UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

PROYECTO FIN DE CARRERA

Control locomotor multidireccional de un robot modular mediante Circuitos Generadores Centrales de Patrones Bioinspirados

Damián Zamorano Martín

Septiembre 2011

Control locomotor multidireccional de un robot modular mediante Circuitos Generadores Centrales de Patrones Bioinspirados

AUTOR: Damián Zamorano Martín
TUTOR: Pablo Varona Martinez

GNB
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Septiembre 2011

PROYECTO FIN DE CARRERA

Título: Control locomotor multidireccional de un robot modular mediante Circuitos Generadores Centrales de Patrones Bioinspirados

Autor: D. Damián Zamorano Martín

Tutor: D. Pablo Varona Martínez

Resumen:

El presente proyecto muestra la obtención y el análisis de varias modalidades de locomoción para un robot modular de 8 módulos empleando generadores centrales de patrones (CPGs, siglas en inglés). Los CPGs son redes neuronales que emplean los seres vivos para producir actividad motora que implique cierta ritmicidad, tal como puede ser el caminar en el ser humano. A la hora de buscar un diseño bio-inspirado para la locomoción de robots, puede ser muy útil emplear CPGs simulados, pues estos aportan al sistema robustez, autonomía y recuperación ante posibles errores. El modelo de neurona empleado en este proyecto vendrá definido por una serie de mapas iterados, y la interconexión entre las neuronas tendrá una serie de reglas tales como la localidad (una neurona solo puede conectarse con las de su entorno) y la competición (conexiones inhibidoras). El resultado serán unos CPGs capaces de realizar 5 movimientos diferentes. El robot estará compuesto por módulos cada uno de los cuales se coloca con el eje perpendicular respecto con su anterior. El movimiento de este robot ya se ha estudiado mediante un control con señales sinusoidales desfasadas, por lo tanto, a la hora de obtener los movimientos, no habrá más que ver los parámetros existentes (amplitud y fases) y obtenerlos en nuestro CPG. Se muestra por tanto una técnica de obtención de amplitudes, y tres técnicas para la obtención de fases (cada una de ellas para una situación en particular).

Abstract:

The present investigation shows the obtaining and the analysis of new motion techniques for a modular robot of eight modules using CPGs. CPGs are neural networks that living beings use to generate motor activity, which is involved in motion that require periodicity, for instance, human being locomotion. The use of simulated CPGs in the context of bio-inspired design for robot locomotion is highly appropriate because CPGs are found to contribute to system robustness, autonomy and stability, in the sense of error recovery capability. In the present project, the neuron model used is defined by a set of iterated equations, and interconnection among neurons is determined by a set of rules, for instance, the principle of locality, which states that a neuron only can be connected with neighbour neurons. The result will be a global CPG, with the capability to make five different movements, oscillating some parameters of a set of equations. The robot will be composed by eight modules, where each module axes are disposed perpendicular with respect to the axes of the previous module. The motion of this robot has already been studied, even though; it is interesting to make a rigorous study for obtaining new movements by

simulations. It is important to stand out the fact that the movement of this robot used to do with out of phase sinusoidal signals. Thus, it is only necessary in this case to attend to parameters, such as amplitude and phases, and obtaining them for the own CPG in order to obtain motion. It is showed, therefore, a technique for obtaining amplitudes and three techniques for obtaining phases, each ones used for concrete situations.

Palabras clave:
CPG, Redes neuronales, neural network, Robot modular bio-inspirado, Bio-inspired modular robot

Agradecimientos

Este proyecto se lo agradezco a Sarita, por ser la mejor.

ÍNDICE DE CONTENIDOS

INDICE DE FIGURAS

ÍNDICE DE TABLAS

1 Introducción

Este proyecto fin de carrera es un proyecto de investigación en el campo del control locomotor multidireccional de un robot modular mediante circuitos Generadores Centrales de Patrones (CPGs) bioinspirados".

Para empezar habría que mencionar el contexto en el que se realiza este proyecto. Mi trabajo aquí no parte de cero, pues es la continuación del trabajo de mucha otra gente, entre la que se encuentra Fernando Herrero, que ya para su doctorado ha trabajado con un proyecto similar pero para un robot que se movía con una grado de libertad, en un movimiento similar al de un gusano [26].

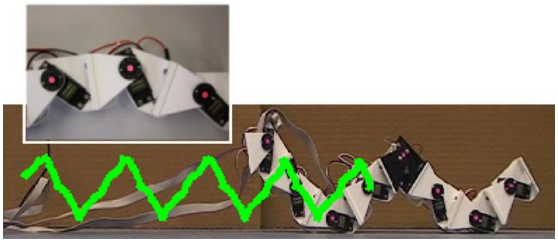

Figura 1 - Robot modular con locomoción controlada por CPGs en un grado de libertad de Fernando Herrero

El robot a emplear es fruto del trabajo de Juan González, que ha investigado mucho este tipo de robots modulares con diferentes configuraciones y los diferentes movimientos que se pueden conseguir con ellos utilizando señales sinusoidales desfasadas [27]. A continuación, se muestra el robot construido para este proyecto, que fue diseñado por Juan, y cuyo manual de instalación muestra en su web:

Figura 2 - Robot construido para el desarrollo de este proyecto a partir del diseño de Juan González

Por último, una mención especial al Grupo de Neurocomputación Biológica (GNB) de la EPS, el cual aporta ideas y conocimientos para crear este tipo de sistemas bioinspirados, y que me ha dado la facilidad para investigar en una de sus parcelas.

Por tanto, el presente trabajo puede dividirse en dos partes. La primera consistirá en estudiar y reproducir el trabajo ya realizado. Para ello, habrá que crear un robot modular que sea capaz de producir los modos de locomoción previstos. . Este robot está compuesto por piezas de plástico mandadas a construir para este proyecto, y por servomotores Futaba (normalmente empleados en aeromodelismo). Posteriormente, se ensamblan las piezas y se cablea debidamente (hay ocho motores y cada uno tiene tres cables). En este robot, se ha elegido unir los diferentes módulos con bridas, a diferencia de otros casos anteriores que estaban unidos por tornillos. Tal decisión se tomó para hacer más fácil el cambio de configuración, ya que las bridas agilizan mucho la unión de módulos. Los vectores de movimientos son generados por un programa en Matlab, que los almacena en un fichero. Para introducir los vectores de movimiento en el robot se emplea un programa en phyton que habrá que modificar para adaptarlo al robot en concreto.

A continuación habrá que hacer un programa (primero en Matlab y posteriormente en C++ para aumentar así su rendimiento) que simule los circuitos generadores de patrones. Estos circuitos generadores de patrones son modelados por una serie de mapas iterados. El programa realizado se ha diseñado con el fin de ser lo más manejable posible, ya sea para hacer las pruebas de este proyecto, o para su uso en otros robot similares con diferentes configuraciones y número de módulos.

También hay que mencionar el trabajo realizado en el simulador. Dicho simulador ha sido empleado con anterioridad en otros proyectos similares, pero en este caso se deseaba realizar pruebas de rendimiento para los diferentes movimientos al ir cambiando algunos parámetros. Se ha eliminado la parte gráfica del simulador, pues además de ser bastante pesada computacionalmente, no aportaba datos relevantes para este fin. Tras eso, se ha modificado para que a partir de los vectores de movimientos de almacenado en un fichero, correspondientes a miles de experimentos, generase otro fichero con la longitud y dirección recorrida por el robot para cada experimento y así seleccionar los mejores casos. En esta fase se llegaron a hacer pruebas donde el programa estaba funcionando durante dos días. Las pruebas hechas en el simulador corresponden a señales sinusoidales, y no a las provenientes del CPG, pues lo que se pretende es identificar movimientos eficientes con amplitudes y fases, para posteriormente identificar amplitudes y fases con configuraciones de CPG, y así simplificar el problema.

La segunda parte será la más creativa del proyecto, dado que tendré que aportar ideas propias, y buscar configuraciones eficientes del CGP para conseguir los movimientos del robot en tres dimensiones. El diseño de un CPG no es algo trivial pues requiere la plena comprensión de las ecuaciones que componen su modelo matemático. Para empezar, se estudió cómo conseguir fases y amplitudes con configuraciones más sencillas (empezando por dos módulos) y, posteriormente, extrapolar dichos resultados a nuestra configuración. Se han propuesto varias formas diferentes de cambiar las fases entre módulos, y combinando todas ellas se han creado cinco movimientos que puede realizar el robot. Estos movimientos son los siguientes:

- Locomoción Sinusoidal: El robot se mueve similar a como lo haría un gusano en una trayectoria recta.
- Locomoción Turning: El robot se mueve similar al movimiento sinusoidal, pero con una trayectoria curva, en la cual se puede variar su radio.

- Locomoción Lateral Shifting: El robot se mueve de manera similar a como lo hace una serpiente de cascabel.
- Locomoción Lateral: El robot se mueve lateralmente desplazando en un paso la cabeza y cola y en otro paso desplazando el cuerpo.
- Locomoción Rolling: El robot rueda sobre sí mismo para desplazarse lateralmente.

En el siguiente enlace pueden verse los 5 movimientos descritos en el mismo orden de secuencia:

http://www.youtube.com/watch?v=jtgie0xjon4

1.1 Motivación

Cuando hablamos de redes neuronales, lo primero que viene a la cabeza son los sistemas de auto-aprendizaje. En el cerebro humano, por ejemplo, las neuronas están interconectadas entre sí con otras neuronas de su entorno, formando complejas redes y un número de caminos posibles muy elevado. El proceso de aprendizaje consiste en fortalecer unos caminos determinados de esa red neuronal frente a otros caminos posibles. Así por ejemplo, cuando aprendemos una actividad nueva que requiera cierta habilidad, como puede ser un deporte, estaremos fortaleciendo unos caminos frente a otros, adquiriendo mayor destreza a medida que se practica el deporte. A esto comúnmente se le llama "mecanizar los movimientos", pues lo que al principio es un movimiento lento y torpe se convierte con la práctica en un movimiento rápido y fluido. Las redes neuronales de aprendizaje hacen esto mismo, a los caminos más comunes se les porta más peso frente a los menos usados.

En este proyecto, el objetivo no es el aprendizaje, pues ya directamente obtendremos de forma artificial los pesos de cada camino que hacen obtener el movimiento deseado. Para entrenar una red neuronal necesitamos un objetivo definido y un feedback que nos diga si estamos mejorando o empeorando, y puesto que el robot que utilizaremos carece de sentidos, no será posible hacerlo así. Por lo tanto, el objetivo en este proyecto es obtener diferentes señales de control de locomoción autónoma con redes neuronales, utilizando como variables las mismas que los organismos vivos emplean de forma natural, pero sin auto-aprendizaje por repetición.

En el transcurso de la evolución, los organismos han ido adquiriendo diversos diseños para la adaptación eficaz a las exigencias ambientales. La gran diversidad de formas de vida existentes no es más que diversas soluciones dadas, probadas y válidas ante cada uno de los problemas en cuestión. Un aspecto muy interesante, es ver cómo algunos organismos han adquirido soluciones semejantes ante problemas similares pero por dos ramas evolutivas diferentes. A este proceso, se le denomina "convergencia evolutiva", y consiste en un fenómeno evolutivo por el que organismos diferentes, relativamente alejados filogenéticamente, tienden, bajo presiones ambientales equivalentes, a desarrollar en su evolución características (morfológicas, fisiológicas, etológicas, etc.) semejantes (estructuras análogas). El ejemplo que suele darse cuando se habla de convergencia

evolutiva es el caso de los tiburones, delfines e ictiosaurios, que muestran semejanzas sin tener un parentesco especial (Ver Figura 1).

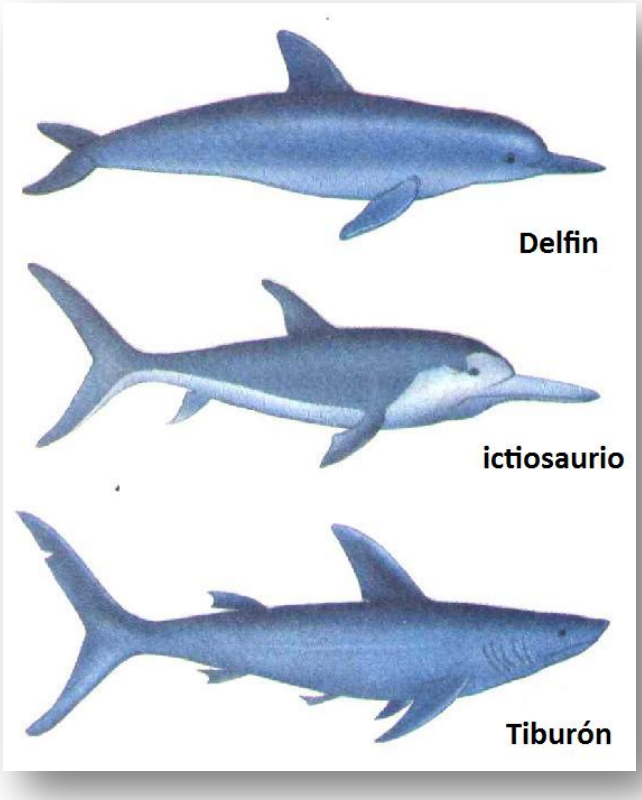

Figura 3 - Evolución convergente

En este sentido, y centrándonos en este caso en la función de locomoción, el estudio del fundamento neurológico ha facilitado el descubrimiento de estrategias que pueden ser aplicadas a diseños en robótica. La investigación de sistemas de locomoción bio-inspirados tiene grandes expectativas de futuro, pues existe un gran número de resultados novedosos en el estudio del control motor realizados en el sistema nervioso de seres vivos y que aún no se han estudiado para la robótica [1,2,3]. En particular, y centrándonos en el control motor, a la robótica le interesa el estudio de los generadores centrales de patrones (CPG en adelante) [4,5]. Los CPG son unas redes neuronales existentes en muchos seres vivos (en el caso de los mamíferos, se encuentra en la columna vertebral), que generan actividades rítmicas capaces de controlar neuronas motoras que se encargan de producir aquellos movimientos que requieran cierta periodicidad, robustez y/o precisión (tales como caminar, volar, nadar, reptar, etc.). Hay que hacer una mención especial a la autonomía que los CPG´s puede aportar a la robótica, pues son capaces de resolver situaciones inesperadas al introducir un feedback sensorial. Además, son redes autónomas, lo cual quiere decir, que no es preciso recibir un estímulo externo para producir un determinado patrón. Por otro lado, la información proveniente de los sentidos, son capaces de modular la actividad de los CPGs, con el fin de ajustarlo a las condiciones externas.

Debido a la complejidad de los CPGs en organismos más evolucionados, la mayoría de los estudios en redes neuronales en el campo de la neurociencia, han sido sobre CPGs de

organismos invertebrados. Recientes estudios demuestran que estos circuitos neuronales tienen como componente básico la inhibición mutua [5,6,7], y que están formados por las neuronas y las sinapsis que se producen entre éstas. Además, muestran una dinámica con diferentes escalas temporales, que rápidamente pueden producir una fuerte secuencia de activaciones [8]. Otra característica que se muestra en los circuitos neuronales es una dinámica invariante para presentar ritmos que sean robustos y flexibles al mismo tiempo [1, 2 ,9]. Se manifiesta también la existencia de ciertos códigos que permiten a las células transmitir información neuronal, conociendo la célula receptora cuál ha sido la célula emisora del mensaje [10,11].

Dentro del mundo de la robótica, cabe destacar la flexibilidad que puede proporcionar un diseño modular, y en el que se pueden estudiar diferentes técnicas de locomoción [12, 13]. En sí mismo, el diseño modular es bio-inspirado, pues en el proceso evolutivo se generan seres vivos compuesto por diferentes módulos (cochinilla, ciempiés, etc.). En un salto evolutivo, estos organismos pueden producir nuevos módulos idénticos al resto, o modificar cada uno de ellos de forma idéntica paralelamente. El diseño modular explota las ventajas de poder crear robots de diferentes tamaños y configuraciones de topología a partir de módulos homogéneos. La modularidad puede emplearse además

en sistemas distribuidos, donde se puedan diseñar patrones que sean reproducidos localmente en cada módulo, manteniendo una locomoción adecuada.

Los estudios realizados hasta la fecha de los CPGs en el entorno de la neurociencia, han aportado ideas en el ámbito de robótica, aplicando dichos conocimientos en robots en diferentes aspectos. Ejemplo de esto mismo es el desarrollo de un modelo del movimiento de la extremidad de un crustáceo bastante realista [16]; el desarrollo de una red neuronal artificial para controlar un robot hexápodo [17]; el estudio del control de aletas [18] y alas [19] realizando un análisis riguroso de la convergencia y estabilidad de los controladores; la locomoción bípeda empleando CPGs [20, 21]; la locomoción modular [22]; o los diseños de CPG genéricos no en línea y los métodos de optimización en línea [24, 25]

Sin embargo, en la investigación de CPGs, el campo de la neurociencia avanza mucho más que el campo de la robótica, pues en muchas ocasiones resulta más cómodo emplear osciladores para la locomoción. En este proyecto se defiende la flexibilidad y la capacidad de negociación de ritmos que el uso de CPGs realistas puede aportar a la robótica, así como una mayor autonomía.

Los CPGs que estudiaremos tendrán unas ciertas características para asegurarnos que se cumplen las bondades ya descritas. Debe ser una topología no abierta, es decir, que toda neurona del CPG recibe al menos una conexión de otra neurona de ese circuito. El CPG debe operar con varias escalas temporales, combinando dinámicas lentas y rápidas para generar los ritmos. Por último, decir que el mecanismo por el cual las neuronas se relacionan será mediante la competencia asimétrica entre ellas por generar las señales de control.

1.2 Objetivos

El objetivo de este proyecto es el estudio de CPGs realistas con el fin de conseguir configuraciones capaces de realizar distintas modalidades de locomoción con dos grados de libertad en el robot modular descrito anteriormente.

Decimos que tiene dos grados de libertad (o movimientos en 3 dimensiones) porque cada uno de los diferentes módulos que componen el robot, está posicionado con el eje del motor perpendicular respecto al anterior. Cuando se haga referencia al robot con un grado de libertad o con movimientos en 2 dimensiones, nos referiremos a aquel robot cuyos módulos están posicionados con el eje en paralelo respecto al anterior. A pesar de que lo único que cambia entre un robot y otro es el posicionamiento de los módulos, la complejidad y número de movimientos que puede realizar aumentan muchísimo.

A pesar de la complejidad que conlleva el estudio de este tipo de robots, todo puede simplificarse a la obtención de señales oscilatorias con diferentes fases y amplitudes utilizando los CPGs. Para ello es necesario diseñar la conectividad de los CPGs y combinar la riqueza dinámica de los modelos neuronales para conseguir las distintas modalidades de locomoción.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

Estado del arte: En este apartado, se presentarán los principios básicos de las neuronas desde el punto de vista de la biología, con el fin de detectar reglas básicas que constituirán el fundamento para el diseño del CPG. Además, se expondrá el modelo de mapa iterado, empleado con anterioridad, y los sistemas de locomoción de robots.

Diseño: En este apartado, se mostrará el diseño del sistema. Para ello, se empezará por tratar los diferentes movimientos que pueden efectuarse con la configuración elegida de los módulos. Este estudio de los movimientos, se hará con señales sinusoidales y se hablará tanto de los movimientos ya estudiados con anterioridad, como de movimientos obtenidos por primera vez. Además, se realizará una introducción del CPG que se usará en el proyecto, no con el fin de entender su funcionamiento en este punto, sino para representar las conexiones y las neuronas del CPG.

Desarrollo: En este apartado, nos centraremos en el funcionamiento del CPG. Se empezará explicando el programa desarrollado para tal fin con sus diferentes niveles (de módulo, neurona y modelo). Posteriormente, se llevará a cabo un estudio básico del funcionamiento de las neuronas, obteniendo las tres configuraciones básicas para la obtención de fases y un método para cambiar la amplitud. Finalmente, se expondrán tres modelos, correspondientes a cinco movimientos ya definitivos con resultados analizados por simulación.

Integración, Pruebas y resultados: En este apartado, se hablará de las pruebas realizadas con el robot y algunas medidas de su funcionamiento.

Referencias: Aquí se listarán las referencias de los artículos, libros y webs consultadas para la realización de este proyecto.

Glosario: Se expondrán en este apartado las siglas empleadas en el proyecto.

2 Estado del arte

2.1 Principios de la neurotransmisión química: Sinapsis

A la hora de hacer un diseño "bio-inspirado", es necesario conocer, aunque sólo sea en aspectos generales y sin entrar en profundidad, un poco de la materia a estudiar. En nuestro caso, vamos a hacer una introducción a la neurotransmisión química, para poder así entender mejor los mecanismos de conectividad entre neuronas.

La neurotransmisión química tiene lugar en las sinapsis, lugares especializados que conectan dos neuronas. Las neuronas están organizadas de forma que tanto pueden enviar información sináptica a otras neuronas como recibirla. Esto se consigue por un largo axón que se ramifica en fibras terminales dispuestas para hacer el contacto sináptico con otras neuronas (véase Figura 2).

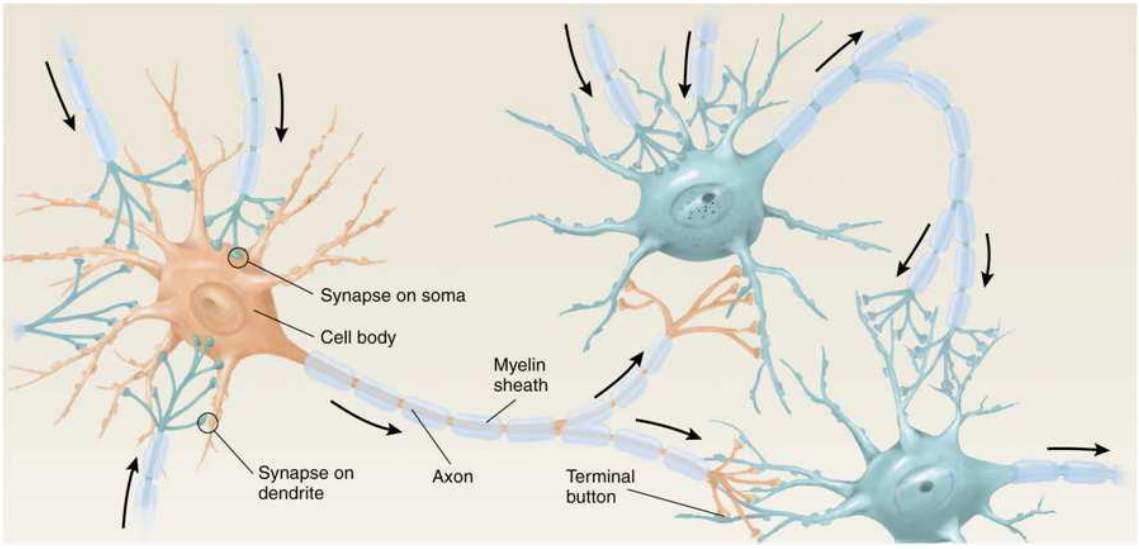

Figura 4 - Conexión de las neuronas

La neurotransmisión química puede describirse en tres dimensiones: el espacio, el tiempo y la función.

2.1.1 El espacio: el sistema nervioso anatómicamente concebido

Tradicionalmente, el sistema nervioso se ha representado como una serie de conexiones de "cables" entre neuronas, no muy diferente a millones de hilos telefónicos dentro de miles y miles de cables. A esta idea, se hace referencia cuando se habla del sistema nervioso "anatómicamente concebido". El cerebro anatómicamente concebido es así un diagrama complejo de hilos conductores, que transporta impulsos eléctricos allá donde "el hilo" tiene una conexión (es decir, una sinapsis).
Las neuronas envían impulsos eléctricos de una parte de la célula a otra parte de la misma célula, pero estos impulsos eléctricos no saltan directamente a otras neuronas. Las neuronas se comunican al lanzar una neurona un mensajero químico o neurotransmisor a

los receptores de una segunda neurona. Esto ocurre solamente en los lugares de conexiones sinápticas entre ellas. La comunicación entre neuronas es, principalmente, química, no eléctrica. Esto es, un impulso eléctrico en la primera neurona se convierte en una señal química en la sinapsis entre esta y una segunda neurona, en un proceso conocido como neurotransmisión química. Esto ocurre solamente en una dirección, desde el terminal del axón pre sináptico a uno cualquiera de una serie de sitios de una segunda neurona post sináptica.

2.1.2 El espacio: el sistema nervioso químicamente concebido

El mensajero químico enviado por una neurona a otra puede propagarse por difusión a otros lugares alejados de la sinapsis. Así, la neurotransmisión puede ocurrir en cualquier receptor compatible dentro del radio del neurotransmisor, de forma equivalente a la comunicación moderna con teléfonos celulares que funcionan dentro del radio de transmisión de una célula dada. Este concepto es llamado sistema nervioso "químicamente concebido" en el que la neurotransmisión ocurre en "soplos" químicos. Así, el cerebro es no solamente una colección de alambres, sino también una sofisticada "sopa química".

2.1.3 El tiempo: señales rápidas frente a señales lentas

Algunas señales del neurotransmisor son muy breves, ya que sólo duran unos milisegundos. Dos de los mejores ejemplos de señales rápidas son los de los neurotransmisores glutamato y acido gamma-aminobutirico (GABA). El glutamato es un neurotransmisor que estimula a las neuronas receptoras, mientras que el GABA es un mensajero que resulta en una inhibición. Ambos neurotransmisores actúan por medio de señales rápidas. Por otra parte, las señales de otros neurotransmisores pueden ser más largas, y durar muchos milisegundos e incluso varios segundos de tiempo. A veces, estos neurotransmisores que actúan más tiempo se llaman neuromoduladores, dado que las señales lentas pueden durar lo suficiente para persistir y modular una neurotransmisión ulterior producida por otro neurotransmisor. Así, una señal neuromoduladora de acción larga puede determinar el tono de una neurona, y puede influirla no solamente por su acción primaria propia, sino también por la acción modificadora sobre la neurotransmisión de un segundo mensaje químico enviado antes de que la primera señal se haya desvanecido. Ejemplos de neurotransmisores de señalización lenta son la norepinefrina, la serotonina y varios neuropéptidos.

2.1.4 Función

La tercera dimensión de la neurotransmisión química es la de la función, es decir, esa cascada de acontecimientos moleculares y celulares puestos en marcha por el proceso de señalización química. Un impulso eléctrico en la primera neurona se convierte en una señal química en la sinapsis por un proceso conocido como acoplamiento excitación-secreción.

Una vez que un impulso eléctrico invade el terminal del axón pre sináptico en la primera neurona, causa la liberación del neurotransmisor químico almacenado allí. El camino para que se produzca va siendo allanado por la síntesis previa y el almacenamiento del neurotransmisor en el axón terminal pre sináptico de la primera neurona. Los enzimas, los

receptores y otros materiales químicos son enviados allí desde el núcleo, el "centro de mando" neuronal, siendo transportados a lo largo del axón hasta los terminales que actúan como "oficinas de campo" de esa neurona por todo el cerebro. El neurotransmisor es empaquetado y almacenado en la neurona pre sináptica en vesículas, como una pistola cargada, lista para disparar (véase Figura 3). Estos procesos ocurren también con su propia escala temporal.

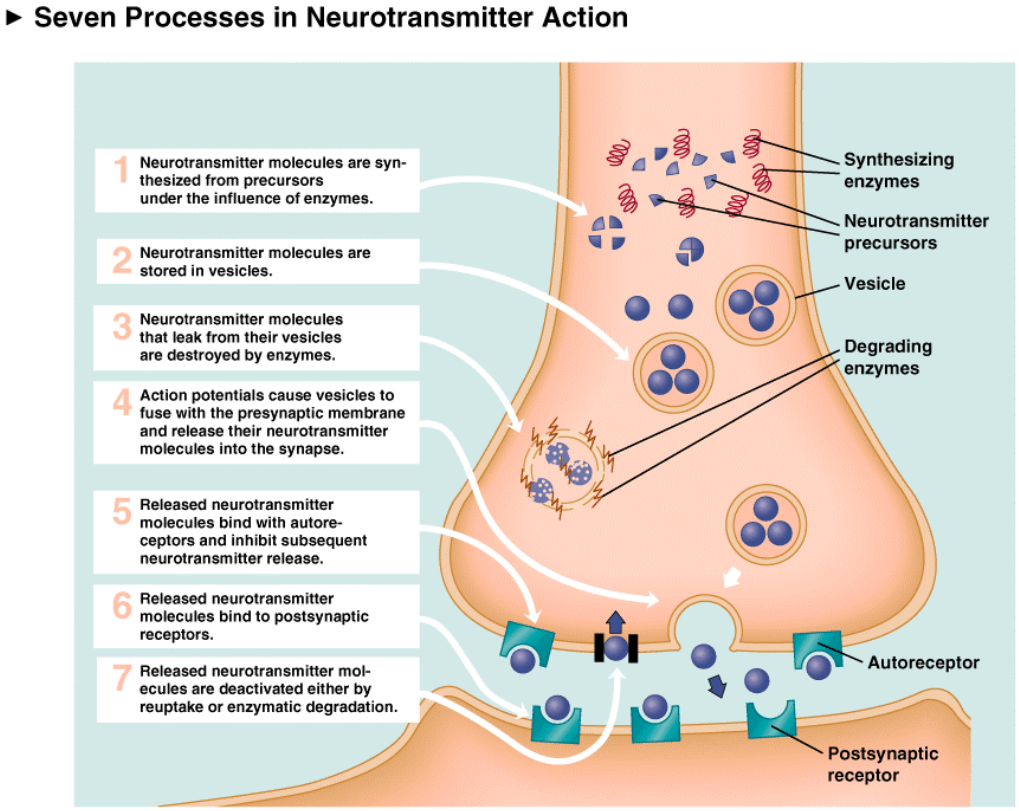

Figura 5 - Sinapsis

Una vez el neurotransmisor ha sido disparado desde la neurona pre sináptica, atraviesa la sinapsis buscando e impactando los lugares diana de los receptores, que son muy selectivos para ese neurotransmisor. La ocupación del receptor por el neurotransmisor que se une a lugares muy específicos es muy similar a las enzimas que se ligan a sustratos en sus sitios activos. El neurotransmisor actúa como una llave que encaja bastante selectivamente en una cerradura receptora. Esto inicia un proceso que reconvierte el mensaje químico en un impulso eléctrico en el segundo nervio. También, desencadena numerosas consecuencias bioquímicas en la segunda neurona.

La ocupación del receptor por el neurotransmisor –el primer mensajero- provoca que ocurran varios procesos intracelulares, comenzando con mensajeros adicionales dentro de la célula. El "segundo mensajero" es un producto químico intracelular creado por el primer mensajero neurotransmisor, al ocupar el receptor fuera de la célula en la conexión sináptica entre la primera y la segunda neurona. Los mejores ejemplos de segundos mensajeros son el adenosis monofosfato (AMP) y el fosfatidil inositol (PI). Algunos receptores están ligados a un tipo de segundo mensajero, y otros a diferentes segundos mensajeros.

2.2 Reglas básicas

A partir del estudio de la neurotransmisión se infieren una serie de reglas básicas a la hora de realizar un diseño bio-inspirado. Estas reglas son el principio de localidad, la presencia de distintas escalas temporales y la competencia entre neuronas[26].

2.2.1 Principio de localidad

No todas las conexiones son posibles, pues en organismos reales, las neuronas están conectadas en gran medida con sus vecinas y, por tanto, conexiones entre neuronas muy separadas o correspondientes a subsistemas distintos no tienen sentido. Una solución muy fácil para conseguir determinados movimientos del robot, era interconectar la neurona de cabeza con la de cola, pero fue descartada por no cumplir el principio de localidad para el diseño modular.

2.2.2 Escalas temporales

El CPG trabaja con varias escalas temporales. Se emplean pulsos muy rápidos a modo de disparo que emulan a los potenciales de acción de las neuronas junto con ondas depolarizantes lentas. Se ha demostrado que esta combinación resulta en una dinámica negociadora en las neuronas que revierte en la generación de ritmos flexibles y a la vez robustos [26] en el CPG. Adicionalmente, cuando se desee cambiar de movimiento habrá que modificar los parámetros de la sinapsis, teniéndose así un periodo de transición entre un movimiento y otro. Este cambio de los parámetros de la sinapsis se asemeja al cambio del modo de funcionamiento en organismos biológicos, provocado por la acción de los neuromoduladores [2,4]

2.2.3 Competencia

El mecanismo por el cual muchos tipos de neuronas se relacionan entre sí, no es más que la competencia mediante coctividad inhibitoria asimétrica para generar ritmos. Esto ocurre en la mayor parte de los CPGs conocidos [5,8]. Por lo tanto, cualquier otro mecanismo alternativo que se puede emplear para controlar los parámetros de amplitud y fase del CPG quedará fuera de lugar en este proyecto.

2.3 Redes Neuronales

Las Redes Neuronales Artificiales (ANNs de Artificial Neural Networks) fueron originalmente una simulación abstracta de los sistemas nerviosos biológicos, formados por un conjunto de unidades llamadas "neuronas" o "nodos" conectadas unas con otras. Estas

conexiones tienen una gran semejanza con las dendritas y los axones en los sistemas nerviosos biológicos [28,29].

El primer modelo de red neuronal fue propuesto en 1943 por McCulloch y Pitts, en términos de un modelo computacional de "actividad nerviosa". El modelo de McCulloch-Pitts es un modelo binario, y cada neurona tiene un escalón o umbral prefijado. Este primer modelo sirvió de ejemplo para los modelos posteriores de John Von Neumann, Marvin Minsky, Frank Rosenblatt, y muchos otros.

Una primera clasificación de los modelos de ANNs podría ser, atendiendo a su similitud con la realidad biológica:

- Los modelos de tipo biológico. Este comprende las redes que tratan de simular los sistemas neuronales biológicos, así como las funciones sensoriales, de coordinación o motoras. En este trabajo se emplearán este tipo de redes.

- El modelo dirigido a aplicación (por ejemplo clasificación de patrones, predicción, optimización, etc). Estos modelos no tienen por qué guardar similitud con los sistemas biológicos. Sus arquitecturas están fuertemente ligadas a las necesidades de las aplicaciones para las que son diseñados.

2.3.1 Redes Neuronales de tipo Biológico

Se estima que el cerebro humano contiene más de cien mil millones de neuronas y sinapsis. Estudios sobre la anatomía del cerebro humano concluyen que en promedio hay más de 1000 sinapsis a la entrada y a la salida de cada neurona. Es importante notar que, aunque el tiempo de conmutación de la neurona (unos pocos milisegundos) es casi un millón de veces menor que en las actuales elementos de las computadoras, ellas tienen una conectividad miles de veces superior que las actuales supercomputadoras.

El objetivo principal de las redes neuronales de tipo biológico es desarrollar un elemento sintético para verificar las hipótesis que conciernen a los sistemas biológicos. En otros contextos, estas redes se utilizan para incorporar elementos de inspiración en un diseño de ingeniería.

Las neuronas y las conexiones entre ellas (sinapsis) constituyen la clave para el procesado de la información.

La mayor parte de las neuronas poseen una estructura de árbol llamadas dendritas que reciben las señales de entrada procedentes de otras neuronas a través de las uniones llamadas sinápsis.

Hay tres partes en una neurona:

1. El cuerpo de la neurona,

2. Ramas de extensión llamadas dendritas para recibir las entradas, y

3. Un axón que lleva la salida de la neurona a las dendritas de otras neuronas.

En general, una neurona envía su salida a otras por su axón. El axón lleva la información por medio de diferencias de potencial. Este proceso es a menudo modelado como una regla

de propagación representada por la función de red. La neurona recoge las señales por su sinapsis sumando todas las influencias excitadoras e inhibidoras. Si las influencias excitadoras positivas dominan, entonces la neurona genera una señal de salida y manda este mensaje a otras neuronas..

2.3.2 Redes Neuronales para aplicaciones concretas

Las ANNs dirigidas a aplicación, están en general poco ligadas a las redes neuronales biológicas. Dado que el conocimiento que se posee sobre el sistema nervioso en general no es completo, se han de definir otras funcionalidades y estructuras de conexión distintas a las vistas desde la perspectiva biológica. Las características principales de este tipo de ANNs son las siguientes:

- Auto Organización y Adaptatividad: Utilizan algoritmos de aprendizaje adaptativo y auto organización, por lo que ofrecen posibilidades de procesado robusto y adaptativo.

- Procesado No Lineal: Aumenta la capacidad de la red de aproximar, clasificar y su inmunidad frente al ruido.

- Procesado paralelo: Normalmente se usa un gran número de células de procesado por el alto nivel de interconectividad.

Estas características juegan un importante papel en las ANNs aplicadas al procesado de señal e imagen. Una red para una determinada aplicación presenta una arquitectura muy concreta, que comprende elementos de procesado adaptativo masivo paralelo combinadas con estructuras de interconexión de red jerárquica.

2.3.3 Taxonomía de las Redes Neuronales

Existen dos fases en toda aplicación de las redes neuronales: la fase de aprendizaje o entrenamiento y la fase de prueba. En la fase de entrenamiento, se usa un conjunto de datos o patrones de entrenamiento para determinar los pesos de las conexiones que definen el modelo neuronal. Una vez entrenado este modelo, se usará en la llamada fase de prueba o funcionamiento directo, en la que se procesan los patrones de prueba que constituyen la entrada habitual de la red, analizándose de esta manera las prestaciones definitivas de la red.

- Fase de Prueba: Los parámetros de diseño de la red neuronal se han obtenido a partir de unos patrones representativos de las entradas que se denominan patrones de entrenamiento. Los resultados pueden ser tanto calculados de una vez como adaptados interactivamente, según el tipo de red neuronal, y en función de las

ecuaciones dinámicas de prueba. Una vez calculados los pesos de la red, los valores de las neuronas de la última capa, se comparan con la salida deseada para determinar la validez del diseño.

- Fase de Aprendizaje: Una característica de las redes neuronales es su capacidad de aprender. Aprenden por la actualización o cambio de los pesos sinápticos que caracterizan a las conexiones. Los pesos son adaptados de acuerdo a la información extraída de los patrones de entrenamiento nuevos que se van presentando.

Atendiendo al tipo de entrenamiento, una posible taxonomía de redes neuronales artificiales clásiscas es:

Tabla 1 - Ejemplos de redes neuronales artificiales

Fijo	**No supervisado**	**Supervisado**
Red de Hamming	Mapa de características	Basadas en decisión
Red de Hopfield	Aprendizaje competitivo	Perceptrón
		ADALINE (LMS)
		Perceptrón Multicapa
		Modelos Temporales Dinámicos
		Modelos Ocultos de Markov

Las redes neuronales se clasifican comúnmente en términos de sus correspondientes algoritmos o métodos de entrenamiento: redes de pesos fijos, redes no supervisadas, y redes de entrenamiento supervisado. Para las redes de pesos fijos no existe ningún tipo de entrenamiento.

En este proyecto, se emplearán modelos de CPGs que corresponden a redes neuronales biológicas. Para ello cada neurona vendrá representada por un mapa iterado que simula el potencial de membrana de la neurona. La conectividad se basará fundamentalmente en la inhibición mutua entre neuronas.

2.4 Locomoción en robots

Si bien es cierto que resulta más sencillo controlar un robot simple como el usado en este proyecto con señales sinusoidales, el estudio de CPGs puede contribuir a una mayor autonomía en el control y a una robustez adicional frente a situaciones imprevistas [26]

Actualmente, se están empleando CPGs para conseguir movimientos mucho más complejos, como pueden ser por ejemplo, caminar [14,15]. Como ya se ha descrito, en los humanos y los animales, los movimientos periódicos y repetitivos como los asociados a caminar o a respirar, son controlados por generadores centrales de patrones. En los sistemas bio-inspirados, se simulan estos CPGs para ejercer el control de la locomoción.

. Variando los parámetros de un CPG se pueden obtener una infinidad de movimientos, es decir, la capacidad para generar distintos ritmos está ahí, pero es necesario encontrar los parámetros y el régimen adecuado de funcionamiento. Esto es especialmente relevante en un modelo de CPG que incorpora una dinámica bio-inspirada tanto para las neuronas como para las conexiones.

En el desarrollo de este proyecto, se propone un modelo de CPG global, con las conexiones lógicas que puede tener éste, y a partir de ahí, se estudiará qué valor deben tener las variables para que se puedan realizar cada uno de los movimientos esperados. Otra forma de hacer este CPG global es obtener un CPG para cada movimiento, y posteriormente, combinar todos ellos para obtener el CPG global. Con esta segunda manera de trabajar se descubre que la mayoría de las conexiones entre un CPG y otro son idénticas, y además, que son pequeñas variaciones las que definen un movimiento u otro.

Esto, añadido a las otras ventajas de emplear CPGs ya expuestas (como la recuperación ante errores), hace que el uso de CPGs en la locomoción de robots sea un campo de estudio interesante a largo plazo.

2.5 Aplicaciones reales del estudio de los CPGs

Pese a que el objetivo de este proyecto es el estudio de los CPGs para la el movimiento de robots, dentro de los sistemas bio-inspirados, el campo de estudio de los CPGs es mucho más amplio.

Se sabe que en la columna vertebral de mamíferos hay CPGs que controlan el movimiento de las extremidades para generar movimientos. Al seccionarse la columna vertebral por algún lado, las señales del cerebro dejan de llegar a las extremidades, pero estos CPGs siguen estando ahí de manera inactiva. El estudio de estos CPGs podría hacer que de manera externa se devolviera la movilidad a las piernas a personas que han quedado inválidas por alguna lesión en la médula espinal.

Esto que a simple vista parece ciencia ficción, está teniendo ya buenos resultados. En la revista "Nature Neuroscience" se publicó que investigadores de Estados Unidos, Rusia y Suiza utilizaron fármacos y estímulos eléctricos para que animales, cuya médula espinal había sido cortada, lograran correr en una rueda giratoria. Estos movimientos eran casi idénticos a los pasos normales, sin embargo, no se trataban de movimientos controlados por la mente[30].

Es bien sabido que si se aplica una corriente eléctrica en el nervio, justo debajo de la herida, se produce contracción del musculo. Sin embargo, el acto de caminar requiere una secuencia compleja de este tipo de contracciones que deben ser producidas en el momento correcto, de manera que las piernas puedan soportar el peso del cuerpo y moverlo hacia delante. Estos patrones de señales eléctricas que los músculos utilizan para caminar (señales motoras) los controla la médula. Es decir, el cerebro da la orden de caminar y los CPGs de la médula generan las señales motoras.

Al quedar la medula seccionada se pierde la información del cerebro, y estas neuronas, pasan a estar en estado durmiente.

La investigación, por tanto, consiste en simular la acción del cerebro usando estímulos farmacológicos y eléctricos aplicados a la médula espinal, y transformar a estas neuronas de un estado durmiente a un estado totalmente funcional. De este modo, se logra estimular los circuitos motores no utilizados para poder activar los CPG y producir el movimiento de los pasos en las patas.

Así, a pesar de no tener conexión entre el cerebro y los músculos, se ha logrado que ratas caminen tras haberles cortado la médula. Después de que se lograse promover la locomoción en el aparato de ejercicios, una semana después de la lesión, se entrenaron a los animales durante 20 minutos cada dos días. Tras dos meses de rehabilitación, las ratas paralizadas lograron caminar cargando todo el peso de su cuerpo. Sin embargo, los movimientos no eran voluntarios, sino causados por los fármacos y estímulos eléctricos.

Esta misma técnica podría provocar niveles importantes de control motor en las piernas, y podría utilizarse con aparatos de neuroprótesis que se podrían implantar en el cerebro para ayudar a reparar la lesión de la médula espinal. Sin embargo, aún queda mucha investigación hasta que esto que se ha conseguido en ratas llegue a humanos, pero al menos ya se sabe que es posible. Es aquí donde se manifiesta la importancia de estudiar CPGs de otros animales más simples, como en este proyecto, pues este estudio ayudará a conocer CPGs complejos.

3 Diseño

3.1 Simulación con señales sinusoidales:

3.1.1 Movimientos básicos:

A pesar de que la finalidad del proyecto es que el robot sea capaz de moverse utilizando CPGs en vez de señales sinusoidales, es interesante replicar el estudio del movimiento con señales sinusoidales como primer paso, pues después solo habrá que ver cómo los CPGs pueden variar las fases y amplitudes entre los distintos módulos.

El robot diseñado por Juan González consta de 8 módulos idénticos. Para diferenciar los dos planos en los que se separan sus movimientos se ha elegido hacer cuatro módulos en blanco y otros cuatro módulos en negro. Cada módulo está formado por piezas de plástico, un motor Futaba S3003 y los tornillos para unirlos. Las piezas se han encargado al servicio de apoyo a la investigación de la UAM SEGAINVEX. Las piezas de cada módulo están diseñadas para que los motores Futaba encajen perfectamente, por lo tanto, cuesta separarlo una vez unidos. Antes de encajar los motores en las piezas, hay que llevarlos a la posición "0" (es decir, cero grados).

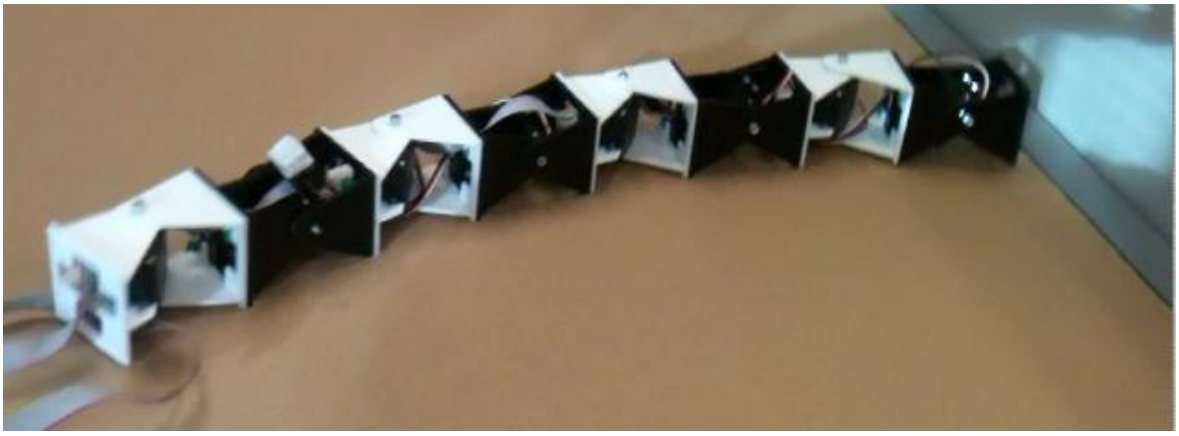

Figura 6 – Robot construido para este proyecto

Una vez encajado, se observa un ligero error respecto al cero en el módulo. Para que este error no afecte al correcto funcionamiento del robot, se ha medido utilizando una regla de nivel. Posteriormente, se corregirá ese error con una simple resta. Los módulos se han etiquetado y numerado de uno a ocho y los errores obtenidos son:

Módulo	Error(º)
1	6
2	0
3	5
4	1
5	-1
6	-3
7	6
8	-2

Hay que observar que el máximo error en término absoluto es de 6º. Esto solo afectará al correcto funcionamiento del robot cuando se quiera llegar a los ángulos máximos que permiten los motores (es decir, disminuye 6º el margen dinámico de los motores).

Una vez montados todos los módulos, etiquetados y medido su error se procede a conectarlos entre sí. Para conectar los módulos se ha escogido hacerlo mediante bridas, cuatro para conectar cada módulo con el adyacente. La elección de usar bridas en vez de tornillos se debe a que hace más fácil el montaje y posibles cambios (las bridas son de un solo uso y baratas). Además se utilizan estos lazos para agarrar los cables de los motores. Para la conexión de los módulos no se ha elegido la misma numeración que la de los módulos. Ésta es la correspondencia (desde la cabeza a la cola) de los módulos con su respectivo número.

Módulos: Cabeza ➔ 7-4-3-2-6-1-8-5 ➔ Cola

El siguiente paso es el cableado de los motores a la placa. Para ello, he partido de la información que se muestra en la web [31]sobre cómo conectar los motores al puerto B de la placa y obtengo estas dos matrices de interconexión (donde el rojo es Vcc y el negro es tierra)

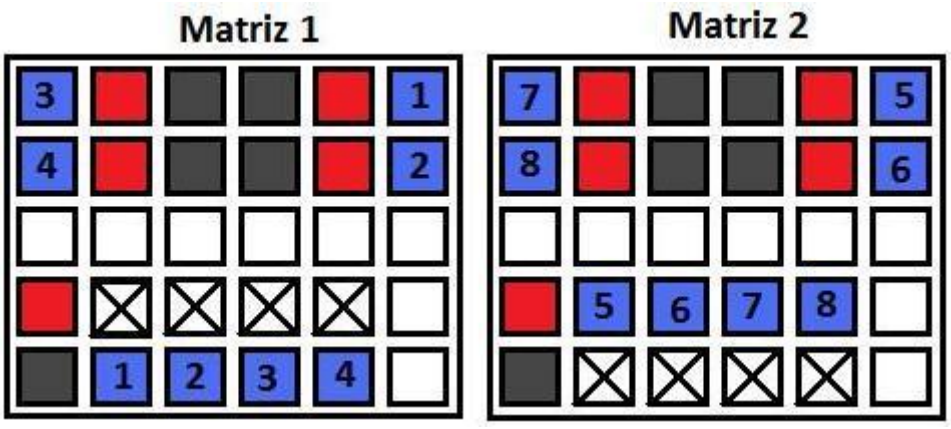

Figura 7 – Placa de conexiones.

Ambas matrices de interconexión han sido creadas partiendo de un PCB con taladros metalizados, en el cual se han soldado las conexiones descritas en el dibujo. En un principio han dado bastantes problemas estas dos matrices de interconexiones, pues las soldaduras se saltaban al poner el robot en funcionamiento. Tras una revisión de Guillermo González (profesor de robótica) decidí volver a hacer las matrices de interconexión, pues, debido a mi inexperiencia soldando, había hecho "soldaduras frías", las cuales no soportaban los movimientos del robot. Estos nuevos circuitos fueron recubiertos de silicona protectora para que aguanten mejor los movimientos mecánicos a los que son expuestos.

Una vez montado el robot, era la hora de hacer una primera prueba de funcionamiento de los motores. Para mi sorpresa la matriz de conexiones no correspondía a la realidad. Las alternativas para solucionar este problema eran volver a hacer las matrices de interconexión (con lo cual el trabajo realizado antes no valdría para nada), cambiar las salidas de los pines de la placa o hacer un cambio enteramente software en la asignación de los motores (solución más sencilla que implica poner unas líneas más de código, pero que debido al lento funcionamiento del robot no afecta para nada a la velocidad). Se ha optado por esta última opción, y además, se ha contrarrestado el error de cada módulo. A continuación, aparece el resultado (en python):

```
if motor == 1:
    motorr = 7
    error = 6
elif motor == 2:
    motorr = 5
    error = 1
elif motor == 3:
    motorr = 3
    error = 5
elif motor == 4:
    motorr = 1
    error = 0
elif motor == 5:
    motorr = 2
    error = -3
elif motor == 6:
    motorr = 4
    error = 6
elif motor == 7:
    motorr = 6
    error = -2
else:
    motorr = 8
    error = -1
```

El siguiente problema fue que los motores no respondían con suficiente fuerza para algunos movimientos. Esto se resolvió acortando el bus, y aunque en algunos casos no tiene fuerza suficiente para elevar todo el robot (cuando tienen que girarse los módulos centrales) ya posee la suficiente fuerza para los movimientos naturales del robot.

Ya con el robot funcionando correctamente, es hora de probarlo usando las señales sinusoidales para comprobar si es capaz de realizar todos los movimientos que ya con anterioridad salían con el simulador. Los resultados han sido bastante buenos. Las únicas diferencias que se observan entre el funcionamiento real y en la simulación es en los extremos en los cuales tiene que cambiar de movimiento, que no son tan exactos (hay regiones de funcionamiento en los que no hace nada claro) pero es normal, pues se ha pasado de un modelo en el cual se controlan un número finito de variables a algo real.

Como conclusión, sólo decir que ha sido bastante sencillo el montaje del robot gracias a la web de Juan González y casi todo ha salido a la primera. Ahora que el robot funciona correctamente es hora de seguir el trabajo y probarlo con los modelos de CPGs.

Toda la información necesaria la obtenemos en el artículo [27] y la presentación [31].

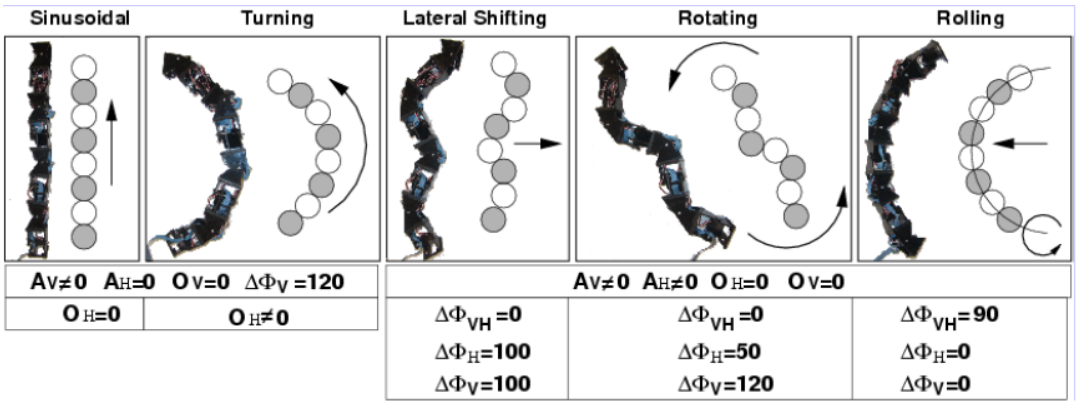

Figura 8 - Movimientos del robot (Figura adaptada de [27]).

En la Figura 4 se detalla cómo variando únicamente ocho parámetros se consiguen hacer todos los movimientos, donde:

AV=Amplitud Vertical
AH=Amplitud Horizontal
ΔϕV=Fase Vertical
ΔϕH=Fase Horizontal
ΔϕVH=Fase entre planos
OV=Constante Vertical
OH=Constante Horizontal

El trabajo que habrá que realizar con los CPGs no será sino conseguir crear señales sinusoidales con esas características de amplitudes, fases y constantes iniciales.

3.1.2 Búsqueda de nuevas modalidades de locomoción

El hecho de que se puedan conseguir todas las modalidades de locomoción descritas variando esas siete variables me ha animado a buscar, mediante simulaciones, nuevos tipos de movimientos más eficientes. Debido a que las combinaciones con siete variables eran muy grandes, se ha dejado de lado a las constantes iniciales y se ha trabajado solo con las otras cinco variables.

En el caso de hacer la simulación para una sola dimensión (por ejemplo, para que la amplitud horizontal obtenga mayor velocidad de desplazamiento), se ha de decir que se obtiene de forma casi inmediata.

1. Hacer un array con todos los puntos en donde queremos estudiar la variable AH (puntos equiespaciados entre un máximo y un mínimo).
2. Con un programa (en C++) obtener todos los vectores de movimientos resultantes de cada uno de los parámetros (debidamente etiquetados para saber qué vectores corresponden a cada parámetro).
3. Modificar el simulador para que a partir de unos vectores de movimientos devuelva el desplazamiento realizado.
4. Con un programa (Matlab) obtener los máximos locales correspondientes al desplazamiento en función de cada parámetro.

Modificar el simulador es el punto más trabajoso de todos, porque implica comprender el funcionamiento del simulador. De este modo, lo primero ha sido suprimir la parte gráfica, pues es la que más tiempo de procesamiento necesita. A continuación, se han añadido una serie de etiquetas que indican el comienzo de una simulación, el fin de la misma... Para ello, ha sido necesario que las condiciones volviesen a su origen tras cada simulación, es decir, que la posición inicial, velocidad, etc volviesen al punto de partida, para que así una simulación no afectase a la siguiente. Por último, tras cada simulación se ha obtenido en un fichero con posiciones en las que finaliza el robot en cada simulación.

En el caso en el que se trabaja con cinco parámetros en vez de con uno, hay que introducir una serie de cambios:

1. El array que antes era de una dimensión ahora es de cinco dimensiones (para recorrerlo hacen falta cinco bucles “for” anidados).
2. El programa en Matlab que antes detectaba máximos de forma sencilla ahora debe detectar los máximos comparando cada punto con todos los adyacentes (en 5 dimensiones). Para evitar resultados “raros”, es decir, poco estables, (debido a que el simulador no es perfecto) se ha hecho un filtro de mediana que elimina todos esos máximos con amplitudes desproporcionadamente grandes con los vecinos.

Para esta simulación, ha sido necesario dejar el ordenador trabajando dos días seguidos, pues al aumentar el número de dimensiones, el tiempo necesario crece exponencialmente (ha habido un total de 52.488 simulaciones). Casi todo el tiempo necesario ha sido

utilizado por el simulador, que a pesar de haberle quitado la parte gráfica, aún tiene demasiada carga computacional.

Debido al gran número de máximos locales que se han obtenido, sólo se han seleccionado los que están por encima del 80% del desplazamiento máximo. Los resultados obtenidos han sido los siguientes:

Tabla 2 - Nuevos movimientos

Desplazamiento relativo al máximo	Dirección	AV	AH	FV	FH	FVH
100	-88	80	80	135	20	39
99	88	70	90	34	156	156
99	88	50	90	101	20	20
99	-84	80	60	67	156	156
99	80	70	70	67	156	39
97	90	50	90	101	39	117
97	82	80	90	101	39	156
96	83	60	90	34	20	20
94	74	50	60	67	20	78
93	82	70	90	67	176	20
93	67	50	90	67	20	59
92	73	90	20	101	59	78
92	69	60	90	101	20	176
92	-86	70	80	34	20	176
90	86	90	90	34	156	156
89	88	70	70	67	20	78
89	-90	90	90	34	20	20
88	85	90	90	34	59	98
85	-69	40	90	34	20	156
85	77	90	90	34	59	59
82	-86	80	90	34	156	117
81	-78	90	90	34	20	137
81	75	70	80	67	59	20
80	-76	90	80	67	117	59
80	79	90	70	67	59	176
80	-80	60	80	34	78	78

La tabla anterior debe interpretarse como los mejores resultados de todas las simulaciones hechas, donde cada columna representa el desplazamiento máximo (ponderado al mayor de todos en tanto por ciento), la dirección en grados, las amplitudes que deben adoptar los módulos horizontales y verticales, y las diferentes fases en grados.

3.2 CPG Diseñado

Más adelante, se explicarán las razones por las que se ha propuesto este CPG. La comprensión de los gráficos puede resultar muy complicada debido al gran número de conexiones entre neuronas. El objetivo de este apartado será pues, mostrar el CPG que se va a emplear en este estudio, y si bien no se entrará aún en el porqué de esta configuración, sí que se intentará obtener un modelo de representación claro y simplificado para a partir de ahí hacer la comprensión más sencilla.

Éste es pues, el modelo de conexiones sobre el que vamos a trabajar:

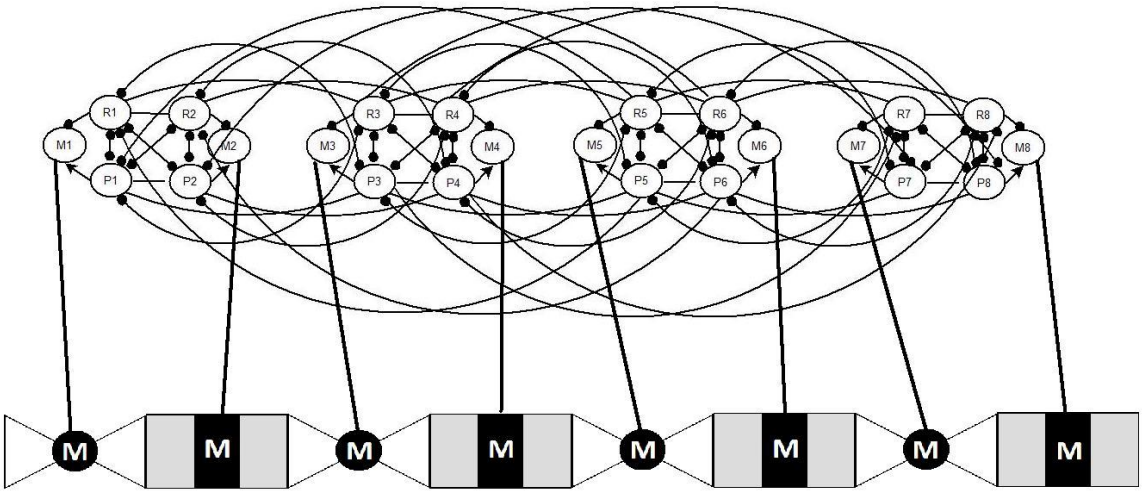

Figura 9 - CPG global. Se muestra la disposición alternante de planos perpendiculares de los módulos del robot. Los motores están señalados con la letra M.

En la Figura 5 se muestran 24 neuronas (ocho neuronas R, ocho neuronas P y ocho neuronas M). Cada neurona está representada por un círculo con el tipo en su interior y una numeración que corresponde con el número de motor sobre el que actúa.

Las neuromas M son las llamadas "motoneuronas" y son responsables de generar la señal para los motores. Las neurona R y P son las que forman propiamente la red (tienen al menos una conexión de entrada y otra de salida) y son las llamadas neuronas "remotoras" y neuronas "premotoras" [26]

Las líneas que unen unas neuronas con otras son las conexiones. Para simplificar se han unido las conexiones de dos en dos (pues suelen ser simétricas), indicando en los extremos qué tipo de conexión se trata.

Las conexiones terminadas en círculo representan conexiones de tipo inhibitorias, mientras que las terminadas en flecha ilustran conexiones excitatorias. Las que no tienen nada significa que pueden ser conexiones de tipo bien excitatorias bien inhibitorias, o por el contrario, no haber conexión en ese sentido (peso cero).

En el caso de haber un movimiento para el cual la conexión entre dos neuronas tenga peso cero, se eliminará la conexión para simplificar, pues dicha conexión no aportará información.

A pesar del número de conexiones tan grande, no se ha considerado aportar a cada conexión un pesoparticular, sino que se agrupan en varios tipos, cada uno de ellos con un peso diferente.

A continuación, se muestran las agrupaciones que se hacen de las conexiones y su respectiva simplificación, que será usada en adelante para una mejor comprensión. Para identificar cada configuración, se han agrupado aquellas conexiones cuyo peso sináptico es idéntico.

Conjunto de Pesos 1:

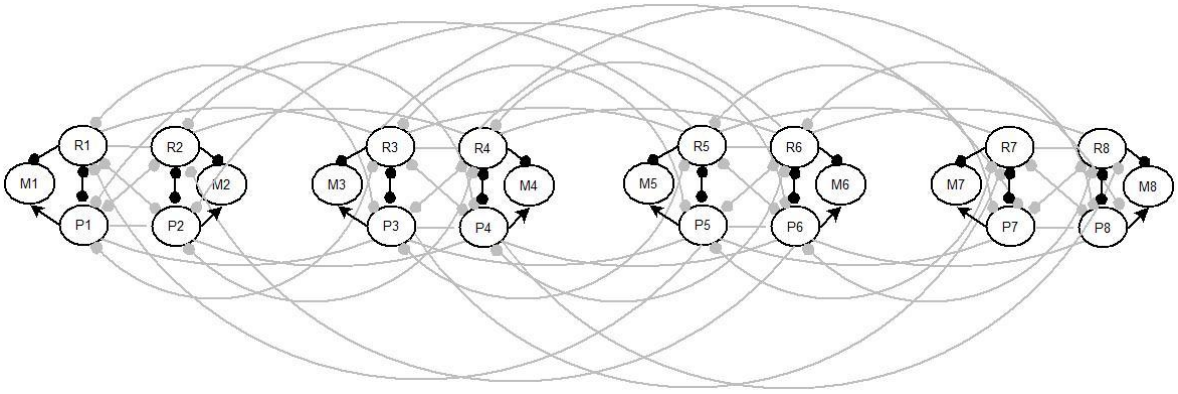

Figura 10 - Peso 1

Este es un caso particular, pues no todas las conexiones señaladas tienen el mismo peso, sino que los peso serán diferentes dependiendo si conectan una neurona P con una neurona R, o si conecta R con M o si conecta P con M, pero siempre idénticas modulo a modulo.

En este caso no es una única ganancia, sino que se trata de tres. La ganancia que une las neuronas R con las neuronas P es un parámetro del sistema llamado Gsyn (peso de sinapsis) y es el caso más básico que permite sincronizar dos neuronas. La neurona M por su lado obtiene contribuciones de las neuronas R y P, pero no aporta ninguna. La función de esta neurona es obtener un ritmo sinusoidal relacionado con sus dos neuronas asociadas. A cada trío de neurona R, P y M se le ha asociado a un módulo, y se ha agrupado en un cluster ficticio llamado C para simplificar su comprensión.

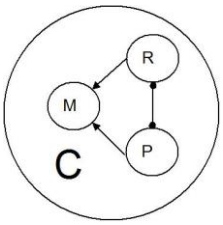

Figura 11 – Agrupación de neuronas en el cluster C para facilitar la descripción

Por lo tanto, el cluster C queda establecido a partir de ahora como las neuronas R, P y M y sus cuatro conexiones (RP, PR, PM, RM). El sistema resultante queda pues:

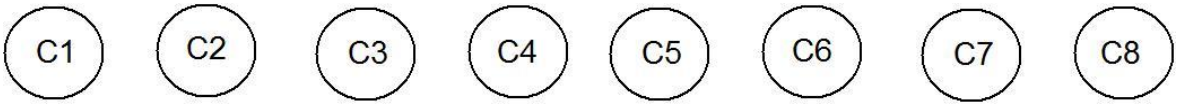

Figura 12 - Resultado Peso 1

En la figura 8, no se muestra ninguna conexión, pues todas ellas están dentro de los módulos ocultos a la vista.

Peso 2:

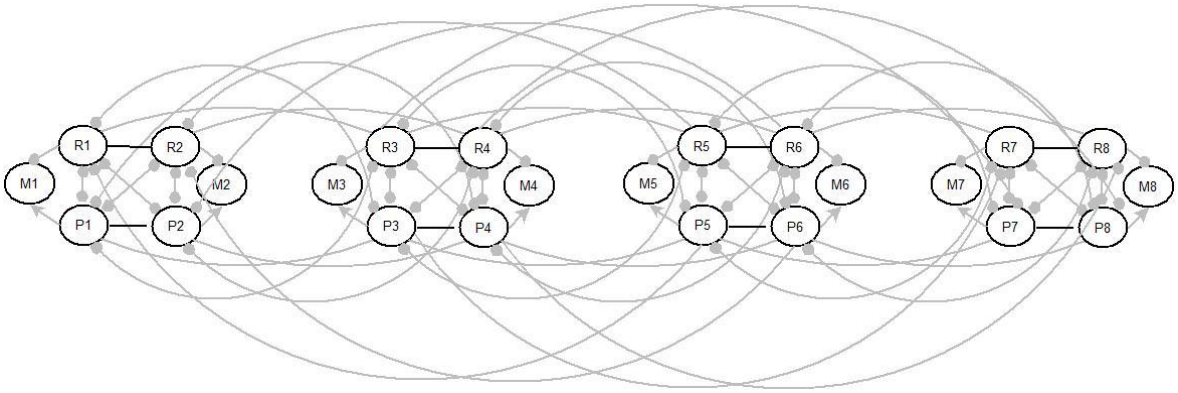

Figura 13 - Peso 2

Esta ganancia corresponde a la que une los bloques horizontales con verticales con correspondencia RR o PP. Posteriormente, se verá que dicha ganancia controla la fase entre los planos horizontal y vertical. Las conexiones podrán ser excitatorias o inhibitorias.

El sistema resumido quedará como:

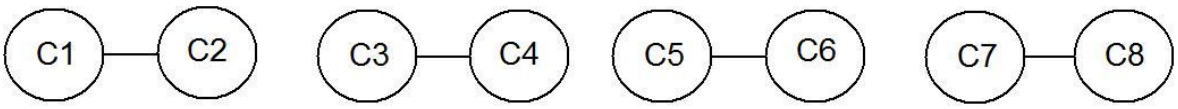

Figura 14 - Resultado Peso 2

Peso 3:

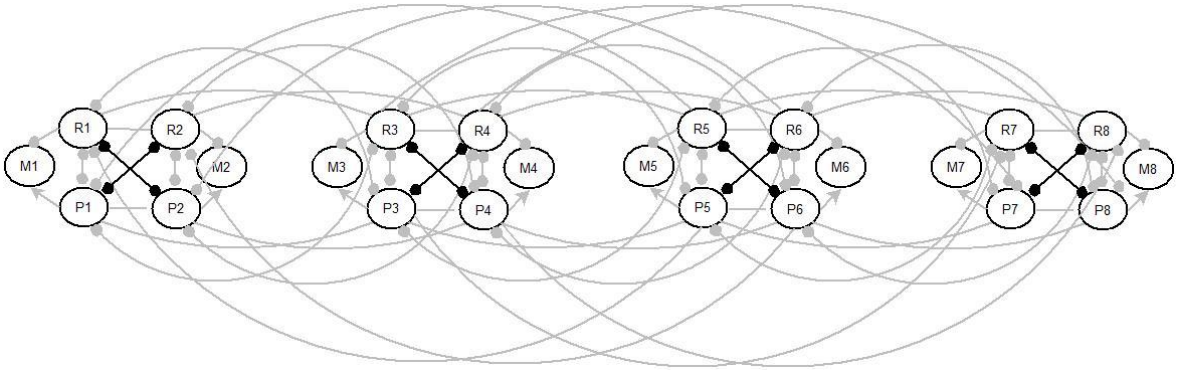

Figura 15 - Peso 3

Este caso también interconecta los bloques horizontales con los verticales, pero en este caso con conexiones RP y PR. Estas conexiones siempre serán inhibitorias, y su función es dar una fase de cero grado entre los dos planos.

El sistema resumido quedará como:

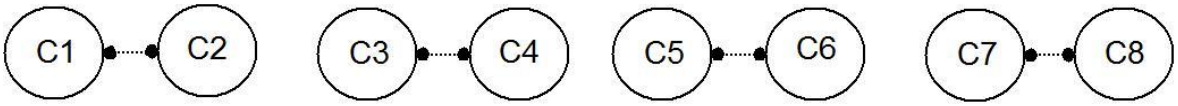

Figura 16 - Resultado Peso 3

Nota: Las ganancias 2 y 3 son incompatibles entre sí, siempre debe haber una de ellas igual a cero.

Peso 4:

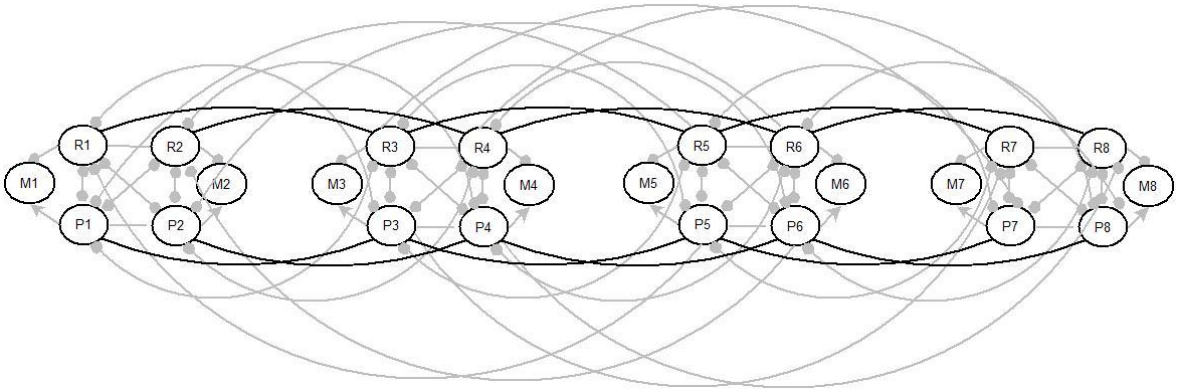

Figura 17 - Peso 4

Esta ganancia corresponde a la que conecta los módulos del mismo plano con correspondencia RR y PP. La conexión así mismo podrá ser excitatoria o inhibitoria, y valdrá para obtener una fase en el mismo plano. Con esta ganancia se controla la fase dentro del mismo módulo.

El sistema resumido quedará como:

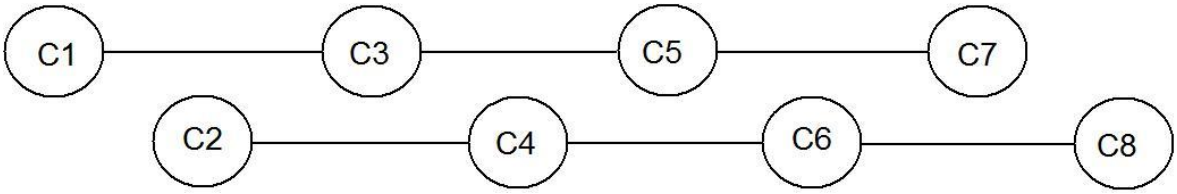

Figura 18 - Resultado Peso 4

Peso 5:

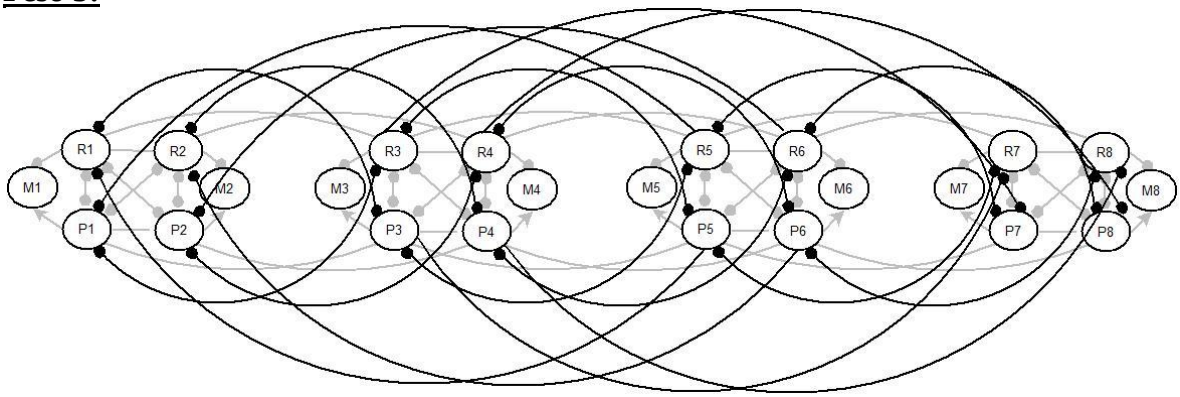

Figura 19 - Peso 5

Esta ganancia corresponde a la conexión de los bloques del mismo plano con correspondencia RP y PR. En este caso las conexiones siempre serán inhibitorias y valdrá para obtener una fase de cero grados en el mismo plano.

El sistema resumido quedará como:

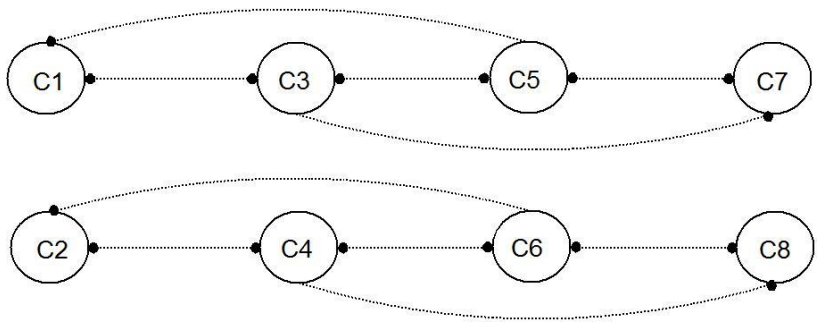

Figura 20 - Resultado Peso 5

La razón para que los módulos C1, C2, C7 y C8 reciban señal de neuronas que no son sus vecinas inmediatas, es que todas obtengan la misma contribución (dos entradas).

Nota: Las ganancias 4 y 5 no pueden estar activas al mismo tiempo, siempre debe haber al menos una igual a cero.

4 Desarrollo

4.1 Modelo neuronal

4.1.1 Descripción programa

Debemos modelar un sistema que simule una serie de neuronas y sus interconexiones.

Más adelante, se estudiarán las ecuaciones detenidamente para la obtención de resultados, pero es importante que el programa que lo ejecute sea lo más flexible y sencillo posible, para poder hacer así pruebas sin complicaciones. En un diseño Top-down partimos de tres niveles: Nivel de módulo, nivel de neurona y nivel de modelo.

4.1.1.1 Nivel de módulo

En este nivel se trabajan con módulos y conexiones al más alto nivel entre ellos. Los módulos usados son los descritos en el apartado 3.2 (módulos C), compuestos por el trío de neuronas R-P-M.

Un ejemplo de conexiones posibles es el siguiente:

```
for (int i= 0; i<TamCPG; i++){
        if(i==0){ //                * 1º            * 2º             * 3º           * 4º             * 5º
                CH[0].Avanzar(      &(CH[1])       ,NULL            ,&(CV[0])      ,NULL            ,NULL);
                CV[0].Avanzar      (&(CV[1])       ,NULL            ,NULL          ,&(CH[0])        ,NULL);
        }else if(i== TamCPG-1){ //  * 1º         * 2º               * 3º           * 4º             * 5º
                CH[TamCPG-1].Avanzar(NULL          ,&(CH[TamCPG-2]),&(CV[TamCPG-1]),NULL           ,NULL;
                CV[TamCPG-1].Avanzar(NULL          ,&(CV[TamCPG-2]), NULL          ,&(CH[TamCPG-1]) ,NULL)
        }else{//                    * 1º           * 2º             * 3º           * 4º             * 5º
                CH[i].Avanzar(      &(CH[i+1])     ,&(CH[i-1])      ,&(CV[i])      ,NULL            ,NULL);
                CV[i].Avanzar(      &(CV[i+1])     ,&(CV[i-1])      ,NULL          ,&(CH[i])        ,NULL);
        }

}
```

Para conseguir todas las conexiones descritas en el apartado 3.2, se han creado dos funciones de avanzar, una que conecta las neuronas con su homólogo (P-P y R-R) y otra que las conecta cruzadas (P-R, R-P):

```
void NeuronsRP::Avanzar(NeuronsRP* N1,NeuronsRP* N2,NeuronsRP* N3,NeuronsRP* N4,NeuronsRP* N5)

void NeuronsRP::AvanzarCruzado(NeuronsRP* N1,NeuronsRP* N2,NeuronsRP* N3,NeuronsRP* N4,NeuronsRP* N5)
```

Se permite hasta un máximo de cinco conexiones de entrada por cada módulo (o lo que es lo mismo, una neurona puede recibir influencia de hasta seis neuronas diferentes, cinco externas más la pareja de su mismo modulo).

Cada conexión tendrá una ganancia propia, y para habilitar la opción de tener conexiones excitatorias e inhibitorias al mismo tiempo, se ha optado por la opción de tener dos tipos de funcionamiento simultáneos (uno corresponderá con las posiciones pares y otro con las posiciones impares).

A estos módulos se les ha llamado “NeuronaRP”

4.1.1.2 Nivel de neurona

A este nivel están implementadas las neuronas tal y como uno puede imaginárselas a partir del modelo. En la práctica se podría trabajar a este nivel, pero el gran número de conexiones que hay, hace que sea más complejo visualizar las conexiones trabajando directamente con las neuronas.

- Con el modelo simplificado trabajamos con ocho módulos, lo que corresponde a 24 neuronas (ocho neuronas P + ocho neuronas R + ocho neuronas M).
- Con el modelo simplificado tenemos un máximo de 40 conexiones (ocho módulos * cinco entradas), lo que corresponde a 112 conexiones (8 neuronas P* 6 entradas + ocho neuronas R* 6 entradas + ocho neuronas M* dos entradas)

Aún así usaremos este nivel a la hora de estudiar el funcionamiento de las neuronas a nivel de módulo.

4.1.1.3 Nivel de modelo

Es a este nivel donde se encuentran reflejadas las ecuaciones elegidas en este proyecto como modelo de neurona.

Se compondrá de dos partes. Por un lado, están las neuronas R y P que comparten modelo (Modelo de Rulkov [32]), y está formado por el siguiente mapa iterado:

Donde *x* representa al voltaje de la membrana con una respuesta de impulsos rápidos e *y* s aporta la respuesta lenta del sistema. Estas dos variables producen una actividad conocida como comportamiento spiking-bursting [26].

La sinapsis entre neuronas (sinapsis químicas excitadoras o inhibidoras) se modelan de la siguiente forma [26]:

Donde es la actividad sináptica, y contiene la contribución temporal del circuito e I representa a la contribución de otras neuronas. Las neuronas M tendrán por el contrario un modelo diferente, que corresponde con las siguientes ecuaciones

Si fuese el caso de querer emplear otro modelo diferente, solo habría que modificar este nivel, y dejar el resto igual, pues hay independencia de un nivel con otro.

Adicionalmente, es posible acceder a todas las variables del sistema, para hacer más fácil el acceso a estas desde cualquier punto del programa.

El esquema UML resultante se muestra en la siguiente página:

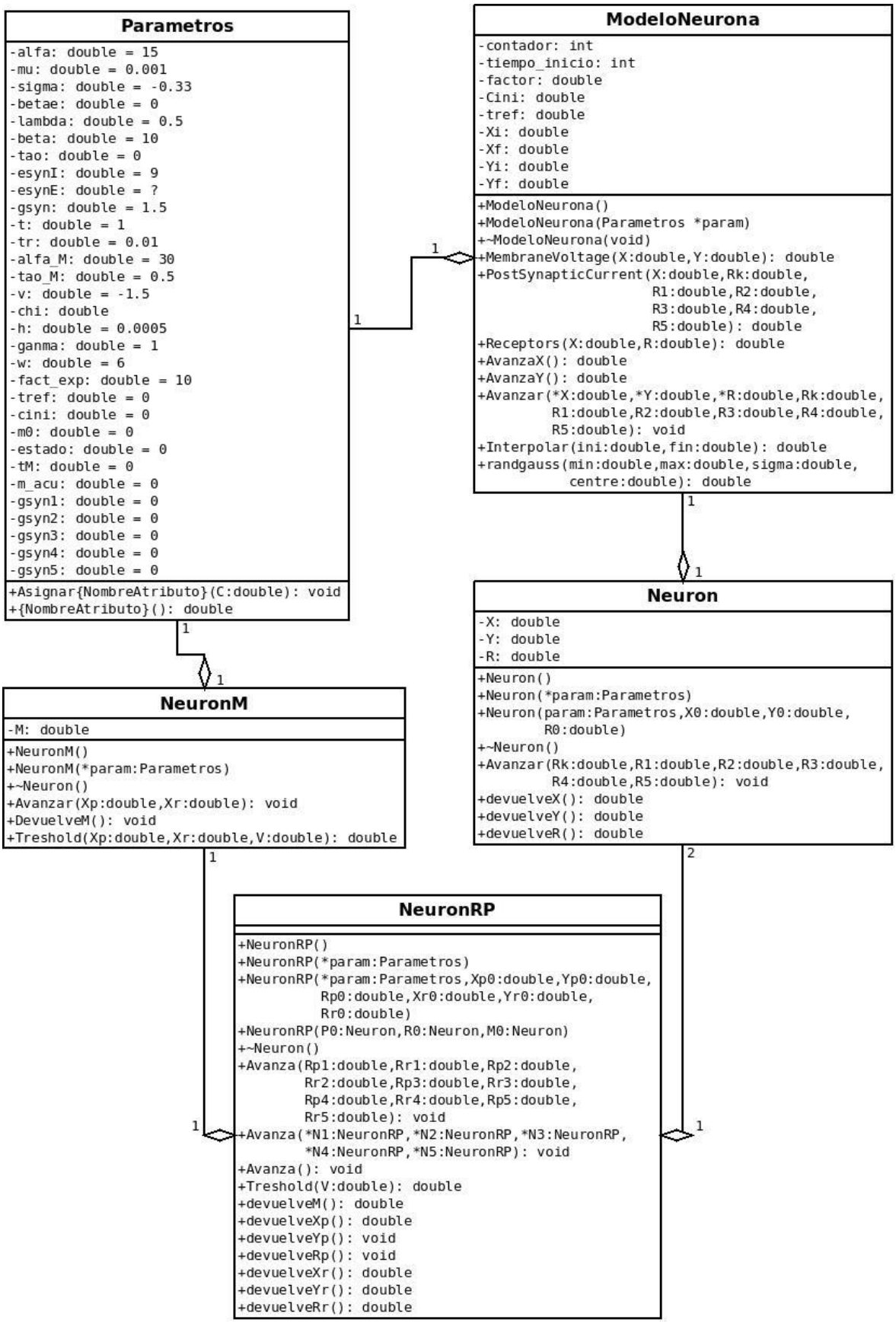

Figura 21 - Modelo UML

Para realizar el programa que simule una red neuronal, se ha utilizado el lenguaje C++. Se ha intentado que el código sea lo más reutilizable posible, haciendo un diseño modular con clases.

La clase *"Parámetros"* recoge todas las constantes del sistema, de forma que modificándola aquí se cambie en cualquier lugar. Esta clase es creada una única vez en todo el programa y cualquier función que quiera acceder a ella, lo hace en forma de puntero.

La clase *"ModeloNeurona"* es la que recoge todo el modelo matemático para el subcircuito de las neuronas R y P. Con esta implementación es posible cambiar de modelo sin más que tocar esta clase, modificando los algoritmos de cada función.

La clase *"Neuron"* es la que implementa las neuronas R y P. En las funciones que usa, se olvida el modelo matemático, que ya está en *"ModeloNeuron"*.

La clase *"NeuronM"* es la que implementa la neurona M (motoneurona).
La clase *"NeuronRP"* es una clase que crea los vínculos entre las neuronas R, P y M, creando una pack de tres neuronas que podrán ser conectadas con otras. De esta forma simplifica mucho el programa final, pues todo queda reducido.

4.1.2 Análisis de las neuronas de un mismo bloque

El primer paso para entender las redes neuronales, será el estudio de las neuronas de un mismo módulo, y poder más tarde extrapolar resultados.

En una primera aproximación, se han conectado las neuronas R, P y M obteniendo los siguientes resultados:

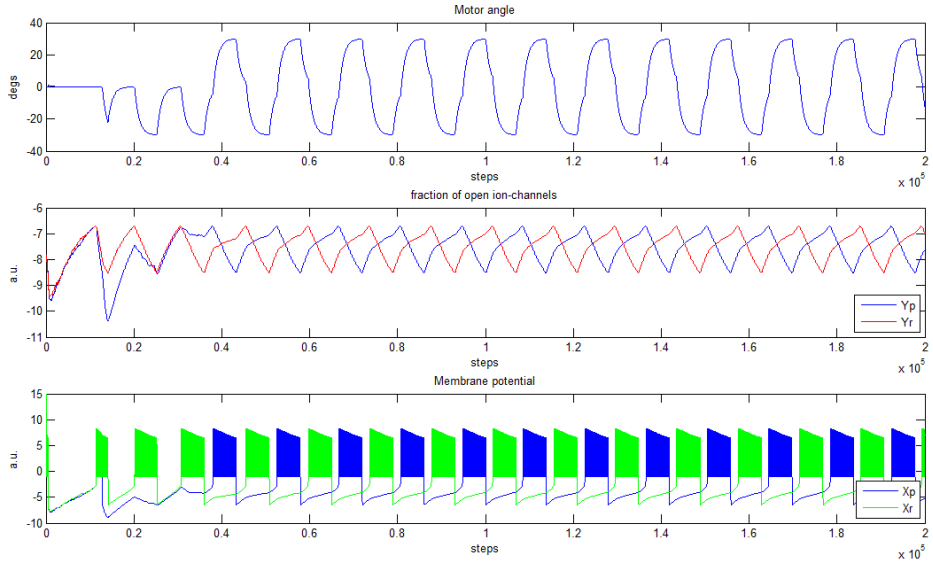

Figura 22 - Análisis RPM 1. Se muestra las graficas del "motor angle", fraction of open ion-channels" y "membrane potential"

Como se observa las neuronas se sincronizan con bastante rapidez, en tan sólo tres periodos. La señal que interesa es "Motor angle" correspondiente a la salida de la neurona M, que da la señal lenta que conectaremos a los motores. Esta señal sale directamente de operar las señales "Membrane potential" de las neuronas R y P, que tienen una respuesta mucho más rápida, a modo de disparo. A continuación, se muestra una ampliación de estas

señales para ver mejor como una neurona hace que la señal “Motor angle” crezca y la otra hace que decrezca:

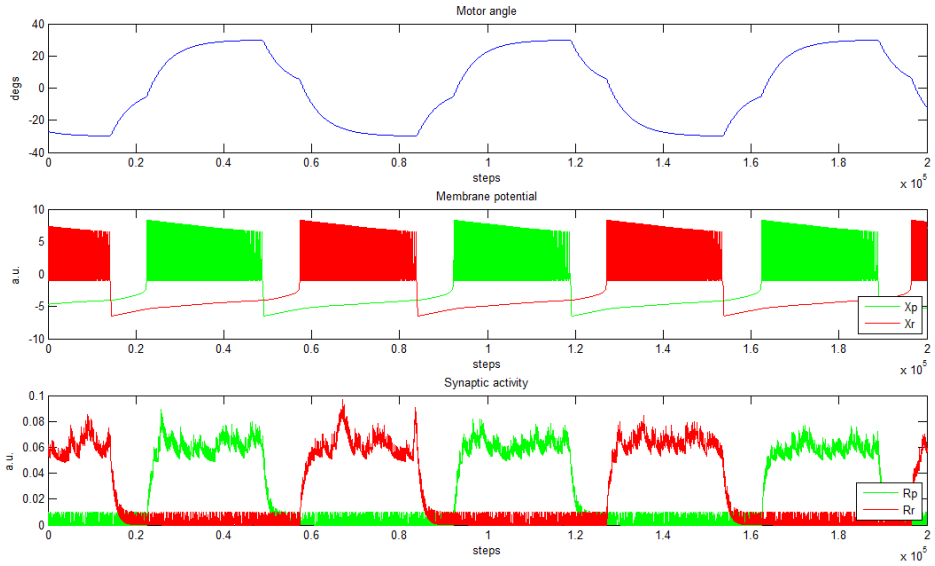

Figura 23 - Análisis RPM 2. Se representan las graficas de “motor angle”, membrane potential” y “synaptic activity”

En esta misma grafica, también se muestra la actividad sináptica entre las neuronas que hace que mientras una está funcionando, la otra esté inhibida.

Hay que decir que los resultados no siempre son tan buenos, pues el sistema tiene varias soluciones y hay algunas que no interesan:

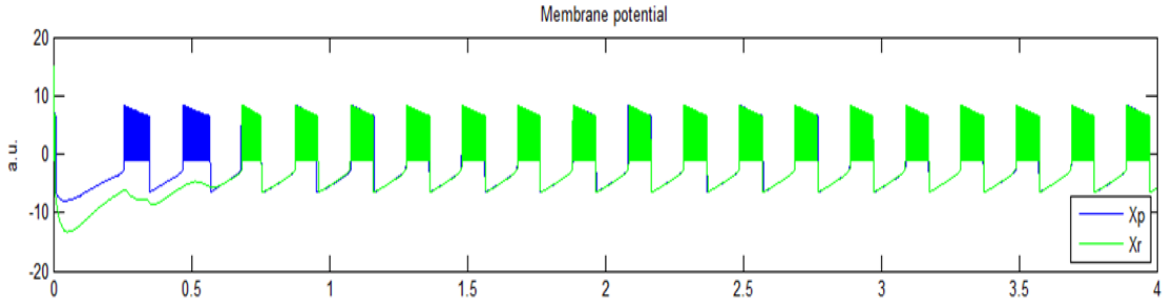

Figura 24 - Problema de sincronización

Esto ocurre cuando las neuronas se sincronizan a nivel de disparo por unas parámetros o unas condiciones iniciales desfavorables. Como hemos dicho es otra solución del sistema, pero no la óptima. La señal “Motor angle” quedará como ruido en este caso. Para solucionar esto se ha hecho uso del algoritmo de la temperatura.

Este algoritmo viene a decir que para obtener una solución óptima para un problema que tiene soluciones sub-óptimas, hay que introducir un ruido que lo saquen de este estado. Es por esto que se ha optado por introducir un ruido muy fuerte al principio que decrezca exponencialmente, y ayude a producir la oscilaciones alternantes entre estas neuronas. Aunque ayuda y mucho el introducir este ruido, hay que decir que no evita este tipo de

situaciones al 100%, pero afortunadamente, cuantas más neuronas y conexiones haya en el sistema, menos probabilidades hay de que se den este tipo de situaciones.

El ruido al comienzo de la ejecución puede verse bien en la actividad sináptica:

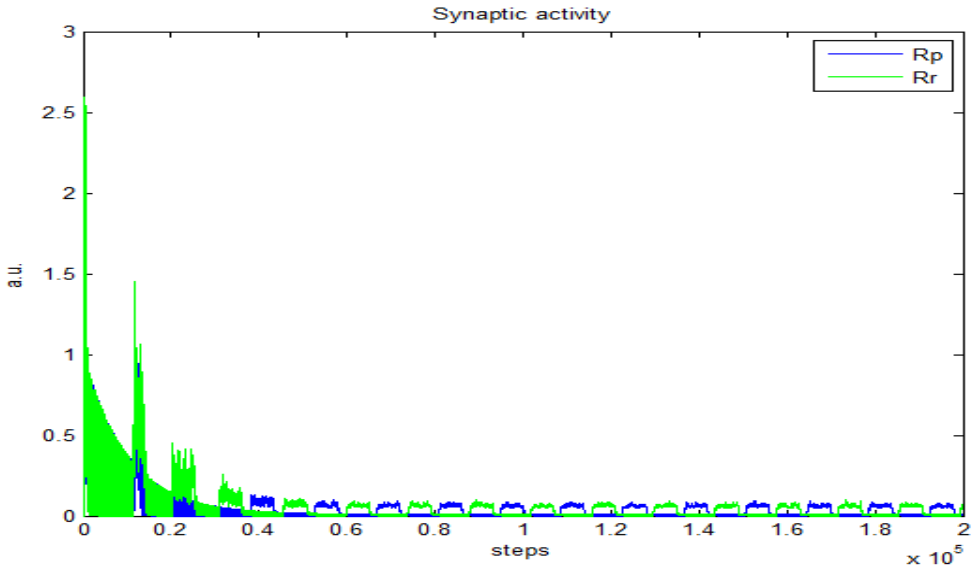

Figura 25 - Ruido inicial

El siguiente punto a estudiar es conectar dos módulos y obtener fases entre las diferentes señales de las neuronas M:

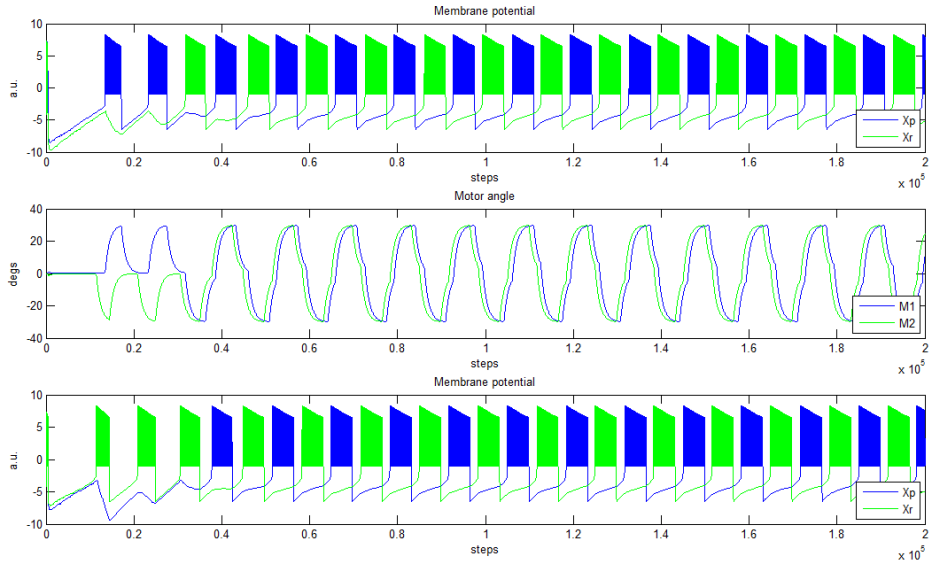

Figura 26 - Sincronización de dos neuronas. Se muestra arriba y abajo la gráfica de "membrane potential" de dos bloques diferentes, y en medio el "motor angle" de los dos módulos superpuestos.

Esto quedará resuelto en el siguiente apartado.

4.1.2 Obtención de amplitudes y desfases

Una vez conseguida la señal sinusoidal con el trío de neuronas RPM, el problema se reduce a conseguir diferentes fases y amplitudes para cada trío de neuronas. Este objetivo se debe de conseguir preferiblemente, variando parámetros de las neuronas R o P, pues son las que representan el modelo de neurona real.

Control de la amplitud

Como se explica en el artículo [26], la amplitud de la señal de salida se puede conseguir individualmente cambiando el parámetro α de la ecuación de x_n.

Este parámetro controla el porcentaje de tiempo que la neurona está disparando, y esto a su vez afecta directamente a la señal de la neurona M.

En la siguiente gráfica se representa la amplitud de la señal de la neurona M variando el parámetro α de 0.1 a 1 con un paso de 0.1. La razón de elegir un paso tan grande y obtener una gráfica escalonada en vez de la recta que se obtendría si elige un paso más fino no es más que comprobar cómo de rápido el cambio de dicho parámetro afecta a la salida y asegurarnos de que se obtiene estabilidad.

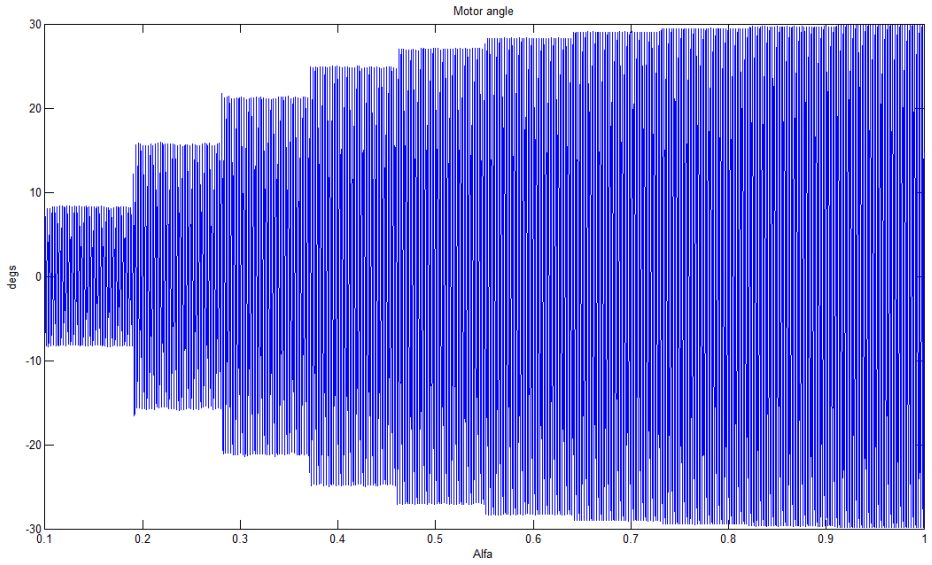

Figura 27 - Cambio de amplitud. Para esta grafica solo se ha empleado un módulo.

Para esta gráfica se han utilizado los siguientes parámetros:

Tabla 3 – Parámetros 1

α	0.1:0.1:1	t_r	0.01
μ	0.001	G	2.5
σ	-0.33	E_{syn}	9
λ	0.5	ν	-1.5
[T]	1	γ	30
β	10	τ	0.5

Control de la fase

En el control de la fase es necesaria ya la interacción entre neuronas de diferentes módulos. En la obtención de resultados habrá que valorar por encima de todo la estabilidad, pues uno de los objetivos de utilizar CPGs en vez de otros métodos es éste. En las simulaciones realizadas, se han elegido tres configuraciones, cada una de ellas válida en determinadas circunstancias.

Las ecuaciones que controla la interacción entre las neuronas son las siguientes:

Analizándolas más a fondo, se observan cuatro parámetros, de los cuales se han usado con éxito tres de ellos para conseguir el desfase deseado.

En este apartado se han hecho las pruebas únicamente con dos módulos para simplificar y, más tarde, extrapolar los resultados al resto.

Configuración 1:

En este caso, se ha conectado cada neurona R o P con su homóloga, y se ha variado el parámetro λ. Con esta solución se ha obtenido una fase muy estable desde cero hasta aproximadamente 60º. Las fases por encima de los 60º no se pueden conseguir con este método, pero si se desea una fase superior sólo hay que intercambiar los módulos. La siguiente gráfica se ha conseguido con este método.

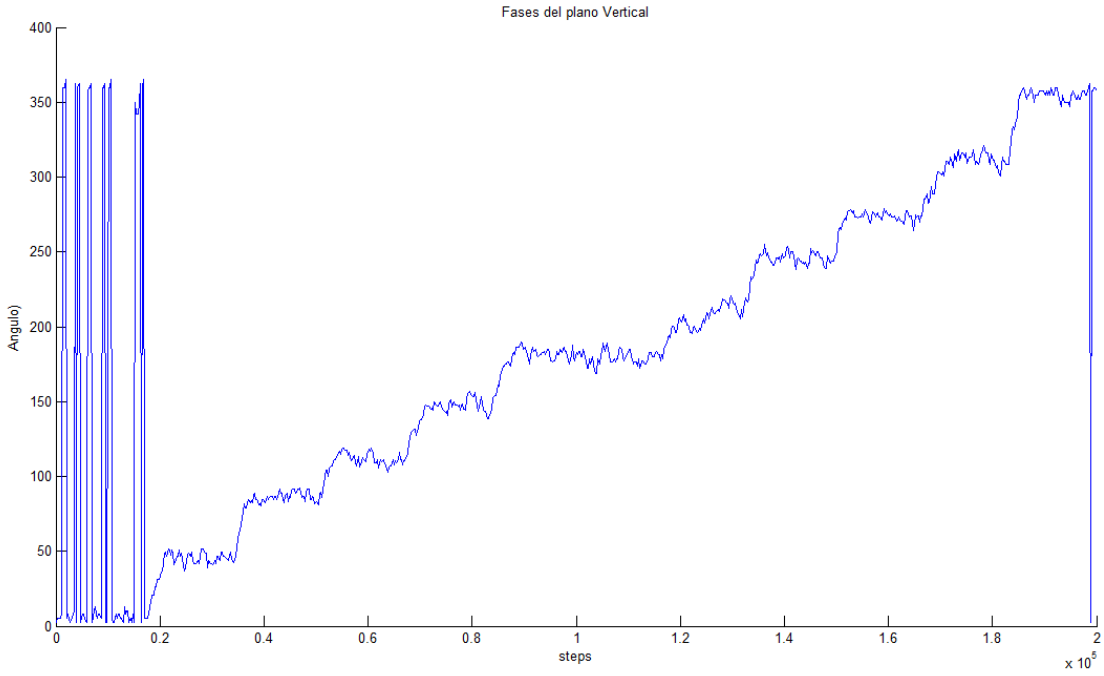

Figura 28 - Configuración 1

Para ello se ha variado el parámetro λ, de 0.35 a 0.6 para conseguir fases desde cero hasta 180 grados y, posteriormente, se ha cambiado de 0.6 a 0.35 pero intercambiando las configuraciones.

Esta configuración funciona muy bien cuando se quiere conseguir una fase determinada entre dos módulos, pero si se desea controlar la fase entre más módulos, los resultados no son buenos, pues para cada módulo extra hay que aumentar el parámetro λ de forma cuadrática para obtener el mismo resultado, y llegado a un punto éste deja de ser estable.
Es, por tanto, que resulta útil para controlar la fase entre los planos horizontal y vertical, pero no para controlar la fase en el mismo plano.

Los parámetros usados en esta gráfica son los siguientes:

Tabla 4 - Parámetros 2

α	10	t_r	0.01
μ	0.001	G	2.5
σ	-0.33	E_{syn1}	9
λ1, λ2	0.5,0.35:0.05:0.6	ν	-1.5
[T]	1	γ	30
β	10	τ	0.5
G1	0.2	G2	0.7

Configuración 2:

En este caso, se pretende conseguir una fase estable entre módulos del mismo plano. Por tanto, hay que conseguir que la fase obtenida sea la misma al aumentar el número de módulos. La solución dada ha sido conectar los bloques (cada neurona con su homóloga) con una inhibición débil y una excitación fuerte. Al variar el parámetro g de la excitación fuerte, la fase varía lentamente, aunque sin cubrir todo el rango de fases como sí ocurría en la primera configuración. Pese a que hay limitaciones en esta configuración, para nuestro caso interesa conseguir fases de 100 y 120 grados, y esto se consigue fácilmente usando este método.

Configuración 3:

Esta configuración será útil únicamente cuando se quiera tener dos o más módulos en fase. En este caso, lo que cambia es la interconexión, que en lugar de conectar una neurona con su homóloga, se conectan neuronas P con neuronas R.

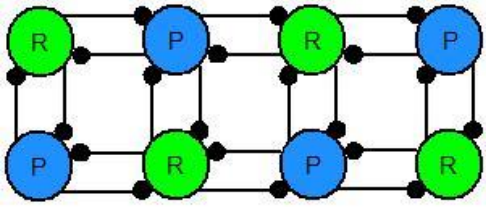

Figura 29 - Configuración 3

De esta forma, obtenemos una sincronización igual que la obtenida dentro de un módulo, pero con muchas más neuronas. La única restricción será que la suma de todas las g que llegan a una neurona debe ser constante (2.5 en nuestro caso)

4.2 Estudio CPG

4.2.1 Estudio 1: Rolling

En este estudio, se pretende conseguir el movimiento de rolling. Para ello es necesario conseguir una fase en cada plano de cero grados y una frase entre planos de 90 grados:

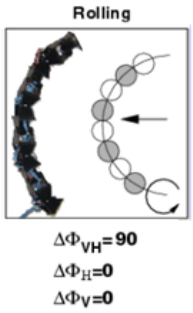

Figura 30 - Rolling

Para conseguir la fase del mismo plano igual a cero, se ha elegido la configuración 3 descrita en el punto 7. Por lo tanto, las neuronas P solo están conectadas con neuronas R en cada plano, y viceversa:

Sin embargo, las conexiones entre planos siguen siendo de una neurona con su homóloga del plano diferente.

Como se observa en la siguiente gráfica, la sincronización de ambos planos por separado queda exacta.

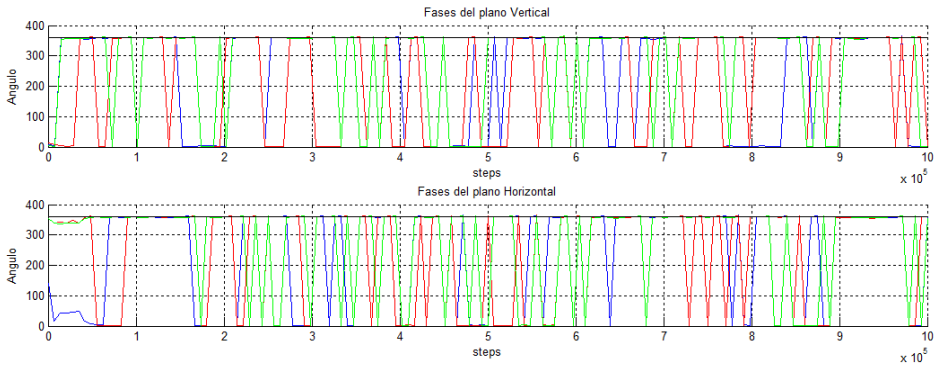

Figura 31 - Fases Rolling

Para conseguir una fase de 90º entre los dos planos, se ha usado la configuración 1, para el cual se ha usado la gráfica dada en el apartado donde se explica dicha configuración como primera aproximación, y posteriormente se ha encontrado un punto más aproximado a dicha fase haciendo pruebas. Esto es porque al cambiar otros parámetros, la gráfica de la configuración 1 no es exacta, pero si se aproxima bastante, y hay que encontrar el punto exacto haciendo pruebas en torno a ese punto.

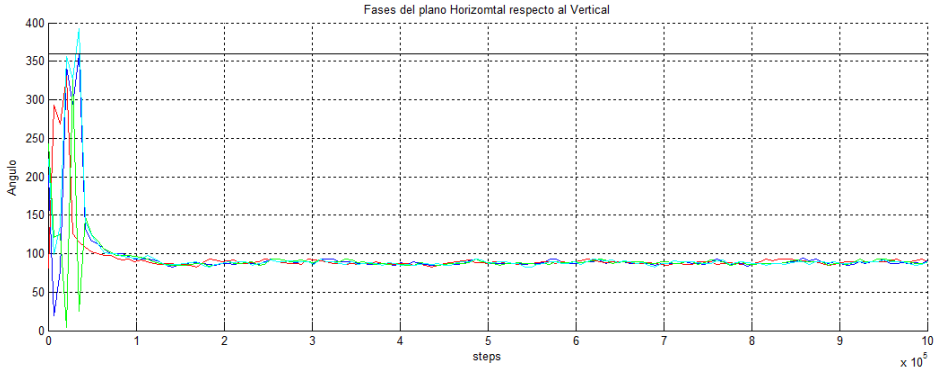

Figura 32 - Fase entre planos Rolling

El resumen de las interconexiones necesarias para hacer estas graficas son las siguientes:

```
/* Parameters for P y R: */             /* Parameters for M: */                 p->AsignarEsynI(9);
p->AsignarAlfa(15);                     p->AsignarAlfa_M(1);                    p->AsignarEsynE(9);
p->AsignarMu(0.001);                    p->AsignarTao_M(0.5);                   p->AsignarGsyn(0.8);   //I
p->AsignarSigma(-0.33);                 p->AsignarV(-1.5);                      p->AsignarGsyn1(0.8);  //I
/* Parameters for inhibitory synapses   p->AsignarChi(30);                      p->AsignarGsyn2(0.8);  //I
between P and R */                      /* otros parametros */                  p->AsignarGsyn3(0.7);  //I
p->AsignarLambda(0.5);                  p->AsignarH(0.0005);                    p->AsignarGsyn4(0.05); //I
p->AsignarBeta(10);                     p->AsignarGanma(1);
p->AsignarT(1);                         p->AsignarW(6);                         baseA = 0.5;
p->AsignarTr(0.01);                     p->AsignarFact_exp(10);                 baseB = 0.53;


for (int i= 0; i<TamCPG; i++){
        if(i==0){ //                 * 1º         * 2º           * 3º           * 4º            * 5º
                p->AsignarLambda(baseA);
                CH[0].AvanzarCruzado(&(CH[1])    ,&(CH[3])       ,&(CV[0])       ,NULL           ,NULL);
                p->AsignarLambda(baseB);
                CV[0].AvanzarCruzado(&(CV[1])    ,&(CV[3])       ,NULL           ,&(CH[0])       ,NULL);
        }else if(i== TamCPG-1){ //    * 1º         * 2º           * 3º           * 4º            * 5º
        p->AsignarLambda(baseA);
        CH[TamCPG-1].AvanzarCruzado(&(CH[TamCPG-4]),&(CH[TamCPG-2]),&(CV[TamCPG-1]),NULL          ,NULL;
        p->AsignarLambda(baseB);
        CV[TamCPG-1].AvanzarCruzado(&(CV[TamCPG-4]),&(CV[TamCPG-2]), NULL         ,&(CH[TamCPG-1]) ,NULL)
        }else{//                     * 1º         * 2º           * 3º           * 4º            * 5º
                p->AsignarLambda(baseA);
                CH[i].AvanzarCruzado(&(CH[i-1])   ,&(CH[i+1])     ,&(CV[i])       ,NULL           ,NULL);
                p->AsignarLambda(baseB);
                CV[i].AvanzarCruzado(&(CV[i-1])   ,&(CV[i+1])     ,NULL           ,&(CH[i])       ,NULL);
        }

}
```

En este código se observa la asignación de valores a las constantes de las ecuaciones, seguido de la conexión de cada modulo. La numeración de las Gsin corresponde a la conexión que esté en la posición de entrada de la función “Avanzar”. Por el contrario hay dos Esyn, una de ellas correspondiente a las posiciones 1, 3 y 5 y la otra correspondiente a la posición 2 y 4.

Una vez conseguidas las fases, se pueden obtener dos movimientos a partir de aquí. Para ello habrá que cambiar la amplitud (con el parámetro α), para la cual habrá un umbral en el que deja de ser un movimiento lateral y se considera un movimiento rotatorio:

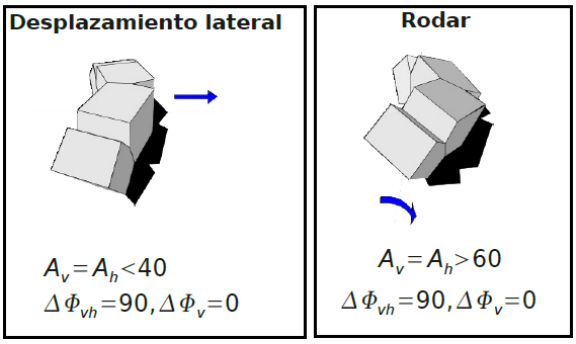

Figura 33 - Movimientos Rolling

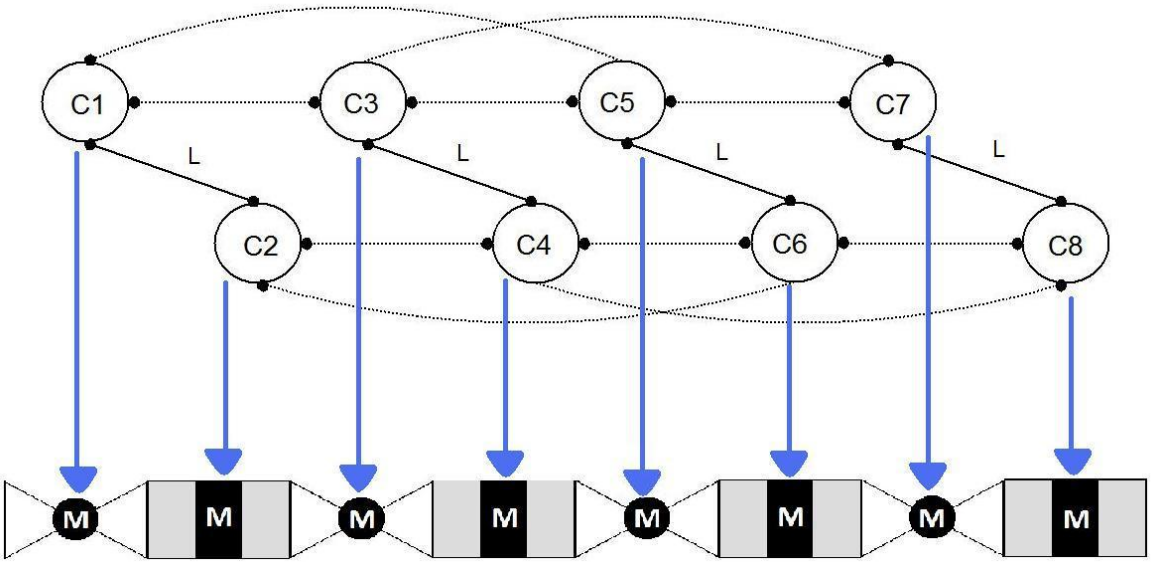

Figura 34 - CPG Rolling

4.2.1 Estudio 2: Sinusoidal

En este estudio, se pretende conseguir el movimiento sinusoidal. Para ello es necesario conseguir una fase en uno de los planos de 120 grados e inhibir por completo el otro plano, dejando una amplitud nula:

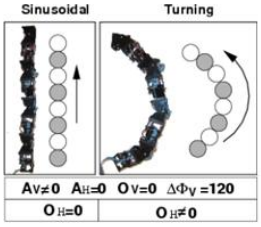

Figura 35 - Sinusoidal

Para conseguir la fase de uno de los planos igual a 120 grados, se ha elegido la configuración 2 descrita en el punto 7. Por lo tanto, el sincronismo se consigue con una inhibición débil combinada con una excitación fuerte que ajusta la fase. Como se observa en la siguiente gráfica, la sincronización del plano vertical queda exacta.

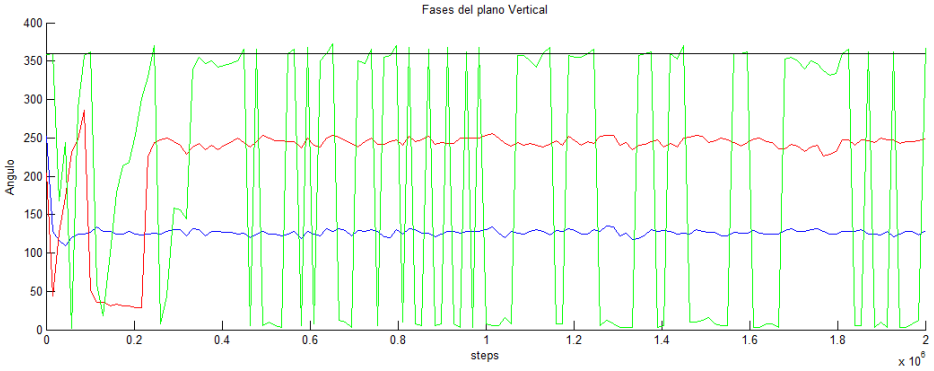

Figura 36 - Fases sinusoidal

El plano horizontal queda inhibido en este caso:

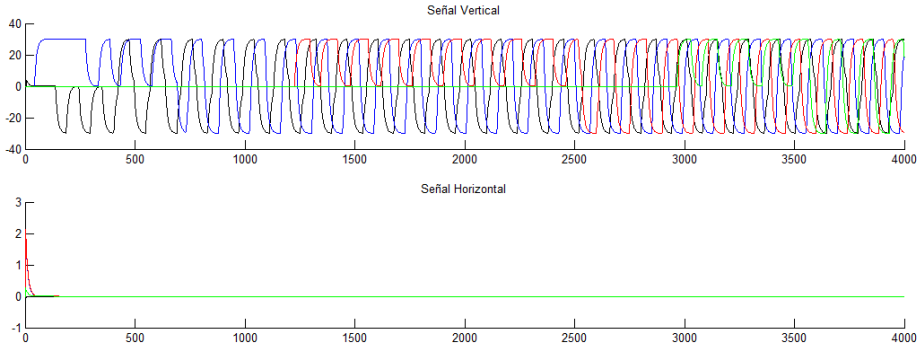

Figura 37 - Amplitud sinusoidal

El resumen de las interconexiones necesarias para hacer estas gráficas son las siguientes:

```
/* Parameters for P y R: */
p->AsignarAlfa(15);
p->AsignarMu(0.001);
p->AsignarSigma(-0.33);
/* Parameters for inhibitory synapses
between P and R */
p->AsignarLambda(0.5);
p->AsignarBeta(10);
p->AsignarGsyn(2.5);
p->AsignarT(1);
p->AsignarTr(0.01);

/* Parameters for M: */
p->AsignarAlfa_M(1);
p->AsignarTao_M(0.5);
p->AsignarV(-1.5);
p->AsignarChi(30);
/* otros parametros */
p->AsignarH(0.0005);
p->AsignarGanma(1);
p->AsignarW(6);
p->AsignarFact_exp(10);

p->AsignarEsynI(9);
p->AsignarEsynE(-9);
p->AsignarGsyn1(0.2);  //I
p->AsignarGsyn2(6.2);  //E
p->AsignarGsyn3(200);  //I
p->AsignarGsyn4(0);    //E

baseA = 0.5;
baseB = 0.53;
```

```
for (int i= 0; i<TamCPG; i++){
        if(i==0){ //                * 1º           * 2º            * 3º           * 4º            * 5º
                CH[0].Avanzar(      &(CH[1])      ,NULL           ,&(CV[0])      ,NULL           ,NULL);
                CV[0].Avanzar      (&(CV[1])      ,NULL           ,NULL          ,&(CH[0])       ,NULL);
        }else if(i== TamCPG-1){ //  * 1º         * 2º             * 3º           * 4º            * 5º
                CH[TamCPG-1].Avanzar(NULL          ,&(CH[TamCPG-2]),&(CV[TamCPG-1]),NULL          ,NULL;
                CV[TamCPG-1].Avanzar(NULL          ,&(CV[TamCPG-2]), NULL         ,&(CH[TamCPG-1]) ,NULL)
        }else{//                    * 1º           * 2º            * 3º           * 4º            * 5º
                CH[i].Avanzar(      &(CH[i+1])    ,&(CH[i-1])     ,&(CV[i])      ,NULL           ,NULL);
                CV[i].Avanzar(      &(CV[i+1])    ,&(CV[i-1])     ,NULL          ,&(CH[i])       ,NULL);
        }

}
```

En este código se observa la asignación de valores a las constantes de las ecuaciones, seguido de la conexión de cada modulo. La numeración de las Gsin corresponde a la conexión que esté en la posición de entrada de la función “Avanzar”. Por el contrario hay dos Esyn, una de ellas correspondiente a las posiciones 1, 3 y 5 y la otra correspondiente a la posición 2 y 4.

Una vez conseguidas las fases, se pueden obtener dos movimientos a partir de aquí. Para ello habrá que cambiar el offset, que hará que el movimiento sea en línea recta o en arco:

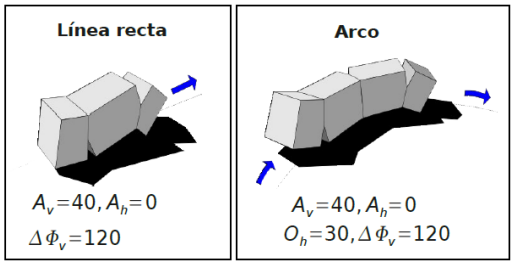

Figura 38 - Movimiento sinusoidal

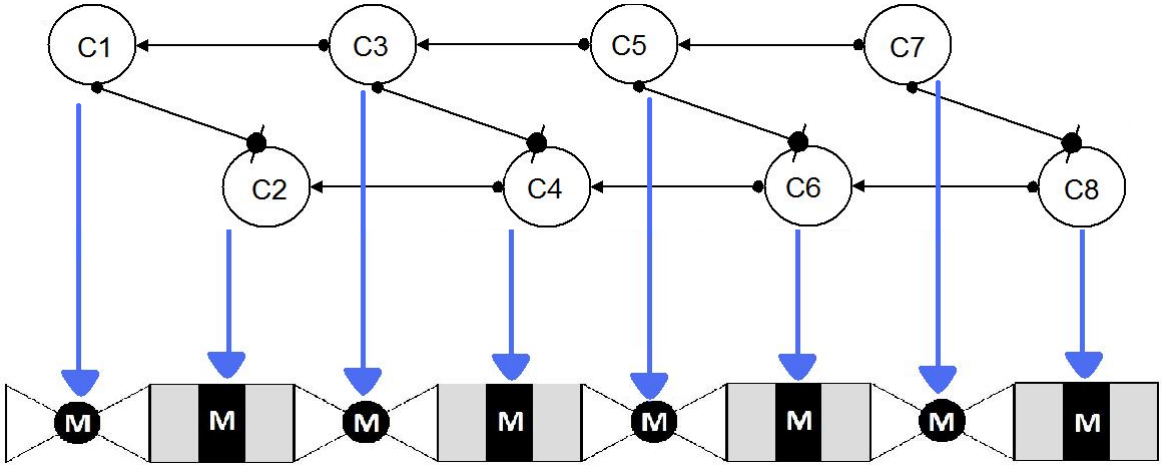

Figura 39 - CPG sinusoidal

4.2.2 Estudio 3: Lateral shifting

En este estudio, se pretende conseguir el movimiento de desplazamiento lateral. Para ello es necesario conseguir una fase en cada plano de 100 grados y una frase entre planos de cero grados:

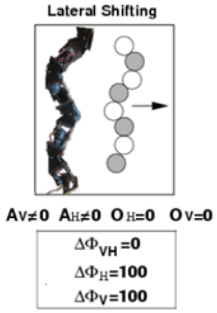

Figura 40 - Shifting

Para conseguir una fase de 100º en el mismo plano se ha usado la configuración 1, para el cual se ha usado la gráfica dada en el apartado donde se explica dicha configuración como primera aproximación, y posteriormente se ha encontrado un punto más aproximado a dicha fase haciendo pruebas. Esto es así porque al cambiar otros parámetros, la gráfica de la configuración 1 no es exacta, pero sí se aproxima bastante, y hay que encontrar el punto exacto haciendo pruebas en torno a ese punto.

A continuación, se muestran las fases de cada módulo con el primero:

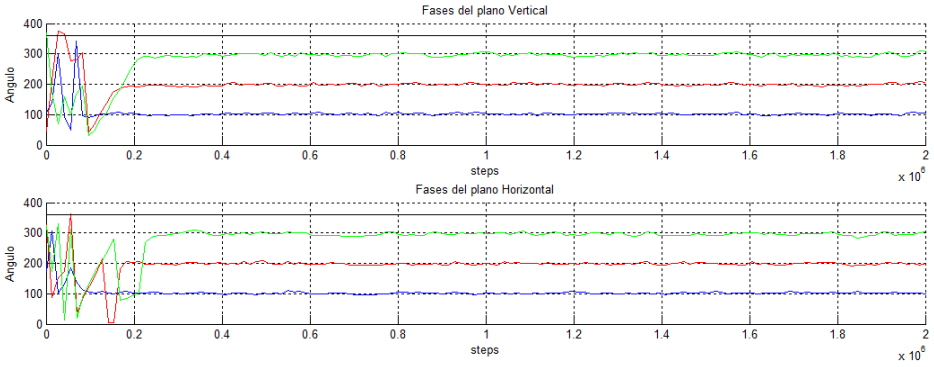

Figura 41 - Fases shifting

En esta otra gráfica se muestra la fase de cada plano con el anterior:

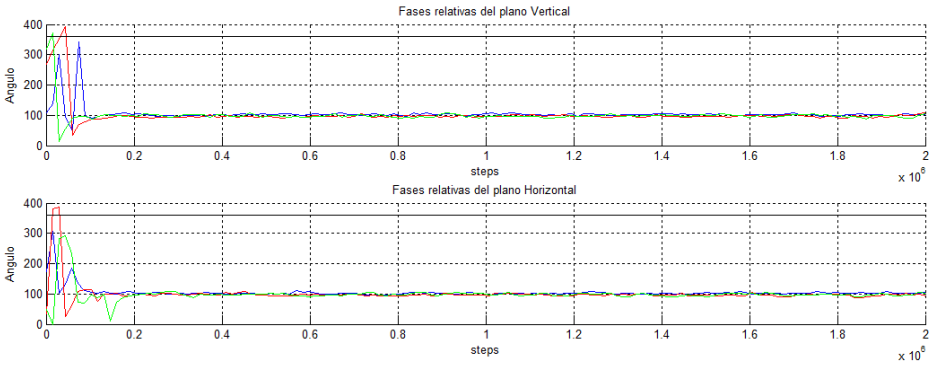

Figura 42 - Fases relativas shiftig

Para conseguir la fase de diferentes planos igual a cero, se ha elegido la configuración 3 descrita en el punto 7. Por lo tanto, las neuronas P sólo están conectadas con neuronas R entre los dos planos, y viceversa:

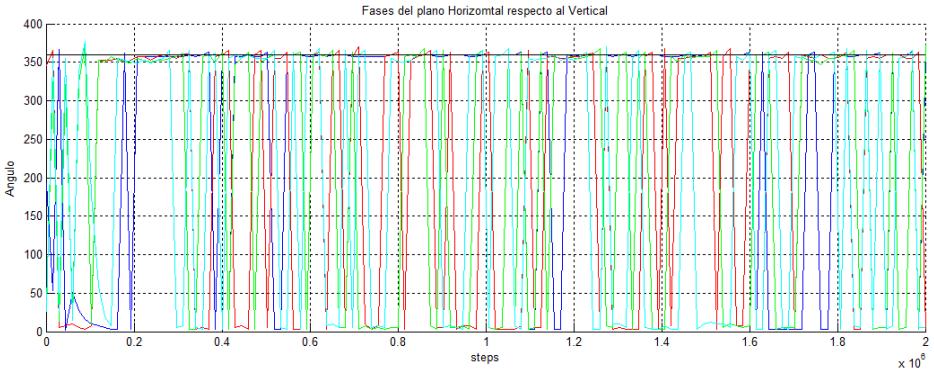

Figura 43 - Fases entre planos shifting

El resumen de las interconexiones necesarias para hacer estas graficas son las siguientes:

```
/* Parameters for P y R: */
p->AsignarAlfa(15);
p->AsignarMu(0.001);
p->AsignarSigma(-0.33);
/* Parameters for inhibitory synapses
between P and R */
p->AsignarLambda(0.5);
p->AsignarBeta(10);
p->AsignarT(1);
p->AsignarTr(0.01);
```

```
/* Parameters for M: */
p->AsignarAlfa_M(1);
p->AsignarTao_M(0.5);
p->AsignarV(-1.5);
p->AsignarChi(30);
/* otros parametros */
p->AsignarH(0.0005);
p->AsignarGanma(1);
p->AsignarW(6);
p->AsignarFact_exp(10);
```

```
p->AsignarEsynI(9);
p->AsignarEsynE(-9);
p->AsignarGsyn(1.2);   //I
p->AsignarGsyn1(1.2);  //I
p->AsignarGsyn4(4.4); //I
p->AsignarGsyn5(0.2); //I

baseA = 0.5;
baseB = 0.53;
```

```
for (int i= 0; i<TamCPG; i++){
    if(i==0){ //                    * 1º          * 2º          * 3º          * 4º             * 5º
         CH[0].AvanzarCruzado(      &(CV[0])      ,NULL         ,NULL         ,NULL            ,(&(CH[1]));
         CV[0].AvanzarCruzado(      &(CH[0])      ,NULL         ,NULL         ,NULL            ,(&(CV[1]));
    }else if(i== TamCPG-1){ //      * 1º          * 2º          * 3º          * 4º             * 5º
         CH[TamCPG-1].AvanzarCruzado(&(CV[TamCPG-1]),NULL       ,NULL         ,&(CH[TamCPG-2]) ,NULL;
         CV[TamCPG-1].AvanzarCruzado(&(CH[TamCPG-1]),NULL       ,NULL         ,&(CV[TamCPG-2]) ,NULL);
    }else{//                        * 1º          * 2º          * 3º          * 4º             * 5º
         CH[i].AvanzarCruzado(      &(CV[i])      ,NULL         ,NULL         ,&(CH[i-1])      ,&(CH[i+1]));
         CV[i].AvanzarCruzado(      &(CH[i])      ,NULL         ,NULL         ,&(CV[i-1])      ,&(CV[i+1]));
    }

}
```

En este código se observa la asignación de valores a las constantes de las ecuaciones, seguido de la conexión de cada modulo. La numeración de las Gsin corresponde a la conexión que esté en la posición de entrada de la función “Avanzar”. Por el contrario hay dos Esyn, una de ellas correspondiente a las posiciones 1, 3 y 5 y la otra correspondiente a la posición 2 y 4.

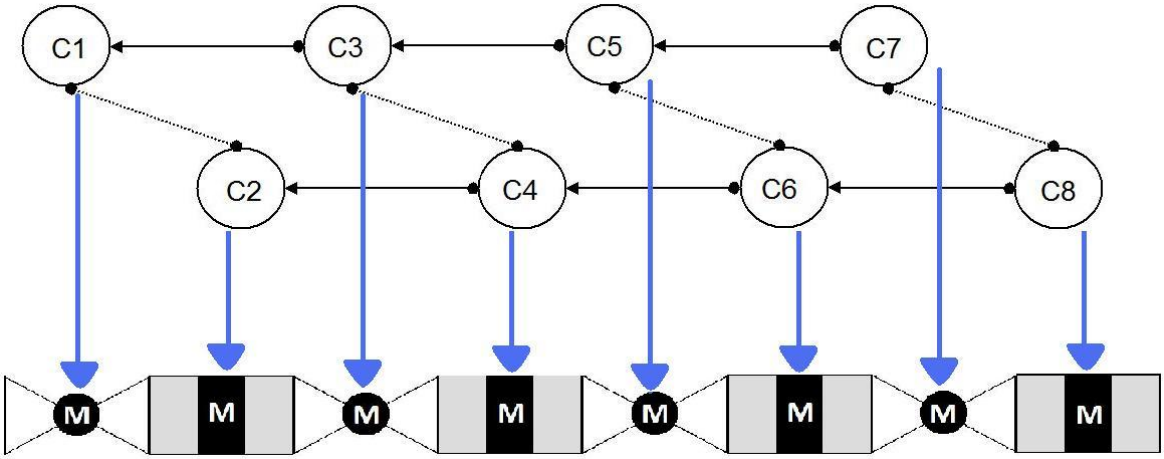

Figura 44 - CPG shifting

5 Integración, pruebas y resultados

La metodología de trabajo para la obtención de los resultados descritos en este proyecto ha sido la siguiente:

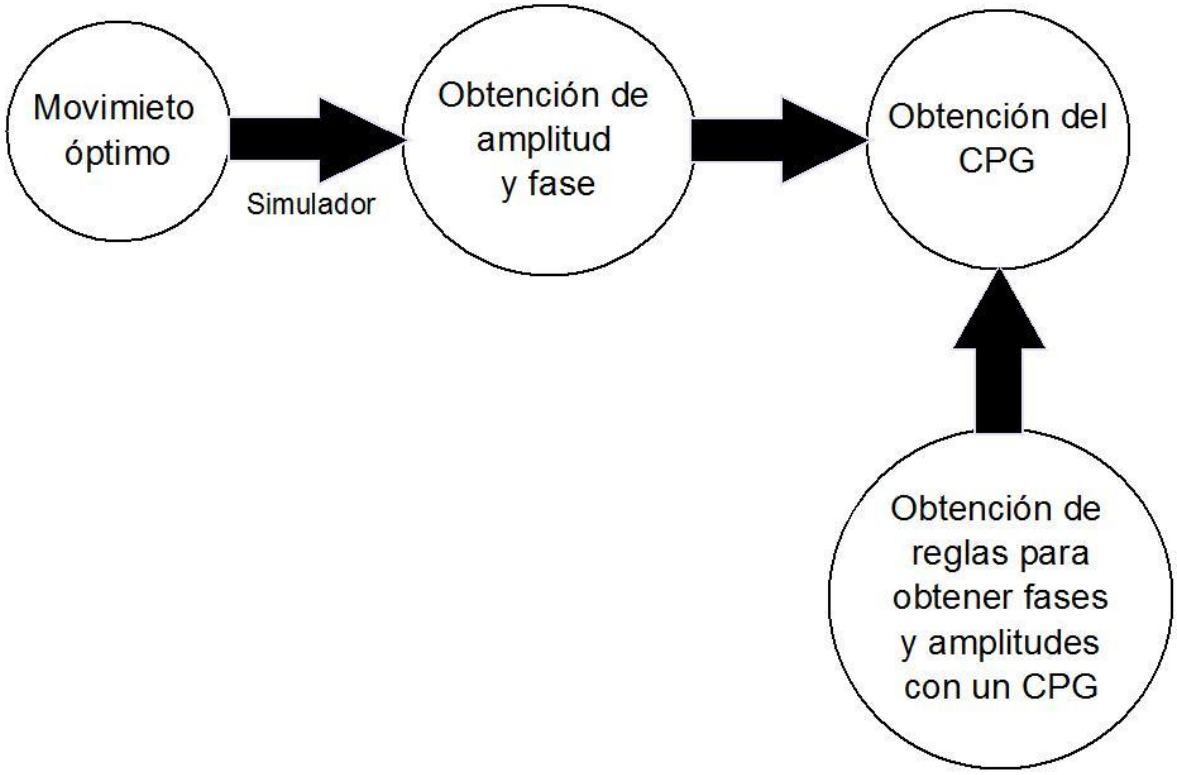

Figura 45 - Diagrama del trabajo realizado

Con círculos se representan puntos de partida y objetivos, y con flechas se representa el trabajo a hacer. En primer lugar hay que conseguir las fases y amplitudes que hacen los movimientos óptimos. Seguidamente hay que obtener una metodología para conseguir las fases y amplitudes cambiando parámetros de un CPG. Por ultimo hay que integrar ambos resultados para obtener el CPG final.

El primer paso en la obtención de resultados consiste en un análisis mediante Matlab de la señales que van a ir a los motores. De estas pruebas se obtienen amplitudes y fases que se comparan con los objetivos perseguidos. Una muestra de estas gráficas son las expuestas en el punto 4.2, donde se aprecian las fases resultantes del análisis de las señales de los CPGs estudiados. Estos resultados son puramente matemáticos, y no dan información del comportamiento físico del robot.

El siguiente paso consiste en realizar pruebas en un simulador con interfaz gráfico que permite estimar de forma aproximada lo que va a hacer el robot en la realidad. Este simulador incorpora un análisis físico del sistema y, por tanto, más realista. Debido a la lentitud y subjetividad que aporta la parte gráfica, se ha eliminado del simulador y se han establecido criterios objetivos para determinar cuáles son los mejores objetivos. El criterio elegido en este caso ha sido la mayor distancia recorrida, aunque podrían haberse elegido otros criterios como mayor ángulo de giro, mejor trayectoria… Cada criterio dependerá de

los objetivos que se persigan, y en este caso es la rapidez. La modificación del simulador permite realizar numerosas simulaciones en bloque sin necesidad de supervisión. De esta forma, se crea un fichero con todos los vectores de movimientos correspondientes a las diferentes simulaciones y el simulador genera otro fichero con los resultados obtenidos. Las pruebas en el simulador han sido para obtener movimientos a partir de señales sinusoidales desfasadas. Son cinco las variables modificadas:

- Amplitud horizontal
- Amplitud vertical
- Fase horizontal
- Fase vertical
- Fase entre planos

Al tener cinco variables, la variación de cada variable no puede ser muy grande (las pruebas a realizar serían demasiadas). Sin embargo, el simulador tampoco es perfecto, y aunque se pudiesen hacer pruebas más finas, los resultados que se obtendrían seguirían siendo aproximados.

Una vez estudiado el sistema con el simulador, es hora de la implementación física. Para ello se ha implementado un robot de ocho módulos. Existen algunas consideraciones a tener en cuenta que no aportaba el simulador:

- El robot está unido mediante un cable al ordenador, y por tanto, limita los movimientos.
- Hay piezas para la interconexión a lo largo del robot que pueden entorpecer el movimiento.
- Hay cantos de diferentes materiales que resbalan en determinadas superficies (para las pruebas se ha usado un suelo de cartón que aporta más fricción).
- La distribución del peso en el robot no es la misma que en el simulador.

Todo esto y más, hacen que los movimientos del robot no sean exactamente como se vieron en el simulador, pero se aproximan bastante.

En un primer momento, se realizaron las pruebas en el robot con señales sinusoidales, pues es un trabajo que ya se había hecho y se aseguraba así reducir la procedencia de posibles fallos (si no funciona es problema del hardware y no del software).

Cuando el robot (hardware) funcionaba correctamente era hora de probarlo con las señales de los CPGs. Los resultados fueron buenos desde el principio, tan sólo hubo que modificar la escala temporal utilizada, pues generaba los movimientos demasiado rápidos y los motores no eran capaces de seguir dicha señal. Tras esta calibración, se podían producir todos los movimientos estudiados. Hubo que calibrar la amplitud para cada movimiento para optimizarlos y sacarlo de situaciones algo inestables, y de esta forma, obtener movimientos similares a los obtenidos con las señales sinusoidales.

Hay que decir que los resultados obtenidos han sido quizás mejores de lo esperado, pues podía pensarse que el resultado de los CPG fuese un movimiento mucho más sucio que el de las señales sinusoidales, pero han sido bastante buenos.

Cuando se parte de reposo, el CPG debe llegar a una situación estable, pasando por un periodo transitorio en el cual el robot se mueve incontroladamente con espasmos hasta llegar el movimiento esperado. En las pruebas se han desechado las primeras muestras, para llegar así al movimiento esperado en régimen permanente. Esta decisión ha sido más que nada para evitar líos con el cable que conecta al robot con el ordenador.

Durante la presentación del proyecto podrá verse una demostración del movimiento del robot, ya sea haciendo una prueba si las circunstancias lo permiten.

http://www.youtube.com/watch?v=jtgie0xjon4

A continuación, puede verse un video con los cinco movimientos que el robot es capaz de hacer con los CPGs:

En el video se aprecian los siguientes movimientos:

Sinusoidal: Es el movimiento que puede realizar el robot con la configuración en un solo plano (el tipo de locomoción estudiada por Fernando Herrero [26]). El movimiento es ondulatorio, similar a la locomoción de un gusano. Su eficiencia viene determinada en gran medida por la fricción del suelo donde se mueve. En este movimiento el plano horizontal está completamente inhibido y sólo se mueve el plano vertical.

Figura 46 - Sinusoidal

Turning o arco: Este movimiento es muy similar al anterior, con la salvedad de que tener una torsión en el plano horizontal que le permite hacer el movimiento en una trayectoria circular.

Figura 47 - Arco

Lateral Shifting: El robot se mueve de forma similar a las serpientes, más concretamente, al movimiento de una serpiente sobre la arena (como la serpiente de cascabel). A este movimiento se le conoce en biología como "movimiento golpe de costado".

Figura 48 - Cascabel

Figura 49 – Lateral Shifting

Lateral: En este movimiento el robot se desplaza en dos fases. En la primera fase, desplaza la cabeza y cola al mismo tiempo apoyándose sobre el costado. En la segunda fase, desplaza el costado apoyándose en la cabeza y la cola.

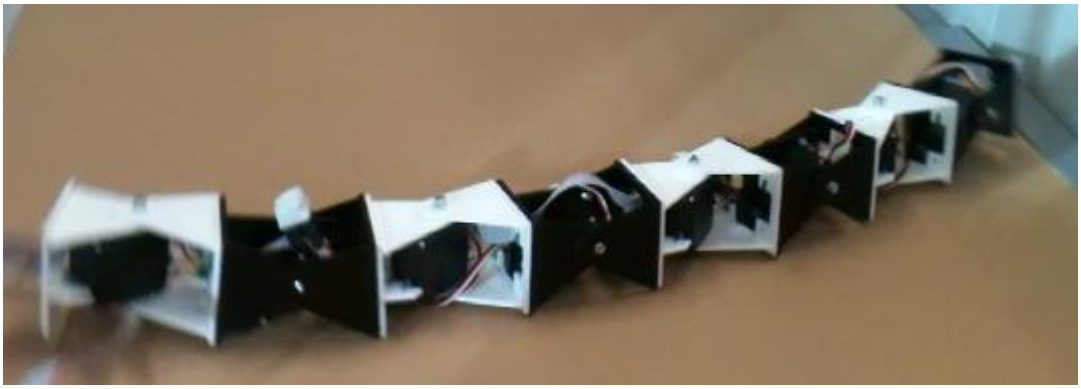

Figura 50 - Lateral

Rolling o rodar: El robot es capaz de rodar sobre su propio eje anteponiendo el costado al resto del cuerpo. Es el movimiento mas eficiente de todos, pues se alcanza la mayor velocidad. Por contraposición, no resulta tan bueno cuando el robot se encuentra unido con un cable.

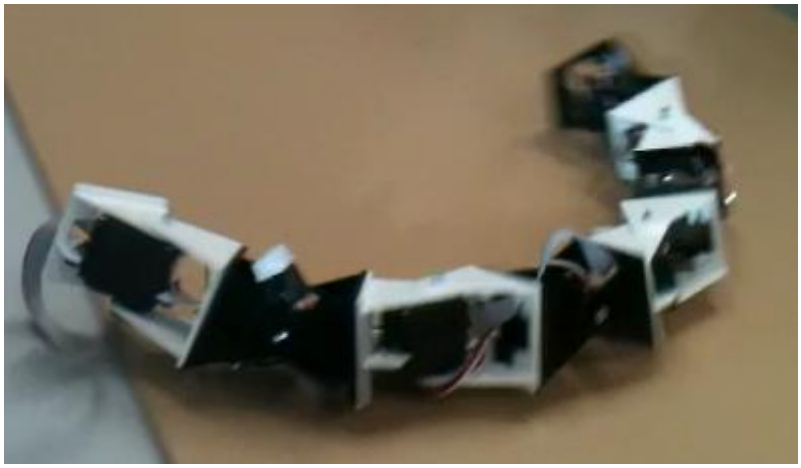

Figura 51 - Rolling

6 Conclusiones y trabajo futuro

6.1 Conclusiones

Este proyecto tenía un objetivo claro, el conseguir una serie de movimientos del robot propuesto, utilizando configuraciones de CPG bioinspirados válidas. Podemos concluir pues, que el objetivo se ha cumplido satisfactoriamente.

Así mismo, las soluciones diseñadas están adaptadas para cada caso en particular, obteniendo pues un modelo de CPG medianamente sencillo para cada movimiento pero bastante complejo si el objetivo es tener un único modelo que lo englobe todo.

En un principio podía pensarse en obtener un modelo general de CPG, para el cual, al cambiar una serie de parámetros controlados, el robot realizase los movimientos esperados. Sin embargo, tal solución no ha sido posible, y el tema de la utilización de CPG se ha visto más complicado de lo esperado a simple vista, teniéndose así que estudiar caso por caso y obtener una solución a medida. Esto hace que nuevos movimientos impliquen nuevos estudios, y que por cada nuevo movimiento, el CPG resultante (obtenido como intersección de todos los demás) aumente su complejidad en gran medida.

Sin embargo, las tres configuraciones estudiadas aportan más agilidad en trabajos futuro, pues ayudan a dividir el problema en varias partes y abordar cada una de ellas por separado.

No hay que olvidar los objetivos de utilizar CPG en vez de señales sinusoidales para la obtención de estos movimientos, pues bien es cierto que los recursos computaciones empleados son mucho mayores, pero a cambio obtenemos una mayor robustez ante posibles errores aportando una recuperación segura sea cual sea el estado en el que se encuentre. Quizás tal complejidad computacional no compense tanto, pero los recursos aquí empleados no supondrán ningún problema en los sistemas embebidos de bajo coste, que estarán en el mercado en un periodo de tiempo no muy largo, por lo cual desde mi opinión es acertado investigar este tipo de materias con la vista puesta en la tecnología de dentro de unos años, pues pueden dar lugar a robots eficientes y baratos.

Hay que destacar que una de las mayores ventajas que aportan los CPGs es la autonomía, y la capacidad de desenvolverse ante situaciones imprevistas. Si bien es cierto que para este proyecto no se ha introducido feedback sensorial y los movimientos empleados están prefijados, no sería difícil adecuar todo este trabajo realizado para conseguir esta autonomía que hace tan útil a los CPGs.

6.2 Trabajo futuro

A partir del trabajo realizado en este proyecto, se pueden abrir nuevas áreas de estudio. Las opciones que aquí se exponen irán dirigidas al estudio de otros CPG para otros robots y la ampliación de funcionalidades de este robot.

Como se ha dicho, las tres configuraciones obtenidas dan mucho juego a la hora de obtener nuevos movimientos, pues ayudan a obtener fases deseadas con unas determinadas características. Esto puede ser útil a la hora de estudiar el movimiento de otros robots más complejos con CPG.

Por otro lado, se ha planteado continuar trabajando con este mismo robot e incorporarle un feedback sensorial, por el cual pueda cambiar de movimiento al recibir un estímulo determinado. Para ello se puede emplear todo el código desarrollado en este proyecto e incorporar funciones que actúen en tiempo real (pues actualmente los movimientos se guardan en un fichero que es posteriormente leído por la tarjeta). También, es necesario aportar al robot algún sensor que actúe como sentido, para el cual se ha pensado ya en una nariz electrónica.

Referencias

[1] Grillner S 2006 Biological pattern generation: the cellular and computational logic of networks in motion *Neuron* **52** 751–66

[2] Marder E and Bucher D 2007 Understanding circuit dynamics using the stomatogastric nervous system of lobsters and crabs *Annu. Rev. Physiol.* **69** 291–316

[3] Smarandache C, Hall W M and Mulloney B 2009 Coordination of rhythmic motor activity by gradients of synaptic strength in a neural circuit that couples modular neural oscillators *J. Neurosci.* **29** 9351–60

[4] Marder E and Bucher D 2001 Central pattern generators and the control of rhythmic movements *Curr. Biol.* **11** R986–96

[5] Selverston A I., Rabinovich M I, Abarbanel H D I, Elson R, Szucs A, Pinto R D, Huerta R and Varona P 2000 Reliable circuits from irregular neurons: a dynamical approach to understanding central pattern generators *J. Physiol. Paris* **94** 357–74

[6] Huerta R, Varona P, Rabinovich M I and Abarbanel H D 2001 Topology selection by chaotic neurons of a pyloric central pattern generator *Biol. Cybern.* **84** L1–8

[7] Stiesberg G R, Reyes M B, Varona P, Pinto R D and Huerta R 2007 Connection topology selection in central pattern generators by maximizing the gain of information *Neural Comput.* **19** 974–93

[8] Rabinovich M I, Varona P, Selverston A I and Abarbanel H D I 2006 Dynamical principles in neuroscience. *Rev. Mod. Phys.* **78** 1213–65

[9] Reyes M, Huerta R, Rabinovich M and Selverston A 2008 Artificial synaptic modification reveals a dynamical
invariant in the pyloric CPG *Eur. J. Appl. Physiol.* **102** 667–75

[10] Szucs A, Pinto R D, Rabinovich M I, Abarbanel H D and Selverston A I 2003 Synaptic modulation of the
interspike interval signatures of bursting pyloric neurons *J. Neurophysiol.* **89** 1363–77

[11] Latorre R, Rodr´ıguez F B and Varona P 2006 Neural signatures: multiple coding in spiking-bursting cells *Biol. Cybern.* **95** 169–83

[12] Kurokawa H, Tomita K, Kamimura A, Kokaji S, Hasuo T and Murata S 2008 Distributed self-reconfiguration of M-TRAN III modular robotic system *Int. J. Robot. Res.* **27** 373–86

[13] Yim M, Duff D G and Roufas K D 2000 PolyBot: a modular reconfigurable robot *Proc. IEEE Int. Conf. Robotics and Automation* vol 1 pp 514–20

[14] Ijspeert A 2008 Central pattern generators for locomotion control in animals and robots: a review *Neural Netw.* **21** 642–53

[15] Degallier S and Ijspeert A J 2010 Modeling discrete and rhythmic movements through motor primitives: a review *Biol. Cybern.* **103** 319–38

[16] Ayers J and Witting J 2007 Biomimetic approaches to the control of underwater walking machines *Phil. Trans. R. Soc.* A **365** 273–95

[17] Arena P, Fortuna L, Frasca M and Patane L 2005 A CNN-based chip for robot locomotion control *IEEE Trans. Circuits Syst.* **52** 1862–71

[18] Chung S-J and Dorothy M 2009 Neurobiologically inspired control of engineered flapping flight *J. Guid. Control Dyn.* **33** 440–53

[19] Seo K, Chung S-J and Slotine J J 2010 CPG-based control of a turtle-like underwater vehicle *Auton. Robots* **28** 247–69

[20] Manoonpong P, Geng T, Kulvicius T, Porr B and W¨org¨otter F 2007 Adaptive, fast walking in a biped robot under neuronal control and learning *PLoS Comput. Biol.* **3** 1305–20

[21] Aoi S and Tsuchiya K 2006 Stability analysis of a simple walking model driven by an oscillator with a phase reset using sensory feedback *IEEE Trans. Robot.* **22** 391–7

[22] Ijspeert A J and Kodjabachian J 1999 Evolution and development of a central pattern generator for the swimming of a lamprey *Artif. Life* **5** 247–69

[23] Kamimura A, Kurokawa H, Toshida E, Tomita K, Murata S and Kokaji S 2003 Automatic locomotion pattern generation for modular robots *IEEE Int. Conf. Robotics and Automation* **vol 1** pp 714–20

[24] Crespi A and Ijspeert A J 2008 Online optimization of swimming and crawling in an amphibious snake robot *IEEE Trans. Robot.* **24** 75–87

[25] Sproewitz A, Moeckel R, Maye J and Ijspeert A J 2008 Learning to move in modular robots using central pattern generators and online optimization *Int. J. Robot. Res.* 27 423–43

[26] F Herrero-Carrón 2011 Bio-inspired design strategies for central pattern generator control in modular robotics Bioinsp. Biomim. 6 (2011) 016006

[27] Gonzalez-Gomez J.,Zhang J.,Boemo E. and Zhang J. Locomotion Capabilities of a Modular Robot with Eight Pitch-Yaw-Connecting Modules

[28] Fundamentals of neural networks. Architectures, algorithms and applications. Laurene Fausett. Prentice Hall. 1994.

[29] Neural networks. A comprehensive foundation. Simon Haykin. Prentice Hall. 1999.

[30] http://www.nature.com/neuro/journal/v12/n10/full/nn.2401.html

[31]http://www.iearobotics.com/personal/juan/publicaciones/art14/download/clawar-2006-hypercube.pdf

[32] Rulkov N F 2002 Modeling of spiking-bursting neural behavior using two-dimensional map Phys. Rev. E 65 041922.

Glosario

CPG	Generador Central de Patrones
ANNs	Artificial Neural Networks
GNB	Grupo de Neurocomputación Biológica de la Escuela Politécnica Superior, UAM.

PRESUPUESTO

1) Ejecución Material

- 8 Motores Futaba ..... 80 €
- 8 Módulos de metacrilato..... 20 €
- Cables y conectores …….5 €

2) Total presupuesto

- Total Presupuesto..... 105 €

Madrid, septiembre de 2011

Fdo.: Damián Zamorano Martín
Ingeniero Superior de Telecomunicación

PLIEGO DE CONDICIONES

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un control del sistema de locomoción de robots utilizando CPGs. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es

obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.