



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

DISCA
DEPARTAMENTO DE INFORMÁTICA
DE SISTEMAS Y COMPUTADORES



Departamento de Ingeniería de Sistemas y Automática
Departamento de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia

Trabajo Fin de Máster

Máster Universitario en Automática e Informática Industrial

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Documentos:

- 1 Memoria
- 2 Manual de uso
- 3 Manual del programador
- 4 Presupuesto

Autor:

D. Javier Dadone Ravera

Tutor:

D. Eduardo Quiles Cucarella

D. Martín Mellado Arteché

Valencia, septiembre de 2017

Dedicada a mi familia y amigos.

Agradecer al instituto de automática e informática industrial (AI2) por su amabilidad y facilidad a la hora de trabajar con el robot, así como el apoyo técnico recibido. Agradecer también a mi tutor D. Eduardo Quiles Cucarella por su dedicación al proyecto y por facilitarme las herramientas necesarias para la realización del mismo.

No olvidarme de familiares y amigos que sin ellos esto no hubiera sido posible, especialmente a mi padre, un referente y excelente profesional.

Sin esfuerzo y sacrificio no hay beneficio.

Resumen

Este trabajo tiene como objetivo diseñar un interfaz cerebro computador basado en señales electroencefalográficas (EEG) para controlar de manera voluntaria los movimientos y acciones de un robot industrial de seis grados de libertad. Se trabajará en la adquisición y calibración de la señal EEG, en diseñar la interfaz más adecuada para el control del robot, así como en desarrollar la aplicación de control y validar la calidad de los resultados obtenidos mediante pruebas reales.

Como resultado, se ha obtenido una aplicación multiplataforma en C++, que junto a Openvibe es capaz de controlar un robot industrial de la compañía Staübli basándose en el paradigma Steady State Visually Evoked Potentials (SSVEP).

Palabras clave: EEG, SSVEP, Openvibe, Brain Computer Interfaces (BCI), control, robótica, robot, movimiento de robot, C++, OpenSource.

Summary

The project consists in the design a brain computer interface based on electroencephalographic (EEG) signals to control the movements and actions voluntarily of an industrial robot. We will work on the acquisition and calibration of the EEG signal, designing the most appropriate interface to move the robot, developing the robot control application and validating the quality of the results obtained through real tests.

As a result, a cross-platform application in C ++ has been obtained, which together with Openvibe is able to control an industrial robot from the Staubbli company based on the Steady State Visually Evoked Potentials (SSVEP) paradigm.

Keywords: EEG, SSVEP, Openvibe, Brain Computer Interfaces (BCI), control, robotics, robot, robot movement, C++, OpenSource.

Resum

Aquest treball té com a objectiu dissenyar una interfície cervell computador basat en senyals electroencefalogràfics (EEG) per controlar de manera voluntària els moviments i accions d'un robot industrial de sis graus de llibertat. Es treballarà en l'adquisició i calibratge del senyal EEG, en dissenyar la interfície més adequada per al control del robot, així com en desenvolupar l'aplicació de control i validar la qualitat dels resultats obtinguts mitjançant proves reals.

Com a resultat, s'ha obtingut una aplicació multiplataforma en C ++, que al costat de Openvibe és capaç de controlar un robot industrial de la companyia Stäubli basant-se en el paradigma Steady State Visually Evoked potentials (SSVEP).

Paraules clau: EEG, SSVEP, Openvibe, Brain Computer Interfaces (BCI), control, robòtica, robot, moviment de robot, C++, OpenSource.



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

DISCA
DEPARTAMENTO DE INFORMÁTICA
DE SISTEMAS Y COMPUTADORES



Departamento de Ingeniería de Sistemas y Automática
Departamento de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia

Trabajo Fin de Máster

Máster Universitario en Automática e Informática Industrial

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

1.- MEMORIA

Autor:

D. Javier Dadone Ravera

Tutor:

D. Eduardo Quiles Cucarella

D. Martín Mellado Arteché

Valencia, septiembre de 2017

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DEL PROYECTO

ÍNDICE DE IMÁGENES.....	3
ÍNDICE DE TABLAS.....	4
1. OBJETO DEL PROYECTO	1
2. ESTADO DEL ARTE.....	1
2.1. Funcionamiento del cerebro.....	1
2.2. Colocación de los electrodos	3
2.3. Sistema de posicionamiento de los electrodos superficiales.....	4
3. JUSTIFICACIÓN DEL PROYECTO.....	6
4. SOLUCIONES ALTERNATIVAS	9
4.1. Paradigma	9
4.1.1. Motor Imagery y Ritmos MU.....	9
4.1.2. P300.....	10
4.1.3. Steady State Visually Evoked Potentials (SSVEP).....	12
4.1.4. Selección del paradigma.....	13
4.2. Aplicación BCI	15
4.2.1. OpenVibe	15
4.2.2. BCI2000	16
4.2.3. Matlab + Toolbox.....	17
4.2.4. Selección de la Aplicación BCI	18
5. DESCRIPCIÓN DETALLADA DE LA SOLUCIÓN.....	21
5.1. Componentes y materiales utilizados	21
5.2. Funcionamiento de la aplicación	23
5.3. Comunicación	25
5.4. Controles.....	28
5.5. Programación Concurrente	29
6. JUSTIFICACIÓN DE LA SOLUCIÓN.....	30
6.1. Descripción del ensayo	30
6.2. Resultados obtenidos	31

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

6.2.1.	Valores Teóricos	31
6.2.2.	Valores Experimentales.....	32
6.2.3.	Encuesta del ensayo	37
7.	TRABAJO FUTURO.....	38
7.1.	Aplicación I. Control del brazo robot industrial.....	38
7.1.1.	Diseño de la aplicación	38
7.1.2.	Funcionalidad.....	38
7.1.3.	Aplicación	38
7.2.	Aplicación II. Comprobación de resultados	39
8.	CONCLUSIONES	39
9.	BIBLIOGRAFIA.....	40
	ANEXO I.....	41

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DE IMÁGENES

Figura 1: Vista lateral izquierda de los lóbulos del cerebro, Cinteco Psicología Clínica y Psiquiátrica: Lóbulos cerebrales (vista lateral) ©, 2011.....	2
Figura 2: Vista de perfil en la situación de los electrodos Fz, Cz y Pz.	5
Figura 3: Vista de frente en la situación de los electrodos T3, T4, C3 y C4.....	5
Figura 4: Sistema de colocación de electrodos 10-20.	6
Figura 5: Aplicación de una interfaz cerebro-computador para rehabilitación en un hospital.	7
Figura 6: Resumen gráfico del proyecto de investigación de la Universidad de Minnesota (USA).....	8
Figura 7: Configuración de electrodos para el paradigma Motor Imagery.	9
Figura 8: Configuración de electrodos para el paradigma P300.	10
Figura 9: Respuesta del paradigma P300.	11
Figura 10: Interfaz Cerebro-Computador del paradigma P300.	11
Figura 11: Configuración de electrodos para el paradigma SSVEP.....	12
Figura 12: Entorno de programación de OpenVibe.....	15
Figura 13: Entorno de BCI2000. Ejemplo de aplicación con P300.....	16
Figura 14: Entorno de la librería BCILAB de Matlab.....	17
Figura 15: Entorno de la librería EEGLAB de Matlab.....	18
Figura 16: Casco Enobio3G 8 Canales.....	21
Figura 17: Electrodos húmedos (Izquierda), Electrodos secos (Derecha).....	22
Figura 18: Robot Stäubli TX60 en el departamento AI2 (UPV).....	22
Figura 19: Entorno de trabajo compuesto por la aplicación de control y el robot Stäubli.	23
Figura 20: Pantalla principal de la aplicación ssvep-robot-bci-5-online-test-shooter....	24
Figura 21: Resumen del conexionado entres aplicaciones.	27
Figura 22: Combinación de teclas programadas y su funcionalidad.	28
Figura 23: Diagrama de bloques de la distribución de hilos de ejecución	29
Figura 24: Disposición de los objetivos y orden de aparición.....	31

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DE TABLAS

Tabla 1: Comparación entre paradigmas.	14
Tabla 2: Comparación entre aplicaciones.	19
Tabla 3: Tabla de estímulos más utilizados.	27
Tabla 4: Movimiento teórico mínimo para el ensayo.	31
Tabla 5: Comparativa resultados temporales entre sujetos.	35
Tabla 6: Comparativa de tasa de éxito entre sujetos.	36
Tabla 7: Resultados de los ensayos.	41

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

1. OBJETO DEL PROYECTO

El objetivo principal del presente proyecto es implementar un interfaz cerebro-computador para poder realizar una comunicación entre el pensamiento de un sujeto y el control de un sistema real. Para el desarrollo del mismo se evaluarán las mejores técnicas y paradigmas necesarios para realizar un control óptimo del sistema minimizando una serie de factores imprescindibles en la comunicación cerebro-computador como son el tiempo de entrenamiento y los errores o falsos positivos.

La realización de este proyecto tiene como objeto tres cuestiones principales:

- 1- Seleccionar el paradigma óptimo. Estudiar la colocación de los electrodos para aumentar la tasa de éxito, así como configurar y seleccionar de manera adecuada el programa de análisis y tratamiento de las señales recibidas.
- 2- Realizar una aplicación sencilla que permita al sujeto interactuar. Ha de ser de fácil configuración y uso. La aplicación diseñada junto con el programa de tratamiento y análisis de señales seleccionado en el apartado anterior han de establecer un protocolo de comunicación con la finalidad de realizar acciones externas en consecuencia a los deseos del sujeto.
- 3- Se dispone de una aplicación externa la cual se controla el guiado de un robot industrial. Para ello es necesario establecer un protocolo de comunicación entre la aplicación diseñada y la aplicación externa.

2. ESTADO DEL ARTE

2.1. Funcionamiento del cerebro

En este apartado se tratará de explicar de forma básica y simple las áreas y funciones principales del cerebro, que es una de las partes de las que se compone el encéfalo.

El cerebro consta de dos mitades llamadas hemisferios cerebrales que se encuentran unidas mediante el cuerpo calloso. Cada hemisferio funciona de modo

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

diferente y se relaciona con distintas actividades. El hemisferio derecho se encarga de controlar el lado izquierdo del cuerpo y se encuentra relacionado con la expresión no verbal (voces, gestos, melodías, percepción táctil, visualización espacial, etc.), mientras que el hemisferio izquierdo es el encargado de controlar el lado derecho del cuerpo y está vinculado con la expresión verbal y el razonamiento lógico.

Además, cada hemisferio cerebral puede ser dividido en cuatro lóbulos, denominándose igual que los huesos craneales que tienen encima como se observa en la *Figura 1*:

- Lóbulo frontal, se asocian a las funciones mentales superiores como la capacidad de concentración, planificación y secuenciación de acciones de forma concreta.
- Lóbulo parietal, que comprende dos funciones. La primera función integra información sensorial para formar una sola percepción, mientras que la segunda función construye un sistema coordinado espacial para representar el mundo alrededor del cuerpo.
- Lóbulo temporal procesa la información sensorial auditiva y se encargan de darle un significado apropiado.
- Lóbulo occipital es el encargado del procesamiento visual y espacial, de la discriminación del movimiento y de la discriminación del color.

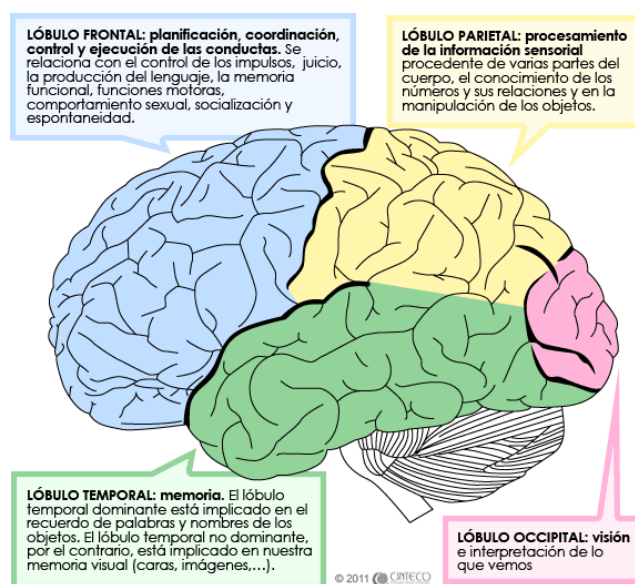


Figura 1: Vista lateral izquierda de los lóbulos del cerebro, Cinteco Psicología Clínica y Psiquiátrica: Lóbulos cerebrales (vista lateral) ©, 2011.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

2.2. Colocación de los electrodos

En este apartado se informará al lector de los principales procedimientos para realizar la captación de la actividad bioeléctrica cerebral del sujeto, así como incidiendo en los tipos de electrodos utilizados para el fin.

La actividad bioeléctrica cerebral puede captarse por diversos procedimientos y su nomenclatura hace referencia al mismo. El procedimiento puede ser invasivo o no invasivo dependiendo de donde y como se realicen las mediciones, los principales son los siguientes:

- Sobre el cuero cabelludo
- En la base del cráneo
- En el cerebro expuesto
- En localizaciones cerebrales profundas

Para captar la señal es necesario el uso de electrodos que hay de diversos tipos:

- Electrodo superficial: Se aplican sobre el cuero cabelludo directamente y existen varios tipos:
 - o Adheridos o adhesivos: Consisten en un pequeño disco metálico de aproximadamente 5 mm de diámetro con resistencia de contacto muy baja el cual se fija a la piel con una pasta conductora que puede ser colodión o bentonita.
 - o De contacto: Consta de un tubo de cloruro de plata enroscado sobre un soporte de plástico que queda sujeto a la base del cráneo mediante bandas eléctricas. El extremo inferior del tubo debe estar en contacto con una almohadilla humedecida en solución salina y a su vez en contacto con el cuero cabelludo del sujeto.
 - o Gorro elástico o casco de malla: La base es similar a los electrodos de contacto, excepto por el hecho de que los electrodos se encuentran ubicados en un casco elástico de forma que facilitan la colocación y mejoran la comodidad al sujeto.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

- De aguja: Su uso es cada vez menor puesto que su colocación es dolorosa y necesitan ser colocados cuidadosamente y con mucha precisión. Los electrodos han de ser esterilizados previamente suponiendo sobrecostes.
- Electrodo basales: se aplican en la base del cráneo sin necesidad de procedimiento quirúrgico
- Electrodo quirúrgicos: para su aplicación es necesario realizar cirugía al sujeto y pueden ser corticales o intracerebrales. Su colocación ha de ser precisa y se consiguen resultados y señales de alta calidad y fiabilidad.

Tras analizar las distintas posibilidades y en base al material disponible, se ha seleccionado para realizar este trabajo un procedimiento no invasivo mediante electrodos superficiales utilizando un gorro elástico.

2.3. Sistema de posicionamiento de los electrodos superficiales

Una vez seleccionado el procedimiento y los electrodos, es necesario conocer el sistema de posicionamiento de los electrodos para garantizar una constancia en la medición, así como una cierta calidad en la adquisición de las mismas.

Los electrodos superficiales pueden ser posicionados de diferentes maneras. Para este proyecto se ha seguido el sistema internacional 10-20 que se explica a continuación:

- Los números pares corresponden al hemisferio cerebral derecho y los impares al izquierdo.
- Las letras representan las áreas frontales (Fp y F), centrales (C), temporales (T), parietales (P), occipitales (O) y las apófisis mastoides (M).
- La letra Z corresponde con los electrodos situados en la línea media.
- Se mide la distancia entre los huesos nasion e inion, pasando por el vértex. A una distancia del 10% sobre el nasion se coloca el electrodo Fpz (frontal polar en la línea media) y a la misma distancia pero sobre el inion se coloca el Oz (occipital en la línea media).
- Entre los puntos Fpz y Oz se sitúan tres puntos más a una distancia de un 20% de la medida nasion-inion entre electrodo y electrodo, situando así Fz, Cz y Pz (*Figura 2*).

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

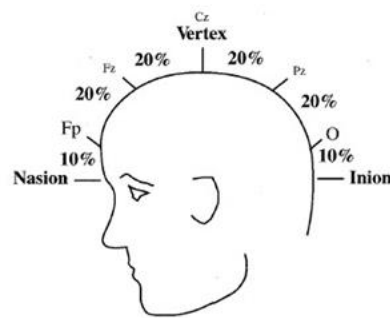


Figura 2: Vista de perfil en la situación de los electrodos Fz, Cz y Pz.

- Se mide la distancia entre los puntos preauriculares (pasando por el vertex), y a una separación del 10% de la medida total se sitúan los electrodos T3 y T4 sobre cada uno de dichos puntos.
- A un 20% de esta última medida coronal, se sitúan los electrodos C3 y C4 sobre la línea que va desde los electrodos temporales hasta el vertex (*Figura 3*).

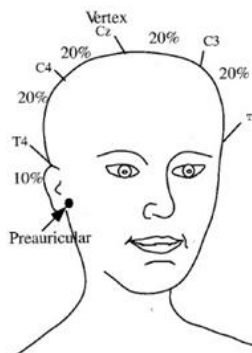


Figura 3: Vista de frente en la situación de los electrodos T3, T4, C3 y C4.

- Los electrodos F3 y F4 están situados a una distancia equidistante entre el electrodo Fz y la línea coronal. De la misma forma, los electrodos P3 y P4 equidistan entre esa línea y Pz.
- Midiendo la distancia entre Fpz y Oz pasando por T3, se pueden colocar los electrodos Fp1 y Fp a una distancia del 10% respecto a Fpz. De igual modo, se pueden colocar los electrodos O1 y O2 a una separación del 10% respecto de Oz.
- El electrodo F7 (y F8) se sitúa a una distancia equidistante entre Fp1 (o Fp2) y T3 (o T4), al igual que ocurre con P7 (y P8), que se sitúa equidistantemente entre T3 (o T4) y O1 (u O2).

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

- Los electrodos mastoides M1 y M2 se colocan junto las apófisis mastoides.

NOTA: La *Figura 4* muestra la colocación final de los electrodos de los últimos cuatro pasos.

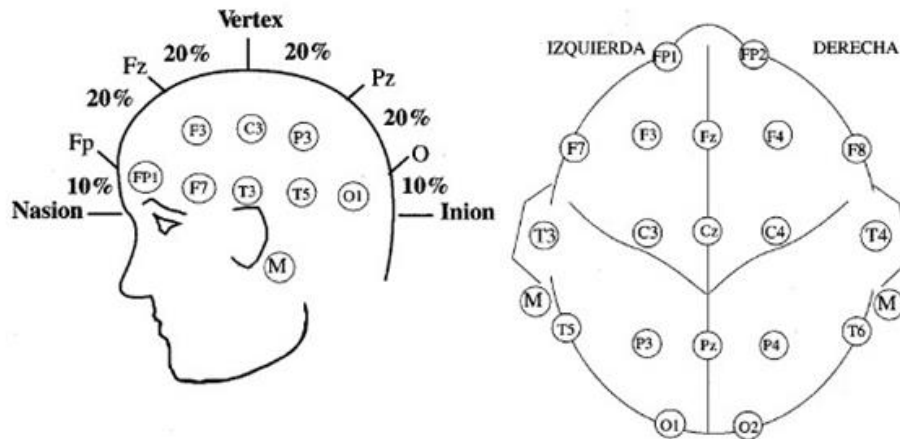


Figura 4: Sistema de colocación de electrodos 10-20.

3. JUSTIFICACIÓN DEL PROYECTO

Hoy en día la robótica es un campo que se encuentra en continuo crecimiento. Las mejoras y avances en la electrónica y en los sistemas de control, así como el conocimiento y la evolución en materiales hacen de la robótica un campo futuro. Actualmente existen robots que realizan numerosas actividades en el campo de la industria y además la robótica ha avanzado este último siglo por el lado servicial, lo conocido como robótica de servicios en la cual se engloban todos aquellos robots que facilitan labores humanas como limpieza, por ejemplo.

Desde hace unos años se está dando mucha importancia a la terminología de robot colaborativo y su utilidad. El concepto hace referencia a un robot que pueda coexistir con un humano sin éste sufrir ningún tipo de peligro. Actualmente se están implantando robots colaborativos para rehabilitación o incluso en industria realizando tareas de pick and place, ensamblaje, empaquetado ...

Por otro lado, la neurociencia es un área interdisciplinaria cuya función principal es estudiar cómo funciona el sistema nervioso, más concretamente, el cerebro. Además,

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

es un campo muy extenso en el que participan otras ciencias como: neuroquímica, neuroanatomía, neurofarmacología, neuropsicología, etc.

Quizá, una de las disciplinas más importantes hoy en día es la neurociencia cognitiva, creada a partir de la combinación de la neurociencia con la psicología cognitiva. Dicha ciencia, se encarga de estudiar los procesos mentales que afectan a la conducta y a otros procesos como la atención, memoria, aprendizaje, pensamientos, etc.

Un mejor conocimiento en esta área implica avances en el entendimiento de enfermedades neurológicas (como en el caso del alzhéimer) y psiquiátricas (como en el caso de la esquizofrenia) que afectan al sistema nervioso.

La unión de estas dos disciplinas y junto con el gran avance en los algoritmos de clasificación que se ha producido este siglo permiten solventar problemas variados. Por ejemplo, un equipo de trabajo de Minnesota (USA) liderado por Kai Keng Ang [1] en el año 2009 han publicado una aplicación de rehabilitación la cual usa el paradigma Motor-Imagery como base. La *Figura 5* muestra a un sujeto utilizando el prototipo desarrollado para rehabilitación.

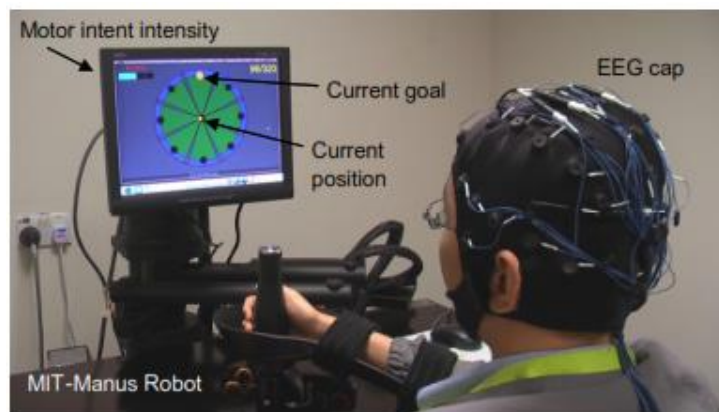


Figura 5: Aplicación de una interfaz cerebro-computador para rehabilitación en un hospital.

En el proyecto utilizan el movimiento imaginado del paciente para realizar acciones repetitivas de rehabilitación. Así una persona con discapacidad o con problemas de movilidad puede realizar los movimientos en su casa sin necesidad de un profesional. Acoplan además electrodos cutáneos en la extremidad a movilizar para acompañar el movimiento intencionado y que el sistema complemente la fuerza necesaria para acabar el ejercicio.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

El departamento de ingeniería biomédica de Minnesota (USA) ha planteado recientemente el control de un brazo robot para realizar tareas de agarre de piezas mediante el uso del paradigma Motor-Imagery [2]. La aplicación permite el control bidimensional de un cursor virtual o de un brazo robot. El movimiento imaginado de mano izquierda, derecha, ambas manos y la relajación permiten el movimiento a izquierdas, derechas, arriba y abajo del robot. La *Figura 6* muestra un resumen de la funcionalidad del proyecto.

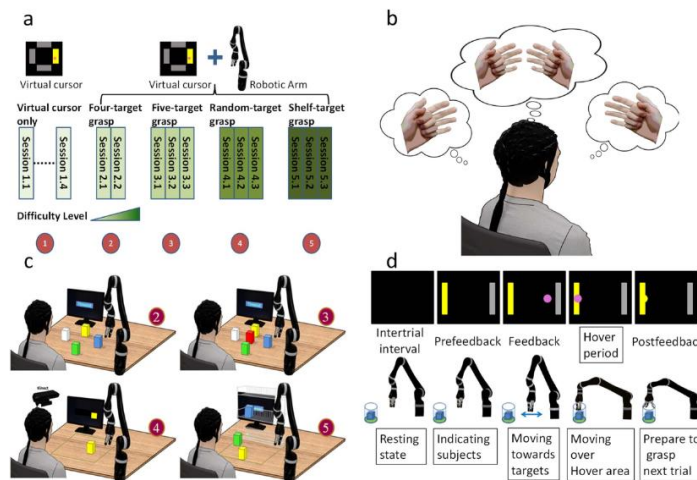


Figura 6: Resumen gráfico del proyecto de investigación de la Universidad de Minnesota (USA).

Investigadores de la Universidad politécnica de Bari (Italia) plantean en 2004 una aplicación que permite a un usuario caminar por un entorno de realidad virtual haciendo uso del paradigma SSVEP [3]. Para ello utilizan tres estímulos oscilando a distinta frecuencia que permiten caminar, girar a derecha o a izquierda. En este proyecto se hace uso de una red neuronal de convolución para realizar la clasificación.

En 2013 se introducen aplicaciones con el paradigma SSVEP que resultan curiosas como el control de una silla de ruedas con el uso de este paradigma [4] y su correspondiente estudio de fatiga realizado un año después [5]. Esto muestra que el campo de aplicación de esta tecnología aumenta con los años llegando a controlarse drones e incluso robots móviles.

Se han indicado solo cuatro de los numerosos proyectos que combinan estas tecnologías. En el presente proyecto se pretende introducir el paradigma de SSVEP como metodología de control.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

4. SOLUCIONES ALTERNATIVAS

4.1. Paradigma

En el siguiente apartado se comentarán tres de los paradigmas más usuales en las aplicaciones BCI (Brain Control Interface), basadas en electroencefalografía (EEG).

4.1.1. Motor Imagery y Ritmos MU

Los ritmos mu se le llama a la actividad eléctrica oscilante que se registra en las zonas motoras centrales del cerebro. Medidas con un electroencefalograma se obtiene señales que oscilan entre ocho y trece hercios de frecuencia y que su amplitud varía con la actividad motora.

Cuando el sujeto realiza, observa o imagina una acción motora, un gran número de neuronas entran en sincronía variando la amplitud de los patrones mu.

La *Figura 7* muestra una posible configuración de electrodos para el paradigma de movimiento imaginario. Con esta configuración y tras un periodo de entrenamiento se puede discernir entre el movimiento imaginario de pies, mano derecha o mano izquierda.

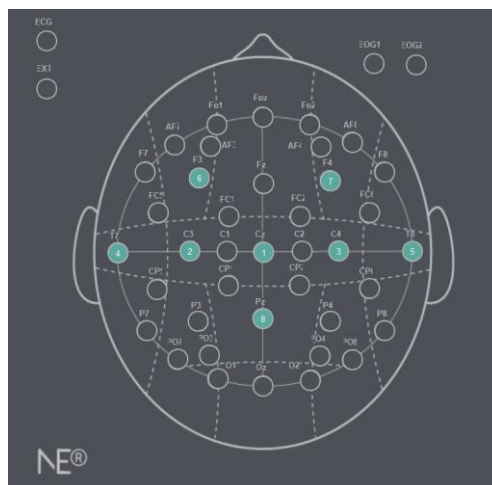


Figura 7: Configuración de electrodos para el paradigma Motor Imagery.

El presente paradigma es usado en numerosas aplicaciones, se dispone de tres variables de control que permiten abarcar cualquier tipo de aplicación.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

4.1.2. P300

Event Related Potentials (ERPs o Potenciales Evocados en castellano) es el nombre que recibe la medida de la respuesta cerebral ante un resultado directo de un evento sensorial, cognitivo o de movimiento. El paradigma P300 es uno de los paradigmas más usados basados en ERP. Se trata de un paradigma cuya lectura se corresponde con un pico en la señal positiva registrada mediante encefalografía exactamente a 300 ms después de haberse producido un estímulo al sujeto.

Este paradigma además presenta una serie de propiedades que lo hacen popular. Los electrodos como se muestra en la *Figura 8* se encuentran ubicados en la zona occipital del cerebro correspondiente como se ha visto en apartados anteriores a estímulos visuales. Además, su ubicación es completamente simétrica frente al electrodo central Cz y se encuentran colocados en la *Figura 8* según la distribución internacional estándar de electrodos 10-20.

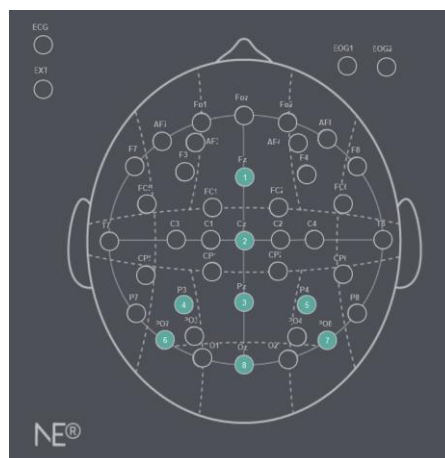


Figura 8: Configuración de electrodos para el paradigma P300.

El pico registrado y por lo que hace característico este paradigma tiene un rango de amplitud comprendido entre 2 y 5 μV con una duración comprendida entre 150 y 200 ms. Esta forma de onda característica se registra a 300 ms de haberse producido un estímulo. La *Figura 9* muestra la forma de onda característica.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

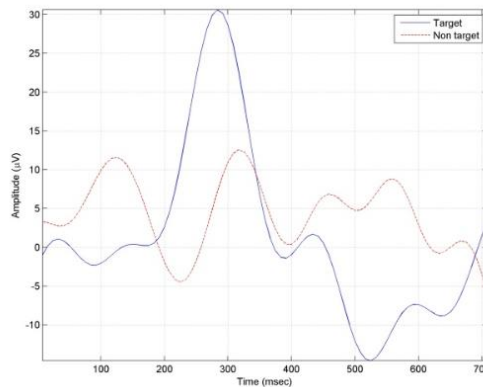


Figura 9: Respuesta del paradigma P300.

El principal uso de este paradigma ha sido como ayuda en la escritura a personas con discapacidad mediante el uso de un teclado virtual que permite deletrear las palabras y así poder interactuar con computadoras, maquinas e humanos. El ejemplo más extendido de interfaz cerebro-computador es el que se muestra en la *Figura 10*.



Figura 10: Interfaz Cerebro-Computador del paradigma P300.

El funcionamiento es simple, las filas y columnas se van iluminando según un patrón programado. El sujeto fija la atención en una letra y se analiza su respuesta cerebral (P300). Mediante un correcto preprocesado de la señal, una correcta extracción de las características y un óptimo clasificador se puede saber la letra que el sujeto está mirando.

Usando una interfaz modificada, pero manteniendo la misma forma de provocar el estímulo (por filas y columnas), se puede abordar cualquier tipo de problema, en este caso se evalúa la opción de controlar un robot industrial utilizando este paradigma.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

4.1.3. Steady State Visually Evoked Potentials (SSVEP)

El paradigma SSVEP como el descrito anteriormente trabaja con estímulos visuales (La distribución de electrodos se refleja en la *Figura 11*). Los estímulos de este paradigma se encuentran oscilando a una cierta frecuencia y cuando el sujeto visualiza el estímulo, su retina se excita y genera una señal eléctrica de la misma frecuencia que el estímulo (incluyendo armónicos). El rango de frecuencias en la cual la retina puede ser excitada se comprende entre los 3.5 Hz y los 75 Hz.

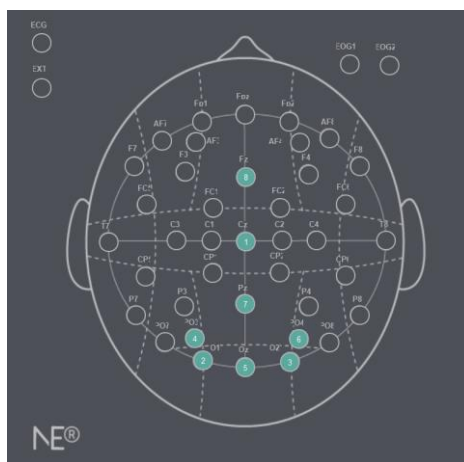


Figura 11: Configuración de electrodos para el paradigma SSVEP.

El paradigma pretende discernir entre los estímulos visualizados realizando un filtrado y una clasificación por frecuencias. De este modo se dispone de un gran número de variables de control para cualquier tipo de aplicaciones. Sus principales características son:

- Las señales son claramente diferenciables
- Se consiguen resultados óptimos con un mínimo tiempo de entrenamiento
- El usuario puede parpadear y tragar sin que afecte a la señal.
- Se permiten retrasos de unos segundos.

Destacar que los estímulos pueden ser virtuales (Led parpadeando en una pantalla) o se puede disponer de hardware que permita emitir una señal periódica, como un diodo LED.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Estímulo virtual: Permite una mayor flexibilidad ya que no es necesario conectado y además es fácilmente actualizable. El principal problema es el software. La frecuencia es un parámetro crítico que pequeñas variaciones debidas a retrasos en software puede llevar a falsas lecturas. Otro problema para tener en cuenta es la pantalla, que presenta una oscilación propia que suele ser 50 o 60 Hz, que se soluciona seleccionando estímulos múltiplos de la frecuencia de oscilación de la pantalla.

Estímulo hardware: Lo más común es usar diodos LED que mediante un pequeño circuito electrónico se consigue emitir la frecuencia deseada.

Los principales problemas que presenta este paradigma es que satura la percepción visual, cansando al sujeto pudiendo llegar a ser molesto y agotador y hay un riesgo de epilepsia con estímulos oscilando entre 15 y 25 Hz.

Además de su uso médico para diagnosticar patologías visuales, este paradigma ha recobrado importancia en el campo de las interfaces cerebro-computador debido a que dispone de diferentes posibilidades y configuraciones, así como la aplicación directa a numerosos procesos.

4.1.4. Selección del paradigma

El presente proyecto, como se ha comentado consiste en el control de un brazo robot industrial mediante algún paradigma que sea capaz de transcribir los deseos del sujeto a acciones de control válidas. Para poder realizar una selección del mejor paradigma se ha realizado una tabla comparativa (ver *Tabla 1*) en la cual se analizan los siguientes aspectos:

- Tiempo de entrenamiento: Mide el tiempo medio que el sujeto ha de entrenar (tanto clasificadores como filtros) para garantizar unos buenos resultados.
- Velocidad de respuesta: Con este parámetro se pretende cuantificar la velocidad desde que comienza el estímulo hasta que se aplica la acción de control al proceso real.
- Nivel de concentración: Refleja la actividad mental del sujeto, así como medidor del cansancio de manera indirecta.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

- Ruido de la señal: Estos parámetros reflejan la calidad de la señal recibida y los problemas que existen cuando el sujeto se mueve o realiza acciones comunes como parpadear o tragar saliva.
- Número de electrodos: Este parámetro cuantifica si es necesario una alta cantidad de electrodos para obtener buenos resultados o no.

		BAJO	MEDIO	ALTO
Tiempo de entrenamiento	Motor Imagery			X
	P300		X	
	SSVEP	X		
Velocidad en la respuesta	Motor Imagery	X		
	P300		X	
	SSVEP			X
Nivel de concentración	Motor Imagery			X
	P300			X
	SSVEP	X		
Ruido en la señal debido a movimientos del sujeto	Motor Imagery			X
	P300			X
	SSVEP		X	
Ruido en la señal tras parpadear o tragar saliva	Motor Imagery	X		
	P300			X
	SSVEP		X	
Número de electrodos mínimos necesarios	Motor Imagery			X
	P300			X
	SSVEP	X		

Tabla 1: Comparación entre paradigmas.

El control del robot se realizará en bucle abierto y en tiempo real, despreciando retrasos de señal. De ahí que sea imprescindible una alta velocidad de respuesta. Con esta restricción el único candidato posible es el paradigma SSVEP.

No solo aporta una alta velocidad de comunicación, sino que el sujeto requiere una baja concentración y el cansancio debido a la concentración es bajo. Como se ha visto en apartados anteriores, puede conllevar a cansancio visual debido a los múltiples estímulos oscilando. Además, el tiempo de entrenamiento es de minutos, mientras que los demás paradigmas estudiados requieren días, semanas incluso meses para obtener resultados similares. Al hablar del ruido, se observa que tras realizar movimientos tanto voluntarios como involuntarios (parpadeo o tragar saliva) el paradigma puede generar falsos positivos, pero en general es robusto. Finalmente comentar que este paradigma necesita como mínimo tres electrodos, el resto se pueden configurar para atenuar ruidos y mejorar la respuesta mientras que los otros paradigmas competidores requieren como mínimo ocho.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

4.2. Aplicación BCI

En el siguiente apartado se comentarán cuatro de las aplicaciones más utilizadas hoy en día para la realización de aplicaciones BCI.

4.2.1. OpenVibe

Openvibe tiene sus inicios en Bliff C++. Bliff C++ Library es una librería de código abierto que permite realizar ensayos offline y comienza a introducir en 2008 clases que permiten el diseño y testado de aplicaciones online. Fabian Lotte, investigador de INRIA (Institut National de Recherche en Informatique et en Automatique, Francia), fue uno de sus creadores. En el año 2007 nace el proyecto Openvibe y en 2009 se deja de dar soporte a la librería Bliff C++.

Openvibe es un software libre desarrollado por INRIA con el propósito de diseñar, testear y utilizar interfaces cerebro-computador. Se trata de una librería en C++ la cual permite adquirir señales, filtrarlas y acondicionarlas para posteriormente realizar un clasificado o visualización de las mismas en tiempo real. Dispone de una fácil programación ya que la lógica del programa se distribuye en diagrama de bloques como se observa en la *Figura 12*.

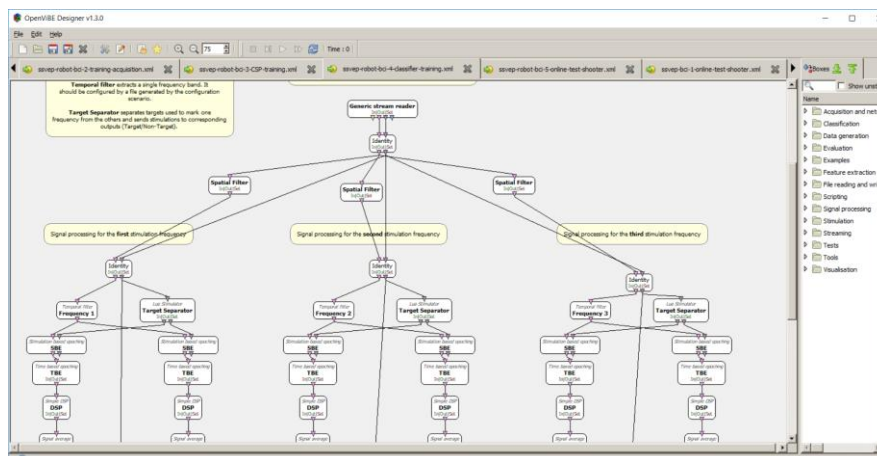


Figura 12: Entorno de programación de OpenVibe.

Desde el 2007 hasta la actualidad, numerosos investigadores han aportado su conocimiento al proyecto dotándolo de una mayor funcionalidad. La librería incluye una

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

serie de demostraciones de los paradigmas más utilizados, así como una serie de drivers que lo hacen compatible con una alta gama de software de adquisición de señales.

4.2.2. BCI2000

BCI2000 es un software para realizar investigación en el campo de las interfaces cerebro-computador. Es usado mayormente para adquisición de señales, creación y muestra de estímulos y monitorización. El proyecto nace en el año 2000, pero la primera versión del programa emerge en 2001 y fue liderado por un grupo de investigadores del departamento de salud de Albany (Estados Unidos).

Cinco años después del nacimiento del proyecto, han realizado una serie de workshops anuales explicando la teoría y la aplicación en el programa. El programa posee una amplia documentación que contiene tutoriales, un amplio y comprensivo manual de usuario y de referencias técnicas.

Destacar que BCI2000 incluye diferentes herramientas de conversión de datos haciéndolo compatible con herramientas de cómputo como es Matlab.

Los principales paradigmas con los que trabaja BCI2000 son los movimientos imaginarios y P300 como se observa en la *Figura 13*.

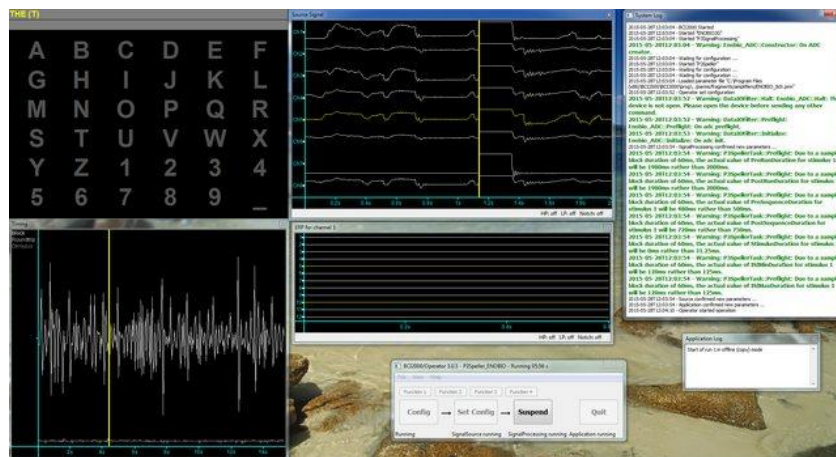


Figura 13: Entorno de BCI2000. Ejemplo de aplicación con P300.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

4.2.3. Matlab + Toolbox

Matlab es una herramienta multiplataforma de software matemático que ofrece un entorno de programación con un lenguaje de programación propio. Se trata de uno de los softwares más importantes de cálculo matricial y representación de datos. Matlab destaca por la capacidad de programar aplicaciones, así como la posibilidad de añadir más funcionalidad mediante toolboxes y librerías externas.

BCILAB se trata de una librería para Matlab desarrollada para investigar como las actividades neuronales codifican la información y poder desarrollar así aplicaciones que aprovechen las señales neuronales para controlar sistemas externos. La librería fue desarrollada para el ámbito clínico en el cual ayuda a personas tetrapléjicas y con problemas de movilidad muscular para controlar dispositivos externos como sillas de ruedas o deletreado de palabras. Los paradigmas utilizados son movimiento imaginario y P300. La *Figura 14* muestra el entorno y la librería en cuestión.

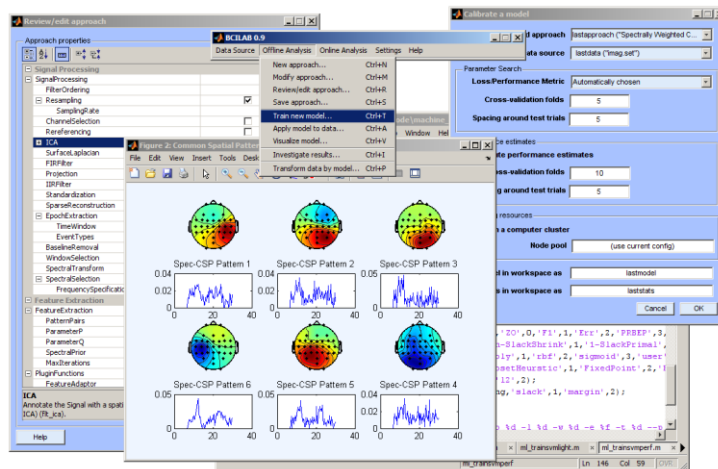


Figura 14: Entorno de la librería BCILAB de Matlab.

EEGLAB se trata de una librería de Matlab para el procesamiento continuo de señales EEG. La librería dispone de una interfaz gráfica interactiva que permite al usuario navegar con flexibilidad pudiendo así procesar los datos de forma dinámica usando métodos de separación multivariable como el ICA (Independent component analysis) o análisis temporales y frecuenciales (TFA). EEGLAB incorpora una serie de tutoriales que facilitan su uso, así como workshops anuales que se realizan desde el 2004. El último se

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ha realizado en Aspet, Francia (Julio de 2017). La *Figura 15* muestra un ejemplo de aplicación y el entorno.

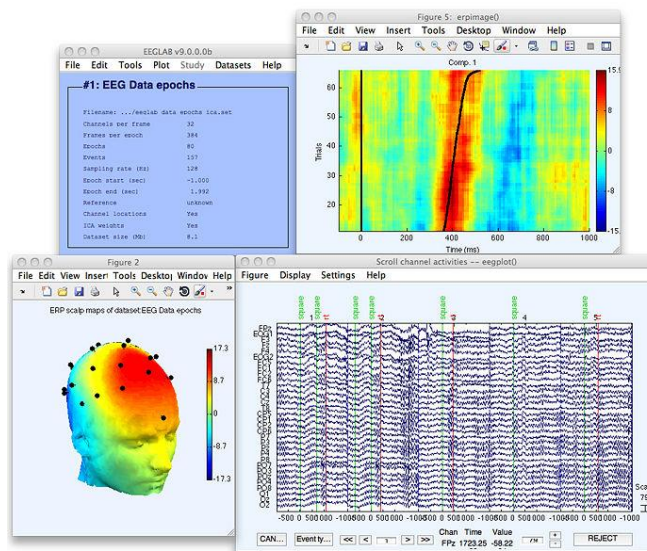


Figura 15: Entorno de la librería EEGLAB de Matlab.

4.2.4. Selección de la Aplicación BCI

En el presente apartado se han expuesto los principales programas de diseño de aplicaciones BCI. A continuación, en la *Tabla 2* se muestra una comparativa entre los programas.

Se evalúan los siguientes apartados:

- En la facilidad de programación e integración de nuevas aplicaciones se evalúa el lenguaje de programación, así como el entorno de programación.
- La integración de las aplicaciones en Linux es un apartado importante. La futura aplicación se compilará en un sistema embebido para conseguir un producto cerrado y funcional.
- Que la aplicación sea OpenSource es un punto positivo puesto que se trata de un tema importante ya que permite crear aplicaciones e integrarlas sin coste adicional de licencias.
- Se dispone de un casco Enobio3G de 8 canales, por lo que es imprescindible que la aplicación sea compatible con el dispositivo.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

- Como se ha visto en el apartado anterior, el paradigma seleccionado es SSVEP. Se valora la aplicación que permita una fácil adaptación e uso del paradigma, así como una amplia documentación en el mismo.
- La comunicación con el robot Stäubli se realiza mediante una serie de funciones programadas en lenguaje C++ de ahí que se valore que la aplicación permita una rápida integración de la librería.

		BAJO	MEDIO	ALTO
Facilidad de programación	OpenVibe			X
	BCI2000		X	
	Matlab + Toolbox		X	
Facilidad de integración de nuevas aplicaciones	OpenVibe			X
	BCI2000		X	
	Matlab + Toolbox		X	
Integración de nuevas aplicaciones en sistema operativo linux.	OpenVibe			X
	BCI2000			X
	Matlab + Toolbox		X	
Aplicaciones OpenSource	OpenVibe			X
	BCI2000			X
	Matlab + Toolbox	X		
Compatibilidad con Enobio3G (Conexión directa)	OpenVibe			X
	BCI2000			X
	Matlab + Toolbox		X	
Compatibilidad con el paradigma seleccionado (SSVEP)	OpenVibe			X
	BCI2000	X		
	Matlab + Toolbox		X	
Documentación respecto al paradigma seleccionado (SSVEP)	OpenVibe			X
	BCI2000	X		
	Matlab + Toolbox	X		
Facil comunicación con el robot Stäubli (Librería en C++)	OpenVibe			X
	BCI2000			X
	Matlab + Toolbox		X	

Tabla 2: Comparación entre aplicaciones.

Tras evaluar las aplicaciones, el programa seleccionado para la realización del proyecto ha sido Openvibe.

La aplicación se encuentra compilada en C++ por lo que permite una fácil y rápida integración de la librería de comunicación con el robot. Se trata de una aplicación OpenSource que puede compilarse en cualquier sistema operativo.

La principal característica que decanta la decisión ha sido la incorporación del paradigma SSVEP entre las aplicaciones DEMO del programa, así como una amplia documentación sobre el funcionamiento del paradigma.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Mediante las aplicaciones DEMO se puede crear aplicaciones nuevas de manera sencilla e integrarlas en la compilación del programa.

Openvibe dispone de un driver en evaluación que permite comunicarse con el casco Enobio3G, pero es un driver 100% funcional.

La aplicación BCI2000 se ha descartado principalmente por su incompatibilidad con el paradigma seleccionado, mientras que Matlab se ha descartado principalmente por ser un programa con licencia y por presentar incompatibilidades con la librería de comunicación con el robot.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

5. DESCRIPCIÓN DETALLADA DE LA SOLUCIÓN

5.1. Componentes y materiales utilizados

5.1.1. Casco Enobio 3G 8 Canales

Se trata de un gorro elástico bluetooth que presenta una serie de agujeros nombrados según el sistema internacional 10-20 [6], que facilitan la colocación de electrodos húmedos y secos.

En el proyecto se ha utilizado la versión de Enobio 8, esto quiere decir que dispone de un convertidor y un amplificador interno apto para ocho canales simultáneos. En la *Figura 16* se observa el gorro junto con el amplificador (naranja), los electrodos, así como el receptor bluetooth USB.

NOTA: La distribución y colocación de los electrodos se explica en el apartado 2.2 del presente documento.



Figura 16: Casco Enobio3G 8 Canales.

El casco dispone de dos modelos de electrodos, electrodos húmedos y electrodos secos (*Figura 17*). Para el presente proyecto se han utilizado electrodos secos debido a su fácil colocación y comodidad para el sujeto.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

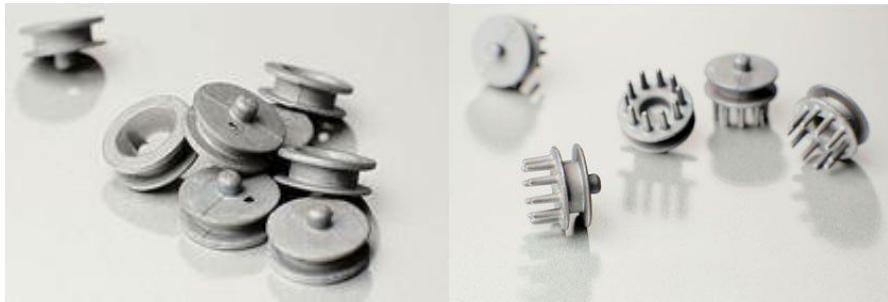


Figura 17: Electrodo húmedo (Izquierda), Electrodo seco (Derecha).

5.1.2. Robot Stäubli TX60

Como proceso a controlar se ha seleccionado un robot industrial de la marca Stäubli y modelo TX60 proporcionado por el Instituto de Automática e Informática Industrial (AI2) de la Universidad Politécnica de Valencia.

Se trata de un modelo de 6 ejes, con 6 grados de libertad para obtener una máxima flexibilidad. Se trata de un robot industrial de carga ligera, es decir, que la carga máxima es de 9 kg en ciertas posiciones. El robot pesa 3.5 Kg y su alcance máximo es de 670 mm. [7] La *Figura 18* muestra el robot descrito.



Figura 18: Robot Stäubli TX60 en el departamento AI2 (UPV).

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

5.1.3. Ordenador completo con teclado ratón y monitor

La aplicación se ha desarrollado en entorno Windows por lo que es necesario un ordenador con sistema operativo Windows, un ratón para facilitar la navegación y la edición, un teclado para enviar comandos e interactuar con la aplicación y lo más importante, un monitor con una frecuencia de refresco de la pantalla de 60 Hz que permita visualizar los estímulos.

En la *Figura 19* se observa el entorno de trabajo compuesto por el ordenador con la aplicación en ejecución, el sujeto con el casco Enobio instalado y con los electrodos en la configuración apropiada y el robot Stäubli TX60 que es el proceso por controlar.

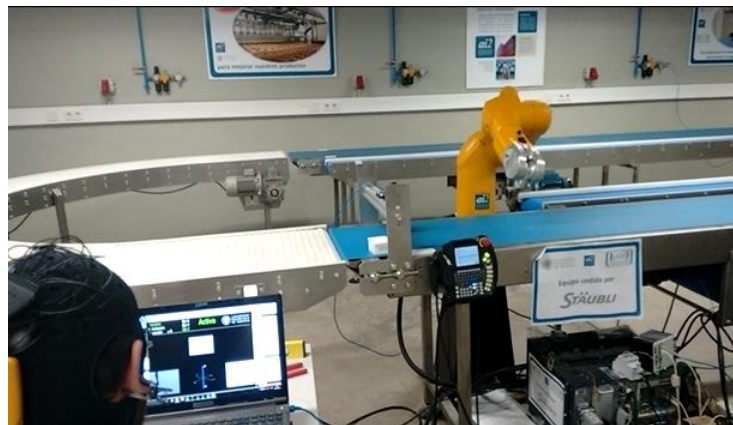


Figura 19: Entorno de trabajo compuesto por la aplicación de control y el robot Stäubli.

5.2. Funcionamiento de la aplicación

La aplicación diseñada para el control se observa en la *Figura 20* donde en el siguiente apartado se comentará su funcionamiento principal, así como la descripción de todos los elementos que aparecen en ella.

Según un estudio realizado en la Universidad de Macau (China) [8], el contraste entre el blanco y el negro provoca una mayor excitación en la retina del sujeto provocando así unos mejores resultados. Afirman que el contraste entre el rojo y el negro en determinadas frecuencias se obtienen resultados similares incluso mejores. Esto lo afirman investigadores brasileños en el XXIV congreso de Ingeniería Biomédica en

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Brasil en 2014 [9]. Asimismo, el contraste entre el blanco y el negro reduce la fatiga tras ensayos realizados por ambos estudios, de ahí que se seleccionara esta combinación de colores para la aplicación.

Para el control del robot se han separado los seis grados de libertad disponibles del robot en posición (control lineal) y orientación (control angular). En la esquina superior derecha se encuentran los dos indicadores que muestran al sujeto el conjunto de grados de libertad del robot que se encuentran habilitados. Más adelante se explicará cómo alternar de modo.

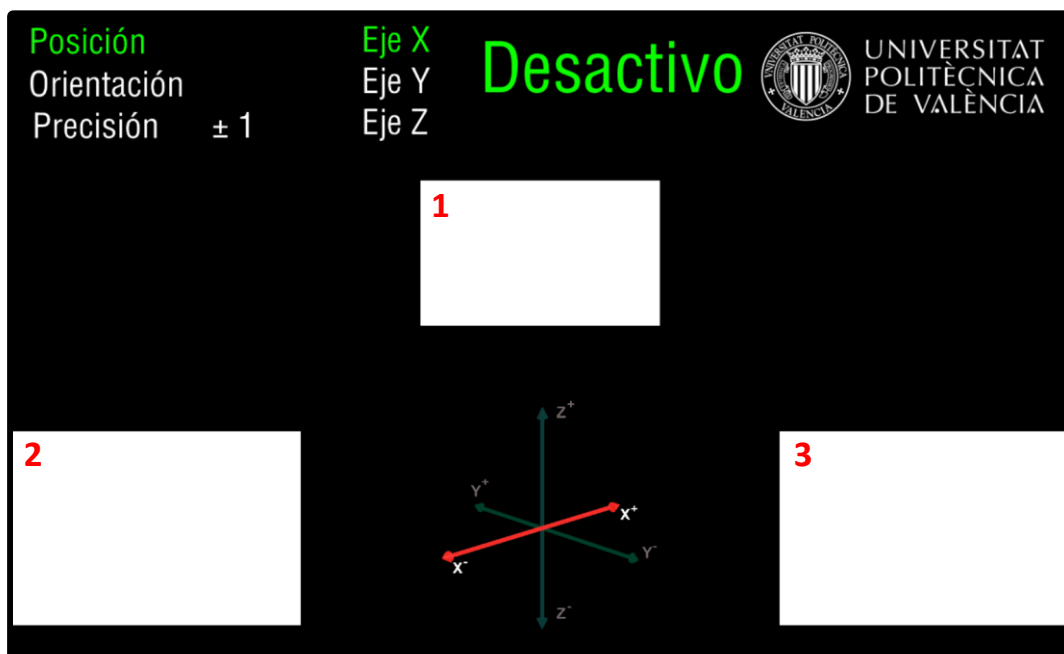


Figura 20: Pantalla principal de la aplicación [ssvep-robot-bci-5-online-test-shooter](#).

Tanto en el centro superior de la aplicación como en los ejes de coordenadas, el sujeto puede visualizar que grado de libertad está controlando, así como el sentido de movimiento. Asimismo, el sujeto puede ver en todo momento si sus órdenes se traspasan al robot mediante la activación o desactivación del mismo.

En el centro de la pantalla se encuentran los tres estímulos visuales necesarios para el control. Ubicados de forma triangular, el rectángulo número 1 corresponde con la acción de control de cambio de grado de libertad. Se encuentra oscilando a una frecuencia de 20Hz y cuando el sujeto visualiza este estímulo, en el caso de encontrarse la posición activada se alternaría entre los tres ejes principales (X, Y y Z) mientras que, en caso de

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

tener la orientación seleccionada, se alternaría entre los tres ángulos principales (Alfa, Beta y Gamma).

Los estímulos inferiores corresponden con incrementos positivos o negativos. El estímulo 2, cuya frecuencia es de 15 Hz, incrementa negativamente mientras que el estímulo 3, programado a una frecuencia de 12 Hz, realiza los incrementos positivamente. Los incrementos van relacionados con la precisión seleccionada y el grado de libertad a controlar, siendo esta magnitud milímetros en caso de movimiento lineal o grados en caso de movimiento angular. El sujeto puede variar los milímetros o grados de desplazamiento del robot con el indicador que se encuentra en la esquina superior izquierda de la aplicación.

Cabe destacar que las frecuencias seleccionadas son 20,15 y 12 Hz, múltiplos de 60Hz que es la frecuencia de refresco del monitor en el cual se ejecuta la aplicación.

5.3. Comunicación

En este apartado se comentará como se ha realizado el conexionado para llegar a realizar el control de un sistema real como es el robot Stäubli.

5.3.1. VRPN

La Red de Periféricos de Realidad Virtual (VRPN) es un conjunto de clases dentro de una librería y un conjunto de servidores que están diseñados para implementar una interfaz transparente entre aplicaciones y un conjunto de dispositivos físicos. En el proyecto se ha utilizado el mismo fin para interconectar la aplicación diseñada en Openvibe con una aplicación externa diseñada en VisualStudio (ver *Figura 21*).

El protocolo VRPN dispone de dos servidores, Analog VRPN Server y Button VRPN Server. La diferencia principal entre ambos queda reflejada en el nombre, el servidor analógico es capaz de recibir una conexión de un cliente analógico y enviar señales analógicas, por ejemplo las señales provenientes del casco Enobio. El servidor Button es simplemente un servidor digital que recibiendo la conexión de un cliente digital puede enviar señales lógicas similar al mecanismo de un botón.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

5.3.2. TCP

Transmission Control Protocol (TCP) es el nombre que recibe uno de los protocolos más importantes de la capa de transporte de un modelo TCP/IP de comunicación en internet.

Se trata de un protocolo de transmisión ordenada de datos entre dos o más ordenadores anfitriones. La principal característica y por lo que lo hace muy importante y muy usado es que garantiza el orden y el recibimiento de los datos enviados, evitando pérdida de datos y transmisiones erróneas. Además, permite el monitoreo del flujo de datos evitando así que se sature la red, así como multiplexar la información recibida de diferentes fuentes. Por último, este protocolo permite comenzar y finalizar la comunicación de manera controlada.

Con el uso del protocolo TCP, las aplicaciones pueden comunicarse en forma segura (gracias al sistema de acuse de recibo del protocolo TCP) independientemente de las capas inferiores. Esto significa que los routers (que funcionan en la capa de Internet) solo tienen que enviar los datos en forma de datagramas, sin preocuparse con el monitoreo de datos porque esta función la cumple la capa de transporte (o más específicamente el protocolo TCP).

Durante una comunicación usando el protocolo TCP, las dos máquinas deben establecer una conexión. La máquina emisora (la que solicita la conexión) se llama cliente, y la máquina receptora se llama servidor. Por eso se conoce como entorno Cliente-Servidor.

Las máquinas de dicho entorno se comunican en modo en línea, es decir, que la comunicación se realiza en ambas direcciones.

5.3.3. Resumen del Conexionado

Openvibe Acquisition Server es el nombre que recibe el programa de Openvibe encargado de la comunicación con periféricos. Mediante un driver es capaz de interactuar con el casco Enobio 3G de 8 canales que se conecta mediante bluetooth a la aplicación. La propia aplicación gestiona las lecturas recibidas, las adapta, les otorga un formato y crea un socket TCP ubicado en el puerto 1024 para enviar los datos.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Openvibe Designer es la aplicación que permite el diseño de aplicaciones en Openvibe así como realizar un procesamiento y tratamiento de la señal recibida y una actuación en consecuencia. Los leídos del socket en el puerto 1024 son procesados y se obtiene una acción de control.

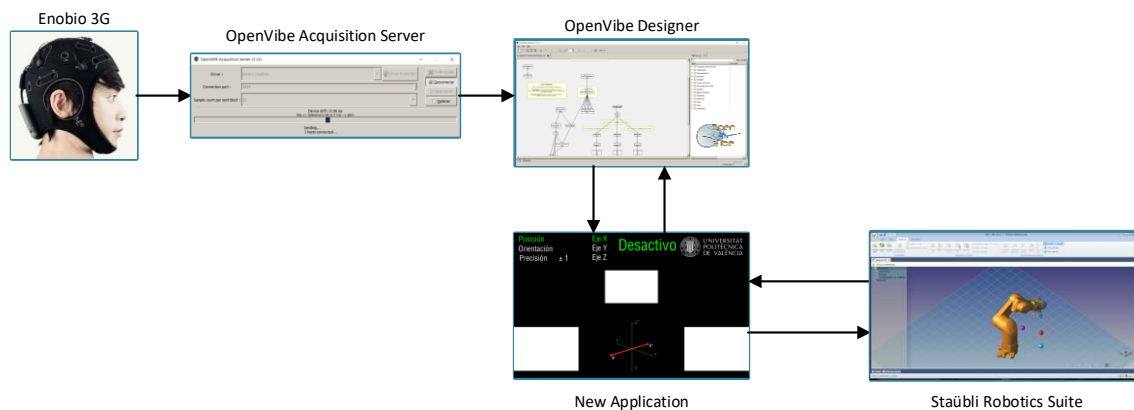


Figura 21: Resumen del conexionado entre aplicaciones.

Se ha programado una aplicación intermedia llamada `openvibe-ssvep-demo-robot` la cual recibe las acciones de control del Openvibe Designer y actúa sobre el robot. La simulación de la aplicación genera una serie de eventos y acciones de control que como se ha comentado en apartados anteriores son valores internos de la propia aplicación. En la *Tabla 3* se observa un extracto de los principales estímulos utilizados en la aplicación, se trata de valores en hexadecimal y se envían mediante el protocolo VRPN. Destacar el `OVTk_SimulationID_ExperimentStart` y `OVTk_SimulationID_ExperimentStop` como señales principales de control de la simulación, así como `Label_00` hasta `Label_03` usados para el control del robot.

TAG	VALOR	DESCRIPTION
<code>OVTk_SimulationID_ExperimentStart</code>	<code>0X00008001</code>	Inicio de la simulación
<code>OVTk_SimulationID_ExperimentStop</code>	<code>0X00008002</code>	Fin de la simulación
<code>OVTk_SimulationID_Label_00</code>	<code>0X00008100</code>	Estímulo para el cambio de eje
<code>OVTk_SimulationID_Label_01</code>	<code>0X00008101</code>	Estímulo para incremento negativo
<code>OVTk_SimulationID_Label_02</code>	<code>0X00008102</code>	Estímulo para incremento positivo

Tabla 3: Tabla de estímulos más utilizados.

Una vez se recibe la acción de control mediante estímulo, la aplicación envía al robot la modificación de la posición u orientación acorde con la precisión y deseos del sujeto. La aplicación se comunica con el robot mediante TCP/IP. Se dispone de una

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

librería proporcionada por el instituto AI2 que facilita el acceso y el control del robot mediante funciones. Cada función tanto de actuación como de información envía una trama de datos cerrada por el socket creado y el robot actúa en consecuencia.

5.4. Controles

En la aplicación no todos los controles los realiza el sujeto con la mente, hay algunos controles de seguridad, que permiten a los usuarios experimentados manipular la aplicación. La *Figura 22* resume el conjunto de botones seleccionado y su principal utilidad.

La aplicación se inicia con la tecla Espacio. Una vez iniciada, el robot por seguridad se encuentra deshabilitado, el programador puede habilitar o deshabilitar los movimientos del robot mediante las teclas pertinentes. Cabe destacar que deshabilitar el robot no corta la comunicación, por lo que es una herramienta útil para programadores a la hora de depurar la respuesta obtenida.

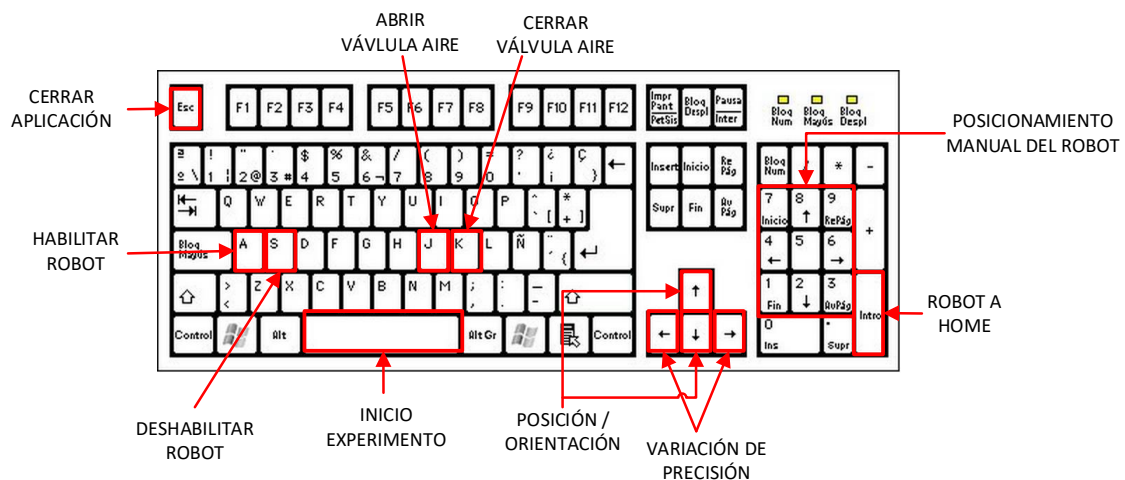


Figura 22: Combinación de teclas programadas y su funcionalidad.

Las teclas J y K permiten la apertura y cierre de una pequeña electroválvula neumática. Gracias a este accionamiento externo se puede por ejemplo dotar al robot de una garra para realizar tareas de pick and place o similares.

Con las flechas se puede navegar por el menú. El usuario puede seleccionar si desea controlar la posición o la orientación del robot, así como la precisión (milímetros o grados de movilidad).

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Finalmente se han programado nueve posiciones finales al robot de forma que mantienen la misma cota de altura y varía su posición en el plano. De esta forma se puede llegar a un objeto con mayor velocidad. La tecla de Enter devuelve el robot a su posición inicial.

5.5. Programación Concurrente

Gracias a la programación concurrente se ha podido realizar el proyecto y cumplir los objetivos.

Los principales problemas que llevaron a la programación concurrente son que el robot necesita continuamente estar comunicando por lo que siempre ha de disponer de un servidor activo, asimismo la aplicación ha de garantizar que los estímulos mantengan la misma frecuencia de parpadeo para evitar falsos positivos e incluso acciones de control erróneas. Como último, la pantalla se encuentra continuamente refrescándose y hay una serie de eventos externos proveniente de teclado que hay que atender y realizar acciones a consecuencia.

Para solventar los problemas la aplicación se ha distribuido según la *Figura 23*:

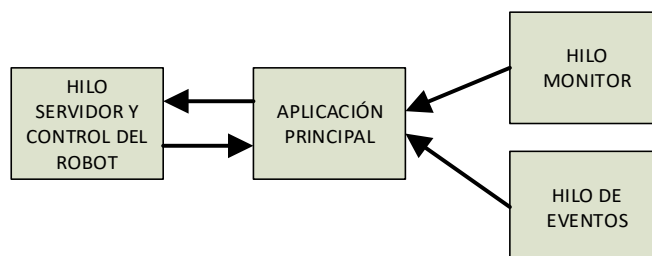


Figura 23: Diagrama de bloques de la distribución de hilos de ejecución

La aplicación principal se ejecuta de forma secuencial y es la encargada de mantener los estímulos oscilando a las frecuencias deseadas. Esta aplicación principal dispone de tres hilos de ejecución que complementan la funcionalidad de la aplicación.

- Hilo servidor y control del robot, en la aplicación recibe el nombre de MoveRobot, es el encargado de realizar, mantener y recuperar la conexión con el robot. Se le ha implementado la funcionalidad del control del robot.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

- Hilo monitor, llamado ControlScreen en la aplicación, es el encargado de refrescar la pantalla cada vez que hay una modificación en ella. Gracias a una serie de mutex y variables condición se condiciona la ejecución del mismo solo cuando hay modificaciones en la pantalla.
- Hilo de eventos o llamado ManualControl, es el encargado de gestionar todos los eventos externos a la aplicación y provenientes de teclado. Como el hilo anterior, se ejecuta cuando hay un evento y eso ha sido posible gracias al uso de mutex y variables condición.

La comunicación entre la aplicación principal y el robot se produce mediante una variable global llamada m_RobotStäubli la cual incluye todas las variables necesarias para la comunicación.

6. JUSTIFICACIÓN DE LA SOLUCIÓN

6.1. Descripción del ensayo

Pasada la simulación es importante para el estudio futuro disponer de una tasa de éxito de la aplicación, así como resultados de fiabilidad. Puesto que se trata del control de un robot a tiempo real esta cuantificación no es viable realizarla por lo que se ha hecho uso de una demo alternativa para cuantificar los resultados.

Se explica en detalle el funcionamiento de la aplicación en el apartado 5 del documento **Manual de Uso**.

La configuración de los electrodos para el ensayo es idéntica al ensayo anterior y la configuración se puede comprobar en el apartado 2 del documento **Manual de Uso**.

Los archivos de configuración, los filtros y clasificadores entrenados son similares, esto quiere decir que el sujeto no ha de volver a entrenarse para realizar el segundo experimento.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

6.2. Resultados obtenidos

Se ha realizado el experimento a cinco sujetos de diferente edad y sexo. Se dispone a comentar los resultados obtenidos en comparación con los valores teóricos calculados. La tabla con los valores se adjunta en el ANEXO I.

6.2.1. Valores Teóricos

En el siguiente apartado se calcularán los valores teóricos que permitirán realizar las estadísticas de éxito. Para ello es necesario conocer que los objetivos se han repartido cada 45° . El orden de aparición de los estímulos, así como su distribución se observa en la *Figura 24*.

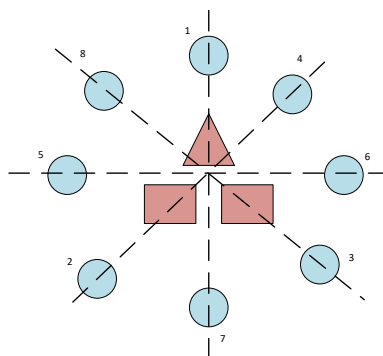


Figura 24: Disposición de los objetivos y orden de aparición.

Se ha configurado el movimiento horario y antihorario de la nave en 3° ya que es un múltiplo de la posición de los objetivos. La *Tabla 4* refleja el movimiento teórico óptimo del ensayo. La nave se encuentra apuntando al primero objetivo por lo que no es necesario movimiento, en cambio para alcanzar el primer estímulo es necesario un movimiento antihorario de 135° que corresponde con 45 pasos teóricos en la aplicación.

Movimiento	Giro AntiHorario	Giro Horario	Movimientos Teóricos
0 - 1	0	0	0
1 - 2	135	0	45
2 - 3	90	0	30
3 - 4	90	0	30
4 - 5	135	0	45
5 - 6	180	0	60
6 - 7	0	90	30
7 - 8	0	135	45
TOTAL			285

Tabla 4: Movimiento teórico mínimo para el ensayo.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Para la realización del ensayo se necesitan 285 movimientos sin contar los disparos.

6.2.2. Valores Experimentales

El primer sujeto evaluado es un varón de 19 años el cual realizó tres repeticiones del experimento.

Se observa en el *Gráfico 1* como el primer experimento el sujeto realiza la tarea en un tiempo de 3.95 minutos. La segunda vez que realiza el experimento, consigue bajar su marca a 3.21 mientras que la última vez, debido al cansancio de las simulaciones el tiempo ha aumentado a 4.30 minutos.

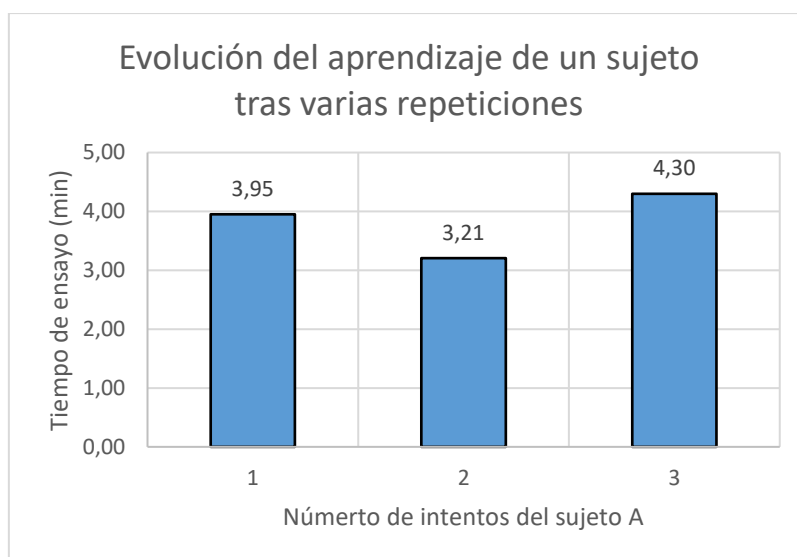


Gráfico 1: Evolución del tiempo de ensayo del sujeto A.

En el *Gráfico 2* se observa el porcentaje de éxito de movimiento calculado respecto al movimiento teórico. Se observa que en el primer ensayo se obtiene un porcentaje de 61.88 %, mientras que el segundo ensayo ha sido más rápido y con mejores resultados, un 78.30%. El último ensayo tras el cansancio el rendimiento ha disminuido del sujeto obteniendo como resultado un 59.45%.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

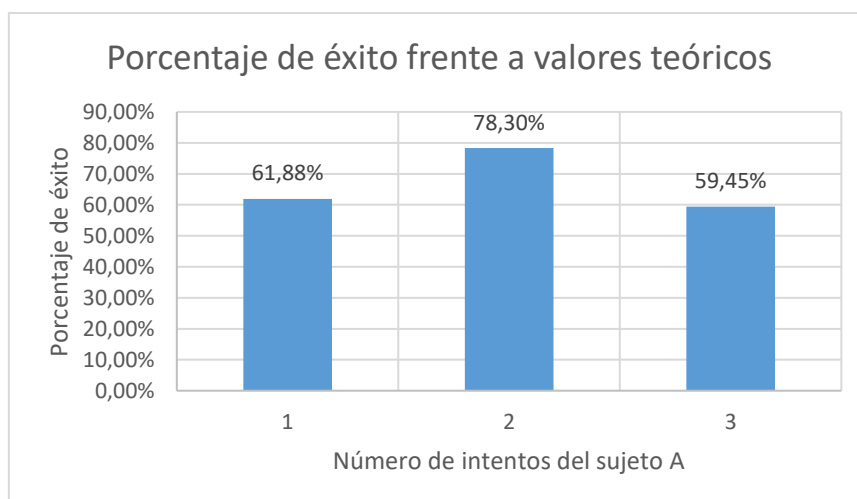


Gráfico 2: Porcentaje de éxito del sujeto A.

El segundo sujeto evaluado es un varón de 30 años el cual realizó cuatro repeticiones del experimento obteniendo muy buenos resultados. Se observa en el *Gráfico 3* que el primer ensayo realizado consigue igualar los tiempos del sujeto anterior llegando incluso a disminuir los 3 minutos.

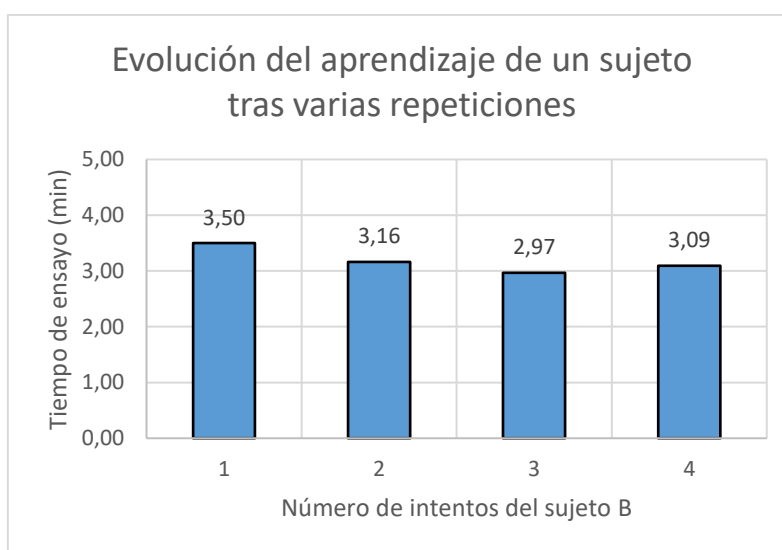


Gráfico 3: Evolución del tiempo de ensayo del sujeto B.

Se observa en la *Gráfico 4* como en el segundo experimento el sujeto supera el 90 % de efectividad con un tiempo que ronda los 3 minutos. El tercer ensayo a pesar de ser más rápido, los movimientos no han sido los mínimos. Cabe destacar que los movimientos se han comparado con la variable teórica obtenida en el apartado anterior

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

por lo que correcciones de la posición de la nave para alcanzar los objetivo no se han contabilizado.

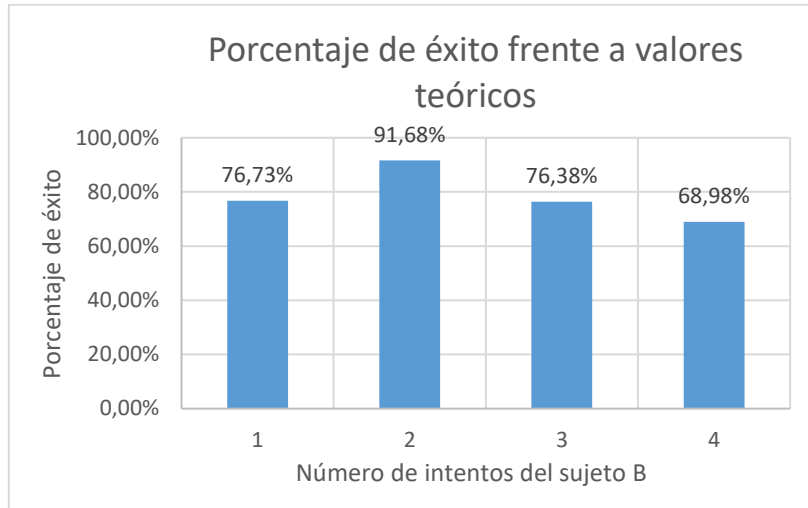


Gráfico 4: Porcentaje de éxito del sujeto B.

El tercer y último sujeto por comentar, es una mujer de 21 años que ha realizado el experimento cinco veces. Este sujeto se caracteriza por la rapidez de sus ensayos como se aprecia en el *Gráfico 5*, así como su alto nivel de mejora tras las repeticiones llegando a realizar el ensayo en menos de dos minutos y medio.

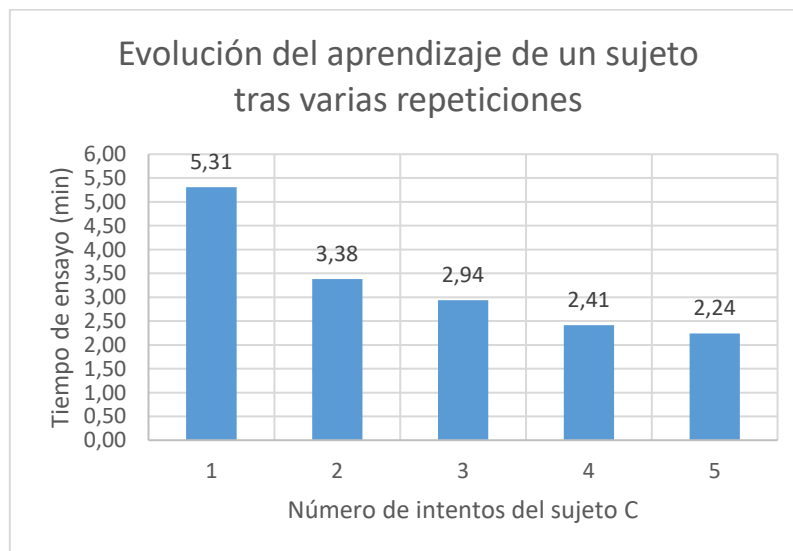


Gráfico 5: Evolución del tiempo de ensayo del sujeto C.

Tras las repeticiones se observa en el *Gráfico 6* como el porcentaje de acierto aumenta consiguiendo valores en torno al 85 %. Esto quiere decir que este sujeto es capaz

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

de realizar ocho tareas en menos de dos minutos y medio con una fiabilidad en torno al 85 %.

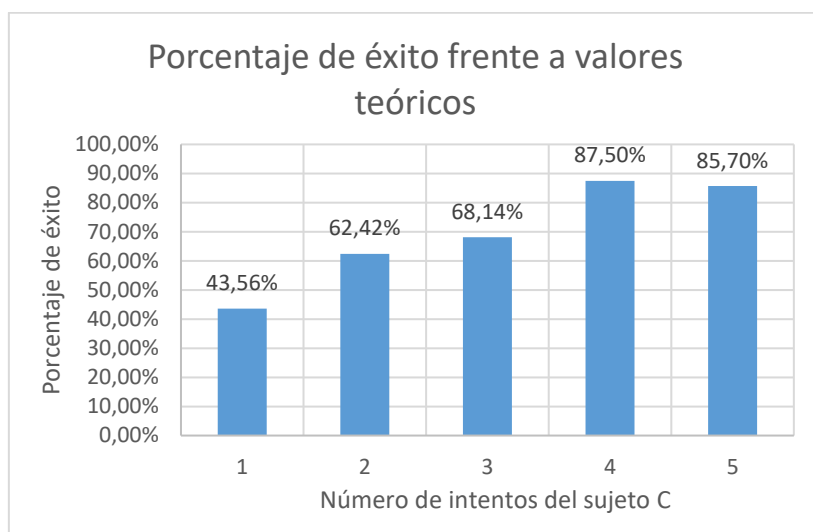


Gráfico 6: Porcentaje de éxito del sujeto C.

A continuación, se realiza una pequeña comparativa de tiempo medio y éxito medio entre los sujetos expuestos anteriormente.

En la *Tabla 5* se observa la comparativa de tiempo medio entre sujetos. La media se ha realizado entre el número de ensayos realizado por cada uno. Se observa como los tres sujetos ensayados consiguen finalizar la simulación en poco más de tres minutos. Esto quiere decir que los sujetos son capaces de eliminar cerca de tres bolas por minuto con la complejidad que requiere el posicionamiento y el disparo (acciones combinadas).

	ENSAYOS					MEDIA
	1	2	3	4	5	
SUJETO A / TIEMPO (min)	3,949	3,205	4,300	-	-	3,818
SUJETO B / TIEMPO (min)	3,501	3,165	2,967	3,093	-	3,181
SUJETO C / TIEMPO (min)	5,310	3,379	2,939	2,414	2,241	3,257

Tabla 5: Comparativa resultados temporales entre sujetos.

En la *Tabla 6* se compara la tasa de éxito media de los sujetos tras realizar las simulaciones. La media, como en el caso anterior, se ha calculado en base a los ensayos realizados. Se observa como los sujetos rondan el 70 % de éxito en la tarea. Destacar el sujeto B que dispone de una media temporal próximo a los tres minutos y su tasa de éxito supera el 75 %.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

	ENSAYOS					MEDIA
	1	2	3	4	5	
SUJETO A / Éxito (%)	61,88%	78,30%	59,45%	-	-	66,54%
SUJETO B / Éxito (%)	76,73%	91,68%	76,38%	68,98%	-	78,44%
SUJETO C / Éxito (%)	43,56%	62,42%	68,14%	87,50%	85,70%	69,46%

Tabla 6: Comparativa de tasa de éxito entre sujetos.

Los ensayos se han realizado sin conocimiento previo de la aplicación, por lo que con constancia y realizando más simulaciones se conseguirían mejores resultados. Un claro ejemplo es la evolución que ha experimentado el sujeto C tras realizar cinco simulaciones.

En la *Gráfico 7* se compara el tiempo empleado frente a los movimientos realizados. Se observa un conjunto de sujetos que necesitan más movimientos para alcanzar un objetivo por lo que necesitan de más tiempo para ello. Hay un conjunto de sujetos que se encuentran en torno a cuatro minutos con un alto número de movimientos y finalmente se observa un conjunto de sujetos que su rendimiento oscila el 80% y consigue acabar el ensayo en torno a los tres minutos y con una media de cien movimientos.

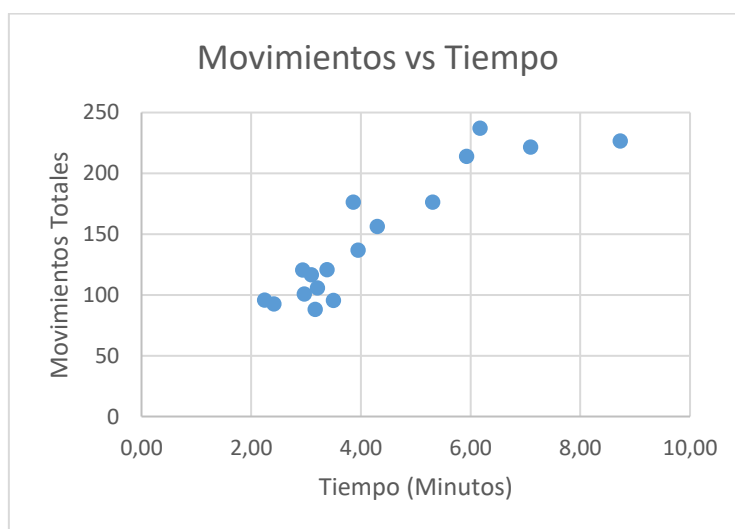


Gráfico 7: Relación tiempo y la media de movimientos totales.

Finalmente, la *Gráfico 8* muestra el conjunto de sujetos con su porcentaje de éxito en el movimiento. Se observa como hay una serie de sujetos que su rendimiento medio es del 80% mientras que otros sujetos que su rendimiento medio oscila el 30%.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

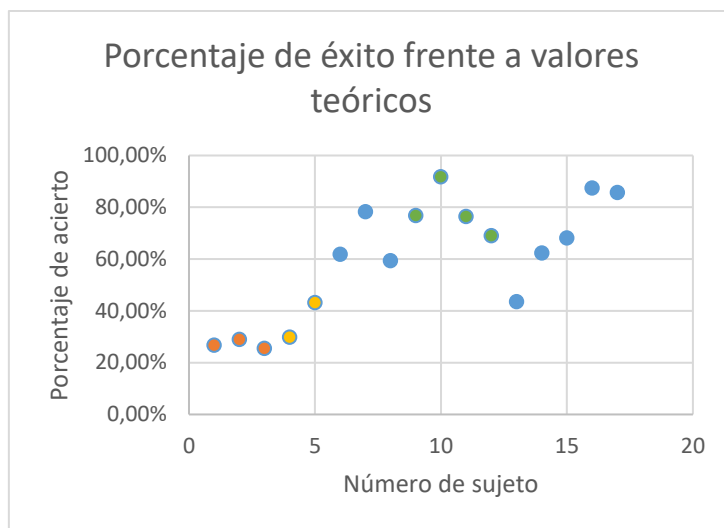


Gráfico 8: Porcentaje de éxito de la población seleccionada.

6.2.3. Encuesta del ensayo

Durante el ensayo se realizaron una serie de preguntas a los sujetos. Todos los sujetos afirman que el ensayo es cansador pero aguantable, se encuentran todos de acuerdo que para un periodo corto es viable mientras que su uso continuado durante horas e incluso días es inaguantable.

Todos coinciden que la aplicación realiza sus deseos salvo en contadas ocasiones que malinterpreta la orden y ejecuta otro comando. Dos de los sujetos cuyos resultados no han sido concluyentes han comentado que la aplicación era lenta y poco funcional.

En cuanto a la colocación de los electrodos y el dolor percibido por los mismos, todos coinciden que la colocación ha sido sencilla y el dolor es aguantable. Existe la posibilidad de utilizar los electrodos húmedos en su lugar para evitar molestias, pero los sujetos prefirieron los secos por evitar el residuo de gel que se queda en el cabello.

Todos los sujetos han recibido una charla de formación previa al ensayo y cuatro de los cinco sujetos han realizado la tarea sin complicaciones salvo uno que ha necesitado guiado para la realización del ensayo. Este sujeto afirma que la interfaz no es intuitiva.

Todos los sujetos salen con un resultado satisfactorio y satisfechos tras haber completado el ensayo.

7. TRABAJO FUTURO

7.1. Aplicación I. Control del brazo robot industrial

7.1.1. Diseño de la aplicación

En cuanto a las mejoras a realizar en la aplicación se incluiría una ventana alternativa de configuración en la propia aplicación que permita modificar todo tipo de parámetros internos sin necesidad de recompilar la aplicación. Interesa controlar los tiempos de exposición, así como los tiempos de refresco de las acciones y comandos enviados al robot.

Es interesante el hecho de disponer de una ventana de configuración de los estímulos a tiempo real, pudiendo así modificar el tamaño y la ubicación de los estímulos para adaptarse a las necesidades de comodidad de cada sujeto.

7.1.2. Funcionalidad

Actualmente la aplicación permite el control de tres grados de libertad y la alternancia se realiza de forma manual. Como mejora futura es intentar reducir al mínimo los comandos manuales existentes incorporando más funcionalidad a la aplicación. Un ejemplo podría ser añadir un nuevo estímulo de frecuencia diferente (10 Hz) que permita alternar entre movimientos lineales o angulares pudiendo así controlar los seis grados de libertad del robot.

La aplicación actual realiza una acción de pick and place de forma manual con el teclado numérico. Una posible mejora sería realizar un sistema híbrido entre SSVEP y P300 y permitir al sujeto realizar una tarea controlada.

7.1.3. Aplicación

Como posibles aplicaciones futuras, se podría realizar un conexionado físico para no depender de un monitor. Es posible hacer oscilar tres o más diodos LED y conseguir los mismos resultados. La aplicación se ha realizado en lenguaje C++ y OpenSource por lo que tiene una fácil integración en sistemas empujados como RaspBerry Pi o BeagleBone.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

En cuanto al sistema híbrido se podría realizar cualquier tipo de aplicación de pick and place. La aplicación actual se ha preparado para que el sujeto pueda jugar al tres en raya accionando comandos y el robot ejecutando las acciones.

7.2. Aplicación II. Comprobación de resultados

La mayoría de sujetos han indicado que les resultaba difícil el ensayo debido al movimiento continuo de la nave y los estímulos. Una posible mejora sería fijar los estímulos y que simplemente rotara una flecha. Esto aumentaría la efectividad de los sujetos, disminuiría el cansancio y la fatiga y disminuiría el tiempo de realización del ensayo.

8. CONCLUSIONES

El campo de la neurociencia se encuentra en continuo cambio. Resultados como los obtenidos en nuevos paradigmas abren la posibilidad de realizar numerosos proyectos y aplicaciones. Las interfaces cerebro-computador aportan una nueva forma de comunicación y de control de aplicaciones externas. Esta tecnología puede llegar a mejorar la vida de personas discapacitadas o con deficiencias motoras puesto que se trata de un complemento que permite realizar funciones básicas sin necesidad de terceros.

En la memoria se ha planteado el uso del paradigma SSVEP para el guiado de un robot industrial basándose en la aplicación openvibe para el análisis y procesamiento de los datos obtenidos. Para ello se ha realizado y gestionado el conexionado apropiado para su funcionalidad.

En cuanto a los resultados, se ha obtenido de media en torno al 75% de éxito en la aplicación y con un tiempo medio de simulación de en torno a los 3 minutos. Esto quiere decir que un sujeto es capaz de realizar acciones diferentes cada 23 segundos y con una fiabilidad del más del 75% con electrodos no invasivos.

Con una investigación más profunda del paradigma se pueden crear aplicaciones con una mayor fiabilidad y una gran aplicabilidad.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

9. BIBLIOGRAFIA

- [1] K. K. Ang, C. Guan, K. S. G. Chua, B. T. Ang, C. Kuah, C. Wang, K. S. Phua, Z. Y. Chin y H. Zhang, «A clinical study of motor imagery-based brain-computer interface».
- [2] J. Meng, S. Zhang, A. Bekyo, J. Olsoe, B. Baxter y B. He, «Noninvasive Electroencephalogram Based Control of a Robotic Arm for Reach and Grasp Tasks».
- [3] C. Loconsole, D. Leonardis, M. Barsotti, A. Frisoli y M. Bergamasco, «A novel BCI-SSVEP based approach for control of walking in Virtual Environment using a Convolutional Neural Network».
- [4] R. Singla, A. Khosla y R. Jha, «Influence of stimuli color on steady-state visual evoked potentials based BCI wheelchair control».
- [5] H. A. LAMTI, M. M. B. KHELIFA, A. M. ALIMI y P. GORCE, «EFFECT OF FATIGUE ON SSVEP DURING VIRTUAL WHEELCHAIR NAVIGATION».
- [6] N. Neuroelectronics, «Enobio Products,» [En línea]. Available: <http://www.neuroelectronics.com/products/enobio/>.
- [7] Staubli, «Robots industriales de 6 ejes TX60,» [En línea]. Available: <https://www.staubli.com/es/robotics/brazos-roboticos/robots-de-carga-ligera/tx60/>.
- [8] T. Cao, F. Wan, P. U. Mak, P.-I. Mak, M. I. Vai y Y. Hu, «Flashing Color on the Performance of SSVEP-based BrainComputer».
- [9] S. M. M. T. F. B. a. A. F. R. M. Tello, «EVALUATION OF DIFFERENT STIMULI COLOR FOR AN SSVEP-BASED».
- [10] C. Vialfa, «CCM Benchmark Group,» 10 Julio 2017. [En línea]. Available: <http://es.ccm.net/contents/281-protocolo-tcp>.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ANEXO I

Se adjuntan la *Tabla 7* con los datos extraídos en el ensayo realizado en septiembre de 2017 a cinco sujetos de distinto sexo y edad con edades comprendidas entre 18 y 30 años.

SUJETO Nº	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
NOMBRE	MARIA	MARIA	MARIA	MARCOS	MARCOS	DIEGO	DIEGO	DIEGO	DANI	DANI	DANI	DANI	SARA	SARA	SARA	SARA	SARA
EDAD	23	23	23	21	21	19	19	19	30	30	30	30	21	21	21	21	21
INTENTOS	1	2	3	1	2	1	2	3	1	2	3	4	1	2	3	4	5
TIEMPO TOTAL (CICLOS)	523719	425684	370374	355609	231709	236911	192302	257970	210035	189885	178039	185592	318599	202739	176346	144848	134441
TIEMPO TOTAL (SEG)	523,72	425,68	370,37	355,61	231,71	236,91	192,30	257,97	210,04	189,89	178,04	185,59	318,60	202,74	176,35	144,85	134,44
TIEMPO TOTAL (MIN)	8,73	7,09	6,17	5,93	3,86	3,95	3,21	4,30	3,50	3,16	2,97	3,09	5,31	3,38	2,94	2,41	2,24
MOV. 1 ESTIMULO	BOLA 1	2	6	40	2	6	8	2	1	2	2	2	1	2	2	2	1
	BOLA 2	42	82	18	29	30	86	7	39	18	15	18	7	10	28	20	22
	BOLA 3	30	29	28	6	34	10	20	83	9	34	10	22	41	22	29	15
	BOLA 4	42	94	21	21	52	54	26	17	26	15	8	18	8	33	15	13
	BOLA 5	29	26	32	158	44	38	18	19	18	22	68	62	25	11	32	32
	BOLA 6	21	31	17	11	23	51	43	125	12	28	19	18	129	21	36	24
	BOLA 7	49	30	63	49	44	30	60	40	12	14	10	8	18	39	59	19
	BOLA 8	23	21	26	24	22	11	32	88	16	30	17	72	33	10	39	44
	TOTAL	238	319	245	300	255	288	208	412	113	160	152	209	265	166	232	171
MEDIA	30	40	31	38	32	36	26	52	14	20	19	26	33	21	29	21	
MOV. 2 ESTIMULO	BOLA 1	0	74	85	10	11	22	11	10	10	11	5	10	16	5	11	10
	BOLA 2	166	149	0	80	79	140	79	73	104	82	81	80	109	73	75	81
	BOLA 3	197	4	148	47	265	68	52	106	50	46	55	63	126	71	60	48
	BOLA 4	170	262	151	67	72	46	65	53	46	57	51	46	47	92	60	51
	BOLA 5	25	77	198	580	288	114	81	87	119	76	148	140	104	111	86	77
	BOLA 6	167	169	121	114	117	30	105	168	0	0	105	103	161	0	18	0
	BOLA 7	77	0	94	212	149	0	36	0	0	0	0	33	27	71	72	12
	BOLA 8	46	3	147	140	141	0	12	46	17	21	0	10	0	0	25	0
	TOTAL	848	738	944	1250	1122	420	441	543	346	293	445	485	590	423	382	304
MEDIA	106	92	118	156	140	53	55	68	43	37	56	61	74	53	48	38	
MOV. 3 ESTIMULO	BOLA 1	0	74	85	0	0	18	0	0	0	0	6	9	0	0	0	0
	BOLA 2	84	76	126	0	0	49	4	0	23	0	0	0	31	0	0	0
	BOLA 3	145	150	101	0	10	21	0	47	0	0	0	0	71	20	0	0
	BOLA 4	119	208	98	9	19	0	10	8	0	0	0	0	0	28	15	0
	BOLA 5	153	0	110	79	0	31	0	0	30	0	67	56	30	27	0	0
	BOLA 6	63	75	21	9	0	129	0	69	106	108	0	0	256	109	130	94
	BOLA 7	130	42	140	65	4	62	89	52	52	53	57	90	78	114	116	68
	BOLA 8	32	90	27	0	0	76	93	120	93	90	84	86	80	78	89	104
	TOTAL	726	715	708	162	33	386	196	296	304	251	208	238	555	376	350	266
MEDIA	91	89	89	20	4	48	25	37	38	31	26	30	69	47	44	33	
MOVIMIENTOS TOTALES	1812	1772	1897	1712	1410	1094	845	1251	763	704	805	932	1410	965	964	741	
MOVIMIENTOS SIN DISPARO	899	830	944	807	660	461	364	479	371	311	373	413	654	457	418	326	
SUMA MEDIA MOVIMIENTOS	227	222	237	214	176	137	106	156	95	88	101	117	176	121	121	93	

Tabla 7: Resultados de los ensayos.



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

DISCA
DEPARTAMENTO DE INFORMÁTICA
DE SISTEMAS Y COMPUTADORES



Departamento de Ingeniería de Sistemas y Automática
Departamento de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia

Trabajo Fin de Máster

Máster Universitario en Automática e Informática Industrial

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

3.- MANUAL DE USO

Autor:

D. Javier Dadone Ravera

Tutor:

D. Eduardo Quiles Cucarella

D. Martín Mellado Arteché

Valencia, septiembre de 2017

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DEL PROYECTO

ÍNDICE DE IMÁGENES.....	2
1. OBJETIVO.....	1
2. CONFIGURACIÓN.....	1
3. ENTRENAMIENTO.....	6
4. SIMULACIÓN.....	10
5. COMPROBACIÓN DE RESULTADOS	12
5.1. Análisis de resultados	13

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DE IMÁGENES

Figura 1: Casco Enobio 8 Canales.....	1
Figura 2: Configuración del casco vía bluetooth en Windows 10.....	2
Figura 3: Ventana de configuración de dispositivos bluetooth	3
Figura 4: Programa Enobio NIC. Colocación de los electrodos	4
Figura 5: Openvibe Acquisition Server. Pantalla principal	4
Figura 6: Openvibe Acquisition Server. Configuración del driver Enobio 3G.....	5
Figura 7: Openvibe Acquisition Server. Ventana de selección de electrodos.....	5
Figura 8: Openvibe Acquisition Server. Configuraciones globales	6
Figura 9: Ventana emergente de abrir escenarios en Openvibe	6
Figura 10: Escenario 1. Configuración de la simulación.....	7
Figura 11: Escenario 2. Adquisición de la señal de entrenamiento.....	8
Figura 12: Escenario 2. Aplicación externa en pantalla principal (Izq), Aplicación externa en funcionamiento (Der).....	9
Figura 13: Aplicación diseñada tras ejecutar ssvep-robot-bci-5-online-test-shooter	10
Figura 14: CMD. Mensajes informativos durante la ejecución de la aplicación.....	11
Figura 15: Aplicación para cuantificar éxito del paradigma SSVEP.	12
Figura 16: Resultados de la simulación. Variable m_ResultTime.	14

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

1. OBJETIVO

El objetivo principal de este documento es dotar al lector de unos conocimientos para entrenar y usar la aplicación, así como técnicas para mejorar la adquisición.

El usuario será capaz de realizar un ensayo completo desde la configuración de electrodos necesaria, el entrenamiento de los clasificadores hasta la prueba de la aplicación y comprobación de resultados.

2. CONFIGURACIÓN

En este apartado se le enseñará al lector a ubicar el casco en la posición óptima para el ensayo, así como colocar los electrodos.

1. Ubicar los electrodos en el casco Enobio según la configuración apropiada. La *Figura 1* muestra el casco y sus principales componentes.

La configuración de electrodos es Cz, O1, O2, PO3, Oz, PO4, Pz y Fz respetando los canales del 1 al 8. Colocar los electrodos secos en el casco de goma según la serigrafía y encender el amplificador.



Figura 1: Casco Enobio 8 Canales

2.- Conectar el Casco Enobio al PC via bluetooth

2.1- Ordenador sin Bluetooth

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Si el ordenador no dispone de bluetooth, el casco Enobio dispone de un conector USB que permite realizar la conexión con el casco. Simplemente conectar el USB y encender el casco Enobio para comunicar.

2.2- Ordenador con Bluetooth incorporado

Para ordenadores con bluetooth incorporado, es necesario añadir el dispositivo en las configuraciones para que se reconozca. Para ello, en un ordenador con Windows 10 se realiza de la siguiente forma:

- 1.- Abrir Configuraciones de Windows 10
- 2.- Clicar sobre Bluetooth y Otros Dispositivos
- 3.- Hacer Click en Agregar Bluetooth u otro dispositivo
- 4.- Seleccionar la opción de dispositivo Bluetooth y esperar a que encuentre el dispositivo con el nombre de NE-ENOBIO8 como se observa en la *Figura 2*.

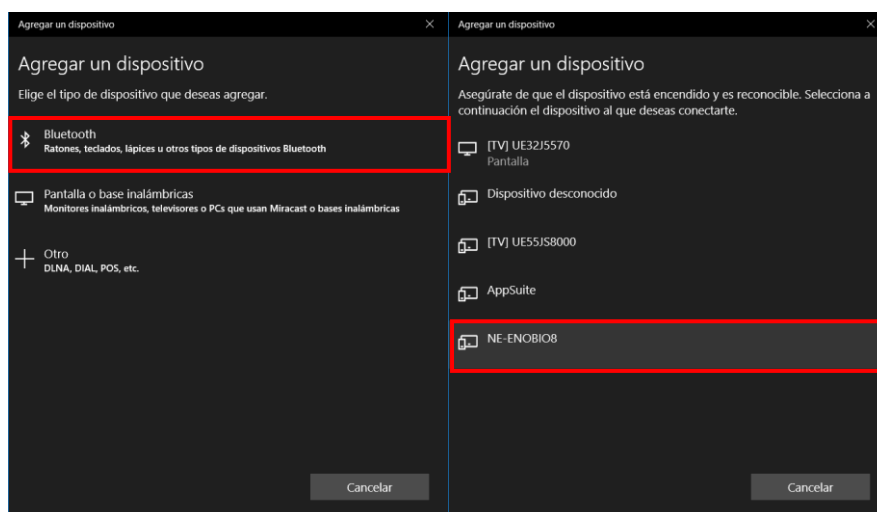


Figura 2: Configuración del casco vía bluetooth en Windows 10

5.- Tras la conjuración, sale una ventana emergente avisando al usuario que se ha configurado correctamente el dispositivo y está listo para usarse. La pantalla de configuración de dispositivo ha de quedar como la *Figura 3*.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

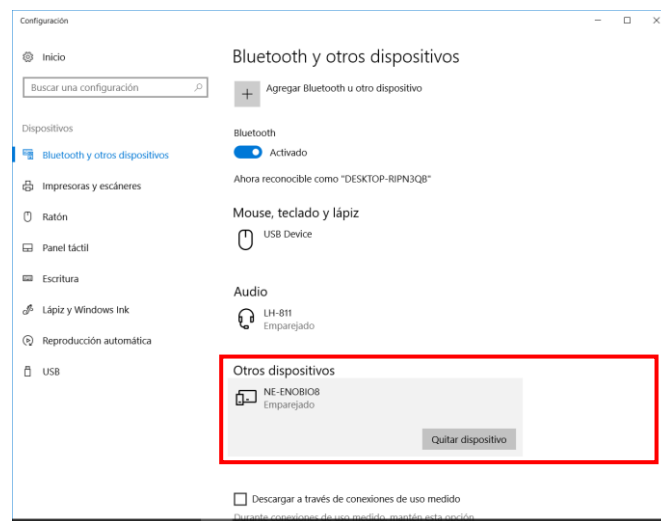


Figura 3: Ventana de configuración de dispositivos bluetooth

3.- Abrir la aplicación de ENOBIO NIC proporcionada por el fabricante

Si el casco está correctamente conectado y configurado, en los ajustes de la derecha se observa el indicador de bluetooth activado y señales de entrada como la batería restante.

4.- Conectar el Casco a la aplicación -> Pestaña de ubicación. (Placement)

En esta ventana (*Figura 4*) es importante configurar la ubicación de los electrodos en el casco, así como comprobar si la señal llega correctamente. Para ello clicar en el botón OFF del submenú Signal Monitoring y los electrodos cambiarán de color entre Rojo, Amarillo y Verde.

- Rojo: La señal es de mala calidad, necesario reajustar la ubicación porque no se reciben los datos correctamente
- Amarillo: La señal es débil, se podría trabajar con ella pero es recomendable una reubicación del electrodo.
- Verde: La señal es perfecta, electrodo configurado y listo para trabajar.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

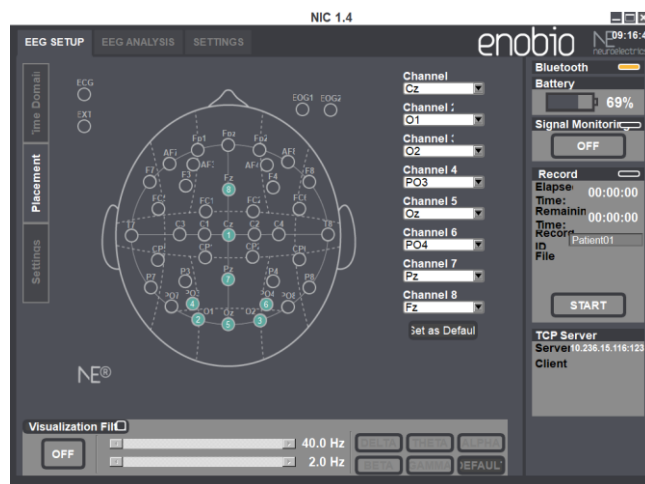


Figura 4: Programa Enobio NIC. Colocación de los electrodos

- 5.- Colocarse el casco según la configuración 10-20 explicada en el apartado 2.3 de la memoria.
- 6.- Configurar todos los electrodos en la aplicación de Enobio de tal forma que todos se encuentren emitiendo en color verde.
- 7.- Cerrar la aplicación de Enobio
- 8.- Abrir la aplicación Openvibe Acquisition Server (*Figura 5*).

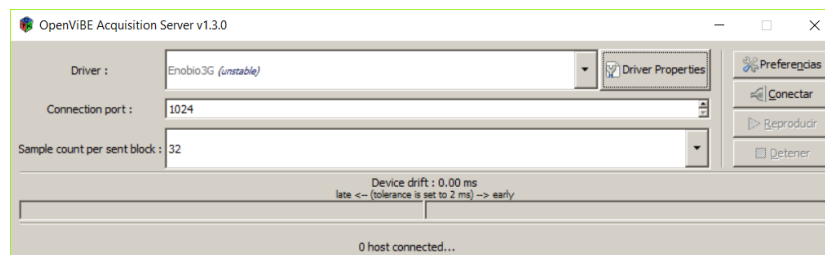


Figura 5: Openvibe Acquisition Server. Pantalla principal

- 8.1. En el desplegable Driver seleccionar “Enobio3G”
- 8.2. Hacer click en el botón Driver Properties.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

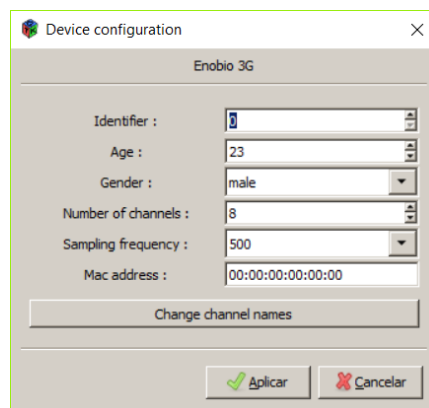


Figura 6: Openvibe Acquisition Server. Configuración del driver Enobio 3G

8.3. Configurar el número de canales, los datos del sujeto y la dirección MAC del casco Enobio como se observa en la *Figura 6*.

8.4. Configurar los canales en el botón “change channel names” tal y como se observa en la *Figura 7*.

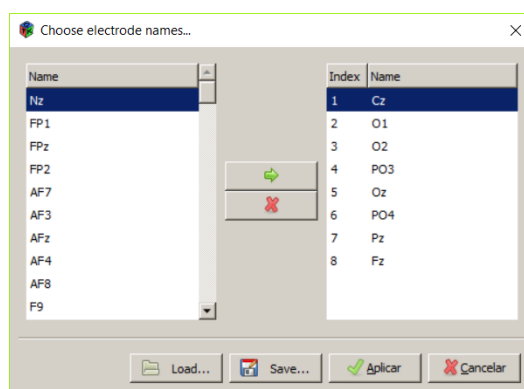


Figura 7: Openvibe Acquisition Server. Ventana de selección de electrodos

8.5. Pulsando en el botón “Preferencias” se puede configurar el puerto TCP entre otras configuraciones que se dejarán por defecto (*Figura 8*).

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

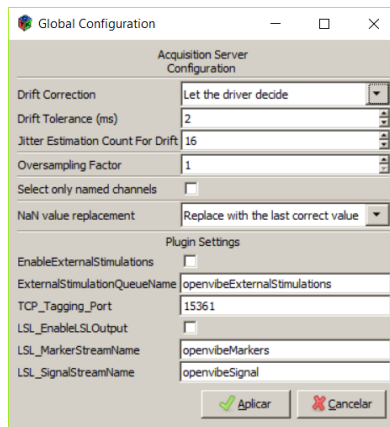


Figura 8: Openvibe Acquisition Server. Configuraciones globales

9.- Hacer click en Conectar y posteriormente en Reproducir para empezar a comunicar.

3. ENTRENAMIENTO

En este apartado, el lector aprenderá a realizar un entrenamiento completo a un sujeto. Para empezar, es necesario abrir el programa de diseño propio de Openvibe llamado Openvibe Designer.

1- Clicar el icono de abrir escenario. La Figura 9 muestra la ventana emergente aparecida. Navegar por la carpeta del proyecto y acudir a la ruta:

`\openvibe-1.3.0-src\dist\share\openvibe\scenarios\bci-examples\ssvep-robot`

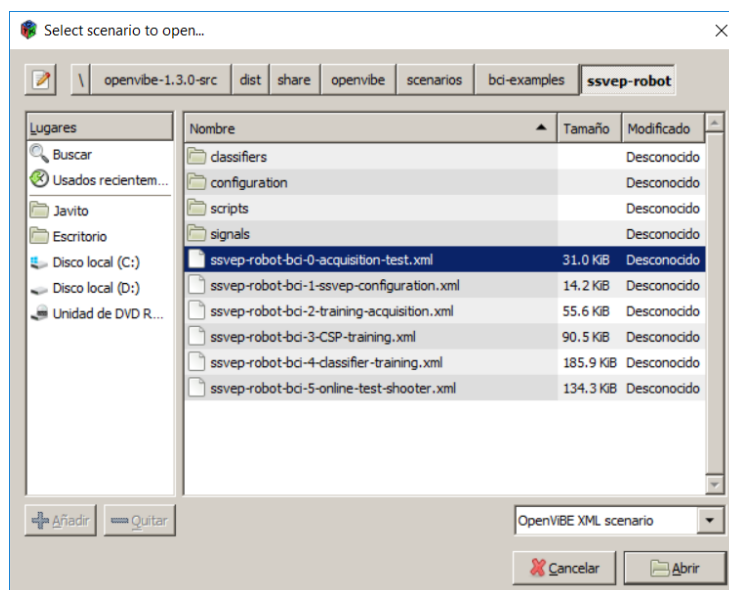


Figura 9: Ventana emergente de abrir escenarios en Openvibe

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

La ruta contiene cuatro carpetas y seis escenarios distintos que se explicarán a continuación.

ssvep-robot-bci-0-acquisition-test -> Se trata de un escenario de prueba. Se puede comprobar la señal recibida y la calidad de la misma. (Paso innecesario tras la comprobación de los electrodos con el programa original de Enobio en el apartado anterior)

ssvep-robot-bci-1-ssvep-configuration -> La *Figura 10* muestra un resumen del escenario en el cual se configuran algunos de los parámetros más importantes de la aplicación. Se observa en el primer desplegable como se selecciona la frecuencia de refresco del monitor, en este caso son 60 Hz.

En el segundo desplegable se puede configurar el color y la frecuencia de los estímulos, así como el color del fondo, entre otros datos.

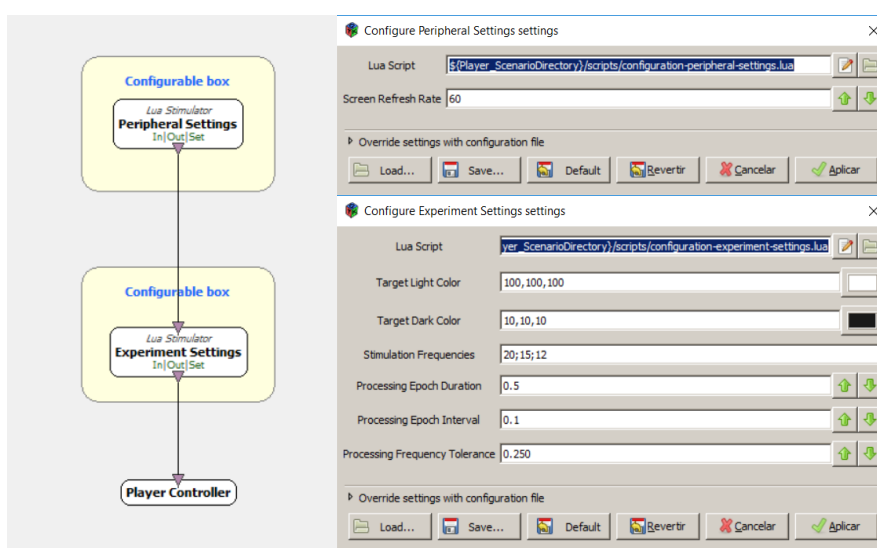


Figura 10: Escenario 1. Configuración de la simulación

ssvep-robot-bci-2-training-acquisition -> Este escenario es el más importante puesto que se trata de la grabación de la señal que se usará para el entrenamiento del filtro espacial y del clasificador (*Figura 11*).

Al abrir el escenario a la derecha se encuentra el comando que inicia la simulación y ejecuta una aplicación externa que en el proyecto se ha modificado. La rama de la izquierda varía los estímulos y controla los tiempos de exposición y reposo.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

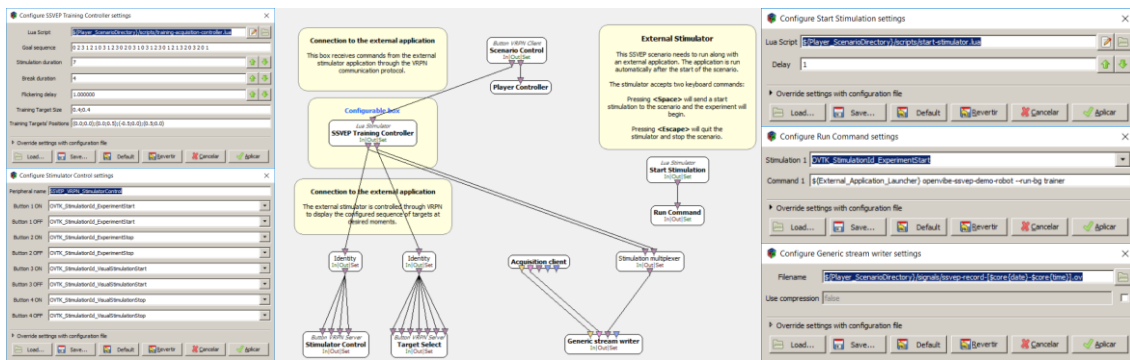


Figura 11: Escenario 2. Adquisición de la señal de entrenamiento

Cuando se inicia la simulación el sujeto visualizará la pantalla que se muestra en *Figura 12* (izquierda) a espera de confirmación de inicio de experimento. En este momento el sujeto ha de cumplir una serie de condiciones para que la lectura se realice bien y se obtengan unos buenos resultados en el futuro.

- El sujeto ha de sentarse en una posición cómoda, evitando movilizar la cabeza durante el ensayo.
- La pantalla ha de estar ubicada a una altura cómoda para facilitar el ensayo. Así mismo la pantalla ha de estar completamente perpendicular al campo visual del sujeto y disponer de una frecuencia de refresco igual a la configurada en el paso previo.
- En la pantalla se muestran cuatro estímulos. Tres de color blanco situados de manera triangular y en el centro del triángulo hay un estímulo que corresponde con el fondo (Su color es gris oscuro).
- Para realizar el ensayo el sujeto visualizará el estímulo parpadeando, en ese momento ha de fijar la mirada al estímulo seleccionado con un triángulo verde *Figura 12* (derecha). Se expondrá al estímulo durante siete segundos (Tiempo configurable).
- Pasado ese tiempo los estímulos dejan de parpadear y se varía el objetivo a mirar. Dispone de cuatro segundos (Tiempo configurable) para descansar la vista, parpadear y tragar saliva.
- Es importante que la cabeza realice el mínimo movimiento para ello la alternancia entre estímulos se realizará de manera visual.
- Aproximadamente el experimento dura siete minutos.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

- Al finalizar, la aplicación se cierra automáticamente.

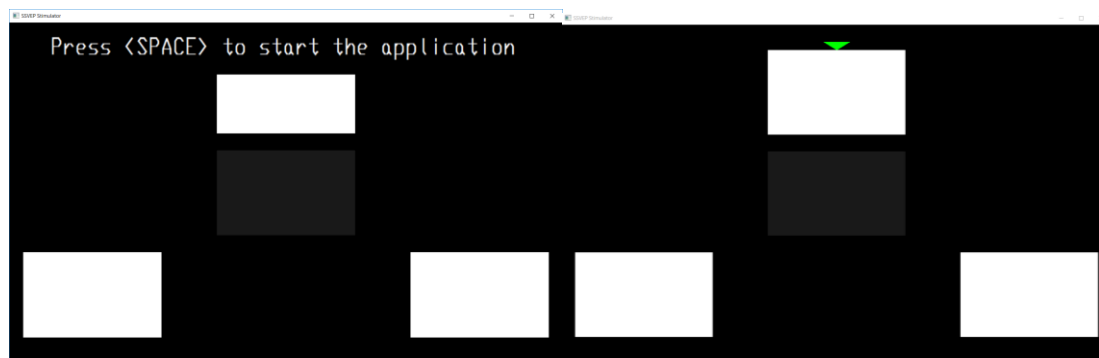


Figura 12: Escenario 2. Aplicación externa en pantalla principal (Izq), Aplicación externa en funcionamiento (Der)

ssvep-robot-bci-3-CSP-training -> Una vez obtenida la grabación de entrenamiento es necesario realizar un entrenamiento del filtro espacial CSP. Para ello es necesario seleccionar el archivo que se ha generado en el escenario anterior y ejecutar la simulación. La duración de esta simulación es igual al tiempo de grabación del archivo por lo que ronda los siete minutos. Se puede aprovechar este tiempo para realizar algún tipo de cuestionario al sujeto.

ssvep-robot-bci-4-classifier-training -> En este escenario se entrena el clasificador, para ello es importante seleccionar el archivo grabado en el escenario 2 así como configurar la ubicación de los filtros espaciales generados en el escenario anterior. El sujeto no ha de realizar ninguna acción. El tiempo de simulación ronda los siete minutos como los escenarios anteriores. Se puede aprovechar este tiempo para realizar algún tipo de cuestionario como la simulación anterior.

En cuanto a las carpetas se encuentran en la raíz (*Figura 9*). Destacar que en la carpeta classifiers se almacenarán de manera automática los clasificadores entrenados para el sujeto en cuestión así como los filtros espaciales CSP.

La carpeta Configuration contiene los archivos de configuración necesarios y generados en el primer escenario ejecutado. La carpeta Scripts contiene los archivos en lenguaje LUA que utiliza el programa Openvibe para gestionar las señales. Finalmente, en la carpeta Signals se encuentran las señales grabadas ordenadas por día y hora de ensayo.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

4. SIMULACIÓN

En este apartado, el lector aprenderá a realizar una simulación completa a un sujeto. Para empezar, es necesario abrir el programa de diseño propio de Openvibe llamado Openvibe Designer y en la misma carpeta del apartado (*Figura 9*) anterior ejecutar los archivos restantes.

Para la simulación es necesario tener abierto y configurado el entorno de Robot Stäubli Suite tal y como se comenta en el **Manual de Instalación**, Apartado 5.

Una vez configurado y el simulador del robot en marcha, se procede a realizar el ensayo de funcionamiento del paradigma. Para ello es necesario ejecutar el escenario siguiente:

ssvep-robot-bci-5-online-test-shooter -> En este escenario el sujeto puede probar y controlar un proceso real como es el robot Stäubli. Para ello es necesario comprobar que los clasificadores y los filtros espaciales que se ejecutan pertenecen al sujeto evaluado. El tiempo de simulación es ilimitado. La señal en este escenario se almacena para su posterior estudio si fuera necesario.

Tras ejecutar el escenario se inicia la aplicación diseñada como se observa en la *Figura 13*. La lógica y el funcionamiento detallado se explica en el documento **Memoria**, Apartado 5.2.



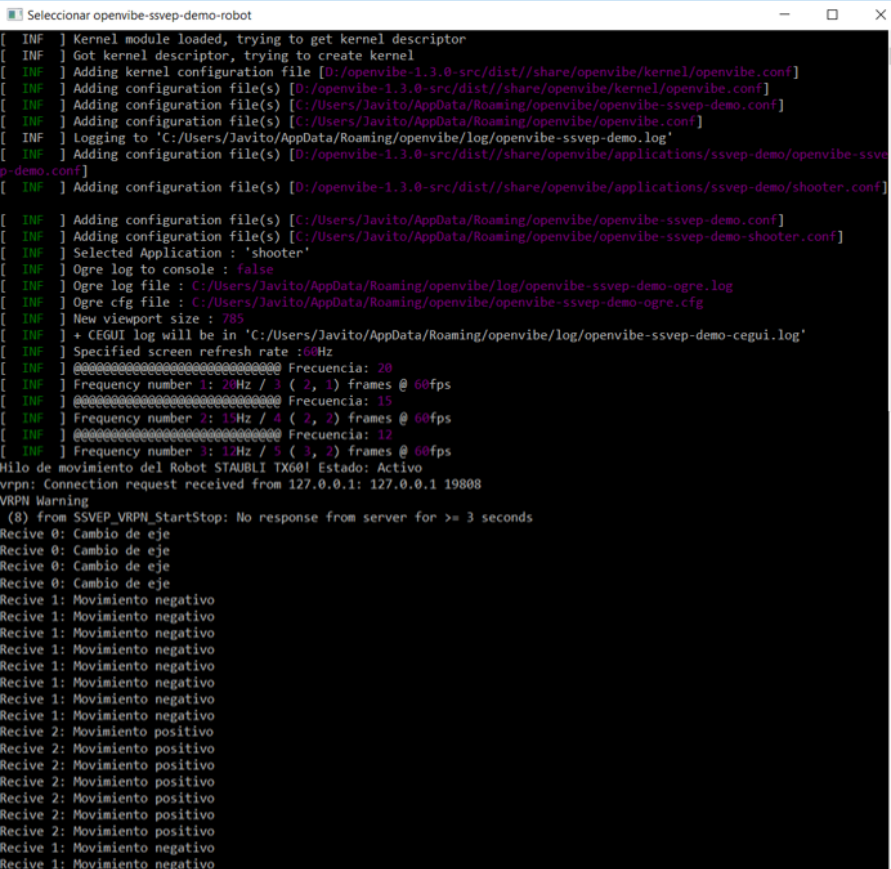
Figura 13: Aplicación diseñada tras ejecutar sssvep-robot-bci-5-online-test-shooter

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

De forma paralela a la pantalla principal se ejecuta una consola en la cual el programador puede comprobar en todo momento en funcionamiento de la aplicación. Se ha de realizar un pequeño test de comprobación de las señales, en el cual el sujeto ha de mirar fijamente a los tres estímulos en el orden que le diga el especialista. Se ha de comprobar que la señal correspondiente al estímulo indicado aparece en la consola.

En la *Figura 14* se observa que tras un periodo de inicialización que se emiten mensajes informativos, aparece el primer mensaje de interés y avisa que el hilo de robot se encuentra en ejecución, pero el robot inicialmente se encuentra desactivado. Tras el mensaje se habilita la conexión entre las aplicaciones mediante VRPN como se ha comentado en el apartado 5.3 del documento de **Memoria**.

Pasada las configuraciones, se realiza una prueba de los estímulos. Se observa como aparecen las órdenes recibidas y van alternando dependiendo del estímulo visualizado.



```
Selecionar openvibe-ssvep-demo-robot
[INF ] Kernel module loaded, trying to get kernel descriptor
[INF ] Got kernel descriptor, trying to create kernel
[INF ] Adding kernel configuration file [D:/openvibe-1.3.0-src/dist//share/openvibe/kernel/openvibe.conf]
[INF ] Adding configuration file(s) [D:/openvibe-1.3.0-src/dist//share/openvibe/kernel/openvibe.conf]
[INF ] Adding configuration file(s) [C:/Users/Javito/AppData/Roaming/openvibe/openvibe-ssvep-demo.conf]
[INF ] Adding configuration file(s) [C:/Users/Javito/AppData/Roaming/openvibe/openvibe.conf]
[INF ] Logging to 'C:/Users/Javito/AppData/Roaming/openvibe/log/openvibe-ssvep-demo.log'
[INF ] Adding configuration file(s) [D:/openvibe-1.3.0-src/dist//share/openvibe/applications/ssvep-demo/openvibe-ssvep-demo.conf]
[INF ] Adding configuration file(s) [D:/openvibe-1.3.0-src/dist//share/openvibe/applications/ssvep-demo/shooter.conf]
[INF ] Adding configuration file(s) [C:/Users/Javito/AppData/Roaming/openvibe/openvibe-ssvep-demo.conf]
[INF ] Adding configuration file(s) [C:/Users/Javito/AppData/Roaming/openvibe/openvibe-ssvep-demo-shooter.conf]
[INF ] Selected Application : 'shooter'
[INF ] Ogre log to console : false
[INF ] Ogre log file : C:/Users/Javito/AppData/Roaming/openvibe/log/openvibe-ssvep-demo-ogre.log
[INF ] Ogre cfg file : C:/Users/Javito/AppData/Roaming/openvibe/openvibe-ssvep-demo-ogre.cfg
[INF ] New viewport size : 785
[INF ] + CEGUI log will be in 'C:/Users/Javito/AppData/Roaming/openvibe/log/openvibe-ssvep-demo-cegui.log'
[INF ] Specified screen refresh rate :60Hz
[INF ] @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ Frecuencia: 20
[INF ] Frequency number 1: 20Hz / 3 ( 2, 1) frames @ 60fps
[INF ] @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ Frecuencia: 15
[INF ] Frequency number 2: 15Hz / 4 ( 2, 2) frames @ 60fps
[INF ] @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ Frecuencia: 12
[INF ] Frequency number 3: 12Hz / 5 ( 3, 2) frames @ 60fps
Hilo de movimiento del Robot STAUPLI TX60! Estado: Activo
vrpn: Connection request received from 127.0.0.1: 127.0.0.1 19808
VRPN Warning
(8) from SSVEP_VRPN_StartStop: No response from server for >= 3 seconds
Recive 0: Cambio de eje
Recive 0: Cambio de eje
Recive 0: Cambio de eje
Recive 0: Cambio de eje
Recive 1: Movimiento negativo
Recive 1: Movimiento negativo
Recive 1: Movimiento negativo
Recive 1: Movimiento negativo
Recive 1: Movimiento negativo
Recive 1: Movimiento negativo
Recive 2: Movimiento positivo
Recive 2: Movimiento positivo
Recive 2: Movimiento positivo
Recive 2: Movimiento positivo
Recive 2: Movimiento positivo
Recive 1: Movimiento negativo
Recive 1: Movimiento negativo
```

Figura 14: CMD. Mensajes informativos durante la ejecución de la aplicación

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Para habilitar el movimiento el robot o realizar los controles manuales por teclado, visualizar el apartado 5.4 Controles del documento **Memoria**.

Mediante esta simulación el sujeto puede controlar el Robot Stäubli tanto en la simulación como el robot real puesto que el comportamiento es idéntico.

5. COMPROBACIÓN DE RESULTADOS

Para este apartado es necesario ejecutar una demo de openvibe modificada para cuantificar tiempos. Se encuentra en la siguiente ruta:

D:\openvibe-1.3.0-src\dist\share\openvibe\scenarios\bci-examples\ssvep

ssvep-bci-1-online-test-shooter -> Se trata de un escenario de prueba del paradigma SSVEP, la funcionalidad es similar a la aplicación de control del robot pero esta posee otra funcionalidad. La *Figura 15* muestra la ventana emergente tras iniciar la simulación. Los datos de los clasificadores y filtros espaciales se obtienen automáticamente de las simulaciones anteriores.

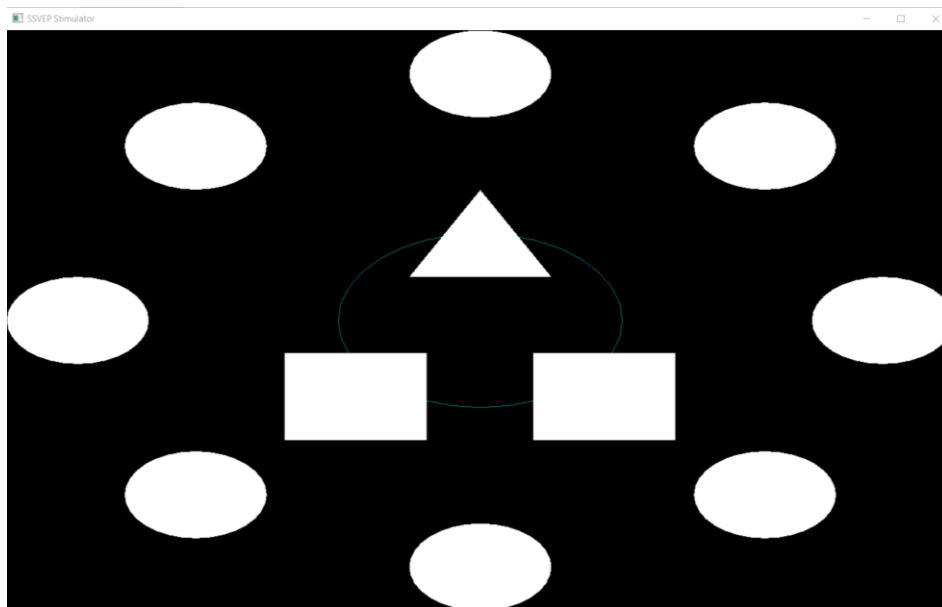


Figura 15: Aplicación para cuantificar éxito del paradigma SSVEP.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

El objetivo de la aplicación es eliminar todos los objetivos y para ello se dispone de una nave compuesta por dos cuadrados y un triángulo. Los cuadrados permiten el movimiento horario o antihorario de la nave y el triángulo permite disparar.

Los círculos u objetivos aparecen de manera secuencial, es decir, cuando se elimina un objetivo desaparece y aparece el nuevo. Hay un total de ocho objetivos a eliminar y se computariza el tiempo fin del experimento, así como el número de movimientos necesarios para eliminar los objetivos.

Para la realización del ensayo es necesario colocar el casco y los electrodos tal y como se explica en el apartado 2 de este documento.

La configuración del experimento se mantiene del primer ensayo realizado por lo que la gama de colores, la frecuencia, así como el tiempo de exposición se mantienen constantes. Para variar la configuración es necesario modificar y ejecutar el archivo ssvep-robot-bci-1-ssvep-configuration como se explica en el apartado 3 de este documento.

Los objetivos se han colocado cada 45° por lo que hay un total de ocho objetivos a eliminar para culminar la tarea con éxito. Se ha fijado el giro de la nave a tres grados por paso puesto que se trata de un valor múltiplo de las posiciones de los objetivos.

5.1. Análisis de resultados

Los resultados de la simulación se almacenan en una variable interna de programa llamada `m_ResultTime` la cual incorpora una variable que contabiliza el tiempo final de ejecución de la aplicación (en ciclos de reloj) y tres vectores de ocho componentes que almacenan el número de movimientos necesarios de cada uno de los estímulos para alcanzar cada uno de los ocho objetivos. La variable también incorpora variables auxiliares como posición del vector.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

	BOLA 1	BOLA 2	BOLA 3	BOLA 4	BOLA 5	BOLA 6	BOLA 7	BOLA 8
m_Stimulus1	2	20	29	15	32	36	59	39
m_Stimulus2	11	75	60	60	86	18	72	0
m_Stimulus3	0	0	0	15	0	130	116	89
m_FinalTime	176346							

Figura 16: Resultados de la simulación. Variable m_ResultTime.

Una vez obtenido los resultados de la simulación para obtener el tiempo en segundos es necesario dividir el tiempo obtenido entre el número de ciclos de reloj por segundo. La librería *time.h* dispone de una variable que contabiliza los ciclos de reloj por segundo cuyo nombre es `CLOCKS_PER_SEC`, la cual tiene un valor de 1000. En la *Figura 16* se observa un tiempo final de 176346 ciclos de reloj que aplicando la conversión son 176.346 segundos que corresponden a 2.94 minutos.

Para obtener los movimientos reales de la aplicación, se ha calibrado el sistema realizando un giro de 360° y comprobando el número de movimientos obtenidos. De manera teórica, el número de movimientos es de 120 ($360^\circ/3^\circ$ por paso) mientras que reales el valor obtenido ha sido de 210 unidades. Con estos dos valores se realiza una conversión que se aplicará posteriormente a los resultados obtenidos del vector. El factor obtenido es de 0.57.

En la *Figura 16* se observan los vectores tras la simulación. El primer vector (`m_Stimulus1`) corresponde con los disparos realizados. Este vector para los futuros cálculos realizados de efectividad no se tiene en cuenta.

El vector `m_Stimulus2` corresponde con el movimiento antihorario mientras que el vector `m_Stimulus3` corresponde con el horario. Se observa como las primeras posiciones del segundo vector tienen un valor alto ya que la distribución de los objetivos hace que el movimiento más rápido sea el antihorario. Se observa también como para el cuarto objetivo (cuarta posición del vector), el sujeto ha tenido que rectificar 15 unidades en el sentido horario para alcanzarlo. Finalmente, los últimos tres objetivos se realizan en sentido horario de ahí el aumento de valor en el vector tres.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

6. BIBLIOGRAFÍA

- [1] Ibonnet, «SSVEP: Steady-State Visual-Evoked Potentials,» 1 Septiembre 2011. [En línea]. Available: <http://openvibe.inria.fr/steady-state-visual-evoked-potentials/>.
- [2] Ibonnet, «Acquisition Server,» 30 Agosto 2011. [En línea]. Available: <http://openvibe.inria.fr/acquisition-server/>.
- [3] Ibonnet, «Designer Overview,» 30 Agosto 2011. [En línea]. Available: <http://openvibe.inria.fr/designer/>.
- [4] N. Neuroelectrics, «NIC,» [En línea]. Available: <http://www.neuroelectrics.com/products/software/nic/>.



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

DISCA
DEPARTAMENTO DE INFORMÁTICA
DE SISTEMAS Y COMPUTADORES



Departamento de Ingeniería de Sistemas y Automática
Departamento de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia

Trabajo Fin de Máster

Máster Universitario en Automática e Informática Industrial

**DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE
LA ADQUISICIÓN DE SEÑALES EEG**

4.- MANUAL DEL PROGRAMADOR

Autor:

D. Javier Dadone Ravera

Tutor:

D. Eduardo Quiles Cucarella

D. Martín Mellado Arteché

Valencia, septiembre de 2017

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DEL DOCUMENTO

ÍNDICE DE IMÁGENES	2
1. OBJETIVO.....	1
2. INTRODUCCIÓN.....	1
3. PREPARACIÓN DEL COMPUTADOR	2
4. INSTALACIÓN DE OPENVIBE.....	2
4.1. Descarga de la aplicación.....	2
4.2. Instalación de dependencias	3
4.3. Nuevas aplicaciones	4
4.3.1. Compilación de la aplicación	5
4.3.1.1. Compilación e instalación tipo 1	5
4.3.1.2. Compilación e instalación tipo 2.....	5
4.3.2. Resultado de la compilación e instalación.....	6
5. PREPARACIÓN DEL ENTORNO STÄUBLI ROBOTICS SUITE	6
6. APLICACIÓN DE VISUAL STUDIO	11
6.1. Primer inicio de la aplicación.....	11
6.2. Insertar Objetos	11
6.3. Otorgar propiedades a los objetos	12
6.5. Programación concurrente.....	14
6.5.1. Control de imágenes y textos.....	14
6.5.2. Control de acciones manuales	14
6.6. Codificación de los estímulos	15
6.7. Código de comunicaciones.....	15
6.8. Estructura de datos	17
6.8.1. Modificación de los ejes.....	17
6.8.2. Modificación del sentido de movimiento.....	18
6.8.3. Movimiento del robot.....	19
7. CONCLUSIONES	20
8. BIBLIOGRAFIA.....	20

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DE IMÁGENES

Figura 1: Menú informativo de los programas utilizados: Visual Studio(Derecha), Stäubli Robot (Izquierda)	2
Figura 2: Página web de Openvibe.....	2
Figura 3: Estructura de carpetas del programa Openvibe V1.3.0.....	3
Figura 4: Pantalla de instalación de dependencias necesarias para instalar Openvibe.....	4
Figura 5: Configuración de dependencias adicionales en Visual Studio	5
Figura 6: Openvibe Designer. Pantalla principal.....	6
Figura 7: Mensaje de advertencia, Modo demostración activado	7
Figura 8: Configuración de la celda del Robot.....	7
Figura 9: Pasos para arrancar el emulador.....	8
Figura 10: Pasos para arrancar la simulación	8
Figura 11: Emulador CS8C, Pantalla principal	9
Figura 12: Emulador CS8C, Pantalla tras configuración	9
Figura 13: Entorno de Stäubli Robotic Suite con el robot en comunicación	10
Figura 14: Ventana emergente de OGRE Engine Rendering Setup.....	11
Figura 15: Ventana emergente de error en Visual Studio	13

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

1. OBJETIVO

El objetivo principal de este documento es dotar al lector de unos conocimientos para instalar y compilar la aplicación de Openvibe así como técnicas para el diseño de nuevas aplicaciones.

El usuario será capaz de realizar un ensayo de la aplicación configurando previamente el periférico bluetooth Enobio 3G y la aplicación de simulación de robots Stäubli Robotics Suite.

Finalmente, el usuario dispondrá de los conocimientos básicos para añadir pequeñas modificaciones al proyecto, así como nociones para navegar por el mismo.

2. INTRODUCCIÓN

Openvibe es un software libre desarrollado por INRIA con el propósito de diseñar, testear y utilizar interfaces cerebro-computador. Se trata de una librería en C++ la cual permite adquirir señales, filtrarlas y acondicionarlas para posteriormente realizar un clasificado o visualización de las mismas en tiempo real. En el presente documento se explicará con detalle la metodología de instalación y la puesta a punto del programa incorporando las mejoras descritas en la memoria.

VisualStudio es un entorno de desarrollo integrado para sistemas operativos Windows. Permite un desarrollo de aplicaciones en numerosos lenguajes de programación. En el presente documento se explicará con detalle la metodología de compilación del proyecto openvibe así como ejemplo del código programado que servirá al lector para entender y poder realizar pequeñas modificaciones futuras en el código.

Stäubli Robotics Suite se trata de un programa para aplicaciones robóticas de desarrollo y mantenimiento. Dispone de una aplicación de programación propia, así como la visualización de los movimientos y trayectorias del robot en un escenario 3D. Es necesaria licencia para la programación y configuración de un nuevo robot que, proporcionada por el AI2, se ha configurado y preparado el entorno para poder realizar simulaciones offline sin necesidad de disponer de una licencia. En el presente documento se explicará al lector como configurar el programa y ejecutar una aplicación interna para permitir la comunicación TCP/IP entre el robot y una aplicación externa.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

3. PREPARACIÓN DEL COMPUTADOR

Previo a la instalación es necesario preparar el computador instalando los programas necesarios. Es necesario disponer de la herramienta Visual Studio y Stäubli Robotics Suite (*Figura 1*).

Para el proyecto se ha utilizado Visual Studio Ultimate 2013 con licencia proporcionada por la UPV y Stäubli Robotics Suite 2016 con licencia proporcionada por el AI2 (UPV)



Figura 1: Menú informativo de los programas utilizados: Visual Studio(Derecha), Stäubli Robot (Izquierda)

4. INSTALACIÓN DE OPENVIBE

4.1. Descarga de la aplicación

Para la instalación de openvibe es necesario descargar los archivos de código fuente proporcionados en la página web (*Figura 2*).

1. Acceder a la página web de OpenVibe: <http://openvibe.inria.fr/>
2. Seleccionar la pestaña “Downloads”
3. Hacer click en “Get latest sources from GIT”

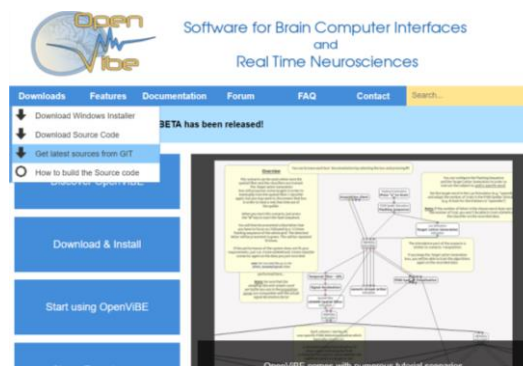


Figura 2: Página web de Openvibe

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

NOTA: La última versión de openvibe disponible en la web puede no ser la misma a la utilizada en el proyecto.

La versión de openvibe utilizada para el desarrollo del proyecto es la V1.3.0 que se adjunta, pero las actualizaciones de versión disponen una distribución de carpetas similar a la *Figura 3*.

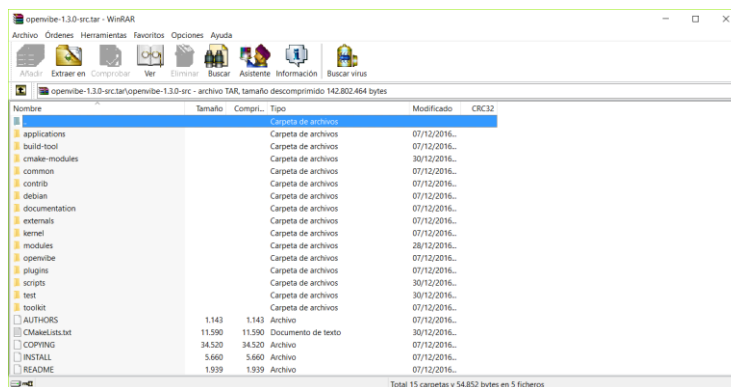


Figura 3: Estructura de carpetas del programa Openvibe V1.3.0

Es recomendable extraer la carpeta en una ruta corta puesto que será la carpeta de trabajo durante el proyecto. La carpeta en el presente proyecto se ubica en la raíz de un disco auxiliar con ruta D:\.

Una de las carpetas más importantes para el desarrollo del proyecto es la carpeta “applications”. En ella se encuentra el código fuente de todas las aplicaciones de Openvibe. La aplicación diseñada en este proyecto ha de incluirse en esa carpeta como se indicará a continuación.

Destacar de las numerosas carpetas, la carpeta “scripts”. En ella se encuentran archivos ejecutables que permiten generar proyectos para VisualStudio, Instalar las dependencias necesarias e incluso compilar la aplicación.

4.2. Instalación de dependencias

En este apartado se procede a instalar las dependencias necesarias para la compilación de Openvibe en Windows. Para instalarlas es necesario:

1. Acceder a la carpeta “Scripts” mencionada en el apartado anterior.
2. Ejecutar el archivo “win32-install_dependencies.exe”

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

NOTA: No es necesario especificar ruta de instalación puesto que se instala en la propia carpeta del programa.

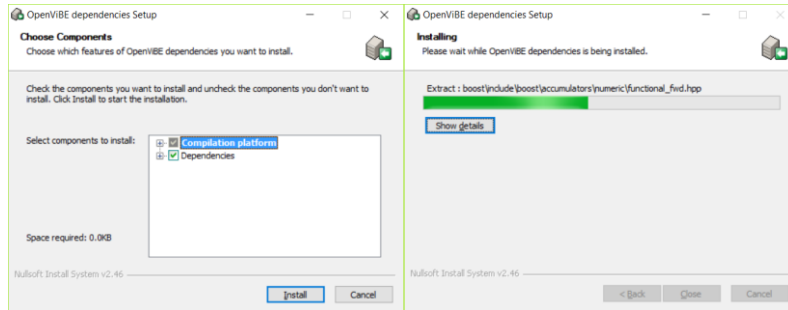


Figura 4: Pantalla de instalación de dependencias necesarias para instalar Openvibe

3. Una vez acabada la instalación comprobar que en la raíz del programa se dispone de una nueva carpeta llamada “dependencies” (Necesaria para la futura compilación de la aplicación).

4.3. Nuevas aplicaciones

Se han añadido creado una nueva aplicación que como se ha comentado en la memoria, conecta Openvibe con la aplicación Stäubli Robotics Suite. Para ello es necesario compilar la nueva aplicación y añadirla al programa original de Openvibe.

1. Las nuevas aplicaciones creadas en este proyecto se encuentran subidas en un repositorio en GitHub cuyo enlace es:

<https://github.com/Vitodad/New-OpenVibe.git>

NOTA: Se adjuntan las modificaciones en un archivo comprimido .zip

2. Para añadir las nuevas aplicaciones es necesario eliminar la carpeta “applications” de la raíz de la aplicación original y copiar la nueva carpeta descargada.
3. Comprobar en la ruta `\openvibe-1.3.0-src\applications\demos\` que se encuentran las carpetas `ssvep-demo` y `ssvep-demo-robot` que han sido las modificadas.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

4.3.1. Compilación de la aplicación

Hay dos formas de compilar e instalar la aplicación, se recomienda utilizar la primera salvo que sea necesario realizar algún tipo de modificación en el código que la segunda es la óptima.

4.3.1.1. Compilación e instalación tipo 1

1. Acceder a la carpeta “scripts” en la raíz del proyecto.
2. Ejecutar el archivo “win32-build.cmd”

NOTA: El archivo comprueba que se dispone de un compilador apropiado, se generan los Makefiles y se instala la aplicación

4.3.1.2. Compilación e instalación tipo 2

1. Acceder a la carpeta “scripts” en la raíz del proyecto.
2. Necesario instalar previamente las dependencias como se ha expuesto en 4.2 de este documento.
3. Ejecutar el archivo “win32-generate-vc-proj.cmd”

NOTA: Se ha generado una nueva carpeta llamada “local-tmp” que contiene el proyecto generado.

4. Ejecutar el archivo “win32-launch-vc.cmd” en la misma carpeta.
5. Una vez iniciado el proyecto, hacer clic derecho en el proyecto “openvibe-ssvep-demo-robot” -> Propiedades
6. En la pestaña “Configuration properties” -> “Linker” -> ”Input” -> ”Additional Dependencies” añadir la librería pthreadCV2.lib como se observa en la *Figura 5*.

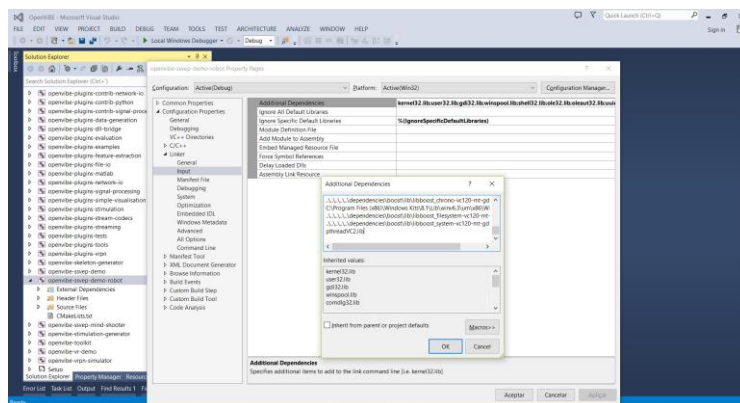


Figura 5: Configuración de dependencias adicionales en Visual Studio

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

7. Aplicar los cambios y aceptar.
8. En la pestaña Build, clicar en Build Solution.
9. Para instalar la aplicación, hacer clic derecho en el proyecto INSTALL y seleccionar Build.

4.3.2. Resultado de la compilación e instalación

Tras realizar el compilado y la instalación de la aplicación de Openvibe modificada, en la carpeta raíz se ha creado una nueva carpeta llamada “dist”. En ella se encuentra la aplicación instalada.

Para comprobar que la instalación se ha realizado con éxito, acudir a la ruta “\dist\” y ejecutar el archivo “openvibe-designer.cmd”. Ha de ejecutar el programa de diseño de aplicaciones de openvibe y se asemeja a la *Figura 6*.

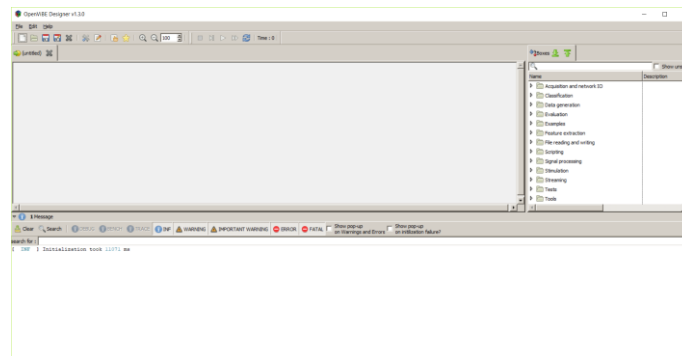


Figura 6: Openvibe Designer. Pantalla principal

5. PREPARACIÓN DEL ENTORNO STÄUBLI ROBOTICS SUITE

En este apartado se procede a configurar el entorno de trabajo del programa destinado a la simulación del robot, para ello es necesario cumplimentar los siguientes pasos:

1. Descargar el archivo comprimido del repositorio de GitHub.

<https://github.com/Vitodad/Staubli-Cell.git>

NOTA: Si adjunta el archivo comprimido en los anexos del proyecto.

2. Descomprimir el archivo y pegar la carpeta llamada TX60 en la siguiente ruta:
“Lugar de instalación”\Stäubli\SRS

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

NOTA: La ruta es importante puesto que en caso contrario el programa no iniciará el proyecto de forma correcta.

La primera vez que se ejecuta el programa es necesario generar el proyecto con una licencia válida. Una vez configurado y guardado los cambios, se permite simular la celda de trabajo sin necesidad de licencia pero las modificaciones ya no están permitidas.

3. Iniciar el programa Stäubli Robotics Suite y abrir el archivo TX60.cell ubicado en la ruta del paso anterior.

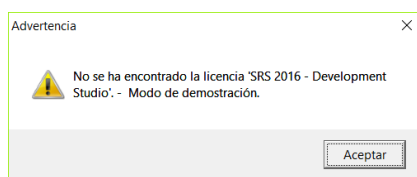


Figura 7: Mensaje de advertencia, Modo demostración activado

4. Aceptar la advertencia que comunica que no se dispone de licencia y por tanto el programa se encuentra en modo demostración, suficiente para simular el robot.

NOTA: La configuración cargada de la celda se observa en la Figura 8, importante la versión del controlador para poder seguir con la configuración (s7.3.1):

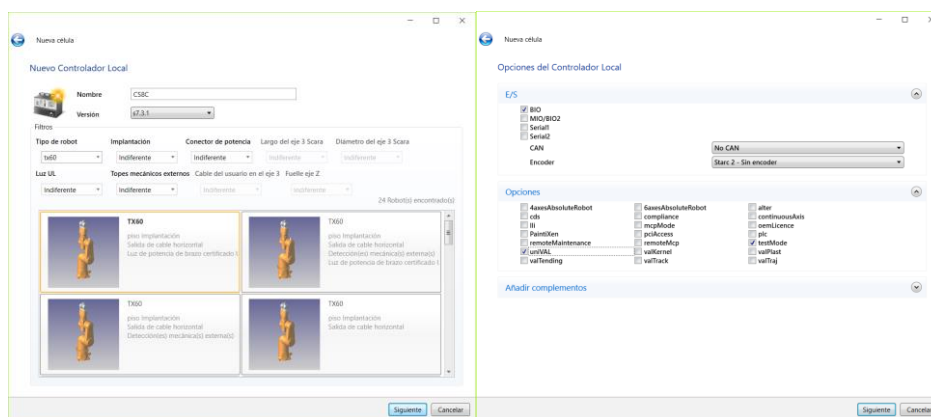


Figura 8: Configuración de la celda del Robot.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

5. Arrancar el emulador tal como se indica en la *Figura 9*.

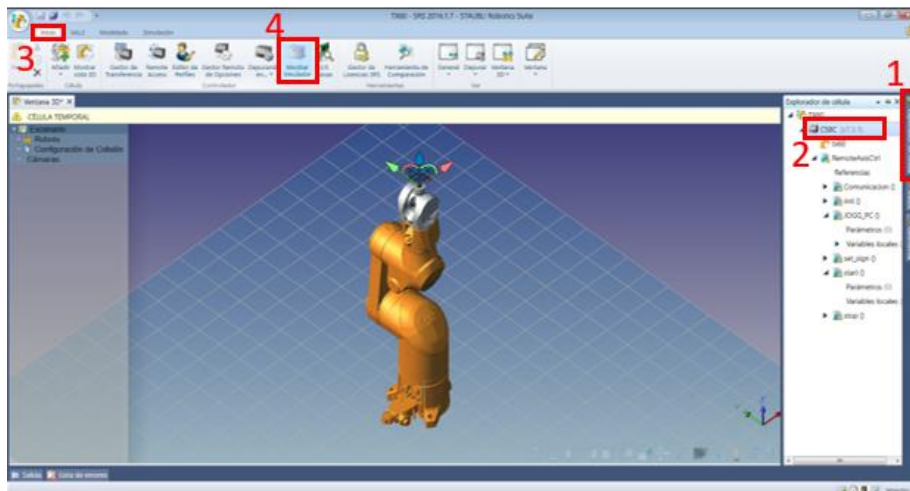


Figura 9: Pasos para arrancar el emulador

6. Empezar la simulación del robot siguiendo los pasos indicados en la *Figura 10*.

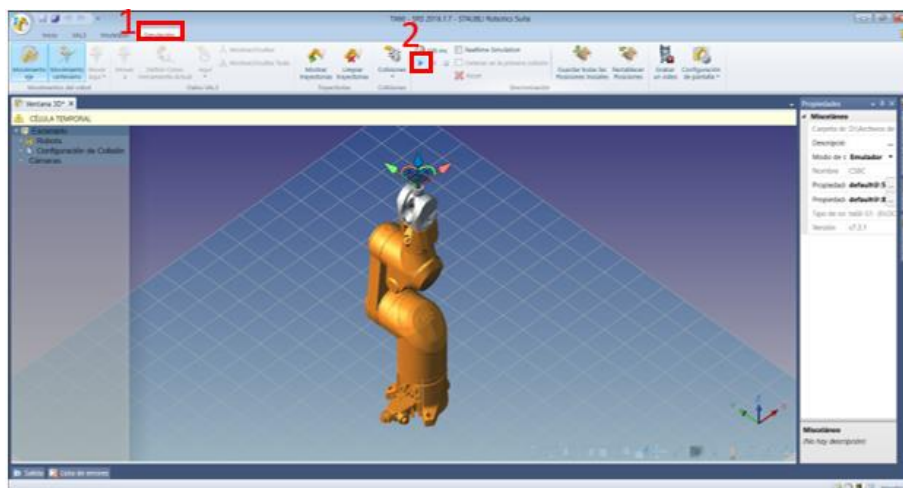


Figura 10: Pasos para arrancar la simulación

7. Configurar las comunicaciones. Para ello es necesario ejecutar el programa que viene incluido en el archivo comprimido cargado en el paso 2. Necesario usar el emulador.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

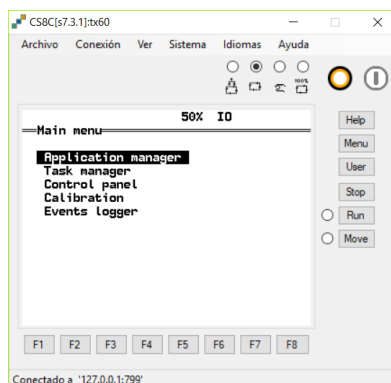


Figura 11: Emulador CS8C, Pantalla principal

8. Realizar la siguiente secuencia con las flechas de teclado:
 - Flecha Derecha -> Flecha Derecha -> Flecha Derecha -> Flecha Abajo -> Tecla F8

9. El resultado ha de ser tal como la *Figura 12* en la cual aparece cargada la aplicación con fecha y hora.

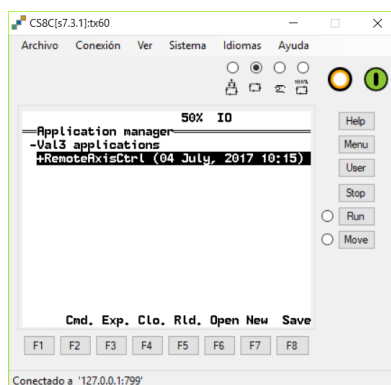


Figura 12: Emulador CS8C, Pantalla tras configuración

10. Clicar el punto blanco (Esquina superior derecha) del emulador hasta que se el robot se encuentre en modo automático (Posición de la *Figura 13*) y encender el emulador (botón verde).

NOTA: Ha de encontrarse el led de Move parpadeando y el led Run apagado.

11. Pulsar la tecla Run y a continuación la tecla F8.

NOTA: Cuando se llega a este paso, el led de Run se encuentra activo mientras que el led de Move se encuentra en intermitencia.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

12. Clicar el Botón de Move, observe que ambos leds se encuentran fijos y activos.

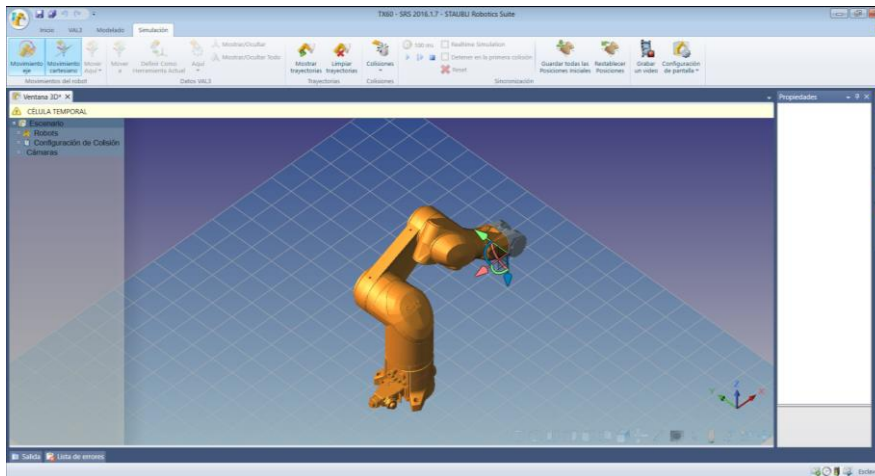


Figura 13: Entorno de Stäubli Robotic Suite con el robot en comunicación

NOTA: Una vez configurado el robot ha inicializado el programa de comunicación y está listo para establecer una comunicación TCP/IP. Observe que el robot se ha ubicado en la posición de Home, listo para comenzar el experimento.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

6. APLICACIÓN DE VISUAL STUDIO

6.1. Primer inicio de la aplicación

Cuando se ejecuta por primera vez la aplicación, se le pregunta al usuario por la versión de OGRE 3G a utilizar para el renderizado y se genera en la ruta C:\Users\Javier\AppData\Roaming\openvibe los archivos temporales de configuración y ejecución. (OGRE, 2017)



Figura 14: Ventana emergente de OGRE Engine Rendering Setup

6.2. Insertar Objetos

Se ha creado una clase llamada CBasicPainter que incluye funciones que permiten dibujar geometrías básicas como un rectángulo, un triángulo y un círculo.

Como ejemplo ilustrativo, para la creación de los estímulos para el proyecto se han utilizado tres rectángulos de distinto tamaño. En el archivo ovassvepRobotCommand.cpp hay una clase llamada CRobotControl que es la encargada de dibujar los estímulos que posteriormente y como se ha visto, sirven para controlar el robot.

En la línea 40 y 41 se definen los lados del robot mediante la inicialización de un vector que proporciona la librería OGRE 3D con valores constantes.

```
39
40   Ogre::RealRect l_oWingRectangle = { -rRadius , rRadius , rRadius, -rRadius};
41   Ogre::RealRect l_oWingRectangle2 = { -rRadius*0.75f, rRadius*0.75f, rRadius*0.75f, -rRadius*0.75f };
42
```

Posteriormente, a partir de la línea 46 se declara el objeto. Destacar que es necesario crear un nodo que contenga los objetos. En este caso como el fin es que los objetos oscilen a una cierta frecuencia, se dibujarán dos rectángulos idénticos de colores

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

distintos y se asignaran al mismo nodo. Finalmente, en la línea 56 se posiciona en el espacio en coordenadas cartesianas.

```
45
46     l_poDrawnObjectNode = m_poRobotNode->createChildSceneNode();
47
48     l_poDrawnObject = l_poPainter->paintRectangle(l_oWingRectangle2, l_oLightColour);
49     l_poDrawnObject->setVisible(true);
50     l_poDrawnObjectNode->attachObject(l_poDrawnObject);
51
52     l_poDrawnObject = l_poPainter->paintRectangle(l_oWingRectangle2, l_oDarkColour);
53     l_poDrawnObject->setVisible(false);
54     l_poDrawnObjectNode->attachObject(l_poDrawnObject);
55
56     l_poDrawnObjectNode->setPosition(0.0f, 0.25f, 0.0f);
57
```

6.3. Otorgar propiedades a los objetos

El nodo que se ha creado en el apartado anterior de por sí solo no realiza ninguna acción. Para conseguir que oscile a una cierta frecuencia, se ha creado una clase llamada CSSVEPFlickeringObject que permite alternar la visibilidad de los rectángulos.

```
57
58     m_poChangeAxes = new CSSVEPFlickeringObject(l_poDrawnObjectNode, (*pFrequencies)[1].first, (*pFrequencies)[1].second);
59
```

En la línea 58 se crea una variable global para destacar el nuevo objeto. Se observa como los argumentos de la función son los nodos y las frecuencias de oscilación.

6.4. Añadir textos e imágenes en la aplicación

Hay dos formas de añadir texto. En la clase CBasicPainter anterior se encuentra la función PaintText la cual es necesario introducirle el color, un vector de posición en cartesianas y el color.

En el proyecto se han utilizado imágenes de textos de 472 x 118 pixeles. Como ventaja principal es que se puede jugar con la propiedad de visible tratándose de un objeto más en la pantalla.

Para introducir una imagen o texto y tomando como ejemplo la aplicación en concreto, la definición se encuentra en la clase CShooterApplication.

```
237
238     m_poTextX = m_poGUIWindowManager->createWindow("TaharezLook/StaticImage", "TextX");
239     m_poTextX->setPosition(CEGUI::UVector2(cegui_absdim(350.0f), cegui_absdim(10.0f)));
240     m_poTextX->setSize(CEGUI::USize(CEGUI::UDim(0.0f, 236.f), CEGUI::UDim(0.0f, 59.f)));
241
```

En las líneas 238 hasta la 240 se crea una ventana y se nombra. Se le otorga una posición, en este caso mediante coordenadas cartesianas absolutas y se le otorga una

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

dimensión. Es importante que la dimensión sea múltiplo al tamaño en pixeles de la imagen para que no sufra distorsiones.

```
414
415     m_poSheet->addChild(m_poTextX);
416     CEGUI::ImageManager::getSingleton().addFromFile("TextX", "TextX.png");
417     m_poTextX->setProperty("Image", "TextX");
418
```

Acto seguido, de la línea 415 a la 417 se asigna la imagen como hija de la ventana creada anteriormente. Es en este momento donde se carga el archivo “Imagen”.png.

```
589     m_poTextX->setProperty("FrameEnabled", "False");
590     m_poTextX->setProperty("BackgroundEnabled", "False");
591     m_poTextX->setVisible(false);
592
```

Finalmente, se le asignan las propiedades iniciales que dispone el objeto como la visibilidad por ejemplo (Línea 591).

Para añadir cualquier imagen al proyecto, se procede de la misma forma siendo necesaria la creación de tags globales de la clase CEGUI::Window*.

Todas las modificaciones de código e imágenes han de realizarse en la carpeta applications de la carpeta de openvibe. Cualquier archivo fuera de esa carpeta no será compilado. Las imágenes se encuentran, para este proyecto en cuestión en:

`\\applications\demos\\ssvep-demo-robot\\share\\resources\\trainer`

Cuando se compila y se instala el proyecto, las imágenes se instalan en la siguiente ruta:

`\\dist\\share\\openvibe\\applications\\ssvep-demo-robot\\resources\\trainer`

La ubicación de los archivos es crítica de hecho, un error muy frecuente es el siguiente:

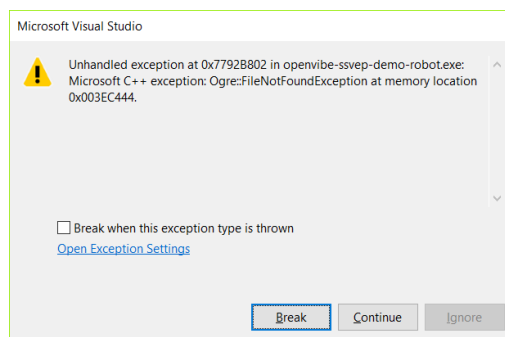


Figura 15: Ventana emergente de error en Visual Studio

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

Eso quiere decir que el archivo no se encuentra en la ubicación apropiada o que el nombre está mal escrito.

6.5. Programación concurrente

6.5.1. Control de imágenes y textos

Como se ha comentado en el apartado anterior se trabaja con objetos para poder condicionar su propiedad de visible. El encargado de este control es un hilo de ejecución en paralelo ubicado en "ovassvepKeyboardCommand.cpp".

ControlScreen, se trata de un hilo que se ejecuta la función handle_Control_Screen basándose en eventos. Dispone de un mutex condicionado a una variable que se activa cuando es necesario refrescar la pantalla.

```
545 pthread_t ManualControl, ControlScreen;
546 pthread_attr_t attr;
547 pthread_attr_init(&attr);
548 pthread_create(&ManualControl, &attr, handle_ManualControl, args);
549 pthread_create(&ControlScreen, &attr, handle_Control_Screen, args);
550 }
```

6.5.2. Control de acciones manuales

En numerosas ocasiones es necesario introducir acciones manuales para debug o simplemente acciones. En este proyecto se han programado rutinas manuales de pick and place del robot simplemente pulsando el teclado numérico. Así mismo se puede controlar el arranque del robot, así como habilitarlo o deshabilitarlo manualmente en caso de peligro.

La rutina manual la controla el hilo ManualControl ejecutando de manera eventual la función handle_ManualControl. Este hilo como el anterior ejecuta la acción dependiendo la tecla pulsada por el usuario.

```
561
562 if (oKey == OIS::KC_A)
563 {
564     m_poApplication->getLogManager() << LogLevel_Info << "***** A\n";
565     l_poShooterApplication->m_RobotStaubli->Enable_Robot = true;
566     pthread_cond_signal(&condition_ControlScreen);
567 }
568
569 if (oKey == OIS::KC_S)
570 {
571     m_poApplication->getLogManager() << LogLevel_Info << "***** S\n";
572     l_poShooterApplication->m_RobotStaubli->Enable_Robot = false;
573     pthread_cond_signal(&condition_ControlScreen);
574 }
```

La configuración de las acciones se realiza en la clase CCommandKeyBoard. Se observa en la línea 562 hasta la 567 como se programa la habilitación del robot cuando

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

se pulsa la tecla A. A partir de la 569 se programa la deshabilitación del robot pulsando la tecla S.

6.6. Codificación de los estímulos

El programa Openvibe ha codificado una serie de estímulos con un valor en hexadecimal. Gracias a esta codificación se permite realizar una comunicación manteniendo el mismo protocolo de transmisión de estímulos. La lista de estímulos y su codificación se encuentran definidas y se encuentran disponibles en la página web del programa (Ibonnet, 2011). La *Figura 16* muestra un ejemplo de la codificación.

OpenViBE stimulation codes

1	#define OVTK_StimulationId_BaselineStart	0x00000007	// 32775
2	#define OVTK_StimulationId_BaselineStop	0x00000008	// 32776
3	#define OVTK_StimulationId_Keyp	0x00000002	// 32782
4	#define OVTK_StimulationId_Button1_Pressed	0x00000012	// 32786
5	#define OVTK_StimulationId_Button1_Released	0x00000013	
6	#define OVTK_StimulationId_Button2_Pressed	0x00000014	
7	#define OVTK_StimulationId_Button2_Released	0x00000015	
8	#define OVTK_StimulationId_Button3_Pressed	0x00000016	
9	#define OVTK_StimulationId_Button3_Released	0x00000017	
10	#define OVTK_StimulationId_Button_Pressed	0x00000018	
11	#define OVTK_StimulationId_Button_Released	0x00000019	
12	#define OVTK_StimulationId_DoubleKeyp	0x00000203	// 32813
13	#define OVTK_StimulationId_EndOfFile	0x00000204	// 32814
14	#define OVTK_StimulationId_ExperimentStart	0x00000001	// 32769
15	#define OVTK_StimulationId_ExperimentStop	0x00000002	// 32770
16	#define OVTK_StimulationId_Label_00	0x00000100	// 33024
17	#define OVTK_StimulationId_Label_01	0x00000101	
18	#define OVTK_StimulationId_Label_02	0x00000102	
19	#define OVTK_StimulationId_Label_03	0x00000103	
20	#define OVTK_StimulationId_Label_04	0x00000104	
21	#define OVTK_StimulationId_Label_05	0x00000105	
22	#define OVTK_StimulationId_Label_06	0x00000106	
23	#define OVTK_StimulationId_Label_07	0x00000107	
24	#define OVTK_StimulationId_Label_08	0x00000108	
25	#define OVTK_StimulationId_Label_09	0x00000109	
26	#define OVTK_StimulationId_Label_0A	0x0000010A	
27	#define OVTK_StimulationId_Label_0B	0x0000010B	
28	#define OVTK_StimulationId_Label_0C	0x0000010C	
29	#define OVTK_StimulationId_Label_0D	0x0000010D	
30	#define OVTK_StimulationId_Label_0E	0x0000010E	
31	#define OVTK_StimulationId_Label_0F	0x0000010F	
32	#define OVTK_StimulationId_Label_10	0x00000110	
33	#define OVTK_StimulationId_Label_11	0x00000111	
34	#define OVTK_StimulationId_Label_12	0x00000112	
35	#define OVTK_StimulationId_Label_13	0x00000113	
36	#define OVTK_StimulationId_Label_14	0x00000114	
37	#define OVTK_StimulationId_Label_15	0x00000115	
38	#define OVTK_StimulationId_Label_16	0x00000116	
39	#define OVTK_StimulationId_Label_17	0x00000117	
40	#define OVTK_StimulationId_Label_18	0x00000118	
41	#define OVTK_StimulationId_Label_19	0x00000119	
42	#define OVTK_StimulationId_Label_1A	0x0000011A	
43	#define OVTK_StimulationId_Label_1B	0x0000011B	
44	#define OVTK_StimulationId_Label_1C	0x0000011C	

Figura 16: Codificación de Openvibe.

6.7. Código de comunicaciones

La Red de Periféricos de Realidad Virtual (VRPN) es un conjunto de clases dentro de una librería y un conjunto de servidores que están diseñados para implementar una interfaz transparente entre aplicaciones y un conjunto de dispositivos físicos. En el proyecto se ha utilizado el mismo fin para interconectar la aplicación diseñada en Openvibe con una aplicación externa diseñada en VisualStudio.

El protocolo VRPN dispone de dos servidores, Analog VRPN Server y Button VRPN Server. La diferencia principal entre ambos queda reflejada en el nombre, el servidor analógico es capaz de recibir una conexión de un cliente analógico y enviar señales analógicas, por ejemplo las señales provenientes del casco Enobio. El servidor Button es simplemente un servidor digital que recibiendo la conexión de un cliente digital puede enviar señales lógicas similar al mecanismo de un botón.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

En el proyecto se han utilizado servidores digitales (Button VRPN Server) para realizar acciones. Como ejemplo nombrar el servidor SSVEP_VRPN_StartStop. Este servidor vinculado al inicio y paro del programa es usado para controlar el experimento. Como ejemplo identificativo se puede observar el nombre del periférico en la primera línea y dos botones que sirven para comenzar y detener el experimento. Se observa que los valores enviados corresponden con los códigos de estímulo tal y como se ha expuesto en el apartado anterior.

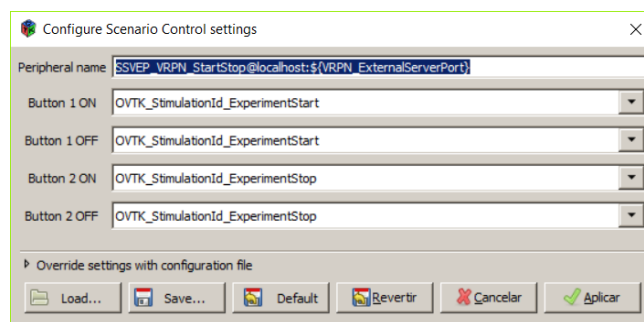


Figura 17: Configuración del escenario. Comunicación VRPN.

En el programa creado en VisualStudio, se ha implementado una clase llamada CCommandStartStop la cual contiene las dos funciones siguientes. Se observa que tras pulsar el botón espacio del teclado, se envía por VRPN el estado del botón y el programa actúa en consecuencia, empezando o acabando el experimento.

```
31 void CCommandStartStop::receiveKeyPressedEvent( const OIS::KeyCode oKey )
32 {
33     if (oKey == OIS::KC_SPACE)
34     {
35         m_poApplication->getLogManager() << LogLevel_Info << "Start message sent\n";
36         m_poVRPNServer->changeButtonState("SSVEP_VRPN_StartStop", 0, 1);
37     }
38 }
39
40
41 if (oKey == OIS::KC_ESCAPE)
42 {
43     m_poApplication->exit();
44 }
45 }
46
47 void CCommandStartStop::receiveKeyReleasedEvent( const OIS::KeyCode oKey )
48 {
49     if (oKey == OIS::KC_SPACE)
50     {
51         m_poVRPNServer->changeButtonState("SSVEP_VRPN_StartStop", 0, 0);
52     }
53 }
54 }
55 }
```

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

6.8. Estructura de datos

En el siguiente apartado se muestra la estructura global de datos utilizada para la comunicación entre hilos, así como las formas de modificar y acceder a esa estructura.

La variable recibe el nombre de `m_RobotStäubli` y se trata de un puntero a la estructura `arguments` la cual contiene:

- Una variable de la clase `communication` llamada `Robot_TX60`. Esta variable contiene todas las funciones y variables necesarias para el control del robot.
- Una variable entera para determinar el grado de libertad.
- Una variable entera que determina el sentido del movimiento
- Un par de variables booleanas que habilitan o deshabilitan el movimiento y el robot.
- Un par de variables que permiten configurar el puerto y el host del robot.

```
13
14  struct arguments {
15
16      communication::STAUBLI* Robot_TX60;
17      int Robot_axis_MODE;
18      int Robot_direction;
19      bool Enable_Movement;
20      bool Enable_Robot;
21      int Increment;
22      //char *Origen;
23      std::string RobotHost;
24      int RobotPort;
25
26  };
27
```

6.8.1. Modificación de los ejes

A continuación se muestra un extracto de la clase `CCommandShipControl`, la cual realiza el control de cambio de eje y control del movimiento según la orden recibida.

Cuando la clase recibe desde `Openvibe` vía `VRPN` un valor de `iButton` igual a cero, entra en el primer bloque de la casuística en el cual dependiendo de si se encuentra en la posición u orientación seleccionada en pantalla (de forma manual), almacena en la variable `Robot_axis_MODE` el valor del eje correspondiente a mover.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

```
20
21 void CCommandShipControl::execute(int iButton, int iState)
22 {
23     CShooterApplication* l_poShooterApplication = dynamic_cast<CShooterApplication*>(m_poApplication);
24
25     InInterrupt = clock();
26
27     switch (iButton)
28     {
29     case 0:
30         if (InInterrupt - InCase0 > 4000){
31             cout << "Recive 0: Cambio de eje" << endl;
32
33             if (l_poCommandKeyBoard->getSelectAxis() == 0){
34                 if (l_poCommandKeyBoard->getSelectVertical() == 2){
35                     //cout << "Recive 0: Cambio de eje XXXXXXXX" << endl;
36                     l_poShooterApplication->m_RobotStaubli->Robot_axis_MODE = ST_AXIS_X;
37                 }
38                 else if (l_poCommandKeyBoard->getSelectVertical() == 1){
39                     //cout << "Recive 0: Cambio de eje RRRRRRRRXXXXXXXX" << endl;
40                     l_poShooterApplication->m_RobotStaubli->Robot_axis_MODE = ST_AXIS_RX;
41                 }
42                 l_poShooterApplication->m_RobotStaubli->Enable_Movement = false;
43                 l_poCommandKeyBoard->setSelectAxis(l_poCommandKeyBoard->getSelectAxis() + 1);
44                 l_poCommandKeyBoard->viewAxis = 0;
45             }
46         }
```

Como la funcionalidad de este estímulo es realizar un cambio de ejes, no es apropiado que tenga un rápido refresco, de ahí que en este caso tenga un retraso de cuatro segundos. El sujeto con esta condición puede cambiar de ejes cada cuatro segundos.

6.8.2. Modificación del sentido de movimiento

En la misma clase que el apartado anterior, hay dos casuísticas más que corresponden con el cambio de sentido de movimiento. Se observa en la figura que la lógica es muy sencilla, se modifica la variable interna que se corresponde con el sentido (Robot_direction) y se habilita el movimiento del robot para ese instante.

El hilo de ejecución que controla el robot será el encargado de realizar el movimiento como se verá a continuación.

```
79     case 1:
80         if (InInterrupt - InCase1 > 100){
81             cout << "Recive 1: Movimiento negativo" << endl;
82             l_poShooterApplication->m_RobotStaubli->Robot_direction = ST_NEGATIVE_MOV;
83             l_poShooterApplication->m_RobotStaubli->Enable_Movement = true;
84             InCase1 = clock();
85
86         }
87         break;
88     case 2:
89         if (InInterrupt - InCase2 > 100){
90             cout << "Recive 2: Movimiento positivo" << endl;
91             l_poShooterApplication->m_RobotStaubli->Robot_direction = ST_POSITIVE_MOV;
92             l_poShooterApplication->m_RobotStaubli->Enable_Movement = true;
93             InCase2 = clock();
94
95         }
96         break;
```

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

6.8.3. Movimiento del robot

Como se ha comentado a lo largo de la memoria, la forma de garantizar la visualización de los estímulos garantizando la constancia en la frecuencia, así como comunicarse con un proceso externo y controlarlo es posible gracias a la programación concurrente. Para ello se ha implementado un hilo de ejecución que se encarga de la comunicación con el robot Stäubli y de su control. A continuación se observa la rutina principal de control.

```
71 while (l_args->Enable_Robot == true && !error){
72     if (l_args->Enable_Movement == true) {
73         ret = l_args->Robot_TX60->configure_move_increment(l_args->Increment);
74         if (ret == 0){
75             ret = l_args->Robot_TX60->set_move_single_axis(l_args->Robot_axis_MODEE, l_args->Robot_direction);
76             l_args->Robot_TX60->move_arm();
77             wait_end_move(l_args->Robot_TX60);
78             printf("\n\n *** Movimiento de Robot: Dirección %d ***\n\n", l_args->Robot_direction);
79             l_args->Enable_Movement = false;
80         }
81         else{
82             printf("ERROR: Lost connection with STAUBLI\n");
83             error = true;
84         }
85     }
86     else{
87         printf("Esperando Orden de Movimiento\n");
88     }
89     Sleep(500);
90 }
91 printf("Esperando Start Experimento\n");
92 Sleep(500);
93 }
```

Se observa como la rutina del robot espera a que se encuentre el robot habilitado para realizar cualquier tipo de cambios en él, en caso de que no lo estuviera estaría esperando la activación.

Una vez el robot se encuentra habilitado, en caso de tener orden de movimiento (recibida en el apartado anterior), se configura el incremento de movimiento según el estado actual (recuerde que se puede modificar este parámetro por pantalla), se le envía el comando de configuración de movimiento de un solo eje (set_move_single_axis) ya que como se ha visto en apartados anteriores se realizan movimientos solo de un eje. Una vez configurado se le ordena al robot que se mueva con el comando move_arm y se espera a que finalice el movimiento con el comando wait_end_move, para acabar se deshabilita la orden de movimiento a la espera de recibir otra orden.

En caso de que el robot perdiera la conexión, el programa automáticamente intenta reconectarse y volver al estado en que se quedó.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

7. CONCLUSIONES

En este documento se han otorgado al lector conocimientos para instalar las aplicaciones necesarias, así como nociones básicas de modificación de código.

Las posibilidades de diseño e utilidad en la aplicación son infinitas y se lo otorga al lector las herramientas para que, con creatividad e ingenio, pueda realizar mejoras al producto así como añadir nuevas funcionalidades.

8. BIBLIOGRAFIA

Ibonnet. (2 de Septiembre de 2011). *Stimulation Codes*. Obtenido de <http://openvibe.inria.fr/stimulation-codes/>

OGRE. (1 de Septiembre de 2017). *OGRE Wiki, Support and community documentation for Ogre3D*. Obtenido de <http://www.ogre3d.org/tikiwiki/tiki-index.php>



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

DISCA
DEPARTAMENTO DE INFORMÁTICA
DE SISTEMAS Y COMPUTADORES



Departamento de Ingeniería de Sistemas y Automática
Departamento de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia

Trabajo Fin de Máster

Máster Universitario en Automática e Informática Industrial

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

4.- PRESUPUESTO

Autor:

D. Javier Dadone Ravera

Tutor:

D. Eduardo Quiles Cucarella

D. Martín Mellado Arteché

Valencia, septiembre de 20

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DEL PROYECTO

ÍNDICE DE TABLAS.....	3
1. PRESUPUESTO PARCIAL.....	1
1.1. SOFTWARE Y HARDWARE.....	1
1.2. PRECIOS MANO DE OBRA.....	2
1.3. SALARIOS.....	2
2. PRESUPUESTO TOTAL.....	3

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

ÍNDICE DE TABLAS

Tabla 1: Presupuesto de Software y Hardware.....	1
Tabla 2: Coste por hora de un ingeniero electrónico	2
Tabla 3: Coste salarial por la mano de obra	2
Tabla 4: Presupuesto parcial sin IVA	3
Tabla 5: Presupuesto total con IVA incluido	3

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

1. PRESUPUESTO PARCIAL

El objetivo principal de este apartado es realizar un presupuesto lo más aproximado posible a la realidad, para ello se tiene en cuenta el coste del material tanto hardware como software, así como el coste salarial de los ingenieros.

1.1. SOFTWARE Y HARDWARE

El software y hardware informático que se ha utilizado para el desarrollo del proyecto puede apreciarse desglosado en la *Tabla 1*.

Como software destacar el programa de desarrollo y mantenimiento de aplicaciones robóticas Staübli Robotics Suite el cual ha sido necesario para la simulación del robot industrial Staübli TX60 utilizado y el programa OpenVibe necesario para el análisis y tratamiento de las señales EEG.

En cuanto al hardware destacar el casco bluetooth Enobio3G de NE Neuroelectrics, necesario para la adquisición de señales EEG y un PC completo usado como base de cómputo.

Software y Hardware

Referencia	Descripción	Unidad	Nº de Unidades	Precio (€)	Total (€)
Stäubli Robotics Suite	Sistema de Desarrollo y Mantenimiento de Aplicaciones Robóticas	Ud	1	25000*	25000*
PC Completo	Ordenador de mesa <i>Fujitsu</i>	Ud	1	725,95	725,95
Enobio	Enobio 8 canales, electrodos, USB, pinza	Ud	1	7.000,00	7000
OpenVibe	<i>Software</i> de análisis EEG	Ud	1	0	0
Total					32725,95

Tabla 1: Presupuesto de Software y Hardware.

El total de este apartado asciende a TREINTA Y DOS MIL SETECIENTOS VEINTICINCO EUROS CON NOVENTA Y CINCO CENTIMOS (32725.95€).

* El coste del software Staübli se ha fijado aproximadamente a 25000€ debido a que se trata de un hardware y software cedido por la empresa al instituto AI2.

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

1.2. PRECIOS MANO DE OBRA

A continuación, en la *Tabla 2* se detalla el precio descompuesto de los trabajadores, este presupuesto incluye el salario neto del trabajador junto a su cotización en la Seguridad Social, sus dietas y transporte.

Coste por Hora de mano de obra de Ingeniero Electrónico				
Descripción	Coste Anual	Días/Año	Horas/Días	€/Hora
Salario Base	31.000,00	220	8	17,61
Seguridad Social (28,3%)	8.773,00	220	8	4,98
Dietas	850,00	220	8	0,48
Transporte	240,00	220	8	0,14
Total (€)				23,22

Tabla 2: Coste por hora de un ingeniero electrónico.

El proyecto ha sido desarrollado por un ingeniero electrónico por lo que el coste horario del personal asciende a VEINTITRES EUROS CON VEINTIDOS CENTIMOS LA HORA (23.22€/hora).

1.3. SALARIOS

A continuación, se muestra el presupuesto de la mano de obra que ha sido necesaria para el desarrollo del proyecto. Las horas reflejadas por el ingeniero electrónico corresponden a las horas de investigación, realización y redacción del proyecto. El resultado se expresa en la *Tabla 3*.

<i>Salario</i>				
Descripción	Unidad	Nº de Unidades	Precio (€)	Importe Total (€)
Ingeniero Electrónico	h	450	23,22	10.449,00
Total (€)				10.449,00

Tabla 3: Coste salarial por la mano de obra.

El importe salarial total en base a las horas trabajadas el ingeniero electrónico, contabilizando unas treinta horas por crédito, asciende a DIEZ MIL CUATROCIENTOS CUARENTA Y NUEVE EUROS (10449.00€)

DISEÑO DE UN SISTEMA DE GUIADO DE ROBOTS MEDIANTE LA ADQUISICIÓN DE SEÑALES EEG

2. PRESUPUESTO TOTAL

En este apartado se muestra el presupuesto total del proyecto en el cual está incluida la suma de los costes expuestos en el apartado anterior al cual se le añade un 15% de costes generales para cubrir gastos indirectos en la realización del proyecto.

En la *Tabla 4* se muestra el presupuesto parcial antes de incluir el IVA.

<i>Coste Parcial</i>	
Descripción	Precio (€)
Total de los costes directos y ejecución del material	43.174,95
Total de los gastos generales 15%	6.476,24
Coste Parcial del Proyecto	49.651,19

Tabla 4: Presupuesto parcial sin IVA.

En la *Tabla 5* se muestra el presupuesto total del proyecto con el IVA incluido.

<i>Coste Total</i>	
Descripción	Precio (€)
Coste Parcial del Proyecto	49.651,19
21% IVA	10.426,75
COSTE TOTAL DEL PROYECTO	60.077,94

Tabla 5: Presupuesto total con IVA incluido.

El total del proyecto asciende a SESENTA MIL SETENTA Y SIETE EUROS CON NOVENTA Y CUATRO CENTIMOS (60077.94€).