Índice General

[1 INTEGRACIÓN CON LA INTERFAZ DE NAVEGACIÓN EXISTENTE 2](#_Toc242720110)

[1.1 Adecuación de los mundos modelados 2](#_Toc242720111)

[1.1.1 Colisiones: sensores de proximidad 2](#_Toc242720112)

[1.1.2 Conversión a VRML 97 4](#_Toc242720113)

[1.2 Descripción del Interfaz gráfico de navegación 7](#_Toc242720114)

[1.3 Diagrama de funcionamiento del sistema BCI 9](#_Toc242720115)

[1.3.1 Adquisición de los datos 10](#_Toc242720116)

[1.3.2 Procesado de los datos 10](#_Toc242720117)

[1.3.3 Clasificador 12](#_Toc242720118)

[1.3.4 Realimentación o biofeedback al sujeto. 12](#_Toc242720119)

[1.4 Integración de Vivienda y Escuela (ETSIT) 12](#_Toc242720120)

[1.5 Integración de Simulador de Vuelo 14](#_Toc242720121)

[2 PRUEBAS Y EVALUACIÓN 18](#_Toc242720122)

Índice de Figuras

[Figura 1‑1: Vistas Left, Front, Top y Perspective en la creación de un sensor. 4](#_Toc242720097)

[Figura 1‑2: Cuadro de diálogo VRML 97 Exporter 5](#_Toc242720098)

[Figura 1‑3: Interfaz NC de selección de comando 8](#_Toc242720099)

[Figura 1‑4: Interfaz CI de selección de comando 9](#_Toc242720100)

[Figura 1‑5: Interfaz de navegación completa 9](#_Toc242720101)

[Figura 1‑6: Diagrama funcional del sistema BCI 10](#_Toc242720102)

[Figura 1‑7: Superposición de secuencias analizadas 11](#_Toc242720103)

[Figura 1‑8: Interfaz NC en la Vivienda 13](#_Toc242720104)

[Figura 1‑9: Interfaz CI en la Escuela 13](#_Toc242720105)

[Figura 1‑10: Interfaz gráfico (VRML) de navegación 15](#_Toc242720106)

[Figura 1‑11: Integración. Visualización desde el punto de vista 15](#_Toc242720107)

[Figura 1‑13: Giro a la derecha del avión 17](#_Toc242720108)

[Figura 1‑14: Descenso del avión 18](#_Toc242720109)

# INTEGRACIÓN CON LA INTERFAZ DE NAVEGACIÓN EXISTENTE

## Adecuación de los mundos modelados

Antes de realizar la integración, es necesario adaptar algunas de las características de los mundos virtuales desarrollados para adecuar y facilitar la integración con la interfaz de navegación existente. El objetivo es garantizar el correcto funcionamiento y respuesta del mundo virtual tras la elección de un movimiento seleccionado en la Interfaz de navegación.

### Colisiones: sensores de proximidad

En los mundos virtuales de la Vivienda y de la Escuela hay que dotar a los objetos que los fabrican y decoran de la capacidad de producir eventos al aproximarse o colisionar con ellos.

Cuando un ambiente virtual contiene varios objetos con los que se debe interactuar, la detección de colisiones es uno de los problemas fundamentales, ya que si no se presta atención especial a la intersección entre los objetos, se podrían originar estados no deseados entre ellos.

Todos los elementos de los mundo virtuales modelados deben de poder interactuar con el usuario observador. El observador es capaz de introducirse en el mundo virtual a través de un objeto cámara o punto de vista (ViewPoint) a través del cual percibe la escena en la que se encuentra y éste debe ser capaz de interactuar con todos los elementos que se encuentra a su paso, desde las propias paredes que limitan la estructura básica hasta los elementos de decoración que se pueden interponer en su camino y con los puede colisionar.

Se deben evitar así situaciones inverosímiles en una situación real como los casos en los que el observador pueda por ejemplo atravesar paredes o situarse físicamente en el mismo espacio geométrico que ya esté ocupado por un elemento de decoración, columnas o incluso otros observadores.

De esta forma también se delimita el itinerario de navegación que se puede realizar a lo largo del mundo virtual, se puede prohibir por ejemplo entrar en una habitación o recinto o transcurrir por determinados lugares de la escena.

Para dotar a los objetos de un mundo virtual de estas características se utilizan los sensores de proximidad o *ProximitySensor*.

Estos sensores definen una región del espacio, normalmente en forma de caja, que permite detectar cuando un usuario ha entrado, salido o se mueve alrededor del interior de ella, reportando la localización y orientación del usuario dentro de la región.

Por tanto es necesario tanto para el mundo virtual de la Vivienda como de la Escuela rodear cada uno de los elementos que las componen de una región rectangular en forma de caja que disparen eventos de colisión cuando el punto de vista del usuario observador penetre en ellos. Esta regiones rectangulares se pueden fabricar directamente utilizando 3DStudio, como hemos hecho durante todo el modelado de los mundos de realidad virtual elaborados, y al final es tan sencillol como crear cajas con las dimensiones oportunas que envuelvan las geometrías con las cuales se quiere detectar la colisión.

Estas cajas se modelan con el objeto de 3DStudio *ProxSensor* ubicado en el menú de creación *Create/Helpers/VRML97* y tras la exportación a VRML se realiza su equivalencia en nodos *ProximitySensor.*

Como cualquier caja los parámetros que pueden ser configurados son su centro y dimensiones (alto, largo y ancho) además de un flag “*Enable*” que indica si el sensor esta activo o desactivo.

En el siguiente ejemplo se muestra como se modela, dimensiona y ubica un sensor de proximidad con 3DStudio para el elemento de decoración que da forma los tablones de notas en el mundo virtual de la Escuela de Telecomunicación, procedimiento que debe seguirse para cada uno de los elementos tanto de este mundo como para los de la Vivienda.

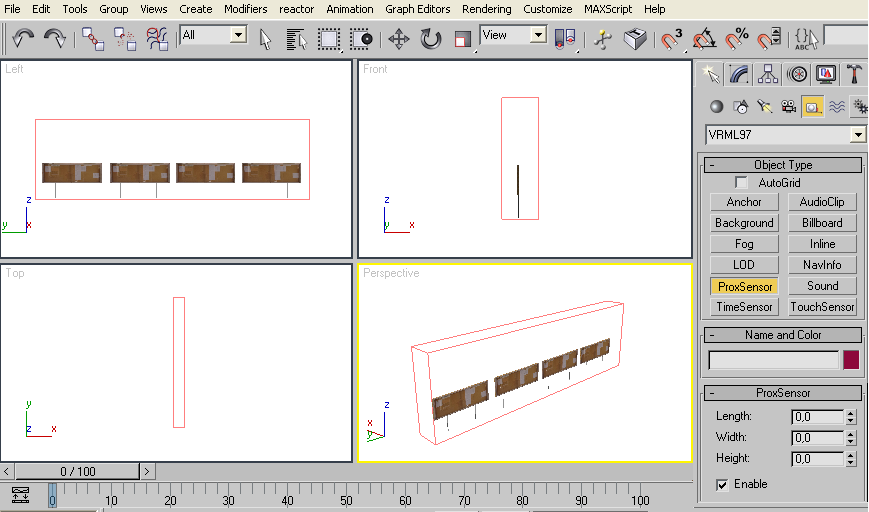


Figura ‑: Vistas Left, Front, Top y Perspective en la creación de un sensor.

Y la traducción del sensor anterior a VRML sería la siguiente:

DEF ProxSensorTablones Transform {

translation -9.53 0.005 61.4

children [

DEF ProxSensorTablones ProximitySensor {

enabled TRUE

center 0 2.86 0

size 0.747 5.72 8.56

}

]

}

### Conversión a VRML 97

El interfaz existente que hay que integrar para realizar la navegación en el interior de los mundos está desarrollado en VRML, Además el procesado de los procesos mentales, tras la adquisición de las señales EEG, se realiza utilizando el Toolbox de realidad virtual de Matlab, que es capaz de interactuar y dar órdenes a entornos virtuales implementados en VRML, por tanto es necesario realizar la conversión de los mundos virtuales a este leguaje.

Desde la versión 5, 3DStudio crea ficheros VRML (con extensión “.wrl”) que son compatibles con el estándar VRML97.

Cuando se realiza la exportación de un entorno virtual desarrollado en 3DStudio al lenguaje propio de VRML, se tiene la oportunidad de configurar un número importante de parámetros en el cuadro de diálogo *VRML 97 Exporter* que se encuentra en el menú *File/Export* de este programa y que se muestra a continuación:

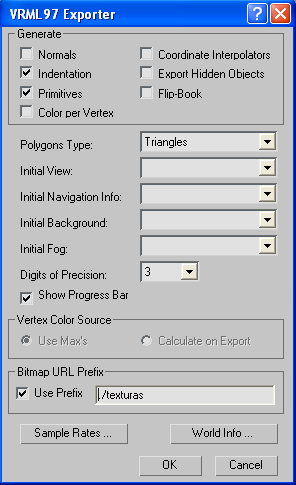


Figura ‑: Cuadro de diálogo VRML 97 Exporter

Los parámetros que se pueden configurar a la hora de realizar la exportación son:

* *Normals*: crea normales reales, creando fichero VRML más grandes. Esta casilla se selecciona si la geometría usada en el modelado necesita de suavizado o “*smoothing*”, y los visores de VRML utilizados para visualizar el mundo requieren de estas normales para la correcta visualización.

Los mundos virtuales del la Vivienda y de la Escuela no tienen geometrías que necesiten de esta propiedad, además se comprueba que con los visores VRML utilizados, es decir, Cortona 3D Viewer y Orbisnap (visor VRML de Matlab), se obtienen resultados muy óptimos sin marcar esta opción.

* *Indentation*: realiza el sangrado apropiado del fichero VRML para una mejor lectura del mismo.

Los ficheros VRML resultados de la exportación son ficheros de texto plano. Si no se realiza un sangrado adecuado del código resultado, es prácticamente imposible distinguir los objetos y parámetros de cada objeto en el fichero WRL, por tanto se marca esta opción para obtener un código más legible.

* *Primitives*: exporta las primitivas de VRML en lugar de las primitivas de 3DStudio. Se consiguen ficheros más cortos.

Es importante que el fichero resultado de la exportación tenga el menor contenido posible. De esta manera al visor VRML le será más fácil procesar el código y menor uso de CPU será necesario para visualizar el mundo virtual en tiempo real.

* *Color per Vertex*: exporta el color de los vértices de la geometría.

No es de especial relevancia que los vértices de las geometrías que componen el mundo virtual se exporten con un color concreto, ni con mucha precisión. Lo que sí interesa es que las caras de los polígonos que limitan esos vértices sí se exporten con el color correctamente.

* *Coordinate Interpolators*: habilita exportar animaciones que utilizan interpolación de coordenadas, no sólo las transformaciones simples de movimiento, rotación y escalado habituales. Se crean ficheros VRML más extensos.

Para los mundos virtuales creados donde la animación y movimiento de la escena se produce en fases posteriores a la exportación no es necesaria.

* *Export Hidden Objects*: si se marca la exportación incluye objetos que se hayan escondido, y serán visibles en VRML. Normalmente se deja desmarcada.
* *Flip-Book*: exporta la escena en múltiples ficheros como una tasa de muestreo especificada. Se utiliza para escenas con animaciones.
* *Polygon Types*: determina cómo son traducidas las caras de la geometría como nodos “*IndexedFaceSet*” de VRML. El tipo por defecto es “*Triangles*”, que traduce las caras como composiciones triangulares. “*Ngons*” traduce tantas caras como sea posible. “*Visible edges*” traduce, únicamente, las caras que son visibles.

A mayor complicación en la segmentación de las caras de la geometría, mayor es el tamaño del fichero resultado. Ninguno de los mundos elaborados tiene formas complejas y se ha selecciona el tipo “Triangles” tal y como viene por defecto.

* *Initial View*: lista todas las cámaras o puntos de vista de la escena y pone en primer plano la vista seleccionada inicialmente. Determina qué vista es la primera en experimentar el usuario.

En los mundos virtuales elaborados, previamente a la integración con el interfaz de navegación, no existe un punto de vista o cámara por lo que no tenemos ninguno que seleccionar. No obstante en las pruebas realizadas se sitúa un punto de vista de ensayo en ciertas ubicaciones desde las cuales un usuario observador pueda visualizar la escena. Así durante los ensayos realizados, cara a mostrar el efecto conseguido, se selecciona alguno de estos punto de vista como punto de vista inicial.

* *Digits of Precision*: Número de dígitos decimales usados para calcular las dimensiones. Por defecto es 4. Un número menos da como resultado un fichero VRML de menor tamaño, pero un número mayor puede ser necesario si partes de la escena se han situado 100.000 unidades, o más, alejadas del centro de la escena.

En ninguno de los mundos diseñados existen dimensiones superiores a las 500 unidades de separación de las geometrías al centro de coordenadas de la escena, tanto es así que es suficiente utilizar 3 dígitos de precisión para este parámetro con el fin de obtener un fichero wrl de pequeño tamaño.

* *Bitmap URL prefix*: especifica la localización de las texturas asignadas a los objetos de la escena. Si no se selecciona, los ficheros de imagen deben situarse en la misma ubicación del fichero VRML.

En este parámetro sólo hay que tener cuidado en colocar las imagen de textura que se han aplicado al mundo virtual en el directorio que se determine en este parámetro, para que al visualizar el mundo en el visor de VRML sepa dónde buscarlas.

El resto de parámetros no son de demasiado interés y no se detallan en este proyecto.

El resultado de este proceso es un fichero, con extensión “wrl”, de texto plano en el que es fácilmente distinguible cada uno de los nodos VRML que forman parte del mundo virtual. La correspondencia de un elemento en 3DStudio y su equivalente en un nodo VRML es usualmente 1 a 1, aunque puede que existan agrupaciones de elementos en 3DStudio que al ser exportados a VRML se obtengan en nodos independientes, o viceversa.

## Descripción del Interfaz gráfico de navegación

El sistema de navegación, con el que hay que realizar la integración, distingue entre dos estados en los que los sujetos pueden ir cambiando voluntariamente.

* Estado de No Control (NC): los sujetos pueden estar involucrados a una actividad mental diferente para controlar el BCI.
* Estado de Control Intencionado (CI): en el que los sujetos pueden controlar el sistema a través de tareas específicas.

El sistema espera en el estado NC hasta que el sujeto voluntariamente entra en el estado CI. En cada estado, los sujetos visualizan una interfaz gráfica con la posibilidad es escoger uno de los tres comandos de navegación: avanzar adelante, girar a la derecha y girar a la izquierda. Después de que un comando haya sido seleccionado el entorno grafico (mundo virtual) responde con la acción apropiada y vuelve al estado NC (No Control). El procedimiento para elegir un comando es el siguiente:

1. El sistema espera en el estado NC en el que una primera interfaz gráfica, Figura 1‑3, es mostrada al sujeto para recordarse que se encuentra en este estado. La interfaz NC consiste en una barra vertical azul semitransparente situada en el centro de la escena virtual. El largo de la barra es calculado cada 62,5 ms como resultado de la clasificación LDA: si el clasificador determina que la tarea mental imaginar el movimiento de la mano derecha, la barra aumenta; por el contrario (estado relajado), la barra mengua hasta su tamaño mínimo. Cuando la longitud de la barra sobrepasa un umbral impuesto durante un tiempo determinado el sistema cambia al estado CI. Una vez que el umbral es excedido, la barra cambia inmediatamente al color amarillo, y después, progresivamente al rojo mientras la longitud se mantenga por encima del umbral, hasta que el tiempo de selección termine. Si la longitud de la barra es temporalmente inferior al umbral, el color de la barra no cambia mientras no tarde más de un tiempo de reseteo en volver a exceder el umbral; en caso contrario el color vuelve a ser azul otra vez, y el tiempo de selección se resetea. El color de la barra representa otro tipo de feedback que permite a los sujetos conocer cuánto tiempo (aproximadamente) necesitan mantener su tarea mental para con conseguir su elección.

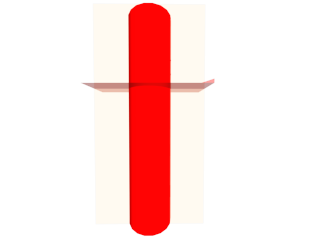
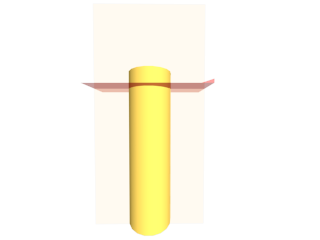


Figura ‑: Interfaz NC de selección de comando

1. El interfaz presentado cuando el sujeto se encuentra en el estado CI, Figura 1‑4, se basa en un circulo dividido en tres regiones, cada una corresponde a los comandos de navegación posibles (avanzar adelante, girar derecha y girar izquierda), con una barra delgada situada en el centro del circulo y recorriéndolo a modo de manecilla de un reloj, de forma que la barra se desplaza para poder seleccionar uno de los comandos que señale en cada momento. La forma de selección de esta barra funciona de la misma forma descrita en el punto 1 (interfaz NC), con el mismo tiempo de selección, reset y nivel del umbral y colores.

Sin embargo se define un nuevo nivel de umbral, o umbral de stop. Cuando la barra lo excede, detiene su rotación para facilitar al sujeto la selección del comando actual. La velocidad de rotación se fija a 2.5 grados por cada iteración del cálculo, por tanto se tardan 9 segundos por vuelta si en ningún momento se sobrepasa el umbral de stop.

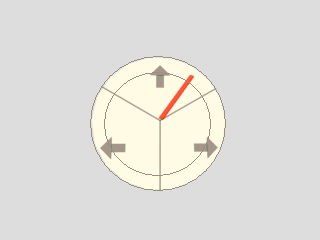
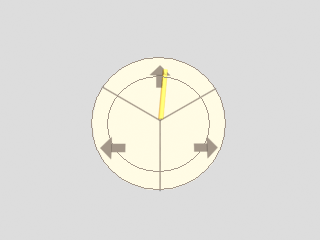
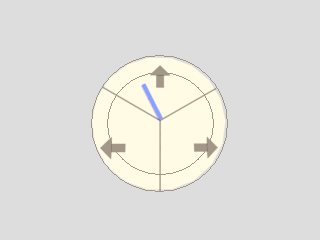


Figura ‑: Interfaz CI de selección de comando

Cuando un comando es seleccionado, el sujeto (usuario y silla de ruedas) se mueve en el interior del mundo virtual, respondiendo éste a las órdenes del comando y visualizándose el movimiento efectuado en el entorno virtual:

* + girar 90 grados a la derecha o izquierda,
  + avanzar hacia delante durante una distancia determinada (1 metro) o hasta que colisiona con un obstáculo del mundo,

después el sistema vuelve al estado NC.

La interfaz de navegación completa, Figura 1‑3, está compuesta por el mecanismo de selección de comandos descrito anteriormente, la silla de ruedas y el punto de vista o cámara a través del cual el sujeto observa el mundo.

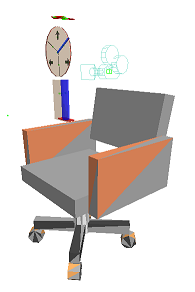


Figura ‑: Interfaz de navegación completa

## Diagrama de funcionamiento del sistema BCI

El funcionamiento del sistema BCI completo se puede resumir en el diagrama siguiente.

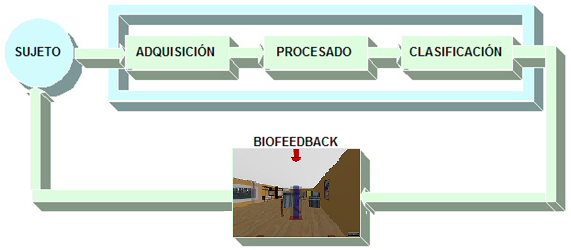


Figura ‑: Diagrama funcional del sistema BCI

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

### Adquisición de los datos

Las señales electroencefalográficas son obtenidas a través de dos canales bipolares de forma que los electrodos activos son situados a 2.5 cm, anterior y posterior, de la posición de los electrodos C3 y C4 (área sensor motora de la mano izquierda y derecha) correspondientes al sistema internacional 10/20. El electrodo masa es situado en la posición FPz.

Después las señales son necesariamente amplificadas por un amplificador de cuatro canales Coulborn V75-08 para garantizar un nivel de señal adecuado para su estudio.

Estas señales analógicas son finalmente digitalizadas a 128 Hz con una tarjeta DAQCard-6025E (National Instruments) de 12 bits de resolución, con el objetivo de realizar su procesado y clasificación utilizando un sistema digital.

### Procesado de los datos

La aplicación realiza distintos procesos sobre cada una de las señales EEG: *enventanado, prefiltrado*, *filtrado*, *enventanado postfiltrado* y *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. 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 1‑4 muestra gráficamente el enventanado de las señales adquiridas.



Figura ‑: Superposición de secuencias analizadas

Para finalizar, comentar que se capturará una nueva ventana de muestras cada 31.25 ms. 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 () se debe realizar todo el procesado de datos y la posterior clasificación de los valores obtenidos.

### 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 el cambio de tamaño de la barra indicadora del proceso mental que se ha descrito en 1.2 Descripción del Interfaz gráfico de navegación.

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



donde:

* pot1 y pot2: 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.
* w0, w1 y w2: son constantes que actúan a modo de pesos, proporcionados por el clasificador tras una fase de entrenamiento del sujeto.

### Realimentación o biofeedback al sujeto.

El bloque de realimentación se ha completado en este proyecto con el desarrollo de mundo de Realidad Virtual, así como la integración con el interfaz de selección de comandos o navegación. El resto de bloques, es decir, adquisición, procesado y clasificación, fueron realizados por proyectos fin de carrera o tesis anteriores dentro del Departamento de Tecnología Electrónica.

La respuesta gráfica del sistema selector de comandos descrito en el apartado anterior, junto con los movimientos que deben producirse en el mundo virtual como consecuencia de cada comando, componen el elemento feedback completo que realimenta e informa al usuario tras la evaluación del sistema BCI.

## Integración de Vivienda y Escuela (ETSIT)

La integración con el interfaz, en los mundos virtuales de la Vivienda y de la Escuela, necesita incluir el nodo VRML “silla” como un elemento más del mundo virtual.

Los movimientos en el interior de la escena se delegan al sistema con el que se realiza la integración, que finalmente es capaz de proporcionar órdenes a la “silla”. Es decir, en esos casos, la experiencia de navegación a través de la escena virtual, en el caso de la Vivienda Virtual o la ETSIT Virtual, se realiza aportando movimiento al nodo “Silla” que realiza las funciones de elemento *feedback* obedeciendo a las ordenes del sistema BCI. La visualización que del mundo virtual tiene el sujeto en cada momento es posible gracias a que con la “silla” se mueve el punto de vista, o cámara, desde donde el sujeto percibe el mundo virtual. Los elementos 3D que componen el mundo virtual, tanto en el caso de la Vivienda como en el de la Escuela, permanecen siempre inmóviles y es el punto de vista, adosado al nodo VRML “Silla” el que emula la sensación de movimiento en la escena.

En las figuras siguientes y podemos ver el sistema de realimentación o feedback completo que visualiza el usuario en cada uno de los estados de decisión descritos en el apartado en el apartado 1.2.



Figura ‑: Interfaz NC en la Vivienda



Figura ‑: Interfaz CI en la Escuela

Se observa en las imágenes anteriores, en la parte inferior de cada una de ellas, como aparece visualizada parte de la silla de ruedas, más exactamente los brazos de la silla.

En cuanto a la detección de colisiones, en cada movimiento que realiza la silla en el interior del mundo virtual se comprueba la posición de la misma. Si la “silla” se encuentra dentro de una zona delimitada por alguno de los sensores de proximidad, implementados en el apartado 1.1.1, se detecta una colisión y el mundo virtual responde deteniéndose frente al objeto colisionado.

## Integración de Simulador de Vuelo

Al igual que se ha realizado con los entornos virtuales de la Vivienda y de la Escuela, el Simulador de Vuelo debe ser integrado con el sistema interfaz de navegación.

En el caso del Simulador de Vuelo, la integración con el interfaz de navegación es algo más compleja, ya que son los propios elementos que modelan el Simulador de Vuelo, y no el nodo VRML de navegación, los que hacen que la escena tome vida y movimiento.

El proceso de integración de la Interfaz de navegación en el mundo virtual del Simulador de vuelo consta de dos pasos:

1. Importación de la interfaz gráfica (nodo VRML) al mundo virtual.
2. Implementación de algoritmos y métodos en código Matlab, para realizar los movimientos sobre los elementos 3D que componen el Simulador de Vuelo, es decir, sobre el modelo del avión y el plano de vuelo.

**INTEGRACIÓN DE LA INTERFAZ GRÁFICA DE NAVEGACIÓN**

En el proceso de selección de un comando es idéntico al descrito en el apartado 1.2, aunque la interfaz gráfica de navegación, que se muestra en la Figura 1‑6, es algo distinta. Consta de tres elementos que se agrupan bajo un mismo nodo VRML:

* Punto de vista o cámara a través de la cual el usuario percibe el mundo virtual.
* Interfaz gráfico para el estado NC (No control).
* Interfaz gráfico para el estado CI (Control Intencionado). Esta interfaz presenta igualmente una rueda de comandos, pero esta vez con cuatro opciones:
  + Ascensos: el avión toma altura sobre el relieve de la ciudad.
  + Giros a la derecha del avión.
  + Descensos: O lo que es lo mismo, pérdida de altura de vuelo.
  + Giros a la izquierda del avión.

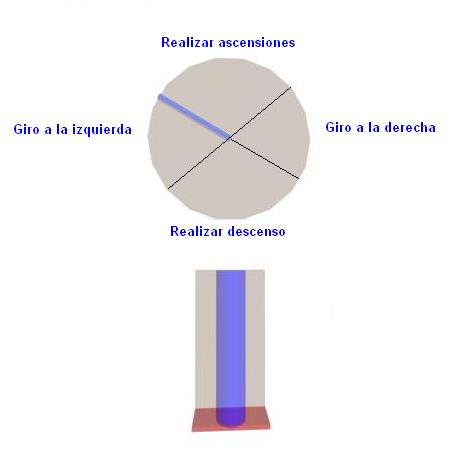


Figura ‑: Interfaz gráfico (VRML) de navegación

Una vez que disponemos del mundo virtual del Simulador de Vuelo bajo el lenguaje VRML, para integrar el interfaz de navegación es necesario incluir el nodo VRML que modela el interfaz de navegación. Esto se lleva a cabo editando el fichero VRML del Simulador de Vuelo en un editor de texto cualquiera e incluyendo el nodo que diseña el interfaz de navegación.

El resultado de la integración visualizado desde el punto de vista desde donde sujeto percibe el mundo virtual es el siguiente.



Figura ‑: Integración. Visualización desde el punto de vista

**IMPLEMENTACIÓN DEL MOVIMIENTO**

A través del interfaz se escoge una de las cuatro opciones de navegación que deben traducirse a los correspondientes desplazamientos de los elementos del mundo virtual para producir el movimiento de la escena.

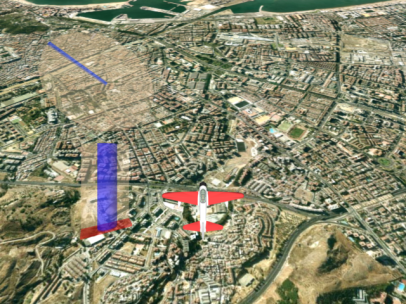
En el caso del Simulador de Vuelo al interpretar un comando de navegación, el movimiento de la escena no se consigue simulando los desplazamientos sobre el punto de vista, sino que son los propios elementos de la escena los que se mueven, mientras el punto de vista debe permanecer en todo momento inmóvil. De esta manera se describe a continuación cómo se ha implementado cada una de las cuatro opciones de navegación que soporta la interfaz con el que se ha realizado la integración.

Antes de nada hay que tener en cuenta que el avión nunca puede estar parado en el aire ya que en caso contrario la tripulación sufriría un grave accidente. Esta circunstancia obliga a que, aunque no se haya tomado alguna de las opciones de navegación, el avión debe seguir avanzando, sobrevolando el terreno en la dirección actual y en línea recta. Lo primero que se debe explicar es cómo se recrea esta situación con los elementos de que disponemos.

Como se explicó en el epígrafe 5.4.2.2 Texturización del plano de vuelo, la recreación de la sensación del avión sobrevolando el terreno, no se realiza mediante un desplazamiento de avance del modelo 3D del avión sobre el terreno, ni desplazamientos del punto de vista del observador (como se realiza en la Vivienda y Escuela con el nodo “silla”), sino que es el propio terreno (textura del plano) el que se desplaza, permaneciendo en todo momento el modelo del avión y el punto de vista inmóviles en sus coordenadas originales.

1. **Avance continuo del avión**

Se consigue realizando el desplazamiento de la textura, más bien del *Gizmo* de la textura, en dirección –Y del eje de coordenadas. Esta dirección en realidad es contraria al avance natural que tendrían que realizar el avión y el punto de vista, pero la sensación final es la misma, es decir, parece que es el avión el que está avanzando, y en realidad lo que avanza es el terreno pero en sentido contrario como se muestran en las siguiente secuencia.

***Figura 1‑12: Avance continúo del avión***

Los cambios necesarios a nivel de implementación de código Matlab han requerido que en las líneas de código (propias del sistema BCI existente) en las que se realizaban movimientos del punto de vista, ahora lo que se haga sea el desplazamiento de la textura en la dirección que se ha descrito. Este avance se ejecuta en todo momento siempre y cuando no se haya tomado algún comando de navegación.

1. **Giro a la derecha o izquierda**

Estas opciones de navegación se simulan realizando movimientos diagonales de la textura en la dirección contraria al movimiento natural que se pretende conseguir. Además en estos casos, el propio modelo del avión también se modifica, dotando de un movimiento de rotación a su eje longitudinal en la dirección del giro que se haya seleccionado. Es decir, el avión permanece en las coordenadas originales (no se desplaza) , pero se rota su eje hacia la derecha o izquierda según el comando elegido, dotando a la escena de mayor realismo a la hora de recrear cada uno de los movimientos. En realidad la textura avanza y el avión rota y la sensación final se aproxima a un giro y avance natural de un avión real.

La siguiente secuencia de imágenes plasma el resultado de un giro a la derecha.

Figura ‑: Giro a la derecha del avión

1. **Ascensos y descensos**

En este caso la textura debe de seguir avanzando tal y como se describe en el punto 1, pero además entra en juego la ubicación del plano de vuelo texturizado. Si se desplaza paulatinamente el plano en dirección –Y, lo que se está haciendo es alejar el plano del avión, por tanto se experimenta la sensación ganar altura y alejarnos del terreno.

Si, por el contrario, se desplaza el plano en dirección Y, se está acercando a la posición donde permanentemente se localiza el avión y punto de vista, y se consigue así la sensación de estar descendiendo.

Si además se acompañan estos desplazamientos del plano texturizado con la rotación del eje transversal del avión simulando que levanta unos grados el morro en los ascensos y lo baja en los descensos, se consigue así la experiencia completa de estar realmente efectuando cada uno de los movimientos.

En la figura siguiente se puede observar el efecto conseguido en un descenso.

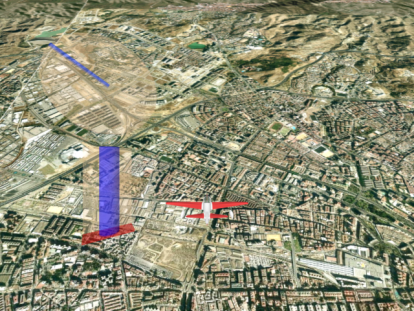
  

Figura ‑: Descenso del avión

Los métodos Matlab necesarios para la implementación de estos comandos se incluyen en las líneas de código pertinentes del sistema BCI existente. Los métodos desarrollados se muestran en el ANEXO XXX.

# PRUEBAS Y EVALUACIÓN

CONCLUSIONES Y LINEAS FUTURAS