# ARQUITECTURA PARA LA INTEGRACIÓN DE MOTORES DE VIDEOJUEGOS EN APLICACIONES BASADAS EN INTERFACES CEREBRO-COMPUTADOR

José Ignacio Estévez Damas, Jonay Toledo Carrillo, Leopoldo Acosta Sánchez
Dpto. de Ingeniería Informática y de Sistemas
Universidad de La Laguna, iestevez@ull.edu.es

## Resumen

*Hoy en día, los sistemas de desarrollo para videojuegos, la realidad virtual y la realidad aumentada se utilizan para crear aplicaciones sofisticadas con un esfuerzo relativamente pequeño, ya que incluyen una gran cantidad de componentes listos para ser usados junto con una metodología de desarrollo bien probada. Este trabajo analiza qué elementos adicionales se deben tener en cuenta cuando una de estas aplicaciones debe integrarse con una interfaz cerebro-computador. Derivado de este análisis, se propone una arquitectura modular, con énfasis en aspectos como la generación de estímulos y la correcta asignación de marcas temporales bajo una base de tiempos común.*

**Palabras clave:** Interfaz cerebro computador (BCI), motores para videojuegos, realidad virtual y realidad aumentada, potenciales evocados de estado estable (SSEVP), Unreal Engie 4 (UE4)

## 1. Introducción

La utilización de las señales cerebrales para controlar a voluntad un sistema conectado a un computador se logra mediante los determinados interfaces cerebro-computador, también conocidos por sus siglas en inglés BCI [1]. Este artículo presenta una propuesta para el desarrollo modular de soluciones basadas en BCI, admitiendo la inclusión de sistemas de estimulación visual integrados en aplicaciones desarrolladas a partir de motores de videojuegos. Hoy en día, esta característica facilitaría el desarrollo de sistemas BCI para contextos como los juegos de ordenador, realidad virtual e incluso realidad aumentada, permitiendo el uso de diferentes plataformas (gafas de realidad aumentada, cascos de realidad virtual, consolas, ordenadores convencionales con diferentes sistemas operativos, móviles Android / IOS, o navegadores web).

La señal cerebral, medida mediante un electrodo situado sobre la piel de la cabeza del sujeto, tienen su origen en los cambios en el campo electromagnético resultantes de la actividad sincronizada de un gran número de neuronas cerebrales, aquellas situadas en la región espacial cercana al electrodo. El comportamiento de estas agrupaciones de neuronas depende de muchos factores, pero puede modularse con relativa facilidad haciendo que el cerebro responda ante determinados estímulos externos (visuales, acústicos, táctiles) o incluso mediante la generación por el sujeto de determinados pensamientos (por ejemplo, aquellos que involucran actividad motora). La modulación del comportamiento de las neuronas del cortex cerebral y su reflejo en el electroencefalograma o EEG es el principio básico que permite la construcción de los BCI.

En este artículo asumiremos que nuestro sistema de modulación de la actividad cerebral será el de los potenciales evocados visuales de estado estable (steady state visual evoked potentials o SSEVP). Aquí, el estímulo proviene de una fuente de luz o de un monitor de computador que muestra al sujeto iluminación o un patrón visual que es modificado repetitivamente con cierta frecuencia (normalmente entre los 8 Hz y 30 Hz). Este estímulo modula la señal cerebral, especialmente en la región del lóbulo occipital, zona del cortex visual primario, de modo que la componente sinusoidal de la misma frecuencia que el estímulo, y sus armónicos, adquieren una importancia superior en relación a la situación en la que no hay estímulo, destacando generalmente sobre el resto. La generación del SSEVP es propiciada especialmente cuando el sujeto centra su mirada en el estímulo parpadeante, de forma que éste quede en el centro del campo de visión. Esta técnica, frente a otras existentes, es muy utilizada en los sistemas BCI debido a su alta relación señal/ruido y poca necesidad de entrenamiento del sujeto.

El sistema BCI puede diseñarse de diferentes modos para lograr que el usuario genere un comando a partir de su EEG. La estrategia que consideramos aquí se basa en presentar en una pantalla gráfica diferentes elementos visuales, que parpadean simultáneamente a frecuencias diferentes (frecuencias objetivo). Cada elemento visual representa una opción de las posibles. Entonces, el proceso de generación del comando consta de los

https://doi.org/10.17979/spudc.9788497497169.063 

siguientes pasos:

- Se presentan en la pantalla los elementos visuales sin parpadear, con la finalidad de que el usuario dirija su mirada al elemento objetivo. Como se ha dicho, es importante que el estímulo se produzca en el centro del campo de visión del sujeto para amplificar el SSEVP. Esta fase dura un tiempo $T_{gs}$ (un valor típico son 0,5$s$).

- Comienza la producción del parpadeo, es lo que se considera el comienzo del estímulo. Esta fase dura un tiempo $T_s$ (los valores típicos van de 1,5 s a 5 s). Si el sistema BCI hace una detección se ofrece un feedback al sujeto (por ejemplo, marcando en diferente color la opción elegida). Es habitual descomponer esta fase en segmentos (por ejemplo de 0,5 s) y aceptar una detección cuando se produzca un mínimo de coincidencias en la frecuencia objetivo mayoritaria.

- Se suspende el estímulo visual completamente (no se muestran los elementos visuales). Esta fase dura un tiempo $T_{off}$. Después de esta fase, vuelve a comenzar el ciclo.

## 2. Motivación

La integración de BCI (brain computer interfaces) con motores de desarrollo de videojuegos, sistemas de realidad virtual y aumentada persigue facilitar el proceso de construcción de aplicaciones con interfaces BCI, incidiendo tanto en las características del entorno, como de los estímulos, al aprovechar las enormes capacidades técnicas de estos motores para la síntesis de gráficos. Además, esto permite la investigación de muchas más opciones de diseño en cuanto a la producción de los estímulos visuales, así como la obtención de información sobre la bondad y factibilidad del uso de BCI en una variedad de situaciones.

La utilización de BCIs en juegos y entornos de realidad virtual ha sido investigada desde hace bastante años. En 2008 podemos encontrar una revisión [2] que describe ya en esos años, un buen número de diferentes formas de utilizar BCI en conjunción con estas tecnologías, observándose una doble dirección. Por una parte los sistemas BCI pueden utilizarse como un sistema de interacción más, donde no es necesario el movimiento físico del sujeto. Así, hoy en día se continua investigando en sistemas BCI para la manipulación de objetos en entornos de realidad virtual. Es bastante relevante que este tipo de método es citado expresamente en una revisión general de sistemas de interacción en entornos 3D junto con otros muchos en 2015 [3].

Por otra parte, los videojuegos y entornos de realidad virtual controlados con BCIs sirven para mejorar la capacidad de los sujetos para trabajar con BCIs, haciendo que luego puedan interactuar de forma más fluida y robusta en entornos reales. Esta misma impresión, sobre el doble papel de los sistemas BCI en videojuegos y entornos de realiad virtual se obtiene de una revisión de 2012 [4]

En el campo de la realidad aumentada, se están empezando a publicar estudios ahora. Por ejemplo, el estudio de 2018 [5] realiza un análisis interesante y sistemático sobre la utilización de un sistema BCI basado en SSEVP (potencial evocado de estado estable) con realidad aumentada de 3 comandos para dirigir un robot móvil. Dicho estudio preliminar muestra que un interfaz basado en realidad aumentada con los *targets* situados en el plano del robot y moviéndose con él, son preferidos por los usuarios debido a que se ahorran la necesidad de hacer operaciones mentales para mapear los signos visuales con la orientación del robot en cada momento. Es también significativo, que este diseño basado en realidad aumentada no parece tener un empeoramiento reseñable en la precisión del BCI.

## 3. Requisitos para una implementación BCI basada en Potenciales Evocados Estacionarios (SSEVP)

Desde el punto de vista de la arquitectura general de los sistemas involucrados en la implementación de un sistema BCI, se deben considerar un conjunto de requisitos básicos que permitan tanto la correcta producción de los estímulos, como el etiquetado temporal de los principales eventos. Frente a esto, hay que tener en cuenta que por un lado la utilización de sistemas de desarrollo del ámbito de los videojuegos utilizan esquemas donde la temporización de los estímulos visuales queda supeditada entre otros factores a la tasa de *frames* por segundo que logra la combinación hardware / software y a la frecuencia de refresco del dispositivo de visualización. Por otro lado, los diseños modulares donde los objetivos a acometer por la aplicación son distribuidos entre diversas partes del software que se ejecutan en máquinas diferentes, introducen una dificultad en el establecimiento de relaciones temporales, dado que se necesita un reloj común, o en su ausencia, un sistema que estime el offset entre los relojes de cada subsistema

### 3.1. Temporización del estímulo visual

Si hablamos de la inducción de un SSEVP, el estímulo visual debe incluir un parpadeo estable con una frecuencia particular. Este parpadeo puede conseguirse en un sistema basado en gráficos por computador de dos formas: bien por alternancia de patrones o bien por modulación de la intensidad luminosa del estímulo.

El primer método puede aplicarse cuando tenemos acceso en cada frame a cambiar el patrón visualizado, y la tasa de frames que el sistema de visualización utiliza es controlable (por ejemplo, constante). Así, los parámetros que determinan un estímulo $x$, generado por la alternancia de dos patrones visuales $P1$ y $P2$, durante un tiempo finito, serían:

- Instante de tiempo en el que comienza el estímulo: $\tau_s^x$.
- Número de ciclos consecutivos patrón 1 - patrón 2. Esto determina la duración del estímulo: $N_C^x$
- Número de frames consecutivos donde será visible dentro de un ciclo el patrón 1: $n_{P1}^x$
- Número de frames consecutivos donde será visible dentro de un ciclo el patrón 2: $n_{P2}^x$

Si el patrón se proyecta sobre un dispositivo de tasa de refresco constante $FPS$, la frecuencia del estímulo $F_S$ sería:

$$F_S^x = \frac{FPS}{n_{P1}^x + n_{P2}^x} \tag{1}$$

y la duración del estímulo quedaría como:

$$T_s^x = \frac{N_C^x(n_{P1}^x + n_{P2}^x)}{FPS} \tag{2}$$

El segundo de los métodos se basa en la modulación de intensidad luminosa del estímulo visual. Siguiendo a [6] y [7] la modulación de esta iluminación puede realizarse mediante un método aproximado:

$$s(f,i) = \frac{1}{2}(1 + sin(2\pi f \frac{i}{FPS})) \tag{3}$$

donde $f$ es la frecuencia del estímulo, e $i$ es la cuenta del frame.

Mantener la frecuencia del estímulo estable va a requerir que el sistema produzca una tasa de frames por segundo (FPS) constante. Los motores de videojuegos suelen disponer de mecanismos para establecer el comportamiento del parámetro FPS. Estos mecanismos se basan en:

- La llamada señal de sincronización vertical o Vsync. Esta señal determina cuando el buffer de video ha sido utilizado por completo y puede sustituirse sin temor a dejar una parte del frame anterior sin ser representada. En el sistema UE4 si se activa la sincronización VSync, cuando un frame se ha terminado de renderizar, se espera hasta la llegada de la señal VSync para proceder a la presentación del mismo. En cambio, si el frame no está listo, lo cual puede ocurrir por una carga computacional excesiva para los medios existentes, el frame es descartado. Esta circunstancia equivale a una disminución efectiva del parámetro FPS.
- Sistemas de control de FPS internos. Por ejemplo, en UE4 pueden establecerse valores máximos para la tasa de frames por segundo. Esta opción tiene la ventaja de que independientemente del monitor o dispositivo de visualización (la frecuencia de VSync puede cambiar) se producirá un tasa de FPS máximas. Si el cómputo en cada frame no sobrepasa el periodo estipulado por el FPS máximo, se mantendría un FPS constante. En cambio, si se sobrepasa este tiempo, la tasa de FPS disminuiría, pero el frame no sería descartado.

Como vemos, en cualquier caso, para mantener la tasa de FPS constante es necesario que el tiempo necesario para el cómputo de un frame no sobrepase el periodo del FPS requerido. Para aplicaciones de laboratorio destinadas al estudio de SSEVP esto no debería ser un problema, ya que los estímulos empleados son bastante sencillos, pero puede ser problemático en otros ámbitos, fuera del laboratorio y destinados a sistemas BCI aplicados, por ejemplo, el de la realidad aumentada, donde las necesidades de procesamiento pueden crecer de manera importante, especialmente si se combina la inclusión de objetos virtuales en el ambiente con la detección de objetos reales. Para aliviar este problema, los sistemas de renderizado gráfico en tiempo real utilizan buffers donde se pueden almacenar varios frames, de forma que si el procesado de un frame se alarga más de lo debido, aún existe un colchón de algunos frames más para proceder a la presentación sin alterar el valor FPS.

### 3.2. Latencia de las entradas y base temporal común

En una aplicación BCI, los registros de datos procesados del electroencegalograma o EEG deben

acompañarse de "anotaciones" ubicadas correctamente en un eje temporal. Esto puede originar algunas dificultades si se utiliza un motor de videojuegos para la generación de los estímulos. Veamos dos clases de problemas que podemos encontrarnos:

- **Latencia de las entradas:** La presentación de estímulos al sujeto, podría no estar determinada en su ubicación temporal con la exactitud requerida: en principio se conocerá, en un intervalo temporal, cuya medición se podrá realizar en el marco específico del "reloj" implementado en el hilo de ejecución del motor y cuyos valores extremos dependerán de la "latencia de las entradas", es decir, el retraso existente entre el hilo de ejecución del juego (donde se ordena la producción de los estímulos) y la presentación efectiva del frame asociado. Este retraso se debe a la utilización de un buffer de frames, necesario para mantener la estabilidad en la tasa de frames por segundo, incluso cuando el motor realiza operaciones que consumen más tiempo que el intervalo existente entre dos o más frames. El problema es mayor cuando se trabaja con una tasa de frames por segundo relativamente baja: por ejemplo, para una valor de 30 frames por segundos, un buffer de dos frames suponen una latencia de 66 ms.
- **Diferentes bases de tiempo:** Si la producción de estímulos se realiza en un módulo con acceso a un reloj A, y el registro del EEG se produce en otro módulo con acceso a un reloj B, la anotación del registros con los eventos correspondientes a los estímulos requerirá, además de la consideración de una "latencia de las entradas", de la estimación del desplazamiento (offset) entre los dos relojes.

Una solución práctica para estos dos problemas necesitaría:

- Establecer un reloj de referencia. Por ejemplo, podríamos utilizar un reloj de alta precisión accesible por el subsistema de registro del EEG. Los eventos relativos a la captura de datos del EEG serían registrados, acompañados de marcas temporales obtenidas de este reloj de referencia.
- Un sistema de estimación de offsets entre el reloj del sistema encargado de producir los estímulos (motor de videojuegos) y el reloj de referencia.
- Los eventos de la producción de estímulos deben ser registrados con una marca temporal. Esta marca temporal debe incluir el tiempo del reloj local y el offset relativo al reloj de referencia.
- Estimar la latencia de las entradas. Este dato también debería estar disponible en los eventos relativos a la producción de estímulos.

## 4. Arquitectura del sistema propuesto

En la figura 1 se muestra el esquema de bloques del diseño que se propone.

En dicha arquitectura se establecen tres tipos de unidades básicas:

- **Unidad Estación Base (BSU)**. La estación base obtiene los datos de los sensores (EEG y otros) y establece como mínimo dos servicios: distribución de los datos de EEG registrados a los servidores de análisis de datos y control y un reloj de referencia de alta precisión. Además debe participar en un protocolo para la estimación de offsets del resto de relojes de las unidades participantes, respecto a esta referencia temporal.
- **Unidad de generación de estímulos (SGU)**. Estas unidades se encargan de producir los estímulos necesarios para el sujeto, por un lado, y por otro, procesar los comandos derivados del BCI. Pueden ser tan simples como un dispositivo para generar una fuente de luz parpadeante a cierta frecuencia o tan complejos como un sistema de realidad aumentada soportado por un motor de videojuegos que debe controlar un robot móvil. En estas unidades tendremos interfaces con el mundo físico: al menos deberá existir un interfaz para producir estímulos sobre el sujeto, aunque dependiendo de la aplicación también serán habituales los interfaces para el control de dispositivos: por ejemplo para controlar un robot o un vehículo. **Es obligatorio que registren los eventos relativos a los estímulos y los marquen temporalmente con un reloj local y una estimación del offset respecto al reloj de referencia de la estación base**. Además, deben enviar esta información a los servidores de análisis de datos y control. También pueden recibir de estos servidores comandos para realizar cambios en sus propios entornos (por ejemplo enviar una orden al actuador de un automatismo) o modificar el programa de estímulos a aplicar
- **Servidor de análisis de datos y control (DACS)**. Estas unidades reciben la señales

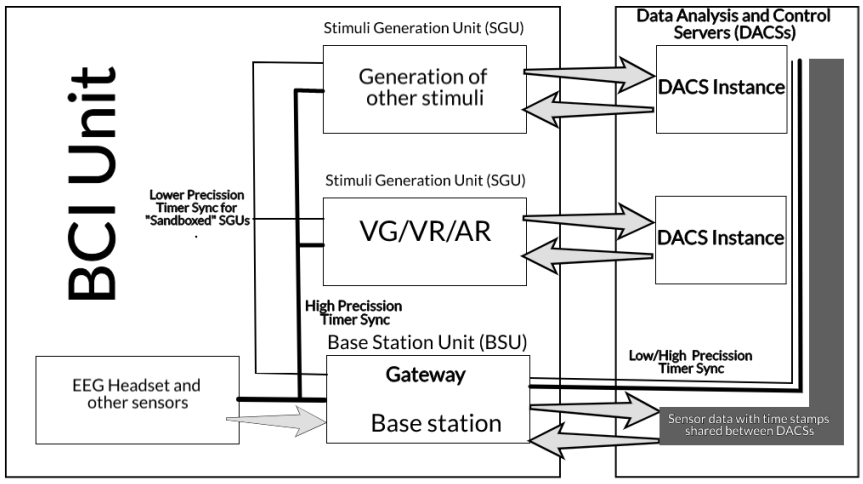


Figura 1: Las líneas continuas más gruesas representan sincronización temporal del alta precisión, mientras que las líneas continuas más finas representan sincronización temporal de menor precisión. Las flechas representan intercambio de datos entre los módulos. La zona gris en el lado derecho representa la transmisión de los datos registrados de EEG a las diferentes unidades de análisis y control.

fisiológicas registradas así como los eventos relativos a los estímulos producidos sobre el sujeto, todo ello con marcas temporales que permiten establecer relaciones temporales con la precisión necesaria. Se encargan del análisis de datos y la estimación de los comandos que el usuario desea aplicar a través del BCI. Dichos comandos son enviados a la unidades de generación de estímulos. Pueden tener otras funcionalidades dependiendo del tipo de aplicación: por ejemplo, pueden enviar comandos a las SGU para modificar el curso de la aplicación, integrar servicios web o servir aplicaciones completas en el caso de que las SGU se basen en un sistema "sandbox" como un navegador.

En el diagrama se establecen dos entornos diferenciados para estas unidades. El entorno "Unidad BCI" comprende los elementos correspondientes a los sensores, la unidad estación base y las unidades de generación de estímulos. Estos elementos pueden ubicarse en una red de comunicaciones local en el entorno próximo al usuario. Por otra parte los servidores de análisis de datos y control pueden situarse en otras redes accesibles desde Internet.

Un aspecto importante a considerar es que las unidades generadores de estímulos (SGU) podrían necesitar ejecutarse en **"sandboxes"** por ejemplo, en **navegadores web**, con un conjunto de recursos limitados y restricciones adicionales. La razón por la que esto puede ser deseable es que así las unidades DACS pueden constituir un sistema centralizado desde donde servir aplicaciones completas consumidas por estas unidades (por ejemplo, aplicaciones HTML5). Esta flexibilidad tiene un precio: posiblemente estos sistemas sandbox no posean las características necesarias para obtener una sincronización con la misma precisión que si todas las unidades estuvieran ejecutando un protocolo de sincronización de alta precisión.

Por ejemplo, en nuestro prototipo utilizamos el protocolo "Laboratory Streaming Layer (LSL)" [8], para conectar la estación base, el servidor de análisis de datos y control y la unidad de generación de estímulos. Los datos del sistema Emotiv Pro son obtenidos del API del software Cortex e inyectados en la capa proporcionada por LSL. El protoclo LSL utiliza un sistema de sincronización temporal sobre UDP que permite una precisión del orden de decenas de milisegundos en el cálculo del offset entre los timer. Para lograr esto, es necesario que la unidad de generación de estímulos, en este caso, un videojuego en primera persona programado con el motor Unreal Engine 4, incorpore las librería de LSL e inyecte datos dentro de esta capa.

Sin embargo, si la SGU fuera programada como una aplicación HTML5 para ser ejecutada por un navegador no dispondríamos de la capa LSL, ya que no forma parte de las funcionalidades del navegador del usuario. Habría que restringirse entonces a las posibilidades previstas por los protocolos a los que puede acceder el navegador. Por ejemplo, se puede utilizar el API de Websockets o un sistema de "long polling" sobre HTML [9] para el intercambio de datos y un sistema de sincronización sobre HTML basado en peticiones periódicas para obtener información del reloj de referencia

a un servidor web que debería implementarse en la estación base. Este tipo de sistema ofrece una precisión menor, pero sería factible, y permitiría disponer de las ventajas de las aplicaciones web modernas.

## 5. Análisis exploratorio sobre errores de sincronización en BCI

En este apartado se muestra un análisis exploratorio que hemos realizado para justificar algunas de las restricciones del modelo propuesto para la construcción de sistemas BCI modulares. Para obtener suficientes datos se ha recurrido al banco de datos generado por los autores del trabajo [10]. La experiencia realizada por Wang et al. contiene las fases de operación descritas en la introducción de este artículo con $T_{gs} = 0{,}5s$, $T_s = 5s$ y $T_{off} = 0{,}5s$. El número de targets es elevado: 40, puesto que está dirigido a una aplicación de teclado virtual. Los datos del EEG están muestreados con una frecuencia de 250 muestras por segundo. Están registradas las fases mencionadas para 35 individuos para cada uno de los 40 targets. De los 35 individuos, 8 tenían experiencia en el uso de BCI y 27 lo utilizaron por primera vez. Para cada individuo se realizó el experimento en 6 ocasiones. Las frecuencias de los targets se distribuyen en el rango entre 8 Hz y 15,8 Hz, con un paso de 0,2 Hz. El método de parpadeo aplicado por los autores del experimento fue el de aproximación por modulación de la intensidad explicado anteriormente.

Para analizar los efectos de no sincronía entre el estímulos y los datos registrados hemos aplicado sistemáticamente un método de análisis de correlación canónica con 3 armónicos, sobre los registros, considerando tiempos de comienzo de la fase de estímulo adelantados progresivamente, de modo que se ha tratado de obtener la frecuencia objetivo mezclando datos de la fase previa a la estimulación, cuando el sujeto está dirigiendo su mirada hacia el objetivo.

En la figura 2 se muestran algunos de los resultados obtenidos. En esta gráfica se han restringido los datos a aquellos individuos para los que el sistema BCI tiene un desempeño promedio de menos de 5 fallos al seleccionar cada uno de los cuarenta objetivos. En el eje horizontal de ambas gráficas se tiene el tiempo de adelante del segmento analizado respecto al tiempo real de aplicación del estímulo en segundos. La gráfica superior muestra como empeoran los resultados del sistema para cada individuo (cada una de las curvas) al adelantar progresivamente el comienzo del segmento (desde 0 ms a 450 ms de adelanto). Se observa gran variabilidad entre los individuos, pero en general un empeoramiento progresivo. De los 18 individuos, 5 acaban con un número de errores sustancialmente por encima de los 5 fallos. La gráfica inferior muestra un promedio sobre todos los individuos del incremento relativo de fallos en términos porcentuales. Las barras verticales que muestran la desviación estándar dan cuenta de la amplia variabilidad entre los individuos pero se sigue observando la tendencia creciente. El dato correspondiente a 150ms, un incremento relativo de 27,8 %, es relevante porque este desfase puede ser bastante típico de determinadas aplicaciones que no incluyeran sistemas de sincronización.

## 6. Ejemplo de caso: adaptación del motor Unreal Engine 4

Unreal Engine 4 es una alternativa atractiva para el desarrollo de aplicaciones que requieran una interacción con el usuario relativamente compleja y que precisen de tecnologías de síntesis de gráficos 3D realistas en tiempo real, tanto en el ámbito de los videojuegos tradicionales como en contextos alternativos como la realidad virtual o la realidad aumentada.

Este equipo de trabajo ha venido desarrollando adaptaciones a este motor tendentes a adaptarlo a la arquitectura descrita más arriba, como unidad de generación de estímulos en una aplicación para la investigación de BCI en realidad aumentada para sillas de ruedas robotizadas.

- **Generación de estímulos**. Se está utilizando el módulo de generación de "sistemas de partículas" del motor para proyectar en el mundo virtual formas geométricas, cuya aparición y desaparición es controlada en cada frame. La parametrización descrita anteriormente para la generación de estímulos SSEVP por alternancia de patrones es la utilizada hasta el momento. Este sistema de partículas puede agregarse como componente al elemento que representa al jugador en primera persona o ser ubicado en otro punto del mundo virtual, o incluso seguir a un objeto real con un marcador fiducial. El motor UE4 puede configurarse para utilizar una tasa fija de frames y sincronizarse con la señal VSync. El módulo de generación de sistemas de partículas de UE4 permite también la modulación en intensidad para la generación aproximada del parpadeo, pero de momento no nos ha sido necesario aplicarlo ya que en esta fase preliminar del proyecto el número de targets es reducido.
- **Integración con Laboratory Streaming Layer (LSL)**. La generación del estímulo de-

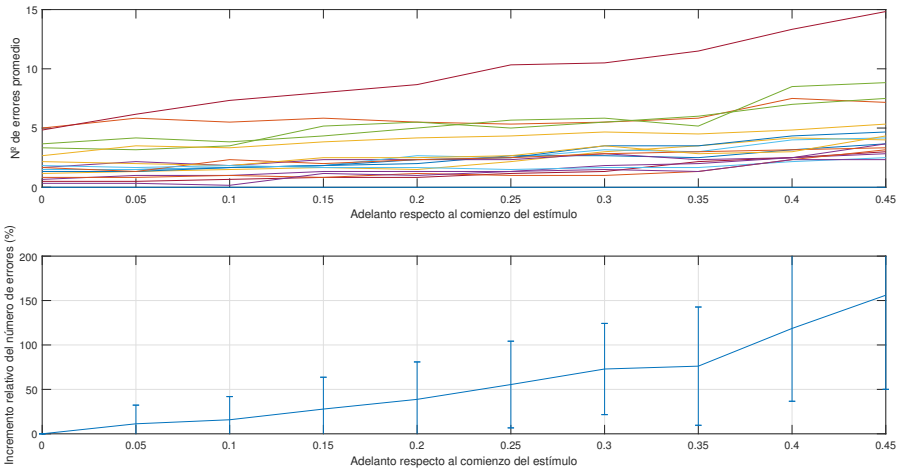


Figura 2: Errores en la identificación del objetivo mediante análisis de correlación canónica con el banco de datos [10], alterando el comienzo de segmento respecto al estímulo real.

be ser registrada con información temporal del reloj local del motor, pero además debe incluir información sobre el offset respecto al reloj de referencia de la estación base. La librería LSL realiza automáticamente la sincronización temporal en una red TCP/IP mediante intercambio de paquetes UDP siguiendo un protocolo similar al conocido como Network Time Protocol.

- **Realidad aumentada y marcadores fiduciales**. El sistema Unreal Engine 4 en la actualidad permite generar aplicaciones de realidad aumentada, contando con plugins como Arcore de Google para aplicaciones Android en dispositivos que soporten esta tecnología. El plugin Arcore permite detectar y parametrizar superficies horizontales lo que es de utilidad para ubicar elementos virtuales en posiciones del entorno. Sin embargo, la detección de marcadores fiduciales no está integrada por defecto en UE4 por lo que ha habido que programar un componente adicional basado en ArUCO Markers para OpenCV y compilar el sistema para la plataforma Android.

## 7. Conclusiones

Este artículo presenta una propuesta de arquitectura modular para aplicaciones BCI donde se han aislado las unidades encargadas de proporcionar estímulos a los usuarios del sistema. El interés de esta forma de enfocar el diseño de estas aplicaciones reside en la posibilidad de utilizar la tecnología de los motores de videojuego con sus sistemas de desarrollo. Así, nos podríamos beneficiar de estos avances en la construcción de videojuegos y sistemas de realidad virtual y aumentada, contextos en los que la aplicaciónde de BCI ha estado y está siendo investigada desde varios puntos de vista, comentados en este artículo.

Tras analizar los requerimientos temporales, especialmente los relativos a la denominada "latencia de las entradas" y la utilización de una base temporal común se han mostrado los resultados experimentales de la aplicación de una técnica convencional para la detección de la selección del individuo mediante SSEVP, sobre un conjunto de datos benchmark introduciendo desplazamientos temporales en el punto estimado de comienzo de la estimulación. De este modo es posible observar el efecto general de un fallo de sincronización de este tipo. Así, con este conjunto de datos se observa una tendencia general hacia el aumento de fallos en la determinación del target, de modo que con solo 150ms de desplazamiento se aumento relativo en el número de errores en un $27,8\,\%$, incluso para un conjunto de individuos con muy buenos resultados bajo una situación sin desplazamiento.

Con estos resultados hay que tener precaución en el diseño de este aplicaciones, especialmente cuando las unidades de estimulación se implementan en "sandboxes" como navegadores, donde las posibilidades de lograr una sincronizacion entre los relojes por debajo de 100ms pueden estar comprometidas.

**Agradecimientos**

El presente trabajo se enmarca dentro de los proyectos DPI2017-9002-R financiado por el Ministerio de Economía, Industria y Competitividad y el MAIH (2016TUR13) financiado por la Fundación

CajaCanarias.

## English summary

## AN ARCHITECTURE FOR INTEGRATING VIDEO GAME ENGINES IN BRAIN COMPUTER INTERFACE (BCI) BASED APPLICATIONS

### Abstract

*Today, development systems for video games, virtual reality and augmented reality are used to build sophisticated applications with relative small-effort, because they include a large number of ready-to-use components together with a well-tested development methodology. This work analyses which additional elements should be taken into consideration when one of this applications has to be integrated with a brain computer interface. Derived from this analysis a modular architecture is proposed, with an emphasis in aspects as the correctness of stimuli generation and production of time stamps for events under a common time-base.*

**Keywords:** Brain-computer interface (BCI), video game engines, steady-state evoked potentials (SSEVP), Unreal Engine 4 (UE4)

## Referencias

[1] Luis J. Barrios, Roberto Hornero, Javier Pérez-Turiel, José L. Pons, Joan Vidal, and José M. Azorín. Estado del Arte en Neurotecnologías para la Asistencia y la Rehabilitación en España: Tecnologías Fundamentales. *Revista Iberoamericana de Automática e Informática Industrial RIAI*, 14(4):346–354, 2017.

[2] Anatole L?cuyer, Fabien Lotte, Richard Reilly, Robert Leeb, Michitaka Hirose, and Mel Slater. *Brain-Computer Interfaces, Virtual Reality, and Videogames*, volume 41. 11 2008.

[3] J Jankowski and M Hachet. Advances in Interaction with 3D Environments. *Computer Graphics Forum*, 34(1):152–190, 2 2015.

[4] Fabien Lotte, Josef Faller, Christoph Guger, Yann Renard, Gert Pfurtscheller, Anatole Lécuyer, and Robert Leeb. Combining BCI with Virtual Reality: Towards New Applications and Improved BCI. In Brendan Z Allison, Stephen Dunne, Robert Leeb, José Del R. Millán, and Anton Nijholt, editors, *Towards Practical Brain-Computer Interfaces: Bridging the Gap from Research to Real-World Applications*, pages 197–220. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[5] Hakim Si-Mohammed, Jimmy Petit, Camille Jeunet, Ferran Argelaguet, Fabien Spindler, Andeol Evain, Nicolas Roussel, Géry Casiez, and Anatole Lecuyer. *Towards BCI-based Interfaces for Augmented Reality: Feasibility, Design and Evaluation*, volume PP. 10 2018.

[6] Xiaogang Chen, Yijun Wang, Shangkai Gao, Tzyy-Ping Jung, and Xiaorong Gao. *Filter bank canonical correlation analysis for implementing a high-speed SSVEP-based brain–computer interface*, volume 12. 6 2015.

[7] Masaki Nakanishi, Yijun Wang, Yu-Te Wang, Yasue Mitsukura, and Tzyy-Ping Jung. *Generating Visual Flickers for Eliciting Robust Steady-State Visual Evoked Potentials at Flexible Frequencies Using Monitor Refresh Rate*, volume 9. 6 2014.

[8] Arnaud Delorme, Tim Mullen, Christian Kothe, Zeynep Akalin Acar, Nima Bigdely-Shamlo, Andre Vankov, and Scott Makeig. *EEGLAB, SIFT, NFT, BCILAB, and ERICA: new tools for advanced EEG processing*, volume 2011. 5 2011.

[9] Victoria Pimentel and Bradford G. Nickerson. *Communicating and Displaying Real-Time Data with WebSocket*, volume 16. 7 2012.

[10] Yijun Wang, Xiaogang Chen, Xiaorong Gao, and Shangkai Gao. *A Benchmark Dataset for SSVEP-Based Brain-Computer Interfaces*, volume PP. 11 2016.