Brain Computer Interface

María Madrid Sobrino
Ingeniería de Telecomunicación

Abstract – This report presents an EEG-based brain-computer interface (BCI) in which subjects could select a picture from a set on a computer screen. The application is centred on detecting steady-state visual evoked potentials (SSVEP) in EEG signals recorded on the scalp of the subject. BCI2000 software platform is used in this project as a basis for the whole system. The platform will link its modules and the developed ones needed to achieve the closed-loop BCI system.
In this context, a C++ computer application with 16 targets and a MATLAB signal processing module were then implemented using the proposed method. In offline tests for a set of frequencies with differences of amplitude up to 15 dB, detection was achieved. Detection was also achieved in online tests.

1. Introducción

1.1 Terminología de los BCI

Una interfaz cerebro-ordenador (llamado por sus siglas en inglés BCI – Brain Computer Interface) es un sistema de comunicaciones en el cual la intención del usuario se transmite al mundo exterior sin hacer uso de los mecanismos naturales, tales como los nervios periféricos y los músculos [1]. El concepto de los BCI surge de la necesidad de una alternativa para incrementar la comunicación y las opciones de comunicación de personas con discapacidad severa, además, sus usos potenciales se extienden desde la rehabilitación de desórdenes neurológicos y la monitorización del estado cerebral hasta la industria del videojuego [2].

Hoy en día hay dos enfoques a la hora de desarrollar un sistema BCI. Un BCI es llamado invasivo si el sistema de adquisición de señal (por ejemplo electrodos) tiene que ser implantado directamente en el cerebro. En otro caso, el sistema puede ser desarrollado sin necesidad de cirugía mediante la colocación del sistema de adquisición en la superficie del cráneo. En este último caso el BCI es llamado no invasivo. Los más prácticos y usables son aquellos basados en electroencefalografía (EEG) no invasiva, de forma que las medidas de señal son tomadas de la superficie craneal. Estas medidas proporcionan información de las ondas cerebrales mediante la grabación de la actividad eléctrica del cráneo. No obstante, otras técnicas como la magnetoencefalografía (MEG) y la imagen por resonancia magnética funcional (fMRI) han sido usadas satisfactoriamente como BCIs no invasivos [3, 4].

Los BCIs no invasivos basados en EEG generalmente utilizan respuestas cerebrales según algunos modelos como los potenciales relacionados con eventos (o ERP – Event-Related Potentials) tales como P300, potenciales evocados visuales (VEP – Visual Evoked Potentials) o potenciales evocados visuales de estado estacionario (SSVEP – Steady-State visual evoked potentials) [2]. Los ERPs son potenciales generados en el cerebro durante la presentación de estímulos. Los estímulos pueden ser generados por un sensor o un evento

psicológico. Esto genera una onda con un retraso temporal que puede ser detectada después de procesar las señales EEG [5]. Por otra parte, VEP consiste en un conjunto de ondas, concretamente aquellas que se derivan de la actividad de la corteza cerebral [6]. Las diferentes formas de onda generadas por los estímulos visuales pueden ser distinguidas según su latencia. Los potenciales VEP son llamados "fugaces" porque la baja tasa de estimulación permite a las vías sensoriales recuperarse antes que el siguiente estímulo aparezca. Cuando los estímulos visuales son presentados a una tasa constante y suficientemente alta se evita que la actividad neuronal evocada regrese a su estado base, entonces la respuesta elegida se vuelve continua. A esto se le llama SSVEP. A altas tasas de estimulación, la respuesta cerebral al estímulo toma forma sinusoidal [6].

El BCI implementado que se presenta usa el paradigma SSVEP. Los BCIs basados en SSVEP dependen de las propiedades psicofisiológicas de las respuestas EEG producidas durante la presentación de estímulos parpadeantes [7]. Cuando los estímulos son presentados al sujeto, su cerebro produce señales EEG a la misma frecuencia del estímulo y a múltiplos de la misma. En algunos trabajos de investigación (como [8]) la fase ha sido también usada como un parámetro para un estímulo visual repetitivo (RVS – Repetitive Visual Stimuli) en particular. La efectividad de los BCIs basados en SSVEP se debe a diferentes factores como la alta relación señal a ruido que se puede obtener [2] y el poco entrenamiento requerido. Sin embargo, ciertas frecuencias de estimulación pueden provocar ataques epilépticos e inducir fatiga [9].

1.2. Implementación propuesta

Con el fin de incrementar la usabilidad y las posibles aplicaciones de los BCIs basados en SSVEP, una aplicación de clasificación es propuesta. Un total de 4 estímulos son simultáneamente presentados al usuario o sujeto, el cual seleccionará uno de ellos centrando su atención en el objeto que corresponda. Cada uno de los objetos presentados está asociado a un RVS que tiene una frecuencia específica [9].

La aplicación presentada permite al usuario seleccionar entre 16 opciones. Las opciones, imágenes, se muestran en 4 grupos expandibles. De esta forma, 16 opciones están disponibles para la selección usando solo 4 estímulos. Sin embargo, el seleccionar una imagen supone una doble selección, es decir, primero el usuario ha de seleccionar uno de los 4 grupos de imágenes y posteriormente, una vez que el grupo se ha expandido en 4 imágenes, el usuario puede seleccionar la imagen deseada. Unas capturas de pantalla de la aplicación son presentadas en las figuras 1 y 2.

Figura 1: Vista de la aplicación antes de la primera selección

Figure 2: Vista de la aplicación después de la primera selección

Para construir por completo un sistema BCI desde cero el desarrollador debe preocuparse de la adquisición de los datos, el procesamiento de los mismos y la presentación de estímulos. Una vez agrupados y acoplados, estos grupos constituirán un sistema de lazo

cerrado. El sistema puede ser soportado por plataformas de software open source tales como BCI2000 [10] o OpenVive [11], las cuales proveen variedad de sistemas de adquisición e implementación de diferentes paradigmas.

Los sistemas de propósito general mencionadas permiten diseñar, probar y usar BCIs. El proyecto que se presenta usa la plataforma BCI2000 como base para el desarrollo. Tanto el módulo de procesado de señal como la aplicación mostrada al usuario fueron creadas y después integradas en la el software de BCI2000.

2. Antecedentes

La mayoría de la información necesaria previa a la realización del presente proyecto está descrita en la sección de introducción, sin embargo, hay alguna información más específica que se muestra a continuación.

Una comparación entre los diferentes rendimientos fue llevada a cabo con el fin de comparar la operación y los métodos de varios sistemas encontrados revisando trabajos previos en el campo hasta la fecha. La tabla 1 muestra los resultados obtenidos. Nota: ITR: Information Transfer Rate – Tasa de transferencia de información.

Tabla 1: Rendimiento de diferentes sistemas

Artículo	Método	ITR [bits/min]	Frecuencias. [Hz]	Objetivos	Tiempo/Selección [s]
Jia et al. [8]	Fourier coefficient projections	60	10, 12, 15	15	2.5*
Wang et al. [12]	Power spectrum analysis	74.5	10, 11, 12	16	3.08
Parini et al. [13]	Spatial Filtering and Channel Combining	51.47	6, 7, …, 17	4	2
Bin et al. [14]	Canonical Correlation analysis (CCA)	58	6.7, 7.5, 8.6, 10, 12, 15	6	2
*2s para la escoger y 0.5s después de cada selección.					

A la hora de buscar información sobre el posicionamiento de los electrodos, 3 artículos fueron examinados. Estos artículos propusieron BCIs basados en SSVEP con diferentes configuraciones de electrodos. En la tabla 2 se listan las configuraciones mencionadas. Las posiciones de los electrodos se han indicado siguiendo el sistema internacional 10-20.

Tabla 2: Posiciones de los electrodos en diferentes sistemas

Artículo	Número de electrodos	Posiciones de los electrodos
Lalor et al. [2]	2	O1, O2
Martinez et al. [15]	5	CPz, Pz, POz, P1, P2, Fz
JJ Vidal et al. [16]	6	Pz-Oz, O1-Oz, O2-Oz, I-Oz, Oz-A, Fz-Pz

3. Método

3.1. BCI2000 y visión general del sistema

La plataforma de software BCI2000 fue elegida como base para el desarrollo del BCI. Tal y como se puede ver en la página web de la plataforma [10], BCI2000 es un sistema de propósito general para la investigación de las interfaces cerebro-ordenador y es gratis para instituciones académicas y de investigación. Puede ser usado para la adquisición de datos y la

presentación de estímulos e incluye aplicaciones para la monitorización cerebral. BCI2000 soporta varios sistemas de adquisición de datos, señales cerebrales y paradigmas. BCI2000 facilita también la interacción con otro software como MATLAB.

El paquete BCI2000 viene con soporte para diferentes sistemas de adquisición de datos, rutinas para el procesado de señal y varios paradigmas implementados. De esta manera, el desarrollador puede montar su sistema BCI solo con instalar el software disponible y uno de los sistemas de adquisición de hardware compatible.

BCI2000 está compuesto por 4 módulos, los 3 más relevantes son el Source module, Signal Processing module y el Application module. Estos se encargan respectivamente de la adquisición de las señales cerebrales, el procesamiento de las mismas y el feedback para el usuario, es decir, la aplicación en sí. El cuarto módulo, Operator module, es la interfaz gráfica proporcionada al usuario que el investigador usa para manejar el resto de los módulos. Uno de los objetivos de BCI2000 es que cada uno de los módulos sea lo más independiente posible respecto al resto de ellos. Como se ha mencionado antes, no es necesario reemplazar o construir ninguno de los módulos para conseguir un sistema que funcione. Sin embargo, para lograr un BCI basado en SSVEP se decidió reemplazar el Application module. El Signal Processing module fue reemplazado también para probar la viabilidad y el rendimiento del software a la hora de usar MATLAB como herramienta para el procesado de señal. En la figura 3 se muestra un esquema de la configuración del sistema desarrollado, incluyendo los módulos relevantes de BCI2000 y sus interacciones con los módulos que han sido diseñados.

Para establecer el entorno de BCI2000, la última versión de su instalador puede ser descargada y luego instalada. El código fuente se puede descargar también para compilarlo posteriormente siguiendo la guía proporcionada en [17].

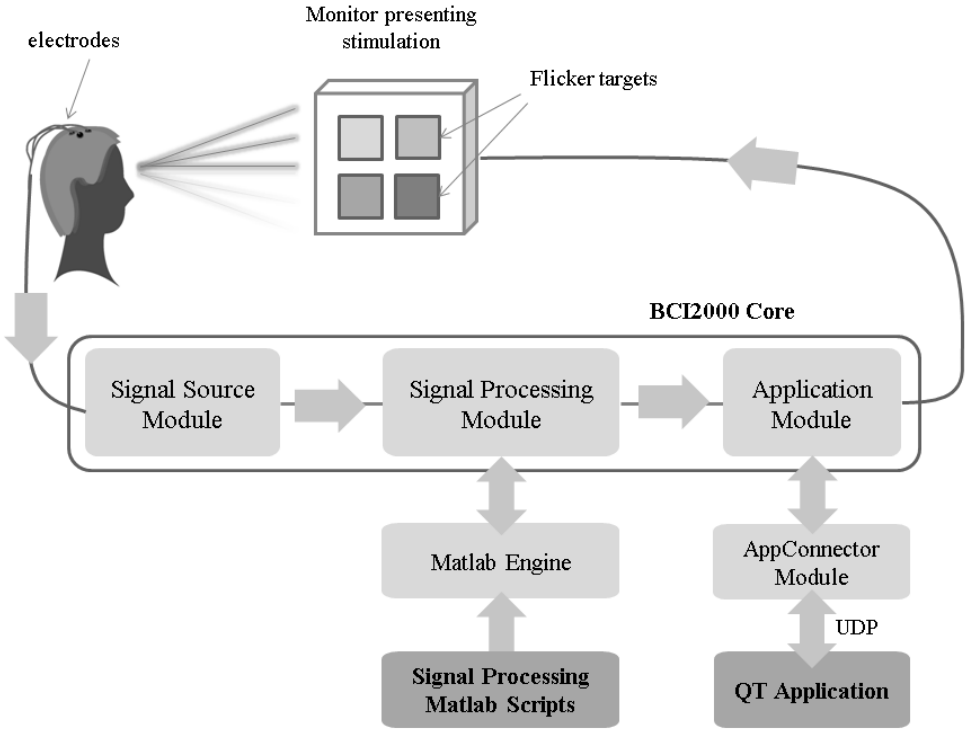

Figura 3: Esquema de la configuración del sistema. Modulos de BCI2000 y Matlab en gris claro. Módulos desarrollados en gris oscuro.

3.2. Signal Processing module

El Signal Processing module de BCI2000 se comporta como una caja negra para el resto del sistema, recibe señales cerebrales desde el Signal Source module y manda señales de control a la aplicación. BCI2000 permite a los desarrolladores usar varios módulos de procesado de señal listos para usar que vienen integrados en la plataforma pero el desarrollador tiene también la posibilidad de crear uno nuevo. Hay dos posibilidades a la hora de desarrollar un módulo de procesado de señal para BCI2000, construyendo un módulo en C++ integrado en BCI2000 o realizando el procesado en MATLAB. MATLAB interactuará con BCI2000 recibiendo datos de entrada y devolviéndole luego los resultados del procesado tal y como se muestra en la figura 3.

En el caso descrito aquí, el procesado con MATLAB fue usado en vez de construir un módulo integrado en BCI2000 con C++. Para ello BCI2000 proporciona una simple interfaz detallada a continuación. El módulo de procesado de señal se implementa como una serie de scripts de MATLAB cuyo formato ha sido previamente definido por BCI2000. El *MatlabSignalProcessing* module permite conectar MATLAB con BCI2000 mediante el protocolo UDP. Cuando BCI2000 está ejecutando, cada bloque de datos se envía a MATLAB y el script pertinente es ejecutado. Los script más relevantes que pueden ser implementados se detallan a continuación:

- *bci_Construct*: Inicializa los parámetros y estados.
- *bci_Preflight*: Es usado para comprobar la consistencia de los parámetros, si son correctos o no. También devuelve las dimensiones de la señal de salida.
- *bci_Initialize*: Determina los coeficientes del filtro.
- *bci_StartRun*: Se utiliza para resetear el estado del filtro al principio de cada ejecución.
- *bci_Process*: Procesa una señal de entrada (un único bloque de datos) de acuerdo a la cadena de procesado, y devuelve el resultado del procesado en una única variable de salida.

Los parámetros son usados para configurar los módulos de adquisición de señal, procesado de señal y la aplicación. Los estados se refieren al estado global del funcionamiento de BCI2000.

Si alguna de las funciones *bci_Preflight*, *bci_Initialize* o *bci_Process* no está disponible, un *warning* se mostrará al usuario. Por tanto, estas funciones deben ser implementadas para que el sistema funcione. Puede encontrarse más información sobre las funciones mencionadas en [18, 19].

Aun así, hay que tener en cuenta el esfuerzo requerido para transformar una implementación existente de procesado *offline* en una implementación online. Mientras que BCI2000 trata de llevar a cabo esta transformación de la manera más simple posible, no se puede eliminar la carga que provoca el manejo de fracciones de datos, lo cual implica la necesidad de usar *buffering,* (en vez de tener acceso inmediato a un conjunto de datos continuo, sería necesario mantener un buffer de datos adicional), y lidiar con la interfaz de MATLAB, manteniendo un estado consistente entre las continuas llamadas al script de procesamiento [20]. Esta necesidad de *buffering* está presente en el script *bci_Process*. Éste sobrelleva la carga que supone usando un *scrolling buffer*. Esto quiere decir que cada vez que un nuevo bloque de datos es recibido del módulo anterior, el bloque más antiguo es desechado y reemplazado por el recién llegado.

El método implementado de procesado de señal que se ha implementado utiliza la siguiente cadena de procesado a los datos obtenidos del módulo de adquisición. Existe primeramente un filtro paso banda que aísla las frecuencias de los estímulos que se presentan en la aplicación SSVEP. Después de ello se obtiene la densidad espectral de potencia de la salida del filtro. El valor de la potencia de las frecuencias de cada estímulo constituirá la salida de nuestro procesado. Esta operación se realiza para cada uno de los canales (a saber, cada uno de los filtros paso banda), creando así una matriz con dimensiones *"canales x número_de_estímulos"* que será posteriormente adaptada al formato requerido por BCI2000. La Figura 4 muestra una descripción gráfica del proceso para cada canal.

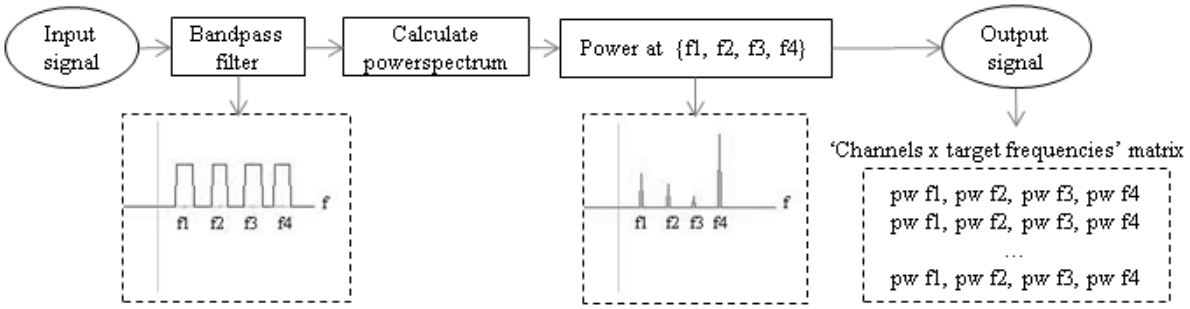

Figura 4: Esquema de procesado de señal

Como el tamaño de los bloques que se intercambian entre los distintos módulos de BCI2000 es uniforme, las dimensiones de la señal de salida deben ser las mismas que las dimensiones de la señal de entrada, que son "*canales x tamaño_de_bloque*". La señal que es producida como salida del procesado de señal descrito tiene dimensiones diferentes, con lo cual, antes de enviarla al siguiente módulo, es expandida hasta la dimensión correcta añadiendo ceros (a esto se le conoce como *zero padding*). El tamaño de bloque puede ser libremente configurado por el usuario y el número de canales es típicamente fijado en el módulo de adquisición. No obstante, el usuario o desarrollador puede a posteriori elegir cuántos canales usar en la opción *Config* disponible en el módulo *Operator*.

Una vez que BCI2000 está ejecutando con el módulo de MATLAB, una ventana con la línea de comandos de MATLAB se mostrará gráficamente. En ese terminal el desarrollador puede ejecutar comandos con el fin de visualizar el contenido de las variables usadas por BCI2000 para la comunicación con MATLAB [21]. Es necesario asegurarse de que las dos plataformas establecen la conexión y se comunican correctamente. Es importante verificar en este punto que los archivos de BCI2000 están localizados en el directorio C/ y comprobar la variable PATH tal como se menciona en [22].

3.3. Application Module

La aplicación ha sido creada usando el lenguaje C++ con ayuda el IDE Qt Creator [23]. Qt Creator es un IDE (*Integrated Development Enviroment*) multiplataforma que funciona sobre los sistemas operativos Windows, Linux y Mac OS y permite a los desarrolladores crear diferentes aplicaciones para múltiples plataformas, tanto de escritorio como móviles. Es un software disponible gratuitamente que se puede descargar desde su web oficial e instalar por sí solo o como parte del Qt SDK (*Software Development Kit*).

A la hora de crear una aplicación para la estimulación visual usando el paradigma SSVEP es importante tener en cuenta el número total de objetivos o comandos. Un número

alto de objetivos ofrece la posibilidad de aumentar por tanto el número de comandos pero puede disminuir la velocidad y precisión del sistema [9]. La aplicación desarrollada en este proyecto permite al usuario seleccionar entre 16 opciones, como se explicó en la introducción (la ventana principal, que es la que presenta la estimulación SSVEP, ha sido mostrada previamente en la Figura 1). La clase *MainWindow* constituye la interfaz gráfica principal de la aplicación. Ésta presenta 4 objetivos en la pantalla y proporciona a la derecha una lista de miniaturas que representan las opciones que ya han sido seleccionadas, así como ciertos botones para su manipulación sin el uso de la estimulación visual. El botón Start inicia la estimulación y el procesado, y los botones de selección ayudan al uso de la aplicación en el caso de que la selección vía BCI2000 falle. Se ha incluido también una barra de menú superior en la que se accede la configuración de la aplicación.

La programación concurrente fue necesaria para poder presentar correctamente el parpadeo simultáneo de las cuatro imágenes. Si solo se hubiera utilizado temporizadores el parpadeo hubiera sido incorrecto y defectuoso, por lo tanto, el uso de un hilo ejecutando el temporizador que controla el parpadeo de cada imagen ha resultado obligatorio. De esta forma cada temporizador se ejecuta de forma autónoma. Para este propósito se ha implementado la clase *TimerThread*, que simplemente consiste en un hilo que contiene un temporizador que controlará la frecuencia de parpadeo de una determinada imagen, de esta forma los temporizadores no se interrumpen entre ellos.

El número de estímulos y la frecuencia de los mismos viene limitada por la frecuencia de refresco del monitor [12], por lo tanto las cuatro frecuencias usadas para los objetos parpadeantes fueron seleccionadas teniendo esto en cuenta. Como el monitor usado tiene una frecuencia de refresco de 60 Hz, las frecuencias disponibles serían las siguientes {60, 30, 20, 15, 10, 8.57, 7.5, 6.66, 6, … , 1 Hz}, las cuales han sido calculadas atendiendo a la relación frecuencia_refresco/frame, siendo frame = {1, 2, 3,…, 60}. Con el fin de evitar frecuencias cercanas el set de frecuencias usado ha sido siguiente {6, 10, 15, 20 Hz}.

La conexión entre la aplicación y BCI2000 se establece mediante la herramienta *AppConnector* proporcionada por BCI2000. Esta interfaz proporciona un enlace bidireccional para intercambiar información con procesos externos siempre que se ejecuten en la misma máquina, o en otra máquina que se encuentre en la misma red local [24]. Como se muestra en la Figura 4, la interfaz *AppConnector* se conecta con BCI2000 usando el protocolo UDP. Por cada bloque de datos procesado por BCI2000 dos tipos de información son enviados pueden ser recibidos por la aplicación externa, los estados internos y la señal de control de BCI2000. Los estados internos describen si el sistema está ejecutando o está suspendido, el tiempo cuando un bloque de datos fue tomado, etc… La señal de control es la salida del procesado de señal y tiene el siguiente formato: *señal(<canal>,<elemento>) = valor decimal en codificación ASCII*, donde <*canal*> es el índice del canal y <*elemento*> es la muestra. En este caso, los valores relevantes de <*canal*> serán {0, 1, 2, 3}, según el formato de la señal esperada descrito en la sección anterior. El formato de los mensajes de *AppConnector* no está relacionado en ningún caso con los mensajes binarios que intercambia BCI2000 entre sus módulos.

El código fuente de BCI2000 incluye un ejemplo de uso de esta herramienta llamado *AppConnectorExample*, el cual permite capturar paquetes UDP provenientes de BCI2000. Este ejemplo has sido creado por el equipo de BCI2000 usando QT e incluye todo el código fuente, con lo cual su integración con la aplicación SSVEP se traduce simplemente en añadir estos archivos al proyecto y conectarlos de forma que sirve los datos recogidos a la aplicación para su post-procesado. En la Figura 5 se puede ver una captura de pantalla del módulo

AppConnector finalmente conectado con la aplicación desarrollada. Alguno de los estados comentados aparece como mensajes recibidos.

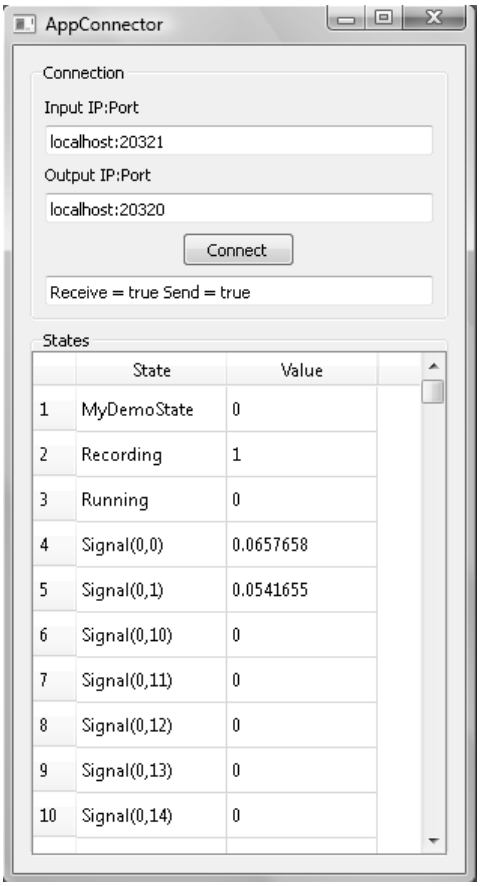

Figure 5: AppConnector recibiendo mensajes

En la aplicación se ha incluido también una funcionalidad de post-procesado para los datos entrantes. Esta funcionalidad ofrece la posibilidad de obtener el resultado de la clasificación promediando todos los canales recibidos, mediante la información de uno sólo de los canales o mediante una suma ponderada de todos los canales. El procesado lo lleva acabo la clase *MainWindow*.

Para escoger cualquiera de estos modos de obtención del resultado, se ha incluido una herramienta gráfica de configuración que es accesible hacienda click en el menú superior de la ventana principal de la aplicación. La que clase *Configuration* que implementa esta herramienta controla la selección una de las tres opciones disponibles y las variables que almacenan la información necesaria para cada una de ellas. Estas variables son usadas posteriormente por la clase *MainWindow* para ejecutar el algoritmo de post-procesado. En la Figura 6 se muestra la interfaz gráfica de la herramienta de configuración.

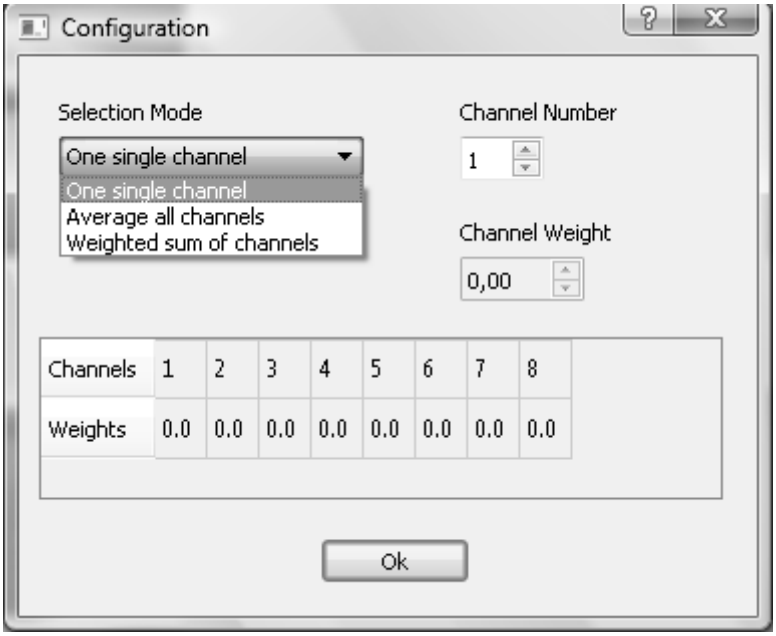

Figure 6: Interfaz de usuario del menú de configuración

Con la información obtenida con *AppConnector* se realiza el post-procesado de los datos y se produce un resultado de clasificación, o lo que es lo mismo, la imagen es finalmente escogida y colocada como miniatura en uno de los huecos destinados a tal efecto en la derecha de la pantalla principal. El programa esperará un pequeño period de 5 segundos para recibir los datos y procesarlos. Un primer resultado se producirá después del procesado y resultará en la expansión de uno de los cuatro grupos de imágenes. Posteriormente se repetirá el proceso para para seleccionar la imagen deseada dentro de las cuatro posibles que han sido expandidas. La imagen elegida se situará en el espacio libre mencionado anteriormente una vez ha sido seleccionada. Si el tiempo de espera expira y el resultado de la clasificación no ha sido recibido, el usuario tendrá que hacer click manualmente en uno de los 4 pequeños botones situados al a derecha del botón *Action!*. Estos botones aparecen en la imagen que se presentó en la Figura 1. Este botón fue añadido para asegurar que la selección fuera posible en cualquier caso, especialmente cuando la conexión entre BCI2000 y la aplicación fuera errónea

3.4. Adquisición de datos EEG

Las medidas de EEG fueron realizadas usando el hardware de adquisición Deymed TruScan 32. Para la recogida de datos fueron utilizados ocho electrodos localizados en los lóbulos central, parietal, occipital y temporal, cuyas posiciones de acuerdo al sistema internacional 10-20 se corresponden con Cz, P3, Pz, P4, O1, O2, T5 y T6. La posición de estos electrodos fue elegida cerca de las regiones parietal y occipital ya que la potencia producida usando el paradigma SSVEP es mayor en las zonas cercanas al córtex parietal y occipital [25]. La frecuencia de muestreo usada fue de 1024 Hz. En la Figura 7 se muestra la configuración de los electrodos empleada.

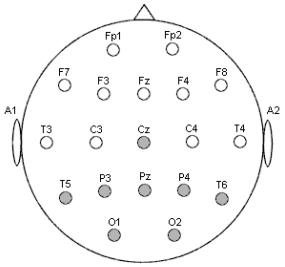

Figure 7: Configuración de los electrodos

Se realizaron una serie de medidas usando el set de frecuencias {6, 10, 15, 20 Hz}. Por cada una de las imágenes fueron grabados 20 segundos de datos. El sujeto fijó su vista durante 10 segundos para la primera selección y repitió a continuación la misma operación durante otros 10 segundos para la segunda y definitiva selección.

4. Resultados

Hay dos opciones principalmente cuando se está probando un Sistema BCI: el análisis online y análisis offline. En este caso, la aproximación más apropiada para la aplicación desarrollada es el análisis online. Esto fue lo que se persiguió en un primer momento pero la conexión entre MALTAB y BCI2000 no resultó satisfactoria. Después de multitud de intentos de solucionar el problema y teniendo en cuenta la restricción temporal para la entrega del proyecto se decidió cambiar al análisis *offline*. El análisis *offline* se llevó a cabo simulando el comportamiento de BCI2000 en MATLAB por medio de un script. Este script genera señales *dummy* con unas componentes frecuenciales específicas de mayor potencia, de forma que esta señal *dummy* está básicamente formada por una combinación lineal de ondas sinusoidales con diferentes amplitudes. Estas señales, (que sustituyen a las que habrían sido generadas, capturadas y transformadas al formato requerido como se ha mencionado anteriormente), atraviesan entonces la cadena de scripts requeridos por BCI2000. Ya que la amplitud de estas componentes frecuenciales que forman la señal es libremente configurable esto proporciona una forma muy sencilla de testear los scripts que tratan la señal. De esta forma, variando la amplitud a diferentes frecuencias se han obtenido los resultados que se pueden ver en la Tabla 3, que resumen cuándo la aplicación es capaz de detectar un objetivo cuando se está realizando la selección. La diferencia de amplitud expresa en dBs cuanto mayor es la amplitud de la frecuencia objeto del testeo en cada caso respecto al resto de frecuencias del set elegido para nuestras imágenes. Por ejemplo, al comprobar el caso de la frecuencia 15 Hz con 15 dB de diferencia, la amplitud para el set {6, 10, 15 20 Hz} sería {1, 1, 31.6, 1}, valores para los cuales la aplicación ya es capaz de discernir que la selección ha sido detectada.

Tabla 3: Detección de frecuencias asociadas a cada objetivo

		Diferencia de amplitud (dB)		
		10	15	20
Frecuencia (Hz)	6	no detectado	no detectado	detectado
	10	detectado	detectado	detectado
	15	no detectado	detectado	detectado
	20	no detectado	detectado	detectado

El análisis online es posible una vez se ha configurado la conexión entre BCI2000 y MATLAB. Para llevarlo a cabo el Sistema se establece de la siguiente manera: Primero se inicia BCI2000 usando la herramienta *BCI2000Launcher* proporcionada por la plataforma y después se seleccionan los módulos *Signal Source*, *Signal Processing* y *Application*. Para el procesado con MATLAB se ha de seleccionar *MatlabSignalProcessing* como módulo de procesado de señal. Hay que ser conscientes de que el sistema espera encontrar los scripts a ejecutar en el *path* de MATLAB. Para usar la aplicación SSVEP desarrollada el módulo *DummyApplication* tiene que ser seleccionado como *Application* module porque ha sido desarrollada como un módulo externo y no integrado en BCI2000. Como estos tests fueron realizados sin un hardware de adquisición de EEGs el módulo *SignalGenerator* ha de ser elegido como *Signal Source* módulo. Una vez se ha lanzado BCI2000 como se ha descrito, la *Command Window* de MATLAB se abrirá y el módulo *Operator* se mostrará para que se puedan configurar las opciones. En la Figura 7 se muestra el escenario esperado.

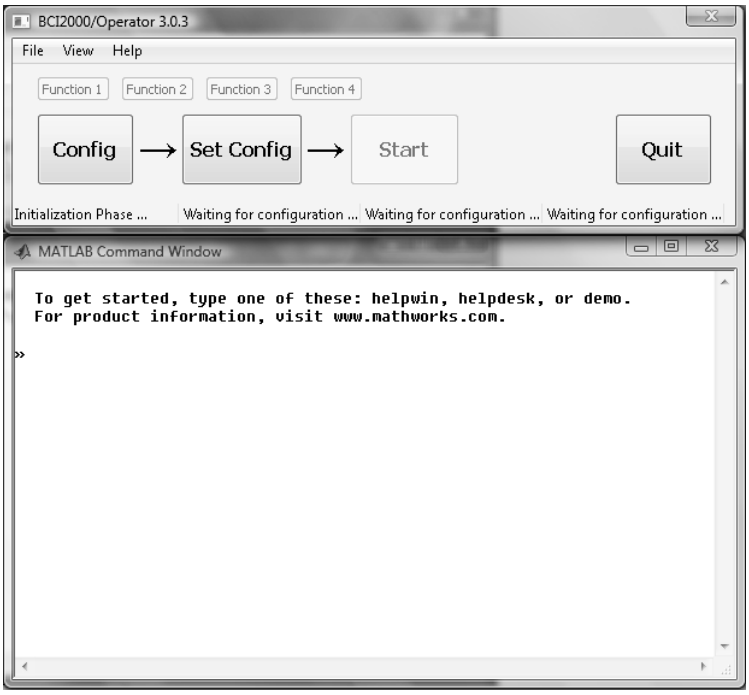

Figura 7: Módulo *Operator* esperando configuración y *Command Window* de MATLAB.

El siguiente paso sería abrir la aplicación SSVEP desarrollada y configurar las opciones para la realización de los tests. Para hacer esto último habría que hacer click en la opción *Config* del módulo *Operator*. En nuestro caso los parámetros configurables son el set de frecuencias y el tiempo de cada selección. También es necesario igualmente configurar la fuente, el generador de señales, para los tests a realizar. Para configurar la conexión UDP entre BCI2000 y la applicación, los parámetros a introducir en la pestaña *Connector* deben permanecer como sigue;*ConnectorInputAddress*: localhost:20320, *ConnectorOutputAddress*: localhost:20321. La configuración del *AppConnector* tiene que ser compatible a través de los siguientes parámetros; *Input IP:Port*: localhost:20321, *Output IP:Port*: localhost:20320.

El modulo del generador de señales fue configurado para producir señales de frecuencias 6, 10 15 y 20 Hz. Por tanto, se esperaba que la aplicación seleccionara los objetivos 1, 2, 3 y 4 respectivamente. Los objetivos 1, 2, 3 y 4 se corresponden con las imagines o grupos de imágenes en las posiciones superior izquierda, superior derecha, inferior

izquierda e inferior derecha. Esto quiere decir que, por ejemplo, cuando el generador produce un seno de 5 Hz el procesado dará como resultado la selección de la imagen superior izquierda o el correspondiente grupo de imágenes que esté en esa posición. En la Tabla 4 se muestran los resultados obtenidos para ciertas amplitudes de señal y de ruido.

Tabla 4: Resultados de clasificación de la aplicación

Frecuencia	Amplitud	Amplitud del ruido blanco	Objetivo detectado
6	100muV	30muV	1 (posición superior izquierda)
10	100muV	30muV	2 (posición superior derecha)
15	100muV	30muV	3 (posición inferior izquierda)
20	100muV	30muV	4 (posición inferior derecha)

5. Discusión

5.1. Paradigma SSVEP

El paradigma SSVEP fue elegido como solución para el BCI por su efectividad. Los sistemas basados en este paradigma logran altas ratios de relación señal a ruido [2] y requieren menor entrenamiento para el sujeto. La señal producida por un estímulo SSVEP es medible en la mayoría de la población y la tarea de extraer la información se reduce tan sólo a hallar el componente frecuencial de la misma con mayor potencia dentro de las componentes para las que se esperan mayores picos de potencias que se corresponden con la frecuencia de parpadeo de cada uno de los objetivos ofertados en el BCI. No obstante es importante tener en cuenta que este tipo de estímulos visuales a ciertas frecuencias pueden provocar ataques epilépticos y ciertos tipos de iluminación podrían dañar la visión del sujeto. Además de ello, algunas frecuencias pueden inducir fatiga [9].

5.2. Análisis y resultados

Los resultados del estudio del análisis offline muestran que diversas respuestas a los SSVEP pueden ser usadas como toma de decisiones o como Sistema de elección de imágenes cuando la diferencia de amplitud de las componentes frecuenciales propuestas es suficientemente alta (más de 15 dB). Para el procesado online, la señal recibida y procesada ha sido una onda senoidal a una determinada frecuencia, por lo que en teoría no existen otros componentes frecuenciales a excepción de la propia frecuencia de la onda, lo que facilita la clasificación.

Aunque no se puede apreciar en las tablas 3 y 4, las frecuencias más altas son las que mejores y más fáciles resultados de clasificación producen. Esto quiere decir que hay mayor diferencia de potencia entre la componente frecuenciales que presentan más potencia. Al testear frecuencias más bajas esta diferencia se aminora. Sin embargo el método de clasificación propuesto solo tiene en cuenta la componente frecuencial de mayor potencia y no proporciona información sobre cómo de fiable es la clasificación en sí.

Ejecutando el análisis online se han podido ver algunos resultados interesantes gracias a la herramienta de visualización de tiempos de BCI2000. La Figura 8 muestra la herramienta *Timing* cuando el módulo de procesado de señal está corriendo.

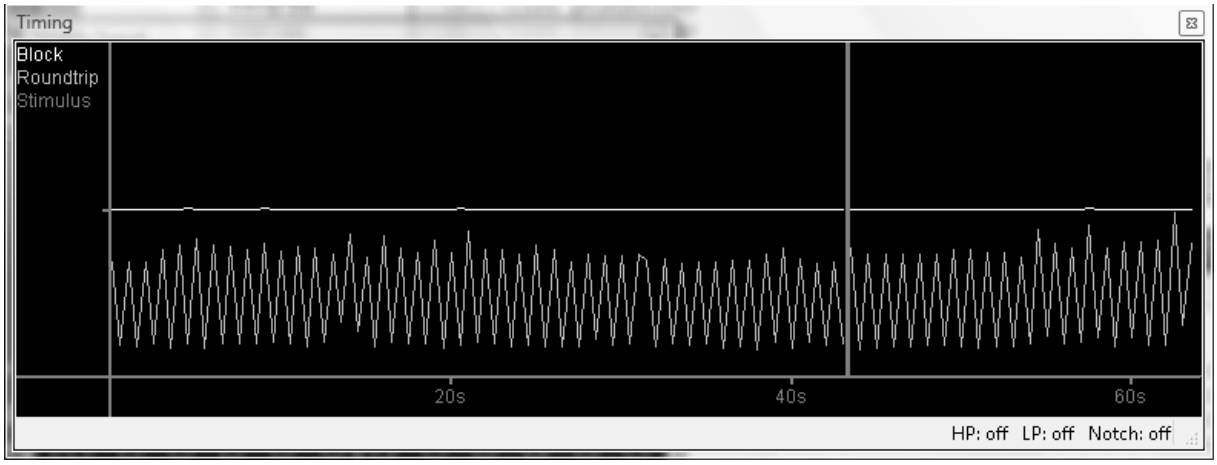

Figura 8: Herramienta de visualización de los tiempos cuando el módulo de procesado está corriendo. El *Timing* es un asunto crítico en un Sistema que procesa datos en tiempo real. En BCI2000, los datos son adquiridos y procesados en bloques regulare e idealmente en intervalos regulares. Para trabajar en tiempo real el Sistema necesita acabar el procesado, mostrar las imágenes y guardar el resultado en la duración de un bloque. El Roundtrip es el tiempo necesario para que un módulo atraviese todos los bloques del sistema. Para cumplir con esta restricción el roundtrip no puede exceder del tiempo de duración físico de un bloque. Para un funcionamiento estable del sistema se ha impuesto la condición de que el tiempo medio de roundtrip se mantenga por debajo de la duración del bloque [26].

Se observa que cuando el tamaño de bloque toma el valor 64, el Roundtrip es mejor que para tamaños menores como 32. También, cuando se usa una tasa de muestreo superior, aparecen picos más acusados, haciendo más cercanos los valores del Roundtrip y Block, lo cual puede producir un *warning* o incluso detener el procesado. El problema llegados a este punto surge si una tasa de 1024 Hz y un tamaño de bloque de 64 son usados, ya que el sistema produce un *warning* al exceder el Roundtrip al tiempo de duración. Esta es la configuración necesaria por el hardware para obtener las señales EEG. Sería obligatorio por tanto si se quiere lograr la funcionalidad completa del Sistema según estos requisitos el mejorar el código MATLAB de forma que fuera más ligero y rápido, cambiar este procesado por un módulo C++ integrado en BCI2000 o bien utilizar un hardware de adquisición de señales EEG alternativo. Las opciones más asequibles serían las que conllevan cambios en el software antes que en lugar del hardware.

5.4. Análisis del EEG

La inexactitud de las frecuencias puede ocurrir cuando se usa un monitor corriente o al uso para la presentación de estímulos SSVEP, debido a la imprecisión inherente del software y la tasa de refresco del monitor. Se observa, por ejemplo, que la frecuencia esperada de 6 Hz termina siendo de 5.313 Hz si uno examina la densidad espectral de potencia de las señales EGG tomadas. Es necesario tener esto en cuenta con el fin de evitar el filtrado de las frecuencias incorrectas por parte del filtro paso banda. También hay que tener cuenta el ruido producido por la red eléctrica a la frecuencia de 50 Hz y sus correspondientes armónicos.

6. Pequeña conclusión

La aplicación y el procesado de señal pueden ser usados como herramienta de investigación, la cual proporciona al investigador la capacidad de realizar diversos tests para diferentes frecuencias y configuraciones de canales con el fin de obtener mejores resultados.

Agradecimientos: La autora agradece "la guía" y soporte del supervisor del proyecto Dr. S. Nasuto. Así mismo, agradece especialmente a Matthew Spencer por su ayuda y apoyo durante la realización y desarrollo del proyecto así como la correspondiente fase de pruebas.

7. Referencias

[1] J.R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan,"Brain-computer interfaces for communication and control" *Clinical Neurophysiology,* vol. 113, no.6, pp. 767–791, 2002.

[2]. E. C. Lalor , S. P. Kelly , C. Finucane , R. Burke , R. Smith , R. B. Reilly , G. McDarby, "Steady-state VEP-based brain-computer interface control in an immersive 3D gaming environment", *EURASIP Journal on Applied Signal Processing,* v.2005 n.1, pp. 3156-3164, 1 January 2005.

[3] J. Mellinger , G. Schalk , C. Braun , H. Preissl , W. Rosenstiel , N. Birbaumer and A. Kuebler "An MEG-based brain-computer interface (BCI)", *NeuroImage,* vol. 36, pp.581 - 593 2007.

[4] S.-S. Yoo, T. Fairneny, N.-K. Chen, et al., "Brain-computer interface using fMRI: spatial navigation by thoughts," *NeuroReport* , vol. 15, no. 10, pp. 1591-1595, 2004.

[5] Reza Fazel-Rezai and Waqas Ahmad (2011). P300-based Brain-Computer Interface Paradigm Design, Recent Advances in Brain-Computer Interface Systems, Prof. Reza Fazel (Ed.), ISBN: 978-953-307-175-6, InTech. Disponible en: http://www.intechopen.com/books/recent-advances-in-brain-computer-interface-systems/p300-based-brain-computer-interface-paradigm-design

[6] Zani A., Mado Proverbio A. (2003). The Cognitive Electrophysiology of Mind and Brain. Academic Press.

[7] H. Segers, A. Combaz, N.V. Manyakov, N. Chumerin, K. Vanderperren, S. Van Huffel, and M.M. Van Hulle, (2011) "Steady State Visual Evoked Potential (SSVEP) -based Brain Spelling System with Synchronous and Asynchronous Typing Modes". IFMBE Proceedings, vol. 34 15. *Nordic-Baltic Conference on Biomedical Engineering and Medical Physics (NBC15).* Aalborg, Denmark, June 14-17, 2011, pp. 164-167.

[8] Jia C, Gao X, Hong B, Gao S (2011) "Frequency and phase mixed coding in SSVEP-based brain–computer interface". *IEEE Trans Biomed* Eng. 58: pp. 200–206.

[9] Zhu D, Bieger J, Molina G G and Aarts R M 2010 "A survey of stimulation methods used in SSVEP-based BCIs". *Comput. Intell. Neurosci.* 1, 2010, p. 702357.

[10] Schalk et al., IEEE Trans Biomed Eng, 2004 Mellinger and Schalk, In: Brain-Computer Interfaces. MIT Press, 2007 [http://www.bci2000.org]

[11] Y. Renard, F. Lotte, G. Gibert, M. Congedo, E. Maby, V. Delannoy, O. Bertrand, A. Lécuyer, "OpenViBE: An Open-Source Software Platform to Design, Test and Use Brain-Computer Interfaces in Real and Virtual Environments", Presence: teleoperators and virtual environments, vol. 19, no 1, 2010.

[12] Wang, Y.; Wang, Y.T.; Jung, T.P. "Visual stimulus design for high-rate SSVEP BCI". Electron. Lett. 2010, 46, pp.1057-1058.

[13] S. Parini, L. Maggi, A. C. Turconi, and G. Andreoni, "A robust and self-paced BCI system based on a four class SSVEP paradigm: algorithms and protocols for a high-transfer-rate direct brain communication," *Computational Intelligence and Neuroscience*, vol. 2009, Article ID 864564, 11 pages, 2009.

[14] G. Bin, X. Gao, Z. Yan, B. Hong, and S. Gao, "An online multi-channel SSVEP-based brain-computer interface using a canonical correlation analysis method," *Journal of Neural Engineering*, vol. 6, no. 4, Article ID 046002, 6 pages, 2009.

[15] P. Martinez, H. Bakardjian, and A. Cichocki, "Fully Online, Multi-Command Brain Computer Interface with Visual Neurofeedback Using SSVEP Paradigm," J. *Computational Intelligence and Neuroscience.*

[16] J. J. Vidal, "Real-time detection of brain events in EEG," *Proc. IEEE*, vol. 65, no.5, pp. 633-641, 1977.

[17] BCI2000 Wiki: Programming Reference: Build System. Disponible en: http://www.bci2000.org/wiki/index.php/Programming_Howto:Building_BCI2000

[18] Schalk G., Mellinger J. (2010). Writing a Custom Matlab Filter. A Practical Guide to Brain–Computer Interfacing with BCI2000, pp. 114-119 Springer.

[19] BCI2000 Wiki: Programming Reference: MatlabFilter. Disponible en: http://www.bci2000.org/wiki/index.php/Programming_Reference:MatlabFilter

[20] BCI2000 Wiki: Programming Tutorial: Implementing a Matlab-based Filter. Disponible en: http://www.bci2000.org/wiki/index.php/Programming_Tutorial:Implementing_a_Matlab-based_Filter

[21] Schalk G., Mellinger J. (2010). MatlabFilter. A Practical Guide to Brain–Computer Interfacing with BCI2000, pp. 179-180 Springer.

[22] BCI2000 Wiki: Troubleshooting at Programming Reference: MatlabFilter. Disponible en: http://www.bci2000.org/wiki/index.php/Programming_Reference:MatlabFilter#Troubleshooting

[23] QT Creator IDE and tools. Disponible en: http://qt.nokia.com/products/developer-tools/

[24] Schalk G., Mellinger J. (2010). AppConnector. A Practical Guide to Brain–Computer Interfacing with BCI2000, pp. 92-96 Springer.

[25] Ding J, Sperling G, Srinivasan R (2006) "Attentional modulation of SSVEP power depends on the network tagged by the flicker frequency". Cereb Cortex 16: pp. 1016–1029.

[26] BCI2000 Wiki: User Reference: Timing. Disponible en: http://www.bci2000.org/wiki/index.php/User_Reference:Timing

Todas las páginas web citadas en este documento fueron verificadas por última vez el 22 de Abril de 2012.