

AGRADECIMIENTOS

Este proyecto no se habría podido acabar sin la continua motivación de mis familiares que me insistieron en la importancia de la finalización del mismo para el desarrollo de mis futuras relaciones laborales. Quiero agradecerles a todos ellos cuanto han hecho por mí, para que este trabajo saliera adelante de la mejor manera posible. Sé de primera mano que siempre han confiado en mí y en mis posibilidades, por eso a su vez quiero dedicar este proyecto a ellos.

Quedo especialmente agradecido con mi tutor del proyecto fin de carrera José Antonio Pérez Carrasco que me ha ayudado y apoyado en todo momento. Ha corregido minuciosamente este trabajo y me dado la posibilidad de mejorarlo. Tengo que agradecerle sus comentarios, direcciones, material proporcionado, sugerencias y las correcciones con las que he podido elaborar una adecuada memoria de todo el trabajo realizado durante todo el periodo en el que he desarrollado este proyecto.

Mis agradecimientos van también a todos los docentes de la Escuela Superior de Ingenieros de Sevilla que me han impartido clase durante mis años de estudiantes. Me han ayudado a formarme no solo como ingeniero si no también como persona, ya que me han animado continuamente a no darme por vencido ante las adversidades.

Por último quiero dar las gracias muy especialmente a Pilar que me ha ayudado, animado, apoyado y querido durante la realización de este proyecto, ya hay que tener valor suficiente para ello.

ÍNDICE

AGRADECIMIENTOS	1
ÍNDICE.....	2
1. INTRODUCCIÓN.....	4
2. MODELO NEURONAL DESCRIPTIVO.....	7
2.1. LA NEURONA	7
2.2. IMPULSO NERVIOSO	10
2.3. SISTEMA NERVIOSO	11
2.4. SINAPSIS.....	12
2.5. PLASTICIDAD NEURONAL	15
2.5.1. APRENDIZAJE Y MEMORIA	17
3. MEMORIA DEL PROYECTO	24
3.1. TIPOS DE SISTEMAS DE CAPTURA DE INFORMACIÓN VISUAL	27
3.1.1. SISTEMAS DE PROCESAMIENTO BASADOS EN EVENTOS FRENTE A LOS BASADOS EN FOTOGRAFÍAS.....	28
3.1.2. ESQUEMAS DE CODIFICACIÓN PARA SISTEMAS BASADOS EN EVENTOS.....	31
3.1.2.1. CODIFICACIÓN BASADA EN LA TASA DE LOS EVENTOS.....	31
3.1.2.2. CODIFICACIÓN BASADA EN EL ORDEN DE LOS EVENTOS.....	32
3.1.3. APLICACIÓN DEL PROTOCOLO AER PARA SISTEMAS BASADOS EN EVENTOS	34
3.2. IMPLEMENTACIÓN	37
3.2.1. DESCRIPCIÓN DEL SISTEMA.....	37
3.2.2. MECANISMO DE APRENDIZAJE STDP.....	46

3.3. RESULTADOS EXPERIMENTALES	51
3.3.1. ESTUDIO DE LA RESPUESTA DE LA NEURONA ANTE DIFERENTES ENTRADAS	57
3.3.2. EVALUACIÓN DEL SISTEMA.....	60
 4. POSIBLES MEJORAS DEL PROYECTO	 63
 5. CONCLUSIONES	 66
 6. APÉNDICES	 67
 6.1. CÓDIGO MATLAB	 67
6.1.1. FUNCIÓN APRENDIZAJE	67
6.1.2. FUNCIÓN ACUMULA PESO	71
6.1.3. FUNCIÓN ACTUALIZACIÓN PESO.....	72
6.1.4. FUNCIÓN CALCULO RATE.....	74
 6.2. ARTICULO URSI	 76
 7. BIBLIOGRAFÍA.....	 80

1. INTRODUCCIÓN

Cada vez es más habitual encontrar sistemas que tratan de implementar el procesamiento del cerebro para resolver problemas complejos. El éxito de estos sistemas queda todavía muy lejos de los objetivos marcados inicialmente por el mundo de la ciencia, por eso el desarrollo tecnológico en esta materia esta en continua evolución. Existen numerosas evidencias que demuestran que el cerebro es capaz de realizar tareas casi impensables para las máquinas más complejas desarrolladas por el hombre a día de hoy, esto provoca un enorme interés en la comunidad científica que se ha dedicado a intentar emular el procesamiento biológico del cerebro. Recientemente los sistemas que están teniendo más éxito son aquéllos basados en redes neuronales, y de ellos, los basados en convoluciones que utilizan algoritmos de entrenamiento como el *backpropagation*. Estos algoritmos están resultando ser un método muy eficiente en el desarrollo de tareas tales como reconocimiento y seguimiento de objetos, clasificación, segmentación, etc [1] [2].

Los sistemas artificiales de visión operan de una manera muy diferente a lo que ocurre en el sistema nervioso visual. Los sistemas artificiales suelen capturar y procesar secuencias de fotogramas. Por ejemplo, una cámara de video captura imágenes a unos 25-30 fotogramas por segundo, que luego son procesados fotograma a fotograma, píxel por píxel. Por lo general aplican operaciones de convolución, para extraer, mejorar y combinar las características hasta conseguir un reconocimiento deseado. Este procesamiento mediante la convolución de fotograma es lento, especialmente si se necesitan realizar muchas convoluciones en secuencia para cada imagen de entrada. Los mecanismos biológicos del sistema visual no funcionan de esta manera, sino que en la retina cada píxel envía pulsos (también llamado eventos) a la corteza cuando su nivel de actividad alcanza un umbral. Por tanto, los píxeles deciden cuándo enviar un evento. Todos estos pulsos se transmiten a medida que están siendo producidos, y no esperan a un tiempo estipulado de transmisión del fotograma (como ocurre en sistemas

artificiales) para enviarlos a la siguiente capa de procesamiento [3]. En puntos posteriores de esta memoria se desarrolla un sistema basado en eventos y no en la captura completa del fotograma. Esto se asemeja mucho más al sistema biológico existente, siendo una mejora bastante reseñable.

Numerosos estudios muestran o representan el sistema nervioso visual estructurado jerárquicamente en capas [5]. Apoyándonos en esta división, cada parte tendrá asignada una función a realizar. Por ejemplo, es ampliamente aceptado que la primera capa de la corteza visual V1 lleve a cabo una operación similar a un banco de filtros de Gabor 2-D con diferentes escalas y orientaciones [2], cuyos parámetros actuales ya fueron medidos [6][7][8]. Este resultado ha sido aprovechado por muchos investigadores a la hora de proponer potentes algoritmos basados en convoluciones para el procesamiento de imágenes [9][10][11]. Sin embargo, las convoluciones son computacionalmente costosas, por lo que parece poco probable que el elevado número de estas operaciones que se llevan a cabo en el cerebro pudiera ser emulado por los programas de software basados en fotogramas que se ejecutan hoy en día.

Según diversos estudios, la solución más adecuada para poder simular un sistema biológico en tiempo real sería considerar un desarrollo mayor del hardware existente combinado con módulos basados en eventos [3][12]. En estos sistemas, las características relevantes de la imagen serían comunicadas y procesadas en primer lugar, lo que ayuda a aumentar la velocidad de procesamiento de la señal. De esta manera, el retardo en el tratamiento del evento de entrada dependerá principalmente del número de capas, y no en la complejidad de los objetos y formas para ser reconocido.

En la actualidad es difícil encontrar sistemas que además de intentar emular la estructura de capas del cerebro utilicen también mecanismos de aprendizaje y comunicación mediante pulsos del mismo. El mecanismo de aprendizaje bioinspirado STDP (*Spike-Timing-Dependent-Plasticity*) [13] es la hipótesis más comúnmente aceptada de mecanismo biológico de aprendizaje. La aplicación del mismo ha dado grandes resultados, aproximándose a resultados obtenidos cuando el procesamiento lo lleva a cabo el cerebro humano. STDP es un proceso que ajusta la fuerza (pesos) de las conexiones entre neuronas basándose en los tiempos relativos entre los pulsos de salida de una neurona y los pulsos recibidos por la misma. La relación entre los tiempos de entrada y salida alteran el potencial de las neuronas provocando que algunos eventos se

asimilen con mayor facilidad. Gracias a la alteración del potencial modelamos el aprendizaje de un patrón determinado en nuestro sistema.

Basándonos en los estudios que han desarrollado esta materia hemos tratado de implementar en este proyecto un pequeño sistema bioinspirado compuesto de dos etapas. La primera de ellas es un banco de filtros de Gabor, que como hemos comentado antes está bastante aceptado como una de las primeras etapas del procesamiento en el cerebro. La segunda etapa es una red neuronal de una etapa, compuesta por diez neuronas de salida cuyas conexiones han sido entrenadas usando STDP. El sistema implementado trata de emular muy ligeramente el mecanismo de aprendizaje y comportamiento del sistema visual humano. Este conjunto recibe como entrada un dígito numérico del 0 a 9 (obtenido de la base de datos MNIST [14]) y sólo la neurona de salida entrenada con ese dígito de entrada es la que presenta una mayor y rápida actividad a la salida.

2. MODELO NEURONAL DESCRIPTIVO

En este punto queremos introducir una serie de definiciones relativas a la naturaleza biológica de la neurona. Lo más reseñable en este punto del informe es que el sistema nervioso se comporta como un conjunto de circuitos por donde pasan impulsos eléctricos, siendo ésto una característica fundamental para el desarrollo del proyecto. Por tanto, gracias al estudio de su constitución se planteará un modelo que simule (de manera lejana) alguna de sus funciones.

2.1. LA NEURONA

La neurona es un tipo de célula del sistema nervioso con unas componentes estructurales básicas que le permiten llevar a cabo la función de transmitir cierto tipo de mensajes, a los que se conoce como impulsos nerviosos. Su principal característica es la excitabilidad eléctrica de su membrana plasmática. Debido a esta cualidad están especializadas en la recepción de estímulos y conducción de los impulsos nerviosos [15].

Algunas de las partes que forman las neuronas son similares a las de las demás células, pero hay otras partes que son distintas. Por eso, no podemos tener el mismo tratamiento que tenemos con el resto de células. A continuación se listan las estructuras principales de la neurona [16]:

- Soma o cuerpo celular: Esta parte incluye el núcleo. Al igual que todas las demás células, las neuronas tienen un núcleo. En esta parte es donde se produce la energía necesaria para el funcionamiento de la neurona.
- Dendritas: Son prolongaciones que salen de diferentes partes del soma, suelen ser muchas y ramificadas. Las dendritas recogen información

proveniente de otras neuronas u órganos del cuerpo y la concentran en el soma de donde, si el mensaje es intenso, pasa al axón.

- **Axón:** Es una sola prolongación que sale del soma en dirección opuesta a las dendritas. La función del axón es la de conducir un impulso nervioso desde el soma hacia otra neurona, músculo o glándula del cuerpo. El axón está compuesto por una capa de mielina (fabricada a su vez por las células de Schwann, también incluidas en el axón) y nódulos de Ranvier que tienen como misión facilitar la transmisión del impulso nervioso. La onda electromagnética producida como impulso nervioso recorre las neuronas recubiertas por mielina, saltando de nodo en nodo de Ranvier.

En la Fig. 1 podemos apreciar todas las componentes que forman la neurona, definidas en este punto [15].

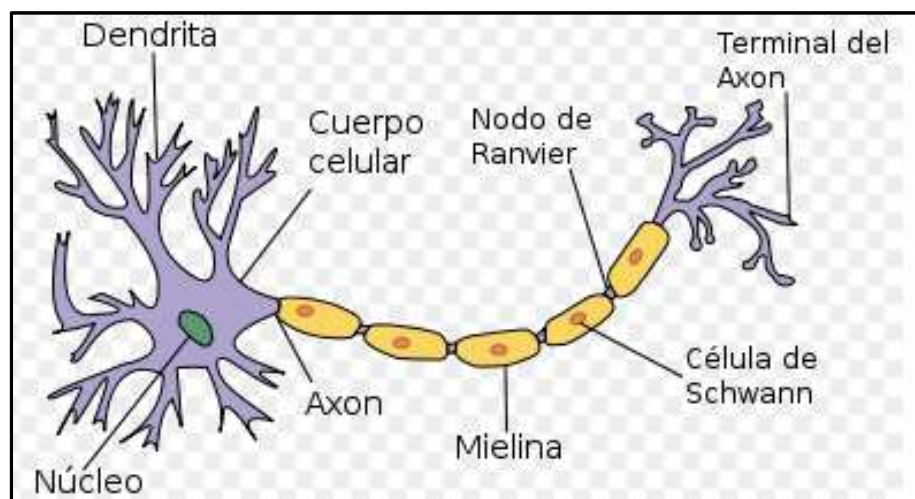


Figura 1. Neurona.

Como hemos comentado, los impulsos nerviosos viajan por toda la neurona comenzando por las dendritas, hasta que llegan a los botones terminales que pueden conectar con otra neurona, fibras musculares o glándulas. Debido a esta función, las neuronas conforman e interconectan los tres componentes del sistema nervioso: sensitivo, motor e integrador o mixto; de esta manera, un estímulo captado en alguna región sensorial entrega cierta información que es conducida a través de las neuronas y es analizada por el componente integrador, que es capaz de elaborar una respuesta cuya

señal es conducida también a través de las neuronas. Dicha respuesta es ejecutada mediante una acción motora, como la contracción muscular o secreción glandular.

2.2. IMPULSO NERVIOSO

El impulso nervioso es una onda eléctrica originada como consecuencia de un cambio transitorio en la permeabilidad de la membrana plasmática de la neurona. Su propagación se debe a una diferencia de potencial (surgido por las distintas concentraciones de iones a ambos lados de la membrana) entre la parte interna y externa de la célula. La carga de una célula nerviosa inactiva se mantiene en valores negativos (el interior respecto al exterior) y varía sólo dentro de unos estrechos márgenes. Cuando el potencial de la membrana de una célula excitable llega más allá de un cierto umbral debido a un estímulo externo (por ejemplo cuando una neurona sensorial capta una señal), la neurona dispara un impulso nervioso gracias a esta diferencia de potencial creando así una corriente eléctrica. Este proceso dura apenas unos milisegundos.

La cantidad de estímulo necesario para provocar la actividad de una neurona se denomina umbral de excitabilidad. Alcanzado este umbral, la respuesta es efectiva independientemente de la interrupción o aumento del estímulo. Durante la despolarización la neurona no es excitable, es decir, está en periodo refractario. Durante la hiperpolarización subsiguiente la neurona es parcialmente excitable y parcialmente refractaria, por tanto, necesitamos un estímulo más intenso para provocar un nuevo potencial de acción ya que ha aumentado el umbral de excitabilidad. Estos factores se debería tener en cuenta a la hora de simular el sistema nervioso [17] [18].

2.3. SISTEMA NERVIOSO

El sistema nervioso es un conjunto de órganos compuesto de unos tejidos y unas células muy similares (por eso se dice que es un sistema y no un aparato) cuya unidad básica es la neurona. Su función primordial es la de captar y procesar rápidamente las señales ejerciendo control y coordinación sobre los demás órganos, para así lograr una oportuna y eficaz interacción con el medio ambiente cambiante. Podemos resumir las actividades del sistema nervioso como una serie de procesos en el que la información sigue un circuito más o menos estándar. Normalmente la señal se inicia en una neurona sensorial aferente, la cual recoge la información a través de su axón (denominado fibra aferente). Esta neurona sensorial transmite la información a otra aleadaña (interneurona), de modo que termine accediendo a un centro de integración donde se procesa la información. Para que la señal llegue a este centro de procesamiento tendrá que pasar por múltiples neuronas que estarán conectadas mediante sinapsis. Finalmente, la respuesta dada irá a parar a las neuronas eferentes que controlan músculos, glándulas u otras estructuras anatómicas. Por tanto, podemos concluir que el circuito neuronal está compuesto por neuronas aferentes, eferentes e interneuronas [19].

Como se aprecia en la Fig. 2 podemos resumir brevemente el sistema nervioso en tres acciones: detección de estímulos, transmisión de información y coordinación general (procesamiento y realización de la acción) [19].

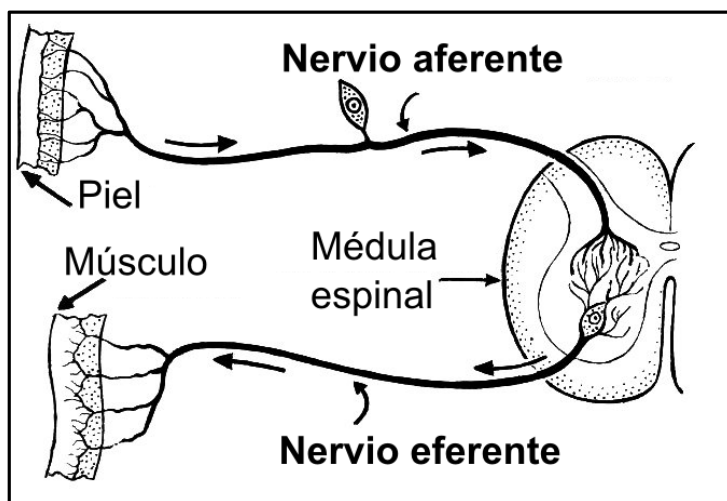


Figura 2. Diagrama aferente-eferente.

2.4. SINAPSIS

La conexión entre una neurona y otra se denomina sinapsis, más concretamente la unión entre el axón de una y la dendrita de otra. Este enlace intracelular lleva a cabo la transmisión del impulso nervioso desde una célula nerviosa a otra. Se inicia con una descarga química que origina una corriente eléctrica en la membrana de la célula sensorial aferente (célula emisora), que como ya hemos dicho es la neurona capaz de captar un evento. La acción de esta componente del sistema nervioso tiene lugar en una etapa presináptica. Una vez que este impulso nervioso alcanza el extremo del axón (la conexión con la otra célula), la propia neurona segrega un tipo de proteínas (neurotransmisores) que se depositan en el espacio sináptico (espacio intermedio entre neurona sensorial y la neurona de salida). Estos neurotransmisores son los encargados de excitar o inhibir la acción de la neurona de salida. La acción que ocurre en ésta última, tiene lugar en la etapa postsináptica. Es muy importante la acción que ocurre en la neurona de salida ya que es en esta etapa donde podemos localizar el aprendizaje del sistema nervioso [20].

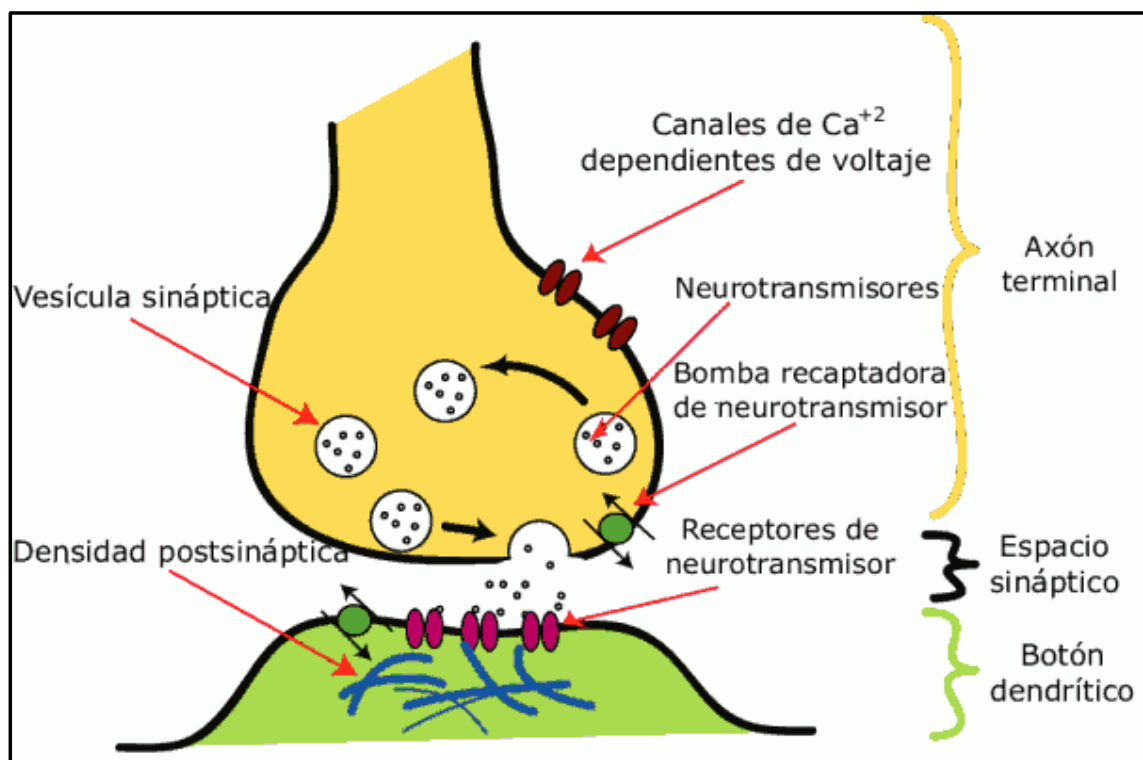


Figura 3. Sinapsis.

La sinapsis se produce en el momento en que se registra actividad químico-eléctrica presináptica y postsináptica. Si esta condición no se da, no se puede hablar de sinapsis. En dicha actividad se liberan neurotransmisores ionizados con base química cuya cancelación de carga provoca la activación de receptores específicos que, a su vez, generan otro tipo de respuestas químico-eléctricas.

Las sinapsis se pueden definir de manera simplificada como un cable. El conjunto de todas ellas permiten a las neuronas del sistema nervioso central formar una red de circuitos neuronales. Una vez que las neuronas son capaces de aprender un determinado suceso, que ocurre repetidas veces, se hacen selectivas. Con esto se quiere decir que son capaces de detectar un evento memorizado con respecto a otro aleatorio, formando unos circuitos virtuales específicos (cuyas neuronas tienen como nexo de unión las sinapsis) para realizar la acción asociada al suceso detectado. Estos circuitos los llamamos virtuales al no ser permanentes, ya que las neuronas pueden olvidar lo aprendido. Por tanto, dicho camino recorrido puede cambiar gracias a una reasignación o modificación de los pesos asociados a cada conexión o sinapsis.

Se distinguen tres tipos principales de transmisión sináptica; los dos primeros mecanismos constituyen las fuerzas principales que rigen la circulación de impulsos nerviosos por los circuitos neuronales:

- Transmisión excitadora: Aquella que incrementa la posibilidad de producir un potencial de acción. La membrana postsináptica reacciona ante el neurotransmisor disminuyendo su potencial de reposo. Por lo tanto, disminuyendo la negatividad interna se aumenta la excitabilidad.
- Transmisión inhibitoria: Aquella que reduce la posibilidad de producir un potencial de acción. La membrana postsináptica se hiperpolariza por el neurotransmisor. Por lo tanto, aumentando la negatividad interna se disminuye la excitabilidad.
- Transmisión moduladora: Aquella que cambia el patrón y/o la frecuencia de la actividad producida por las células involucradas.

El que la acción en una sinapsis sea excitatoria o inhibitoria no depende exclusivamente del neurotransmisor (ya que puede actuar indistintamente de las dos

formas), sino de las características de la membrana postsináptica. Los receptores determinan su respuesta ante un neurotransmisor determinado.

La fuerza de una sinapsis viene dada por el cambio del potencial de la membrana. Dicho cambio ocurre cuando se activan los receptores, ante unos neurotransmisores determinados, alojados en la dendrita de la neurona de salida. Este cambio de voltaje se denomina potencial postsináptico, y es resultado directo de los flujos iónicos a través de los canales receptores. Los cambios en la fuerza sináptica pueden ser:

- A corto plazo y sin cambios permanentes en las estructuras neuronales, con una duración de segundos o minutos.
- De larga duración LTP (*Potenciación a Largo Plazo*), en que la activación continuada o repetida de la sinapsis implica la inducción de la síntesis proteica en el núcleo de la neurona, alterando la estructura de la propia neurona.

Podemos concluir que el aprendizaje y la memoria son resultado de cambios a largo plazo en la fuerza sináptica, mediante un mecanismo de plasticidad sináptica [21].

2.5. PLASTICIDAD NEURONAL

La plasticidad neuronal, también denominada neuroplasticidad, es la propiedad que emerge de la naturaleza y funcionamiento de las neuronas cuando establecen comunicación entre ellas. Es capaz de modular la percepción de los estímulos con el medio, tanto los que entran como los que salen. Por tanto, la red neuronal es un elemento dinámico en continua evolución, donde cualquier pequeño cambio puede producir como resultado una respuesta diferente. Esta propiedad le ha permitido adaptarse al medio sin gran dificultad, pudiendo llegar a asumir una neurona el papel de otra neurona que esté lesionada.

Para subsanar células dañadas el sistema nervioso puede reorganizar el conjunto de sinapsis intervinientes. Para ello, se surte de acciones reguladoras modulando el proceso a través de la aplicación de impulsos excitatorios o inhibitorios según lo requiera. Ésto es posible ya que existen numerosas uniones replicadas, pudiendo así tomar caminos alternativos. Está demostrado que existen conexiones neuronales que incrementan su nivel de actividad cuando ocurre la muerte de un grupo de neuronas que lideraban una determinada función (*desenmascaramiento compensatorio*).

Pero no sólo queda ahí la cosa, porque si la situación lo requiere, el sistema nervioso puede crear nuevas conexiones para que otro grupo de neuronas actúen substituyendo a las erróneas. Así que, podemos clasificar las acciones del sistema nervioso como:

- Constructivas: Cuando crea y amplía nuevas conexiones neuronales.
- Destructivas: Cuando elimina los enlaces poco activos o incluso totalmente inactivos.

La plasticidad anatómica de las neuronas en el sistema nervioso es un fenómeno común en las sinapsis, en las que tanto los estímulos fisiológicos como las condiciones del entorno pueden dar origen a ciertos cambios morfológicos y numéricos. La capacidad de cambio de estos factores es conocido como plasticidad sináptica que a su vez es culpable en gran medida de la facultad de memorizar y olvidar. Por tanto, se demuestra que el aprendizaje y reparación del sistema nervioso es un elemento vivo que

evoluciona con el paso del tiempo, donde diversos sucesos aprendidos pueden ser olvidados y viceversa [21].

2.5.1. APRENDIZAJE Y MEMORIA

La memoria humana es una función cerebral que es resultado de conexiones sinápticas entre neuronas, mediante la que el ser humano puede retener experiencias pasadas. La memorización se obtiene reforzando las conexiones entre las neuronas, a su vez estas uniones se obtienen mediante el continuo entrenamiento al que se le somete. La experiencia sensorial del sistema nervioso es la causa por la que se produce el aprendizaje. A través de la captación de señales se modifica las propiedades de las sinapsis, provocando en ellas:

- Cambios funcionales (reducción del umbral de excitación, mayor liberación de neurotransmisores, mayor persistencia del contacto entre neurotransmisores y receptores, etc.).
- Cambios estructurales (aumenta el número de sinapsis, cambian los contactos sinápticos preexistentes, etc.).

La memoria se puede clasificar según su persistencia. Podemos diferenciar tres grandes grupos: a corto, medio y largo plazo. En la etapa de entrenamiento del sistema se va modificando las uniones sinápticas establecidas, reflejándose en un cambio en la amplitud del potencial de la neurona de salida. Este potencial es responsable de la acción postsináptica, su modificación se puede clasificar en:

- Potenciación sináptica: Aumento de la amplitud de los potenciales en respuesta a impulsos de entrada sucesivos.
- Depresión sináptica: Disminución de la amplitud de los potenciales de la neurona de salida en respuesta a eventos de entrada sucesivos.

Los dos procesos sinápticos mencionados se producen como resultado de cambios en la cantidad de neurotransmisor liberado por cada evento captado. Gracias a esta modificación se va modelando el tipo de memorización según su persistencia. Podemos aplicar de manera global los procesos de modificación de potencial de la neurona de la siguiente forma:

- **Suma Espacial:** Supongamos que, de entre las 10.000 sinapsis posibles, 3.000 están recibiendo señales de excitación y otras tantas de *inhibición*. La suma espacial es el proceso que hace la neurona al elaborar todas esas señales en un mismo ciclo de proceso y producir una respuesta. La respuesta puede aplicar tanto a niveles de potencial de acción como de metabolización de proteínas, neurotransmisores o cualquier otra molécula sea capaz de portar información.
- **Suma Temporal:** Partiendo del mismo supuesto que en el caso de la suma espacial, tomamos como ejemplo una dendrita, en donde se establece sinapsis con una terminación axónica de otra neurona. Si la neurona presináptica produce una ráfaga de estímulos muy seguidos en el tiempo, la neurona que los recibe ha de sumarlos en el tiempo. Se aplica un proceso mediante el cual la neurona establece un resultado a esos estímulos.

En la Fig. 4, se ve cómo va aumentando el potencial de la neurona debido a la acción conducida por las sinapsis. Habrá un momento en que la neurona no pueda aguantar más la presión y dispare para descargar toda la energía acumulada [21].

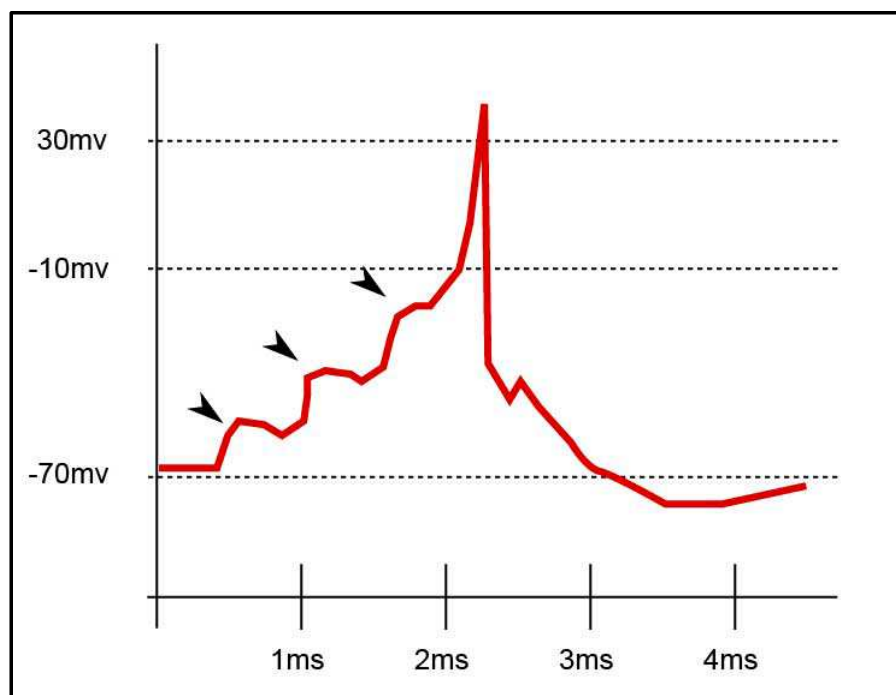


Figura 4. Aumento del potencial almacenado por la neurona debido a la contribución de los impulsos sinápticos.

De aquí se deduce la gran importancia espacio-temporal de los impulsos que transitan las sinapsis. No solo es importante la conexión por la que circula el impulso, sino que también hay que tener en cuenta el momento en el que tiene lugar y el número de veces sucesiva que ocurre el proceso. Ésto será un factor determinante en la memorización del evento captado. Cuánto más se repite el suceso más se refuerza la unión sináptica, contribuyendo así a la asimilación del aprendizaje [21].

Para entender un poco mejor como influye la plasticidad sináptica en el aprendizaje y memorización por parte de una neurona vamos a introducir un pequeño ejemplo. En las Fig. 5, 6 y 7 ilustran dicho ejemplo muy simplificado de un sistema bioinspirado basado en el mecanismo STDP (que a posteriori vamos a desarrollar más detenidamente). En estas figuras se representa una neurona de salida conectada a 3 neuronas que representamos como $c1$, $c2$ y $c3$ que producen un determinado patrón codificado en sus pulsos. Como se puede ver el orden de disparo de las neuronas siempre va a ocurrir en este caso es: primero $c1$, después $c2$ y finalmente $c3$ (como viene representado por la curva de estímulo). El peso de las neuronas se representan en las figuras como un punto negro en las conexiones (cuanto mayor es su tamaño mayor es su valor). Al principio (*paso 1*), todas las conexiones tienen igual valor (los pesos de las 3 sinapsis están pre-configurados con la misma inicialización de $1/3$), de modo que cada vez que llega un pulso a la neurona de salida su potencial se ve incrementado por el peso correspondiente a cada conexión. Se puede ver que en la primera iteración la neurona alcanza el umbral de disparo (de valor '1') cuando le llega el pulso procedente de la neurona $c3$ (instante $t3$). En el primer caso, como el disparo de la neurona de salida se produce posteriormente a los pulsos de entrada, las 3 conexiones se ven reforzadas actualizándose sus valores con un incremento de $1/3$, $2/3$ y 1 respectivamente. En la segunda repetición del patrón de entrada (*paso 2*), será la neurona $c2$ (por tener un *peso*= $2/3$) la que produzca la activación de la neurona de salida en el instante $t2$, con lo cual las conexiones con $c1$ y $c2$ se verán reforzadas y la conexión con $c3$ atenuada debido a que la acción ocurre después de la repuesta de la salida (el peso disminuye en $2/3$) [22].

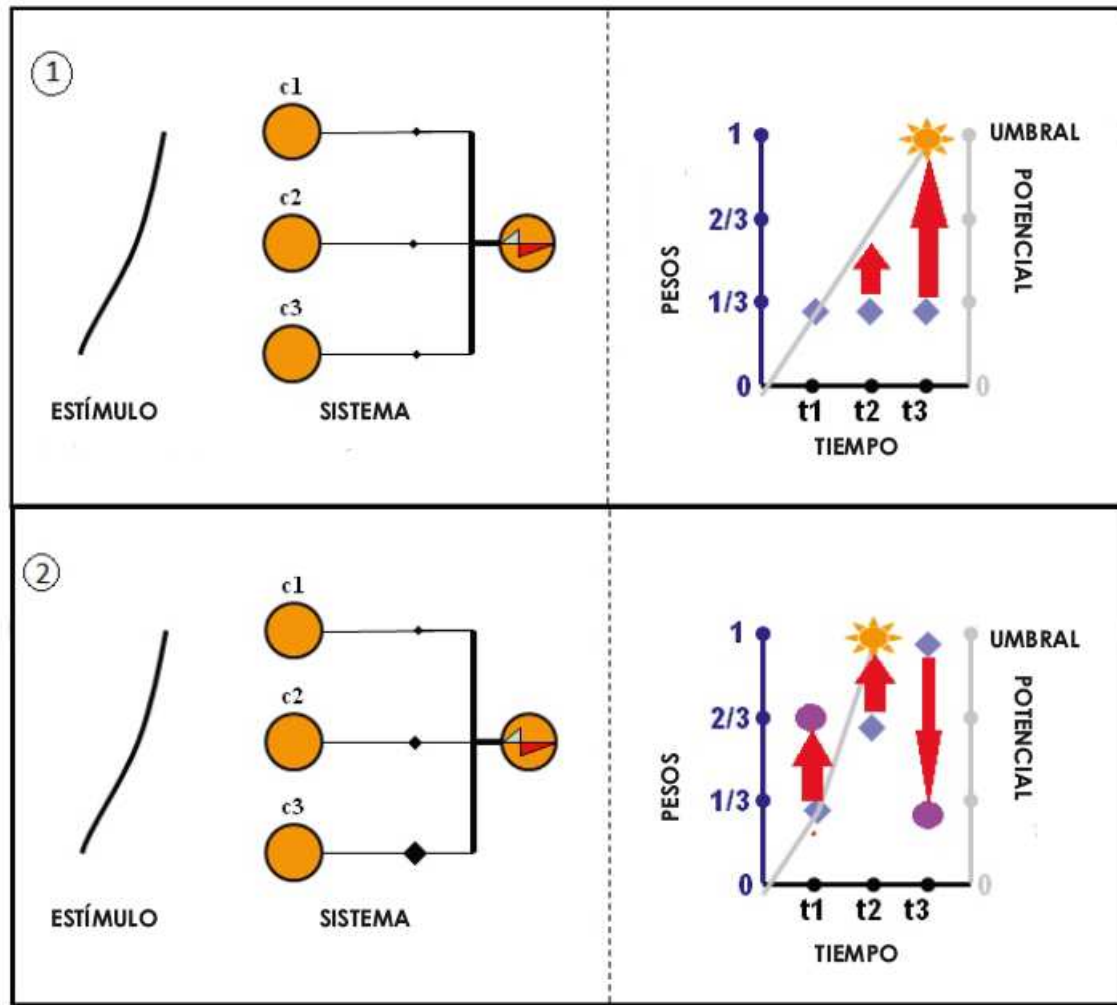


Figura 5. Ejemplo de un mecanismo bioinspirado de aprendizaje STDP (Parte 1).

En una tercera iteración (*paso 3*), será la neurona *c2* de nuevo la causante de producir un disparo a la salida en el instante *t2*, con lo cual la conexión con *c1* será reforzada de nuevo y la conexión con *c3* se verá atenuada totalmente. En la cuarta iteración del ejemplo (*paso 4*), la conexión con la neurona *c1* ya tiene un peso de valor '1' y por tanto es suficiente para activar la neurona de salida por sí misma en el instante *t1* [22].

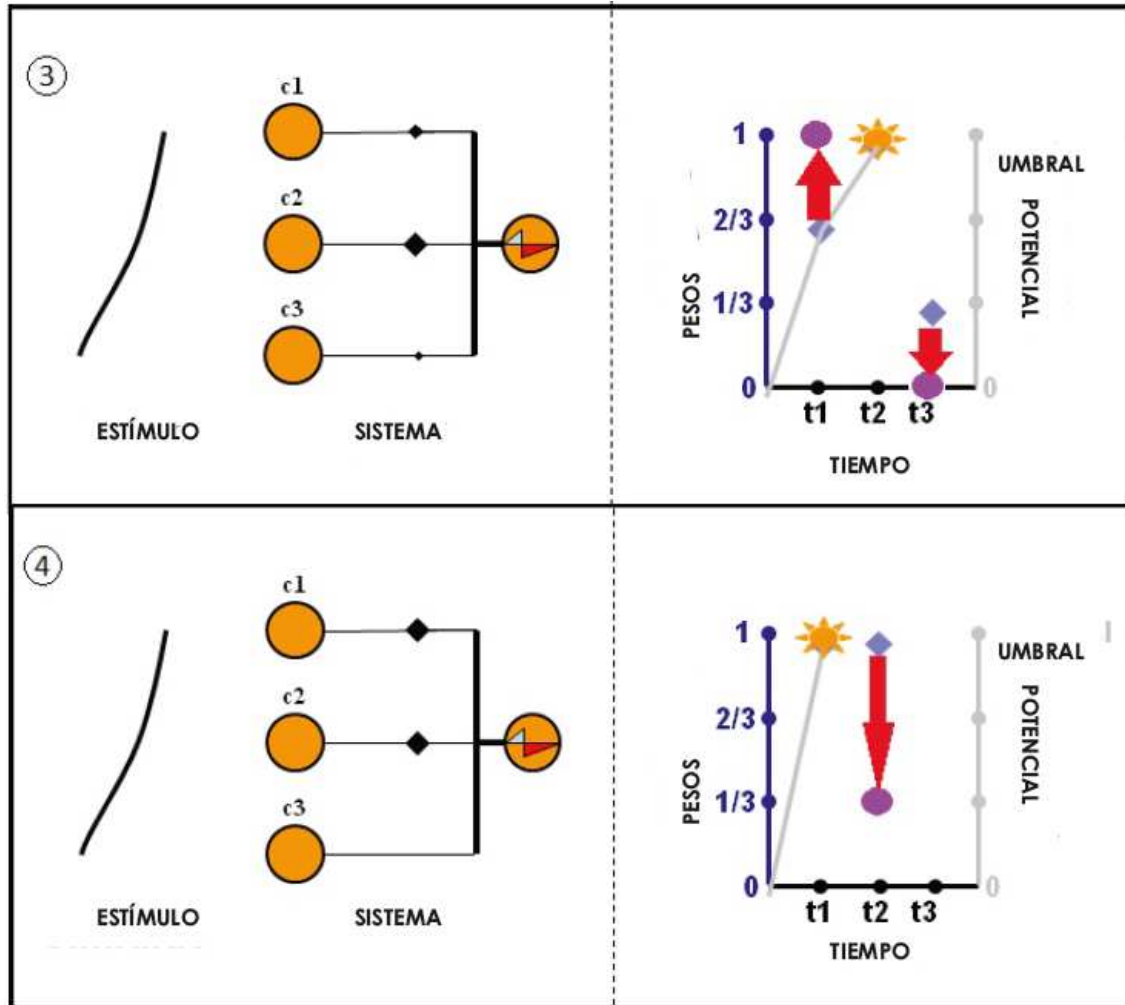


Figura 6. Ejemplo de un mecanismo bioinspirado de aprendizaje STDP (Parte 2).

Las otras dos conexiones (con *c2* y *c3*) se ven atenuadas completamente, como se puede observar sucesivamente en los pasos 5 y 6, donde su contribución a la respuesta del sistema se puede considerar prácticamente nula. En un estado estacionario las conexiones con *c2* y *c3* estarán inhibidas y la conexión con la neurona *c1* bastará para producir la activación con la neurona de salida. Nótese que esto es lo que se esperaba pues la neurona *c1* es la más activa y la más rápida, y el algoritmo de aprendizaje STDP ha permitido que la neurona de salida aprenda esto. La rapidez de este mecanismo beneficia a la detección del patrón sin tener que estudiar por completo el estímulo de entrada.

Como hemos descrito anteriormente la neurona al final siempre responde a la excitación transmitida para la neurona *c1*, pero puede llegar a ocurrir que esto deje de cumplirse y empiece a responder al resto de entradas. Esto puede deberse a que deje de recibir estímulos de la neurona *c1* y empiece a olvidar lo aprendido hasta ahora. Gracias al mecanismo bioinspirado STDP (como haría cualquier sistema “vivo”), el sistema se adapta a la situación y es capaz de aprender a reconocer otro patrón. Por tanto, podemos decir que el sistema es capaz de reciclarse, como ocurre en los sistemas biológicos. Es por ello que podemos asemejar las conexiones establecidas por las sinapsis a circuitos virtuales, ya que tras un largo periodo de inanición de entradas que contienen el patrón entrenado pueden pasar a estar inactivas. En cambio otras sinapsis que antes estaban inactivas pueden pasar a un estado de pleno funcionamiento [22].

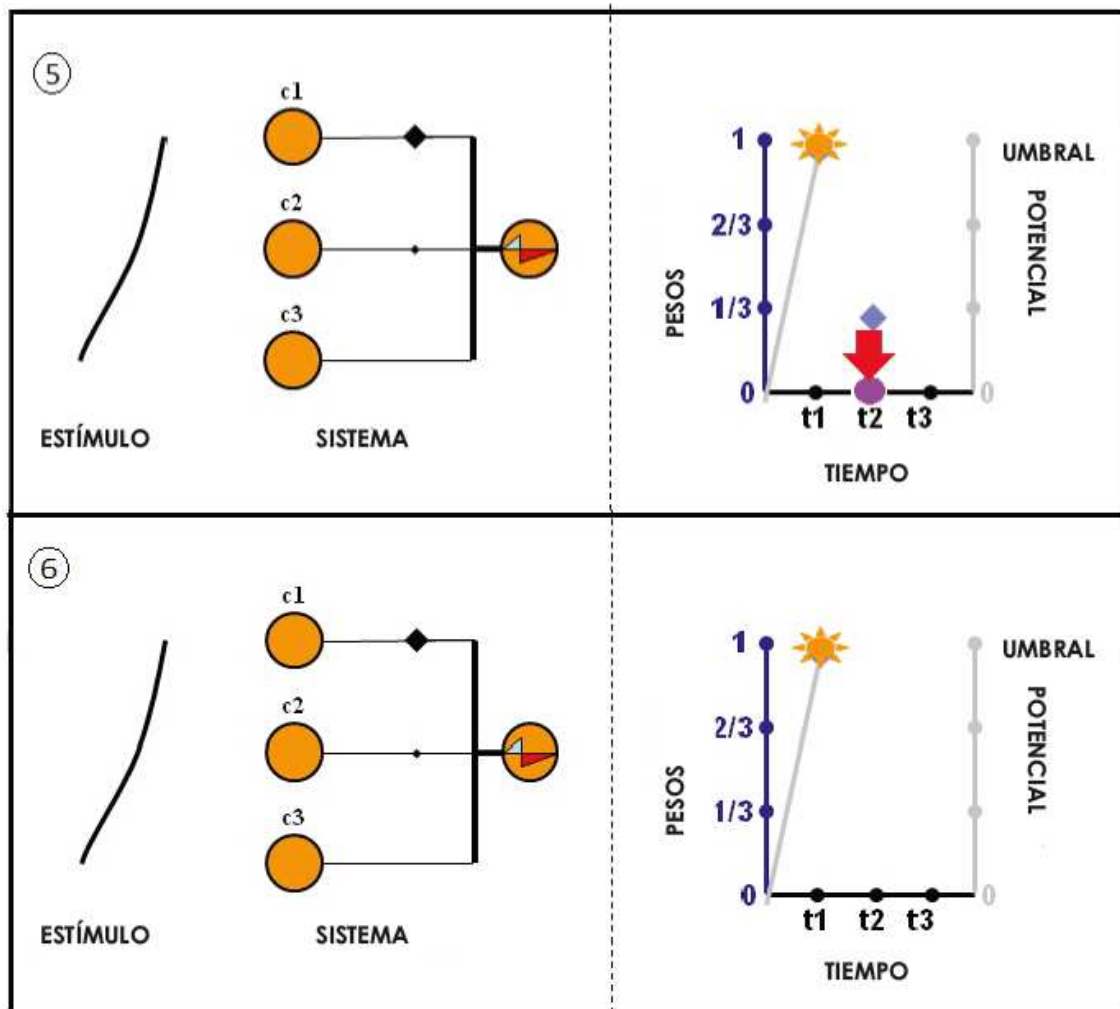


Figura 7. Ejemplo de un mecanismo bioinspirado de aprendizaje STDP (Parte 3).

Quede constancia de que el ejemplo descrito es muy sencillo y que sólo trata de ilustrar la idea que se esconde detrás del mecanismo biológico STDP. Cuando lo aplicamos a un sistema un poquito más complejo, como el de nuestro experimento, el número de combinaciones y conexiones es mucho mayor.

3. MEMORIA DEL PROYECTO

En la etapa de documentación de este proyecto se observa que hay numerosos métodos y mecanismos bioinspirados de aprendizaje que tratan de emular (con mayor o menor éxito) el procesamiento biológico existente en el cerebro. Se pueden destacar como sistemas más exitosos aquellos basados en redes neuronales que utilizan convoluciones para el procesamiento de la señal. Dentro de este grupo destacan aquellos que usan el algoritmo de entrenamiento *backpropagation* propuesto por Yann LeCun [1]. Este mecanismo es muy eficiente a la hora de desarrollar tareas de reconocimiento y seguimiento de objetos. Además, este sistema es capaz de auto-corregir la salida experimental con la deseada. Para ello, realiza los ajustes necesarios en los pesos asociados a las sinapsis para conseguir que el resultado empírico se acerque cada vez más al buscado.

Los mecanismos que utilizan el algoritmo *backpropagation* pertenecen al grupo de sistemas cuyo aprendizaje es *supervisado*. Este método de aprendizaje consiste en un entrenamiento en el que se introducen repetidas veces un determinado patrón, de modo que las respuestas que va dando el sistema se van comparando con las respuestas deseadas. En virtud de estas comparaciones, se reajustan los pesos. El error (diferencia entre la respuesta obtenida y la deseada) se propaga hacia atrás con la misión de modificar los pesos asociados a las neuronas previas. Con este reajuste de la fuerza de las conexiones la red se acerca cada vez más a la respuesta correcta. En la Fig. 8 se presenta un esquema que muestra cómo funcionan los sistemas de aprendizaje supervisados.

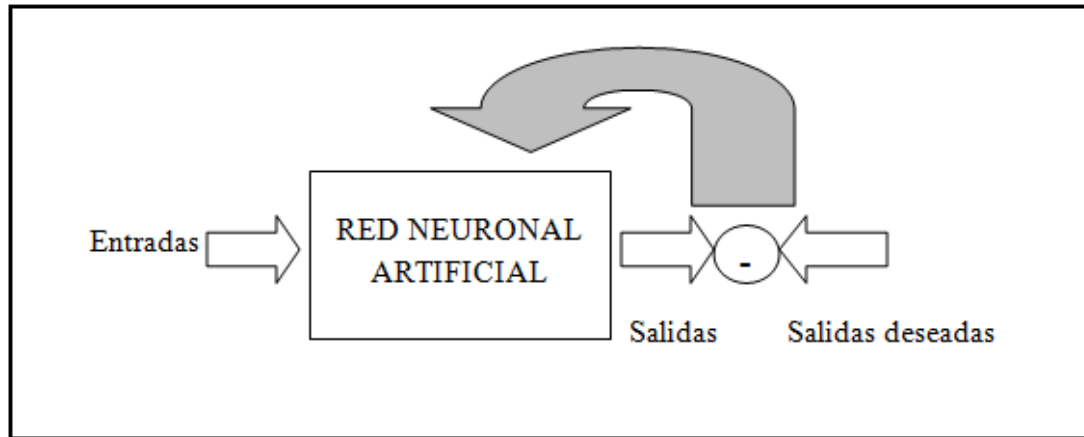


Figura 8. Sistema de aprendizaje supervisado.

Aun existiendo todas las ventajas enumeradas este mecanismo es insuficiente, ya que al ser un sistema de aprendizaje *supervisado* es necesario conocer las entradas y las salidas deseadas en todo momento para poder así minimizar el error. Podemos solucionar este problema utilizando un método de aprendizaje que sea *no supervisado*. Por ello, en nuestro proyecto utilizamos un sistema bioinspirado basado en el mecanismo *no supervisado* STDP (*Spike Timing Dependent Plasticity*) desarrollado por T. Masquelier, R. Guyonneau y S. J. Thorpe [23]. Este mecanismo es capaz de ajustar los pesos sinápticos teniendo solo en cuenta la relación entre los pulsos de entrada y de salida, todo ello sin conocer la salida deseada. Esto significa que no hay información externa que ayude a decidir cómo modificar los pesos de las conexiones. Ha de ser la red la que vaya adecuando los pesos en función de la información interna que vaya recogiendo de las entradas. La modificación de los pesos asociados a las sinapsis modela el proceso de aprendizaje o memorización de la neurona. Cuando la sinapsis se activa, su peso contribuye a la alteración del potencial de la neurona de salida. Por tanto, los sistemas basados en STDP son capaces de implementar el aprendizaje de manera autónoma. Hasta ahora sigue siendo una de las hipótesis con mayor acogida por toda la comunidad científica. En la Fig. 9 se representa un esquema general de los sistemas de aprendizaje *no supervisado*.

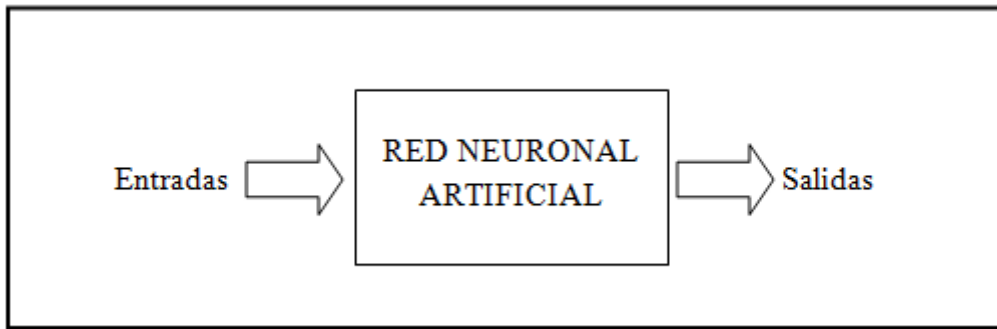


Figura 9. Sistema de aprendizaje *no supervisado*.

En el presente trabajo se utiliza STDP para implementar un mecanismo artificial bioinspirado de aprendizaje. En concreto, nos vamos a basar en el reconocimiento de los diez dígitos numéricos (0 a 9).

En nuestro sistema podemos diferenciar claramente dos etapas:

- La primera etapa se compone de un banco de filtros de Gabor, que está bastante aceptado como una de las primeras etapas del procesamiento en el cerebro. Sus principales tareas son la captación y el acondicionamiento de la señal de entrada.
- La segunda etapa es una red neuronal de una etapa, compuesta por diez neuronas de salida. Cada neurona ha sido entrenada por separado usando STDP.

Aplicaremos los principios de la plasticidad sináptica en nuestro sistema (funcionamiento de las neuronas cuando estas establecen comunicación, modulando la percepción de los estímulos con el medio), ya que la comunidad científica cree que el aprendizaje y almacenamiento de información subyace en la modulación de los circuitos implementados por las sinapsis. Debido a estos circuitos neuronales el sistema nervioso es capaz de transportar el estímulo hasta la neurona. Es posible que el conjunto pueda cambiar de recorrido si el sistema deja de recordar un determinado patrón aprendido con anterioridad, llevando el estímulo a otra neurona diferente. Para realizar este cambio de recorrido se activarían otros circuitos mientras que otros pasarían a un estado de inactividad [23].

3.1. TIPOS DE SISTEMAS DE CAPTURA DE INFORMACIÓN VISUAL

Los sistemas artificiales de visión funcionan de forma diferente a lo que ocurre en el sistema nervioso visual. Por lo general, los sistemas artificiales se dedican a capturar secuencias de fotogramas a las que se les aplican operaciones de convolución, necesarias para alcanzar el reconocimiento deseado de la imagen captada. Este procesamiento mediante convoluciones de fotogramas es lento y costoso computacionalmente, ya que se necesitan realizar muchas convoluciones en secuencia para cada imagen de entrada. En cambio la manera de operar del sistema visual es distinta, los píxeles de la retina envían pulsos a la corteza cada vez que un evento captado alcanza un nivel de actividad superior a un umbral. Todos estos pulsos son transmitidos a medida que están siendo producidos, sin esperar a un tiempo estipulado para la transmisión del fotograma [3].

Diversos estudios representan el sistema nervioso visual estructurado jerárquicamente en capas [5]. Esto simplifica bastante las operaciones computacionales a realizar, ya que cada capa tiene asignada una misión a realizar. Aun así, sigue siendo necesaria la realización de múltiples convoluciones que pueden ser excesivas para los programas de software basados en fotogramas que se ejecutan hoy en día.

Según novedosos estudios la solución más adecuada para resolver el problema del coste computacional vendrá de la mano de un mayor desarrollo del hardware, combinado con un sistema dividido en capas que utilice mecanismos basados en eventos [3] [12]. De esta manera aumentaremos la rapidez en el procesamiento de la señal. Por tanto, el retardo en el tratamiento del evento de entrada dependerá principalmente del número de capas, y no de la complejidad de los objetos y formas para ser reconocido.

3.1.1. SISTEMAS DE PROCESAMIENTO BASADOS EN EVENTOS FRENTE A LOS BASADOS EN FOTOGRAMAS

Para mostrar la velocidad de procesamiento de los sistemas basado en eventos (*Event-Based*), en comparación con los sistemas basado en fotogramas (*Frame-Based*) consideremos la Fig. 10. En ella se ilustra de manera conceptual la diferencia entre un sistema de captura y procesado basado en fotogramas y un sistema basado en eventos (bioinspirado), a la hora de la detección y procesamiento de la entrada. Cada uno utiliza una cámara diferente para captar la realidad.

En la imagen superior, una cámara basada en la captura de fotogramas percibe una secuencia de imágenes, cada una de las cuales se transmite al sistema de procesado. Cada fotograma es tratado por sofisticados algoritmos de procesamiento de imagen para lograr algún tipo de reconocimiento. El sistema informático tiene que tener todos los valores de los píxeles de un fotograma antes de comenzar cualquier cálculo.

En la imagen inferior, un sensor de visión basado en eventos opera sin tener en cuenta el fotograma completo. Cada píxel envía un evento, por lo general codificando una coordenada (x, y) , cuando detecta algo. Normalmente el píxel realiza la acción de envío cuando detecta un cambio de intensidad con respecto a los píxeles vecinos [24] [25]. Los eventos se envían al sistema de procesado, sin esperar a un tiempo marcado por el fotograma detectado. El sistema informático actualiza su estado después de cada evento [3].

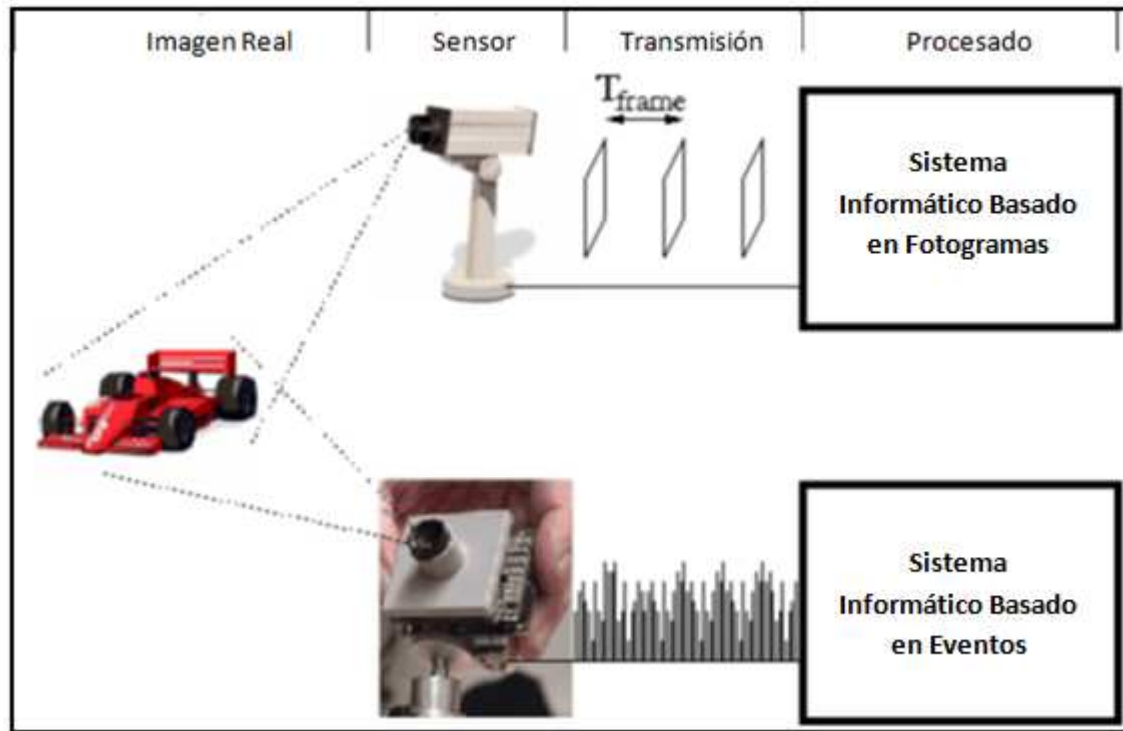


Figura 10. Comparación de los mecanismos de captación y procesado en sistemas basados en fotogramas y en eventos.

La Fig. 11 ilustra la diferencia de tiempo de procesado y detección entre ambos conceptos. En la parte superior (basada en fotogramas), la secuencia de imágenes es tramitada fotograma a fotograma, con un tiempo de detección igual a T_{frame} . Durante el primer fotograma $T1$, la información producida por este método no llega al sistema computacional hasta que el fotograma completo es capturado y transmitido. Así que hay que sumarle un retraso adicional ($T1 + \Delta$). El sistema computacional tiene que procesar la imagen completa, por tanto requiere el manejo de una gran cantidad de datos provocando un alto tiempo de detección. Como se aprecia en la imagen, T_{FC} será el tiempo necesario para procesar la información y que se produzca algún tipo de reconocimiento. En la parte inferior de la Fig. 11, los píxeles de la retina “ven” directamente la escena y generan eventos con pequeños retrasos. Los eventos son procesados a medida que van llegando con un pequeño retraso, debido a la acción de cálculo por parte del sistema. Dicho retraso es representado en la figura como T_{ev} (algunos ns [26]). Para llevar a cabo el reconocimiento no todos los eventos son necesarios. En realidad los acontecimientos más relevantes, por lo general, vienen en primer lugar o con mayor frecuencia. Por consiguiente, el tiempo de reconocimiento T_{rcg} puede ser menor incluso que el tiempo total de los eventos producidos. Nótese que el

reconocimiento es posible antes del tiempo T_1 , lo que resulta un T_{FC}' negativo en comparación con el retraso surgido en el reconocimiento de un sistema basado en fotogramas [3].

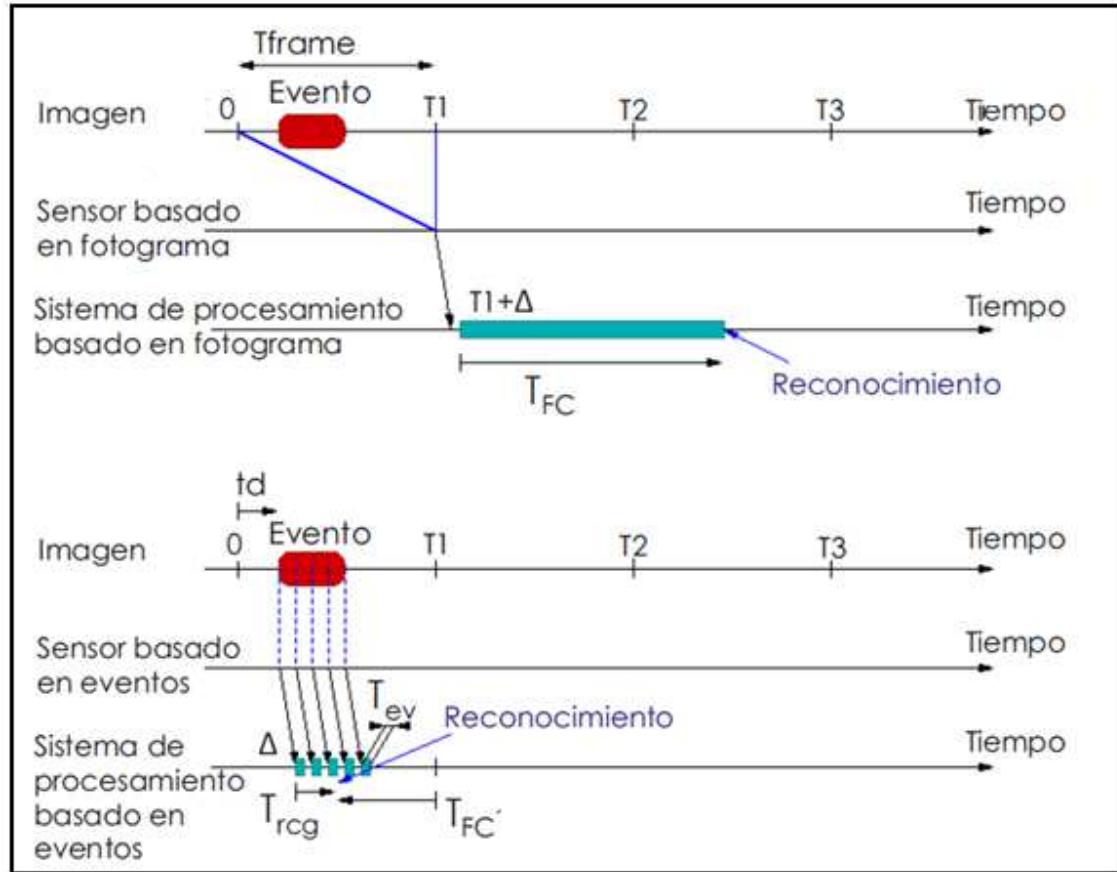


Figura 11. Comparación entre el tiempo de procesado y captación para sistemas basados en fotogramas y en eventos.

Por lo desarrollado anteriormente y dado que pretendemos implementar un sistema ligeramente bioinspirado, podemos concluir que el sistema más útil para nuestro proyecto será el basado en eventos, ya que su retraso es bastante menor. A todo esto podemos añadir que el coste computacional es también menor, ya que no es necesario procesar todos los eventos producidos para producir los resultados deseados.

3.1.2. ESQUEMAS DE CODIFICACIÓN PARA SISTEMAS BASADOS EN EVENTOS

La neurociencia, informática y los investigadores de ingeniería de aplicaciones han desarrollado y utilizado varios esquemas de codificación de la información. Vamos a resumir dos tipos de codificaciones que son las que más se han utilizado hasta ahora: la basada en la tasa de los eventos (*Rate Coding*) y la basada en el orden de los eventos (*Rank Order Coding*). Desarrollando estos esquemas explicaremos por qué nos quedaremos con el segundo.

3.1.2.1. CODIFICACION BASADA EN LA TASA DE LOS EVENTOS

La codificación basada en eventos (*Rate Coding*) es un esquema tradicional de codificación, donde la información está contenida en la tasa de eventos que estimulan la neurona. En la mayoría de los sistemas sensoriales la tasa de disparo aumenta de forma no lineal con el incremento de la intensidad del estímulo [27]. Este concepto ha sido aplicado con éxito durante los últimos 80 años y su origen se remonta al trabajo realizado por Adrian [28]. En dicho trabajo mostró que la tasa con la que disparan las neuronas receptoras, que son responsables del estiramiento en los músculos, está relacionada con la fuerza aplicada al músculo. En las décadas siguientes, la medición de las tasas de disparo se convirtió en una herramienta estándar para describir las propiedades de todos los tipos de neuronas sensoriales. Ésto fue así debido a la relativa facilidad de medición de estas tasas de forma experimental. Sin embargo, este enfoque deja de lado toda la información posible relacionada con el momento exacto de llegada de un impulso. Durante los últimos años estudios experimentales han sugerido que el concepto de tasa de disparo (basado en la tasa promedio temporal de acción en la neurona receptora) puede ser demasiado simplista para describir la actividad del cerebro [29]. Esto es así porque la codificación basada en eventos no tiene en cuenta el orden con el que llegan los estímulos externos, que son responsables de que la neurona detectora dispare con una determinada tasa.

3.1.2.2. CODIFICACIÓN BASADA EN EL ORDEN DE LOS EVENTOS

Si bien la idea de tratar una imagen usando una codificación en tasa puede parecer plausible, recientes trabajos experimentales en la retina de un ratón han hecho que se descarte. La cantidad de información contenida en la tasa de disparo de las neuronas puede ser insuficiente para explicar el comportamiento del animal [30]. En los esquemas de codificación basados en el orden de los eventos (*Rank Order Coding*), propuesto originalmente por Thorpe [31], la información está codificada en el orden relativo de llegada de los estímulos a la población de neuronas que se ven envueltas en el experimento (ver también [32]). En respuesta a un estímulo visual células de la retina se cargarán progresivamente hasta alcanzar un umbral. Ésto provocará que la neurona genere un pulso de salida como respuesta al desbordamiento del acumulador. Gracias a este disparo la neurona de salida es capaz de liberar toda la energía retenida, reiniciando así el proceso. Diversas simulaciones han demostrado que mediante el uso del orden en que disparan las neuronas que captan la imagen, es posible reconstruir el objeto identificado [33]. Ésto se ha aplicado en el estudio de la retina de la salamandra [34]. Por tanto, queda demostrado que se puede conseguir la plasticidad sináptica mediante la modulación según el orden de los eventos.

Los sistemas de codificación basados en la tasa de los eventos y el orden de los eventos se comparan en la Fig. 12. En la parte superior de la imagen, se muestran los estímulos recibidos por tres neuronas diferentes (etiquetadas como NA, NB y NC). Como se puede observar en la codificación basada en la tasa de los eventos (en la parte inferior de la ilustración), las actividades de salida de las tres neuronas varían con la intensidad del estímulo. En este esquema no es necesario utilizar señales de referencia temporales. Por el contrario, en la codificación basada en el orden de los eventos (parte central de la imagen), la información se codifica en el orden relativo en que las tres neuronas disparan. Esta vez sí es necesaria una señal de referencia temporal para indicar el inicio de nuevas ventanas de tiempo. Sin embargo, el número de pulsos de entrada requeridos es menor y por lo tanto requiere un ancho de banda menor. Gracias a este método el coste computacional del sistema disminuye en gran medida, ya que solo tiene en cuenta el momento en el que dispara cada neurona y el orden relativo de disparo entre ellas. Aunque se recorta en procesamiento, el sistema tiene que estar bien sincronizado, para que no se solapen disparos de neuronas que se han producido en slots

de tiempos diferentes. Por ser el sistema más parecido al implementado en el sistema visual humano y además ser más eficiente en ancho de banda y cantidad de información, el sistema de codificación utilizado en este proyecto es el basado en el orden de los eventos [3].

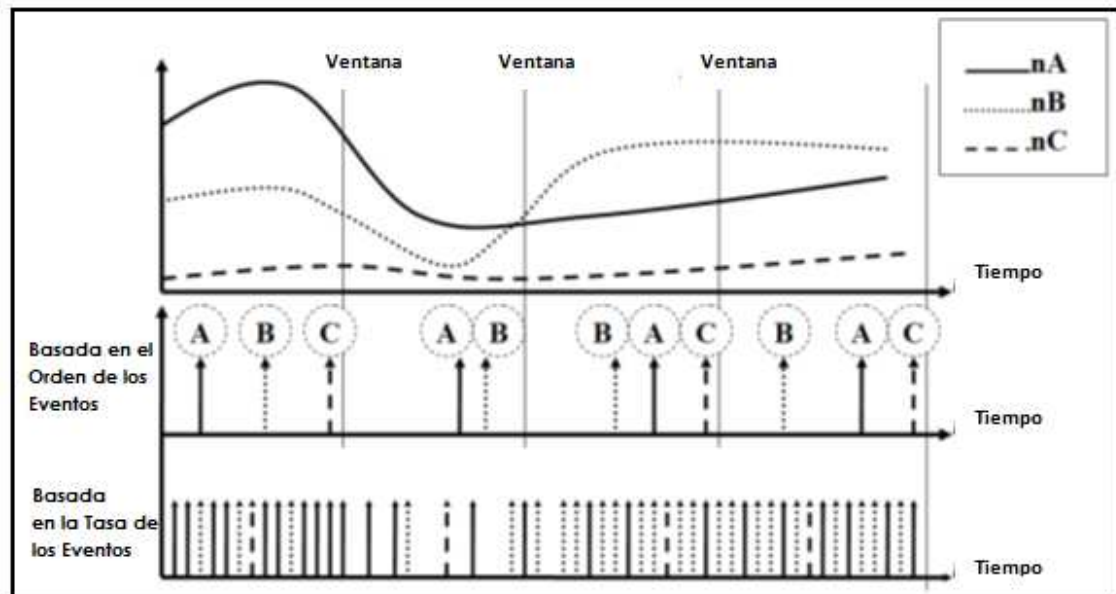


Figura 12. Esquemas de codificación basados en la tasa de los eventos frente a los basados en el orden de los eventos.

3.1.3. APLICACIÓN DEL PROTOCOLO AER PARA SISTEMAS BASADOS EN EVENTOS

Como hemos comentado anteriormente se ha elegido como mecanismo más válido para nuestro proyecto el sistema basado en eventos utilizado con una codificación según el orden en que tienen lugar los eventos, ya que nos proponen una serie de ventajas que se asemejan a la forma de trabajar del sistema nervioso visual. Esta elección no serviría para nada si la tecnología actual no permitiera la implementación de aplicaciones complejas a alta velocidad con una elevada tasa de resultados correctos.

Los sistemas de captación basados en eventos trabajan de manera similar a la retina, cada píxel envía pulsos a la corteza cerebral cuando su nivel de actividad alcanza cierto umbral. Aquellos píxeles muy activos enviarán más pulsos que los menos activos. Todos estos pulsos son transmitidos a medida que están siendo producidos, y no esperan a un tiempo artificial (*tiempo de frame*) antes de enviarlos a la siguiente etapa de procesamiento [4]. Las características extraídas son propagadas y procesadas etapa por etapa tan pronto como han sido producidas, sin esperar a finalizar la recolección y procesamiento de los datos de fotogramas completos. Un problema importante que encuentran los ingenieros cuando tratan de implementar sistemas de procesamiento de visión bioinspirados es conseguir la masiva cantidad de interconexiones hacia delante y de realimentación que aparece entre las etapas neuronales existentes en el sistema de procesamiento de visión humano. La representación de datos basada en direcciones de eventos AER (*Address Event Representation* [35] [36]) es una posible solución. Esta solución fue propuesta por primera vez en 1991 en el Instituto de tecnología de California (Caltech). Este sistema se ha utilizado desde entonces por una amplia comunidad de ingenieros de hardware neuromórficos. La Fig. 13 ilustra la comunicación en un enlace punto a punto AER tradicional cuando se utiliza codificación basada en tasa de eventos [3].

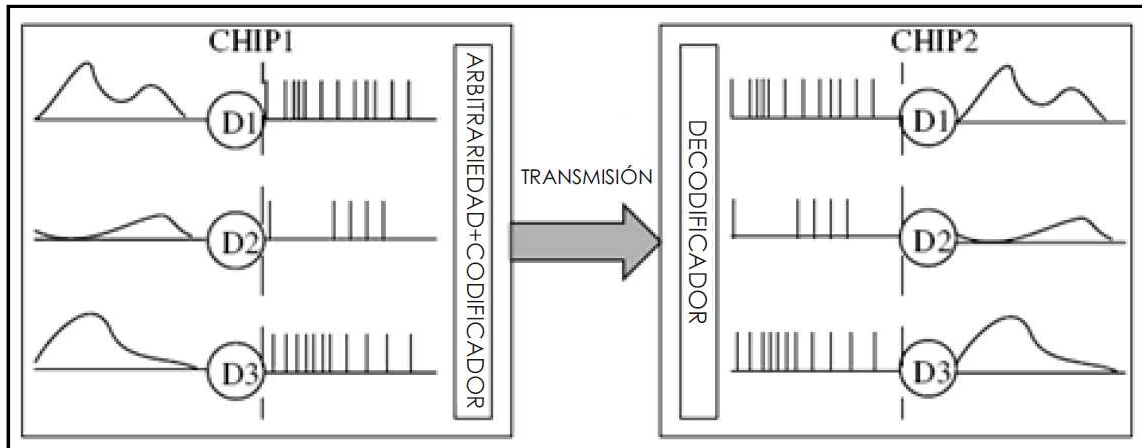


Figura 13. Comunicación interchip punto a punto aplicando protocolo AER.

En el sistema más simple AER y basado en una codificación “rate-based”, el estado continuo en el tiempo de las neuronas emisoras en un chip es transformado a una secuencia de pulsos digitales muy rápidos (eventos) de anchura mínima (del orden de ns) pero con intervalos entre pulsos del orden de ms (similar a las neuronas cerebrales). Este alto intervalo entre pulsos permite una potente multiplexación, y los pulsos generados por las neuronas emisoras pueden ser multiplexados en tiempo en un bus de salida común de alta velocidad. Cada vez que una neurona emite un pulso o evento, la dirección de esa neurona aparece en el bus digital junto con sus señales de establecimiento de la comunicación (*request* y *acknowledged*). Esto se conoce como evento de dirección. El chip receptor lee y decodifica las direcciones de los eventos entrantes y envía pulsos a las neuronas receptoras correspondientes, que integran esos pulsos y son capaces de reproducir el estado de las neuronas emisoras. Esta es la comunicación entre chips basada en AER más simple. Sin embargo, esta comunicación punto a punto puede ser extendida a un esquema multi-emisor o multi-receptor [37], donde rotaciones, traslaciones o procesamientos más complicados como convoluciones pueden ser implementados por chips de procesamiento que reciben estos eventos [38]. Además, la información puede ser trasladada o rotada fácilmente simplemente cambiando las direcciones de los eventos al tiempo que viajan de un chip al siguiente. Existe una creciente comunidad de usuarios del protocolo AER para el diseño de aplicaciones de visión y audición bioinspiradas, robótica, seguimiento y reconocimiento de objetos, etc., como ha sido demostrado por el éxito en los últimos años de los participantes en las ‘Neuromorphic Engineering Workshop series’ [39]. El éxito de esta

comunidad es diseñar sistemas grandes jerárquicamente estructurados multi-chip multi-etapa capaces de implementar procesamientos complejos de matrices en tiempo real. El éxito de tales sistemas dependerá en gran medida de la disponibilidad de herramientas robustas y eficientes de diseño y depuración de sistemas AER [37] [40].

3.2. IMPLEMENTACIÓN

En este punto se van a explicar todos los elementos intervinientes en este proyecto y además se va a desarrollar la base teórica del mecanismo bioinspirado de aprendizaje STDP. Con todo ello, hemos desarrollado una simulación de un sistema artificial de reconocimiento de dígitos a través de programa del cálculo Matlab.

3.2.1. DESCRIPCIÓN DEL SISTEMA

El sistema implementado en este trabajo está compuesto principalmente por dos etapas, como se puede observar en la Fig. 14. La primera de ellas es un banco de filtros de Gabor (1 escala y 6 orientaciones) y la segunda una red neuronal mono-etapa compuesta por diez neuronas de salida. El nexo de unión entre las dos etapas son las sinapsis, que actúan como auténticos circuitos neuronales que conducen los impulsos detectados. Las imágenes de salida de los filtros de Gabor (salida de la primera etapa) están formadas por píxeles, y cada uno de ellos tiene una conexión (sinapsis) con cada una de las neuronas de salida. Según las uniones que se activen nuestro sistema reconocerá una imagen u otra. Teniendo en cuenta esto podemos diferenciar dos etapas [23]:

- **Acción Temporal Presináptica:** La acción ocurre en la primera etapa de nuestro sistema. Los píxeles a la salida de los filtros de Gabor envían pulsos cada vez que un evento es captado. A su vez, dichos píxeles realizan una codificación intensidad-tiempo de la señal percibida antes de transmitirla en forma de pulso a través de las sinapsis.
- **Acción Temporal Postsináptica:** La acción la lleva a cabo una neurona final conectada a los píxeles mediante las sinapsis. Dicha neurona actúa como un auténtico detector de coincidencia, sólo reacciona cuando reconoce una imagen aprendida previamente.

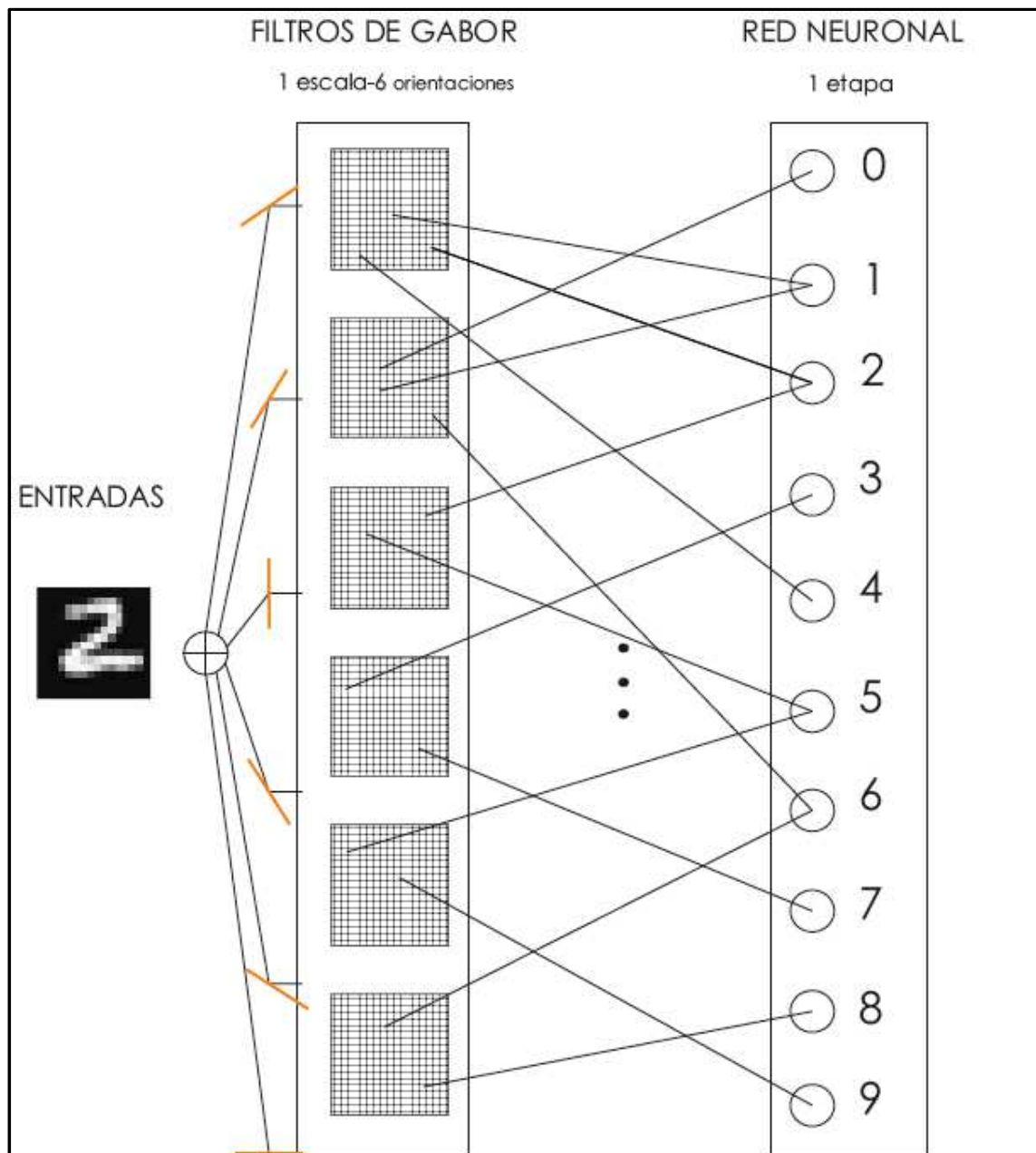


Figura 14. Sistema bioinspirado de aprendizaje de dígitos.

El sistema recibe como entrada dígitos numéricos del rango [0,9], obtenidos de la base de datos MNIST [14]. En la Fig. 15, se muestran ejemplos de los 10 dígitos que representan cada uno de los números naturales, siendo una buena herramienta para el uso en sistemas de reconocimiento y aprendizaje.

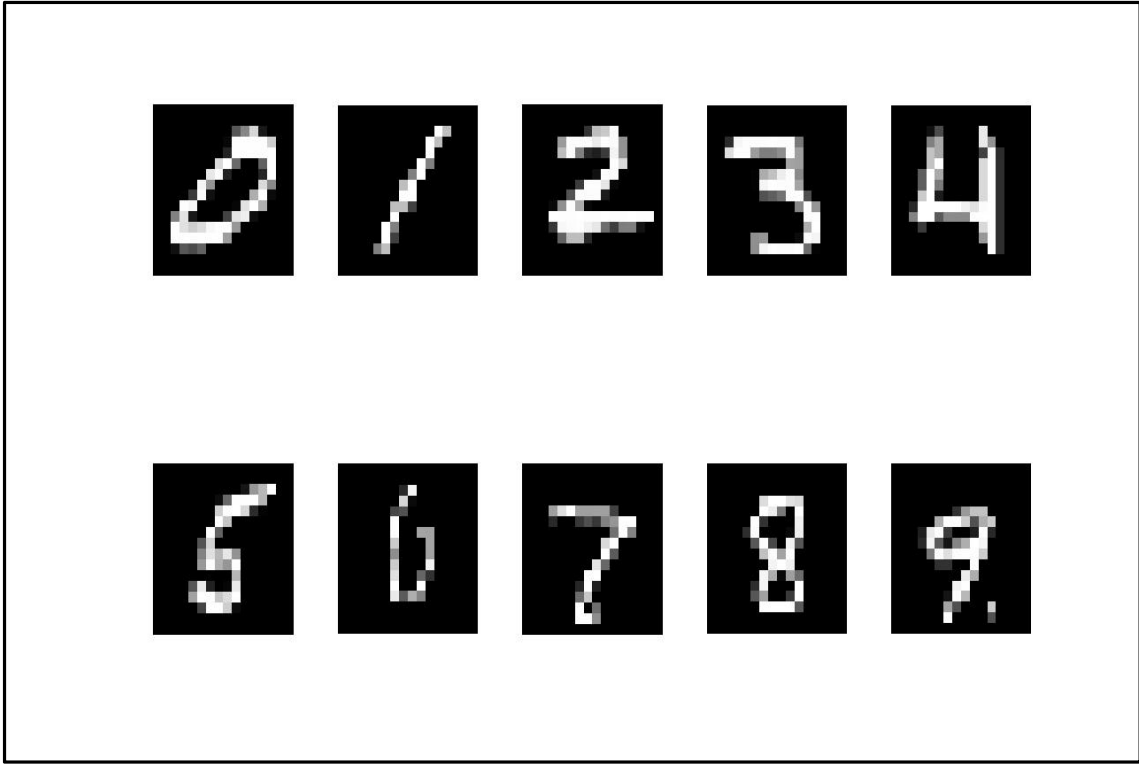


Figura 15. Números naturales de la base de datos MNIST.

El tamaño de las imágenes de entrada es de 32x32, aunque para nuestra aplicación estas imágenes han sido submustradas para tener un tamaño de 16x16. Dichas imágenes son convolucionadas con un conjunto de 6 filtros de Gabor, que implementan una escala y 6 orientaciones. El tamaño de los filtros de Gabor es 10x10. Cada uno de estos filtros se obtiene mediante las siguientes expresiones [3][41]:

$$g(x, y) = \frac{1}{2\pi \cdot \sigma_x \sigma_y} \cdot \exp\left(-\frac{1}{2} \left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right) + j2\pi \cdot Wx\right)$$

Realizando unas determinadas dilataciones y rotaciones a la función de transferencia $g(x, y)$ obtenemos cada filtro. Cada uno codifica una orientación:

$$g_{s,k}(x, y) = a^{-s} \cdot g(x', y')$$

$$x' = a^{-s}(x \cdot \cos \theta + y \cdot \sin \theta)$$

$$y' = a^{-s}(-x \cdot \sin \theta + y \cdot \cos \theta)$$

En estas expresiones θ representa la orientación y S la escala. Los parámetros $\{a, \theta, W, \sigma_x, \sigma_y\}$ son descritos en el método implementado por Manjunath [42]. En la Fig. 16 podemos observar gráficamente las orientaciones de los filtros de Gabor obtenidos a través de las dilataciones y rotaciones mencionadas anteriormente.

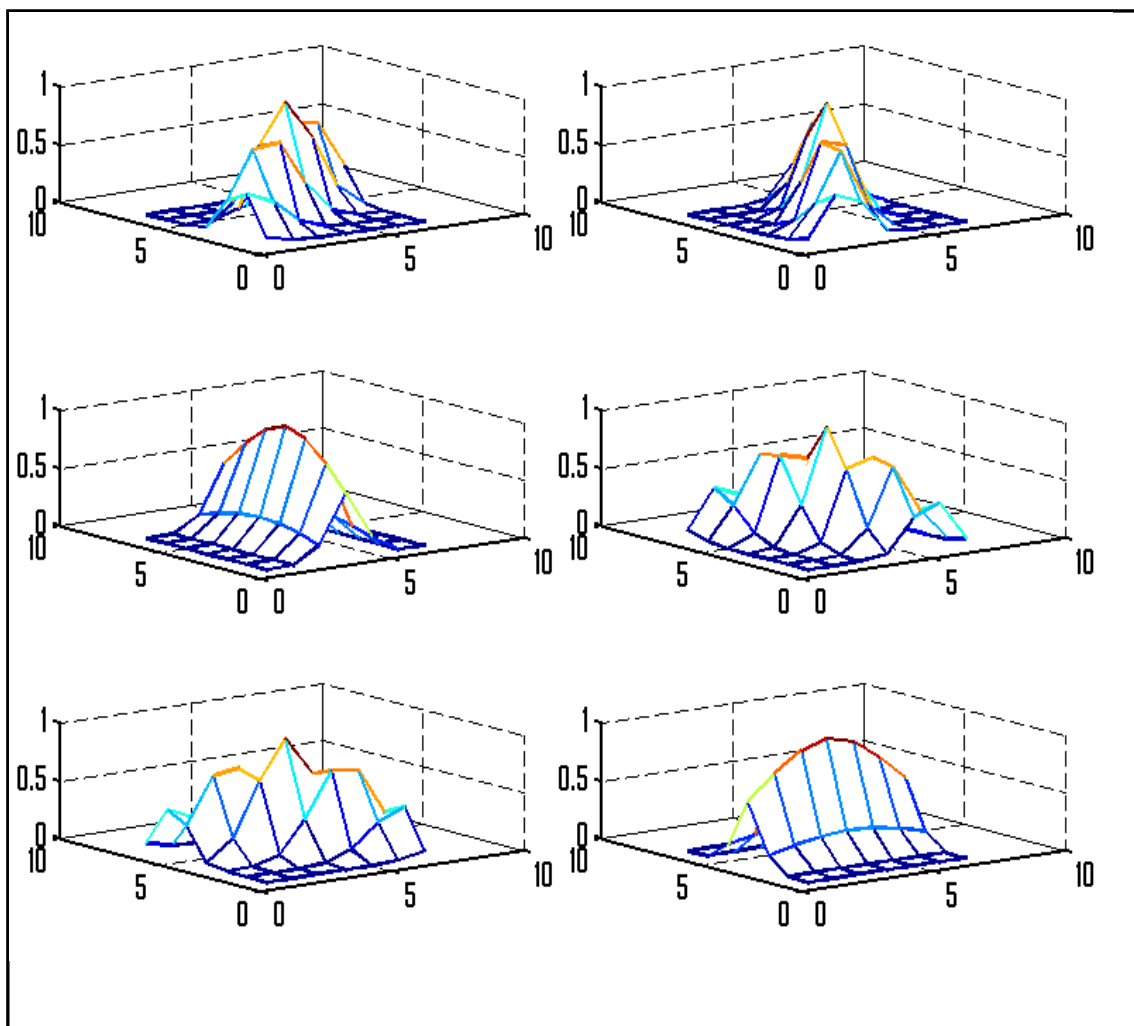


Figura 16. Representación de las 6 orientaciones de los filtros de Gabor.

Para eliminar información redundante y reducir parte del cálculo numérico, se han realizado una serie de simplificaciones en nuestra subrutina codificada en el programa Matlab. Buscando eliminar cualquier elemento superfluo, tras cada convolución de la imagen de entrada con cada filtro de Gabor hemos reducido su tamaño a una matriz 10x10. Se ha comprobado experimentalmente que este conjunto de

píxeles contiene la mayor parte de la información, ya que en la periferia casi todos los valores son iguales a cero (dichos valores no aportan información).

Con la convolución se busca encontrar cuales son las componentes direccionales más dominantes en cada número. Por ejemplo, para el dígito “1” podemos intuir con facilidad que su componente más potente es la vertical. Por tanto, la convolución entre este dígito y la orientación vertical del filtro de Gabor da como resultado el mayor valor posible. El resultado de la convolución es tan alto debido a la similitud existente. Este tipo de rasgos nos ayudan a diferenciar un dígito de otro distinto. Así que podemos utilizar las diferencias existentes en las componentes espaciales y de orientación de cada dígito para lograr que cada neurona de salida aprenda un determinado patrón [41].

En la Fig. 17 muestra un ejemplo de la operación de convolución entre una imagen de entrada (en este caso el dígito “0”) y las 6 componentes de los filtros de Gabor. El resultado muestra la relación de similitud entre el dígito “0” y las 6 orientaciones de los filtros de Gabor. Para este caso se aprecia que el dígito “0” no tiene ninguna componente direccional que destaque sobre las demás, ya que aparecen componentes verticales, horizontales e inclinadas de manera equitativa. Esta característica es útil a la hora de clasificar el aprendizaje del dígito “0”, ya que su patrón está compuesto por todas las componentes existentes.

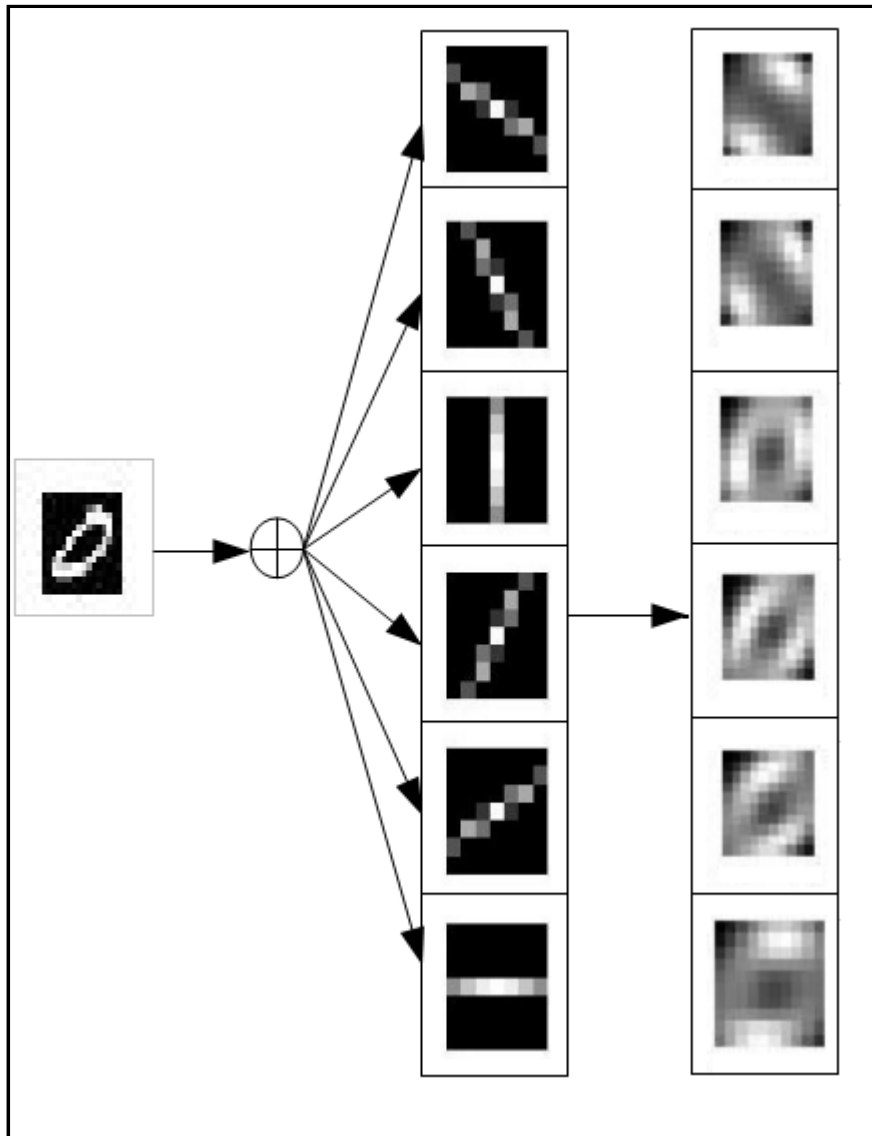


Figura 17. Implementación del sistema bioinspirado.

Como para cada dígito no intervienen todas las orientaciones espaciales (y si intervinieran la cantidad captada para alguna de las orientaciones será ínfima) se pueden descartar diferentes niveles de intensidad sin pérdida de información. En nuestro experimento hemos optado por despreciar aquellos valores a la salida de los Gabor que no superan el 75% del valor más alto para una imagen de entrada. Esta reducción no solo viene motivada por la continua búsqueda de simplificar el sistema para ahorrar en coste computacional, sino también para evitar posibles errores en el aprendizaje de la neurona de salida. Hay que tener cuidado con las contribuciones indeseadas de algunas componentes espaciales que no son características del dígito entrenado, ya que pueden

llevar a respuestas incorrectas en la neurona de salida. Ésto puede desembocar en respuestas erróneas como las falsas alarmas (disparos en instantes erróneos que no se corresponden con la correcta detección del patrón entrenado).

Otra operación realizada consiste en comparar los píxeles de salida de cada filtro de Gabor que coinciden espacialmente. Al usar 6 filtros de Gabor tenemos 6 píxeles por posición (que detectan diferentes orientaciones). Sólo vamos a conservar la información aportada por aquel píxel que contenga un mayor nivel de intensidad. Se desecha la información aportada por el resto de píxeles, ya que podemos considerarla redundante. De este modo se implementa una operación MAX entre las distintas orientaciones y conseguimos un ahorro computacional considerable, sin que afecte al resultado final de todo el proceso constructivo del sistema.

En la segunda etapa de nuestra implementación es donde interviene el sistema de aprendizaje STDP. Este mecanismo biológico de aprendizaje se basa en los tiempos relativos entre los pulsos producidos por una neurona y los que recibe a su entrada y que pueden ser los que le han hecho activarse y disparar. Es por ello que en nuestra implementación necesitamos codificar el valor de cada píxel a la salida de cada convolución en valores temporales. En nuestro sistema hemos realizado una codificación intensidad-tiempo según el orden de llegada de los eventos (*Rank Order Coding* [31]). Cada píxel activo a la salida de los filtros de Gabor (perteneciente todavía a la primera etapa) codifica un pulso en el dominio del tiempo, cuyo valor será inversamente proporcional al valor de intensidad del píxel. Así, un píxel con valor de intensidad cercano a 0 producirá un pulso “tardío” y un píxel con valor de intensidad cercano a 255 producirá un pulso casi instantáneo. Estos pulsos se enviarán a las neuronas de salida a través de las sinapsis.

El tiempo de creación de un pulso vendrá relacionado (inversamente) con el nivel de píxel, según la expresión:

$$t_{pulso} = K \cdot (1 - pixel/255)$$

En nuestra implementación t_{pulso} tiene un valor máximo de 2,5 ms. En la Fig. 18, se representa la curva que modela la codificación en tiempo de los niveles de intensidad de los píxeles activos:

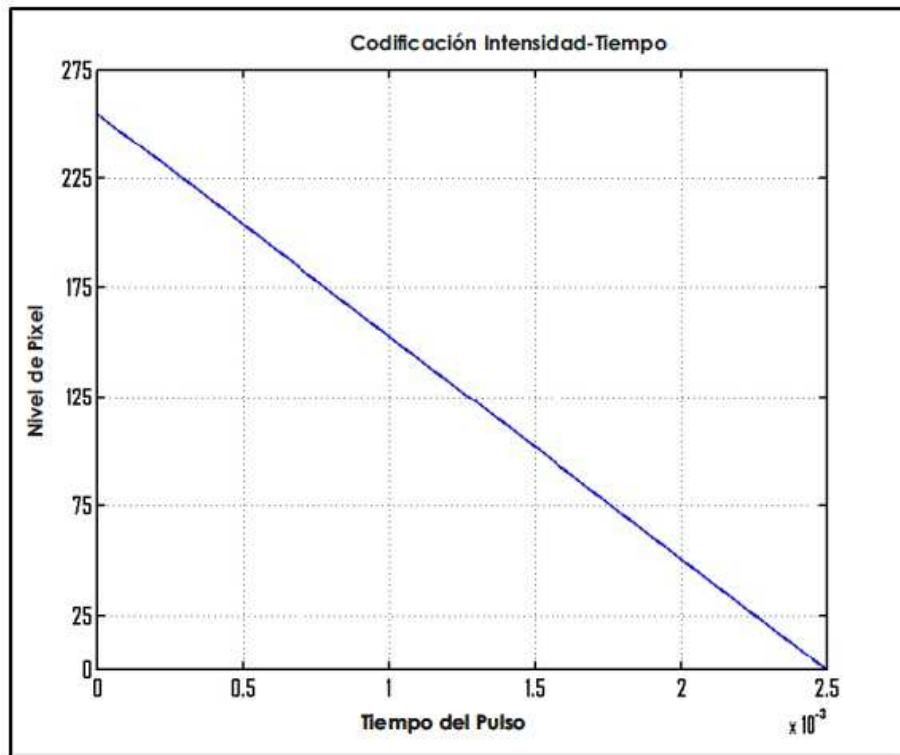


Figura 18. Codificador Intensidad-Tiempo.

En la Fig. 19 se muestra un ejemplo de la codificación del nivel de intensidad de los píxeles en tiempo, como podemos apreciar la relación existente entre ellos es inversamente proporcional. Por tanto, aquellos píxeles que sufren un nivel de intensidad alto toman un valor menor cuando se convierten al dominio del tiempo. Por el contrario, si el valor del nivel del pixel es bajo su transformación al dominio del tiempo adquiere un valor muy alto. Este ejemplo es meramente ilustrativo, ya que realmente los valores no se corresponden con los codificados en este proyecto.

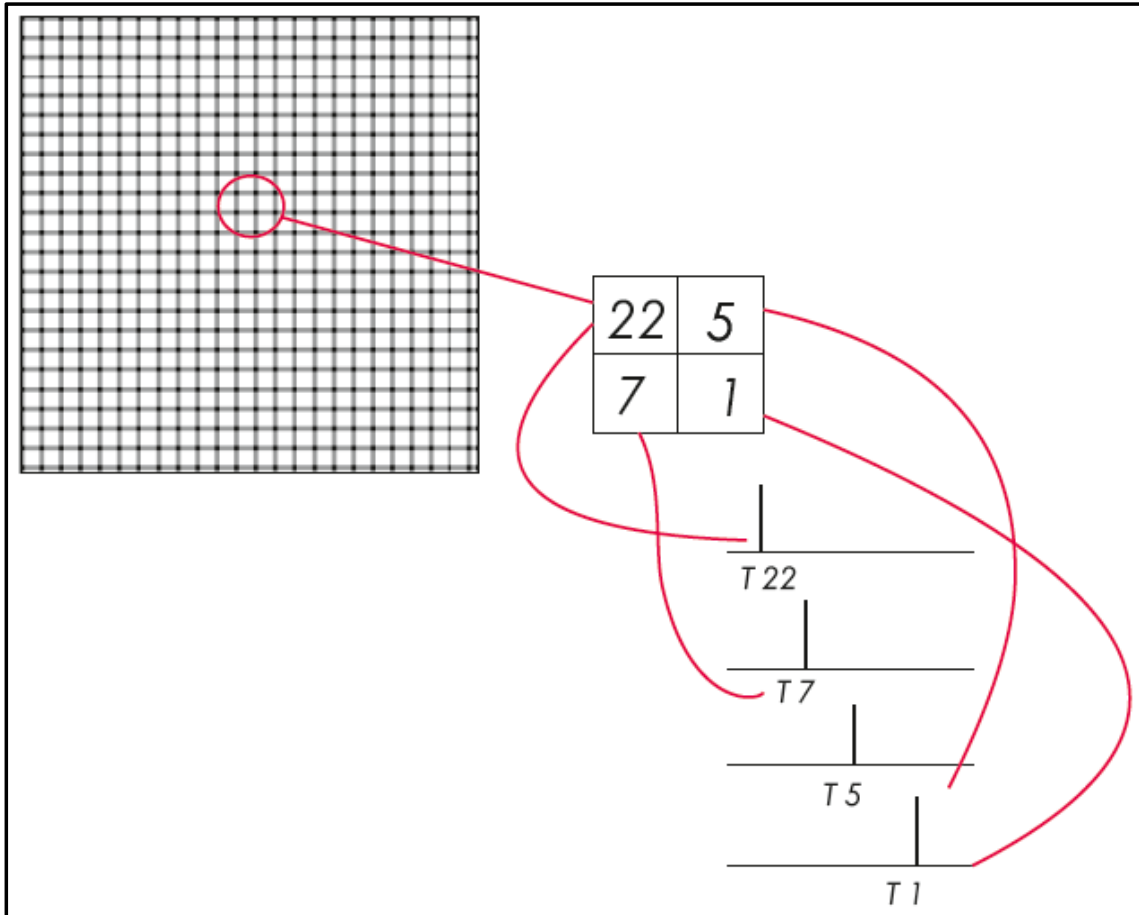


Figura 19. Conversión de nivel de píxel en tiempo.

En nuestro sistema, los píxeles de salida de la primera etapa están conectados con cada una de las 10 neuronas de salida de la segunda etapa. Por lo cual, tenemos 6x100x10 (6 filtros de Gabor, 100 píxeles de salida de cada filtro y 10 neuronas de salida) que hace un total de 6000 conexiones (sinapsis) en nuestro sistema bioinspirado neuronal. Tras la codificación temporal los píxeles envían los pulsos a cada neurona a través de cada sinapsis, empezando así el periodo de aprendizaje y competición entre ellas.

3.2.2. MECANISMO DE APRENDIZAJE STDP

En nuestra implementación cada neurona de salida ha sido entrenada por separado. En una implementación futura y más eficiente se permitirá que todas las neuronas aprendan a la vez. Si además permitiésemos *inhibición* entre ellas conseguiríamos que cada neurona aprendiera un dígito diferente [13]. Por tanto, para la implementación aquí presentada, cada neurona de salida i , correspondiente al dígito de entrada i ($i=0,...,9$), ha sido entrenada por separado utilizando un conjunto de 5000 imágenes correspondientes al mismo dígito. Una vez entrenadas, las neuronas se hacen selectivas al estímulo de entrada, es decir, la respuesta a un dígito que no sea el entrenado es mucho más lenta que si fuera el patrón aprendido. Esta lentitud en la resolución nos lleva a concluir que nuestra neurona no ha memorizado ese dígito.

Pero nuestra pregunta sigue sin ser respondida, ¿cómo se implementa el aprendizaje basado en mecanismos bioinspirados? Esta pregunta la podemos responder gracias al postulado de Hebb (vínculo de causalidad) [43] [44]:

“Se refuerzan aquellas conexiones en la que las neuronas presinápticas disparan un poco antes que las neuronas postsinápticas.”

La primera etapa de nuestro sistema, mostrada en la Fig. 14, implementa las funciones que realizan las neuronas presinápticas. En los sistemas biológicos esta etapa está constituida por las neuronas aferentes (sensorial) cuyo cometido es captar la imagen. Por otro lado, las neuronas postsinápticas se corresponden con las neuronas finales de nuestro sistema, equivalente a la segunda etapa de la Fig. 14. Esta etapa tiene la misión de memorizar el dígito, actuando las neuronas de salida como auténticos detectores de coincidencia.

Así que gracias a la cita anterior podemos afirmar que el mecanismo de aprendizaje STDP (*Spike-Timing-Dependent-Plasticity*) se basa en el principio en el cual la fuerza de las conexiones con las neuronas de salida (“pesos”) se van reforzando o debilitando con la repetición de unos pulsos de entrada determinados [13] [23]. Aquellas conexiones cuyos pulsos transportados provoquen un disparo en la neurona de salida se verán reforzadas. En cambio, aquellas conexiones por las que viajan pulsos

que llegan a la neurona de salida posteriormente al disparo de la misma se verán debilitadas. STDP es un mecanismo *no supervisado*, es decir, las neuronas aprenden de manera autónoma ante la repetición de los patrones de entrada. Gracias a este mecanismo, cuando la neurona haya aprendido un patrón de entrada le bastaran muy pocos pulsos para producir un disparo.

El mecanismo STDP se resume del siguiente modo: Inicialmente el peso de todas las conexiones es el mismo (o puede estar inicializado aleatoriamente), pero cada vez que la neurona de salida produce un pulso los pesos de las conexiones se actualizan de forma positiva o negativa (según en qué instante haya recibido el pulso de entrada la neurona de salida, antes o después de su propio disparo). Los pesos contribuyen de manera acumulativa al potencial de la neurona de salida. Si este potencial alcanza un umbral determinado se produce una reacción en forma de disparo. Una vez que nuestro detector de coincidencia (neurona de salida) responde al estímulo de entrada con un disparo, el acumulador de la neurona entrenada se vacía. Todo este proceso de actualización de los pesos de las sinapsis se rige según las expresiones siguientes [13] [23]:

$$w_o = \Delta w_{inst} + w_{ant}$$

$$\Delta w_{inst} = \begin{cases} a^+ \cdot \exp\left(\frac{t_j - t_i}{\tau^+}\right); & t_j \leq t_i \\ -a^- \cdot \exp\left(-\frac{t_j - t_i}{\tau^-}\right); & t_j > t_i \end{cases}$$

En estas ecuaciones w_o es el peso actualizado de la sinapsis una vez que la neurona de salida dispara. Consiste en la suma de w_{ant} (es el peso anterior al disparo del detector de coincidencia asociado a una de las sinapsis) y Δw_{inst} (es el incremento de peso debido al disparo de la neurona final). Los parámetros a^+ , a^- , τ^+ y τ^- son constantes predefinidas [3], que toman los valores siguientes en nuestra práctica:

$$a^+ = 0.03125;$$

$$a^- = 0.25 \cdot a^+;$$

$$\tau^+ = 16.8e - 3;$$

$$\tau^- = 90e - 3;$$

Por último, el valor $t_j - t_i$ representa la diferencia de tiempo entre el pulso producido por los píxeles activos a la salida de los filtros de Gabor en el instante t_j (cuya conexión con la neurona de salida se está actualizando) y el disparo producido por la neurona final en el instante t_i . La Fig. 20 representa la curva de actualización de pesos. En ella se puede ver que aquellas conexiones cuyos pulsos sean anteriores al disparo de la neurona de salida (y por tanto causantes del mismo) se verán reforzadas. Esto se corresponde con la curva de la parte izquierda, conocida como etapa de potenciación o aprendizaje de la neurona LTP (*Long Term Potentiation*). En cambio, los pesos de aquellas conexiones con pulsos posteriores al disparo de la neurona de salida se verán atenuados. Esto se corresponde con la curva de la derecha, siendo conocida como etapa de depresión u olvido de lo aprendido, LTD (*Long Term Depression*) [23].

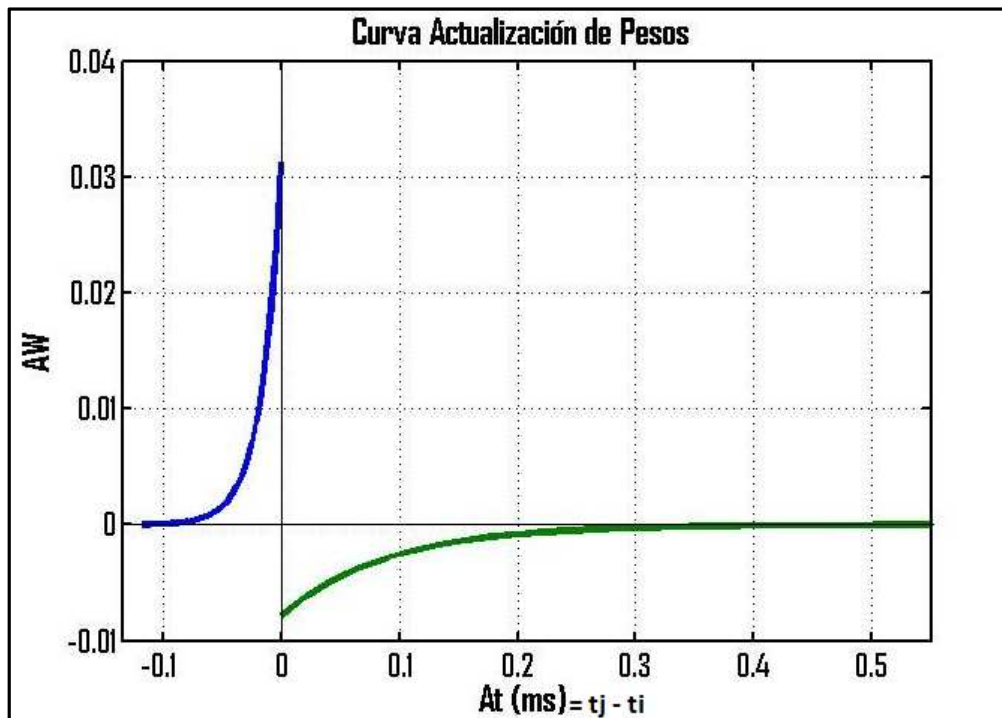


Figura 20. Curva aprendizaje STDP.

Esta actualización de pesos tiene su sentido, ya que si una neurona de salida aprende a reconocer un determinado dígito habrá un grupo de sinapsis que tengan asociadas un mayor peso. Estas sinapsis tendrán una mayor probabilidad de activarse cuando llegue una entrada que sea el dígito entrenado. Podemos utilizar ésto como un factor diferenciador, ya que nos ayuda a que nuestro sistema consiga memorizar un patrón determinado de manera autónoma y rápida (a mayor peso asociado a una conexión mayor probabilidad que la neurona de salida dispare de manera instantánea obviando el resto de enlaces).

En la Fig. 21 podemos observar un ejemplo sobre la actualización de los pesos asociados a las sinapsis, para ello utilizamos la curva de aprendizaje STDP. Los puntos rojos significan aquellos píxeles activos en la etapa 1 (salida de los Gabor). En nuestro modelo, el punto rojo que aparece circulado se corresponde con el píxel activo que envía un pulso (codificado en el dominio del tiempo) a través de una sinapsis y que causa el disparo de la neurona de salida. Podemos decir que la sinapsis que transporta este pulso es casi la principal culpable de que nuestro detector de coincidencia dispare, por eso el incremento de peso será máximo. Los eventos que llegan antes que nuestro sistema responda, provocan una actualización positiva pero en menor medida. Esto se corresponde con la etapa de potenciación de las conexiones involucradas. En cambio los estímulos posteriores a la respuesta final del sistema se ven claramente envueltos en una etapa de depresión, disminuyendo el valor de los pesos asociados a las sinapsis que transportan dichos pulsos.

Si nos detenemos un poco más en la curva de actualización de los pesos, podemos apreciar que hemos premiado con mayor intensidad y pendiente la etapa de potenciación o aprendizaje (parte de la izquierda de la curva) con respecto al periodo de depresión dónde se empieza a olvidar la regla memorizada (parte de la derecha de la curva). Esto es así, ya que tras un cierto periodo de aprendizaje la respuesta del sistema es automática e instantánea. Cuando la neurona de salida haya sido entrenada durante un tiempo, la neurona disparará tras la llegada de unos pocos pulsos tempranos, debido al alto valor de los pesos de aquellas sinapsis que los transportan. Las sinapsis que apenas transporten pulsos o que vengan con mucho retardo irán disminuyendo el valor de sus pesos poco a poco. Estas sinapsis son mayoritarias con respecto a las que intervienen activamente en el disparo de la neurona de salida. Es posible que alguna vez, sinapsis

que suelen mandar pulsos tempranos los envíen posteriormente a aquéllas que producen la activación a la salida, actualizándose sus pesos negativamente. Así que para que no haya muchos errores hemos decidido que la intensidad y pendiente de la curva de depresión sea menor que la de potenciación. Por tanto, la actualización de los pesos en la parte de la curva de depresión es mucho más suave.

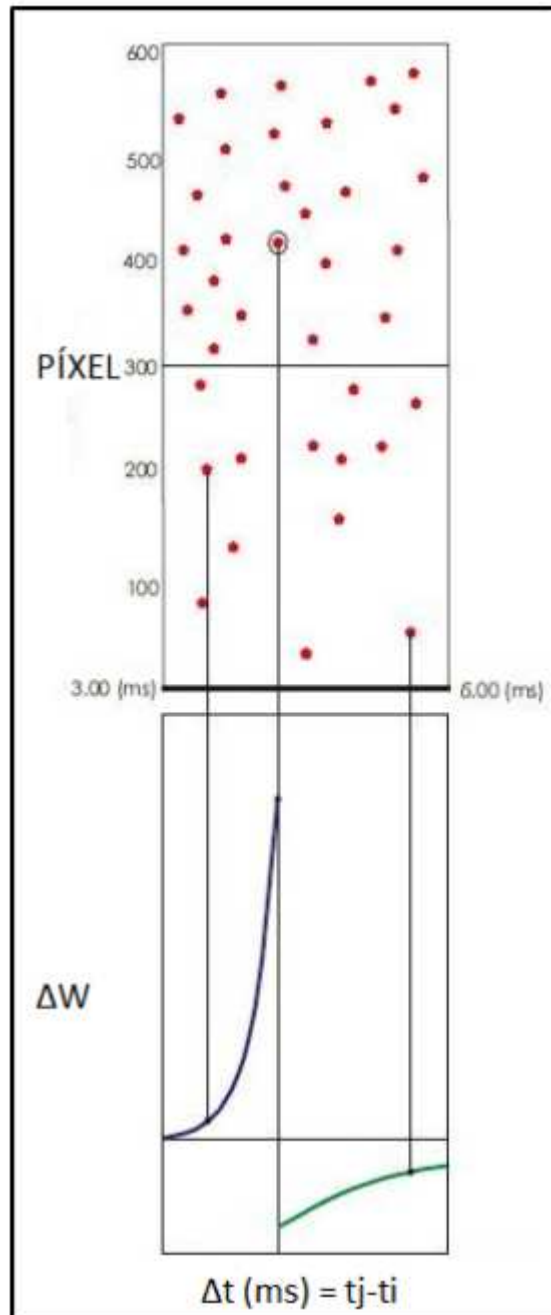


Figura 21. Ejemplo aplicado curva aprendizaje STDP.

3.3. RESULTADOS EXPERIMENTALES

Podemos diferenciar dos fases en nuestro experimento. La primera fase se corresponde con el entrenamiento de cada neurona de salida por separado. La segunda fase se produce cuando la neurona de salida ha interiorizado el entrenamiento. Podemos decir que cada neurona ha memorizado el dígito con el que ha sido entrenada.

En la Fig. 22, hemos representado la fase de entrenamiento correspondiente a la neurona “1”. El eje y de la gráfica superior muestran los píxeles activos que disparan un pulso codificado en el dominio del tiempo (se corresponden con la salida de la imagen convolucionada con el banco de filtros de Gabor). Como recordamos, había 600 píxeles posibles unidos a cada neurona de salida. El pulso producido por cada uno de estos píxeles (sólo se han representado aquellos con un valor superior a un nivel predefinido) se ha representado como un punto rojo. A su vez, aquellos que aparecen circulados son los desencadenantes de que el acumulador de la neurona de salida se desborde y provoque la respuesta de nuestro sistema. El eje x codifica el tiempo en segundos. Cada imagen captada produce una serie de pulsos en una ventana de duración 3 ms (recordemos que el tiempo utilizado en la codificación intensidad-tiempo es de 2,5 ms pero dejamos un margen de 0,5 ms para que no haya solapamiento entre las imágenes detectadas). La imagen inferior, representa el potencial acumulado por la neurona de salida que se está entrenando. Se puede observar en esta etapa que la neurona necesita cargar durante un tiempo largo (del orden de 22ms, ~8 imágenes) la información correspondiente a los patrones para producir un pulso de salida y descargar su acumulador de potencial. En esta etapa inicial no tenemos la certeza de que cuando se produce la respuesta del sistema sea realmente porque se ha detectado un patrón, ya que es posible que haya ocurrido una falsa alarma debida a la acumulación de pesos (recordamos que todas las conexiones al inicio tienen el mismo peso asociado). Esto se va corrigiendo con la repetición sucesiva de entradas. En el momento del disparo entra en juego la actualización de los pesos de cada conexión, contribuyendo así a que en un futuro la neurona de salida detecte rápidamente el patrón de entrada.

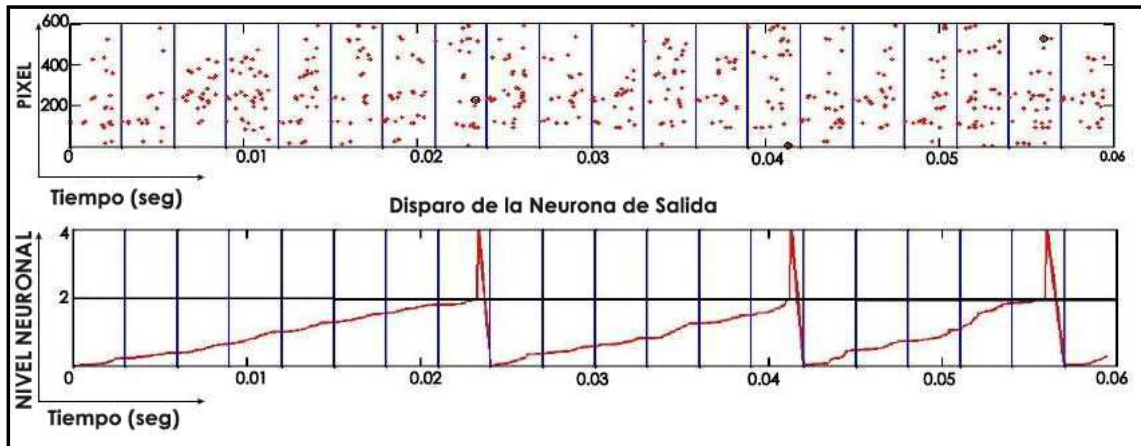


Figura 22. Etapa de entrenamiento de la neurona '1'.

Poco a poco, conforme van llegando las imágenes que representan el dígito “1” el sistema va evolucionando, cada vez habrá menos slots donde no se produzca disparo. Debido al aprendizaje nuestro detector se hace poco a poco más selectivo y le cuesta menos reconocer el patrón entrenado evitando así la reproducción de falsas alarmas.

En la Fig. 23 se muestra la etapa final del entrenamiento. Se puede observar que la neurona sólo necesita unos poquitos pulsos para activarse y producir un disparo. Ésto es prueba de que ya se ha entrenado y ha reconocido el patrón a la entrada (del orden de décimas de ms). La neurona ya ha memorizado el dígito. Además la fiabilidad aumenta, como veremos en puntos posteriores de esta memoria. Podemos apreciar que cada vez que llega una imagen que contiene el patrón entrenado la salida dispara (en cada slot de la Fig. 23 se produce una respuesta del sistema). En esta etapa disminuye considerablemente la ocurrencia de falsas alarmas, ya que al detectar tan rápidamente el patrón memorizado no hay posibilidad de que algún elemento extraño al patrón produzca confusión.

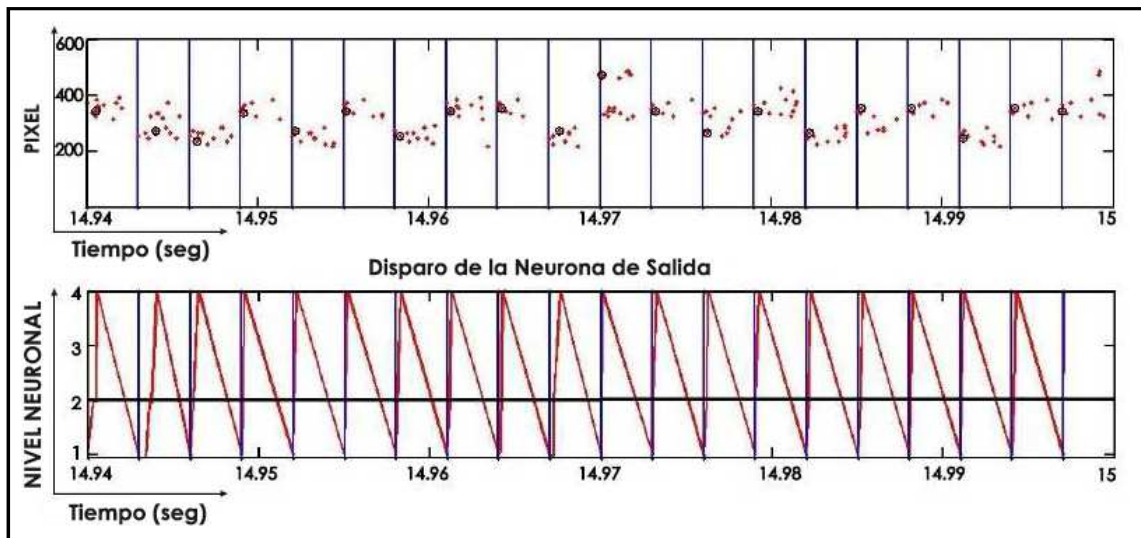


Figura 23. Etapa de entrenamiento de la neurona '1' al final.

Una vez que el sistema asimila el aprendizaje podemos observar como el grupo de píxeles responsables de la activación de la respuesta del sistema suelen ser vecinos, ya que se corresponden con la región donde la componente espacial de Gabor es más fuerte para ese dígito. Por ejemplo en el dígito “1” las componentes más fuertes son las componentes verticales. Por tanto, nuestro mecanismo de aprendizaje se vuelve muy selectivo, sólo responde ante aquellos píxeles que tienen una participación activa en el patrón a memorizar. En este punto sería necesario aplicar el concepto de inhibición si todas las neuronas del sistema compitieran durante la etapa de aprendizaje (esto no se desarrolla en el presente proyecto), ya que como hemos dicho la componente principal del dígito “1” es la vertical al igual que la del dígito “7”. Si la entrada fuera un “7” está claro que la neurona entrenada con el dígito “1” podría ganar frente a la que estuviera aprendiendo a reconocer el dígito “7”. Para que no ocurra esto, la neurona entrenada con el dígito “7” debe mandar una señal de inhibición a la neurona “1” cuando detecte la componente horizontal del número “7”. Así solucionaríamos el problema de solapamiento de aprendizaje entre neuronas.

En Fig. 24 hemos representado todos los píxeles activos culpables de los disparos de la neurona de salida al introducir 5000 entradas que representan el dígito “1”. Esta imagen las hemos relacionado a su vez con los 6 filtros de Gabor utilizados

(parte derecha de la imagen), consiguiendo así localizar las componentes predominantes en el entrenamiento del dígito “1”. Observamos claramente que las componentes vertical e inclinada son las más repetidas, correspondiéndose con los píxeles comprendidos entre [300, 400] y [200, 300]. Esto concuerda con las componentes espaciales y de orientación del dígito “1”.

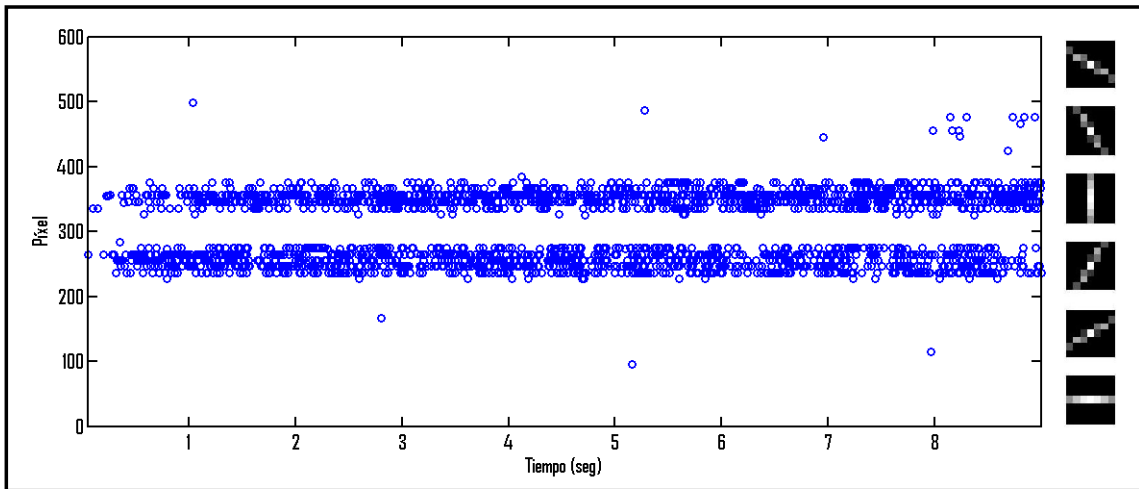


Figura 24. Píxeles activos que provocan el disparo de la neurona de salida cuando se entrena el sistema con el dígito “1”.

En cambio para el caso de aprendizaje del dígito “0” intervienen casi todas las componentes (horizontales, inclinadas y verticales). En la Fig. 25 podemos ver como contribuyen todas las componentes en la respuesta del sistema. Que colaboren todas las orientaciones de los filtros de Gabor no tiene porque ser precisamente malo, ya que podemos utilizar esta característica como un factor diferenciador. Aún así se puede observar que las componentes inclinadas son las más potentes, ya que son las mayores responsables de los disparos de la neurona final.

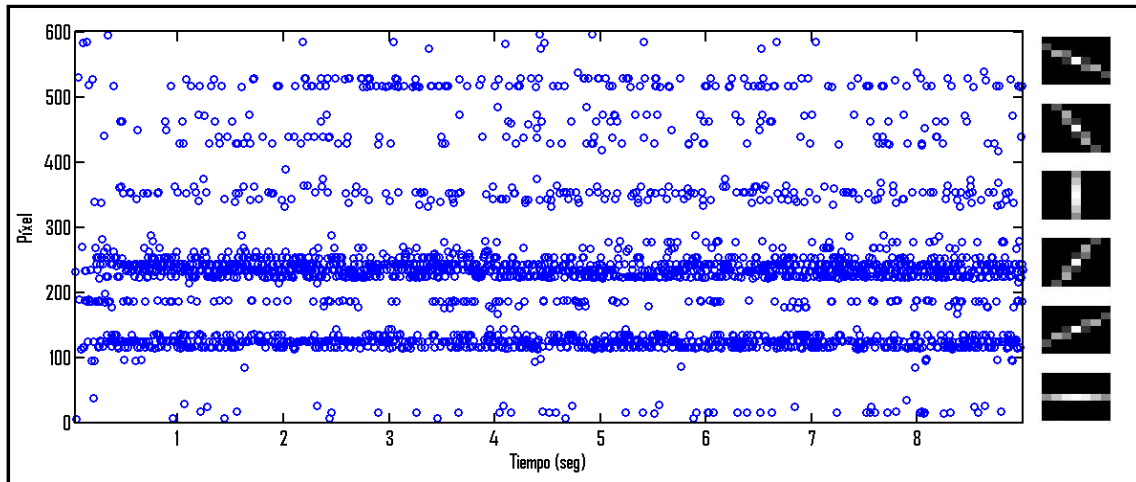


Figura 25. Píxeles activos que provocan el disparo de la neurona de salida cuando se entrena el sistema con el dígito “0”.

Como hemos comentado antes, una de las principales características del mecanismo implementado es la rápida respuesta dada por la neurona de salida cuando ha memorizado una determinada imagen. En la Fig. 22, se aprecia como la neurona de salida tarda una media de 22 ms en disparar ante el continuo bombardeo de imágenes que representan el número natural “1”, siendo un valor muy alto. Debido a la inexperiencia del sistema los pesos asociados a nuestras sinapsis son pequeños e iguales, por lo que los pulsos que recorren las conexiones les cuesta mucho desbordar el umbral del acumulador de la neurona salida y cuando lo consiguen puede que la respuesta no sea correcta (existe la posibilidad que sea una falsa alarma al ser todos los pesos parecidos). En cambio pasa todo lo contrario en la Fig. 23, dónde nuestro modelo ya está muy experimentado y ya existe una gran diferencia entre los pesos de las sinapsis. Habrá sinapsis con pesos elevados que serán responsables del disparo de la neurona de salida. Se reduce drásticamente el tiempo de respuesta a unos pocos milisegundos, esto es, la memorización se ha alcanzado.

La Fig. 26 representa la diferencia de tiempos existentes entre disparos sucesivos de la neurona de salida conforme se va entrenando con el dígito “1”. En el eje y se representa la diferencia de tiempo que tiene lugar entre dos disparos sucesivos y en el eje x se representa el numero de disparos (es por ello que toma valores discretos). Por ejemplo el valor ($x=5$, $y=0.004$) significa que la diferencia de tiempo que existe entre los disparos que ocurren en el quinto y cuarto lugar es de 4 ms. Se observa

perfectamente que el tiempo inicial es alto, conforme la neurona de salida va asimilando el aprendizaje la diferencia de tiempo entre disparos va disminuyendo.

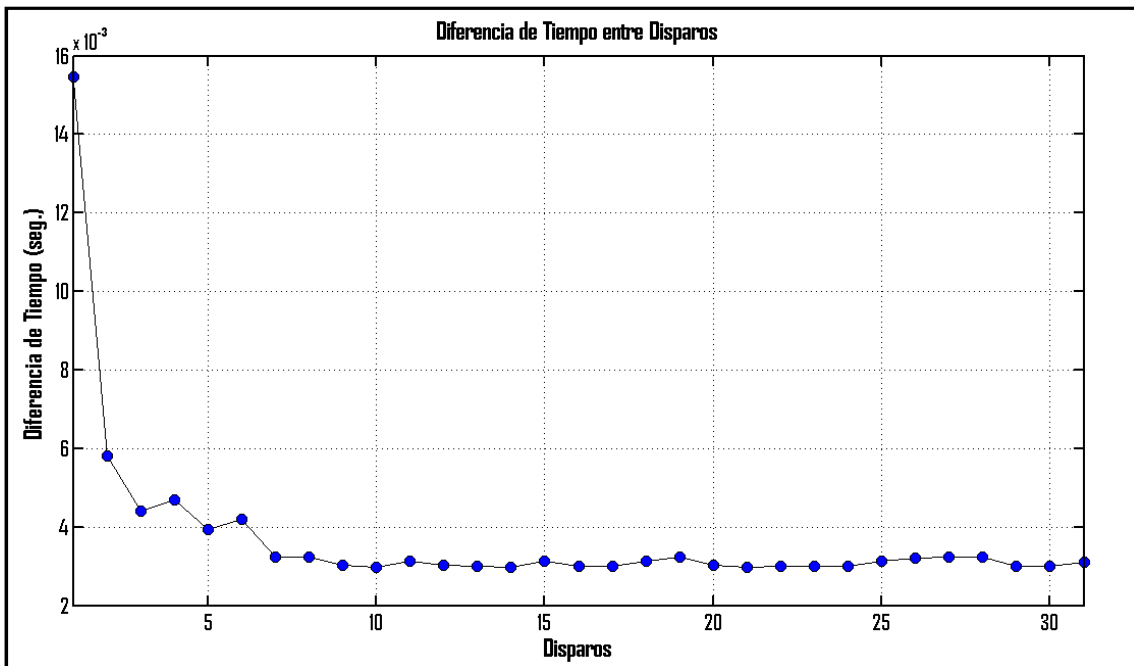


Figura 26. Diferencia de tiempo entre disparos sucesivos de la neurona de salida ($disp_i(t) - disp_{i-1}(t)$).

En la Fig. 27 observamos la representación de los pesos finales asociados a las sinapsis que han contribuido a que la neurona de salida aprenda a reconocer el dígito “1”. Vemos que las únicas componentes importantes de la Fig. 27 son las centrales. Esto quiere decir que los pesos de las componentes verticales e inclinadas son los únicos que se han reforzados positivamente. Queda claro que las orientaciones verticales e inclinadas son las más potentes del dígito entrenado.

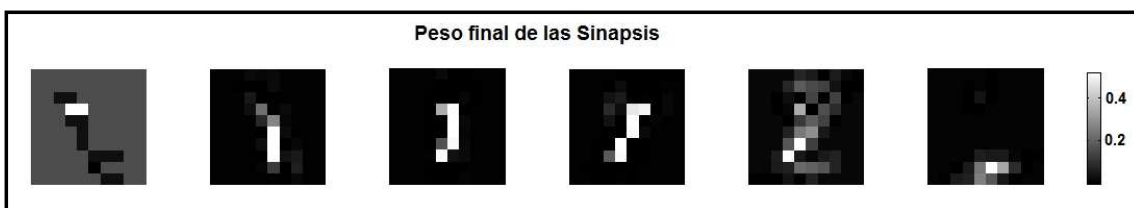


Figura 27. Peso final de las sinapsis cuando el sistema se entrena con 5000 entradas que representan el dígito “1”.

3.3.1. ESTUDIO DE LA RESPUESTA DE LA NEURONA ANTE DIFERENTES ENTRADAS

Una vez desarrollado nuestro sistema, necesitamos acotar el grado de fiabilidad del sistema. Para ello, vamos a realizar un pequeño estudio donde compararemos el grado de selectividad obtenido por la neurona de salida entrenada con un determinado patrón. Básicamente el experimento consiste en estudiar la respuesta del sistema ante un número memorizado y otro que no lo es.

Para la realización de esta práctica nos hemos surtido de los pesos finales de una neurona entrenada con el patrón que representa el número natural “7” (se ha bombardeado el sistema con 5000 entradas que representan el mismo dígito, asegurándonos así que los pesos asociados a las sinapsis son los resultantes cuando la neurona final ha memorizado el patrón). La respuesta del sistema es instantánea cuando se introduce el patrón entrenado, como se aprecia en la Fig. 28. Vemos que no es necesario que el mecanismo procese toda la señal de entrada, ya que le basta con el procesamiento de tan solo unos pocos pulsos enviados por los primeros píxeles activos para dar una respuesta.

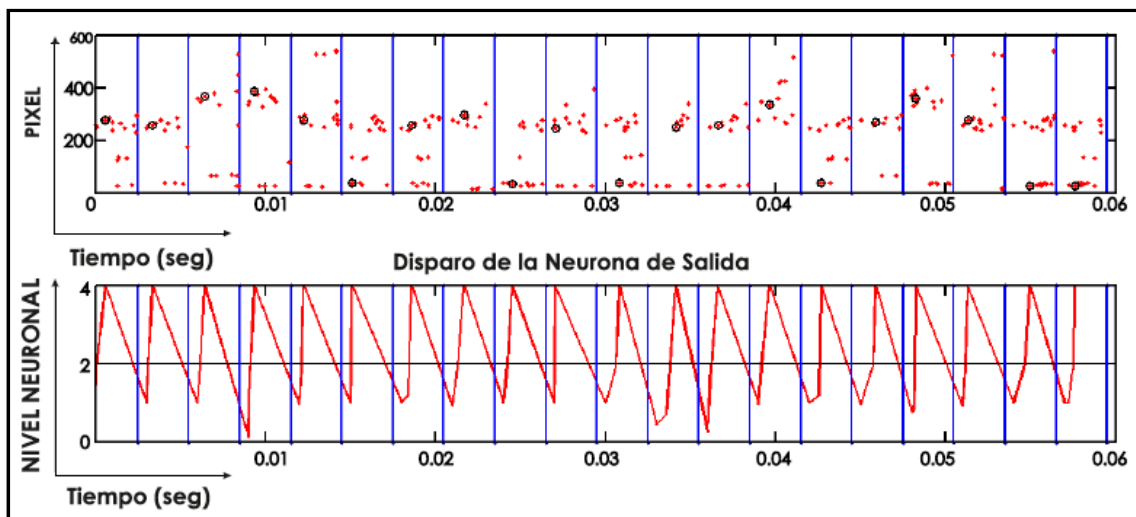


Figura 28. Respuesta ante el dígito aprendido “7”.

En la Fig. 28, vemos la representación gráfica de lo expuesto previamente. Hay muy pocos píxeles que detecten señal debido a que la actividad de los mismos no supera la sensibilidad mínima a la entrada, esto es debido a que el dígito “7” está compuesto principalmente por dos orientaciones (inclinada y horizontal). Otro elemento reseñable en la imagen es la alta fiabilidad de la estructura, en cada slot de tiempo se produce disparo del sistema. Podemos concluir que siempre que se introduzca el patrón entrenado se produce respuesta indicando que ha habido un reconocimiento del dígito.

En la Fig. 29 introducimos al sistema entradas que representan el número natural “6”. Ahora vemos que la respuesta cambia ya que la neurona de salida no reconoce el patrón memorizado en las entradas. Ahora hay muchas componentes activadas que antes no lo estaban, ya que para este nuevo dígito predominan componentes verticales, inclinadas y horizontales. Ahora la respuesta es mucho más lenta e incluso puede que no haya disparo en alguna ventana. La neurona dispara por la continua excitación a la entrada (por la acumulación de la contribución de los pesos de las conexiones), por lo que entendemos que las respuestas del sistema pueden ser falsas alarmas en su mayoría.

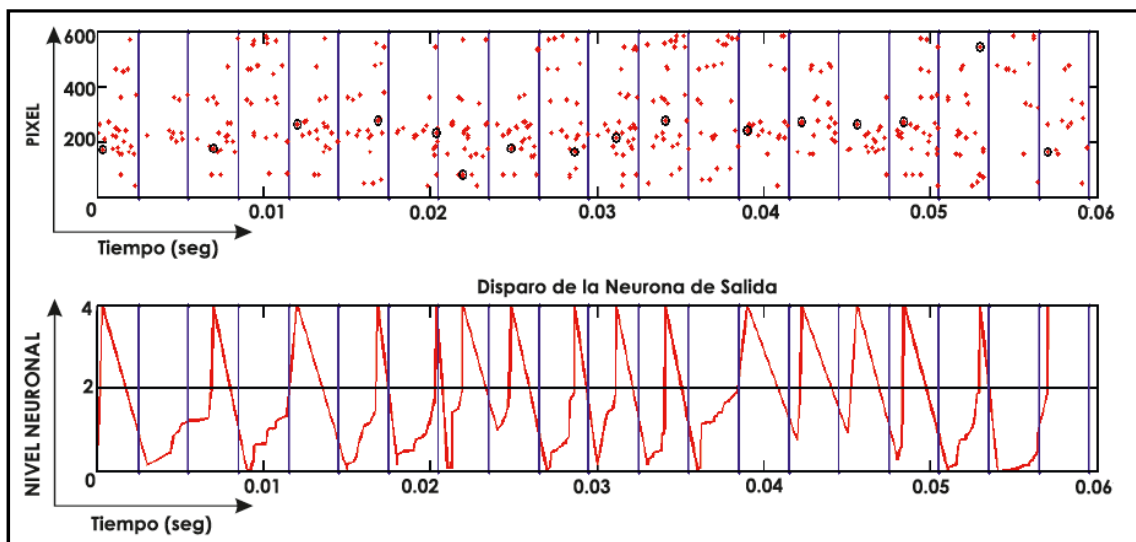


Figura 29. Respuesta ante el dígito no aprendido “6”.

Si la neurona de salida entrenada con el dígito “7” empieza a entrenarse con otra señal (como en este caso el dígito “6”), olvidará lo aprendido hasta ahora y comenzará de

nuevo una etapa de aprendizaje. Así queda demostrado que las neuronas y sus sinapsis pueden sufrir un proceso de reciclaje, donde conexiones que antes no intervenían ahora empiezan un proceso de actividad. Por tanto, podemos hablar de las sinapsis como uniones “semipermanentes”. Las conexiones estarán establecidas durante el periodo de actividad en el que llega el patrón entrenado. En cambio, durante el periodo de inactividad o de llegadas de entradas aleatorias (diferentes a la entrada memorizada) estas conexiones se irán diluyendo poco a poco conforme vaya pasando el tiempo.

3.3.2. EVALUACIÓN DEL SISTEMA

En el apartado anterior comprobamos de manera gráfica como la respuesta de la neurona es más rápida cuando se estimula con un patrón de entrada entrenado frente a otro diferente. Para demostrar empíricamente estos datos se desarrolla a lo largo de este apartado el estudio del *rate* o tasa de éxito. Consideramos éxito el que haya disparado la neurona deseada (aquella entrenada con el dígito que recibe a la entrada) antes que las demás. El cálculo viene expresado según la fórmula:

$$rate = \frac{succ}{total} \cdot 100$$

En esta ecuación, *succ* se corresponde con el número de disparos que se producen correctamente, es decir aquellas entradas aplicadas a neuronas de salida que no han sido entrenadas con dicha entrada provocan un tiempo de disparo mayor que en el del detector que si ha sido entrenado con dicho entrada (sería el éxito de nuestro sistema). Por otro lado la variable *total* se corresponde con el número de disparos totales que se producen en el testing.

Para conseguir un valor fiable del *rate* del sistema se ha entrenado cada una de las diez neuronas de salida con 5000 imágenes que codifican uno de los dígitos (estos dígitos han sido obtenidos de la base de datos MNIST [14]). Una vez que cada neurona ha memorizado su dígito correspondiente, se toma cada una de manera individual y se la expone a excitaciones que se corresponden con cada uno de los dígitos involucrados en este experimento. El siguiente paso es tomar los tiempos de disparo de la neurona de salida y aplicar la fórmula dada por el *rate*.

Sólo a modo de ejemplo, en la Tabla 1 se muestra el tiempo que tardan en disparar las diferentes neuronas, entrenadas cada una previamente con los diferentes dígitos intervinientes, cuando a la entrada se introducen diez imágenes que representa el dígito “0”. La tabla viene sombreada en verde cuando la neurona entrenada con el dígito “0” reconoce su patrón antes que el resto de neuronas entrenadas anteriormente con otros estímulos. En cambio va sombreado en rojo cuando uno de los detectores entrenado con un dígito que no es el “0” dispara antes que la neurona que si lo está.

Neurona	1ºdisp	2ºdisp	3ºdisp	4ºdisp	5ºdisp	6ºdisp	7ºdisp	8ºdisp	9ºdisp	10ºdisp
0	0,78	0,85	1,17	0,65	1,09	0,79	0,92	2,79	0,71	0,77
1	2,22	9,50	1,75	4,89	3,5	5,03	3,50	6,74	4,84	10,86
2	1,28	2,51	1,88	0,99	2,70	0,92	1,68	1,09	1,53	2,38
3	1,20	2,52	3,80	2,71	1,99	0,85	1,61	6,84	2,10	0,83
4	4,65	2,10	5,07	1,59	1,62	1,98	1,83	2,65	1,66	4,59
5	1,17	2,79	2,26	1,97	3,50	2,81	1,31	1,91	1,64	2,31
6	0,95	2,10	1,67	0,92	1,21	1,05	1,54	1,48	1,16	1,63
7	1,36	4,27	2,90	1,95	1,42	1,63	1,91	2,78	4,12	3,57
8	1,04	3,60	1,24	2,72	4,73	2,10	1,98	2,90	3,50	2,89
9	2,11	2,21	6,50	1,74	5,47	4,40	2,82	1,56	1,58	2,81

Tabla 1. Disparos de las neuronas de salida ante el mismo estímulo (ms).

Este caso es bastante ilustrativo y da un buen resultado, ya que en los 10 primeros disparos de cada detector de coincidencia en solo una ocasión se produce una detección errónea. Para este caso último caso, la salida reacciona más rápido ante un patrón no memorizado previamente. Haciendo los cálculos:

$$rate_{N_0} = \frac{succ}{total} \cdot 100 = \frac{9}{10} \cdot 100 = 90\%$$

Haciendo un estudio un poco más exhaustivo para 500 imágenes de entrada a cada neurona obtenemos para cada salida entrenada el caculo de *rate* mostrado en la Tabla 2:

Neurona	Rate
0	93.33
1	94.11
2	75.89
3	78.11
4	79.22
5	80.78
6	83.33
7	76.89
8	80.89
9	76.11

Tabla 2. Tasa de éxito para cada neurona después de 500 imágenes de entrada (%).

Podemos observar en la tabla anterior una aceptable tasa de éxito, ya que el nivel más bajo obtenido es de 75,89 % para la neurona encargada de detectar el dígito “2”. Es posible que este dígito falle más que otros porque comparte componentes parecidas a los dígitos “3” y “7”. Podemos observar en la Tabla 2 que las neuronas encargadas de memorizar estos dígitos también tienen un porcentaje de éxito menor al resto (78.11% y 76.89 respectivamente), debido a la similitud de las direcciones y orientaciones de los patrones entrenados. Por eso, a la hora de realizar un estudio del éxito del sistema las neuronas pueden equivocarse en la detección al confundir entradas parecidas, bajando así la tasa de éxito.

Por último hemos hecho un promediado de todos los *rate*, obteniendo un valor que consideramos aceptable:

$$rate_{total} = \frac{\sum_{i=1}^{n=10} Rate_i}{n} = 81.87 \%$$

Se hace evidente que una gran mejora del sistema será el entrenamiento conjunto de las diez neuronas de salida, donde la inclusión de inhibición entre las diferentes neuronas hará que las características no presentes en un dígito y que aparezcan a la entrada inhiban, haciendo al sistema por tanto más selectivo.

4. POSIBLES MEJORAS DEL PROYECTO

Podemos admitir que nuestro experimento es un poco especulativo, pero quizás con algunas mejoras podría ser plausible que un mecanismo basado en STDP en un sistema como el nuestro emulara de alguna manera mejor el sistema nervioso visual.

Una vez revisado exhaustivamente el sistema hemos encontrado varios puntos bastante mejorables en el mismo. Para ello nos hemos surtido de otros estudios similares realizados por diversos autores nombrados anteriormente en este proyecto [13] [23] [45]. Unos de los puntos más reseñables en el proyecto es la forma de aprendizaje de nuestro sistema, ya que se da en unas condiciones ideales donde no ocurre ninguna alteración externa como ruido gaussiano, jitter, intercalado de otras imágenes que llevan a nuestro detector de coincidencia a confusión e inclusive a error. Entendemos que gracias a la gran robustez del mecanismo STDP esta modificación sería viable, pero seguramente habría que realizar una serie de ligeras modificaciones para adaptar el conjunto a la nueva situación descrita.

Otra característica a mejorar en la simulación es que las neuronas son entrenadas por separado. Ésto claramente no ocurre en la realidad, ya que las neuronas compiten unas con otras. Para poder aplicar aquí un entrenamiento conjunto tendríamos que hablar del concepto *inhibición*, ya que sin él todas las neuronas podrían disparar y aprender el mismo patrón, siendo el experimento así un auténtico fracaso. Al introducir la inhibición, las neuronas que reconociesen una imagen y por tanto dieran una respuesta ante tal estímulo, lanzarían un pulso inhibitorio al resto de neuronas para indicarles que ella ha reconocido el dígito y que deberían disminuir una cierta cantidad su acumulador de pesos para que les costase más reconocer el mismo patrón. La disminución en el acumulador de pesos de la neurona de salida no debería ser tampoco excesiva ya que podríamos bloquear el sistema, ocasionando una paralización del resto de neuronas sin que éstas pudiesen memorizar otra imagen [13].

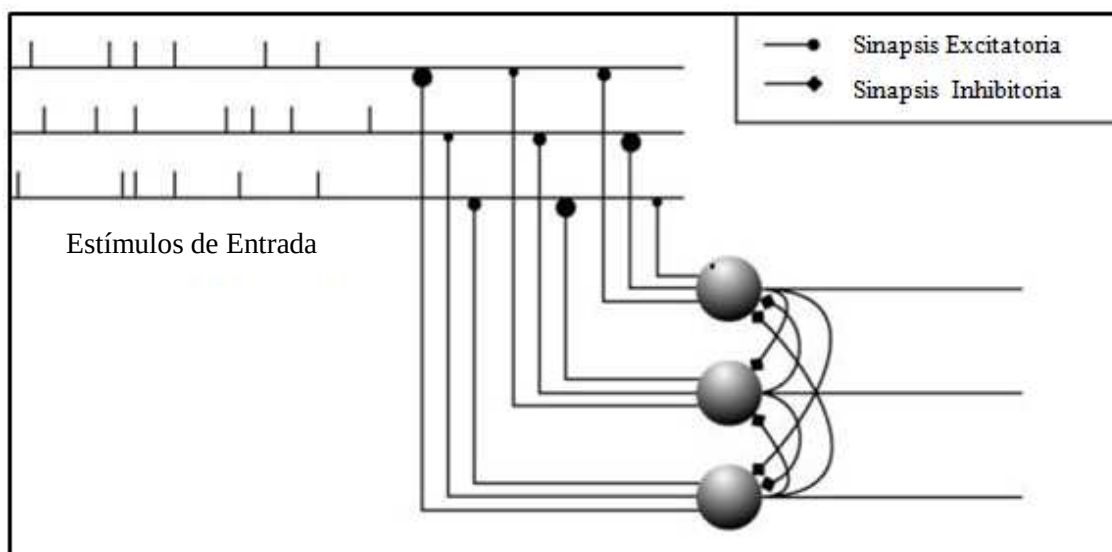


Figura 30. Respuestas inhibitorias de las neuronas.

En la Fig. 30 se ve como todas las neuronas reciben los mismos pulsos de entradas pero por diferentes factores (biológicos, inicialización de los pesos, ambientales...) aprenden a reconocer una determinada imagen. En el momento de respuesta ante la imagen conocida la neurona misma se encarga de lanzar un pulso inhibitorio al resto de detectores, alertándoles que ya ha habido un reconocimiento de una imagen evitando el disparo durante el slot de tiempo que tiene lugar el evento.

Al actuar todo el conjunto de neuronas detectoras existentes en nuestra simulación no podríamos olvidarnos de incluir un factor de aleatoriedad, ya que no es difícil que dos neuronas reconozcan a la misma vez una imagen sin que el pulso de *inhibición* le diese tiempo a surtir su efecto. En ese caso nos tendríamos que poner en manos del azar. Esta suposición no es muy descabellada porque sabemos que esto pasa continuamente en la naturaleza.

Con la introducción de la competición entre las neuronas y el mecanismo inhibitorio conseguiremos alcanzar valores más altos de realismo, porque entendemos que algo parecido debe ocurrir en el entorno biológico.

Con la utilización de un banco de filtro mayor que estuviera constituido por un número superior de orientaciones y escalas se podría mejorar también un poco más el sistema. Está claro que con esta mejora el coste computacional se incrementaría, costaría mucho más alcanzar el aprendizaje del patrón, pero seguramente el número de falsas alarmas se vería disminuido.

El nivel de detalle también podría mejorarse introduciendo el color en aplicaciones donde interviniese. En nuestro sistema, al tratar reconocimiento de números en escala de grises no tiene sentido hablar de color.

5. CONCLUSIONES

En el trabajo presentado se ha demostrado la posibilidad de incluir mecanismos de aprendizaje no supervisados bioinspirados en una aplicación sencilla de reconocimiento de dígitos. Nótese que en este trabajo el objetivo último no era alcanzar una tasa de reconocimiento elevada, sino demostrar cómo un sistema sencillo puede ser entrenado utilizando el algoritmo bioinspirado STDP y proporcionar resultados aceptables. Es posible alcanzar mejores resultados haciendo el sistema un poco más complejo utilizando tal vez mayor resolución en las imágenes, un banco de filtros mayor y añadiendo inhibición entre las neuronas de salida durante la etapa de entrenamiento, de modo que cuando una neurona de salida dispare un pulso, este pulso se utilice para inhibir e impedir que el resto de neuronas “aprendan” el patrón que ha hecho disparar a la primera de ellas [13].

Un sistema como el descrito puede ser implementado totalmente con módulos convolucionales bioinspirados, tales como módulos de convolución basados en el protocolo AER (Address-Event-Representation [35] [36]). Hay numerosos sistemas basados en este protocolo y en el mecanismo bioinspirado STDP. Nuestro sistema, a diferencia de otros, tiene como novedad la capacidad de reconocer un número elevado de patrones diferentes.

Sistemas básicos como el de este proyecto se están implementando en mecanismos hardware como el Spinnaker (*Spiking Neural Network Architecture*) [46] [47], que está siendo desarrollado en la Universidad de Manchester. Consiste en una arquitectura multi-núcleo que se inspira en el funcionamiento del cerebro humano. Este mecanismo está compuesto por millones de procesadores que trabajan en paralelo enviando y procesando pulsos eléctricos como lo hacen las neuronas del sistema nervioso. Por tanto, este hardware proporciona una herramienta muy sólida en el desarrollo de mecanismos artificiales que implementen algunas de las tareas del cerebro humano.

6. APÉNDICES

6.1. CÓDIGO MATLAB

En este apartado vamos a incluir las subrutinas de código en Matlab que son más significativas o que han conllevado un esfuerzo mayor en su implementación, realizando un papel primordial en el desarrollo del proyecto.

6.1.1. FUNCIÓN APRENDIZAJE

Esta subrutina implementa la actividad de aprendizaje en la neurona detectora de patrones de coincidencia. La función va acumulando en cada neurona los pesos de las conexiones y compara esta acumulación de pesos con el umbral de disparo de la neurona. Tras el disparo de la neurona, se actualizan los pesos sirviéndose de STDP y el acumulador neuronal se pone de nuevo a cero para reiniciar la cuenta.

La función recibe como entrada la variable bidimensional *matriz* que contiene todos los resultados obtenidos con los filtros de Gabor, codificados en el tiempo y simplificados a aquellas componentes con mayor valor. La otra variable *n_digitos* introduce el número de dígitos de entrada que estimulan a la neurona.

Como salida saca por pantalla varias imágenes:

- La primera imagen, representa los pesos acumulados finales para cada sinapsis (este valor muestra el aprendizaje de la neurona) según las 6 orientaciones de Gabor existentes.

- La segunda, tercera y cuarta imagen representan los píxeles activos comparándolos en el tiempo con la evolución del potencial acumulado por la neurona de salida. Unos ejemplos de estas ilustraciones aparecen en las Fig. 22 y 23. A su vez se representan los píxeles activos a la salida de los filtros de Gabor que codifican en el dominio del tiempo y que influirán directamente en el acumulador de la neurona de salida.

-La quinta ilustración, muestra todos los píxeles activos que codifican pulsos enviados a partir de las sinapsis que provocan el disparo de nuestra célula nerviosa entrenada.

-La sexta imagen, representa como va disminuyendo el tiempo entre disparos de la neurona detectora de coincidencia conforme el entrenamiento va siendo asimilado.

Podemos destacar en la subrutina la ayuda de dos funciones principales: *acumula_peso* y *actualizacion_peso*. La primera va acumulando el peso de las sinapsis que transportan pulsos a la neurona de salida (debido al proceso de estimulación neuronal) y modifica la variable *disparo* si la neurona detecta el patrón memorizado; la segunda función va actualizando los pesos de todas las sinapsis cuando la neurona detectora dispara. Sus códigos se explicarán más detenidamente en los siguientes apartados.

```
function out=aprendizaje(matriz, n_digitos);

[filas, columnas]=size(matriz);
timeslot =3e-3; %Tiempo de slot es 3 ms.
peso_inicial=0.01;
aumento_peso=peso_inicial*ones(filas,columnas/2); %Matriz que acumula el aumento de los peso de cada sinapsis.
peso_total=0; %Variable que acumula el peso, se pone a cero cuando dispara la neurona de salida.
disparo=0; %Variable binaria que indica si ha reconocido el patrón entrenado.
umbral=2; %Umbral que debe superar "peso_total" para que se desborde el acumulador de la neurona de salida.
var_representacion=[];
var_representacion(1,1)=0;
var_representacion(1,2)=0;
k=2;
afereente_detector=[]; %Variable que me determina que pixel es el desencadenante del disparo de la neurona.
m=1;
tiempo_disparo=[];
tiempo=0;
indice=1;
for j=2:2:columnas
    for i=1:1:filas
        if(matriz(i,j)<inf & disparo<1)

            [disparo,peso_total,var_representacion]=acumula_peso(aumento_peso...
            ...(matriz(i,j-1),j/2),peso_total,umbral,var_representacion,k);

            var_representacion(k,2)=matriz(i,j); %Registra el instante en el que se incrementa el peso.
            k=k+1;
            if(disparo>0) %Se ha producido el reconocimiento de la imagen.
                aumento_peso=actualizacion_peso(matriz,aumento_peso,i,j,(columnas/2),n_digitos,umbral);
```

```

    aferente_detector(m,1)=matriz(i,j-1);
    aferente_detector(m,2)=matriz(i,j);
    m=m+1;
end;
elseif (disparo>0 & i==filas)           Se ha producido disparo debido a una imagen intermedia.
disparo=0;
tiempo_disparo(indice)=aferente_detector(m-1,2)-tiempo;
if(j>2)
    tiempo=((j/2)-1)*3e-3+2.5e-3;      %Actualiza el inicio de la ventana para el siguiente disparo.
else
    tiempo=2.5e-3;                      %Solo para el caso de la primera ventana.
end;
indice=indice+1;
end;
end;

out=aumento_peso(:,columnas/2);
var_aux1=size(aumento_peso);
peso1=aumento_peso(:,var_aux1(2));
representa_imagen_peso(peso1,2);

%REPRESENTACION DE LAS VENTANAS
var_aux=zeros(2,2);                    %Matriz que sirve para la representación de las ventanas.
var_aux(:,1)=umbral;
if n_digitos>75
    n_digitos_ventana=20;              %Número de dígitos para las ventanas de representación.
    ventana=timeslot*n_digitos_ventana;

%PRIMERA VENTANA%
var_aux1=find(var_representacion(:,2)<ventana);          %Acotación de la primera ventana de representación.
var_aux2=find(aferente_detector(:,2)<ventana);
var_aux(2,2)=ventana;                                   %La primera ventana tiene 20 slots de tiempo.
primera_ventana(var_representacion(var_aux1,:),var_aux,n_digitos_ventana,...
...matriz,aferente_detector(var_aux2,:));

%SEGUNDA VENTANA ALTERNATIVA%
mitad_digitos=5*n_digitos_ventana;
mitad_tiempo=5*ventana;
var_aux3=find((var_representacion(:,2)>mitad_tiempo)&(var_representacion(:,2)<(mitad_tiempo+ventana)));
var_aux4=find((aferente_detector(:,2)>mitad_tiempo)&(aferente_detector(:,2)<(mitad_tiempo+ventana)));
var_aux(1,2)=mitad_tiempo;
var_aux(2,2)=mitad_tiempo+ventana;                    %La segunda ventana también tiene 20 slots de tiempo.
segunda_ventana(var_representacion(var_aux3,:),var_aux,mitad_tiempo,matriz(:,(2*mitad_digitos+1):...
...(2*mitad_digitos+40)),aferente_detector(var_aux4,:),n_digitos_ventana);

%TERCERA VENTANA%
inicio_ventana=timeslot*columnas/2-ventana;           %Inicio de la última ventana de representación.

```

```
var_aux5=find(var_representacion(:,2)>inicio_ventana);
var_aux6=find(aferente_detector(:,2)>inicio_ventana);
var_aux(1,2)=inicio_ventana;
var_aux(2,2)=timeslot*columnas/2;
tercera_ventana(var_representacion(var_aux5,:),var_aux,...
... n_digitos_ventana,matriz(:,(2*n_digitos-39):(2*n_digitos)),...
...aferente_detector(var_aux6,:));
end;

%REPRESENTACION DEL PÍXEL QUE HACE QUE LA NEURONA DISPARE%
representa_aferente_detector(aferente_detector)

%DIFERENCIAS DE TIEMPO ENTRE DISPAROS DE LA NEURONA DE SALIDA%
matriz_diferencias=[];
[fil,col]=size(aferente_detector);
for i=1:1:fil
    if(i<fil)
        matriz_diferencias(i)=aferente_detector(i+1,2)-aferente_detector(i,2);
    end
end
variable=[1:fil-1];
diferencias_entre_picos(matriz_diferencias)
```

6.1.2. FUNCIÓN ACUMULA PESO

Esta subrutina es llamada en la función *aprendizaje* que realiza la operación de acumular el peso de las sinapsis que van transmitiendo los pulsos codificados en la primera etapa del sistema. Si el peso acumulado en la variable *peso_total* supera el umbral marcado en el experimento, la función modifica la variable binaria *disparo* pasando a tomar el valor “1”, indicando al proceso que se ha producido desbordamiento del acumulador de la neurona detectora.

Las variables de entradas son:

-*peso_total*: Es la variable que ejerce la función de acumulador de peso a la entrada de la neurona de salida.

-*aumento_peso*: Variable que indica el peso de la sinapsis que transporta un pulso a la neurona de salida. Ayuda a actualizar la variable *peso_total*.

-*umbral*: Constante que marca el punto en el que la neurona de salida dispara (y por tanto reconoce un patrón entrenado). Si la variable *peso_total* supera el valor fijo marcado, nuestro detector reaccionará.

Las variables de salida son *disparo*, *peso_total* y *var_representacion* su misión consiste en ayudar en la actualización del sistema.

```
function [disparo,peso_total,var_representacion]=acumula_peso(aumento_peso,peso_total,umbral,...
...var_representacion,k)

peso_total=peso_total+aumento_peso;           %Actualización del peso total acumulado por la neurona de salida.
if (peso_total<umbral)
    disparo=0;                                %No se dispara, no se reconoce la imagen.
    var_representacion(k,1)=peso_total;
else
    disparo=1;                                %Se produce disparo, se reconoce la imagen.
    peso_total=0;                             %Como se ha disparado la neurona se reinicializa el acumulador de pesos.
    var_representacion(k,1)=2*umbral;
end;
```

6.1.3. FUNCIÓN ACTUALIZACIÓN PESO

Esta subrutina también es llamada en la función principal *aprendizaje*. Su cometido principal consiste en actualizar todos los pesos asociados a las sinapsis cuando la neurona de salida realiza un disparo. Esta actualización se rige según lo postulado en el mecanismo de aprendizaje bioinspirado STDP.

Las variables de entrada son:

-*matriz*: Esta variable contiene el valor de los pesos asociados a las sinapsis en el slot de tiempo en el que se produce el disparo de la neurona de salida.

-*aumento_peso*: Variable que proporciona los pesos de todas las sinapsis antes de actualizarse.

-*umbral*: Constante acordada como valor máximo del acumulador de la neurona de salida.

La salida de la función *actualizacion_peso* es la variable *aumento_peso* pero actualizada una vez aplicada las reglas descritas por el método STDP.

```
function aumento_peso=actualizacion_peso(matriz,aumento_peso,i,j,columnas,n_digitos,umbral)

p=j/2+1; %Variable auxiliar para índice.
if(p>n_digitos)
    p=n_digitos;
end;
aumento_peso(matriz(i,j-1),p)=aumento_peso(matriz(i,j-1),j/2)+incW(0);
if (aumento_peso(matriz(i,j-1),p)>(umbral/2)) %Ninguna actualización de peso supera el valor umbral.
    aumento_peso(matriz(i,j-1),p)=umbral/2;
end;
for k=1:1:i-1
    %Actualización positiva de los pesos.
    aumento_peso(matriz(k,j-1),p)=aumento_peso(matriz(k,j-1),j/2)+incW(matriz(k,j)-matriz(i,j));
    if (aumento_peso(matriz(k,j-1),p)>(umbral/2))
        aumento_peso(matriz(k,j-1),p)=umbral/2;
    %Para que la actualización de los pesos nunca sobrepase el umbral.
end;
end;
for l=i+1:1:600
    if (matriz(l,j)<inf)
        %Actualización negativa de los pesos.
```



```
    aumento_peso(matriz(l,j-1),p)=aumento_peso(matriz(l,j-1),j/2)+incW(matriz(l,j)-matriz(i,j));
end;
if (aumento_peso(matriz(l,j-1),p)<0)
    aumento_peso(matriz(l,j-1),p)=0;
    %Para que la actualización de los pesos no sea inferior a cero.
end;
end;
for j=p+1:1:columnas
    aumento_peso(:,j)=aumento_peso(:,p);    % Se actualiza la variable de salida en el siguiente slot de tiempo.
end;
```

6.1.4. FUNCIÓN CALCULO RATE

Esta función es una parte de la subrutina que calcula la tasa de éxito del experimento, considerando como éxito el que haya disparado la neurona deseada (aquella entrenada con el dígito que recibe a la entrada) antes que las demás:

$$Rate = \frac{succ}{total} \cdot 100$$

En esta ecuación, *succ* como se ha explicado en apartados anteriores se corresponde con el número de disparos que se producen correctamente. Es decir, aquellas entradas aplicadas a neuronas de salida que no han sido entrenadas con dicha entrada provocan un tiempo de disparo mayor que en el del detector que si ha sido entrenado con dicho entrada (sería el éxito de nuestro sistema). Por otro lado la variable *total* se corresponde con el número de disparos totales que se producen en el test. Esta subrutina nos va a devolver la variable *tiempo_disparo*, que muestra todos los tiempos en los que se produce los disparos de la neurona de salida a estudio.

Las variables de entrada a nuestra subrutina son:

-*matriz*: Son los pulsos en el dominio del tiempo enviados a cada neurona de salida a través de las sinapsis.

-*patrón*: Pesos finales de cada sinapsis (se corresponden cuando la neurona esta entrenada).

-*numero_disparos*: Números de disparos totales en el test.

Al igual que la función *aprendizaje* se ayuda de la función *acumula_peso* que nos ayuda actualizar el valor del acumulador de la neurona de salida.

```
function tiempo_disparo=calculo_rate(matriz,patron,numero_disparos)
```

```
[filas,columnas]=size(matriz);
```

```
peso_total=0; %Variable que actualiza el peso en un adicional producidos en un slot de tiempo.
```

```
disparo=0; %Indica si se ha reconocido la imagen o no.
```

```
umbral=2;
```

```
var_representacion=[];
```

```
var_representacion(1,1)=0;
```

```

var_representacion(1,2)=0;
k=2;
aferente_detector=[];
m=1;
tiempo_disparo=[];
tiempo=0;
indice=1;
for j=2:2:columnas
    for i=1:1:filas
        if(matriz(i,j)<inf && disparo<1)

            [disparo,peso_total,var_representacion]=acumula_peso(patron(matriz(i,j-1)),peso_total,imbral,...
            ...var_representacion,k);

            var_representacion(k,2)=matriz(i,j);
            k=k+1;

            if(disparo>0)                                %Se ha producido el reconocimiento de la imagen.
                aferente_detector(m,1)=matriz(i,j-1);    %Función que marca el momento del disparo.
                aferente_detector(m,2)=matriz(i,j);
                m=m+1;
            end;
        elseif (disparo>0 && i==filas)                    %Se ha producido el disparo debido a una imagen intermedia.
            disparo=0;
            tiempo_disparo(indice)=aferente_detector(m-1,2)-tiempo;
            if(j>2)
                tiempo=((j/2)-1)*3e-3+2.5e-3;
            else
                tiempo=2.5e-3;
            end;
            indice=indice+1;
        end;
    if indice> numero_disparos
        break;
    %Se sale del "for" cuando supera el número de disparos que se pasan por línea de comandos.
    end
end;

if indice> numero_disparos
    break;
end
end;

```

6.2. ARTICULO URSI

Reconocimiento de Dígitos Usando Mecanismos Bioinspirados de Aprendizaje

J. A. Pérez-Carrasco, Eneko García-Arana, B. Acha, C. Serrano

jperez2@us.es, eneko.garcia.arana@gmail.com, bacha@us.es, cserrano@us.es

Dpto. Teoría de la Señal, ETSIT, Universidad de Sevilla. Avda de los Descubrimientos, s/n, CP41092

Abstract- This document presents a slightly bioinspired system for recognition of numeric digits. The system is composed by a Gabor filter bank with one scale and six orientations and one one-layer neural network trained using the biological training mechanism called STDP (*Spike-timing-dependent plasticity*). STDP is an unsupervised mechanism, where the network is able to learn autonomously when the inputs are repeated persistently. In the work presented here ten neurons have been trained to recognize ten different digits (0-9). To test and train our network, the MNIST database has been used. We have used 50000 images for training and 2000 images for testing.

I. INTRODUCCIÓN

Cada vez es más habitual encontrar sistemas que tratan de implementar el procesamiento del cerebro para resolver problemas complejos. La evidencia constante de que el cerebro es capaz de realizar tareas casi impensables para la más compleja máquina desarrollada por el hombre a día de hoy está haciendo que la comunidad científica trate de emular el procesamiento biológico implementado en el cerebro. Recientemente los sistemas que están teniendo más éxito son aquéllos basados en redes neuronales, y de ellos, los basados en convoluciones que utilizan algoritmos de entrenamiento como el *backpropagation* están resultando más eficientes en el desarrollo de tareas como reconocimiento y seguimiento de objetos, clasificación, segmentación, etc [1][2]. A pesar de todos estos intentos, aún a día de hoy es difícil encontrar sistemas que además de emular la estructura de etapas del cerebro utilicen también los mecanismos de aprendizaje y comunicación mediante pulsos del mismo. El mecanismo de aprendizaje biológico STDP (*Spike-timing-dependent plasticity*) [3] es la hipótesis más comúnmente aceptada de mecanismo biológico de aprendizaje implementada en el cerebro humano. STDP es un proceso que ajusta la fuerza (pesos) de las conexiones entre neuronas en el cerebro basándose en los tiempos relativos entre los pulsos de salida de una neurona y los pulsos recibidos por la misma y que alteran su potencial.

Tratando de implementar un pequeño sistema bioinspirado hemos desarrollado en el presente trabajo un sistema compuesto de dos etapas. La primera de ellas es un banco de filtros de Gabor, que está bastante aceptado como una

de las primeras etapas del procesamiento en el cerebro. La segunda etapa es una red neuronal de una etapa, compuesta por diez neuronas de salida, cuyas conexiones han sido entrenadas usando STDP. El sistema recibe como entrada un dígito numérico de 0 a 9, obtenido de la base de datos MNIST [4], y sólo la neurona de salida correspondiente a ese dígito de entrada será la que presente una mayor y rápida actividad a la salida.

II. IMPLEMENTACIÓN

A. Descripción del sistema

El sistema implementado en este trabajo está compuesto de dos etapas y se puede ver en la Fig. 1. La primera de ellas es un banco de filtros de Gabor y la segunda una red neuronal monoetapa compuesta por diez neuronas de salida. El tamaño de las imágenes de entrada es de 32x32, aunque para nuestra aplicación, estas imágenes han sido submuestreadas para tener un tamaño de 16x16. Dichas imágenes son convolucionadas con un conjunto de 6 filtros de Gabor, que implementan una escala y seis orientaciones. El tamaño de los filtros de Gabor es 10x10.

Cada uno de estos filtros se obtiene mediante las siguientes expresiones [5]:

$$g(x, y) = \frac{1}{2\pi \cdot \sigma_x \sigma_y} \exp \left[-\frac{1}{2} \left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) + j2\pi \cdot Wx \right], \quad (1)$$

Realizando unas determinadas dilataciones y rotaciones a la función de transferencia obtenida $g(x, y)$ obtenemos cada filtro codificando una orientación y escala:

$$\begin{aligned} g_{s,k}(x, y) &= a^{-s} g(x', y') \\ x' &= a^{-s} (x \cdot \cos\theta + y \cdot \sin\theta) \\ y' &= a^{-s} (-x \cdot \sin\theta + y \cdot \cos\theta) \end{aligned} \quad (2)$$

En estas expresiones θ representa la orientación y S la escala. Los parámetros $\{a, \theta, W, \sigma_x, \sigma_y\}$ son descritos en el método implementado por Manjunath [5].

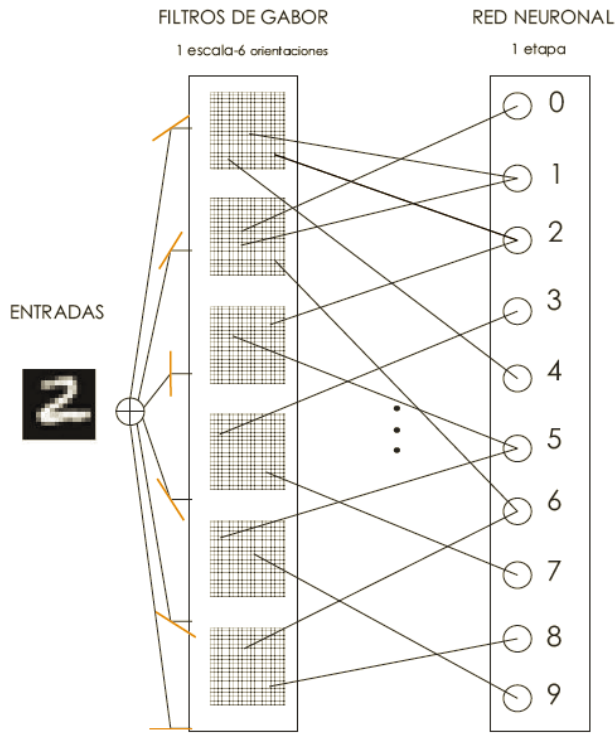


Fig. 1. Sistema de reconocimiento de dígitos con mecanismos bioinspirados de aprendizaje.

De cada imagen de salida tras la convolución hemos seleccionado la parte central de tamaño 10x10, una vez comprobado que en ese conjunto de píxeles estaba contenida la información con valor distinto de cero.

El mecanismo STDP es un mecanismo biológico de aprendizaje que se basa en los tiempos relativos entre los pulsos producidos por una neurona y los pulsos que recibe a su entrada y que pueden ser los que le han hecho activarse y disparar. Es por ello que en nuestra implementación necesitamos codificar el valor de cada píxel a la salida de la convolución con uno de los filtros de Gabor en pulsos. Para nuestra implementación hemos realizado una codificación intensidad-tiempo, de modo que hemos creado un pulso temporal para cada píxel con un tiempo de creación “inversamente proporcional” al valor del píxel. Así, un píxel con valor cercano a 0 producirá un pulso “tardío” y un píxel con valor cercano a 255 producirá un disparo casi instantáneo. El tiempo de creación de un pulso vendrá relacionado con su valor de píxel con la expresión:

$$tpo_pulso = cte_temp \times (1 - pixel_value / 255) \quad (3)$$

En nuestra implementación cte_temp tiene un valor de 3ms. Un ejemplo ilustrativo de esta relación inversamente proporcional puede verse en la Fig. 2.

En el sistema, cada píxel de salida de cada filtro de Gabor está conectado a una de las de las 10 neuronas de salida, con lo cual tenemos 6x100x10 (6 componentes de Gabor, 100 píxeles de salida de cada filtro y 10 neuronas de salida) que hace un total de 6000 conexiones en nuestra etapa neuronal.

B. Mecanismo aprendizaje STDP

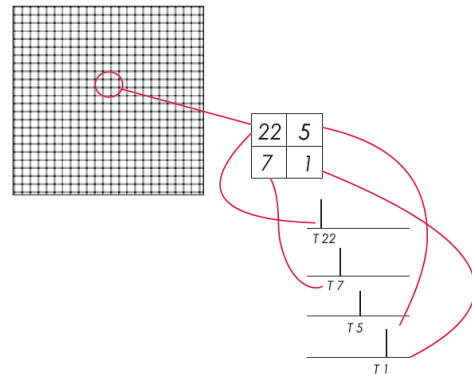


Fig. 2. Conversión de nivel de píxel en tiempo.

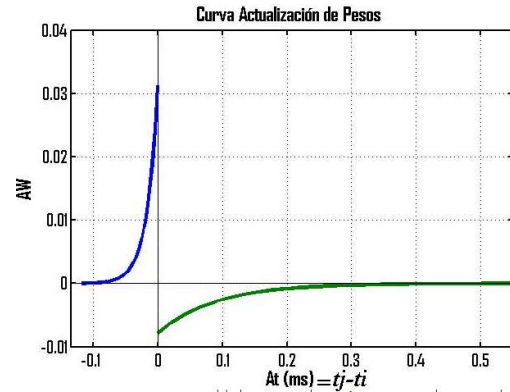


Fig. 3. Curva Aprendizaje STDP.

En nuestra implementación, las conexiones correspondientes a cada neurona de salida han sido entrenadas por separado. En una implementación futura y más eficiente se permitirá que todas las neuronas aprendan a la vez y el hecho de permitir inhibición entre ellas acabará haciendo que cada neurona aprenda un dígito diferente [3]. Por tanto, para la implementación aquí presentada, cada neurona de salida i , correspondiente al dígito de entrada i ($i=0,...,9$) ha sido entrenada por separado utilizando un conjunto de 5000 imágenes correspondientes al mismo dígito. Cuando la neurona i (sus conexiones) ha sido entrenada se procede a entrenar el resto de neuronas del mismo modo.

Pero nuestra pregunta sigue sin ser respondida, ¿cómo se implementa el aprendizaje bioinspirado basado en STDP? El mecanismo de aprendizaje STDP (*Spike Timing Dependent Plasticity*) [3] se basa en el principio de que las conexiones con las neuronas de salida (“pesos”) se van reforzando o debilitando con la repetición de unos pulsos de entrada determinados, de modo que aquellas conexiones cuyos pulsos provoquen un disparo en la neurona de salida se verán reforzadas. En cambio, aquellas conexiones por las cuales viajen pulsos que lleguen a la neurona de salida posteriormente al disparo de la misma se verán debilitadas. STDP es un mecanismo no supervisado, es decir, las neuronas aprenden de manera autónoma ante la repetición de los patrones de entrada. Gracias a este mecanismo, cuando la neurona haya aprendido un patrón de entrada, le bastarán muy pocos pulsos en sólo algunas conexiones de entrada para producir un disparo. El mecanismo STDP se resume del siguiente modo: Inicialmente el peso de todas las conexiones es el mismo (o puede estar inicializado

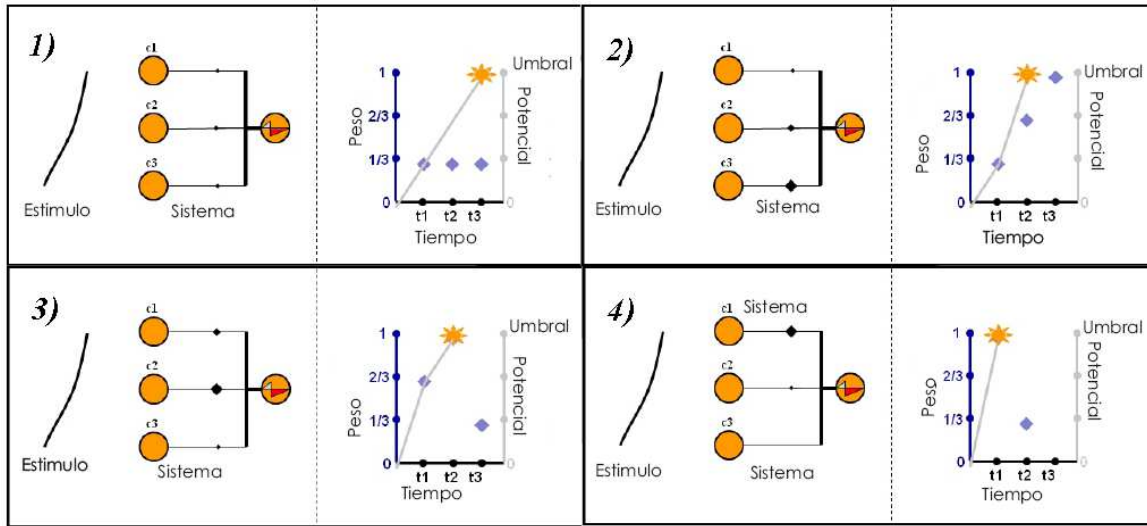


Fig. 4. Mecanismo Bioinspirado de Aprendizaje STDP.

aleatoriamente), pero cada vez que la neurona de salida produce un pulso de salida los pesos de las conexiones se actualizan de acuerdo a las siguientes expresiones [3]:

$$w_o = \Delta w_{inst} + w_{ant}$$

$$\Delta w_{inst} = \begin{cases} a^+ \cdot \exp\left(-\frac{t_j - t_i}{\tau^+}\right) & t_j \leq t_i \\ -a^- \cdot \exp\left(-\frac{t_j - t_i}{\tau^-}\right) & t_j > t_i \end{cases} \quad (4)$$

En estas ecuaciones w_{ant} es el peso anterior en una de las conexiones, Δw_{inst} es el incremento de peso debido al disparo de la neurona final. Los parámetros a^+ y a^- y τ^+ y τ^- son constantes predefinidas [3]. Por último, el valor $t_j - t_i$ representa la diferencia de tiempo entre el pulso producido por la neurona aferente en el instante t_j cuya conexión con la neurona de salida se está actualizando y el disparo producido por la neurona final en el instante t_i . La Fig. 3 representa la curva de actualización de pesos. En ella se puede ver que aquellas conexiones cuyos pulsos sean anteriores al disparo de la neurona de salida (y por tanto causantes del mismo) se verán reforzadas, sin embargo, los pesos de aquellas conexiones con pulsos posteriores al pulso de la neurona de salida se verán atenuados.

En la Fig. 4 se ilustra el mecanismo STDP de un modo simplificado. En esta figura, una neurona de salida está conectada a 3 neuronas que representamos como $c1$, $c2$ y $c3$ que producen un determinado patrón codificado en sus pulsos. Como se puede ver el orden de disparo de las neuronas es primero $c1$, después $c2$ y finalmente $c3$ (como viene representado por la curva de estímulo). El peso de las neuronas se representa en la figura como un punto negro en las conexiones (cuánto más grande mayor es su valor). Al principio (*paso 1*), todas las conexiones tienen igual valor ($pesos=1/3$), de modo que cada vez que llega un pulso a la neurona de salida, su potencial se ve incrementado por el peso correspondiente a cada

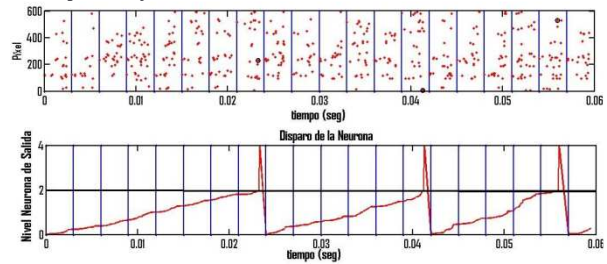


Fig. 5. Etapa de Entrenamiento de la Neurona '0' al inicio.

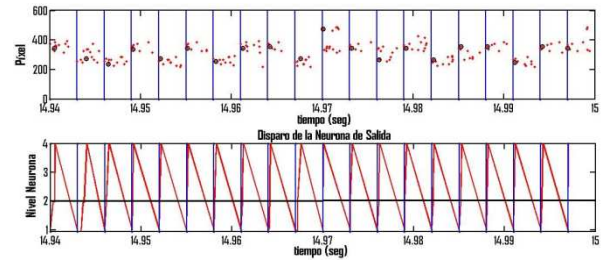


Fig. 6. Etapa de Entrenamiento de la Neurona '0' al final.

conexión. Se puede ver que en la primera iteración la neurona alcanza el umbral de disparo (de valor '1') cuando le llega el pulso procedente de la neurona $c3$ (instante $t3$). En el primer caso, como el disparo de la neurona de salida se produce posteriormente a los pulsos de entrada, las 3 conexiones se ven reforzadas a valores $1/3$, $2/3$ y 1 respectivamente. En la segunda repetición del patrón de entrada (*paso 2*), será la neurona $c2$ (por tener un *peso*= $2/3$) la que produzca la activación de la neurona de salida en el instante $t2$, con lo cual las conexiones con $c1$ y $c2$ se verán reforzadas y la conexión con $c3$ atenuada. En una tercera iteración (*paso 3*), será la neurona $c2$ de nuevo la causante de producir un disparo a la salida en el instante $t2$, con lo cual la conexión con $c1$ será reforzada de nuevo y la conexión con $c3$ se verá atenuada. En la última y cuarta iteración del ejemplo (*paso 4*), la conexión con la neurona $c1$ ya tiene un peso de valor '1' y por tanto es suficiente para activar la neurona de salida en el instante $t1$. Las otras dos conexiones (con $c2$ y $c3$) se ven atenuadas. En un estado estacionario estas dos conexiones estarán inhibidas y la conexión con la neurona $c1$ bastará para producir la

Neurona	1° Disparo	2° Disparo	3° Disparo	4° Disparo	5° Disparo
0	5,37	4,58	4,64	5,17	5,01
1	0,51	0,63	0,61	0,60	0,75
2	1,92	4,75	3,50	2,47	2,93
3	1,85	6,59	0,69	2,53	2,57
4	4,67	2,30	3,78	2,77	3,50
5	1,77	1,43	2,38	3,57	4,04
6	1,15	5,63	1,81	4,88	3,77
7	3,11	4,03	4,97	2,07	7,35
8	0,27	1,55	0,60	0,75	2,13
9	1,27	1,15	2,80	2,84	1,92

Tabla 1. Disparos de las neuronas de salida ante el mismo estímulo (ms).

activación con la neurona de salida. Nótese que esto es lo que se esperaba pues la neurona *c1* es la más activa y la más rápida, y el algoritmo de aprendizaje STDP ha permitido que la neurona de salida aprenda esto. Quede constancia de que el ejemplo aquí descrito es muy sencillo y que sólo trata de ilustrar la idea que se esconde detrás del mecanismo biológico STDP.

III. RESULTADOS EXPERIMENTALES

Podemos definir dos etapas en nuestro experimento. La primera etapa es de entrenamiento de cada neurona de salida por separado. En la Fig. 5 hemos representado esta fase. El eje *y* de la gráfica superior codifica el píxel que dispara un pulso a la salida de cada convolución con el banco de filtros de gabor. Como recordamos había 600 píxeles posibles. El pulso producido por cada uno de estos píxeles (sólo se han representado aquellos con un valor superior a un nivel predefinido) se ha representado como un punto rojo. El eje *x* codifica el tiempo en segundos. Cada imagen utilizada de entrada produce una serie de pulsos en una ventana de duración 3ms (recordemos que es el tiempo utilizado en la codificación intensidad-tiempo). La gráfica inferior de la Fig. 5 representa el potencial acumulado por la neurona de salida que se está entrenando. En esta figura se está representando el entrenamiento correspondiente a la neurona '0'. Se puede observar en esta etapa que la neurona necesita cargar durante un tiempo largo (del orden de 22ms, ~8 imágenes) la información correspondiente a los patrones para producir un pulso de salida y descargarse. En el momento del disparo, entra en juego la actualización de los pesos de cada conexión, esto contribuye en un futuro a que la neurona de salida detecte rápidamente el patrón de entrada. En la Fig. 6 se muestra la etapa final del entrenamiento. Se puede observar que la neurona sólo necesita unos poquitos pulsos para activarse y producir un disparo, prueba de que ya se ha entrenado y ha reconocido el patrón a la entrada (del orden de décimas de ms).

Cada una de las diez neuronas ha sido entrenada con 5000 imágenes codificando uno de los dígitos. Estos dígitos han sido obtenidos de la base de datos MNIST [4]. A modo de ejemplo, en la Tabla 1 se muestra el tiempo que tarda en disparar cada una de las neuronas de salida una vez entrenadas cuando a la entrada se introducen cinco imágenes correspondientes al dígito '1'. Este caso (ilustrativo) no es óptimo ya que para dos de las cinco

imágenes son otras neuronas de salida las que han disparado un pulso a la salida de un modo más rápido (en este caso la neurona que codifica el dígito '8').

El sistema ha sido testeado utilizando 200 imágenes correspondientes a cada dígito. Si consideramos éxito el que haya disparado la neurona deseada (aquella entrenada ante el dígito a la entrada) antes que las demás, podemos calcular la tasa de éxito como:

$$rate = \frac{succ}{total} \cdot 100 \quad (5)$$

En esta ecuación, *succ* corresponde al número de entradas reconocidas correctamente y *total* corresponde al número de imágenes testeadas. En nuestro experimento la tasa de reconocimiento ha sido del 80.56%.

IV. CONCLUSIONES

En el trabajo presentado se ha demostrado la posibilidad de incluir mecanismos de aprendizaje no supervisados bioinspirados en una aplicación sencilla de reconocimiento de dígitos. Nótese que en este trabajo el objetivo último no era alcanzar una tasa de reconocimiento elevada, sino demostrar cómo un sistema sencillo puede ser entrenado utilizando el algoritmo bioinspirado STDP y proporcionar resultados aceptables. Es posible alcanzar mejores resultados haciendo el sistema un poco más complejo utilizando tal vez mayor resolución en las imágenes, un banco de filtros mayor y añadiendo inhibición entre las neuronas de salida durante la etapa de entrenamiento, de modo que cuando una neurona de salida dispare un pulso, este pulso se utilice para inhibir e impedir que el resto de neuronas "aprendan" el patrón que ha hecho disparar a la primera de ellas [3].

Un sistema como el descrito puede ser implementado totalmente con módulos convolucionales bioinspirados, tales como módulos de convolución basados en el protocolo AER (*Address-Event-Representation*) [6].

AGRADECIMIENTOS

Este trabajo ha sido financiado en parte por el proyecto MITRA (TEC2010-21619-C04-02), del Ministerio de Ciencia e Innovación.

REFERENCIAS

- [1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.
- [2] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust object recognition with cortex-like mechanisms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 411–426, 2007.
- [3] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Competitive STDP-Based Spike Pattern Learning," *Neural Comput*, 2009, vol. 21, pp. 1259–1276.
- [4] Yann LeCun, Website: <http://yann.lecun.com/exdb/mnist/>
- [5] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 8, pp. 837–842, Aug. 1996.
- [6] J. A. Pérez-Carrasco, et al., "Advanced vision processing systems: Spike-based simulation and processing", *LNCIS*, vol. 5807, pp. 640–651, 2009.

7. BIBLIOGRAFÍA

- [1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition”, *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.
- [2] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, “Robust object recognition with cortex-like mechanisms”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 411–426, Marzo 2007.
- [3] José Antonio Pérez Carrasco, “Herramienta de simulación para construir y analizar sistemas complejos y jerárquicamente estructurados basados en AER que implementan procesamiento de la información visual”, Tesis Doctoral. Sevilla, Marzo 2011.
- [4] G. M. Shepherd, “The Synaptic Organization of the Brain”, 3rd ed. New York: Oxford Univ. Press, 1990.
- [5] E. T. Rolls and G. Deco, “Computational Neuroscience of Vision”, New York: Oxford Univ. Press, 2002.
- [6] R. DeValois, D. Albrecht, and L. Thorell, “Spatial frequency selectivity of cells in macaque visual cortex”, *Vis. Res.*, vol. 22, pp. 545-559, 1982.
- [7] R. DeValois, E. Yund, and N. Hepler, “The orientation and direction selectivity of cells in macaque visual cortex”, *Vis. Res.*, vol. 22, pp. 531-544, 1982.

- [8] P. H. Schiller, B. L. Finlay, and S. F. Volman, “Quantitative studies of single-cell properties in monkey striate cortex”, *Spatial frequency, J. Neurophysiol.* vol. 39(6), pp. 1334-1351, 1976.
- [9] K. Fukushima and N. Wake, “Handwritten alphanumeric character recognition by the neocognitron”, *IEEE Trans. Neural Netw.*, vol. 2(3), pp. 355-365, Mayo 1991.
- [10] S. Grossberg, E. Mingolla and J. Williamson, “Synthetic aperture radar processing by a multiple scale neural system for boundary and surface representation”, *Neural Netw.*, vol. 8(7/8), pp. 1005-1028, 1995.
- [11] S. Lawrence, C. L. Giles, A. Tsoi, and A. Back, “Face recognition: A convolutional neural network approach”, *IEEE Trans. Neural Netw.*, vol. 8(1), pp. 98-113, Enero 1997.
- [12] T. Masquelier and S. J. Thorpe, “Learning to recognize objects using waves of spikes and Spike Timing-Dependent Plasticity”, 2009.
- [13] T. Masquelier, R. Guyonneau, and S. J. Thorpe, “Competitive STDP-Based Spike Pattern Learning”, *Neural Comput*, 2009, vol 21, pp. 1259-1276.
- [14] Yann LeCun, Website: <http://yann.lecun.com/exdb/mnist/>
- [15] Paniagua, R.; Nistal, M.; Sesma, P.; Álvarez-Uría, M.; Fraile, B.; Anadón, R. y José Sáez, F., “Citología e histología vegetal y animal”, (2002).
- [16] Eddie Marrero, Website: <http://academic.uprm.edu/~eddiem/psic3001/id36.htm>
- [17] Tortora Grabowski, “Anatomía y Fisiología”, Editorial Oxford, 9ª. Edición, México, 2002.
- [18] Website: http://enciclopedia.us.es/index.php/Impulso_nervioso

- [19] Luis Palacios Raufast, Josefina Blasco Mínguez, Teresa Pagés Costas y Vicente Alfaro González, “Fisiología Animal”, Edicions Universitat Barcelona, p.47-48, (2005).
- [20] Bear MF, Connors BW, Paradiso M.A., “Neurociencia: explorando el cerebro”, Barcelona: Masson, 2002.
- [21] Morris, R.G.M. et al, “Elements of a neurobiological theory of the hippocampus: the role of activity dependents synaptic plasticity in memory”, *Phil. Trans. R. Soc. Lond. B*, N° 358, 2003, pp. 773-786.
- [22] R. Guyonneau et al., “Illustration STDP with reproducible spike waves”, *Neural Comp*, 2005.
- [23] T. Masquelier, R. Guyonneau, S. J. Thorpe, “Spike Timing Dependent Plasticity Finds the Start of Repeating Patterns in Continuous Spike Trains”, Enero 2008.
- [24] P. Lichtsteiner, C. Posch, and T. Delbruck, “A 128x128 120 dB 30 mW asynchronous vision sensor that responds to relative intensity change”, *IEEE J. Solid-State Circuits*, vol. 43(2), pp. 566-576, Febrero 2008.
- [25] J. Costas-Santos, T. Serrano-Gotarredona, R. Serrano-Gotarredona, and B. Linares-Barranco, “A contrast retina with on-chip calibration for neuromorphic spike-based AER vision systems”, *IEEE Trans. Circuits Syst. I*, Reg. Papers, vol. 54(7), pp. 1444-1458, Julio 2007.
- [26] R. Serrano-Gotarredona, T. Serrano-Gotarredona, A. Acosta-Jiménez, and B. Linares-Barranco, “A neuromorphic cortical layer microchip for spike based event processing vision systems”, *IEEE Trans. Circuits Syst. I*, Reg. Papers, vol. 53(12), pp. 2548-2566, Diciembre 2006.

- [27] E. Kandel, J. Schwartz, and T. M. Jessel, “Principles of Neural Science. Elsevier”, New York, 1991.
- [28] E. D. Adrian, Y. Zotterman, “The impulses produced by sensory nerve endings: Part II: The response of a single end organ”, *Journal of Physiology*, no 61, pp. 151-71, 1926.
- [29] R. Stein, E. Gossen, K. Jones, “Neuronal variability: noise or part of the signal?”, *Nature Reviews Neuroscience*, vol. 6, pp. 389-397, 2005.
- [30] A. L. Jacobs, et al., “Ruling out and ruling in neural codes,” *Proc Natl Acad Sci U S A*, vol. 106, pp. 5936-41, 2009.
- [31] S. J. Thorpe, “Spike arrival times: A highly efficient coding scheme for neural networks”, *Parallel processing in neural systems and computers*, R. Eckmiller, G. Hartmann, and G. Hauske, Editors. 1990, Elsevier: North-Holland. p. 91-94.
- [32] R. VanRullen, R. Guyonneau, and S. J. Thorpe, “Spike times make sense”, *Trends Neurosci*, vol 28, pp. 1-4, 2005.
- [33] R. VanRullen and S. J. Thorpe, “Rate coding versus temporal order coding: what the retinal ganglion cells tell the visual cortex”, *Neural Comput*, vol 13, pp. 1255-83, 2001.
- [34] T. Gollisch and M. Meister, “Rapid neural coding in the retina with relative spike latencies” *Science*, vol 319, pp. 1108-11, 2008.
- [35] M. Sivilotti, “Wiring considerations in analog VLSI systems with application to field-programmable networks”, *Ph.D. dissertation, Comput. Sci. Div., California Inst. Technol., Pasadena, CA*, 1991.

-
- [36] K. Boahen, “Point-to-point connectivity between neuromorphic chips using address events”, *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 47(5), pp. 416-434, Mayo 2000.
- [37] R. Serrano-Gotarredona, et al, “AER Building Blocks for Multi-Layers Multi-Chips Neu-romorphic Vision Systems, in *Advances in Neural Information Processing Systems*”, Vol. 18, Y. Weiss and B. S. and J. Platt (Eds.), (NIPS'06), MIT Press, Cambridge, MA, 1217-1224, (2006).
- [38] T. Serrano-Gotarredona, A. G. Andreou, and B. Linares-Barranco, “AER image filtering architecture for vision-processing systems”, *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 46(9), pp. 1064-1071, Septiembre 1999.
- [39] A. Cohen, R. Etienne-Cummings, T. Horiuchi, G. Indiveri, S. Shamma, R. Douglas, C. Koch, and T. Sejnowski, “Rep. 2004 Workshop on Neuromorphic Eng. Telluride”, CO, del 27 de Junio al 17 Julio, 2004.
- [40] F. Gómez-Rodríguez, R. Paz, A. Linares-Barranco, M. Rivas, L. Miro, G. Jiménez, A. Civit, “AER tools for Communications and Debugging”, *Proc. IEEE ISCAS06. Kos, Greece*, Mayo 2006.
- [41] José Antonio Pérez-Carrasco, Begoña Acha, Carmen Serrano, Luis Camuñas-Mesa, Teresa Serrano-Gotarredona, Bernabé Linares-Barranco, “Fast Vision Through Frameless Event-Based Sensing and Convolutional Processing: Application to Texture Recognition”, *IEEE Transactions On Neural Networks*, Vol. 21, no. 4, Abril 2010.
- [42] B. S. Manjunath and W. Y. Ma, “Texture features for browsing and retrieval of image data” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol.18, no. 8, pp. 837–842, Agosto. 1996.
- [43] Hebb, D y Penfield, W., “Human behaviour after extensive bilateral removal from the frontal lobes”, *Archives of Neurology and Psychiatry.*, pp. 421–436. (1940).

[44] Shouval, Harel, “The Synaptic basis for Learning and Memory: A theoretical approach”, The University of Texas Health Science Center at Houston. Enero 2005.

[45] Timothee Masquelier and Simon J. Thorpe, “Learning to recognize objects using waves of spikes and Spike Timing-Dependent Plasticity”, 2009.

[46] Website: <http://apt.cs.man.ac.uk/projects/SpiNNaker/>

[47] Website: <http://apt.cs.man.ac.uk/projects/SpiNNaker/architecture/>