
Índice general.

CAPÍTULO 1. INTRODUCCIÓN.....	11
1.1. SISTEMAS BCI (INTERFACES CEREBRO-COMPUTADOR).	11
1.1.1. Definición de sistema BCI.	12
1.1.2. Características de un sistema BCI.	14
1.1.3. Problemas de los sistemas BCI.	16
1.2. REALIDAD VIRTUAL.	17
1.2.1. Definición de Realidad Virtual.....	17
1.2.2. Características de la Realidad Virtual.....	18
1.2.3. Tipos de Realidad Virtual.....	19
1.2.4. Problemas actuales de la Realidad Virtual.....	21
1.2.5. Aplicaciones de la Realidad Virtual.....	23
1.3. REALIDAD VIRTUAL COMO INTERFAZ DE UN SISTEMA DE ENTRENAMIENTO BCI..	23
CAPÍTULO 2. OBJETIVOS.	25
CAPÍTULO 3. HERRAMIENTAS UTILIZADAS EN EL DESARROLLO DE LA APLICACIÓN.	27
3.1. HERRAMIENTAS SOFTWARE.....	27
3.1.1. V-Realm Builder.	27
3.1.1.1. Estructura de los mundos virtuales en V-Realm Builder.	28
3.1.1.2. Crear objetos tridimensionales con V-Realm Builder.....	30
3.1.1.3. Construir objetos tridimensionales complejos.....	39
3.1.1.4. Manipular los objetos tridimensionales.....	40
3.1.1.5. Diseño de un mundo virtual.	44
3.1.2. Virtual Reality Toolbox.	49

3.1.2.1. Virtual Reality Toolbox Viewer.....	50
3.1.2.2. Funciones.....	54
3.1.2.2.1. Funciones de la interfaz de MATLAB.....	54
3.1.2.2.2. Métodos del objeto vrworld.....	55
3.1.2.2.3. Métodos del objeto vnode.....	57
3.1.2.2.4. Métodos del objeto vrfigure.....	57
3.2. HERRAMIENTAS HARDWARE.....	58
 CAPÍTULO 4. DESCRIPCIÓN GENERAL DEL MUNDO VIRTUAL	
DISEÑADO.....	61
4.1. INTRODUCCIÓN AL MUNDO VIRTUAL IMPLEMENTADO.....	61
4.2. DESCRIPCIÓN DE LA ESCENA VIRTUAL BÁSICA.....	64
4.2.1. Implementación del fondo.....	65
4.2.2. Implementación de la carretera.....	65
4.2.3. Efectos implementados sobre la escena virtual.....	67
4.2.3.1. Movimiento de la carretera.....	68
4.2.3.2. Iluminación ambiental de la escena.....	71
4.2.3.3. Sonido.....	71
4.3 DESCRIPCIÓN DE LOS OBJETOS VIRTUALES DISEÑADOS.....	72
4.3.1. Diseño del objeto biofeedback.....	73
4.3.1.1. Implementación del coche.....	73
4.3.1.2. Efecto implementado en el coche.....	74
4.3.2. Diseño de los obstáculos.....	75
4.3.2.1. Implementación del charco.....	76
4.3.2.2. Implementación del muro.....	79
4.3.2.3. Implementación de la rampa.....	85
4.3.2.4. Configuraciones de los obstáculos.....	90
 CAPÍTULO 5. INTEGRACIÓN DE LA INTERFAZ GRÁFICA	
DESARROLLADA CON EL RESTO DEL SISTEMA BCI.....	93
5.1. ANTECEDENTES DE LA INTERFAZ.....	93
5.1.1. Descripción del paradigma de entrenamiento de referencia.....	94
5.1.2. Comparativa de la interfaz inicial y la interfaz desarrollada.....	96
5.2. DIAGRAMA BÁSICO DEL SISTEMA BCI GENERADO.....	98

5.2.1. <i>Adquisición de los datos.</i>	99
5.2.2. <i>Procesado de los datos.</i>	100
5.2.3. <i>Clasificador.</i>	101
5.2.4. <i>Realimentación o biofeedback al sujeto.</i>	102
5.3. DESCRIPCIÓN DE LA APLICACIÓN FINAL DESARROLLADA.....	103
5.3.1. <i>Funcionamiento general del sistema BCI desarrollado.</i>	106
5.3.2.1. <i>Concepto de experimento.</i>	106
5.3.2.2. <i>Concepto de sesión.</i>	107
5.3.2.3. <i>Concepto de ensayo.</i>	107
5.3.2.4. <i>Concepto de prueba</i>	107
5.4. SISTEMA DE MEDIDA DE TIEMPOS DE REACCIÓN.....	109
5.4.1. <i>Descripción de la aplicación desarrollada.</i>	109
5.4.2. <i>Funcionamiento general de la aplicación.</i>	111
CAPÍTULO 6. VENTAJAS, LIMITACIONES Y PRUEBAS DE LAS APLICACIONES DISEÑADAS EN EL PROYECTO. RESULTADOS EXPERIMENTALES.....	117
6.1. SISTEMA BCI.....	117
6.1.1. <i>Ventajas de la aplicación.</i>	118
6.1.2. <i>Limitaciones de la aplicación.</i>	119
6.1.3. <i>Pruebas realizadas durante el proceso de diseño.</i>	121
6.2. SISTEMA DE MEDIDA DE LOS TIEMPOS DE REACCIÓN.....	124
6.2.1. <i>Ventajas de la aplicación.</i>	124
6.2.2 <i>Limitaciones de la aplicación.</i>	125
6.2.3. <i>Pruebas realizadas durante el proceso de diseño.</i>	126
6.3. RESULTADOS EXPERIMENTALES OBTENIDOS.....	127
6.3.1. <i>Pruebas realizadas.</i>	127
6.3.2. <i>Paradigma de entrenamiento.</i>	128
6.3.3. <i>Porcentajes de error.</i>	129
CAPÍTULO 7. CONCLUSIONES Y LÍNEAS FUTURAS DEL SISTEMA BCI.	131
7.1. CONCLUSIONES.....	131
7.2. LÍNEAS FUTURAS.	133
APÉNDICE A. MANUAL DE USUARIO.	135

A.1. FUNCIONAMIENTO DE LA HERRAMIENTA BCI.	135
A.1.1. Datos del Usuario.	136
A.1.2. Tipos de Obstáculos.	137
A.1.3. Organización del Ensayo.	137
A.1.4. Parámetros del Análisis y Estructura de la Prueba.	138
A.1.5. Botones.	139
A.1.6. Organización y almacenamiento de la información.	140
A.1.6.1. Estructura de Directorios.	140
A.1.6.2. Ficheros guardados.	141
A.2. FUNCIONAMIENTO DEL SISTEMA DE MEDIDA DE TIEMPOS DE REACCIÓN.	143
A.2.1. Datos del Usuario.	143
A.2.2. Pruebas del Ensayo.	144
A.2.3. Estructura de la Prueba.	144
A.2.4. Botones.	145
A.2.5. Organización y almacenamiento de la información.	145
A.2.5.1. Estructura de Directorios.	145
A.2.5.2. Ficheros guardados.	146
BIBLIOGRAFÍA.	149

Índice de figuras.

1.1 ESQUEMA SISTEMA BCI	12
3.1 VENTANA DEL PROGRAMA V-REALM BUILDER.....	28
3.2 VENTANA QUE POSEE EL ÁRBOL DE NODOS DE UN MUNDO VIRTUAL.....	29
3.3 ESTRUCTURA JERÁRQUICA DE UN ÁRBOL DE NODOS	29
3.4 VENTANA DE VISUALIZACIÓN DEL V-REALM BUILDER	30
3.5 SISTEMA DE COORDENADAS DEL MUNDO VIRTUAL	30
3.6 NODO SHAPE Y SUS CAMPOS	31
3.7 NODO APPEARANCE Y SUS CAMPOS.....	31
3.8 NODO MATERIAL Y SUS CAMPOS	32
3.9 VENTANA DE LA LIBRERÍA DE MATERIALES	32
3.10 NODO TEXTURE Y SUS CAMPOS	33
3.11 VENTANA DE LA LIBRERÍA DE TEXTURAS.....	33
3.12 ICONOS DE LAS DISTINTAS GEOMETRÍAS	34
3.13 NODO GEOMETRY CON LA GEOMETRÍA DE UN CUBO (BOX)	34
3.14 REFERENCIA DE LOS OBJETOS CREADOS EN V-REALM BUILDER.....	34
3.15 VENTANA PARA ESTABLECER EL TAMAÑO DE LA GEOMETRÍA.....	35
3.16 VENTANA DEL EDITOR ELEVATIONGRID.....	36
3.17 VENTANA DEL EDITOR EXTRUSION	37
3.18 VENTANA DEL EDITOR INDEXED FACESET	37
3.19 ÁRBOL DE NODOS TRAS FINALIZAR EL PROCESO DE CONSTRUCCIÓN DE UN OBJETO SIMPLE.....	38
3.20 NODO GROUP Y SUS CAMPOS.....	39
3.21 NODO SWITCH Y SUS CAMPOS	39

3.22 VENTANA DE VISUALIZACIÓN SOBRE LA QUE SE MANIPULAN LOS OBJETOS DEL MUNDO VIRTUAL.....	40
3.23 V-REALM BUILDER MUESTRA LA VENTANA DE ÁRBOL DE NODOS Y DE VISUALIZACIÓN DEL MUNDO	41
3.24 PERSPECTIVAS DEL MUNDO VIRTUAL	41
3.25 PERSPECTIVA PRINCIPAL DEL MUNDO VIRTUAL	42
3.26 PLANTA DEL MUNDO VIRTUAL	42
3.27 PERFIL DEL MUNDO VIRTUAL	43
3.28 ALZADO DEL MUNDO VIRTUAL	43
3.29 NODO BACKGROUND Y SUS CAMPOS	45
3.30 NODO FOG Y SUS CAMPOS	45
3.31 NODO VIEWPOINT Y SUS CAMPOS.....	46
3.32 NODO DIRECTIONALLIGHT Y SUS CAMPOS.....	47
3.33 NODO POINTLIGHT Y SUS CAMPOS	47
3.34 NODO SPOTLIGHT Y SUS CAMPOS	47
3.35 VENTANA DE LA LIBRERÍA DE OBJETOS.....	48
3.36 NODO WORLDINFO Y SUS CAMPOS	49
3.37 VENTANA DE VISIONADO DE MUNDOS VIRTUALES	51
3.38 PANEL DE NAVEGACIÓN DE LA VENTANA DE VISIONADO	53
3.39 POLÍGRAFO COULBOURN INSTRUMENTS LABLINC	58
3.40 GORRO DE ELECTROS ELECTROCAP.....	59
4.1 FONDO IMPLEMENTADO EN LA ESCENA VIRTUAL BÁSICA.....	65
4.2 DIMENSIONES DEL ASFALTO DE LA CARRETERA.....	66
4.3 DIMENSIONES DE LAS LÍNEAS DE LA CARRETERA.....	66
4.4 ESCENA VIRTUAL BÁSICA	67
4.5 PUNTO DE VISTA DEL MUNDO VIRTUAL	68
4.6 POSICIÓN DE LOS OBJETOS Y SENTIDO DE DESPLAZAMIENTO	69
4.7 DISTANCIA ENTRE ELEMENTOS	70
4.8 ESCENA VIRTUAL AL INICIO Y TRAS HABERSE DESPLAZADO 4 UNIDADES.....	70
4.9 ESCENA VIRTUAL TRAS EL DESPLAZAMIENTO DE 8 Y 12 UNIDADES	70
4.10 ESCENA VIRTUAL TRAS EL DESPLAZAMIENTO DE 16 Y 20 UNIDADES	71
4.11 OBJETO BIOFEEDBACK	73
4.12 OBJETO BIOFEEDBACK SITUADO EN LA ESCENA VIRTUAL	74

4.13 FORMA DEL CHARCO EN EL EDITOR EXTRUSION DE V-REALM BUILDER	76
4.14 CHARCO IMPLEMENTADO	76
4.15 MUNDO VIRTUAL CON EL OBSTÁCULO CHARCO	77
4.16 MARCAS QUE ESTABLECEN EL INTERVALO DE CONTROL SOBRE EL OBJETO BIOFEEDBACK	78
4.17 MUNDO VIRTUAL CON LAS MARCAS QUE INDICAN EL INTERVALO DE CONTROL BIOFEEDBACK	78
4.18 MUNDO VIRTUAL EN UN INSTANTE EN EL QUE AMBAS MARCAS SON VISIBLES	79
4.19 MURO IMPLEMENTADO	79
4.20 MUNDO VIRTUAL CON EL OBSTÁCULO MURO	80
4.21 OBJETO BIOFEEDBACK DESPUÉS DE LA COLISIÓN	81
4.22 EFECTO DE LA COLISIÓN IMPLEMENTADO	81
4.23 VISTA LATERAL DE LA COLISIÓN	82
4.24 INTERVALO DEL EJE X EN EL QUE EL OBJETO BIOFEEDBACK COLISIONA CON EL MURO DERECHO	83
4.25 INTERVALO DEL EJE X EN EL QUE EL OBJETO BIOFEEDBACK COLISIONA CON EL MURO IZQUIERDO	83
4.26 DIMENSIONES DEL MURO PARA LA SEGUNDA APLICACIÓN	84
4.27 COCHE OCUPANDO EL CARRIL CENTRAL	84
4.28 COCHE OCUPANDO EL CARRIL DERECHO	84
4.29 COCHE OCUPANDO EL CARRIL IZQUIERDO	85
4.30 MUNDO VIRTUAL IMPLEMENTADO	85
4.31 RAMPA IMPLEMENTADA	86
4.32 MUNDO VIRTUAL CON EL OBSTÁCULO RAMPA	87
4.33 ASCENSIÓN POR LA PENDIENTE (ACTUALIZACIONES 1 Y 9)	88
4.34 ESTABILIZACIÓN SOBRE LA RASANTE (ACTUALIZACIONES 10 Y 29)	88
4.35 CAÍDA AL ASFALTO (ACTUALIZACIONES 30 Y 32)	88
4.36 CÁLCULO DEL ÁNGULO PARA EL ASCENSO DE LA PENDIENTE	89
4.37 PROCESO DEL EFECTO IMPLEMENTADO SOBRE LA RAMPA	89
4.38 INTERVALO DE EJE X EN EL QUE EL OBJETO BIOFEEDBACK SALTARÁ LA RAMPA DERECHA	89
4.39 INTERVALO DE EJE X EN EL QUE EL OBJETO BIOFEEDBACK SALTARÁ LA RAMPA IZQUIERDA	90
4.40 MUNDO VIRTUAL CON EL OBSTÁCULO CHARCO	90

4.41 MUNDO VIRTUAL CON LOS OBSTÁCULOS MURO Y CHARCO	91
4.42 MUNDO VIRTUAL CON LOS OBSTÁCULOS RAMPA Y CHARCO	91
5.1 TEMPORIZACIÓN DEL SISTEMA BCI DE REFERENCIA	95
5.2 DIAGRAMA BÁSICO DE LA BCI CREADA	98
5.3 PROCESO DE ANÁLISIS SEGUIDO POR LA SEÑAL	100
5.4 SUPERPOSICIÓN DE SECUENCIAS ANALIZADAS	101
5.5 DIAGRAMA DE BLOQUES DEL SISTEMA BCI DESARROLLADO	103
5.6 ESTRUCTURA TEMPORAL DE UNA PRUEBA GENÉRICA	108
5.7 DIAGRAMA DE BLOQUES DEL SISTEMA DE MEDIDA DE LOS TIEMPOS DE REACCIÓN	110
5.8 ESTRUCTURA TEMPORAL DE UNA PRUEBA GENÉRICA	112
5.9 ESTRUCTURA DE UNA PRUEBA CUANDO EL MURO SE MUESTRA SÓLO EN EL INSTANTE DE LA COLISIÓN	113
5.10 INSTANTES TEMPORALES EN LOS QUE SE PUEDE COLOCAR EL MURO ANTES DE LA POSIBLE COLISIÓN	114
5.11 ESTRUCTURA DE UNA PRUEBA EN EL QUE EL MURO NO LLEGA A SER VISIBLE	115
5.12 ESTRUCTURA TEMPORAL EN LA QUE EL MURO ES SIEMPRE VISIBLE	115
6.1 RETRASOS EN LA APLICACIÓN A 128 Hz, 4 MUESTRAS Y 1 ACTUALIZACIÓN	121
6.2 DURACIÓN DE UNA ACTUALIZACIÓN DEL MUNDO VIRTUAL	122
6.3 ESTRUCTURA DEL PARADIGMA DE ENTRENAMIENTO DE LAS PRUEBAS REALIZADAS	128
6.4 MEDIDA DE ERROR EN TODAS LAS SESIONES REALIZADAS POR CADA SUJETO	129
A.1 VENTANA CORRESPONDIENTE AL PANEL DE CONTROL DE LA BCI.....	136
A.2 EJEMPLO DE ESTRUCTURA DE FICHEROS CREADA POR LA HERRAMIENTA DE MEDIDA	140
A.3 PANEL DE CONTROL DE LA HERRAMIENTA DE MEDIDA DE LOS TIEMPOS DE REACCIÓN	143
A.4 EJEMPLO DE ESTRUCTURA DE FICHEROS CREADA POR LA HERRAMIENTA DE MEDIDA	146
A.5 PINES DEL PUERTO PARALELO EMPLEADOS EN LA HERRAMIENTA	147

Índice de tablas.

1.1 BANDAS DE FRECUENCIAS DE CADA TIPO DE RITMO CEREBRAL PROCEDENTE DE SEÑALES EEG	15
3.1 FUNCIONES QUE SE PUEDEN REALIZAR CON LA BARRA DE MENÚS.....	51
3.2 FUNCIONES QUE SE PERMITEN REALIZAR CON LA BARRA DE HERRAMIENTAS	52
3.3 FUNCIONES QUE SE PUEDEN REALIZAR MEDIANTE EL PANEL DE NAVEGACIÓN.....	54
A.1 VARIABLES ALMACENADAS EN RTOTAL.....	142

Capítulo 1.

Introducción.

1.1. Sistemas BCI (Interfaces Cerebro-Computador).

En los últimos años, ha crecido el interés en el desarrollo de una interfaz entre el cerebro humano y un sistema artificial, como un computador. Este tipo de sistema es el que se conoce como Interfaz Cerebro-Computador o más comúnmente sistema BCI (*Brain-Computer Interface*).

Una Interfaz Cerebro-Computador se basa en las medidas y el análisis de las señales electrofisiológicas del cerebro, predominando el uso de las señales electroencefalográficas (EEG) captadas durante algún tipo de actividad mental.

Las medidas de las señales electroencefalográficas proporcionan un nuevo canal “no muscular” para enviar señales y comandos a un mundo externo. Por este motivo, una de las aplicaciones más importantes de la Interfaz Cerebro-Computador está en el campo de la medicina y concretamente en la rehabilitación, consiguiendo establecer un canal de comunicación y control para aquellos individuos con importantes deficiencias en sus funciones motoras.

Este nuevo canal de comunicación hombre-máquina no es sencillo de manejar en una primera toma de contacto, por lo que es necesario un proceso de adaptación e

incluso de entrenamiento. El objetivo del proyecto trata de implementar un sistema de entrenamiento a través de una Interfaz Cerebro-Computador.

1.1.1. Definición de sistema BCI.

Una BCI es un sistema de comunicaciones encargado de adquirir una señal y transformarla en una señal de control capaz de realizar unas acciones determinadas en un ordenador (mover el ratón, seleccionar una letra,..).

El objetivo de un sistema BCI no es obtener las señales que el cerebro produce intentando averiguar sus intenciones, sino enseñarlo a producir ciertas señales eléctricas cerebrales a través de una determinada tarea mental que se sabe es capaz de producir.

A partir de la definición se podría representar el esquema general de un sistema BCI, tal y como se muestra en la figura 1.1.

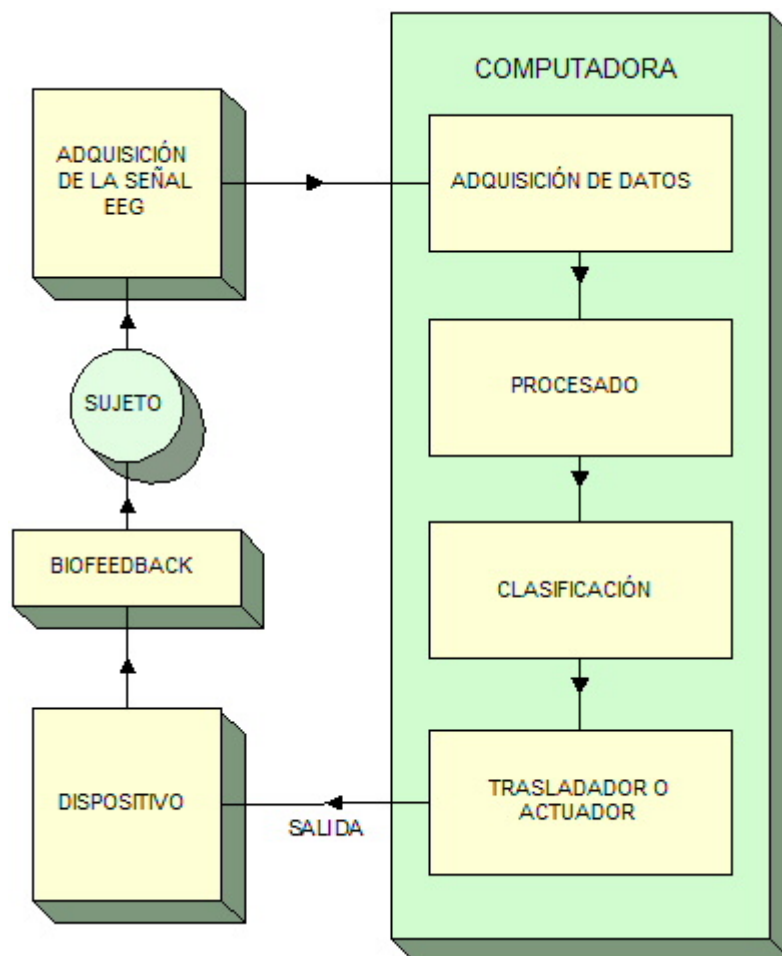


Figura 1.1. Esquema sistema BCI.

A continuación se explica cada uno de los bloques:

- **Adquisición de la señal:** La señal es adquirida mediante un hardware adecuado y es presentada al sistema BCI. Los elementos empleados para la obtención de la señal son electrodos colocados en una posición apropiada en el cuero cabelludo del sujeto. Las señales procedentes de los electrodos pasan por el polígrafo encargado de amplificarlas para que puedan ser tratadas por el ordenador.
- **Adquisición de datos:** Es el primer bloque dentro del ordenador, convierte las señales analógicas provenientes del amplificador, en digitales y las transfiere al bloque de procesado.
- **Procesado:** Es la fase en la que se realizará la transformación de la señal adquirida y se extraerán los parámetros de interés que la caractericen (potencia de la señal, espectro,...). El resultado se empleará en la fase de clasificación.
- **Clasificación:** El clasificador decide a qué estado mental corresponde cada una de las señales que emite el sujeto. Después de obtener el valor que representa a la señal adquirida, según el clasificador, se transmite al elemento trasladador o actuador.
- **Trasladador o Actuador:** Es el encargado de generar en la interfaz de salida los cambios oportunos acorde con los resultados obtenidos en la clasificación. Es precisamente este bloque el que se implementa en el proyecto.
- **Dispositivo:** Es el elemento que soporta la interfaz de salida, en el caso de este proyecto se trata de un monitor y a través de él se puede observar la interfaz gráfica creada.
- **Biofeedback (Bio-realimentación):** Es la manera en que un sistema BCI muestra al usuario los efectos de sus tareas mentales. Se trata de informar al usuario si realmente su objetivo llega a tener efecto en la acción que intenta realizar.

Se ha demostrado que con esta manera de indicar al usuario la consecución de una acción, posee información más detallada que le es útil para mejorar o corregir el proceso de su actividad mental.

Una forma de hacer el feedback más atractivo al sujeto es mediante el empleo de técnicas basadas en Realidad Virtual. Con esto se logra tener un alto grado de atención y

concentración, haciendo que el sujeto se involucre más activamente en el proceso de entrenamiento.

1.1.2. Características de un sistema BCI.

Las características que permiten diferenciar los diversos sistemas BCI son las siguientes:

- **Bidireccional:** El sistema BCI debe ser bidireccional para que se trate de un buen sistema. Es decir, que debe proporcionar y obtener información del cerebro. Proporcionar información al cerebro es sencillo, pero obtenerla a partir del estudio de la señal eléctrica cerebral es más problemático. La complejidad de las medidas de las señales se reduce si las medidas se centran en áreas específicas de la actividad cerebral, como el área concerniente a la función motora.
- **Modo de operación:** Puede ser síncrono, si la clasificación y análisis de las señales se hacen a través de ventanas de tiempo, o bien asíncrono, si el análisis y la clasificación de las señales se hace de forma ininterrumpida.
- **Tipo de registro:** Las señales son registradas a través de electrodos colocados en zonas concretas del cuero cabelludo mediante técnicas no invasivas. O bien, son registradas mediante técnicas invasivas, que requieren cirugía para implantar los electrodos en el cerebro. Las técnicas invasivas son las que obtienen mejores valores de señal, debido a que los valores de la señal están en unidades de microvoltios. Sin embargo, las técnicas no invasivas son las más sencillas de colocar y por tanto, de encontrar voluntarios.
- **Características necesarias de la señal captada:** Las señales que se captan en un sistema BCI son las electroencefalográficas. Estas señales están formadas por los llamados ritmos cerebrales, que son ondas cerebrales asociadas a un estado de concentración concreto y estudiadas en el dominio de la frecuencia.

Los ritmos cerebrales se corresponden a la actividad cerebral que se genera al realizar de forma consciente o no, algún tipo de tarea mental. Se distinguen distintos tipos de señales, que se clasifican en función de la banda de frecuencia que ocupen. En la tabla 1.1 puede verse la clasificación de ritmos cerebrales.

Ritmo Cerebral	Banda de Frecuencia (Hz)
δ	<4
θ	4-8
α , μ	8-12
β	12-32
γ	>32

Tabla 1.1. Bandas de frecuencia de cada tipo de ritmo cerebral procedente de señales EEG.

Las ondas δ , θ y γ no son de interés, porque no están relacionadas con la función motora. De hecho, las ondas δ aparecen sólo durante el sueño, las ondas θ aparecen en períodos de estrés emocional y frustración, y las ondas γ aparecen como respuesta a estímulos sonoros o luces relampagueantes.

La producción de ondas α en la mayoría de las personas se asocia al estado de relajación con los ojos cerrados. Pero en el momento que se realice una actividad física o mental, estas señales desaparecen o se reducen.

La característica más importante de los ritmos μ y β es que están relacionados con las funciones motoras. Se captan sobre las zonas del córtex más directamente relacionadas con las funciones motoras.

Se ha demostrado que imaginar un movimiento (sin llegar a ejecutarlo físicamente) produce efectos similares en estas ondas cerebrales, que el hecho de ejecutar físicamente dicho movimiento. A esto se le denomina imagen motora.

El concepto de imagen motora es el que se utiliza para detectar estados mentales en sistemas BCI, ya que la producción de este tipo de ondas (μ y β), puede ser regulada por la mayoría de las personas tras un entrenamiento y es especialmente interesante en el caso de personas con discapacidades motoras.

- **Estrategia empleada para la tarea mental a ejecutar:** Se trata de determinar que tareas mentales deben realizar los sujetos bajo estudio para que las señales cerebrales correspondientes a ellas sean distinguibles y por tanto, sean fáciles de clasificar.

Las tareas mentales más habituales que se discriminan son el reposo y la imaginación de un movimiento. Pero el sistema no se ha enfocado en la ejecución de ninguna determinada tarea mental.

-
- **Tipo de feedback:** Se suele proporcionar un feedback de tipo visual, es decir, el sujeto podrá ver a través de una pantalla u otro dispositivo de visualización (casco de realidad virtual, gafas estereoscópicas,...), cómo está realizando la tarea mental. Si su actividad es correcta, la interfaz enviará refuerzos positivos para continuar en esa línea y en caso contrario, dará refuerzos negativos para que se ponga empeño en mejorar en la siguiente ocasión.

1.1.3. Problemas de los sistemas BCI.

Después de exponer el concepto de sistema BCI y sus características, hay también que hablar de la problemática que plantea este tipo de sistema.

Una de las principales características que poseen estos sistemas es la necesidad de un proceso de adaptación por parte del sujeto a través de un entrenamiento, debido a que el sistema no se encarga de averiguar las intenciones del sujeto, sino que interpreta una serie de tareas mentales establecidas. Dependiendo del individuo, las actividades mentales se realizarán mejor o peor, por tanto, el tiempo que se emplea en la adaptación es variable, pudiendo resultar en ocasiones excesivo.

Por otro lado, existe un factor importante como es el tiempo de respuesta de un sistema BCI que puede resultar problemático. El motivo se encuentra en el proceso de clasificación de las señales, ya que debe realizarse solamente en un breve instante de tiempo después de la obtención de la señal.

Los sujetos bajo estudio esperan que la interfaz responda en una pequeña fracción de tiempo, y se debe a que necesitan observar casi instantáneamente el resultado producido por su actividad mental. Por ello, se puede concluir que un sistema BCI es crítico en el tiempo.

1.2. Realidad Virtual.

En este apartado, se explicarán algunos conceptos relacionados con esta tecnología. Se comenzará definiendo el concepto de Realidad Virtual y se especificarán algunas de sus características.

1.2.1. Definición de Realidad Virtual.

La Realidad Virtual puede definirse de la siguiente manera:

La realidad virtual es una simulación por computadora, dinámica y tridimensional, con alto contenido gráfico, acústico y táctil, orientada a la visualización de situaciones y variables complejas, durante la cual el usuario ingresa, a través del uso de sofisticados dispositivos de entrada, a “mundos” que aparentan ser reales, resultando inmerso en ambientes altamente participativos, de origen artificial. [1].

En definitiva, la Realidad Virtual es la representación de una serie de elementos a través de medios electrónicos, dando la sensación de estar en una situación real en la que se puede interactuar con el entorno creado.

La definición de Realidad Virtual posee conceptos tales como:

- El “mundo” que contiene “objetos” debe operar en base a unas reglas de juego.
- Se expresa en lenguaje gráfico tridimensional, para la representación del amplio contenido gráfico.
- El comportamiento es dinámico y opera en tiempo real, de forma que el sujeto tenga la sensación de que está sucediendo lo que ve.
- La operación está basada en la incorporación del usuario en el “interior” del medio computarizado. Requiere que, en principio haya una “suspensión de la incredulidad” como recurso para lograr la integración del usuario al mundo virtual al que ingresa.
- Posee la capacidad de reaccionar ante el usuario, ofreciéndole una experiencia inmersiva, interactiva y multisensorial.

1.2.2. Características de la Realidad Virtual.

- **Inmersión:** Es la propiedad que permite al usuario tener la sensación de encontrarse dentro de un mundo tridimensional. Para conseguir esta sensación, se bloquean las distracciones que puedan aparecer al usuario para que éste dirija su atención a la información sobre la cual trabaja y crea que está formando parte de la experiencia virtual.
- **Interacción:** Permite al usuario manipular, operar y controlar la acción dentro de la aplicación. Gracias a esta característica el sistema es capaz de responder a una reacción producida por el sujeto bajo estudio.
- **Tridimensionalidad:** Pretende estimular los sentidos del usuario para darle la impresión de que los objetos del mundo virtual son en tres dimensiones como en el mundo real. Los sonidos suelen tener también efectos estereofónicos.

Para crear elementos tridimensionales se emplean lenguajes como VRML, el cual es un estándar para la creación de mundos virtuales no inmersivos (concepto explicado en el apartado 1.2.3).

VRML es un acrónimo para *Virtual Reality Modeling Language* (Lenguaje para Modelado de Realidad Virtual). VRML provee un conjunto básico de primitivas para el modelado geométrico tridimensional y tiene la capacidad de dar comportamiento a los objetos y de asignar diferentes animaciones, las cuales pueden ser activadas por eventos generados por diferentes usuarios.

- **Tiempo real:** Los cálculos deben realizarse en el instante en el que se obtienen los datos, para que el mundo virtual tenga mayor sensación de realismo y el usuario tenga mayor libertad de acción. La sensación de continuidad en la imagen virtual se consigue cuando se muestran al menos 50 imágenes por segundo por lo que los requerimientos computacionales son elevados.
- **Multisensorialidad:** Este aspecto hace referencia a la percepción de fenómenos simulados mediante Realidad Virtual, donde actualmente el sentido de la vista se complementa con los del oído y el tacto.
- **Visión estereoscópica:** Es una técnica de visualización que mejora la sensación de profundidad, es decir, proporciona una sensación de tridimensionalidad. Crea por separado las imágenes del ojo izquierdo y del ojo derecho, encargándose el cerebro de mezclarlas.

La navegación de mundos virtuales no necesariamente debe utilizar este tipo de técnica, pudiendo emplearse interfaces más convencionales como se verá en el punto 1.2.3.

1.2.3. Tipos de Realidad Virtual.

La Realidad Virtual se puede clasificar en dos tipos:

- *Realidad Virtual Inmersiva.*
- *Realidad Virtual No Inmersiva.*

Los métodos inmersivos de Realidad Virtual se ligán a un ambiente tridimensional creado por computadora que se manipula a través de cascos, guantes u otros dispositivos. Estos dispositivos capturan la posición y rotación de diferentes partes del cuerpo humano. Existen una serie de sistemas de Realidad Virtual que se pueden clasificar como inmersivos:

- **Sistemas de mapeo por vídeo:** Este enfoque se basa en filmar, mediante cámaras de vídeo, a una o más personas e incorporar dichas imágenes en la pantalla del ordenador, donde podrán interactuar, en tiempo real, con otros usuarios o con imágenes gráficas generadas por él. De esta forma, las acciones que el usuario realice en el exterior de la pantalla se reproducen en la pantalla del equipo utilizado, permitiéndole interactuar, desde fuera. El usuario puede, a través de este enfoque, simular su participación en aventuras, deportes y otras formas de interacción física.

Este tipo de sistema puede ser considerado como una forma particular de sistema inmersivo.

- **Sistemas inmersivos:** Son sistemas avanzados de Realidad Virtual que buscan que el usuario pueda sentirse inmerso en el mundo. El fenómeno de inmersión puede presentarse de cuatro formas diferentes, dependiendo del método que se emplee para crear esta ilusión:
 - El operador aislado.
 - La cabina personal.
 - La cabina colectiva.
 - La caverna o cueva (cave).

Normalmente todas estas posibilidades aíslan al usuario de forma que quedan integrados en el mundo virtual de la mejor forma posible.

- **Sistemas de telepresencia:** Consiste en proporcionar al usuario sensaciones recogidas por sensores distantes situados en un determinado campo de operaciones. Esto permitirá teleoperar sondas-robots en el espacio o dirigir vehículos de forma remota.

La telepresencia contempla, obligatoriamente, un grado de inmersión que involucra el uso de control remoto.

- **Sistemas de realidad mixta y aumentada:** Al fusionar los sistemas de telepresencia y Realidad Virtual se obtienen los denominados sistemas de Realidad Mixta. Aquí las entradas generadas por el ordenador se mezclan con entradas de telepresencia y/o con la visión de los usuarios del mundo real.

Este tipo de sistema se orienta a la estrategia de realzar las percepciones del usuario con respecto al mundo real. Se suelen emplear dispositivos HMD (*Head Mounted Devices*, dispositivos colocados en la cabeza).

- **Sistemas de realidad virtual en pecera:** Este sistema combina un monitor de despliegue estereoscópico utilizando lentes LCD con obturador acoplados a un rastreador de cabeza mecánico.

La Realidad Virtual no inmersiva permite interactuar en tiempo real con diferentes espacios y ambientes que en realidad no existen. Todo esto se realiza sin la necesidad de dispositivos adicionales a la computadora. Un sistema no inmersivo típico es:

- **Sistema de ventanas (Window On World System (WoW)):** Utiliza un monitor convencional para mostrar el mundo virtual. Son conocidos como WoW y como Realidad Virtual de escritorio. Intenta que la imagen aparecida en pantalla parezca real y que los objetos representados en ella actúen con realismo. Por esto se llaman de ventana, porque la pantalla se convierte en una ventana a través de la cual se mira al mundo virtual.

La Realidad Virtual no inmersiva ofrece un nuevo mundo a través de una ventana de escritorio. Este enfoque no inmersivo tiene varias ventajas sobre el enfoque inmersivo como son el bajo costo, y la fácil y rápida aceptación de los usuarios. Los dispositivos inmersivos son de alto costo, y generalmente el usuario prefiere manipular el ambiente

virtual por medio de dispositivos familiares, como son el teclado y el ratón, en lugar de medios como cascos pesados o guantes.

1.2.4. Problemas actuales de la Realidad Virtual.

En términos del estado actual de la tecnología, existe aún un número de problemas importantes por resolver para poder garantizar el uso sistemático de esta tecnología a nivel de usuario. Entre estos problemas destacan:

- *Representación.*
- *Realimentación háptica (“haptic feedback”).*
- *Demora (“lag”) en tiempo de respuesta.*
- *Ángulo de visualización.*
- *Malestar por uso prolongado.*

A continuación, se explican los términos mencionados y el porqué de sus inconvenientes:

- **Representación:** Un mundo virtual está constituido por polígonos que son los bloques básicos de la computación gráfica. Los polígonos unidos en “mallas” sirven para representar objetos y escenarios, resultando indispensables en la constitución de mundos virtuales. El número de polígonos utilizados en la descripción de un objeto o escenario influye en la percepción de la imagen. Si el número de polígonos es elevado la imagen es más fina, pero también es necesaria una mayor velocidad de procesamiento para presentar la imagen en tiempo real.

En la actualidad los dispositivos de Realidad Virtual como mucho pueden producir de 7000 a 10.000 polígonos por segundo. Son valores insuficientes ya que se ha estimado que para representar imágenes del mundo real se necesitan entre 80 y 100 millones de polígonos por segundo. Sin embargo, estas necesidades son flexibles gracias a que el ser humano posee una muy adaptable capacidad de percepción. Por ejemplo, los dibujos animados son ampliamente aceptados con un mínimo de 500 polígonos.

La imagen creada a través de Realidad Virtual debe presentar una serie de características:

-
- Poseer tridimensionalidad.
 - Sincronizar los cambios en perspectiva originados por los desplazamientos del usuario, incluyendo la resolución de problemas de visibilidad de múltiples objetos.
 - La imagen requiere de tratamiento mediante sombras y efectos especiales para mantener la credulidad.
 - Existe una información complementaria de sonido, tacto y fuerza.
- **Realimentación háptica:** El problema principal dentro de la realimentación háptica se refiere al denominado “feedback de fuerza”, es decir al efecto que busca imitar a la realidad oponiendo campos de fuerza que permitan, por ejemplo, al chocar o empujar objetos, obtener una oposición o rechazo de parte de los mismos.

La realimentación de fuerza, hasta para los objetos más sencillos, es una muy difícil tarea y los despliegues hápticos no son diseñados como simples máquinas de tacto, sino mas bien como ambientes de los cuales una persona puede alcanzar algún conocimiento de propiedades asociadas con los objetos representados (tales como peso y solidez).
 - **Demora:** La demora es la medida de tiempo entre el momento en el que una persona ejecuta una acción y el momento en el que el computador la registra.

La demora implica un problema en aplicaciones virtuales, puesto que son en tiempo real y exigen una perfecta sincronización entre las acciones del usuario y el mundo virtual.
 - **Ángulo de visión:** Al ángulo de visión resulta difícil precisarle un campo óptimo de visión en Realidad Virtual ya que, lo que en un caso puede resultar adecuado, en otro puede no serlo. Así, por ejemplo, si se le ofrece un amplio campo de visión a una persona que necesita concentrarse para cumplir una tarea específica, son más los problemas que se le crean que los beneficios, porque un amplio campo de visión pudiera ofrecerle muchas distracciones. En el otro extremo, si se le da un campo de visión muy estrecho a una persona que está buscando alcanzar una percepción global, resultará inefectivo.
 - **Malestar por uso prolongado:** Se estima que un 10% de los usuarios de Realidad Virtual están afectados por el malestar derivado del uso prolongado. En este sentido, se han detectado síntomas de incomodidad y hasta de náusea

durante experiencias de Realidad Virtual, si la tasa de cuadros por segundo de la imagen virtual tiene unos valores determinados.

Una forma de combatir el malestar es la inclusión de un período de “entrenamiento” o adaptación a la experiencia virtual. Las investigaciones actuales detectaron que la náusea tiende a ocurrir durante la exposición inicial de un usuario a frecuentes movimientos de arranque y detención, y cambios en la aceleración.

1.2.5. Aplicaciones de la Realidad Virtual.

- **Psicología:** Se emplea para el tratamiento de distintas fobias, como son la claustrofobia o el vértigo.
- **Medicina:** Se crean protocolos para el entrenamiento en técnicas de cirugía complejas a través de simuladores de formación. Los simuladores permiten al médico desarrollar la habilidad y destreza en la técnica correspondiente.
- **Educación:** Se utiliza Realidad Virtual para el desarrollo de herramientas educativas y teleeducación.
- **Ingeniería:** Se usa para el manejo de robots a distancia, como parte integral de un proceso de diseño (diseño de compuestos orgánicos), ingeniería genética, etc.
- **Militar:** Una aplicación muy extendida son los simuladores de vuelo. Actualmente, es una herramienta fundamental para el entrenamiento de pilotos militares. Son sistemas muy sofisticados y costosos porque incorporan todo tipo de interfaces para simular la situación real dentro de un avión.

1.3. Realidad Virtual como Interfaz de un sistema de entrenamiento BCI.

Un sistema de entrenamiento BCI basado en técnicas de Realidad Virtual es de gran ayuda en el proceso de aprendizaje de los sujetos, debido a que resulta una experiencia menos monótona o aburrida.

Al proporcionar una interfaz gráfica más agradable para el usuario, se mejoran las condiciones en el proceso de entrenamiento. Por tanto, podrían ser necesarias menos sesiones de entrenamiento.

En un inicio, el estudio de sistemas BCI se centraba sobre todo en el desarrollo de algoritmos óptimos de procesamiento de señal y formas de clasificación de patrones EEG eficientes. Por este motivo, la mayoría de los interfaces se basaban en entornos de dos dimensiones, donde algún objeto indicaba la dirección en la que se debía desplazar un cursor o una barra mediante una tarea mental.

Algunos grupos de investigación consideraron fundamental el estudio de nuevas técnicas de biofeedback que permitieran al sujeto controlar, de forma fiable su patrón EEG. Es en este instante, cuando surge la idea de emplear técnicas basadas en Realidad Virtual como interfaz de los sistemas BCI.

La Realidad Virtual, aplicada al entrenamiento en sistemas BCI, proporciona un biofeedback que presenta las situaciones de forma realista, sirviéndole al sujeto para mejorar en su motivación concentración y aislamiento, con el fin de controlar su actividad mental y en consecuencia, sus señales electroencefalográficas.

Por tanto, la introducción de técnicas de Realidad Virtual ha sido con la idea de ofrecer al usuario entornos más estimulantes que los tradicionales que resultan más bien aburridos.

El desarrollo de biofeedback, basados en técnicas de Realidad Virtual, se ha convertido en una poderosa herramienta para la creación de programas, encaminados a tratar las necesidades de los usuarios de forma individualizada.

En definitiva, el empleo de Realidad Virtual como interfaz de un sistema BCI es para proporcionar al usuario realismo y ayudarle a mantener los niveles de concentración y motivación adecuados para llegar a controlar sus señales EEG en el menor tiempo posible.

Capítulo 2.

Objetivos.

El objetivo principal del proyecto es el desarrollo de una interfaz multimodal, utilizada como elemento de realimentación en los sistemas implementados en este proyecto. En concreto, se han creado una Interfaz Cerebro-Computador y un Sistema de Medida de Tiempos de Reacción.

El desarrollo de la interfaz multimodal se basa en técnicas de Realidad Virtual, debido a que permiten combinar representaciones en tres dimensiones con sonidos, produciendo un entorno virtual con un efecto más motivador e integrador. Los entornos virtuales provocan en los usuarios un aumento de la concentración e interacción, facilitando la fase de entrenamiento. Por tanto, el usuario podría ser capaz de controlar antes sus señales electroencefalográficas (EEG) en los sistemas BCI.

Una vez planteado el objetivo general, se pasa a describir de manera más extensa las dos aplicaciones desarrolladas en el proyecto.

La primera aplicación creada es un sistema BCI basado en una interfaz multimodal, desarrollándose en ella los conceptos planteados sobre BCI y Realidad Virtual. Para la implementación de la BCI se ha desarrollado una interfaz multimodal, como ya se ha comentado, empleando técnicas basadas en Realidad Virtual a través del *Virtual Reality Toolbox* de MATLAB. La interfaz consistirá en el control de un coche que deberá evitar una serie de obstáculos predefinidos. El coche para sortear los obstáculos realizará

desplazamientos a izquierda o a derecha. Finalmente, la interfaz se integrará en un sistema BCI ya existente y desarrollado en MATLAB por el grupo DIANA del Departamento de Tecnología Electrónica.

La aplicación resultante es una Interfaz Cerebro-Computador que incluye adquisición, filtrado y procesado de las señales EEG del sujeto, además de la comunicación en tiempo real con el mundo virtual (biofeedback). Este biofeedback añade realismo a las sesiones de entrenamiento, puesto que las tareas mentales del sujeto tienen el efecto inmediato esperado en la escena virtual.

La aplicación obtenida es una herramienta completa, flexible y versátil que se empleará como medio de trabajo en este ámbito de investigación.

La segunda aplicación creada en el proyecto es el sistema de medida de tiempos de reacción. En ella se pretende estudiar el tiempo que tarda un sujeto en reaccionar ante un estímulo muy determinado. La interfaz desarrollada será similar a la implementada en el sistema BCI, ya que se diferencia en que sólo se empleará un tipo de obstáculo. Además la aplicación no utilizará ni la adquisición, ni el procesado de la señal EEG debido a que no se capturarán señales, sino que el usuario se comunicará con el mundo virtual a través de los cursores derecho e izquierdo del teclado. Finalmente, la aplicación debe registrar el tiempo de reacción del sujeto, que va desde la aparición del obstáculo hasta la pulsación del cursor para el desplazamiento del coche.

Capítulo 3.

Herramientas utilizadas en el desarrollo de la aplicación.

A continuación, se realiza una amplia exposición de las herramientas software y hardware usadas en el proceso de desarrollo de la aplicación. En este capítulo, se pretende hacer un sencillo manual de usuario del *Virtual Reality Toolbox*, ya que es la primera vez que se emplea en el desarrollo de un proyecto.

3.1. Herramientas Software.

Se han utilizado principalmente dos herramientas. En primer lugar, se emplea el *V-Realm Builder* para la construcción de mundos virtuales, como su propio nombre indica. En él se crean distintos objetos en tres dimensiones, pudiendo darles la forma, color y textura adecuada. Una vez creado el mundo virtual, se emplea el *Virtual Reality Toolbox* de MATLAB, que es una librería de Realidad Virtual que permite el manejo de un entorno en tres dimensiones, con el inconveniente de no poder usar la visión estereoscópica.

3.1.1. V-Realm Builder.

Se trata de una potente herramienta tridimensional para la creación de objetos 3D y mundos virtuales. Además usa el lenguaje VRML, por tanto produce ficheros que

pueden ser leídos por cualquier sistema de visualización VRML 2.0 y a la vez, *V-Realm Builder* puede leer todos los mundos VRML 2.0.

VRML es un lenguaje estándar de modelado de Realidad Virtual. Se usa para definir mundos virtuales que pueden ser visualizados a través de una ventana de visionado.

V-Realm Builder no ha sido desarrollado con la intención de reemplazar otras herramientas de modelado, si no para establecer una herramienta virtual amigable. Posee una interfaz intuitiva, con la que rápidamente se puede operar y construir mundos VRML.

Las herramientas de modelado están disponibles en paneles de comandos y en una barra de herramientas principal, tal y como se muestra en la figura 3.1. Haciendo uso de estas utilidades se han diseñado los objetos 3D empleados en el mundo virtual.

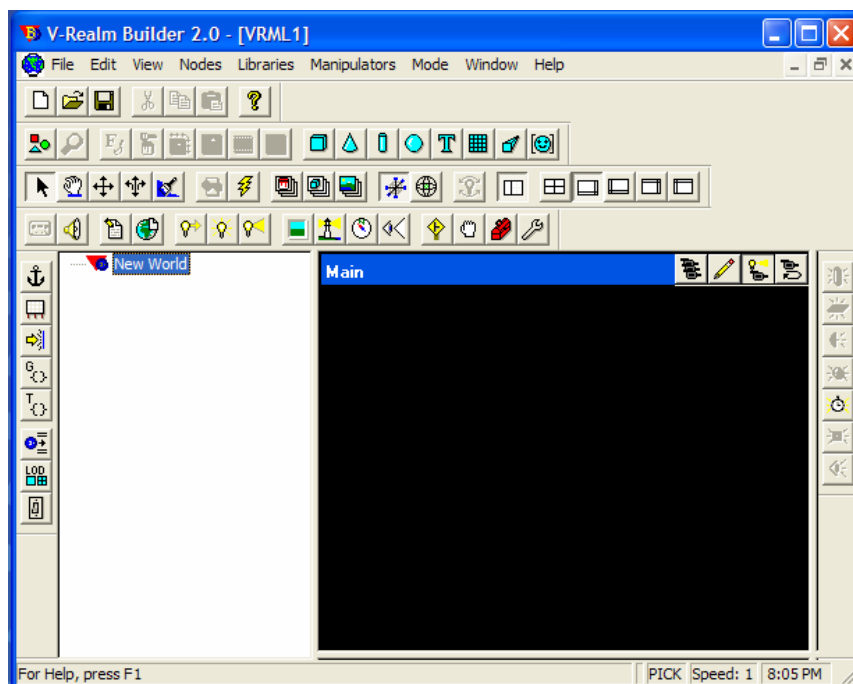


Figura 3.1: Ventana del programa V-Realm Builder.

3.1.1.1. Estructura de los mundos virtuales en V-Realm Builder.

El *V-Realm Builder* crea sus objetos a través de nodos, los cuales pueden ser de distintos tipos y vienen caracterizados por sus campos.

Los mundos virtuales son un conjunto ordenado de nodos (objetos) estructurados en forma de árbol, debido a que los nodos pueden ser tanto padres como hijos de otros nodos. Existe una ventana que representa visualmente el árbol de nodos, tal y como muestra la figura 3.2.



Figura 3.2: Ventana que posee el árbol de nodos de un mundo virtual.

Para poder asignar un nodo al mundo en construcción hay que seleccionar *New World* que es la raíz del árbol de nodos. *New World* adquirirá el nombre del fichero (*.wrl) cuando se grabe. En la figura 3.3 puede verse un ejemplo de una jerarquía concreta. Las características de un nodo padre son inherentes al hijo.

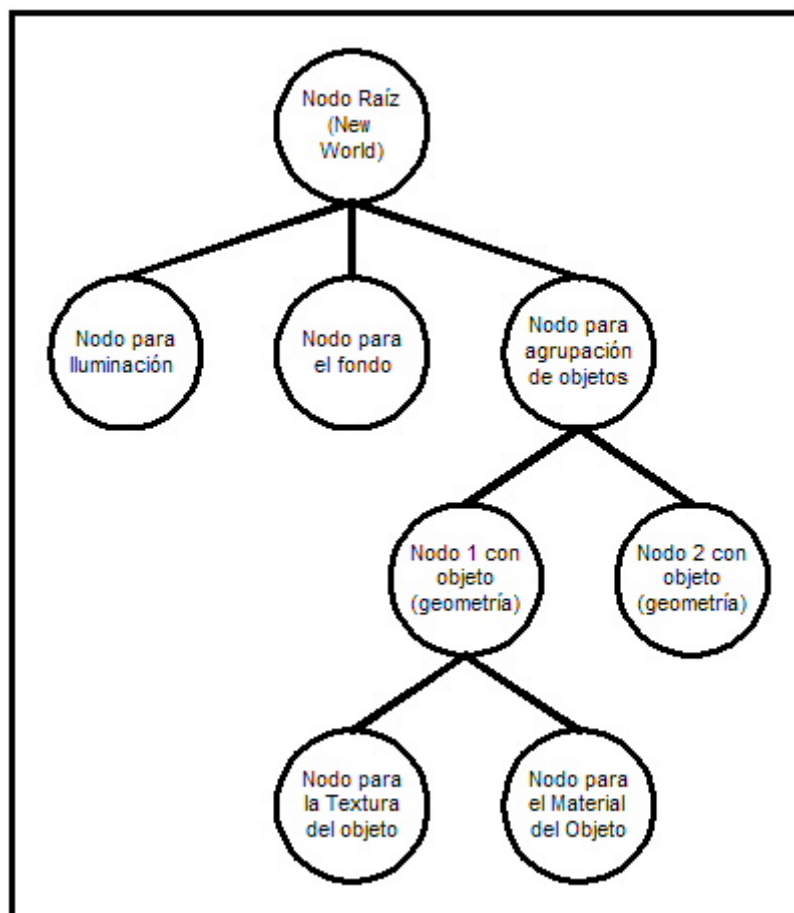


Figura 3.3: Estructura jerárquica de un árbol de nodos.

El mundo virtual creado con el árbol de nodos puede verse a través de una ventana de visualización, a la derecha de la ventana de nodos como muestra la figura 3.4.

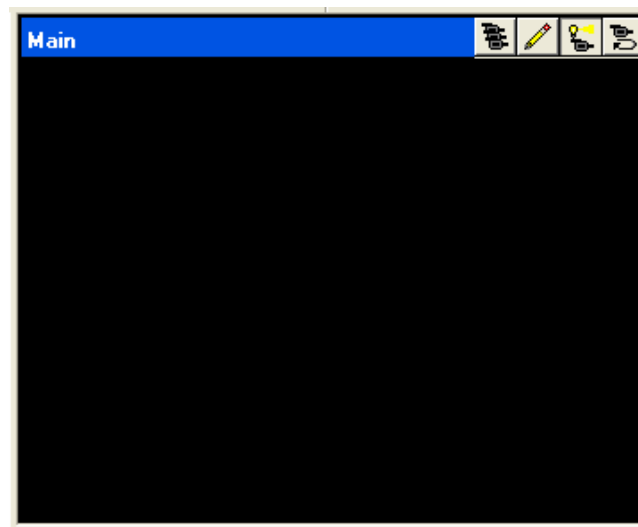


Figura 3.4: Ventana de visualización del V-Realm Builder.

El sistema de coordenadas del *V-Realm Builder* es el que viene representado por la siguiente figura 3.5.

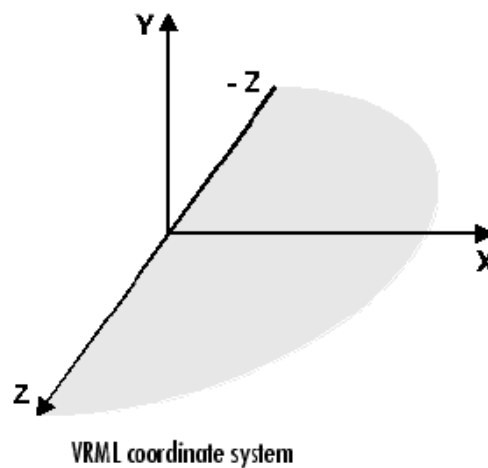



Figura 3.5: Sistema de coordenadas del mundo virtual.

3.1.1.2. Crear objetos tridimensionales con V-Realm Builder.

Inicialmente se va a explicar la forma de construir objetos tridimensionales sencillos y seguidamente, se comentará cómo a partir de esos objetos sencillos pueden generarse elementos tridimensionales más complejos.

Se comienza mostrando los nodos necesarios para la construcción de objetos sencillos:

- **Nodo Shape (Forma)** . Es el nodo que determina la forma y dimensiones del objeto que se está creando. Posee dos campos (ver figura 3.6):

- *Appearance (apariencia)*: Es un nodo hijo del nodo *Shape* que especifica los atributos visuales como el material y la textura del objeto.
- *Geometry (geometría)*: Es también nodo hijo de *Shape* y determina la geometría del objeto. Si el nodo *geometry* no tiene ningún elemento asignado, el objeto no es dibujado. Por defecto, viene vacío.

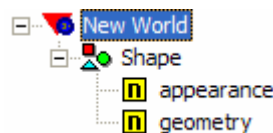



Figura 3.6: Nodo Shape y sus campos.

- **Nodo appearance** . Pulsando el icono cuando está seleccionado el nodo *appearance*, se le asignan los campos *material* y *textura*, tal y como viene reflejado en la figura 3.7:

- *Material*: Especifica las propiedades del material asociado al nodo *geometry*.
- *Texture (Textura)*: Define el mapa de textura asociado a la geometría. La textura viene definida en un fichero (*.gif).

Estos campos son nodos hijo de *appearance* y de *Shape*.

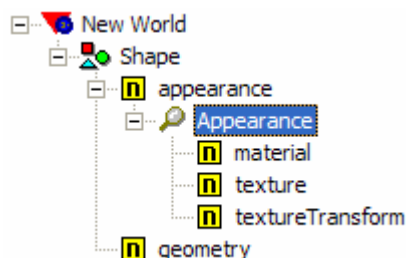



Figura 3.7: Nodo appearance y sus campos.

- **Nodo material** . Pulsando el icono cuando está seleccionado el nodo material, se le asignan los campos: *ambientIntensity*, *diffuseColor*, *emissiveColor*, *shininess* y

transparency como se observa en la figura 3.8. Todos estos campos toman valores entre 0 y 1.

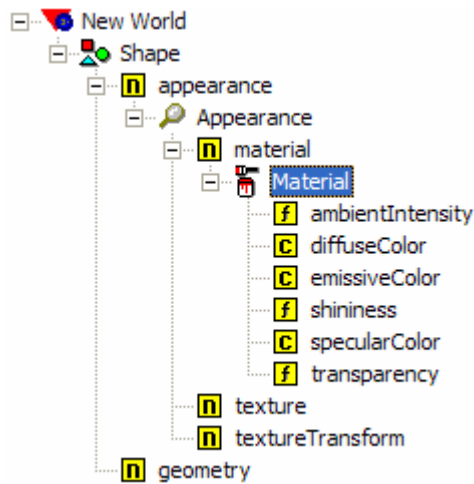



Figura 3.8: Nodo material y sus campos.

Adicionalmente se ofrece una librería de materiales en una ventana flotante que da la disponibilidad de una amplia gama.

Pulsando el botón (*Show/Hide Material Library*)  y arrastrando con el ratón el material escogido hacia el nodo correspondiente se le asignan los campos, pero con los valores asociados al material elegido. La figura 3.9 muestra la ventana a través de la cual se puede seleccionar alguno de los materiales predefinidos.

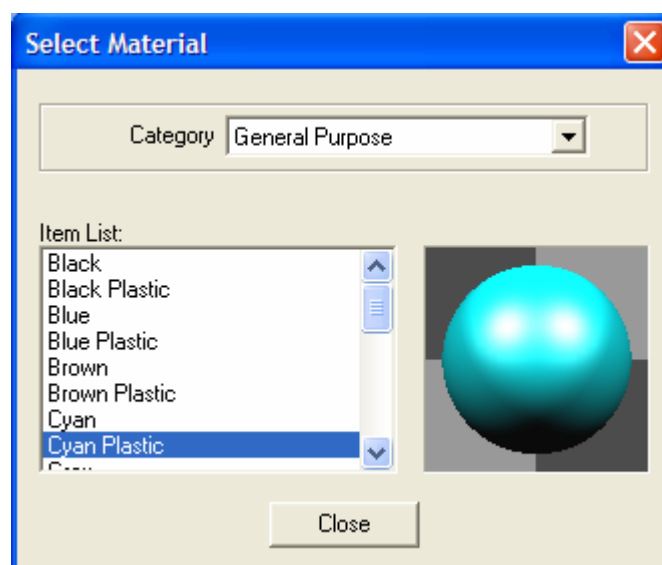



Figura 3.9: Ventana de la librería de materiales.

- **Nodo Texture (Textura)**  Pulsando el icono cuando está seleccionado el nodo *Texture*, se le asigna una serie de campos de entre los que destaca *url*. Con este campo se establece la dirección del fichero que posee la textura que se le asigna al objeto seleccionado. (Ver figura 3.10).

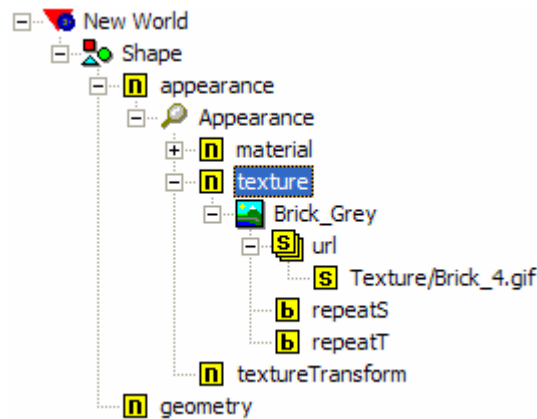



Figura 3.10: Nodo texture y sus campos.

Adicionalmente se ofrece una librería de texturas en una ventana flotante que da la disponibilidad de una amplia gama.

Pulsando el botón (*Show/Hide Textura Library*)  y arrastrando con el ratón la textura escogida hacia el nodo, se le asigna el *url* de esa textura. La figura 3.11 muestra la ventana que permite la selección de una textura predefinida en *V-Realm Builder*.

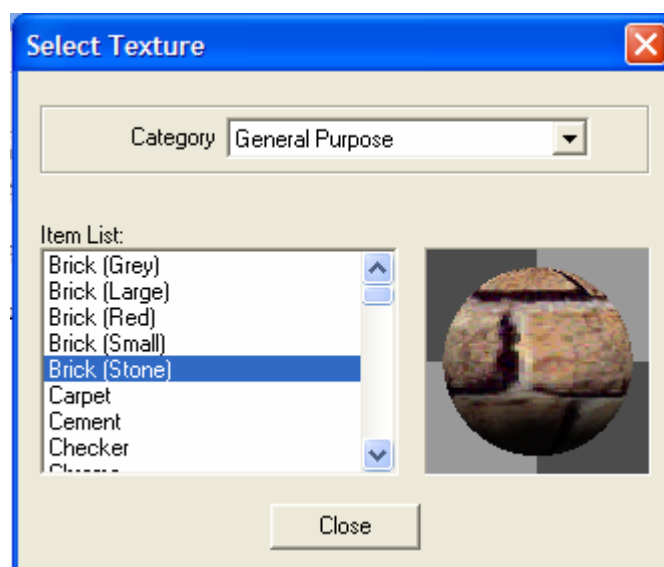


Figura 3.11: Ventana de la librería de texturas.

- **Nodo geometry.** A este nodo se le pueden asignar geometrías primitivas, tales como: *Box*, *Cone*, *Cylinder*, *Sphere*, *Text*, *Elevation Grid*, *Extrusion* e *Indexed Face Set*. Cada una de estas geometrías vienen representadas mediante los iconos mostrados en la figura 3.12.



Figura 3.12: Iconos de las distintas geometrías.

Seleccionando el nodo *geometry* y pulsando cualquiera de los botones anteriores, aparecerá dibujado, en la ventana de visualización del mundo virtual, un objeto con la geometría elegida y con el material y textura establecidos con anterioridad. La figura 3.13 muestra un ejemplo de nodo *geometry*.



Figura 3.13: Nodo *geometry* con la geometría de un cubo (*Box*).

El punto de referencia de las geometrías creadas es su centro y en definitiva, cualquier desplazamiento de un objeto se hará con respecto a este punto. Además, la modificación de las dimensiones de una geometría (objeto) tendrá en cuenta el punto de referencia, ya que una mitad de las dimensiones de la geometría será en el sentido positivo de los ejes de coordenadas y la otra en sentido negativo. La figura 3.14 muestra lo comentado pero en una sola coordenada de las tres que posee el objeto. Los puntos verdes de la imagen determinan la referencia del objeto.

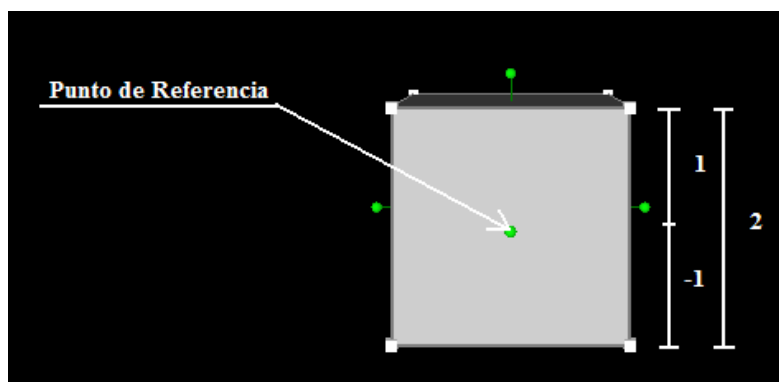


Figura 3.14: Referencia de los objetos creados en V-Realm Builder.

-
- **Box.** Es un cubo centrado en las coordenadas (0,0,0) correspondientes a los ejes (X,Y,Z). Por defecto el tamaño es de 2 unidades en cada dimensión, de -1 a 1 en cada eje. Posee el campo *Size*, que establece la dimensión del cubo según indica la ventana flotante de la figura 3.15 y debe ser mayor a 0.0.

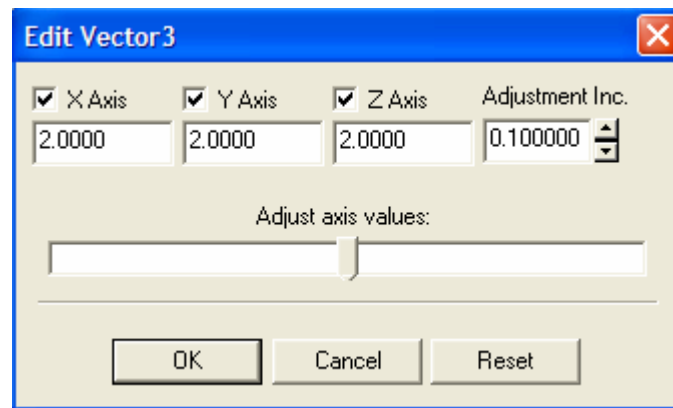



Figura 3.15: Ventana para establecer el tamaño de la geometría.

- **Cone.** Es un cono que está centrado en las coordenadas (0,0,0), cuya altura está orientada en el eje Y. Posee cuatro campos:
 - *BottomRadius*: Especifica el radio de la base del cono. Por defecto, tiene el valor de 1. Debe ser mayor que 0.0.
 - *Height*: Especifica la altura del cono desde el centro de la base. Por defecto, tiene el valor de 2 unidades que va de -1 a 1 en el eje Y. Debe ser mayor que 0.0.
 - *Bottom*: Se trata de un booleano que está a *TRUE*. Si se colocase a *FALSE* desaparecería la base del cono.
 - *Side*: Es igual al campo *bottom* sólo que en esta ocasión se trata del lado del cono.
- **Cylinder.** Es un cilindro centrado en (0,0,0) orientado en el eje Y. Posee cuatro campos:
 - *Bottom*: Se trata de un booleano que está a *TRUE*. Si se colocase a *FALSE* desaparecería la base del cilindro.
 - *Height*: La altura por defecto es de 2 unidades que va de -1 a 1 en el eje Y. Debe ser mayor que 0.0.
 - *Radius*: Especifica el radio del cilindro. Por defecto el radio es de 1. Debe ser mayor que 0.0.

- *Side*: Es igual al campo *bottom* sólo que en esta ocasión se trata del lado del cilindro.
- *Top*: Es igual al campo *bottom* sólo que en esta ocasión se trata de la cara superior.
- **Sphere**. Es una esfera centrada en (0,0,0) en el sistema de coordenadas. Posee un único campo *Radius*, que especifica el radio de la esfera y debe ser mayor que 0.0. Por defecto, el valor es 1.
- **Text**. Muestra un texto en el plano XY y el valor es establecido en el campo *fontStyle*. Para añadir el texto se debe seleccionar el nodo *fontStyle* y pulsar el botón  (*Insert font Style*).
- **ElevationGrid**. Es un rectángulo uniforme de altura variable en el plano XZ del sistema de coordenadas. Se define a través de su editor, tal y como muestra la figura 3.16.

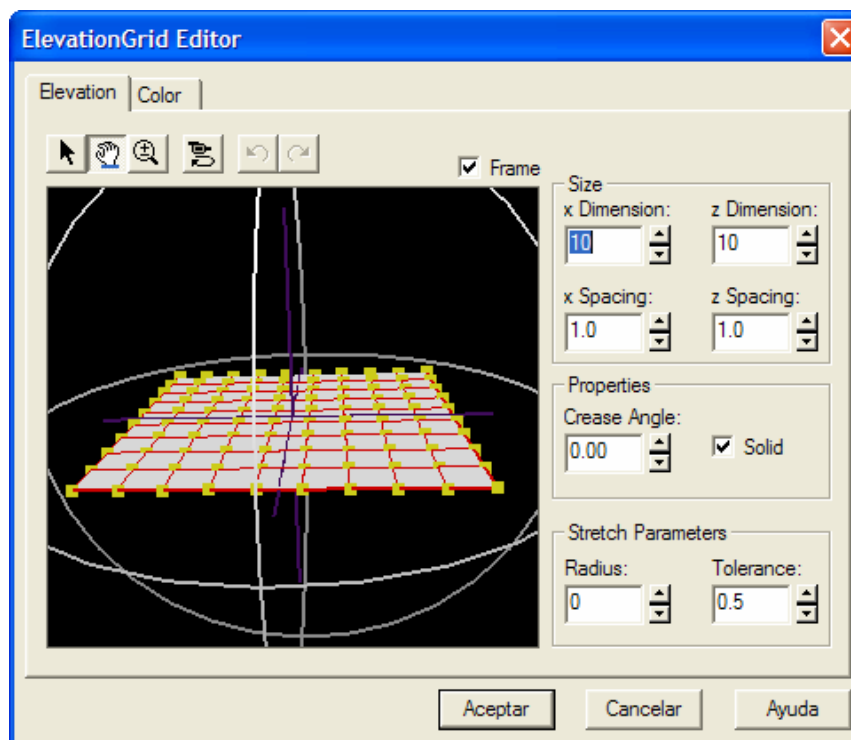


Figura 3.16: Ventana del Editor *ElevationGrid*.

- **Extrusion**. Crea elementos 3D a partir de una sección (*cross section*) 2D en el plano XZ y con el eje Y se da volumen (*spine*). Se define a través de su editor, tal y como se observa en la figura 3.17.

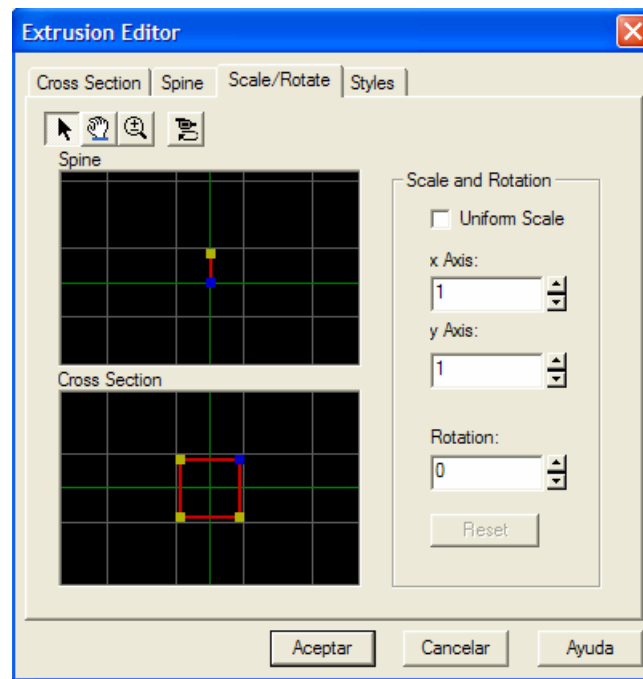


Figura 3.17: Ventana del Editor Extrusion.

- **Indexed FaceSet.** Representa formas 3D mediante la construcción de caras (polígonos). La figura 3.18 muestra el editor para crear el objeto a través de este método.

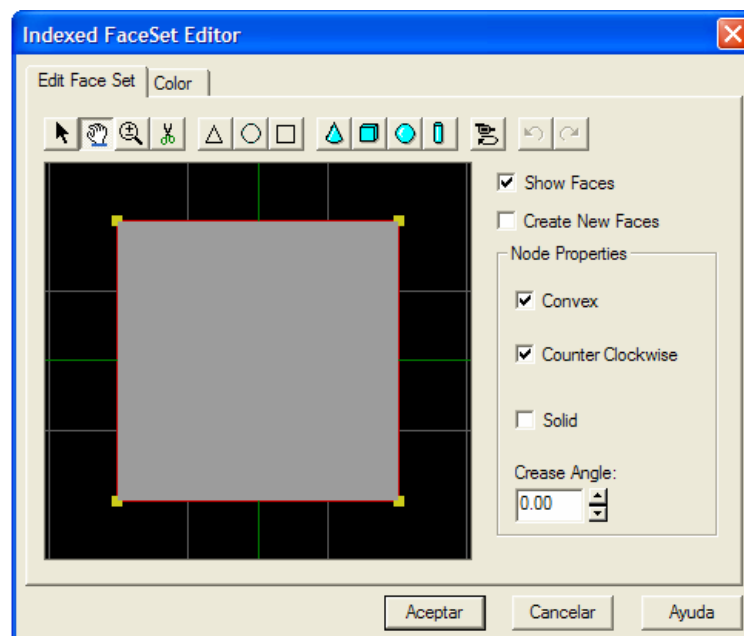


Figura 3.18: Ventana del Editor Indexed FaceSet.

A continuación, se pretende ubicar el objeto creado en una posición determinada en el mundo. Para ello se emplea:

- **Nodo Transform (Tranformación).** Agrupa todos sus nodos hijo y les define un sistema de coordenadas. Los campos más destacados son:

- *Center:* Establece el centro del objeto, inicialmente su valor es (0,0,0).
- *Rotation:* Permite el giro de los objetos hijo del nodo con respecto a cualquiera de los tres planos. El ángulo de giro debe ser un valor comprendido dentro del intervalo que va de -180° a 180°. La rotación de los objetos sigue la regla de la mano derecha.
- *Scale:* Realiza un escalado del objeto con respecto al plano indicado y el valor de escala establecido.
- *Translation:* Desplaza el objeto a la posición (X,Y,Z) indicada.
- *Children:* Hay que pinchar en este nodo y realizar todo el proceso de construcción de un objeto explicado con anterioridad. De hecho, pueden crearse todos los objetos que se deseen.

El árbol de nodos después de todo el proceso debe quedar como muestra la figura 3.19.

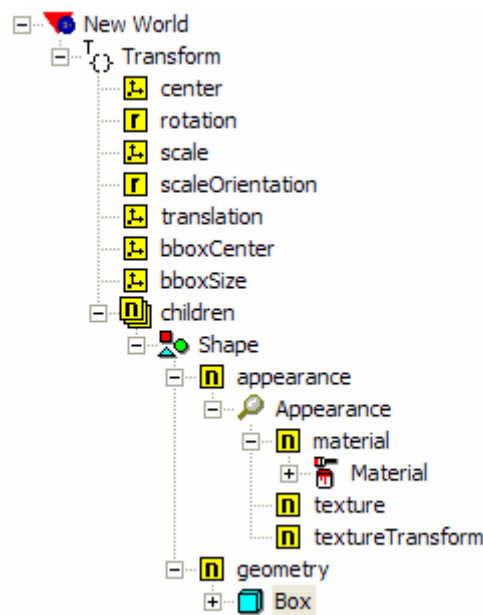



Figura 3.19: Árbol de nodos tras finalizar el proceso de construcción de un objeto simple.

Si se pulsa cualquiera de las teclas , sin haber definido previamente un nodo *Shape*, aparecerá directamente la estructura de nodos anteriormente mostrada en la figura 3.19.

El nombre de los nodos *Transform*, *Shape*, *Box*, *Material* y *Textura* puede ser modificado. El cambio del nombre ayudará a que sea más sencilla la comprensión del mundo creado.

3.1.1.3. Construir objetos tridimensionales complejos.

Partiendo de objetos sencillos y a través de una serie de nodos que agrupan otros nodos, pueden construirse objetos más complejos.

- **Nodo Group (Grupo).** Agrupa todos sus nodos hijo pero no le impone ninguna condición especial. El campo destacable es el *Children*, lugar donde se referirán todos los nodos hijo. (Ver figura 3.20).



Figura 3.20: Nodo Group y sus campos.

- **Nodo Transform.** Agrupa todos sus nodos hijo y además, les define un sistema de coordenadas. Los campos han sido explicados con anterioridad.
- **Nodo Switch (Conmutador).** Permite mostrar un nodo o ningún nodo hijo en la escena creada. Los nodos hijos envían o reciben información incluso si no son mostrados. Los campos son (ver figura 3.21):
 - o *whichChoice*: Establece qué nodo hijo va a ser mostrado. Los nodos hijo son numerados a partir del 0. Cuando no quiera mostrarse ninguno de los nodos se le dará el valor -1.
 - o *Choice*: Se colocarán todos los elementos que se pueden conmutar. Es similar al nodo *Children*.

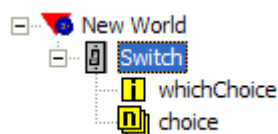


Figura 3.21: Nodo Switch y sus campos.

Los siguientes nodos de agrupación no son empleados porque poseen características que no pueden ser usadas por la manera en la que se simula la interfaz desarrollada:

- **Nodo Billboard.** Modifica sus coordenadas realizando un giro en el eje X, cambiando el punto de vista para que los nodos hijos siempre sean visibles.
- **Nodo Collision (Colisión).** Indica si los nodos que le han sido especificados han colisionado.
- **Nodo LOD (Nivel de Detalles).** Establece el nivel de detalles que posee un objeto del mundo virtual en función de la posición que ocupa en él. Debido a que un objeto muy alejado no necesita ser representado con muchos detalles, en cambio cuando está cerca necesita más. Como nodos hijo tendría el objeto con los distintos niveles de detalles.

Existen además dos nodos que permiten agrupar a otros nodos, pero para usarlos a través de la Web: *Anchor* e *Inline*. No son de interés para el desarrollo del mundo deseado.

3.1.1.4. Manipular los objetos tridimensionales.

Los objetos 3D se manipulan sobre la ventana de visualización del mundo virtual que se observa en la figura 3.22. Para realizar estas operaciones se emplea el botón izquierdo del ratón.

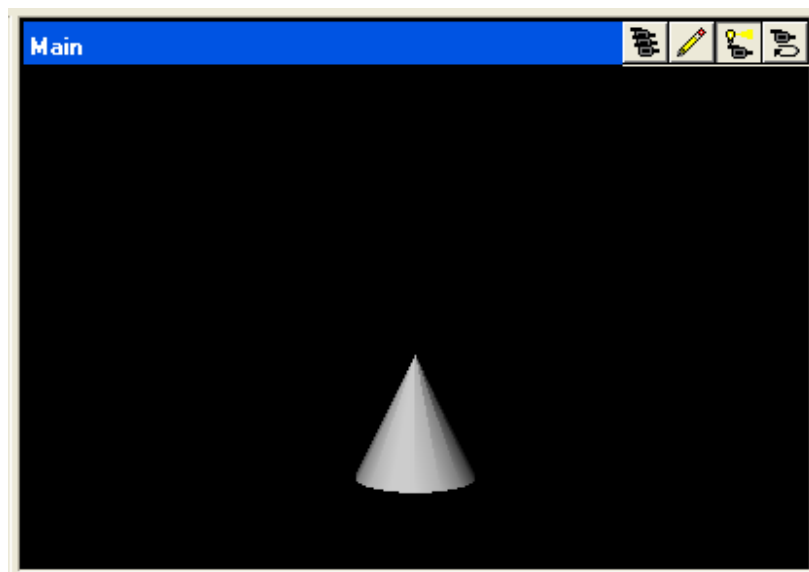
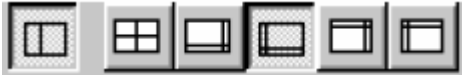



Figura 3.22: Ventana de visualización sobre la que se manipulan los objetos del mundo virtual.

En función de los botones , la configuración de la pantalla será de una manera o de otra.

- **Toggle Node Tree Window** . Si se pulsa este botón en el editor de mundos virtuales, se muestra la ventana donde aparece el árbol de nodos junto con la pantalla de visualización del mundo virtual, tal y como se ve en la figura 3.23. En este caso el mundo virtual no es más que un simple cono.

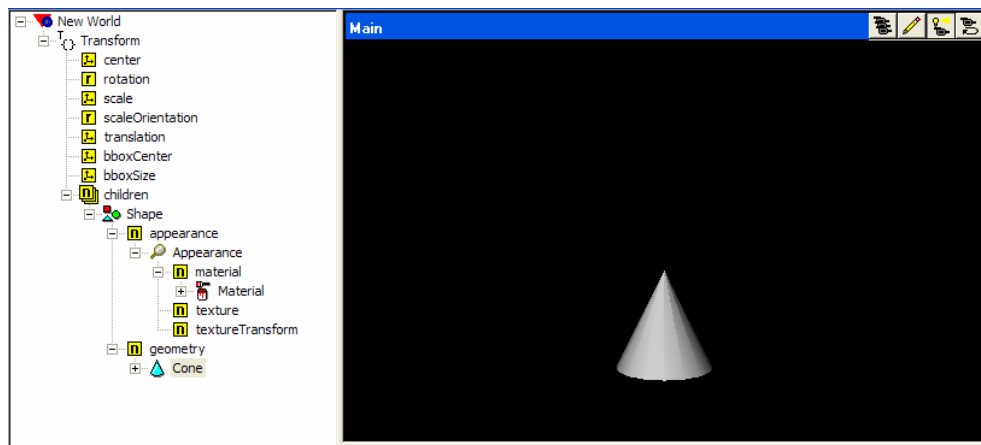



Figura 3.23: V-Realm Builder muestra la ventana de árbol de nodos y de visualización del mundo.

- **Equalize View Pane** . Muestra en pantalla los cuatro puntos de vista del mundo virtual, tal y como se observa en la figura 3.24.

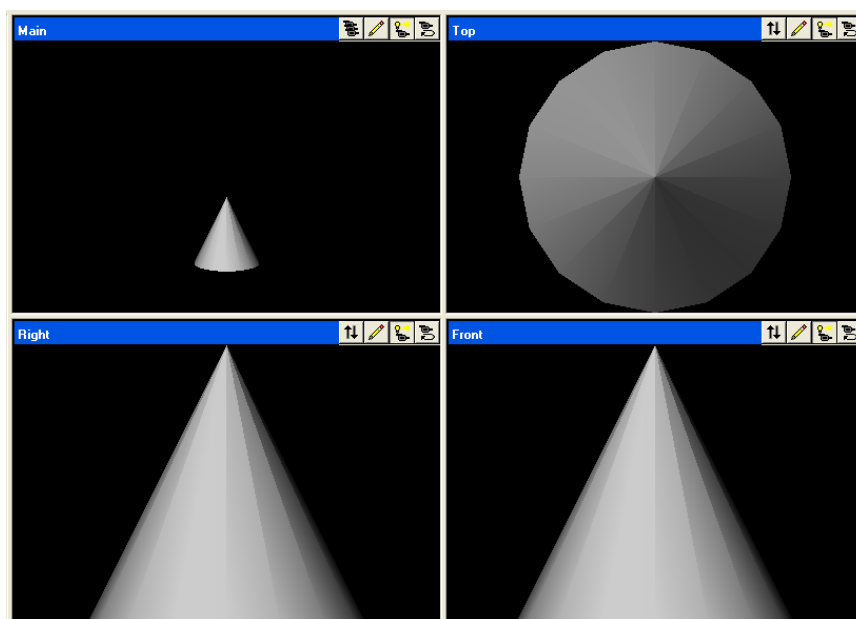



Figura 3.24: Perspectivas del mundo virtual.

-
- **View Main Pane** . Es la configuración por defecto de la ventana de visualización del mundo virtual y se ve en la figura 3.25

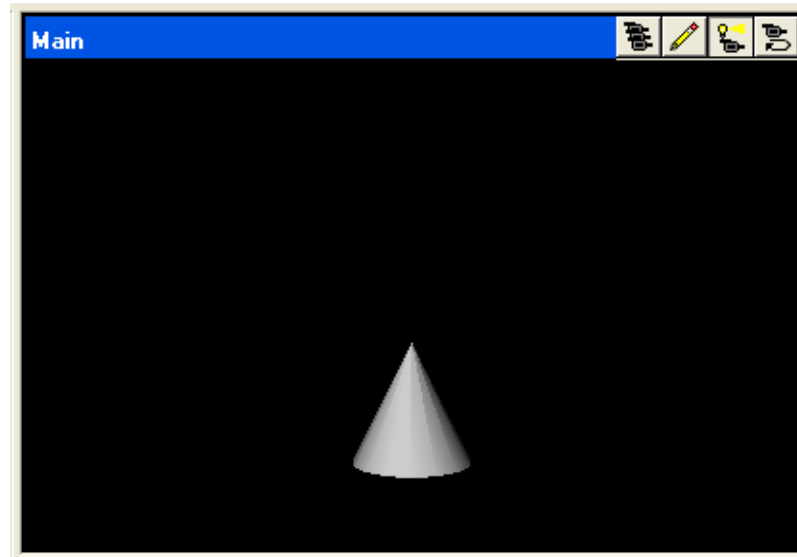



Figura 3.25: Perspectiva principal del mundo virtual.

- **View Top/Bottom Pane** . Muestra el mundo virtual a través de una ventana que tiene como punto de vista la planta. Este punto de vista es mostrado en la figura 3.26.

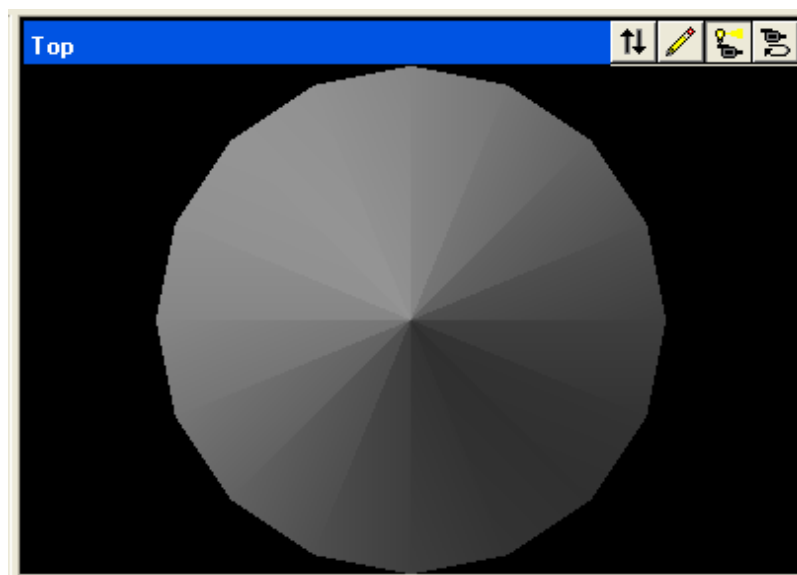



Figura 3.26: Planta del mundo virtual.

-
- **View Left/Right Pane** . Muestra el mundo virtual a través de una ventana que tiene como punto de vista el perfil. La figura 3.27 representa la situación planteada.

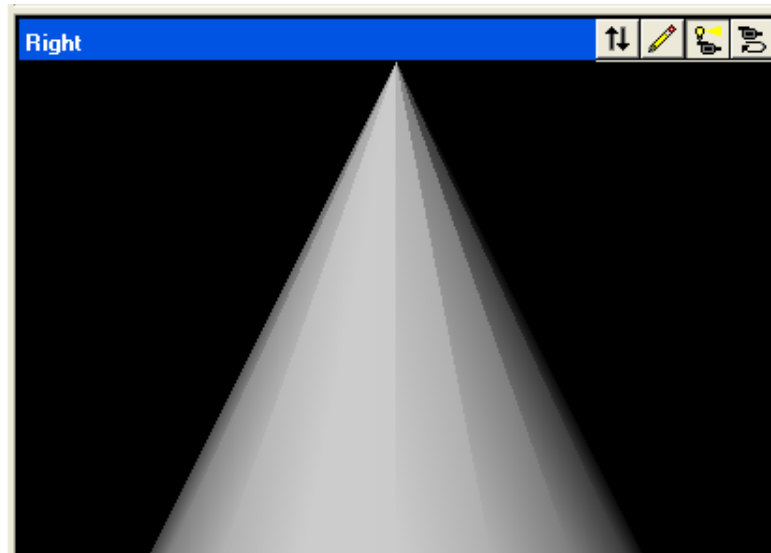



Figura 3.27: Perfil del mundo virtual.

- **View Front/Back Pane** . Muestra el mundo virtual a través de una ventana que tiene como punto de vista el alzado. Esta vista del mundo virtual se observa en la figura 3.28.

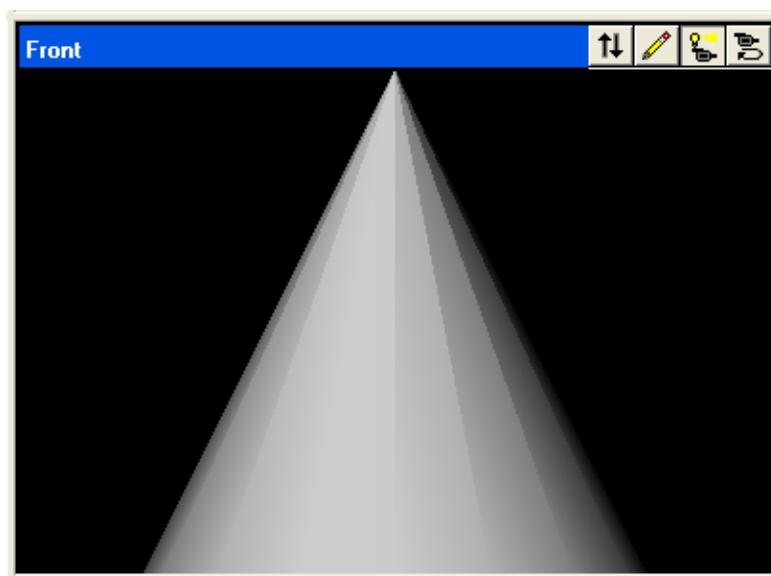








Figura 3.28: Alzado del mundo virtual.

Para poder mover los objetos y el punto de vista del mundo virtual desde la pantalla de visualización, se emplean los botones .

- **Pick Mode** . Con este modo se puede seleccionar el objeto que se desee del mundo y desplazarlo en el plano XY y en el eje Z (para realizar el desplazamiento en Z, además de seleccionar el objeto con el ratón se debe pulsar la tecla CTRL).
- **Model Mode** . Se utiliza para variar el punto de vista del mundo, pudiendo desplazarlo en cualquiera de sus ejes.
- **Pan Mode** . Se utiliza para variar el punto de vista del mundo, pero sólo en el plano XY.
- **Navigation Mode** . Se utiliza para recorrer el mundo virtual. Permite desplazamientos en cualquier dirección.


Si se desea volver al punto de vista original, porque no interesa el establecido, sólo hay que pulsar .

3.1.1.5. Diseño de un mundo virtual.

La construcción de mundos virtuales no se basa únicamente en la creación de objetos. Además puede emplearse fondos, distintos puntos de vista, fuentes de luz, librerías y sonidos, dando la posibilidad de crear mejores mundos.

En este punto se pretende explicar los nodos y herramientas que definen los elementos que se acaban de citar, existentes en el *V-Realm Builder*.

Si se desea establecer un fondo se emplean los nodos:


- **Nodo Background (Fondo)** . Se usa para definir el color del suelo y del cielo. El fondo se situará detrás de todos los elementos que aparecen en la

escena. Posee una serie de campos de entre los que se pueden destacar dos empleados para el establecimiento del color (ver figura 3.29):

- *groundColor*: Determina el color del suelo.
- *skyColor*: Determina el color del cielo.



Figura 3.29: Nodo Background y sus campos.

- **Nodo Fog (Niebla)** . Sirve para el establecimiento de efectos atmosféricos sobre los objetos. Posee los campos (ver figura 3.30):
 - *Color*: Determina el color del efecto atmosférico sobre los objetos.
 - *visibilityRange*: Determina a la distancia a la que el objeto está totalmente oscurecido por el efecto atmosférico.
 - *Set_bind*: Determina si se emplea el nodo (*TRUE*) o no (*FALSE*).

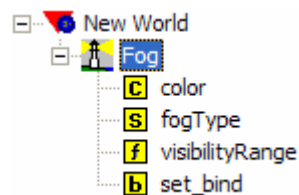



Figura 3.30: Nodo Fog y sus campos.

Si se quiere tener algún punto de vista además del definido por el editor *V-Realm Builder*, pueden crearse otros a través de:

- **Nodo ViewPoint (Punto de Vista)** . Define una ubicación específica en el sistema de coordenadas desde donde el usuario puede ver el mundo virtual. Se

pueden definir varios puntos de vista, pero sólo estará activo uno de ellos. Sus campos más significativos son (ver figura 3.31):

- *Position*: Establece la posición del punto de vista.
- *Orientation*: Establece la orientación del punto de vista con respecto a cualquiera de los tres ejes de coordenadas.
- *Set_bind*: Determina si está activo (*TRUE*) el punto de vista o no (*FALSE*).

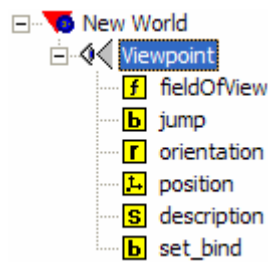





Figura 3.31: Nodo Viewpoint y sus campos.

Si existen varios puntos de vista, el botón (*Viewer Position List*)  permite seleccionar alguno de ellos.

En cambio, si se desea volver al punto de vista inicial se pulsará el botón *Reset Viewer* .

Para establecer una luminosidad en la escena del mundo virtual se pueden emplear los siguientes nodos:

- **Nodo DireccionalLight (Luz direccional)** . Define una fuente de luz que ilumina en una determinada dirección. Posee los campos (ver figura 3.32):
 - *ambientIntensity*: Establece la intensidad ambiental, y su valor varía entre 0 y 1.
 - *Color*: Establece el color de la luz.
 - *Direction*: Establece la dirección hacia donde emite la fuente de luz.
 - *Intensity*: Establece la intensidad de la luz.
 - *On*: Establece si la fuente de luz está activa (*TRUE*) o no (*FALSE*).

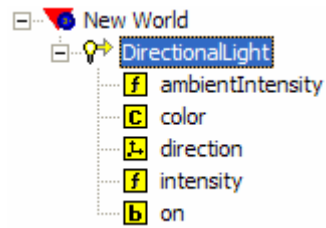



Figura 3.32: Nodo *DirectionalLight* y sus campos.

- **Nodo PointLight (Punto de Luz)** . Define una fuente de luz que emite luz en todas las direcciones, es decir, es omnidireccional. Además de los campos antes mencionados este nodo de luz posee el campo *Radius*, que establece el radio de acción de la fuente de luz. (Ver figura 3.33).

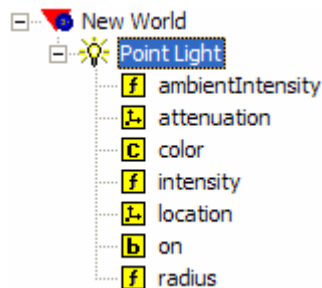



Figura 3.33: Nodo *PointLight* y sus campos.

- **Nodo SpotLight (Foco de Luz)** . Define una fuente de luz que emite desde un punto específico en una determinada dirección. Además de los campos anteriores posee (ver figura 3.34):
 - o *beamWidth*: Establece el ancho del rayo de luz emitido.
 - o *cutoffAngle*: Establece el ángulo de exclusión de luz.

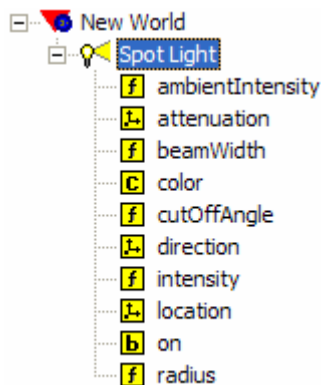





Figura 3.34: Nodo *SpotLight* y sus campos.

Para que la fuente de luz sea visible se emplea el botón (*Light Manipulators*) . Al hacerse visible se puede desplazar por el mundo.

Si se desea insertar objetos existentes se utilizan:

- **Use (Copia de Uso)** . Se emplea para no tener que duplicar información. Si existe un objeto en el mundo que se necesita usar de nuevo, en lugar de copiarlo se crea un *Use*. La copia de uso lo que hace es utilizar la definición de un objeto anteriormente implementado para no tener que volver a crearlo.
- **Librerías de objetos** . Permite al usuario añadir objetos creados directamente en el mundo. Los objetos de la librería están agrupados por categorías. (Ver figura 3.35).

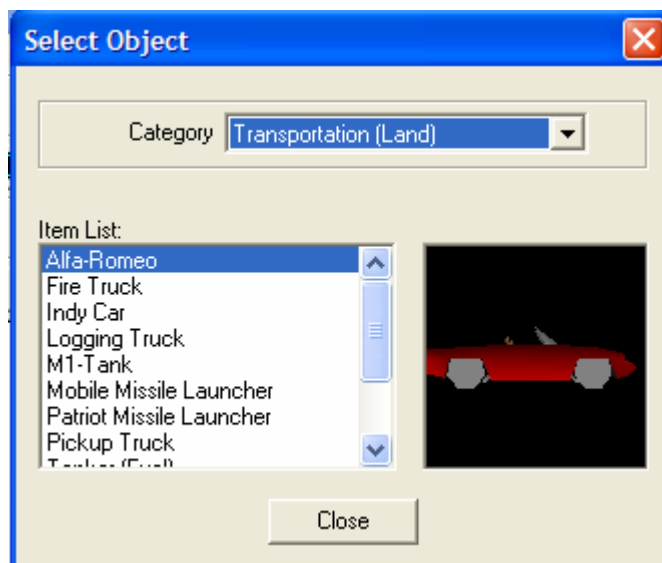




Figura 3.35: Ventana de la librería de objetos.

La inserción de sonido en los mundos virtuales se hace a través de:

- **Nodo Sound (Sonido)** . Describe un sonido en un punto exacto del espacio.
- **Nodo AudioClip** . Es una fuente de datos de audio. El nodo *Sound* requiere este tipo de información. Por este motivo, en el campo *source* del nodo *Sound* se inserta el nodo *AudioClip*.

Los nodos de sonido no son empleados porque la ventana del *Virtual Reality Toolbox* por donde se observa el mundo virtual no reproduce sonido. En definitiva, el uso de los nodos de sonido es inútil si se utilizan para el desarrollo de la interfaz gráfica.

Para finalizar, el mundo virtual permite insertar información sobre él:


- **Nodo worldInfo** . Contiene información sobre el mundo. Este nodo no afecta a la apariencia visual del mundo. Posee dos campos (ver figura 3.36):
 - o *title (Título)*: Es el título que aparece en la ventana de visionado, cuando se simula el mundo virtual.
 - o *Info*: En él se incluye otro tipo de información del mundo virtual, como por ejemplo, el autor.



Figura 3.36: Nodo WorldInfo y sus campos.

3.1.2. Virtual Reality Toolbox.

Se usa para interactuar con modelos de Realidad Virtual a través de MATLAB o Simulink. En el proyecto se utiliza el entorno MATLAB, debido a que el mundo virtual debe manejarse mediante comandos escritos en scripts y funciones en ficheros M (*.m) ya definidos.

Los mundos son creados con el estándar VRML (*Virtual Reality Modeling Language*). El editor empleado para crear y editar código VRML es el *V-Realm Builder*, como se ha comentado en el apartado anterior.

Existen dos maneras de visualizar el mundo virtual. La primera es la ventana de visionado de mundos virtuales que posee el *Virtual Reality Toolbox* y la segunda es empleando el *blaxxun Contact* si el visionado se realiza a través de la Web. En el proyecto se emplea la ventana de visionado interna del *Virtual Reality Toolbox*.

MATLAB puede tomar ficheros-M como entrada y generar aplicaciones que poseen la funcionalidad del *Virtual Reality Toolbox*, incluyendo la ventana de visionado. Las aplicaciones MATLAB que incluyen la funcionalidad del *Virtual Reality Toolbox* (aplicaciones *stand-alone*) tienen las siguientes limitaciones:

- No soporta Simulink, ya que no puede acceder a la librería (*vrlib*) de *Virtual Reality Toolbox Simulink*.
- No tiene la opción de servidor de *Virtual Reality Toolbox*, por tanto no permite conexiones remotas. Debido a este motivo, no se puede usar la ventana de visionado *blaxxun Contact*.
- No se pueden grabar animaciones.
- Las siguientes características de la ventana de visionado de *Virtual Reality Toolbox* no pueden ser usadas en este tipo de aplicaciones:
 - *File > Open in Editor*.
 - *Recording menu*.
 - *Simulation menu*.
 - *Help*.

Para poder usar las características de los mundos virtuales, hay que escribir un fichero-M que use la interfaz de MATLAB para *Virtual Reality Toolbox* (por ejemplo, crear, abrir, y cerrar mundos virtuales).

MATLAB provee comunicación para el control y manipulación de objetos de Realidad Virtual usando objetos MATLAB. Después de crear estos objetos y asociarlos a un mundo virtual, se puede controlar a través de funciones.

3.1.2.1. Virtual Reality Toolbox Viewer.

La ventana de visionado del *Virtual Reality Toolbox* es la que se muestra en la figura 3.37.

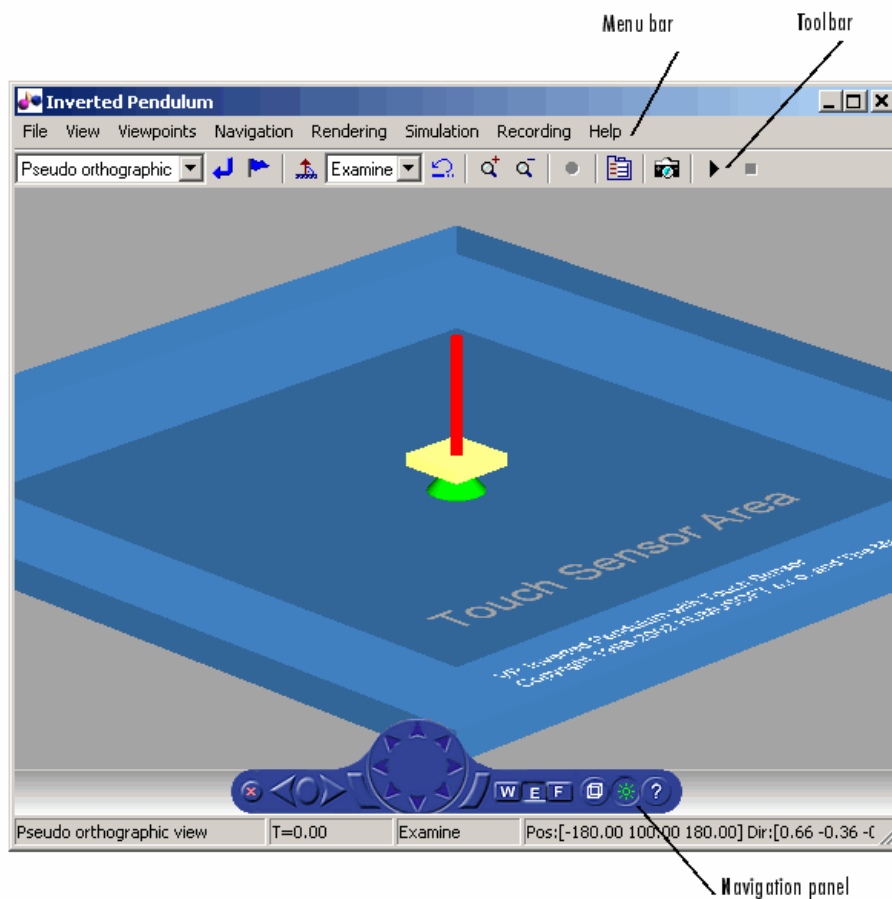


Figura 3.37: Ventana de visionado de mundos virtuales.

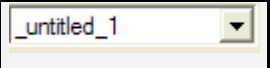

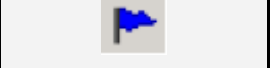
La barra de menús (Menu Bar): La tabla 3.1 muestra las funciones que se pueden realizar con la barra de menús.

FUNCIONES DE LA BARRA DE MENÚS		
File	Operaciones generales sobre un fichero	
	Opción	Función
	<i>New Window</i>	Abre otra ventana para la escena virtual. Es útil si se desea tener varias vistas del mundo
	<i>Open in Editor</i>	Abre el mundo virtual en el editor de entornos virtuales (<i>V-Realm Builder</i>)
	<i>Reload</i>	Vuelve a cargar el mundo virtual con su configuración inicial
	<i>Save as</i>	Salva los cambios realizados en el mundo virtual
	<i>Close</i>	Cierra la ventana de visionado

View	Permite personalizar la ventana de visionado de Realidad Virtual	
	Opción	Función
	<i>Toolbar</i>	Muestra los botones de la barra de herramientas
	<i>Status Bar</i>	Muestra la barra de estado en la parte inferior de la ventana de visionado. Incluye: el punto de vista, el tiempo de simulación, el método de navegación, la posición y dirección de la cámara
	<i>Navigation Zones</i>	Muestra las zonas de navegación
	<i>Navigation Panel</i>	Establece si aparecerá o no el panel navegación
	<i>Zoom In/Out</i>	Aumenta o disminuye el zoom
	<i>Normal (100%)</i>	Vuelve al zoom normal
Viewpoints	Maneja los puntos de vista del mundo virtual	
Navigation	Maneja la navegación de la escena virtual	
Rendering	Maneja la interpretación de la escena	
Simulation	Maneja el inicio y la parada del modelo y sus parámetros	
Recording	Maneja la captura de una animación a través de sus parámetros	
Help	Muestra la ayuda de la ventana de visionado	

Tabla 3.1: Funciones que se pueden realizar con la barra de menús.

Barra de herramientas (Toolbar): En ella se encuentra los botones con las operaciones más usadas, tal y como muestra la tabla 3.2.

FUNCIONES DE LA BARRA DE HERRAMIENTAS	
Botón	Función
	Listado de todos los puntos de vista existentes en el mundo virtual
	Botón que permite volver al punto de vista seleccionado en la lista anterior
	Botón para crear un punto de vista




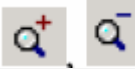





	Botón para enderezar el punto de vista. Es decir, formando un ángulo de 90° con respecto al suelo
	Listado de las opciones de navegación por el mundo virtual: <i>Walk</i> , <i>Examine</i> y <i>Fly</i>
	Botón de deshacer
	Botón de <i>zoom in/out</i>
	Botón de grabación. Sirve para iniciar y parar la grabación
	Botón de los parámetros empleados en el modelo (para Simulink)
	Botón para la captura de imágenes
	Botón para inicio/pausa/continuación de la simulación
	Botón de parada de la simulación

Tabla 3.2: Funciones que se permiten realizar con la barra de herramientas

Panel de navegación (Navigation Panel): Posee los controles para desplazarse por el mundo, como muestra la figura 3.38.

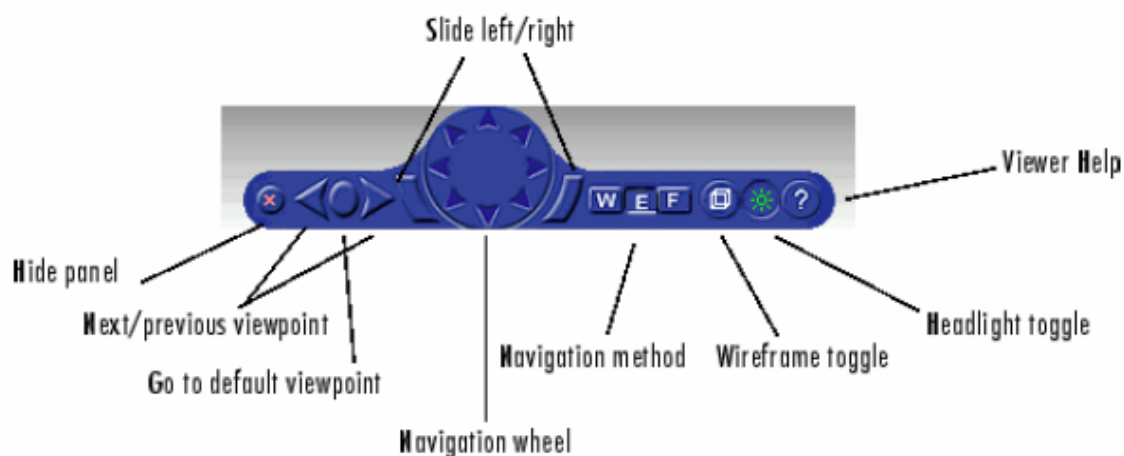


Figura 3.38: Panel de navegación de la ventana de visionado.

En la tabla 3.3 se describen cada uno de los elementos que posee el panel de navegación.

FUNCIONES DEL PANEL DE NAVEGACIÓN	
Botón	Función
<i>Hide panel</i>	Permite ocultar el panel de navegación
<i>Next/previous viewpoint</i>	Da la opción de seleccionar un punto de vista de la lista
<i>Return to default viewpoint</i>	Vuelve al punto de vista original
<i>Slide left/right</i>	Permite desplazar el punto de vista a izquierda y derecha
<i>Navigation Wheel</i>	Mueve el punto de vista en ocho posibles direcciones
<i>Navigation method</i>	Maneja la navegación por la escena virtual
<i>Wireframe toggle</i>	Representa el mundo virtual sólo a través de líneas
<i>Headlight toggle</i>	Activa o desactiva la fuente de luz del mundo creado
<i>Help</i>	Ayuda

Tabla 3.3: Funciones que se pueden realizar mediante el panel de navegación

3.1.2.2. Funciones.

A continuación, se explicarán detalladamente las funciones MATLAB que se pueden emplear para el manejo de los mundos virtuales.

3.1.2.2.1. Funciones de la interfaz de MATLAB.

- ***vrdrawnow***. Actualiza el mundo virtual. Los cambios se almacenan en una cola y son borrados cuando se realizan. La ventana de visionado es actualizada en los tiempos muertos de MATLAB (cuando ningún fichero-M está siendo ejecutado).
- ***vrgetpref***. Valores de las preferencias del *Virtual Reality Toolbox*. Las preferencias posee los valores de configuración de la interfaz entre MATLAB y el *Virtual Reality Toolbox*.
x=vrgetpref. Devuelve los valores de todas las preferencias del *Virtual Reality Toolbox*.

x=vrgetpref('nombre_preferencia'): Devuelve el valor de la preferencia específica.

x=vrgetpref('factory'): Devuelve los valores por defecto de todas las preferencias.

- ***vrinstall***. Instala y chequea los componentes del *Virtual Reality Toolbox*.
vrinstall –install
vrinstall –uninstall
vrinstall –check
- ***vrsetpref***. Cambia las preferencias del *Virtual Reality Toolbox*.
vrsetpref('nombre_preferencia', 'valor_preferencia'): Establece un valor en una determinada preferencia.
vrsetpref('factory'): Establece en todas las preferencias los valores por defecto.
- ***vrwho***. Realiza un listado los mundos virtuales cargados en memoria.
- ***vrwhos***. Realiza un listado de los detalles de los mundos virtuales cargados en memoria.

3.1.2.2.2. Métodos del objeto *vrworld*.

- ***vrworld***. Crea un objeto *vrworld* que representa un mundo virtual.
Mi_mundo=vrworld('nombre_fichero'): El mundo virtual se carga desde un fichero VRML donde se guarda. Si no se especifica la extensión, por defecto es .wrl.
- ***vrworld/close***. Cierra el mundo virtual.
close(objeto_vrworld).
Este método cambia el mundo virtual del estado abierto al cerrado:
 - Si el mundo está abierto más de una vez, hay que usar un número apropiado de *close*.
 - Si el mundo virtual está ya cerrado, *close* no hace nada.

-
- ***vrworld/delete***. Borra el mundo virtual de la memoria.
delete(objeto_vrworld).

 - ***vrworld/get***. Valores de las propiedades del objeto *vrworld*.
get(objeto_vrworld): Devuelve los valores de todas las propiedades del objeto *vrworld* en la ventana de comandos de MATLAB.
x=get(objeto_vrworld): Guarda en una variable los valores de todas las propiedades.
x=get(objeto_vrworld, 'nombre_propiedad'): Devuelve el valor de una propiedad dada.

 - ***vrworld/nodes***. Realiza un listado de los nodos disponibles en el mundo virtual.
nodes(objeto_vrworld, '-full'): Muestra los nodos a través de la ventana de comandos de MATLAB.
x=nodes(objeto_vrworld, '-full'): Guarda en una variable todas los nodos del mundo virtual.
'-full': Es opcional, permite obtener además del listado de nodos, la lista de sus campos.

 - ***vrworld/open***. Abre el mundo virtual.
open(objeto_vrworld).

 - ***vrworld/reload***. Vuelve a cargar el mundo virtual desde el fichero VRML.
reload(objeto_vrworld).

 - ***vrworld/set***. Cambia los valores de las propiedades del objeto *vrworld*.
set(objeto_vrworld, 'nombre_propiedad', 'valor_propiedad'): Establece un valor en una determinada propiedad.

 - ***vrworld/view***. Visualiza el mundo virtual.
view(vrworld_object).
x=view(vrworld_object, -internal): Abre el mundo virtual en la ventana de visionado interna de *Virtual Reality Toolbox*.
x=view(objeto_vrworld, 'web'): Abre el mundo virtual en una ventana Web.

3.1.2.2.3. Métodos del objeto *vrnode*.

- ***vrnode***. Crea un nodo o un manejador a un nodo (*objeto_vrnode*) existente en el mundo virtual.

Mi_nodo=vrnode(objeto_vrworld, 'nombre_nodo').

- ***vrnode/delete***. Borra el objeto *vrnode*.

delete(objeto_vrnode).

- ***vrnode/fields***. Devuelve los campos VRML del objeto *vrnode*.

fields(objeto_vrnode).

- ***vrnode/get***. Valores de las propiedades del objeto *vrnode*.

get(objeto_vrnode): Devuelve los valores de todas las propiedades del objeto *vrnode* a través de la ventana de comandos de MATLAB.

x=get(objeto_vrnode, 'nombre_propiedad'): Devuelve el valor de una propiedad dada.

- ***vrnode/getfield***. Valores de los campos del objeto *vrnode*.

getfield(vrnode_object): Devuelve los nombres de los campos y sus valores para un determinado nodo.

x=getfield(objeto_vrnode, 'nombre_campo'): Devuelve el valor de un determinado campo del objeto nodo.

- ***vrnode/set***. Cambia los valores en las propiedades de un nodo del mundo virtual.

x=set(objeto_vrnode, 'nombre_propiedad', 'valor_propiedad').

- ***vrnode/setfield***. Cambia el valor de un campo del objeto *vrnode*.

x=setfield(objeto_vrnode, 'nombre_campo', 'valor_campo').

3.1.2.2.4. Métodos del objeto *vrfigure*.

- ***vrfigure***. Crea una nueva figura (*objeto_vrfigure*) de Realidad Virtual.

f=vrfigure(objeto_vrworld, posición): Crea una nueva figura de Realidad Virtual en una posición específica. La posición es opcional. Se emplea en lugar del comando *view*.

- ***vrfigure/close***. Cierra la figura de Realidad Virtual.
close(objeto_vrfigure).
- ***vrfigure/get***. Valores de las propiedades del objeto *vrfigure*.
get(vrfigure_object): Listado de todas las propiedades del objeto *vrfigure*.
x=get(vrfigure_object, 'property_name'): Devuelve el valor de una propiedad específica del objeto figura.
- ***vrfigure/set***. Cambia los valores de las propiedades del objeto *vrfigure*.
set(objeto_vrfigure, 'nombre_propiedad', valor_propiedad).

3.2. Herramientas Hardware.

En este aparato se incluyen tanto el equipo de trabajo en el que se ejecutará el sistema final, como el resto de equipos necesarios para la adquisición de datos. A continuación, se enumerarán los equipos empleados:

- Ordenador *Pentium Core2Duo 1.7 GHz, 1 Giga* de memoria *RAM* y *tarjeta gráfica de 256 MB DDR3*. Además posee el sistema operativo *Microsoft Windows XP*, la versión de *Matlab 7.2*, el *Virtual Reality Toolbox 4.3*, el *Signal Processing Toolbox* y el editor *V-Realm Builder*.
- Polígrafo de 4 canales EEG, Modelo *V75-08* de *Coulbourn Instruments LabLinc*. Se utiliza para amplificar las señales EEG del sujeto, que como ya se ha comentado, presentan amplitudes pequeñas (ver figura 3.39).



Figura 3.39: Polígrafo Coulbourn Instruments LabLinc.

-
- Tarjeta de adquisición de datos *DAQCard-6024E* y drivers asociados (NI-DAQ versión 8.0.1) de *Nacional Instruments*. Es el componente encargado de traducir las señales analógicas del sujeto en formato digital para que puedan ser tratadas por el ordenador.
 - Gorro dotado de electrodos o *ElectroCap*. Es el medio para transmitir las ondas cerebrales del sujeto hasta el polígrafo. En la figura 3.40 se puede ver un ejemplo de un *ElectroCap*.



Figura 3.40: Gorro de electrodos ElectroCap.

Capítulo 4.

Descripción general del mundo virtual diseñado.

El objetivo principal del proyecto es el desarrollo de una interfaz multimodal basada en técnicas de Realidad Virtual. Este interfaz se empleará para la implementación de dos aplicaciones como son un sistema BCI y un sistema de medida de tiempos de reacción. Finalmente, esta interfaz se integrará en un sistema BCI ya existente, desarrollado por el grupo DIANA del Departamento de Tecnología Electrónica con la idea de poder reducir el tiempo de entrenamiento del sistema, aumentando la concentración y la motivación del sujeto.

El capítulo pretende describir la interfaz implementada en el proyecto. Como ya se ha comentado, la interfaz será un mundo virtual. El desarrollo del mundo virtual vendrá determinado por el paradigma de entrenamiento del sistema BCI propuesto en el proyecto. Por tanto, es interesante explicar el paradigma de entrenamiento para entender las características del mundo virtual diseñado.

4.1. Introducción al mundo virtual implementado.

El mundo virtual implementado es básicamente el mismo para las dos aplicaciones creadas. De hecho, el entorno virtual del sistema de medida de tiempos de reacción es una particularización de la del sistema BCI.

El escenario virtual para ambas aplicaciones será una carretera rectilínea de tres carriles en la que un vehículo avanza progresivamente. Al mismo tiempo, se puede desplazar a izquierda y a derecha, para esquivar una serie de obstáculos que salen a su paso.

A continuación, se describirán los elementos que constituyen el paradigma de entrenamiento del mundo virtual para el sistema BCI, así como su función en la escena virtual. Seguidamente se explicará la escena virtual del sistema de medida de tiempos de reacción, estableciendo las diferencias con la interfaz del sistema BCI.

Partiendo de la breve descripción anterior, se puede explicar el paradigma de entrenamiento del sistema BCI a partir de los elementos de los que consta el mundo virtual:

- Una **carretera** que constituye la escena virtual principal, ayudando al sujeto a aislarse del exterior y a adentrarse en el mundo virtual.
- Un **coche** como el elemento que proporciona el biofeedback al sujeto. El coche plasma visualmente el resultado de la tarea mental, moviéndose a izquierda o a derecha de la carretera. Por este motivo, al coche se le denomina **objeto biofeedback**.
- Una serie de **obstáculos** que aparecerán en la carretera y que debería evitar el sujeto controlando el desplazamiento del coche hacia el lado adecuado. Los obstáculos implementados son:
 - **Un charco:** Si este obstáculo aparece en el entrenamiento, se situará en uno de los carriles laterales de la calzada, indicando al sujeto que debe desplazar el objeto biofeedback (coche) al lado contrario. El charco aparecerá en un instante indicado y estará presente durante un intervalo establecido. Por tanto, la longitud del charco vendrá determinada por este intervalo, que corresponde al espacio de tiempo en el que el sujeto tiene control sobre el coche.
 - **Un muro:** Si este obstáculo aparece en el entrenamiento, se situará en uno de los carriles laterales junto con el charco y al final de él. Por tanto, es un obstáculo que se evitará en última instancia.
 - **Una rampa:** Si este obstáculo aparece en el entrenamiento, se situará siempre al final del charco pero en el carril lateral contrario. Es un

obstáculo que hay que abordar, para que el objeto biofeedback lo salte.

La forma de superar los obstáculos permite gratificar o castigar al individuo de maneras diferentes. Con esta idea se han implementado dos tipos de realimentación o feedback:

- **Realimentación positiva:** Gratifica al sujeto por haber ejecutado correctamente la tarea mental. No produce nada excepcional cuando se equivoca el sujeto en la realización de la tarea.
- **Realimentación negativa:** Castiga al sujeto por no haber ejecutado correctamente la tarea mental y no conseguir el resultado deseado.

Los posibles obstáculos están pensados para provocar distintos efectos en el sujeto, según el tipo de feedback (positivo o negativo). Los obstáculos que implementan realimentación negativa son:

- **El muro:** Si el sujeto no realiza correctamente su tarea mental, el coche no esquivará el muro, provocando la sensación de ser castigado por no haberlo conseguido. La sensación de castigo se obtiene al recrear visualmente el efecto de una colisión.
- **El charco:** Es un obstáculo en el que también se emplea una realimentación negativa, debido a que si no se realiza bien la tarea mental, la interfaz mostrará como el objeto biofeedback está pisando el charco, produciendo una sensación de castigo en el sujeto.

La realimentación empleada en el muro y el charco posee una clara característica que las diferencia. En el caso del charco, el feedback empleado es continuo, porque durante todo el intervalo de tiempo que se pretende esquivar este obstáculo, se puede corregir la tarea mental cuando se esté realizando mal. En cambio, el muro emplea un feedback de tipo discreto, ya que este obstáculo se esquivará o no dependiendo de la tarea mental realizada durante un solo instante de tiempo. A pesar de hablar de feedback discreto para el funcionamiento del sistema se emplea feedback continuo. El feedback discreto del que se habla es de un instante temporal dentro de toda una realimentación continua en la simulación del sistema.

Finalmente, el obstáculo que implementa realimentación positiva es **la rampa**, debido a que si el sujeto realiza correctamente su tarea mental, el coche ascenderá por la rampa, la rebasará y caerá de nuevo al asfalto de la carretera, dando una recompensa visual. En cambio, si la tarea mental no se realiza bien, el coche esquivará el obstáculo rampa sin ninguna consecuencia extraordinaria. Al igual que el muro, el feedback de este tipo de obstáculo es también discreto.

Para finalizar, se explicará el paradigma de entrenamiento del sistema de medida de tiempos de reacción relacionándolo con la interfaz desarrollada. Los elementos que componen la interfaz son esencialmente los mismos:

- Una **carretera** que al igual que en la interfaz del sistema BCI constituye la escena virtual principal.
- Un **coche** como el elemento que proporciona el **biofeedback** al sujeto. Realiza la misma función que el objeto biofeedback de la interfaz del sistema BCI. Pero se diferencia en que lo que se plasma visualmente no es el resultado de una tarea mental, sino la pulsación del cursor izquierdo o derecho.
- Por último, un **obstáculo** aparecerá en la carretera y deberá ser evitado por el sujeto controlando el desplazamiento del coche hacia el lado adecuado. El obstáculo implementado en este caso es **el muro** y se situará en una posición elegida de manera aleatoria. El obstáculo deberá esquivarse para evitar la colisión.

Como se comentó al inicio de este apartado, las dos interfaces son similares. De hecho, se observa que la segunda interfaz es una particularización de las posibilidades dadas por la primera.

4.2. Descripción de la escena virtual básica.

La escena virtual básica debía ser sencilla y familiar para que influyera positivamente en la motivación y concentración del sujeto, reduciendo las posibilidades de fracaso del entrenamiento. También se ha incluido dinamismo en la escena para dar mayor credibilidad y así, estimular al sujeto en la realización de sus tareas mentales.

Con las ideas anteriores, se ha desarrollado un mundo virtual cuya escena principal es una carretera dividida en tres carriles, que discurre en línea recta. Además, alrededor de la carretera se extiende un terreno verdoso con textura más o menos homogénea, hasta donde alcanza la vista. Para las dos aplicaciones la escena virtual básica es igual.

Para la implementación se ha empleado, como se comentó en el capítulo 3, el editor *V-Realm Builder* y el *Virtual Reality Toolbox*.

4.2.1. Implementación del fondo.

El primer elemento creado en la escena virtual es el fondo, que comprende tanto el cielo como el suelo del paisaje. En este caso, se utilizó el nodo *Background* (Fondo). Ajustando los campos de color se ha obtenido el fondo mostrado en la figura 4.1.

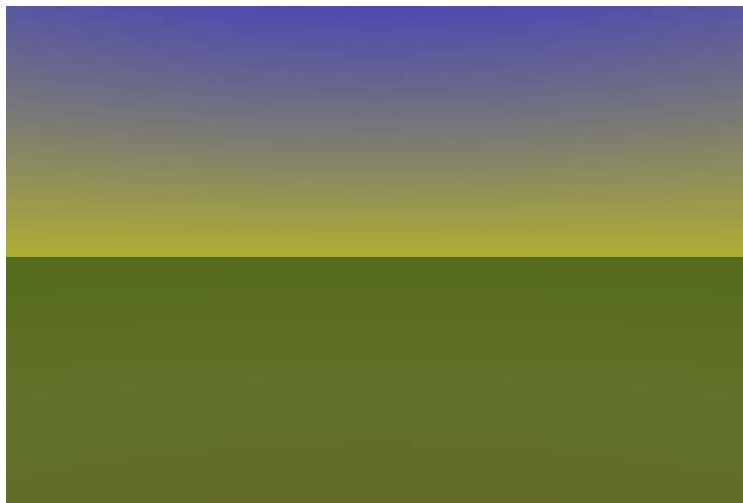


Figura 4.1: Fondo implementado en la escena virtual básica.

4.2.2. Implementación de la carretera.

La carretera implementada en la escena virtual está compuesta por el asfalto y las líneas del suelo. Las líneas serán de dos tipos: continuas y discontinuas.

El asfalto de la carretera se diseñó con un nodo a través del editor *ElevationGrid*, explicado en el apartado 3.1.1.2. La forma creada es un rectángulo alargado de color gris. La figura 4.2 muestra las dimensiones del asfalto.

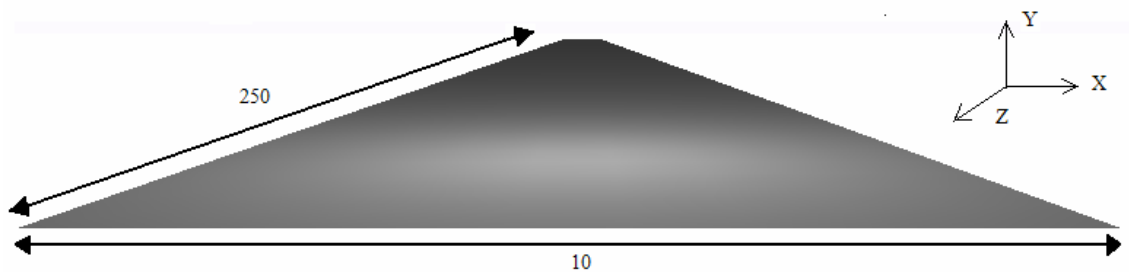


Figura 4.2: Dimensiones del asfalto de la carretera.

La longitud de la carretera es de 250 unidades porque a partir de este valor la sensación de profundidad es la misma, por tanto, era innecesario colocar un elemento mayor que cargue más los recursos del PC. Además, el ancho es de 10 unidades para que abarque toda la ventana de visionado.

Las líneas son empleadas para construir los carriles sobre los que el coche debe desplazarse. Se crearon tres carriles, para los que se necesitarán dos líneas continuas de longitud igual al asfalto y dos líneas discontinuas. Las líneas se implementaron con nodos *Box* (cubo) de color blanco. La figura 4.3 muestra la carretera con las dimensiones de las líneas.

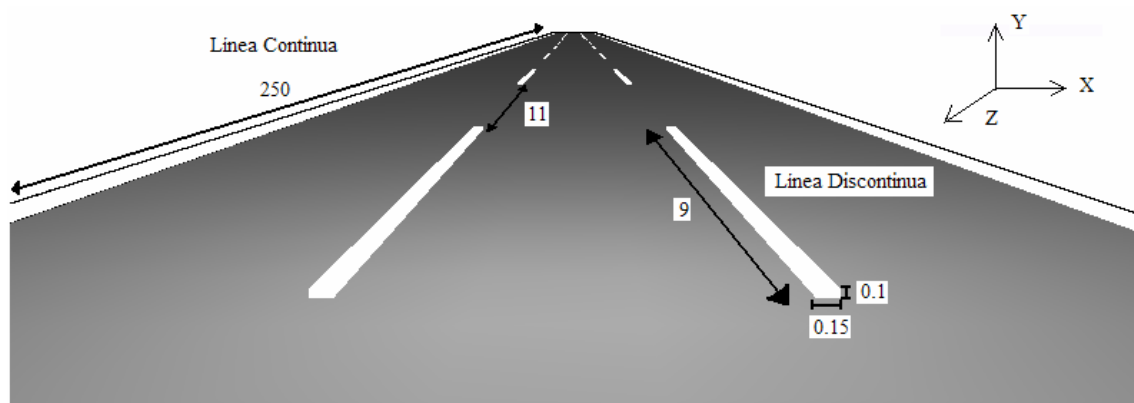


Figura 4.3: Dimensiones de las líneas de la carretera.

Las líneas discontinuas están formadas por una serie de secciones. Se ha implementado una sola sección y el resto son copias de uso (*USE*) desplazadas en el eje Z. Es decir, que todas las secciones son iguales a la original en color y dimensiones.

A lo largo de la carretera y a ambos lados, se ha dispuesto una hilera de árboles para mejorar la estética de la escena. En la creación de estas hileras de árboles también se emplean las copias de uso (*USE*). Cada árbol está separado 20 unidades del anterior y del siguiente. Además, al final de la carretera se han colocado unas montañas. El árbol y las montañas se han obtenido de la librería de objetos (*Object Library*). La figura 4.4 muestra la escena virtual completa.

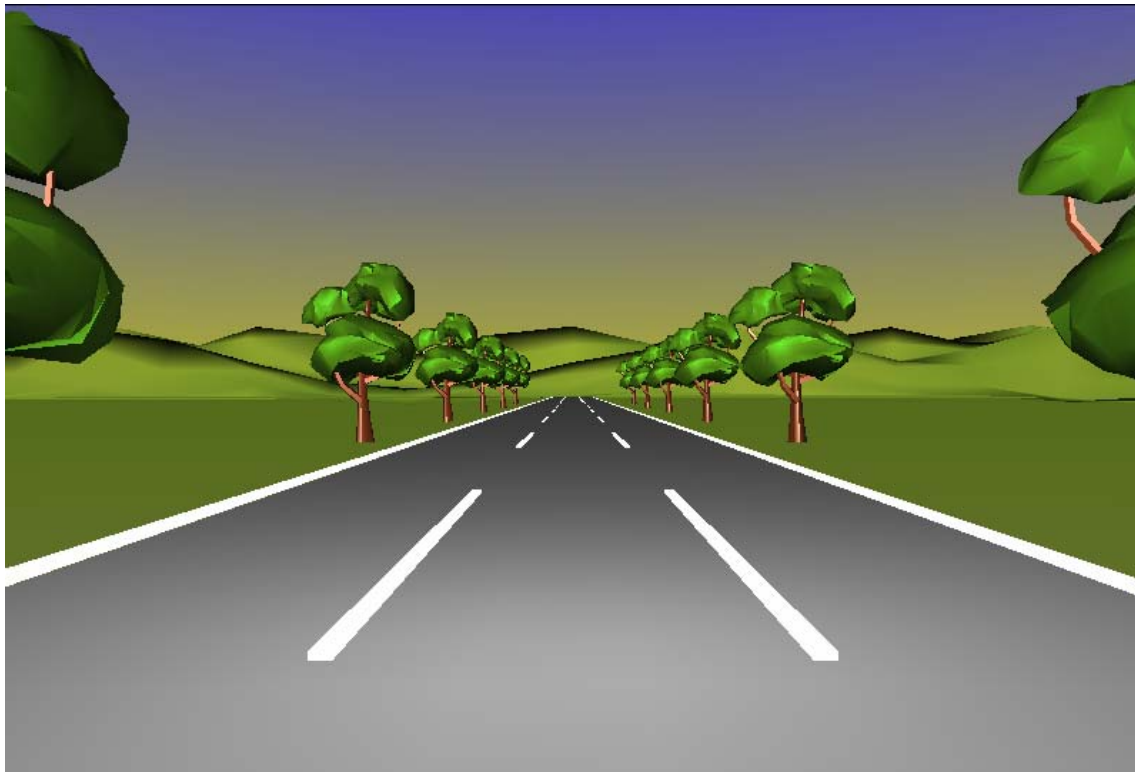


Figura 4.4: Escena virtual básica.

Tanto la carretera como el resto de elementos que forman la escena virtual básica son tratados como objetos individuales por el *V-Realm Builder*.

4.2.3. Efectos implementados sobre la escena virtual.

Una vez expuesta la escena virtual básica, se pasa a detallar los efectos implementados que imprimen realismo y dinamismo al mundo virtual.

4.2.3.1. Movimiento de la carretera.

La herramienta desarrollada mueve la carretera durante el tiempo de simulación. Esta característica es la que da dinamismo y realismo al mundo virtual, favoreciendo la concentración y sensación de inmersión del sujeto.

El movimiento se implementa desplazando algunos objetos de la escena virtual. La posibilidad de desplazar objetos viene dada por una serie de comandos MATLAB definidos en el *Virtual Reality Toolbox*, como se comentó en la sección 3.1.2.

Antes de explicar el efecto de movimiento sobre la carretera, hay que establecer el punto de vista que es el lugar desde donde se observa el mundo virtual a través de la ventana de visionado. El punto de vista se ha situado en la posición (0, 1.6, 10) de XYZ, tal y como se observa en la figura 4.5. Por este motivo, el asfalto de la carretera se ha situado en la posición (-5, 0, -240) para que se viese asfalto desde el inicio de la imagen y centrado. Las líneas del suelo y los árboles de la escena virtual parten de la posición cero en Z.

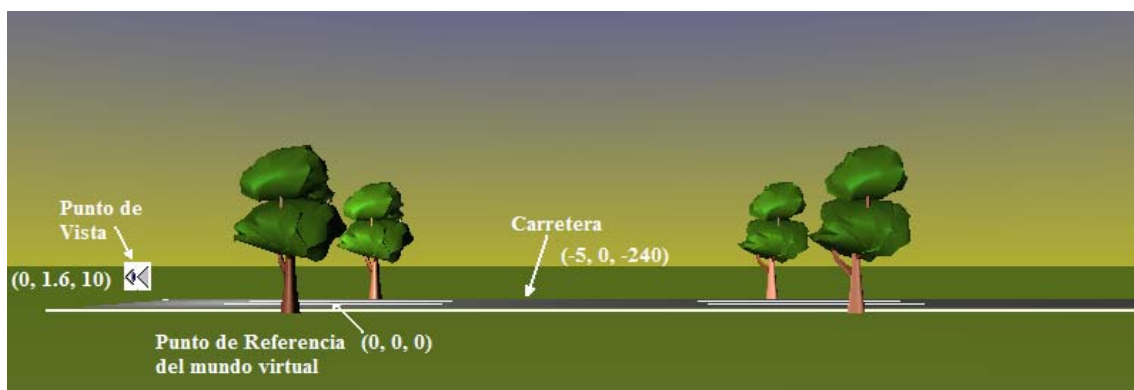


Figura 4.5: Punto de Vista del mundo virtual.

La carretera está orientada en sentido positivo del eje Z. Es decir, hacia a fuera de la pantalla. Por tanto, el avance del mundo virtual será hacia fuera de la pantalla. La figura 4.6 muestra las posiciones de los objetos y la dirección de desplazamiento de los mismos. Además se puede ver como el punto de vista definido muestra la carretera centrada y desde el inicio de la escena.

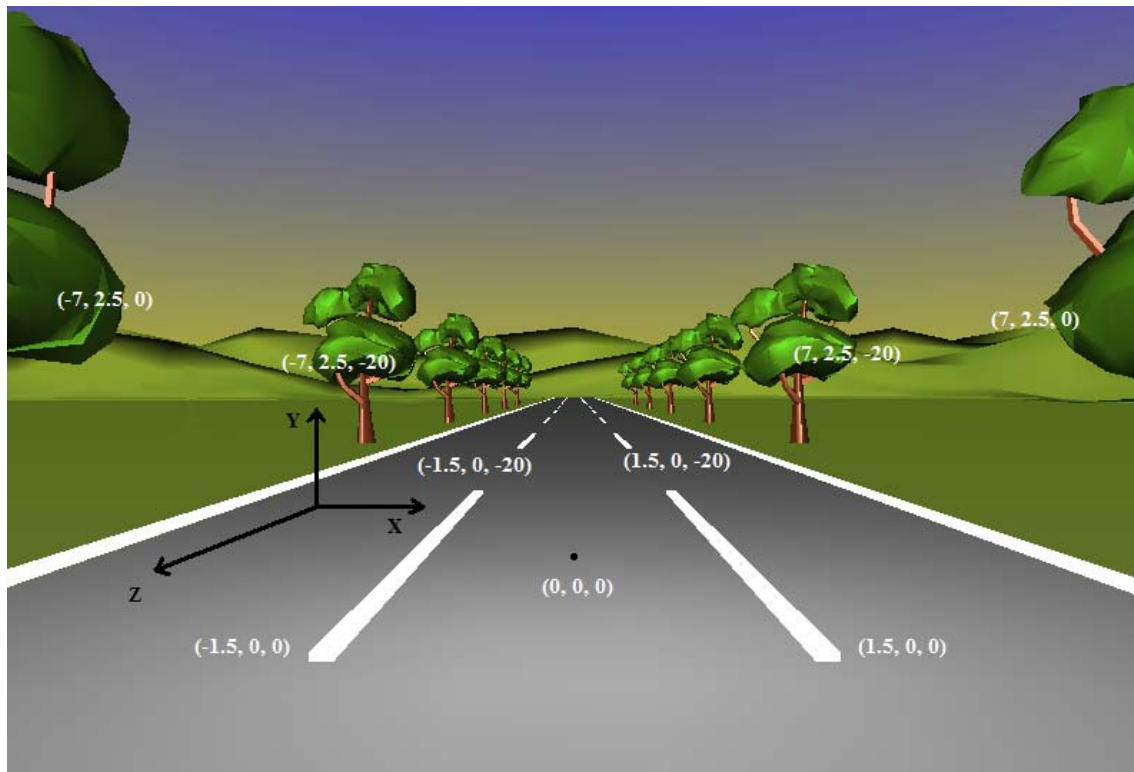


Figura 4.6: Posición de los objetos y sentido de desplazamiento.

Los objetos que se emplean para crear la sensación de movimiento son las líneas discontinuas del suelo y los árboles. El incremento en cada desplazamiento (actualización) es de una unidad en el sentido positivo del eje Z. Cuando los objetos se hayan desplazado 20 unidades volverán a su posición original. Por tanto, lo que se realiza constantemente es un movimiento repetitivo con la intención de emplear el menor número de objetos.

En la figura 4.7 se observa que la distancia entre elementos virtuales consecutivos es de 20 unidades, de aquí viene la razón de emplear este valor para la repetición de una secuencia de movimientos.

Como se comentó en la sección 4.2.2, para crear las líneas discontinuas del suelo y para las hileras de árboles se emplean copias de uso (*USE*). Al utilizar copias de uso, el desplazamiento realizado sobre los elementos originales también se producirá sobre sus copias. Los objetos se desplazarán mediante el campo *translation* que poseen los nodos virtuales que tienen asociado el objeto.

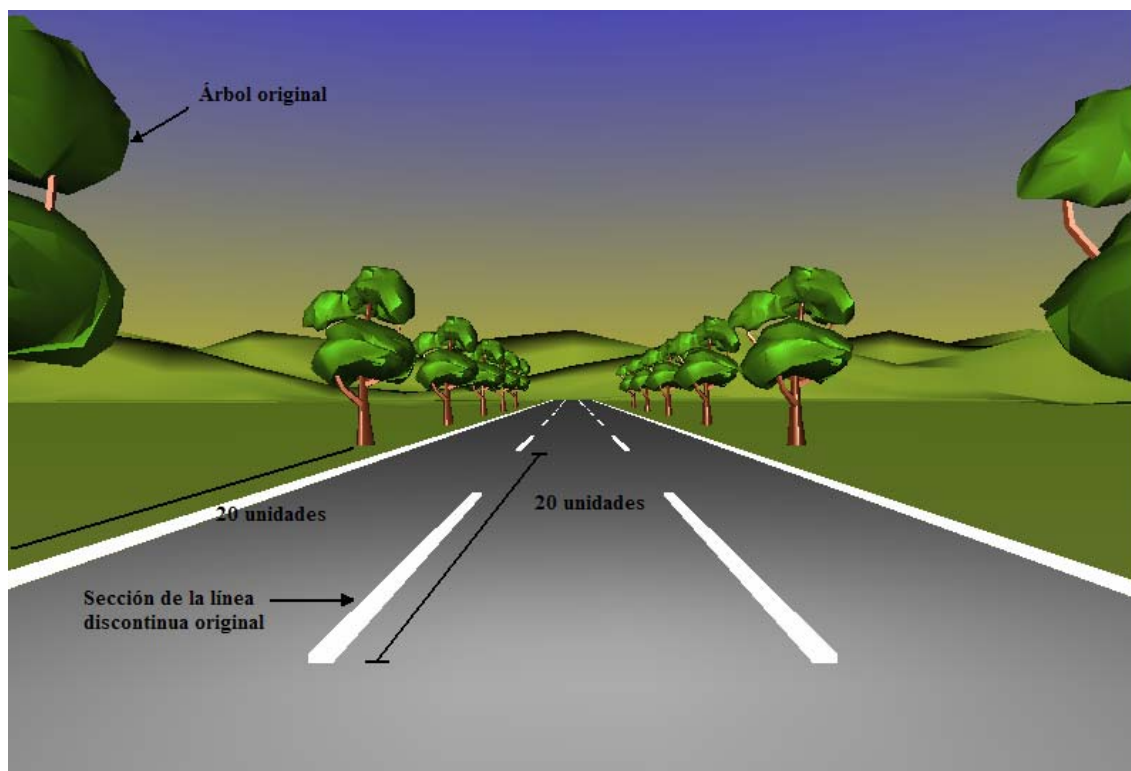


Figura 4.7: Distancia entre elementos.

A continuación, las figuras 4.8, 4.9 y 4.10 mostrarán una secuencia que realiza el desplazamiento de la escena virtual.

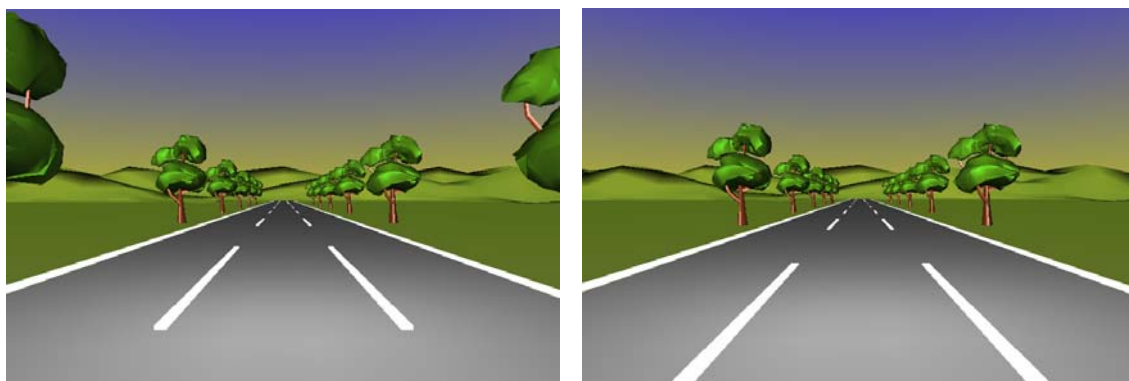


Figura 4.8: Escena virtual al inicio y tras haberse desplazado 4 unidades.

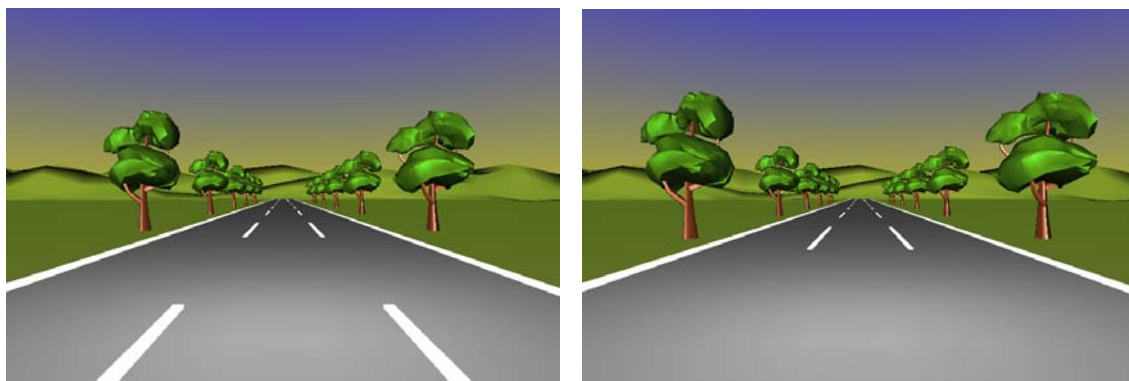


Figura 4.9: Escena virtual tras el desplazamiento de 8 y 12 unidades.

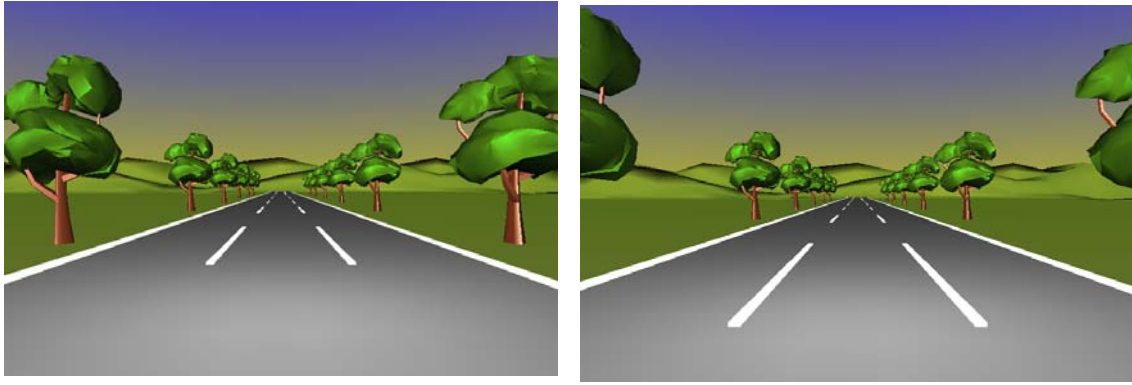


Figura 4.10: Escena virtual tras el desplazamiento de 16 y 20 unidades.

En la secuencia de imágenes puede observarse que tras haberse desplazado 20 unidades la escena virtual, todo queda como al inicio. De aquí, viene la idea de emplear una secuencia repetitiva.

4.2.3.2. Iluminación ambiental de la escena.

La escena se ha iluminado con un determinado tipo de luz:

- **Punto de luz (Point Light):** Define una fuente de luz que emite en todas las direcciones, es decir, es omnidireccional.

El punto de luz permite crear una luz ambiental que iluminará todos los objetos del mundo virtual, considerando que no todas las partes de los objetos son visibles.

El uso de este tipo de luz ha permitido iluminar el mundo virtual con una sola fuente y en consecuencia, la escena supondrá una menor carga de memoria.

4.2.3.3. Sonido.

El sonido es un elemento importante en la escena, porque ayuda en la creación de la sensación de realismo y aislamiento.

El *V-Realm Builder* hace posible la localización del sonido en el espacio, generando un sonido estereofónico. Pero la ventana de visionado del *Virtual Reality Toolbox* no permite la reproducción del sonido implementado en el *V-Realm Builder*.

Debido a este motivo, se ha empleado otra alternativa para presentar sonido en la simulación del mundo virtual. Lo primero que se ha realizado es pasar unos archivos .WAV, que poseen el sonido del motor de un coche y el sonido de rotura de cristales, a ficheros .MAT. Para ello se han empleado los comandos:

```
y=wavread('nombre_fichero.wav');  
save y nombre_fichero.mat
```

siendo y la variable que almacena los datos correspondientes al sonido.

A continuación, se crea un manejador de sonido para cada uno de los elementos que se desean reproducir, cargados previamente de un archivo .MAT. A través de una serie de comandos se ejecuta el sonido, se para o se pausa:

```
load nombre_fichero.mat;  
manejador=audioplayer(y, Fs);  
play(manejador);  
stop(manejador);  
pause(manejador);
```

siendo F_s la frecuencia a la que se reproducen los datos.

El sonido del motor de un coche se asocia al objeto biofeedback (coche), ya que produce la sensación de un vehículo en movimiento. En el caso de fallar la tarea mental cuando se trata de sortear el obstáculo muro, se reproduce el efecto sonoro de cristales rotos, debido a que se colisionará con el muro.

4.3 Descripción de los objetos virtuales diseñados.

Una vez descrita la escena virtual básica, se detallarán los objetos del mundo virtual, que forman parte de la actividad que se pretende desarrollar.

Los objetos virtuales se pueden clasificar en dos tipos: el objeto *biofeedback* (*coche*) y los *obstáculos*. El elemento principal es el objeto biofeedback, ya que el sujeto se identificará con él. Finalmente, los obstáculos indicarán qué tarea mental se debe ejecutar el sujeto en cada momento.

4.3.1. Diseño del objeto biofeedback.

El objeto biofeedback es el elemento sobre el que se plasman las tareas mentales que realiza el sujeto y en consecuencia, proporciona la realimentación necesaria para ejecutar la tarea mental correctamente.

La utilización del objeto biofeedback consigue una mayor motivación y concentración del sujeto, ya que puede ver en tiempo real el resultado de su actividad mental sobre la interfaz desarrollada.

En el proyecto, el objeto biofeedback es representado mediante un coche que se desplaza a la izquierda o a la derecha, en función de las señales EEG del sujeto para el sistema BCI y con los cursores izquierdo y derecho para el sistema de medida de los tiempos de reacción.

4.3.1.1. Implementación del coche.

El coche no ha sido diseñado con el *V-Realm Builder*, sino que se ha tomado de un tutorial MATLAB de Realidad Virtual. El motivo es que el diseño del coche es muy realista con un alto nivel de detalles, como puede observarse en la figura 4.11.



Figura 4.11: Objeto biofeedback.

Para la creación de elementos complejos con muchos detalles se puede emplear el 3D Studio, pudiendo importar el objeto creado a VRML, siendo éste el lenguaje empleado en *V-Realm Builder*. Por tanto, es presumible que el diseño de este coche se halla realizado en 3D Studio e importado a VRML.

4.3.1.2. Efecto implementado en el coche.

El efecto implementado en el coche es un desplazamiento a izquierda y a derecha, que en ningún caso rebasará los límites de la carretera. El movimiento será sobre el eje X en el intervalo de -3 a 3, permitiendo esquivar los obstáculos que aparecen sobre la carretera. Para el sistema BCI, es el clasificador quién determinará si el desplazamiento debe ser hacia la derecha o izquierda, y qué posición concretamente ocupará el coche en el eje X. Para el sistema de medida de los tiempos de reacción, es la pulsación de los cursores quién establecerá la posición del coche en el mundo virtual.

El coche se ha colocado inicialmente en la posición (0, 0.3, 3.5) del mundo virtual para centrarlo en el de carril de en medio y además, para tener una perspectiva completa del vehículo. Así, el sujeto puede observar en todo momento los efectos de su actividad mental sobre el objeto biofeedback desde el punto de vista establecido. La figura 4.12 muestra el mundo virtual con el objeto biofeedback situado en su posición inicial.



Figura 4.12: Objeto biofeedback situado en la escena virtual.

4.3.2. Diseño de los obstáculos

Como se ha comentado en el apartado 4.1, existen tres tipos de obstáculos en el sistema BCI: *charco*, *muro* y *rampa*. Además se comentó que existen dos tipos de realimentación: *positiva* y *negativa*. La realimentación positiva premia visualmente al sujeto por haber realizado correctamente la tarea mental, consiguiendo animarlo para sucesivas pruebas. En cambio, la realimentación negativa castiga visualmente al sujeto para que se esfuerce en realizar las tareas mentales.

- **El charco:** Es un obstáculo que emplea realimentación negativa y que se sitúa en la posición adecuada para indicar al sujeto en qué momento y durante cuánto tiempo el objeto biofeedback puede moverse por la escena virtual, así como cuál es la tarea mental que debe ejecutar. El hecho de ser un intervalo temporal el periodo en el que el coche puede esquivar el charco, se habla de feedback continuo.
- **El muro y la rampa:** Emplean un feedback discreto porque estos obstáculos se esquivarán o abordarán dependiendo de la tarea mental realizada durante un solo instante de tiempo. Por tanto, no hay opción de corregir la tarea mental si se ha realizado de forma incorrecta. Se diferencia en que el muro emplea realimentación negativa y la rampa positiva.

El éxito o el fracaso de la tarea mental realizada en estos tipos de obstáculos es en función de si se ha conseguido eludirlos o no.

Antes de comenzar a describir la implementación de los obstáculos, es importante explicar la manera de elegir los obstáculos que aparecen en el entrenamiento. Esta elección se realizará aleatoriamente, ya que durante la sesión de entrenamiento podrán aparecer distintas configuraciones de obstáculos. Se genera un número aleatorio que se identificará con una configuración de obstáculos. De hecho, existen 3 posibles configuraciones de obstáculos: *charco*, *muro-charco* y *rampa-charco*. También se ha creado otra versión en la que se sustituye el charco por unas marcas en la carretera, que consistirán en unas señales de tráfico. La función tanto del charco como la de las marcas es la misma. Por lo que en esta otra versión las 3 posibles configuraciones de obstáculos son: *marcas*, *muro-marcas* y *rampa-marcas*.

Finalmente, para el sistema de medida de tiempos de reacción se empleará un único tipo de obstáculo de los empleados en la BCI. En este caso, la interfaz gráfica sólo utilizará el obstáculo muro.

A continuación, se detallará tanto la implementación de cada uno de los obstáculos diseñados en las aplicaciones, como los efectos implementados sobre ellos cuando el sujeto consigue esquivarlos o no.

4.3.2.1. Implementación del charco.

El charco implementado en el mundo virtual es básicamente una superficie de agua celeste sobre el asfalto, ocupando uno de los carriles laterales durante el tiempo en el que el sujeto tiene control sobre el objeto biofeedback.

El charco se diseña con un nodo a través del editor *Extrusion*. Su forma viene determinada por *Cross Section*, mostrado en la figura 4.13. *Spine* determina la dimensión del charco en el eje Y y como puede observarse en la figura 4.13, es la menor posible.

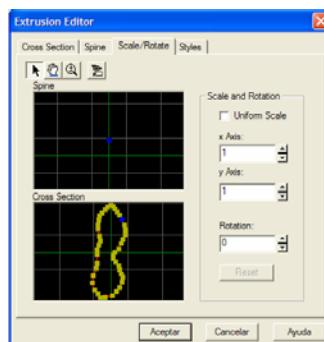


Figura 4.13: Forma del charco en el editor *Extrusion* de *V-Realm Builder*.

La longitud inicial del charco es de 64 unidades, tal y como se muestra en la figura 4.14.

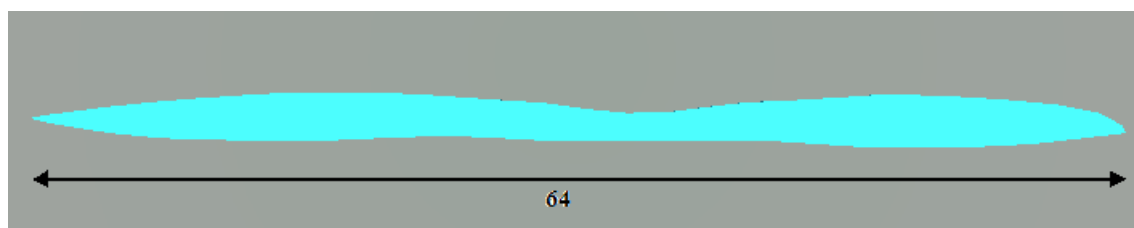


Figura 4.14: Charco implementado.

La longitud del charco durante la simulación vendrá determinada por la duración temporal de la fase en la que el sujeto tiene control sobre el objeto biofeedback (coche). Por ejemplo, si esta fase dura 3.75, el número de unidades que debería tener el charco es 240.

$$\text{Longitud_del_charco} = \text{tiempo_control_del_coche} \times \text{longitud_inicial_charco}$$

Para que el charco adquiriera el número de unidades necesario en cada simulación, hay que escalar el valor inicial establecido en el mundo virtual. Se emplea el campo *scale* del nodo correspondiente.

El charco izquierdo y derecho son iguales, debido a que el charco izquierdo es una copia de uso (*USE*) de la forma del charco derecho. Los charcos se posicionarán en un punto del eje Z del mundo virtual y se desplazarán en el sentido positivo del eje Z, gracias al campo *translation* que tienen asociados. La figura 4.15 muestra el mundo virtual con el obstáculo charco.



Figura 4.15: Mundo virtual con el obstáculo charco.

Existe una segunda versión de la aplicación del sistema BCI que emplea otra forma de indicar el intervalo de control del objeto biofeedback por parte del sujeto. En lugar de utilizar un charco como en el caso anterior, se usan unas señales de tráfico que también se ubicarán en alguno de los carriles laterales, indicando que el vehículo debe dirigirse hacia el carril contrario. La figura 4.16 muestra los elementos empleados que han sido obtenidos de la librería de objetos (*Object Library*).

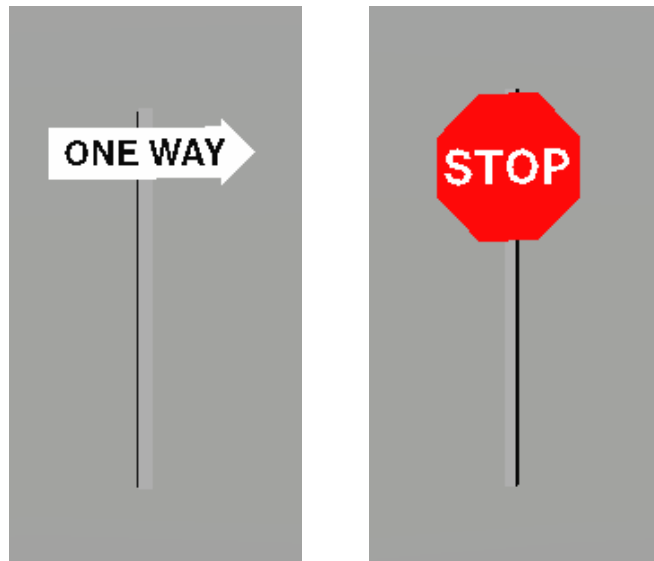


Figura 4.16: Marcas que establecen el intervalo de control sobre el objeto biofeedback.

La marca representada por la señal de tráfico *ONE WAY* establece el inicio del intervalo de control sobre el objeto biofeedback. En cambio, la marca representada por la señal *STOP* indica que este intervalo va a finalizar. La figura 4.17 muestra la ubicación de ambas marcas en el mundo virtual implementado. Se posicionarán fuera de la carretera, próximas al carril izquierdo o derecho.

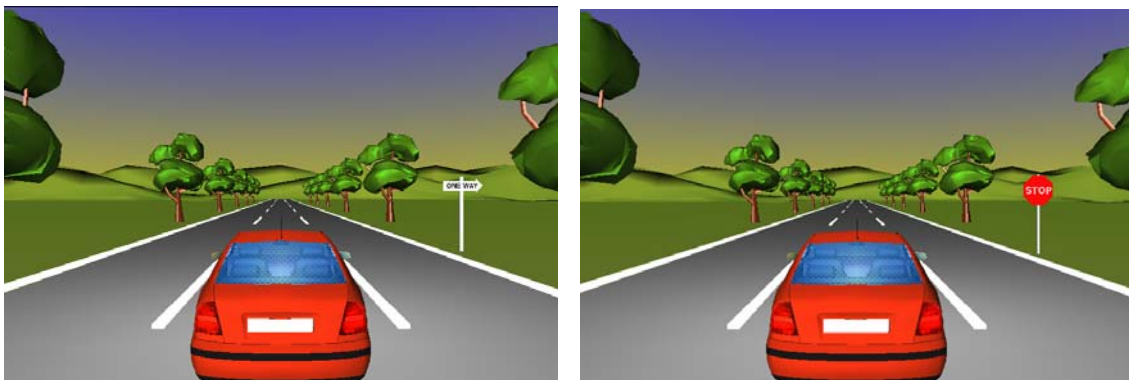


Figura 4.17: Mundo virtual con las marcas que indican el intervalo de control biofeedback.

La figura 4.18 muestra el mundo virtual con las dos marcas encargadas de indicar al usuario la duración del intervalo de control biofeedback. Así como, el instante en el que comienza y finaliza este periodo temporal.

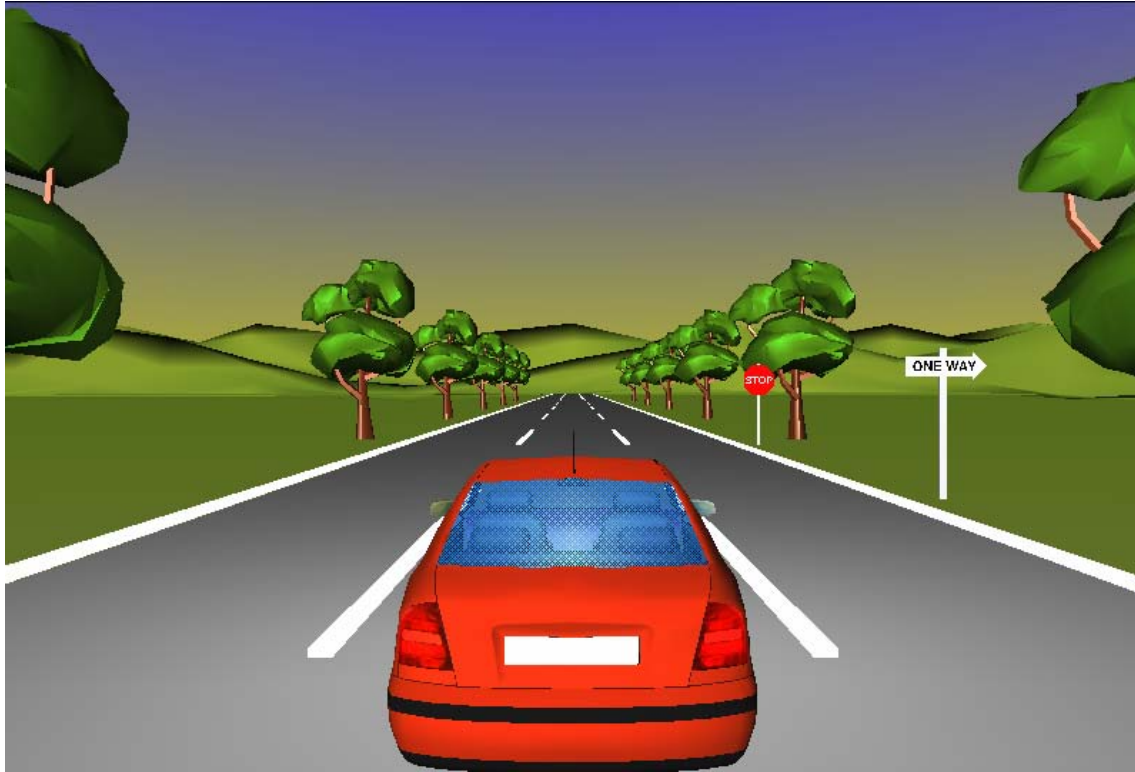


Figura 4.18: Mundo virtual en un instante en el que ambas marcas son visibles.

4.3.2.2. Implementación del muro.

El segundo obstáculo del sistema BCI es un muro creado con *V-Realm Builder*. Se trata de una pared formada por una serie de ladrillos de color rojo. Además ocupará uno de los carriles laterales junto al charco, pero se colocará al final de él.

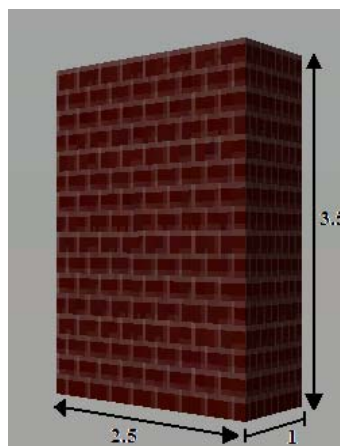


Figura 4.19: Muro implementado.

El muro se ha implementado con un nodo *Box* de dimensiones (2.5, 3.5, 1) en XYZ. El cubo, que da forma al muro, hay que desplazarlo 1.75 unidades en el eje Y, porque el punto de referencia del cubo es su centro. Si no se desplaza, la mitad del objeto creado queda por debajo de la carretera. Para dar textura se ha empleado la librería (*Texture Library*), eligiendo *Brick_Small*. La figura 4.19 muestra la imagen del muro creado.

El muro izquierdo y derecho son iguales, debido a que el muro izquierdo es una copia de uso (*USE*) de la forma del muro derecho. Durante la simulación se posicionarán en un punto del eje Z, que corresponderá al final del charco y en consecuencia, al final del control del objeto biofeedback por parte del sujeto. Además, se desplazarán en el sentido positivo del eje Z como el resto de los elementos del mundo virtual y empleando el campo *translation*.

Finalmente, el muro derecho hay que desplazarlo 3 unidades en el eje X y el muro izquierdo -3 unidades, de modo que no queden en el carril central. La figura 4.20 muestra el mundo virtual con este tipo de obstáculo.



Figura 4.20: Mundo virtual con el obstáculo muro.

Si la tarea mental no se realiza correctamente cuando el obstáculo muro está a la altura del coche, este colisionará (feedback negativo). Para esta situación se ha implementado la acción de un choque, provocándose los siguientes efectos:

- Primero, en el coche se romperán los cristales, como se observa en la figura 4.21. En el instante en el que entren en contacto el coche y el muro, se conmutará el coche empleado hasta ese momento por otro con los cristales rotos.



Figura 4.21: Objeto biofeedback después de la colisión.

- A continuación, se recreará el efecto de una colisión. Si el coche entra en contacto con el muro, se elevará un poco (5°) la parte trasera durante 0.375 segundos, y tras haber transcurrido este tiempo volverá a su posición original. La acción estará acompañada de un sonido, que simulará a unos cristales rotos. Para elevar la parte trasera del coche hay que emplear el campo *rotation* del coche colisionado. La figura 4.22 muestra el efecto de la colisión.

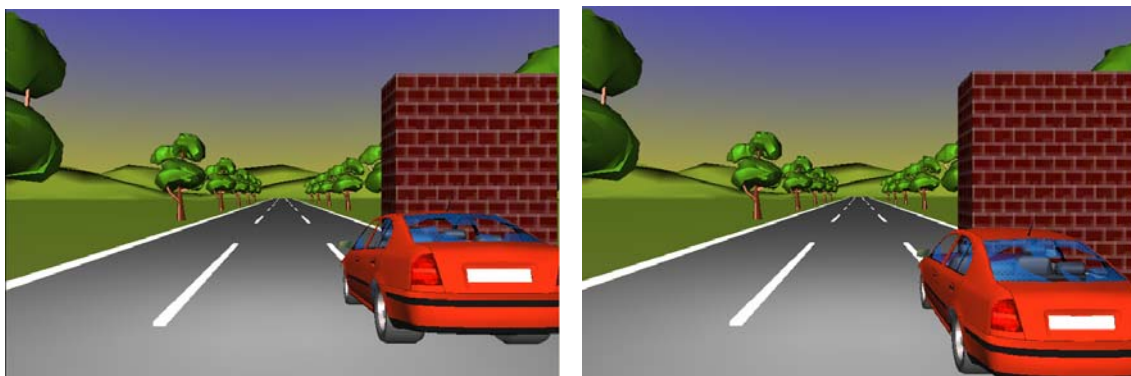


Figura 4.22: Efecto de la colisión implementado.

La figura 4.23 muestra el instante de la colisión en la que la parte trasera del coche se eleva 5°.

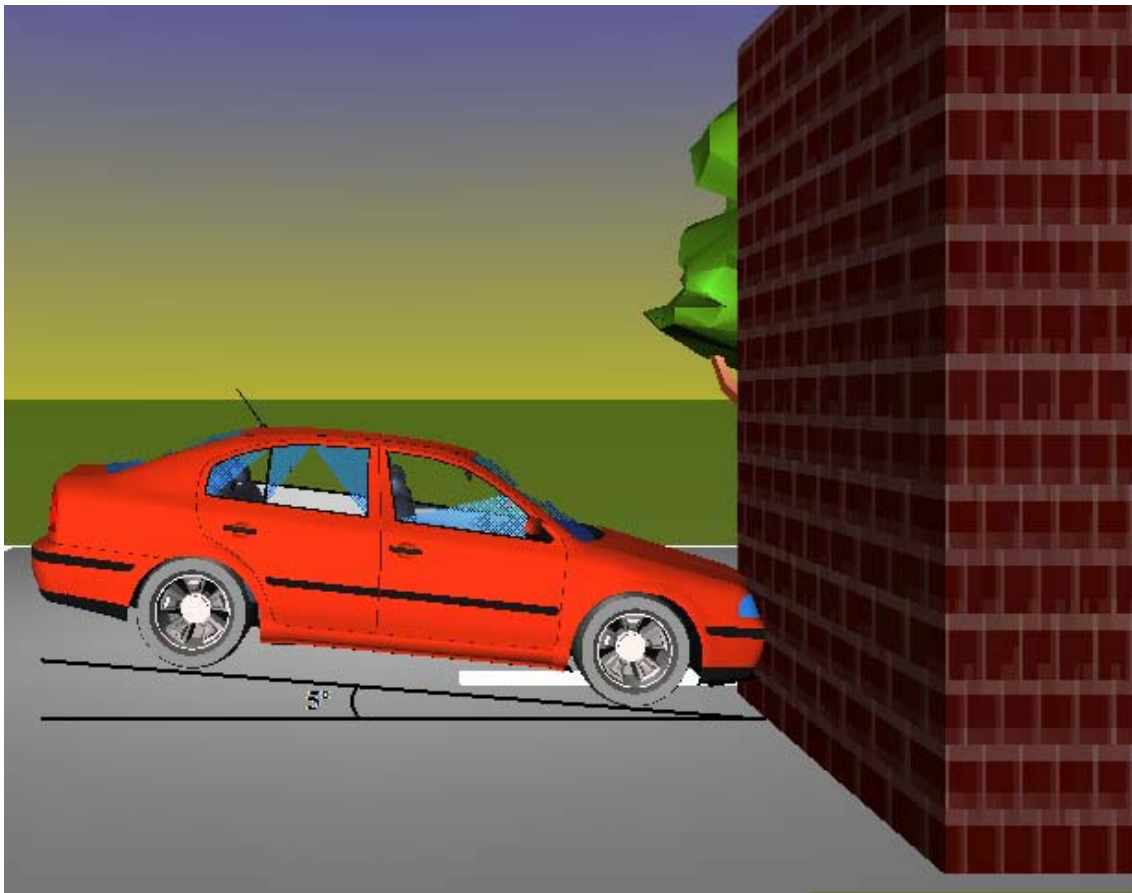


Figura 4.23: Vista lateral de la colisión.

La sensación producida es la de una colisión entre un vehículo y un obstáculo, resultando el vehículo visible y audiblemente perjudicado. Este efecto provoca en el sujeto una impresión negativa, obligándolo a esforzarse más en sucesivas situaciones.

Al producirse la colisión y sus efectos asociados, también se frenarán las actualizaciones (modificaciones que producen la sensación de movimiento) del mundo virtual hasta que finalice la prueba.

El objeto biofeedback colisionará cuando:

- El muro aparece en el carril derecho y el coche se encuentra en una posición dentro del intervalo $(1\ 3]$ del eje X. (Figura 4.24).

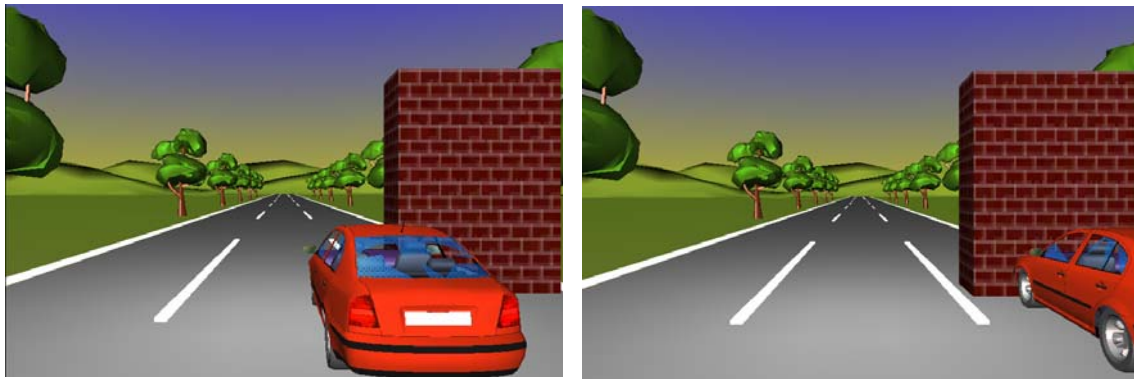


Figura 4.24: Intervalo del eje X en el que el objeto biofeedback colisiona con el muro derecho.

- El muro aparece en el carril izquierdo y el coche se encuentra en una posición dentro del intervalo $(-1 -3]$ del eje X. (Figura 4.25).

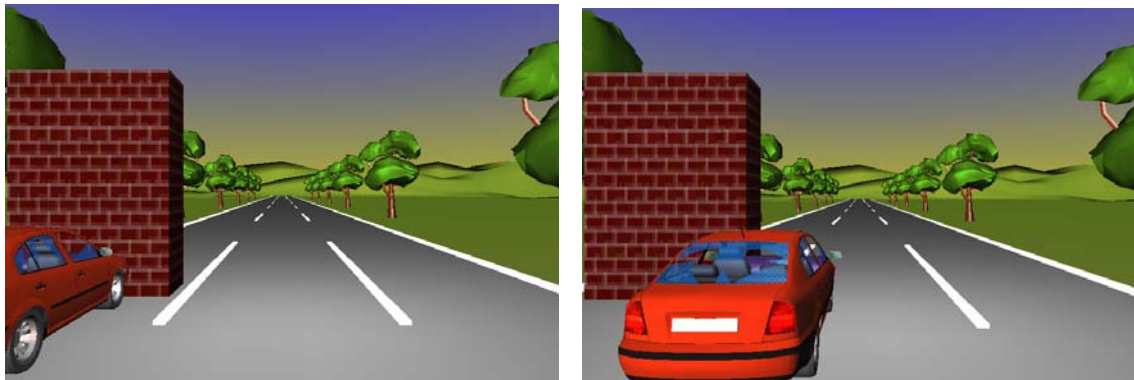


Figura 4.25: Intervalo del eje X en el que el objeto biofeedback colisiona con el muro izquierdo.

La interfaz implementada, en la segunda aplicación, es una particularización de la interfaz gráfica desarrollada para el sistema BCI. Se trata de un mundo virtual que sólo posee el obstáculo muro. Por tanto, lo que se ha hecho es eliminar los obstáculos charco y rampa de la escena virtual, y modificar las dimensiones del muro para que ocupe dos carriles.

Se ha modificado el tamaño del muro en el eje X, duplicándolo para que ocupe los dos carriles exigidos. Las dimensiones del muro son (5, 3.5, 1) como muestra la figura 4.26. El obstáculo utilizado en la interfaz aparecerá a izquierda o derecha de manera aleatoria.

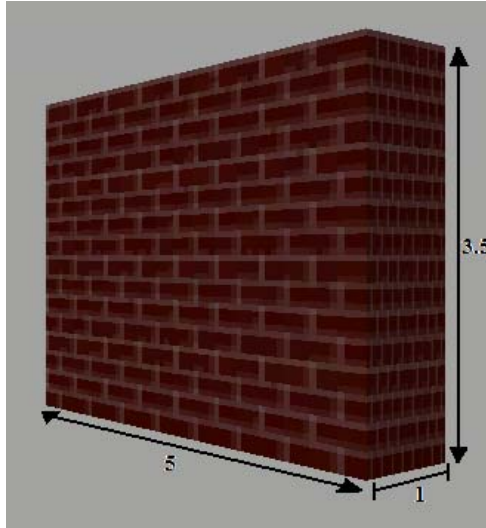


Figura 4.26: Dimensiones del muro para la segunda aplicación.

Además el coche podrá ubicarse en tres posiciones distintas del eje X, que corresponden a los tres carriles de la interfaz gráfica.

- Si la posición del coche en el eje X es 0, ocupará el carril central como muestra la figura 4.27.



Figura 4.27: Coche ocupando el carril central.

- Si la posición del coche en el eje X es 2.5, ocupará el carril derecho como muestra la figura 4.28.



Figura 4.28: Coche ocupando el carril derecho.

-
- Si la posición del coche en el eje X es -2.5, ocupará el carril izquierdo como muestra la figura 4.29.



Figura 4.29: Coche ocupando el carril izquierdo.

El resultado de la interfaz implementada para el sistema de medida de los tiempos de reacción se muestra en la figura 4.30.

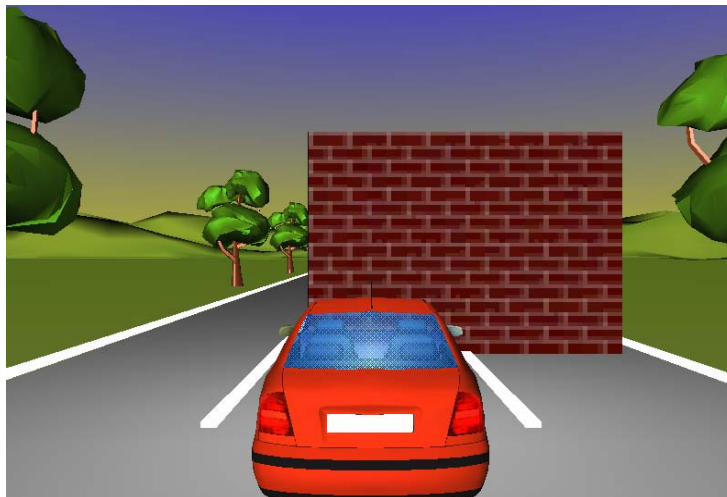


Figura 4.30: Mundo virtual implementado.

Por último, el efecto de colisión implementado en esta aplicación es más sencillo, ya que solamente se romperán los cristales del coche y se reproducirá el sonido que tiene asociado.

4.3.2.3. Implementación de la rampa.

El último obstáculo diseñado en el sistema BCI es una rampa que ocupa alguno de los carriles laterales. Se trata de un obstáculo formado por una serie de ladrillos de color gris. La rampa será abordada por el objeto biofeedback al final del charco.

Dicha rampa está formada por dos elementos: pendiente y rasante. La pendiente es la parte que posee un cierto grado de inclinación. La rasante es la zona paralela a la carretera.

La pendiente se ha implementado con un nodo *Box* de dimensiones (3, 1.4, 8) en XYZ. Para que el cubo que da forma a la pendiente adquiera inclinación, hay que rotarlo 10° en el eje X con el campo *rotation*. Parte del cubo queda por debajo de la carretera creándose una imagen similar a una pendiente. La textura elegida de la librería (*Texture Library*) es *Brick_Grey*.

La rasante se ha implementado con un nodo *Box* de dimensiones (3, 1.4, 20). La textura elegida es *Brick_Grey*, la misma que en la pendiente. Para que la rasante esté a continuación de la pendiente, hay que desplazarla 14 unidades en el sentido negativo del eje Z, y 0.69 unidades en el eje Y. La figura 4.31 muestra las dimensiones de la rampa y los puntos de referencia de los objetos.

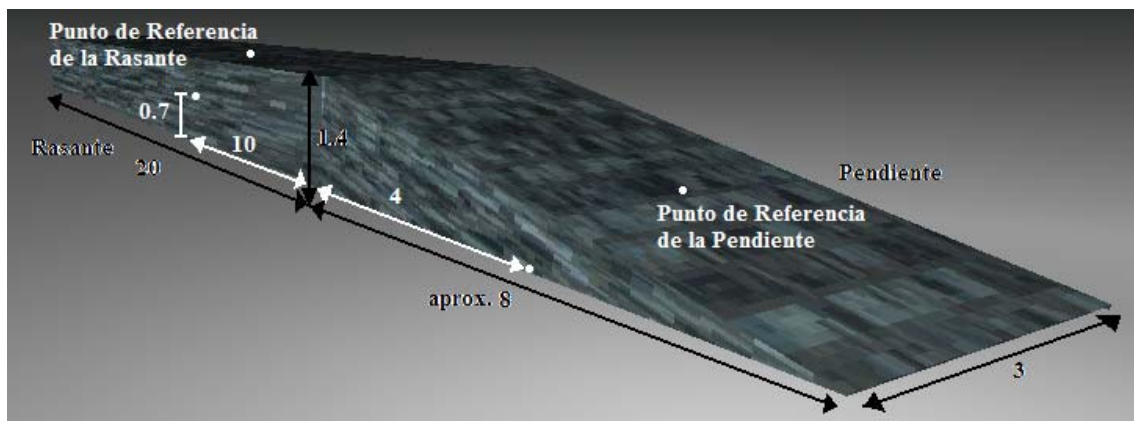


Figura 4.31: Rampa implementada.

Como se observa en la figura anterior, el punto de referencia de los elementos es el centro, por esto el desplazamiento realizado sobre la rasante es la mitad de las dimensiones de cada elemento. Además, al ser la inclinación de la pendiente 10° se eleva 0.69 unidades, $4 \cdot \text{sen}(10^\circ) = 0.69$, que coincide con el desplazamiento en el eje Y de la rasante.

La rampa izquierda y derecha son iguales, debido a que la rampa izquierda emplea copias de uso (*USE*) de la pendiente y la rasante. Durante la simulación se posicionará

en un punto del eje Z, correspondiente al final del charco. Se desplazará en el sentido positivo del eje Z como los demás elementos del mundo virtual.

Finalmente, la rampa derecha hay que desplazarla 3 unidades en el eje X y la rampa izquierda -3 unidades, de manera que no queden en el carril central. La figura 4.32 muestra el mundo virtual con el obstáculo muro.



Figura 4.32: Mundo virtual con el obstáculo rampa.

Si la tarea mental se realiza correctamente, el coche deberá saltar por la rampa (feedback positivo). Para realizar el efecto que reproduce el salto de la rampa son necesarios 0.5 segundos. Todo este proceso requiere de 32 actualizaciones del mundo virtual y se pueden englobar en tres fases:

1. **Ascensión de la pendiente:** Abarca desde que el coche se encuentra justo delante de la rampa hasta que el coche alcanza con el morro la parte más alta. Son necesarias 9 actualizaciones. La figura 4.33 muestra imágenes del inicio y del final de esta acción.



Figura 4.33: Ascensión de la pendiente. (Actualizaciones 1 y 9).

2. **Estabilización sobre la rasante:** Se encarga de estabilizar el coche sobre la rasante, y de avanzar sobre ella, hasta que las ruedas delanteras lleguen al final. Para realizar esta acción son necesarias 20 actualizaciones. La figura 4.34 muestra el inicio y el final de la acción.



Figura 4.34: Estabilización sobre la rasante. (Actualizaciones 10 y 29).

3. **Caída al asfalto:** Se encarga de la caída del coche al asfalto. Son necesarias 3 actualizaciones. La figura 4.35 muestra el inicio y final de la acción.

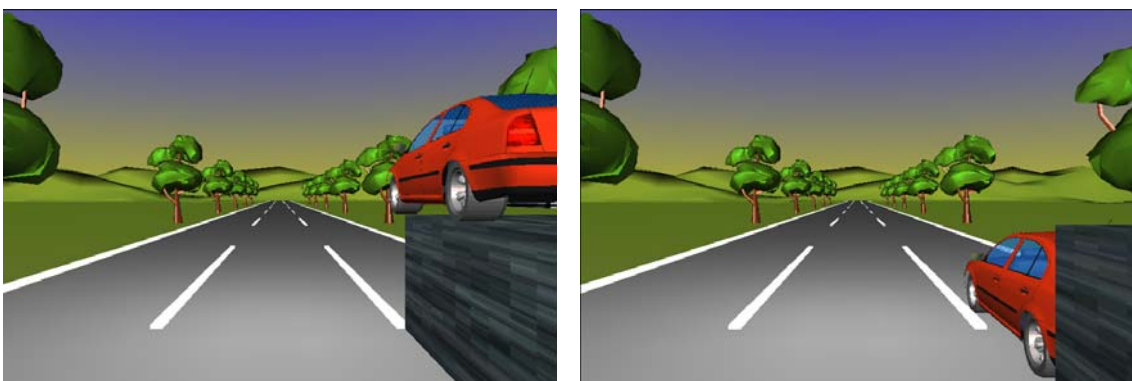


Figura 4.35: Caída al asfalto (Actualizaciones 30 y 32).

La mecánica del salto se ha implementado calculando el ángulo formado entre el objeto biofeedback (coche) y el suelo. El esquema de la figura 4.36 muestra el cálculo del ángulo en la subida, aplicando trigonometría.

$$\text{Ángulo} = \arcsen\left(\frac{\text{Altura_Rampa}}{\text{Longitud_Pendiente}}\right) = 10^\circ$$

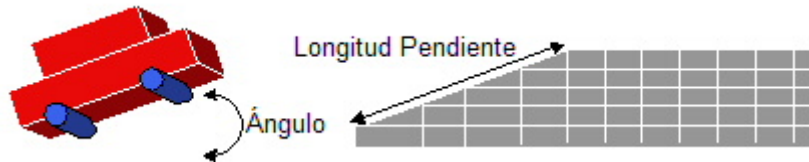


Figura 4.36: Cálculo del ángulo para el ascenso de la pendiente.

El vehículo aumentará el ángulo que forma con el suelo hasta el valor de 10° , instante en el que se posará totalmente sobre la rampa. A partir de aquí, el ángulo no variará hasta que el vehículo llegue a la rasante, lugar donde irá disminuyendo el ángulo hasta hacerse cero. Para conseguir la sensación de ascenso por la rampa, además de variar el ángulo formado por el coche, es necesario modificar su posición vertical (eje Y). El coche caerá al asfalto cuando el ángulo y el desplazamiento en el eje Y sean cero. La figura 4.37 explica el efecto creado.

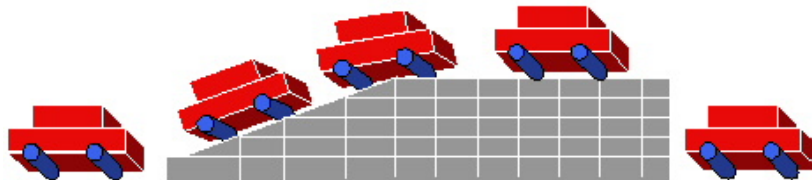


Figura 4.37: Proceso del efecto implementado sobre la rampa.

El objeto biofeedback saltará la rampa cuando:

- La rampa aparece en el carril derecho y el coche se encuentra en una posición dentro del intervalo $(0.8 \ 3]$ del eje X. (Figura 4.38).

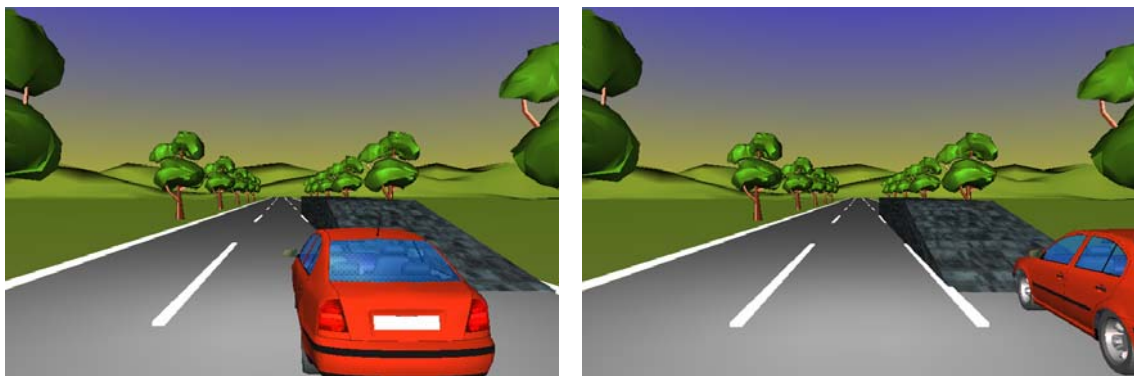


Figura 4.38: Intervalo del eje X en el que el objeto biofeedback saltará la rampa derecha.

-
- La rampa aparece en el carril izquierdo y el coche se encuentra en una posición dentro del intervalo $(-0.8 -3]$ del eje X. (Figura 4.39).



Figura 4.39: Intervalo del eje X en el que el objeto biofeedback saltará la rampa izquierda.

4.3.2.4. Configuraciones de los obstáculos.

Los obstáculos diseñados, como se ha comentado en este apartado, pueden combinarse de la siguiente manera en el sistema BCI:

- **Charco:** El charco puede aparecer sólo, tal y como muestra la figura 4.40. Para ello, en el panel de control únicamente se debe seleccionar este tipo de obstáculo.

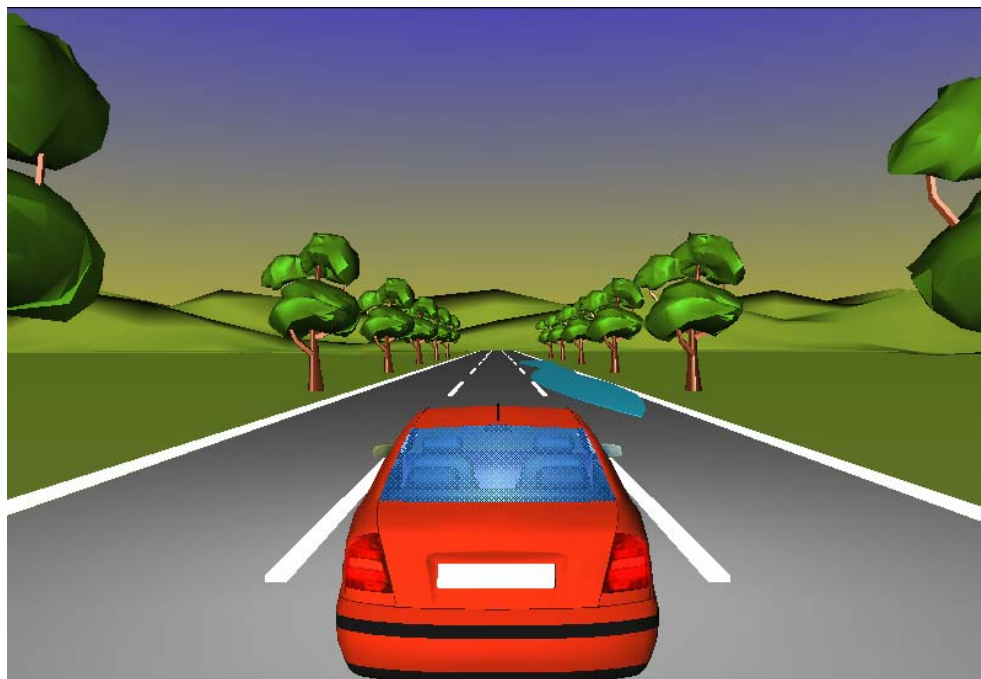


Figura 4.40: Mundo virtual con el obstáculo charco.

-
- **Muro y charco:** El muro aparecerá al final del charco y en el mismo carril. La figura 4.41 muestra esta configuración de los obstáculos en el mundo virtual.



Figura 4.41: Mundo virtual con los obstáculos muro y charco.

- **Rampa y charco:** La rampa aparecerá al final del charco y en el carril contrario. La figura 4.42 muestra esta configuración de los obstáculos en el mundo virtual.

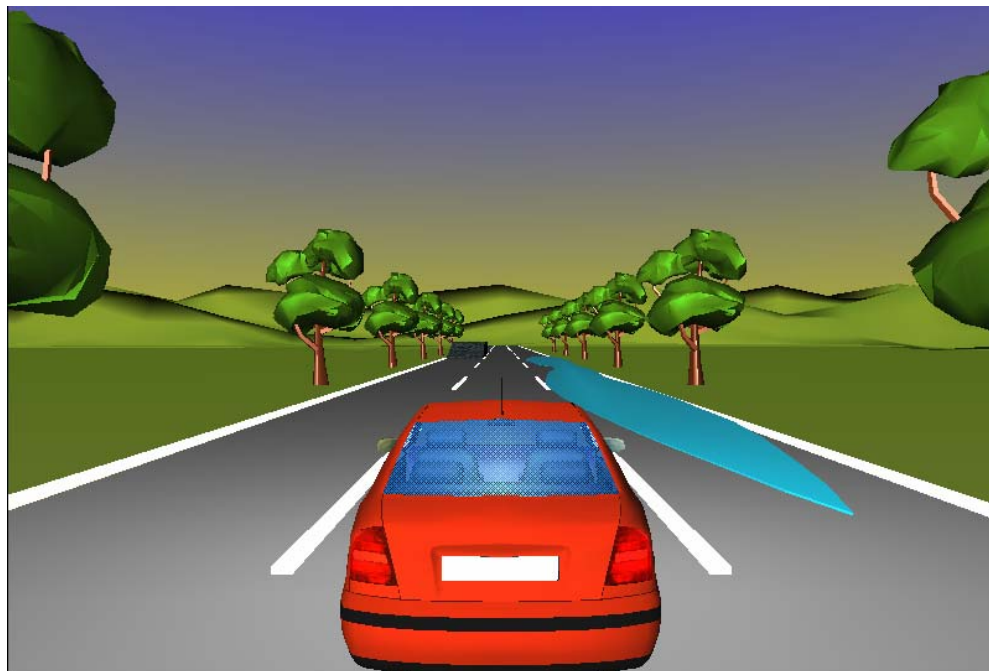


Figura 4.42: Mundo virtual con los obstáculos rampa y charco.

Capítulo 5.

Integración de la interfaz gráfica desarrollada con el resto del sistema BCI.

En este capítulo, se pretende explicar el proceso de integración de la interfaz gráfica en los bloques dedicados a la adquisición y procesado de las señales. Con la aplicación desarrollada se persigue la creación de un sistema de entrenamiento que permita mejorar la comunicación cerebro-computador, a través de un mayor control de las señales mentales.

Finalmente, también se describirá la integración de la interfaz en la segunda aplicación implementada en el proyecto, el sistema de medida de los tiempos de reacción. Se encarga de estudiar el tiempo de reacción de un sujeto ante un estímulo concreto.

5.1. Antecedentes de la interfaz.

El sistema BCI implementado en este proyecto parte de otro desarrollado por el grupo DIANA del Departamento de Tecnología Electrónica de la Universidad de Málaga. La interfaz que poseía inicialmente el sistema BCI intentaba conseguir que un sujeto, a través del control de sus señales cerebrales, extendiera una barra horizontal en la pantalla, a izquierda o derecha según las indicaciones de una flecha que aparecía

previamente. El movimiento de la barra en la pantalla del ordenador informaba al sujeto del estado de su actividad mental, constituyendo la realimentación o biofeedback.

Tras la realización de una serie de pruebas por parte del grupo DIANA, se llegó a la conclusión de que la sencillez de la interfaz podía resultar contraproducente. La simplicidad del paradigma de entrenamiento puede producir, al cabo de ciertas sesiones, sensación de monotonía en el sujeto. La disminución de la motivación y concentración genera un peor control mental.

Para captar la atención del sujeto durante la totalidad de las sesiones de entrenamiento, se optó por el uso de técnicas basadas en Realidad Virtual. Es una tecnología que permite provocar una elevada sensación de inmersión y realismo en el usuario.

El sistema BCI inicial propone un paradigma sencillo en estructura. Basándose en él se ha desarrollado la interfaz en este proyecto.

5.1.1. Descripción del paradigma de entrenamiento de referencia.

El paradigma de entrenamiento del sistema BCI de referencia será más fácil de entender a partir de los elementos que lo componen. Los elementos básicos que se distinguen en el entrenamiento son:

- Una **cruz roja**, en el centro de la pantalla del ordenador, indicando al sujeto el estado de relajación y espera.
- Una **flecha** que apuntará a izquierda o derecha, indicando la tarea mental que se debe realizar.
- Un **objeto biofeedback** presentado con la forma de una **barra horizontal**. Dicha barra proporciona al sujeto información del estado de la tarea mental que se está realizando. La barra horizontal se extiende en mayor o menor medida a izquierda o derecha, dependiendo de la tarea mental realizada por el sujeto. Si la flecha ha aparecido a la derecha y la barra se alarga hacia este lado, el sujeto estará haciendo correctamente la acción mental. La realimentación o feedback empleado en la aplicación es **continuo**, ya que muestra de forma ininterrumpida

el resultado de la tarea mental, al desplazar la barra horizontal a derecha o izquierda. Por tanto, el sujeto puede corregir su actividad mental de manera instantánea.

El entrenamiento del sistema BCI se compone de una serie de fases que se repiten durante todo el tiempo que dura la simulación. De hecho, estas fases de entrenamiento poseen una temporización muy bien estructurada y resulta un factor determinante en la realización del experimento. Se compone de 4 fases que serán consecutivas y se repetirán durante todo el entrenamiento: *inicio y presentación, establecimiento de la tarea a ejecutar, control sobre el objeto biofeedback y descanso*.

- **Inicio y presentación:** Esta primera fase es la que establece el inicio de la serie de fases que se repiten durante todo el entrenamiento. Es aquí donde aparece la cruz roja indicando al sujeto que se relaje y espere.
- **Establecimiento de la tarea a ejecutar:** Es la segunda fase, en ella se pretende indicar al sujeto a través de una flecha la tarea mental que debe realizar.
- **Control del objeto biofeedback:** Se trata de la fase más importante del sistema, debido a que en ella se controla el objeto biofeedback (barra horizontal) a través del análisis y la clasificación de las señales EEG del sujeto bajo prueba. A esta fase también se le denomina **fase de análisis**.
- **Descanso:** Establece un tiempo de descanso antes de pasar nuevamente a la fase de inicio y presentación.

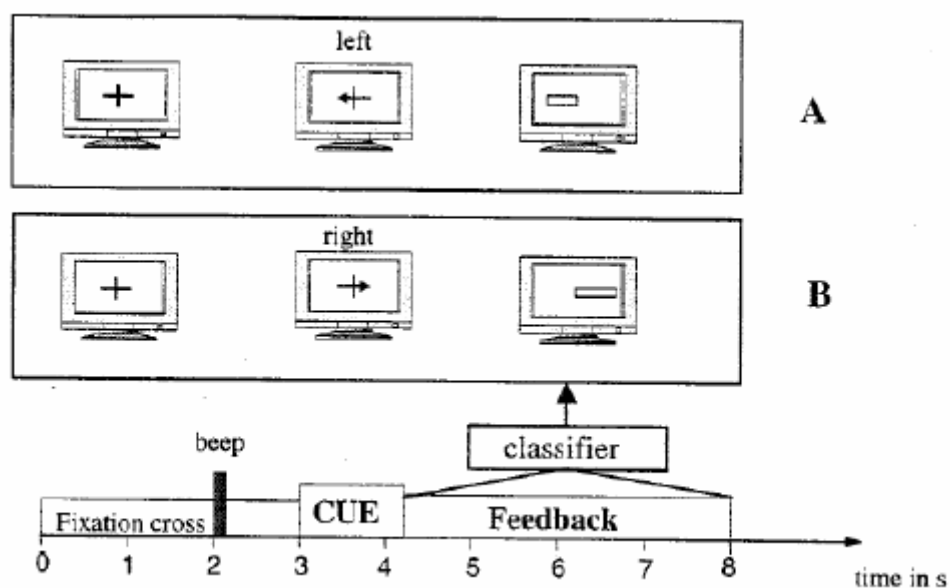


Figura 5.1: Temporización del sistema BCI de referencia.

En la figura 5.1 se muestra un ejemplo de las diferentes fases del entrenamiento de un sistema BCI. La duración de todas las fases en el ejemplo es de 8 segundos. Sin embargo, el eje de tiempos de la figura no tiene en cuenta la fase de descanso.

Fases de entrenamiento del ejemplo:

1. En la fase de inicio aparecerá una cruz roja en el centro de la pantalla.
2. A los 2 segundos sonará un *beep*.
3. En el instante 3 segundos y durante 1.25 segundos (fase de establecimiento de la tarea a ejecutar), una flecha de color verde apuntará a derecha o a izquierda, indicando la tarea mental a realizar.
4. En el instante 4.25 segundos y durante 3.75 segundos (fase de análisis), desaparecerá la flecha y se mostrará una barra en el centro de la pantalla, proporcionando el biofeedback al sujeto.
5. Finalmente, desaparecerá la barra y se iniciará la fase de descanso.

Todo este proceso explicado se repite durante todo el entrenamiento que se realiza con el sistema BCI que se desarrolló por el grupo DIANA del Departamento de Tecnología Electrónica.

5.1.2. Comparativa de la interfaz inicial y la interfaz desarrollada.

Los elementos explicados en el sistema BCI de referencia están relacionados directamente con los desarrollados en el capítulo 4, utilizados en la interfaz gráfica basada en técnicas de Realidad Virtual.

- La carretera implementada se asocia a la interfaz en general, y aparecerá durante toda la simulación. Se asociará a la cruz roja únicamente en la primera fase que indica al sujeto el estado de relajación y espera.
- Los obstáculos implementados se asocian a la flecha, ya que ambos determinan qué tarea mental debe realizarse.
- El coche es el objeto biofeedback que sustituye a la barra horizontal mostrada durante la fase de análisis.

Al igual que en el sistema de referencia, la BCI desarrollada en el proyecto también se compone de 4 fases que serán: *presentación del mundo virtual, establecimiento de la tarea a ejecutar, control sobre el objeto biofeedback y descanso*.

- **Presentación del mundo virtual:** Esta primera fase es la que establece el inicio de la serie de fases que se repiten durante todo el entrenamiento. Es aquí donde aparece la interfaz gráfica desarrollada mediante técnicas de Realidad Virtual, consistente en una carretera. Además, conseguirá que el sujeto evite distracciones y se sienta inmerso en el mundo virtual.
- **Establecimiento de la tarea a ejecutar:** Es la segunda fase y en ella se pretende indicar al sujeto la tarea mental que debe realizar a través de una serie de obstáculos explicados en el capítulo 4.
- **Control del objeto biofeedback:** Se trata de la fase más importante del sistema, debido a que en ella se controla el objeto biofeedback (coche) a través del análisis y la clasificación de las señales EEG del sujeto bajo prueba. A esta fase también se le denomina **fase de análisis**.
- **Descanso:** Establece un tiempo de descanso antes de pasar nuevamente a la fase de inicio y presentación.

Las fases explicadas son las mismas que en el sistema BCI de referencia, diferenciándose solamente en que se han adaptado al mundo virtual implementado para la interfaz gráfica del sistema BCI creado en este proyecto. A continuación, se describirá un ejemplo de las fases de entrenamiento de esta nueva interfaz.

Ejemplo de las fases de entrenamiento con la interfaz desarrollada:

1. En la fase de presentación, el mundo virtual mostrado consistirá en una carretera con el coche (objeto biofeedback) circulando en el centro. Además, se reproducirá el sonido de un motor durante toda la prueba.
2. Al transcurrir un tiempo de 2 segundos se mostrarán los obstáculos (charco, muro y rampa), que indicarán la tarea mental a realizar.
3. La fase de análisis comenzará, en principio, a los 4.25 segundos y durará 3.75 segundos. Se trata de un intervalo temporal en el que el sujeto tiene control sobre el coche a través de un feedback, para intentar sortear los obstáculos desde que el coche se encuentra a la altura del charco hasta que

finaliza. La longitud del charco coincide con la duración de la fase de análisis.

4. La simulación concluirá 1 segundo después de terminar la fase de análisis.
5. Finalmente, el mundo virtual volverá a su posición inicial y se detendrá durante la fase de descanso.

Como se puede observar, lo que se ha realizado es una adaptación de una interfaz desarrollada a un sistema BCI existente.

5.2. Diagrama básico del sistema BCI generado.

El diagrama del sistema BCI que se ha elaborado se puede representar con el esquema de la figura 5.2.

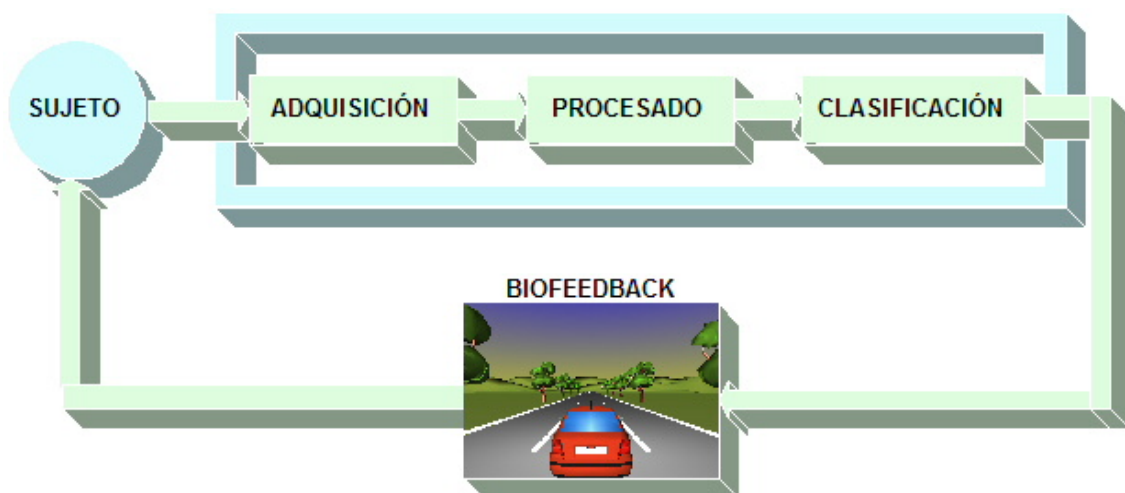


Figura 5.2: Diagrama básico de la BCI creada.

Como se ha podido ver en la figura 5.2, el sistema BCI consta de las siguientes fases: *adquisición*, *procesado*, *clasificación* y *realimentación*.

- **Adquisición:** Obtiene las señales electroencefalográficas del sujeto bajo prueba, mediante una serie de electrodos colocados en el cuero cabelludo. Las señales captadas pasan a un polígrafo, dispositivo encargado de amplificarlas. Los valores de salida del polígrafo son introducidos en una tarjeta de adquisición, dedicada a la conversión analógica-digital para que puedan ser procesadas por el ordenador.

-
- **Procesado de señal:** Extrae los parámetros de interés de las señales adquiridas, que se emplearán posteriormente en la modificación de la interfaz gráfica.
 - **Clasificación:** Traduce los parámetros obtenidos durante el procesado en modificaciones de la interfaz gráfica. El encargado de realizar las modificaciones, en tiempo real, sobre la interfaz es el *trasladador*. En este caso, el trasladador está formado por la interfaz gráfica y los comandos que la gestionan.
 - **Realimentación o biofeedback:** Informa al sujeto de cómo su actividad mental está afectando a lo que se le presenta a través de la interfaz gráfica. Se emplea feedback continuo, permitiendo al sujeto ver durante toda la prueba cómo está ejecutando la tarea mental.

El último bloque es el que se ha desarrollado en este proyecto, así como la integración con el resto de ellos. El resto, es decir, adquisición, procesado y clasificación, fueron realizados por otro proyecto fin de carrera dentro del Departamento de Tecnología Electrónica.

5.2.1. Adquisición de los datos.

Los datos son capturados mediante una tarjeta de adquisición configurada a través del *DAQ Toolbox* de MATLAB. Se encarga de transferir las muestras de las señales adquiridas.

La captura de las señales del sujeto es realizada por la tarjeta *DAQCard-6024E*. La comunicación tarjeta-aplicación es proporcionada por los propios *drivers* de *DAQCard-6024E*, de *National Instruments (NI-DAQ)*.

Los paquetes de datos procedentes de cada canal de adquisición del polígrafo son recibidos por la tarjeta. Cuando esto ocurre, avisa al módulo de adquisición mediante funciones “callback”, que contienen las instrucciones necesarias para el tratamiento de estos datos. Finalmente, es interesante comentar que el sistema BCI implementado en el proyecto solamente utilizará dos de los cuatro canales de los que dispone el sistema de referencia.

5.2.2. Procesado de los datos.

La aplicación realiza distintos procesos sobre cada una de las señales EEG: *enventanado prefiltrado*, *filtrado*, *enventanado postfiltrado* y *análisis*. La figura 5.3 muestra el esquema que sigue el procesado de datos.



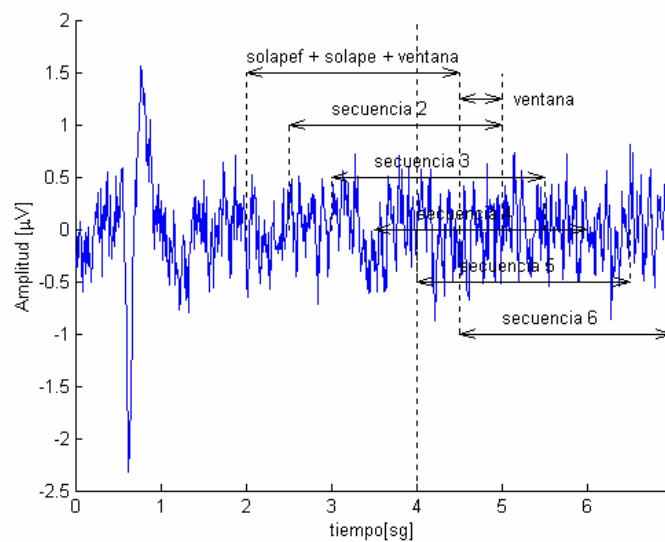
Figura 5.3: Proceso de Análisis seguido por la señal.

A continuación, se comentarán los distintos pasos que sigue la señal en el proceso de análisis:

- **Enventanado prefiltrado:** Toma una secuencia de muestras de longitud *solapef+solape+ventana*. El solape adicional (*solapef*), se incluye con el objetivo de que absorba el efecto del transitorio del filtrado.
- **Filtrado:** La señal pasa por un filtro creado en un editor. Este editor es mostrado al pulsar el botón *Editor Filtros* del panel de control. En él se puede definir el tipo de filtro, el orden, la banda de frecuencia, etc. De hecho, el filtro creado en la aplicación es de *Butterworth* debido a que es la primera opción dada por el editor y a que es fácil de implementar. También es Paso Banda, de orden 5 y suele estar en la banda de frecuencia entre 8 y 12 Hz, que es la banda donde suelen aparecer las ondas μ empleadas en el sistema BCI, aunque dependiendo del individuo la banda de frecuencia será distinta.
- **Enventanado postfiltrado:** Elimina las muestras incluidas por *solapef*, ya que después de haber absorbido el efecto transitorio del filtrado no son necesarias.

-
- **Análisis:** Se analizan las muestras correspondientes a *solape+ventana*. Los parámetros *solape* y *ventana* son elegidos por el usuario a través del panel de control. En esta etapa del procesado de datos se extraen los parámetros de interés de las muestras obtenidas. En este caso, se ha obtenido la potencia de las muestras que forman la ventana. Para ello, se eleva cada muestra al cuadrado, se suman y se dividen por el número de muestras, con esto se consigue realizar un promediado de la potencia.

Una ventana de muestras estará formada por las muestras procedentes de ventanas anteriores (*solape*), más una serie de muestras nuevas (*ventana*). La figura 5.4 muestra gráficamente el enventanado de las señales adquiridas.



5.4: Superposición de secuencias analizadas.

Para finalizar, comentar que se capturará una nueva ventana de muestras cada 31.25 mseg. que corresponden a una frecuencia de muestreo de 128 Hz y un tamaño de ventana de 64 muestras, de las cuales 60 son de *solape* y 4 son muestras nuevas (*ventana*). Por tanto, cada 128 Hz y 4 muestras ($4/128 = 31.25\text{mseg}$) se debe realizar todo el procesado de datos y la posterior clasificación de los valores obtenidos.

5.2.3. Clasificador.

La clasificación establece un valor tras el cálculo de la potencia media de las muestras filtradas. Este valor obtenido se transfiere al mundo virtual, produciendo un desplazamiento a derecha o a izquierda del objeto biofeedback.

El clasificador que utiliza la BCI es de tipo lineal (*LDA, Linear Discriminant Analysis*) y viene caracterizado por la expresión:

$$dist = w_0 + w_1 \cdot pot_1 + w_2 \cdot pot_2$$

donde:

- pot_1 y pot_2 : Representan las potencias medias de los dos canales EEG registrados en un determinado intervalo. Esta potencia media se calcula directamente elevando al cuadrado cada muestra y haciendo un promedio de ellas.
- w_0 , w_1 y w_2 : son constantes que actúan a modo de pesos, proporcionados por el clasificador tras una fase de entrenamiento del sujeto.

El valor $dist$ obtenido es multiplicado por una serie de valores:

- Si $dist > 0$, $dist_der = dist \times Factor_der \times Cte_der$.
- Si $dist < 0$, $dist_izq = dist \times Factor_izq \times Cte_izq$.

donde:

- $dist_der$ y $dist_izq$: Determina en qué medida se desplaza el objeto biofeedback a la derecha y a la izquierda.
- $Factor_der$ y $Factor_izq$: Se emplea para ajustar la potencia obtenida con el clasificador a las condiciones del entrenamiento.
- Cte_der y Cte_izq : Realizan un escalado al desplazamiento para ajustarlo a las dimensiones del mundo virtual implementado debido a que el desplazamiento del objeto biofeedback (coche) va de -3 a 3.

5.2.4. Realimentación o biofeedback al sujeto.

El biofeedback se encarga de presentar al sujeto el resultado de su actividad mental. Para ello, se transfieren los valores $dist_der$ y $dist_izq$ obtenidos en la clasificación a la interfaz gráfica. En el sistema implementado, los cambios producidos se traducen en un desplazamiento a izquierda o derecha del vehículo, que hace de objeto biofeedback en el mundo virtual.

5.3. Descripción de la aplicación final desarrollada.

El programa encargado de gestionar todo el funcionamiento de la aplicación es HM_panel_rv. Las funcionalidades de la aplicación pueden dividirse en tres principalmente:

- **Modelado de Filtros:** Está gestionado a su vez por el programa *ModelarFiltros*.
- **Medida y análisis de la señal:** En este caso, se llaman de forma anidada los programas *Ensayo*, *Present*, *Medir*, *Prueba*, *tiempo2frec* y *desp_barra*.
- **Control de la interfaz implementada:** La interfaz gráfica, desarrollada en Realidad Virtual, es controlada por la principal línea de ejecución que está asociada a los programas *Ensayo*, *Present*, *Medir*, *Prueba*, *tiempo2frec* y *desp_barra*.

El diagrama de bloques asociado a la aplicación desarrollada, puede verse en la figura 5.5:

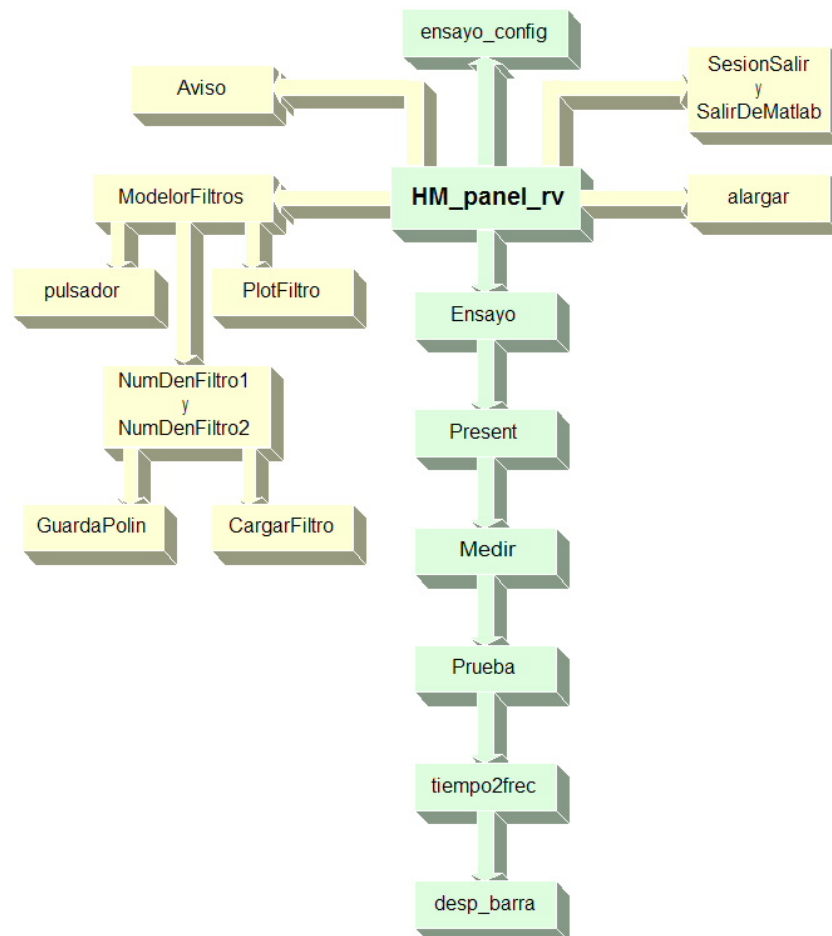


Figura 5.5: Diagrama de bloques del sistema BCI desarrollado.

En el diagrama de bloques, la principal línea de ejecución es la asociada al color verde. Es en estos programas donde se ha integrado la nueva interfaz gráfica implementada en este proyecto.

A continuación, se expondrá brevemente la función realizada por cada uno de los programas y cómo se interrelacionan entre sí, con el fin de que el usuario pueda intentar hacer las modificaciones que considere oportunas en la aplicación.

- **alargar:** Es una función encargada de crear una cadena de caracteres, a partir de un carácter. Por ejemplo, si se le da una letra ‘a’ y se le pide que devuelva una cadena de 5 veces esa letra, devuelve ‘aaaaa’. Resulta útil para guardar en un vector todos los tipos de prueba establecidos en un ensayo.
- **Aviso:** Es la función encargada de presentar cualquier mensaje por pantalla y esperar a que el sujeto confirme su lectura.
- **desp_barra:** Decide sobre el movimiento del objeto biofeedback (coche) y lo ejecuta. La decisión se toma a partir del análisis de las muestras hecho por tiempo2frec. Esta función es la encargada de actualizar el mundo virtual durante la fase de análisis de las señales.
- **ensayo_config.mat:** Es un archivo MAT que guarda la configuración del panel de control. La actualización del archivo puede realizarse en tres ocasiones: al salir del programa, al salir de Matlab, o justo antes de iniciar el ensayo (la primera vez que se pulsa este botón). El único dato que cambia es el nombre del ensayo al que se le añade un número. Para volver a la configuración original, definida por la aplicación, hay que pulsar el botón ‘Resetear Parámetros’.
- **Ensayo:** Es el responsable de la organización general del ensayo. Controla el número de pruebas, llama al programa Present cuando se inicia una prueba y devuelve el mundo virtual a su estado inicial cuando finaliza. Por último, graba la configuración tras el ensayo y antes de salir de la aplicación.
- **Guardapolin:** Guarda en una matriz los bancos de filtros definidos. Cada canal tiene asociado un banco de filtros. Los filtros que forman un banco, se modifican individualmente.
- **HM_panel_rv:** Es el programa encargado de gestionar la Herramienta de Medida. Recoge los datos o parámetros de ensayo desde un panel de control,

vigilando que se introducen correctamente, y avisando por pantalla en caso de error.

- **Medir:** Inicializa la tarjeta de adquisición, configurándola para que llame a la función Prueba cada 4 muestras a una frecuencia de 128 Hz (31.25 mseg.). Cuando pasa el tiempo que dura la prueba, devuelve el control al programa que lo llamó (Present).
- **ModelarFiltros:** Muestra el panel de diseño de filtros y gestiona los datos introducidos a través de él. Cuando se modifica un dato, se actualizan los filtros, reflejándose el resultado en las pequeñas gráficas que hay a la izquierda.
- **NumDenFiltro1, NumDenfiltro2:** Son los encargados de calcular los filtros. Existen dos modos de calcularlos: el modo 1 emplea NumDenFiltro1, y el modo 2 utiliza NumDenFiltro2. Para cada modo se usan distintos parámetros de definición.
- **Present:** Es el encargado de realizar los últimos ajustes en las distintas figuras que hay abiertas antes de llamar al programa Medir.
- **Prueba:** Es el encargado de actualizar el mundo virtual y de hacer aparecer los obstáculos en el momento adecuado, recoge las muestras de la tarjeta y las manda analizar (tiempo2frec) e interpretar (desp_barra). Además graba todas las muestras, utilizando el fichero 'rtotal.mat' cuando finalizan las pruebas de un ensayo, y también graba los filtros que se han diseñado para el ensayo en ficheros 'filtro_chk.mat' donde la 'k' representa el número del canal al que corresponde el filtro.
- **pulsador:** Muestra un pulsador '>>' en la esquina inferior derecha de cualquier figura. Debe pulsarse antes de devolver el control al programa que lo llamó.
- **PlotFiltro:** Es llamado por ModelarFiltros y representa la respuesta de los filtros creados.
- **SesionSalir:** Se usa para salir de la aplicación, pero no de MATLAB y guarda los parámetros del ensayo en ensayo_config.mat.
- **SalirDeMatlab:** Cierra la aplicación y también sale de MATLAB. Además guarda la configuración en ensayo_config.mat.

-
- **tiempo2frec:** Es el encargado de filtrar la señal y realizar su análisis en frecuencia. Calcula la potencia media de las señales capturadas por los dos canales establecidos. Los resultados son interpretados por desp_barra.

La aplicación implementada no se puede utilizar de manera aislada, ya que necesita recibir las señales del sujeto (el resultado de su adquisición, procesado y clasificación), para traducirlas al entorno virtual.

Al ejecutar la aplicación, aparecen una serie de ventanas que permiten al usuario configurar las condiciones y parámetros de cada ensayo, que son básicamente una serie de pruebas de entrenamiento para el sujeto.

5.3.1. Funcionamiento general del sistema BCI desarrollado.

Una vez definida la interfaz virtual y su conexión con el resto de la aplicación, es necesario describir el protocolo de entrenamiento seguido para cada sujeto. Para ello, se utilizan los términos de experimento, sesión, ensayo y prueba. Una sesión de entrenamiento se organiza en una serie de ensayos que a su vez están divididos en pruebas.

Posteriormente, se detalla la estructura de las pruebas de entrenamiento, es decir, el paradigma de entrenamiento, ya que forma parte de las especificaciones iniciales y condicionan la programación de la aplicación.

5.3.2.1. Concepto de experimento.

Un experimento son las señales EEG captadas en un ensayo a través de electrodos situados en lugares estandarizados del cuero cabelludo, mediante lo que se llama *ElectroCap*. Se trata de un gorro con una serie de conectores a través de los que se conectan los electrodos. Otras características como la conductancia de la piel son también captadas y transferidas al polígrafo a través de electrodos pregelados desechables.

5.3.2.2. Concepto de sesión.

Una sesión es una cita con un determinado sujeto al que se pretende hacer una serie de experimentos. La aplicación desarrollada permite configurar los ensayos de manera diferente dentro de una misma sesión.

5.3.2.3. Concepto de ensayo.

Un ensayo es un conjunto de pruebas de igual o distinto tipo, orientadas a un objetivo común. Las sesiones iniciales deberían estar formadas por un solo tipo de prueba, favoreciendo la concentración del sujeto en un único tipo de actividad mental. De forma progresiva, se introducirán pruebas de distinto tipo en un mismo ensayo, para que el sujeto mejore su capacidad de cambiar entre estados mentales.

5.3.2.4. Concepto de prueba

Una prueba es el elemento más pequeño en un experimento, el registro mínimo que se puede hacer de las señales de un sujeto. Las pruebas están constituidas de una serie de fases comentadas con anterioridad como son: *presentación del mundo virtual*, *establecimiento de la tarea a ejecutar*, *control sobre el objeto biofeedback* y *descanso*.

Las pruebas pueden ser de dos tipos: *con feedback* o *sin feedback*.

- **Prueba feedback:** Mostrará el resultado de la tarea mental realizada por el sujeto durante toda la fase de análisis de las señales. Los resultados de la actividad mental producirán cambios en el objeto biofeedback del mundo virtual. El coche es el objeto biofeedback, como ya se comentó, y se desplazará a izquierda o a derecha en función del parámetro que se obtenga en la fase de análisis y clasificación.
- **Prueba no feedback:** Captura las señales EEG del sujeto bajo prueba y las procesa, pero no deja que se clasifiquen y actúen sobre la escena virtual. Por tanto, las señales del sujeto no tendrán ningún efecto en el coche.

Tarea mental a realizar.

Las tareas mentales que se deben realizar y los efectos asociados son los siguientes:

- Si la tarea mental que se realiza es pensar en el movimiento de la mano derecha, el coche se desplazará a la derecha.
- Si por otro lado, el sujeto se relaja (mantiene la mente en blanco), el coche se desplazará a la izquierda.

Ambas tareas mentales se realizan con el objetivo de sortear los obstáculos implementados en el sistema BCI, durante el tiempo de análisis de la prueba.

Estructura temporal de una prueba.

La figura 5.6 muestra la estructura temporal de una prueba genérica, que consta de los parámetros que se definen a continuación:

- **Inicio:** Instante en el que aparece el mundo virtual con su configuración inicial.
- **t_{objetivo} :** Instante en que aparecen los obstáculos, indicando de qué punto a qué punto a lo largo de la carretera tendrá lugar el intervalo de análisis.
- **$t_{\text{análisis}}$:** Instante en que comienza el análisis.

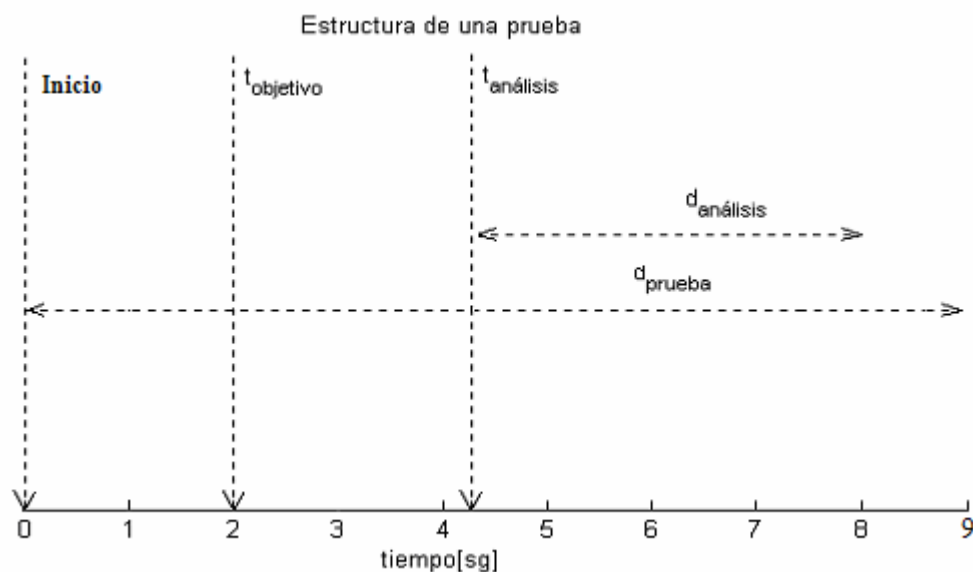


Figura 5.6: Estructura temporal de una prueba genérica.

- **$d_{\text{análisis}}$:** Duración del período de análisis, intervalo temporal donde se procesarán las señales EEG del sujeto (y provocarán cambios en el mundo virtual si es una prueba de tipo feedback). Se mide a partir de $t_{\text{análisis}}$.
- **d_{prueba} :** Duración total de la prueba.

-
- $t_{desc_ep_fijo}$ y $t_{desc_ep_aleat}$: Valores temporales que dan lugar a un tiempo de espera entre pruebas. Este tiempo se compone de una parte fija $t_{desc_ep_fijo}$ y una parte aleatoria $t_{desc_ep_aleat}$. La finalidad de estas dos componentes, es que haya un pequeño descanso entre pruebas, pero que no sea siempre exactamente el mismo.

Después de establecer las distintas fases en las que se divide una prueba, es necesario puntualizar ciertos aspectos de dichas etapas.

- En primer lugar, el objeto biofeedback y los obstáculos deben estar presentes cuando comience la fase de análisis.

$$0 \leq t_{objetivo} < t_{análisis}$$

- En segundo lugar, la fase de análisis nunca excederá en tiempo a la duración de la prueba.

$$t_{análisis} + d_{análisis} \leq d_{prueba}$$

5.4. Sistema de medida de tiempos de reacción.

Se ha desarrollado una segunda aplicación a partir del sistema BCI implementado. Se pretende medir el tiempo que tarda un sujeto en reaccionar, desde que aparece un tipo de obstáculo determinado hasta que realiza la acción necesaria para esquivarlo.

5.4.1. Descripción de la aplicación desarrollada.

La funcionalidad de la aplicación es el control de la interfaz implementada. La interfaz gráfica, desarrollada en Realidad Virtual, es controlada por la línea de ejecución formada por los programas *HM_panel_tr*, *Ensayo*, *Present*, *Medir*, *Prueba* y *pulsar*.

El diagrama de bloques asociado a la aplicación desarrollada, puede verse en la figura 5.7:

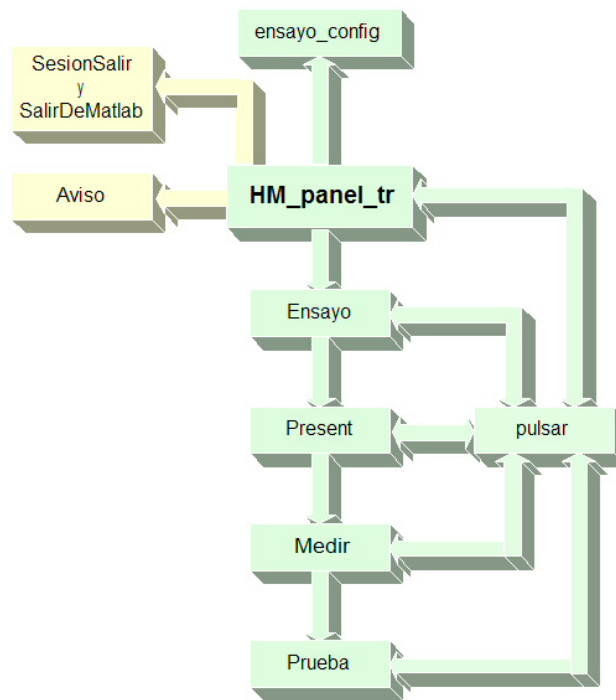


Figura 5.7: Diagrama de bloques del sistema de medida de los tiempos de reacción.

Puede verse que esta segunda aplicación no emplea ningún programa relacionado con la adquisición y análisis de las señales. Se ha tomado únicamente los programas relacionados con la gestión de las pruebas.

A continuación, se expondrá brevemente las modificaciones que se han realizado sobre algunos de los programas.

- **Medir:** La aplicación desarrollada posee dos versiones y difieren principalmente en este punto. La primera versión emplea la tarjeta de adquisición, por tanto hay que inicializarla y configurarla en este programa, para que llame a la función Prueba cada 4 muestras a una frecuencia de 128 Hz (31.25 mseg.). Y en la segunda versión no se usa la tarjeta de adquisición, por tanto se debe controlar el tiempo de ejecución de cada actualización del mundo virtual para que se cumplan los 31.25 mseg. Cuando pasa el tiempo que dura la prueba, devuelve el control al programa que lo llamó (Present).
- **Prueba:** Es el encargado de actualizar el mundo virtual y de hacer aparecer el obstáculo muro en el momento adecuado. Además se le ha eliminado todo lo relacionado con la adquisición y el análisis.
- **pulsar:** Es una función *callback* definida para que el sujeto bajo prueba interaccione con el mundo virtual.

5.4.2. Funcionamiento general de la aplicación.

El sistema de medida de los tiempos de reacción, como ya se ha comentado con anterioridad, es una particularización del sistema BCI implementado (objetivo principal del proyecto). Es básicamente, la BCI sin adquisición y sin procesado de la señal, por tanto la aplicación gira totalmente entorno a la gestión de la interfaz gráfica utilizada.

Para definir el funcionamiento general de la segunda aplicación creada en este proyecto, son necesarios conceptos usados con el sistema BCI, tales como: *sesión*, *ensayo* y *prueba*.

- Una sesión es una cita con un determinado sujeto al que se pretende hacer una serie de ensayos.
- Un ensayo es un conjunto de pruebas orientadas a un objetivo común.
- Una prueba es el elemento más pequeño en un experimento.

Tarea a realizar dentro de una prueba.

La tarea consiste en mover el coche, que aparece en el mundo virtual, consiguiendo esquivar el obstáculo muro. La colisión entre el muro y el coche se evitará empleando los cursores izquierdo y derecho, encargados del movimiento del vehículo.

- Si se pulsa el cursor derecho, el coche se desplazará hacia la derecha del sujeto.
- Si se pulsa el cursor izquierdo, el coche se desplazará a la izquierda.

Si el coche está posicionado en un carril lateral y se quiere ir hasta el otro, se debe pulsar dos veces el cursor asociado a ese desplazamiento. En ningún caso, el coche rebasará los límites de los carriles laterales. Es decir, si el coche se encuentra en el carril izquierdo y se pulsa el cursor izquierdo no sucederá nada, porque está situado en la posición más a la izquierda posible.

Con esta actividad se pretende medir el tiempo que tarda el sujeto bajo prueba en reaccionar, desde que aparece el muro hasta que se pulsa una tecla (cursor izquierdo o derecho). Es decir, cuanto tiempo tarda el sujeto en reaccionar ante un estímulo determinado.

En función del resultado de la prueba se enviará distintos valores por el puerto paralelo (Se detallará más detenidamente en el Manual de Usuario).

Estructura temporal de una prueba.

La estructura temporal de una prueba genérica mostrada en la figura 5.8, vendrá determinada por los siguientes parámetros temporales:

- **t_muro1:** Indica cuánto tiempo (en segundos) es visible el muro antes de que esté a la altura del coche.
- **t_muro2:** Hace lo mismo que t_muro1.
- **t_colision:** Instante (en segundos) máximo que tarda el muro en estar a la altura del coche. En función, de esta variable se posicionará el muro en el mundo virtual. La aplicación elige aleatoriamente un valor entre 1 y t_colision, para establecer el tiempo que tarda el muro en estar a la altura del coche desde que empezó la ejecución.
- **Duración de la prueba:** Duración (en segundos) de la prueba.
- **Desc. entre pruebas:** Descanso (en segundos) entre pruebas como la suma de dos valores, uno fijo y otro aleatorio.

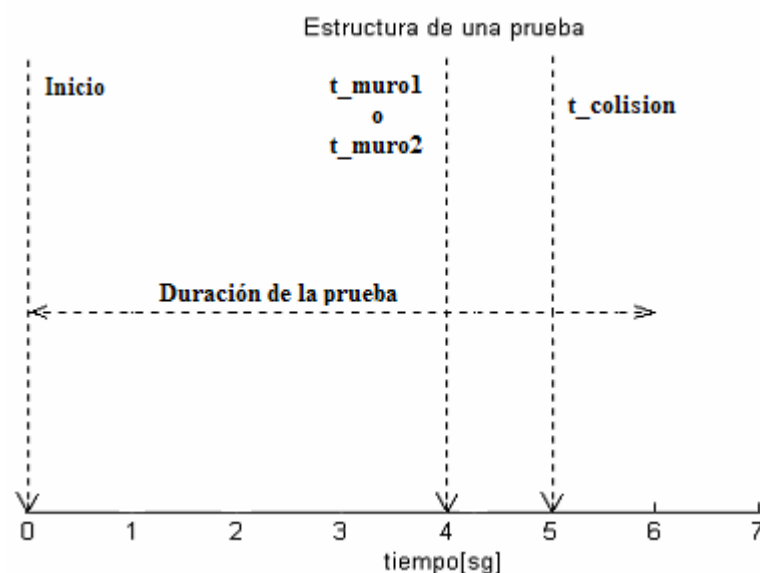


Figura 5.8: Estructura temporal de una prueba genérica.

En la figura 5.8, se observa que t_muro1 y t_muro2 establecen el tiempo durante el cual el obstáculo es visible antes de llegar a la altura del coche. Es decir, el tiempo del que se dispone para poder esquivar el muro. En el ejemplo de la figura 5.8, el muro será

visible a los 4 segundos y estará a la altura del coche a los 5. Si el coche en t_{colision} no esquivo el muro, colisionará. A partir del diagrama temporal de la figura, se puede deducir que t_{muro1} o t_{muro2} debe tener en el panel de control el valor 1. Las variables t_{muro1} y t_{muro2} pueden tomar valores distintos.

La mitad de las pruebas de un ensayo emplea el tiempo t_{muro1} y la otra mitad t_{muro2} , el establecimiento de estos tiempos en cada prueba se hace de forma aleatoria. Además, el muro aparecerá aleatoriamente a la derecha o a la izquierda de la carretera, pero en este caso, no existirá ninguna restricción con respecto al número de veces que debe aparecer a cada lado. Debido a la longitud de la carretera, el muro sólo podrá ser visible como máximo durante 3.75 segundos, por tanto no tienen sentido valores de tiempo mayores a 3.75 segundos en t_{muro1} y t_{muro2} .

Si t_{muro1} o t_{muro2} es cero, el muro aparecerá cuando se ha colisionado sin poder hacer nada. En la figura 5.9 se puede observar como los instantes temporales t_{muro} y t_{colision} coinciden. Esta situación demuestra que el muro será visible justo en el momento que se encuentre a la altura del coche. Por este motivo, se colisionará sin poder evitarse.

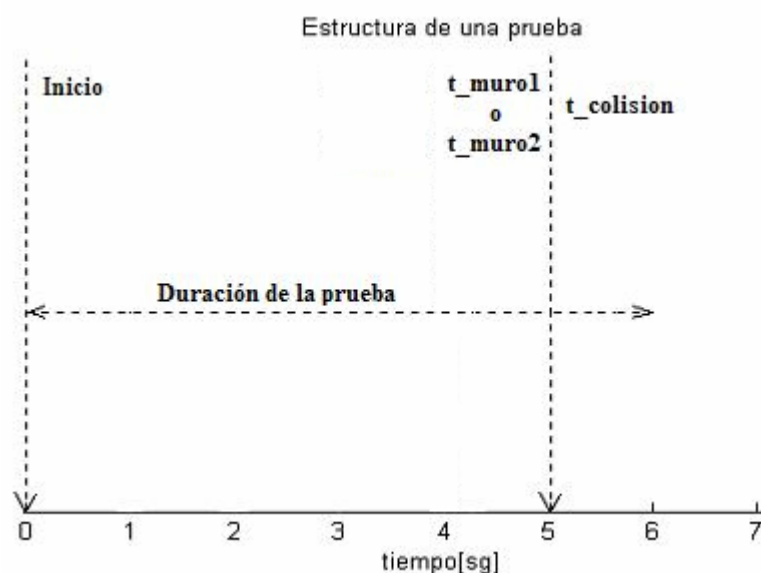


Figura 5.9: Estructura de una prueba cuando el muro se muestra sólo en el instante de la colisión.

En definitiva, t_muro1 y t_muro2 deben cumplir con un determinado intervalo de valores para que el resultado de las pruebas sea el más adecuado. De todas las situaciones planteadas, se ha extraído la siguiente restricción:

$$0 < t_muro1, t_muro2 \leq 3.75$$

Como se ha comentado anteriormente, el tiempo que tarda en llegar el muro a la altura del coche no tiene porque coincidir con el valor de la variable $t_colision$, el cual ha sido introducido a través del panel de control. En las figuras 5.8 y 5.9 se ha supuesto que estos dos valores coinciden.

Si por ejemplo la variable $t_colision$ toma el valor 6 en el panel de control, se creará un intervalo de valores de 1 a 6. Seguidamente se elegirá de manera aleatoria uno de esos valores, que determinarán la posición del muro en la prueba. La figura 5.10 muestra la situación planteada.

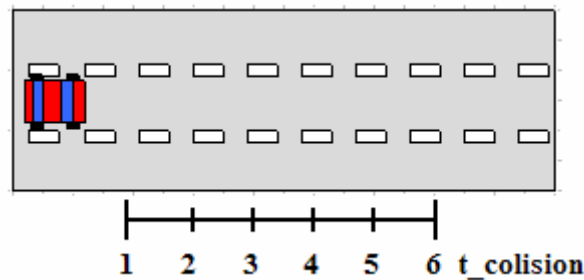


Figura 5.10: Instantes temporales en los que se puede colocar el muro antes de la posible colisión.

Si $t_colision$ es mayor que el tiempo de duración de la prueba, puede que el muro no aparezca porque se posicione muy lejos y no llegue a ser visible. La figura 5.11 muestra la estructura temporal de una prueba en la que el obstáculo muro no llega a verse por ser el valor de $t_colision$ mayor que la duración de una prueba. El tiempo que tarda en llegar el muro a la altura del coche es igual a $t_colision$, tal y como sucedía en la figura 5.8 y 5.9.

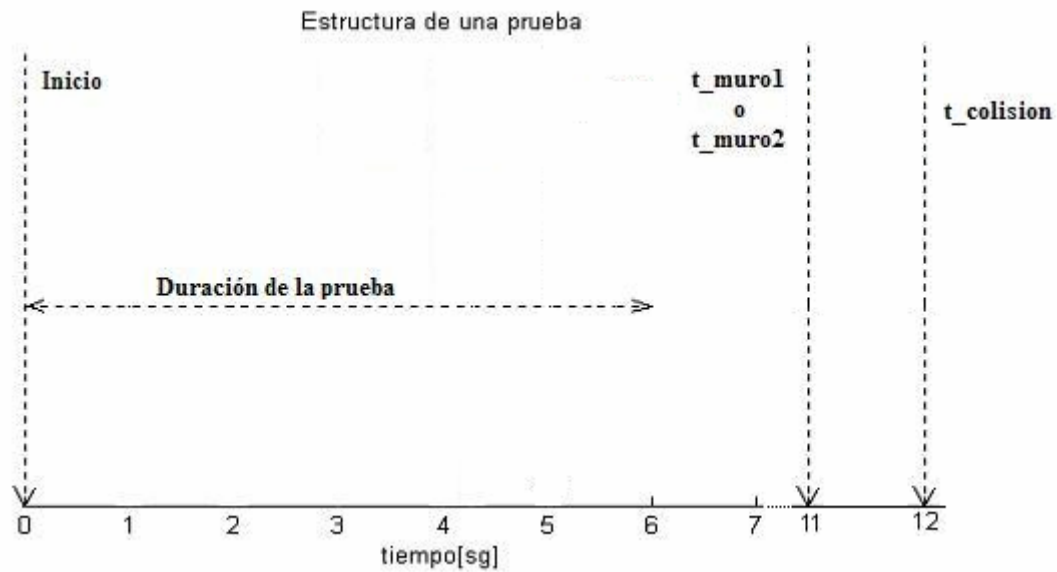


Figura 5.11: Estructura de una prueba en la que el muro no llega a ser visible.

El muro siempre debe llegar a la altura del coche, a partir de esta premisa puede extraerse una restricción en los parámetros que definen la estructura temporal de una prueba.

$$t_colision \leq duraci3n_de_la_prueba$$

En cambio, si $t_colision$ es menor que el t_muro1 o t_muro2 , el obstáculo será visible desde que se posiciona en el mundo virtual. La figura 5.12 muestra como desde el inicio de la prueba el muro es visible.

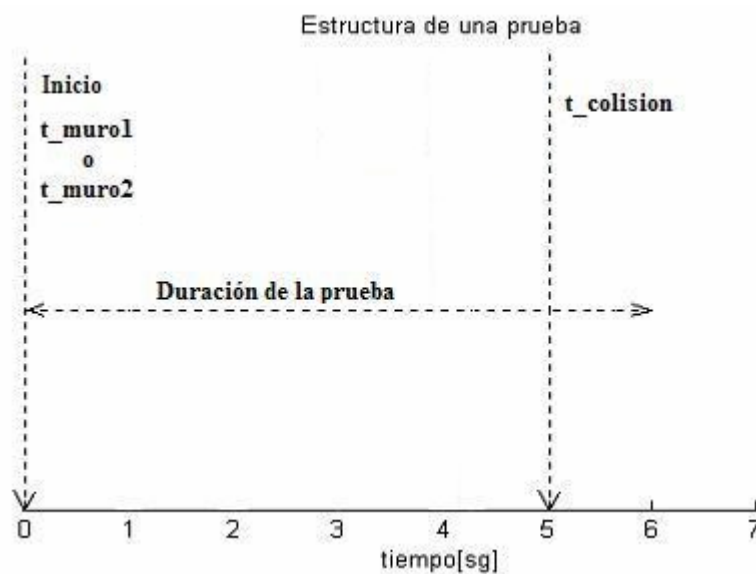


Figura 5.12: Estructura temporal en la que el muro es siempre visible.

Finalmente, si el coche colisiona con el muro, se debe esperar a que transcurra el tiempo restante de la prueba en ejecución antes de comenzar una nueva prueba.

Capítulo 6.

Ventajas, limitaciones y pruebas de las aplicaciones diseñadas en el proyecto.

Resultados experimentales.

En este capítulo, se presentan las ventajas de las dos herramientas desarrolladas, al mismo tiempo que se exponen las restricciones a las que están sometidas. Las limitaciones son debidas principalmente a las propias del procesador y de la tarjeta gráfica del equipo empleado. Al haberse desarrollado las herramientas en un equipo concreto, si se ejecutan en otro de menores prestaciones no se obtendrán los mismos resultados.

Finalmente, se plantearán los resultados de los experimentos realizados que se corresponde con la última fase del proyecto.

6.1. Sistema BCI.

La interfaz creada en el proyecto se ha integrado en una herramienta desarrollada por el grupo DIANA del Departamento de Tecnología Electrónica, constituyendo un sistema BCI completo. A continuación, se detallarán tanto las ventajas como las limitaciones que posee la aplicación final resultante. Así como, las pruebas realizadas durante el proceso de desarrollo de la aplicación.

6.1.1. Ventajas de la aplicación.

La aplicación que define la herramienta encargada de la gestión del sistema BCI, presenta una serie de características beneficiosas para su uso. En los siguientes puntos se plantearán las ventajas que presenta esta primera aplicación:

- La herramienta es flexible y fácilmente configurable gracias a la librería de adquisición de datos (*DAQ*) de MATLAB. Por ese motivo, pueden emplearse distintos valores para la frecuencia de muestreo, tamaño de la ventana, solape, etc.
- Permite incorporar de manera sencilla nuevos algoritmos para el procesado, gracias a la modularidad del código y a la disponibilidad de numerosos algoritmos de análisis de señal.
- La construcción de nuevos mundos virtuales para la aplicación es rápida e intuitiva, gracias al programa *V-Realm Builder* que incorpora MATLAB.
- Además se puede incorporar, sin dificultad, nuevos mundos virtuales con nuevos comportamientos, debido a que el *Virtual Reality Toolbox* da la posibilidad de un manejo sencillo del mundo virtual a través de líneas de comandos.
- La totalidad del código de la herramienta está escrito en código MATLAB, lo cual la hace flexible para introducir nuevos cambios que no hayan sido contemplados en la actual versión.
- Una ventaja adicional es que al tratarse de código MATLAB, este es un lenguaje de programación más amigable que el lenguaje C/C++.
- La idea de emplear mundos virtuales en sistemas BCI no es nueva. De hecho, existe una aplicación similar desarrollada con *WorldToolKit* (WTK), una librería de funciones escritas en lenguaje C que permite implementar aplicaciones de Realidad Virtual. Pero *WorldToolKit* presenta una serie de limitaciones que está obligando a desarrollar este tipo de aplicaciones en otros lenguajes como MATLAB. *WorldToolKit* emplea un hardware muy determinado, una mochila, que en estos momentos es insustituible porque ya no se puede adquirir, en cambio MATLAB no necesita ningún elemento hardware de ese tipo. Además, *WorldToolKit* es una herramienta asíncrona, y en consecuencia, presenta problemas de sincronización, al contrario que MATLAB que es totalmente síncrona.

6.1.2. Limitaciones de la aplicación.

En este apartado, se explican las limitaciones de la interfaz desarrollada. El proceso de diseño de esta interfaz está condicionado por las especificaciones temporales que se deben cumplir y muy especialmente por el ordenador en donde se va a ejecutar. Todo esto ha provocado ciertas restricciones en la herramienta que se detallan a continuación.

- **Limitación en el número de obstáculos:** La aplicación implementada ofrece al usuario la posibilidad de configurar, al inicio de cada ensayo, los obstáculos (*charco*, *muro* y *rampa*) que aparecerán. Sin embargo, no todos los obstáculos que se han seleccionado en los cuadros de diálogo (se detallará en el Manual de Usuario) se presentarán en cada prueba.

Todos los obstáculos que aparecen en el mundo virtual son cargados al iniciar el ensayo. Los que no son utilizados sólo ocuparán recursos de memoria y serán “invisibles” para el sujeto. El motivo de que no sea posible cargar el obstáculo en cada prueba, es que retrasaría mucho el sistema y no se cumpliría el procesado en tiempo real de las señales cerebrales. Dicha especificación es la más exigente.

También existe una limitación en el número de obstáculos gestionados a la vez en la aplicación, ya que más de dos obstáculos producen retardos en la ejecución de la herramienta.

- **Limitación en las actualizaciones por segundo del mundo virtual:** El sistema BCI desarrollado funciona con una frecuencia de muestreo de 128 Hz y un tamaño de ventana de 64 muestras, de las cuales solo 4 serán por lo general muestras nuevas de entre ventanas consecutivas. En estas circunstancias se configura la tarjeta de adquisición para que capture datos cada 31.25 mseg.

$$\frac{\text{Número_de_muestras_nuevas}}{\text{Frecuencia_muestreo}} = \frac{4}{128} = 31.25\text{mseg.}$$

Por tanto, en los 31.25 mseg debe procesarse la información obtenida y además actualizarse el mundo virtual. Si se realiza una sola actualización del mundo cada 31.25 mseg, la imagen mostrada se modificará a una frecuencia de 32 Hz. Es una frecuencia suficiente para mantener una continuidad temporal, pero producirá una imagen de mala calidad porque el movimiento del entorno

será muy lento. En cambio, si se realiza dos actualizaciones del mundo virtual, la imagen mostrada aumenta su velocidad de modificación a 64 Hz y en definitiva, será más dinámica. Finalmente, más de dos actualizaciones del mundo no tiene sentido porque superaría los 31.25 mseg de los que se dispone.

Esta restricción también ha influido en el diseño del entorno virtual, ya que incluir demasiados elementos y efectos en la escena sobrecargaría el sistema e impediría que se ejecute correctamente la temporización, porque los posibles retardos serán acumulativos.

- **Condiciones temporales de la prueba según el obstáculo:** La ejecución de cada prueba estará influenciada por el tipo de obstáculo que se presente al sujeto. Esto es debido a que cada uno de los obstáculos produce un efecto diferente en el mundo virtual. Como se comentó en el capítulo 4, si es el muro el que aparece y no se realiza correctamente la tarea mental, el objeto biofeedback (coche) choca contra él, escuchándose el sonido de rotura de cristales y por último, si aparece la rampa y el sujeto acierta con la acción a realizar, se asciende por ella y se cae de nuevo a la carretera tras permanecer un breve instante en el aire.

Cada uno de los efectos creados requiere de un tiempo para realizarse. Se reproducen justo después de la fase análisis de las señales del sujeto y antes del intervalo de descanso.

El muro y la rampa necesitan, aproximadamente, medio segundo adicional después del análisis, para dar tiempo a que se ejecute el efecto correspondiente. Por ejemplo, si el análisis dura 4 segundos y comienza en el instante 3 segundos, la duración de la prueba será de 8 segundos, el medio segundo asociado al efecto de los obstáculos y medio más por posibles pequeños retardos, debido a que el programa se ejecuta muy al límite de las posibilidades del PC.

Esta restricción está contemplada en el diálogo que permite configurar los parámetros del mundo virtual.

- Una limitación adicional es que al tratarse de **código interpretado** su velocidad de ejecución es menor a la del código compilado (como es C/C++, por ejemplo), y además consume muchos más recursos. De hecho, el programa ha sido probado y desarrollado en un ordenador potente con una buena tarjeta gráfica (Procesador Core2Duo 1.7 GHz y tarjeta gráfica 256 MB de memoria DDR3), y aún así la aplicación funciona al límite de sus posibilidades temporales.

6.1.3. Pruebas realizadas durante el proceso de diseño.

En el proceso de desarrollo del sistema BCI, ha sido necesario realizar una serie de pruebas para comprobar que todo se estaba implementando correctamente. Al inicio, la aplicación se desarrollaba en un portátil de 1.6 GHz y 512 MB de memoria RAM. Pero las características de este ordenador eran insuficientes y por tanto, la aplicación no funcionaba correctamente. El problema se debía principalmente a que la tarjeta gráfica era integrada y para el manejo de gráficos en 3D era necesaria una buena tarjeta externa.

Antes de cambiar de ordenador, se realizaron algunas pruebas para ver hasta que punto era necesario un nuevo equipo y además, para comprobar los límites temporales. La primera prueba se hizo configurando la tarjeta de adquisición a 128 Hz y 4 muestras nuevas en cada ventana de datos capturados. Además, se decidió que la duración de cada prueba fuera de 9 segundos, al igual que en los experimentos realizados. Bajo estas condiciones, la tarjeta de adquisición realizó un total de 288 llamadas por prueba ($128 \cdot 9/4 = 288$).

$$\text{Número_llamadas_por_prueba} = \frac{\text{Frecuencia_muestreo} \times \text{Duración_prueba}}{\text{Número_muestras_nuevas}}$$

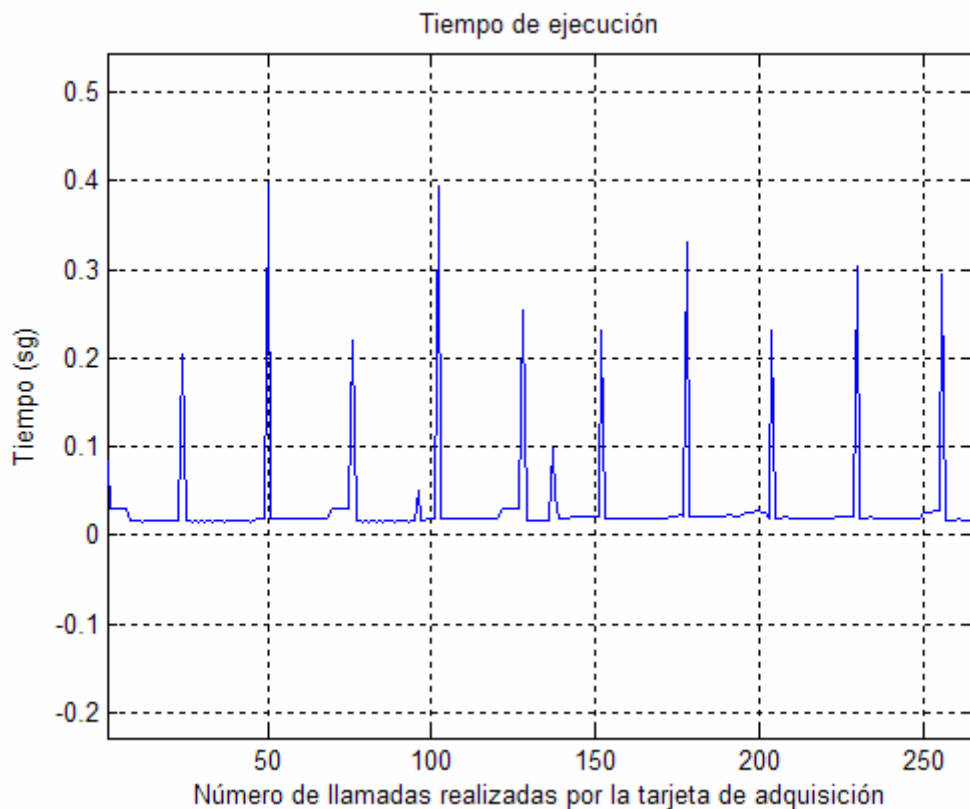


Figura 6.1: Retrasos en la aplicación a 128 Hz, 4 muestras y 1 actualización.

En la figura 6.1 puede verse que la aplicación a 128 Hz, 4 muestras y 1 actualización del mundo virtual por llamada, produce picos de retraso con una cierta periodicidad. Se observa que los picos aparecen aproximadamente cada 25 llamadas de la tarjeta de adquisición.

La simulación con una actualización produce retrasos que están diez veces por encima del límite temporal de cada llamada. Si cada llamada dura 31.25 mseg, el pico de retraso alcanza los 300 o 400 mseg. Por tanto, la simulación ideal que es con dos actualizaciones, como se comentó en el apartado 6.1.2, empeorará aún más los tiempos. Además, se observa que no se ejecutan todas las llamadas debido al retraso, ejecutándose únicamente 268.

En la figura 6.2, se observa que cada actualización del mundo virtual tarda 0.015 seg. Por lo que si se consigue eliminar los picos de retraso, se podrían realizar dos actualizaciones por llamada porque no se sobrepasaría el límite temporal (31.25 mseg.).

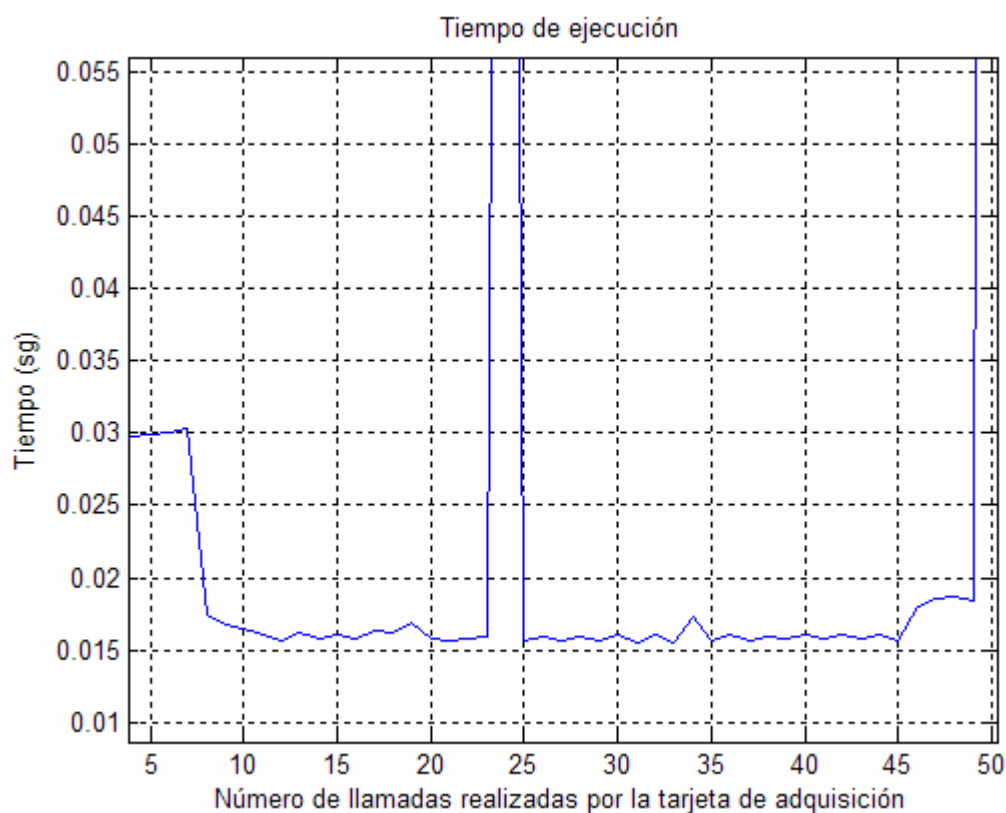


Figura 6.2: Duración de una actualización del mundo virtual.

Al no funcionar la aplicación correctamente con las especificaciones de 128 Hz y 4 muestras nuevas, se probó con otras configuraciones en la tarjeta de adquisición. Para ello, se dejó la misma frecuencia de muestreo y se fue variando el número de muestras nuevas por ventana. Los valores que se probaron fueron 8, 16 y 32 muestras, pero en ninguno de estos casos, funcionó la aplicación porque aún seguían apareciendo picos de retraso. En definitiva, se llegó a la conclusión de que era necesario un equipo más potente. En concreto, un ordenador con una buena tarjeta gráfica al estar empleándose Realidad Virtual.

A partir de ese momento, se empezó a utilizar un ordenador *Core2Duo de 1.7 GHz, 1 Giga de memoria RAM y tarjeta gráfica de 256 MB de memoria DDR3*. Con este equipo la aplicación iba mucho mejor, pero al introducir todas las especificaciones tanto las relativas al procesado de datos como las concernientes al manejo del mundo virtual, seguía existiendo un pequeño retraso.

La aplicación final quedó configurada a 128 Hz, 4 muestras y 2 actualizaciones del mundo virtual por llamada. El pequeño retraso que posee la aplicación es debido a la fase de análisis. Esta fase introduce un retardo de 3 mseg. por llamada. Si la duración de la fase de análisis es de 3.75 segundos, como en los experimentos realizados, se harán 120 llamadas por prueba. Con estos valores, se produce un retraso de 0,36 segundos ($120 \cdot 0.003 = 0.36$).

Además, la aplicación presentaba un problema con la forma de reservar memoria que tenían los arrays encargados de almacenar las muestras capturadas. En las primeras pruebas, no se notaba nada en particular, pero a partir de la prueba número 20 se producían grandes retrasos. Por este motivo, se cambió la manera de reservar memoria. Ahora la reserva de memoria se hace al inicio de la simulación, para que el retardo introducido en el proceso de almacenado no sea variable sino constante. Esta operación introdujo 0,1 segundo más de retraso. En conclusión, la suma total de los retrasos hace que la simulación acabe medio segundo más tarde. Este pequeño retraso al ser constante puede ser asumible, pero no es lo ideal.

En el intento de evitar cualquier tipo de retraso, se probó con una configuración de 128 Hz y 8 muestras. El resultado de la prueba fue positivo, porque se consiguió evitar

los retrasos. El inconveniente es que el resto de los sistemas BCI desarrollados por el grupo DIANA funcionan con 128 Hz y 4 muestras. Por lo que si se desea seguir trabajando igual, hay que asumir un pequeño retraso. En cambio, si se introducen más muestras, se evitarán los retrasos pero disminuirá el control que tiene el individuo sobre el objeto biofeedback. Debido a que el número de llamadas que se realizarían durante la prueba, se reducirían a la mitad ($128 \cdot 9/8 = 144$). En consecuencia, las llamadas durante la fase de análisis serán 60 y esto producirá que el usuario en lugar de modificar el mundo 120 veces, lo haga sólo 60.

Para que el cambio en el número de muestras nuevas por ventana fuera sencillo y rápido, se realizó una mejora que permitía introducir cualquier valor que se pueda expresar en base 2, excepto el 1 que es 2^0 , y la aplicación seguiría funcionando igual, en cuanto a la actualización del mundo virtual.

Finalmente, se comprobó que la aplicación funcionaba mejor cuando la tarjeta gráfica estaba a 70 Hz, porque iba más rápida y la interfaz gráfica producía una buena imagen continua. Se probó también con 80 Hz, pero la imagen producida daba la sensación de ir a saltos.

6.2. Sistema de medida de los tiempos de reacción.

El sistema de medida de los tiempos de reacción es una herramienta desarrollada a partir de la interfaz implementada en el proyecto. Además no poseerá ni adquisición de datos ni procesado. Por tanto, todas las ventajas y limitaciones de la aplicación girarán entorno al mundo virtual empleado. A continuación, se detallarán tanto las ventajas como las limitaciones que posee la aplicación, y las pruebas realizadas en el proceso de diseño.

6.2.1. Ventajas de la aplicación.

La aplicación que se encarga de la gestión de la herramienta definida presenta una serie de ventajas que se plantean en los siguientes puntos.

-
- La herramienta no necesita adquirir ni procesar información, esto permitirá que las condiciones temporales de la aplicación no sean tan restrictivas como en el sistema BCI. De hecho, se han implementado dos versiones de la aplicación, la primera utiliza la tarjeta de adquisición para marcar el ritmo de ejecución del mundo virtual de forma precisa, en cambio la segunda versión no usa la tarjeta de adquisición y a pesar de no ser tan precisa, resultará ventajosa porque no es necesaria ningún hardware adicional.
 - La modularidad del código del sistema BCI ha permitido desarrollar esta segunda aplicación más rápido. Para crear la nueva herramienta sólo ha sido necesario eliminar los módulos correspondientes al procesado, modificar los restantes y además, se ha creado un nuevo bloque que controla el teclado usado para interactuar con el escenario virtual.
 - Finalmente, al ser una aplicación que sólo gestiona el mundo virtual, poseerá todas las ventajas relacionadas con la construcción y gestión de mundos virtuales a través de *V-Realm Builder* y *Virtual Reality Toolbox* comentadas para el sistema BCI.

6.2.2 Limitaciones de la aplicación.

En este apartado se explican las limitaciones de la aplicación, que en su mayoría estarán relacionadas con la interfaz implementada. De hecho, las restricciones son similares a las del sistema BCI.

- **Limitación en el número de obstáculos:** El único obstáculo que aparecerá en esta aplicación es el muro, pero no para evitar retardos sino porque se trata de una especificación. Al igual que sucedía en el sistema BCI, el obstáculo que aparece en el mundo virtual es cargado al iniciar el ensayo, para evitar que el sistema se retrase.
- **Limitación en las actualizaciones por segundo del mundo virtual:** En el sistema de medida de los tiempos de reacción sucederá lo mismo que en el sistema BCI, se realizarán dos actualizaciones del mundo virtual cada 31.25 mseg. En este caso, el tiempo no es tan crítico porque no se procesa información.

-
- Finalmente, la aplicación se ha desarrollado de **dos maneras distintas**, como se comentó en el apartado anterior. La primera versión funciona configurando la tarjeta de adquisición con los mismos parámetros que el sistema BCI, es decir con una frecuencia de muestreo de 128 Hz y una ventana de 4 muestras nuevas. Pero la tarjeta no se utilizará para capturar datos sino para controlar la temporización de la aplicación. Además tiene el inconveniente de necesitar este hardware adicional (tarjeta de adquisición). En la segunda versión no se emplea la tarjeta de adquisición, lo que se realiza es un cálculo del tiempo que tarda MATLAB en ejecutar las dos actualizaciones y si es menor a 31.25 mseg. se introducirá una pausa. El inconveniente de esta segunda versión es la temporización, debido a que se ha ajustado para el ordenador en el que se ha sido desarrollado. Por tanto, si se ejecuta en un ordenador de distintas características los tiempos no serán los mismos pudiendo retrasarse o adelantarse la simulación.

6.2.3. Pruebas realizadas durante el proceso de diseño.

El sistema de medida de los tiempos de reacción tiene los mismos problemas que el sistema BCI, debido a que esta aplicación se implementó a partir del sistema BCI. Así que, todo el proceso que se realizó para establecer el número de actualizaciones del mundo virtual en cada llamada, ha sido aprovechado en esta aplicación.

Esta segunda aplicación posee una ventaja sobre la primera y es que no captura ni procesa información, por lo que no se producen retrasos.

El control del coche a través de los cursores del teclado resultó complicado. El motivo fue que existía una función callback que testeaba el teclado, pero no servía para la ventana de visionado de mundos virtuales. Por lo que se creó una ventana sólo para controlar la pulsación de los cursores izquierdo y derecho y así, poder modificar la posición del coche en el mundo virtual. Además, la ventana definida para el control del teclado no es visible.

Finalmente, se desarrolló el control de los valores enviados por el puerto paralelo, tras la ocurrencia de un evento. Para comprobar si los valores enviados en una prueba eran correctos, se simuló una prueba y con un polímetro se confirmó que los valores en el puerto eran los adecuados.

6.3. Resultados experimentales obtenidos.

La Realidad Virtual se ha empleado como técnica para implementar una interfaz de salida de un sistema Interfaz Cerebro-Computadora, que tiene como objetivo proporcionar un entrenamiento eficaz para que los sujetos que lo realicen puedan aprender a controlar de forma fiable sus señales electroencefalográficas.

Para comprobar que se consiguen los objetivos deseados, se han realizado algunas pruebas con algunos sujetos durante varias sesiones.

6.3.1. Pruebas realizadas.

Las pruebas realizadas con el sistema BCI creado, se basan en un paradigma de entrenamiento, que discrimina entre el estado de reposo y la imaginación del movimiento de la mano derecha. Gracias a que, la interfaz gráfica perteneciente al sistema BCI emplea de técnicas de Realidad Virtual, se mejoran las sesiones de entrenamiento para controlar las señales del sujeto, al aumentar el interés y la concentración.

Además, es necesario observar los efectos que tiene esta tecnología en el tiempo de entrenamiento, pues uno de los objetivos que se pretende alcanzar con ella es la reducción de este tiempo, pero con una tasa de aciertos aceptable.

Finalmente, recordar que el feedback es continuo, ya que se proporciona información en tiempo real sobre las acciones que el sujeto está llevando a cabo, de manera que los cambios que provoquen estas acciones en la interfaz, le ayuden a rectificar de manera instantánea en función de lo que observa.

6.3.2. Paradigma de entrenamiento.

El protocolo o paradigma de entrenamiento de las pruebas realizadas tiene la estructura representada en la figura 6.3.

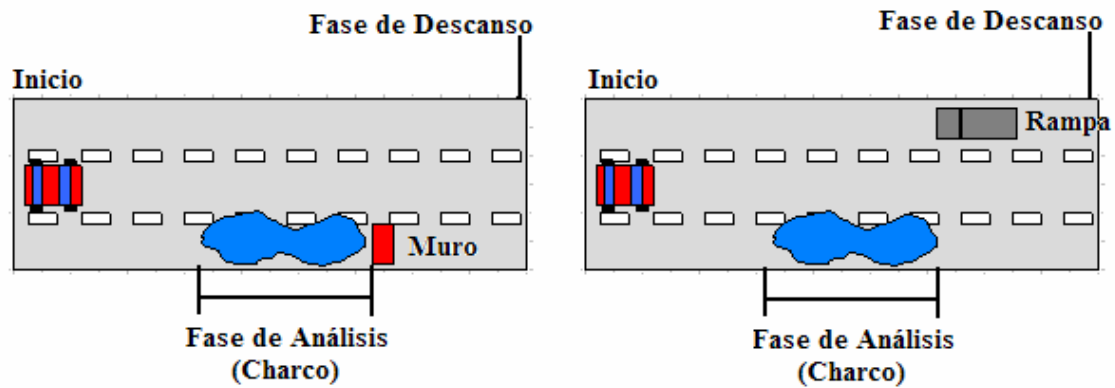


Figura 6.3: Estructura del paradigma de entrenamiento de las pruebas realizadas.

Cada sujeto realizó 4 sesiones. Cada sesión de entrenamiento se compone a su vez, de cuatro ensayos, cada uno organizado en 40 pruebas individuales. El ensayo mostrará el charco a la izquierda en 20 pruebas y a la derecha en otras 20. A su vez, las 20 pruebas se dividirán en 10, que usarán el muro y en otras 10, que usarán la rampa. Cada prueba se sitúa en una carretera rectilínea de tres carriles que se encuentra en movimiento. El sujeto se identificará con un vehículo que aparecerá en el carril central, avanzando a lo largo de dicha carretera. En el instante 2 segundos después de haberse iniciado la prueba, aparece un charco en uno de los carriles laterales. El charco se identifica con la tarea mental que se debe realizar, es decir, pensar en mover la mano derecha, para desplazar el coche a la derecha o mantenerse en reposo, para desplazar el coche al carril izquierdo. En el instante aproximado correspondiente a los 4.25 segundos, comienza la fase de análisis lugar donde el coche habrá alcanzado el charco y es este momento en el que el sujeto debe evitarlo realizando su tarea mental.

La fase de análisis, que dura unos 3.75 segundos, el sujeto debe realizar la tarea mental e intentar mantenerla, lo cual forma parte del entrenamiento que se está llevando a cabo. Si el coche no consigue esquivar el charco, el coche lo pisará. Además justo al final de la fase de análisis aparecerá otro obstáculo como pueden ser el muro o la rampa, que habrá que esquivarlo o saltarlo.

Tras finalizar esta fase, es decir, cuando el coche deja el obstáculo atrás, se inicia la fase de descanso instantáneo en el que se detiene la ejecución durante un periodo establecido, indicando al sujeto que puede descansar. Después de este periodo de tiempo comienza otra prueba igual a la anterior. Cada prueba dura 9 segundos en total.

6.3.3. Porcentajes de error.

Una vez realizado las pruebas a 3 sujetos, se obtuvieron conclusiones a partir de las gráficas construidas con los resultados.

Las gráficas representan los porcentajes de error en la actividad mental ejecutada en cada una de las sesiones. Se considera acierto cuando el sujeto consigue esquivar el obstáculo y fallo o error en caso contrario. Esta medida de éxito se realiza con las muestras de las señales del sujeto, adquiridas cada medio segundo. De hecho, todo este proceso se realiza con una aplicación ya existente.

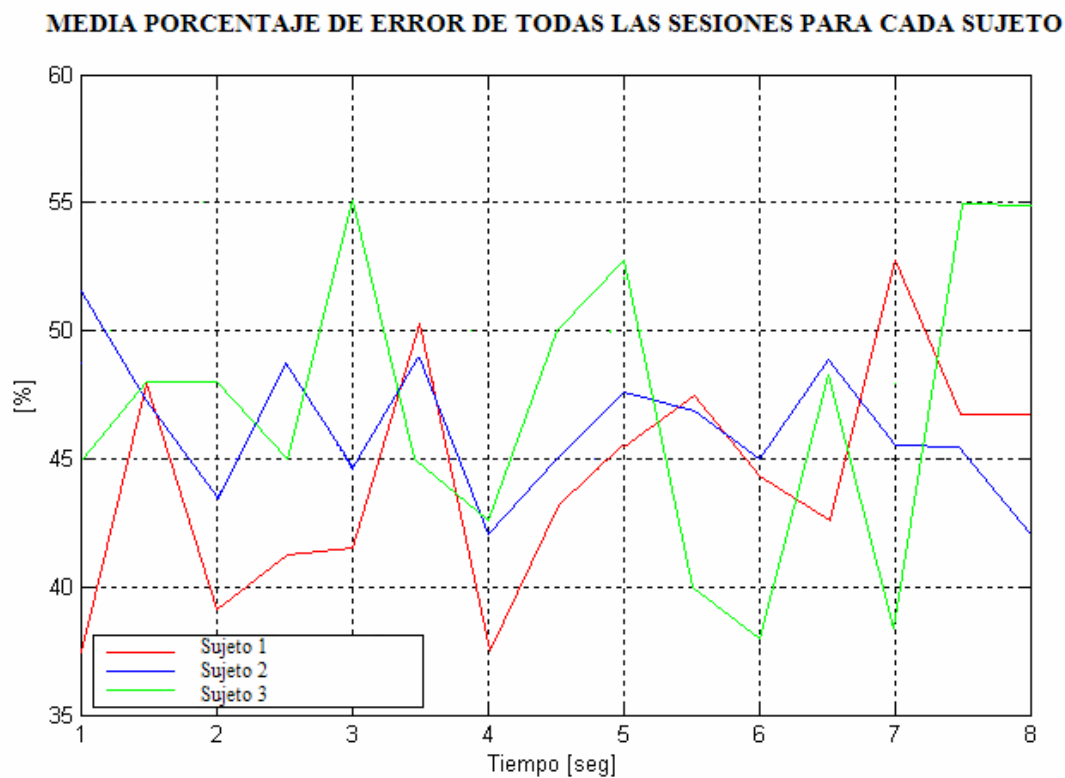


Figura 6.4: Medida de error en todas las sesiones realizadas por cada sujeto.

Estos porcentajes facilitan, en función del tiempo, el promedio de errores y aciertos realizados por el sujeto al realizar la tarea mental, para el mismo instante de tiempo de las 160 pruebas en cada sesión (4 ensayos de 40 pruebas cada uno). En la figura 6.4 puede observarse una de estas gráficas en la que además, se han promediado los resultados de las 4 sesiones para cada sujeto.

Lo ideal es que el porcentaje de error disminuya entre los instantes 4.25 y 8 (fase de análisis), y como se puede observar en la figura 6.4, no es así. De hecho, el error oscila entorno al 45%. El motivo de que el porcentaje de error sea una curva que se mantiene constantemente irregular y sin un decaimiento claro durante el transcurso del tiempo, es que los sujetos bajo estudio no eran muy buenos. Los resultados dependen de algo muy subjetivo como es la capacidad de adaptación del individuo al sistema.

Capítulo 7.

Conclusiones y líneas futuras del sistema BCI.

7.1. Conclusiones.

En el proyecto, como se ha comentando durante toda la memoria se han desarrollado dos aplicaciones: el sistema BCI y el sistema de medida de tiempos de reacción.

En ambos casos, el escenario virtual empleado es el mismo, diferenciándose únicamente en los obstáculos utilizados. De hecho, la interfaz gráfica del sistema de medida de tiempos de reacción es una particularización de la interfaz del sistema BCI, porque únicamente usará el obstáculo muro. Para el desarrollo de estos mundos virtuales se ha empleado el *Virtual Reality Toolbox* de MATLAB.

La interfaz desarrollada en el proyecto, haciendo uso de técnicas de Realidad Virtual, completa el sistema BCI creado por el grupo de investigación DIANA en el Departamento de Tecnología Electrónica.

El proceso de programación ha sido complicado, porque además de cumplir con las especificaciones establecidas en la interfaz, se debe cumplir, o como mínimo no contradecir las especificaciones de los demás bloques ya creados que forman parte del sistema BCI.

Finalmente, se ha logrado uno de los objetivos principales del proyecto al conseguir combinar los distintos escenarios virtuales y adaptarlos a la estructura temporal de las pruebas. Para ello, ha sido necesario realizar un estudio tanto del tiempo empleado por el mundo virtual en cada una de sus actualizaciones como del que se dispone, debido a que un exceso significativo en los límites temporales produce un mal funcionamiento de la herramienta. Por otro lado, se ha tratado de centrar la atención en crear interfaces llamativos que incidieran positivamente en la motivación de los sujetos. Además, la herramienta final resulta flexible y versátil para futuros cambios que se quieran llevar a cabo.

La herramienta es flexible porque permite al operador actuar sobre múltiples parámetros del mundo virtual. Es posible establecer qué obstáculos, de los posibles (*charco*, *charco-muro*, *charco-rampa* o *ninguno*) aparecerán en las distintas pruebas. Por último, es versátil al ser posible cambiar la configuración elegida, cada vez que comience un ensayo diferente en una misma sesión de entrenamiento.

Posteriormente, se realizaron todas las pruebas diferentes que pueden configurarse con la herramienta. Esto proporcionó la información acerca de cómo acelerar y mejorar el entrenamiento, que constituye un objetivo esencial en este momento.

La segunda aplicación, el sistema de medida de tiempos de reacción, permite medir el tiempo que tarda en ejecutar una determinada acción un individuo después de haber recibido un estímulo concreto. El sujeto bajo estudio debe intentar realizar la acción siempre en el menor tiempo posible.

Esta segunda aplicación se ha desarrollado como colaboración con el Dr. Rolando Grave de Peralta Menéndez del Instituto Suizo de Tecnología. El Dr. Rolando Grave pretende crear un sistema que se anticipe a unas determinadas actividades motoras ante unas situaciones concretas, ya que se ha observado que antes de realizar una acción motora de forma consciente, se produce una actividad cerebral previa que conduce al desarrollo de esta acción. El objetivo final busca emplear este tipo de sistemas en los coches. Con la intención de poder ganar tiempo de reacción, de manera que se puedan evitar más accidentes. Por este motivo, se está intentando medir el tiempo transcurrido desde que se produce la actividad cerebral hasta la acción motora resultante.

El sistema de medida de los tiempos de reacción cubre la parte relacionada con la acción motora, debido a que se enviarán eventos a través del puerto paralelo cuando aparezca el muro y cuando se pulse la tecla adecuada para esquivarlo. Los valores enviados por el puerto paralelo van al polígrafo PIOSEMI, que es el encargado de registrar las señales cerebrales.

7.2. Líneas futuras.

La interfaz inicial del sistema BCI no utilizaba la Realidad Virtual, en este proyecto se ha hecho uso de los mundos virtuales para visualizar el biofeedback del usuario.

Además, sería interesante estudiar las señales procedentes de los movimientos oculares, parpadeos y respuestas a estímulos externos (potenciales evocados), ya que afectan de forma negativa al biofeedback. Quizás estas señales podrían emplearse para mejorar el entrenamiento, por ejemplo, indicando al sujeto que está parpadeando.

El empleo de dispositivos de visión estereoscópica no es posible porque el *Virtual Reality Toolbox* no hace uso de esta técnica, por tanto, esta vía de investigación está cerrada en la interfaz desarrollada en el proyecto.

A pesar de la limitación anterior, existen otras líneas de desarrollo que también pueden resultar interesantes, como el reto que supone incluir a varios usuarios en un mismo mundo virtual.

Para poder llevar a cabo actividades en grupo usando Realidad Virtual es necesario aplicar el concepto de Ambientes Virtuales Colaborativos. Con este concepto, la experiencia de un sujeto en una aplicación individual de entrenamiento BCI, se puede ver enriquecida si puede competir con otro usuario. Es decir, la posibilidad de que una competición entre dos sujetos que entrenan el control de sus ritmos cerebrales a través de un sistema BCI, pueda mejorar sus resultados obtenidos en caso de entrenamiento individual.

En la aplicación BCI creada, pueden introducirse una serie de mejoras que repercutirían directamente en el protocolo de entrenamiento. Una posible modificación es la implementación de más obstáculos, consiguiendo un mayor número de configuraciones en las escenas virtuales. Otra idea de interés es el control de la velocidad a la que se desplaza el mundo virtual, de esta manera se podría estudiar la influencia de este parámetro en el manejo de la interfaz gráfica por parte del sujeto bajo estudio. Finalmente, puede resultar adecuado hacer una sola aplicación con las dos versiones implementadas del sistema BCI en este proyecto. Es decir, incluir las marcas representadas por señales de tráfico como una posibilidad dentro de la primera versión.

Es también interesante estudiar como se crean los ejecutables en MATLAB, para que sea más sencilla y cómoda la ejecución tanto del sistema BCI como del sistema de medida de tiempos y así, no tener que depender del código creado ni del programa MATLAB.

Los efectos de sonido de las aplicaciones desarrolladas en el proyecto, no han sido implementados a través de técnicas basadas en Realidad Virtual, porque la ventana de visionado de MATLAB no reproduce sonido. Otra posible línea de desarrollo de la aplicación, es estudiar como visualizar el mundo virtual para que el sonido pueda introducirse mediante técnicas de Realidad Virtual y además, reproducirse.

Para finalizar, podría resultar útil el estudio de la herramienta Simulink relacionada al *Virtual Reality Toolbox*. El objetivo de este estudio sería el de poder desarrollar sistemas similares a los implementados en el proyecto.

Apéndice A.

Manual de Usuario.

En este apéndice se pretende explicar cómo utilizar de manera general las dos aplicaciones implementadas en el proyecto, tanto el sistema BCI con la inclusión de la interfaz gráfica desarrollada como el sistema de medida de tiempos de reacción. En primer lugar, se describirá el manejo del sistema BCI y posteriormente, el referido al sistema de medida de tiempos de reacción. En ambos casos, se detallará el proceso de configuración de la herramienta por el usuario y la manera en que se almacena la información recogida a lo largo de las distintas pruebas.

A.1. Funcionamiento de la herramienta BCI.

Una vez definidos los conceptos utilizados en la BCI se procede a describir la interfaz de usuario que permitirá configurar cada sesión de pruebas.

Todos los cuadros de diálogo están englobados en un panel de control que se encarga de gestionar la introducción de datos. Uno de los cuadros de diálogo permite configurar los obstáculos del mundo virtual que pueden aparecer en cada ensayo (si se ejecuta la primera versión de la aplicación podrán aparecer el charco para la delimitación de la fase de análisis, el muro o la rampa y en el caso de que se ejecute la segunda versión podrán aparecer las marcas representadas por señales de tráfico para delimitar la fase de análisis, la rampa o el muro), el resto de los cuadros de diálogo son los que ya formaban parte de la herramienta que existía en el Departamento.

Al ejecutar el programa *HM_panel_rv.m*, aparece una ventana que constituye el panel de control como muestra la figura A.1, donde aparecerá la totalidad de opciones a especificar por parte del usuario.

Figura A.1: Ventana correspondiente al panel de control de la BCI.

En ella se distinguen cinco secciones claramente diferenciadas: *Datos del Usuario*, *Tipos de Obstáculos*, *Organización del Ensayo*, *Parámetros del Análisis* y *Estructura de la Prueba* y *Botones*. A continuación se pasa a explicar cada sección por separado.

A.1.1. Datos del Usuario.

Permite el establecimiento del nombre del sujeto, nombre del ensayo, fecha y un identificador que ayudará no sólo a identificar al sujeto, sino también a guardar

ordenadamente los resultados del ensayo, ya que se crea una estructura de directorios a partir del directorio en que se encuentren los programas de la Herramienta de Medida, con la siguiente sintaxis:

`\Sesiones\<Identificador del Sujeto>\'Sesion_\'<Fecha>\<Nombre del Ensayo>`

Posteriormente en la sección A.1.2 se hablará un poco más de esta estructura y se pondrá un ejemplo.

Por tanto, cada ensayo tendrá su propio directorio. Si el programa detecta que ya existe ese directorio, simplemente avisa al usuario, y se deja a la libre elección de éste cambiar el nombre o no. Si no se cambia el nombre, el programa borrará ese directorio la primera vez que se pulse el botón de *Iniciar Ensayo*. De todas formas, el programa nada mas abrirse, busca en el fichero de configuración los parámetros que se dejaron la última vez que se usó, entre esos parámetros está el nombre del ensayo, y para evitar que tenga el mismo nombre, se le añade automáticamente un número que se irá incrementando también automáticamente en sucesivas llamadas.

A.1.2. Tipos de Obstáculos.

Ofrece la posibilidad de seleccionar cuáles de los posibles obstáculos que pueden aparecer en la sesión estarán activos. Cualquiera de las combinaciones por la que se opte es válida, siendo también factible la opción de *Ninguno*, lo que significa que no aparecerá ningún obstáculo en el entrenamiento. Recordar que el resto de las alternativas son *charco*, *muro* y *rampa*. En caso de seleccionar varias, el obstáculo que aparecerá en cada prueba se elegirá aleatoriamente. También existe otra versión en la que se puede establecer que la fase de análisis esté delimitada por unas señales de tráfico, en lugar de un charco de agua.

A.1.3. Organización del Ensayo.

En esta sección del panel de control pueden combinarse los distintos tipos de pruebas. Para ello, hay que establecer las pruebas que desean realizarse y la cantidad. El

entrenamiento puede realizarse con dos tipos de pruebas diferentes, con feedback o sin feedback. El feedback que se emplea es de tipo continuo y se caracteriza porque el objeto biofeedback (coche) se desplaza en tiempo real durante todo el tiempo que dura el análisis. Por ejemplo, en la figura A.1 se ha organizado un ensayo de 5 pruebas usando feedback continuo (*Feed_cont*).

A la derecha se observan cinco botones: los cuatro de la parte superior permiten seleccionar automáticamente una configuración preestablecida para el ensayo ('*N-F*', '*F-D*', '*F-C*', '*C-Mx*'), que se consideró útil en la versión antigua de la BCI. Se han dejado por si en algún momento se considera que pueden resultar beneficioso reconfigurarlos.

El botón de abajo (*Crear fichero de info adicional*), permite crear un fichero de texto si se desea explicar algo más acerca de la prueba. Resulta muy útil el que sea de tipo texto, pues al moverse por la estructura de directorios puede saberse rápidamente lo que hay en uno de ellos utilizando el *notepad* de Windows.

A.1.4. Parámetros del Análisis y Estructura de la Prueba.

En esta sección se observan 3 columnas bien diferenciadas:

- La primera incluye los parámetros de muestreo y análisis de la señal, como son la frecuencia de muestreo, el tamaño de la ventana que se quiere usar, el solape de muestras anteriores que quiere añadirse a la ventana, el método de estimación espectral, y el orden del mismo. En este proyecto, el método de estimación espectral empleado es el cálculo de la potencia. Todos estos parámetros son comunes para los cuatro canales que como máximo pueden grabarse.
- En la segunda columna se observa que pueden capturarse como máximo cuatro canales, así como darles nombre (se les ha dado los nombres *C3*, *C4*, *EOG*, *EMG*). Todos estos canales se capturan a la misma frecuencia indicada en la columna anterior. En este proyecto sólo se emplean dos canales, el *C3* y *C4*, tal y como se observa en la figura A.1. Además está el botón que llama

al *Editor de Filtros*, una herramienta que será muy útil para definir los filtros.

- Finalmente, en la columna que queda se piden los tiempos que definen la estructura de la prueba, que ya se vieron anteriormente al definir ‘prueba’ y ‘ensayo’. Así, los valores que se les dan son: instante en que aparece el obstáculo (t. Objetivo) = 2sg., instante en que comienza el análisis (Inicio) = 4.25sg., duración del análisis (Duración) = 3.75sg., duración total de la prueba (Duración de la prueba) = 9sg. El instante t. Cursor que muestra la figura A.1 no se emplea en el sistema BCI desarrollado.

Además hay un tiempo aleatorio de separación entre prueba y prueba, que está formado por una parte fija (1sg) más una parte aleatoria (1sg), ambos indicados en la línea ‘Desc. entre pruebas’.

Se observa que además hay un tiempo de descanso entre ensayos, este tiempo no es utilizado por el programa en sí, sino que sirve más bien para recordar al monitor del experimento que debe hacer un descanso entre ensayo y ensayo.

A.1.5. Botones.

La última sección del panel de control está compuesta por una serie de botones, tales como:

- ‘**Iniciar Ensayo**’, tras ser pulsado una vez, muestra un mensaje de aviso alertando al encargado del experimento de que compruebe que todo está listo para empezar y que vuelva a pulsarlo para confirmar.
- **Ver(t)** y **Ver(f)**, sirven para habilitar o inhabilitar la salida por pantalla de la señal ya sea en el dominio del tiempo o en el de la frecuencia, respectivamente. La BCI desarrollada en este proyecto posee estas dos posibilidades como herencia de la versión anterior, pero no es empleada. Se ha dejado por si en un futuro puede resultar útil.
- ‘**Configuración Avanzada**’, evita el cambio accidental de los parámetros que definen un experimento, es decir, todos aquellos parámetros que deben

permanecer fijos a lo largo de un experimento. Para poder modificarlos habrá que pulsar este botón.

- Finalmente, '**Resetear Parámetros**' borra todos los campos de la primera sección, y modifica el resto de parámetros que hay en las demás secciones en base a una configuración preestablecida. Dicha configuración, se consideró útil en su momento, y por tanto no hay que verla como una configuración ideal. Además, este botón dispone de un seguro (del tipo *radiobutton*) para evitar un reseteo accidental y en consecuencia, borrar la configuración.

A.1.6. Organización y almacenamiento de la información.

A.1.6.1. Estructura de Directorios.

La Herramienta de Medida crea una estructura de directorios para guardar la información recopilada a lo largo de todas las sesiones. Dicha estructura parte siempre del directorio donde se encuentran todos los programas MATLAB que forman la Herramienta. Como se mostró con anterioridad la información almacenada vendrá dada por la siguiente dirección:

`C:\...\HM_graz_coche_obstaculos_definitivo\Sesiones\<Identificador_Sujeto>\'Sesion _'<Fecha>\<NombreEnsayo>`

A continuación, en la figura A.2 se observa un ejemplo de estructura de directorios tras realizar varias sesiones para distintos sujetos:

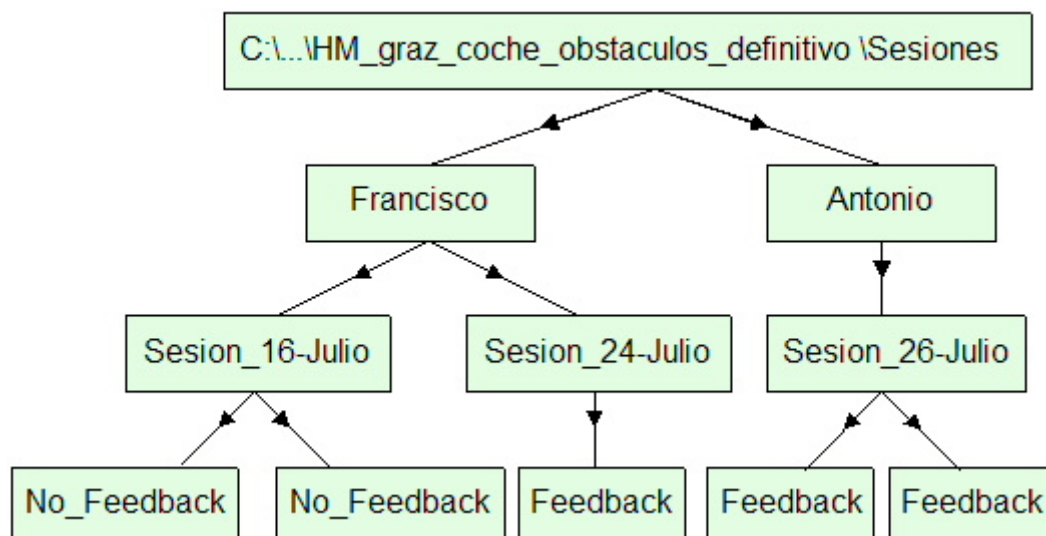


Figura A.2: Ejemplo de estructura de ficheros creada por la Herramienta de Medida.

A.1.6.2. Ficheros guardados.

Son varios los ficheros grabados a lo largo de la ejecución del programa. A continuación, se detallarán las distintas variables almacenadas en los ficheros y sus funciones.

- **ensayo_config.mat:** Contiene, como ya se ha dicho, la configuración que se utilizó por última vez para organizar un ensayo. Se guarda en el mismo directorio en que están los programas que forman la Herramienta de Medida. Contiene un gran número de variables, exponerlas aquí no aportaría nada, ya que casi todas son de uso interno del programa.
- **filtro_ch1.mat, filtro_ch2.mat, filtro_ch3.mat y filtro_ch4.mat:** Contienen los 4 filtros que forman el banco de filtros que se utilizó para cada canal. Dichos filtros se guardan como dos matrices, 'Num' y 'Den' que contienen respectivamente todos los numeradores y denominadores de los 4 filtros. Se guardan en el directorio Sesiones.
- **rtotal.mat:** Este fichero contiene las variables que identifican al sujeto que realiza el ensayo, las variables que definen la estructura temporal de las pruebas y los resultados obtenidos tras la realización del ensayo. Se graba en el directorio Sesiones al igual que filtro_chi. A continuación, se muestra la tabla A.1 que contiene todas las variables almacenadas en este fichero.

Variable	Función
sujeto	Nombre del sujeto bajo prueba
ensayo	Nombre del ensayo que se va a realizar
Fs	Frecuencia de muestreo de las señales EEG
muestratotal1, muestratotal2, muestratotal3, muestratotal4	Se tratan de las muestras totales obtenidas de los canales registrados. El sistema BCI creado en el proyecto sólo usará dos de los cuatro canales disponibles en la BCI de referencia
trigtot	Es una variable que indica el intervalo en el que se realiza la tarea mental. Se establece una traza
tipologia	Indica el número de pruebas que se van a realizar y de qué tipo

tiempototal	Es un vector en el que almacena los instantes temporales en el que se captura la información
posición	Indica si el charco ocupa el carril izquierdo (valor 1) o derecho (valor 2)
t_objetivo	Indica el instante a partir del cual los obstáculos son visibles
t_analisis	Establece el instante en el que se inicia el análisis
d_analisis	Indica la duración del análisis
d_descanso	Tiempo de descanso entre pruebas
dist_barra	Almacena los resultados obtenidos en un ensayo tras el proceso de clasificación de las señales.
wo, w1 y w2	Pesos empleados en el proceso de clasificación.

Tabla A.1: Variables almacenadas en rtotal.

- **sesion.mat:** Almacena en el directorio Sesiones la variable ‘**y**’. Esta variable posee las muestras de los dos canales empleados en los ensayos así como la traza que establece el intervalo en el que se realiza la tarea mental.
- **clase.mat:** Guarda en la carpeta Sesiones la variable ‘**z**’ y en ella se almacena el tipo de prueba realizada, es decir, si los obstáculos aparecen en el carril izquierdo o derecho. La variable **z** es una matriz de dos filas y de número de columnas igual al número de pruebas realizadas en el ensayo.
 - Si el obstáculo aparece en el carril derecho, la variable **z** almacena los valores $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$.
 - Si el obstáculo aparece en el carril izquierdo, la variable **z** almacena los valores $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

A.2. Funcionamiento del sistema de medida de tiempos de reacción.

La herramienta gestiona la introducción de datos a través de una serie de cuadros de diálogo que están englobados en un panel de control.

Al ejecutar el programa *HM_panel_tr.m*, aparece el panel de control como muestra la figura A.3, en él se distinguen cuatro secciones diferenciadas: *Datos del Usuario*, *Pruebas del Ensayo*, *Estructura de la Prueba* y *Botones*. A continuación se pasa a explicar cada sección por separado.

A.3: Panel de control de la herramienta de medida de los tiempos de reacción.

A.2.1. Datos del Usuario.

Permite el establecimiento del nombre del sujeto, nombre del ensayo, fecha y un identificador que ayudará no sólo a identificar al sujeto, sino también a guardar ordenadamente los resultados del ensayo, ya que se crea una estructura de directorios a

partir del directorio en que se encuentren los programas de la Herramienta de Medida, con la siguiente sintaxis:

`\Sesiones\<Identificador del Sujeto>\'Sesion_\'<Fecha>\<Nombre del Ensayo>`

Al ser una aplicación desarrollada a partir del sistema BCI implementado en el proyecto, heredará algunas de sus características como la estructura de directorios generados en cada sesión.

A.2.2. Pruebas del Ensayo.

A diferencia del sistema BCI, el sistema de medida de tiempos de reacción sólo posee un tipo de prueba. Por este motivo, la sección únicamente sirve para establecer la cantidad de pruebas que van a realizarse en el ensayo. Por ejemplo, en la figura A.3 se ha organizado un ensayo de 5 pruebas.

En el caso de que sea necesario explicar algo más acerca de la prueba, al igual que sucedía en el sistema BCI, la sección posee el botón *Crear fichero de info adicional*.

A.2.3. Estructura de la Prueba.

En esta sección se observan una serie de parámetros que permiten determinar la estructura de la prueba:

- Los primeros parámetros que aparecen son la frecuencia de muestreo y el tamaño de la ventana. A pesar de que no se capturan ni se procesan señales, se usan estos parámetros debido a que con ellos se establecen referencias temporales.
- Finalmente, están los parámetros estructura de la prueba, que ya se vieron anteriormente al definir ‘prueba’ y ‘ensayo’. Así, los valores que se les dan son: instante en que aparece el muro antes de la posible colisión (t_{muro1}) = 0.5sg. y (t_{muro2}) = 1sg., instante máximo que tardará el muro en estar a la altura del coche (t_{colision}) = 5sg., duración total de la prueba (Duración de la prueba) = 6sg.

Además hay un tiempo aleatorio de separación entre pruebas, que está formado por una parte fija (0sg) más una parte aleatoria (0sg), ambos indicados en la línea ‘Desc. entre pruebas’. En el ejemplo, se ha eliminado este periodo temporal.

Se observa que además hay un tiempo de descanso entre ensayos, este tiempo no es utilizado por el programa en sí, sino que sirve más bien para recordar que se debe hacer un descanso entre ensayos.

A.2.4. Botones.

Está última sección del panel de control está compuesta una serie de botones, que también han sido empleados en el sistema BCI. De hecho, cumplirán las mismas funciones en ambos sistemas.

- ‘**Iniciar Ensayo**’, se emplea para confirmar que todos los elementos del experimento son correctos y para comenzar el ensayo que se desea realizar. Este botón debe pulsarse dos veces, la primera para comprobar que todo es correcto y la segunda para iniciar el ensayo.
- ‘**Configuración Avanzada**’, se usa para evitar el cambio accidental de los parámetros que definen el ensayo. En el caso de que se desee modificarlos habrá que pulsar este botón.
- Finalmente, ‘**Resetear Parámetros**’ borra todos los campos de la primera sección, y modifica el resto de parámetros que hay en las demás secciones en base a una configuración preestablecida. Además, este botón dispone de un seguro (del tipo *radiobutton*) para evitar un reseteo accidental y en consecuencia, borrar la configuración.

A.2.5. Organización y almacenamiento de la información.

A.2.5.1. Estructura de Directorios.

La organización de la información en esta segunda herramienta sigue la misma estructura de directorios creada en el sistema BCI. La información vendrá almacenada por la siguiente dirección:

`C:\...\HM_graz_coche_muro_sintarjeta\Sesiones\<Identificador_Sujeto>\'Sesion_\'<Fecha>\<NombreEnsayo>`

En este caso se ha tomado la versión que no utiliza la tarjeta de adquisición.

A continuación, en la figura A.4 se observa un ejemplo de estructura de directorios tras realizar varias sesiones para distintos sujetos:

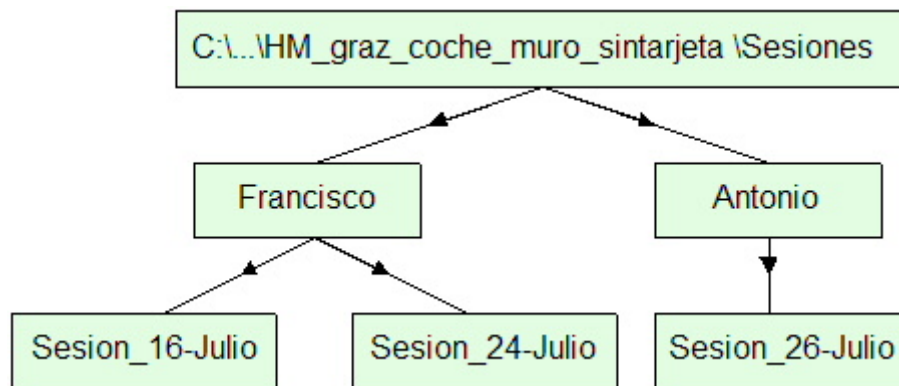


Figura A.4: Ejemplo de estructura de ficheros creada por la Herramienta de Medida.

A.2.5.2. Ficheros guardados.

Los ficheros que son grabados a lo largo de la ejecución del programa, se verán a continuación y además, se detallarán las distintas variables que almacena cada uno de ellos y su función.

- **ensayo_config.mat:** Contiene al igual que el sistema BCI, la configuración que se utilizó por última vez para organizar un ensayo. Se guarda en el mismo directorio en que están los programas que forman la Herramienta de Medida.
- **clase_muro.mat:** Guarda en la carpeta Sesiones las variables '**muro_der_izq**' y '**t_muro**'.
 - o **muro_der_izq:** Almacena la posición del muro en cada prueba realizada durante un ensayo.
 - Si aparece a la derecha de la carretera, la variable tomará el valor 0.
 - En cambio, si aparece a la izquierda, la variable tomará el valor 1.

- **t_muro:** Almacena que variable se ha empleado de t_muro1 y t_muro2 en cada prueba realizada en un ensayo.
 - Si se utiliza t_muro1, la variable tomará el valor 0.
 - Si se utiliza t_muro2, la variable tomará el valor 1.
- **clase_teclado.mat:** Guarda en la carpeta Sesiones las variables ‘pulsacion’ y ‘dif_tiempo’.
 - **pulsación:** Almacena la tecla pulsada en cada prueba.
 - Si se pulsa el cursor derecho, el valor que tomará la variable es 0.
 - Si se pulsa el cursor izquierdo, el valor que tomará la variable es 1.
 - Finalmente, si no se pulsa ninguna tecla, el valor que tomará la variable es 2.
 - **dif_tiempo:** Almacena la diferencia de tiempo que hay desde que aparece el obstáculo muro y se pulsa una tecla en cada prueba. Es decir, el tiempo de reacción.

Finalmente, es importante comentar que se envían unos valores por el puerto paralelo en cada prueba cuando suceden unos eventos y para ello, se usarán una serie pines del puerto tal y como se muestra en la figura A.5:

- **Pin 2:** Se le asignará el valor 1, si se emplea t_muro1 y 0 si no se usa.
- **Pin 3:** Se le asignará el valor 1, si se emplea t_muro2 y 0 si no se usa.
- **Pin 4:** Se le asignará el valor 1, si se pulsa el cursor derecho y 0 si no se pulsa.
- **Pin 5:** Se le asignará el valor 1, si se pulsa el cursor izquierdo y 0 si no se pulsa.

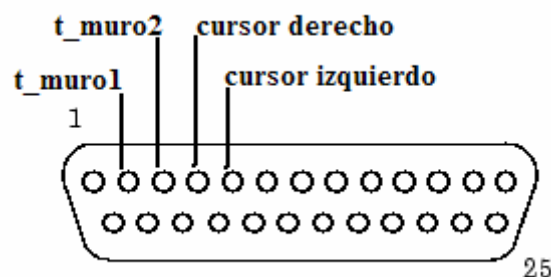


Figura A.5: Pines del puerto paralelo empleados en la herramienta.

Bibliografía.

- [1] La Realidad Virtual. Está disponible en <http://www.monografias.com/>.
- [2] *Manual VRML97. Internacional Standard ISO/IEC 14772-1:1997*. The VRML Consortium Incorporated. Disponible en <http://tecfa.unige.ch/guides/vrml/vrml97/spec/>.
- [3] *La guía VRML de Floppy*. Vapour Technology Ltd. 1998-2002. Está disponible en http://web3d.vapourtech.com/tutorials/vrml97/index_es.html.
- [4] *Tutorial VRML97*. Narcís Parés i Burgués. 2000. Está disponible en <http://www.iaa.upf.es/~npares/docencia/vrml/tutorial.htm>.
- [5] Parra Márquez, Juan Carlos; García Alvarado, Rodrigo; Santalices Malfanti, Iván. *Introducción Práctica a la Realidad Virtual*. Ediciones U. Bio-Bio, Concepción, 2001. Está disponible en <http://zeus.dci.ubiobio.cl/~sigradi/libros/>.
- [6] De la Torre Peláez, J.: *Estudio de la Viabilidad del Reconocimiento de Tareas Mentales a través del Análisis del Electroencefalograma*, Proyecto Fin de Carrera, Departamento de Tecnología Electrónica, Universidad de Málaga, 2002.
- [7] García-Malea López, M.: *Desarrollo de mundos virtuales aplicados al entrenamiento de interfaces cerebro-computadora*, Proyecto Fin de Carrera, Departamento de Tecnología Electrónica, Universidad de Málaga, 2004.

[8] *MATLAB and Simulink for Technical Computing*. 1994-2007. The Mathworks Inc. Está disponible en www.mathworks.com.

[9] *Virtual Reality Toolbox For Use with MATLAB and Simulink. User's Guide*. Version 4. 2001-2006 by Mathworks.

[10] *V-Realm Builder. User's Guide and Reference*. 1996-1997 Ligos Corporation. California, USA.

[11] *MATLAB and Simulink. Student Version. Learning MATLAB 7*. 1984-2005 by The Mathworks inc.

[12] *Características del Puerto Paralelo. Conector DB25M*. Está disponible en www.zator.com/Hardware/H2_5_2.htm.