CONTROL INTELIGENTE DE UNA PRÓTESIS MIOELÉCTRICA DE MIEMBRO SUPERIOR BASADO EN REDES NEURONALES

LAURA PATRICIA DÍAZ GÓMEZ
COD: 18201003

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
ESCUELA DE CIENCIAS NATURALES E INGENIERÍA
FACULTAD DE INGENIERÍA MECATRÓNICA
BUCARAMANGA
2006

CONTROL INTELIGENTE DE UNA PRÓTESIS MIOLECTRICA DE MIEMBRO SUPERIOR BASADO EN REDES NEURONALES

LAURA PATRICIA DÍAZ GÓMEZ
COD: 18201003

DIRECTOR: Ph.D.Dr.Sc.Ing. Antonio Faustino Muñoz Moner

ASESOR: Ing. Diego Tibaduiza
Ing. Andrés Cruz Ballesteros (Egresado UNAB – Actual Gerente de BIONIX)

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
ESCUELA DE CIENCIAS NATURALES E INGENIERÍA
FACULTAD DE INGENIERÍA MECATRÓNICA
BUCARAMANGA
2006

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

OBJETIVO GENERAL

Diseñar un sistema inteligente, basado en Redes Neuronales Artificiales (RNA'S), para controlar el movimiento de una pinza para una prótesis de brazo a partir de dos señales electromiográficas ya tratadas.

OBJETIVOS ESPECÍFICOS

- Investigar los tipos de sistemas inteligentes con Redes Neuronales Artificiales (RNA'S) y basado en esto, elegir la estructura de acuerdo al tipo de respuesta que se requiere para llevar acabo el control de la prótesis.

- Implementar en el microcontrolador una forma de capturar y digitalizar la información necesaria de las señales electromiográficas para su posterior almacenamiento y procesamiento.

- Estipular los parámetros de salidas deseadas para la Red Neuronal Artificial (RNA) respecto a la información capturada, digitalizada y almacenada de las señales electromiográficas, de la persona a usar la prótesis.

- Diseñar el algoritmo de entrenamiento de la Red Neuronal Artificial (RNA) ya escogida, para que el funcionamiento de la prótesis sea adaptable a las señales electromiográficas, de la persona a usar la prótesis, ya almacenadas.

- Diseñar el algoritmo para ejecutar la Red Neuronal Artificial (RNA) ya escogida, teniendo en cuenta que el procesamiento se realice mediante los datos resultantes de la etapa de entrenamiento.

- Acondicionar la salida de la Red Neuronal Artificial (RNA) al hardware que genera el movimiento de la pinza en la prótesis.

- Probar el algoritmo de control inteligente en el sistema de la prótesis desarrollada por Bionix y con esto brindar una solución a la empresa para que sus pacientes se rehabiliten cómodamente, de manera que la prótesis que la prótesis sea quien se adapte a ellos.

INTRODUCCIÓN

Actualmente en nuestro país vivimos un conflicto armado en el cual cada segundo miles de inocentes pierden partes de su cuerpo e incluso su vida. Basado en estos hechos empresas nacionales como BIONIX, especializadas en el área de la bioelectrónica buscan trabajar innovando tecnología para brindar solución a dichos problemas.

Con el fin de ayudar a una gran mayoría de discapacitados BIONIX desarrolla prótesis mioeléctricas de miembro superior y viendo la necesidad de continuar en una constante investigación, la empresa ofrece una posibilidad a estudiantes de practica académica para que se involucren y brinden soluciones a la constante evolución que requiere este tipo de tecnología.

Bionix está conformado por egresados de la facultad, quienes desarrollaron como proyecto de grado una prótesis mioeléctrica a nivel didáctico, aunque la nueva y moderna prótesis que ellos desarrollan tuvo su base en el anterior proyecto, este proyecto es aplicado a la nueva prótesis, por su sistema de mecanismo y actuadores[1] modernos y por el tipo de procesador en el que voy a aplicar el control inteligente de la prótesis.

Las prótesis que BIONIX desarrolla al igual que muchas prótesis del mundo no ofrecen un nivel de entrenamiento en los pacientes para que sus señales electromiográficas[2] sean mejor implementadas en el mecanismo, para lo cual la empresa propuso el desarrollo de un algoritmo inteligente basado en redes

[1]Actuadores: motores, pistones o artefactos que reaccionen a una entrada y generen movimiento.
[2] Señales Electromiogáficas: potencial de acción generado por los músculos

neuronales que de una solución más eficiente de adaptación de aparatos protésicos a un problema social como son los discapacitados en Colombia.

Este proyecto esta basado primordialmente en lograr crear nuevas soluciones a problemas existentes en el mejoramiento de la integrabilidad a sistemas de prótesis de brazo para mejorar la calidad de vida en personas parcial o totalmente discapacitadas, aprovechando al máximo todas las ventajas que nos ofrece el procesamiento de las señales electromiográficas, mediante el diseño de un sistema inteligente, basado en Redes Neuronales Artificiales (RNA'S)[3], para controlar el movimiento de una pinza en una prótesis de brazo a partir de dos señales electromiográficas ya tratadas, implementado bajo la tecnología de circuitos programables; teniendo en cuenta que la recepción de estas señales se tomarán directamente de un paciente o persona a utilizar la prótesis.

Durante las etapas de entrenamiento y de ejecución de la Red Neuronal Artificial (RNA), se iniciará la captura, digitalización y almacenamiento de los datos a partir de la información que suministra las señales electromiográficas. Los datos almacenados serán utilizados para el entrenamiento de la Red Neuronal Artificial (RNA) en una etapa llamada Training[4] con el fin de estandarizar el sistema de control para la persona a usar la prótesis, y procesados en una segunda etapa llamada Run o de ejecución de la RNA con el fin de obtener ya los resultados óptimos del pasado entrenamiento.

El sistema guiará constantemente, mediante ayuda visual o audible, al paciente o persona a manipular la prótesis, indicándole los pasos a seguir durante el

[3] RNA'S: Tipo de algoritmo de control inteligente que emula la lógica del sistema neuronal en el cerebro humano.
[4] Training: proceso de entrenamiento previo que se realiza antes de ejecutar una red neuronal.

funcionamiento de la misma, sobre todo al momento de capturar los datos a utilizar en la etapa de entrenamiento de la RNA.

Este diseño se caracterizará por tener dos etapas en su diseño, etapa Training y etapa Run[5]. En la etapa Training se encarga de tomar la información de una serie de muestras de las señales electromiográficas, ya almacenada, del paciente o persona que quiera usar la prótesis, para entrenar la RNA establecida. Y en la etapa Run se ejecuta la RNA ya entrenada y sus salidas son transformadas al movimiento necesario de la pinza para el óptimo funcionamiento de la prótesis en el paciente o persona que se estableció el entrenamiento de la RNA en la etapa Training del diseño.

[5] Run: modo de ejecución para correr la red neuronal.

CAPITULO 1
ESTADO DEL ARTE

Los sistemas de control de las diferentes prótesis que existen hoy en día en el mercado y las que surgen a través de las diferentes investigaciones o tesis de grado, son diseñas en laboratorio de manera personalizada; esto significa que una prótesis es de uso exclusivo para un paciente.

En la última década el mundo ha cambiado mucho y también sus necesidades. Estos cambios también se han producido en el área de la tecnología. Dado a que este mercado se esta expandiendo debido a desarrollos tecnológicos cada vez mas complejos para crear nuevas y mejores soluciones.

La recepción de señales y procesamiento de la información es una de las funciones naturales más importantes usadas por el hombre. Estas funciones son desde hace unas décadas, símbolo de inspiración del hombre en la búsqueda de mejores soluciones; un ejemplo claro de esto es el de querer simular el funcionamiento de las diferentes partes del cuerpo humano de una forma artificial mediante las mismas señales eléctricas que este genera.

Teniendo en cuenta la importancia del tema a nivel mundial, se han creado ya diseños de algoritmos inteligentes, no solo basados en las redes neuronales, si no también en diferentes métodos que se aplican a la inteligencia artificial.

El hombre se ha caracterizado siempre por su búsqueda constante de nuevas vías

para mejorar sus condiciones de vida. Estos esfuerzos le han servido para reducir el trabajo en aquellas operaciones en las que la fuerza juega un papel primordial.

Los progresos obtenidos han permitido dirigir estos esfuerzos a otros campos, como por ejemplo, a la construcción de máquinas calculadoras que ayuden a resolver de forma automática y rápida determinadas operaciones que resultan tediosas cuando se realizan a mano.

Uno de los primeros en acometer esta empresa fue Charles Babbage, quien trató infructuosamente de construir una máquina capaz de resolver problemas matemáticos.

Posteriormente otros tantos intentaron construir máquinas similares, pero no fue hasta la Segunda Guerra Mundial, cuando ya se disponía de instrumentos electrónicos, que se empezaron a recoger los primeros frutos. En 1946 se construyó la primera computadora electrónica, ENIAC. Desde entonces los desarrollos en este campo han tenido un auge espectacular.

Estas máquinas permiten implementar fácilmente algoritmos para resolver multitud de problemas que antes resultaban engorrosos de resolver. Sin embargo, se observa una limitación importante: ¿qué ocurre cuando el problema que se quiere resolver no admite un tratamiento algorítmico[6], como es el caso, por ejemplo, de la clasificación de objetos por rasgos comunes? Este ejemplo demuestra que la construcción de nuevas máquinas más versátiles requiere un enfoque del problema desde otro punto de vista. Los desarrollos actuales de los científicos se dirigen al estudio de las capacidades humanas como una fuente de nuevas ideas para el diseño de las nuevas máquinas. Así, la inteligencia artificial es un intento

[6] Tratamiento algorítmico: procesamiento de diseño que resuelve problemas.

por descubrir y describir aspectos de la inteligencia humana que pueden ser simulados mediante máquinas. Esta disciplina se ha desarrollado fuertemente en los últimos años teniendo aplicación en algunos campos como visión artificial, demostración de teoremas, procesamiento de información expresada mediante lenguajes humanos, etc.

Las redes neuronales no son más que otra forma de emular[7] ciertas características propias de los humanos, como la capacidad de memorizar y de asociar hechos. Si se examinan con atención aquellos problemas que no pueden expresarse a través de un algoritmo, se observará que todos ellos tienen una característica en común: la experiencia. El hombre es capaz de resolver estas situaciones acudiendo a la experiencia acumulada. Así, parece claro que una forma de aproximarse al problema consista en la construcción de sistemas que sean capaces de reproducir esta característica humana. En definitiva, las redes neuronales no son más que un modelo artificial y simplificado del cerebro humano, que es el ejemplo más perfecto del que disponemos para un sistema que es capaz de adquirir conocimiento a través de la experiencia. Una red neuronal es "un nuevo sistema para el tratamiento de la información, cuya unidad básica de procesamiento está inspirada en la célula fundamental del sistema nervioso humano: *la neurona*".

Todos los procesos del cuerpo humano se relacionan en alguna u otra forma con la (in)actividad de estas neuronas. Las mismas son un componente relativamente simple del ser humano, pero cuando millares de ellas se conectan en forma conjunta se hacen muy poderosas. Lo que básicamente ocurre en una neurona biológica es lo siguiente: la neurona es estimulada o excitada a través de sus *entradas* (inputs) y cuando se alcanza un cierto umbral, la neurona se dispara o

[7] Emular: Imitar una capacidad o características propias

activa, pasando una señal hacia el *axon*[8]. Posteriores investigaciones condujeron al descubrimiento de que estos procesos son el resultado de eventos electroquímicos.

Como ya se sabe, el pensamiento tiene lugar en el cerebro, que consta de billones de neuronas interconectadas. Así, el secreto de la "inteligencia" -sin importar como se defina- se sitúa dentro de estas neuronas interconectadas y de su interacción.

También, es bien conocido que los humanos son capaces de aprender. Aprendizaje significa que aquellos problemas que inicialmente no pueden resolverse, pueden ser resueltos después de obtener más información acerca del problema. Por lo tanto, las Redes Neuronales[1]:

_ Consisten de unidades de procesamiento que intercambian datos o información.

_ Se utilizan para reconocer patrones, incluyendo imágenes, manuscritos y secuencias de tiempo (por ejemplo: tendencias financieras).

_ Tienen capacidad de aprender y mejorar su funcionamiento. Una primera clasificación de los modelos de redes neuronales podría ser, atendiendo a su similitud con la realidad biológica:

1) El modelo de tipo biológico. Este comprende las redes que tratan de simular los sistemas neuronales biológicos, así como las funciones auditivas o algunas funciones básicas de la visión.

[8] Axon: parte de la neurona que se encarga del transporte de orgánulos y sustancias y la conducción del impulso nervioso.

2) El modelo dirigido a aplicación. Este modelo no tiene por qué guardar similitud con los sistemas biológicos. Su arquitectura está fuertemente ligada a las necesidades de las aplicaciones para la que es diseñada[1].

CAPITULO 2

PLANTEAMIENTO DEL PROBLEMA

En la figura que se observa a continuación se sintetiza el diagrama del proceso que se realizó en el proyecto.

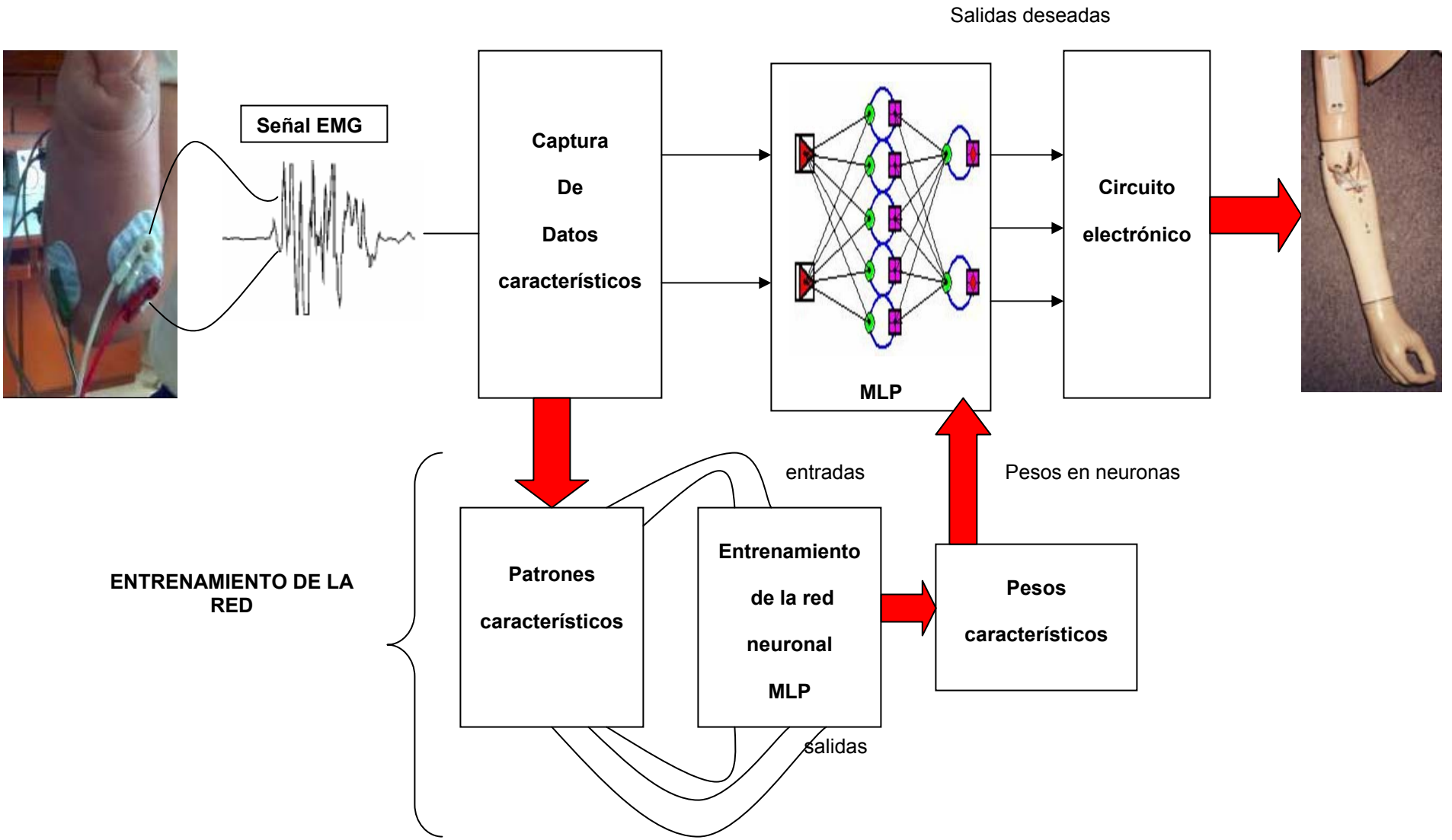
El funcionamiento del sistema depende del seguimiento de los pasos que se siguen en el diagrama.

Después de acondicionar la señal, se caracterizan unas entradas para capturar y unas salidas a las cuales se quiere llegar.

Para esto, se necesita que los datos pasen por un entrenamiento previo, es decir un proceso matemático que permite modificar los pesos dentro del sistema y llegar a una respuesta optima y deseada para el paciente.

Teniendo los pesos ya procesados, se llevan a la bloque de ejecución donde se retoman formulas que llevan a activar o desactivar el motor y darle ciertas características de velocidad.

Figura 1. Diagrama del diseño control inteligente



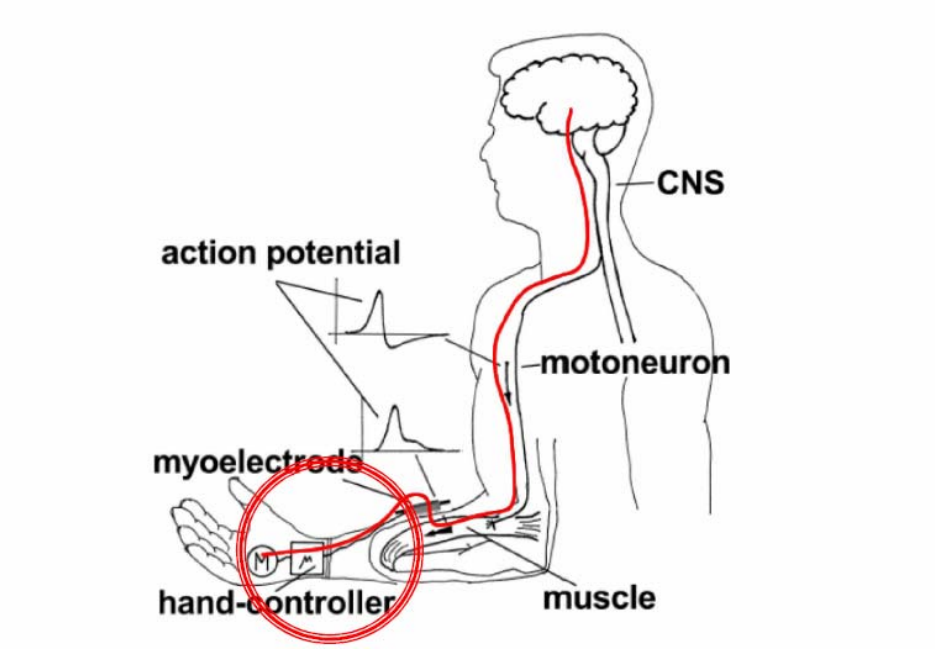
CAPITULO 3

MARCO TEÓRICO

3.1 ELECTROMIOGRAFÍA[2]

La base de toda exploración electrofisiológica es el registro de los potenciales de las células excitables. La **electromiografía** se ocupa del registro de dichos potenciales evocados voluntariamente en el músculo.

Figura 2. Proceso neuronal de las señales electromiográficas



Merletti, Roberto. Parker, Philip A. Electromyography. Physiology, engineering, and noninvasive applications. Edit, IEEE Press Series in Biomedical Engineering. 2004

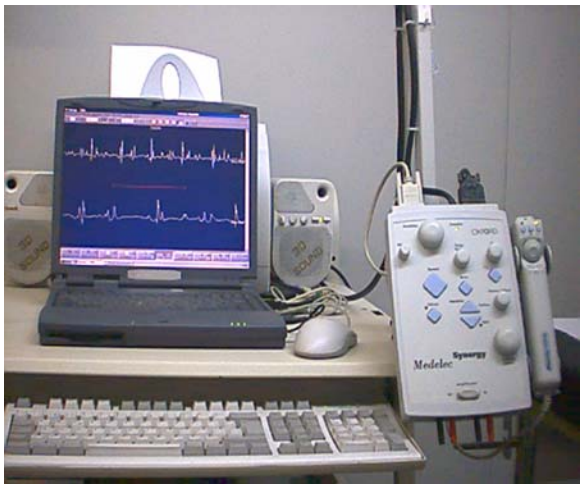
Las *propiedades eléctricas de las fibras excitables*, nerviosas y musculares, derivan de la existencia de una membrana semipermeable que separa fluidos intracelulares y extracelulares con diferente concentración iónica que origina un potencial transmembrana. El espacio intracelular del axón contiene una alta concentración de ión K y otros aniones así como de aminoácidos y proteínas de carga negativa. En el espacio extracelular predomina el ión Na y el ión Cl. La impermeabilidad de la membrana en reposo no solo a las moléculas proteicas sino también, en diferente proporción, a estos iones, es la causa del mantenimiento de la diferencia de potencial entre ambos lados, negativa en el interior, de unos -70-90 mV. Potenciales electrotónicos de suficiente intensidad en la membrana axonal inducen cambios en la actividad de los canales específicos lo que permite el paso de los iones, fundamentalmente del Na, a través de la membrana.

Se generan de este modo *potenciales de acción*[9] que suceden a la inversión de la carga eléctrica entre ambos lados de la membrana, que la sitúan en los +30mV que corresponde al potencial de equilibrio para el Na. La bomba de Na-K es capaz posteriormente de reequilibrar la concentración iónica transportándolos contra gradiente en un sistema que consume energía. El potencial de acción creado es capaz entonces de inducir corrientes electrotónicas[10] en la membrana que inducen en las zonas inmediatamente cercanas el mismo proceso de cambios estructurales en los canales iónicos que dependen del voltaje. Se produce así un nuevo potencial de acción que de esta forma se propaga a lo largo del axón o de la fibra muscular. [2]

[9] Potencial de acción: cambio de energía que responde a un estimulo mecánico o eléctrico.
[10] Corrientes electrotónicas: propiedad química en la propagación de información en las neuronas

3.1.1 Equipo Electromiográfico[2]

Figura 3. Equipo usado para realizar electromiografias



http://personales.ya.com/emgnm/emg.htm

La electromiografía es el estudio electrofisiológico[11] del sistema neuromuscular. No es una prueba complementaria, sino la prolongación del estudio clínico neurológico. Dicha exploración se diseña en cada caso en función de la historia clínica y de la exploración neurológica, y puede modificarse según los datos que se vayan obteniendo.

- El electromiógrafo es un osciloscopio de tecnología avanzada que debe cumplir dos requisitos básicos imprescindibles: 1) precisión y exactitud en la medida de las señales.
- Seguridad en uso:

[11] Electrofisiológico: es un procedimiento para examinar la actividad eléctrica y sus patologías.

Debe ir equipado con dos o más amplificadores cuyas ganancias puedan ir desde menos de un micro voltio hasta 10 milivoltios; el barrido del osciloscopio debe poder oscilar entre 1 msg/división y un segundo o más por división; el rango de frecuencias debe ir desde 2 Hz hasta 20 kHz.

Es imprescindible que esté equipado con un promediador y con el dispositivo de disparo (trigger) de la señal y línea de retraso.

Los estimuladores eléctricos (uno al menos) son parte integral del electromiógrafo y deben poder generar estímulos de duraciones y frecuencia variables con posibilidad de aplicar trenes de estímulos.

Para la estimulación magnética transcraneal[12] se emplea un estimulador magnético independiente conectado al electromiógrafo.

El osciloscopio donde se visualiza la señal se complementa con los altavoces. En la actualidad, la mayoría de las técnicas se realizan mediante un programa informático con posibilidades de archivo parcial o total de los datos obtenidos y un sistema de impresión.

Aparte del electromiógrafo es obligado tener aparatos de cuantificación de la sensibilidad (térmica, vibratoria etc.), dinamómetros, algómetros[13].

Los electrodos y el material fungible deben cumplir criterios de seguridad, precisión y exactitud técnica, así como duración (longevidad) máxima.

[12] Transcraneal: técnica neurofisiológica que permite una estimulación no invasiva de la corteza cerebral
[13] Algómetro: son utilizados para cuantificar la. sensibilidad de presión de un punto muscular

3.2 LA SEÑAL EMG PARA EL CONTROL DE PRÓTESIS [3]

Para utilizar la señal EMG se necesita conocer sus características, la forma en que se genera y la manera en que puede ser registrada.

Una característica muy importante de la contracción muscular es que la respuesta del músculo es el resultado de una suma en el espacio (agrupando las unidades motoras dispersas en el músculo). y en el tiempo (aumentando la relación de disparo de las unidades motoras individuales), de un gran número de eventos químicos y mecánicos individuales. La tensión generada por las interacciones de la actina-miosina[14] se combinan al nivel del sarcómero[15]; las tensiones del sarcómero se suman a nivel de las miofibrillas[16] y las tensiones de las miofibrillas, se suman a nivel de la fibra muscular; las tensiones de la fibra muscular se suman a nivel de la unidad motora, donde finalmente las tensiones de la unidad motora se suman en el tendón del músculo. El proceso anteriormente descrito es la generación de la señal EMG.

En 1849, Dubois-Reymond registró cambios en los potenciales eléctricos de los músculos humanos. En el año de 1945, se sugirió que las señales mioeléctricas detectadas en los músculos del antebrazo se podían usar para el control de prótesis; acto seguido, se hicieron experimentos clínicos y años más tarde (1960), en Rusia, se colocaron a amputados manos eléctricas controladas mioeléctricamente.

Uno de los usos de la actividad mioeléctrica es precisamente utilizarla como señal de control; por ejemplo, existe la prótesis mioeléctrica de dos funciones,

[14] Actina-miosina: Proteínas propias de los músculos
[15] Sarcomero: pequeñas fibras musculares
[16] Miofibrillas: unión de sarcomeros

que es controlada por dos señales mioeléctricas aisladas que manejan un motor bidireccional (abertura y cierre de la mano); sin embargo, es difícil para el amputado operar funciones adicionales aislando más señales mioeléctricas. Las señales en muchos casos interfieren, es decir, los movimientos protésicos no pueden ser ejecutados independientemente.

Se ha demostrado que una persona normal deja la organización de los movimientos de sus miembros a su subconsciente; por lo tanto, los objetivos en el diseño de un sistema protésico, son los de lograr que el substituto artificial trabaje en la misma forma que el miembro perdido, es decir, permitir que el amputado pueda concentrarse en otras tareas, diferentes a la manipulación de su prótesis, operándola subconscientemente.

Si las vías neuronales de los músculos están intactas, el amputado puede producir un flujo de señales neuronales, en la misma forma como en el caso normal, usando la imagen del miembro perdido. Se sabe que la pérdida de un miembro puede estar seguida por una “ilusión” de que el miembro aún está ahí.

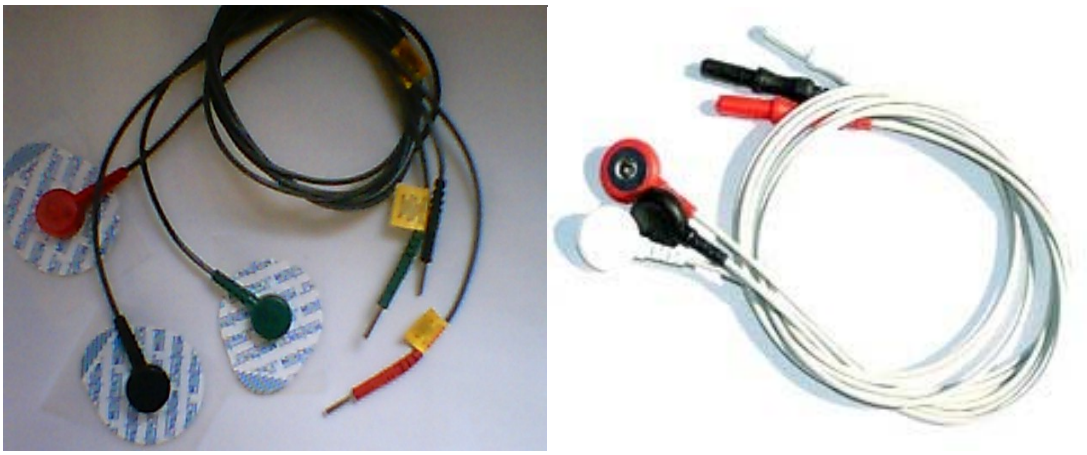
3.2.1 Electrodos Para Electromiografía [5]: La empresa Kendall, ofrece una gama de electrodos adhesivos desechables llamados: MediTrace 200 series.

Figura 4. Electrodos Autoadhesivos

Kendall LTP-Monitoring & Diagnostic

3.2.2 Cable Blindado, Conector De Electrodos [6]: Estos cables tienen una extensión de 24 pulgadas con terminales de seguridad DIN, que conectan al electrodo. El cable blindado minimiza interferencia electromagnética producida por ruido externo.

Figura 5. Conectores entre electrodos y equipo de adquisición



Esparza Franco, C.H. Equipo de Adquisición de señales de Electro conducción. Tesis, 2003.

3.3 EQUIPO PARA EL ESTUDIO DE BIOSEÑALES

3.3.1 Biopac Research [4]: BIOPAC es un sistema compuesto por hardware y software para la adquisición, análisis y tratamiento digital de bioseñales.

Figura 6. Equipo Biopac, para el análisis de bioseñales

http://www.teoloyucan.com/biopac_pri.html

Este sistema es el estándar internacional en adquisición, tratamiento y análisis de señales biológicas. Al manejarlo, se puede ver la forma de las diferentes señales electromiográficas (EMG) tanto en personas sanas como en pacientes con diferentes lesiones del sistema nervioso o músculo esquelético entre ellos los amputados. También estudia su espectro de frecuencia a través del algoritmo de la transformada rápida de Fourier (FFT), para tener elementos de juicio técnicos en la determinación de las frecuencias de corte de los filtros de la etapa de procesamiento análogo de la señal. Así mismo, con la aplicación de los filtros digitales FIR que posee el BIOPAC se podrán simular diferentes tipos de filtros para luego implementarlos con hardware.

Además del hardware y software para bioseñales, también se manejan otros equipos del laboratorio como son el osciloscopio, multímetro, un equipo de telemetría, fuentes y generador de señales entre otros.

3.4 INTELIGENCIA ARTIFICIAL

La inteligencia artificial es la disciplina en la cual se desarrollan algoritmos de control que emulan ciertas características propias de los organismos biológicos inteligentes. Básicamente son tres tipos de algoritmos:

3.4.1 Las *Redes Neuronales Artificiales* (RNAs) [7]: Emulan las redes neuronales biológicas y se utilizan para aprender estrategias de control observando la forma como una persona lo hace, son sistemas que aprenden con ejemplos no requieren que la tarea a ejecutar se programe. También se utilizan para aprender on-line la mejor forma para controlar un sistema aplicando entradas, evaluado la calidad de las respuestas para estas entradas, y ajustando la fórmula que genera las acciones de control para mejorar la respuesta del sistema.

3.4.2 Los *Sistemas Difusos* o *Lógica Borrosa* (fuzzy logic) [8]: Emulan la manera en que el cerebro razona o piensa. La Lógica Borrosa se introduce para manejar eficazmente conceptos vagos e imprecisos como los empleados en la vida cotidiana, y que nuestro cerebro está acostumbrado a tratar. Por ejemplo, en la realidad el agua no se presenta en tan sólo dos estados, caliente o fría, como diría la lógica booleana, si no más bien gélida, fría, templada, caliente o quemando. A partir de estos conceptos, y como la generalización de la reglas de la lógica booleana, base de nuestros sistemas digitales, los sistemas borrosos llevan a cabo un tipo de razonamiento aproximado semejante al desarrollado por el cerebro.

3.4.3 Los *algoritmos genéticos* (AGs) es una técnica de búsqueda iterativa inspirada en los principios de selección natural, los AGs no buscan modelar la

evolución biológica si no derivar estrategias de optimización., se pueden aplicar a cualquier problema que requiera encontrar un máximo o un mínimo.

En la robótica, los algoritmos genéticos se utilizan para evolucionar controladores a partir de una población y utilizando el principio de *supervivencia de los mejores* donde *mejo*r se define por la calidad de la respuesta lograda con el controlador.

Las redes neuronales son ideales para toda clase problemas, porque como sus compañeras biológicas, una red neuronal artificial puede aprender y luego ser entrenada para encontrar soluciones, reconocer patrones, clasificar datos y hacer previsión de eventos futuros.

Las principales características que diferencian a las redes neuronales de otras técnicas de inteligencia artificial ya comentadas, son su capacidad de aprendizaje a partir de la experiencia, su velocidad de respuesta una vez concluido el entrenamiento y también su robustez, en el sentido de que el conocimiento que contienen tras el entrenamiento esta repartido en toda la red, de forma que el fallo o supresión de algunas de sus neuronas no impide un cierto numero de respuestas correctas frente a estímulos presentados[9].

3.5 LAS REDES NEURONALES ARTIFICIALES.

Años de investigaciones ha dado lugar al nacimiento de las redes neuronales, se da por la ambición de simular el comportamiento del cerebro humano. Inspirados en el modo en el que las redes neuronales biológicas del cerebro procesan la información.

Las redes neuronales artificiales, mediante un estilo de computación paralelo, distribuido y adaptativo, son capaces de aprender a partir de ejemplos. Estos sistemas imitan esquemáticamente la estructura hardware (neuronal) del cerebro para tratar de reproducir algunas de sus capacidades. En la práctica, una red neuronal artificial puede simularse mediante un programa de ordenador, o bien en circuitos electrónicos específicos.

“Redes neuronales artificiales son redes interconectadas masivamente en paralelo de elementos simples (usualmente adaptativos) y con organización jerárquica, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico.” [10]

3.5.1 Ventajas de las Redes Neuronales Artificiales[11]: Dado a que las redes neuronales artificiales son semejantes a las redes neuronales biológicas, su constitución y fundamentos son parecidos a las del cerebro. Las RNA se caracterizan principalmente por:

● Tener una inclinación natural a adquirir el conocimiento a través de la experiencia, el cual es almacenado, al igual que en el cerebro, en el peso relativo de las conexiones interneuronales.

● Tienen una altísima plasticidad y gran adaptabilidad, son capaces de cambiar dinámicamente junto con el medio.

● Poseen un alto nivel de tolerancia a fallas, es decir, pueden sufrir un daño considerable y continuar teniendo un buen comportamiento, al igual como ocurre en los sistemas biológicos.

● Poseen un comportamiento altamente no-lineal, lo que les permite procesar información procedente de otros fenómenos no-lineales.

3.5.2 Aplicaciones: Las RNAs han sido aplicadas a un creciente número de problemas reales de considerable complejidad, por ejemplo: reconocimiento de

patrones, clasificación de datos, predicciones, etc. Su ventaja más importante está en solucionar problemas que son demasiados complejos para las técnicas convencionales; problemas que no tienen un algoritmo específico para su solución, o cuyo algoritmo es demasiado complejo para ser encontrado.
En general, las Redes Neuronales Artificiales han sido claramente aceptadas como nuevos sistemas muy eficaces para el procesamiento de la información en muchas disciplinas. Estas han dado como resultado una variedad de aplicaciones comerciales de esta tecnología. Algunos campos donde se aplican las redes neuronales son: Finanzas, Tratamiento de textos y proceso de formas, Transportes y Comunicaciones, Negocios, Industria manufacturera, Medicina y salud, Alimentación, Energía, Ciencia e ingeniería.

3.5.3 La Neurona Biológica: La neurona es la unidad biológica básica del cerebro. La figura 7. muestra un esquema simplificado de la fisiología de una neurona típica, algunos elementos ha destacar de su estructura son:

● Las *dentritas*, son la vía de entrada de las señales que se combinan en el cuerpo de la neurona. El cuerpo de la célula, realiza la suma de esas señales de entrada.
● El *axón* es una fibra que lleva la señal desde el cuerpo de la célula hacia otras neuronas.
● La *sinápsis,* es donde se tiene lugar el intercambio de información entre dos neuronas. En los terminales de las sinápsis se encuentran unas vesículas que contienen unas sustancias químicas llamadas neurotransmisores, que ayudan a la propagación de las señales electroquímicas de una neurona a otra.

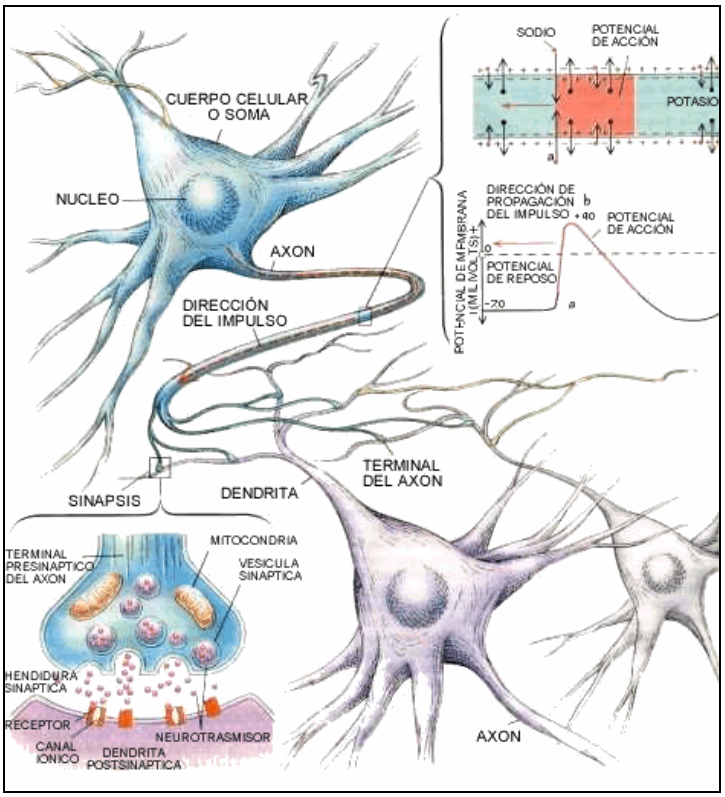
Cuando una neurona esta en reposo, su membrana externa mantiene una diferencia de potencial de -70 mV (la superficie interior es negativa respecto de la superficie exterior). En reposo, la membrana es más permeable a los iones de

potasio que a los iones de sodio. Cuando se estimula la célula, la permeabilidad al sodio se incrementa, lo cual produce una entrada repentina de cargas positivas. Esta entrada produce un impulso – una inversión momentánea del potencial de la membrana. El impulso se inicia en la unión del cuerpo celular y el axón, y se propaga lejos del cuerpo celular. Cuando el impulso alcanza los terminales del axón de la neurona presináptica, esta induce la liberación de moléculas neurotransmisoras. Los transmisores se difunden y alcanzan los receptores de la membrana postsináptica[7].

De esta manera la información se transmite de unas neuronas a otras, es procesada a través de las conexiones sinápticas y las propias neuronas. El aprendizaje de las redes neuronales se produce mediante la variación de la efectividad de las sinapsis, de esta manera cambia la influencia que unas neuronas ejercen sobre otras, de aquí se deduce que la arquitectura, el tipo y la efectividad de las conexiones en un momento dado, representan en cierto modo la memoria o estado de conocimiento de la red.

En la figura 7. se muestra un ejemplo de una neurona biológica. Se estima que hay 26.000 millones (2.6×10^{10}) de neuronas en el cerebro y en un área de un milímetro cuadrado hay aproximadamente 50.000. El tamaño y forma de las neuronas es variable pero poseen las mismas subdivisiones anatómicas.

Figura 7. Fisiología de una neurona biológica

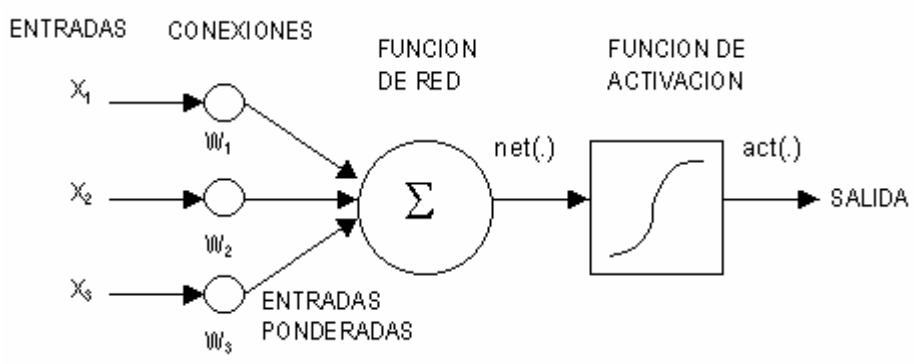


Fuente: Universidad UTP. Redes Neuronales. [En línea]. Colombia. Google. 2004. Disponible en Internet: http://ohm.utp.edu.co/neuronales/main.htm

3.5.4 Modelo de una Neurona Artificial: Concebida originalmente como un intento de modelar la biofisiología del cerebro humano (entender y explicar como funciona y opera el cerebro). En figura 8 se muestra un modelo de una neurona artificial es una imitación del proceso de la célula fundamental del sistema nervioso humano, la *neurona.* Para ello se expresa un modelo matemático simplificado:

$$y(t) = f\left(\sum w_{ij} x_j - \theta\right) \qquad (1)$$

Figura 8. Neurona artificial



Fuente: Instituto de investigación en inteligencia artificial. [En línea]. España. Google. 2004. Disponible en Internet. http://www.iiia.csic.es/~mario/rna/tutorial/RNA_marcos.html

Los elementos fundamentales que constituyen una neurona artificial son los siguientes:

● Las entradas X_i representan las señales que provienen de otras neuronas y que son capturadas por las dendritas.
● Los pesos W_i son la intensidad de la sinápsis que conecta dos neuronas; tanto X_i como W_i son valores reales.
● La Σ es la suma de todos los valores X_i de entradas a la neurona, multiplicados por su correspondiente peso W_i.
● θ es la función umbral que la neurona debe sobrepasar para activarse; este proceso ocurre biológicamente en el cuerpo de la célula.

Las señales de entrada a una neurona artificial X_1, X_2,.., X_n son variables continuas en lugar de pulsos discretos, como se presentan en una neurona biológica. Cada señal de entrada pasa a través de una ganancia o peso, llamado

peso sináptico o fortaleza de la conexión cuya función es análoga a la de la función sináptica de la neurona biológica.

Los pesos pueden ser positivos (excitatorios), o negativos (inhibitorios), el nodo sumatoria acumula todas las señales de entradas multiplicadas por los pesos o ponderadas y las pasa a la salida a través de una función umbral o función de transferencia. La entrada neta a cada unidad puede escribirse de la siguiente manera:

$$neta_i = \sum_{i=0}^{n} w_i x_i = \vec{x}\vec{w} \qquad (2)$$

Una vez que se ha calculado la activación del nodo, el valor de salida equivale a:

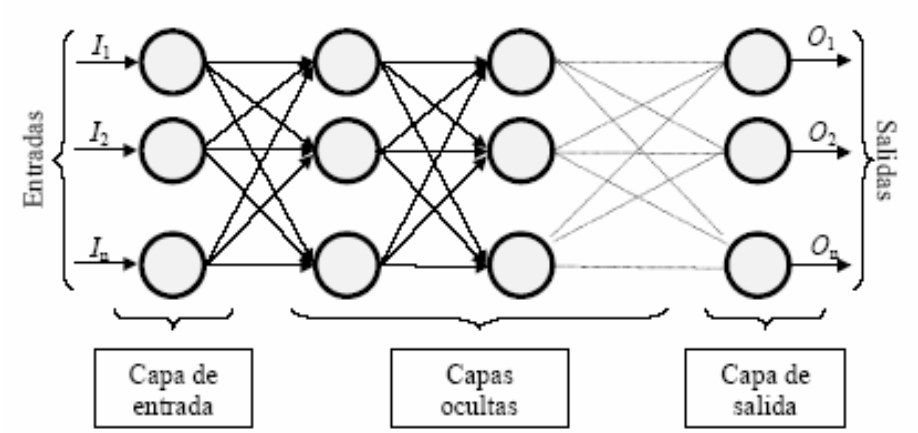
$$x_i = f_i(neta_i) \qquad (3)$$

Donde, f_i representa la *función de activación* para esa unidad, que corresponde a la función escogida para transformar la entrada neta x_i, en el valor de salida y que depende de las características específicas de cada red.

3.6 ELEMENTOS BÁSICOS QUE COMPONEN UNA RED NEURONAL[1]

A continuación se puede ver, en la Figura 9, un esquema de una red neuronal:

Figura 9. Ejemplo de una red neuronal totalmente conectada.



PDF Redes neuronales. Conceptos básicos y aplicaciones. Universidad Tecnológica Nacional – Facultad Regional Rosario. Departamento de Ingeniería Química. Grupo de Investigación Aplicada a la Ingeniería Química (GIAIQ).

La misma está constituida por neuronas interconectadas y arregladas en tres capas (esto último puede variar). Los datos ingresan por medio de la "*capa de entrada*", pasan a través de la "*capa oculta*" y salen por la "*capa de salida*". Cabe mencionar que la capa oculta puede estar constituida por varias capas.

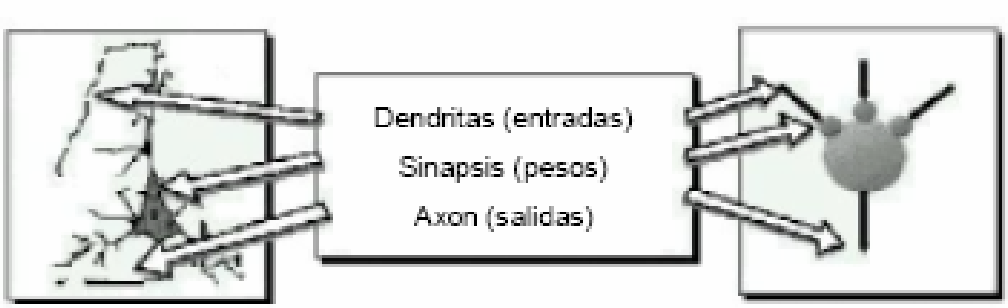
-*Capa de entrada:* es la capa que recibe directamente la información proveniente de las fuentes externas de la red.

- Capas *Ocultas*: son internas a la red y no tienen contacto directo con el entorno exterior. El número de niveles ocultos puede estar entre cero y un número elevado. Las neuronas de las capas ocultas pueden estar interconectadas de

distintas maneras, lo que determina, junto con su número, las distintas topologías de redes neuronales.
- Capas de *salidas*: transfieren información de la red hacia el exterior.

Antes de comenzar el estudio sobre las redes neuronales, se debe aprender algo sobre las neuronas y de cómo ellas son utilizadas por una red neuronal.
En la Figura 10 se compara una neurona biológica con una neurona artificial. En la misma se pueden observar las similitudes entre ambas (tienen entradas, utilizan pesos y generan salidas).

Figura 10. Comparación entre una neurona biológica (izquierda) y una artificial (derecha).



PDF Redes neuronales. Conceptos básicos y aplicaciones. Universidad Tecnológica Nacional – Facultad Regional Rosario. Departamento de Ingeniería Química. Grupo de Investigación Aplicada a la Ingeniería Química (GIAIQ).

Mientras una neurona es muy pequeña en sí misma, cuando se combinan cientos, miles o millones de ellas pueden resolver problemas muy complejos. Por ejemplo el cerebro humano se compone de billones de tales neuronas.

3.6.1 Función De Entrada (*input function*): La neurona trata a muchos valores de entrada como si fueran uno solo; esto recibe el nombre de *entrada global*. Por lo tanto, ahora nos enfrentamos al problema de cómo se pueden combinar estas simples entradas (*ini*1, *ini*2, ...) dentro de la entrada global, *gini*. Esto se logra a través de la *función de entrada*, la cual se calcula a partir del *vector entrada*. La función de entrada puede describirse como sigue: *inputi = (ini1 • wi1)* (ini2 • wi2)*... (inin • win),* donde: * representa al operador apropiado (por ejemplo: máximo, sumatoria, productoria, etc.), *n* al número de entradas a la neurona *Ni* y *wi* al peso.

Los valores de entrada se multiplican por los pesos anteriormente ingresados a la neurona. Por consiguiente, los pesos que generalmente no están restringidos cambian la medida de influencia que tienen los valores de entrada. Es decir, que permiten que un gran valor de entrada tenga solamente una pequeña influencia, si estos son lo suficientemente pequeños.

3.6.2 Función De Activación (*activation function*): Una neurona biológica puede estar activa (excitada) o inactiva (no excitada); es decir, que tiene un "*estado de activación*". Las neuronas artificiales también tienen diferentes estados de activación; algunas de ellas solamente dos, al igual que las biológicas, pero otras pueden tomar cualquier valor dentro de un conjunto determinado.

La *función activación* calcula el estado de actividad de una neurona; transformando la entrada global (menos el umbral, Θi) en un valor (estado) de activación, cuyo rango normalmente va de (0 a 1) o de (–1 a 1). Esto es así, porque una neurona puede estar totalmente inactiva (0 o –1) o activa (1).

La función activación, es una función de la entrada global (*gini*) menos el umbral (Θi). Las funciones de activación más comúnmente utilizadas se detallan a continuación:

1) Función Lineal:

$$f(x) = \begin{cases} -1 & x \le -1/a \\ a * x & -1/a < x < 1/a \\ 1 & x \ge 1/a \end{cases}$$

$$\text{con } x = gin_i - \Theta_i, \text{ y } a > 0.$$

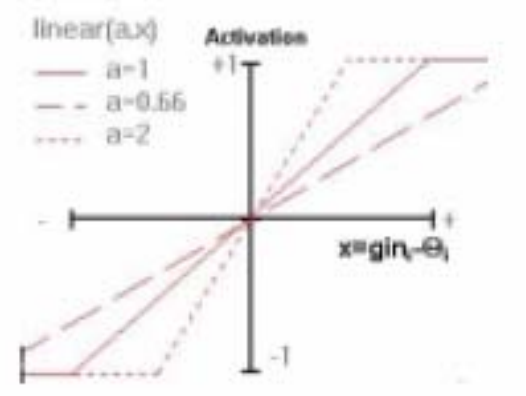


Figura 11. Función de activación lineal

Los valores de salida obtenidos por medio de esta función de activación serán: a·(*gini* - Θi), cuando el argumento de (*gini* - Θi) esté comprendido dentro del rango (-1/a, 1/a). Por encima o por debajo de esta zona se fija la salida en 1 o –1, respectivamente. Cuando a = 1 (siendo que la misma afecta la pendiente de la gráfica), la salida es igual a la entrada.

2) Función Sigmoidea:

$$f(x) = \frac{1}{1 + e^{-g\,x}}, \text{ con } x = gin_i - \Theta_i.$$

Los valores de salida que proporciona esta función están comprendidos dentro de un rango que va de 0 a 1. Al modificar el valor de *g* se ve afectada la pendiente de la función de activación.

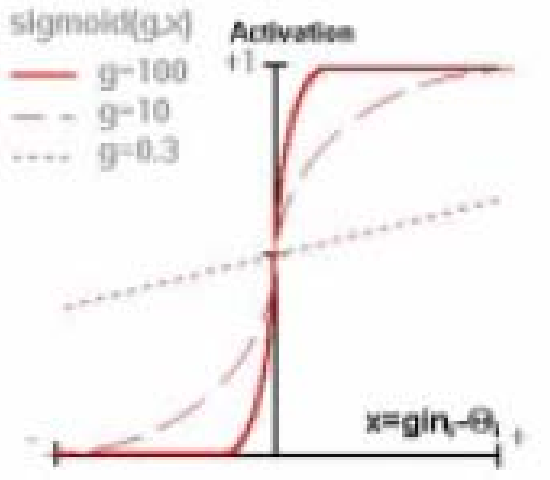


Figura 12. Función de activación Sigmoidea

3) Función Tangente hiperbólica:

$$f(x) = \frac{e^{gx} - e^{-gx}}{e^{gx} + e^{-gx}}, \text{ con } x = gin_i - \Theta_i.$$

Los valores de salida de la función tangente hiperbólica están comprendidos dentro de un rango que va de -1 a 1. Al modificar el valor de *g* se ve afectada la pendiente de la función de activación.

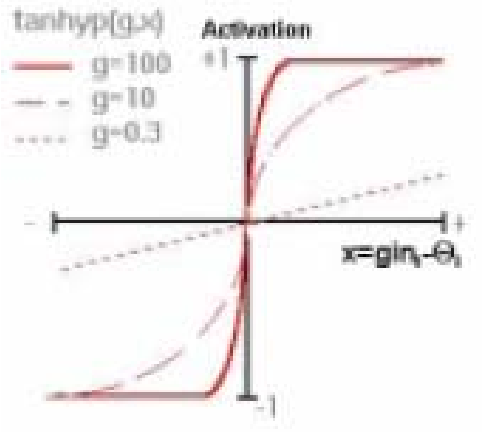


Figura 13. Función de activación Tangente hiperbólica

Para explicar porque se utilizan estas funciones de activación se suele emplear la analogía a la aceleración de un automóvil. Cuando un auto inicia su movimiento necesita una potencia elevada para comenzar a acelerar. Pero al ir tomando velocidad, este demanda un menor incremento de dicha potencia para mantener la aceleración. Al llegar a altas velocidades, nuevamente un amplio incremento en la potencia es necesario para obtener una pequeña ganancia de velocidad.

En resumen, en ambos extremos del rango de aceleración de un automóvil se demanda una mayor potencia para la aceleración que en la mitad de dicho rango.

3.6.3 Función De Salida (*output function*): El último componente que una neurona necesita es la *función de salida*. El valor resultante de esta función es la salida de la neurona *i* (*outi*); por ende, la función de salida determina que valor se transfiere a las neuronas vinculadas. Si la función de activación está por debajo de un umbral determinado, ninguna salida se pasa a la neurona subsiguiente. Normalmente, no cualquier valor es permitido como una entrada para una neurona, por lo tanto, los valores de salida están comprendidos en el rango [0, 1] o [-1, 1]. También pueden ser binarios {0, 1} o {-1, 1}.

3.7 MECANISMOS DE APRENDIZAJE [1]

Se ha visto que los datos de entrada se procesan a través de la red neuronal con el propósito de lograr una salida. También se dijo que las redes neuronales extraen generalizaciones desde un conjunto determinado de ejemplos anteriores de tales problemas de decisión. Una red neuronal debe aprender a calcular la salida correcta para cada constelación (arreglo o vector) de entrada en el conjunto de ejemplos. Este proceso de aprendizaje se denomina: *proceso de entrenamiento o acondicionamiento*. El conjunto de datos (o conjunto de ejemplos) sobre el cual este proceso se basa es, por ende, llamado: *conjunto de datos de entrenamiento*. Si la topología de la red y las diferentes funciones de cada neurona (entrada, activación y salida) no pueden cambiar durante el aprendizaje, mientras que los pesos sobre cada una de las conexiones si pueden hacerlo; el aprendizaje de una red neuronal significa: *adaptación de los pesos*.

En otras palabras el aprendizaje es el proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios que se producen durante el mismo se reducen a la destrucción, modificación y creación de conexiones entre las neuronas. En los sistemas biológicos existe una continua destrucción y creación de conexiones entre las neuronas. En los modelos de redes neuronales artificiales, la creación de una nueva conexión implica que el peso de la misma pasa a tener un valor distinto de cero. De la misma manera, una conexión se destruye cuando su peso pasa a ser cero.

Durante el proceso de aprendizaje, los pesos de las conexiones de la red sufren modificaciones, por lo tanto, se puede afirmar que este proceso ha terminado (la red ha aprendido) cuando los valores de los pesos permanecen estables ($dw_{ij}/dt = 0$). Un aspecto importante respecto al aprendizaje de las redes neuronales es el conocer cómo se modifican los valores de los pesos, es decir,

cuáles son los criterios que se siguen para cambiar el valor asignado a las conexiones cuando se pretende que la red aprenda una nueva información. Hay dos métodos de aprendizaje importantes que pueden distinguirse:

a- Aprendizaje supervisado.
b- Aprendizaje no supervisado.

Otro criterio que se puede utilizar para diferenciar las reglas de aprendizaje se basa en considerar si la red puede aprender durante su funcionamiento habitual o si el aprendizaje supone la desconexión de la red, es decir, su inhabilitación hasta que el proceso termine. En el primer caso, se trataría de un aprendizaje *on line*, mientras que el segundo es lo que se conoce como *off line*. Cuando el aprendizaje es off line, se distingue entre una *fase de aprendizaje o entrenamiento* y una *fase de operación o funcionamiento*, existiendo un conjunto de datos de entrenamiento y un conjunto de datos de test o prueba, que serán utilizados en la correspondiente fase. Además, los pesos de las conexiones permanecen fijos después que termina la etapa de entrenamiento de la red. Debido precisamente a su carácter estático, estos sistemas no presentan problemas de estabilidad en su funcionamiento.

Una generalización de la fórmula o regla para decir los cambios en los pesos es la siguiente:

$$\textit{Peso Nuevo} = \textit{Peso Viejo} + \textit{Cambio de Peso}$$

Matemáticamente esto es:

$$wij(t+1) = wij(t) + \Delta wij(t) \qquad (4)$$

donde *t* hace referencia a la etapa de aprendizaje, *wij(t*+1) al peso nuevo y *wij*(*t*) al peso viejo.

3.7.1 Aprendizaje supervisado: El aprendizaje supervisado se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada.

En este tipo de aprendizaje se suelen considerar, a su vez, tres formas de llevarlo a cabo, que dan lugar a los siguientes aprendizajes supervisados:

1) Aprendizaje por corrección de error.
2) Aprendizaje por refuerzo.
3) Aprendizaje estocástico.

3.7.1.1 Aprendizaje por corrección de error: Consiste en ajustar los pesos de las conexiones de la red en función de la diferencia entre los valores deseados y los obtenidos a la salida de la red, es decir, en función del error cometido en la salida.

Un ejemplo de este tipo de algoritmos lo constituye la *regla de aprendizaje del Perceptron*, utilizada en el entrenamiento de la red del mismo nombre que desarrolló Rosenblatt en 1958. Esta es una regla muy simple, para cada neurona en la *capa de salida* se le calcula la desviación a la salida objetivo como el error, δ. El cual luego se utiliza para cambiar los pesos sobre la conexión de la neurona precedente. El cambio de los pesos por medio de la regla de aprendizaje del Perceptron se realiza según la siguiente regla:

$$\Delta wij = \sigma * outj * (aqi - outi) \quad (5)$$

donde: a*qi* es la salida deseada/objetivo de la neurona de salida *Ni*, δi = (a*qi* – *outi*) la desviación objetivo de la neurona *Ni* y σ el aprendizaje.

La salida de la neurona *Nj* (*outj*) se utiliza, porque este valor influye en la entrada global y, por ende, en la activación y luego en la salida de la neurona *Ni*. Esto es semejante a un "efecto en cadena". Ver Figura 14

Figura 14. Influencia de la salida de la neurona Nj en la entrada de la neurona Ni.

PDF Redes neuronales. Conceptos basicos y aplicaciones. Universidad Tecnológica Nacional – Facultad Regional Rosario. Departamento de Ingeniería Química. Grupo de Investigación Aplicada a la Ingeniería Química (GIAIQ).

Otro algoritmo muy conocido y que pertenece a esta clasificación es la *regla de aprendizaje Delta* o regla del mínimo error cuadrado (LMS Error: Least Mean Squared Error), que también utiliza la desviación a la salida objetivo, pero toma en

consideración a todas las neuronas predecesoras que tiene la neurona de salida. Esto permite cuantificar el error global cometido en cualquier momento durante el proceso de entrenamiento de la red, lo cual es importante, ya que cuanto más información se tenga sobre el error cometido, más rápido se puede aprender. Luego el error calculado (δ) es igualmente repartido entre las conexiones de las neuronas predecesoras.

Por último se debe mencionar la *regla de aprendizaje de propagación hacia atrás o de backpropagation*, también conocido como regla LMS multicapa, la cual es una generalización de la regla de aprendizaje Delta. Esta es la primer regla de aprendizaje que permitió realizar cambios sobre los pesos en las conexiones de la capa oculta.

3.7.1.2 Aprendizaje por refuerzo: Se trata de un aprendizaje supervisado, más lento que el anterior, que se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado, es decir, de no indicar durante el entrenamiento exactamente la salida que se desea que proporcione la red ante una determinada entrada.

En el aprendizaje por refuerzo la función del supervisor se reduce a indicar mediante una señal de refuerzo si la salida obtenida en la red se ajusta a la deseada (éxito = +1 o fracaso = -1), y en función de ello se ajustan los pesos basándose en un mecanismo de probabilidades. Se podría decir que en este tipo de aprendizaje la función del supervisor se asemeja más a la de un crítico (que opina sobre la respuesta de la red) que a la de un maestro (que indica a la red la respuesta concreta que debe generar), como ocurría en el caso de supervisión por corrección del error.

3.7.1.3 Aprendizaje estocástico: Consiste básicamente en realizar cambios aleatorios en los valores de los pesos de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de distribuciones de probabilidad. En el aprendizaje estocástico se suele hacer una analogía en términos termodinámicos, asociando a la red neuronal con un sólido físico que tiene cierto estado energético. En el caso de la red, la energía de la misma representaría el grado de estabilidad de la red, de tal forma que el estado de mínima energía correspondería a una situación en la que los pesos de las conexiones consiguen que su funcionamiento sea el que más se ajusta al objetivo deseado.

Según lo anterior, el aprendizaje consistiría en realizar un cambio aleatorio de los valores de los pesos y determinar la energía de la red (habitualmente la función energía es una función de Liapunov). Si la energía es menor después del cambio, es decir, si el comportamiento de la red se acerca al deseado, se acepta el cambio; si, por el contrario, la energía no es menor, se aceptaría el cambio en función de una determinada y preestablecida distribución de probabilidades.

3.7.2 Aprendizaje no supervisado: Las redes con aprendizaje no supervisado (también conocido como autosupervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta.

Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada. Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado. En algunos casos, la salida representa el grado de familiaridad o similitud entre la información que se le está presentando en la entrada y las

informaciones que se le han mostrado hasta entonces (en el pasado). En otro caso, podría realizar una clusterización (clustering) o establecimiento de categorías, indicando la red a la salida a qué categoría pertenece la información presentada a la entrada, siendo la propia red quien debe encontrar las categorías apropiadas a partir de las correlaciones entre las informaciones presentadas. En cuanto a los algoritmos de aprendizaje no supervisado, en general se suelen considerar dos tipos, que dan lugar a los siguientes aprendizajes:

1) Aprendizaje hebbiano.
2) Aprendizaje competitivo y comparativo.

3.7.2.1 Aprendizaje hebbiano: Esta regla de aprendizaje es la base de muchas otras, la cual pretende medir la familiaridad o extraer características de los datos de entrada. El fundamento es una suposición bastante simple: si dos neuronas *Ni* y *Nj* toman el mismo estado simultáneamente (ambas activas o ambas inactivas), el peso de la conexión entre ambas se incrementa.

Las entradas y salidas permitidas a la neurona son: {-1, 1} o {0, 1} (neuronas binarias). Esto puede explicarse porque la regla de aprendizaje de Hebb se originó a partir de la neurona biológica clásica, que solamente puede tener dos estados: activa o inactiva.

3.7.2.2 Aprendizaje competitivo y comparativo: Se orienta a la clusterización o clasificación de los datos de entrada. Como característica principal del aprendizaje competitivo se puede decir que, si un patrón nuevo se determina que pertenece a una clase reconocida previamente, entonces la inclusión de este nuevo patrón a esta clase matizará la representación de la misma. Si el patrón de entrada se

determinó que no pertenece a ninguna de las clases reconocidas anteriormente, entonces la estructura y los pesos de la red neuronal serán ajustados para reconocer la nueva clase.

3.7.3 Cuestiones A Resolver Al Trabajar Con Una Red Neuronal : Muchos problemas aparecen cuando se trabaja con redes neuronales. Primeramente se debe analizar el dominio del problema y decidir a que clase pertenece. Luego debe decidirse si una red neuronal es adecuada para resolver dicho problema. Esto es lo que se llama: *etapa preliminar*. Concluida esta etapa, las siguientes preguntas han de responderse:

a- Origen de los datos:
_ ¿Qué datos son de importancia para la situación del problema definido?
_ ¿Qué variables son relevantes?
_ ¿De dónde pueden obtenerse los datos?

b- Preparación y codificación de los datos:
_ ¿Cómo preparar y codificar los datos?

c- Topología de la red (dependiendo parcialmente del ítem b-):
_ ¿Qué tipo de red debe escogerse?
_ ¿Cuántas capas ocultas y con cuántas neuronas son necesarias?
_ ¿Cuántas neuronas en la capa de salida (según la codificación escogida)?
_ ¿Qué tipos de neuronas deben escogerse?
_ ¿Qué regla de aprendizaje escoger?

d- Decisiones concernientes al proceso de aprendizaje:
_ ¿Cuántos ciclos de aprendizaje?
_ ¿Qué inicialización para los pesos?

3.8 PRINCIPALES TOPOLOGÍAS [1]

3.8.1 Topología de las redes neuronales: La topología o arquitectura de una red neuronal consiste en la organización y disposición de las neuronas en la misma, formando capas o agrupaciones de neuronas más o menos alejadas de la entrada y salida de dicha red. En este sentido, los parámetros fundamentales de la red son: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas.

3.8.2 Redes monocapa: En las redes monocapa, se establecen conexiones entre las neuronas que pertenecen a la única capa que constituye la red. Las redes monocapas se utilizan generalmente en tareas relacionadas con lo que se conoce como autoasociación (regenerar información de entrada que se presenta a la red de forma incompleta o distorsionada).

3.8.3 Redes multicapa: Las redes multicapas son aquellas que disponen de un conjunto de neuronas agrupadas en varios (2, 3, etc.) niveles o capas. En estos casos, una forma para distinguir la capa a la que pertenece una neurona, consistiría en fijarse en el origen de las señales que recibe a la entrada y el destino de la señal de salida. Normalmente, todas las neuronas de una capa reciben señales de entrada desde otra capa anterior (la cual está más cerca a la entrada de la red), y envían señales de salida a una capa posterior (que está más cerca a la salida de la red). A estas conexiones se las denomina *conexiones hacia adelante o feedforward*. Sin embargo, en un gran número de estas redes también existe la posibilidad de conectar la salida de las neuronas de capas posteriores a la entrada de capas anteriores; a estas conexiones se las denomina *conexiones hacia atrás o feedback*.

Estas dos posibilidades permiten distinguir entre dos tipos de redes con múltiples capas: las redes con conexiones hacia adelante o *redes feedforward*, y las redes que disponen de conexiones tanto hacia adelante como hacia atrás o *redes feedforward/feedback*.

3.8.4 Conexión entre neuronas: La conectividad entre los nodos de una red neuronal está relacionada con la forma en que las salidas de las neuronas están canalizadas para convertirse en entradas de otras neuronas. La señal de salida de un nodo puede ser una entrada de otro elemento de proceso, o incluso ser una entrada de sí mismo (*conexión autorrecurrente*). Cuando ninguna salida de las neuronas es entrada de neuronas del mismo nivel o de niveles precedentes, la red se describe como de *conexión hacia delante*. Cuando las salidas pueden ser conectadas como entradas de neuronas de niveles previos o del mismo nivel, incluyéndose ellas mismas, la red es de *conexión hacia atrás*.

Las redes de propagación hacia atrás que tienen lazos cerrados son llamadas: *sistemas recurrentes*.

3.8.5 Redes de propagación hacia atrás (*backpropagation*): El nombre de backpropagation resulta de la forma en que el error es propagado hacia atrás a través de la red neuronal, en otras palabras el error se propaga hacia atrás desde la capa de salida. Esto permite que los pesos sobre las conexiones de las neuronas ubicadas en las capas ocultas cambien durante el entrenamiento.

El cambio de los pesos en las conexiones de las neuronas además de influir sobre la entrada global, influye en la activación y por consiguiente en la salida de una neurona. Por lo tanto, es de gran utilidad considerar las variaciones de la función activación al modificarse el valor de los pesos. Esto se llama *sensibilidad* de la función activación, de acuerdo al cambio en los pesos.

3.8.6 Estructura de la Red Hopfield: La Red Hopfield es recurrente y completamente interconectada. Funciona como una memoria asociativa no lineal, que puede almacenar internamente patrones presentados de forma incompleta o con ruido. De esta forma puede ser usada como una herramienta de optimización; también se han utilizado en aplicaciones de segmentación y restauración de imágenes y optimización combinatoria.

La Red Hopfield consta de un número de neuronas simétrica e íntegramente conectadas, como ya se mencionó anteriormente. Esto significa que si existe una conexión desde la neurona N_i a la neurona N_j, también existe la conexión desde N_j a N_i; ambas exhibendo el mismo peso ($w_{ij} = w_{ji}$). Vale aclarar que la conexión de una neurona con sí misma no está permitida.

El conjunto permitido de valores de entrada y salida es {0, 1} (o en alguna oportunidad {-1, 1}); o sea, es un conjunto binario. De esta manera todas las neuronas en una Red Hopfield son binarias, tomando solamente uno de los dos estados posibles: activo (1) o inactivo (-1 o 0).

Las Redes Hopfield se emplean para reconocer patrones. Después que el aprendizaje haya llegado a su fin, la red neuronal debe ser capaz de dar una salida correcta para cada patrón de entrada dado, aun cuando este sea ruidoso.

La clave del aprendizaje Hopfield es que si un patrón que tiene que ser aprendido se conoce, los pesos sobre cada conexión de la red neuronal pueden ser calculados. En esta circunstancia, solamente el estado de las neuronas cambia durante el proceso de aprendizaje. Este cálculo garantiza que cada patrón aprendido corresponda a un mínimo de la función energía. Es importante entender que para este tipo de redes la definición de aprendizaje es diferente al dado anteriormente, donde aprendizaje significaba simplemente la adaptación de

los pesos. En una Red Hopfield los pesos se pueden calcular y se mantienen fijos durante el aprendizaje de los patrones. Solamente cambia el estado de las neuronas.

Para calcular el peso de una conexión cualquiera, w_{ij} (y por simetría para la conexión w_{ji}), en una Red Hopfield se utiliza la siguiente ecuación:

$$w_{ij} = \sum_{q=1}^{Q} (2 * e_{qi} - 1) * (2 * e_{qj} - 1), \quad i <> j \qquad (6)$$

siendo Q el número de patrones y e_{qi} la entrada a la neurona N_i. Generalmente es aconsejable trabajar con esta ecuación cuando los patrones que se han de aprender no son muy semejantes unos a otros, y si el número de ceros y unos son similares para todos los patrones. Con respecto al número de ceros y unos, el *umbral* de cada neurona puede utilizarse para regular esto, distinguiéndose así dos casos posibles:

a- Si hay más 0s que 1s el umbral tiene que disminuirse, porque que las neuronas tienen una probabilidad más alta para hacerse inactivas que para hacerse activas.

b- Si hay mas 1s que 0s el umbral tiene que incrementarse, porque las neuronas tienen una probabilidad más alta para hacerse activas que para hacerse inactivas.

3.8.7 Simulated Annealing aplicada a una Red Hopfield: En muchos problemas, la tarea no es justamente encontrar cualquier mínimo local, sino la de encontrar el óptimo global. Lo que significa que para una entrada determinada se debe encontrar una salida que resulte en un mínimo de la función energía. Utilizando una Red Hopfield, se encuentra que un mínimo yace cerca del vector de

entrada dado, porque la energía decrece paso a paso. El cual puede ser un mínimo local.

En una Red Hopfield todos los mínimos locales son un estado estable. Un problema similar se origina en termodinámica durante el proceso de cristalización. Durante un enfriamiento lento, el cristal crece con una estructura casi perfecta, ya que cada átomo tiene bastante tiempo para saltar a otra posición dentro de la cuadrícula, de tal forma que la energía total del cristal decrezca. Para realizar dicho salto se necesita energía, es decir, que si el cristal tiene la energía suficiente (si su temperatura es aun bastante alta), todos los átomos disponen de una chance para cambiar su posición. Pero para permitir que esto ocurra la energía de un átomo tiene que incrementarse por un corto tiempo, de lo contrario el átomo descansaría en su vieja posición.

Tener una chance se puede interpretar como *"hay una probabilidad"*. Esta probabilidad depende de la activación que un átomo muestra a una determinada temperatura y tiempo del sistema. Utilizando esta técnica donde el cristal comienza a una temperatura elevada y que luego decrece paso a paso, se les da a los átomos una posibilidad de cambiar sus estados independientemente de la activación, por medio de un incremento en la energía de los mismos de un paso a otro. Cuando la temperatura se reduce, la cuadrícula vibra menos, y el sistema (la cuadrícula) alcanza un estado estable; haciéndose gradualmente más dificultoso para un átomo encontrar la energía para saltar a otra posición. Esta es la idea de Simulated Annealing, que luego se aplica a la Red Hopfield cuando se intenta encontrar un óptimo global.

3.8.8 Asociaciones entre la información de entrada y salida: Ya se sabe que las redes neuronales son sistemas que almacenan cierta información *aprendida*. Esta información se registra de forma distribuida en los pesos asociados a las

conexiones entre neuronas. Por tanto, puede imaginarse una red como cierto tipo de memoria que almacena datos de forma estable, datos que se grabarán en dicha memoria como consecuencia del aprendizaje de la red y que podrán ser leídos a la salida como respuesta a cierta información de entrada, comportándose entonces la red como lo que habitualmente se conoce por memoria asociativa: cuando se aplica un estímulo (dato de entrada) la red responde con una salida asociada a dicha información de entrada.

Existen dos formas primarias de realizar esta asociación entre entradas/salidas que se corresponden con la naturaleza de la información almacenada en la red. Una primera sería la denominada *heteroasociación,* que se refiere al caso en el que la red aprende parejas de datos [$(A_1,B_1), (A_2, B_2),... (A_N,B_N)$], de tal forma que cuando se presente cierta información de entrada A_i, deberá responder generando la correspondiente salida asociada B_i. La segunda se conoce como *autoasociación,* donde la red *aprende* ciertas informaciones $A_1, A_2, ..., A_N$; de tal forma que cuando se le presenta una información de entrada realizará una autocorrelación, respondiendo con uno de los datos almacenados, el más parecido al de entrada.

Estos dos mecanismos de asociación dan lugar a dos tipos de redes neuronales: las redes heteroasociativas y las autoasociativas. Una *red heteroasociativa* podría considerarse como aquella que computa cierta función, que en la mayoría de los casos no podría expresarse analíticamente, entre un conjunto de entradas y un conjunto de salidas, correspondiendo a cada posible entrada una determinada salida.

Por otra parte, una *red autoasociativa* es una red cuya principal misión es reconstruir una determinada información de entrada que se presente incompleta o distorsionada (le asocia el dato almacenado más parecido). En realidad estos dos

tipos de modelos de redes no son diferentes en principio, porque una red heteroasiciativa puede siempre ser reducida a una asociativa mediante la concatenación de una información de entrada y su salida (respuesta) asociada, para obtener la información de entrada de la red autoasociativa equivalente.

También puede conseguirse que una red autoasociativa se comporte como heteroasociativa, simplemente presentando, como entrada parcial de la autoasociativa, la información de entrada para la heteroasociativa y haciendo que la red complete la información para producir lo que sería la salida de la red heteroasociativa equivalente.

3.8.9 Redes heteroasociativas: Las redes heteroasociativas, al asociar informaciones de entrada con diferentes informaciones de salida, precisan al menos de dos capas, una para captar y retener la información de entrada y otra para mantener la salida con la información asociada. Si esto no fuese así, se perdería la información inicial al obtenerse el dato asociado {3}, lo cual no debe ocurrir, ya que en el proceso de obtención de la salida se puede necesitar acceder varias veces a esta información que, por tanto, deberá permanecer en la capa de entrada.

En cuanto a su conectividad, pueden ser del tipo con conexión hacia adelante (*o feedforward*) o con conexión hacia atrás (*feddforward/feedback*), o bien con conexiones laterales.

3.8.9.1 Redes autoasociativas: Una red autoasociativa asocia una información de entrada con el ejemplar más parecido de los almacenados *conocidos* por la red. Estos tipos de redes pueden implementarse con una sola capa de neuronas. Esta capa comenzará reteniendo la información inicial a la entrada, y terminará representando la información autoasociada. Si se quiere mantener la información

de entrada y salida, se deberían añadir capas adicionales, sin embargo, la funcionalidad de la red puede conseguirse en una sola capa.

En cuanto a su conectividad, existen de conexiones laterales y, en algunos casos, conexiones autorrecurrentes[17].

[17] Autorrecurrentes: salida de una neurona aplicada a su propia entrada

CAPITULO 4

DISEÑO DEL SISTEMA

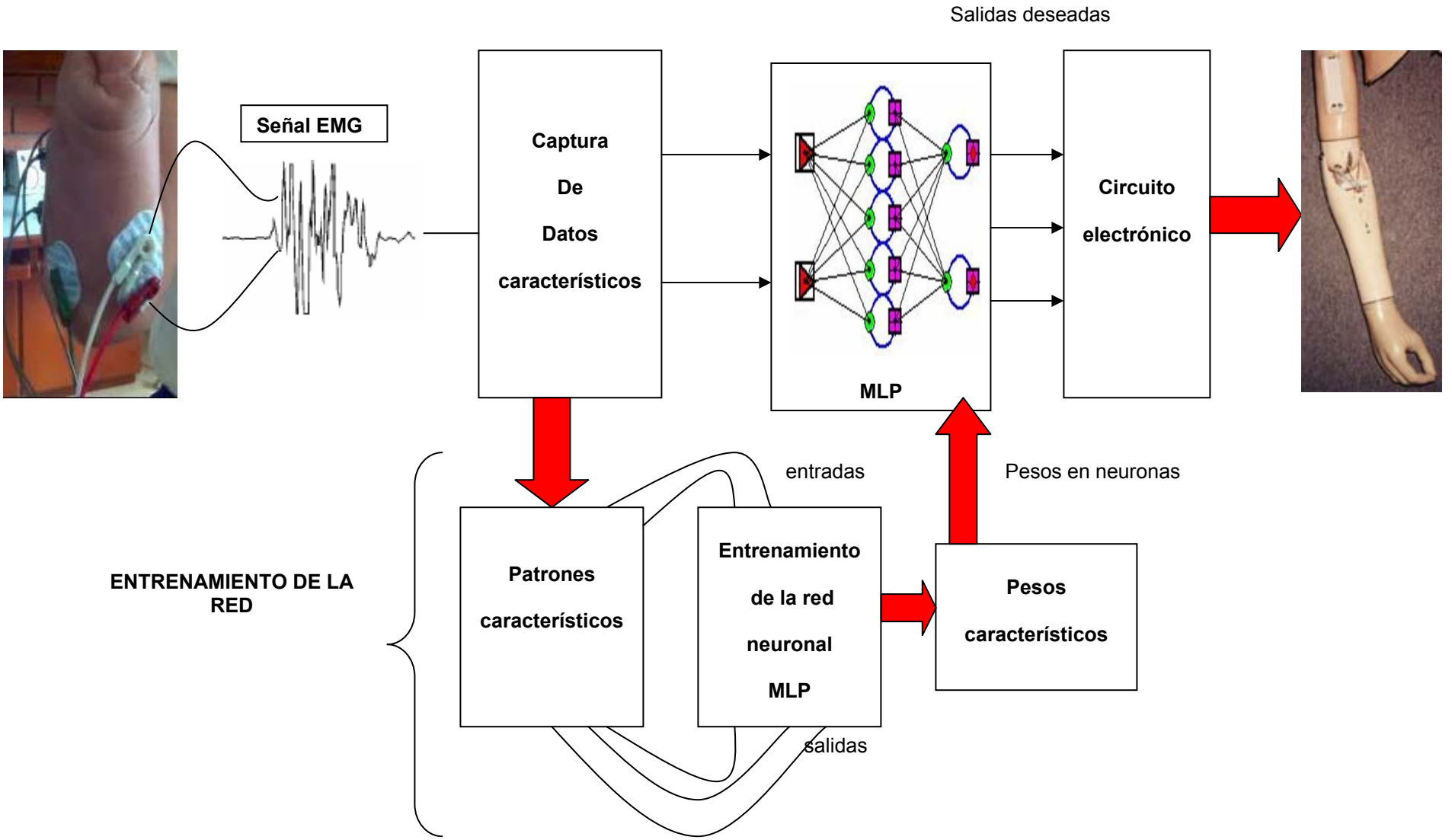
Para el diseño del sistema se contó con la colaboración de diferentes pacientes para realizar el análisis de sus señales electromiográficas dependiendo su grado de amputación y tipo de trauma causante de dicha discapacidad.

Con dichos análisis se realizó un estudio previo con la tarjeta de acondicionamiento Bionix.

Para realizar el diseño inteligente implementando redes neuronales se necesito realizar un segundo acondicionamiento a la señal para poder discretizar de manera más exacta los datos de entrada a la red que se va a implementar.

De esta forma se implementa el algoritmo diseñado, entrenando, ejecutando y probando la red en un paciente determinado.

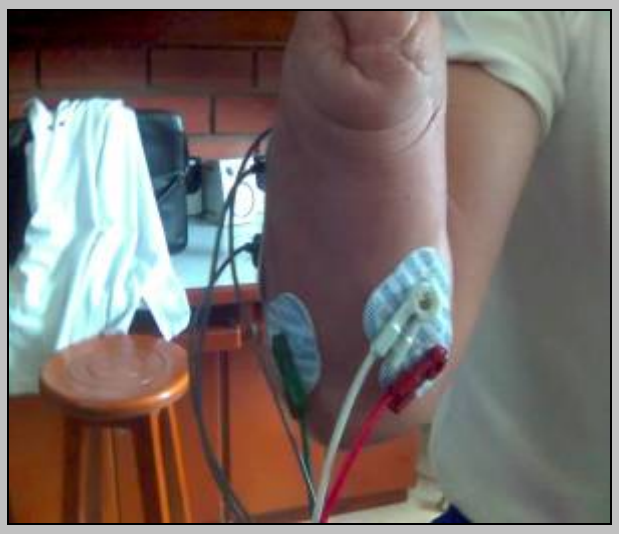
Figura 1. Diagrama del diseño control inteligente



4.1 SEÑAL EMG

A continuación se muestra el anales de las señales electromiográficas realizado a uno de los pacientes discapacitados.

Tabla 1. Anamnesis del paciente elegido

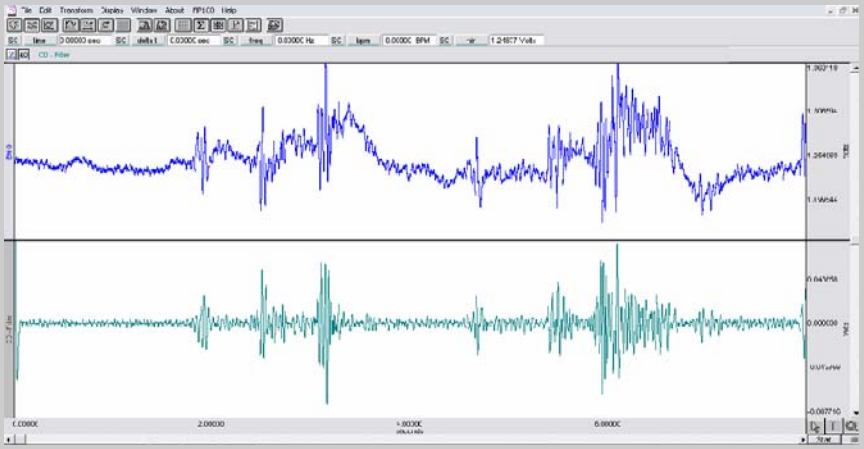
ANAMNESIS DEL PACIENTE	
	
Nombre del paciente	L. A. L. P.
Edad	47 años
Identificación	2.100.079 de Guadalupe /Santander
Ocupación	Policía retirado
Escolaridad	Bachiller
Dominancia	Miembro superior derecho
Mecanismo de amputación	Amputación traumática en un atentado con artefacto explosivo
Nivel de amputación	Tercio distal del antebrazo del miembro superior derecho
Evolución	18 años
Usa prótesis?	Si
Tipo de prótesis	Mecánica - Gancho
Tiempo de uso	17 años

4.1.1 Sesión toma de señales y análisis de los resultados: A continuación se muestran las señales adquiridas en el paciente, tomadas con un solo set de electrodos colocados en el grupo muscular flexor para medir el comportamiento agonista y antagonista.

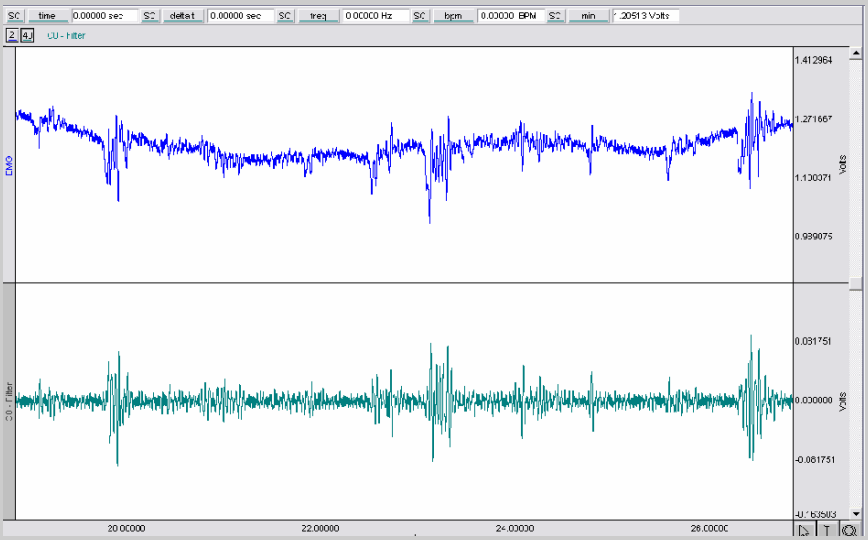
Figura 15. Toma de señales al paciente, realizada con el equipo BIOPAC

REGISTRO DE EXTENSIÓN Y FLEXIÓN EN 3 PASOS (MÍNIMO – MEDIO - MÁXIMO)
FECHA: ABRIL 15 DE 2006

A



B



El paciente realizó flexión (A) y extensión (B) con tres fuerzas cada una levemente más fuerte que la anterior. El objetivo de este ejercicio era encontrar el modelo de control más eficiente del mecanismo.

Analizamos el comportamiento de las señales del paciente obtenidas de los músculos flexores y extensores, donde se le pedía realizar tres tipos de esfuerzos diferentes para asir un objeto y luego soltarlo.

La mejor respuesta obtenida es notable en la imagen A donde las tres fuerzas se distinguen fácilmente. Mientras que en la imagen B es más difícil encontrar los tres tipos de esfuerzos diferentes y tan solo dos se alcanzan a diferenciar un poco del resto de la señal.

Observamos que con el corto tiempo de entrenamiento, el paciente es capaz de realizar diferentes fuerzas las cuales pueden ser muy útiles a la hora de controlar velocidad o fuerza de agarre en la prótesis.

4.1.2 Configuración de electrodos: A continuación se muestra la adquisición de señales realizada en el paciente, con dos sets de electrodos ubicados uno sobre cada punto motor de los músculos flexores y extensores hallados en el antebrazo.

4.1.2.1 Respuesta de Adquisición con una configuración determinada de electrodos: Para la toma de estas señales el paciente realizó continuamente movimientos de flexión y extensión. Orientado por la fisioterapeuta, se realizaban ejercicios de articulación de la muñeca con el brazo sano, y de esta manera se imaginaba el ejercicio en el brazo amputado, y así se generaban las señales observadas en la imagen anterior. La señal de mayor magnitud es la señal de

extensión, sin embargo la de flexión aparece pero de una menor magnitud. Esta señal de extensión daría la orden de apertura en la prótesis.

Figura 16. Configuración de electrodos

AMPUTACIÓN

Tipo: Transradial tercio distal
Canal 1 Músculos extensores
Canal2 Músculos flexores

Separación Electrodos: 1.5 CM

Configuración Electrodos

Rojo + Máximo Potencial de acción
Negro - Potencial de Acción medio
Verde (N) Potencial de Acción Neutral

Figura 17. Respuesta a la configuración de electrodos

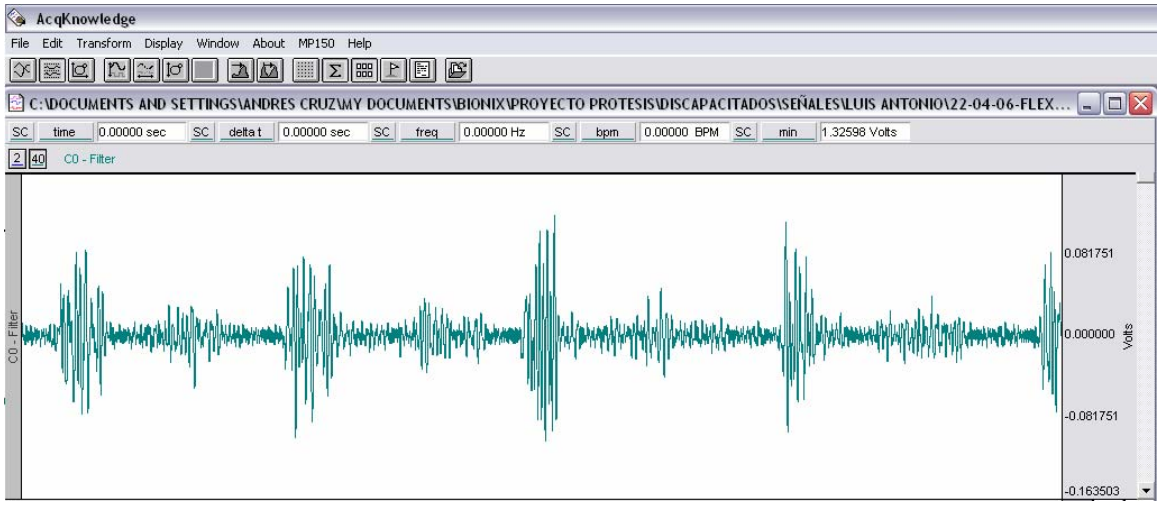
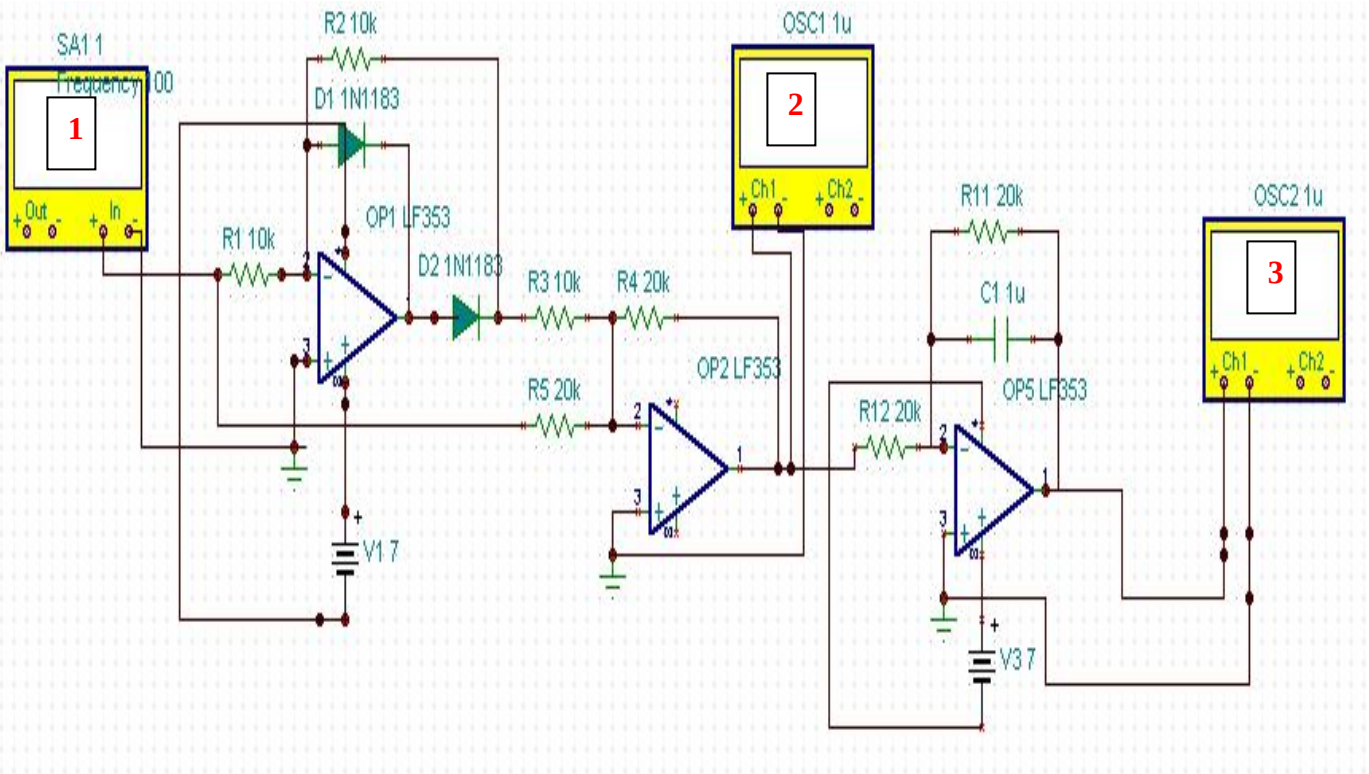


Figura tomada desde el equipo BIOPAC y su software de tratamiento digital.

4.2 ACONDICIONAMIENTO DE LA SEÑAL EMG

Para obtener mejores resultados en el entrenamiento y ejecución en la red neuronal a elegir se requiere de un segundo acondicionamiento, adicional al de la tarjeta de Bionix, donde se trata la señal inicial como tal pasando por una amplificación y un filtrado que permite ver y estudiar la señal como se pudo ver en las figuras anteriores.

Figura 18. Circuito esquemático del diseño de nuevo acondicionamiento



Diseño realizado en software de simulación llamado tina

Figura 19. Captura de señal tarjeta Bionix para el acondicionamiento (1)

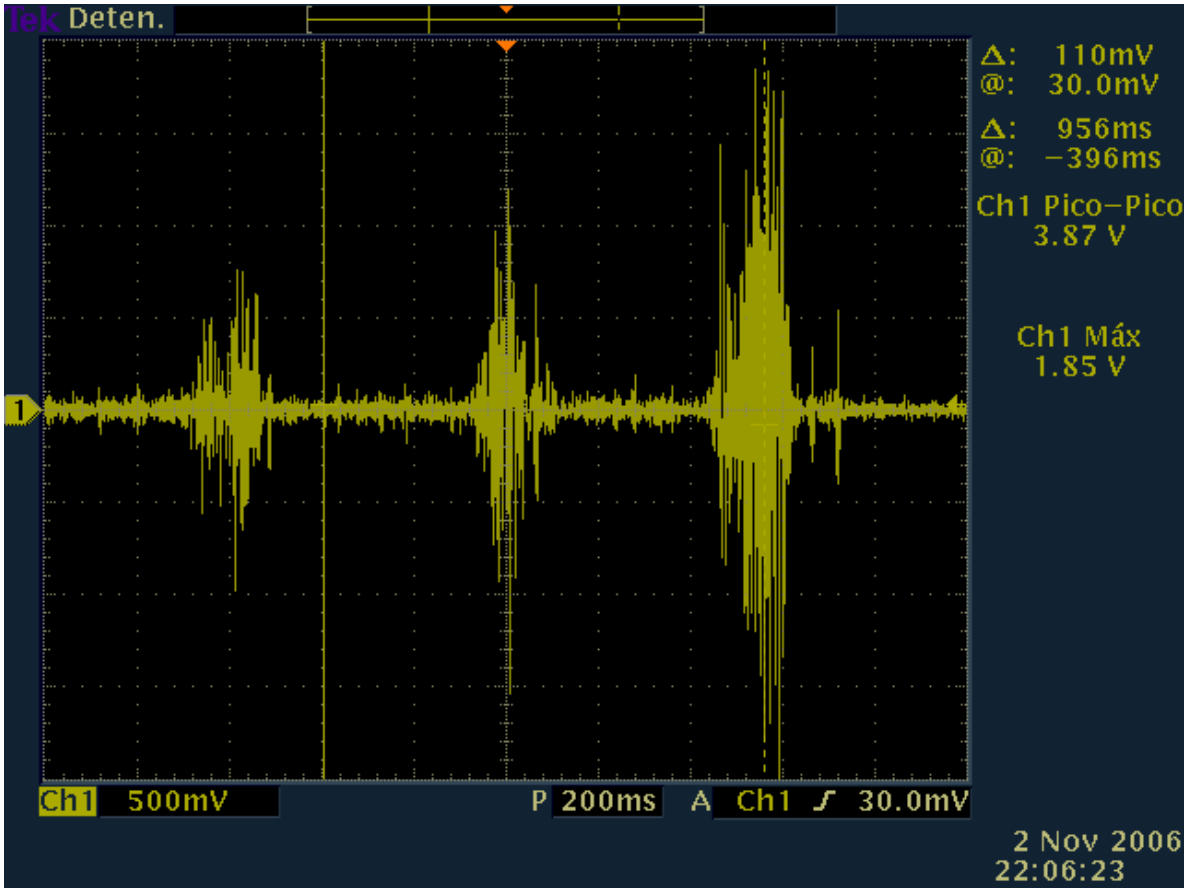


Figura obtenida de la prueba de la tarjeta de acondicionamiento de señal EMG de Bionix

4.2.1 Tratamiento de la señal de entrada: Se diseñó un circuito de rectificación de la señal con amplificadores operacionales LF353 fabricado por Fairchild[18].

4.2.1.1 Rectificador de onda completa de alta precisión: La señal de entrada ingresa al circuito de amplificadores para obtener una alta impedancia.

[18] Empresa lider en semiconductores. www.fairchildsemi.com/

Figura 20. Resultado de la rectificación (2)

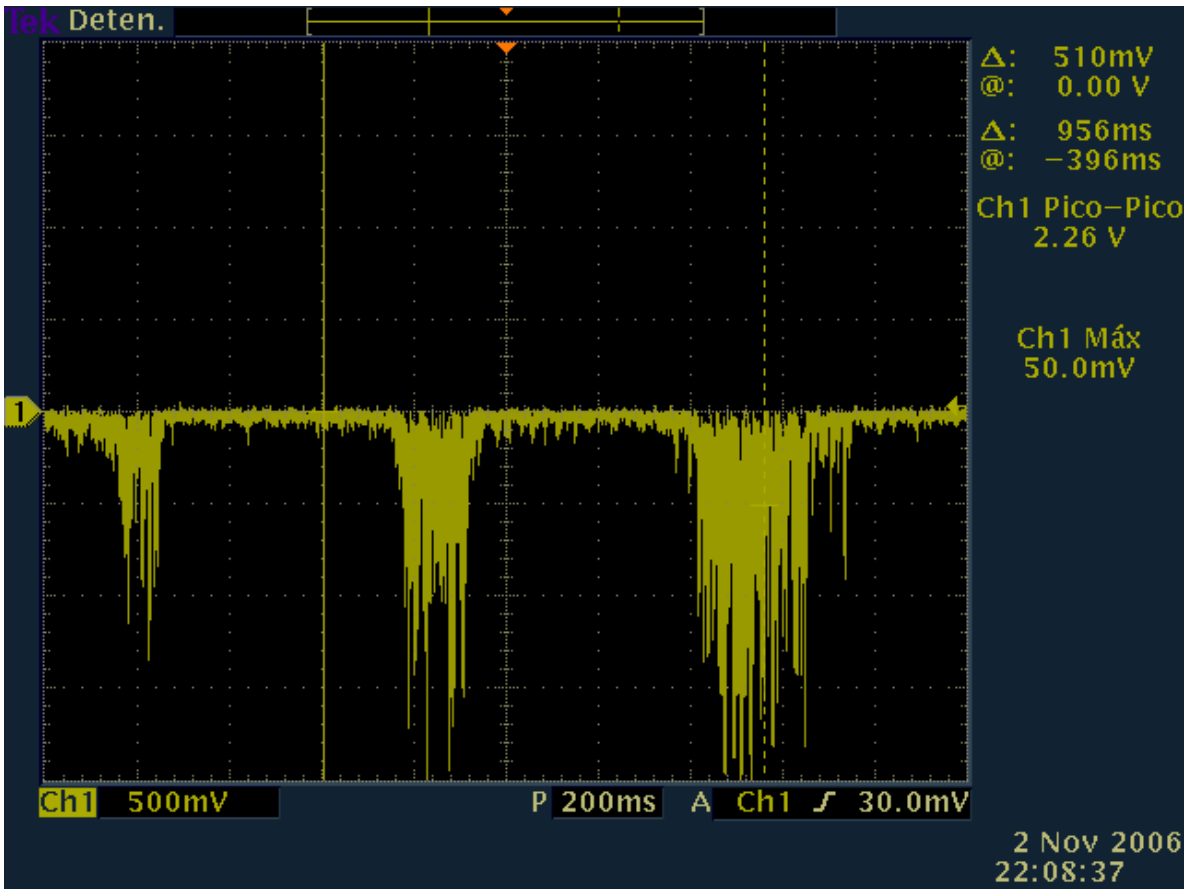
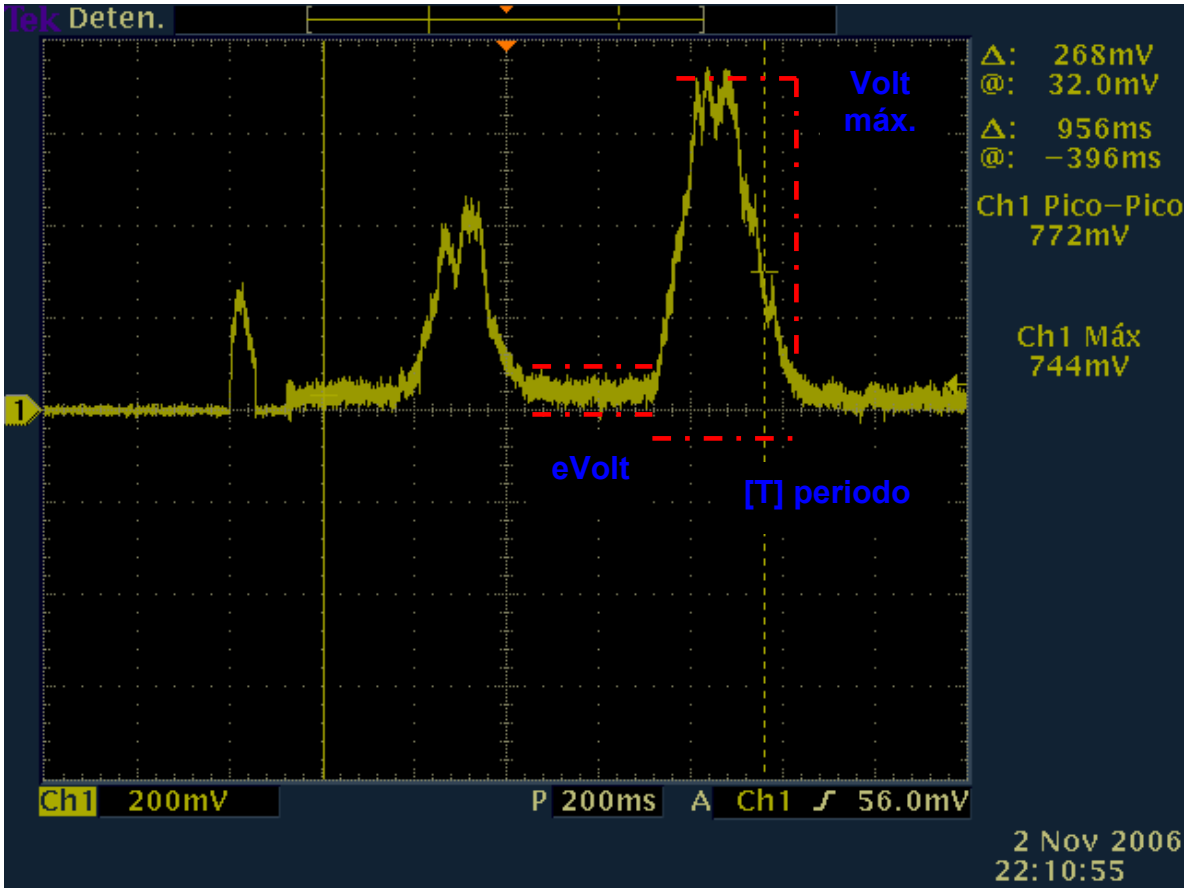


Figura obtenida del nuevo diseño de acondicionamiento para la entrada a la red neuronal

4.2.1.2 Circuito integrador: Proporciona el área bajo la curva de la señal ya rectificada .

Figura 21. Resultado obtenido de la integración de la señal



Obtenida del ingreso de la señal rectificada al circuito integrador

4.3 CARACTERIZACIÓN DE LA SEÑAL

Para realizar el muestreo de la señal, se necesitan tomar datos patrones de la señal que entra a la red neuronal.

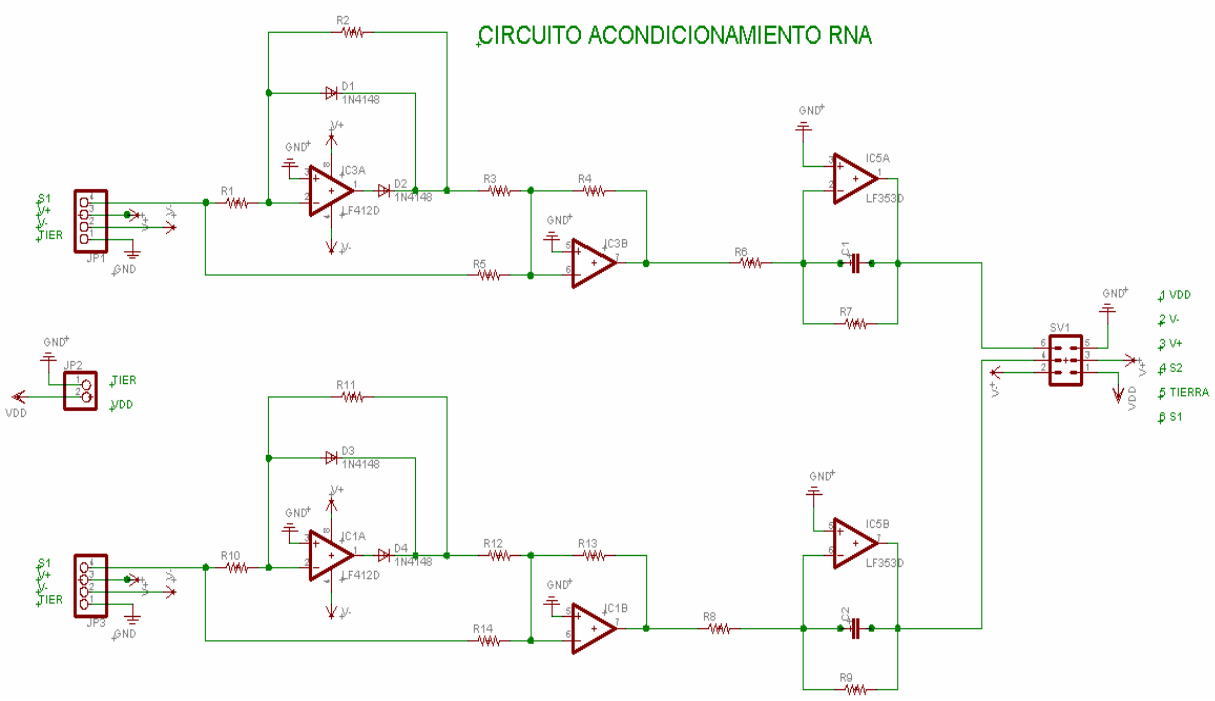
- *Amplitud máximo:* Voltaje limite que se toma para muestrear la señal
- *Amplitud offset:* Voltaje sobre el cual se encuentra montada la señal o referenciada.

- *Amplitud Error de voltaje:* Voltaje que se toma en cuenta del rizado del ruido para iniciar los patrones de entrada por encima de este.
- *Periodo:* Tiempo en el cual se captura la señal y de esta manera se muestrea.

Tabla 2. Valores patrones de la señal integrada

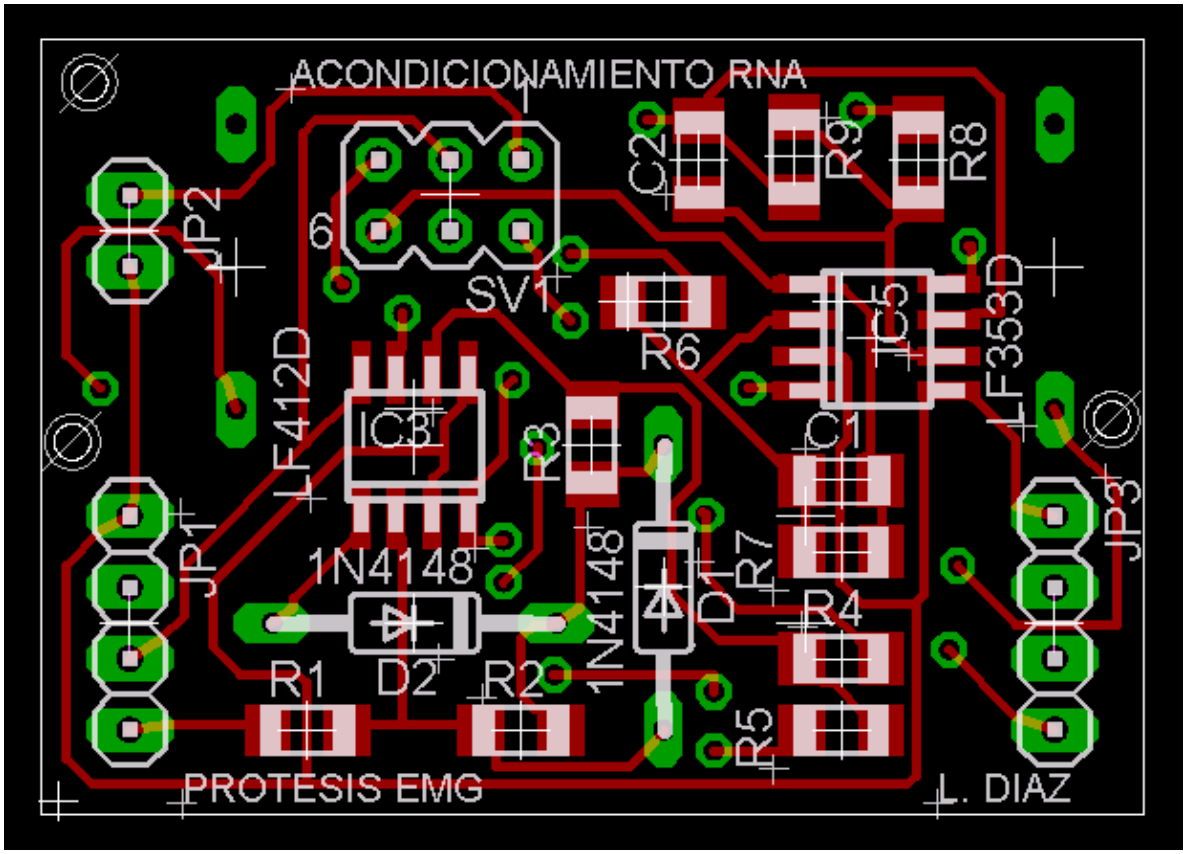
DATO	VALOR
Valor máx. de amplitud	3 Volts
Amplitud offset	1.5 Volts
Amplitud error de voltaje	0.140 Volts
Periodo de la señal	550 ms

Figura 22. Esquemático circuito acondicionamiento



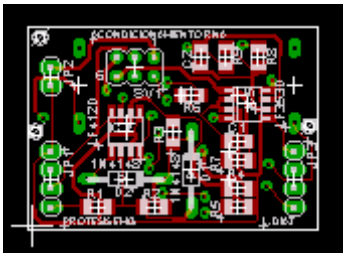
Esquemático realizado en Eagle

Figura 23. Diseño de la Board a imprimir



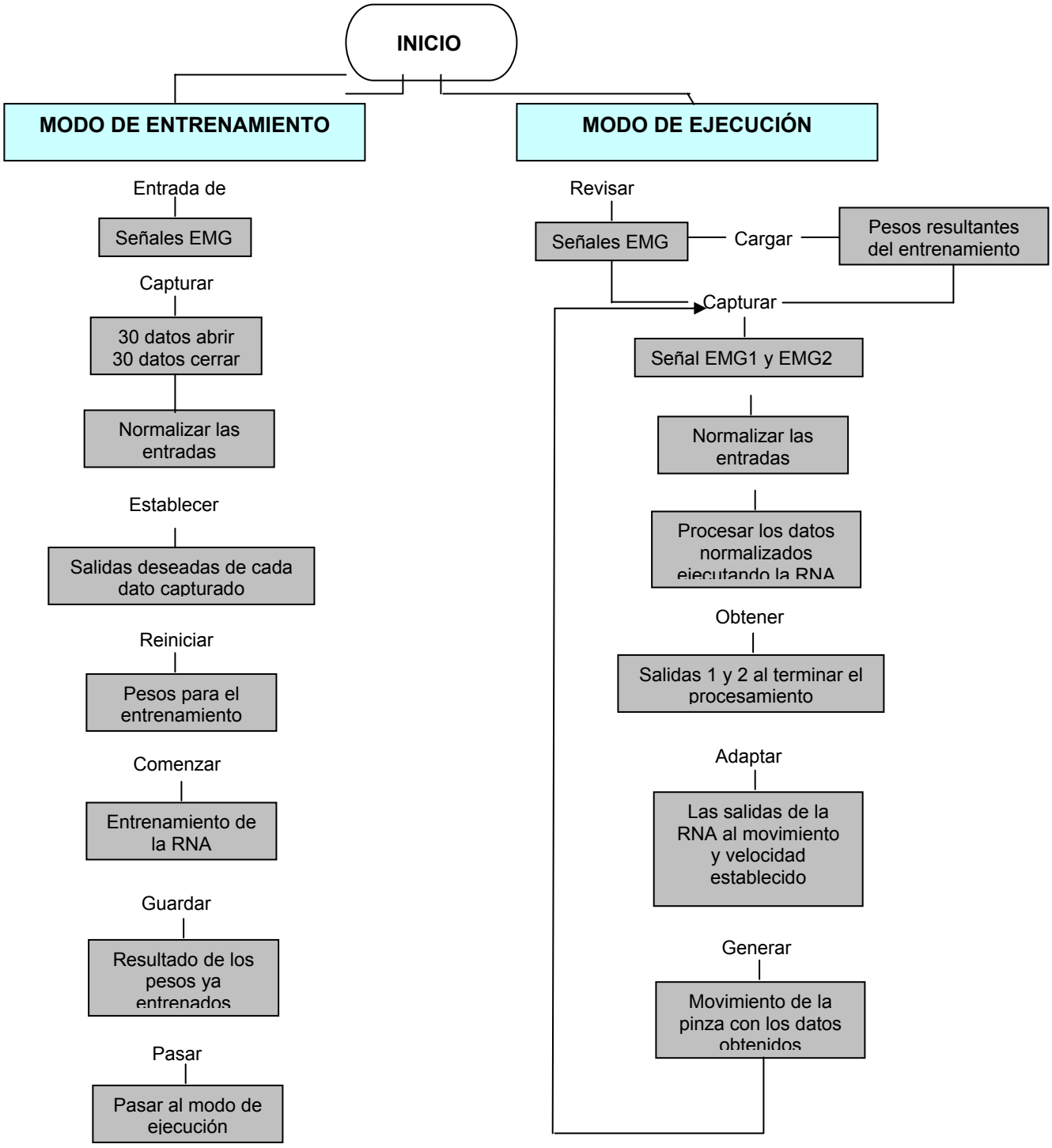
Board realizado en Eagle

Figura 24. Diseño de la Board a en tamaño real



Diseñado con elementos de superficie, Tamaño 3.6 cm X 4.9 cm

4.4 DIAGRAMA GENERAL DEL ALGORITMO DE CONTROL



4.5 SELECCIÓN DE TIPO DE RED

Debido a parámetros no lineales de la información adquirida, la red Perceptron no puede ser tomada como método de procesamiento en este sistema, debido a que esta solo es utilizada para solucionar problemas sencillos donde los patrones de entrenamiento son linealmente separables en un hiperplano. Dada la naturaleza de los datos y la imposible separación lineal se escoge la red Percetron Multicapa (MLP) con algoritmo de aprendizaje Backpropagation que es capaz de soportar la complejidad de este problema.

La red *Backpropagation* es una regla de aprendizaje que se puede aplicar en modelos de redes con más de dos capas de neuronas, su aprendizaje es supervisado, el cual necesita conocer cual es la salida esperada asociada a cada una de las entradas que actualiza pesos.

La importancia de esta red radica en su capacidad de auto-adaptar los pesos de las neuronas de las capas intermedias para aprender la relación que existe entre un conjunto de patrones dados como ejemplo y sus salidas correspondientes. Para poder aplicar esa misma relación, después del entrenamiento, a nuevos vectores de entrada con ruidos o incompletas, dando una salida activa si la nueva entrada es parecida a las presentadas durante el aprendizaje. Esta característica exige a los sistemas la capacidad de generalización, entendida como las la facilidad de dar salidas satisfactorias a entradas que el sistema no ha visto nunca en su fase de entrenamiento[19].

[19] Hilera, José R. y Martínez, Víctor J. Redes neuronales Artificiales. RAMA. Madrid, 2000.

4.5.1 Estructura de la red: Durante el estudio de la red se defino los parámetros estructurales de la red MPL. Hay datos que se dejan tanto constantes como variables, es constante cuando se define por completo el diseño de la red, el numero de capas, el numero de neuronas en la capa de entrada y de salida, ya que de esto no depende la variación del entrenamiento. El número de neuronas en la capa oculta es variable al igual que el numero de épocas y su rata de aprendizaje pues de estos datos depende un buen resultado.

- *Numero de Capas*: Es el parámetro general en el cual se divide la red. Capa de entrada, capas ocultas y capa de salida. Para este caso específico se tomaron en cuenta 3 capas; la de entrada, una capa oculta y la capa de salida.

- *Numero de neuronas*: Son los elementos necesarios para una capa neuronal que definen la capacidad de aprendizaje y/o la capacidad de procesamiento en cada una de las capas.
    - Capa de entrada: 2 neuronas; Señal EMG1 y Señal EMG2
    - Capa oculta: 5 o 8 neuronas; es un numero suficiente para el procesamiento, de esta manera la red esta aprendiendo y no requiere de mucho tiempo en este proceso. Es decir a mayor numero de neuronas en las o en la capa oculta mayor aprendizaje y mayor tiempo de procesamiento. Cuando se exagera en el numero de estas, la red puede sobre entrenarse y demorarse horas realizando un entrenamiento, esto se define de acuerdo a la complejidad del problema y sus datos, además es un proceso que se realiza a prueba y error, buscando obtener la mejor respuesta.
    - Capa de salida: 2 neuronas; una que maneja la dirección del motor (abrir o cerrar la pinza) y otra que maneja su velocidad.

- *Épocas o Número de iteraciones*: El numero de veces que pasan todos datos de entrenamiento por la red neuronal al momento de su aprendizaje. Se va a definir 50 épocas, para así no demorar mucho el procesamiento.

- *Rata de aprendizaje*: Es una constante que la controla el tamaño del cambio de los pesos en cada iteración. Se deben evitar dos extremos: un ritmo de aprendizaje demasiado pequeño puede ocasionar una disminución importante en la velocidad de convergencia y la posibilidad de acabar atrapado en un mínimo local; en cambio, un ritmo de aprendizaje demasiado grande puede conducir a inestabilidades en la función de error, lo cual evitará que se produzca la convergencia debido a que se darán saltos en torno al mínimo sin alcanzarlo. Por tanto, se recomienda elegir un ritmo de aprendizaje lo más grande posible sin que provoque grandes oscilaciones. En general, el valor de la tasa de aprendizaje suele estar comprendida entre 0.05 y 0.5 [12].

Figura 25. Estructura de una neurona artificial

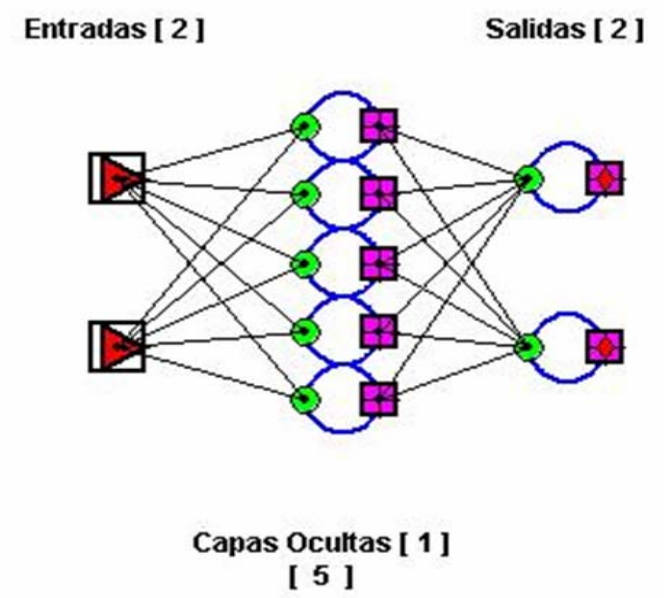


Figura realizada en un programa interactivo hecho en matlab que permite dibujar la estructura de la red.

4.6 ENTRENAMIENTO DE LA RED

El entrenamiento de la red neuronal es el proceso más importante de ella. Para realizar un algoritmo de entrenamiento se debe seguir un proceso matemático que me permite modificar y procesar los pesos que se le dan a cada una de las neuronas de entrada, salida y capa oculta.

En la etapa de aprendizaje, el objetivo que se persigue es hacer mínima la discrepancia o error entre la salida obtenida por la red y la salida deseada por el usuario ante la presentación de un conjunto de patrones denominado grupo de entrenamiento. Por este motivo, se dice que el aprendizaje en las redes *backpropagation* es de tipo supervisado, debido a el usuario (o supervisor) determina la salida deseada ante la presentación de un determinado patrón de entrada [13].

La función de error o error cuadrático medio es quien determina por su valor si la red esta aprendiendo, este valor esta dado por la siguiente ecuación[13]:

$$E_p = \frac{1}{2}\sum_{k=1}^{M}(d_{pk} - y_{pk})^2 \quad (7)$$

donde d_{pk} es la salida deseada para la neurona de salida k ante la presentación del patrón p. A partir de esta expresión se puede obtener una medida general de error mediante:

$$E = \sum_{p=1}^{P} E_p \quad (8)$$

[13]La base matemática del algoritmo *backpropagation* para la modificación de los pesos es la técnica conocida como gradiente decreciente[20].

Teniendo en cuenta que Ep es función de todos los pesos de la red, el gradiente de Ep es un vector igual a la derivada parcial de Ep respecto a cada uno de los pesos. El gradiente toma la dirección que determina el incremento más rápido en el error, mientras que la dirección opuesta, es decir, la dirección negativa, determina el decremento más rápido en el error. Por tanto, el error puede reducirse ajustando cada peso en la dirección:

$$-\sum_{p=1}^{P}\frac{\partial E_p}{\partial w_{ji}} \qquad (9)$$

De acuerdo al algoritmo que se sigue tomamos en cuenta las formulas para la modificación de los pesos tomando en cuenta una tasa de aprendizaje:

$$\Delta v_{kj}(n+1) = \eta\sum_{p=1}^{P}\delta_{pk}b_{pj} \qquad (10)$$

Donde,

$$\delta_{pk} = (d_{pk} - y_{pk})f'(net_{pk}) \qquad (11)$$

y n indica la iteración. Cuando se trata del peso de una neurona oculta:

[20] Rumelhart, Hinton y Williams, 1986

$$\Delta w_{ji}(n+1) = \eta \sum_{p=1}^{P} \delta_{pj} x_{pi} \qquad (12)$$

Donde,

$$\delta_{pj} = f'(net_{pj}) \sum_{k=1}^{M} \delta_{pk} v_{kj} \qquad (13)$$

Se puede observar que el error o valor delta asociado a una neurona oculta j, viene determinado por la suma de los errores que se cometen en las k neuronas de salida que reciben como entrada la salida de esa neurona oculta j. De ahí que el algoritmo también se denomine propagación del error hacia atrás.

Para la modificación de los pesos, la actualización se realiza después de haber presentado todos los patrones de entrenamiento. Este es el modo habitual de proceder y se denomina aprendizaje por lotes o modo *batch*.

Con el fin de acelerar el proceso de convergencia de los pesos, Rumelhart, Hinton y Williams (1986) sugirieron añadir en la expresión del incremento de los pesos un factor momento, α, el cual tiene en cuenta la dirección del incremento tomada en la iteración anterior. Así, cuando se trata del peso de una neurona de salida:

$$\Delta v_{kj}(n+1) = \eta \left(\sum_{p=1}^{P} \delta_{pk} b_{pj} \right) + \alpha \Delta v_{kj}(n) \qquad (14)$$

Cuando se trata del peso de una neurona oculta:

$$\Delta w_{ji}(n+1) = \eta\left(\sum_{p=1}^{P} \delta_{pj} x_{pi}\right) + \alpha \Delta w_{ji}(n) \quad (15)$$

Durante la etapa de aprendizaje de la red, los pesos son modificados de forma iterativa de acuerdo con los valores del grupo de entrenamiento, con el objeto de minimizar el error cometido entre la salida obtenida por la red y la salida deseada por el usuario. Sin embargo, como ya se ha comentado, cuando el número de parámetros o pesos es excesivo en relación al problema ocurre el fenómeno del sobreajuste o sobreentrenamiento, el modelo se ajusta demasiado a las particularidades irrelevantes presentes en los patrones de entrenamiento en vez de ajustarse a la función subyacente que relaciona entradas y salidas, perdiendo su habilidad de generalizar su aprendizaje a casos nuevos.

Para evitar el problema del sobreajuste, es aconsejable utilizar un segundo grupo de datos diferentes a los de entrenamiento, el grupo de validación, que permita controlar el proceso de aprendizaje. Durante el aprendizaje la red va modificando los pesos en función de los datos de entrenamiento y de forma alternada se va obteniendo el error que comete la red ante los datos de validación. Este proceso se ve representado en el entrenamiento que se realizó en el entorno de matlab. Allí se puede observar cómo el error de entrenamiento (error max) y el error de validación (error cuadrático medio) van disminuyendo a medida que aumenta el número de iteraciones, hasta alcanzar un mínimo, momento en el que podemos parar el aprendizaje de la red.

4.6.1 Elección de los pesos iniciales: Cuando una red neuronal es diseñada por primera vez, se deben asignar valores a los pesos a partir de los cuales se comienza etapa de entrenamiento. Los pesos de umbral y e conexión se pueden

inicializar de forma totalmente aleatoria, si bien es conveniente seguir algunas sencillas reglas que permitirán minimizar la duración del entrenamiento.

Es conveniente que la entrada neta a cada unidad sea cero, independientemente del valor que tomen los datos de entrada.

4.6.2 Función de activación de las neuronas ocultas y de salida: Se puede que para obtener el valor de salida de las neuronas de la capa oculta y de salida, se aplica una función, denominada función de activación, que sea continua y, por tanto, derivable para poder obtener el error o valor delta de las neuronas. Se debe tener en cuenta que la función pueda resolver problemas complejos y que no solamente sean lineales, para lo cual se va a implementar la función tangente hiperbólica [13].

$$f(x) = \frac{e^{gx} - e^{-gx}}{e^{gx} + e^{-gx}}, \text{ con } x = gin_i - \Theta_i. \qquad (16)$$

Los valores de salida de la función tangente hiperbólica están comprendidos dentro de un rango que va de -1 a 1. Al modificar el valor de *g* se ve afectada la pendiente de la función de activación.

4.7. Etapa de entrenamiento diseñado en matlab: Teniendo en cuenta la facilidad de lenguaje en el que se programa el software matlab, se realzó un código de entrenamiento para simular y probar la estructura elegida para la red.

4.7.1 Datos de entrada a la red: Son la señales EMG que se generan con el movimiento de flexión y extensión en el paciente. Se toman 6 muestras de los movimientos de forma intercalada (abrir - cerrar), 3 de cada uno.

Tabla 3. Datos de la señal abrir

EMG1	EMG2
347	344
378	334
406	324
483	363
575	415
582	410
544	373
542	372
555	401
510	395

EMG1	EMG2
347	344
385	342
409	326
476	356
571	411
587	416
551	379
538	368
547	394
512	397

EMG1	EMG2
347	344
383	339
408	326
478	358
572	413
586	414
549	377
539	369
550	396
512	396

Tabla 4. Datos de la señal cerrar

EMG1	EMG2
345	347
339	380
326	407
358	480
412	574
414	584
377	547
369	540
396	552
396	511

EMG1	EMG2
344	347
334	375
324	406
363	486
415	577
410	579
373	542
372	544
401	557
394	509

EMG1	EMG2
345	347
340	381
326	407
357	480
412	574
414	584
377	547
369	540
396	552
396	511

4.7.2 Normalización de los datos de entrada a la red: Los datos de entrada para entrenar le red corresponden al muestreo de dos señales, flexión (Abrir) y extensión (cerrar).

Los datos obtenidos fueron normalizados a niveles entre -1 y 1, para que la red pueda converger y llegue a un error cuadrático medio deseado.

Tabla 5. Datos de la señal abrir normalizados

EMG1	EMG2
-0.8251	-0.8479
-0.5894	-0.9240
-0.3764	-10.000
0.2091	-0.7034
0.9087	-0.3080
0.9620	-0.3460
0.6730	-0.6274
0.6578	-0.6350
0.7567	-0.4144
0.4144	-0.4601

EMG1	EMG2
-0.8251	-0.8479
-0.5361	-0.8631
-0.3536	-0.9848
0.1559	-0.7567
0.8783	-0.3384
10.000	-0.3004
0.7262	-0.5817
0.6274	-0.6654
0.6958	-0.4677
0.4297	-0.4449

EMG1	EMG2
-0.8251	-0.8479
-0.5513	-0.8859
-0.3612	-0.9848
0.1711	-0.7414
0.8859	-0.3232
0.9924	-0.3156
0.7110	-0.5970
0.6350	-0.6578
0.7186	-0.4525
0.4297	-0.4525

Tabla 6. Datos de la señal cerrar normalizados

EMG1	EMG2
-0.8403	-0.8251
-0.8859	-0.5741
-0.9848	-0.3688
-0.7414	0.1863
-0.3308	0.9011
-0.3156	0.9772
-0.5970	0.6958
-0.6578	0.6426
-0.4525	0.7338
-0.4525	0.4221

EMG1	EMG2
-0.8479	-0.8251
-0.9240	-0.6122
-10.000	-0.3764
-0.7034	0.2319
-0.3080	0.9240
-0.3460	0.9392
-0.6274	0.6578
-0.6350	0.6730
-0.4144	0.7719
-0.4677	0.4068

EMG1	EMG2
-0.8403	-0.8251
-0.8783	-0.5665
-0.9848	-0.3688
-0.7490	0.1863
-0.3308	0.9011
-0.3156	0.9772
-0.5970	0.6958
-0.6578	0.6426
-0.4525	0.7338
-0.4525	0.4221

4.7.3 Salidas deseadas de los datos ya normalizados: El tipo de red es una red supervisada lo que requiere tener unas salidas deseadas prederminada por los estados del pwm de motor.

Tabla 7. Datos de salida señal abrir normalizados

OUT1 (Velocidad)	OUT2 (Dirección)
-1.0000	1.0000
-1.0000	1.0000
-0.5083	1.0000
-0.8667	1.0000
0.9000	1.0000
-0.4750	1.0000
0.6417	1.0000
-0.7917	1.0000
0.7333	1.0000
-0.6000	1.0000

OUT1 (Velocidad)	OUT2 (Dirección)
-1.0000	1.0000
-1.0000	1.0000
-0.4833	1.0000
-0.9250	1.0000
0.8667	1.0000
-0.4250	1.0000
0.7000	1.0000
-0.8250	1.0000
0.6667	1.0000
-0.5833	1.0000

OUT1 (Velocidad)	OUT2 (Dirección)
-1.0000	1.0000
-1.0000	1.0000
-0.4917	1.0000
-0.9083	1.0000
0.8750	1.0000
-0.4417	1.0000
0.6833	1.0000
-0.8167	1.0000
0.6917	1.0000
-0.5917	1.0000

- Out1 [velocidad del motor]: (2*((EMG1-UEMG1) / (EMGMax-UEMG1))-1)
- UEMG1, umbral mínimo de captura de la señal EMG1.

Tabla 8. Datos de salida señal cerrar normalizados

OUT1 (Velocidad)	OUT2 (Dirección)
-1.0000	-1.0000
-0.7250	-1.0000
-1.0000	-1.0000
0.1083	-1.0000
-0.4583	-1.0000
0.9750	-1.0000
-0.7500	-1.0000
0.6083	-1.0000
-0.5917	-1.0000
0.3667	-1.0000

OUT1 (Velocidad)	OUT2 (Dirección)
-1.0000	-1.0000
-0.7667	-1.0000
-1.0000	-1.0000
0.1583	-1.0000
-0.4333	-1.0000
0.9333	-1.0000
-0.7833	-1.0000
0.6417	-1.0000
-0.5500	-1.0000
0.3500	-1.0000

OUT1 (Velocidad)	OUT2 (Dirección)
-1.0000	-1.0000
-0.7167	-1.0000
-1.0000	-1.0000
0.1083	-1.0000
-0.4583	-1.0000
0.9750	-1.0000
-0.7500	-1.0000
0.6083	-1.0000
-0.5917	-1.0000
0.3667	-1.0000

- Out1 [velocidad del motor]: (2*((EMG2-UEMG2) / (EMGMax-UEMG2))-1)
- UEMG2, umbral mínimo de captura de la señal EMG2.

4.7.4 Resultados Matlab

Figura 26. Datos de entrada del programa

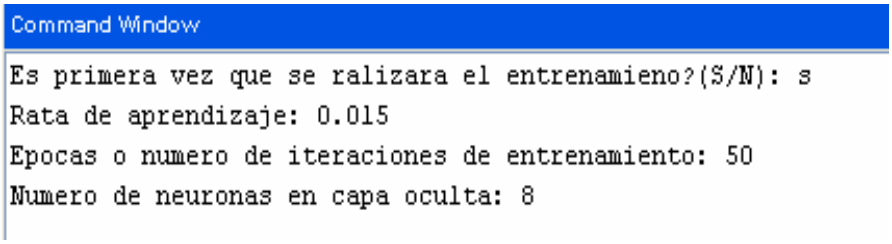


Figura 27. Proceso de entrenamiento

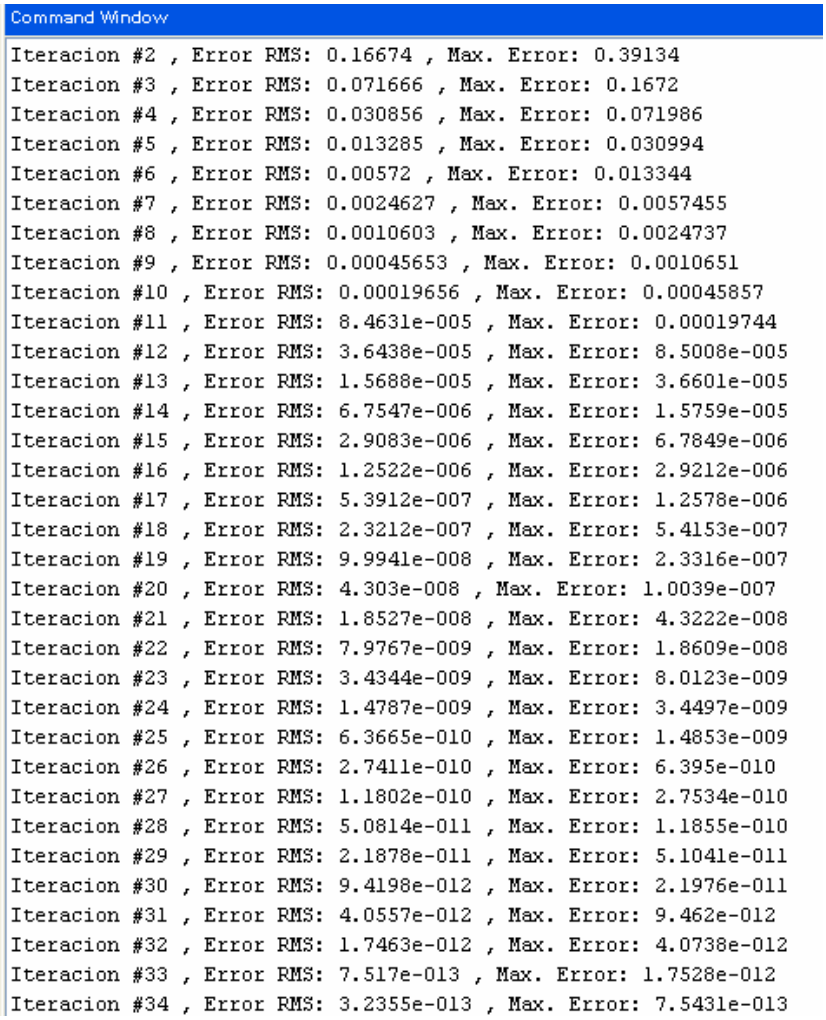
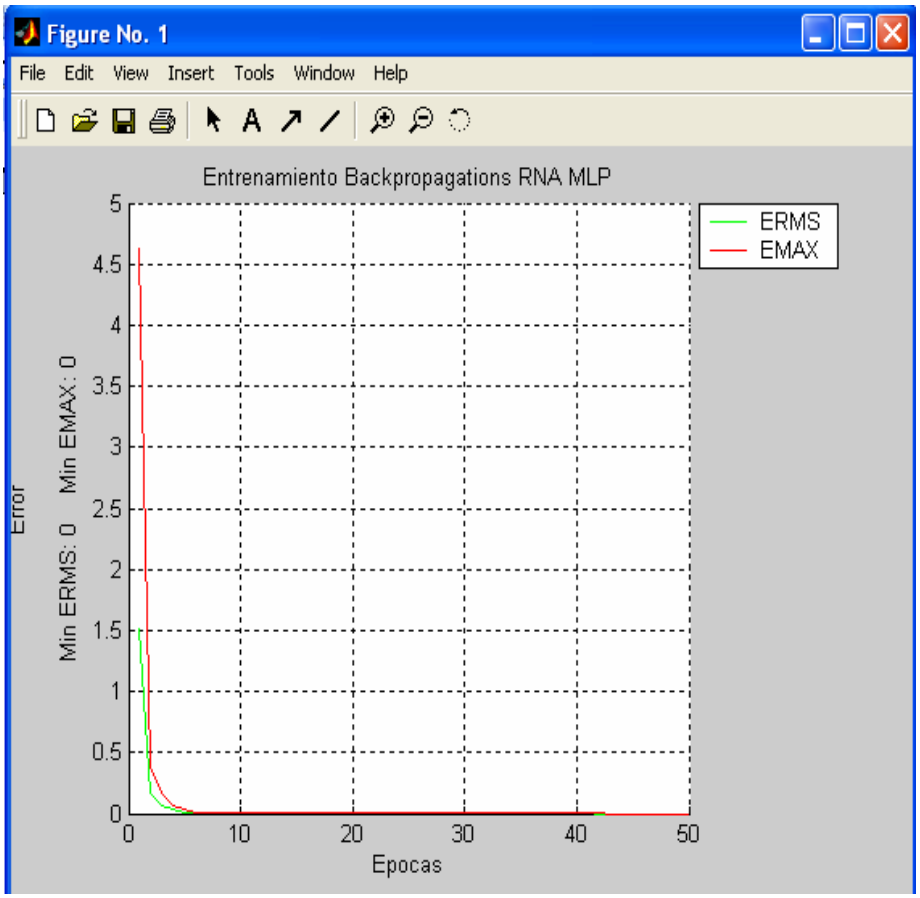


Figura 28. Grafica del error máximo por iteración y error cuadrático medio



4.8 EJECUCIÓN DE LA RED NEURONAL EN EL MICROCONTROLADOR

Para que el diseño del sistema sea independiente de un computador, se realizó la programación de toda la red en un microcontrolador PIC18F6722, el cual cumple con los requerimientos mínimos de memoria y velocidad para poder procesar un red neuronal.

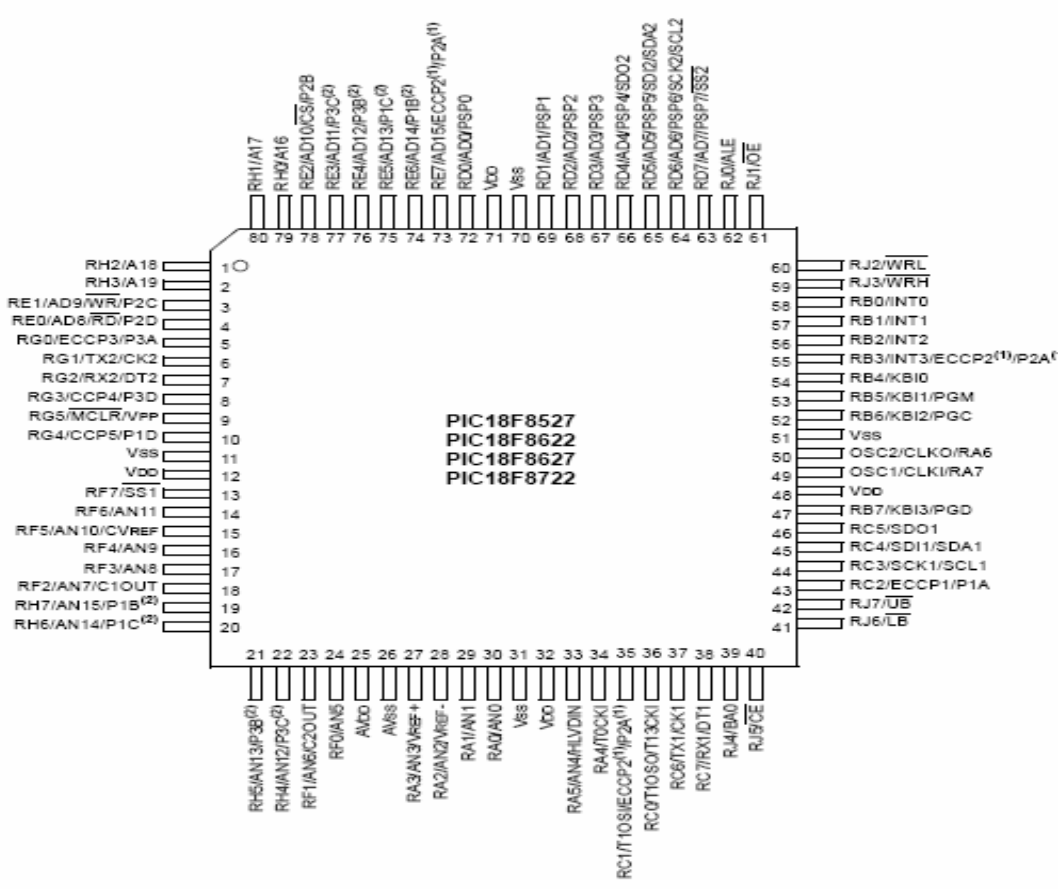
4.8.1 Características del microcontrolador PIC18F6722: En la siguiente tabla se muestran las características que se tuvieron en cuenta para programar la red en un microcontrolador de gama mejorada.

Tabla 9. Características principales del PIC

Device	Program Memory		Data Memory		I/O	10-bit A/D (ch)	CCP/ ECCP (PWM)	MSSP			EUSART	Comparators	Timers 8/16-bit	External Bus
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI™	Master I^2C™				
PIC18F6527	48K	24576	3936	1024	54	12	2/3	2	Y	Y	2	2	2/3	N
PIC18F6622	64K	32768	3936	1024	54	12	2/3	2	Y	Y	2	2	2/3	N
PIC18F6627	96K	49152	3936	1024	54	12	2/3	2	Y	Y	2	2	2/3	N
PIC18F6722	128K	65536	3936	1024	54	12	2/3	2	Y	Y	2	2	2/3	N
PIC18F8527	48K	24576	3936	1024	70	16	2/3	2	Y	Y	2	2	2/3	Y
PIC18F8622	64K	32768	3936	1024	70	16	2/3	2	Y	Y	2	2	2/3	Y
PIC18F8627	96K	49152	3936	1024	70	16	2/3	2	Y	Y	2	2	2/3	Y
PIC18F8722	128K	65536	3936	1024	70	16	2/3	2	Y	Y	2	2	2/3	Y

PDF PIC18F6722. Anexo

Figura 29. Diagrama de pines del microcontrolador



4.8.2 Requerimientos de software para la programación:

- **Software MATLAB 6.5:** Pruebas de entrenamiento para elegir la estructura de la red neuronal óptima para el diseño.
- **Software MPLAB I DE 7.1:** Plataforma para el diseño de algoritmos en los microcontroladores y programación de los mismos.
- **Software compilador CCS C PIC COMPLILER 3.1:** Compilador del algoritmo de los microcontroladores que están escritos en lenguaje C.
- **PLUING MPLAB CCS C PIC COMPILER:** Pluing (función adicional) que permite crear una interfaz entre el software compilador CCS C PIC COMPILER y el MPLAB IDE.
- **Software PROTEUS 6.7:** Emulador del diseño total de la prótesis.

4.8.2 Emulación del sistema

Figura 30. Emulación total del proyecto en proteus

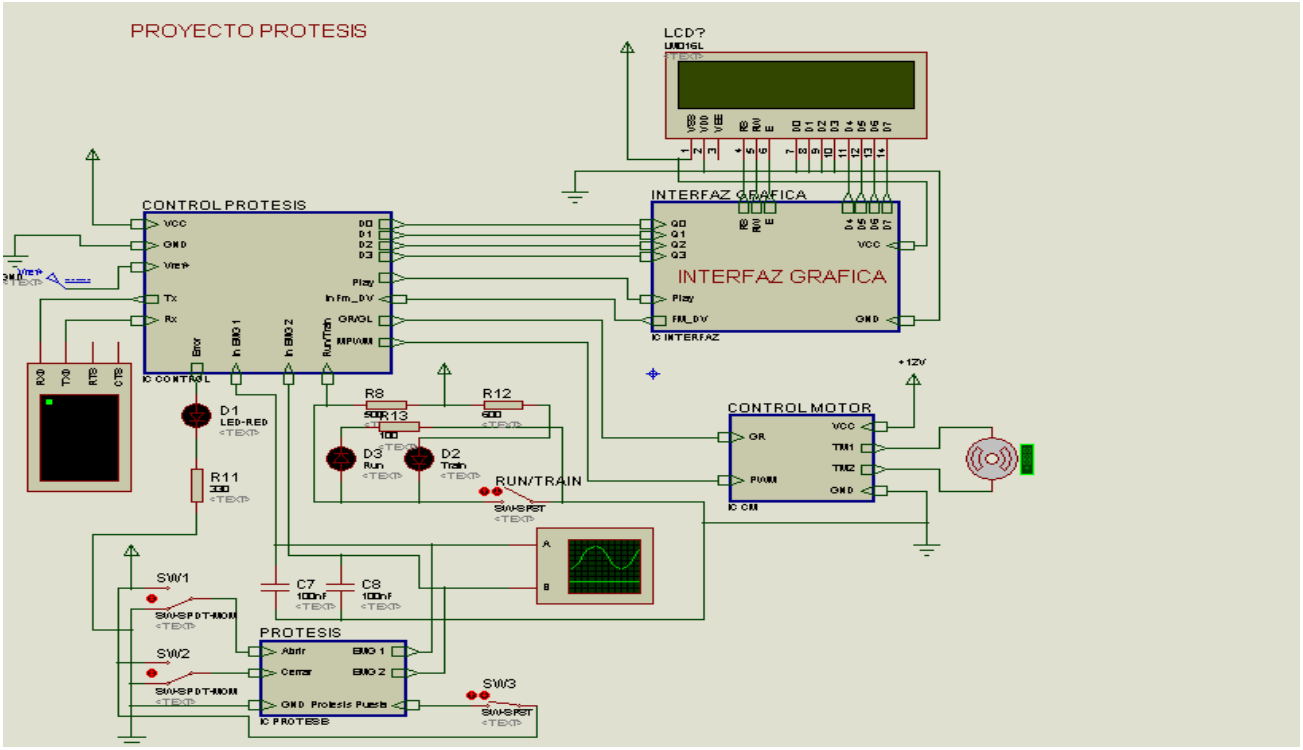


Figura 31. Emulación del control en el microcontrolador

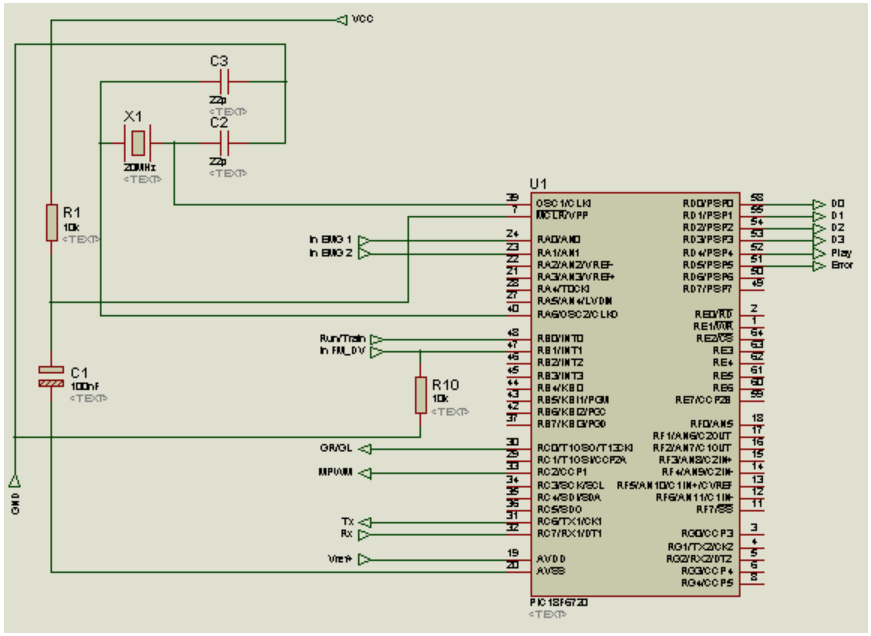


Figura 32. Emulación de la interfaz gráfica

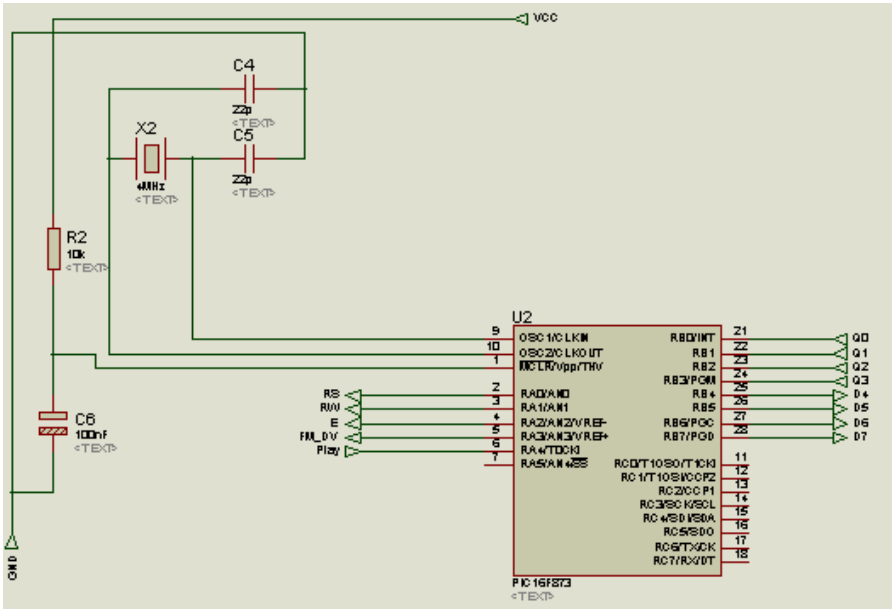


Figura 33. Esquemático del sistema de control

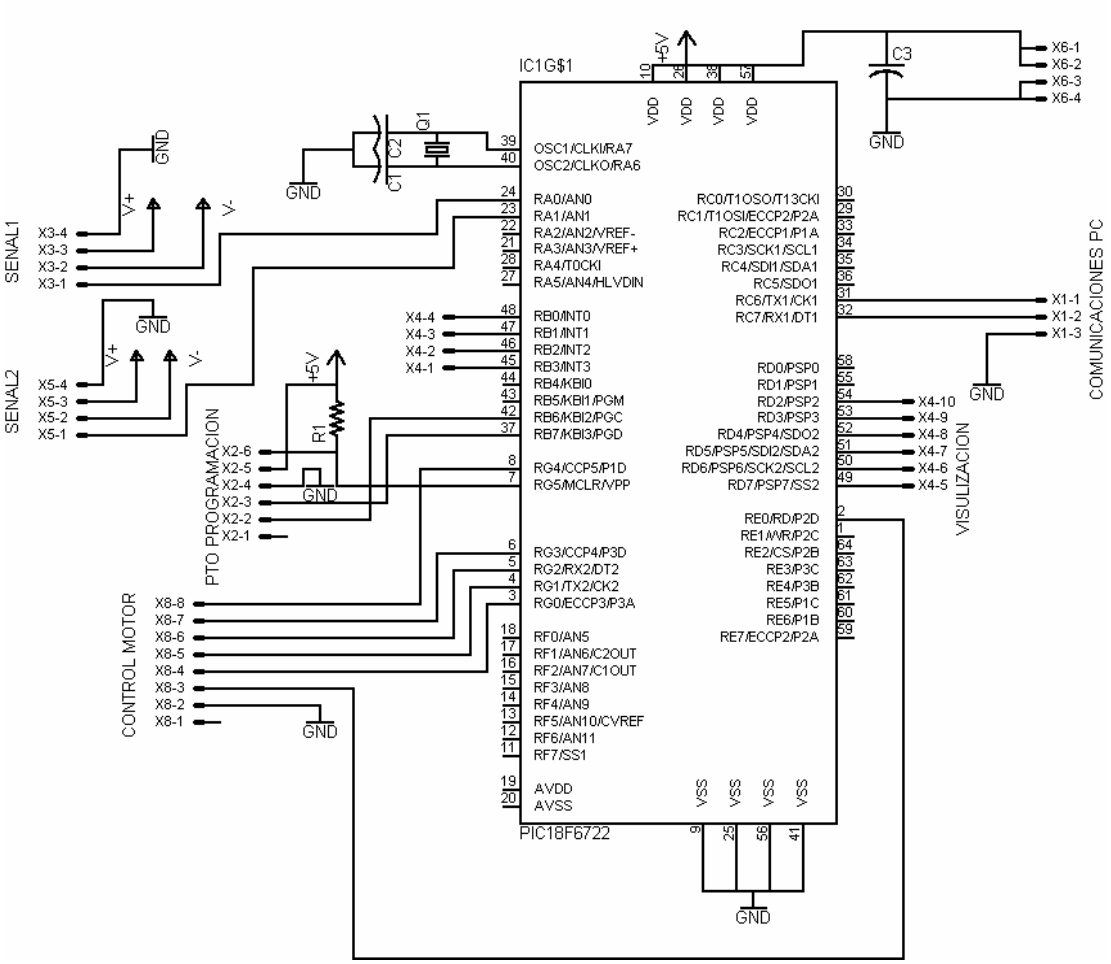
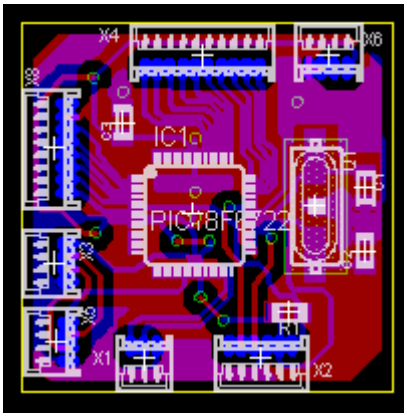


Figura 34. Board en eagle del sistema de control



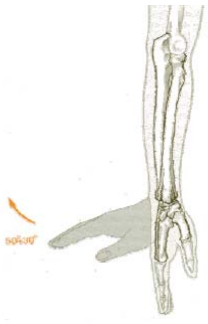
4.9 METODOLOGÍA DEL FUNCIONAMIENTO

Existen dos modos de funcionamiento que se seleccionan por medio de un switche: el modo TRAIN (entrenamiento) y el Modo RUN (ejecución).

Nota: Como sistema de seguridad se implementó un jumper en el hardware, con el fin de asegurar un entrenamiento previo en la prótesis, es decir, al iniciar el modo TRAIN, el sistema pedirá remover dicho elemento para proceder a el entrenamiento a menos que no haya una red entrenada en el sistema y al finalizar el proceso deberá colocarse de nuevo el jumper para asegurar el entrenamiento realizado.

4.9.1 Modo Train (Entrenamiento): Es donde se realiza la captura de datos y el entrenamiento de estos. Estos datos son capturados mediante los movimientos de flexión (abrir) y extensión (cerrar).

Flexión: Articulación imaginaria de la muñeca, que contrae los músculos flexores.



Extensión: Articulación imaginaria de la muñeca, que contrae los músculos extensores.

Figura 35. Merletti, Roberto. Parker, Philip A. Electromyography. Physiology, engineering, and noninvasive applications. Edit, IEEE Press Series in Biomedical Engineering. 2004

Durante el proceso de captura el paciente será guiado por un modulo externo que le indicará el estado de las señales iniciales y el tiempo en el que se deben generar los diferentes movimientos.

El sistema al finalizar totalmente el entrenamiento le indicará al paciente la conclusión del proceso y permitirá su paso al modo RUN (ejecución), para comenzar a ejecutar la red ya entrenada.

4.9.2 Modo Run (Ejecución): A partir de la captura se procesan los datos de entrada a la prótesis en tiempo real por medio de la red neuronal previamente entrenada, lo que da paso a los movimientos que el paciente desee realizar, ya sea abrir o cerrar la prótesis.

Nota: El modulo de guía no tendrá utilidad en el modo RUN , adicionalmente se encuentra de forma visual un led rojo que oscilara cuando se presente algún tipo de error en el sistema; cuando la oscilación sea rápida y constante es porque no existe un entrenamiento previo de la red, cuando la oscilación sea lenta y constante es porque las señales EMG iniciales no están listas y cuando el encendido del led sea constante es porque no hay jumper de seguridad conectado, habiendo una red previamente entrenada.

CAPITULO 5
CONCLUSIONES

- El proceso de entrenamiento, involucró experimentar con muchos tipos de redes tratando de encontrar no solo una configuración óptima, si no también un algoritmo que además de rapidez garantizara estabilidad en el resultado final. La red escogida involucró un gran número de parámetros, sin embargo alcanzó convergencia en menos tiempo que otras redes de una y tres capas ocultas con similar números de neuronas, de esta forma se confirma que no existe un procedimiento establecido para determinar el modelo de red que debe emplearse en cada aplicación y que sólo con la práctica puede determinarse cual es la configuración de red que garantiza mejores resultados.

- La implementación de la red neuronal en el microcontrolador requiere de gran capacidad de memoria RAM, EEPROM de datos y ciclos anidados, lo que llevó a escoger una gama alta de microcontroladores, los PIC18F6722.

- Los pasos necesarios para llevar a cabo la captura de los datos y el entrenamiento de la prótesis son complejos a la hora de ejecutarlos, el módulo guía permitió facilitar al usuario realizar cada una de las instrucciones a seguir durante este proceso para el óptimo funcionamiento de la prótesis.

- De acuerdo a las pruebas que se realizaron del programa, aunque el toolbox de matlab permite simular redes neuronales, no es complementario cuando se desea interpretar y programar directamente sobre un hardware.

- Cuando se realiza la captura de los datos de entrada a la red, no se esta completamente seguro si estos son lineales o no lineales, pues las señales pueden tener picos de voltaje que pueden generar parámetros no lineales, para lo cual se implementó una red que no solo procesa datos lineales y con esto se previenen esos picos de voltaje que pueden de alguna forma evitar una buena ejecución en el proceso.

- Para un perfecto funcionamiento del sistema programado, no solo se requiere de una buena caracterización de las señales e implementación de la red, también es conveniente realizar un entrenamiento previo al paciente como tal, para que estimule las señales y recuerde de forma correcta los movimientos que se generan en la prótesis.

BIBLIOGRAFÍA

[1]. PDF Redes neuronales. Conceptos básicos y aplicaciones. Grupo de Investigación Aplicada a la Ingeniería Química (GIAIQ). Universidad Tecnológica Nacional – Facultad Regional Rosario. Departamento de Ingeniería Química.

[2]. Http://personales.ya.com/emgnm/emg.htm

[3]. Barea navarro, Rafael. Instrumentación Biomédica. Departamento electrónica. Unviversidad Alcalá.

[4]. http://www.teoloyucan.com/biopac_pri.html

[5]. kendall medical Electrodos. www.kendall-ltp.com/pdf/acage.pdf

[6]. Surface electromyography systems, EMG electrode. www.delsys.com

[7]. Delgado, Alberto. Inteligencia Artificial y Minirobots. ECOE Ediciones. Santa Fe de Bogota, 1998.

[8]. Martín del Brío, B. y Sanz, A. Redes Neuronales y Sistemas Difusos. RA-MA. 2ª edición, Madrid, 2000.

[9]. Peñaloza, Jaime. Gómez, Jaime. Tesis de Grado: Diseño Y Construcción De Un Sistema Identificador De Colores Básicos, Bucaramanga, 2004.

[10]. Hilera, José R. y Martínez, Víctor J. Redes neuronales Artificiales. RAMA. Madrid, 2000.

[11]. Http://www.monografias.com/trabajos12/redneur/redneur.shtml

[12]. Rumelhart, D.E., Hinton, G.E. y Williams, R.J. (1986). Learning internal representations by error propagation. En: D.E. Rumelhart y J.L. McClelland (Eds.) *Parallel distributed processing* (pp. 318-362). Cambridge, MA: MIT Press.

[13]. Tutorial sobre Redes Neuronales Artificiales: El Perceptrón Multicapa. Palmer, A., Montaño, J.J. y Jiménez, R. Área de Metodología de las Ciencias del Comportamiento. Facultad de Psicología.

- E. H., Biomedical Engineering, edit John Wiley & Sons, Inc.

- Muzumdar, Ashok. Powered Upper Limb Prostheses. Control, Implementation and clinical Application. Edit, Springer, 2004.

- Merletti, Roberto. Parker, Philip A. Electromyography. Physiology, engineering, and noninvasive applications. Edit, IEEE Press Series in Biomedical Engineering. 2004

- Angulo Usatergui, José María. Microcontroladores PIC. Diseño practico y aplicaciones. Edición, Mc Graw Hill. 2000.

- Freeman, James. Skapura, David. Redes Neuronales. Algoritmos, aplicaciones, y tecnicas de programación. 1991.

- Martín del Brio. Bonifacio. Redes Neuronales y sistemas difusos. Segunda Edición Ampliada y revizada. 2005.

ANEXOS

ANEXO 1: Algoritmo de control realizado en matlab

ANEXO 2: Algoritmo de todo el sistema de control

ANEXO 3: Hoja de datos Amplificador Operacional LF353

ANEXO 4: Hoja de datos Microcontrolador PIC18F6722

ANEXO 1

ALGORITMO DE CONTROL REALIZADO EN MATLAB

```
close all;
clc;
clear all;
%cargar entradas X y salidas deseadas YD respecto a NE
load 'input.mat'
NE=size(X,2);%Numero de Ejemplos

Min=min(min(X'));
Max=max(max(X'));

k=0;
for i=1:1:10
k=k+1;
Tem(1,k)=X(1,i);
Tem(2,k)=X(2,i);
k=k+1;
Tem(1,k)=X(1,(10+i));
Tem(2,k)=X(2,(10+i));
end

for i=21:1:30
k=k+1;
Tem(1,k)=X(1,i);
Tem(2,k)=X(2,i);
k=k+1;
Tem(1,k)=X(1,(10+i));
Tem(2,k)=X(2,(10+i));
end

for i=41:1:50
k=k+1;
Tem(1,k)=X(1,i);
Tem(2,k)=X(2,i);
k=k+1;
Tem(1,k)=X(1,(10+i));
Tem(2,k)=X(2,(10+i));
end
```

```
VE1max=1640;
Offset=1500;
VE2max=1640;
DV=(5/1023)*1000;
UEMG0=VE1max; %VEMG1max;
UEMG0=floor((UEMG0+((UEMG0-Offset)*0.4)));
UEMG0=floor((UEMG0/DV));
UEMG1=VE2max; %VEMG1max;
UEMG1=floor((UEMG1+((UEMG1-Offset)*0.4)));
UEMG1=floor((UEMG1/DV));
n=0;

for k=1:1:NE

if(n==0)
Xn(1,k)=(2*((Tem(1,k)-Min)/(Max-Min))-1);
YD(1,k)=(2*((Tem(1,k)-UEMG0)/(Max-U+EMG0))-1);
YD(2,k)=1;
Xn(2,k)=(2*((Tem(2,k)-Min)/(Max-Min))-1);
n=1;
else
Xn(2,k)=(2*((Tem(2,k)-Min)/(Max-Min))-1);
YD(1,k)=(2*((Tem(2,k)-UEMG1)/(Max-UEMG1))-1);
YD(2,k)=-1;
Xn(1,k)=(2*((Tem(1,k)-Min)/(Max-Min))-1);
n=0;
end

if(YD(1,k)>1)
YD(1,k)=1;
end
if(YD(1,k)<-1)
YD(1,k)=-1;
end
end
A=input('Es primera vez que se ralizara el entrenamieno?(S/N): ','s');
RA=input('Rata de aprendizaje: ');
```

```
Epoc=input('Epocas o numero de iteraciones de entrenamiento: ');

if((A=='N')||(A=='n'))
%Cargar pesos respecto a NO
load 'pesos.mat' pesos
NO=size(pesos{2,1},2);
else
NO=input('Numero de neuronas en capa oculta: ');
%Reinicio de los pesos
for i=1:1:NO
pesos{1,2}(1,i)=(0.6-(randn));
pesos{2,3}(1,i)=(0.6-(randn));
pesos{1,2}(2,i)=(0.6-(randn));
pesos{2,3}(2,i)=(0.6-(randn));
pesos{2,1}(i)=(0.6-(randn));
end
end
figure;
hold on;
title(['Entrenamiento Backpropagations RNA MLP']);
xlabel('Epocas');
ylabel('Error');
for it=1:1:Epoc
emax=0;
erms=0;

for k=1:1:NE
Y(1)=0;
Y(2)=0;

%Propagacion
for i=1:1:NO
B=pesos{1,2}(1,i)*X(1,k)+pesos{1,2}(2,i)*X(2,k)+pesos{2,1}(i);
Y(1)=Y(1)+(pesos{2,3}(1,i)*tansig(B));
Y(2)=Y(2)+(pesos{2,3}(2,i)*tansig(B));
end

e1=Y(1)-YD(1);
```

```
e2=Y(2)-YD(2);

%Correccion de pesos
for i=1:1:NO
B=pesos{1,2}(1,i)*X(1,k)+pesos{1,2}(2,i)*X(2,k)+pesos{2,1}(i);
AE=e1*pesos{2,3}(1,i)+e2*pesos{2,3}(2,i);

pesos{2,3}(1,i)=pesos{2,3}(1,i)-(RA*e1*tansig(B));
pesos{2,3}(2,i)=pesos{2,3}(2,i)-(RA*e2*tansig(B));
pesos{2,1}(i)=pesos{2,1}(i)-(RA*AE*dtansig(B,tansig(B)));
pesos{1,2}(1,i)=pesos{1,2}(1,i)-(RA*AE*X(1,k)*dtansig(B,tansig(B)));
pesos{1,2}(2,i)=pesos{1,2}(2,i)-(RA*AE*X(2,k)*dtansig(B,tansig(B)));
end

%Maximo error por iteracion

if(emax < sqrt(e1*e1+e2*e2))
emax=sqrt(e1*e1+e2*e2);%no intereza
end

erms=erms+(e1*e1+e2*e2);

end

erms=sqrt(erms/NE);

fprintf(['\nIteracion #%d , Error RMS: ',num2str(erms),' , Max. Error: ',num2str(emax)],it);

if(it >1 )
plot([(it-1) it],[Aerms erms],'-g');
plot([(it-1) it],[Aemax emax],'r');
end
Aerms=erms;
Aemax=emax;
end

axis;
grid on;
```

```
legend({'ERMS','EMAX'},5);
xlabel('Epocas');
ylabel({'Error';''};['Min ERMS: ',num2str(erms),'    Min EMAX: ',num2str(emax)]});
hold off;

if((A=='N')||(A=='n'))
A=input('\n\nDesea salvar estos ultimos pesos?(S/N): ','s');
if((A=='S')||(A=='s'))
%guardar los pesos resultantes
save 'pesos.mat' pesos Y -append
clear all;
end
else
%guardar los pesos resultantes
save 'pesos.mat' pesos Y -append
end
```

ANEXO 2

ALGORITMO DE CONTROL DE TODO EL SISTEMA

```
#include <18f6722.h>
#device ADC=10
#fuses HS,NOPROTECT
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <float.h>
#use delay(clock=20000000,RESTART_WDT)
#use RS232(BAUD=9600, BITS=8, PARITY=N, XMIT=PIN_C6, RCV=PIN_C7, RESTART_WDT)
#use fast_io(A)
#use fast_io(B)
#use fast_io(C)
#use fast_io(D)
#byte porta=3968
#byte portb=3969
#byte portc=3970
#byte portd=3971

#define Control pin_b0
#define In_FM_DV pin_b1
#define JST pin_b2
#define Pinza pin_c0
#define MPWM pin_c2
#define Play pin_d4
#define Error pin_d5
#define Open 0
#define Close 1
#define EMG1 0
#define EMG2 1
#define Frec 3
//*****Valores de las señales***
#define VE1max 1640      //Vmax de Error en EMG1=1640 mv
#define VE2max 1640      //Vmax de Error en EMG2=1640 mv
#define Offset 1500
#define TE 4000          //Periodo del Error=4000 us
#define DV 4.88758       //Vref= 5 voltios => DV ADC=4.88758 mv =(5v/1023)

#define VEMG1max 3000//Voltaje EMG1 maximo=2800 mv
```

```
#define VEMG2max 3000//Voltaje EMG1 maximo=2800 mv
#define TEMG1 550//Periodo de la señal EMG1=100 ms
#define TEMG2 550//Periodo de la señal EMG2=100 ms
#define NM 10//Numero de muestras a tomar

#define NO 8//Numero de neuronas en capa oculta
#define Epoc 50////Numero de iteraciones para el entrenamiento
#define NE 60//Numero de ejemplos
#define RA 0.015//Rata de aprendizaje

int16 Min=1023,Max=0;

#INT_EXT
Interrupcion_RB0()
{disable_interrupts(INT_EXT);
output_high(Error);
delay_ms(200);
output_low(Error);
delay_ms(200);
output_high(Error);
delay_ms(200);
output_low(Error);
delay_ms(200);
output_high(Error);
delay_ms(200);
output_low(Error);
delay_ms(200);
output_high(Error);
delay_ms(200);
output_low(Error);
reset_cpu();
}

int16 Leer_ADC(int Var)
{set_adc_channel(Var);
delay_us(10);
return(read_adc());
}
```

```
int Leer_EEPROM(int16 Dir)
{return (read_eeprom(Dir));}

Escribir_EEPROM(int16 Dir,int Data)
{write_eeprom(Dir,Data);}

Compruebe_EMG()
{int i=0;
int16 EMG_ADC=0,A=0,B=0,TSE=0;

Interfaz(3); //Comprobando señales EMG
A=VE1max;
A=(Offset+((A-Offset)/4));
A=(A/DV);
TSE=TE;
TSE=(TSE/20);
EMG_ADC=Leer_ADC(EMG1);
while((EMG_ADC<A)|(EMG_ADC>(A+10)))
{
if(input(Error)==0)
{output_high(Error);}
else
{output_low(Error);}
Interfaz(4); //Ajuste señal EMG1
for(i=1;i<=20;i++)
{delay_us(TSE);
EMG_ADC=Leer_ADC(EMG1);
if((EMG_ADC>=A)&(EMG_ADC<=(A+10)))
{break;}
}
}
output_low(Error);
Interfaz(5); //EMG1 Lista

B=VE2max;
B=(Offset+((B-Offset)/4));
B=(B/DV);
```

```
EMG_ADC=Leer_ADC(EMG2);
while((EMG_ADC<B)|(EMG_ADC>(B+10)))
{
if(input(Error)==0)
{output_high(Error);}
else
{output_low(Error);}
Interfaz(6); //Ajuste señal EMG2
for(i=1;i<=20;i++)
{delay_us(TSE);
EMG_ADC=Leer_ADC(EMG2);
if((EMG_ADC>=B)&(EMG_ADC<=(B+10)))
{break;}
}
}
output_low(Error);
Interfaz(7); //EMG2 Lista
}

void main()
{int j=0,i=0,B0,B1,B2,B3,A1[3];
int16 EMG_ADC=0,A=0,B=0,TSE=0,n=0;
signed int32 T,T2;
float C,D,M;

set_tris_a(0b00000011);
set_tris_b(0b10001111);
set_tris_c(0b10000000);
set_tris_d(0b00000000);
portd=0;
setup_ccp1(CCP_OFF);
output_low(Error);
setup_adc_ports(ANALOG_AN0_TO_AN1|VSS_VDD); //In analogas RA0-RA1, Range 0-Vdd, Justificacion a la derecha
setup_adc(ADC_CLOCK_DIV_16);
output_low(MPWM);
output_high(Play);
delay_ms(2000);
```

```
if(input(Control)==0)//Configuracion de interrupcion INT0
{
EXT_INT_EDGE(L_TO_H);                                   // Interupcion INT con flanco
de subida
enable_interrupts(INT_EXT);                             // Seleccionar interrupciones por RB0
enable_interrupts(GLOBAL);
}
else
{
EXT_INT_EDGE(H_TO_L);                                   // Interupcion INT con flanco
de subida
enable_interrupts(INT_EXT);                             // Seleccionar interrupciones por RB0
enable_interrupts(GLOBAL);
}

if(input(Control)==0)
{
i=0;
while(i==0)
{i=Interfaz(2);//Modo TRAIN
}

if(Leer_EEPROM(0)==1)
{
while(input(JST)==0)
{Interfaz(0);} //Retire el Jumper De Seguridad RNA
}
Compruebe_EMG();
Escribir_EEPROM(0,0);//Desactivar entrenamineto completado
//**********Emulacion Abrir (1)**********
Interfaz(8); //Emulacion de accion abrir
Capture_Datos(EMG1,1);

//**********Emulacion Cerrar (1)**********
Interfaz(11); //Emulacion de accion cerrar
Capture_Datos(EMG2,1);
```

```
//***********Emulacion Abrir (2)***********
Interfaz(8); //Emulacion de accion abrir
Capture_Datos(EMG1,2);

//***********Emulacion Cerrar (2)***********
Interfaz(11); //Emulacion de accion cerrar
Capture_Datos(EMG2,2);

//***********Emulacion Abrir (3)***********
Interfaz(8); //Emulacion de accion abrir
Capture_Datos(EMG1,3);

//***********Emulacion Cerrar (3)***********
Interfaz(11); //Emulacion de accion cerrar
Capture_Datos(EMG2,3);

Escribir_EEPROM(6,make8(Min,0));
Escribir_EEPROM(7,make8(Min,1));
Escribir_EEPROM(8,make8(Max,0));
Escribir_EEPROM(9,make8(Max,1));

Entrenamiento();
Escribir_EEPROM(0,1);//Activar entrenamineto completado
while(1)
{
output_high(Error);
delay_ms(500);
output_low(Error);
Interfaz(14);//Entrenamiento terminado
output_high(Error);
Interfaz(1); //Ponga el Jumper De Seguridad RNA
output_low(Error);
delay_ms(500);
output_high(Error);
delay_ms(500);
output_low(Error);
Interfaz(15);//Pasar a modo RUN
```

```
}
}
else
{

while((Leer_EEPROM(0))!=1)//Mira si no hay Programa RNA de Control
{
output_high(Error);
delay_ms(200);
output_low(Error);
delay_ms(200);
output_high(Error);
delay_ms(100);
output_low(Error);
delay_ms(100);
}

Compruebe_EMG();
Ejecute_RNA();
}

}
```

ANEXO 3

HOJA DE DATOS LF353

ANEXO 4

HOJA DE DATOS PIC18F6722