kursmtrl/dVISE_instr/VERSION4/share/platforms.html)

<http://www.ima.hk->

[r.se/ima_web/kursmtrl/dVISE_instr/VERSION4/dvreview/concepts/4rev_concepts.html](http://www.ima.hk-r.se/ima_web/kursmtrl/dVISE_instr/VERSION4/dvreview/concepts/4rev_concepts.html)

ÍNDICE

SUMARIO	iii
INTRODUCCIÓN	1
1 DEFINICIÓN DE CONCEPTOS	3
1.1 Realidad Virtual	4
1.1.1 Mundo Virtual	5
1.1.2 Espacio Virtual	5
1.2 Sistema de RV	5
1.3 Características de los Sistemas de RV	5
1.3.1 Capacidad Sintética o Respuesta en Tiempo Real	6
1.3.2 Tridimensionalidad	6
1.3.3 Estereoscopia	7
1.3.3.1 Claves de Profundidad	7
1.3.4 Interactividad o Manipulación	9
1.3.5 Navegación	10
1.3.6 Ilusión de Realidad o Simulación de Comportamiento	10
1.3.7 Inmersión	10
1.3.7.1 Inmersión de bajo nivel	11
1.3.7.2 Inmersión de alto nivel	11
1.3.8 Punto de Vista	12

1.4 Hardware de un Sistema de RV	12
1.4.1 Dispositivos de Entrada	12
1.4.1.1 Dispositivo de Localización	12
1.4.1.1.1 Latencia	16
1.4.1.2 Dispositivo de Control	16
1.4.1.2.1 Mareo de Inmersión	17
1.4.2 Dispositivos de Salida	18
1.4.2.1 Dispositivo de Presentación	18
1.4.2.2 Dispositivo de Audio	19
1.4.2.3 Dispositivo de Realimentación Táctil	19
1.4.2.4 Dispositivos Móviles	20
1.4.3 Computadora	21
1.5 Software de un Sistema de RV	23
1.5.1 <i>Capacidades de un Software de RV</i>	24
1.6 Otros	26
1.6.1 Luces	26
1.6.2 Ejes de Coordenadas	27
1.7 Tomas Furness y HITL	28
2 HITL (Human Interface Laboratory, sucursal South)	34
2.1 El Diseñador	36

2.2 Hardware para un Sistema de RV	37
2.1.1 Dispositivos de Entrada	37
2.2.1.1 Dispositivo de Localización	38
2.2.1.2 Dispositivo de Control	39
2.2.2 Dispositivos de Salida	39
2.2.2.1 Dispositivo de Presentación	40
2.2.2.2 Dispositivo de Audio	40
2.2.3 Computadora	41
2.3 Software para un Sistema de RV	42
2.3.1 VRML	43
2.3.2 SENSE8	44
2.3.3 DIVISION	44
3 CREACIÓN DE UN SISTEMA DE REALIDAD VIRTUAL	47
3.1 DIVISION	49
3.1.1 dVS	49
3.1.2 dVISE	52
3.2 De Espacio Tridimensional a Espacio Virtual	57
3.2.1 Convertidores	57
3.2.1.1 3ds2vdi	59
3.2.1.1.1 img2vtx	60

3.3 Instrucciones Básicas de dVISE	61
3.3.1 Cargar Mundo y Abrir dVISE	62
3.3.2 Salir de dVISE	64
3.4 Propiedades a cada Objeto	64
3.4.1 Control Panel de dVISE	64
3.4.1.1 Zona	66
3.4.1.2 Ligar	68
3.4.1.3 Luces	70
3.4.1.4 Restricciones	71
3.4.1.5 Colisiones	73
3.4.1.6 Comportamiento	74
3.5 Activar periféricos	78
3.5.1 Tool Box	78
CONCLUSIONES GENERALES	81
ANEXO A	86
ANEXO B	90

BIBLIOGRAFÍA

117

ÍNDICE

120