Universidad Veracruzana

Doctorado en Inteligencia Artificial

Arquitectura Modular Aplicada a la Robótica Evolutiva Inspirada en la Acción del Neurotransmisor Acetilcolina sobre Neuronas Efectoras

TESIS

Que para obtener el grado de:

Doctor en Inteligencia Artificial

Presenta:

Fernando Aldana Franco

Director de tesis:

Dr. Fernando Martín Montes González

Revisores:

Dr. Alberto Ochoa Zezzatti
Dra. Rusalky Del Ángel Ortiz
Dra. Ericka Janet Rechy Ramírez
Dra. Marcela Quiroz Castellanos
Dr. Antonio Marín Hernández

Xalapa de Enríquez, Veracruz.
22 de noviembre de 2017.

Dedicatoria

Esta tesis fue realizada con la ayuda de una **beca CONACYT nacional** número **236204** otorgada para el periodo **Agosto 2013 – Julio 2017**, y una **beca CONACYT mixta** número **290842** otorgada por el periodo **Febrero - Agosto 2015**.

A mi esposa amada Mónica por todo tu amor, paciencia, apoyo y cariño.

A mi hijo Rodrigo que es mi adoración. Espero que en un futuro tengas la oportunidad de leer este trabajo y te sientas muy orgulloso de tu padre.

A mi madre Rosario de quien me siento profundamente orgulloso, por todo su apoyo. Pero sobre todo por ser mi ejemplo y mí guía no sólo en la vida en general, sino dentro de la investigación.

A mi hermano Miguel por ser mi amigo y compañero de muchas etapas en la vida. Pero también por mostrarme que la disciplina, el empeño y la tenacidad son claves para alcanzar nuestras metas.

A mi abuelo, a quien no dejo de extrañarlo un sólo minuto. Siempre estuviste y estarás conmigo.

A mi profesor Fernando Montes, por ayudarme y guiarme con esta investigación.

Al profesor Stefano Nolfi, por su apoyo y enseñanzas. Tuve el privilegio de conocer a uno de los máximos investigadores de mi campo de trabajo. Lo que me permitió mejorar mis habilidades y perspectivas de trabajo.

A mis revisores por el tiempo dedicado a la mejora de este trabajo de investigación.

A mis profesores que durante mi formación me han ayudado con sus enseñanzas.

A todas aquellas personas que de manera directa o indirecta contribuyeron a este trabajo.

Anexo 5. Implementation of a simulation model for ROV in a decision task problem.

Anexo 6. Acetyl-modulated architecture for evolutionary robotics.

Índice de figuras.

Índice de tablas.

Índice de gráficas.

Resumen.

La Robótica Evolutiva (RE) es un área de la Inteligencia Artificial (IA) que en los últimos veinte años ha cobrado una especial relevancia gracias al enfoque bio-inspirado y transdiciplinar utilizado para producir resultados tanto a nivel de hardware (morfología) como a nivel de software (control).

Al tratarse de una rama del conocimiento relativamente nueva, ya que su inicio se remonta a la última década del siglo XX, se encuentra abierta a múltiples retos y problemáticas de distinta índole. Un buen ejemplo de ello son los esfuerzos por disminuir los tiempos de convergencia de los algoritmos utilizados, generar nuevas estructuras de control y morfología que permitan resolver problemas más complejos en términos de comportamientos o metas, además de los métodos propuestos para generar componentes morfológicos evolutivos.

La influencia de los métodos bio-inspirados en el área de la Inteligencia Artificial y en particular de la Robótica Evolutiva es muy fuerte, lo que produce el interés de grupos alrededor del mundo por vincular sus investigaciones con este enfoque. Un buen ejemplo de ello es la modularidad en los seres vivos.

La modularidad se apoya en el fenómeno presente en todos los seres vivos en donde diferentes secciones de varios aparatos se encargan de solventar diversas funciones. Por ejemplo, en el sistema digestivo interactúan diferentes aparatos como el estómago, los intestinos, el esófago, para lograr una meta común. Así se divide una meta, propósito o trabajo por desarrollar para hacerlo de una manera eficiente mediante la coordinación de diferentes aparatos, regiones, sistemas y estructuras. Se pueden identificar dos tipos de modularidad en los organismos: interna y externa. La modularidad interna se refiere a la capacidad de algunos aparatos u órganos para resolver más de una tarea. Por ejemplo, el lóbulo temporal del cerebro humano que atiende funciones de audición, lenguaje y memoria. La modularidad externa es la capacidad de algunos aparatos u órganos para colaborar con una fracción de la resolución de una tarea general. Un ejemplo de ello es la coordinación que existe entre las regiones del lóbulo temporal, el área de Wernicke, y el Lóbulo Parietal del cerebro humano para la resolución de tareas de comprensión de lenguaje.

La modularidad en robótica evolutiva, en una forma de representación abstracta, puede ser modelado y aplicado en forma de sistemas de control. Esto permite producir sistemas de control más grandes e intrincados en términos de la cantidad de módulos y su interacción. Lo que a su vez representa la posibilidad de evitar fenómenos como el bloqueo de espacios de búsqueda. Un ejemplo de este bloqueo aparece cuando un sistema de control, en particular una Red Neuronal Artificial (RNA), debe ser evolucionada para cumplir con dos o más tareas, pero el algoritmo de optimización no puede entregar buenos resultados. Éste fenómeno computacional es provocado por que los óptimos locales que favorecen a la primera tarea no son los mismos o son distantes de los que favorecen a la segunda tarea. Debido a ello el proceso evolutivo es incapaz de producir sistemas de control que cumplen con la tarea requerida. Así, los módulos representan potencialmente la división en fragmentos de ese espacio de soluciones inicial, en donde cada

módulo resulta una fracción del problema inicial. Por ello la función más importante de la modularidad en el área de la RE es que permite reestructurar espacios complejos de búsqueda, lo que permite generar nuevos espacios que sean más amigables con el proceso evolutivo en términos de su exploración, y obtener comportamientos adecuados para resolver las diferentes tareas.

La finalidad de esta investigación es la construcción y validación de una arquitectura modular de comportamientos haciendo uso de la robótica evolutiva. La arquitectura propuesta está inspirada en la acción del neurotransmisor acetilcolina (ACh) sobre las neuronas motoras del cuerpo humano y la acción de neuronas de Renshaw como colaterales inhibidoras. La acetilcolina regula la sinapsis química de las neuronas efectoras, permitiendo así la activación del movimiento de los músculos y el esqueleto en general; se trata de un neurotransmisor que está relacionado con otros órganos del cuerpo y con enfermedades como el Parkinson y la adicción a la nicotina.

Como parte del proceso de construcción de la arquitectura propuesta, se realizó un experimento exploratorio para encontrar una topología de red que evitara el fenómeno de la monopolización modular, cambiando el dominio de aplicación de la RE a una tarea de clasificación de la base de datos Fisher Iris, que permitió cuantificar el funcionamiento y la activación correctos de diferentes módulos.

Además, se diseñaron un par de experimentos comparativos que validaron el funcionamiento adecuado de la arquitectura propuesta en tareas de robots evolutivos en dos entornos con tareas diferentes. Se contrastaron los resultados de tres diferentes sistemas de control basados en redes neuronales artificiales: no modular, *arquitectura modular emergente* (modularidad interna) y la arquitectura *acetilmodulada* (modularidad externa). Los resultados para la validación de la arquitectura fueron obtenidos a partir de dos entornos diferentes de tareas evolutivas: limpieza de entorno y cambio de cuarto. En ambos casos el simulador empleado fue Webots (Michel, 1998), se empleó el modelo computacional del robot e-puck y el proceso evolutivo estuvo a cargo de un algoritmo genético generacional. El proceso de optimización se basó en la descomposición de términos de la función de calidad en sub-tareas representadas en cada módulo.

Finalmente, se desarrolló un experimento analítico sobre la relación que existe entre los sistemas de comunicación y los sistemas de control modulares aplicados a robots evolutivos. Los sistemas de comunicación proveen una herramienta adicional de descripción y exploración del medio a los robots. Para ello se utilizó la tarea de limpieza de entornos para un grupo de robots y se analizaron la interacción y los beneficios de ambos factores dentro del marco de la tarea.

Este documento se encuentra dividido como se describe a continuación. En el **Capítulo 1** titulado **Introducción** se delimita la investigación a partir de la definición del problema, la justificación de su estudio, el planteamiento de los objetivos, de las limitaciones y de la hipótesis. El **Capítulo 2** que tiene como título **Marco teórico**, se presentan trabajos relacionados y definiciones vinculados con esta investigación desde el punto de vista de la robótica evolutiva (morfología y control), los procesos evolutivos en el área, la modularidad, la sinapsis química, la acetilcolina, las neuronas de

Renshaw y los sistemas de comunicación. Para el **Capítulo 3**, nombrado **Metodología**, se muestra el marco metodológico que incluye detalle de la arquitectura propuesta, las herramientas utilizadas y la descripción de los experimentos realizados. El **Capítulo 4** muestra los resultados de los experimentos ejecutados y una discusión sobre los mismos, el título de este capítulo es **Resultados**. Finalmente, el **Capítulo 5** expresa las conclusiones sobre la investigación, el trabajo relacionado y el trabajo futuro, el nombre de este capítulo es **Conclusiones**. En los anexos se incluye los artículos publicados como producto de la investigación.

Resumen.

Capítulo 1.

Introducción.

En los párrafos que forman el Capítulo 1, se delimita el problema a resolver mediante el planteamiento, la justificación de la investigación, la hipótesis de trabajo, los objetivos y las limitaciones de la investigación. De esta manera, se busca introducir al lector a la problemática resuelta durante esta investigación.

1.1 Planteamiento del problema.

La Robótica Evolutiva (RE) es una rama del conocimiento que surgió a principios de 1990. Su autoría se disputa entre dos grupos de investigación en el mundo, uno en la Universidad de Sussex en Inglaterra y otro en el instituto de Psicología del Consejo Nacional de Investigaciones de Italia. Se trata de una rama de la robótica con la que se busca crear sistemas de control y morfología para robots, basados en la acción de procesos evolutivos artificiales. Para ello, se utilizan herramientas de Inteligencia Artificial (IA) como las Redes Neuronales Artificiales (RNA) y los algoritmos del Cómputo Evolutivo (CE), mediante esta combinación se busca crear entornos que provoquen presiones evolutivas en los individuos a evolucionar u optimizar. Dichas presiones producen una mejora en la aptitud de los individuos, lo que permite alcanzar diversos objetivos evolutivos.

La mayoría de las investigaciones, en el ámbito de la robótica evolutiva, utilizan RNAs como elemento principal de control, en gran medida por la capacidad de discriminación y adaptabilidad de las mismas. Así, se optimizan neuro-controladores con la capacidad de resolver diferentes tareas o desarrollar comportamientos diversos, pero existen tareas en robótica evolutiva en las cuales no es posible optimizar un sistema de control en forma de una sola red neuronal artificial. Por ejemplo, en tareas de limpieza de entornos resulta complicado, para los algoritmos de computación evolutiva, optimizar una red neuronal artificial que cumpla con las tareas de recolección y depósito de basura (Calabretta, Nolfi, Parisi y Wagner, 1998a). En dicho caso, una sola red neuronal es incapaz de producir que el robot levante y deposite basura del entorno, obteniendo individuos aptos para realizar, con buenos niveles de desempeño sólo una de las dos tareas. Esto puede quedar más claro con el siguiente ejemplo: un espacio de soluciones está compuesto por las configuraciones de un robot con una determinada cantidad de sensores y actuadores, el robot debe resolver dos tareas (A y B), pero los óptimos locales de la tarea A pueden no ser los mismos de una tarea B. Si los espacios de las tareas A y B se tratan como espacios individuales, los problemas por interferencia entre funciones se pueden evitar y así generar tiempos de convergencia considerablemente menores.

En otras palabras, la incapacidad de optimizar un sistema de control con una sola red neuronal artificial (por sus siglas RNA) ocurre porque existe un bloqueo entre tareas que afecta al espacio de soluciones potenciales que el algoritmo de computación evolutiva busca atacar. Esto quiere decir, que la responsabilidad de los malos resultados para alcanzar los objetivos evolutivos

producidos por este efecto recae directamente en la herramienta de control utilizada (RNA) y su representación en el espacio de búsqueda. De acuerdo a Jacobs, Jordan y Barto (1991), dicho problema se conoce como bloqueo de espacios de soluciones o bloqueo neuronal cuando la estructura a optimizar involucrada es una RNA.

1.2 Justificación.

Uno de los retos abiertos en el área de la robótica evolutiva es generar estructuras de control complejas para funciones, tareas o comportamientos complejo (Meyer, 1998). Para Floreano y Urzelai (2000), otros campos que cubren los retos abiertos en el área son: Proceso de evaluación, Genética, Plasticidad, Nivel y Modo de Búsqueda, Operadores y Desempeño.

El primer concepto relevante es la modularidad, que es un fenómeno que se encuentra con frecuencia en los seres vivos, particularmente en el ser humano. Para Moschovitch y Umilta (1991), la cognición humana está compuesta por una infinidad de módulos especializados que permiten el desarrollo de conductas complejas.

Para Togelius (2004) la modularidad es un camino para generar modelos de control útiles en RE, por lo que debe ser rescatada y aplicada en el área. Así los robots evolutivos podrán estar equipados con un sistema de control dividido en varios sub-sistemas, cuyo propósito final es resolver una tarea general. Este pensamiento coincide con las ideas de Santos y Duro (2005), quienes aseveran que, al tratar con comportamientos, funciones o tareas complejas en robots, una buena solución es utilizar la estrategia “divide y vencerás”.

Pero la modularidad, apreciada desde el punto de vista computacional, no se reduce a la construcción, mediante bloques, de sistemas de control complejos en RE. El enfoque puede transferir al problema de interés que es el de las búsquedas en el espacio de soluciones y los bloqueos que puedan existir en ellos; esto corresponde a rutas de investigación sugeridas en el párrafo previo: Nivel y modo de búsqueda. El nivel se refiere a la cantidad del espacio de soluciones que se puede explorar, así como su dimensionalidad. En el caso del modo es la manera en que se realiza la exploración.

De esta manera, un espacio de soluciones complejo o difícil de explorar puede ser dividido en sub-objetivos, lo que genera espacios de búsqueda más sencillos de explorar. Lo que coincide con lo propuesto por Harvey, Husbands y Cliff (1993), quienes consideran que uno de los principales problemas de la RE es el alto costo computacional, en términos de tiempo y espacio del proceso evolutivo.

Existen soluciones con las que se busca atacar este problema desde diversos enfoques, pero la mayoría acarrean problemas adicionales. Entre ellos destacan la monopolización de recursos por parte de una región de la red neuronal durante la toma de decisiones. Esto quiere decir que, durante todo el tiempo de prueba de un neuro-controlador, sólo uno de los módulos toma parte en las soluciones. Otro inconveniente con las soluciones sobre el bloqueo en espacios de

búsqueda es la necesidad del uso de operadores especiales en el proceso evolutivo, o el gran consumo de recursos computacionales (tiempo y espacio de procesamiento) por parte del proceso evolutivo artificial. O la necesidad de interconectar diferentes redes neuronales, volviendo las salidas de etapas previas en las entradas de las siguientes etapas, que es el resultado del diseño de modelos jerárquicos. Todas estas razones vuelven a las soluciones existentes, soluciones incompletas. Por ello resulta importante proponer una nueva técnica que permita solucionar el bloqueo que ocurre en los espacios de soluciones potenciales.

1.3 Hipótesis.

"Un modelo computacional de control para robótica evolutiva basado en redes neuronales artificiales que funcionen con el enfoque de los módulos externos, e inspirado en componentes neuronales, representa una solución al problema de los bloqueos en espacios de soluciones potenciales en robótica evolutiva".

1.4 Objetivos.

1.4.1 Objetivo general.

El objetivo principal de la investigación es diseñar y validar una arquitectura de control para robótica evolutiva basada en redes neuronales artificiales mediante el fenómeno de la modularidad externa.

1.4.2 Objetivos específicos.

Los objetivos secundarios son:

- Diseñar un sistema de control de RNAs con un sistema de interacción, basado en la modularidad externa, las propiedades de la sinapsis química y la acetilcolina como neurotransmisor, además de las neuronas de Renshaw para lograr inhibiciones a células laterales.
- Realizar un estudio exploratorio para encontrar la mejor configuración de módulos externos para redes neuronales que evite la monopolización de recursos por parte de un sólo módulo. El uso de esta base de datos permite la cuantificación de la activación de los módulos dadas las clases.
- Comparar el desempeño de la arquitectura propuesta contra el funcionamiento de un control no modular y uno de módulos internos en un entorno en donde el bloqueo entre tareas ha sido identificado previamente: limpieza de entorno.
- Comparar el desempeño de la arquitectura propuesta contra el funcionamiento de un control no modular y uno de módulos internos, en un entorno que requiere una mayor cantidad de módulos para la solución: cambio de cuarto.
- Analizar la interacción entre el sistema modular y un sistema de comunicación emergente en una tarea de limpieza de entorno para un grupo de robots evolutivos.

1.5 Antecedentes.

Existen sistemas de modularidad interna aplicada a robots evolutivos que han probado ser efectivos puesto que generan mecanismos de presión evolutiva que permiten la especialización de ciertos módulos de una red para resolver un problema determinado, con un costo de monopolización de los recursos que reducen al mínimo la participación en la solución general y la interacción del resto de los módulos (Nolfi, 1997). Sin embargo, con este enfoque se logran estructuras de una sola red divididas en diferentes regiones, en donde cada una es responsable de resolver una parte del problema general. Además, la evolución de los controladores basados en modularidad interna depende de operadores genéticos especiales, como lo es el de duplicación para mejorar su desempeño (Calabretta, Nolfi, Parisi y Wagner, 1998b). Lo que se realiza con este operador es una copia funcional de algunos pesos de la red, para ejercer presión evolutiva y mejorar el rendimiento de los módulos.

Un resultado que demuestra que el control modular externo es una buena solución al problema de los bloqueos en espacios de solución es el trabajo de Aldana-Franco y Montes (2015), en donde se compara una arquitectura heterogénea (*monored*) y una modular dividida (dos redes sin niveles de interacción entre sí) aplicadas al control de tareas motrices y de emisión de sonidos para robots e-puck. En dicha investigación se muestra que el proceso de evolución de un control heterogéneo de ciertas tareas motrices bloquea el proceso evolutivo del mismo control para algunas tareas de emisión de sonidos, lo que provoca una calidad baja en los individuos. Esta tendencia se modifica cuando se dividen los espacios de búsqueda en términos de los actuadores que utiliza el robot para desarrollar sus funciones, lo que a su vez representa una arquitectura modular externa con dos sistemas de control cuya interacción se reduce al mínimo debido a que no contienden por el uso de recursos.

Existen algunas representaciones de modularidad externa que se basan en modelos jerárquicos. Por ejemplo, Manoonpong, Pasemann y Fischer (2005) proponen un sistema basado en tres redes neuronales para el control de robots con patas. Dos de las redes son de nivel más bajo y sus salidas alimentan las entradas de la tercera red. O la investigación de Duarte, Oliveira y Christensen (2015) en donde se busca adaptar la arquitectura de subsumsión de Brooks (1986) a redes neuronales. En ella se encuentran diferentes comportamientos representados por redes neuronales, en donde los de mayor jerarquía subsumen a los de menor.

En la presente investigación se plantea como estrategia generar una solución mediante el uso de la modularidad externa para el problema de la redimensionalización de los espacios de exploración para el ajuste de los pesos de redes neuronales artificiales. La solución propuesta involucra un modelo basado en un sistema de redes neuronales artificiales, cada una encargada de solucionar una parte de un problema más grande. Además de contar con una red neuronal artificial que regula las interacciones de los módulos, utilizando el enfoque de los neurotransmisores, como la acetilcolina cuya función es regular el movimiento voluntario de los seres humanos, el fenómeno de la sinapsis química y los mecanismos de inhibición para neuronas laterales, como el de las neuronas de Renshaw.

En otras palabras, en la arquitectura modular propuesta utiliza un sistema de interacción que permite a los diversos módulos competir por el control de los recursos del robot. El sistema propuesto está inspirado en la acción del neurotransmisor acetilcolina. Dicha red produce cantidades de un neurotransmisor que actúa sobre las neuronas de salida (efectoras) de cada uno de los módulos. La presencia o no del neurotransmisor en cada módulo, así como la cantidad de concentración, es el mecanismo de mediación para otorgar el control de los motores a los diferentes módulos. Dicho sistema de interacción es definido por la interacción del robot con el medio (emergencia del sistema), y no por el conocimiento *a priori* del programador.

1.6 Limitaciones.

Entre las principales limitaciones de este trabajo se encuentran:

- Experimentación con los robots e-puck en un ambiente virtual del simulador Webots.
- El proceso de evolución artificial es ejecutado exclusivamente mediante algoritmos genéticos tradicionales.
- Los elementos de control básicos para cada módulo son las redes neuronales artificiales.
- El proceso de división modular está basado en la función de calidad y sus diferentes términos matemáticos.
- Las sustancias manejadas en los sistemas son virtuales y se trata de modelos básicos computacionales.
- El sistema de comunicación utilizado está basado en los LEDs de los robots y sus cámaras. La interpretación de cada señal, así como la decisión del robot con el que se entabla el ciclo de comunicación depende del proceso evolutivo.

Capítulo 2.

Marco teórico.

En este capítulo se muestra una amplia perspectiva de la rama del conocimiento de la robótica evolutiva, poniendo especial atención a las investigaciones sobre morfología, control y el proceso evolutivo. De la misma manera, se exhiben los avances de investigaciones en la materia de modularidad aplicada a redes neuronales artificiales, principalmente cuando se emplean como elementos de control de robots evolutivos. También se abordan los temas relacionados con los neurotransmisores, en especial la acetilcolina utilizado como sustancia precursora de la sinapsis química en neuronas motoras. Adicionalmente se encuentran referencias a los sistemas de comunicación aplicados a robots evolutivos y sus beneficios potenciales. Todos esos trabajos sirven como antecedentes de este trabajo de investigación.

2.1 Robótica Evolutiva.

Dentro de la robótica existen muchas sub-disciplinas con notables contrastes entre sí, tal es el caso de la tele operación (Janipireddy, Saeed y Saeed, 2017), (Aldana-Franco, Montes-González y Ochoa, 2017b), la conductual (Gage y Murphy, 2003), la autónoma (Sharkey y Ziemke, 1998), entre muchas otras. Una de estas ramas es la robótica evolutiva que nace a principios de la década de los 90s. El área está envuelta en un marco de bio-inspiración muy importante, acentuado en los aspectos que involucran a la teoría de la evolución y la adaptabilidad de las especies (Eiben y Smith, 2016).

En esta área del conocimiento es posible combinar estructuras de inteligencia artificial como las redes neuronales artificiales, las redes bayesianas, los árboles de decisión, los sistemas de lógica difusa, entre otras, con herramientas del cómputo evolutivo. Dicha combinación permite optimizar o evolucionar el control o morfología de los robots.

Con referencia a los orígenes del área, éstos se remontan a su aparición como una alternativa a la robótica basada en comportamientos que buscaba evitar la complejidad en el diseño y las interacciones de los sistemas de control de la robótica conductual. Este proceso debe desarrollarse a partir de la experiencia del robot en su medio.

Así, el principal propósito de la RE es optimizar los comportamientos a generar, y no que sean desarrollados por un programador bajo su experiencia (Cliff, Harvey y Husbands, 1992). Dicho de otra manera, lo que se busca en RE no es que el sistema de control o de morfología se desarrolle basado en la experiencia del programador, sino en la experiencia del robot mismo. Esto ha generado que el diseño de los componentes desarrollados se base en el punto de vista proximal del problema, tratando de evitar en la medida de lo posible el punto de vista distal. Esto es remarcado en su momento por uno de los fundadores del área, Harvey (200), el autor considera que utilizando la evolución ciega para un sistema proximal es posible obtener sistemas dinámicos complejos que pueden resultar opacos e incomprensibles para el analista humano. Sin embargo,

en los últimos años se ha flexibilizado este criterio, (SIlva, Duarte, Correia, Oliveira y Christensen, 2016). Lo que ha permitido que en algunas ocasiones los diseñadores puedan brindar un impulso inicial a los sistemas en búsqueda (Rubinacci, Ponticorvo, Gigliotta y Moglino, 2017), (Lei, Manier y Wang, 2017), (Gravina, Liapis y Yannakakis, 2017) para la obtención de sistemas de control variados que permitan resolver tareas más difíciles. Esto sin perder de vista que la generación de los comportamientos o de los mecanismos para la resolución de la tarea proviene del proceso evolutivo mismo. Entre las tareas más difíciles que se pretenden resolver en la actualidad se encuentra la resolución de laberintos con formas variadas, modelado de sistemas de control para robots con múltiples patas, trabajos para comunidades de robots e inclusive modelos de locomoción en forma de gateo para robots humanoides (Mouret y Chatzilygeroudis, 2017), (André y Nolfi, 2016).

Una definición para la RE es propuesta por Bongard (2008). En ella se considera a la RE como la rama del conocimiento que es utilizada para la obtención de componentes artificiales que intentan reproducir características de los seres vivos, y que en su mayoría toman la forma de sistemas complejos. Desde un punto de vista diferente, el que involucra a la Vida Artificial como ciencia, Floreano y Mondada (1998) consideran que la RE explora la vida tal como es y cómo podría ser.

Según señalan Harvey, Di Paolo, Wood, Quinn y Tuci (2005), en RE se recogen los modelos de la neurociencia computacional para ser insertados en cuerpos y entornos simulados, como también lo sugieren Zimmer y Doncieux (2017) quienes llaman a esta técnica neuro-evolución. Harvey *et al*. (2005), agregan que el proceso de evolución es regulado por una función matemática que busca satisfacer una tarea o conducta. Además mencionan que se debe hacer lo posible por probar los modelos en entornos reales, aunque esto último ha dejado de intentarse en los últimos años debido al creciente poder de cómputo de los simuladores.

Clark (1998) conceptualiza a la RE como una conjunción de diferentes áreas de la Inteligencia Artificial y otras áreas de la ciencia cognitiva que trata de estudiar al cerebro, el cuerpo y el ambiente como un todo. Este todo se representa mediante un aparato sensorio motriz que se modifica mediante la interacción con su entorno y con su morfología externa (forma del robot) e interna (sistema nervioso del robot), expresando capacidades cognitivas (Ziemke, Bergfeldt, Buason, Susi y Svensson, 2004).

Barandiaran (2003) dice que la RE es una herramienta que busca generar artefactos inteligentes. Para ello utiliza un proceso de simulación como un símil de la evolución natural, que permite estudiar la interacción entre conocimiento y evolución. Se trata entonces de una fusión conceptual entre mecanismos biológicos y cognitivismo, que permite estudiar la interacción entre diferentes factores como: evolución, acoplamiento sensorio-motriz al entorno, mecanismos neuronales y autorregulación de sistemas cognitivos.

En esta investigación se propone que la RE se puede definir como la rama del conocimiento que produce elementos de control o morfología de robots. En ella se utilizan herramientas del cómputo evolutivo y de la inteligencia artificial, mediante un proceso de búsqueda en un espacio de solución que trata de replicar los procesos evolutivos y de adaptación presentes en la naturaleza.

Nolfi y Floreano (2000) definen el trabajo de investigación en RE como el producto de colocar un agente cognitivo (simulado y corpóreo) situado en un entorno sensorio-motor, cuyos mecanismos son capaces de reproducir conductas cognitivas.

Para Pratihar (2003) en la robótica evolutiva existen dos factores que interaccionan: La evolución y el aprendizaje. Son dos mecanismos de acción biológica, que aplicados a la RE actúan en tiempos diferentes pero que pueden resultar complementarios. La evolución captura pequeños cambios en el ambiente que afectan al nivel generacional. Por su parte el aprendizaje produce cambios adaptativos durante la vida de los individuos. Según este mismo autor, un robot evolutivo es aquel que se somete a un proceso evolutivo artificial con la finalidad de obtener algún componente de morfología o control.

Calabretta y Parisi (2005) aseveran que la RE representa una corriente del pensamiento conocida como Conexionismo Evolutivo, en donde la mente es vista como el resultado de interacciones fisicoquímicas que tienen lugar en las redes neuronales. Este proceso es resultado de la interacción entre procesos evolutivos y de aprendizaje. Este enfoque representa un proceso de diseño automático, que se encuentra presente en la naturaleza mediante tres procedimientos muy importantes: Evolución, desarrollo en tiempo de vida y aprendizaje (Kodjabachian y Meyer, 1998).

En el área de la RE, el uso del primero (evolución) es el pilar fundamental. El segundo (desarrollo en tiempo de vida) es un elemento clave que permite evaluar a los individuos de un proceso evolutivo. Mientras que el tercero (aprendizaje) ha cobrado importancia en los últimos años ya que permite el desarrollo de habilidades que no pueden ser codificadas en el Genotipo del individuo. Los ajustes a las competencias de los robots son desarrolladas por diferentes algoritmos de aprendizaje automático dentro del proceso evolutivo (Meeden, 1996).

Un robot evolutivo puede dividirse en tres componentes: morfología y control, proceso evolutivo y proceso de evaluación. La morfología y el control involucran a los elementos físicos del robot y a la herramienta de IA destinada al manejo de las situaciones en las que el robot se ve inmerso. El proceso evolutivo es el proceso de optimización o de búsqueda computacional para encontrar la mejor configuración del elemento de control o morfología del robot. En la actualidad se combina con procesos de aprendizaje automático para la refinación de la búsqueda (Palacios-Leyva, Cruz, Montes-González, Rascón y Santos, 2013). El proceso de evaluación permite evaluar a cada uno de los individuos durante todo el proceso evolutivo. Se utiliza una función matemática que guía el proceso de búsqueda. Dicha función expresa las características buscadas en el individuo ideal del problema a resolver.

2.1.1 Morfología y control.

En cuanto a la evolución de la morfología, ésta no ha sido tan explotada por los grupos de investigación tal vez debido a la falta de herramientas que permitan simular la evolución artificial de los cuerpos de robots. Esto puede ser una consecuencia del alto costo computacional que puede implicar este proceso de optimización. Sin embargo, la optimización de la morfología de los robots involucra un proceso de evolución más sensible a la vista del público. Por ende, puede generar una sensación de que lo realizado es más cercano a lo que ocurre en el mundo real, ya que los componentes físicos se van modificando durante el proceso evolutivo.

Un ejemplo del trabajo en morfología de la robótica evolutiva es presentado por Thompson (1997), el cual consiste en un modelo de reestructuración del diseño electrónico a partir de un FPGA (Field Programmable Gate Array por sus siglas en inglés) para un robot Khepera (Harlan, Levine y McClarigan, 2001) que debe evitar las colisiones con los obstáculos presentes en su ambiente.

Otro ejemplo es reportado por Dellaert y Beer (1996), quienes utilizan herramientas del cómputo evolutivo como la coevolución competitiva. Contrario a lo que podría pensarse, este proceso no se aplica a la morfología y al control como dos entes que compiten por mejorar generación tras generación. Se aplica a individuos compuestos por morfología y control que compiten con otros individuos para encontrar la mejor configuración posible, dadas las restricciones del espacio de búsqueda. Un ejemplo final es presentado por Jelisavcic *et al.* (2017), quienes proponen una plataforma para el desarrollo de robot en un entorno real, el cuál va a adecuando piezas según las presiones evolutivas que se van generando en el medio ambiente.

En cuanto a los sistemas de control, existen algunas estructuras de IA que han sido utilizadas como herramientas para generar sistemas de control de robots. Entre ellas se encuentran los sistemas de lógica difusa. Mendel (1995) describe un sistema de control para un robot móvil autónomo basado en un sistema de lógica difusa, cuyas funciones de pertenencia son ajustadas mediante un proceso evolutivo artificial. O el trabajo de Jamil, Jalani y Ahmad (2016) en donde se optimiza un sistema de lógica difusa que funcione como sistema de control para el movimiento de un brazo robótico.

Otra estructura utilizada como sistema de control para robots en el campo de la RE son los sistemas de clasificación basados en reglas. En este caso el proceso evolutivo tiene como finalidad optimizar y generar las reglas que componen al sistema de control. Tal es el caso de la investigación Kodjabachian y Meyer (1995), en donde a partir de programas genéticos es posible crear un sistema de control basado en árboles de decisión.

Las redes bayesianas también son utilizadas como herramienta de control de robots evolutivos. El proceso evolutivo puede involucrar la optimización de las estructuras de la red y/o las probabilidades asociadas a cada nodo. Se trata de un caso muy similar a lo que sucede con las redes neuronales artificiales. Un ejemplo del uso de redes bayesianas en RE es presentado por

Inamura, Inaba e Inoue (2000) quienes evolucionan un controlador de robots basado en una red bayesiana, en donde cada nodo corresponde con el estatus de un comportamiento. O el trabajo presentado por Calandra, Seyfarth, Peters y Deisenroth (2016), en donde se utiliza una red bayesiana para el control de locomoción de un robot humanoide.

Aunque algunos grupos de investigadores en el mundo han tratado de diversificar sus investigaciones, la mayoría de los trabajos en el área se enfocan a la construcción de sistemas de control para robots usando redes neuronales artificiales como eje fundamental de los mismos. Por ello es importante revisar la historia de estas estructuras y su aplicación al área.

2.1.2 Redes Neuronales Artificiales.

En 1943 McCulloch y Pits (1943) presentaron el primer modelo de red neuronal. Dicho modelo consiste en funciones de activación de dos estados: encendido y apagado. Algunos años después, para ser más precisos en 1949, Hebb agrega la primera regla que le permitía a las redes el aprendizaje de patrones, y que en la actualidad ha dado origen al aprendizaje Hebbiana (Hopfield, 1982), (Sanborn y Chater, 2017), que se basa en la modificación de los pesos en proporción de la entrada, la salida de la red y un ritmo de aprendizaje.

Pero no fue hasta 1958 que la investigación de nuevos modelos de redes neuronales mostró un avance importante. El Perceptrón, desarrollado por Rosenblatt (1958), es un modelo de RNA de una sola capa cuya principal característica es la clasificación de problemas linealmente separables y una regla de aprendizaje que permite la corrección del error. El gran aporte de este modelo es que toma en cuenta la suma ponderada de las entradas para generar una señal de salida del tipo lógica. Dicha salida depende de un umbral de activación. Esto es algo común en los modelos multicapas con los que se trabaja en la actualidad.

Es precisamente el uso de modelos multicapas el hecho que terminó revolucionando el campo de estudio en la década de los 80s. Este tipo de modelos permiten resolver problemas que no son linealmente separables mediante la regionalización del espacio de clasificación y el uso del algoritmo de retro-propagación del error como sistema de aprendizaje (Whittington y Bogacz, 2017).

Una RNA según Rusell y Norving (2004), es un modelo compuesto por nodos que se conectan entre sí a partir de pesos numéricos que se ajustan durante el proceso de aprendizaje. En el caso de la RE los pesos se ajustan durante el proceso de evolución artificial, pero también pueden ser ajustados por un subproceso usando un algoritmo de aprendizaje.

Cazorla *et al.* (1997) apuntan algunos de los elementos que se encuentran de manera tradicional en un RNA son (ver Figura 1):

1. Nodos o neuronas. Las unidades que componen la red.
2. Entradas. Los patrones que son presentados a la red.
3. Pesos sinápticos. Sirven para determinar la influencia que tiene una entrada en la activación de cierto nodo dentro de la red. Son números reales que en caso de ser positivos representan conexiones excitatorias, y en el caso de números negativos representan conexiones inhibitorias. Las conexiones pueden ser: Feedforward, Backward, Estratificada, Acíclica, Totalmente conectadas, Modular, entre otras.
4. Función de combinación, entrada neta (NET) o de entrada. Comprende la influencia que tienen los pesos sinápticos en todas las entradas.
5. Función de activación. Determina, como su nombre lo indica, si una neurona se encuentra en estado activo o no. Puede tomar la forma de diversas funciones matemáticas, entre las más comunes están: escalón unitario, sigmoidal y rampa.
6. Función de salida. Permite determinar el valor que devuelve la neurona.

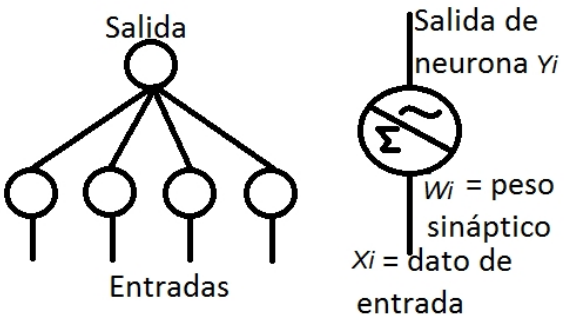


Figura 1. Representación gráfica de una red neuronal multicapa. Se muestra la capa de entrada y la capa de salida de una red. Además de los componentes de una neurona artificial, entre los que destacan los datos de entrada, el peso sináptico, la función NET, la función de activación y su salida.

Como señalan Nolfi, Elman y Parisi (1994), las RNAs son especialmente útiles en RE porque son altamente tolerantes a fallos y ruidos que pueden provenir de los sensores y actuadores del robot en su entorno físico. En un ambiente simulado deben agregarse niveles de ruido como parte del enriquecimiento del mundo representado. Al trabajar con estas herramientas computacionales no es necesario conocer en su totalidad la naturaleza del fenómeno a representar o modelar, sobre todo en términos de sus relaciones internas.

El proceso evolutivo artificial en las redes neuronales puede afectar dos componentes de la red (ver Figura 2): la estructura interna de la red y los pesos de las conexiones sinápticas. Cuando se optimiza la estructura de la red, se utiliza una codificación especial en el genotipo que involucra las conexiones de cada neurona, así como la cantidad de neuronas por capa y la cantidad de capas.

Es posible encontrar procesos de optimización que involucren ambos aspectos, pero existe la posibilidad de que los espacios de búsquedas no sean convergentes. En algunas ocasiones se utiliza un algoritmo de aprendizaje para ajustar los pesos de la red y el algoritmo evolutivo para optimizar la estructura de la red. Lo que evita problemas de interferencia entre los espacios de soluciones potenciales que impiden ajustar ambas características. Sin embargo, la mayoría de las investigaciones en el área se enfocan solo en la optimización de los pesos sinápticos que ponderan las conexiones entre neuronas, dejando la estructura de la red a elección del diseñador.

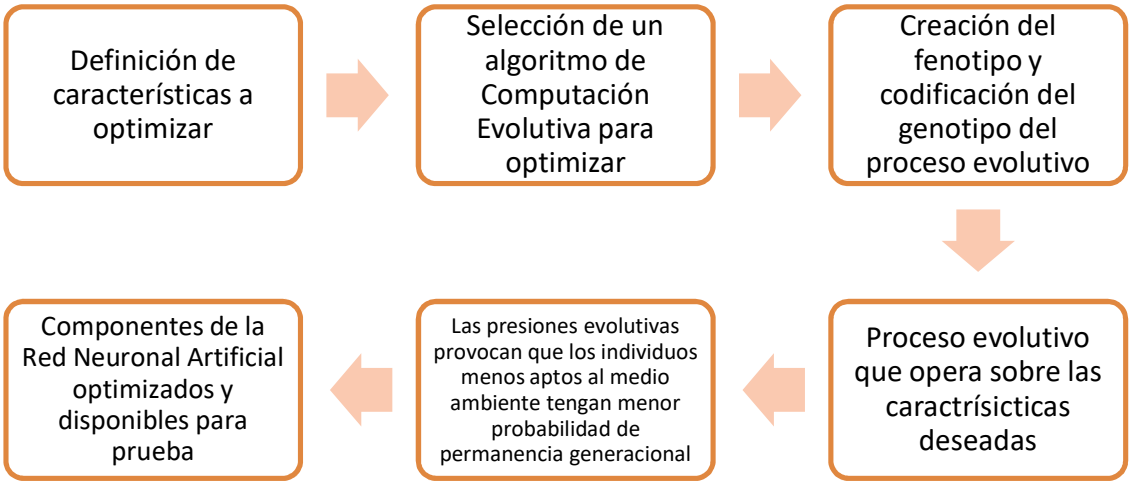


Figura 2. Proceso de evolución de redes neuronales artificiales en robótica evolutiva. Después de definir los componentes a evolucionar y los mecanismos de evolución, se deben codificar en su forma cromosómica. El proceso computacional evolutivo recibe las presiones del medio para fomentar que los individuos más aptos tengan mayores probabilidades de permanecer dentro del mismo. Finalmente se obtienen los componentes evolucionados y listos para prueba.

Existen muchas variantes en cuanto al tipo de conexiones que permiten las capas y el número de neuronas existentes en ellas. Una de las más empleadas es la *Feedforward* o con conexiones hacia adelante. Este tipo de arquitectura representa un control puramente reactivo (Parisi, Cecconi y Nolfi, 1990). También es posible implementar las arquitecturas *Backward*, modulares, cíclicas y acíclicas. En el caso de las conexiones cíclicas, en especial cuando se encuentran en capas ocultas de las RNAs, simbolizan controles deliberativos. Inclusive pueden representar bloques de memoria que le permiten recordar al robot sobre acciones pasadas (Yamauchi y Beer, 1994).

Concerniente a las redes modulares, que son el tema principal de este trabajo de investigación, se utilizan cuando se requiere dividir el control de las tareas en segmentos o cuando se requiere que una red atienda más de una tarea. A partir de la literatura presente en el área, se puede identificar dos tipos de modularidad: interna y externa. La modularidad interna se refiere a la división de una

sola RNA en varios fragmentos capaces de procesar por separado un determinado problema. Mientras que la modularidad externa se refiere al uso de diferentes RNAs que procesan diferentes tareas y que pueden comunicarse mediante algún medio. En la Sección 2.2 se aborda este tema con mayor detalle.

Con respecto al número de neuronas por capas y el número de capas de los controladores (ver Figura 3), en la capa de entrada se suelen ubicar las neuronas asociadas con los sensores del robot que pueden ser representados de manera directa o indirecta. Se realiza de manera directa cuando la neurona recibe como entrada la lectura del sensor asociado. Se realiza de manera indirecta cuando la lectura de un sensor es pre-procesada, cambiada de dominio o fusionada a la información de más sensores. Un ejemplo de esto último es una cámara de baja resolución cuya lectura es procesada en términos de la presencia o ausencia de sus componentes RGB (Red-Green-Blue por sus siglas en inglés) y no por el valor mismo en un determinado pixel. Otro ejemplo ocurre cuando para disminuir la carga de trabajo del algoritmo de computación evolutiva que optimiza los pesos de una red, se discretizan las lecturas provenientes de los sensores. Esto es un hecho práctico solo cuando no se requiere de la totalidad de la sensibilidad proporcionada por la lectura del sensor.

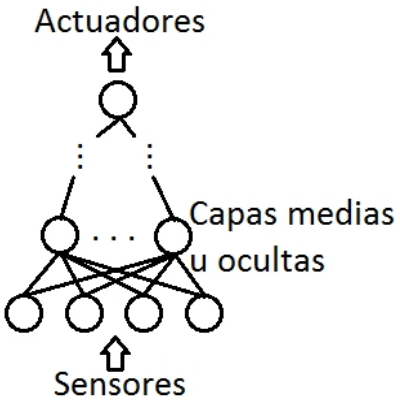


Figura 3. Prototipo de red neuronal artificial usada en robótica evolutiva. Se compone por una capa de entrada que recibe la información sensoria (medio ambiente e interna del robot). Puede o no incluir capas medias que aumentan el grado de deliberación del neuro-controlador. La capa de salida tiene efecto sobre los actuadores del robot, como pueden ser los motores del mismo para producir movimiento.

La presencia o no de capas ocultas en una RNA obedece al nivel de reactividad y procesamiento necesarios para que el sistema de control opere en los parámetros requeridos. En otras palabras, la función de las capas ocultas o intermedias en un sistema de control de un robot evolutivo es la de agregar capacidad de discriminación (capacidad para separar clases). De esta manera, si la tarea a realizar no necesita de mayor discriminación, no se deben agregar capas ocultas a la red ya que puede aumentar la dimensionalidad del espacio de soluciones. El número de neuronas en las capas ocultas está en función de la cantidad de neuronas de las capas previas. Por ejemplo, en una arquitectura con conexiones hacia delante de tres capas, la cantidad de neuronas idealmente debe ser menor al número de neuronas en la capa de salida, pero mayor al de la capa de entrada.

Por su parte, en la capa de salida se encuentran las neuronas que se conectan con los actuadores. De la misma manera que las neuronas de sensado, existen dos tipos de neuronas en la capa de salida que pueden ser identificables. El primer tipo es de codificación directa, y permiten enlazar el funcionamiento de los actuadores del robot directamente con la función de salida de la neurona. En el otro tipo de neurona, la función de salida de la neurona debe recibir un post-procesamiento para enlazarse con el actuador del motor. Por ello, se considera una neurona de codificación indirecta.

En cuanto a las funciones de activación de las neuronas que componen el sistema de control de un robot, la mayoría de ellas tienen un efecto marcado. Tal es el caso de la función escalón unitario que se utiliza para robots puramente reactivos. O la función sigmoidal que produce cambios menos inmediatos en la activación de la neurona. De esta forma, entre mayor sea la inmediatez del cambio producto de la función de activación, mayor será la capacidad reactiva del sistema de control.

Es común encontrar en la literatura el uso de algoritmos genéticos para optimizar RNAs usadas como elementos de control de robots. La razón es la facilidad para codificar los pesos sinápticos en el genotipo de cada individuo dentro del proceso evolutivo (Nolfi, Bongard, Husbands y Floreano, 2016). También la posibilidad de codificar más características como la cantidad de neuronas, capas y conexiones en la red.

2.1.3 Proceso evolutivo.

Todos los elementos computacionales y matemáticos involucrados en la optimización de las estructuras de control y/o morfología forman parte del proceso evolutivo artificial. De esa manera el proceso evolutivo comprende al algoritmo de computación evolutiva y todo lo referente a éste. Es importante remarcar que el proceso de evaluación que se menciona al inicio de este capítulo puede considerarse como parte del proceso evolutivo, ya que a partir de éste se guía la optimización.

La finalidad del proceso evolutivo artificial es recrear las condiciones, por lo menos de manera parcial, a las que han sido sometidos los seres vivos con el paso de los años en los entornos reales (Bongard, 2013). Así al finalizar esta búsqueda y de manera ideal se obtiene a los individuos capaces de contender con el medio de acuerdo a las especificaciones del proceso de evaluación. En el mundo real, los procesos evolutivos involucran una gran cantidad de años. Por ejemplo, el caso de los seres humanos es de aproximadamente 7 millones de años. En el caso de los robots evolutivos, el tiempo para obtener resultados es menor gracias a la velocidad de procesamiento de las computadoras o en su defecto de los microprocesadores.

Existen tres métodos de evolución artificial utilizados por los diversos grupos de investigación en el mundo (Vargas, Di Paolo, Harvey y Husbands, 2014): en entorno real, en entorno virtual y en entornos mixtos. Se dice que un proceso de evolutivo se desarrolla en un entorno real cuando el algoritmo de computación evolutiva corre en los robots reales. Este tipo de procesos resultan muy

costosos en términos de tiempo de prueba. Por ejemplo, en el caso de un algoritmo genético cuya duración máxima comprende 100 generaciones cada una de 100 individuos y el tiempo de prueba de cada individuo es de 30 segundos. El tiempo total procesamiento es de aproximadamente 83 horas continuas solo para la evaluación. En términos prácticos esto es algo complicado de llevar a cabo debido a la duración de pila y degaste de componentes electrónico-mecánicos.

Al realizar las pruebas solamente en entornos físicos no existen fenómenos que desajusten los elementos evolucionados. Aunque la cantidad y tiempo de pruebas debe representar un nivel de riqueza importante. Esto permite que los robots estén expuestos a una mayor cantidad de situaciones potenciales que deben resolver. Un buen ejemplo de este tipo de procesos es presentado por Floreano y Mondada (1996). El proceso evolutivo en entornos reales reduce el tiempo de prueba a un mínimo de 300mS por vida para cada individuo. No obstante, bajo este esquema también se reducen las situaciones a las que el robot se enfrenta. Existen variantes de la evolución en robots físicos. Karafotias, Haasdijk y Eiben (2011) describen un proceso de optimización basado en estrategias evolutivas, y que es distribuido. De esa manera, cada robot representa una solución particular al problema y el conjunto de la población representa la solución final.

El segundo método de evolución artificial en RE es el simulado. Tal vez se trata del más popular entre los investigadores ya que los tiempos de prueba se reducen. Además, solo es necesario contar con el modelo de simulación de un robot y no con elemento físico para llevarlo a cabo. Aunque no siempre es indispensable comprobar el funcionamiento final del resultado del proceso evolutivo en un entorno real, esto agrega una característica extra al modelo representado. La principal desventaja de la optimización simulada es la aparición de desajustes, producto de la disparidad entre modelos virtuales y físicos de los entornos. Este problema es común en la robótica de simulación y se conoce como *Reality Gap* (Jakobi, 1997). En la actualidad el aumento en los costos de los robots reales y la necesidad de salvaguardar sus partes han llevado a que las investigaciones utilicen un enfoque puramente virtual (Preen y Bull, 2017), (Corucci, Cheney, Kriegman, Bongard y Laschi, 2017), (Ozdemir, Gauci y GroB, 2017), (Woodford, Du Plessis y Pretorius, 2017). Sobre todo, cuando se tratan comportamientos elaborados o cuando los modelos evolucionados no tienen equivalencias directas en los entornos reales (Ficici, Watson y Pollack, 1999) y que pueden producir desajustes incorregibles en modelos reales.

Uno de los primeros ejemplos de evolución en entornos simulados es presentado por Nolfi, Floreano, Miglino y Mondada (1994). En él se utiliza un entorno de simulación mínimo, que contempla las respuestas de los sensores y actuadores en entornos reales. Con el paso de los años esta investigación dio pie a la aparición de dos simuladores especializados en la experimentación de RE: EvoRobot (Nolfi y Gigliotta, 2017) y FARSA (Massera, Ferrauto, Gigliotta y Nolfi, 2013).

El tercer método es una combinación entre los ambientes reales y virtuales. El proceso en un inicio funciona sobre un entorno virtual, lo que permite que el tiempo del proceso sea corto en esta primera fase. Sobre las generaciones finales o la parte final del proceso evolutivo el entorno de prueba se traslada a ambientes reales. Así se disminuyen los desajustes provenientes de los

modelos virtuales. Esta metodología es usada por Miglino, Lund y Nolfi (1995) bajo la idea que durante las primeras generaciones que se prueban en un entorno real los saltos del proceso evolutivo son grandes. Durante la etapa final estos saltos disminuyen y el proceso entrega los individuos aptos para su prueba en entornos reales.

Referente a los algoritmos de computación evolutiva que pueden ser utilizados para desarrollar elementos de morfología y control de robots, se permite el uso de cualquiera de ellos. Sin embargo, no todos los algoritmos existentes han sido probados en el campo de trabajo. Entre los algoritmos probados se encuentra el trabajo de Cruz (2012) y Cruz, Montes-González, Mezura y Santos (2012) en donde se utiliza la Evolución Diferencial en dos versiones diferentes, así como el algoritmo de Optimización de Partículas (PSO por sus siglas en ingles) para la evolución artificial de neuro-controladores que se encargan del manejo de tareas simples como seguimiento de paredes. Las Estrategias Evolutivas en sus dos versiones (*1 + 1* y *u + 1*) son otra opción para optimizar elementos de control o morfología de robots. Becerra, Santos y Duro (1999) usan este algoritmo para optimizar el comportamiento de seguir paredes. Reportan que son necesarias menos generaciones del algoritmo utilizado respecto a los AGs para tareas de exploración del entorno y elusión de obstáculos.

La Programación Evolutiva (PE) y Programación Genética (PG) constituye otra opción para desarrollar los procesos de evolución artificial. La PE, desarrollada por Fogel (1993) en 1964, es un programa de alto nivel generalmente asociado con una estructura de Autómata de Estados Finitos, y cuyo principal operador es la mutación. La PG, desarrollada por Koza (1994) en 1989, es también un programa de alto nivel representado por un árbol, pero que está basado en el operador de cruza. Es una variante de la PE. Un ejemplo de aplicación en RE de la PG es el trabajo de Nordin. Banzhaf y Brameier (1998), quienes evolucionan el control de un robot Khepera mediante la manipulación de código ensamblador.

Aunque algunos algoritmos de computación evolutiva han demostrado una mejor influencia en términos del tiempo de convergencia o de los óptimos encontrados. La opción preferida por los investigadores del área son los algoritmos genéticos. Esto se debe posiblemente a la combinación que representan con las RNAs. Y es que no es difícil imaginar que la codificación de los fenotipos de las RNAs a genotipos utilizados en los AG se realiza de manera sencilla y directa. Por ejemplo, la codificación de pesos se realiza mediante arreglos continuos de valores lógicos binarios; inclusive pueden codificarse características presentes y ausentes de la red mediante estos valores.

Los algoritmos genéticos fueron implementados por primera vez por Holland (1992) en el año de 1975. Su propósito era abstraer y explicar los procesos de adaptación presentes en los sistemas naturales, así como el diseño de sistemas artificiales que emulen los mecanismos de los sistemas naturales. Según Goldberg (2000) los AG son un tipo de algoritmo computacional basado en búsquedas inspiradas en mecanismos de selección natural y genética. En ellos los individuos más aptos sobreviven para intercambiar información y así promover la herencia de sus características.

Para Coello (2004) un AG, en su concepción básica, es una representación binaria que codifica las soluciones a un determinado problema, que es evolucionado mediante los operadores básicos de mutación y cruza. Un AG está constituido por las siguientes partes:

- Genotipo. La representación con la que trabaja el algoritmo de manera directa. A ella se le aplican los operadores. También se conoce como cromosoma.
- Fenotipo. Es la representación de un nivel alto de las características a optimizar.
- Función de calidad o aptitud. Es una función matemática que otorga un puntaje a cada uno de los individuos probados. Expresa las características deseables del individuo ideal.
- Operadores. Son las modificaciones que se le aplican al genotipo para variar la búsqueda. Los básicos son elitismo, sustitución, selección, mutación y cruza (ver Figura 4).

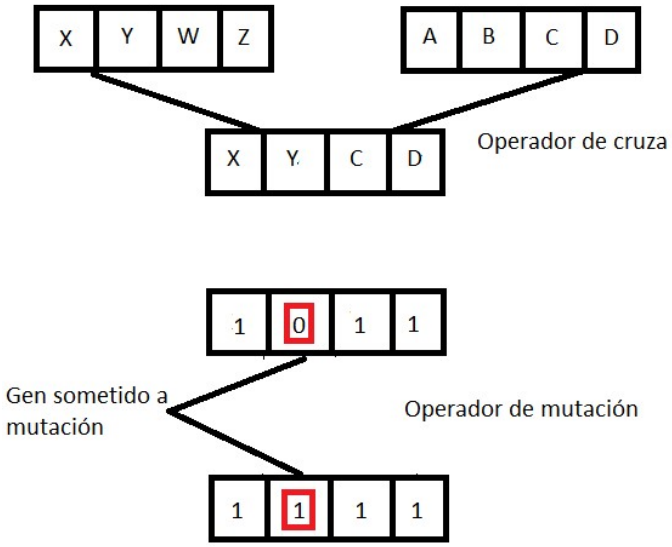


Figura 4. Operadores genéticos de cruza y mutación. La cruza consiste en la combinación de los cromosomas de dos o más individuos para generar un nuevo individuo. El operador de mutación en las representaciones binarias consiste en la operación del complemento binario de un gen seleccionado.

El proceso básico de funcionamiento de un AG en el campo de la robótica evolutiva es el siguiente (ver Figura 5):

1. Generar una población aleatoria inicial.
2. Calcular la aptitud de cada individuo.
3. Aplicación de operadores.
4. Generación de una nueva población a partir de los operadores aplicados en el paso previo.
5. Si se cumple con la condición de parada, terminar el proceso. En caso contrario la generación constituida en el cuarto sustituye a la anterior y se retoma el proceso en el segundo paso.

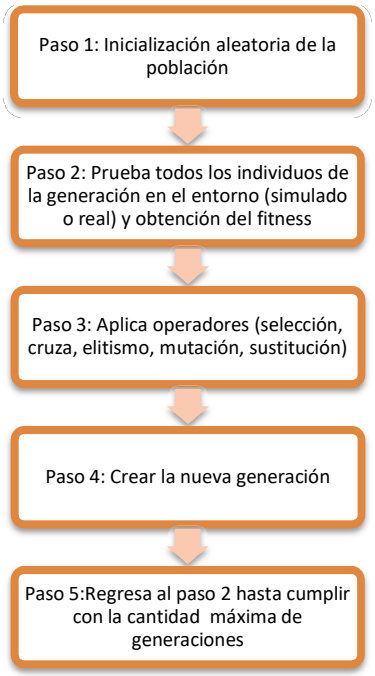


Figura 5. Prototipo de algoritmo genético usado en robótica evolutiva. El proceso comienza con la inicialización de los genotipos de los individuos. Posteriormente se prueban los individuos en el entorno correspondiente (simulado o real) para obtener el puntaje de aptitud o *fitness*. Con ello es posible aplicar los operadores y crear la nueva generación. El proceso regresa a evaluar a la nueva generación en tanto el número de generaciones totales no se alcance.

En cuanto a los operadores, el de elitismo funciona como medio para conservar a los mejores individuos de la población para que su genotipo se mantenga intacto en la conformación de la nueva generación. Se recomienda que este operador afecte a una cantidad baja de individuos, de manera que no influencie la diversidad que los demás operadores brindan. Esto representa, en términos de búsquedas computacionales, la posibilidad de seguir explorando el espacio de soluciones posibles sin perder un óptimo local encontrado previamente.

La mutación es un operador que, en su concepto básico, sustituye el valor lógico de un gen del cromosoma por su valor contrario. Opera sobre la totalidad de los genes de la nueva generación, ocasionando pequeños cambios en los cromosomas de los individuos. La cantidad de genes que se ven expuestos a este operador se expresa en forma de número y es conocido como porcentaje de mutación. Este operador representa, en términos de búsqueda computacional, saltos en el espacio de soluciones que tratan de impedir mantenerse en óptimos locales. Se recomienda que el porcentaje de mutación en los procesos evolutivos del área de RE se mantenga bajo para no producir saltos continuos en el espacio de búsqueda. En especial al final del proceso evolutivo en el que se espera un refinamiento del sistema de control.

Los operadores de selección, cruza y sustitución se encuentran fuertemente relacionados ya que a partir del primero se buscan los candidatos para aplicar los otros dos. Entonces el operador de selección sirve para obtener los candidatos para una etapa posterior. Puede funcionar, en su forma básica de dos maneras diferentes: torneo y ruleta. Cuando la selección se realiza a partir de

un torneo, se comparan los puntajes de aptitud de dos o más individuos para obtener vencedores. Los ganadores del torneo son los individuos que se cruzaran para obtener descendencia. Por otro lado, cuando se utiliza el método de la ruleta se asigna una probabilidad a cada individuo. Dicha probabilidad está basada en el puntaje de aptitud, en donde los individuos con mayor puntaje son los que mayores probabilidades tienen de reproducirse.

El operador de cruza funciona a partir del genotipo de los individuos seleccionados. Éste combina, a partir de puntos de cruza, los cromosomas de dos o más padres. Los nuevos cromosomas obtenidos a partir de este operador sustituyen a los individuos de la generación previa. En términos de búsquedas computacionales, este operador es el camino por el cual se obtiene la diversidad suficiente para encontrar óptimos. La sensibilidad de este movimiento en el espacio de búsqueda depende de la cantidad de padres que forman un nuevo individuo y de los puntos de cruza utilizados. El operador de sustitución permite conservar a un número de individuos de la generación previa. Esto provoca que los saltos en el espacio de selección no produzcan cambios repentinos y por tanto que se pierda algún óptimo local ya encontrado.

El operador de selección determina la cantidad de individuos hijos y padres que formarán la nueva generación. En términos de búsquedas computacionales representa una vía para agregar variabilidad controlable y conservar puntos aleatorios previamente explorados que pueden representar una ruta para encontrar óptimos locales. Es decir, brinda una nueva posibilidad a los puntos encontrados previamente a que durante la siguiente generación puedan generar una mejor recombinación o mutación.

Existen otros operadores que son utilizados en RE para refinar las búsquedas, como lo son los operadores de inversión, que cambian el sentido de los genes de los cromosomas. También existen variantes de los AG que les permiten trabajar con diferentes especies de individuos, como es el caso de la coevolución.

La coevolución permite evolucionar en un mismo proceso computacional diversos organismos o características diversas de un solo organismo. La coevolución también puede ser vista como una manera de reestructurar espacios de potenciales soluciones complejos en espacios, en donde se generan procesos simbióticos (Watson y Pollack, 2000). Ésta puede ser de dos tipos: Competitiva y cooperativa. Cuando se habla de coevolución competitiva, se deben considerar entonces organismos contrarios o contra-posicionados. Entonces se genera un mecanismo de competencia entre las funciones de calidad que le permiten al proceso obtener los resultados deseados. El caso contrario es la coevolución cooperativa. Los organismos son complementarios y buscan coadyuvar en la solución de un problema. Entonces los mecanismos generados por las funciones de calidad son de ayuda entre los individuos o características optimizadas. Un ejemplo del uso de la coevolución es presentado por Montes-González, Ochoa, Marín y Aguilar-Sánchez (2010), en donde se obtiene un modelo presa-depredador a partir de la versión competitiva de esta variante de los procesos evolutivos artificiales.

2.2 Sistemas bio-inspirados y robótica evolutiva.

La bio-inspiración en el campo de la robótica evolutiva es un tema que está adherido fuertemente desde el surgimiento del área en la década de los años 90s. Lo que se buscaba desde un inicio fue tomar un fenómeno natural como es la evolución y adaptación natural, y trasladarlo al campo de las ciencias computacionales. La presencia de las redes neuronales artificiales como herramienta de control es otro claro ejemplo de este espíritu de inspiración en fenómenos naturales. A partir de estas estructuras se buscan obtener sistemas nerviosos artificiales (Gurkiewicz y Kornegreen, 2007). Dichos sistemas son expuestos a presiones evolutivas con el fin de obtener el desarrollo de comportamientos o algunos componentes funcionales útiles bajo ciertas circunstancias para generar modelos computacionales (Webb, 2009). Esta rama de trabajo de la RE se combina con una rama conocida como neurobiología computacional. Esta combinación es una plataforma de pruebas de sistemas muy importante (Floreano, Husbands y Nolfi, 2008) con implicaciones tanto en las neurociencias como en la robótica. Por ello es que resulta particularmente importante el estudio de este tipo de sistemas, tal como lo mencionan Husbands *et al.* (2014).

La finalidad de esta combinación es crear cerebros artificiales para equiparlos en robots evolutivos. En ellos se realizan pruebas de diferentes fenómenos neurológicos a distintas escalas y complejidades (Mayer, 2011), (Pasemann, 2013), (Jeason y White, 2012), (Fernando, Szathmary y Husbands, 2012). Al crear modelos computacionales basados en modelos naturales neurológicos, se busca seguir un camino que la naturaleza ha tomado y perfeccionado durante muchos años mediante la evolución.

El modelo realizado durante esta investigación doctoral busca rescatar algunos componentes neurofisiológicos como son la sinapsis química, la producción y la acción de la acetilcolina como neurotransmisor y modulador de la activada de las moto-neuronas, también la acción inhibitoria colateral de las células de Rensahw, y aplicarlos a un modelo computacional de redes neuronales que funcionan como módulos. Dichos conceptos y sus características se explican en los párrafos siguientes.

2.2.1 Sinapsis química y física.

Las células del sistema nervioso central, también conocidas como neuronas, poseen diferentes propiedades y características que les permiten funcionar como una unión. Una de ellas es la capacidad para transmitir información a través de los impulsos eléctricos. Dicho proceso (Pereda, 2014) es conocido como sinapsis y se puede dividir en dos categorías que dependen de la naturaleza de los medios empleados para llevarlo a cabo: eléctrica y química. La primera es la más replicada dentro del campo de la inteligencia artificial dentro de los modelos de redes neuronales artificiales. Para que se lleve a cabo este tipo de sinapsis, debe existir contacto entre el axón de la neurona pre-sináptica y la dendrita de la neurona post-sináptica que permitan la transmisión de corrientes eléctricas y la despolarización de las neuronas, que ocurre de manera casi inmediata entre las neuronas pre-sináptica y post-sináptica (ver Figura 6).

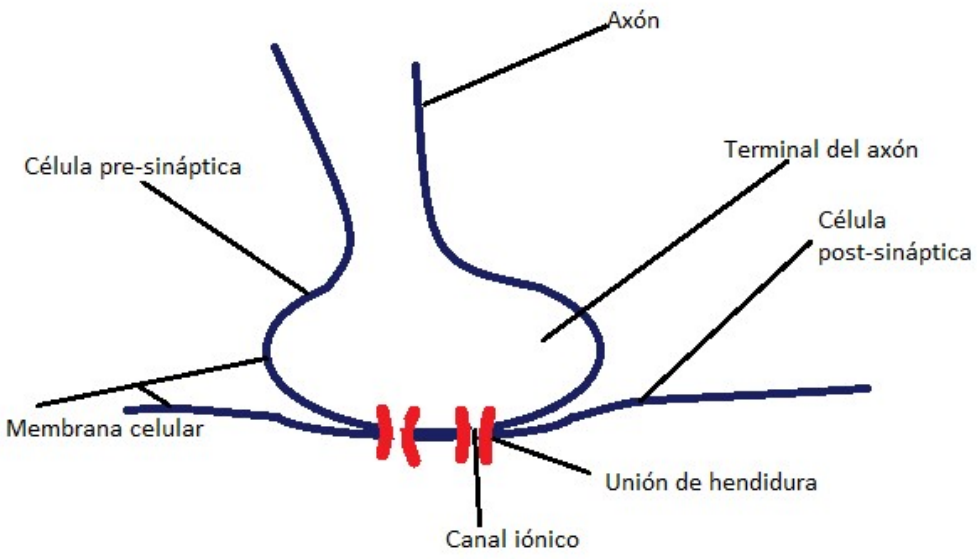


Figura 6. Esquema de la sinapsis eléctrica. Una neurona pre-sináptica envía una corriente eléctrica a una neurona post-sináptico. Lo cual provoca que la neurona receptora produzca un potencial de acción.

Por su parte, la sinapsis química (ver Figura 7) opera a través de sustancias que son liberadas por una neurona previa o pre-sináptica. Las sustancias son capturadas por la neurona que hará sinapsis y que se conoce como post-sináptica. Al realizar el proceso de recepción, las neuronas se despolarizan y producen una corriente eléctrica. Este procedimiento no es instantáneo como en el caso de la sinapsis eléctrica ya que existe un pequeño retardo producto del proceso de segregación y captura de las sustancias transmisores. Además, el efecto de los neurotransmisores puede amplificarse, algo que no ocurre dentro de la sinapsis eléctrica. Existen mecanismos que permiten recuperar las sustancias liberadas en la sinapsis y reintegradas al proceso. Esta forma de transmisión de información a nivel neuronal no es modelado de forma común en las redes neuronales artificiales.

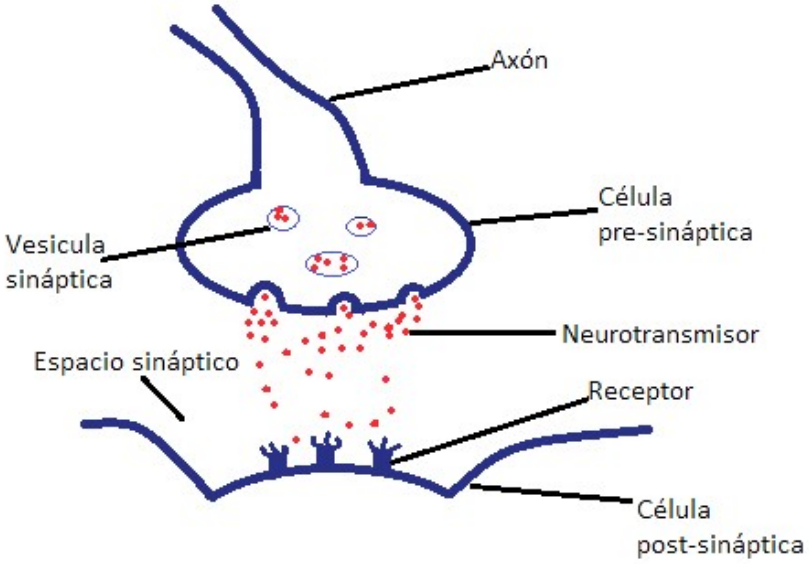


Figura 7. Esquema de la sinapsis química. Una neurona pre-sináptica envía sustancias neurotransmisoras que son capturadas por receptores en la neurona post-sináptica. Esto produce un potencial de acción en la neurona receptora.

Cada tipo de sinapsis representa ciertas ventajas y desventajas. Por ejemplo, el tiempo de transmisión en la eléctrica es menor que en la química. Por su parte la química tiene señales que pueden amplificarse a partir del incremento de los transmisores. La sinapsis química es la que se reproduce con mayor frecuencia en las células. La mayoría de las neuronas basan su funcionamiento en el intercambio de información por sustancias químicas y no por contacto directo. Las sustancias responsables de producir sinapsis química en un par de neuronas son los neurotransmisores (Katz, 1969) (Spitze, 2017). Son sustancias que permiten controlar diferentes funciones. Por ello es posible verlos como los mecanismos que regulan la interacción de los componentes o módulos del sistema nervioso.

2.2.2 Neurotransmisores.

La sinapsis química, así como otros muchos procesos neuronales, están regulados por sustancias producidas en el cerebro llamadas neurotransmisores. Por ello un neurotransmisor es una sustancia química que permite el desarrollo de la sinapsis química. En la naturaleza existen diferentes sustancias neurotransmisoras, entre las que destacan: acetilcolina, dopamina, serotonina, noradrenalina, endorfina, glutamato. Cada uno afecta a diferentes partes del sistema nervioso. Algunos de ellos son complementarios, precursores o antagonistas. Este tipo de sustancias pueden clasificarse en:

- Aminas.
- Aminoácidos.
- Purinas.
- Gases.
- Péptidos.
- Ésteres.

Los neurotransmisores se producen dentro del mismo sistema nervioso, muchos de ellos producidos en los ganglios basales (formaciones neuronales pertenecientes al cerebro primitivo). El camino que siguen desde sus centros de transmisión hasta sus centros de almacenamiento se conocen como vías. Una vez llegado el neurotransmisor a su destino, es almacenado en vesículas especiales hasta instantes previos de su liberación.

Estas sustancias tienen algunas características fundamentales que les otorgan su identidad dentro del sistema nervioso central y periférico. Los neurotransmisores deben cumplir con las siguientes características para que puedan considerarse sustancias de este tipo (Seiggelbaum *y* Hudspeth, 2000):

1.- La producción se realiza por neuronas.

2.- Está presente en la neurona pre-sináptica, y se libera para ejercer un efecto o acción definida en la neurona post-sináptica u órgano efector.

3.- Cuando se ministra desde el exterior y en concentraciones razonables, imita exactamente la acción del transmisor de liberación endógena (exceptuando la acetilcolina).

4.- Existe un mecanismo de eliminación en la hendidura sináptica.

Es posible producir neurotransmisores virtuales basados en los modelos de redes neuronales actuales, especialmente en robótica evolutiva. Para ello se pueden usar neuronas de salida que emitan niveles de diferentes sustancias virtuales. Dichas sustancias pueden regular a otras redes mediante sus diferentes niveles. Su generación puede ajustarse a determinadas condiciones sensoriales del robot.

Así, las sustancias virtuales producidas provienen de estructuras neuronales. Pueden llevar su efecto a otras redes o neuronas. La presencia de dicha sustancia virtual puede servir como un candado para impedir la transmisión de un efecto entre neuronas. Además, se podría suministrar desde el exterior aumentando los niveles de salida de las neuronas productoras. Pueden existir mecanismos inmediatos y paulatinos para su eliminación. Finalmente, la sustancia puede ser utilizada para comunicar estados de diferentes procesos o fomentar la aparición de ciertas salidas (Smith y Philippides, 2000).

Las sustancias neurotransmisoras tienen muchas propiedades que pueden ser incorporadas a la robótica evolutiva (Husbands *et al.*, 1998). Desde procesos que modifiquen las salidas de una red, pasando por el modelado de emociones y estados de ánimo, hasta sistemas complejos en donde existe una modulación entre procesos.

Un ejemplo del uso de los neurotransmisores como mecanismo de excitación e inhibición de células es presentado por Husbands, Smith, Jakobi y O'shea (1998). Ellos proponen un mecanismo de control para robots evolutivos basado en una red de Gas. Lo que buscan es regular la concentración de una sustancia virtual que asemeja el funcionamiento de los neurotransmisores en las neuronas. La finalidad es modular el funcionamiento de las diferentes neuronas mediante el concepto general de neurotransmisor. De esta manera, se cuenta con un mecanismo adicional a las entradas de los sensores para producir una respuesta de salida (Philippides, Husbands, Smith y O'shea, 2005).

2.2.3 Acetilcolina.

La acetilcolina (ACh) es el neurotransmisor encargado de regular las funciones de las neuronas motoras. Por ello es importante estudiarla dentro de los sistemas de control de robots evolutivos, así como tratar de imitar su funcionamiento. Este neurotransmisor es producto de la mezcla de varias sustancias como la Coilina, que se encuentra de manera natural en las células pre-sinápticas. También participa la Acetilcoenzima A, que proviene del proceso de glucólisis de las neuronas (ver Figura 8). Para producir moléculas de esta enzima, la neurona deja de transformar algunas moléculas en energía dentro del proceso de glucólisis. Por ello se considera que la acetilcolina es contraria al proceso energético natural de las neuronas.

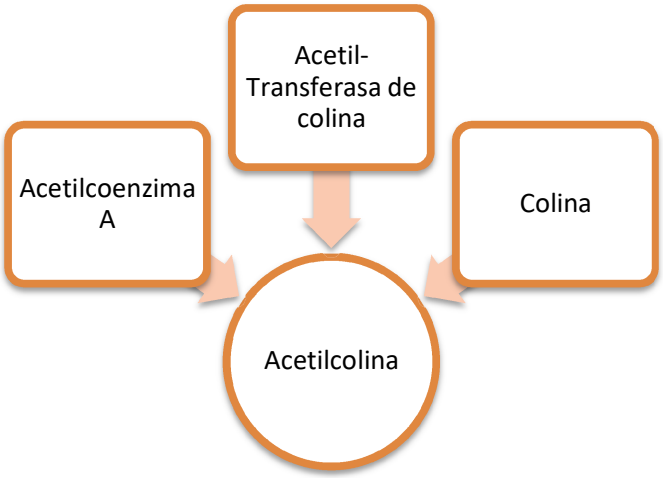


Figura 8. Proceso de formación de la acetilcolina (ACh). Las moléculas de Acetilcoenzima A se unen a las de la colina, utilizando como catalizador la colina acetiltransferasa. El neurotransmisor se guarda en las vesículas de las neuronas pre-sinápticas.

El catalizador es una sustancia conocida como Acetiltransferasa de Colina, que es la encargada de retirar de la enzima el grupo Acetil y permitir la unión con la colina. La producción de la acetilcolina se puede dar en neuronas de Renshaw, el Núcleo Pedunculo Pontino, el Núcleo Basal de Meynert, entre otros ganglios basales que formen parte de alguna vía colinérgica.

Una vez que la acetilcolina es formada, se libera desde la neurona presináptica y es capturada por receptores en la neurona post-sináptica. Dichos receptores pueden ser nicotínicos o muscarínicos. Los primeros se asocian con funciones excitatorias en el aparato músculo esquelético, y en algunos órganos de movimiento involuntario como el corazón. Por otro lado, los receptores muscarínicos se asocian a procesos inhibitorios y se relacionan con el sistema parasimpático. Mediante receptores nicotínicos tipo 1 (N1) regula la sinapsis en las uniones neuromusculares, es decir se encarga de controlar los músculos que deseamos mover. Con receptores N2, se regulan algunos ganglios del sistema autónomo, en especial del sistema parasimpático activados en procesos de relajación. Los receptores muscarínicos M1 se encuentran en neuronas relacionadas con la memoria, los M2 con el movimiento del corazón y el sistema parasimpático, los M3 con los intestinos, y los M4 con retroalimentaciones negativas para detener la producción de ciertos neurotransmisores.

Una vez capturado el neurotransmisor y ocurrida la sinapsis química, la neurona post-sináptica utiliza acetilcolinesterasa para quitar el grupo acetil del neurotransmisor y liberar las moléculas capturadas de colina (ver Figura 9). Lo mismo sucede con las moléculas del neurotransmisor que fueron liberadas, pero no capturadas por los receptores. Las neuronas pre-sinápticas recapturan la colina y la almacenan para seguir produciendo cantidades del neurotransmisor.

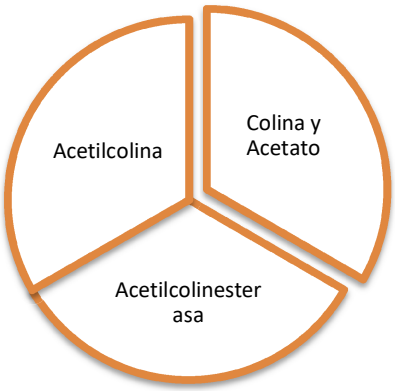


Figura 9. Proceso de separación de la acetilcolina (ACh). Las cantidades de neurotransmisor que no son captadas por los receptores post-sinápticos y las capturadas por los receptores son disueltas por la sustancia acetilcolinesterasa. Dicha mezcla produce dos sustancias: colina y acetato.

La acetilcolina está presente en todos los seres vivos, pero la única vía de producción en el cerebro humano es la síntesis dentro de las neuronas. A diferencia de otros neurotransmisores, la acetilcolina ingerida mediante la dieta no se rescata y almacena en regiones cerebrales. Lo que sucede con las cantidades ingeridas del neurotransmisor en la dieta es que se destruyen tan

pronto llegan al sistema digestivo. El autor de la destrucción de las moléculas de acetilcolina es una sustancia agonista llamada Butirilcolinesterasa que sirve de protección antes los efectos del neurotransmisor. Por la misma razón, la acetilcolina no puede ser inyectada como sustancia desde el exterior. De no existir este mecanismo, los músculos involucrados en el proceso de digestión podrían verse afectados, produciendo movimientos involuntarios.

Las deficiencias de acetilcolina se asocian con enfermedades como: Alzheimer, Parkinson, miastenia grave y parálisis. Una de las principales enfermedades asociadas es la adicción a la Nicotina. La nicotina es una sustancia agonista, que puede reemplazar la acción de la acetilcolina. Lo que produce es inhibición del apetito, en algunas ocasiones pequeños estallidos de energía, aumento de la memoria durante las primeras dosis del fármaco y pérdida de memoria de manera paulatina, produce movimientos indeseados en los músculos, cambiar el ritmo cardiaco y respiratorio, muchos efectos colaterales como los causados en el sistema circulatorio, y sobre todo una dependencia cada vez mayor de dicha sustancia debido al aumento de la necesidad de producción de neurotransmisores como la dopamina.

2.2.4 Neuronas de Renshaw.

Un papel muy importante del neurotransmisor acetilcolina se encuentra en las neuronas de Renshaw. Uno de los mecanismos con los que cuenta el sistema nervioso central para evitar que neuronas colaterales ejerzan acciones post-sinápticas es la inhibición. Este mecanismo consiste en mandar señales eléctricas o químicas de una neurona a otra para evitar conseguir un efecto contrario de la excitación. Las células más importantes en este sentido son las conocidas como células o neuronas de Renshaw, nombradas en honor al neuro-científico Birdsey Renshaw. Las neuronas de este tipo se encuentran en la médula espinal, particularmente en las astas anteriores de la médula espinal y se consideran inter-neuronas (Aznavurian y Aguilar Rebolledo, 2006).

Su función es mandar señales del tipo inhibitorias entre motoneuronas laterales. Esto sucede cuando una motoneurona es excitada y a través de sus conexiones laterales envía una señal a la neurona de Renshaw para que a su vez puedan mandar señales de inhibición a otras motoneuronas laterales (Alvarez y Fyffe, 2007). Este mecanismo de acción entre neuronas se conoce como inhibición lateral y es utilizado en funciones de movimiento.

Dada la naturaleza de su funcionamiento, este tipo de neuronas se conocen como un mecanismo de retroalimentación negativa. Dentro del aparato musculo-esquelético su acción es muy importante porque permite contraer músculos dada la elongación de otros. Es decir, cuando una motoneurona recibe la señal de excitación produce una extensión en su músculo, pero también envía una señal a las inter-neuronas para que el músculo contrario sea inhibido y realice la acción contraria del músculo excitado.

Este tipo de neuronas tienen una relación especial con la Glicina, un neurotransmisor inhibidor principalmente en la médula espinal (Lawton, Perry, Yamaguchi y Zornik, 2017). Pero también tiene una relación con otros neurotransmisores, principalmente con la Acetilcolina. Los receptores

nicotínicos tipo II se encuentra en las motoneuronas y realizan el trabajo de excitación en las mismas. Por lo que la sinapsis provocada a través del transmisor sirve como señal de excitación a las inter-neuronas de Renshaw quienes se encargan de inhibir las neuronas colaterales y bloquear el efecto de la acetilcolina en ellas (Kravos y Malesic, 2017).

2.3 Modularidad en Robótica Evolutiva.

La modularidad, como ya se mencionó en el capítulo previo, es un fenómeno muy importante en los seres vivos. Cho y Shimohara (1997) consideran que existe evidencia suficiente en el campo de la neurofisiología para afirmar que el cerebro humano procesa información mediante módulos. Dichos módulos son divisiones en partes identificables y que poseen sus propias tareas o propósitos. Añaden que la modularidad cerebral es resultado del proceso evolutivo en donde subsistemas especializados emergen interactivamente. Al trasladar este concepto a la RE, un módulo es un bloque de construcción, que unido con otros puede generar un mecanismo complejo de interacción.

Para Ballard (1986) la aparición de módulos en el cerebro humano es una consecuencia de la limitante del número de neuronas. El cerebro humano tiene un aproximado de cien mil millones de neuronas con 7000 conexiones por cada neurona. Por su parte Geschwind y Galaburda (1985) afirman que la especialización de algunas estructuras cerebrales es una consecuencia de las interacciones entre módulos (Jacobs *et al.*, 2017). Dicha característica es esencial para el éxito de sistemas multifuncionales (Potter, Meeden y Schultz, 2001).

Otro concepto a destacar sobre la modularidad y la biología es la de Bolker (2000), quien considera que el fenómeno permite describir y explicar la organización de los organismos vivos. A partir de esta propiedad define a la modularidad como el conjunto de unidades interconectadas, que poseen un nivel físico y uno funcional. Siguiendo esa tendencia, Newman (2006) menciona que la importancia en el estudio de la modularidad como fenómeno natural se encuentra en el papel que juega está en la biología: muchos de los organismos son divididos en módulos, lo que vuelve fundamental el estudio de la caracterización y detección de estructuras modulares.

Según Hart, Kammeyer y Belew (1994), la modularización computacional puede ser tratada como una operación de descomposición que reconstruye el espacio de búsqueda, de tal manera que sea más sencillo de explorar con los procesos evolutivos artificiales de cada componente o módulo. Un enfoque biológico-computacional es presentado por Wagner (1996), quien describe que la aparición de la modularidad en los entes biológicos es responsabilidad de las fuerzas de selección y evolución natural. Agrega que la organización modular es una propiedad evolutiva que proviene de la duplicación de genes con efectos pleiotrópicos. En especial en los caracteres complejos que tienen participación en funciones similares, lo que provoca que evolucionen juntos como en la co-evolución.

Reisinger (2003) menciona tres puntos fundamentales de la modularidad vista a través de los sistemas evolutivos artificiales. La capacidad de evolución que determina la mejor representación genotípica y fenotípica de los datos, así como la forma del paisaje de calidad. La especialización, que es la base de la adaptación a diferentes funciones con pequeña o escaza interferencia respecto a otras funciones. Y la canalización que es la robustez del fenotipo respecto a las mutaciones. Wagner, Mezey y Calabretta (2001) también brinda una clasificación sobre los métodos para emerger modularidad a través de los procesos evolutivos artificiales: La selección directa resultado de una selección con ventaja, la evolución dinámica con un efecto directo de las presiones evolutivas, y la simbiogénesis producto de factores externos añadidos al proceso evolutivo.

Desde el punto de vista de las RNAs, Caeli, Guan y Wen (1999) definen a la modularidad como elementos que se agrupan en subredes en función de la tarea a realizar. Así generan una taxonomía de los modelos modulares utilizados. El primero es un modelo mixto Gaussiano asociativo (AGM por sus siglas en inglés), basado en jerarquías en donde cada módulo está supervisado y cuentan con un sistema de mediación entre módulos. Otro modelo es el mixto jerárquico de expertos (HME por sus siglas en inglés) que consta de diferentes salidas que representan expertos, y que son combinados mediante distribuciones de probabilidad. Las redes de redes son modelos multinivel y en algunas ocasiones jerárquicos que interconectan módulos similares en cuanto a forma, generalmente en forma de fractales. En ellos pueden existir o no jerarquías, dependiendo del tipo de conexiones utilizadas.

Dentro del campo de la RE, Khare, Yao, Sendhoff, Jin y Wersing (2005) mencionan que un módulo representa una parte de una solución a un problema general, en donde la cooperación con otros módulos resulta fundamental. Bajo ese enfoque, la función de calidad de los individuos debe tomar en cuenta el proceso de cooperación. Sugieren además una clasificación de modularidad basada en la descomposición de las partes del problema. La descomposición serial consiste en módulos que requieren de una secuencia de activación. La descomposición paralela busca la generación de módulos que puedan atender sub-tareas simultáneamente, pero de manera separada.

Para Floreano, Dürr y Mattiussi (2008) la modularidad aplicada a la RE es una implementación de RNAs arregladas de tal manera que algunos bloques se encargan del procesamiento sensorial, mientras que otros del valor del sistema, y otros del mecanismo de selección de acción. Dürr, Floreano y Mattussi (2010) creen que la modularidad permite desarrollar estructuras de control mediante procesos automáticos de descomposición de tareas para robots evolutivos. Para ello se utiliza un enfoque funcional y un proceso evolutivo artificial.

Uno de los trabajos pioneros en la modularidad basada en RNAs y aplicada al control de robots es el realizado por Gruau (1994), quien considera que la modularidad puede equipararse con procedimientos jerárquicos computacionales, que al ser trasladados al dominio de las RNAs, en donde cada una corresponde a una porción de la red. Además, conceptualiza el fenómeno como la capacidad de realizar diferentes copias de una sección de una RNA. Utiliza para ello un proceso

evolutivo artificial, basado en un algoritmo genético paralelo, que define la morfología y los pesos de la red neuronal modular vista como una codificación celular. Los resultados de su investigación indican que el algoritmo de computación evolutiva genera regularidades en la topología de la red; se trata de un fenómeno que también se encuentra en el sistema nervioso de los seres vivos y que es resultado de las presiones evolutivas naturales.

Un trabajo derivado de la investigación previamente mencionada es presentado por NourAshrafoddin, Vahdat y Ebadzadeh (2007), quienes optimizan la estructura y los pesos de redes neuronales artificiales a partir de programas genéticos. Destacan que las estructuras con mejor rendimiento son aquéllas en las que su estructura resulta sencilla. Mientras que las estructuras complejas tienen un rendimiento inferior. Sin embargo, su aplicación sólo abarca la clasificación de basas de datos, no se aplica al control de robots. El resultado anterior es confirmado por Cho (1997), quien evoluciona módulos internos que poseen neuronas con conexiones inhibitorias y excitatorias utilizando pesos positivos y negativos, lo que genera competencia modular entre neuronas artificiales. Se indica que durante las primeras etapas del proceso evolutivo aparecen estructuras complejas que van desapareciendo conforme el proceso se hace maduro.

Otro de los trabajos fundamentales en el área, y que sirve como punto de comparación para esta investigación, es el que contempla la emergencia de arquitecturas modulares y que es reportado por Nolfi (1997). El enfoque de dicha investigación está destinado totalmente a la robótica, en donde la modularidad se utiliza como pequeños componentes que regulan los comportamientos del sistema de control. Sugiere utilizar el punto de vista proximal en la descripción de los comportamientos, es decir, utilizando el bloque sensorio-motriz del robot. Así se privilegia la experiencia del robot en su entorno y no la del programador. De esta manera el enfoque distal, el que tiene que ver con la óptica del programador, debe quedar de lado. De esa manera se cumple con el objetivo principal de la robótica evolutiva, que es generar una metodología para sintetizar sistemas de control que produzcan comportamientos más elaborados. También se busca que el robot pueda seleccionar por sí mismo aquellas situaciones que son favorables para la estructura sensorio-motriz, evitando esas situaciones que no le favorecen.

En la experimentación realizada por Nolfi, el robot Khepera simulado se encuentra inmerso en un entorno delimitado por cuatro paredes y en su interior se encuentran cuatro cilindros. Su objetivo es limpiar el entorno, sacando los cilindros del área. El estudio compara cinco RNAs con diferente arquitectura: RNA de dos capas, RNA de tres capas, RNA de dos capas y retro-conexiones entre algunas neuronas de la capa de salida a algunas neuronas de la capa de entrada, RNA de dos capas con dos módulos externos, y RNA de dos capas con módulos internos (ver Figura 10).

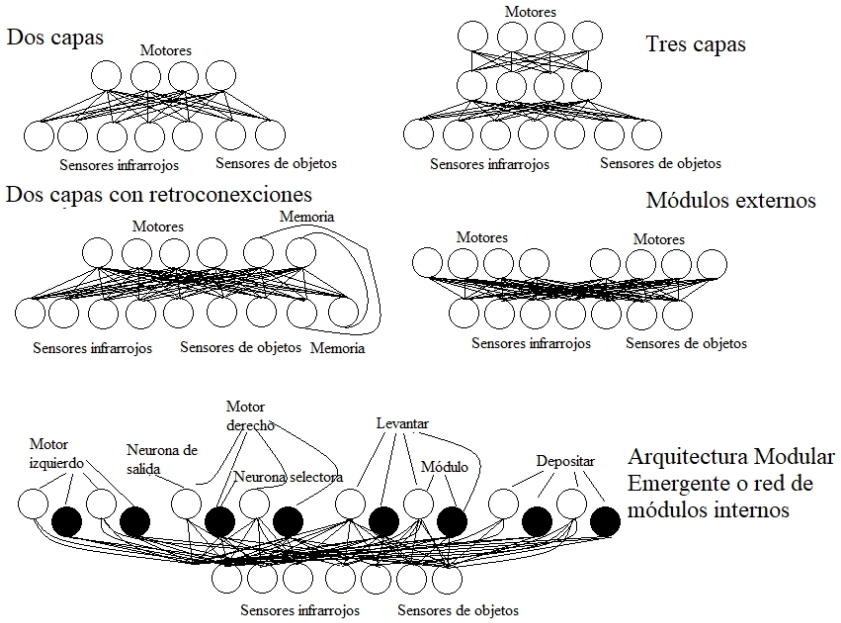


Figura 10. Estudio comparativo de diversas redes neuronales artificiales incluidas redes modulares (Nolfi, 1997). Se muestra una red tradicional (A) con una capa de entrada y una de salida, una red con capa media (B), una red con retro-conexiones (C), una red de módulos externos (D), y una red de módulos internos o red modular emergente (E).

Cada uno de estos neuro-controladores es sometido a un proceso de optimización o evolución artificial mediante un AG que optimizan los pesos de las redes. En el caso de la cuarta arquitectura, está conformada por dos redes neuronales. Cada una actúa de manera independiente y la entrada en funcionamiento de cada una depende del estado del robot. Si el robot ha capturado una pieza se activa la red A. Pero si el robot no ha levantado una pieza, el control le corresponde al módulo B. De esta forma, no existe un mecanismo de interacción que permite mejorar el dinamismo del control de módulos externos.

La quinta arquitectura está compuesta por un par de módulos por cada actuador o comportamiento relacionado con la capa de salida. Cada módulo contiene una neurona que codifica el comportamiento o actuador, y una neurona selectora. La función de esta neurona es otorgar el control del actuador o comportamiento al módulo que representan. Así se otorga el control al módulo con la mayor activación en dicha neurona. Esto genera una presión evolutiva similar a la de un proceso de evolución competitiva en donde, los módulos hacen prevalecer su solución ante los demás.

Los resultados muestran que el proceso evolutivo de la quinta arquitectura es el que tiene mejores resultados en términos de *fitness*, seguido de las arquitecturas segunda, cuarta, tercera y primera. En términos de pruebas en entornos reales, la *Arquitectura Modular Emergente* (*AME*) propuesta por Nolfi supera el rendimiento de las demás, seguida por las arquitecturas segunda, tercera, cuarta y primera. Pero en este caso, el bloquear los cambios en el módulo duplicado genera que el módulo que se sigue optimizando monopolice los recursos del robot para resolver una tarea y no

permita la participación del módulo competidor. Como consecuencia sólo uno de los dos módulos toma el control de un actuador en específico en la mayoría de las veces durante el tiempo de vida del robot (monopolización de los recursos del robot).

Entre las conclusiones se destaca que en el caso de la *AME* solo uno de los dos módulos que controlan el actuador del motor izquierdo se activa. Esto es producto de la emergencia y especialización del sistema de control. Al utilizar un mecanismo de presión competitiva que permite que dos módulos luchen por el acceso a los actuadores del robot, se genera una presión evolutiva que desencadena en un monopolio en el uso de los recursos. En cuanto a la arquitectura de módulos externos, el pobre desempeño puede ser causado por la falta de un sistema de interacción más flexible.

En un esfuerzo por parte de Ziemke, Carlsonn y Bodén (1999) por expandir el trabajo de Nolfi, se prueban tres arquitecturas neuronales artificiales adicionales derivadas de la AME. La primera incluye unidades de memoria, la segunda una capa intermedia, y la tercera unidades de memoria combinadas con una capa intermedia. Adicionalmente se utilizan las cinco arquitecturas definidas en el trabajo de Nolfi para comparar los resultados con las nuevas propuestas de arquitectura de control.

Los resultados confirman lo mostrado por Nolfi, en donde el mejor resultado se obtiene mediante la AME. Sin embargo, se muestra una mejoría en los resultados mostrados por parte de la arquitectura de módulos externos. En cuanto a las tres nuevas arquitecturas, únicamente la que cuenta con unidades de memoria tuvo un rendimiento intermedio entre las arquitecturas modulares y la reactiva simple. En el caso de las dos arquitecturas restantes el rendimiento es inferior al peor de los rendimientos mostrados en el trabajo de Nolfi. Como es concluido por Ziemke (2000), las redes neuronales recurrentes también pueden realizar funciones modulares en forma diacrónica. Es decir, como un mapeo del aparato sensorio-motriz del robot que se adapta de manera dinámica al ambiente en el que se encuentra inmerso.

En términos de los neuro-controladores utilizados en el área, es posible distinguir dos posibles versiones de modularidad: Interna y externa. La interna se refiere a la agrupación de neuronas dentro de una red con un propósito específico como la *arquitectura modular emergente*. La externa tiene que ver con redes con funciones específicas y que pueden interaccionan entre sí. La primera versión es generada para forzar especialización en los módulos, pero no permite interacción entre ellos de manera directa ya que sólo uno de ellos toma parte en la solución de todas las tareas. La segunda sí tiene la capacidad de producir interacción entre módulos que permite que todos puedan participar en la solución general del problema.

Una línea de investigación subsecuente al trabajo de Nolfi y paralela a los trabajos de Ziemke, se enfoca al aspecto evolutivo del control modular interno de robot mediante RNAs. Calabretta *et al.* (1998) utilizan la *AME* para el control del robot Khepera y la tarea de limpieza del entorno previamente descrita. Ahí se señala que una red neuronal modular es aquella que puede usar módulos en diferentes situaciones, con la ventaja respecto a arquitecturas no modulares de

generar diferentes respuestas a patrones sensoriales semejantes. El propósito fundamental de la investigación desarrollada por Calabretta es entender los mecanismos evolutivos que generan estructuras de control modulares especializadas (Calabretta *et al.*, 1998a). También lo es el comprender los mecanismos que propician la evolución de los módulos (Pereira-Leal y Teichmann, 2005). Se considera entonces que el cerebro humano es una colección de componentes que interactúan con una especialización estructural y funcional. La descripción de estos módulos no solo se encuentra en el nivel anatómico, sino también en el genético. Así los sistemas nerviosos que controlan a los organismos virtuales deben estar capacitados para resolver múltiples tareas.

El resultado de la investigación de Calabretta, Nolfi, Parisi y Wagner (1998a) es una extensión de la AME, y recibe el nombre de Arquitectura Modular Basada en Duplicaciones (AMBD). A diferencia de la AME, los módulos emergen del proceso evolutivo artificial mediante el uso del operador de *Duplicación*. Dicho operador consiste en copiar y duplicar los pesos de los módulos que compiten por un actuador en el punto medio de la evolución, para que el proceso evolutivo pueda optimizar únicamente los pesos del módulo duplicado y no del original. El operador de duplicación es el encargado de especializar los módulos internos de la red, pero su aplicación en exceso puede afectar a la emergencia de módulos funcionales y la monopolización de los recursos (Schlessinger, Bentley y Beau Lotto, 2006).

Entre los resultados se reporta que no existe correspondencia entre los módulos y las conductas resultantes. Esto corresponde a lo mostrado previamente en la *arquitectura modular emergente*, y que es consecuencia del punto de vista utilizado para la descomposición de los comportamientos. Se añade que la comparación del proceso evolutivo entre la *AME* y la AMBD no arroja diferencias notorias. Sin embargo, en términos de monopolización de módulos la AMBD parece provocar la aparición de este fenómeno con mayor frecuencia.

Una versión diferente para modularización interna es reportada por Reisinger, Stanley y Miikkulainen (2004), en cuyo caso se utiliza a la coevolución para la descomposición automática de módulos internos de redes neuronales con sus respectivas combinaciones. El algoritmo NEAT, según expresan sus creadores, permite modularizar y ajustar los pesos que componen una red neuronal mediante la reestructuración del espacio de soluciones potenciales. Para ello se utiliza un historial de mutaciones que evitan la interferencia genética. Tiene una velocidad de convergencia mayor a la presentada por otros algoritmos (Reeder, Miguez, Sparks, Georgiopoulos y Anagnostopoulos, 2008).

Por su parte, Di Fernando, Calabretta y Parisi (2001) justifican el uso de la modularidad en robots evolutivos. Consideran que en un sistema de neurocontrol que debe lidiar con una tarea sencilla no necesita utilizar módulos. Por otro lado, si se cuenta con varias tareas o con una tarea compleja entonces el uso de modularidad cobra sentido. La modularidad es una forma de resolver el problema de la interferencia neuronal, que es una variante del problema de bloqueo en espacios de soluciones. Este fenómeno ocurre cuando en una RNA que debe resolver dos tareas, el algoritmo de aprendizaje de los pesos es incapaz de encontrar una buena configuración debido a que existe un bloqueo muto entre ellas. Este fenómeno en términos de búsquedas

computacionales representa un espacio de soluciones complejo o difícil de explorar, que resulta inseparable. Esto puede resolverse reduciendo la dimensionalidad del mismo. La redimensionalización del espacio de búsqueda ocurre cuando la tarea se divide en módulos. La interferencia neuronal fue reportada por primera vez en los trabajos de Jacobs y Jordan (1992) en donde se intenta resolver una tarea bio-inspirada en procesos cerebrales visuales que es conocida como "*What and where task*", y consiste en la identificación visual de objetos.

Calabretta, Di Fernando, Parisi y Keil (2008) muestran que la interferencia neuronal entre dos tareas se da durante las primeras épocas del proceso de aprendizaje o de evolución, debido a que en ellas los saltos dentro del espacio de soluciones son mayores. Algunos de estos saltos pueden favorecer a una tarea, pero desfavorecer a la otra. Christensen y Dorigo (2006) presentan una solución basada en la evolución incremental combinada con la modularidad. También indican que para su aplicación es necesario un esfuerzo ingenieril para organizar el sistema en sub-tareas de complejidad ascendente. Éste aspecto puede contraponerse al enfoque de la robótica evolutiva o reducir la capacidad de emergencia de conductas.

Una solución alterna a la evolución incremental es la co-evolución de módulos. En ese caso el resultado es producto de las interacciones evolutivas entre módulos, sin tomar en cuenta su complejidad. La co-evolución puede ser de naturaleza competitiva y cooperativa. En este caso, y como es sugerido por Marshall, Blank y Meeden (2004), se deben balancear las presiones evolutivas dentro del proceso.

Pero el fenómeno de la interferencia en los espacios de solución también es inherente a los procesos evolutivos. Calabretta (2007) reporta la existencia de un fenómeno de interferencia genética. Ésta reduce la eficiencia en el proceso de selección durante la evolución y afecta la emergencia de los módulos. La interferencia genética es una extensión de la interferencia entre espacios de soluciones complejos. En ella, se reporta, no se pueden optimizar estructuras de una RNA durante el mismo proceso evolutivo que los pesos de la misma red. El fenómeno puede combatirse mediante el uso de algoritmos de aprendizaje combinados con los procesos evolutivos artificiales. De tal manera que el algoritmo de computación evolutiva se centre en la optimización de la estructura, mientras que el aprendizaje en el ajuste de los pesos (Calabretta, Di Fernando, Wagner y Parisi, 2003).

Una solución adicional al problema de la interferencia genética es el algoritmo ESP diseñado por Bryant y Miikkulainen (2003). Su principal característica es la división en poblaciones de un mismo proceso evolutivo de individuos. Se optimiza por una parte los pesos de una RNA y por otro su morfología. Este algoritmo tiene la capacidad de generar un tipo especial de modularidad interna, conocida como táctica. Dicha modularidad consiste en la creación de sub-módulos requeridos para implementar un sub-comportamiento. Este enfoque es, como lo señalan Téllez y Angulo (2006), contrario a la modularidad estratégica que busca los comportamientos requeridos para obtener un comportamiento global.

Una metodología adicional de creación de módulos es presentada por Thangavelautham y D'Eleuterio (2004). Se trata de la Red Emergente de Descomposición de tareas (ETDN por sus siglas en inglés). Ésta consiste en un conjunto de neuronas de decisión cuya función es la de regular la competición por dominancia de una red de módulos internos. Por su parte Praczyk (2013) reporta la evolución artificial de estructuras complejas de RNAs que utilizan modularidad interna y regularidad. El método utilizado (Assambler Encoding) se enfoca en la creación de la red, dejando de lado el ajuste de los pesos (Praczy, 2008). Se indica que la cantidad de módulos producidos bajo el método varía en cuanto a tipo y tamaño. Por lo que van desde pequeños módulos con pocas unidades o neuronas hasta varios módulos con una estructura compleja. Basado en el concepto de pre-adaptación de Darwin, que indica una estructura no se utiliza para el mismo propósito durante toda la evolución (Darwin, 2009). Así las estructuras se originan mediante la adaptación para la solución de diferentes problemas. Mouret y Doncieux (2009) proponen una codificación que combinada con un algoritmo de optimización multi-objetivo (MOEA) para la evolución de estructuras modulares internas.

En cuanto a la modularidad externa, Sharkey (2012) realiza una clasificación que tiene una elevada importancia cuando se trata con la interacción de diferentes redes. Señala que existen cuatro tipos de sistemas: Competitivo cuando los módulos buscan tomar el control de los recursos del robot, cooperativo cuando todos los módulos contribuyen con el funcionamiento requerido, secuencial cuando la salida de un módulo es la entrada de otro, y supervisado cuando existe un módulo que verifica el funcionamiento de otro. Un primer trabajo sobre módulos externos y secuenciales de redes neuronales artificiales, pero en este caso aplicadas al *clustering*, es presentado por Cao, Ahmadu y Shridhar (1997). Ahí se describe un sistema compuesto por un módulo de extracción de características principales que sirve como entrada a un módulo de *clustering*.

Tal vez uno de los trabajos más importantes sobre modularidad externa es la investigación descrita por Manoonpong, Pasemann y Fischer (2005). En él se describe un sistema de control para un robot caminante de cuatro patas, que está compuesto por diferentes módulos externos de redes neuronales de distintas estructuras y con interacciones entre ellos de diversos tipos. El sistema contempla tres módulos diferentes. El primero es llamado "Procesamiento de entrada de sensores" que está basado en una red mínima recurrente (MRC por sus siglas en inglés), recibe y procesa señales de dos sensores infrarrojos apostados al frente del robot, y cuyas salidas se conectan al módulo de velocidad del robot. El segundo es el módulo de movimientos rítmicos basado en una red neuronal osciladora (Kimura, Akiyama y Sakurama, 1999). Es utilizado para mantener una locomoción rítmica. El tercer módulo es conocido como regulador de velocidad y se utiliza para regular la velocidad y dirección de movimiento, incluyendo los giros. La interacción entre el primer y segundo módulo se realiza mediante secuenciación del tercero sobre el primero. Es decir, las salidas el primer módulo son las entradas del tercero. Lo mismo sucede con la interacción del segundo y tercer módulo. No se reportan los mecanismos de optimización individual y conjunta de cada módulo, pero puede presuponerse el uso de técnicas incrementarles para llevarlo a cabo. Lo que sí es señalado es el hecho que el sistema de control desarrollado es robusto a los comportamientos de eludir obstáculos y explorar el medio (Manoonpong, Paseman y

Roth, 2007). Así como que puede ser extendido para el control de robots con múltiples patas (Manoonpong, Paseman y Wörgötter, 2008).

Siguiendo con la descripción de sistemas de control modulares externos, De Nardi, Togelius, Holland y Lucas (2006) reportan la construcción de uno para el control de un helicóptero autónomo. En este sistema se emplean técnicas de evolución incremental para ajustar los pesos de módulos externos en forma de redes neuronales de perceptrón multicapa conectadas de manera secuencial. Se dice que una menor cantidad de conexiones entre módulos reduce la dimensionalidad del espacio de búsqueda (Rice, 2000). Este fenómeno es descrito por Cliff *et al.* (1992) como un procedimiento a priori, pero con beneficios evidentes.

Una de las principales ventajas del trabajo con módulos neuronales externos es que permiten optimizar todos los pesos de manera independiente, excepto aquellos involucrados en la interacción con otros módulos. Esto puede desembocar en módulos reutilizables. Una evidencia de que esto es posible la reportan Lara, Hülse y Paseman (2001), quienes evolucionan de manera independiente las conexiones sinápticas de un par de módulos asociados con tareas que involucran el uso de los mismos sensores. Lo más relevante de este trabajo es el hecho que los módulos compiten por el uso de algunos actuadores de manera eficiente y no monopolizan los recursos como en las arquitecturas de módulos internos. Al terminar el proceso evolutivo independiente se genera un nuevo proceso evolutivo para ajustar los pesos de unas neuronas de interfaz. Estas neuronas permiten a los módulos entrar en competencia por los recursos del robot.

Cuando se trabaja con modularidad, la descomposición de las tareas puede ocurrir en dos formas: automática incluida en el proceso evolutivo artificial, y por parte del diseñador mediante el conocimiento previo del problema. El primer camino puede llevar a problemas como el de la interferencia genética, o la falta de recursos computacionales para crear procesos de modularización automática combinados con procesos de ajuste de pesos o estructura del módulo. Es decir, representa un aumento en la dimensionalidad del espacio de búsqueda. Un ejemplo de la descomposición automática de tareas en módulos internos es la presentada por Miglino, Nolfi y Parisi (1996). En dicho trabajo se optimizan los pesos y las conexiones de los módulos internos mediante un enfoque incremental y discontinuo.

El segundo método representa una tendencia contraria a la presente en los principios de la robótica evolutiva: el diseñador debe reducir al mínimo su influencia e interpretación del entorno que transmite al robot. En un intento por reducir este efecto, Weitzenfield (2000) propone un proceso de modularización o partición de tareas mediante esquemas etológicos, que explican el funcionamiento del robot a nivel de comportamientos biológicos. Como se explicó previamente el punto de vista distal puede resultar complejo comparado con el proximal. Así la traducción de los comportamientos etológicos a su equivalente proximal resulta complicada en muchos de los casos.

Según lo expresan Meeden y Kumar (1998), el papel del diseñador en RE debe reducirse a determinar aspectos operativos de las redes neuronales, representaciones de entradas y salidas, características físicas del robot y del entorno. En el resto de los aspectos la influencia del diseñador debe ser indirecta o nula. A partir de ello es necesario buscar mecanismos semiautomáticos que permitan descomponer las tareas mediante un procedimiento que no requiera grandes cantidades de recursos computacionales. Pero que eviten también influir en la experiencia del robot de manera directa. Éste es el caso del proceso de modularización mediante los términos de la función de calidad, utilizado en esta tesis doctoral (Aldana-Franco, 2017). La base de esta metodología se encuentra en los trabajos en el ramo de la economía y los sistemas evolutivos de Frenken, Marengo y Valente (1999) y Frenken (2006). En ellos se demuestra que un sistema evolutivo puede ser descompuesto en términos matemáticos de una función de calidad sin afectar el proceso evolutivo.

2.4 Comunicación en Robótica Evolutiva.

Los sistemas de comunicación en los seres vivos son muy importantes. Les permite intercambiar información de todo tipo: medio ambiente, estados personales, información colectiva. Los canales de comunicación en la naturaleza son muy variados y van desde señales químicas o eléctricas, sonoras, visuales, olores (Scott-Philips, Blythe, Gardner y West, 2012).

En el caso de los robots evolutivos, los sistemas de comunicación tienen un valor importante para la solución de las tareas (Aldana-Franco, 2011), (Palacios-Leyva, Aldana-Franco, Lara-Guzmán y Montes-González, 2017). Es conocido que los sistemas de comunicación evolutivos deben tener asociado un alto valor adaptativo para las especies que se desarrollan en el mismo (Nolfi, 2013), (Steyven, Hart y Paechte, 2003). En caso contrario, la emergencia o el establecimiento de un sistema de señales es poco probable.

Los canales que sirven para establecer comunicación en los robots evolutivos pueden ser muy variados. Pueden utilizar un sistema de comunicación inalámbrico, uno sonoro, o alguno basado en señales luminosas o movimientos. La manera en que se establece un mecanismo de comunicación depende de los sensores y actuadores disponibles. Otro de los factores importantes a tomar en cuenta a la hora de pretender establecer un mecanismo de comunicación evolutivo es la complejidad del entorno y de la tarea a resolver (Aldana-Franco, Montes y Nolfi, 2017).

Resulta casi imposible imaginar un sistema evolutivo de comunicación que pueda emerger en tareas donde un sólo robot es capaz de resolverla (Loula, Gudwin y Queiroz, 2010). Los sistemas de comunicación requieren de una población de individuos. Si el sistema de comunicación no aporta nada a la población de robots, entonces existen pocas posibilidades que emerja y nulas de que se establezca (Montes y Aldana-Franco, 2011).

Al estar vinculado la presencia de los sistemas de comunicación con tareas complejas, que son resueltas por individuos en una población (Mitri, Floreano y Keller, 2009), (Mitri, Floreano y Keller, 2010). Esto puede ser un indicio de una relación probable con el fenómeno de la modularidad. Como menciona Steels (2003), es importante comprender y estudiar todos aquellos factores que pueden ayudar o estar involucrados en la emergencia de sistemas de comunicación y como estos ayudan a desarrollar nuevas soluciones tanto cooperativas como competitivas.

La comunicación, combinada con sistemas de control modular, puede ayudar a los robots a resolver de mejor manera las tareas difíciles de resolver con sistemas de control tradicionales. Esto debido a que los robots tienen mayores herramientas a su alcance para conseguir su propósito. Aunque inicialmente un sistema de comunicación aumenta la complejidad de la búsqueda, el uso de la modularidad puede reducirla, formando una relación que favorece ambas características.

2.5 Trabajo relacionado.

El enfoque que se utiliza dentro de esta investigación es el de dar solución a un problema computacional, específicamente a un problema de una herramienta muy utilizada en la Inteligencia Artificial. Esto se logra mediante el modelado de algunas características de los sistemas nerviosos de los seres vivos. En otras palabras, se trata de una solución bio-inspirada y transdisciplinar. En comparación con algunos trabajos previos (ver Tabla 1), el principal aporte es la inclusión de los conceptos de sinapsis química, la acción de un neurotransmisor artificial que tiene un efecto similar al de la acetilcolina, y la acción de inhibición de las neuronas de Renshaw.

Tabla 1. Cuadro comparativo de los trabajos más importantes de modularidad en robótica evolutiva.

Herramienta	Autores	Aporte	Año
Arquitectura Modular Emergente	Nolfi (1997)	Creación de una arquitectura modular aplicada a robots evolutivos, que representa el enfoque de modularidad interna. Establece que la tarea limpieza de entornos para robots evolutivos presenta bloqueos en espacios de solución y que puede ser utilizada para validar arquitecturas modulares.	1997.
Extensión de *Arquitectura Modular Emergente*	Ziemke *et al.* (1999)	Ponen a prueba la *arquitectura modular emergente* contra algunas variantes de la misma, confirmando el buen rendimiento de esta arquitectura en la tarea de limpieza de entorno.	1999.
Control Neuronal Modular para máquinas caminantes	Manoonpong *et al.* (2008)	Un sistema de control basado en modularidad y creado para el control de robot cuya locomoción se realiza mediante patas. Se utilizan tres redes neuronales artificiales. La arquitectura está basada en jerarquía en donde las salidas de las dos primeras redes son las entradas de la siguiente red. Representa a la modularidad externa.	2005.
Proceso evolutivo para la optimización de la morfología de redes modulares	Mouret y Doncieux (2009)	Creación de un procedimiento para optimizar la morfología de redes neuronales artificiales y facilitar la creación de sistemas de control bajo el enfoque de la modularidad interna.	2009.
Modelo de selección de Acción	Krichmar (2012)	Creación de un modelo de selección de acción basado en la acción de los neurotransmisores Dopamina, Serotonina y Acetilcolina que afecta los estados emocionales de un robot.	2012.
Evolución de neuro-controladores híbridos para tareas complejas	Duarte *et al.* (2015)	Propuesta de un par de tareas para probar arquitecturas modulares: rescate de compañero y cambio de cuarto. Propuesta de un modelo jerárquico de control basado en redes neuronales aplicable al robot e-puck.	2014.
Aprendizaje y neuro-transmisores artificiales	Noruuzzadeh y Clune (2016)	Se demuestra mediante una plataforma robótica que la inclusión de neuromodulación en redes neuronales artificiales es un factor que mejora el aprendizaje de las mismas.	2016.
Arquitectura Acetilmodulada	Aldana-Franco, Montes-González y Ochoa (2017a)	Propuesta de una arquitectura de módulos externos regulada por un modelo de la función de la acetilcolina el los seres vivos, su acción en la sinapsis química y el trabajo inhibitorio de las neuronas de Renshaw.	2017.

Capítulo 3.

Metodología.

En esta sección se describe detalladamente la metodología utilizada para dar respuesta a la problemática señalada en el primer capítulo. La investigación se dividió en cuatro experimentos, uno de ellos es exploratorio, dos comparativos y uno descriptivo. En cada uno se presentan las herramientas utilizadas, la predicción de los resultados y el diseño experimental.

3.1 Experimento 1: estudio exploratorio.

El primer paso dentro de la investigación fue el diseño de la red modular (módulos externos). Se implementó un sistema de redes neuronales vistas como un conjunto de módulos capaces de resolver diferentes tareas. Para ello se probaron cinco configuraciones propias y dos configuraciones probadas previamente (Ziemke, 2017).

3.1.1 Problemática.

Como se expone en el capítulo anterior, uno de los principales problemas con las arquitecturas modulares internas en la robótica evolutiva es su tendencia a la monopolización de los recursos. Este fenómeno provoca que sólo uno de los módulos capaces de resolver cierta tarea acapare el control de los actuadores del robot. De esta manera, se buscó que las topologías de red puestas a prueba eviten este problema, permitiendo que los diferentes módulos intervengan en la solución general del problema planteado.

En el caso de los sistemas de control utilizados en robots evolutivos, es difícil cuantificar cuándo una activación de ciertos módulos es correcta o no. Esto debido a que depende de las circunstancias y el dinamismo del medio. Por ello el dominio de aplicación de las redes neuronales probadas fue una tarea de clasificación, aplicada a la base de datos Fisher Iris (Bazdek, Keller, Krishnapuram, Kuncheva y Pal, 1999), la cual se detalla en la Sección 3.1.2. De esta manera, fue posible conocer bajo qué circunstancias cada módulo toma parte en la solución del problema.

3.1.2 Objetivo del experimento.

El propósito del experimento fue encontrar aquella configuración que evitará el fenómeno de la monopolización de recursos, y que además tenga el mejor rendimiento en el dominio de la tarea de clasificación. También es propósito brindar un panorama del comportamiento de algunas configuraciones de redes modulares en una tarea en la que es posible cuantificar el éxito de la discriminación. Es decir, se trata de un experimento con fines de diseño de un controlador de robots evolutivos, en donde la tarea se cambia a un domino de aplicación en donde es posible conocer las salidas correctas ante la excitación presentada a la red. Pero que también permitió conocer si un módulo tomaba parte de la solución a la entrada presentada de manera correcta. Lo

cual implicó que la red solución evitó que un sólo módulo tome parte de la solución en todos los casos, dejando al resto inutilizables.

3.1.3 Software y hardware.

Tanto los procesos de optimización como los de validación de las topologías se realizaron a través del software libre de licencia *CodeBlocks* 13.12 y el lenguaje de programación C++ para la construcción del algoritmo genético generacional y las redes neuronales artificiales. Para disminuir los efectos por el uso de diferentes computadoras, todos los procesos computacionales corrieron en una sola computadora bajo el sistema operativo Windows 8.1, con procesador AMDA8 @ 1.60GHz y 8GB en RAM.

3.1.4 Base de datos.

La base de datos Iris de Fisher se compone de 150 ejemplos: tres clases y cuatro atributos. Se trata de una colección de información para la clasificación de un tipo de flor llamada iris y sus subespecies. De esta manera las tres clases corresponden a las subespecies: *setosa*, *veriscolor* y *virgínica*. Los cuatro atributos representan las medidas del largo y ancho del sépalo, así como el largo y ancho del pétalo. Para el primer atributo los valores máximos y mínimos son 7.9 y 4.3, respectivamente. El valor máximo del segundo atributo es 4.4, mientras que el mínimo es 2.0. En el caso del tercer atributo, los valores máximo y mínimo son 6.9 y 1.0, respectivamente. Finalmente, el cuarto atributo tiene como valor máximo 2.5 y mínimo 0.1. La base no presenta datos faltantes.

3.1.5 Topologías.

Cada topología puesta a prueba (ver Figura 11), constaba de tres módulos independientes que correspondían con cada una de las tres clases presentes en la base de datos. De esta manera, la modularización se realizó en función de la clase o lo que es lo mismo, a través de componentes matemáticos básicos que representan la función de aptitud general. Gracias a ello era posible conocer cuántas veces un módulo se activaba en la clase correspondiente.

La capa de entrada de cada uno de los tres módulos en las siete diferentes configuraciones de red neuronal probadas estaba integrada por cuatro neuronas. Cada una de ellas corresponde con los cuatro atributos que tiene la base de datos.

La capa media de todos los módulos estaba constituida por tres neuronas. Por su parte la capa de salida de cada módulo contaba con una neurona que determinaba si el ejemplo presentado correspondía a la clase que representaba o no. La función de transferencia utilizada en todas las neuronas que componen cada topología fue la sigmoidal.

En la primera topología puesta a prueba, conocida como *G1*, se incluyó una neurona adicional en la capa de salida. Su función era si el módulo estaba activo o no. Además, se agregó una capa superior (de interacción) compuesta de tres neuronas en donde se integraban los estados de las

neuronas adicionales de las capas de salida. Las conexiones a las neuronas de la capa de interacción eran de dos tipos: inhibitorias y excitatorias. Las conexiones inhibitorias provenían de las neuronas de activación de módulos distintos al correspondiente con la neurona de la capa superior. Por su lado las conexiones excitatorias provenían de la neurona de activación asociada a la neurona de la capa superior. El mecanismo para conocer el módulo que interviene en la clasificación de cada ejemplo presentado fue el máximo nivel de salida de las neuronas en la capa de interacción.

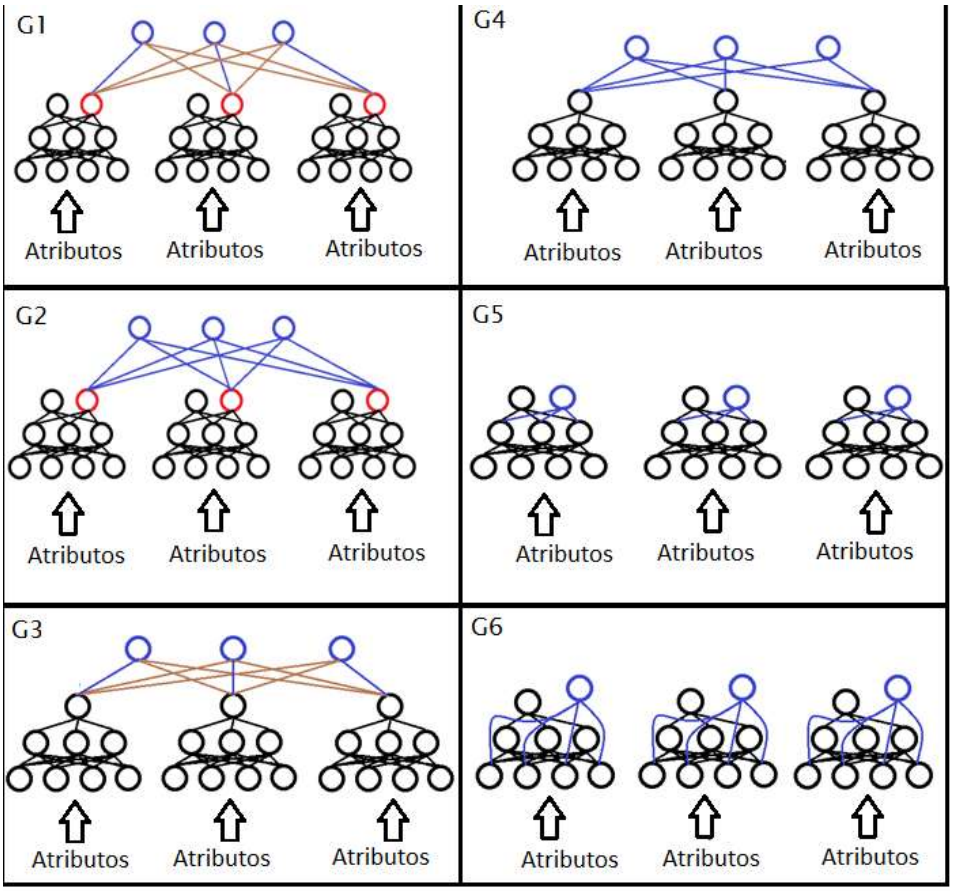


Figura 11. Topologías del estudio exploratorio *G1* a *G6*. Las neuronas en negro se involucran en la tarea de clasificación, las neuronas en azul intervienen en funciones de interacción, las líneas en color café son conexiones inhibitorias, mientras que las líneas en azul representan conexiones excitatorias. Las neuronas en rojo sirven para conocer el estado de la actividad del módulo. Todos los módulos reciben como entrada los cuatro atributos de la base (ancho y largo del pétalo, ancho y largo del sépalo).

La segunda topología, conocida como *G2* incluyó una configuración similar a *G1*. En este caso se omitió la presencia de conexiones inhibitorias en las neuronas de la capa de interacción modular. Así las neuronas en dicha capa recibían conexiones excitatorias de las neuronas con actividad en los tres módulos.

En cuanto a la tercera topología (*G3*), se excluyó la neurona de actividad neuronal presente en *G1* y *G2*. De esta manera, las conexiones sinápticas que abastecen de información a las neuronas de interacción provenían de la neurona de salida de cada uno de los módulos. Dichas conexiones podían ser excitatorias si provenían del mismo módulo, o inhibitorias si provenían de otro módulo.

Para la cuarta topología (*G4*), se repitió la configuración de *G3* pero sin incluir conexiones inhibitorias a las neuronas de interconexión. De esta manera todas las neuronas de interacción modular recibían únicamente conexiones del tipo excitatorio. Dichas conexiones provenían de las neuronas de la capa de salida de cada uno de los módulos.

La quinta topología (*G5*) puesta a prueba es una variante de la *Arquitectura Modular Emergente* de Nolfi, en donde una red neuronal de módulos internos se utiliza como sistema de control para el robot Khepera que debe limpiar su entorno depositando cilindros de colores fuera de la arena. En el caso de la topología *G5*, se realizó un arreglo para módulos externos y las conexiones de la neurona selectora de cada módulo provenientes de la capa media. Así las neuronas de competitividad recibían conexiones sinápticas de la capa intermedia de cada módulo. El módulo que tomaba el control de la solución de cada instancia es aquél cuya neurona de competitividad posee el mayor nivel de salida.

La sexta topología (*G6*) es otra versión la *Arquitectura Modular Emergente* arreglada como módulos externos. La principal diferencia entre la topología *G5* y la *G6* fue que las conexiones de las neuronas de decisión estaban directamente ligadas a la capa de entrada. La competencia entre módulos se basa en las neuronas de competitividad incluidas en la sexta topología. Así el sistema de interacción entre módulos funcionaba igual que el de la arquitectura *G5*.

Finalmente, la topología *G7* o red *acetilmodulada* (ver Figura 12) estaba conformada por tres módulos, que clasificaban las diferentes instancias presentadas con cuatro neuronas en la capa de entrada, tres en la capa media y una neurona en la capa de salida. También tenía una red moduladora constituida por las mismas entradas que los módulos, pero adicionando neuronas de estados que contienen la información de cuál de los módulos se activó en la clasificación anterior. Dicha red neuronal representaba un ganglio basal y estaba integrada por cuatro neuronas de entrada que representan a los atributos de la base de datos. Además de tres neuronas que representaban el estado anterior de cada uno de los tres módulos. La capa estaba constituida por tres neuronas. En la capa de salida tenía tres grupos de dos neuronas. Cada grupo contenía una neurona que indica si se produce o no una sustancia artificial conocida modulador. Esto se realizaba a partir de neuronas binarias de salida. La otra neurona producía un nivel de activación del módulo, con un valor entero continuo que va de 0.0 a 1.0.

En caso de existir producción del modulador artificial, éste viajaba y se concentraba en las neuronas de la capa intermedia para ser liberada sobre las neuronas de la capa de salida o neuronas efectoras en un instante de tiempo (ver Figura 13). Este funcionamiento emula el fenómeno de los neurotransmisores, especialmente la acetilcolina que actúa sobre las neuronas motoras. La sustancia química se guarda en las vesículas de la neurona pre-sináptica de la inter-

neurona (capa media), de donde se liberan para producir sinapsis química con una neurona post-sináptica a través de los receptores colinérgicos. De esta manera la red reguladora no debía enviar el modulador a más de un módulo o red. Se utilizaron dos medios de verificación para impedirlo. El primero es la neurona de producción del modulador. En caso de que más de una neurona produjera la sustancia artificial, entonces se verificaba el mayor de los niveles en la neurona de intensidad para determinar cuál de los módulos brindará solución a la instancia presentada. Entonces, la inhibición de los módulos que no participaban en la solución se realizaba mediante un mecanismo similar al de las neuronas de Renshaw. Es decir, se inhibían lateralmente las neuronas de productoras de acetilcolina de los módulos o ganglios que no formarían parte de la solución. Esta topología combinaba una representación de la modularidad externa, el modelado de un mecanismo neurofisiológico como lo es la acetilcolina y su acción en las neuronas motoras, el efecto de las neuronas de Renshaw, así como la acción de la sinapsis química en las neuronas de las estructuras cerebrales de los seres vivos.

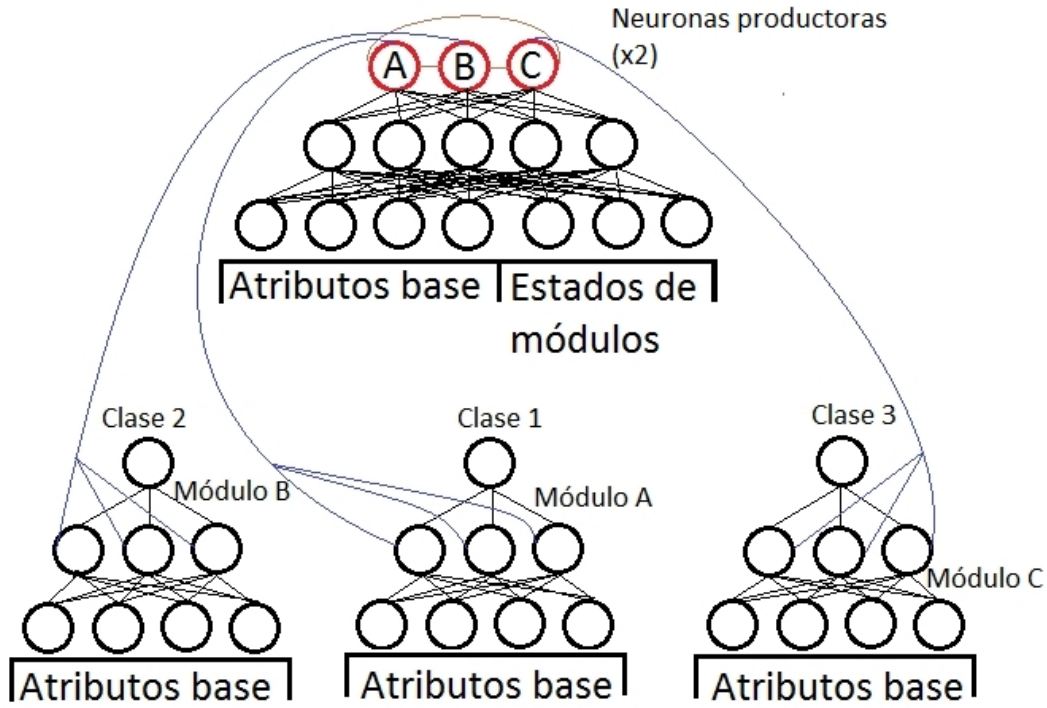


Figura 12. Topología *G7* para el estudio exploratorio. Esquema de conexión de la topología *G7* o red *acetilmodulada* para el estudio exploratorio, que consta de tres módulos que clasifican cada una de las tres clases posibles. También se incluye una red productora de ACh virtual. Las conexiones en azul son los canales de la transmisión de acetilcolina virtual hacia las inter-neuronas de cada módulo. Las conexiones en café son inhibitorias laterales entre módulos. Cada grupo está conformado por dos neuronas: una que habilita o inhibe la producción y otra que determina el nivel de producción.

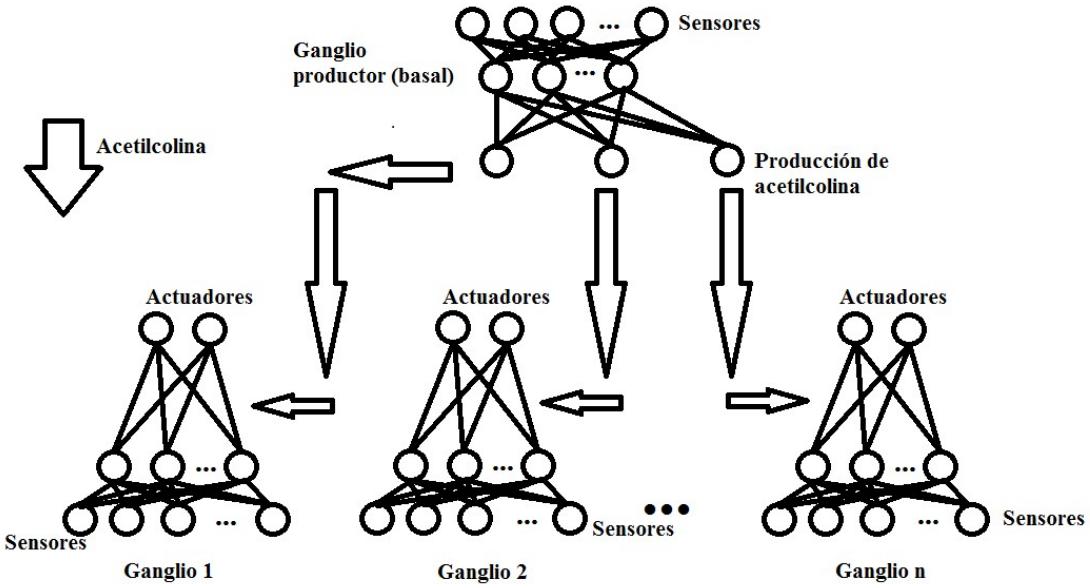


Figura 13. Red *acetilmodulada*. Un conjunto de ganglios o redes en forma de módulos compiten por el acceso al aparato motriz del robot (actuadores). Una red reguladora produce niveles de acetilcolina virtual, la cual de ser seleccionada viaja a las inter-neuronas de los ganglios y les brinda la posibilidad de realizar sinapsis con la motoneurona, regulando de esta manera el acceso a los recursos. La producción de acetilcolina virtual es excluyente entre las neuronas gracias al mecanismo inspirado en la acción de las neuronas de Renshaw. Con ello se garantiza que sólo una de las neuronas productoras envíe niveles del transmisor virtual al módulo correspondiente y señales inhibitorias al resto de neuronas de salida de la red reguladora. El envío de la acetilcolina virtual se asocia al valor más alto de activación en las neuronas de salida de la red reguladora.

3.1.6 Proceso evolutivo.

El proceso evolutivo al que fueron sometidas las topologías constaba de dos partes. La optimización de los pesos de aquellas conexiones involucradas en la clasificación de una instancia y la optimización de los pesos de las conexiones involucradas en la interacción modular. La primera fase optimizaba de manera independiente cada uno de los módulos. Mientras que el segunda el proceso evolutivo afecta a todos los módulos al mismo tiempo, pero únicamente a los pesos de las interacciones modulares. Así se dejaban intactos los pesos de las tareas de clasificación de cada módulo.

Se utilizó un algoritmo genético generacional para todos los procesos de optimización. En todos los casos los parámetros son los mismos. La representación genética fue binaria (9 bits), con valores de 0.0 a 5.0 para conexiones excitatorias y -5.0 a 0.0 para conexiones inhibitorias. El número de generaciones fue 100 y cada una estaba compuesta de 100 individuos. Se utilizó 2% de mutaciones aplicados al total de los bits de toda una generación. Para conservar intacto al mejor individuo de cada generación se empleó el operador elitismo.

Referente a la recombinación de individuos el operador de selección se basó en sorteo, eligiendo cuatro posibles candidatos aleatoriamente por cada padre potencial. Se elegía al de *fitness* más alto para aplicar el operador de cruza. Así el operador de cruza funcionó con un punto de cruza fijo y dos padres. También se utilizó un mecanismo de sustitución que conservaba hasta el 20% de individuos de la generación anterior.

La función de calidad para la optimización de los módulos en cualquiera de los dos posibles procesos se basó en la cantidad de ejemplos bien clasificados. Se probaron 105 ejemplos, que representan el 70% de la base de datos, elegidos aleatoriamente para cada individuo para la optimización. La validación utilizó los 150 ejemplos de la base.

3.1.7 Diseño Experimental.

El diseño experimental factorial busca cubrir la necesidad de conocer diferencias estadísticas entre grupos experimentales y mantener la validez interna. Para ello se crean diferentes grupos experimentales que representan a las diferentes topologías. De esta manera cada proceso evolutivo de las diferentes topologías representa a un individuo de prueba. Los procesos evolutivos de cada grupo experimental se repitieron 90 veces para cada topología. Se utilizó al mejor de los individuos de la última generación de cada proceso evolutivo para ser sometidos a la totalidad de la base de datos y medir las variables de interés. En todos los casos, se aplicaron mediciones post-prueba a cada individuo en los grupos experimentales.

Se midieron dos variables dependientes:

- La primera conocida como activación modular sirvió para conocer cuando un módulo se activó de manera correcta. Se contabilizó el número de ocasiones en las que la clasificación del ejemplo provenía del módulo corresponde a la clase en cuestión (1, 2 ó 3).
- La segunda variable, ejemplos clasificados correctamente, cuantificó la cantidad de ejemplos clasificados correctamente al exponer cada topología de módulos a la base completa. En ambos casos, los valores posibles son números enteros positivos de 0 a 150.

La variable independiente correspondió a las siete diferentes topologías puestas a prueba. Por ello sus valores posibles son los números enteros: 1, 2, 3, 4, 5 y 6, 7 (ver Figura 14). Los primeros cuatro grupos corresponden a las topologías propuestas en base a las acciones de inhibición y excitación de neuronas, el grupo 5 representa a la topología propuesta por Nolfi (1997), el grupo 6 representa a una de las topologías puestas a prueba por Ziemke *et al*. (2004), y el *G7* fue una arquitectura cuya propuesta está inspirada en la acción del neurotransmisor acetilcolina en las neuronas motoras y la sinapsis química llamada *acetilmodulada*.

Las principales variables de control fueron las que intervienen en la capacidad de procesamiento como el procesador, la cantidad de memoria de la computadora de pruebas y el sistema operativo. Por ello los experimentos se realizaron en la misma computadora, evitando así la influencia de los cambios de las características entre equipos de cómputo.

Para buscar diferencias significativas se utilizó una prueba estadística de Kruskal Wallis o ANOVA de una vía para rangos aplicada a cada uno de los siete grupos experimentales en busca de diferencias significativas que permitan establecer a la mejor topología (ver Figura 14). Cada grupo experimental estaba compuesto por 90 mediciones de las dos variables dependientes. En la selección de la muestra intervinieron factores como tiempo de procesamiento de los procesos evolutivos. Se utilizó una prueba post-hoc Student -Newmann-Keuls. En ambos casos se utilizó un criterio de $p<0.05$.

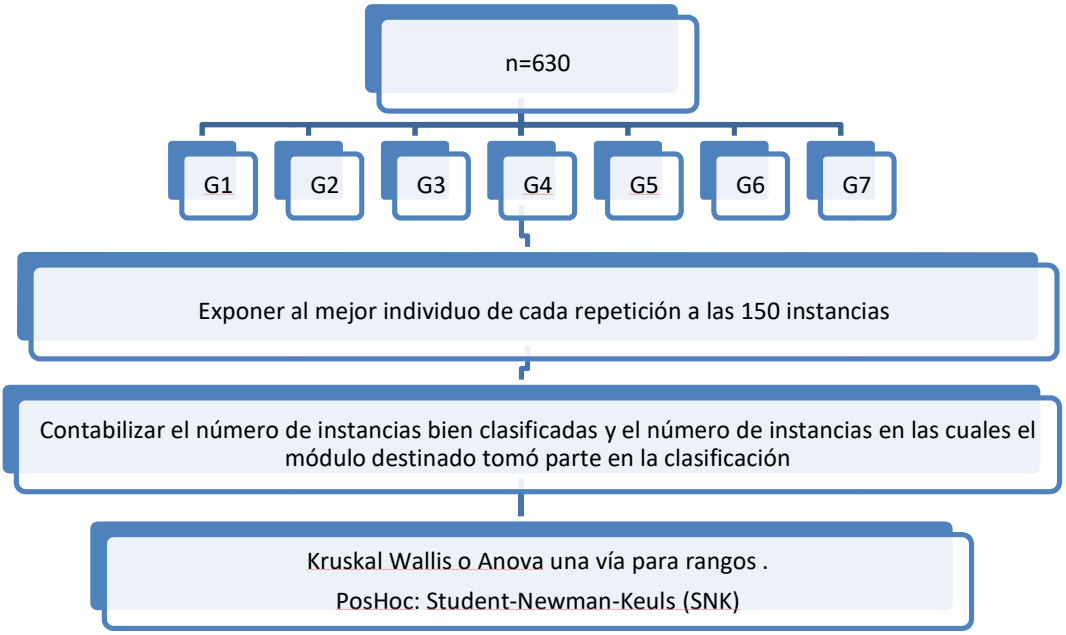


Figura 14. Diseño experimental (experimento 1). Se comparan siete grupos experimentales mediante dos variables dependientes: ejemplos bien clasificados y módulos activados de manera correcta.

3.2 Experimento 2: Limpieza del entorno (estudio comparativo).

3.2.1 Problemática.

Para poder validar el funcionamiento correcto de la *arquitectura acetilmodulada* presentada en el experimento anterior, y conocer la capacidad de la misma para evitar los bloqueos en espacios de soluciones se optó por la construcción de un experimento que permitiera comparar tres diferentes arquitecturas: una no modular, una de módulos internos y otra de módulos externos. El entorno y la tarea fue elegida debido a dos factores: la posibilidad de dividir las tareas en módulos a partir de su función de calidad y que fue previamente utilizada en otras investigaciones relacionadas con fenómenos modulares (Ziemke, Carlsonn, y Bodén, 1999). La tarea seleccionada representa un potencial escenario de bloqueo entre tareas: levantar y tirar basura.

3.2.2 Objetivo del experimento.

Mediante este experimento se esperaba observar un desempeño similar o mejor de la arquitectura de módulos externos respecto a la de módulos internos. La arquitectura de módulos internos fue probada en investigaciones previas (Nolfi, 1997), mostrando buenos resultados en términos de resolución de la misma, pero con efectos de monopolización de recursos por parte de algunos módulos. También se esperaba observar que el peor rendimiento correspondiera a la arquitectura no modular. Esto debido a que en investigaciones previas (Nolfi, 1997), (Ziemke, 2017) se ha demostrado que una sola red neuronal es incapaz de solventar el control de robots para esta tarea.

3.2.3 Entorno, tarea, y robot.

La tarea elegida para el segundo experimento es la de limpieza de entorno con el modelo computacional. Mismo que fue usado para validar el uso de la *Arquitectura Modular Emergente*. El entorno original consistía en una arena delimitada por paredes, y dentro se encontraban elementos de recolección (cilindros). Un robot Khepera (Mondada, Franzi y Guignard, 1999) debía ser capaz de recoger los objetos con la pinza, y depositarlos fuera de la arena. Por cada objeto que el robot recogiera y colocara fuera, recibía un punto en su función de aptitud.

Así, se realizó una adaptación al experimento para el robot e-Puck, que no cuenta con una pinza para poder manipular objetos. El entorno (ver Figura 15) consiste en una arena delimitada por cuatro paredes de color azul. El color del piso de la arena fue gris y existían dos zonas circulares para recolección y depósito de basura. Las zonas eran identificables para el robot a través de los sensores de piso. En el caso de la zona de recolección, el color para identificar era el blanco. Por otra parte, la zona para depositar basura estaba identificada con el color negro. Adicionalmente, se colocó un cilindro de color rojo a las zonas de depósito y recolección de basura. La finalidad fue crear un apoyo adicional al sistema sensorial del robot y que pudiera identificar de mejor manera las zonas, utilizando los componentes RGB de la cámara. Un robot recolectaba una unidad de basura por cada paso del simulador o "*step*" que pasaba dentro de la zona correspondiente. Lo mismo sucedía con el depósito de basura, dejando una unidad por *step* transcurrido en la zona negra.

El robot e-puck (Mondada *et al.*, 2009) (ver Figura 16) es una plataforma open hardware, creada con fines educativos. Fue construido por el Instituto Politécnico de Lausana, y la versión básica cuenta con diferentes sensores y actuadores como: infrarrojo, sensor de luz, cámara VGA de 640x480 pixeles, motores a pasos, anillo de LEDs, giroscopio, micrófono, bocina, puerto serial, comunicación Bluetooth, y varios puertos para extender sus características electrónicas. Se basa en un dispositivo microcontrolador del fabricante Microchip: dsPIC30F6014A. Es un dispositivo electrónico digital de la familia TTL, que puede programarse mediante lenguajes de bajo nivel como el ensamblador, y de alto nivel como C. Los modelos computacionales del robot, sus sensores y actuadores, se encuentran disponibles en muchos simuladores comerciales y de libre uso. Entre ellos destacan: Matlab (Samiloglu, Cayirpunar, Gazi y Koku, 2008), Webots (Michel,

1998) y el simulador especializado en robótica evolutiva FARSA (Massera, Ferrauto, Gigliotta y Nolfi, 2013).

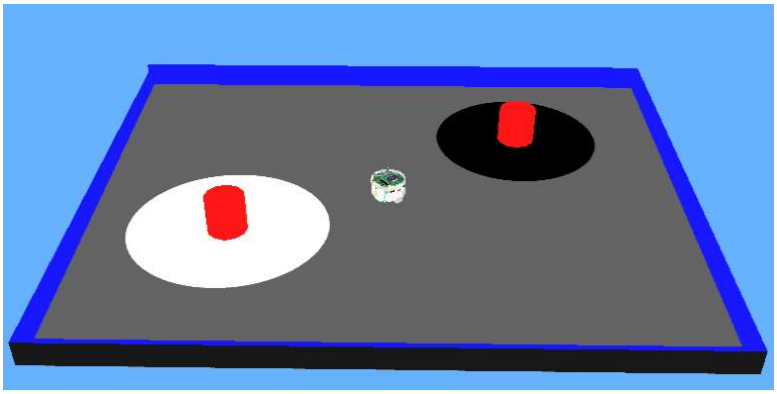


Figura 15. Medio ambiente virtual para el experimento limpieza de entorno. Un robot debe visitar la zona circular blanca, en donde por cada step recoge una unidad de basura virtual. En la zona de depósito (área circular objetivo de color negro o "target") el robot deposita una unidad de basura por cada step en ella.

Figura 16. Robot e-puck. Es un robot fabricado con propósitos educativos basado en un micro-controlador de la familia PIC. Tomada de: openswarm.org/supported-hardware/

Para resolver la tarea, el modelo virtual del robot utilizó los sensores infrarrojos para detectar obstáculos cercanos. De la misma manera, se emplearon las componentes Roja y Azul de la cámara VGA, los sensores de piso para detectar tres colores (gris, blanco y azul). Los actuadores que fueron empleados en este experimento fueron los motores a pasos.

Se utilizó el simulador Webots en la versión 6.1.3 como plataforma virtual para llevar a cabo los experimentos. Este simulador que cuenta con una gran variedad de modelos computacionales para robots. Permite la creación de una amplia variedad de escenarios para diversas tareas. Opera a partir de programas que controlan los modelos virtuales de los robots. Dichos programas pueden ser realizados a partir de diferentes lenguajes de programación. En el caso de este experimento, el lenguaje de programación utilizado fue C++.

3.2.4 Topologías.

Se probaron tres tipos diferentes de redes durante este experimento (ver Tabla 2): arquitectura no modular, *arquitectura modular emergente* y *arquitectura acetilmodulada*. El caso de la primera es un sistema de control basado en una sola red neuronal que reguló las acciones de recolección y depósito de basura. Constaba de 27 neuronas (ver Figura 17) distribuidas de la siguiente manera: 20 correspondientes a los sensores, cinco en la capa media y dos salidas. Se utilizó la arquitectura de red *Feedforward* y la función de transferencia del tipo *sigmoide* que aumenta el grado de deliberación del sistema. Este par de características son variables controladas para el resto de las topologías del experimento, y por lo tanto fueron iguales para los otros dos sistemas de control.

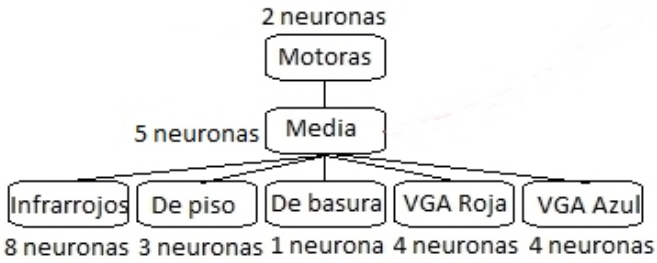


Figura 17. *Monored* para el control de robots dentro del experimento limpieza de entorno. Una sola red neuronal artificial modifica el comportamiento de los robots evolutivos. La red está constituida por ocho sensores infrarrojos, tres de piso, uno de basura, cuatro segmentos de la cámara para la componente Roja y cuatro para la componente azul, cinco neuronas en la capa media, y dos neuronas para controlar los motores.

La capa de entrada constaba de ocho sensores infrarrojos con codificación binaria, en donde el número 0 corresponde a la ausencia de objetos y el 1 a la presencia de los mismos. Tres neuronas de la capa de entrada estaban vinculadas a los sensores de piso que operaban de forma especializada y con valores binarios. Esto quiere decir que cada una de las neuronas se especializó en la detección de un color (gris, blanco y negro), cuyo valor era el número 1 en presencia del color y 0 en ausencia. Una de las neuronas de la capa de entrada estaba asociada al estado de recolección de basura. Este nivel fue representado con una variable que se sufre un decremento en cada *step* que el robot pasaba en una zona de depósito. La misma variable incrementa su valor por cada *step* en la zona de recolección. Existía un valor mínimo y un valor máximo, y la neurona recibía un valor escalado entre 0.0 y 1.0. En cuanto a la cámara VGA, se utilizó un arreglo de neuronas que dividían en segmentos y en componentes las imágenes recibidas. De tal manera, se tenían cuatro neuronas que correspondían a la componente roja de una imagen. Cada una determinaba la presencia o no de dicha componente en un sector de 72 pixeles. Lo mismo ocurre con cuatro neuronas para la detección de la componente azul. De esta manera, se divide la detección de los colores en la imagen y brinda información al robot sobre la orientación que debe seguir el robot respecto a los objetos detectados. Las neuronas de este tipo también operaban en valores binarios, en donde el número 0 corresponde a la ausencia de la componente a detectar y 1 a la presencia del color detectado.

Se utilizaron cinco neuronas en la capa media. Las neuronas de la capa de salida controlaban la velocidad con la que el robot avanza. Así, cada una de las neuronas corresponde con la velocidad que se representa en el robot. No se consideraron velocidades negativas para las salidas de los motores.

Por su parte, la red que representó a la *arquitectura modular emergente* (ver Figura 18). Una red dividida en dos regiones en la capa de salida tenía de 36 neuronas distribuidas de la siguiente manera: 20 neuronas en la capa de entrada, diez en la capa de media y ocho en la capa de salida. Esta arquitectura representa el fenómeno de la modularidad interna. Se utilizaron los mismos sensores que los reportados en la arquitectura anterior: ocho sensores infrarrojos, tres sensores de piso, un sensor virtual para manejar el estado de la carga de basura, cuatro sensores de la componente roja de la cámara y cuatro de la componente azul. En la capa media se utilizaron 10 neuronas. El mayor cambio ocurrió en la capa de salida en donde se formaron grupos de dos neuronas que corresponden a los diferentes módulos de la arquitectura. Dentro de dichos grupos, una neurona devolvía el posible valor aplicable a la velocidad del motor. La otra neurona sirvió para determinar a cuál de los dos módulos se otorga el recurso por el cual compiten. De tal manera, que la activación más alta determinaba el módulo que toma el control del motor. Dos de los grupos se enfocaron en el manejo del motor derecho (A y B), compitiendo por el acceso al actuador, y los otros dos módulos en el control del motor izquierdo del robot (C Y D). Se utilizó el enfoque proximal para la descomposición de la tarea, por ello se consideraron los motores para modularizar el sistema.

Tabla 2. Características de las redes comparadas en el experimento limpieza de entorno.

Característica	*Acetilmodulada (ACh)*	*Monored*	*Arquitectura Modular Emergente (AME)*
Neuronas	Reguladora: 38, Módulos: 32	27	38
Pesos	Reguladora:304, Módulos: 240	130	300
Entradas	Reguladora: 22((8 infrarrojos, 3 piso, 1 basura, 4 R y 4 B, 2 estado módulos), Módulos: 20 (8 infrarrojos, 3 piso, 1 basura, 4 R y 4 B)	20 (8 infrarrojos, 3 piso, 1 basura, 4 R y 4 B)	20 (8 infrarrojos, 3 piso, 1 basura, 4 R y 4 B)
Capa media	10	5	10
Salidas	Reguladora: 4, Módulos: 2	2	8
Funciones de transferencia	*Sigmoidal*	*Sigmoidal*	*Sigmoidal*
Arquitectura	*FeedForward*	*FeedForward*	*FeedForward*

Por último, la *arquitectura acetilmodulada* (ver Figura 19) constaba de tres redes neuronales: una para controlar la recolección de basura, otra para el depósito de basura y la última para la regulación de la interacción de las redes a partir de la producción de un transmisor artificial similar a la acetilcolina.

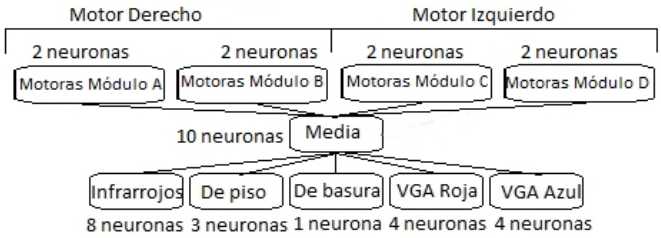


Figura 18. Red de módulos internos para el control de robots dentro del experimento limpieza de entorno. Una sola red neurona artificial dividida en dos módulos internos modifica el comportamiento de los robots evolutivos. La red está constituida por ocho sensores infrarrojos, tres de piso, uno de basura, cuatro segmentos de la cámara para la componente Roja y cuatro para la componente azul, diez neuronas en la capa media, dos neuronas para controlar los motores y una neurona selectora en cada módulo. El control de los actuadores se le otorga al módulo con el valor de salida en la neurona de selección más alto.

Tanto la red para el control de la recolección como para la red de depósito, el número de neuronas que las constituían fue de 32 que corresponden a: 20 neuronas en la capa de entrada,

diez neuronas en la capa media y dos neuronas en la capa de salida. Se utilizó el mismo aparato sensorial de las redes anteriores para este par de redes, que consta de: ocho sensores infrarrojos, tres de piso, uno de estado de carga de basura, cuatro componentes rojas y cuatro componentes azules de la cámara VGA. Las salidas de las dos neuronas de la capa de salida controlan la velocidad de los motores del robot, siempre y cuando el mecanismo de producción permita la llegada del neurotransmisor a la entrada de las neuronas para hacer sinapsis química.

La red moduladora estaba formada por 38 neuronas: 24 en la capa de entrada, diez en la capa media y cuatro en la capa de salida. En esta red se produjo el transmisor artificial que permitía la activación de los dos módulos. El aparato sensorial estaba constituido por: ocho sensores infrarrojos, tres sensores de piso, un sensor de estado de basura, cuatro neuronas para la componente roja y cuatro para la componente azul de la cámara, además de dos sensores que brindaban información sobre los estados de activación de los módulos en el *step* anterior.

Todos los sensores operaban bajo los parámetros mencionados en los sistemas de control anteriores, excluyendo a los sensores de estado. Estos operaban en forma binaria, con un valor igual a uno si el módulo al que representaba había hecho sinapsis y cero en caso contrario. Las neuronas de la capa de salida de esta red neuronal operaban de la siguiente manera: Se formaron dos grupos de dos neuronas. Cada una representaba a uno de los módulos que podían tomar el control del robot. Una de las neuronas de cada grupo tenía una salida del tipo binaria que determina si se producía o no el transmisor artificial. La otra neurona determinaba la intensidad de la sustancia virtual en un rango de valores continuos de 0.0 a 1.0.

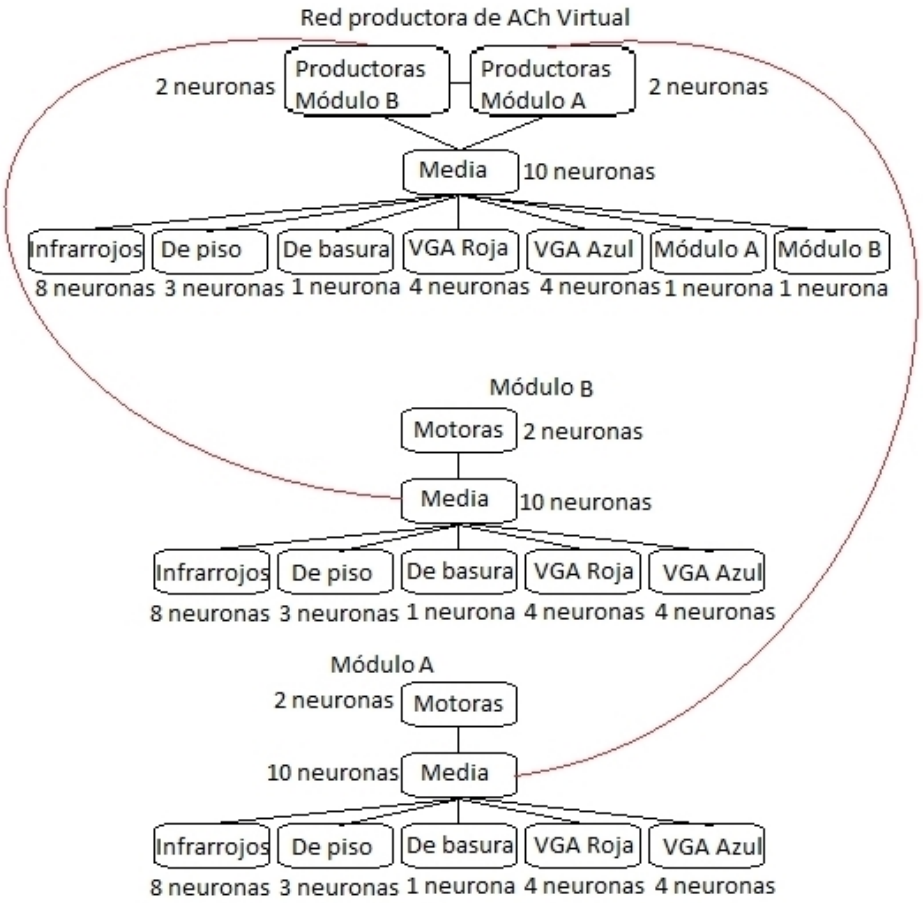


Figura 19. Red *acetilmodulada* para el control de robots dentro del experimento limpieza de entorno. Dos módulos compiten por el control de los actuadores del robot. El control se otorga a aquel módulo que reciba vía acetilcolinergica el neurotransmisor virtual a las inter-neuronas (neuronas de capa media) correspondientes. La red productora de ACh virtual determina cuál de los módulos tendrá el control de los actuadores a partir de la información del entorno y de los módulos. Al mismo tiempo inhibe la sinapsis en el módulo que no participará en la solución en un instante de tiempo.

El primer criterio para otorgar el control de los actuadores del robot a los diferentes módulos es la presencia de la sustancia virtual. En caso de que el valor de salida de las dos neuronas binarias fuera el número 1, entonces se tomaba en cuenta el valor de intensidad. Así se concedía el control al módulo con mayor intensidad en las neuronas de salida, permitiendo la sinapsis en sus neuronas efectoras.

3.2.5 Proceso evolutivo.

El proceso evolutivo de los tres sistemas de control basados en redes neuronales estuvo a cargo de un algoritmo genético. Dicho algoritmo era del tipo generacional operado a partir de selección por torneo, un sólo punto fijo de cruza aleatorio, mutación, sustitución y elitismo para conservar al mejor individuo de cada generación. En cuanto a la codificación de los pesos, se utilizó un código binario natural de 9 bits. Los valores que podía tomar iban del 0.0 al 1.0, y una resolución por

cambio de bit de $1.99x10^{-4}$. Los operadores de mutación y sustitución actúan directamente sobre los valores binarios de cada peso.

Durante cada proceso evolutivo la población inicial se prueba con la finalidad de obtener los puntajes de calidad de cada individuo. Se aplicó el operador de selección, el cual decide entre dos candidatos a padre elegidos aleatoriamente. La selección se basó en el puntaje de aptitud más alto. El siguiente operador en uso es el de cruza. Se toman dos padres de una lista realizada en el paso anterior y se eligió un punto de cruza aleatorio *M*. El cromosoma de los nuevos individuos se formó a partir de los cromosomas del padre A hasta el punto de cruza *M*, y del padre B después del punto de cruza.

Posteriormente se realizó la mutación de los individuos. Para ello se calculó el total de mutación por realizar basado en el porcentaje de mutación (2%) y el total de pesos que conforman el cromosoma de la red. Al final se aplicaron las mutaciones binarias (cambio de estado) a cada uno de los individuos padres. Posteriormente se utilizó un mecanismo de sustitución que podía mantener hasta el 20% de la población original. Además de conservar el cromosoma intacto del mejor individuo en cada generación. Una vez realizada la sustitución, se procedió a probar a los nuevos individuos dentro de una nueva iteración del algoritmo genético.

Los procesos evolutivos de las dos primeras topologías estuvieron compuestos por 100 generaciones. Para la tercera topología, el proceso de los módulos fue independiente y constaba también de 100 generaciones. Con los valores de los módulos optimizados de manera individual, se optimizaban únicamente los pesos de la red moduladora del transmisor virtual. Este proceso tomaba 100 generaciones.

Cada individuo fue puesto a prueba en un entorno virtual dentro del simulador Webots, durante tres evaluaciones o "*trials*" de 600 *steps* cada uno. La posición de los individuos al inicio de cada trial se determinaba de manera aleatoria. La función de calidad otorgaba un punto a los individuos que se encontraban en la zona de recolección de basura, siempre que no superaran la cantidad de basura máxima permitida (200 unidades). También otorgaba un punto por cada *step* que los robots se mantuvieran en la zona de depósito, siempre y cuando no hubieran alcanzado el mínimo de unidades de basura (cero unidades).

3.2.6 Diseño experimental.

Se utilizados tres grupos experimentales que correspondían con las tres topologías a comparar (ver Figura 20). El nombre de los tres grupos fue: *monored* que representaba a la arquitectura no modular, *AME* que correspondió a la *arquitectura modular emergente* y que representa a la modularidad interna, y *Acetilmodulada* que representaba a la arquitectura propuesta de módulos externos. Así, el diseño experimental estaba basado en un grupo control y dos tratamientos diferentes que buscan solucionar el bloqueo en espacios de búsqueda. Para los tres grupos experimentales se utilizó la medición post-prueba. Con este diseño se mantuvo la validez interna del experimento.

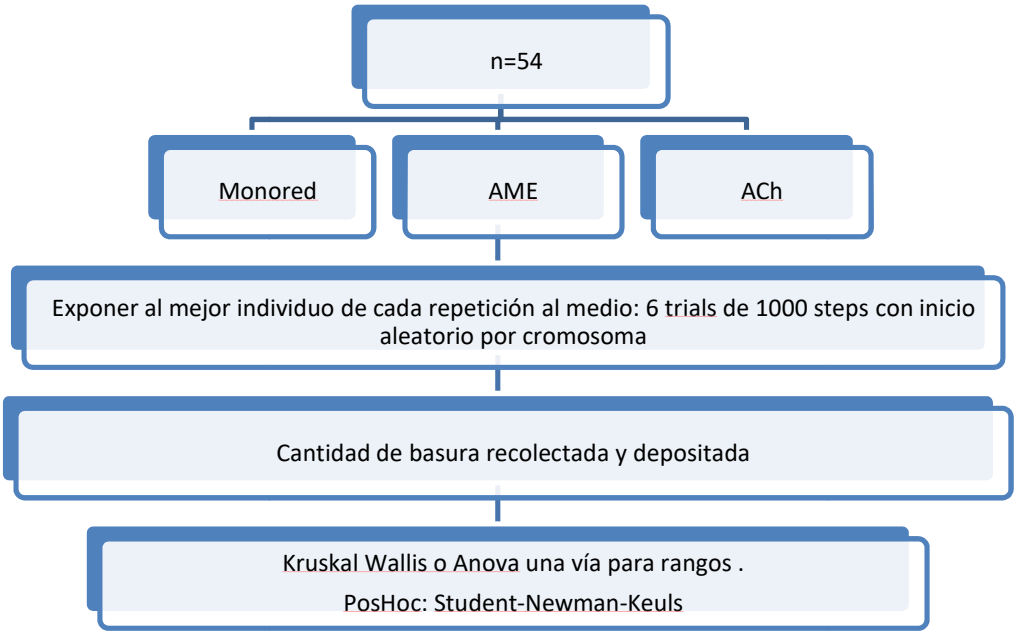


Figura 20. Diseño experimental (experimento limpieza de entorno). Se comparan tres arquitecturas de control para robots evolutivos basadas en diferentes configuraciones de redes neuronales. La variable a medir es el puntaje de aptitud en una prueba independiente al proceso evolutivo.

Cada grupo experimental constaba de 18 repeticiones de la evolución de la topología representada. Una vez finalizada cada evolución, se eligió al mejor individuo de cada repetición y fue probado en el entorno de la tarea. Con ello se obtuvo el valor de la variable dependiente, que en este caso es un puntaje de aptitud. Para ello se probó el mejor individuo de cada repetición en seis *trials* con posición inicial aleatoria, de 1000 *steps* cada uno. Este procedimiento post-evolución se repitió para cada grupo experimental. El valor de aptitud de los seis *trials* de cada individuo se promedió para obtener el puntaje de aptitud. La variable independiente o a manipular fueron los diferentes tipos de topologías, y la variable dependiente el puntaje de aptitud en la prueba post-evolución. Se utilizó dicha variable porque se esperaba que una mejor topología produjera individuos con mejor capacidad para realizar la tarea.

Así la predicción de este experimento en términos del diseño experimental fue que la topología basada en la acetilmodulación produce mejores individuos que el resto de las topologías, o al menos iguales que los de la arquitectura de módulos internos. Esto producto de la riqueza y robustez de su diseño y a la presencia del mecanismo de regulación modular propuesto.

Para comprobar la predicción del experimento, se utilizó una prueba estadística de Kruskal Wallis o ANOVA de una vía para rangos para buscar diferencias estadísticas, y una prueba post hoc Student -Newmann-Keuls para la comparación entre grupos dada la diferencia estadística. Ambos métodos estadísticos utilizaron un valor de significancia $p<0.05$.

Se utilizó este diseño experimental en función de las variables a medir: una variable dependiente y una independiente. También por la necesidad de conocer si existen diferencias estadísticas producto del uso de alguna topología en especial. El tamaño de la muestra está restringido al tiempo de procesamiento de los procesos evolutivos en los ambientes simulados.

3.3 Experimento 3: Cambio de cuarto (estudio comparativo).

3.3.1 Problemática.

Los sistemas modulares no siempre se reducen a un par de módulos entre los cuales se puede elegir a la mejor opción para resolver un problema general. Al aumentar la cantidad de módulos disponibles, también aumenta la complejidad de los espacios de búsqueda y pueden resultar más difíciles de explorar. Además de la posibilidad de otros problemas que pueden originarse derivados de esta situación. Por ello es importante estudiar la arquitectura propuesta y verificar que no sea susceptible al aumento de en la cantidad de módulos.

3.3.2 Objetivo del experimento.

Se compararon las mismas arquitecturas del experimento limpieza de entorno, y en todos los casos se realizaron las adecuaciones necesarias para el nuevo entorno. En particular, en los casos de la *arquitectura acetilmodulada* y la *arquitectura modular emergente* se aumentó la cantidad de módulos y la interacción entre ellos. Esto aumentó el nivel de complejidad del sistema de control en términos de las interacciones modulares.

3.3.3 Entorno, tarea, y robot.

La tarea y el entorno fueron una adaptación de un experimento ideado para comprobar arquitecturas modulares, en particular para generar el aprendizaje de una tarea en específico. En él se permite que los robots evolutivos adquieran habilidades para aplicarlas en un entorno más complejo como los laberintos y las tareas de rescate de compañeros (Duarte *et al.*, 2015). En la versión original, un robot e-puck se encuentra en una arena delimitada por cuatro paredes. En el centro de la arena se coloca una quinta pared que sirve de división para separar la arena en dos habitaciones. También incluye una pequeña puerta por la que se esperaba que el robot pueda abrir y cambiar de cuarto. Junto a la pared se coloca un dispositivo mecánico que, al ser empujado, el robot abre la puerta.

El entorno adaptado para este experimento (ver Figura 21) consistió en un cuarto delimitado por paredes azules y una arena de color gris. Justo a la mitad se incluyó una pared roja para formar los dos cuartos. En medio de la pared, se encontraba una puerta. Debajo de la zona de la pared roja, se colocó un segmento blanco en la arena como ayuda para señalizar al robot la cercanía de la puerta. En el extremo más lejano a la pared roja, se agregó una zona negra en la arena. El propósito fue generar un área en la que el robot pudiera realizar la acción de apertura de la

puerta. Se optó por un dispositivo que estuviera relacionado con el sensor de piso como mecanismo sensorio motriz asociado a la apertura de la puerta.

La tarea consistió en que el robot comenzaba dentro de una posición aleatoria en el cuarto del lado donde se encontraba el dispositivo de apertura. De esta manera, los individuos debían buscar y posicionarse sobre el espacio de color negro en la arena para poder abrir la puerta. Posteriormente debían identificar la puerta y cruzar a través de ella para llegar al otro cuarto.

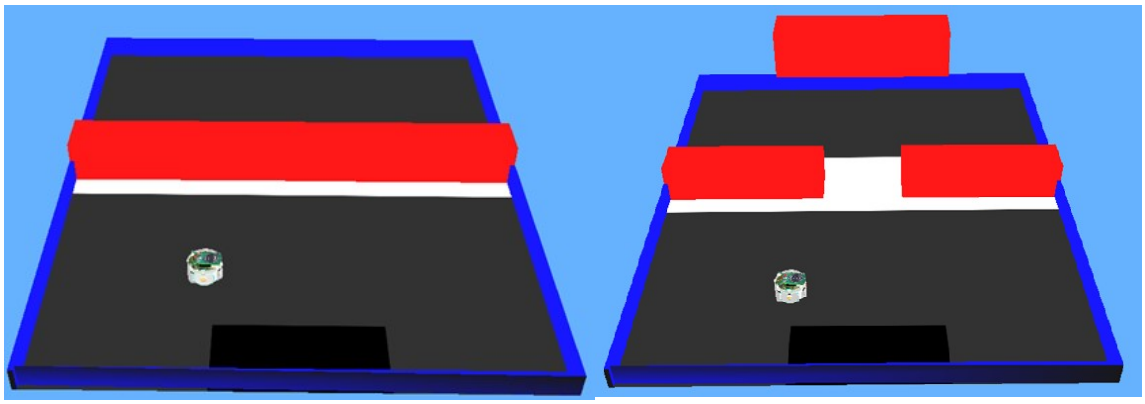


Figura 21. Medio ambiente virtual para el experimento cambio de cuarto. Un robot debe visitar la zona negra para poder abrir la puerta al final del cuarto. Posteriormente debe encontrar la entrada del pasillo y concretar el cambio de cuarto.

Como en el caso del experimento limpieza de entorno, se utilizó un robot e-puck en un ambiente del simulador Webots. Se utilizaron los sensores infrarrojos, sensores de piso, las componentes roja y azul de la cámara VGA del robot. Como actuadores, se utilizaron las velocidades de los dos motores del robot.

3.3.4 Topologías.

Se pusieron a prueba tres tipos de control basado en redes neuronales artificiales (ver Tabla 3): arquitectura no modular, *arquitectura modular emergente* y *arquitectura acetilmodulada*. La primera es una sola red neuronal artificial (ver Figura 22) que reguló las acciones apertura de la puerta y cambio de cuarto del robot. Dicha red constaba de 35 neuronas con la siguiente distribución: 22 neuronas en la capa de entrada, 11 en la capa media y dos en la capa de salida. Se utilizó la arquitectura de red *FeedForward* y la función de transferencia del tipo escalón unitario. Estas últimas representaron variables controladas del experimento.

La capa de entrada del sistema de control *monored* estaba formada por ocho sensores binarios infrarrojos, tres sensores binarios de piso especializados en la detección de los colores blanco, negro y gris. También tres sensores binarios de estado de la tarea, que cambiaban a nivel alto si el robot abría la puerta, encontraba la entrada o encontraba el final del mismo respectivamente. Se usaron cuatro sensores por cada componente de la cámara VGA (rojo y azul), que dividían la imagen en cuatro segmentos de 72 pixeles cada uno. Las neuronas de la capa de salida

controlaban la velocidad de giro de los motores del robot. Para ello, se multiplicó el valor de salida de cada neurona por una constante. Las velocidades negativas no formaban parte de la imagen de dicha función, pero si el valor 0.0.

La *arquitectura modular emergente* tenía de 41 neuronas distribuidas de la siguiente manera: 22 neuronas en la capa de entrada, 11 en la capa de media y ocho en la capa de salida (ver Figura 23). Es importante recordar que dicha arquitectura es una representación de la modularidad interna de las regiones cerebrales. El aparato sensorial de esta red fue similar al de la *monored*, constituida por: ocho sensores infrarrojos, tres sensores de piso, tres sensores de estado de la tarea, cuatro sensores de la componente roja y cuatro de la azul de la cámara. Se utilizó el enfoque proximal para la modularización. Por ello existían dos módulos que competían por el acceso al motor derecho, y otros dos para acceder al motor izquierdo. Una de las neuronas codificaba la velocidad del motor, mientras que la otra se usaba como neurona de activación cuyo nivel determina el acceso al actuador.

Tabla 3. Características de las redes comparadas en el experimento cambio de cuarto.

Característica	***Acetilmodulada***	***Monored***	***Arquitectura Modular Emergente***
Neuronas	Reguladora: 42, Módulos: 35	35	41
Pesos	Reguladora: 366, Módulos: 286	286	286
Entradas	Reguladora: 25(8 infrarrojos, 3 piso, 3 estado abre puerta, pasillo y meta, 4 R y 4 B, 3 estado módulos), Módulos: 22 (8 infrarrojos, 3 piso, 3 estado, 4 R y 4 B)	22 (8 infrarrojos, 3 piso, 3 estado abre puerta, pasillo y meta, 4 R y 4 B)	22 (8 infrarrojos, 3 piso, 3 estado abre puerta, pasillo y meta, 4 R y 4 B)
Capa media	11	11	11
Salidas	Reguladora: 6, Módulos: 2	2	8
Funciones de transferencia	*Sigmoidal*	*Sigmoidal*	*Sigmoidal*
Arquitectura	*FeedForward*	*FeedForward*	*FeedForward*

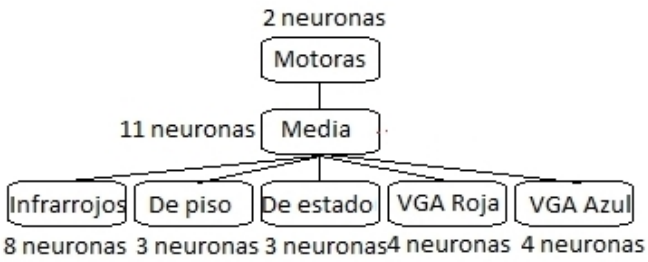


Figura 22. *Monored* para el control de robots dentro del experimento cambio de cuarto. La red está constituida por ocho sensores infrarrojos, tres de piso, uno de basura, tres sensores binarios que brindan información sobre el estado de la tarea, cuatro segmentos de la cámara para la componente Roja y cuatro para la componente azul, once neuronas en la capa media, y dos neuronas para controlar los motores.

Como fue el caso del experimento anterior con la *arquitectura modular emergente*, se agruparon dos neuronas para formar cada módulo. Se pusieron dos módulos a competir por cada recurso del robot, por ello dos de los módulos competían por el acceso al control del motor derecho y dos módulos para el motor izquierdo. En cada módulo existía una neurona de salida que podía controlar la velocidad del motor del robot, y una neurona de activación que le permitía al robot competir por el acceso a recursos. En cada pareja de módulos, se verificaba el mayor valor de la neurona de activación para que se encargara del control de la velocidad del motor correspondiente en cada *step* de prueba.

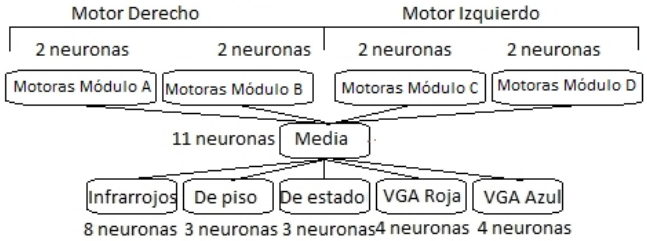


Figura 23. Red de módulos internos para el control de robots dentro del experimento cambio de cuarto. Una sola red neurona artificial dividida en tres módulos internos modifica el comportamiento de los robots evolutivos. La red está constituida por ocho sensores infrarrojos, tres de piso, tres que codifican el estado de la tarea, cuatro segmentos de la cámara para la componente roja y cuatro para la componente azul, once neuronas en la capa media, dos neuronas para controlar los motores y una neurona selectora en cada uno de los tres módulos. El control de los actuadores se le otorga al módulo con el valor de salida en la neurona de selección más alto.

Finalmente, para la *arquitectura acetilmodulada*, se emplearon tres redes neuronales como módulos y una red de modulación (ver Figura 24). Los tres módulos correspondían a una de las fracciones o componentes de la función de evaluación. Por ello, el primer módulo se encargó de controlar al robot para que abriera la puerta, el segundo para encontrar la entrada del pasillo, y el tercero para cruzar el mismo y cambiar del cuarto. Las redes modulares estaban integradas por 35 neuronas: 22 en la capa de entrada, 11 en la capa media y dos en la capa de salida. El aparato sensorial estaba integrado por ocho sensores infrarrojos, tres sensores de piso para la detección de los colores negro, blanco y gris, tres sensores binarios de estado virtual de la tarea (abre

puerta, encuentra pasillo y llega a la meta), cuatro neuronas de la componente roja y cuatro neuronas de la componente azul de la cámara VGA.

En el caso de la red reguladora de acetilcolina artificial estaba integrada por 42 neuronas: 25 en la capa de entrada, 11 en la capa media y seis en la capa de salida. Los sensores utilizados fueron los ocho sensores infrarrojos, tres sensores de piso, tres sensores de estado de la tarea, cuatro neuronas asociadas a segmentos de la cámara para la componente roja, y cuatro para la componente azul. Adicionalmente se crearon tres neuronas que recibían información binaria sobre el estado de activación de los módulos en el *step* anterior.

Se formaron grupos de dos neuronas en la capa de salida. Cada grupo representaba a los tres módulos presentes en la topología. Los grupos de neuronas de salida estaban compuestos por una neurona que indicaba la presencia o no de la sustancia artificial que se transporta a las neuronas de la capa intermedia de los módulos y que permitía que las neuronas de la capa de salida de los módulos hicieran sinapsis bajo el esquema químico. La segunda neurona permitió conocer la cantidad de sustancia artificial que viaja a los módulos. En caso de existir activación en varios módulos, determinaba cuál de ellos efectuaba el proceso de sinapsis química artificial. Así, el primer criterio para otorgar el control de los actuadores del robot a los diferentes módulos fue la presencia de la sustancia virtual. En caso de que el valor de salida de las dos neuronas binarias fuera uno, entonces se tomaba en cuenta el valor de intensidad de la sustancia virtual. De esta manera se concedía el control al módulo con mayor intensidad en las neuronas de salida, permitiendo la sinapsis en sus neuronas efectoras.

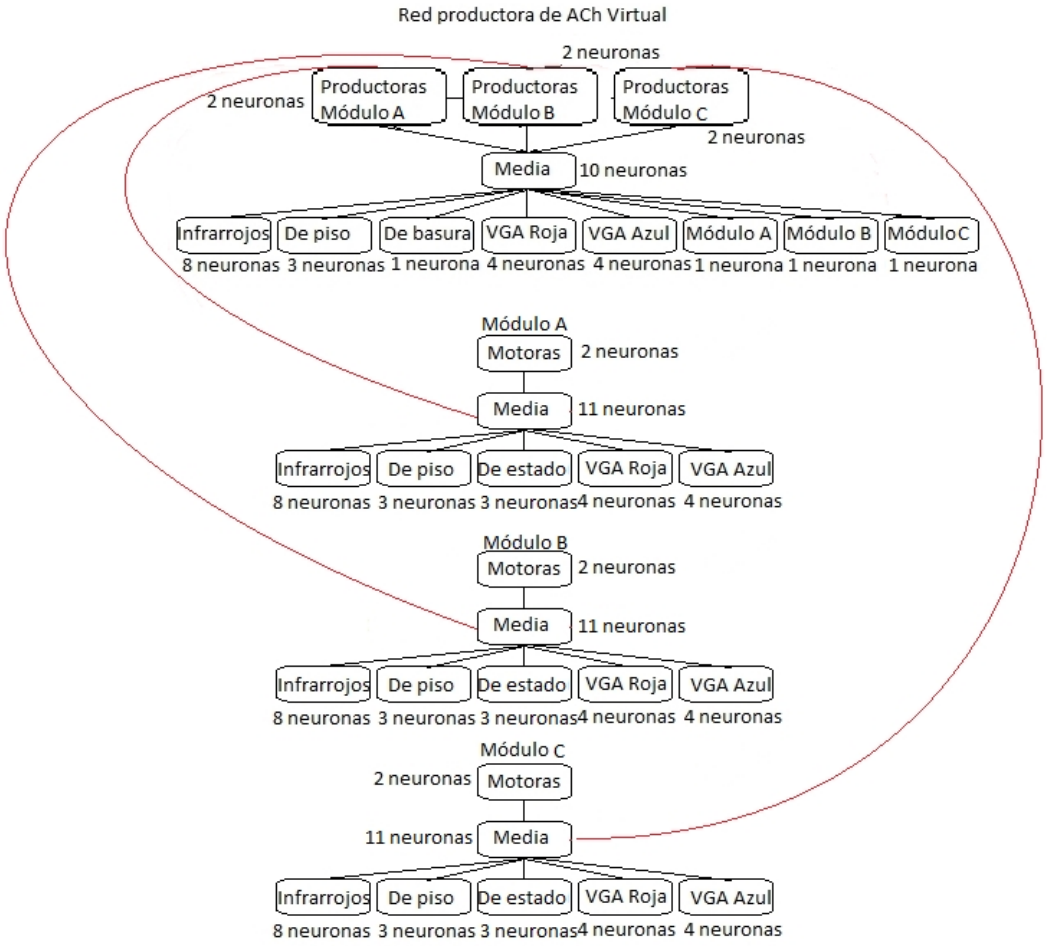


Figura 24. Red *acetilmodulada* para el control de robots dentro del experimento cambio de cuarto. Tres módulos compiten por el control de los actuadores del robot. El control se otorga a aquel módulo que reciba vía acetilcolinergica el neurotransmisor virtual a las inter-neuronas (neuronas de capa media) correspondientes. La red productora de ACh virtual determina cuál de los módulos tendrá el control de los actuadores a partir de la información del entorno y de los módulos. Al mismo tiempo inhibe la sinapsis en el módulo que no participará en la solución en un instante de tiempo.

3.3.5 Proceso evolutivo.

El proceso evolutivo fue utilizado para optimizar los pesos de las diferentes topologías de redes neuronales. El algoritmo evolutivo utilizado fue un algoritmo genético tipo generacional como el utilizado en el experimento limpieza de entorno. En las tres topologías puestas a prueba, los parámetros del proceso evolutivo fueron: 20 individuos, mutación del 2%, elitismo del mejor individuo, punto de cruza aleatorio, selección por torneo y sustitución en donde se podía conservar hasta el 20% de los padres puestos a prueba en la generación previa. Las primeras dos topologías tuvieron un proceso evolutivo compuesto por 100 generaciones. La tercera topología tuvo un proceso evolutivo diferente. Primero se optimizaron los módulos de manera independiente. Cada proceso independiente constó de 100 generaciones. Con los valores de los módulos, se optimizó la red reguladora con 100 generaciones más.

Los individuos de las tres topologías fueron probados en un entorno virtual dentro del simulador Webots, con seis *trials* de 800 *steps* cada uno. La posición de los individuos al inicio de cada trial se determinaba de manera aleatoria, siempre dentro del cuarto con la zona para abrir la puerta. La función de calidad otorgaba tres puntos a los individuos que abrían la puerta, 11.0 a los que encontraban la entrada del pasillo, y 29.0 a los que cambiaban de cuarto. De esta manera, los individuos que cumplían con la tarea obtenían un *fitness* máximo de 43.0 por trial. Los individuos que sólo abrían la puerta y encontraban la entrada del pasillo recibían un *fitness* de 14.0 por trial. Y finalmente, los que sólo abrían la puerta recibían 3.0 puntos de *fitness* por trial.

3.3.6 Diseño experimental.

Se utilizaron tres grupos que representan a cada topología a comparar. Los nombres de los tres grupos fueron: *monored* que representaba a la arquitectura no modular, *AME* que corresponde a la *arquitectura modular emergente* y que representaba a la modularidad interna, y *acetilmodulada* que representaba a la arquitectura propuesta de módulos externos. De esta manera, el diseño experimental estaba basado en un grupo control y dos tratamientos diferentes que buscan solucionar el bloqueo en espacios de búsqueda. Para los tres grupos experimentales se utilizó la medición post-prueba. Con el diseño experimental se mantuvo la validez interna del experimento.

Cada grupo experimental estaba formado por 18 repeticiones del proceso evolutivo para cada topología (ver Figura 25). Se utilizó al mejor individuo de cada repetición para obtener el valor de la variable dependiente. Dicha variable fue el puntaje de aptitud en una prueba post-evolución. Para obtenerlo se probó al mejor individuo de cada repetición en seis *trials* con posición inicial aleatoria, de 1000 *steps* cada uno. El valor de aptitud de los seis *trials* se promedió y así se obtuvo el valor de aptitud post-evolución de cada individuo. Este procedimiento se repitió para cada grupo experimental.

La variable independiente estuvo representada por las diferentes topologías puestas a prueba, y la variable dependiente el puntaje de aptitud de la prueba post-evolución. Así la predicción de este experimento contempló que al aumentar la cantidad de módulos la topología basada en la acetilmodulación tiene un mejor rendimiento que las demás. Aumentar la cantidad de módulos trae consigo un aumento en la dimensionalidad del espacio de soluciones y vuelve potencialmente más difícil encontrar óptimos a través del proceso de búsqueda.

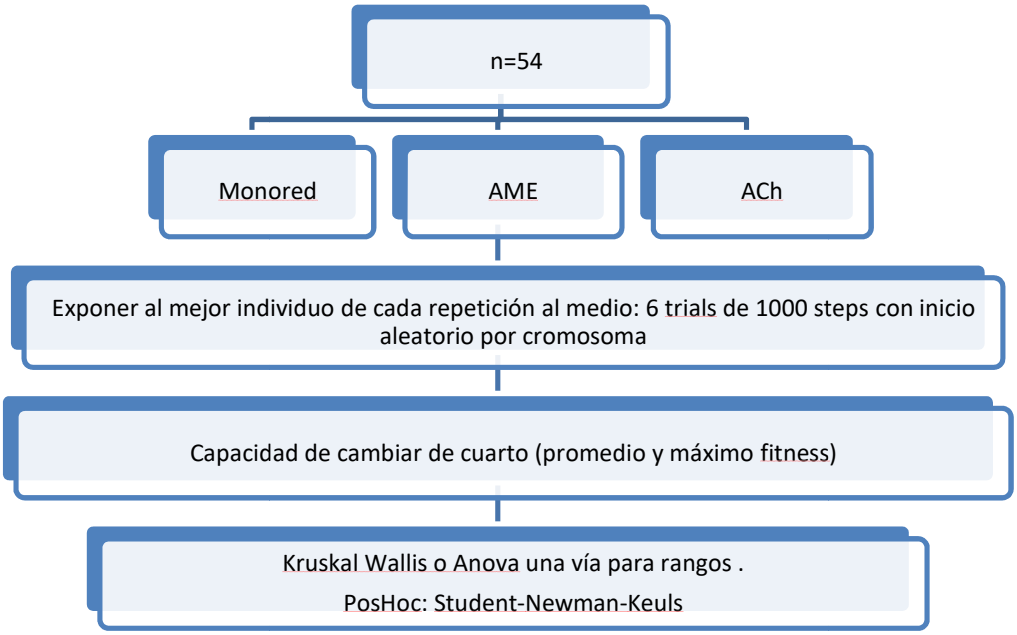


Figura 25. Diseño experimental (experimento cambio de cuarto). Se comparan tres arquitecturas de control para robots evolutivos basadas en diferentes configuraciones de redes neuronales. La variable a medir es el puntaje de aptitud en una prueba independiente al proceso evolutivo.

Para comprobar la predicción del experimento, se utilizó una prueba estadística de Kruskal Wallis o ANOVA de una vía para rangos para buscar diferencias estadísticas, y una prueba post hoc Student -Newmann-Keuls para la comparación entre grupos dada la diferencia estadística. Ambos métodos estadísticos utilizaron un valor de significancia $p<0.05$.

El diseño experimental fue seleccionado debido a la cantidad de variables utilizadas: una variable dependiente y una independiente. Esto permitió conocer las diferencias estadísticas entre grupos experimentales y con ello cuál de las topologías tiene un mejor rendimiento. El tamaño de la muestra está restringido al tiempo de procesamiento de los procesos evolutivos en los ambientes simulados.

3.4 Experimento 4: Modularidad y comunicación.

3.4.1 Problemática.

La modularidad es una herramienta utilizada para solucionar bloqueos en espacios de búsqueda muy complejos. Dichos espacios son el producto de una tarea compleja, que al aplicar módulos se descompone en sub-problemas que tienen una complejidad menor al problema original. Pero existen otras técnicas que podrían mejorar la solución provista por la modularidad. Uno de esos casos es el de los sistemas de comunicación mediante la adición de nuevas herramientas de coordinación. Estos últimos no se pueden producir sin la existencia de una población de robots. Esta solución brinda capacidad a los robots para intercambiar información, y solucionar el problema en equipo. Así la complejidad del espacio de búsqueda influenciada por la complejidad

del entorno y la tarea, también puede estar ligada al nivel de cooperación entre individuos para su resolución (Bernard, André y Bredeche, 2016). La combinación de herramientas en el área de la RE puede ayudar a solucionar el problema de los espacios de búsqueda que resultan difíciles de explorar para lograr sistemas de control lo suficientemente robustos para cumplir con una tarea o un comportamiento. Así, la mezcla de las herramientas modularidad y comunicación pueden mejorar el proceso evolutivo de los robots dada una tarea en particular. Por lo cual es importante estudiar, desde el punto de vista estadístico, si existen diferencias significativas por la aplicación de las dos herramientas y si existe un factor de interacción entre ellas.

3.4.2 Objetivo del experimento.

Por ello se creó este experimento para permitir comparar la manipulación de las variables y sus efectos en el proceso evolutivo artificial. Ambas variables fueron del tipo binario, por lo cual podían estar presentes o no en los cuatro grupos experimentales configurados.

3.4.3 Entorno, tarea, y robot.

Para el desarrollo del cuarto experimento se utilizó una versión de la tarea limpieza de entorno mediante un grupo de tres robots. Con esto se buscó generar una tarea en la cual se pudiera utilizar el intercambio de datos sobre la solución de la tarea y sobre situaciones particulares de los individuos que forman los grupos. Tal y como se utilizó en el experimento limpieza de entorno, el medio ambiente virtual consistió en una arena cuadrada delimitada por cuatro paredes azules. Dentro de la arena, de color gris, se encontraban dos targets circulares blanco y negro. El target blanco representaba una zona de recolección de basura virtual (ver Figura 26). Por su parte, el target negro representaba la zona de depósito de basura. Los targets de recolección y depósito de basura contenían un cilindro verde adicional en el centro de cada una. La función del cilindro es servir como un apoyo visual adicional para la localización de los targets.

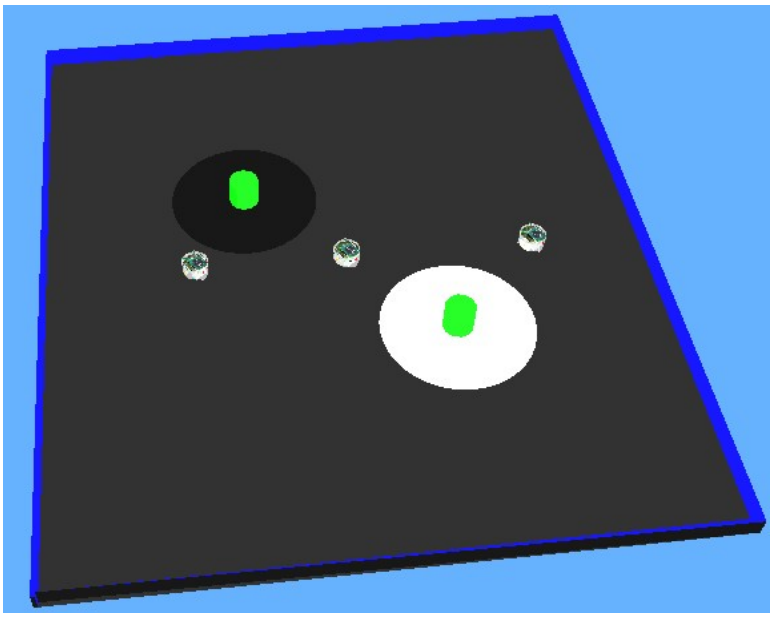


Figura 26. Medio ambiente virtual para el experimento cambio de cuarto. Un grupo de cuatro robots deben visitar la zona circular blanca, en donde por cada *step* recoge una unidad de basura virtual. En la zona de depósito (target negra) los robots depositan una unidad de basura por cada *step* en ella.

Cada *step* que los robots permanecían en una zona de recolección representaba una unidad de basura que se adicionaba a su carga. En el caso de la zona de depósito, cada *step* que los robots permanecían, la carga disminuía una unidad. Se utilizaron robots e-puck en un ambiente del simulador Webots. Se utilizaron los sensores infrarrojos, sensores de piso, sensor virtual de basura, las componentes: roja, verde y azul de la cámara VGA del robot para detectar zonas de depósito y recolección de basura y delimitar la cercanía de las paredes, y la componente roja como medio de recepción de la comunicación. Como actuadores, se utilizaron las velocidades de los dos motores del robot y el anillo de LEDs con el color rojo y negro. Tanto la componente roja de la cámara, como el anillo de LEDs fueron los medios para establecer la comunicación entre robots.

3.4.4 Topologías.

Para este experimento se configuraron cuatro topologías de red distintas (ver Tabla 4), que tenían la función de controlar a los individuos en las cuatro variantes del experimento. El primer grupo representaba una arquitectura no modular sin comunicación que fue el grupo control del experimento. La RNA estaba constituida por 37 Neuronas: 20 en la capa de entrada, 15 en la capa media, dos en la capa de salida. Se usó la arquitectura *feedforward* y la función de transferencia de *sigmoidal* (ver Figura 27).

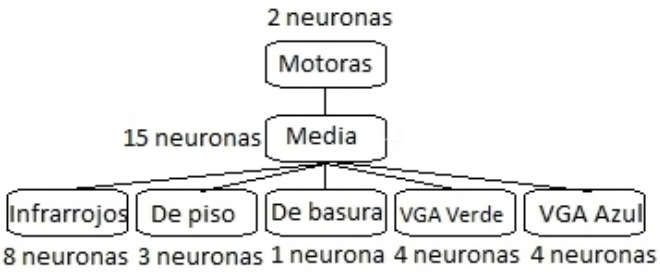


Figura 27. Red neuronal para el control de robots para el grupo control dentro del experimento modularidad y comunicación. Una red neurona artificial sin modularidad y sin comunicación. La red está constituida por ocho sensores infrarrojos, tres de piso, uno de basura, cuatro segmentos de la cámara para la componente verde y cuatro para la componente azul, quince neuronas en la capa media, y dos neuronas para controlar los motores.

Se utilizaron los siguientes sensores: ocho infrarrojos binarios, tres sensores binarios de piso para detectar los colores negro, blanco y gris, un sensor con la cantidad de basura que porta el robot, cuatro de la componente verde y cuatro de la azul de la cámara en donde cada una representa un sector de 72 pixeles de la imagen adquirida por la cámara VGA. En la capa de salida cuenta con dos neuronas para codificar el valor de la velocidad de cada uno de los motores.

El segundo grupo representaba la arquitectura no modular con comunicación, la red neuronal estaba constituida por 42 neuronas, de las cuales 24 estaban en la capa de entrada, 15 en la capa media y tres en la capa de salida (ver Figura 28). Como en el caso anterior, se usó la arquitectura *feedforward* y la función de transferencia de *sigmoide*. Se utilizaron los siguientes sensores: ocho infrarrojos binarios, tres sensores binarios de piso para detectar los colores negro, blanco y gris, un sensor con la cantidad de basura que porta el robot, cuatro neuronas de la componente roja, cuatro de la componente verde y cuatro de la azul de la cámara en donde cada una representa un sector de 72 pixeles de la imagen adquirida por la cámara VGA. En la capa de salida cuenta con tres neuronas, en donde dos salidas funcionaron para codificar el valor de la velocidad de cada uno de los motores, y una neurona para emitir alguna señal a través del anillo de LEDs.

El tercer grupo representaba un modelo modular sin comunicación, conformado por dos módulos y una red reguladora (ver Figura 29). Uno de los módulos se ocupaba de la resolución de la tarea de recolección, mientras que el otro se encargó del depósito de basura. Cada módulo estaba constituido por 37 neuronas: 20 en la capa de entrada, 15 en la capa de media y dos en la capa de salida. El aparato sensorial contaba con ocho sensores infrarrojos, tres sensores de piso especializados en la detección de los colores negro, blanco y gris, un sensor con el nivel de basura, cuatro neuronas asociadas a cuatro segmentos de 72 pixeles del componente verde, y cuatro de la componente azul de la cámara VGA. En la capa de salida, los módulos tenían dos neuronas asociadas con la velocidad de los motores de cada robot.

La red moduladora estaba formada por 41 neuronas: 22 en la capa de entrada, 15 en la capa media y cuatro en la capa de salida. En la capa de entrada se encontraban ocho sensores infrarrojos, tres sensores de piso para detectar los colores negro, blanco y gris, sensor de nivel de

basura, cuatro sensores por cada componente de la cámara VGA (GB), que dividían la imagen en sectores de 72 pixeles, y un sensor por cada módulo que servía para conocer el estado de activación de los mismos en el tiempo anterior (dos). La capa de salida tenía cuatro neuronas, que conformaban dos bloques de dos unidades cada una. La misión de cada bloque era producir niveles de un modulador artificial para que fuera enviado a un módulo, se produjera sinapsis y así tomará el control de los actuadores del robot. Una de las neuronas de cada bloque indicaba si se producía o no el modulador. La otra servía para indicar la intensidad del mismo. En caso que ambos bloques produjeran la sustancia virtual, la neurona con mayor intensidad determinaba el módulo que tomaba el control de los actuadores.

Tabla 4. Características de las redes comparadas en el experimento modularidad y comunicación.

Característica	Sin modularidad sin comunicación	Sin modularidad con comunicación	Con modularidad sin comunicación	Con modularidad con comunicación
Neuronas	37	42	Reguladora: 41, módulos: 37	Reguladora: 45, módulos: 42
Pesos	350	429	Reguladora: 412, módulos: 350	Reguladora: 429, módulos: 37
Entradas	20 (8 infrarrojos, 3 piso, 1 basura, 4 G y 4 B)	24 (8 infrarrojos, 3 piso, 1 basura, 4 R, 4 G y 4 B)	Reguladora: 22 (8 infrarrojos, 3 piso, 1 basura, 4 G y 4 B, 2 estado módulos), Módulos: 20 (8 infrarrojos, 3 piso, 1 basura, 4 G y 4 B)	Reguladora: 26 (8 infrarrojos, 3 piso, 1 basura, 4 R, 4 G y 4 B, 2 estado módulos), Módulos: 24 (8 infrarrojos, 3 piso, 1 basura, 4 R, 4 G y 4 B)
Capa media	15	15	15	15
Salidas	2	3	Reguladora: 4, módulos: 2	Reguladora: 4, módulos: 3
Funciones de transferencia	*Sigmoidal*	*Sigmoidal*	*Sigmoidal*	*Sigmoidal*
Arquitectura	*FeedForward*	*FeedForward*	*FeedForward*	*FeedForward*

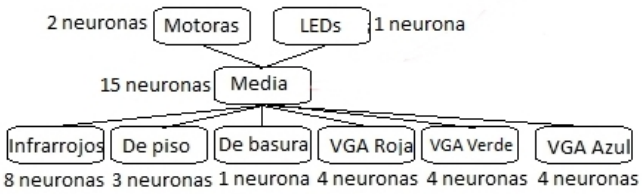


Figura 28. Red neuronal para el control de robots para el segundo grupo dentro del experimento modularidad y comunicación. Una red neurona artificial sin modularidad y con comunicación. La red está constituida por ocho sensores infrarrojos, tres de piso, uno de basura, cuatro segmentos de la cámara para la componente roja, cuatro para la componente verde y cuatro para la componente azul, quince neuronas en la capa media, dos neuronas para controlar los motores y una para controlar el anillo de LEDs rojos.

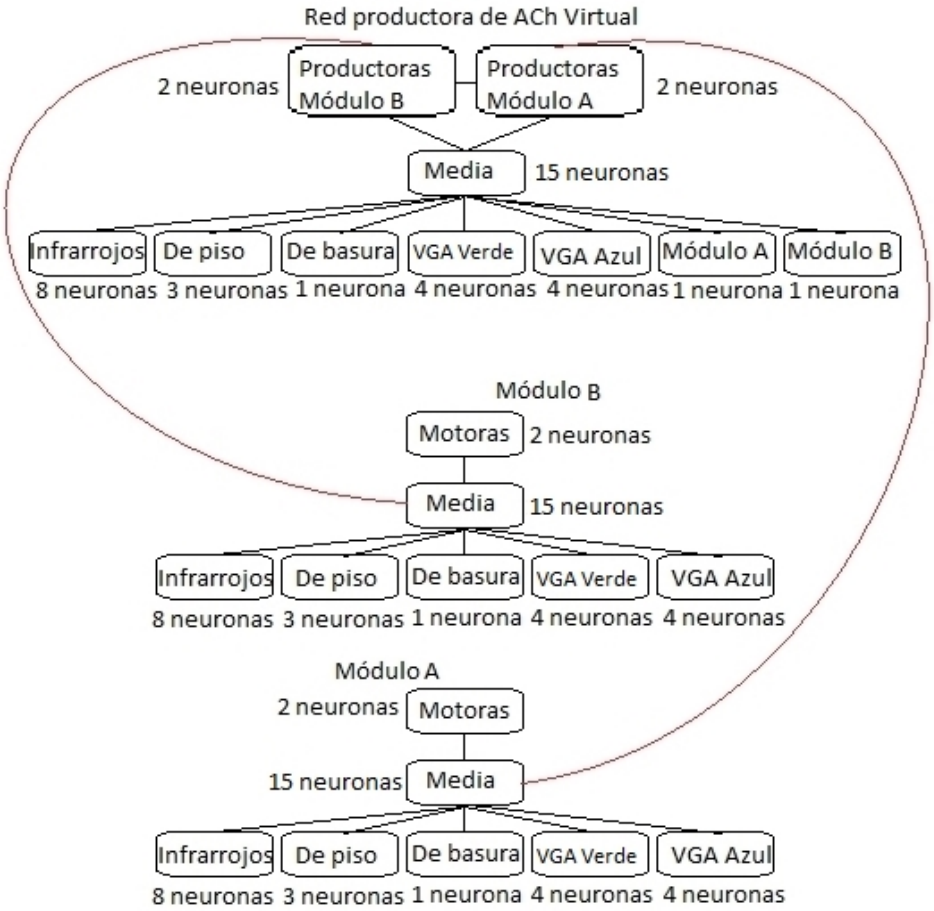


Figura 29. Control de robots para el tercer grupo dentro del experimento modularidad y comunicación. Un par de módulos compiten por el control de los actuadores del robot. El grupo de robots no cuenta con un sistema de comunicación. La red productora determina, mediante niveles de acetilcolina virtual, cuál de los dos módulos tendrá acceso a los actuadores del robot. Uno de los módulos (A) se especializa en recolección, mientras que el otro (B) en el depósito de basura virtual.

El cuarto grupo representó a una arquitectura modular y con habilidades comunicativas (ver Figura 30). De esta manera, el sistema de control de cada robot estaba integrado por dos módulos y una red reguladora. La constitución de los módulos era exactamente la misma que en el grupo anterior, la variación era la inclusión de cuatro neuronas en la capa de entrada para la componente roja de la cámara y una neurona en la capa de salida para el control del apagado o encendido del anillo de LEDs. La red moduladora estaba constituida en la misma forma que en el grupo anterior, pero también incluía las neuronas de receptoras de mensaje en la capa de entrada (componente roja cámara VGA). Con estos ajustes los módulos estaban integrados por 42 neuronas: 24 en la capa de entrada, 15 en la capa media y tres en la capa de salida. La red reguladora estaba integrada por 45 neuronas: 26 en la capa de entrada, 15 en la capa media y cuatro en la capa de salida.

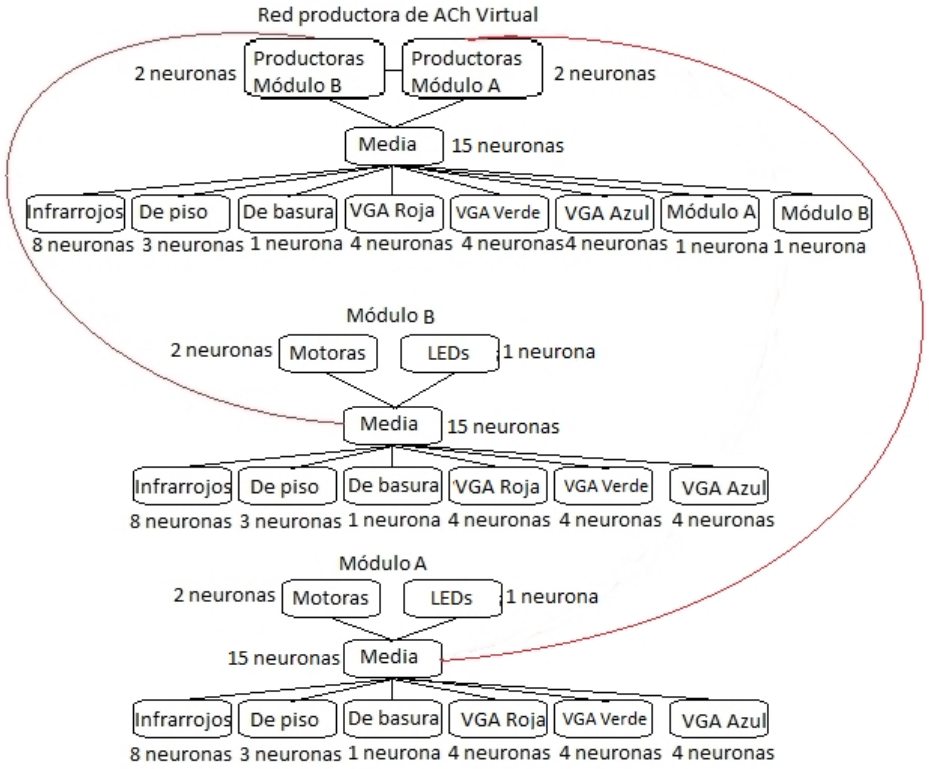


Figura 30. Control de robots para el tercer grupo dentro del experimento modularidad y comunicación. Un par de módulos compiten por el control de los actuadores del robot. El grupo de robots está equipado con un sistema de comunicación. La red productora determina, mediante niveles de acetilcolina virtual, cuál de los dos módulos tendrá acceso a los actuadores del robot. Uno de los módulos (A) se especializa en recolección, mientras que el otro (B) en el depósito de basura virtual.

3.4.5 Proceso evolutivo.

Se utilizó un proceso evolutivo para optimizar los pesos de las diferentes topologías de redes neuronales que componen el diseño experimental. El algoritmo evolutivo utilizado fue un algoritmo genético tipo generacional como el utilizado en los experimentos limpieza de entorno y

cambio de cuarto. En las cuatro configuraciones de neuro-controladores puestas a prueba, los parámetros del proceso evolutivo fueron: 20 individuos, mutación del 2%, elitismo del mejor individuo, punto de cruza aleatorio, selección por torneo y sustitución en donde se podía conservar hasta el 20% de los padres puestos a prueba en la generación previa. La codificación fue binaria de los pesos a partir de nueve bits y valores positivos entre 0.0 y 1.0.

Todos los controladores experimentales tuvieron procesos evolutivos de 500 generaciones. Para las topologías modulares, se creó un proceso previo para optimizar los módulos de manera separada. Dicho proceso duró 100 generaciones. Con los módulos optimizados, se procedió a optimizar el sistema modular, cuyo proceso duró 500 generaciones.

Los procesos evolutivos fueron probados en un entorno virtual dentro del simulador Webots, con seis *trials* de 1200 *steps* cada uno. La posición de los individuos al inicio de cada trial se determinaba de manera aleatoria. La función de calidad otorgaba un punto por cada elemento de basura recolectado y un punto por cada elemento de basura depositado. Los grupos estaban constituidos por tres individuos genéticamente iguales (clones) cuyo genotipo representa a cada individuo dentro del proceso de optimización. El sistema de comunicación fue totalmente emergente y evolutivo, lo cual indica que no existían reglas previas para la emisión o interpretación de las señales.

3.4.6 Diseño experimental.

El experimento utilizó dos variables con dos niveles. La variable comunicación en donde los grupos pueden o no tener la capacidad de comunicarse entre sí (ver Figura 31). Los robots con esta capacidad fueron equipados en su aparato sensorio motriz con un transmisor y un receptor para el intercambio de información mediante el encendido y apagado del anillo de LEDs. El nivel de ausencia de la variable producía que los robots no contaran con estos actuadores y sensores. Por su parte, la variable modularidad podía estar presente o ausente. Cuando estaba ausente, significaba que los sistemas de control utilizaban una sola red neuronal. Ante la presencia de la variable, el sistema de control estaba basado en la *arquitectura acetilmodulada*. Así, el diseño experimental utilizaba un diseño factorial con un grupo de control y la manipulación de dos variables independientes. Para los cuatro grupos experimentales se utilizó la medición post-prueba. El diseño buscaba mantener la validez interna del experimento.

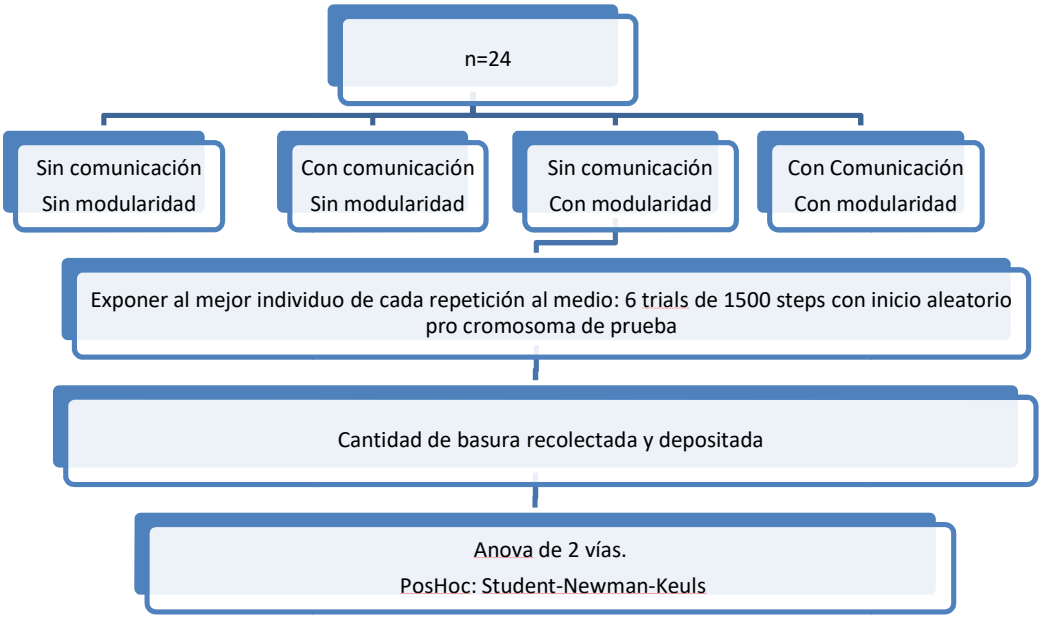


Figura 31. Diseño experimental (experimento modularidad y comunicación). Se compara la acción de dos variables independientes (comunicación y modularidad). La variable a medir es el puntaje de aptitud en una prueba independiente al proceso evolutivo.

Se utilizaron cuatro grupos que representan a las cuatro condiciones a comparar: sin comunicación sin modularidad (SCSM o grupo control), con comunicación sin modularidad (CCSM), sin comunicación con modularidad (SCCM), con comunicación con modularidad (CCCM). Cada grupo experimental estaba formado por seis repeticiones del proceso evolutivo. Se utilizó al mejor individuo de cada repetición para obtener el valor de la variable dependiente en una prueba post-evolución en forma de puntaje de aptitud. Se probó al mejor individuo de cada repetición en seis *trials* con posición inicial aleatoria, de 1500 *steps* cada uno. El valor de aptitud de los seis *trials* se promedió y así se obtuvo el valor de aptitud de cada individuo. Este procedimiento se repitió con cada uno de los grupos experimentales.

La variable dependiente fue el puntaje de aptitud de una prueba post-evolución, ya que ésta refleja la mejoría producto de la manipulación de las variables independientes. Mientras que las variables independientes fueron los sistemas de comunicación y los sistemas modulares en niveles de presencia y ausencia. Como complemento para detectar las señales producidas por los sistemas de comunicación, se utilizó una técnica etológica llamada muestreo tipo *scan* o barrido con un tipo de registro instantáneo durante un lapso de tiempo de dos minutos para el mejor individuo de la última generación para cada una de las repeticiones de los diferentes grupos experimentales que contaran con un sistema de comunicación habilitado. Se consideró un sistema de comunicación estable como aquel que no cambia su estrategia de señalización dentro de las últimas generaciones (Marocco, Cangelosi y Nolfi, 2003).

La predicción de este experimento fue que los sistemas de comunicación por sí solos no pueden resolver el bloqueo en los espacios de búsqueda, pero combinados con modularidad pueden producir sistemas más competentes que los que únicamente son modulares. Para comprobar la predicción del experimento, se utilizó una prueba estadística de ANOVA de dos vías para buscar diferencias estadísticas, y una prueba post hoc Student -Newmann-Keuls para la comparación entre grupos dada la diferencia estadística. Ambos métodos estadísticos utilizaron un valor de significancia $p<0.05$.

El diseño experimental estaba fundamentado en un diseño del tipo factorial. En estos experimentos es posible combinar en los grupos experimentales los efectos de dos factores o variables independientes. Las diferencias significativas indican la influencia que tienen uno sobre otro, y sobre la variable independiente. El tamaño de la muestra seleccionada fue limitado por los tiempos de procesamiento del simulador ante los procesos evolutivos.

Capítulo 4.

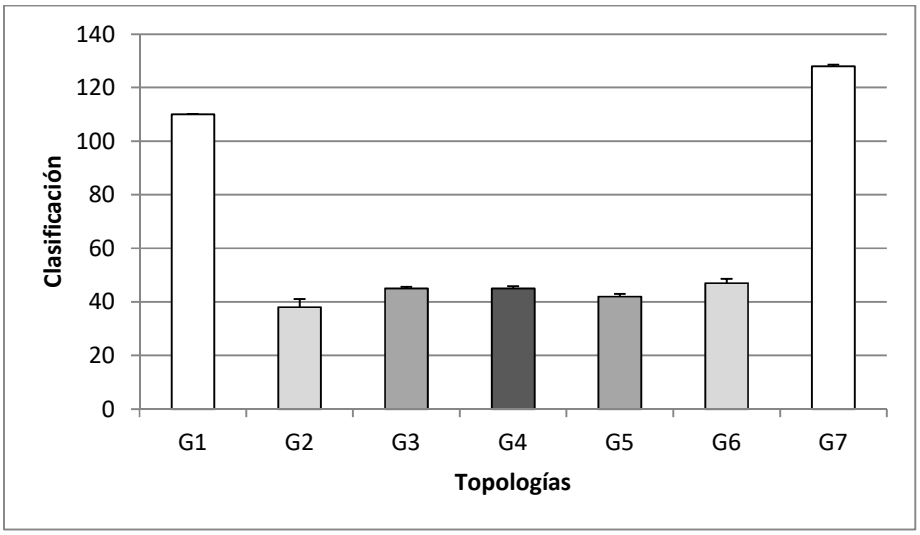
Resultados y discusión.

Esta sección está destinada a la presentación de los resultados obtenidos en cada uno de los experimentos mostrados en el marco metodológico y su discusión. La estructura del capítulo busca presentar la discusión de cada experimento después de presentarse los resultados de cada uno para presentar el contexto completo.

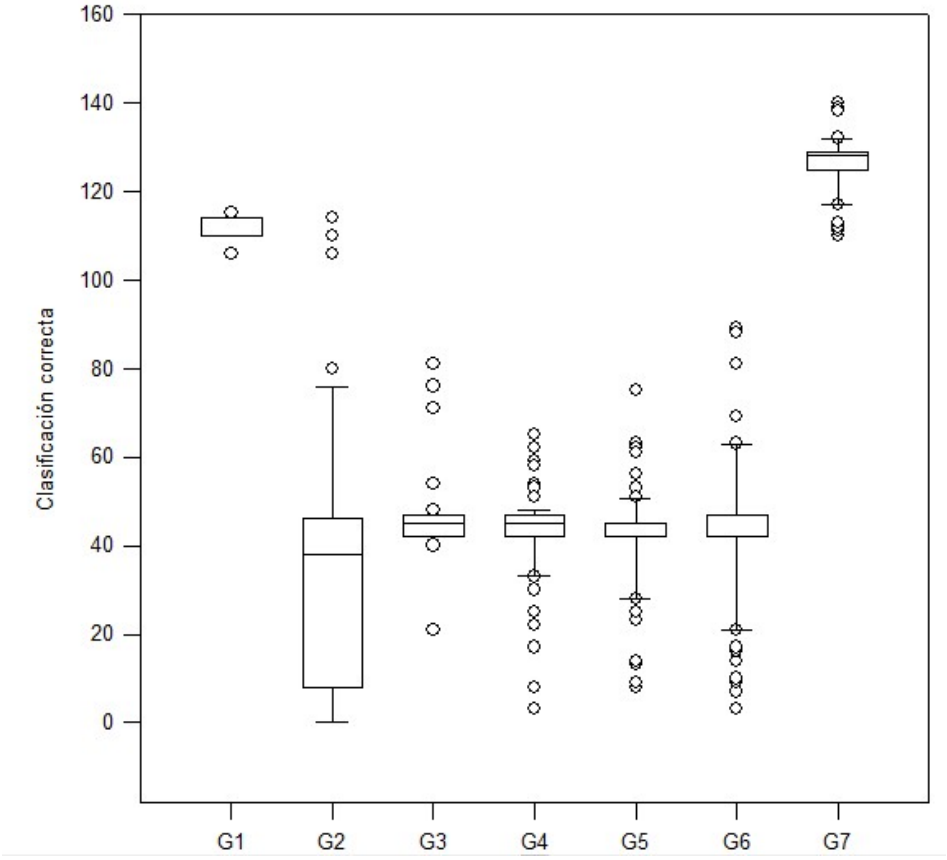
4.1 Resultados experimento 1: estudio exploratorio.

Los resultados de la prueba estadística Kruskal Wallis o ANOVA de una vía para rangos (Shapiro-Wilk $p < 0.05$, no cumple con normalidad) aplicada a la variable dependiente "*clasificación correcta*" (dos niveles: presencia y ausencia) indicaron la presencia de diferencias significativas ($H_{(419.004)}$, $p \leq 0.001$, $6df$) entre los siete grupos puestos a prueba. La prueba post-hoc reveló que las topologías *G1* (110 ± 0.248 ejemplos bien clasificados) y *G7* o acetilmodulada (*126 ± 0.706* ejemplos bien clasificados) son significativamente diferentes ($p<0.05$) entre sí y diferente al resto (*G2 = 38 ± 3.166, G3 = 45 ± 0.703, G4 = 45 ± 0.974, G5 = 42 ± 1.080, G6 = 47 ± 1.696*).

Se muestran los resultados de las medianas de rendimiento de los mejores individuos al final de cada proceso evolutivo en cada grupo experimental (ver Gráficas 1 y 2), de acuerdo a la variable medida. El valor máximo de instancias bien clasificadas esperado fue de *150*, que conforman la totalidad de los ejemplos de la base de datos. El mejor rendimiento corresponde a la *red acetilmodulada* o topología *G7*, seguida de la topología *G1*. Por su parte, el resto de arquitecturas tienen un desempeño bajo en cuanto al número de ejemplos clasificados correctamente (ver Tabla 5).



Gráfica 1. Medianas y error estándar de los grupos del primer experimento (clasificación correcta). El grupo *G7* contó con la mediana más alta y el error estándar más pequeño, seguida de *G1*. Los grupos restantes tienen medianas similares, con error estándar similar.



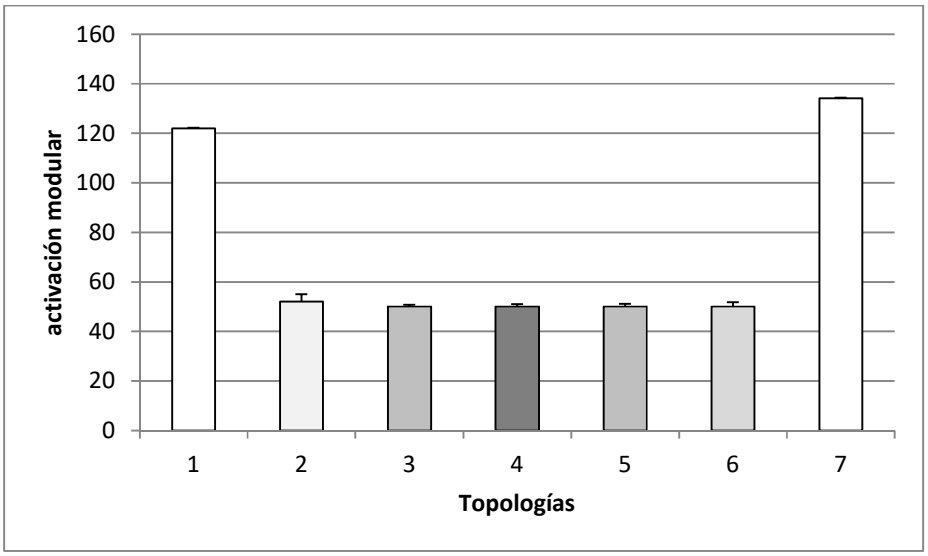
Gráfica 2. Gráfica de cajas de los resultados del experimento exploratorio para la variable clasificación correcta (máximo, mínimo y mediana). La arquitectura *G1* y la *G7* (*acetilmodulada*) son las de mejores resultados en términos de clasificación.

Tabla 5. Kruskal Wallis o ANOVA de una vía para rangos experimento 1 para la variable clasificación correcta.

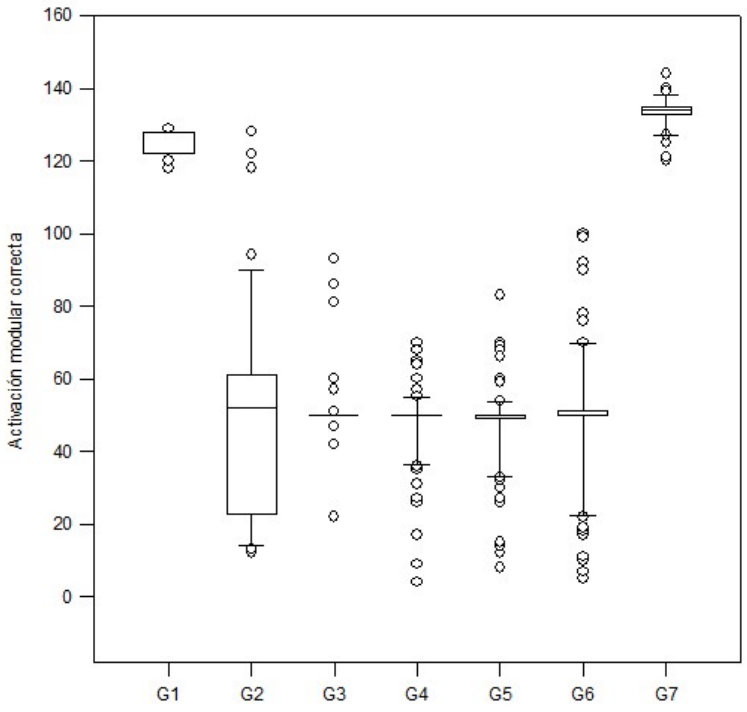
Topología	*G1*	*G2*	*G3*	*G4*	*G5*	*G6*	*G7*
Mediana	*110**	*38*	*45*	*45*	*42*	*47*	*126**
Media + error estándar	*110.978 ± 0.248*	*38.156 ± 3.166*	*45.578 ± 0.703*	*42.878 ± 0.974*	*42.244 ± 1.080*	*45.600 ± 1.696*	*126.4 ± 0.706*
Desviación estándar	*2.351*	*30.034*	*6.665*	*9.236*	*10.245*	*16.089*	*6.699*
				H=419.004, p≤0.001, n=90			
				*SNK comparación entre grupos			

Los resultados de la prueba estadística Kruskal Wallis o ANOVA de una vía para rangos (Shapiro-Wilk $p < 0.05$, no cumple con normalidad) aplicada a la variable dependiente "*activación modular*" mostraron diferencias significativas ($H_{(422.065)}$, $p<0.001$, *6df*) en los siete grupos puestos a prueba. La prueba post-hoc reveló que la primera topología *G1* (*122 ± 0.340*) y la topología *G7* o *acetilmodulada* (*134 ± 0.494*) fueron significativamente diferentes ($p<0.05$) entre sí y al resto (*G2 = 52 ± 3.1117, G3 = 50 ± 0.792, G4 = 50 ± 1.05, G5 = 50 ± 1.186, G6 = 50 ± 1.878*).

Se muestra los resultados de las medianas de rendimiento de los mejores individuos al final de cada proceso evolutivo en cada grupo experimental (ver Gráficas 3 y 4). El valor máximo de módulos activos respecto a su clase fue de *150*, que conforman la totalidad de los ejemplos de la base de datos. El mejor rendimiento corresponde a la red *acetilmodulada*, seguida de la topología *G1*. El resto de arquitecturas tienen un desempeño bajo. Este análisis estadístico indica que la red *acetilmodulada* elige de mejor manera el módulo que pude clasificar una instancia en particular, y que evita el fenómeno que ocurre con los otros grupos en donde un sólo módulo es el que actúa en la totalidad de las instancias (ver Tabla 6).



Gráfica 3. Medianas y error estándar de los grupos del primer experimento (*activación modular*). El grupo *G1* contó con la mediana más alta y el error estándar más pequeño. Los grupos restantes tienen medianas similares, con error estándar similar.



Gráfica 4. Gráfica de cajas de los resultados del experimento exploratorio de la variable activación modular correcta (máximo, mínimo y media). La arquitectura *G1* y la *G7* (*acetilmodulada*) son las de mejores resultados.

Tabla 6. Kruskal Wallis o ANOVA de una vía para rangos experimento 1 para la variable activación modular.

Topología	*G1*	*G2*	*G3*	*G4*	*G5*	*G6*	*G7*
Mediana	*122**	*52*	*50*	*50*	*50*	*50*	*134**
Media + error estándar	*123.662 ± 0.340*	*51.867 ± 3.117*	*51.011 ± 0.792*	*48.156 ± 1.05*	*47.884 ± 1.186*	*50.378 ± 1.878*	*133.556 ± 0.494*
Desviación estándar	*3.228*	*29.568*	*7.513*	*9.959*	*11.247*	*17.813*	*4.691*
			H=422.065, p≤0.001, n=90				
			SNK comparación entre grupos				

4.2 Discusión experimento 1: estudio exploratorio.

Es importante recordar que el dominio de aplicación de este experimento se reubicó a la clasificación de una base de datos debido a la necesidad de conocer el comportamiento de las redes modulares en términos de la monopolización de recursos, así como cuantificar tanto la cantidad de ejemplos clasificados correctamente como la activación de los diferentes módulos.

Los resultados de las variables dependientes "*clasificados correcta*" y "*activación modular*" demuestran que dos redes tienen un rendimiento adecuado en la tarea de clasificación de una base de datos. Así las redes *acetilmodulada* y la topología *G1* tuvieron un desempeño superior al resto de los grupos.

En el caso de la topología *G1*, el buen rendimiento se debe a las dos diferencias principales que tienen con respecto a las topologías *G2* a G6. La presencia de una neurona de activación modular brindó la posibilidad a las redes neuronales de separar las tareas de clasificación y aporte a la solución. Lo que no podría ocurrir con una sola neurona de salida, la cual sería la encargada de decidir la salida correcta y si debe participar o no en la decisión de un conjunto de datos de entrada.

El segundo factor es la presencia de conexiones inhibitorias y excitatorias. Éstas permiten que el umbral de disparo de la neurona pueda operar en un rango mayor. Pero, sobre todo, el trabajo con conexiones inhibitorias (pesos negativos) permite que, dada la activación mayor en un módulo, el resto de las neuronas de decisión tengan un nivel bajo. En otras palabras, esto favorece que las activaciones modulares sean más altas cuando un conjunto de entrada es presentado y el módulo identifica la capacidad de brindar la solución al mismo. Es decir, esta topología tiene implícito el funcionamiento de las neuronas de Renshaw, encargado de la inhibición lateral de módulos.

Un fenómeno importante identificado en los resultados es el de la monopolización de los recursos observable en la variable medible "activación modular correcta". Con dicha variable se cuantificó la cantidad de veces que un módulo interviene en la solución cuando el ejemplo presentado correspondía a la clase para la cual el módulo se especializó.

La monopolización estuvo presente en los grupos de las topologías *G2* a *G6*. En dichas topologías, sólo uno de los módulos fue capaz de tomar el control de la tarea de clasificación en cada una de las *150* instancias puestas a prueba. Las topologías *G1* y *G7* lograron evitar este fenómeno, permitiendo que los diversos módulos tomaran el control de la tarea de clasificación ante los diferentes ejemplos presentados a la red. De esta manera, dichas topologías son una buena solución cuando se requiere un conjunto de varias redes neuronales que se deban coordinar para resolver una o varias tareas. En otras palabras, este par de redes, que son representaciones de la modularidad externa, tienen la capacidad de evitar el principal problema de la modularidad interna: la monopolización de recursos.

Pero el caso destacado en el estudio exploratorio es el desempeño de la red modular *G7* o *acetilmodulada* para ambas variables medibles. En el caso de la topología *G7*, la inclusión de un mecanismo de modulación externo, permite que la cantidad de ejemplos bien clasificados aumente en el caso de dicha red. Pero esto es una consecuencia de la activación correcta de los módulos ante los ejemplos clasificados. Esto quiere decir que, si una red de módulos externos es capaz de activar sus componentes en la situación deseada, el funcionamiento del sistema como conjunto mejora. Por lo que es posible mencionar que una de las grandes claves para la construcción de sistemas de módulos externos, es la capacidad del mismo sistema para elegir cuál de los diferentes módulos debe tomar parte en la solución de un problema presentado.

El modelo *acetilmodulado* fue el resultado de la corriente de bio-inspiración y la transdisciplinaridad en la ciencia. Una característica importante del modelo de la red *acetilmodulada* es la manera en que se conmuta el uso de los diferentes módulos. Los seres vivos tienen estructuras modulares, lo cual permite que los organismos atiendan diferentes procesos a la vez. Por ejemplo, el sistema circulatorio, el aparato digestivo, el sistema nervioso. Entre todos ellos existe un nivel de interacción que permite que los seres vivos puedan realizar diferentes funciones.

En el caso del sistema nervioso central, en particular el de los seres humanos, la interacción entre sus diferentes módulos o componentes se realiza a partir de los neurotransmisores. Es exactamente eso lo que busca reproducir el modelo de la red *acetilmodulada*. Así la interacción en la red propuesta está determinada por una red que no tiene la capacidad de participar de manera directa en la solución del problema, pero que lo hace de manera indirecta a través de la producción de un nivel virtual de una sustancia. Dicha sustancia es utilizada por las neuronas de la capa intermedia para producir sinapsis con las neuronas de la capa de salida de su módulo. Cuando las neuronas de la capa media no reciben niveles de la sustancia artificial, es imposible producir una respuesta de salida en un módulo. Los módulos que no producen respuesta, no son capaces de participar en la solución ante alguna instancia o ejemplo presentado. Para que la

sustancia virtual pueda viajar de la red moduladora productora a los módulos, se utilizó un sistema parecido al de las neuronas de Renshaw en el sistema motriz voluntario. Dichas neuronas una vez excitadas, inhiben mediante conexiones laterales a las neuronas vecinas para generar contracciones y elongaciones de las fibras musculares. De esta manera, las neuronas en el modelo *acetilmodulado* que producen niveles menores del transmisor son inhibidas por la neurona que produce el nivel más alto de la sustancia virtual. Así la regulación entre la interacción o activación de los módulos ocurre como una respuesta de una red neuronal con conexiones inhibitorias.

La sustancia producida por la red moduladora es un equivalente de la acetilcolina. Que en los seres vivos es la sustancia encargada de regular las activaciones de las neuronas efectoras del sistema muscular. Además de tratarse del neurotransmisor primario, que puede regular la producción de otros neuromoduladores en el sistema nervioso. De esta manera, la acción de la modulación de que tienen la inter-neuronas (neuronas de capa media) hacia las neuronas efectoras (neuronas de capa de salida) en cada módulo dentro del modelo *acetilmodulado* está inspirado en su mayoría en la acción de este transmisor. Es decir, el transmisor virtual creado tiene una acción después de su producción que va desde una neurona pre-sináptica (inter-neurona) hacia una neurona post-sináptica (efectora), logrando reproducir en cierta medida el fenómeno de la sinapsis química.

Así la sustancia virtual producida por el modelo propuesto cumple con las características de los neurotransmisores en los seres vivos como son: la producción se realiza por neuronas, su acción va de la neurona pre-sináptica a la post-sináptica, existe un mecanismo de eliminación. En el caso de la acetilcolina, ésta no puede ser inyectada a los seres vivos desde el exterior para comprobar que tenga un efecto similar al producido por la sustancia que se produce de manera natural en el sistema nervioso. La causa es la acción que podría tener sobre el movimiento de los músculos y órganos, pero sobre todo porque la sustancia antagonista (butilcolinesterasa) se encarga de dividir la acetilcolina tan pronto llega al organismo de los seres vivos. Por ello, esa característica queda fuera del modelo virtual que se construye dentro de la *arquitectura acetilmodulada*.

La capacidad de permitir interactuar entre los diferentes módulos de manera adecuada en el modelo *acetilmodulado* depende de la capacidad de selección y discriminación de la red modular productora de acetilcolina artificial. En otras palabras, para la construcción de un modelo de control basado en redes neuronales artificiales arregladas como módulos externos, se debe prever de un sistema de interacción que favorezca la selección entre los diferentes módulos y evitar el fenómeno de monopolización modular. La red modular es un mecanismo que se optimiza con la finalidad de discriminar cuando un módulo debe activarse sobre otros.

Pero contar con un mecanismo de selección no es una condición suficiente, pero sí necesaria para el éxito de un sistema de control de módulos externos. La otra parte depende de la buena optimización de los módulos que puedan ser seleccionados para proporcionar soluciones a los problemas que el agente inteligente puede encontrar en su tiempo de prueba. Así el sistema tendrá los elementos necesarios para su buen funcionamiento con un sistema que puede elegir entre diversos módulos que son optimizados para atender propósitos u objetivos muy concretos. Esto puede equipararse al armado de un rompecabezas, en donde es necesario que las piezas

encajen para lograr el objetivo final. Lo cual puede entenderse mejor empleando el caso del dominio de aplicación de clasificación. Si la topología *acetilmodulada* no hubiese contado con un sistema de interacción entre módulos bien ajustado, el problema de la monopolización podría aparecer. Esto pudo provocar que sólo uno de los módulos estuviera activo en la totalidad de la clasificación. Pero aun contando con un mecanismo de selección, si la optimización de los módulos no fuera la adecuada, simplemente el sistema de selección no tendría opciones adecuadas para operar. Este fenómeno es similar a lo que sucede en los seres vivos, en particular en algunas regiones del cerebro humano, en donde existen regiones que intervienen e interactúan entre sí para resolver problemas específicos. Por ejemplo: manejo de memoria de corto y largo plazo, movimiento ocular, o las regiones sensorio-motrices. Si alguna región de la corteza sensorio-motriz sufre alguna afectación, la capacidad del individuo para resolver determinadas tareas puede perderse o sufrir daños importantes. Aunque en algunos casos es posible que las neuronas cercanas logren un proceso de plasticidad, que es una capacidad de las neuronas sanas para asumir funciones ante algún daño en una región (Carvalho y Nolfi, 2016), las funciones de las regiones dañadas no se cumplen en su totalidad.

Por otro lado, el procedimiento utilizado para redimensionar el espacio de búsqueda original en el experimento exploratorio se basa en el uso de una cantidad total de clases presentes en la base de datos. Es decir, se trata de generar módulos capaces de reconocer las tres clases presentes. En conjunto tienen la capacidad de resolver cualquiera de los ejemplos presentados. El buen funcionamiento depende de la optimización del sistema de interacción modular de cada topología.

Así para el caso de todas las topologías puestas a prueba, el espacio de soluciones potenciales se descompone en tres sub-espacios que corresponden a cada uno de los módulos. También se incluye una reestructura el espacio inicial, cambiando la complejidad de problema. Esta misma pasa de la clasificación entre tres clases posibles a la selección del módulo que mejor puede apoyar la solución a una instancia presentada. Es por ello que se puede afirmar que en el caso de la topología *acetilmodulada*, la reestructura del espacio de solución resulta la mejor solución. El sistema de interacción basado en la acción del neurotransmisor acetilcolina permite que los módulos participen de mejor manera en la solución de las instancias presentadas.

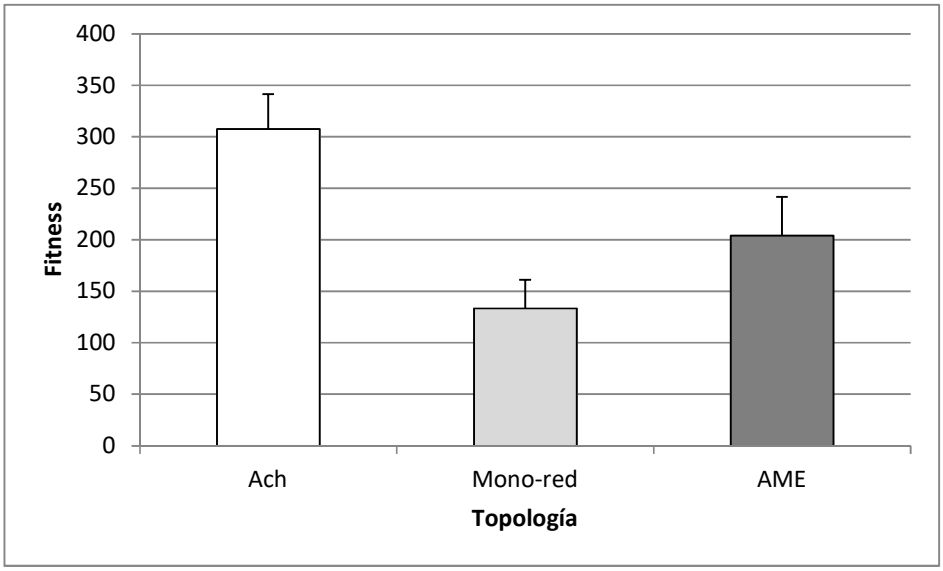
Finalmente, este experimento muestra una propuesta para mejorar el desarrollo metodológico y la construcción de los neuro-controladores en RE, que es uno de los problemas abiertos en el campo de la RE (Doncieux, Bredeche, Mouret y Eiben, 2015), (Trianni, 2014), (Trianni y López-Ibáñez, 2015). Aunque el domino de aplicación no es propiamente dicha área, si tiene la intención de probar diferentes redes neuronales para ser probadas en un dominio de aplicación diferente. Como ya se ha mencionado, la intención del estudio exploratorio fue encontrar un buen modelo redes neuronales arregladas como módulos externos y conocer la naturaleza del funcionamiento de esta representación. Principalmente se buscaba una configuración que permitiera evitar monopolización de la modularidad interna promoviendo la participación de los diversos módulos en la solución de un problema particular. El dominio de clasificación permite estudiar la solución correcta de un sistema ante una instancia dada, lo que en el dominio de robótica evolutiva es

difícil de conocer. De esta manera, este experimento puede considerarse como parte del proceso de diseño de un controlador para un robot evolutivo. Y por lo tanto es un aporte en el intento de brindar de un marco metodológico robusto al área de investigación (König, 2015).

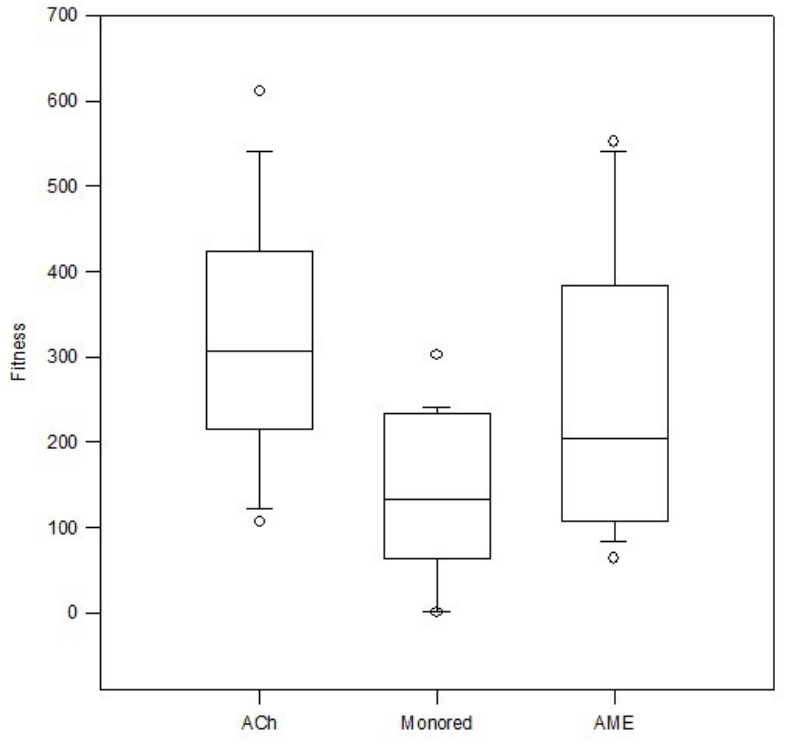
4.3 Resultados experimento 2: Limpieza de entorno (estudio comparativo).

Los resultados de la prueba estadística Kruskal Wallis o ANOVA de una vía para rangos (Shapiro-Wilk $p < 0.05$, no cumple con normalidad) aplicada a la variable dependiente "cantidad de basura recolectada y depositada" de los diferentes procesos evolutivos (ver Gráficas 5 y 6) mostró diferencias significativas entre las medianas de los grupos ($H_{(13.622)}$, $p=0.001$, $2df$). Por su parte la prueba post-hoc reveló que todos los grupos (*Acetilmodulada = 307.66 ± 33.812, Monored = 133.33 ± 21.737, Arquitectura Modular Emergente = 204.00 ± 37.660*) eran diferentes entre sí ($p<0.05$).

El número máximo esperado en la variable medible fue de *700*. Correspondiente a *350* puntos por unidades de basura levantada y *350* por unidades de basura depositadas. El grupo experimental con el mayor puntaje fue el que representaba a la arquitectura *acetilmodulada*. En él, la mayoría de individuos fue capaz de resolver la tarea evolutiva propuesta. En el caso de la *arquitectura modular emergente*, la cantidad de individuos capaces de resolver la tarea es menor que en el caso de la arquitectura *acetilmodulada* y mayor a un *monored* o *control no modular*. Esto último coincide con lo reportado por Nolfi (1997) dentro de su marco experimental, en donde se dice que el rendimiento del *control no modular* es superado por un control modular (ver Tabla 7).



Gráfica 5. Medianas y error estándar de los grupos del segundo experimento. Los grupos que representan arquitecturas modulares obtienen un mejor resultado que la *monored*. Esto es debido a que la modularidad permite evitar los bloqueos o interferencias entre tareas para redes neuronales.

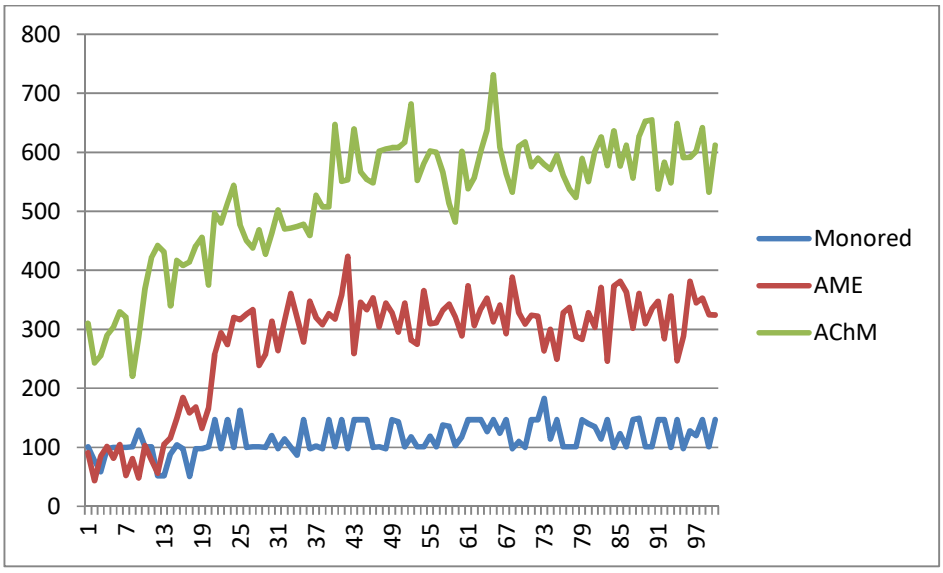


Gráfica 6. Gráfica de cajas de los resultados del experimento 2 (máximo, mínimo y mediana). La arquitectura acetilmodulada tiene un desempeño diferente y superior que el resto de las arquitecturas.

Tabla 7. Kruskal Wallis o ANOVA de una vía para rangos experimento 2 "limpieza de entorno".

	Acetilmodulada	*Monored*	*Arquitectura Modular Emergente*
Mediana	*307.660 ± 33.812**	*133.330 ± 21.737**	*204.00 ± 37.660**
Media	*318.617*	*133.579*	*250.553*
Desviación estándar	*143.453*	*92.223*	*159.777*
		H=13.622, p=0.001, n=18	
		**SNK* comparación entre grupos	

Los procesos evolutivos de la *monored* (ver Gráfica 7) estuvieron marcados por subidas y bajadas de la función de aptitud ante la imposibilidad de ajustar los pesos de la red para el cumplimiento de ambas tareas. En el caso de la *AME*, los pesos lograban estabilizarse alrededor de un valor de *calidad* o *"fitness"* (entre *250* y *400* puntos). El mismo caso ocurrió para los procesos evolutivos de la *arquitectura acetilmodulada*, en donde después de un desempeño inicial bajo el proceso tiende a estabilizarse en un rango de valores del *fitness* (entre *450 y 700* puntos).



Gráfica 7. *Fitness* promedio del proceso evolutivo de una repetición de las tres arquitecturas en el experimento limpieza de entorno. Se aprecia la incapacidad del algoritmo evolutivo para establecer un crecimiento constante de la aptitud de los individuos en el caso de la *monored*. En el caso de la *AME* existe un crecimiento constante en el *fitness* de la población durante las primeras etapas del proceso evolutivo. Finalmente, para el caso de la AChM es posible observar un crecimiento constante en el *fitness* de la población durante las primeras etapas del proceso evolutivo, que resulta mayor que en cualquiera de las otras dos arquitecturas.

4.4 Discusión experimento 2: Limpieza de entorno (estudio comparativo).

Este experimento tenía la intención de poner a prueba la *arquitectura acetilmodulada* en el domino de aplicación de RE, en particular su capacidad para resolver bloqueos en espacios de soluciones. Además de comparar su rendimiento con una *arquitectura de módulos internos* y una *arquitectura no modular*. Lo primero que se debe resaltar es que los resultados muestran que la tarea de limpieza de entorno representa una fuente de estudio para la interferencia neuronal. Lo que conlleva la confirmación de los resultados reportados por Nolfi (1997) y Ziemke *et al.* (2004) sobre las arquitecturas *no modular* (*monored*) y la *modular emergente*. Ellos reportan que la tarea de limpieza de entorno provoca un bloqueo en las arquitecturas no modulares, que provoca sistemas de control capaces de adquirir la habilidad de levantar basura, pero no de depositarla.

La mediana del grupo experimental *monored* ronda los *133* puntos de *fitness*. En la gráfica es posible apreciar que, en el mejor de los casos, este grupo experimental produce individuos capaces de levantar basura y depositar una parte de ella. Pero en el peor de los casos produce individuos que no son capaces de completar su máxima carga de recolección de basura virtual o simplemente incapaces de lograr la primera de las tareas (levantar basura). Por ello, el grupo de la *monored* se puede considerar como un grupo control experimental que permite conocer la naturaleza del problema de limpieza de entorno en RE. El rendimiento puede ser explicado por los espacios complejos de solución. Es probable que las configuraciones óptimas de los pesos (óptimos locales) para que los robots cumplan con las tareas de recolección y depósito de basura

se encuentren distantes en el espacio de búsqueda. Por ello el algoritmo evolutivo sufre de muchas dificultades para obtener buenos resultados (Jacobs *et al.*, 1991).

En el caso de la *red modular emergente*, la estructura de la misma y las presiones evolutivas permiten especializar a los módulos para recoger y medianamente depositar basura. Lo que representa una mejoría respecto a la *monored*. La causa principal de dicha mejoría es la modularización del trabajo. En el mejor de los casos es capaz de obtener individuos capaces de llegar casi a la meta propuesta, pero en el peor produce individuos que pueden recolectar un nivel considerable de basura. Pero en la mayoría de los casos produce individuos que cumplen bien con la tarea de recolección con una mediana apenas por sobre los *200* puntos de *fitness*.

La *arquitectura modular emergente* produce monopolización en los módulos. Esto genera que sólo uno de los módulos sea capaz de brindar solución al problema. Dicha monopolización en la AME es la razón por la cual no consigue mayores niveles de aptitud. La monopolización provoca que sólo uno de los módulos sea tomado en cuenta en la solución del problema. Lo que conlleva en algunos de los casos a un proceso de bloqueo en el espacio de soluciones producto del uso y configuración de las neuronas selectoras. Es decir, la neurona selectora de uno de los módulos siempre tendrá un nivel menor a la neurona del otro módulo producto de la monopolización de recursos. Por lo que para mejorar su funcionamiento es necesario considerar un procedimiento que posterior a la optimización de los pesos correspondientes a los módulos y que únicamente afecte a los pesos de las neuronas selectoras. A partir de ello es posible clasificar al problema de la monopolización de recursos como un sub-problema del bloqueo en los espacios de solución.

Por su parte, la *arquitectura acetilmodulada* tiene los mejores resultados comparada con los otros dos grupos experimentales. La arquitectura propuesta garantiza obtener al menos, individuos con un valor de aptitud mayor a la mediana de los individuos del grupo *AME*. En el mejor de los casos, los robots obtenidos pueden levantar y depositar basura en más de una ocasión. Y en la mayoría de los casos obtiene controladores que le permiten a los robots cumplir de buena manera con ambas tareas.

La modularidad externa, representada por la *arquitectura acetilmodulada*, es la propiedad de los sistemas de los seres vivos, que permite a diferentes órganos contribuir con la solución de una funcionalidad general específica. Es decir, se trata de una representación de bloques de construcción que pueden unirse y generar comportamientos o funciones que pueden ser considerados complejos. La complejidad de los comportamientos tiene una relación importante con la configuración del medio ambiente y la del aparato sensorio-motriz del robot (Bongard, 2015). Este par de aspectos son los encargados de ejercer presiones evolutivas sobre el proceso de búsqueda y definir la dimensionalidad del mismo, respectivamente. En el caso de la búsqueda computacional se busca simplificar los espacios para los algoritmos de ajuste u optimización mediante la especialización de un sistema de control para la resolución de un sub-problema o sub-objetivos. Este enfoque también se basa en el paralelismo del cerebro entre diferentes regiones que cooperan en la solución de una tarea.

Lo que se busca es dividir una tarea o meta general y dividirla entre diferentes redes neuronales. Cada una de ellas representa un espacio de búsqueda independiente, por lo que el ajuste de los componentes de las redes debe realizarse de una manera más amigable para el algoritmo encargado de optimizar el componente; para garantizar que exista un sub-espacio de soluciones en el que el bloqueo presentado desaparezca.

Usando el enfoque de la modularidad externa no es necesario crear operadores especiales dentro del proceso evolutivo. Basta con optimizar las redes de manera separada y crear un sistema de interacción que conjunte el trabajo. Esto ocurre cuando una red de alto nivel selecciona aquel módulo que puede brindar la mejor solución en un tiempo y situación determinados para el sistema entero. La creación de estructuras de control en forma de módulos externos puede representar, además una solución al problema de los bloqueos en los espacios de solución, una oportunidad para generar estructuras de control más complejas que obedecen a la necesidad de contender con ambientes más complejos con sistemas sensorio-motrices también más complejos.

En los seres vivos la regulación y sincronización de los diferentes componentes cerebrales, así como su interacción principalmente las representaciones de la modularidad externa, se lleva a cabo mediante diferentes sustancias que les permiten intercambiar información o generar procesos de sinapsis en las neuronas. En el caso de los músculos, la sustancia encargada de realizar los procesos de activación de neuronas efectoras es la Acetilcolina. Además, se considera un neurotransmisor primitivo, presente en todos los seres vivos y que es capaz de regular la producción de otros neurotransmisores. La acetilcolina está relacionada con enfermedades como el Parkinson y su efecto puede ser sustituido por algunas sustancias como la Nicotina, que provoca adicción a los seres humanos. También toma parte en el ciclo de la producción energética celular, en donde se produce una sustancia que posteriormente se transforma en el nuero-transmisor.

El modelado y aplicación de diferentes componentes del cerebro humano a sistemas de robótica evolutiva, pueden brindar un nivel de robustez adicional. Esto debido a que los sistemas de funcionamiento del cerebro tienen procesos que son el resultado de la evolución, el paso del tiempo y la adaptación a los diferentes medios. Al ser imitados, se asegura que se está siguiendo un camino ya probado por la naturaleza que tiene un buen funcionamiento. Así, la importancia del uso de la acetilcolina como inspiración para la creación del sistema de interacción viene de las funciones que la misma sustancia cumple en el sistema nervioso del ser humano. Se trata de la sustancia reguladora de la acción de diferentes regiones cerebrales en las neuronas motoras. Pero, como todo proceso de neurotransmisión, requiere de ciertos mecanismos de producción, traslado a las regiones de influencia, y recuperación que deben ser representados en un modelo computacional. Por todo ello es que su estudio es de particular importancia en las neurociencias, y es fuente para el modelado y como referencia de los sistemas bio-inspirados como la RE.

En los seres vivos la modularidad permite que los individuos tengan el funcionamiento coordinado de diferentes aparatos y sistemas. En el caso de los agentes inteligentes, especialmente en el caso de los robots evolutivos, la modularidad tiene el potencial para permitir aumentar la capacidad de resolución de tareas complicadas de optimizar mediante un sistema no modular en donde se

presentan interferencias neuronales. Con un arreglo de sistemas y módulos lo suficientemente extenso y combinado con la capacidad de producir niveles de inteligencia a partir de presiones evolutivas artificiales (Corucci, 2017), es posible representar una manera diferente de organizar los elementos de pensamiento, raciocinio, memoria y funcionalidad de agentes inteligentes y en particular robots evolutivos (Scheper, Tijmons, De Visser y De Croon, 2016).

Por otro lado, en términos de exploración de espacio de solución la red *acetilmodulada* representa una manera de descomponer en sub espacios más amigables para los algoritmos de solución. Cada uno de los módulos se encarga de atender un fragmento del problema original. Existe un sistema que discrimina entre la totalidad de los módulos, basado en la información del entorno. Determina de esta manera cuál de ellos puede representar una mejor opción para intervenir en una situación determinada.

Cuando se habla de optimización de pesos de las redes neuronales, el espacio de soluciones potenciales o espacio de búsqueda está integrado por la totalidad de configuraciones posibles dadas las restricciones de los valores de pesos. Entre mayor sea la cantidad de pesos por optimizar, el proceso de búsqueda se hace más complejo (Huizinga, Mouret y Clune, 2017). Pero también se vuelve más complejo cuando las cualidades buscadas en la red se encuentran en regiones diferentes del espacio de solución. Por ello obtener una configuración suficientemente robusta para solucionar un problema se vuelve una tarea complicada para el algoritmo de optimización. Una de las soluciones es buscar puntos medios que permitan resolver los problemas que representan configuraciones de pesos distantes en el espacio de soluciones (Rollins y Schrum, 2017). La modularización representa una solución diferente: Dividir el espacio de soluciones en fragmentos del problema original garantiza encontrar una reconfiguración en la que se puedan encontrar soluciones fragmentadas para el problema original.

Como ocurre con la *arquitectura acetilmodulada*, el problema de optimizar una sola red para controlar más de un proceso de clasificación o control se convierte en diversos sub-procesos de optimización de diferentes redes neuronales. El proceso finaliza con la optimización de la red que va a producir la selección de cuál de los módulos será elegido para solucionar el problema en un instante de tiempo. Esto representa también una redimencionalización del espacio de búsqueda y un cambio de objetivo: En lugar de optimizar la red para que cumpla con diferentes objetivos, la optimización tiene como finalidad encontrar una configuración de pesos para que una red decida cuando concede el control del sistema a alguno de los módulos previamente optimizados. En conjunto, tanto el espacio dividido en los módulos como el cambio de espacio de soluciones que representa el mecanismo de regulación modular resultan un espacio conjunto más amigables de explorar, en comparación con el espacio de búsqueda original.

El buen funcionamiento se debe entonces a la modularización del problema. Aunque en la *AME* también se modularizó dividiendo el trabajo en levantamiento y depósito, la interacción de la *arquitectura acetilmodulada* es diferente. En primer lugar, los módulos son optimizados por separado, lo que elimina el riesgo de perder óptimos locales para una tarea por buscar óptimos de la tarea contraria. Con los módulos optimizados, es posible transformar el espacio de búsqueda

original (levantamiento y depósito de basura) en un problema de búsqueda computacional en el cual se elige el módulo que puede proveer la mejor solución en un instante de tiempo determinado. Es decir, no se optimiza la red reguladora para alcanzar el comportamiento global, sino para regular la interacción de los módulos y encontrar patrones que permitan determinar cuándo un módulo se debe activar o no.

En el caso del experimento limpieza de entorno, la cantidad de módulos en el modelo *acetilmodulado* fueron dos. El sistema de interacción, que es la red productora de acetilcolina virtual, debe elegir en cuál de las dos neuronas de salida deberá producir un mayor nivel. Lo que provoca que el sistema funcione como un *switch* con tres posibilidades: activar el módulo de recolección, activar el módulo de depósito, o no activar ningún módulo.

El rendimiento de la *arquitectura acetilmodulada* resulta particularmente contrastante respecto a la *monored*. Mientras la red *acetilmodulada* fue capaz de producir a los mejores individuos durante este experimento, la *monored* no puede producir individuos lo suficientemente buenos para cumplir la tarea en su totalidad. Este resultado es producto nuevamente de la modularización de la tarea. Mientras que en el proceso evolutivo de la *monored* la búsqueda se debe encargar de ajustar dos tareas, en la red *acetilmodulada* el proceso evolutivo tiene diferentes etapas que evitan la interferencia de las tareas.

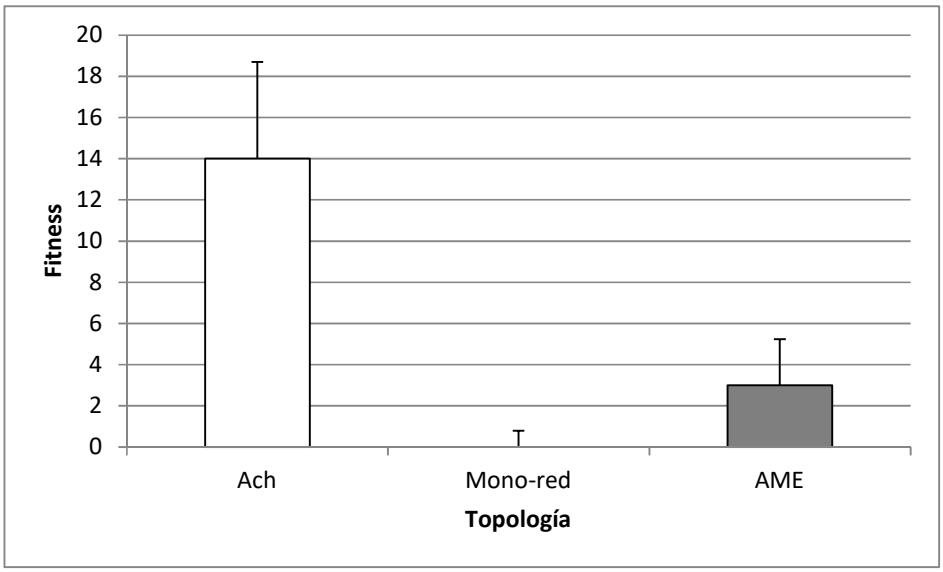
Uno de los problemas abiertos en el mundo de la RE es la generación de estructuras de control capaces de lidiar con tareas cada vez más complejas o elaboradas (Eiben, 2014). Una de las formas para lograrlo es también a través de los enfoques bio-inspirados y la transdisciplinaridad de la Inteligencia Artificial como el utilizado en la *arquitectura acetilmodulada*. En donde el uso de las propiedades de los neurotransmisores para la coordinación de diferentes tareas en el sistema nervioso, y las células de Renshaw y su capacidad de inhibición de neuronas colaterales, son utilizados como modelos para generar un sistema de control en forma de módulos externos.

Finalmente, es importante destacar que existen componentes en el modelo obtenido que no tienen equivalentes en el modelo físico del robot. Tal es el caso del sensor virtual de basura, que depende de manera directa de un supervisor de posición incluido dentro del simulador. Lo cual puede causar diferencias lo suficientemente grandes para considerarlas incorregibles y que tienden a aumentar el problema del *reality gap* (Ficici *et al.,* 1999).

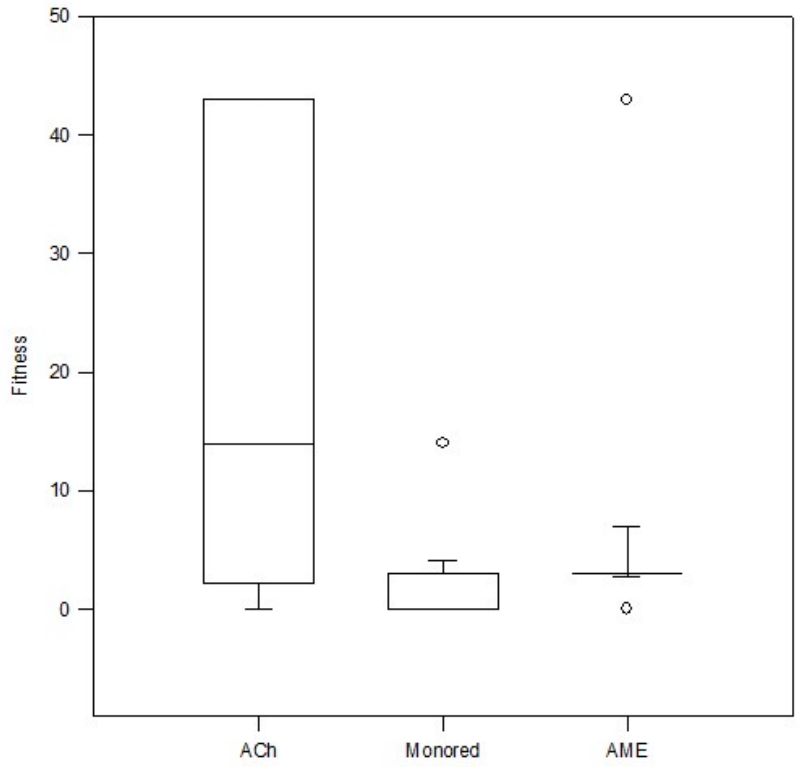
4.5 Resultados experimento 3: Cambio de cuarto (estudio comparativo).

Los resultados de la prueba estadística Kruskal Wallis o ANOVA de una vía para rangos (Shapiro-Wilk $p < 0.05$, no cumple con normalidad) aplicada a la variable dependiente "*fitness*" (ver Gráficas 8 y 9) mostraron diferencias significativas ($H_{(13.890)}$, $p<=0.001$, $2df$) entre las medianas de los diferentes procesos evolutivos en los tres grupos experimentales puestos a prueba. La prueba post-hoc reveló que el grupo que representa a la *arquitectura Acetilmodulada* (*14 ± 4.693*) es significativamente diferente ($p<0.05$) respecto a los otros dos grupos (*Monored = 0 ± 0.79*, *Arquitectura Modular Emergente = 3 ± 2.238*).

En este experimento, el mayor *fitness* esperado fue de 43. El grupo experimental que representa a la *arquitectura acetilmodulada*, tuvo el mejor rendimiento. Muchos de los individuos de dicho grupo experimental fueron capaces de cambiar de cuarto. En el caso de la *arquitectura modular emergente*, los individuos eran capaces de abrir la puerta y sólo uno de ellos pudo cruzar el cuarto. Finalmente, el grupo *monored* tuvo individuos capaces de abrir la puerta, pero también incapaces de realizar ninguno de los componentes de la tarea evolutiva (ver Tabla 8).



Gráfica 8. Medianas y error estándar de los grupos del tercer experimento. En este experimento se ponen a prueba las arquitecturas utilizadas en el experimento previo, con una tarea que necesita de mayores habilidades por parte de los robots para resolverlas.

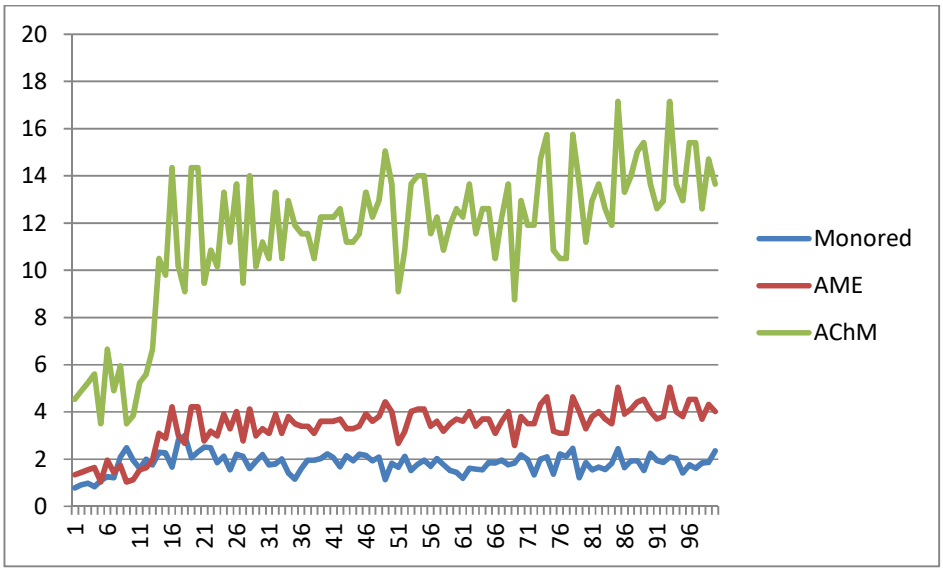


Gráfica 9. Gráfica de cajas de los resultados del experimento 3 (máximo, mínimo y mediana). La *arquitectura acetilmodulada* es la que mejor desempeño tiene, seguida de la AME.

Tabla 8. ANOVA de una vía para rango experimento 3 cambio de cuarto.

	Acetilmodulada	*Monored*	*Arquitectura Modular Emergente*
Mediana	***14****	***0***	***3***
Media + error estándar	***21.994 ± 4.693***	***1.944 ± 0.79***	***5.056 ± 2.238***
Desviación estándar	***19.910***	***3.351***	***9.496***
		H=13.890, p≤0.001, n=18	
		***SNK* comparación entre grupos**	

Los procesos evolutivos de la *monored* (ver Gráfica 10) muestran un bloqueo en el espacio de soluciones, por lo que la aptitud del proceso evolutivo tiene variaciones constantes. En los casos de la *AME*, es posible apreciar el efecto del bloqueo en el espacio de soluciones que vuelve inestable el proceso evolutivo, y la *arquitectura acetilmodulada*, después de un desempeño inicial bajo el proceso tiende a estabilizarse en un rango de valores del *fitness*. Aunque en el caso de la *arquitectura acetilmodulada* resultó un valor mayor (entre *11* y *14* puntos *fitness* mayor).



Gráfica 10. *Fitness* promedio del proceso evolutivo de una repetición de las tres arquitecturas en el experimento cambio de cuarto. Es posible apreciar el efecto del bloqueo en el espacio de soluciones que vuelve inestable el proceso evolutivo en el caso de la *monored*. También es posible observar un crecimiento constante en el *fitness* de la población durante las primeras etapas del proceso evolutivo de la AME. Existe un crecimiento constante en el *fitness* de la población durante las primeras etapas del proceso evolutivo de la *AChM*, el cual resulta mayor comparado con las otras dos arquitecturas.

4.6 Discusión experimento 3: Cambio de cuarto (estudio comparativo).

El tercer experimento tenía como finalidad probar la *arquitectura acetilmodulada* en el dominio de aplicación de la RE añadiendo complejidad al sistema de control mediante la adición de un módulo o una tarea adicional. De esta manera, una de las cuestiones que se deseaba observar es la capacidad de las arquitecturas modulares ante el incremento del número de módulos. Esto se debe a la posibilidad de encontrar arquitecturas modulares que funcionen de buena manera ante la presencia de dos módulos entre los cuales decidir, pero que su mecanismo de interacción resulte no adecuado ante la presencia de más de dos opciones a elegir.

La terea obliga a los individuos a adquirir tres habilidades básicas: buscar la zona para abrir la puerta, encontrar la entrada del pasillo y cruzarlo para cambiar de cuarto. Adicionalmente, existe una secuencia implícita para el cumplimiento de la tarea. El robot debía, en primer lugar, abrir la puerta para realizar el cambio de habitación. El siguiente paso debía ser encontrar la entrada y finalmente cambiar la habitación. Esta secuencia se vuelve un punto clave en la evolución de los diferentes grupos experimentales. Los datos muestran que, de las tres arquitecturas puestas a prueba, la que mejores resultados tuvo fue la red *acetilmodulada*.

En el caso de la *monored*, el proceso evolutivo produjo individuos no capaces de abrir la puerta o de realizar alguna de las otras acciones. En el caso de este sistema de control, el espacio de soluciones contaba con un componente de complejidad mayor. Aunque los individuos fueran capaces de encontrar la entrada del pasillo, de nada serviría sino eran capaces de abrir la puerta.

Esto evidencia un dato importante sobre la naturaleza de la tarea. Y es que los puntos óptimos que brindan solución a la tarea de apertura de puerta no convergen en el espacio de soluciones con aquellos que dan solución a la búsqueda del pasillo. En otras palabras, en la *monored* existe interferencia en los espacios de soluciones potenciales entre las tareas de apertura de puerta y búsqueda de la entrada del pasillo.

Con ello, y a partir de los datos obtenidos se puede decir que el cambio de cuarto es una situación o tarea adecuada para estudiar el fenómeno de la modularidad en los robots evolutivos. Esto debido a que una red neuronal por sí sola no es capaz de resolver esta tarea. Pero que a diferencia de la tarea limpieza del entorno, puede ser dividida en varios módulos y evitar que los modelos de selección sean los adecuados para resolver dos tareas mutuamente excluyentes. Lo que obliga a los sistemas de selección o de regulación de interacciones a buscar entre más de dos opciones la opción correcta que resuelva la tarea.

Pero también es importante destacar que existe un componente secuencial de aprendizaje, que, de adquirirse en el modo adecuado, puede generar buenos resultados. Este componente lleva entonces a un tipo de interferencia en espacios de búsqueda: interferencia secuencial. En ella, aunque los espacios de soluciones potenciales tengan regiones de convergencia entre diferentes tareas, si el conocimiento no se adquiere en un orden determinado, los algoritmos de búsqueda no obtendrán buenos resultados.

En el caso del grupo de robots con control *AME*, el proceso evolutivo tuvo poca variabilidad de resultados. La *arquitectura modular emergente*, tal cual es optimizada en este experimento, tiene la capacidad de producir individuos aptos para encontrar la zona de apertura de puerta. Pero fue incapaz de producir individuos que cumplieran con el resto de la tarea global.

La respuesta se encuentra en la naturaleza monopolizadora de esta arquitectura de control de robots evolutivos. Como es reportado por Nolfi (1997), la arquitectura tiende a generar que uno de los módulos domine al resto. Cuando se cuenta con dos módulos, como en el caso del experimento limpieza de entorno, el sistema de interacción entre módulos puede realizar la función de un *switch* apagando y prendiendo los módulos. Pero en el caso de contar con más de dos opciones, la selección se hace más difícil y la monopolización se vuelve más difícil de romper. Además de existir un componente secuencial en esta tarea, que impide a los individuos adquirir la capacidad de resolución de la tarea si no se adquieren la habilidad básica inicial (apertura de puerta). Una opción para mejorar el rendimiento de la *AME* en este problema puede ser el uso de la evolución incremental. De tal manera que la red de módulos internos vaya ajustando los pesos de los diferentes módulos conforme se van añadiendo componentes nuevos al *fitness* de manera secuencial (Faíña, Jacobsen y Risi, 2017). Esto puede producir que la especialización ocurra entre dos módulos que resuelven la misma tarea, pero no entre módulos que resuelven diferentes tareas.

Por su parte, la *arquitectura acetilmodulada* sí es capaz de lograr algunos casos de éxito. Aunque la mayoría de los individuos tienden a cumplir con dos terceras partes de la tarea global. Los buenos resultados de la arquitectura propuesta se deben a que con ella es posible evitar tanto los bloqueos entre sub-tareas como el bloque secuencial. Es decir, la modularidad externa no sólo tiene la capacidad de evitar el bloqueo espacios de búsqueda tradicional, sino que también puede evitar la incapacidad de los robots de adquirir habilidades relacionadas con un orden o secuencia de para la resolución de un problema determinado.

Al realizar la optimización de los tres módulos de manera separada, los individuos son expuestos a los diferentes estímulos del medio en donde se encuentran inmersos. De tal manera que, para la optimización del módulo encargado de resolver la apertura de la puerta, el proceso evolutivo sólo busca a los individuos con las mejores características para cumplir con dicha tarea. Para el módulo encargado de regular la búsqueda de la entrada del pasillo, el proceso evolutivo obviara la capacidad de los individuos para abrir la puerta. Por ello, durante esta fase de la evolución global la puerta permanece abierta. Y finalmente, el proceso de optimización del módulo encargado de hacer cruzar los robots a través del pasillo contempla únicamente dicha capacidad.

Y como sucedió en el experimento limpieza de entorno, la optimización del mecanismo de interacción o la red productora de acetilcolina artificial, tiene como principal tarea elegir a cuál de los módulos conviene otorgarle el control de los actuadores (tener la capacidad de hacer sinapsis química) bajo determinadas circunstancias o estímulos. Los resultados probaron que, para una variedad mayor a dos, el proceso de optimización dividido produce sistemas de interacción capaces de seleccionar el módulo correcto en una situación determinada.

Así la arquitectura propuesta rompe el bloque espacios de soluciones a partir de la modularización y la redimencionalización del mismo espacio. Pero también rompe la incapacidad de las redes neuronales para aprender por sí solas una tarea secuencial, exponiendo a los diferentes módulos a los estímulos indicados.

Finalmente es importante destacar algunos aspectos que deben ser analizados al utilizar una arquitectura de módulos externos. La modularidad de manera ideal permite la descomposición de un espacio de búsqueda en sub-espacios que tienen el potencial para resultar más sencillos de explorar. Esto reduce la carga de trabajo del algoritmo evolutivo y aumenta las probabilidades de encontrar configuraciones para que los neuro-controladores puedan cumplir con sus objetivos. Por lo que el éxito de dichas arquitecturas de control recae en la elección del módulo que dará solución de manera adecuada a una situación o estímulo en la que se encuentra inmerso el robot evolutivo. Las redes de módulos externos son una solución para los problemas de bloqueo en espacios de búsqueda. Resuelven además el caso especial de bloqueo por componentes de aprendizaje secuencial, como lo demostró la red *acetilmodulada* durante este experimento. Por lo que es posible afirmar que la arquitectura presentada durante los experimentos de esta investigación es capaz de descomponer los espacios de solución en sub-espacios que resultan más sencillos de explorar, y transforma el proceso de búsqueda computacional inicial a uno en donde

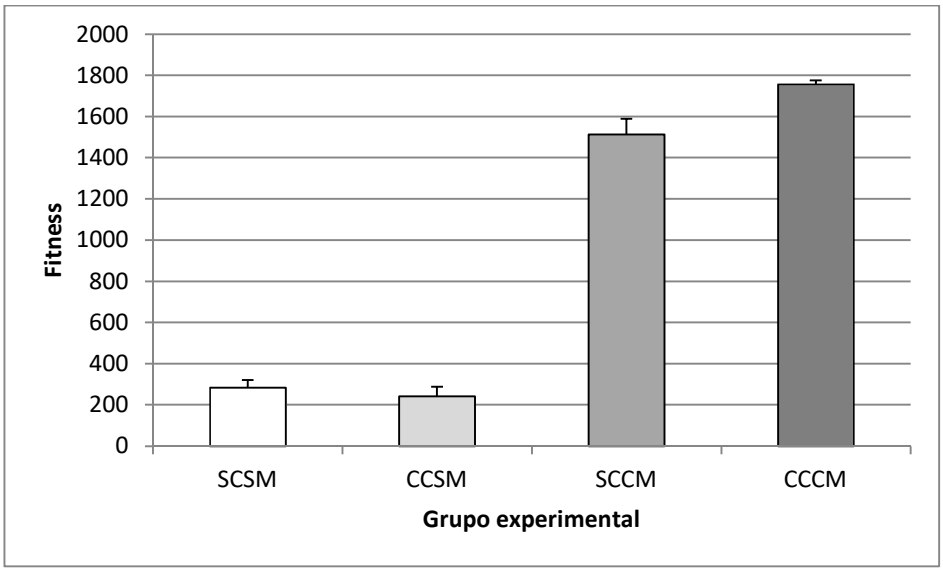
se debe encontrar la configuración correcta para la producción del modulador artificial y enviarlo al módulo que resuelve de mejor manera cada situación presentada a los robots evolutivos.

4.7 Resultados experimento 4: Modularidad y comunicación.

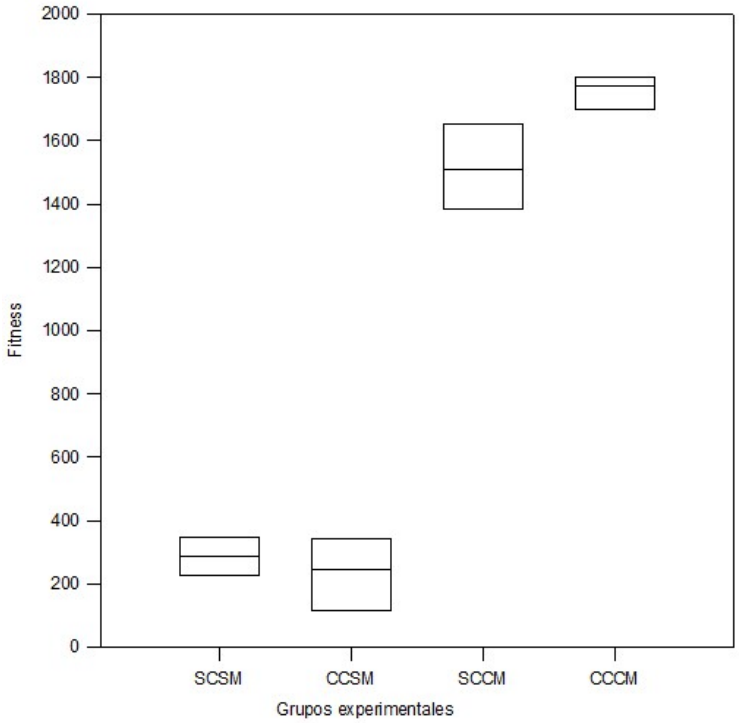
Los resultados de la prueba estadística ANOVA de dos vías (Shapiro-Wilk *P = 0.724*, Equivalencia de varianza *P = 0.098*, ambos cumplen) aplicada a la variable dependiente "*fitness*" (ver Gráficas 11 y 12) mostraron diferencias significativas dada la manipulación de la variable *modularidad* ($F_{(725.057)}$, *p<0.001, 1df*), pero sin diferencias significativas en la manipulación de la variable *comunicación* ($F_{(3.924)}$, *p=0.062, 1df*). Es decir, el efecto de los diferentes niveles de comunicación depende de la presencia de modularidad (ver Tabla 9). La prueba Post-Hoc determinó diferencias significativas entre la presencia y ausencia de la variable modularidad (*p<0.05*). Por su parte la variable comunicación en los cuatro grupos experimentales no presenta diferencias significativas (*p=0.62, 1df*), por lo que es posible determinar que no existen efectos estadísticos producto de la presencia o ausencia de dicha variable.

Adicionalmente la prueba estadística arrojó diferencias significativas producidas por la interacción entre los factores *modularidad* y *comunicación* ($F_{(8.038)}$, *p=0.010, 23df*). La prueba de interacciones Post-hoc mostró que existen diferencias significativas (*p<0.05*) dada la presencia y ausencia de la variable comunicación en los grupos con la presencia de la variable modularidad. Un resultado igual se generó dada la presencia y ausencia de la variable *modularidad* dada la ausencia de la variable *comunicación* (*p<0.05*), y dada la presencia de la variable *comunicación* ante la ausencia y la presencia de la *modularidad* (*p<0.05*). Por último, la prueba Post-hoc no determinó diferencias significativas dada la ausencia de la variable *modularidad* ante la presencia y ausencia de la *comunicación* (*p=0.553*).

En este experimento, el mayor *fitness* esperado es de *1800* puntos. El grupo experimental que representa la presencia de ambas variables (*modularidad* y *comunicación*) tuvo el mejor rendimiento (*1756 ± 20.061*). El segundo mejor desempeño tuvo lugar en el grupo con presencia de *modularidad* y ausencia de sistema de *comunicación* (*1511.63 ± 78.737*). El otro par de grupos experimentales tuvo un rendimiento bajo (Sin *modularidad* y sin *comunicación = 284.122 ± 37.179*, Con *comunicación* y sin *modularidad = 240.607 ± 48.892*).



Gráfica 11. Medias y error estándar de los grupos del cuarto experimento. En este experimento se puso a prueba la presencia y ausencia de las variables experimentales comunicación y modularidad, así como su interacción.



Gráfica 12. Gráfica de cajas de los resultados del experimento 4 (máximo, mínimo y media). Los grupos experimentales sin variable modularidad son los que peor desempeño tuvieron.

Tabla 9. ANOVA de dos vías experimento modularidad y comunicación.

Media + error estándar (desviación estándar)	Comunicación (ausencia)	Comunicación (presencia)	Sin comunicación con modularidad	Con comunicación con modularidad
Modularidad (ausencia)	***284.122 ± 37.1709 (91.069)***	***240.607 ± 48.892 (119.761)***	***262.364 ± 30.008 (103.950)***	***F=725.057l, p<0.001***
Modularidad (presencia)	***1511.63 ± 78.737 (192.865) ****	***1756.333 ± 20.061 (49.132) ****	***1633.698 ± 53.550 (185.504)***	
	998.470 ± 229.889 (796.359)	***897.592 ± 189.569 (659.687)***	***F=8.038, p=0.010, n=24***	
	F=3.924, p=0.062			
**SNK* en relación a la comparación entre variables				

El muestreo de comportamientos aplicado a los grupos con el sistema de comunicación habilitado mostró que los sistemas de comunicación evolutivos se establecieron de mejor manera en el grupo que cuenta con la variable de *modularidad* presente (ver Tabla 10). Además, mostró que las señales se producen en su mayoría dentro de la zona de las zonas de depósito y recolección de basura.

Tabla 10. Señales emergentes identificadas.

	Con comunicación sin modularidad	Con comunicación con modularidad
Número de repeticiones que desarrollan un sistema de comunicación estable (Marocco *et al.*, 2003)	***2/6***	***5/6***
Número de repeticiones que desarrollan señales en zonas de recolección de basura virtual	***1/6***	***3/6***
Número de repeticiones que desarrollan señales en zonas de recolección de basura virtual	***0/6***	***2/6***

4.8 Discusión experimento 4: Modularidad y comunicación.

La finalidad de este experimento fue la de conocer los efectos de la combinación de dos herramientas que buscan dotar de mayores recursos a los robots evolutivos, en particular como solución al problema del bloque en espacios de búsqueda: *modularidad* y *comunicación*. La comunicación es una herramienta que permite a las comunidades de robots intercambiar información. Pero lo más interesante de los robots evolutivos es que los sistemas de comunicación emergen del proceso evolutivo. Por ello, este tipo de experimentos se vuelven una plataforma para el estudio de los orígenes y evolución de los sistemas de comunicación.

Lo primero a destacar es la escalabilidad de la tarea de limpieza de entorno. Dicha tarea pudo ser resuelta por un grupo de robots. Y lo que resulta mejor es que permite el estudio de sistemas de comunicación, ya que es necesario un nivel de coordinación entre los individuos que conforman un equipo.

En algunas ocasiones es necesario variar las condiciones del entorno para favorecer la emergencia de la comunicación. Dichas manipulaciones buscan en ocasiones modificar el nivel de complejidad o dificultad de la tarea. Es decir, se busca que la tarea resulte un estímulo adecuado para que las presiones evolutivas produzcan la emergencia de las señales. Entonces la complejidad del medio o dificultad de la tarea es un factor determinante para la emergencia del sistema de comunicación (Aldana-Franco, Montes y Nolfi, 2017). Por lo que, en entornos difíciles o complejos, los sistemas de comunicación tienden a emerger también en una forma compleja. Esta complejidad debe entenderse en términos de la cantidad de señales emergentes con un valor léxico.

Una ventaja adicional de la resolución de tareas por grupos de robots es que cuando la tarea pasa de ser resuelta por un sólo individuo a ser resuelta por varios, la cantidad de estímulos a los que son expuestas las poblaciones de robots evolutivos también aumentan. Es decir, al aumentar la cantidad de individuos que participan en la resolución de la tarea, también se están aumentando la cantidad de situaciones a las cuales son expuestas las redes neuronales artificiales. Lo que debe provocar una aceleración en la convergencia de los algoritmos evolutivos de búsqueda (Karafotias *et al.*, 2011).

Por su parte la modularidad, como se mostró en los dos experimentos anteriores, es una herramienta enfocada a mejorar los procesos evolutivos y en particular los espacios de búsqueda. Es una herramienta que permite sortear los bloqueos en espacios de búsqueda, dividiendo los espacios difíciles de explorar que cuentan con puntos de óptimos de convergencia distantes, en sub espacios más sencillos de explorar. Desde ese mismo punto de vista, los sistemas de comunicación pueden ser vistos como una variable más que cambia la dimensionalidad del espacio de búsqueda. Esto puede ser un hecho favorable o desfavorable para la búsqueda computacional. Será desfavorable cuando a un espacio difícil de explorar se le aumenta el nivel de complejidad de búsqueda a partir de la adición de una nueva variable de optimización o evolución. Pero puede ser favorable porque le brinda al espacio de búsqueda una nueva variable que puede ser optimizada y así alcanzar la meta de evolución. Es decir, un robot evolutivo al que se le brinda

la posibilidad de comunicarse con otro, cuenta una herramienta adicional con la cual resolver una tarea. De esta manera la adición de la comunicación puede facilitar la resolución de una terea que sin ella sería imposible de lograr. Entonces en términos del espacio de búsqueda, la comunicación corresponde a una nueva variable que puede representar la potencial solución a una determinada tarea.

Tanto en el caso de la modularidad como con los sistemas de comunicación, el fin principal es mejorar el desempeño de los robots evolutivos inmersos en un ambiente determinado para resolver alguna tarea determinada. Por lo que esperar que la interacción entre ellas sea positiva y ayude a los sistemas de control de robots evolutivos no resulta una idea descabellada.

Como lo muestran los resultados, la tarea no pudo resolverse sin la presencia de ambas variables. Es decir, el grupo control en donde los robots no contaban con el control modular ni con el sistema de comunicación tuvo el peor de los resultados. La respuesta radica en dos factores importantes. El primero es que, como se demostró en el experimento limpieza de entorno, los sistemas de control no modulares no son capaces de resolver la tarea de limpieza de entorno tal cual se ha presentado en esta investigación. El segundo factor es provocado por los sistemas de comunicación: la interacción entre los robots de un equipo. La cual resulta un elemento importante, ya que en algunas tareas los sistemas de comunicación derivan en la organización de los robots para resolver la tarea. En dichos casos los robots se especializan en resolver una determinada parte de la tarea. La sincronización de las sub-tareas corre a cargo del sistema de comunicación. Al carecer de dicho sistema, no existen los medios suficientes para generar un nivel de coordinación entre los individuos de un grupo y los resultados no son buenos.

El segundo grupo representó a un sistema de control del tipo *monored* o no modular combinado con los elementos que pueden dar luz a la emergencia potencial de un sistema de comunicación. Sin embargo, añadir una salida adicional a una *monored* para hacer funcionar un actuador de comunicación, resulta en un caso desfavorable de comunicación. Esto combinado a la imposibilidad de los robots de ajustar los pesos para ambas tareas, aumenta un factor adicional al bloque en espacios de búsqueda. Por lo que resulta difícil para el algoritmo de búsqueda ajustar los pesos para cumplir con las tareas de limpieza y de comunicación. Aunque existe el potencial de emergencia de coordinación entre los robots, el bloqueo presente entre las tareas de recolección y depósito de basura virtual impiden que el algoritmo genético tenga la capacidad para encontrar una combinación de pesos para la red neuronal en un punto del espacio de búsqueda. En el caso de la tarea de recolección de basura se puede comprobar que en los casos para las cuales existe un bloque entre dos tareas, al agregar una tercera el bloque prevalece.

Este resultado también resulta relevante para el estudio de los sistemas de comunicación debido a que demuestra que una de las variables para la emergencia de los mismos sistemas es la inexistencia de los bloqueos de espacios de soluciones. Cuyo resultado final es la imposibilidad de emergencia de señales, ya que los pesos de los neuro-controladores no pueden ser optimizados. En el caso de este segundo grupo experimental, la tendencia fue que la mayoría de los robots tienden a aprender la tarea de recolección en el mejor de los casos, y ninguna de las tareas en el

peor. Tampoco desarrollaron un sistema de comunicación con estrategias de señalización que resulten estables y destacables.

El grupo tres correspondió a la *arquitectura acetilmodulada* aplicada a un grupo de robots evolutivos. Los resultados muestran un rendimiento muy similar a los mostrados en el experimento limpieza de entorno, en particular a los grupos de la misma arquitectura. Los grupos de robots son capaces de cumplir con las tareas, pero el *nivel de fitness* no corresponde al máximo dentro del experimento modularidad y comunicación. La causa de esto es que no existe una coordinación real entre los robots, ya que no se cuenta con los medios para realizarlo. Lo que provoca que cada uno de los robots actúe como entes individuales. Aunque parezca un factor bastante obvio, es necesario contar con canales de comunicación para que exista coordinación entre miembros de un grupo de robots evolutivos. Este canal de comunicación puede ser tan sofisticado como sencillo, y obedecer a diferentes naturalezas como mecanismos visuales de movimiento percibidos por sensores infrarrojos o ultrasónicos, hasta sistemas basados en la emisión de sonidos, o el cambio de tonalidades de LEDs (Moritz y Mostaghim, 2017).

Esto es lo que sucede con el último grupo experimental. En él se combinan los factores necesarios para lograr el mejor desempeño de este experimento: modularidad y comunicación. El primero es la modularidad que se encarga de evitar los bloqueos entre tareas a los que se ven expuestos los procesos evolutivos. Es decir, dividen el espacio original de soluciones potenciales para la tarea de limpieza de entorno en dos sub-espacios (levantar y tirar basura). También se genera un nuevo espacio de soluciones para un problema de selección de módulo. El segundo factor que actúa sobre estas poblaciones evolucionadas es la comunicación. Al existir un medio de comunicación, los robots evolutivos tienen un mecanismo por el cual coordinarse como grupo. Por ello la resolución de la tarea resultó más efectiva que en cualquiera de los otros grupos. Existen diferentes estrategias de señalización que evolucionaron en las diferentes repeticiones del grupo experimental. Entre ellas destaca la señalización de los espacios de depósito y/o recolección de basura. Esto permitió atraer a los robots a dichas áreas y de esta manera no perder el tiempo en la exploración y localización de los mismos.

Los resultados de este experimento permiten aseverar algunas cuestiones. Lo primero es que los sistemas de comunicación son los principales causantes de la emergencia de comportamientos coordinados en grupos de robots evolutivos (Stender, Yan, Karayaka, Tay y Adams, 2017). Como en todos los sistemas de comunicación de la naturaleza, es necesario que existan algunos componentes necesarios: emisor, receptor, canal y señal. Los individuos que hacen posible la comunicación son el receptor y el emisor que interpretan y expresan información. Pero no basta contar con un par de individuos que puedan realizar estas operaciones básicas del circuito del habla. También es fundamental contar con un canal de comunicación adecuado que sirve como medio de transmisión para las señales. Y finalmente con la señal misma, que es la instancia por la cual se comparte la información. Si alguno de estos componentes falta, la comunicación no puede establecerse y los mecanismos de coordinación necesarios no están disponibles para los robots que los requieren.

En cuanto a la modularidad, los resultados son consistentes respecto a los resultados presentados en los experimentos exploratorios, limpieza de entorno y cambio de cuarto. La modularidad por si sola es una herramienta capaz de evitar el fenómeno de la interferencia entre espacios de soluciones. En particular, la modularidad externa tiene la capacidad de descomponer los espacios de estados en sub-espacios más amigables para la exploración de los algoritmos de optimización en robótica evolutiva. Los sistemas de comunicación por si solos no tienen dicha capacidad, pero al combinarlos con sistemas de control modulares el rendimiento de las poblaciones mejora. Por ello se debe considerar que los sistemas de comunicación no son una herramienta fundamental para resolver el problema de la interferencia de los espacios de búsqueda. El uso de dichos sistemas fue una herramienta de apoyo que puede mejorar el desempeño de la evolución en grupos de robots evolutivos que deben solucionar una tarea. Es el medio adecuado para otorgar un nivel de interacción a los robots evolutivos que tratan de resolver un problema con un cierto nivel de cooperación y/o coordinación.

Capítulo 5.

Conclusiones.

Dentro de este quinto capítulo se presentan las conclusiones de la investigación. También se muestra el aporte de la investigación y su comparación respecto a otros trabajos similares. Finalmente se mencionan los retos y la perspectiva derivados de los resultados de la investigación derivados.

5.1 Conclusiones.

La *arquitectura acetilmodulada* es una solución al problema de los bloqueos en los espacios de búsqueda potenciales o bloqueo neuronal, como se muestra en los experimentos exploratorio, limpieza de entorno, cambio de cuarto, modularidad y comunicación. En particular se probó que la arquitectura propuesta tiene la capacidad de enfrentar los bloqueos a partir de la estrategia "divide y vencerás". De esta manera lo que realiza es una división del espacio de soluciones en fragmentos en los que no existan bloqueos. Esta arquitectura para sistemas de control de robots evolutivos está basada en el enfoque de módulos externos, un sistema de regulación de interacción modular basado en la acción de la acetilcolina en los músculos voluntarios de los seres vivos y su acción en la sinapsis química, así como la acción de neuronas de Renshaw para inhibir células colaterales. Por ello el modelo propuesto se considera bio-inspirado con el cual se busca rescatar algunas características que en la naturaleza representan una solución a un problema de la IA. El neurotransmisor cumple con los preceptos de las sustancias naturales que actúan en los seres vivos como son: la producción se realiza por neuronas, su acción va de la neurona pre-sináptica a la post-sináptica, existe un mecanismo de eliminación. Con esto se está replicando la capacidad de los organismos para funcionar en regiones, órganos, aparatos y sistemas en donde cada uno resuelve tareas específicas y cuya sincronización significa la supervivencia de los individuos. Así el modelo propuesto traslada el funcionamiento a módulos o bloques que deben resolver una fracción de un problema, y que relacionados representan la solución total a los problemas que el robot evolutivo enfrenta.

El experimento del estudio exploratorio tiene como principal aporte la propuesta de un método de diseño que en la gran mayoría de las ocasiones se omite dentro de los experimentos en robótica evolutiva. Este método permite comparar diferentes estructuras de redes neuronales artificiales con la finalidad de encontrar la que mejor resuelva el problema. Pero también es un esfuerzo por entender la naturaleza de los mecanismos que intervienen en la modularización de problemas utilizando redes neuronales artificiales. Es decir, se trata de un estudio exploratorio con un cambio de dominio de aplicación en donde es posible conocer totalmente la respuesta esperada por las RNAs. El resultado relevante de este experimento para la investigación es que la red *acetilmodulada* tiene la capacidad de evitar la monopolización de los recursos por parte de un módulo. Es decir, en la solución de una instancia determinada, todos los módulos tienen la misma probabilidad inicial de participar en su solución.

Dentro del experimento limpieza de entorno se presentó una comparación entre la arquitectura de módulos externos (*acetilmodulada*), una arquitectura de módulos internos (*arquitectura modular emergente*), y una *monored*. La tarea ya ha sido utilizada para validar el funcionamiento de controles modulares. Así se pudo confirmar, a través de los puntajes de aptitud de la *monored*, que existe un bloqueo potencial entre las tareas de levantar y depositar basura. Bajo las condiciones experimentales presentadas, la red *acetilmodulada* muestra tener un mejor rendimiento que el resto. Lo que representa que este modelo de control para robots evolutivos es capaz de solucionar bloqueos en los espacios de búsqueda. Además, confirma la tendencia de la arquitectura de módulos internos por generar módulos que tienden a monopolizar los recursos. Esto quiere decir que de los dos módulos que tratan de solventar la misma problemática, sólo uno estará activo a través del nivel más alto de su neurona de activación. Pero dado el mecanismo utilizado para la modularización, en donde la función de calidad original se divide en sus dos términos matemáticos, existe monopolización de los recursos inclusive entre módulos que deben solucionar diferentes partes de la tarea global.

Aunado a esto, el experimento cambio de cuarto revela un nuevo componente en la naturaleza del fenómeno de los bloqueos en los espacios de solución: aprendizaje secuencial. La *arquitectura acetilmodulada* tiene la capacidad de resolver cualquier problema relacionado con este componente, debido a la descomposición del espacio original de búsqueda en sub-espacios que resultan más sencillos de explorar para el algoritmo de optimización. Pero también demuestra que la arquitectura de control propuesta, en especial el sistema de interacción modular, tiene la capacidad de funcionar de funcionar en con un número mayor a dos módulos. Es decir, la arquitectura propuesta no sólo funciona con diseños de módulos mutuamente excluyentes, también tiene la capacidad de hacerlo con una mayor cantidad de ellos.

Con el experimento modularidad y comunicación se comprobó que la comunicación es una herramienta de apoyo para un grupo de robots evolutivos que deben resolver una tarea que representa un bloqueo en el espacio de búsqueda. Ésta genera las condiciones necesarias para la emergencia de la cooperación entre individuos. Por lo que la modularidad combinada con los sistemas de comunicación tiene la capacidad de mejorar las soluciones obtenidas mediante el proceso evolutivo a través de habilidades comunicativas de los individuos que permiten establecer nuevas soluciones para resolver las tareas. Además, la modularidad favorece a los sistemas de comunicación al ayudar a eliminar bloqueos en el espacio de exploración del algoritmo evolutivo.

Finalmente, la red *acetilmodulada* permite dividir un espacio de soluciones que presenta bloqueos. Esta subdivisión del espacio original conlleva una redimencionalización del mismo y permite que los algoritmos de optimización puedan actuar de mejor manera. Por lo que es recomendable su uso cuando exista un bloque en el espacio de soluciones que impida obtener buenos individuos en los procesos evolutivos de robots. Aunque, de no existir el bloqueo también puede ser utilizada. Además, este modelo de redes neuronales artificiales también puede ser extendido a otros dominios de aplicación como la clasificación y el aprendizaje automático.

Como consecuencia de este trabajo de investigación, se produjeron los siguientes artículos y reportes técnicos:

- A Basic Modular and Divided Control versus Heterogeneous Control: Separable Search Spaces Case. FODIClyT 2015.
- The Impact of population composition for cooperation emergence in evolutionary robotics. IJICOPI. 2017.
- Software de Optimización de Redes Neuronales Artificiales para Controladores Proporcionales (SORNA_CP) en un Sistema de Control retroalimentado: Reporte Técnico. Mayo 2017.
- Acetyl-Modulated Architecture for Evolutionary Robotics. IJICOPI. 2017.
- Implementation of a simulation model for ROV in a decision task problem. Workshop Hybrid Intelligent System MICAI 2017 y Journal Research in Computing Science.
- Environmental factors that affect the emergence of signal in population of evolutionary robots. Journal of Artificial Intelligence Research. En revision. 2017.

5.2 Trabajo futuro.

A partir de los resultados obtenidos durante esta investigación, es posible crear nuevas preguntas de investigación a resolver que buscan profundizar diferentes aspectos tanto de la naturaleza del fenómeno de la modularidad a través de la *arquitectura acetilmodulada*. Un aspecto importante es analizar el comportamiento de la arquitectura propuesta ante la posibilidad de ajustes morfológicos de los módulos y por supuesto de la red moduladora. La finalidad sería optimizar los sistemas de control modulares, y reducir el tamaño de los mismos. Esto último tiene el potencial para reducir la dimensionalidad del espacio de soluciones potenciales y así aligerar la carga de trabajo al algoritmo evolutivo.

La *arquitectura acetilmodulada* es un modelo de la acción de diferentes componentes del sistema nervioso central en los seres vivos. Indagar sobre la naturaleza de la respuesta de algunos de ellos es un reto abierto y un camino para mejorar el modelo mismo. Entre ellos destaca la relación inversa que existe en la naturaleza entre la creación de la acetilcolina y el consumo energético de las células (glicólisis). Esto implicaría la adición de una nueva variable experimental que contempla el consumo y la producción de energía, necesaria para la supervivencia del robot evolutivo.

Adicionalmente, la acetilcolina es el único neurotransmisor que no puede ser consumido de manera directa por los seres vivos. Los componentes necesarios para su producción provienen de la ingesta. Aunque todos los alimentos que consumimos los seres vivos tienen cantidades de acetilcolina, ésta es neutralizada por una sustancia antagonista conocida como butilcolinesterasa. El objetivo de esto es conseguir que los músculos involucrados en el sistema digestivo realicen movimientos involuntarios producto de la presencia del neuromodulador. Sin embargo, existen algunas sustancias agonistas del neurotransmisor que logran un efecto similar en los seres vivos. Entre ellas se encuentra la nicotina. Dicha sustancia tiene propiedades que pueden sustituir a corto plazo el efecto de la acetilcolina, pero también produce efectos adictivos. Lo que sucede con

ella es que produce un aumento en la cantidad de receptores post-sinápticos. Lo que quiere decir que los consumidores requieren una mayor cantidad de la sustancia para que sus efectos se perciban en el cuerpo del consumidor. Este efecto puede ser tomado en cuenta dentro del modelo computacional con la finalidad de tener respuestas similares a las producidas en la naturaleza.

Aumentar la complejidad del modelo en términos de su funcionamiento en los seres vivos puede permitir que otras áreas del conocimiento como las neurociencias, puedan utilizar a los robots evolutivos como plataforma de experimentación. Esto tendría un efecto en la disminución de especímenes vivos experimentales que sufren daños, pero también ayudaría al estudio de enfermedades graves y crónico-degenerativas como el Alzheimer o el Parkinson asociadas con los niveles de acetilcolina en el cuerpo. Lo cual resulta una de las principales aplicaciones potenciales de este modelo computacionales, y de todos los que involucran neurotransmisores.

También resulta factible profundizar sobre el estudio de la respuesta de la sinapsis química propuesta en el modelo computacional de la *arquitectura acetilmodulada*. Uno de los principales efectos que se deben tratar de modelar es el retardo en la propagación de las señales sinápticas, la posibilidad de amplificación en relación con las cantidades de receptores en las neuronas post-sinápticas, así como el mismo efecto en la velocidad e intensidad de las sinapsis de las neuronas artificiales en relación con el aumento de los receptores.

Otro de los caminos que pueden derivar de esta investigación, y que representan la aplicación potencial número uno de la arquitectura propuesta, es la creación de sistemas nerviosos para robots. Inclusive sistemas y aparatos bien definidos para robots que permitan aumentar la capacidad de resolución de tareas. En otras palabras, este modelo de arquitectura de control puede crear las condiciones para que la organización de los robots evolutivos tome un nuevo enfoque, y les permita lograr nuevos niveles de inteligencia. Este proceso puede involucrar la relación de la acetilcolina con la producción y regulación de otros neurotransmisores que resultan importantes en los procesos cognitivos y de memoria. Por ejemplo, se puede plantear un modelo que involucre la acetilcolina como medio de regulación del aparato motriz de los robots, pero también como un requisito para la producción de otro neurotransmisor involucrado en las tareas de aprendizaje que se encuentren involucradas en una actividad específica.

Una ruta natural que la investigación puede tomar es la de probar la arquitectura propuesta contra otras arquitecturas que han sido creadas, y también en otras tareas que pueden resultar más demandantes. Por ejemplo, la tarea de rescate de compañeros. Esta involucra un laberinto *T-Maze* por el cual los robots evolutivos deben moverse y encontrar a un compañero para regresarlo. El nivel de complejidad de esta tarea resulta mayor y derivará en la necesidad de una mayor cantidad de módulos para resolverla.

El estudio de los sistemas de comunicación asociados con la arquitectura modular propuesta puede resultar otro camino a seguir. La arquitectura provee una plataforma de estudio que aumenta la variedad de las tareas que se pueden conseguir. Pero también representa una posibilidad para crear módulos especializados en la adquisición de léxico, estructuras gramaticales, así como el estudio de la sintaxis y la semántica del lenguaje. Es decir, es posible modularizar los elementos involucrados en la naturaleza del lenguaje. De tal manera se tendrían estructuras especializadas en algún componente del lenguaje que al interactuar pueden derivar en sistemas de comunicación más complejos y completos para el apoyo en la resolución de tareas más demandantes.

Referencias.

Aldana-Franco, F. (2011). *Desarrollo de un Esquema Comunicativo Evolutivo-Cooperativo para un Grupo de Robots* (Tesis de maestría). Maestría en Inteligencia Artificial, Universidad Veracruzana, Xalapa, Veracruz, México.

Aldana-Franco, F. (2017). *Arquitectura Modular Aplicada a la Robótica Evolutiva Inspirada en la Acción del Neurotransmisor Acetilcolina sobre Neuronas Efectoras* (Tesis de Doctorado). Centro de Investigaciones de Inteligencia Artificial, Universidad Veracruzana.

Aldana-Franco, F., y Montes-González, F. (2015). *Basic Modular and Divided Control versus Heterogeneous Control: Separable Search Spaces Case*. En V Foro Nacional de Divulgación Científica y Tecnológica. Facultad de Ingeniería Mecánica Eléctrica Xalapa, Universidad Veracruzana. Xalapa, Veracruz México.

Aldana-Franco, F., y Montes-González, F., Nolfi, S. (Por Publicar 2017). *Environmental factors that affect the emergence of signal in population of evolutionary robots*.

Aldana-Franco, F., Montes-González, F., y Ochoa, A. (2017a). *Acetyl-Modulated Architecture for Evolutionary Robotics*. International Journal of Combinatorial Optimization Problems and Informatics.

Aldana-Franco, F., Montes-González, F., y Ochoa, A. (2017b). *Implementation of a simulation model for ROV in a decision task problem*. Journal Research in Computing Science. Por publicar.

Alvarez, F.J., y Fyffe, R.E. (2007). *The continuing case for the Renshaw cell*. The Journal of physiology, 584(1), 31-45.

André, J.B., y Nolfi, S. (2016). *Evolutionary Robotics simulations help explain why reciprocity is rare in nature*. Scientific Reports, 6.

Aznavurian, A., y Aguilar-Rebolledo, F. (2006). *Espasticidad ¿Qué es y qué no es? Nuevos Horizontes en la Restauración Neurológica.* Plasticidad y Restauración Neurológica, 5(2).

Ballard, D.H. (1986). *Cortical Connnections and Parallel Processing. Structure and Function*. The Behavioural and Brain Science, 9(1). 67-90.

Barandiaran, X. (2003). *Breve Introducción a la Robótica en Teoría del Conocimiento* (Reporte técnico). Universidad del País Vasco, España.

Barandiaran, X. (2003). *La Robótica Evolutiva como Epistemología Experimental Naturalizada* (Reporte técnico). Universidad Del País Vasco, España.

Bates, E. (1993). *Modularity, domain specificity and the development of language*. Discussions in Neuroscience, 10(1), 136-148.

Becerra, J., Santos, J., y Duro, R. (1999). *Progressive Construction of Compound Behavior Controllers for Autonomous Robot Using Temporal Information*. Advances in Artificial Life, 324-328.

Bernard, A., André, J. B., y Bredeche, N. (2016). *To cooperate or not to cooperate: why behavioural mechanisms matter*. PLoS Comput Biol, 12(5), e 1004886.

Bezdek, J.C., Keller, J.M., Krishnapuram, R., Kuncheva, L.I., y Pal, N.R. (1999). *Will the real iris data please stand up?* IEEE Transactions on Fuzzy Systems, 7(3), 368-369.

Bolker, J.A. (2000). *Modularity in Development and Why It Matters to Evo-Devo*. Amercian Zoologist, 40(5), 770-776.

Bongard, J. (2008). *Behavior Changing-Incremental Behavior Integration on Evolutionary Robotics*. Artificial Life, 11, 64-71.

Bongard, J. (2013). *Evolutionary Robotics*. Communications of the ACM, 56(8).

Bongard, J. (2015). *Using robots to investigate the evolution of adaptive behavior*. Current Opinion in Behavioral Sciences, 6, 168-173.

Brooks, R. (1986). *A robust layered control system for a mobile robot*. IEEE journal on robotics and automation, 2(1), 14-23.

Bryant, B.D., y Miikkulainen, R. (2003). *Neuroevolution for Adaptive Teams*. En Proceedings of the 2003 Conference on Evolutionary Computation, (3), 2194-2201.

Caelli, T., Guan, L., y Wen, W. (1999). *Modularity in Neural Computing*. Proceedings of the IEEE, 87(9), 1497-1518.

Calabretta, R. (2007). *Genetic interference reduces the evolvability of modular and nonmodular visual neural networks*. Philosophical Transcations of The Royal Society of London B: Biological Sciences, 362(1479), 403-410.

Calabretta, R., Di Fernando, A., Parisi, D., y Keil, F.C. (2008). *How to learn multiple tasks*. Biological Theory, 3(1), 30-41.

Calabretta, R., Di Fernando, A., Wagner, G.P., y Parisi, D. (2003). *What does it take to evolve behaviorally complex organisms?* BioSystems, 69(2), 245-262.

Calabretta, R., Nolfi, S., Parisi, D., y Wagner, G.P. (1998a). *Emergence of Functional Modularity in Robots*. From Animals to Animats, 5, 497-504.

Calabretta, R., Nolfi, S., Parisi, D., y Wagner, G.P. (1998b). *A case of the evolution of modularity: towards a bridge between evolutionary biology, artificial life, neuro- and cognitive science*. Proceedings of Sixth International Conference on Artificial Life, (6), 275-284.

Calabretta, R., y Parisi, D. (2005). *Evolutionary Connectionism and Mind/Brain Modularity*. En Modularity, 309.

Calandra, R., Seyfarth, A., Peters, J., y Deisenroth, M.P. (2016). *Bayesian optimization for learning gaits under uncertainty*. Annnals of Mathematics and Artificial Intelligence, 76(1-2), 5-23.

Cao, J., Ahmadi, M., y Shridhar, M. (1997). *A Hierarchical Neural Network Architecture for Handwritten Numeral Recognition*. Pattern Recognition, 30(2), 289-294.

Carvalho, J., y Nolfi, S. (2016). *Affordance generation enables behavioral plasticity and cognitive offloading in evolving robots*. En 2016 IEEE Symposium Series on Computational Intelligence (SSCI), 1-8. IEEE.

Cazorla, M., Colomina, O., Escolano, F., Gallardo, D., Rizo, R., y Satorre, R. (1997). *Técnicas de inteligencia artificial*. Editorial Club Universitario, Alicante, España.

Cho, S. (1997). *Combining Modular Neural Networks Developed by Evolutionary Algorithm.* En IEEE International Conference on Evolutionary Computation 1997, 647-650.

Cho, S.B., y Shimohara, K. (1997). *Emergence of Structure and Function in Evolutionary Modular Neural Networks*. En Proceedings of Fourth European Conference on Artificial Life, 197-204.

Christensen, A., y Dorigo, M. (2006). *Incremental Evolution of Robot Controllers for a Highly Integrated Task*. En International Conference on Simulation of adaptive Behavior, 473-484.

Clark, A. (1998). *Being There: Putting brain, body and world together again*. The MIT Press.

Cliff, D., Harvey, I., y Husbands, P. (1992). *Explorations in evolutionary robotics*. Adaptive Behavior, 2(1), 73-110.

Cliff, D., Harvey, I., y Husbands, P., (1992). *Incremental Evolution of Neural Network Architectures for Adaptive Behavior*. En Proceedings of the European Symposium on Artificial Neural Networks (ESANN'93), 39-44.

Coello, A.C., y Zacatenco, C.S.P. (2004) *Introducción a la Computación Evolutiva* (Notas de clase). Departamento de Ingeniería Eléctrica, Sección de computación, CINVESTAV-IPN. Ciudad de México, México.

Corucci, F. (2017). *Evolutionary Developmental Soft Robotics: towards adaptive and intelligence soft machines following nature's approach to design*. En Soft Robotics: Trends, Applications and Challenges, 111-116. Springer International Publishing.

Corucci, F., Cheney, N., Kriegman, S., Bongard, J., y Laschi, C. (2017). *Evolutionary Developmental Soft Robotics As a Framework to Study Intelligence and Adaptive Behavior in Animals and Plants*. Frontiers in Robotics and AI, (4), 34.

Cruz, V. (2012). *Una propuesta evolutiva para el desarrollo de comportamientos robóticos* (Tesis de maestría). Maestría en Inteligencia Artificial, Universidad Veracruzana. Xalapa, Veracruz, México.

Cruz, V., Montes-González, F., Mezura, E. y Santos J. (2012). *Robotic Behavior Implementation Using Two Different Differential Evolution Variants*. Proceedings Mexican International Conference on Artificial Intelligence (MICAI 2012), 212-226. Springer Berlin Heidelberg.

Darwin, C. (2009). *The origin of species by means of natural selection: or, the preservation of favored races in the struggle for life*.

Delleart, F., y Beer, R.D. (1996). *A Developmental Model for the Evolution of Complete Autonomous Agents*. En Proceedings of the fourth international conference on simulation of adaptive behavior, 393-401.

Di Fernando A., Calabretta, R., y Parisi, D. (2001). *Evolving Modular Architectures for Neural Networks*. Proceedings of the Sixth Neural Computation and Psychology Workshop: Evolution Learning and Development, 253-262. Springer Verlag.

Di Nardi, R., Togelius, J., Holland, O.E., y Lucas, S.M. (2006). *Evolution of Neural Network for Helicopter Control: Why Modularity Matters*. En IEEE congress on evolutionary Computation 2006, 1799-1806. IEEE.

Docieux, S., Bredeche, N., Mouret, J.B., y Eiben, A. (2015). *Evolutionary Robotics: what, why, and where to*. Frontiers in Robotics and Artificial Intelligence, 2(4).

Duarte, M., Oliveira, S.M., y Christensen, A.L. (2015). *Evolution of Hybrid Robotic Controllers for Complex Tasks*. Journal of Intelligent and Robotic Systems, 78 (3-4), 463.

Dürr, P., Floreano, D., y Mattussi, C. (2010). *Genetic Representation and Evolvability of Modular Neural Controllers*. IEEE Computational Intelligence Magazine, 5(3), 10-19.

Eiben, A. (2014). *Grand Challenges for evolutionary robotics*. Frontiers in Robotics and IA, 1(4).

Eiben, A.E., y Smith, J.E. (2016). *Towards the Evolution of Things*. ACM SIGEVOlution, 8(3), 3-6.

Faíña, A., Jacobsen, L.T., y Risi, S. (2017). *Automating the incremental Evolution of Controllers for Physical Robots*. Artificial Life.

Fernando, C.T., Szathmary, E., y Husbands, P. (2012). *Selectionist and evolutionary approaches to brain function: a critical appraisal*. Frontiers in computational neuroscience, 6, 24.

Ficci, S.G., Watson, R.A., y Pollack, J.A. (1999). *Embodied evolution: A response to challenges in Evolutionary Robotics*. En Proceedings of the eight European workshop on learning robots, 14-22.

Floreano, D., Dürr, P., y Mattiussi, C. (2008). *Neuroevolution: From architectures to learning*. Evolutionary Intelligence, 1(1), 47-62.

Floreano, D., Husbands, P., y Nolfi, S. (2008). *Evolutionary Robotics*. En Springer handbook of robotics, 1432-1451. Springer Berlin Heidelberg.

Floreando, D., y Mondada, F. (1996). *Evolution of Homing Navegation in a Real Mobile Robot*. IEEE transactions on Systems, Man, and Cybernetics Part B, 26(3),396-407.

Floreano, D., y Mondada, F. (1998). *Evolutionary Neurocontrollers for Autonomous Mobile Robots*. Neural Networks, 11(7), 1461-1478.

Floreano, D., y Urzelai, J. (2000). *Evolutionary Robotics: The Next Generation*. En Evolutionary Robotics III: AAIBooks, 231-226.

Fogel, D.B. (1993). *Applying evolutionary programming to selected traveling salesman problems*. Cybernetics and systems, 24(1), 27-36.

Frenken, K. (2006). *A fitness landscape approach to technological complexity, modularity, and vertical disintegration*. Structural Change and Economic Dynamics, 17(3), 288-305.

Frenken, K., Marengo, L., y Valente, M. (1999). *Interdependencies near-decomposibility and adaption*. Computational techniques for modeling learning in economics, 209-244.

Gage, A., y Murphy R.R. (2003). *Principles and experiences in usign legos to teach behavioral robotics*. En Frontiers in Education 2003. FIE 2003 33rd Annual, (2), F4E-23. IEEE.

Gecshwind, N., y Galaburda, A.M. (1985). *Cerebral Lateralization: Biological Mechanism, Association, and Pathology: I.A. hypotesis and a program for search*. Archives of nerology, 42(5), 428-459.

Goldberg, D (2000). *Genetic Algorithms in Search Optimization & Machine Learning*. E.U.A: Addisonn-Wisley.

Gravina, D., Liapis, A., y Yannakakis, G.N. (2017). *Coupling Novelty and Surprise Evolutionary Divergence*. En Proceedings of the Genetic and Evolutionary Computation Conference. 2017.

Gruau, F. (1994). *Automatic Definition of Modular Neural Networks*. Adaptive Behaviour, 3(2), 151-183.

Gurkiewicz, M., y Kornegreen, A. (2007). *A numerical approach to ion channel modelling using whole-cell voltage-clamp recordings and genetic algorithm*. PLoS Comput Biol, 3(8), e169.

Harlan, R.M., Levine, D.B., y McClarigan, S. (2001). *The Khepera Robot and the kRobot Class: A plataform for Introducing Robotics in the Undergraduate Curriculum*. ACM SIGCSE Bulletin, 33(1), 105-109.

Hart, W.E., Kammeyer, T.E., y Belew, R.K. (1994). *The role of development in genetic algorithms*. Doundations of genetic algorithms, 3, 315-332.

Harvey, I. (2000). *Robotics: Philosophy of Mind Using Screwdriver*. From Intelligent Robots to Artificial Life, 3, 207-230.

Harvey, I., Di Paolo, E., Wood, R., Quinn, M., y Tuci, E. (2005). *Evolutionary Robotics: A new scientific tool for studying cognition*. Artificial Life, 11(1-2), 79-98.

Harvey, I., Husbands, P., y Cliff, D. (1993). *Issues in Evolutionary Robotics. Proceedings of the second international conference on Simulation of Adaptive Behavior*. MIT Press.

Holland, J.H. (1992). *Genetic algorithms*. Scientific american, 267(1), 66-73.

Hopfield, J.J. (1982). *Neural networks and physical systems with emergent collective computational abilities*. Proceedings of the National Academy of Sciences, 79(8), 2554-2558.

Huizinga, J., Mouret, J.B., y Clune, J. (2016). *Does Aligning Phenotypic and Genotypic Modularity Improve the Evolution of Neural Networks?* En Proceedings of the 2016 on Genetic and Evolutionary Computation Conference, 125-132. ACM.

Husbands, P., Moioli, R., Shim, Y., Philippides, A., Vargas, P., y O'shea, M. (2014). *Evolutionary robotics and neuroscience*. The horizons of evolutionary roboticas, 17-63.

Husbands, P., Smith, T., Jakobi, N., y O'shea, M. (1998). *Better living through cheminestry: Evolving GasNets for Robot Control*. Connection Science, 10(3-4), 185-210.

Husbands, P., Smith, T., O'shea, M., Jakobi, N., Anderson J., y Philippides A. (1998). *Brains, Gases and Robots*. En Proceeding ICANN, (98).

Inamura, T., Inaba, M., e Inoue, H. (2000). *User adaption of human-robot interaction model based on Bayesian network and introspection of interaction experience*. En Proceedings of the 2000 IEE/JRS International Conference on Intelligent Robots and Systems, (3), 2139-2144.

Jacobs, E.G., Weiss, B., Makris, N., Whitefield-Gabrieli, S., Buka, S.L., Klibanski, A., y Goldstein, J.M. (2017). *Reorganization of functional networks in verbal working memory circuitry in early midlife: the impact of sex and menopausal status*. Cerebral Cortex, 27(5), 2857-2870.

Jacobs, R. y Jordan, M. (1992). *Computational Consequences of a bias toward short connections*. Journal of Cognitive Neuroscience, 4(4), 323-335.

Jacobs, R., Jordan, M., y Barto, A. (1991). *Task decomposition through competition in a modular connectionist architecture: The what and where vision task*. Cognitive Science, 15(2), 219-250.

Jakobi, N. (1997). *Evolutionary Robotics and the Radical Envelope of Noise Hypothesis*. Adaptive Behavior, 6(2), 325-368.

Jamil, M.F.A., Jalani, J., y Ahmad, A. (2016). *A new approach of active compliance control via fuzzy logic control for multifingered robot hand*. In First International Workshop on Pattern Recognition, (10011), id 1001111 9.

Janipireddy, S.B, Saeed, Z.A. y Saeed, M.Z. (2017). *Role of robotics in trauma and orthopedics*. International Journal of Research in Medical Sciences, 5(8), 3268-3272.

Jeason, F., y White, A. (2012). *Evolving Axonal Delay Neural Networks for Robot Control*. In Porceedings of the 14th annual conference on Genetic and Evolutionary Computation, 121-128. ACM.

Jelisavcic, M., De Carlo, M., Hupkes, E., Eustratiadis, P., Orlowski, J., Haasdijk, E., Auerbach, J.E., y Eiben, A.E. (2017). *Real-World Evolution of Robot Morphologies: A Proof of Concept*. Artificial Life.

Karafotias, G., Haasdijk, E., y Eiben, A. (2011). *An Algorithm for Distributed On-Line, On-Board Evolutionary Robotics*. Proceedings of the 13th annual conference on Genetic and Evolutionary Computation, 171-178.

Katz, B.(1969). *The release of neural transmitter substances*. Liverpool University Press.

Khare, V., Yao, X., Sendhoff, B., Jin, Y., y Wersing, H. (2005). *Co-evolutionary Modular Neural Network for Automatic Problem Decomposition*. En Proceedings of the 2005 IEEE congress on Evolutionary Computation, (3), 2691-2698.

Kimura, H., Akiyama, S., y Sakurama, K. (1999). *Realization of Dynamic Walking and Running of the Quadruped Using Neural Osillator*. Autonomous Robots, 7(3), 247-258.

Kodjabachian, J., y Meyer, J. (1995). *Evolution and Development of Control Architectures in Animats*. Robotics and Autonomous Systmes, 16(2-4), 161-182.

Kodjabachian, J., y Meyer, J. (1998). *Evolution and Developmental of Neural Controllers for locomotion, gradient-following, and obstacle-avoidance in artificial insects*. IEEE transactions on nerual networks, 9(5), 796-812.

Koza, J.R. (1994). *Genetic programming II: Automatic discovery of reusable subprograms*. Cambridge, MA, E.U.A.

König, L. (2015). *Complex Behavior in Evolutionary Robotics*. Walter de Gruyter GmbH & Co. Kg.

Kravos, M., y Malesic, I. (2017). *The Role of Glutamate Dehydrogenase Activity in Development of Neurodegenerative Disorders*. World Journal of Neuroscience, 7, 181-192.

Krichmar, J.L. (2012). *A Biological inspired action selection algorithm based on principles of neuromodulation*. The 2012 International Joint Conference in Neural Networks, 1-8.

Lara, B., Hülse, M., y Paseman, F. (2001). *Evolving Neuro-Modules and their Interface to Control Autonomous Robots*. Proceedings of The 5th World Multi-Conference on Systemics, Cybernetics and Informatics.

Lawton, K., Perry, W., Yamaguchi, A., y Zornik, E. (2017). *Motor Neurons Tune Premotor Activity in a Vertebrate Central Pattern Generator*. The Journal of Neuroscience, 37(12), 3264-3275.

Lei, W., Manier, H., Manier, M.A., y Wang, X. (2017). *A Hybrid Quantum Evolutionary Algorithm with Improved Decoding Shecme for a Robotic Flowshop Scheduling Problem*. Mathematical Problems in Engineering.

Loula, A., Gudwin, R., y Queiroz, J. (2010). *On the emergence of indexical and symbolic interpretation in artificial creatures, or What Is this I Hear?* En ALIFE 2010, 862-868.

Manoonpong, P., Pasemann, F., y Fischer, J. (2005). *Modular Neural Control for a Reactive Behavior of Walking Machines*. IEEE International Symposium on Computational Intelligence in Robotics and Automation 2005, 403-408.

Manoonpong, P., Paseman, F., y Roth, H. (2007). *Modular Reactive Neurocontrol for Biologically-Inspired Walking Machines*. The International Journal of Robotics Research, 26(3), 301-331.

Manoonpong, P., Paseman, F. y Wörgötter, F. (2008). *Sensor-driven neural control for omnidirectional locomotion and versatile reactive behaviors of walking machines*. Robotics and Autonomous Systems, 56(3), 265-288.

Marocco, D., Cangelosi, A., y Nolfi, S. (2003). *The emergence of communication in evolutionary robots*. Philosophical transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 361(1811), 2397-2421.

Marshall, J., Blank, D., y Meeden, L. (2004). *An Emergent ramework for Self-Motivation in Developmental Robotics*. Proceedings of the 3rd international conference on developmental and learning, (10).

Massera, G., Ferrauto, T., Gigliotta, O., y Nolfi, S. (2013). *FARSA: An open software tool for embodied cognitive science*. En ECAL, 538-545.

Mayer, H. (2011). *Evolution of Robotic Neurocontrollers with Intrinsic Noise and their Behavior in Noisy Environments*. En The 2011 International Joint Conference on Neural Networks (IJCNN), 1975-1980. IEEE.

McCulloch, W.S., y Pitts, W. (1943). *A logical calculus of the ideas immanent in nervous activity*. The bulletin of mathematical biophysics, 5(4), 115-133.

Meeden, L. (1996). *An incremental approach to developing intelligent neural network controller for robots*. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 26 (3), 474-485.

Meeden, L., y Kumar, D. (1998). *Trends in Evolutionary Robotics. Soft Computing for Intelligence Robotic Systems*. En Soft Computing for Intelligent Robotic Systems, 215-233. Physica-Verlag HD.

Mendel, J. (1995). *Fuzzy Logic Systems for Engineering: a tutorial*. Proceedings of the IEEE Transactions on Fuzzy Systems, 83(3), 345-377.

Meyer, J. (1998). *Evolutionary Approaches to Neural Control in Mobile Robots*. En Proceeding of the IEEE International Conference on Systems, Man and Cybernetics, (3), 2418-2423.

Michel, O. (1998). *Webots: Symbiosis between virtual and real mobile robots*. En International Conference on Virtual Works, 254-263. Springer, Berlin, Heidelberg.

Miglino, O., Lund, H., y Nolfi, S. (1995). *Evolving Mobile Robots in Simulated and Real Enviroments*. En Artificial Life, 2(4), 417-434.

Miglino, O., Nolfi, S., y Parisi, D. (1996). *Discontinuity in evolution: how different levels of organization imply pre-adaption*. En Proceedings Santa Fe Institute Studies in the Sciences of Complexity, (26), 399-416. Addison-Wesley Publishing Co.

Mitri, S., Floreano, D., y Keller, L. (2009). *The evolution of information suppression in communicating robots with conflict of interests*. Proceedings of the National Academy of Sciences, 106(37), 15786-15790.

Mitri, S., Floreano, D., y Keller, L. (2010). *Relatedness influences signal reliability in evolving robots*. Proceedings of The Royal Society of London B: Biological Science, (278), 378-383.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.C., Floreano, D., y Martinoli, A. (2009). *The e-puck robot, designed for education in engineering*. En Proceedings of the 9th conference on autonomous robot systems and competitions, (1), 59-65. IPCB: Instituto Politectino de Castelo Branco.

Mondada, F., Franzi, E., y Guignard, A. (1999). *The Development of Khepera*. Proceedings of the first International Khepera Workshop 1999, 7-14.

Montes-Gonzalez, F., y Aldana-Franco, F. (2011). *The evolution of signal communication for the e-puck Robot*. Advances in Artificial Intelligence, 466.477.

Montes-González, F., Ochoa, C., Marín, L., y Aguilar-Sánchez, J. (2010). *A Hybryd Approach in the Development of Behavior Based Robotics*. Computación y sistemas, 13(4), 385-397.

Moritz, R.L., y Mostaghim, S. (2017). *Heterogeneous Evolutionary Swarms with Partial Redundancy Solving Multi-Objective Tasks*. In international Conference of Evolutionary Multi-Criterion Optimization, 453-468. Springer, Cham.

Moscovitch, M., y Umiltá, C. (1991). *Conscious and nonconscious aspects of memory: a neuropsychological framework of modules and central system*. Perspectives on Cognitive Neurosciences, 299-365. Oxford University Press.

Mouret, J.B., y Chatzilygeroudis, K. (2017). *20 Years of Reality Gap: a few Thoughts about Simulators in Evolutionary Robotics*. En Workshop "Simulation in Evolutionary Robotics", Genetic and Evolutionary Computation Conference.

Mouret, J., y Doncieux, S. (2009). *Evolving modular neural-networks thorugh exptation*. En IEEE congress on Evolutionary Computation 2009 CEC'09, 1570-1577. IEEE.

Newman, M. (2006). *Modularity and community structure in networks*. Proceedings of the National Academy of Science, 103(23), 8577-8582.

Nolfi, S. (1997).*Using emergent modularity to develop control systems for mobile robots*. Adaptive Behavoir, 5(3-4), 343-363.

Nolfi, S. (2013). *Emergence of communication and language in evolving robots*. New Perspectives on the Origins Language, 144, 533-554.

Nolfi, S., Bongard, J., Husbands, P., y Floreano, D. (2016). *Evolutionary Robotics*. En Springer Handbook of Robotics, 2035-2068. Springer International Publishing. Springer-Verlag.

Nolfi, S., Elman, J., y Parisi, D. (1994). *Learning and Evolution in Neural Networks*. Adaptive Behavior, 3(1), 5-28.

Nolfi, S., y Floreano, D. (2000). *Evolutionary Robotics: The biology, intelligence and technology of self-organization machines*. The MIT Press.

Nolfi, S., Floreano, D., Miglino, O., y Mondada, F. (1994). *How to Evolve Autonomous Robots: Different approaches in evolutionary robotics*. Artificial Life IV: Proceedings of the 4th International Workshop on Artificial Life, 190-197. The MIT Press.

Nolfi, S., y Gigliotta, O. (2010). *Evorobot*. En Evolution of communication and language in embodied agents, 297-301. Springer Berlin Heidelberg.

Nordin, P., Banzhaf, W., y Brameier, M. (1998). *Evolution of a World Model for a Miniature Robot Using Genetic Programming*. Robotics and Automation Systems, 25(1-2), 105-116.

Norouzzadeh, M.S., y Clune, J. (2016). *Neuromodulation Improves the Evolution for Forward Models*. En Porceedings of the 2016 on Genetic and Evolutionary Computation Conference, 157-164. ACM.

NourAshrafoddin, N., Vahdat, A., y Ebadzadeh, M. (2007). *Automatic Design of Modular Neural Networks Using Genetic Programming*. Artificial Neural Networks-ICANN 2007, 788-798.

Ozdemir, A., Gauci, M., y GroB, R. (2017). *Shepherding With Robots That Do Not Compute*. En Proceedings of the 14th European Conference on Artificial Life. MIT Press.

Palacios-Leyva, R., Aldana-Franco, F., Lara-Guzmán, B., y Montes-González, F. (2017). *The impact of population composition for cooperation emergence in evolutionary robotics*. International Journal of Combinatorial Optimization Problems and Informatics.

Palacios-Leyva, R., Cruz, V., Montes-González, F., Rascón, L., y Santos, J. (2013). *Combination of reinforcement learning with evolution for automatically obtaining robot neural controllers*. En IEEE congress on Evolutionary Computation, 119-126. IEEE.

Parisi, D., Cecconi, F., y Nolfi, S. (1990). *Econets: Neural Networks that Learn in an Enviroment*. Network: Computation in Neural Systems, 1(2), 149-168.

Pasemann, F. (2013). *Self-regulating Neurons in the Sensorimotor Loop*. En International Work-Conference on Artificial Neural Networks, 481.491. Springer Berlin Heidelberg.

Pereda, A.E. (2014). *Electrical synapses and their functional interactions with chemical synapses*. Nature Reviews Neuroscience, 15(4), 250-263.

Pereira-Leal, J., y Teichmann, S. (2005). *Novel specificities emerge by stepwise duplication of functional modules*. Genome Research, 15(4), 552-559.

Philippides, A., Husbands, P., Smith, T., y O'shea, M. (2005). *Flexible Couplings: Diffusing Neuromodulators and Adaptive Robotics*. Artificial Life, 11(1-2), 139-160.

Potter, M., Meeden, L., y Schultz, A. (2001). *Heterogeneity in the Coevolved Behaviors of Mobile Robots: The Emergent of Specialists*. En Proceedings of International joint conference on artificial intelligence, (17), 1337-1343. Lawrence Erlbaum Associates LTD.

Preen, R.J., y Bull, L. (2017). *On Design Mining: Coevolution and Surrogate Models*. Artificial Life, 23(2), 186.

Praczyk, T. (2008). *Modularity networks in Assembler Encoding*. Computational Methods in Science and Technology, 14(1), 27-38.

Praczyk, T. (2013). *Modularity and Regularity in Neural Networks Produced with Assembler Code*. Computational Methods in Science and Technology, 19(3), 145-155.

Pratihar, D. (2003). *Evolutionary robotics: A review*. Sadhana, 28(6), 999-1009.

Reeder, J., Miguez, R., Sparks, J., Georgiopoulos, M., y Anagnostopoulos, G. (2008). *Interactively Evolved Modular Neural Networks for Game Agent Control*. En IEEE Symposium on Computational Intelligence and Games 2008, 167-174.

Reisinger, J. (2003). *An Overview of Modularity in Artificial Evolutionary Systems*. Proceedings of Parallel Problem Solving from Nature.

Reisinger J., Stanley, K., y Miikkulainen, R. (2004). *Evolving Reusable Neural Modules*. En Genetic and Evolutionary Computation Conference, 69-81. Springer Berlin Heidelberg.

Rice, S. (2000). *The evolution of developmental interactions: epistasis, canalization and integration*. En Epistasis and the Evolutionary Process, 82-98.

Rollins, A., y Schruum, J. (2017). *Balancing Selection Pressures, Multi Objetives, and Neural Modularity to Coevolve Cooperative Agent Behavior*. arXiv preprint arXiv:1703.08577..

Rosenblatt, F. (1958). *The perceptron: A probabilistic model for information storage and organization in the brain*. Psychological review, 65(6), 386.

Rubinacci, F., Ponticorvo, M., Gigliotta, O., y Miglino, O. (2017). *Breeding Robots to Learn How to Rule Complex Systems*. En Robotics in Education, 137-142. Srpinger International Publishing.

Russel, S. y Norving, P. (2004). *Inteligencia Artificial: Un enfoque moderno*. Prentice Hall.

Samiloglu, A.T., Cayirpunar, O., Gazi, V., y Koku, A.B. (2008). *An experimental set-up for multi-robot applications*. En International Workshop on Standarts and Common Platforms for Robotics (SCPR2008), 539-550.

Sanborn, A.N., y Chater, N. (2017). *The sampling brain*. Trends in Cognitive Sciences, 21(7), 492-493.

Santos, J. y Duro, R. (2005). *Evolución Artificial y Robótica Autónoma*. RA-MA. Madrid, España.

Scheper, K. Y., Tijmons, S., De Visser, C. C., y De Croon, G.C. (2016). *Behavior trees for evolutionary robotics*. Artificial life.

Schlessinger, E., Bentley, P. y Beau Lotto, R. (2006). *Modular Thinking: Evolving Modular Neural Networks for Visual Guidance of Agents*. En Proceedings of the 8th annual conference on Genetic and Evolutionary Computation, 215-222. ACM.

Scott-Philips, T., Blythe, R., Gardner, A., y West, H. (2012). *How do communication systems emerge?* Proceedings of The Royal Society of London B: Biological Sciences, 279(1735), 1943-1949.

Seiggelbaum, S.A., & Hudspeth, A.J. (2000). *Principles of neural science*. E.R. Kandel, J.H. Schwart, & T.M. Jessell (eds). New York: McGraw-Hill.

Sharkey, A. (2012). *Combining Artificial Neural Nets: essemble and modular multi-net systems*. Springer Science and Business Media.

Sharkey, N.E., y Ziemke, T. (1998). *A consideration of the biological and psychological foundations of autonomous robotics*. Connection Science, 10 (3-4), 361-391.

Silva, F., Duarte, M., Correia, L., Oliveira, S.M., y Christensen, A.L. (2016). *Open Issues in Evolutionary Robotics*. Evolutionary Computation, 24(2), 205-236.

Smith, T. y Philippides, A. (2000). *Nitric Oxide in Real and Artificial Neural Networks*. BT Technology Journal, 18(4), 140-149.

Spitzer, N.C. (2017). *Neurotransmitter Switching in the Developing and Adult Brain*. Annual Review of Neuroscience, (0).

Steels, L. (2003). *Evolving grounded communication for robots*. Trends in cognitive sciences, 7 (7), 308-312.

Stender, M., Yan, Y., Karayaka, H.B., Tay, P., y Adams, R. (2017). *Simulating micro-robots to find a point of interest under noise and with limited communication using Particle Swarm Optimization*. En SoutheastCon 2017, 1-8. IEEE.

Steyven, A., Hart, E., y Paechter, B. (2015). *The cost of communication: Environmental pressure and survivability in mEda*. En Proceedings of the Companion Publication of the 2015 on Annual Conference on Genetic and Evolutionary Computation Conference, 1239-1240. ACM.

Téllez, R., y Angulo, C. (2006). *Tactical modularity for evolutionary animats*. Frontiers in Artificial Intelligence and Applications, 146. 71-79.

Thangavelautham, J., y D'Eleuterio, G. (2004). *A Neuroevolutionary Approach to Emergent Task Decomposition*. Parallel Problem Solving from Nature-PSSN VIII, 991-1000. Springer Berlin/Heidelberg.

Thompson, A. (1997). *Artificial Evolution in the Physical World*. In Evolutionary robotics: From intelligent robots to artificial life, 101-125. AAI Books.

Togelius, J. (2004). *Evolution of a Subsumption Architecture Neurocontroller*. Journal of Intelligent and Fuzzy Systems, 15(1), 15-20.

Trianni, V. (2014). *Evolutionary Robotics: model or design?* Frontiers in Robotics and IA, 1, 13.

Trianni, V., y López-Ibáñez, M. (2015). *Advantages of Task-Specific Multi-Objective Optimization in Evolutionary Robotics*. PLoS one, 10(8), e0136406.

Vargas, P.A., Di Paolo, E.A., Harvey, I., y Husbands, P. (2014). *The horizons of evolutionary robotics*. MIT Press.

Wagner, G. (1996). *Homologues, Natural Kinds and the Evolution of Modularity*. Amer. Zoologist, 36(1), 36-43.

Wagner, G., Mezey, J., y Calabretta, R. (2001). *Natural Selection and the Origin of Modules. Modularity. Understanding the development and evolution of complex natural systems*. The MIT Press.

Watson, R., y Pollack, J. (2000). *Symbiotic Combination as an Alternative to Sexual Recombination in Genetic Algorithms*. En Parallel Problem Solving from Nature, VI, 425-434. Springer Berlin Heidelberg.

Webb, B. (2009). *Animals versus animats: or why not model the real iguana?* Adaptive Behavior, 17 (4), 269-286.

Weitzenfield, A. (2000). *From Schemas to Neural Networks: A Multi-level Modeling Approach to Biologically-Inspired Autonomous Robotic Systems*. Robotics and Autonomous Systems, 56(2), 177-197.

Whittington, J.C., y Bogacz, R. (2017). *An approximation of the error backpropagation algorithm in a predictive coding network with local hebbian synaptic plasticity*. Neural computation, 29(5), 1229-1262.

Woodford, G.W., Du Plessis, M.C., y Pretorius, C.J. (2017). *Concurrent controller and Simulator Neural Network development for a snake-like robot in Evolutionary Robotics*. Robotics and Autonomous Systems, 88, 37-50.

Yamauchi, B., y Beer, R. (1994). *Integrating Reactive, Sequential, and Learning Behavior Using Dynamical Neural Network*. From Animals to Animats, 3, 382-391.

Ziemke, T. (2000). *On 'Parts' and 'Wholes' of Adaptive Behavior: Functional Modularity and Diachronic Structure in Recurrent Neural Robot Controllers*. From Animals to Animats, 6, 171-180.

Ziemke, T., Bergfeldt, N., Buason, G., Susi, T., y Svensson, H. (2004). *Evolving Cognitive Scaffolding and Enviroment Adaption: A new research direction for Evolutionary Robotics*. Connection Science, 16(4), 339-350.

Ziemke, T., Carlsonn, J., y Bodén, M. (1999). *An Experimental Comparison of Weight Evolution in Neural Control Architectures for a 'Garbage-Collecting' Khepera Robot*. En Proceedings of the First International Khepera Workshop, 31-40.

Zimmer, M., y Doncieux, S. (2017). *Bootstrapping Q-Learning for Robotics form Neuro-Evolution Results*. IEEE Transactions on Cognitive and Developmental Systems.

Anexo 1.

A Basic Modular and Divided Control versus Heterogeneous Control: Separable Search Spaces Case.

A Basic Modular and Divided Control versus Heterogeneous Control: Separable Search Spaces Case.

Aldana-Franco F. y Montes-González F.

Universidad Veracruzana.

Abstarct.

This article shows a case about separability in search spaces applied to Evolutionary Robotics and the E-puck robot. Robot tasks evolved are: avoid collisions, finding targets and emitting sounds. Two architectures are tested: a homogeneous control and a divided and independent control. Each one is represented as a Neural Network and optimaized by Genetic Algorithms with mutation rate variations. Also, future work is presented.

Resumen.

En este artículo se presenta un caso de inseparabilidad de espacios de solución para un proceso evolutivo de un sistema de control para un robot e-puck. Las tareas que se evolucionan son: elusión de obstáculos, búsqueda de áreas objetivo y emisión de sonidos. Se optimizan dos tipos de arquitecturas de control, la primera es un control homogéneo en donde todas las tareas recaen en una misma red neuronal, mientras que la segunda es un control dividido independiente. Finalmente, se muestran la visión a futuro de esta investigación.

1 Introducción.

La Robótica Evolutiva es un campo que estudia los procesos evolutivos artificiales aplicados a la generación de estructuras de control y morfología de robots [10]. No obstante, la mayoría de las investigaciones en el área se enfocan en el desarrollo de los sistemas de control, típicamente caracterizados en forma de Redes Neuronales Artificiales ([23], [22] y [11]) y combinados con Algoritmos Genéticos [17].

En el área es muy común observar estructuras de control homogéneas que se encargan de la supervisión de varias tareas ([2], [13] y [21]). Sin embargo, uno de los principales problemas con este tipo de estructuras es que en ocasiones no es posible, para el algoritmo de Computación Evolutiva, encontrar soluciones lo suficientemente óptimas para dos o más funciones o tareas. Esta problemática puede traducirse en términos de la complejidad del espacio de búsqueda: entre mayor sea la cantidad de tareas a optimizar, mayor será la complejidad del espacio de búsqueda.

Existen múltiples enfoques para solucionar esta problemática que han sido propuestos por diversos grupos de investigación alrededor del mundo. La mayoría de estas soluciones son un intento por simplificar los espacios de soluciones posibles, que también puede ser visto como un problema de escalabilidad del diseño evolutivo de neurocontroladores.

Una de las opciones es tratarlo con evolución incremental. En este caso, durante el proceso evolutivo artificial, se cambia la función de calidad de acuerdo a los resultados obtenidos y al número de generaciones transcurridas, aumentando la complejidad de la tarea o comportamiento optimizado ([30] y [14]).

Pero este mecanismo de evolución artificial puede generar que después de encontrar una solución óptima en un punto del espacio de soluciones para una tarea de menor complejidad, el mismo algoritmo se alejará de ese punto debido al cambio del objetivo de optimización, inclusive manteniendo un término en la función de calidad que haga referencia a esa tarea menos compleja. Esto ocurriría cuando la tarea siguiente domine a las soluciones ya encontradas, y su punto óptimo se encuentre alejado en el hiperplano de búsqueda.

Una segunda opción para tratar de simplificar los espacios de búsqueda fue planteada por Togelius [29], quien propone usar la arquitectura de subsunción de Brooks [3] pero utilizando procesos evolutivos artificiales para la optimización de cada bloque de comportamientos.

Existen dos problemas potenciales en esta solución. Si uno de los bloques es una combinación de dos o más comportamientos, es posible que el problema se mantenga. El segundo consiste en que el proceso de la división de los bloques de comportamiento debe ser realizado por parte del programador, tendencia contraria a la corriente de robótica evolutiva en donde el programador debe influenciar lo menos posible, a través de su visión del entorno, al robot y procurando que este construya su sistema de control a partir de su experiencia con el ambiente donde se encuentra inmerso [16].

Una tercera opción es el uso de la modularidad. Se trata de un fenómeno que se encuentra frecuente en los seres vivos, particularmente en el ser humano, por ejemplo para Moschovitch y Umilta [24] el proceso cognitivo del ser humano está compuesto por una infinidad de módulos especializados.

1.1 Modularidad en robótica evolutiva.

La modularidad en el área es vista como una fuente generación de elementos robóticos complejos, a través de la aplicación de la estrategia “divide y vencerás” [28]. Una de las tantas aplicaciones potencial de la modularidad en el área es mostrada por Marbach y Ijspeert [18], quienes proponen coevolucionar diferentes partes de la morfología del robot como forma de fabricación y cuyo destino final puede ser la producción en masa de dichas partes.

Sobre la modularidad aplicada a los sistemas de control, Pasemann et al [27] señalan que estos pueden ser vistos como pequeños bloques de construcción cuya unión representa la posibilidad de generar comportamientos complejos.

En uno de los primeros trabajos sobre modularidad se reporta la evolución de redes neuronales artificiales como elementos de control de robots, en donde se optimizan no solo los pesos de las conexiones sinápticas, también los elementos de la arquitectura de la red a un alto costo computacional [15].

Una de las investigaciones más relevantes fue realizada por Nolfi [25], en donde se evalúan algunas arquitecturas de redes neuronales entre las que se encuentran una arquitectura con módulos externos (varias redes neuronales) y una con módulos internos (bloques de neuronas).

Los resultados reportados muestran que la arquitectura con mejor rendimiento es la que cuenta con módulos internos, mientras que la arquitectura con módulos externos estuvo en el medio del rendimiento respecto a las otras arquitecturas. En ese caso la arquitectura modular externa no reduce los espacios de solución porque la división de funciones se realiza a partir del estado de un sensor, y no a partir de las funciones que deben cumplir.

El mismo Nolfi afirma que existen tres niveles de trabajo en modularidad: Genético, Sistema Nervioso y Comportamientos. En el nivel del sistema de control o sistema nervioso se pueden distinguir dos divisiones claras: Modularidad Interna y Modularidad Externa.

Existen muchos trabajos derivados de la investigación de Nolfi sobre la modularidad interna, que combinan el nivel del sistema nervioso con el nivel genético, con especial atención al proceso evolutivo artificial y la definición genética de los módulos [31].

Aunque la modularidad externa no tuvo los resultados esperados, su utilidad puede ser replanteada y utilizarla como una herramienta para disminuir los espacios de búsqueda, reduciendo la carga de trabajo para el algoritmo de computación evolutiva encargado del proceso evolutivo artificial.

Una muestra que el enfoque tiene buenos resultados es el trabajo de Montes y Aldana [20], en donde se genera un sistema de comunicación basado en un control dividido que separa las tareas de movimiento y de emisión de sonidos aplicándolo a un grupo de robots e-puck.

En el presente trabajo se reportan datos obtenidos sobre el proceso evolutivo artificial de par de arquitecturas de de control optimizadas para un grupo de tareas cuyos espacios de soluciones resultan no convergentes o inseparables. La organización es la siguiente: En la sección 3 se abordan las características del robot y del simulador, la sección 4 detalla la configuración del experimento, la sección 5 muestra los resultados y la sección 6 las conclusiones y el trabajo futuro.

2 Robot y Simulador.

El robot utilizado para la parte experimental de esta investigación es el e-puck [19]. Se trata de un robot con ocho sensores infrarrojos que pueden ser utilizados para conocer la cercanía de objetos o para obtener mediciones sobre la presencia de luz ambiental. Está equipado con dos ruedas diferenciales operadas por motores a pasos capaces de moverlo a una velocidad máxima de 13cm/s. También posee tres micrófonos, una bocina, acelerómetro 3D, cámara CMOS 640x480 pixeles, anillo de leds, memoria RAM de 8KB y un procesador dsPIC30 @ 30MHz. Se utiliza la extensión del sensor de piso, que mide cambios en la tonalidad de la superficie que recorre el robot.

Este robot cuenta con modelos computacionales en diferentes simuladores como Evorobot* o player/stage, sin embargo se decide utilizar Webots [12] por su flexibilidad para crear diferentes entornos de prueba y sistemas de control. Este simulador, como muchos otros, no cuenta con modelos computacionales para los micrófonos y la bocina. Los modelos construidos y añadidos al simulador son los mismos repostados por Aldana [1]; en ese mismo trabajo se reporta la implementación y adaptación de un Algoritmo Genético combinado con el simulador.

3 Configuración experimental.

Se utiliza un algoritmo genético de representación binaria para evolucionar redes neuronales utilizadas como estructuras de control para robots. El algoritmo genético consta de cuatro operadores básicos: reproducción de los 50 mejores individuos de la generación utilizando como mecanismo de selección la ruleta, cruza en dos puntos fijos, mutación que varía entre 1% y 2% del total de los bits del total de individuos por generación, y elitismo para conservar al mejor individuo de cada generación.

La cantidad de individuos por generación es 100, mientras que el total de generaciones que componen el proceso evolutivo es 50. El algoritmo es programado en una plataforma de distribución libre para el lenguaje C, desde la cual se invoca al simulador para evaluar a los individuos. El tiempo de evaluación por individuo es de 10 vidas compuestas por 200 ciclos.

Las tareas que el robot debe realizar consisten en: alejarse de los obstáculos, acercarse a áreas con tonalidad diferente al del resto de la arena y emitir sonidos en ciertas situaciones deseadas (ver figura 1). Se utilizan dos tipos de control neuronal: homogéneo y dividido.

$$fitness_{movimineto_{general}} = (1 - maxifr) + (target * x) \qquad fitness_{emision\ de\ señales} = tx + voc$$

Figura 1. Ecuaciones de la función de calidad. (I) Optimiza la elusión de colisiones y la búsqueda de áreas objetivo, en donde: maxifr es el valor máximo de los sensores infrarrojos en el tiempo t, target es un valor binario (0, 1) que cuantifica cambios en la tonalidad del piso, y X un factor de multiplicación comúnmente igual a 1. (D) Optimiza la emisión de sonidos en ciertas circunstancias del robot, en donde: tx es un valor binario (-1, 1) que cuantifica si el robot emite sonidos o no, y voc es un valor binario (-1, 1) que determina si el robot emite la señal en las circunstancias deseadas. El proceso evolutivo del controlador homogéneo combina las dos funciones de calidad, mientras que el proceso del controlador dividido las utiliza por separado.

El control homogéneo consta de una red neuronal optimizada para controlar todas las funciones del robot, con tres capas configuradas de la siguiente manera: la capa de entrada con 11 neuronas con conexiones directas provenientes de los sensores, la capa media con 8 neuronas, y la capa de salida con 5 neuronas de las cuales 2 codifican con su función de salida el movimiento de los motores, mientras que la restantes sirven para la emisión de sonidos (ver figura 2).

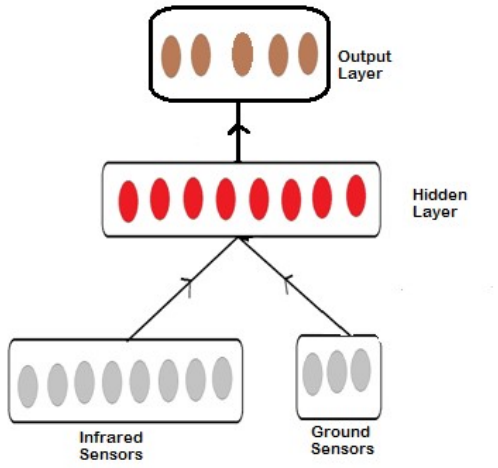


Figura 2. Control homogéneo: 11 neuronas en la capa de entrada (8 sensores infrarrojos y 3 sensores de piso), 8 neuronas en la capa intermedia y 5 neuronas en la capa de salida. 3 de las neuronas en la capa de salida codifica la emisión de sonidos, mientras que las 2 restantes codifican el movimiento del robot.

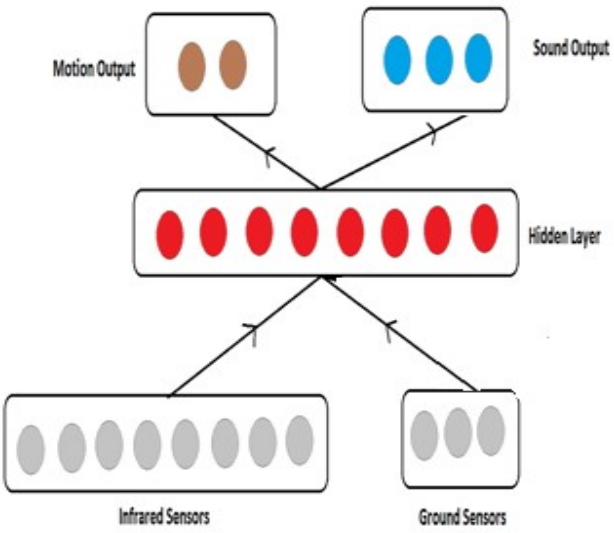


Figura 3. Control dividido: Ambas redes cuentan con 11 neuronas en la capa de entrada (8 sensores infrarrojos y 3 sensores de piso), 8 neuronas en la capa media con pesos independientes para cada estructura. En la capa de salida para la red de control motriz se tienen 2 neuronas que codifican el movimiento del robot. En la capa de salida de la red de control de emisión de sonidos se cuenta con 3 neuronas, cuyo mayor estado de activación determina la señal emitida.

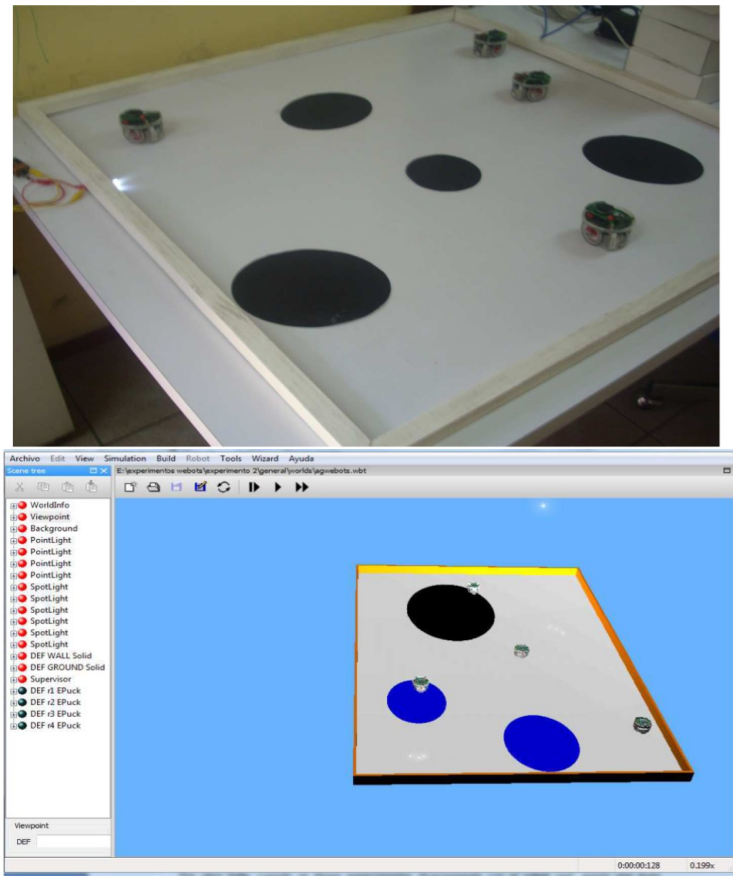


Figura 4. Configuración de la arena virtual (Inferior) con 4 robots, 4 paredes, 4 robots y 3 áreas objetivo de color diferente al del resto de la arena. El ambiente real (superior) trata de replicar el ambiente real.

En el control dividido se tiene dos estructuras neuronales independientes. Ambas reciben los mismos datos de los mismos sensores, cuentan con una capa oculta pero independiente entre ambas redes. La capa de salida de la primera red neuronal cuenta con dos neuronas que codifican el movimiento del robot, mientras que la otra estructura neuronal cuenta con tres neuronas que codifican la emisión de sonidos (ver figura 3). El proceso evolutivo del control dividido optimiza durante 50 generaciones el control de movimiento del robot, y durante otras 50 generaciones posteriores el control de emisión de sonido. La aptitud final de los individuos es la suma de ambos procesos evolutivos.

Para el proceso evolutivo del control dividido el porcentaje de mutación se mantiene constante en 1. En el caso del control homogéneo se generan dos procesos evolutivos diferentes: uno con mutación del 1% y otro con mutación del 2%.

La arena virtual de pruebas se encuentra delimitada por cuatro paredes, y contiene 3 círculos con de color oscuro, diferente al del resto. La cantidad de robots presentes en la arena durante cada prueba del proceso evolutivo es 4, sin embargo, al probar el individuo mejor calificado al final del proceso se realiza con solo 1. La finalidad de mantener 4 individuos en la arena durante el proceso

evolutivo es disminuir el tiempo de prueba, además de agregar obstáculos dinámicos al entorno. El ambiente real de prueba es una réplica del ambiente virtual (ver figura 4).

4 Resultados.

Al concluir los procesos evolutivos en sus diferentes versiones, para cada arquitectura y para sus 10 repeticiones, la aptitud de los individuos del control homogéneo de la mejor corrida es siempre menor al de los individuos del control dividido de la mejor corrida (ver figuras 5 y 6).

La combinación de los espacios de solución para la evolución del controlador homogéneo impide obtener buenos resultados. Es un fenómeno similar a lo reportado por Calabretta et al [5], en donde se cuantifica un proceso evolutivo para dos tareas, y que a diferencia de lo mostrado en este trabajo, una de las tareas resulta dominante.

Aquella interferencia reportada fue causada por mutaciones ventajosas y desventajosas. También se indica que el fenómeno ocurre cuando la tarea A ha sido optimizada y el proceso de optimización de la tarea B es deficiente.

Para Calabretta [4] la aparición de la modularidad no se reduce al aspecto evolutivo, sino que también depende de un proceso de aprendizaje. En términos prácticos, es posible combinar la evolución artificial con un proceso de aprendizaje automático cuyo resultado es la refinación de la búsqueda [26].

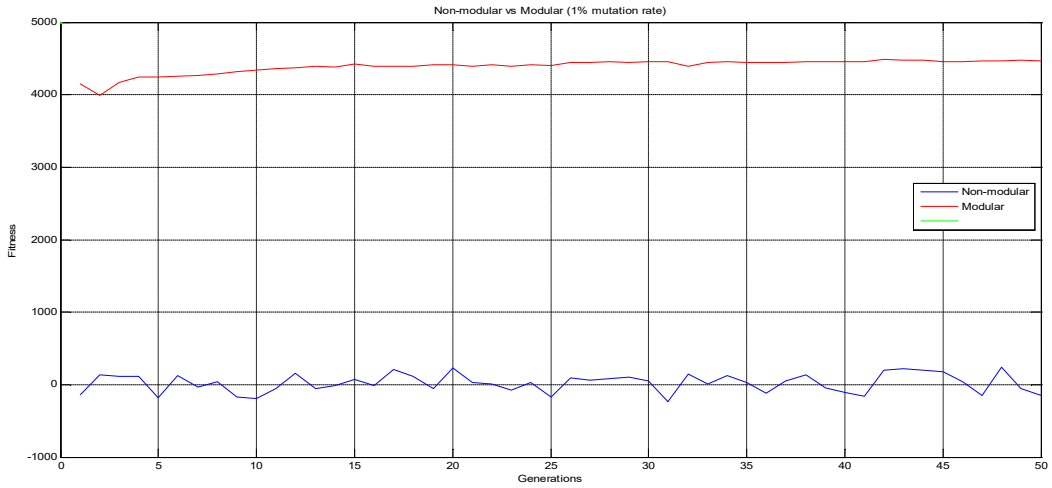


Figura 5. Aptitud del mejor individuo por cada generación del proceso evolutivo para la arquitectura homogénea y la dividida, con un porcentaje de mutación de 1 para ambas. El control dividido muestra tener una aptitud superior al control homogéneo en todas las generaciones.

En términos de los resultados presentados en este artículo, el efecto de una posible combinación de evolución y aprendizaje puede ser un pequeño aumento en la función de calidad del mejor individuo por generación de la arquitectura homogénea, pero de ninguna manera facilita la

separabilidad de las soluciones en el espacio combinado debido a que le proceso de aprendizaje solo refina las soluciones encontradas por el proceso evolutivo.

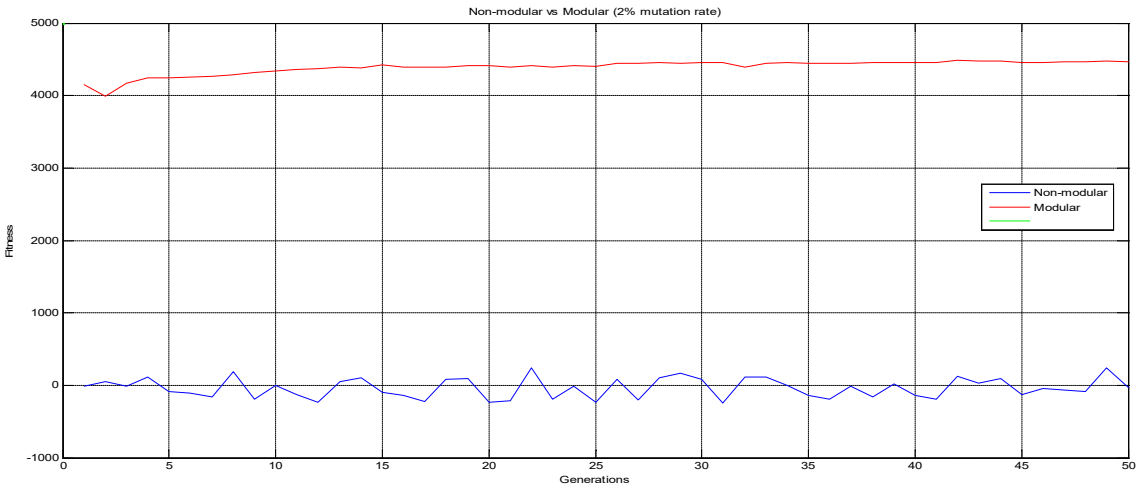


Figura 6. Aptitud del mejor individuo por cada generación del proceso evolutivo para la arquitectura homogénea y la dividida, con un porcentaje de mutación de 2 para el control homogéneo y 1 para el control dividido. El control homogéneo muestra una desventaja en cuanto a la aptitud de su mejor individuo por generación respecto al control dividido. El aumento en el porcentaje de mutación genera una gráfica con más picos, lo que representa los saltos en el espacio de soluciones que provoca el operador genético.

Por esto, el papel del aprendizaje en la modularidad se reduce a promover la especialización de los módulos. Por otro lado, la evolución artificial es el proceso a través del cual se explora el espacio de búsqueda: esta es una posición que complementa lo reportado por Calabretta et al [8].

Los resultados presentados en este trabajo confirman lo mostrado por Montes y Aldana [20] sobre la alternativa y funcionalidad de los sistemas de control dividido, que pueden ser vistos como un caso básico de modularidad en donde no existe competencia entre los módulos por el control de los recursos.

5 Conclusiones y trabajo futuro.

En suma a lo dicho por Calabretta et al [6], la interferencia genética no es solo un caso relacionado con las mutaciones de un proceso evolutivo artificial. Se trata de un problema relacionado con los espacios de solución y la capacidad que tiene el algoritmo de computación evolutiva para explorarlos.

Las mutaciones, en su forma básica, permiten que los algoritmos continúen explorando el espacio de soluciones, evitando que queden estancados en un máximo o en un mínimo local. Se ha mostrado que al variar el porcentaje de mutaciones, para el espacio de búsqueda presentado de la arquitectura homogénea, no existen cambios.

La mejora en la aptitud de los individuos del control dividido es una consecuencia de la división del espacio de soluciones que el algoritmo de computación evolutiva debe explorar. Es también una clara muestra que la modularidad interna [7] no es la única técnica que entrega buenos resultados, aunque con diferente modularidad.

La modularización externa de un sistema de control también puede ser exitosa, si la división de las tareas o comportamientos que debe controlar cada red neuronal se realiza con la finalidad de reducir el espacio de soluciones, como sucede en los resultados presentados.

En los próximos meses se plantea explorar la extensión de una arquitectura modular aplicada a la robótica evolutiva, cuya finalidad es disminuir los espacios de búsqueda tratando de apegarse a la corriente de pensamiento Evolutivo-Conexionista [9]. Para ello, se planea que la división de los módulos se realice a partir de los términos presentes en la función de calidad global y no a partir de la experiencia del programador.

La arquitectura seleccionada es un conjunto de redes neuronales multicapa vistas como bloques de construcción sin orden jerárquico (ver figura 7). La manera de contender por los recursos del robot es mediante neuronas especializadas de competitividad con una función similar a la presentada por Nolfi [25].

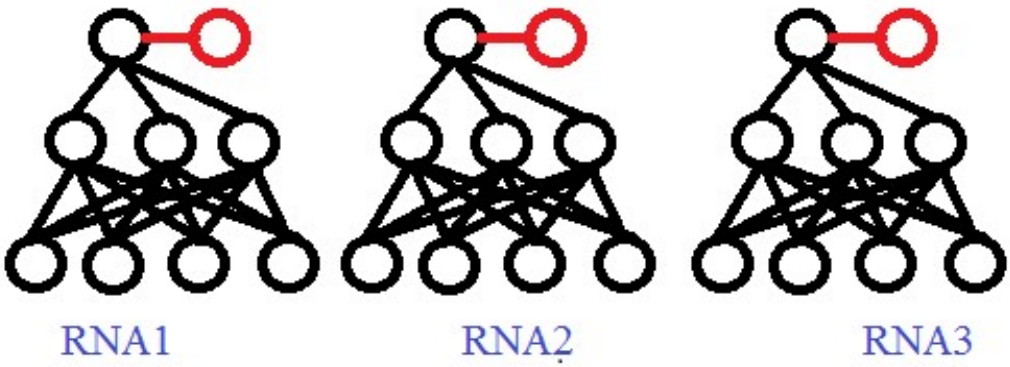


Figura 7. Arquitectura modular competitiva propuesta como trabajo futuro. Cada red neuronal es optimizada de acuerdo con una división de la función de calidad global, lo que permite disminuir la complejidad o dimensionalidad del espacio de soluciones mediante procesos evolutivos independientes. La neurona especial en la capa de salida tiene la función de otorgar el control de los recursos del robot, en caso de existir competitividad.

De esta manera, cada bloque representa un término de la función de calidad global y genera la simplificación en la dimensionalidad del espacio de búsqueda. Debe existir un subproceso dentro de la evolución artificial que permita conjuntar el funcionamiento de los individuos optimizando solo los pesos involucrados directamente con las neuronas de competitividad.

6 Referencias.

[1] Aldana F. Desarrollo de un Esquema Evolutivo Comunicativo para un Grupo de Robots. Tesis de maestría: Inteligencia Artificial, Universidad Veracruzana. Octubre 2011.

[2] Baldassarre G., Nolfi S. y Parisi D. Evolving Mobile Robots Able to Display Collective Behaviors. Artificial Life 9. Pp 255-267. 2003.

[3] Brooks R. A Robust Layered Control System for a Mobile Robot. IEE Journal on robotics and Automation: 14-23. 1986.

[4] Calabretta R. Genetic interference reduces the evolvability of modular and nonmodular visual neural networks.Philosophicla Transactions of The Royal Society B: Biological Sciences, 362. Pp. 403-413.

[5] Calabretta R., Di Fernando A., Wagner G. y Parisi D. What does it take to evolve behaviorally complex organisms?. BioSystems, 69. Pp. 245-262. 2003.

[6] Calabretta R., Nolfi S., Parisi D. y Wagner G. A case study of the evolution of modularity: towards a bridge between evolution biology, artificial life, neuro- and cognitive science. Proceedings of Sixth International Conference on Artificial Life. pp. 275-248.MIT Press. 1997.

[7] Calabretta R., Nolfi S., Parisi D. y Wagner G. Emergence of Functional Modularity in Robots. From Animals to Animats 5: 497-504. MIT Press. 1998.

[8] Calabretta R., Nolfi S., Parisi D. y Wagner G. Duplication of modules facilitates the evolution of functional specialization. Artificial Life 6: 69-84. 2000.

[9] Calabretta R. y Parisi D. Evolutionary Connectionism and Mind/Brain Modularity. Modularity: Understanding the development and evolution of complex natural systems. MIT Press. 2001.

[10] Clif D., Harvey I. y Husbands P. Explorations in Evolutionary Robotics. Adaptive Behavior 2. Pp. 73-110. 1993.

[11] Cruz V., Montes F., Mezura E. y Santos J. Robotic Behavior Implementation Using Two Different Differential Evolution Variants. Proceedings MICAI 2012. Pp. 216-226. México 2012.

[12] Cyberbotics, Webots: Commercial Mobile Robot Simulation Software.

[13] Floreano D. y Mondada F. Evolution of Homing Navegation in a Real Mobile Robot. IEEE Transactions on systems, man, and cybernetics, Part-B, 26. Pp. 396-407. 1996.

[14] Gomez F. y Miikkulainen R. Incremental Evolution of Complex General Behavior. 1996.

[15] Gruau F. Automatic Definition of Modular Neural Networks. Adaptative Behavior 2, 151-13. 1995.

[16] Harvey I., Di Paolo E., Tuci E., Wood R y Quinn M. Evolutionary Robotics: A New Scientific Tool for Studying Cognition. Artificial Life 11. Pp. 495-502. 2007.

[17] Husbands P. Harvey I., Cliff D., Thompson A. y Jakobi N. The Artificial Evolution of Control Systems. Proceedings of the second international conference on adaptive computing in engineering design and control. Pp. 50-61. 1996.

[18] Marbach D. y Ijspeert A. Coevolution of Configuraton and Control for homogenous Modular Robots. 2004.

[19] Mondada F. y Bonani M. The e-puck educational robot.

[20] Montes F. y Aldana F. The Evolution of Signal Communication or the e-puck Robot. Springer Advances in Artificial Intelligence, MICAI 2011. Pp. 466-477. Mexico 2011.

[21] Montes F., Aldana F., Marín L. y Ríos H. Evolving a Heterogenous Foranging Robot Task. Proceedings MICAI 2010. 2010.

[22] Montes F., Palacios R., Aldana F., Cruz V. The Coevolution of Behavior and Motivated Action Selection.

[23] Montes F., Velásquez J., Aldana F. y Palacios R. The Development of an Ultrasonic Turrent Extension for the Khepera Robot to Avoid Legged Objects.

[24] Moscovitch M. y Umiltá C. Conscious and nonconscious aspects of memory: a neuropsychological framework of modules and central system. Perspectives on Cognitive Neurosciences. Pp- 299-365. Oxford University Press. 1991.

[25] Nolfi S. Using emergent modularity to develop control systems for mobile robots. Adaptative Behavior 5: 343-363. 1996.

[26] Palacios R., Cruz V., Montes L., Rascón L. y Santos J. Combination of reinforcement learning with evolution for automatically obtaining robot neural controllers. IEEE Evolutionary Computation (CEC) 2013. Pp 119-126. México 2013.

[27] Pasemann F., Steinmetz U., Hülse M y Lara B. Robot Control and the Evolution of Modular Neurodynamics. Theory in Biosciences. Pp. 311-326. 2001.

[28] Santos J. y Duro R. Evolución Artificial y Robótica Autónoma. Alfaomega. Primera edición. México. 2005.

[29] Togelius J. Evolution of a Subsumption Architecture Neurocontroller. 2003.

[30] Urzelai J. y Floreano D. Incremental Evolution with Minimal Resource. IKW99. 1999.

[31] Wagner G., Mezey J. y Calabretta R. Natural Selection and the Origin of Modules. . Modularity: Understanding the development and evolution of complex natural systems. 2001.

Anexo 2.

The impact of population composition for cooperation emergence in evolutionary robotics.

The impact of population composition for cooperation emergence in evolutionary robotics

Rodrigo Palacios-Leyva[1], Fernando Aldana-Franco[1], Bruno Lara-Guzmán[2], and Fernando Montes-González[1]

1 Artificial Intelligence Research Center. University of Veracruz, Xalapa, Mexico.

2 Sciences Department, Autonomous University of Morelos, Cuernavaca, Mexico

fmontes@uv.mx

Abstract. *Communication is an important tool for evolutionary robotics. Some important aspects are the emergence of signals, the environment, and manipulation of social and evolutionary variables. In this paper we focus on social aspects related to exploration in poisoned and food environments. These aspects are as follows: a) intermediate levels of heterogeneity in population of evolutionary robots, and b) cooperation of robots for fitness contribution to regulate the emergence of communication signals. The FARSA simulator and Marxbot robot are used in order to optimize the weights of neural networks using a steady state genetic algorithm. A basic communication system is developed based on color LEDs and linear cameras.*

Keywords: Evolutionary Robotics, Neural Networks, Communication Signals, FARSA, Marxbot.

This work was supported by the Sistema Nacional de Investigadores (Exp. 30026 [FMMG], Exp. 200825 [BLG]). F. Aldana-Franco (Reg. 377475) and R. Palacios-Leyva (236198), both received a fellowship from CONACyT

1. Introduction

Evolutionary Robotics (ER) was created with the idea of developing morphological and control structures as a result of an artificial evolutionary process. In this field, an algorithm adjusts an artificial intelligent neural structure in order to control a robot. The most common representation for control systems in ER are Artificial Neural Networks (ANNs), and their weights are typically optimized by a Genetic Algorithm (Bongard [3]).

In nature communication is an important characteristic of individuals and communities. As for robots, communicative skills are used for sharing information such as personal, environmental, social, internal, and external states. Furthermore, communication is an important part of cooperative behavior (Sperati et al [19]).

Several methods allow communication to emerge in evolutionary robotics, through different channels, based on different sensors, e. g. color LEDs, sound, radiofrequency, and movement. The way a communication system is established depends on sensors and actuators with which robots are constituted (Marocco et al [9]).

In ER populations are evolved in order to solve a common task which can be accomplished using individual or group strategies. Additionally, populations can be either homogenous sharing the same chromosome information, or heterogeneous using information from different chromosomes. Then, in order to score their fitness, the population individuals have to be tested for some iterations in the simulator. Thus, in a homogenous population a population of *n* genetically identical individuals is built based in a single chromosome. In contrast, in a heterogeneous population all their individuals are not genetically identical. However, in this case the fitness function must have a mechanism to combine the score from different chromosomes within the population.

In their work Floreano et al. (2008) show that homogenous populations use communication systems to benefit the overall population. In contrast in heterogeneous populations communication systems emerge as a mechanism to drive away different individuals from food zones. Hence, is very important to understand the mechanisms that allow communication systems to emerge in populations of robots having different configurations (Steels [20]).

Additionally, the emergence of signal communication has an important relation with the conditions of the environment, control systems, and configuration of population (Montes-Gonzalez & Aldana-Franco [14]). Signals are emitted in situations where the possibility exists of collaboratively exploiting a common resource by the individuals of a homogenous population sharing useful information. Thus, signals are highly correlated with behavior which are prone to build up a basic communicative system. Environmental and social conditions both influence the emergence of signals (Nolfi [15], [16]).

In order to attract individuals when homogenous robot populations are evolved, in both poisoned and safe environments, signals mainly emerge towards the food zones. In the case of heterogeneous populations, signals emerge on non-favorable zones in order to keep away robots from feeding zones. Therefore, signals are associated with a conflict interest level in the population and finally egotistical behavior (Floreano et al. [5]).

In nature, the overall behavior of a community is regulated by those individuals with a high influence over the others (Mengistu, et. al [11]). This could be represented as a social advantage in the evolutionary population (Mitri, et. al [12]). For example, in hierarchical societies like ants (Triani et. al [20]) and honeybees (Zahadat & Shcmickl [24]), individuals that have a greater social value route the behavior of socially inferior individuals.

Experimental research (Floreano, et. al [6]) shows the importance of kin structure and the level of selection in the evolution to develop a stable cooperative communication. Also, this study demonstrates that cooperative communication and signaling can be evolved in groups of robots with simple artificial neural networks. Besides, the authors show that the evolutionary principles, ruling the evolution of social life, operate in groups of robotic agents mainly shaped by selection. This feature demonstrates that efficient groups of cooperative robots can be designed based on the transfer of knowledge carried out by artificial evolution.

Because communication has rooted social components, its own emergence is related to social factors in ER. Leaders in evolutionary societies must have an evolutionary advantage over the rest

of the population. Hence, they are able to regulate the emergence of signals in order to define its meaning and usefulness.

In this paper, we revise the impact of environmental variables in the emergence of cooperation in evolutionary robots. Additionally, we analyze to what extent the inclusion of communication leaders, in the population, affects the emergence of signal communication.

Two experiments were configured in order to prove that intermediate levels of heterogeneity produce intermediate level of conflict of interest, and the inclusion of robots with a high fitness contribution can guide the emergence of alerting signals. The purpose of the first experiment is to show that the level of heterogeneity is an important factor for the emergence of signal communication in robots. As for the second experiment we study how the emergence of signal communication is affected by the presence of those individuals with a large evolutionary contribution over the main population.

This paper is organized as follows. Firstly we introduce the MarXbot and FARSA Simulator. Secondly, in the methodology and materials section we provide the details of our two experiments. Thirdly, the results section shows the outcomes of the experiments. Next, in the discussion we expand the description of our findings. Finally, we draw some conclusions about our work.

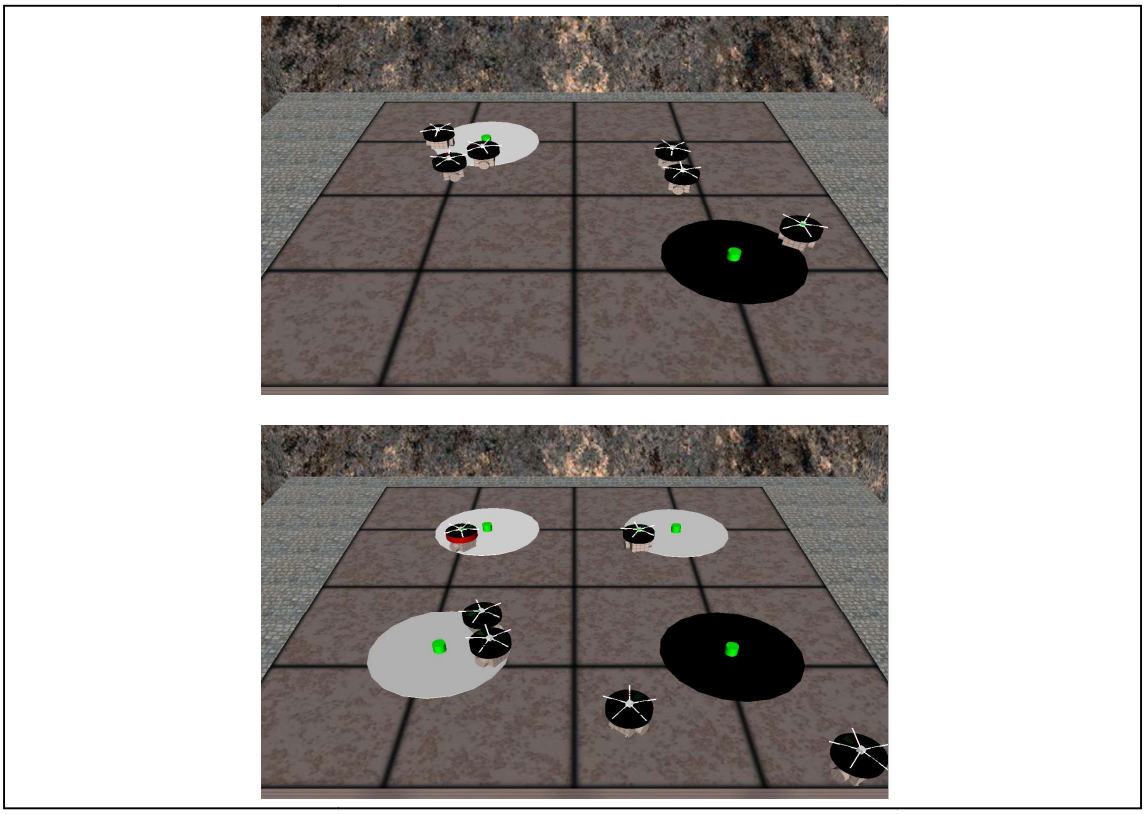


Figure 1. Examples of poisoned and food environments in FARSA

2. The MarXbot and FARSA Simulator

All experiments were conducted in a virtual world based in the FARSA simulator (Figure 1). Neural networks were used in order to control a group of MarXbot robots (Bonani et. al [1]). For evolution the simulator offers two versions of the Genetic Algorithm, the Elementary GA 1-1 (Davis [2]) and the Steady State (Shwehm [18]). In our case for the optimization of the weights on the ANNs we modified the steady state genetic algorithm. Additionally, FARSA is an open-source tool for experimental research on embodied cognitive science and adaptive behavior developed at the Institute of Cognitive Sciences and Technologies (ISTC-CNR) in Rome, Italy (Massera et. al, [10]).

FARSA provides a set of integrated libraries to create several components of embodied models and simulate their interactions with the environment in which they are situated. The graphical interface *Total99* allows the visualization of the experimental components and analysis of the cognitive processes derived from the interactions between the agent and the environment. Also, FARSA has a modular architecture based on three main concepts: 'components', 'configuration file', and 'plugins'.

The components are modules organized hierarchically which represent a process (an evolutionary process) or an object (neural network controller). Next, the configuration file is a text file where all the components, used in a particular project, and their parameter values are specified. This file can be modified through the graphical interface or directly from a text editor. Components facilitate the separation of the code in the main library from new code and provides an easy way to develop new experiments. Plugins can also be edited with the compiled code of existent components and new components developed by the users.

Evolution in FARSA starts with a random population, which is tested in a predefined environment. Next, a population of children is created, which are the result of the application of the mutation operator. Children run in the simulator and their adaptability-scores are compared with those of their parents. A selection of the worst parents is replaced with their best children and a new generation is spanned. In this algorithm the parameters to configure are: percentage of initial mutation, final mutation rate, and decreased rate of mutation. Moreover, the simulator (*worldsim*) is a complete library that allows the development of robots and environments. Thus, FARSA supports several robotic platforms including the marXbot robot (Figure 2).

Figure 2. The MarXbot Robotic Platform

The marXbot robot is a modular miniature mobile robot designed mainly for collective-robotic experiments (Bonani et. al, [1]). The base module has a combination of tracks and wheels (treels) that enables rough-terrain mobility and provides energy through a hot-swappable battery. The robot is equipped with a set of sensors like proximity sensors, 3D accelerometer, and a 3-axis gyroscope. This feature allows computing a rough estimate of the direction and distance of nearby robots. Also, the robot is provided with an attachment module that allows self-assembly with other similar robots. The marXbot is equipped with a distance scanner module and a main computer module provides an onboard linux-based operating system that allows the robot to build a 2D map of its surroundings.

3. Methodology and materials

In relation to the environment, we configured an adaptation of the poison and food experiment by Floreano, et. al [4]. A group of robots was placed in an arena without walls. Robots had to find and spent most of their time in the food zones and avoid poisoned areas. Food zones were represented by a white circular target area and poisoned zones with black circular target areas. A green cylinder was aggregated in the center of each target area as an extra visual reference (see Figure 1).

The steady state genetic algorithm was based on the mutation of initial individuals and the worst parents were replaced by their improved children. For every iteration in FARSA the mutation rate, of each generation, was decreased from an initial value of 50% to a minimal value of 1%. The fitness function rewarded with a positive value for each step that a robot spent in a food zone, and punished with negative values when the robot was next to a poisoned area. The complete evolutionary process was composed by 500 generations of 20 teams of six robots with random initial positions. Next, a repetition consists of a run of the complete evolutionary process. The rest of the parameters were as follows: selection of 20 individuals for reproduction; 1% decrease mutation rate; 1 trial of 300 steps; and then 10 trials for a team of six robots with 12 repetitions for each experimental group (6,000 generations in total).

Robots used 24 infrared sensors encoded in 8 average measures of 3 group sensors (FARSA allows grouping and fusing a certain number of infrared sensors). As for the ground sensors, 3 are employed for the detection of gray, black, and white colors. Also, the linear camera detects 5 segments of 72° of red, green, and blue components. As for the actuators, robots controlled 2 motors using only angle information (orientation-based), and the ring of LEDs flashing binary-coded red and blue colors. The neural controller was composed by 26 neurons at the input layer, 15 for the hidden layer, and 4 neurons at the output layer. In total 460 weights were optimized in a feed- forward structure.

As for the signal coding, it was implemented using a binary function that outputs 0 if the value is below 0.5 and 1 otherwise. There are two neurons for coding color signals, respectively representing red and blue colors. However, two more colors can be represented using a combination of the output of these two neurons. Therefore, codification in pairs is as follows, OFF, OFF = BLACK; ON, OFF = RED; OFF, ON = BLUE; ON, ON = PURPLE (Figure 3).

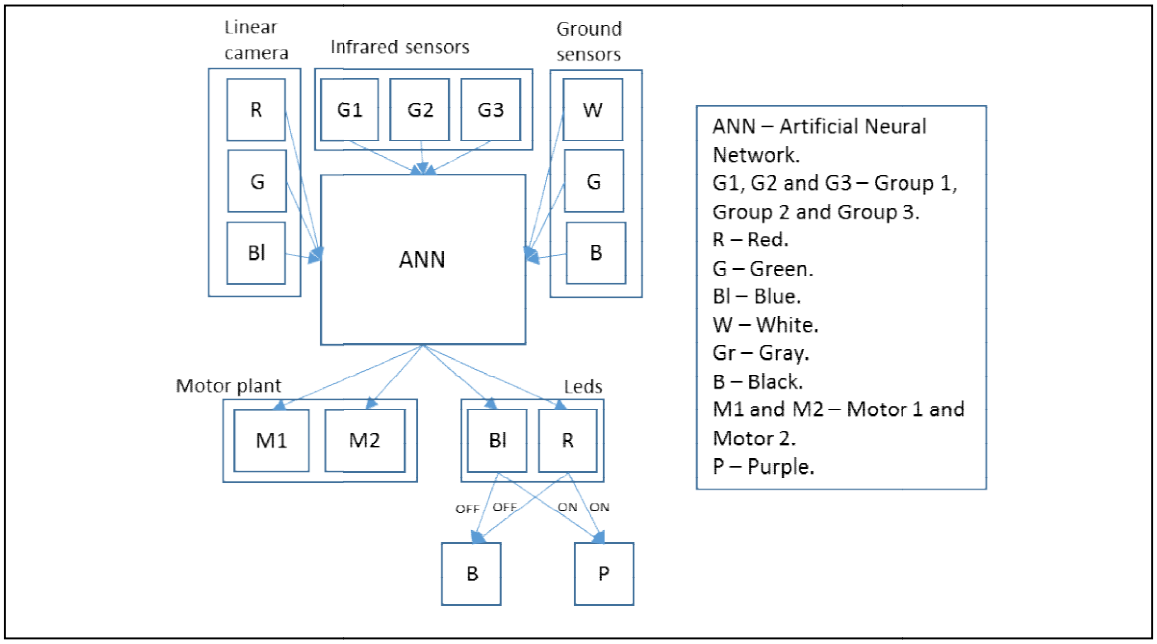


Figure 3. The neural architecture for our experiments

3.1 Experiment 1. Homogenous vs. heterogeneous populations

The aim of this first experiment was to find out to what extent the level of heterogeneity affects the emergence of communication signals in robots. Also to probe that the production of signals is related to the level of heterogeneity. Additionally, we anticipated that intermediate values of heterogeneity produced both intermediate levels of fitness and signal production. In relation to heterogeneity a 'clone' is a concept related to the way the population is integrated. For a homogenous population their individuals share the same chromosome (clones); on the contrary in a heterogeneous population their individual chromosomes are all different.

For the experiment we employed 4 experimental groups that represented different team configurations based on different variations of the heterogeneity level. The groups depending on their heterogeneity level were formed as follows: level 0 with identical chromosomes for each member of the team (control group); level 1 composed by 2 different chromosomes; level 2 composed by 3 different clones; and level 3 composed by a team of 6 different chromosomes.

Two dependent variables were measured: a) the fitness function level and b) the locations where robots emitted signals (food area, poisoned area, another robot presence, or no signals). In our experiments we used the fitness levels because communication emergence brings about additional benefits to robots by sharing food information and increasing their fitness (Marocco et al [9]). The average fitness score of the last two hundred best individuals of each repetition was used as the output variable. This in order to consider that the communication system was stable, evolutionary signals are considered stable when there is no change in signaling strategy after 200 generations of the evolutionary process. The statistical test used for finding differences between groups was a One way ANOVA on ranks (α=0.95) and a post-hoc Student-Newmann-Keuls with fitness level as the dependent variable and heterogeneity level as the independent variable. Signalization was

quantified with 4 locations, or situations, where commonly robots emit signals (food, poison, another robot and no signal). Signals were registered for 10 minutes through an instantaneous scan sampling of the best individuals in the last generation of all the repetitions in the experimental groups.

3.2 Experiment 2. The presence of communication leaders affects the emergence of signals

This second experiment was developed to demonstrate that the emergence of signal communication was affected by the presence of individuals with a large evolutionary contribution over populations of genetic clones. The experiment was integrated by 4 groups. Two free variables were used: the fitness contribution associated with the number of signals emitted in 2 levels (2 colors for a pair of individuals with a high fitness contribution and 4 colors of an equal fitness signal contribution); and the number of localized food target areas in 2 channels, i.e. 1 and 3 target areas.

The first group named 'control-group' was configured with 20 populations of 6 robots with an equal fitness contribution (+1,-1), 1 target food-area, and 1 poisoned-food area. The second group 'group-1' was composed by 6 individuals with the same contribution of fitness (+1,-1), 3 food zones, and 1 poisoned zone. The third 'group-2' was formed by 2 individuals with a high fitness contribution (+1,-1), 4 individuals with a low contribution (+0.5,-1), 1 food zone, and 1 poisoned zone. Finally, the fourth 'group-3' included 2 individuals with a high fitness contribution (+1,-1), 4 individuals with a low contribution (+0.5,-1), and 3 food zones with 1 poisoned zone.

The fitness function measured the last two hundred best individuals for each repetition, which reflected changes in behavior during evolution under experimental conditions. In order to compare the fitness levels of experimental groups, a two way ANOVA was used as statistical test (α=0.095). Next, it was complemented with a post-hoc of Student-Newmann-Keuls with $p<0.001$ and comparison of two factors: a dependent variable based on statistical tests using the individual fitness level; and an independent variable related to the number of food zones and to the presence or absence of individuals with high fitness contributions. Signals were measured for the last 10 minutes of the best individuals in 4 different places, or positions, where the robots emitted signals (food, poison, another robot, and no signal). A special case is when the robot is under a situation where the LED rapidly flashes one color and changes to another, e.g. the robot finds food, and emits a blue signal, afterwards finds a cylinder and emits a green signal.

4. Results

4.1 Experiment 1

A competitive race between species decreases the performance of individuals, and has an important effect on signal emergence. When the competitiveness level increases, between species, the altruism of individuals decreases (Waibel et al [22]). The One Way ANOVA on Ranks (Figure 4) showed that statistical differences were evident between the four experimental groups ($P=<0.001$, 3df, n=12). Furthermore, the Student-Newmann-Keuls Pos-Hoc test showed differences between all groups ($P<0.05$) related to variations in the fitness level produced by changes in the heterogeneity level.

Firstly, at the highest level of heterogeneity (six different individuals), signalization occurred 42.9% in poisoned zones, 14.2% in food zones, 28.7% in the presence of another robot, and 14.2% of the repetitions did not develop a communication system. Secondly, 25% of the signals for the 3-clones repetitions emerged in poisoned zones, 31.25% for black zones, 31.25% in the presence of another robot, and 12.5% repetitions did not develop any kind of communication. Thirdly, the 2-clones repetitions developed 21.4% of the signals at food zones, 28.5% at black zones, 14.2% in the presence of another robot, and 35.9% no communication system at all. On the other hand, homogeneous population signals were emitted most of the time at beneficial places (80%), and to a lesser extent in poisoned zones (20%).

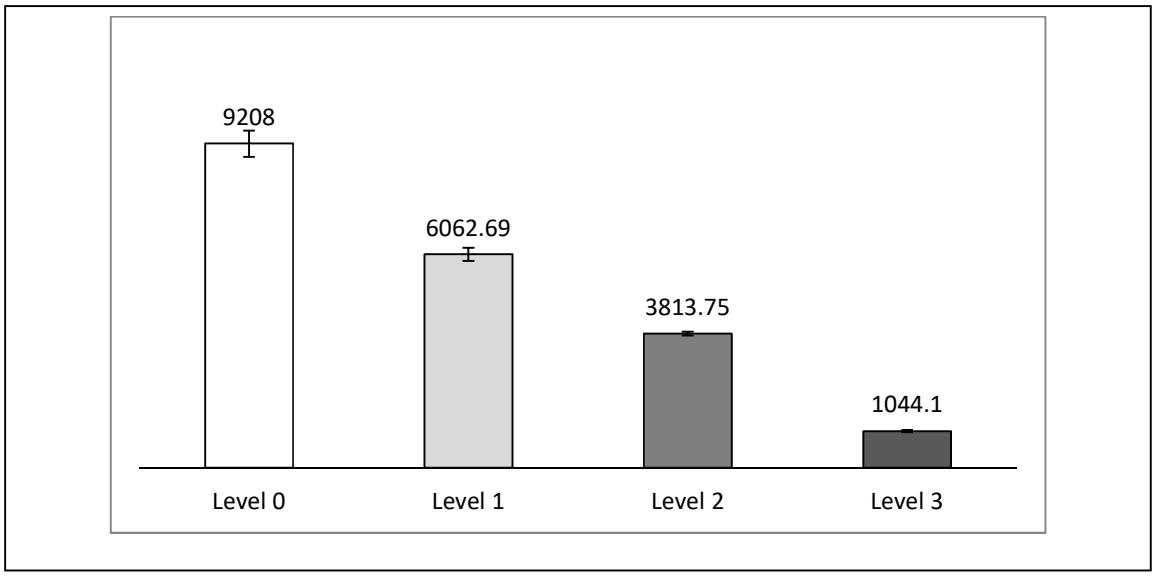


Figure 4. The average fitness and standard error of groups for experiment 1

4.2 Experiment 2

The Two Way ANOVA showed that there were statistical differences between the experimental groups ($P<0.001$, 44df residual, 47df total, a factor of interaction between factors $p=<0.001$ and $n=12$). The Student-Newmann-Keuls Pos-Hoc test showed differences between two factors: the presence of individuals with a high fitness contribution and the number of food zones ($P<0.05$). In consequence the best fitness level was scored by group-1 (see Figure 5).

In relation to the production of signals in the control group, robots produced them 80% of the time in beneficial places, 20% in poisoned zones. For group-1, signals emerged 90% of the time in food zones (25% at food zone 1, 40% at food zone 2, and 25% at food zone 3), 3.3% in poisoned zones, and 7.3% in the presence of another robot. As for group-2, 70% of the time signals emerged in food zones, 15% in poisoned zones, and 10% in the presence of another robot and 5% did not develop a communication system. Finally, in group-3 signals emerged 80% of the time in food zones, 10% in poisoned zones, and 10% in the presence of another robot. Robots in the control group employed an average of 2.2 signals; 3.1 signals in group-1, 1.1 for group-2, and 1.8 signals for group-3.

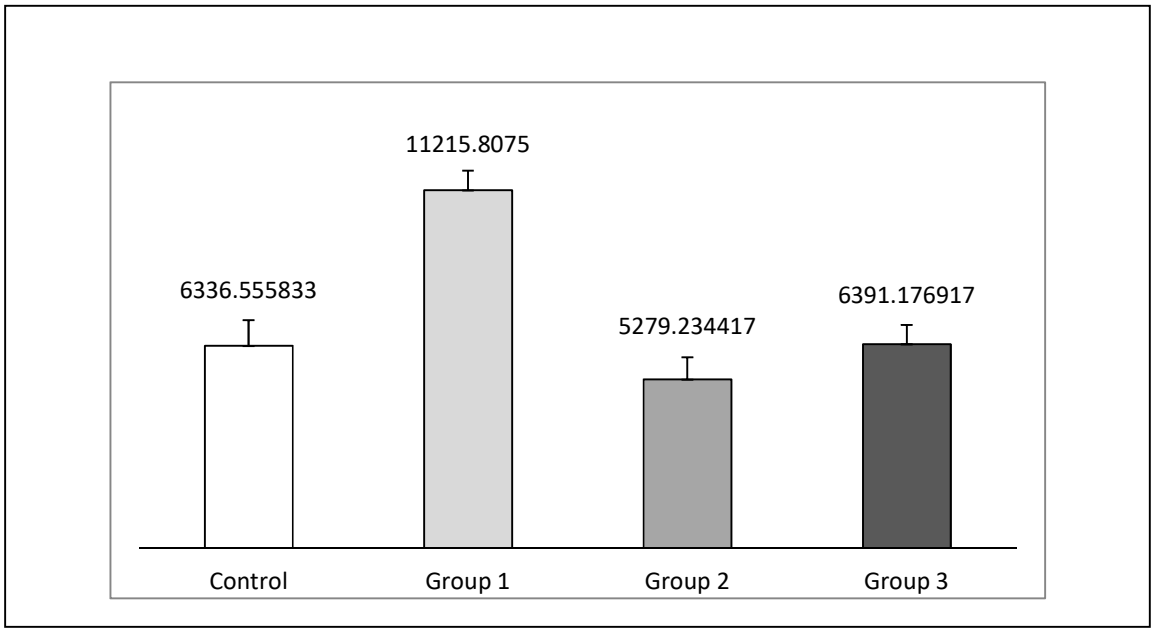


Figure 5. The average fitness and standard error of groups for experiment 2

5. Discussion

In the first experiment, as showed by Mitri et al. [11], the use of one clone facilitated that signals emerged when robots were in safe places like food zones. The results from these authors were confirmed in our experiments for the heterogeneous groups. Hence, when a group of robots was composed by two, three, or six clones; signals were used for attracting robots outside of the safe zones and then attract them to dangerous zones. The statistical tests confirmed that the manipulation of the independent variable (level of heterogeneity) has an effect on the fitness due to the emergence of non-altruistic behavior. In evolution this kind of behavior reduces the level of fitness and affects the emergence of communication signals. Intermediate levels of heterogeneity, confirmed a trend, which in turn results in a decrease of the fitness level.

In the case of groups composed by 2 and 3 clones the tendency of decreasing performance was maintained for intermediate values compared to groups of 1 and 6 clones. We can safely assume that the level of heterogeneity is inversely proportional to the fitness and level of altruism. Overall we found low performance at high levels of heterogeneity in the population. The fitness function level showed differences between groups as a result of different levels of individual contributions over evolution.

In the same experiment, after the total number of generations (6,000), high heterogeneity levels did not develop a stable communication system and this can be due to the complexity of the solution space. In contrast, for a homogenous population where competition between individuals does not exist, all replications produced a stable communication system. This could be related to the value of signals because if individuals did not produce them, a population would not solve the task and reach

high levels of fitness. Therefore, evolved communication systems in heterogeneous populations have low levels of altruism amongst identical individuals or species.

Here we observed two strategies, one is the emission of misleading signals and the other is the absence in the production of signals. In the first case a particular specie may produce a communication system to mislead individuals, of other species, from reaching the food zones which can be interpreted as a race competition between species. Eventually a strategy such as this may have better results because it uses one of its members to send competitors away while the rest of his teammates can visit empty food areas. The second strategy is related to the absence in production of signals which in turn facilitates the development of a common color identity of the group. Thus, in this case the failure to establish a communication system may be favorable for the population sharing the same color code and misleading for the others.

As for the second experiment we observed that robots in the control group emitted signals in food zones and before a collision. Furthermore, the availability of more than one food zone causes signals to emerge in a different way. This is the case of group-1 and group-3 where robots used a different signals for each located food zone. Our findings demonstrated that signal emergence depends on the utility and the complexity of the environment.

The use of fitness function level as a variable shows that experimental groups that have more than one food zone available showed better performance. Therefore, the availability of various food zones facilitates the emergence of complex communication system producing different signals having a significant associated lexical value. Furthermore, the use of fitness shows that there are statistical differences between experimental groups as a result of the manipulation of the independent variables, i.e. leaders and multiple targets. We found that there is a correlation between these variables due to the level of statistical significance of the test interaction between factors. Also, we observed that experimental groups are susceptible to manipulation of variables. For example, an increase in the number of food zones rises the individual fitness and produces more emergent signals. Additionally, the existence of communication leaders causes fitness decays in the other groups (the control-group and group 1 reach higher fitness levels than groups 2 and 3 that do not have a leader).

Cultural learning in a communication system helps to understand the usefulness in preserving or not a signal with its initial content. Furthermore, after some generations a signal can be developed and its original content be changed or even more can be suppressed. Leaders help to produce initial signals that can be imitated by others, which can be interpreted as a cultural learning that occurs during evolution. In our second experiment we observed that evolution guided from the team leaders helped to establish a stable communication system with an economy in the production of signals. Here, individuals with high fitness contributions produced a social benefit for the rest of the population which in turn tended to imitate signal behavior from the leaders. As many as 4 robots were able to produce blue and red signals in contrast to 2 robots which were able to produce only red signals. Despite the fact that leaders with two signals can be used to code 4 different signals, evolution optimized the selection of leaders that emitted only one color signal. Hence, after the evolutionary processes is finished, all robots emitted red signals in the food zones. We can summarize this behavior as an example of social learning (Heinerman et al [7]).

In relation to the number of available food zones, this has a major impact on the number of robots that are able to identify them. The group that develops stable identification signals related to food areas will have more individuals reaching them and in turn scoring high levels of fitness. However, the presence of leaders regulates the number of emergent signals because signals are mainly developed to point to the food zones and the rest of the population tend to mimic the behavior of the leaders. The population follows the leader even though the population has the possibility to produce additional communication signals.

In summary, in the first experiment we showed the importance of heterogeneity in a population and demonstrated that intermediate levels of heterogeneity produced intermediate levels of altruism. On the other hand in the second experiment we showed that fitness can be increased over evolution by combining social information with fitness information. This was observed in this experiment with group-2 where more fitted individuals acted as leaders even though they were not the strongest in the group. An increase in the availability of food zones make robots to choose zones accordingly to signal information from the leaders of the group. As a consequence, the rest of robots in the population associated different signals to the remaining the food zones. Also, in the second experiment we demonstrated that is possible to share a common communication signal set, during evolution, by the influence of the communication leaders.

6. Conclusions

For evolutionary robotics the heterogeneity level has an important effect on the emergence of communication systems. Evidence from our results confirms that intermediate levels of heterogeneity produce intermediate fitness values. The availability of reachable places is an important factor for developing communication systems. Furthermore, environmental manipulation favors the production of emergent signals with associated lexical values. Hence, the existence of communication leaders in the group, as a form of manipulation, accelerates the emergence of effective signal communication towards the final steps of evolution.

The utility of leaders for developing a stable communication system can be explained because the social composition of the group is a decisive factor for individual evolutionary development. At an initial stage gene modification is a secondary effect from behavior related to the sensory-motor systems. Later on, a more adapted sensory motor system produces more fitted individuals, which in turn are preserved and their genes are transmitted over the next generations. Hence, our results confirmed that social information has a great influence in gene expression, i.e. epigenetic changes, because sensorial systems induce neural transduction and next they are preserved by adaptation and natural selection (Robinson et al [17]).

Finally, as in nature where individuals with a particular advantage over the others regulate social and cultural processes; in our second experiment we observed that leaders in the groups are capable of producing signals with a clear lexical meaning.

References

[1] Bonani M., Longchamp V., Magnenat S., Retornaz P., Burnier D., Roulet G., et al. (2010). The marXbot a miniature mobile robot opening new perspectives for the collective-robotic research, IROS.

[2] Davis, L., Ed. (1991), Handbook of Genetic Algorithms, New York, NY: Van Nostrand Reinhold.

[3] Bongard J. (2013). Evolutionary robotics. Communications of the ACM 56(8). Pp. 74-85.

[4] Floreano D., Mitri S., Magnenat S. and Keller L. (2007). Evolutionary Conditions for the Emergence of Communication in Robots. Current Biology 17. Pp. 514-519.

[5] Floreano D., Mitri S., Pérez-Uribe A. and Keller L. (2008). Evolution of Altruistic Robots. Computational Intelligence: Research Frontiers. Pp. 232-248.

[6] Floreano D., Mitri S., Magnenat S., and Keller L. (2007). Evolutionary conditions for the Emergence of Communication in Robots, Current Biology 17. Pp. 1-6.

[7] Heinerman, J., Rango, M., & Eiben, A. E. (2015). Evolution, individual learning, and social learning in a swarm of real robots. In Computational Intelligence, 2015 IEEE Symposium Series on IEEE. Pp. 1055-1062

[8] Kernbach S. (2013). Handbook of Collective Robotics Fundamentals and Challenges. Pan Standford Publishing.

[9] Marocco D., Cangelosi A., and Nolfi S. (2003). The emergence f communication in evolutionary robots. Philosophical transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 361 (1811). Pp. 397-421.

[10] Massera M., Ferrauto T., Gigliotta O., and Nolfi S. (2013). Farsa: An open software tool for embodied cognitive science. In advances in Artificial Life, ECAL, volume 12, pages 538-545, 2013.

[11] Mengistu, H., Huizinga, J., Mouret, J. B., & Clune, J. (2016). The evolutionary origins of hierarchy. PLOS Comput Biol, 12(6).

[12] Mitri S., Floreano D. and Keller L. (2009). The evolution of information suppression in communicating robots with conflict of interests. Proceedings of the National Academy of Sciences, 106(37). Pp. 15786-15790.

[13] Mitri, S., Wischmann, S., Floreano, D., & Keller, L. (2013). Using robots to understand social behaviour. Biological Reviews, 88(1), Pp. 31-39.

[14] Montes-Gonzalez F. and Aldana-Franco F. (2011). The evolution of signal communication for the e-puck robot. In Proceedings of the 10th Mexican international conference on Advances in Artificial Intelligence. Springer Berlin Heidelberg. Pp. 466-477.

[15] Nolfi S. (1998). Evolutionary Robotics: Exploiting the full power of self-organization. Connection Science, 10 (3-4). Pp. 167-184. 1998.

[16] Nolfi S. (2013). Emergence of communication and language in evolving robots. New perspectives on the origins of language. Pp. 533-554.

[17] Robinson, G. E., Fernald, R. D., & Clayton, D. F. (2008). Genes and social behavior. Science, 322(5903), Pp. 896-900.

[18] Schwehm, M. (1996). Parallel population models for genetic algorithms. Universität Erlangen-Nürnberg, Pp. 2-8.

[19] Sperati, V., Trianni, V., & Nolfi, S. (2011). Self-organised path formation in a swarm of robots. Swarm Intelligence, 5(2). Pp. 97-119.

[20] Steels L. (2003). Evolving grounded communication for robots. Trends in cognitive sciences, 7 (7). Pp. 308-312.

[21] Trianni V., Labella T., y Dorigo M. (2004). Evolution of direct communication for a swarm-bot performing hole advance. Ant Colony Optimization and Swarm Intelligence. Springer Berlin Heidelberg. Pp. 131-140.

[22] Vargas P., Di Paolo A., Harvey I., y Husbands P. (2014). The Horizons of Evolutionary Robotics. The MIT Press.

[23] Waibel, M., Keller, L., & Floreano, D. (2009). Genetic team composition and level of selection in the evolution of cooperation. IEEE Transactions on Evolutionary Computation, 13(3). Pp. 648-660.

[24] Zahadat, P., & Schmickl, T. (2016). Division of labor in a swarm of autonomous underwater robots by improved partitioning social inhibition. Adaptive Behavior 24(2). Pp. 87-101.

Anexo 3.

Software de Optimización de Redes Neuronales Artificiales para Controladores Proporcionales (SORNA_CP) en un Sistema de control retroalimentado: Resporte Técnico.

Software de Optimización de Redes Neuronales Artificiales para Controladores Proporcionales (SORNA_CP) en un sistema de control retroalimentado: Reporte Técnico.

Fernando Aldana Franco.
fernando_aldana_franco@hotmail.com
Doctorado en Inteligencia Artificial. Universidad Veracruzana.
Mayo 2017.

Introducción.

En el campo de la Ingeniería, las herramientas de control son muy importantes. Esto debido a que permiten la automatización de procesos y sistemas. Por ello pueden representar sistemas de naturaleza física diversa como eléctricos, mecánicos, electrónicos, térmicos, entre otros. Dichos sistemas tienen diferentes componentes modelados con funciones matemáticas conocidas como función de transferencia, que es la relación entre la señal de salida y la señal de entrada del componente o un sistema (ver ecuación 1).

$$G(s) = \frac{C(s)}{E(s)} \qquad ec.1$$

Para obtener esta función es necesario transferir la ecuación que modela el sistema en el dominio del tiempo (t) al plano S, mediante la transformada de Laplace (ver figura 1). La transformada de Laplace sobre los términos diferenciales e integrales permite el cambio en los términos matemáticos asociados a la variación del tiempo. Esto permite simplificar el análisis matemático del sistema transformado (Chen [2]).

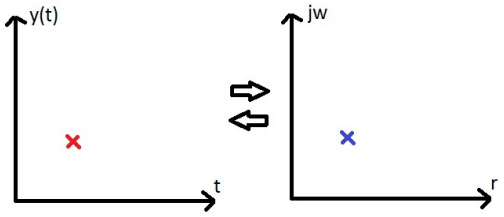


Figura 1. Transformación de una función en el dominio del tiempo al plano S mediante la transformada de Laplace.

El componente más importante de un sistema de control es la planta, que es el dispositivo que se quiere controlar (Shinskey [17]). Este componente recibe como entrada, en el caso de los sistemas retroalimentados, una señal de error. Dicha señal de error se define como la diferencia entre la señal de salida del sistema y la señal de entrada del mismo (ver figura 2 y ecuación 2).

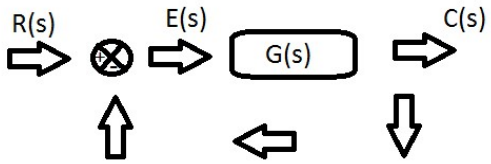


Figura 2. Prototipo de un sistema de control de lazo cerrado con una planta. R(s) es una señal de entrada, E(s) es la señal de error, C(s) es la salida del sistema y G(s) es la función de transferencia de la planta, todos en el dominio del plano S.

$$E(s) = R(s) - C(s) \qquad ec.2$$

Las funciones de transferencias se pueden descomponer en numerador y denominador. De tal manera que las raíces del numerador se conocen como ceros de las función. Mientras que las raíces del denominador son los polos del sistema. Estos últimos son muy importantes porque de ellos depende la estabilidad del mismo.

Los sistemas de control tienen una respuesta en el dominio del tiempo que puede ser descompuesto en dos componentes (Hernández-Guzmán [7]). La primera es la respuesta transitoria y que está compuesta por aquellos componentes que desaparecen en el tiempo, por ejemplo los componentes exponenciales negativos. La segunda es la respuesta estable que afecta al sistema desde el inicio hasta el final de la operación del mismo (no desaparece).

Un sistema de control se considera estable si su respuesta en el dominio del tiempo tiende a estabilizarse. Es decir, si los componentes transitorios del sistema tienden a un valor igual a cero después de un tiempo (exponenciales negativos) y la respuesta estacionaria se mantiene en un margen de operación deseable (Hellerstein et al [6]).

Considérese la respuesta de un sistema hipotético dado por la ec.3. La respuesta y(t) del sistema está compuesta por un término exponencial negativo y un término periódico de la función seno. La respuesta transitoria tiende a cero siempre y cuando el valor del exponencial sea negativo. En caso contrario, el valor de y(t) para ese término crece conforme el tiempo avanza. En cuyo caso, se considera al sistema como inestable.

$$y(t) = e^{-10t} + \sin(\omega t + \emptyset) \qquad ec.3$$

Como se mencionó previamente, la estabilidad de los sistemas de control depende de los polos de la función de transferencia en lazo cerrado del sistema. Si las raíces del denominador son positivas, la respuesta transitoria del sistema provocará inestabilidad.

Existen diferentes metodologías para indagar sobre su naturaleza estable de un sistema de control. Por ejemplo, en el dominio del tiempo el sistema puede ser analizado mediante el criterio de Routh-Hurwitz. Pero este criterio de estabilidad no brinda mayor información de las causas de la inestabilidad del sistema, es decir es concluyente pero no explicativo (Kuo [10]).

Otro de los criterios para conocer la estabilidad de los sistemas es el método del Lugar Geométrico de las Raíces. En él se analiza la estabilidad del sistema afectada por la variación de un parámetro conocido como ganancia (Ogata [13]). Con esta metodología sí es posible determinar la causa de la inestabilidad de los sistemas y proponer un valor para controlarla a través de la ganancia (K).

Por ejemplo, suponer que un sistema de control tiene la siguiente función de transferencia en lazo cerrado:

$$G(s) = k\left(\frac{s^2 + 2s + 4}{s^5 + 11.4s^4 + 39s^3 + 43.6s^2 + 24s + 0}\right) \qquad ec. 4$$

en donde K es la ganancia del sistema.

Si se aplica el método del lugar geométrico de las raíces (ver figura 3), se puede observar que el sistema es estable para valores de ganancia K entre 0 y 11.2. Estos valores de K del sistema pueden estar asociados con un componente adicional que es el controlador que se conecta a la planta (ver figura 4).

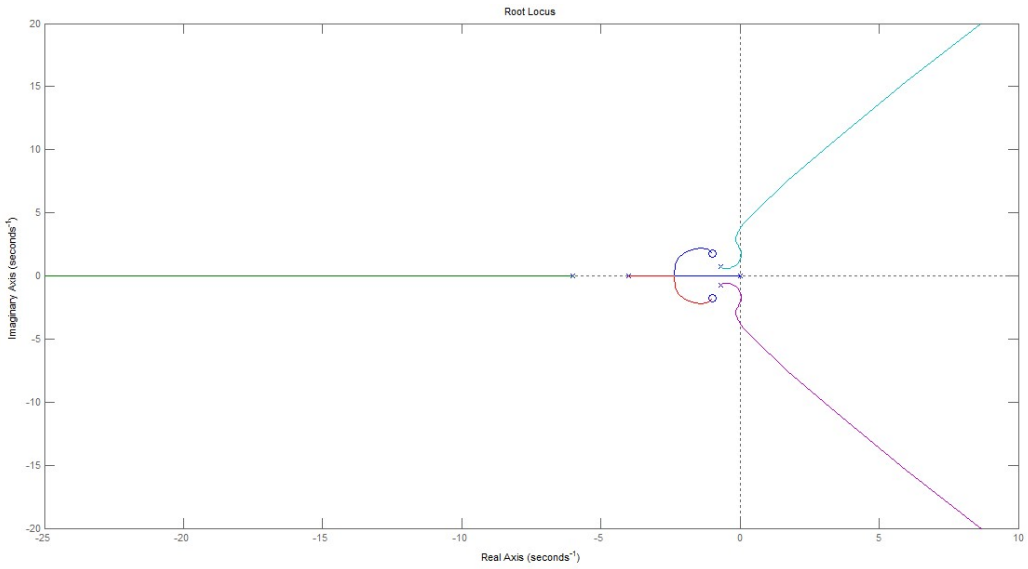


Figura 3. Gráfica del Lugar Geométrico de las Raíces de la función de transferencia de la ecuación 4. Muestra los polos (x) y los ceros (o) del sistema. Mientras los polos del sistema (x) se encuentren en el semiplano izquierdo del plano S, el sistema tendrá una salida que se estabilizará. Si los polos se encuentran en el semiplano derecho, el sistema tendrá una salida inestable. Si los polos se encuentran en el eje jW, el sistema tendrá una salida críticamente estable. La posición de los polos se mueven conforme el valor de la ganancia K se modifica.

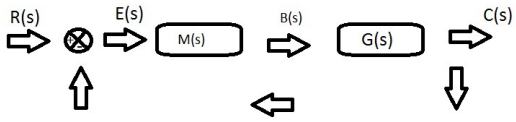


Figura 4. Sistema de control de lazo cerrado con una planta y un controlador. R(s) es una señal de entrada, E(s) es la señal de error, B(s) es la señal de salida del controlador, C(s) es la salida del sistema, M(s) es la función de transferencia del controlador y G(s) es la función de transferencia de la planta, todos en el dominio del plano S.

Un controlador es un dispositivo que tiene la función de corregir una señal de error mediante una acción de control para que la planta opere en los parámetros deseados. Los controladores pueden adicionar polos y ceros a la función, pero también pueden modificar la ganancia del sistema. En ese caso, el controlador se conoce como proporcional y el valor de su constante proporcional tiene efectos sobre los polos del sistema.

El efecto del controlador proporcional es mover los polos y los ceros en el plano S conforme aumenta o disminuye su valor. Así en muchas ocasiones es suficiente con la acción de un controlador proporcional para retirar la inestabilidad a un sistema de control. Por ello resulta muy importante estudiar este tipo de controladores en la teoría de control clásica.

Por otra parte, en los últimos años ha surgido y cobrado mucha importancia una rama del control que utiliza algunas técnicas de la Inteligencia Artificial para lograr los objetivos de control (Vassilyev et al [18]). Dicha rama de control es conocida como Control Inteligente (Galán et al [4]).

La primera herramienta en ser utilizada como controlador inteligente fue la Lógica Difusa (LD). Los controladores producidos se conocen como difusos. En ellos se asocian valores lingüísticos a las variables de entrada y salida (Chen y Pham [3]). Las variables están asociadas a una función de membrecía que indica el grado de pertenencia de un valor respecto a los conjuntos difusos. Y mediante reglas if-then se asocian los valores de entrada con valores de salida. Esto combinado con un motor de inferencia produce acciones de control que han probado ser tan eficientes como los controladores más robustos de otras ramas del control como el Proporcional- Integrador-Derivador (PID) (Ross [15]).

Dentro de las primeras aplicaciones de este tipo de sistemas de control se encuentran las lavadoras domésticas de ropa. En donde a partir de los valores determinados por el sistema y el usuario se determinaba el tiempo de lavado, la intensidad, así como otras variables de salida.

En la actualidad es posible utilizar a las Redes Neuronales Artificiales como herramienta de control, o a los algoritmos evolutivos como medio de optimización. En el primer caso se obtienen controladores neuronales. En el caso de los segundos se obtienen controladores evolutivos. También es posible combinar las diferentes técnicas a fin de obtener controladores neuro-difusos, difusos-evolutivos, neuro-evolutivos o neuro-difuso-evolutivos.

Una Red Neuronal Artificial (RNA) es un sistema de interconexiones y pesos que ponderan las entradas de los sistemas para encontrar una salida deseada (Hagan et al [5]). Se trata de un modelo que trata de reproducir la sinapsis eléctrica de las células del sistema nervioso de los seres

vivos (Kriesel [9]). Producen modelos que se encargan de discriminar situaciones diversas, principalmente aplicables a la clasificación y el control de sistemas (Jinn et al [8], Montes-González y Aldana-Franco [11], Santos [16]).

Las redes neuronales artificiales son herramientas muy poderosas por su capacidad a tolerar fallos y ruidos del entorno donde se encuentran inmersas (Nolfi et al [12]). En el caso de los sistemas de control representan la posibilidad de involucrar múltiples entradas y vincularlas con diferentes salidas. Esto es una característica que mediante la teoría de control clásica y sus herramientas no es posible lograr, ya que los sistemas son del tipo una entrada-una salida.

De esta manera las redes neuronales artificiales están a la altura de los sistemas de control creados a través de la teoría de control moderno. En ella se utilizan ecuaciones de estados y matrices para representar a los sistemas y analizarlos. Pero en el caso de las RNAs el análisis y diseño tiene un mayor alcance porque las relaciones entre entradas y salidas pueden ser tan complejas como el problema requiera.

Los algoritmos genéticos son herramientas computacionales que permiten la optimización de funciones matemáticas y la exploración multipuntos de un espacios de búsqueda. Al combinarlos con las redes neuronales artificiales, la capacidad de exploración del espacio de soluciones potenciales crece comparado con un algoritmo de aprendizaje tradicional como BackPropagation. El uso de este tipo de herramientas aumentan el potencial de las RNAs usadas como controladores.

Uno de las principales dificultades al trabajar con un neurocontrolador es que difícilmente se puede conocer el aporte a la función de transferencia del mismo. Principalmente porque el aporte depende en cierta parte por los datos que van modificándose de acuerdo con las condiciones del entorno del sistema. Una de las principales implicaciones de esto es que el análisis de control no puede desarrollarse de manera tradicional con herramientas de la teoría de control clásico. Por lo cual simplemente puede garantizarse que la señal de salida del controlador sea la adecuada ante diferentes condiciones del sistema.

Sin embargo Boza y Carvalho [1] desarrollaron un método con el cuál es posible conocer la función de transferencia de un controlador neuronal como si se tratara de un controlador proporcional. Con lo cual es posible observar si el sistema se estabilizará en términos de la acción de los pesos de las capas de salida y los pesos de las capas de entrada.

En este método se consideran a las redes neuronales como un elemento más de un sistema de control tradicional. Como consecuencia se aplican a las diferentes unidades neuronales los preceptos del álgebra de bloques (Rodríguez [14]). Con lo cual una neurona que tiene más de una conexión de entrada y pesos asociados a ellos se consideran puntos de suma. El efecto de múltiples capas son considerados conexiones seriales de bloques. Es importante mencionar que la función de transferencia del controlador es una función matemática que relaciona mediante un cociente la salida y la entrada del mismo. En ningún momento se debe confundir con la función de

transferencia interna de la red, la cual mapea el valor de la función NET con un valor de salida específico.

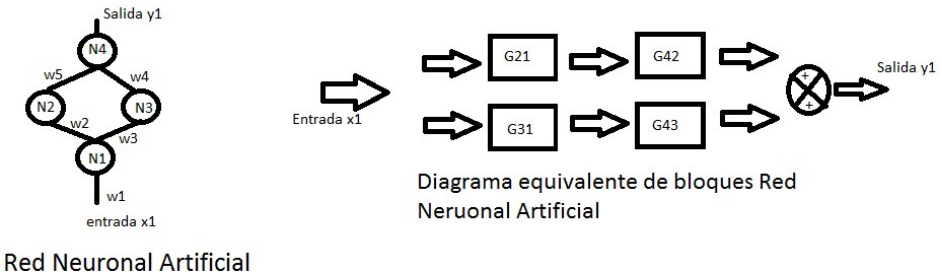


Figura 5. Red neuronal artificial de tres capas y el diagrama equivalente de la red como sistema de control.

La función de transferencia de cada unidad se representa por el aporte del peso de la conexión de esa neurona a la capa superior entre la sumatoria de los pesos que llegan a la misma neurona provenientes de la capa inferior (ver ecuación 5).

$$G_a^b = \frac{|W_a^b|}{\sum_{a=1}^{n} W_n^b} \qquad ec.5$$

en donde a es la conexión de la neurona de la capa previa y b es la neurona de la capa actual o neurona que recibe el peso.

De esa manera la función de transferencia de la red neuronal artificial utilizada como controlador proporcional aplica las reglas del álgebra de bloque en donde dos bloques conectados en serie multiplican su función de transferencia, y dos bloques en paralelo unidos por un punto de suma con dos símbolos positivos es la suma de las funciones de transferencia de los bloques.

Esta metodología que calcula el aporte de la red neuronal a la función de transferencia como si se tratara de un controlador proporcional no se encuentra incorporada en los principales programas de análisis matemático utilizados en el mundo de los sistemas de control. En donde en el mejor de los casos, es posible probar el controlador simulado conectado a la planta para verificar que las condiciones del entorno produzcan las salidas deseadas. Como es el caso de Matlab y sus toolbox Simulink y Neural Network Toolbox.

Debido a ello se creó un Software de Optimización de Redes Neuronales Artificiales (SORNA) que utiliza un algoritmo genético generacional para ajustar los pesos de la red, y un mecanismo computacional que calcula la función de transferencia de dicho controlador en su forma proporcional. El programa funciona en una interfaz producida en Matlab, pero el núcleo de procesamiento principal ocurre en lenguaje C++. El programa no requiere que el usuario cuente con una licencia de Matlab, ya que se distribuye con un instalador que incluye todo lo necesario para su funcionamiento (Matlab Compiler Runtime). Finalmente el sistema cuenta con un componente para la obtención de un análisis del Lugar Geométrico de las Raíces que el usuario puede utilizar para verificar que el valor de la ganancia del sistema se encuentre en los márgenes

de operaciones ideales. Adicionalmente se creó un comprobador auxiliar para conocer la salida de la red optimizada dado un ejemplo o instancia. En los siguientes apartados de este documento se muestra los componentes de construcción, sus principales características funcionamiento de ambos programas.

Construcción y características.

El software de optimización (SORNA) combina el poder de procesamiento de datos en lenguaje C++ y el procesamiento matemático y de gráficos en Matlab. La interfaz fue construida a partir del manejador de interfaces de usuario o GUIDE de Matlab.

La interfaz permite la configuración de todos los componentes necesarios por parte del usuario, y realiza una llamada un programa escrito en lenguaje C++. Este programa es un algoritmo genético generacional en donde el número de individuos y generaciones son definidos por el usuario. La representación del genotipo es con números binarios de nueve bits. Cada conjunto de bits representa a un peso de la red neuronal que se mapea a un valor de -1.0 a 1.0, con una resolución de 0.00390625. En cada cadena de representación de los pesos, los ocho bits menos significativos corresponden al valor numérico, mientras que el bit más significativo es el signo. En donde el 0 corresponde a un valor positivo y 1 que corresponde a un valor negativo.

El algoritmo genético está basado en los operadores: selección, cruza, mutación, elitismo y sustitución. El operador de selección opera por torneo. Es decir, se seleccionan un par de individuos candidatos potenciales al azar entre el conjunto de padres. Se comparan sus puntajes de aptitud y el mejor se utiliza para recombinar su genotipo y crear los individuos de la siguiente generación.

El operador de cruza utiliza un punto fijo que se calcula al inicio del algoritmo y es la mitad de la cadena de bits que representan a los genotipos de los individuos. Para generar a un hijo se utilizan dos padres tomados mediante el operador de selección.

La mutación es un valor por default del 2% del total de bits en la representación cromosómica de la población. Pero puede ser modificada por el usuario en la interfaz a través de un coeficiente de mutación. Al tener representación binaria en el genotipo de los individuos, el operador aplica la operación binaria de negación con la finalidad de intercambiar el valor del bit seleccionado aleatoriamente.

El elitismo busca y conserva al mejor individuo de la generación presente. Esto con la intención de salvaguardar un óptimos locales. Finalmente la sustitución elige hasta un 20% de individuos de la generación previa elegidos aleatoriamente para conservarlos en la nueva generación. El resto de los individuos corresponden a los hijos generados y al individuo salvaguardado mediante el operador de elitismo.

Existen tres versiones de redes neuronales con arquitectura feedforward que pueden ser optimizadas como controladores neuronales: red sin capa media, red con capa media, red con

capa media y conexiones directas entre las unidades de entrada y las de salida (ver figura 6). Todas las redes utilizan una función de salida senoidal.

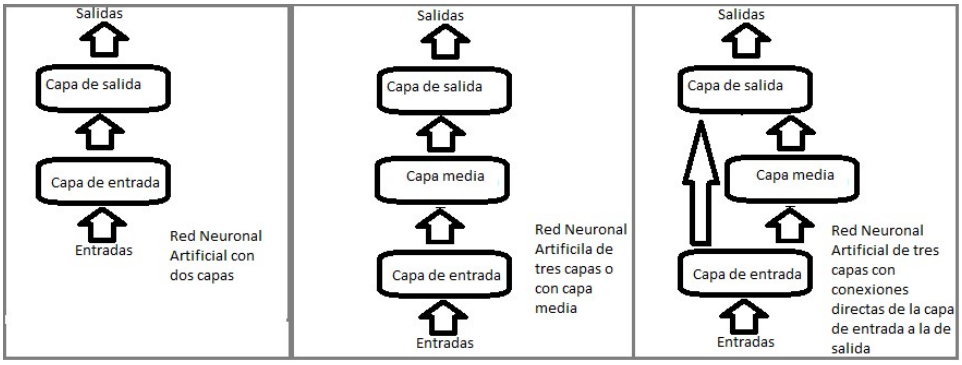


Figura 6. Configuraciones de redes neuronales artificiales que son posibles de optimizar mediante SORNA.

Una vez optimizada la red neuronal seleccionada, el programa calcula el aporte de los pesos en forma de función de transferencia para cada salida. Dicho valor se imprime en la interfaz y corresponde a un equivalente a un controlador proporcional aplicable a una función de transferencia.

Dentro de las funciones de la interfaz creada con Matlab, se grafica como salida las curvas de evolución del sistema mostrando el mejor individuo, el promedio del puntaje de calidad y al peor de los individuos en cada generación.

El software funciona a través de la configuración de archivos que son leídos tanto por la interfaz como por el programa en C. Además de un archivo llamado "base.txt" que debe ser proporcionado por el usuario y que contiene la base de datos para la optimización. El programa en C imprime archivos con los datos del paisaje de calidad con los mejores individuos, promedio de individuos y peor individuos por cada generación. También devuelve el valor de la función de transferencia de la red neuronal artificial equivalente a un controlador del tipo proporcional.

Adicionalmente es posible imprimir un gráfico dentro de la interfaz del lugar geométrico de las raíces de un sistema de control. Con él, dado una función de transferencia el usuario está en la posibilidad de verificar si el valor del controlador neuronal proporcional cumple con los requisitos de estabilidad del sistema o no. Esta función está basada en el comando rlocus de Matlab utilizado para el análisis del Lugar Geométrico de las Raíces.

Finalmente la interfaz adicional (comprobador SORNA) tiene la finalidad de comprobar el funcionamiento de la red con los pesos optimizados por el SORNA. Esta interfaz es un comprobador asociado a SORNA creado con la misma combinación del optimizador (Matlab y C). En él es posible probar tres tipos de redes neuronales artificiales: dos capas, con capa medio (tres capas), con conexiones directas entre la capa de entrada y salida además de la capa media. Este software auxiliar devuelve el valor de salida de la red, el valor límite esperado según la clase y si la clasificación de la instancia presentada es correcta o no.

Su funcionamiento recae en dos archivos de texto. El primero es el archivo de salida generado por el software de optimización y que contiene los pesos del mejor individuos. El segundo es un archivo con los datos de la configuración de la red, los atributos y las clase. Este par de archivos son leídos por un programa en C que calcula la salida de la red bajo los parámetros que utilizan las redes neuronales en el SORNA.

Funcionamiento.

Tanto SORNA como el comprobador SORNA se encuentran dentro de un par de archivos pkg. El usuario debe descomprimir y seguir el procedimiento de instalación. De contar con una instalación de Matlab o el Matlab Compile Runtime, es necesario seleccionar una carpeta en donde el archivo se pueda descomprimir y colocar la aplicación producida en el proceso de desempaquetado. Es este caso no existe un proceso previo y la aplicación puede ser abierta sin mayores contratiempos.

En caso de no contar con una versión de Matlab o el MCR, el desempaquetado lo detectará y permitirá la instalación de un versión de MCR. Esto no conlleva la compra de alguna licencia o producto de la compañía MathWorks. MCR es una versión básica del compilador del lenguaje de programación que permite correr interfaces desarrollados en él.

La instalación de MCR requiere que el usuario determine la carpeta en donde se colocará el programa, así como aceptar los términos y condiciones de la licencia. Una vez que la instalación se complete, el software de optimización se desempaquetará en la carpeta en donde se encuentre el archivo empaquetado.

En el caso de SORNA, el archivo pkg contiene cuatro archivos: Instrucciones.txt, readme.txt, AG.exe y SORNA_CP.exe. El primer archivo contiene una pequeña guía de los aspectos básicos que componen el software de optimización. El segundo contiene algunas instrucciones de instalación vinculadas con el MCR. El tercer archivo en un complemento del software que contiene al algoritmo genético de optimización y las redes neuronales artificiales. Finalmente la interfaz es el cuarto archivo que el usuario debe abrir para poder realizar los procesos de optimización de los neurocontroladores (ver figura 7).

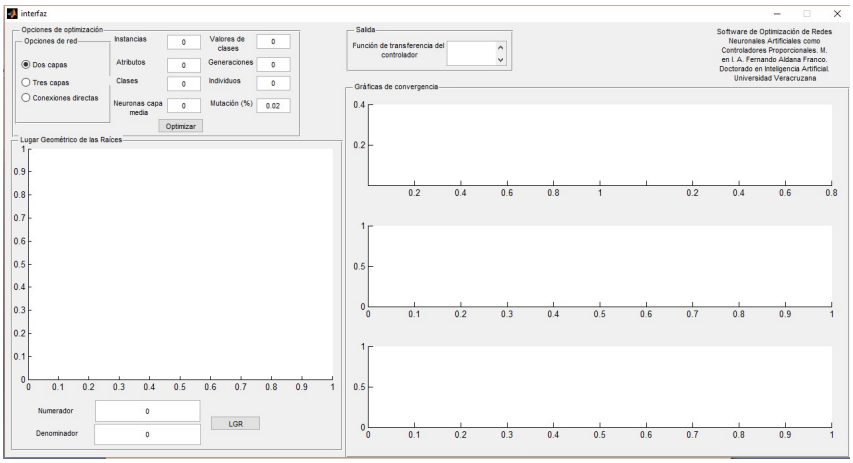


Figura 7. Vista de la interfaz del programa de optimización de redes neuronales.

La interfaz le ofrece al usuario varios aspectos por configurar. Lo primero que se debe seleccionar es el tipo de red a optimizar (ver figura 8). Existen tres tipos de redes neuronales que puedes ser optimizadas: Dos capas, Capa media y Capa media con conexiones directas. Estas opciones se encuentran en la esquina superior izquierda del programa.

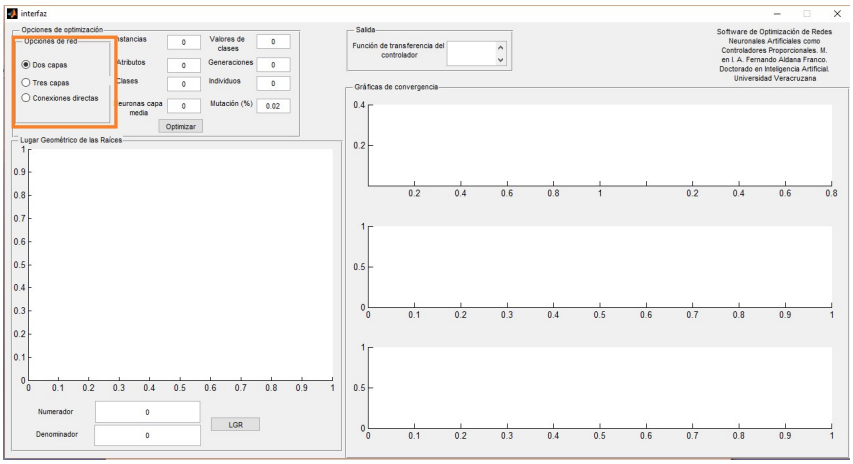


Figura 8. Sección de selección del tipo de neurocontrolador a optimizar.

El siguiente punto a configurar son los aspectos al rededor de la base de datos, la configuración de la red y del algoritmo genético (ver figura 9). Lo primero que el usuario necesita determinar es la cantidad de ejemplos que contiene la base de datos. También se deben configurar el número de atributos, las clases y los valores de la clase. Además se debe definir el número de generaciones deseadas, la cantidad de individuos por generación y el coeficiente de mutación. Finalmente, en caso de elegir las dos últimas opciones de redes neuronales se puede indicar el número de neuronas en la capa media.

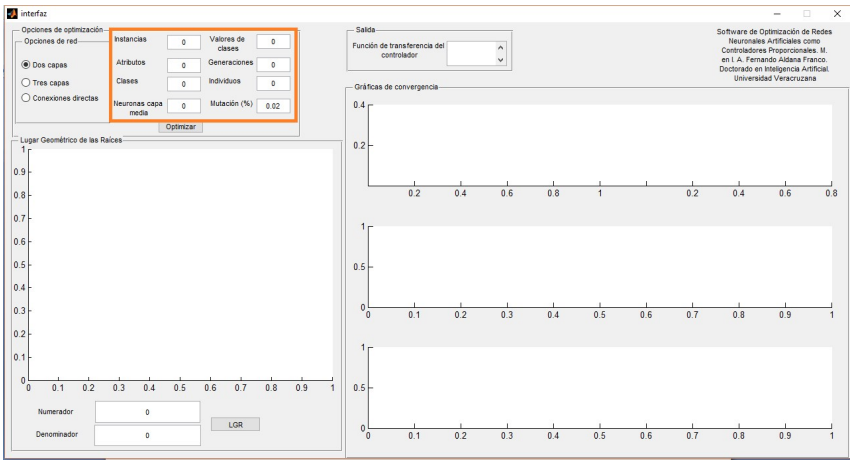


Figura 9. Sección de configuración de parámetros de la interfaz. El coeficiente de mutación inicial es 0.02.

Antes de iniciar el procedimiento de optimización, el usuario debe asegurarse de incluir un archivo llamado "base.txt" dentro de la carpeta en donde se encuentre el software de optimización. Dicho archivo debe contener la base de datos que servirá para optimizar el neuro-controlador. El archivo debe seguir el siguiente orden y configuración: los valores de las instancias en forma de columnas y se listarán sin contener etiquetas, después se incluyen las clases con sus valores y sin etiquetas. Los valores de una instancia deben separarse mediante espacios, y la separación entre ejemplos se realiza mediante saltos de línea. Los atributos pueden representarse mediante números reales. Las clases se representan con números enteros positivos (incluido el 0). El software no contiene ningún archivo de discretización o de interpretación de valores lingüísticos.

Para iniciar con el proceso de optimización, el usuario necesita presionar el botón de optimizar (ver figura 10). Una vez terminado el proceso de optimización, en la interfaz aparecen las gráficas de convergencia (ver figura 11) y el valor de la función de transferencia equivalente a un controlador proporcional. Este valor puede ser utilizado por el usuario para comprobar si el valor del controlador tendrá algún efecto estabilizador sobre la planta del sistema de control.

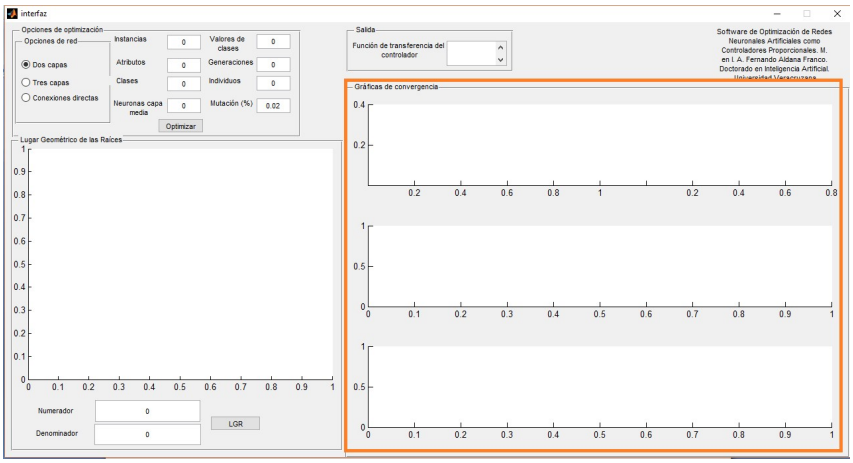


Figura 10. Sección de gráficas de convergencia de la interfaz. La primera gráfica corresponde al mejor individuo en cada generación. La gráfica intermedia es el promedio de los individuos en cada generación. La última gráfica muestra el paisaje de calidad del peor individuo en cada generación.

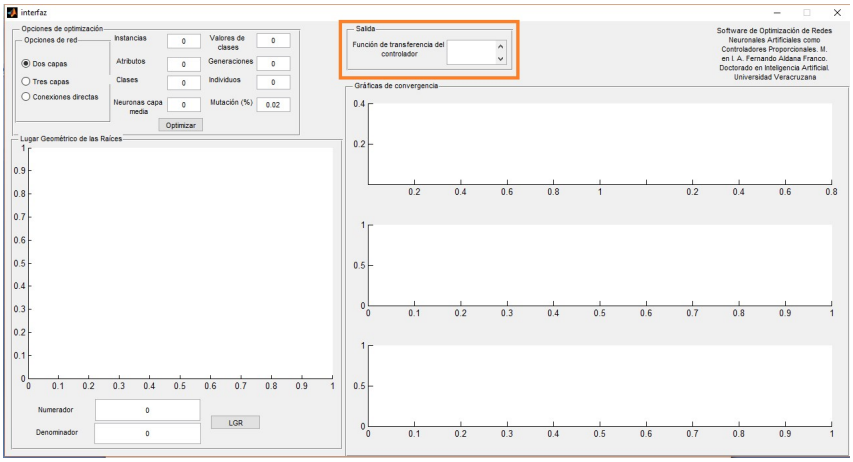


Figura 11. Sección en donde se muestran los valores de la función de transferencia de un controlador proporcional equivalente a cada salida de la red neuronal artificial.

El proceso evolutivo también tiene como salida un par de archivos *.txt. El primero de ellos es "Mejor.txt", en donde se guardan los pesos de la mejor red neuronal de la última generación. El segundo es "G_Final.txt" en donde se guardan lo pesos de cada uno de los individuos de la última generación. Este par de archivos se encuentran en la carpeta que contiene el software de optimización.

Finalmente, el botón "LGR" puede ser utilizado por el usuario para generar una gráfica de lugar geométrico de las raíces de un sistema de control con una función de transferencia en específico (ver figura 12). Previo a esto, se necesita configurar la función de transferencia dividida en los espacios numerador y denominador. En ambos casos se deben indicar en forma de vector, los coeficientes de los polinomios correspondientes cuidando que ambos vectores sean del mismo tamaño y comenzando desde la derecha con el término independiente hasta la izquierda con el

coeficiente asociado al término de mayor potencia en S. De tal manera que la función de transferencia siguiente:

$$G(s) = \frac{s^2 + 3s + 2}{2s^3 + 3s^2 + s + 23}$$

Los vectores deberán representarse como:

$$numerador = [0\ 1\ 3\ 2]$$

$$denominador = [2\ 3\ 1\ 23]$$

Estos valores tienen que ser escritos dentro de la interfaz sin incluir los corchetes.

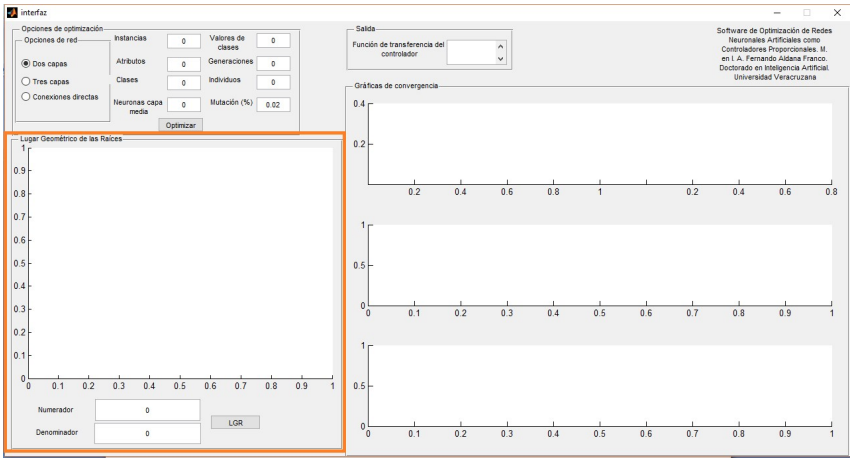


Figura 12. Región en donde se muestra la estabilidad del sistema en base a un análisis de lugar geométrico de las raíces. Los polos deben aparecer en el semiplano izquierdo del plano complejo S.

Con esto, el usuario está en la posibilidad de verificar que el valor de la constante de proporcionalidad produzca un efecto de estabilidad en el sistema. Lo que implica que el valor de los polos en el valor K en la gráfica del lugar geométrico de las raíces sea igual al valor de la constante de proporcionalidad del neuro-controlador. Y por supuesto observando que el valor de los polos en ese valor de K se encuentren todos en el semiplano izquierdo del plano S (ver figura 13).

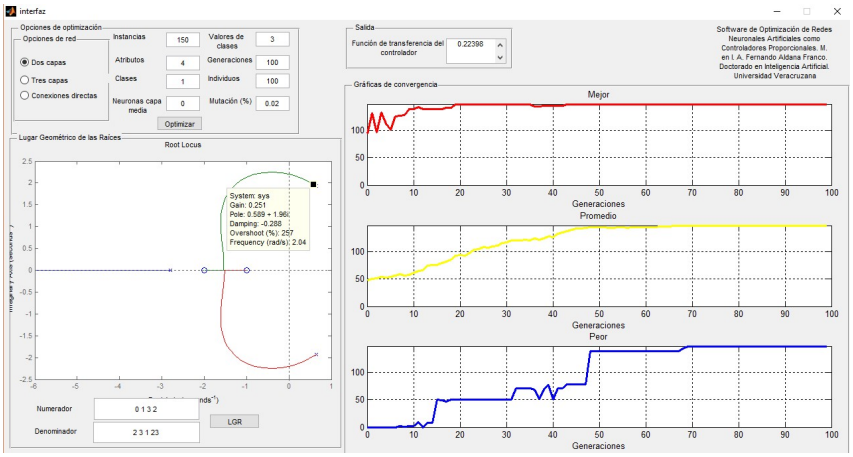


Figura 13. Corrida de prueba del software. El análisis mediante lugar geométrico de las raíces muestra que el sistema resulta inestable en el punto de la ganacia aproximada a 0.2. El sistema se estabiliza con valores superiores a dicha ganancia ya que los polos complejos iguales se trasladan al semiplano izquierdo.

Es importante destacar SORNA cuantifica el valor correcto de cada neurona en la capa de salida. De tal manera que si la red consta de dos neuronas (clases A y B) en la capa de salida y se tienen 50 instancias para optimizar, el valor de aptitud máximo esperado es de 100 que corresponde a 50 instancias de la clase A y 50 de la clase B.

Una vez obtenido una red optimizada se puede utilizar el comprobador SORNA en donde el usuario deberá seguir los mismos pasos de desempaquetado que con el software de optimización. En este caso los archivos descomprimidos son cuatro: la interfaz en Matlab llamada "comprobador_SORNA", un archivo ejecutable llamado "red_calcula" , un archivo readme con las instrucciones de manejo de las interfaces de Matlab y el MCR, y un archivo de texto llamado "instrucciones" que contiene una guía básica del funcionamiento del software de comprobación. El primero de ellos con el cual el usuario deberá trabajar.

Para hacer funcionar el comprobador es necesario copiar y pegar el archivo "Mejor.txt" resultado del proceso de optimización con SORNA. Es este archivo el que contendrá los pesos con los cuales será probada la red. Dentro de la interfaz, el usuario deberá configurar el tipo de red eligiendo entre tres opciones (ver figura 14): Dos Capas, Tres Capas, Conexiones Directas. Una vez seleccionada la topología de la red, es necesario indicar la cantidad de atributos del ejemplo que se desea clasificar, así como el número de clases, los valores posibles para las clases y el número de neuronas en capa media si la topología lo requiere (ver figura 15). Además se deberán definir los valores de atributos y la clase esperada (ver figura 16). Y finalmente presionar el botón de comprobación para generar una salida (ver figura 17).

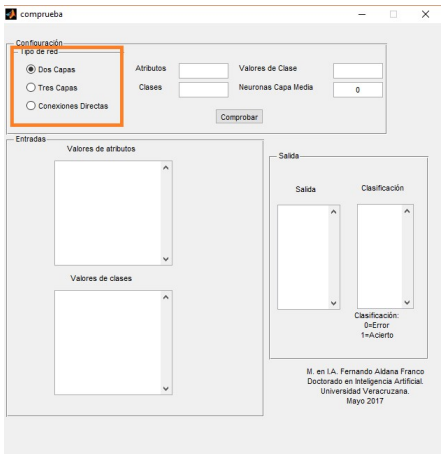


Figura 14. Sección de selección del tipo de neurocontrolador a comprobar.

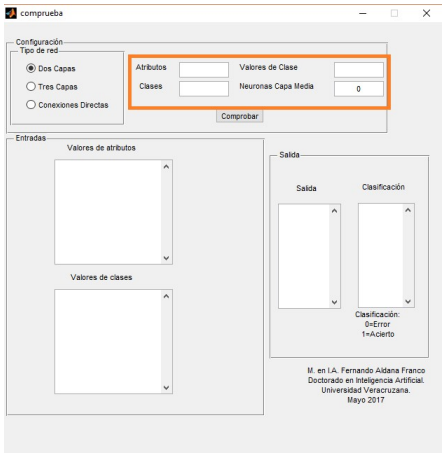


Figura 15. Sección de configuración de número de atributos y clases, número de valores posibles de clases y número de neuronas en la capa media.

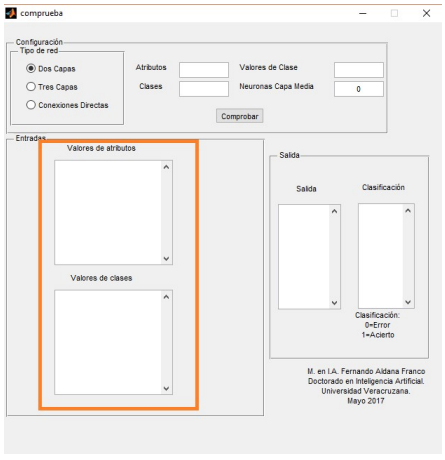


Figura 16. Sección de configuración de los valores de atributos y clases. El usuario deberá listar el ejemplo que desea comprobar.

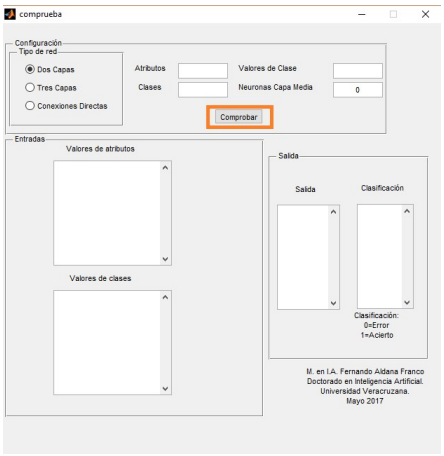


Figura 17. Botón de comprobación que desencadena el proceso computacional para el cálculo de salida de una red neuronal seleccionada dados los datos de un ejemplo.

El proceso computacional del software devolverá dos valores la sección de salidas (ver figura 18): salida de la red y el valor de clasificación. El primero es la salida de la red calculada a través de la salida neta y el ajuste de la función de transferencia de salida de la red del tipo sigmoidal. El segundo es un valor binario que determina si la red clasificó de manera correcta la instancia presentada (valor=1) o lo hace de manera incorrecta (valor = 0). De esta manera el usuario podrá conocer el valor exacto de la salida de la red y si la misma clasifica de manera correcta una instancia (ver figura 19). Por ello este software complementario representa un mecanismo para validar las estructuras de RNA obtenidas a través de SORNA.

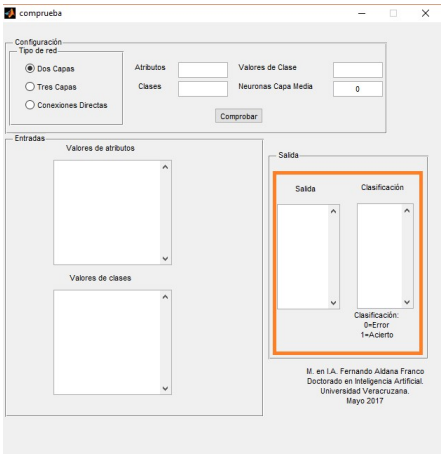


Figura 18. Sección de salida del software de comprobación SORNA. Se muestran dos listas en donde aparecerá el valor de la salida de la red y un indicador binario para conocer si la red clasificó de manera correcta el ejemplo dado.

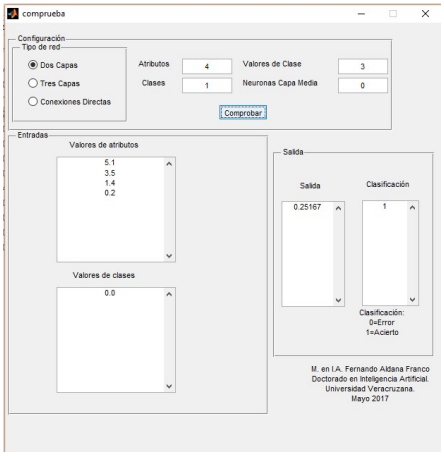


Figura 19. Corrida de prueba del software de comprobación. Se comprueba el funcionamiento de una red de dos capas, con cuatro atributos y una salida con tres valores posibles. El ejemplo es bien clasificado como se muestra en la sección de salida.

Referencias.

[1] Boza Condorena E.G. y Cravalho da Costa A. A method of transfer functions and block diagrams to study the contribution of variables in Artificial Neural Network Process Models. Proceedings of the World Congress on Engineering and Computer Science 2011, vol. 2. 2011.

[2] Chen C. Analog and Digital Control System Design: Trasnfer-Function, Statte-Space, and Algebraic Methods. Saunders College Publishing. 1995.

[3] Chen G., y Pham T. Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems. CRC-Press. 2001.

[4] Galán R., Jiménez A., Sanz R., y Matía F. Control Inteligente. Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial 4(10), 43-48. 2000.

[5] Hagan M., Demuth H., Hudson Bale M., y De Jesús O. Neural Networks Design. Segunda edición. 2014.

[6] Hellerstein J., Diao Y., Parekh S., y Tilbury D. Feedback Control of Computing Systems. Wiley-Interscience-IEEE. 2004.

[7] Hernández-Guzmán V., Silva-Ortigoza R., y Carrillo Serrano R. Control Automático: Teoría de diseño, construcción de prototipos, modelado, identificación y pruebas experimentales. Primera Edición. Colección CIDETEC IPN. Enero 2013.

[8] Jinn J., Ma X., y Kosonen I. An intelligent control system for traffic lights with simulation-based evaluation. Control Engineering Practice 58, 24-33. 2017.

[9] Kriesel D. A Brief Introduction to Neural Networks. 2007.

[10] Kuo B. Sistemas de Control Automático. Séptima Edición. Pearson Prentice Hall. México. 1996.

[11] Montes González F. y Aldana-Franco F. The evolution of Signal Communication for the e-puck Robot. Advances in Artificial Intelligence, 466-477. 2011.

[12] Nolfi S., Elman J., y Parisi D. Learning and Evolution in Neural Networks. Adaptative Behavior 3. Pp. 5-28. 1994.

[13] Ogatta K. Ingeniería de Control Moderna. Tercera edición. Pearson Prentice Hall. México. 1998.

[14] Rodríguez A. A Practical Approach To Signals, Systems and Controls. Arizona State University. Septiémbre 1998.

[15] Ross T. Fuzzy Logic with Engineering Applications. Tercera edición. Wiley. 2010.

[16] Santos M. Un Enfoque Aplicado del Control Inteligente. Revista Iberoamericana de Automática e Informática Industrial 8, 283-296. 2011.

[17] Shinskey F. Process-Control Systems Application, Design, Adjusment. Segunda edición. Mc-Graw Hill. 1979.

[18] Vassilyev S., Kelina A., Kuinov Y., y Pashchenko F. Intelligent Control Systems. Procedia Computer Science 103, 623-628. 2017.

Anexo 4.

Environmental factors that affect the emergence of signal in population of evolutionary robots.

Journal of Artificial Intelligence Research - (2017) - Submitted -; published -

Environmental factors that affect the emergence of signals in population of evolutionary robots

Fernando Aldana-Franco FALDANA@UV.MX
Fernando Montes-Gonzlez FMONTES@UV.MX
Artificial Intelligence Research Center. University of Veracruz, Xalapa, Veracruz, Mexico,

Stefano Nolfi STEFANO.NOLFI@ISTC.CNR.IT
Institute of Cognitive Sciences and Technologies. Consiglio Nazionale delle Ricerche,, Rome, Lazzio, Italy.

Abstract

Communication systems represent an evolutionary advantage for a group of robots that solve particular coordinated and uncoordinated tasks. They represent a tool that is used during evolution for interchange personal and environmental information. In our research, we focus on some aspects that have important effects on the emergence of evolutionary communication systems, and through their modification impact the evolutionary development: i) the evolutionary importance of the task, ii) complexity of the environment, and iii) complexity of the communication system. The FARSA simulator is used as a computational platform to develop evolutionary processes for optimizing the weights of neural networks that control homogeneous populations of Marxbot robots, which emit signals using color LEDs. A coordinated poison and food task is configured in order to conduct our research.

1. Introduction

Evolutionary Robotics (ER) is a field where morphological and/or control structures are the result of iterative pressure performed by an artificial evolutionary process (Nolfi, 1998). The typical representations of control systems in ER are Artificial Neural Networks (ANNs), optimized by evolutionary algorithms such as the Genetic Algorithm (GA) (Holland, 1992). Efforts in the area are concentrated on evolving primitive brains as control systems that emerge under specific environmental conditions. For that reason is important to understand how these environmental features influence evolutionary processes (Bongard, 2013).

Communication is an important characteristic for some species and individuals that allows them to interchange messages about the environment, personal status or situations, and collective information (Mitri, Floreano, & Keller, 2010). On evolutionary robotics, communication systems let robots interchange information and develop coordinate behavior (Trianni, Labella, & Dorigo, 2004).

In nature, communication systems have different channels for transmitting information: sound, visual, chemical, odor, and electrical. On robotics, communication is typically based on actuators such as wireless and radiofrequency signals, sound, color changes, and movements. The way in which a group of robots establishes communication depends on their the actuators and sensors, and the way they interact (Scott-Phillips, Blythe, Gardner, & West, 2012). For that reason is very important to understand all the mechanisms that allow communication systems to emerge in ER (Steels, 2003).

Communicative skills tend to emerge on communities and groups of robots that have to develop collaborative or competitive tasks (Ampatzis, Tuci, Trianni, & Dorigo, 2008). An example of that is the experiment known as poison and food task for evolutionary robots which was configured by (Floreano, Mitri, Magnenat, & Keller, 2007). They develop an evolutionary mechanism for the emergence of communication in robots. In this experiment, the environment is configured with four S-bots in a virtual arena; two devices produce red lights and represent food and poisoned zones. Robots have to find food zones, linger there, and avoid poisoned zones. As for the communication robots are equipped granted with the skills for sharing, or not, information.

In the experiment, robots are tested in teams of ten robots. Team heterogeneity is set as a selection level. Next, a team can be defined as either homogeneous or heterogeneous. When a team is homogeneous, each member has the same genotype. In a heterogeneous team each robot is configures with a different genotype. The evolutionary process is composed by 500 generations. In order to test the robots, at the end of evolutionary processes, robots teams are selected in four different ways: homogeneous population; heterogeneous population; the best homogeneous individuals, and the best heterogeneous individuals.

In their results (Floreano, Mitri, Perez-Uribe, & Keller, 2008) show that the emergence of a communication system improves the performance of the task increasing the fitness of the task for groups with this kind of system. Additionally, team composition is a factor which improves the solution of the task when emerged signals were more meaningful than in homogeneous populations. The fitness of heterogeneous populations is lower than in homogeneous populations. This is due to the evolutionary advantage associated to the interchange of information and composition of populations. Whereas in homogenous populations robots signalize food areas in order to attract other robots to improve their fitness (Mitri et al., 2010); in contrast heterogeneous robots are unable to maintain a reliable communication associated with a level of altruistic behaviors in the population (Waibel, Keller, & Floreano, 2009).

As a result, signalized food areas offers an evolutionary advantage that allows robots to fulfill the task through prompting robots to move to the food areas. Hence, this task represents an experimental platform for testing emerging developing communication systems and the manipulation of different experimental variables in order to understand some of the underlying mechanisms for the emergence of communication systems. For that reason, we based our research in the poison and food environment.

However, the emergence of signal communication in evolutionary robots is not affected only by modifying the population composition. Emergence of signals has an important relation with the conditions of the environment and control systems (Montes-Gonzalez & Aldana-Franco, 2011). In this way, signals are emitted in situations with a high level of evolutionary advantage (Steyven, Hart, & Paechter, 2015). An evolutionary advantage is a beneficial condition that helps individuals in an artificial evolutionary process to better adapt to the environment. As a consequence, individuals increase the possibilities of transmitting their genetic information into the next generations.

Signals and cues might be 6the result of an environmental value associated with a lexical value and correlated with behaviors. When a group of evolutionary robots associate a signal with a particular situation like food zones, a conceptualization process happens and signals receive a specific value in the communication system (Solé, Corominas-Murtra, Valverde,

& Steels, 2010). Thus emitters and receptors develop abilities for understand signals and their meanings, e.g. lexical value (Loula, Gudwin, & Queiroz, 2010). This signals produce behavioral responses on receivers and emitters. Then, environmental characteristics creates the adaptive conditions for changes in the emergence of semantic and syntax, communication skills and behavior (Nolfi,).

Therefore, in order to associate lexical values to cues, the environment must provide the necessary conditions for the optimization processes to set an evolutionary pressure onto the population of robots to allow the emergence and preservation of signals. Lexical acquisition represents one of the most relevant fields of research on evolutionary and communicative robots (Rasheed & Amin, 2016). Hence, we named a defined-signal as one that has a defined lexical value and is emitted in a particular situation or context which produces a behavioral response on receivers.

In our work, through the manipulation of factors and environmental variables we configured three experiments. The first experiment, named The Blocking Task Experiment, is related to changes in the evolutionary advantage of communication signals; the second, The Multiple Signals Experiment, is related to the complexity of the environment; and the third, The Signals in a Complex Scenario Experiment, is related to the complexity of the communication. In general, our proposal is to demonstrate these manipulations had an impact on the emergence of signals and performance of robots that employed a color based communication system. The communication system was set in terms of fitness, the number of emerged signals, and places where signals originated.

The paper is organized as follows: in the experiment and results section we present three experiments, their configuration and description; next the discussion section focus on our experimental results; and finally in the conclusions we propose our future work.

2. Experiments and Results

All of our experiments were conducted in a virtual world using the Marxbot robots and FARSA simulator, which is a Framework for Autonomous Robotics Simulation and Analysis (Massera, Ferrauto, Gigliotta, & Nolfi, 2013). This simulator is an open source simulator for evolutionary robotics that includes a group of tools and algorithms, neural networks, and computational models for other robots such as Khepera, E-puck, Marxbot and iCub robots. The weights of the artificial neural structures, through the evolutionary process, were optimized using a steady state genetic algorithm (Schwehm, 1996). Our steady state version in FARSA is based on the mutation of an initial population and the substitution of the worse parents instead of their improved sons. The mutation rate for the algorithm decreases in each generation until a minimal value is reached.

In the experiments a full evolutionary process (i.e. a replication) consisted of 500 generations. The population was made of 20 individuals for reproduction with an initial mutation rate of 50%, decreasing 2% by iteration until a final steady rate of 1% is maintained. A single trial consisted of 300 steps for a team of robots, and then the team was tested by 10 trials. We allowed every experimental group to evolve by 12 replications. The fitness function rewarded one point to robots that spent one step in the food area, and one negative point each time the robot was in a poisoned area.

On the other hand, the Marxbot is a modular robot equipped with 24 infrared sensors, 12 ground sensors, a ring of LEDs, a camera (one Ultra Extended Graphics Array Camera - UXGA), two pairs of Treels which are two motors each associated with a rubber track and an additional wheel, Linux 2.6 Aseba Operative System onboard, 10 micro-controllers dsPIC33, and connections for adding extra electronic devices. We selected this robots because of its rings of LEDs and camera, that allow to develop visual communication systems.

About the sensory-motor components, robots were equipped with 24 infrared sensors encoded in eight groups of the averaged measures of three sensors, ground sensors specialized in the detection of the food and poisoned areas, and five segments of 72 for the RGB components of the linear camera, the two motors, and the ring of LEDs.

2.1 The Blocking Task Experiment

Here we intended to prove that the emergence of a signal in a particular place had a relation with the evolutionary advantages of pointing beneficial situations. In the literature we have found an example of the emergence of signals in homogeneous populations near beneficial places (Mitri, Floreano, & Keller, 2009) like food zones. In other words, the emergence of signals is related to the place where a lexical value of the signal is more useful for the evolutionary process and individuals. For this reason, we designed an experiment to demonstrate that signals tend to emerge in a different places or situations (lexical value) when the evolutionary advantage of pointing food zone is reduced. Then, a blocking condition was programmed, which set the motor speeds to zero using FARSA. Thus, as soon as a robot arrived at the food zone, it was not allowed to leave the zone. That condition reduced the beneficial evolutionary cost associated with signalize food areas. It was expected that the places where signals emerged changed, thus increasing the number of replications of emerged signals in poisoned zones or outside of food zones. Our experiment was composed by two groups: a) one with a blocking condition, and b) the second without a blocking condition.

A population of six robots was set to solve the task. The population is homogeneous, which means that all robots were genetically identical. Two target areas colored in grayscale composed the environment: light gray (0-0.4 gray level) for the food zone, and a dark gray (0.6-1.0 gray level) for the poisoned zone (see figure 1). Each target area had an extra visual signal with a green cylinder that sums up to the visual information of the environment. The color of the ground was gray (RGB = 127, 127, 127).

Each control system included a feed-forward neural network with 132 weights and 27 neurons (see figure 2): 19 inputs (eight groups of three infrared sensors; one ground sensor for a color change identification in grayscale; five blue and five green component sensors for the linear camera); five hidden neurons with a bias input; and three outputs with a bias input (two motors and one for signal codification). Robots were allowed to use two signals codified with one binary neuron: one signal for dark tones (0, 0, 0 in RGB) with a binary value of zero, and the other signal for a blue tone in RGB = 0, 0, 255 which corresponds to a binary value of one.

Two dependent variables were measured: the fitness function level and the place where robots emitted signals (food area, poisoned area, other robot presence, or no signals). We used fitness levels because the emergence of a communication system brings direct benefits

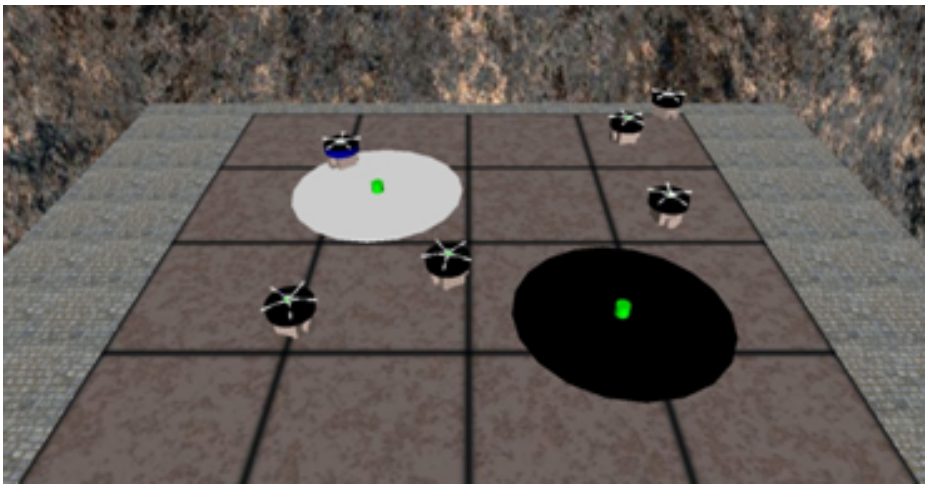


Figure 1: The setup for the virtual environment, in FARSA, is presented where the poisoned and food task for the blocking task and multiple signals experiments are developed. Six robots started in random positions and had to find a white food area, also avoid a black zone that represents a poisoned area. The rest of arena was colored in gray. Robots used their cameras and LEDs as a communication system.

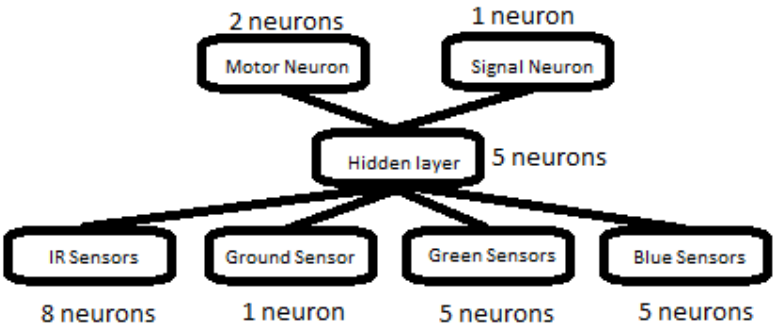


Figure 2: The configuration of the ANN used on experiment blocking task. The evolutionary process optimized a feed-forward neural network composed by 27 neurons: 19 neurons at the input layer, 5 in the hidden layer, and 3 at the output layer.

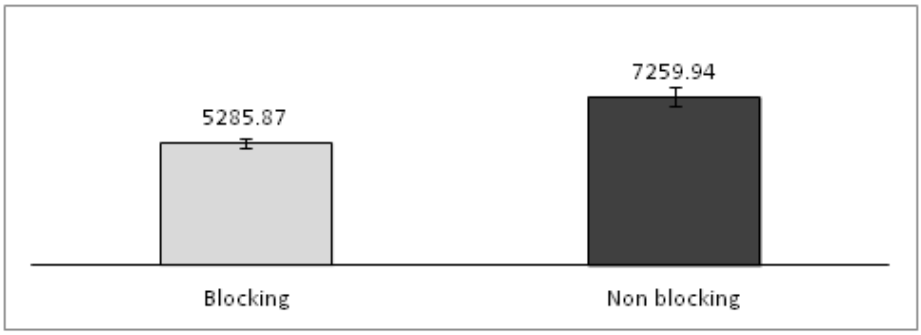


Figure 3: The median fitness and standard error of groups for experiment 1 - blocking task.

to robots (Marocco, Cangelosi, & Nolfi, 2003). The fitness function was quantified as an average fitness score of the last 200 of the best individuals for each replication. Then, a communication system is considered stable when the strategy of robots to signalize does not change in 200 generations. The statistical test used for finding differences between groups was a t-test (=0.95), with fitness scores as the dependent variable and the blocking condition variable as the independent variable (presence and absence). Signalization was quantified for four different places or situations where robots could emit signals (food, poison, other robot, and no signal). The experiment was registered for 10 minutes through an instantaneous scan sampling of the best individuals at the last generation for all replications in the experimental groups.

The results exhibited changes in the performance of robots and the place where robots emitted signals in the experimental groups. In relation to the statistical analysis, there were statistical differences between groups (p=¡ 0.001, n=12) in terms of fitness, as showed by a T-test. The best fitness corresponded to the group with no blocking condition (see figure 3).

The comparison of both groups in terms of places where signals emerged were as follows, for individuals in the group with no blocking condition, 40% of the seeds did not develop signal communication and 60% of seeds sent signals in the food zones. On the contrary, 30% of replications evolved with a blocking condition, which in turn did not develop a communication system, 20% sent signals in the poisoned areas, and 50% emitted signals near another robot.

2.2 The Multiple Signals Experiment

For this experiment we compared the performance of a multi-signal communication system affected by two levels of complexity (low and high). Two experimental groups were set up: a basic complexity group (bcomplex-group) and a high complexity group (hcomplex-group). The bcomplex-group used white and black colors on targets and two ground sensor unities for detecting them. Reduced detection facilitates the search in the solution space. On the other hand, the hcomplex-group used grayscale colors for targets and used one ground

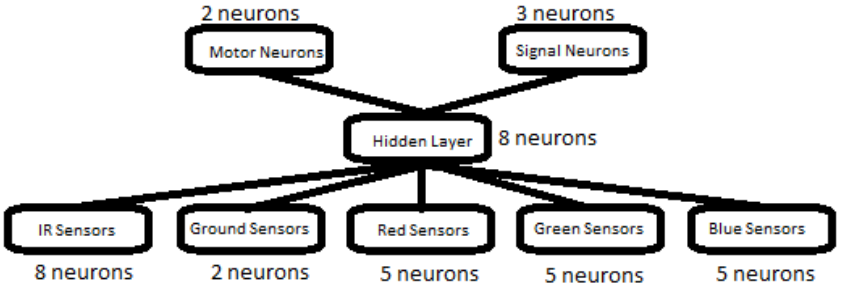


Figure 4: The configuration of the feed-forward ANN used for individuals of the bcomplex-group in the second experiment named multiple signals. The evolutionary process optimized the weights of a neural network composed of 38 neurons: 25 neurons at the input layer, 8 in the hidden layer, and 5 at the output layer.

sensor unity as in the first experiment. In the last experimental scenario the complexity of the search space is more elaborated.

We proposed that the inclusion of different levels of complexity in the world configuration produced changes in performance and emergence of signals. It means that a more complex communication system produces changes in collective signalization strategies. As in experiment 1, a green cylinder was used for extra visual information for finding target areas. The proposed population for solving the task was composed of six genetically identical robots. The color of the ground corresponded to gray (RGB = 127, 127, 127).

The ANN for the bcomplex-group had 278 weights and 38 neurons (see figure 4). The inputs form an array of 25 elements from: eight infrared sensors; two ground sensors - one for white color detection and one for black color detection; five red, five green, and five blue component sensors for the linear camera that represent segments of 72 degrees of the field of view; eight hidden neurons with a bias input; and five outputs with a bias input (two motors and three binary color neurons).

As for the ANN of the hcomplex-group was composed by 269 weights and 37 neurons (see figure 5). Thus, 24 inputs were set in array from: eight infrared sensors; one ground sensor for the identification of the grayscale color changes; five red, five green, and five blue component sensors for the linear camera; eight hidden neurons with a bias input; and five outputs with a bias input (two motors and three binary color neurons).

The communication was based on RGB binary combinations controlled by three neurons at the output layer producing eight signals or combinations of colors. In this way, a black color signal was produced by the combination of low levels in the three neurons, and white color was the result of combining high levels of the three neurons.

We measured changes in the experimental conditions by averaging the fitness scores of the last two hundred best individual of each replication. A Mann-Whitney Rank Sum statistical test (=0.95) was used to compare fitness levels of experimental groups as the dependent variable, and the level of complexity as the independent variable (basic and high). Signalization was quantified on four different places, or situations, where robots

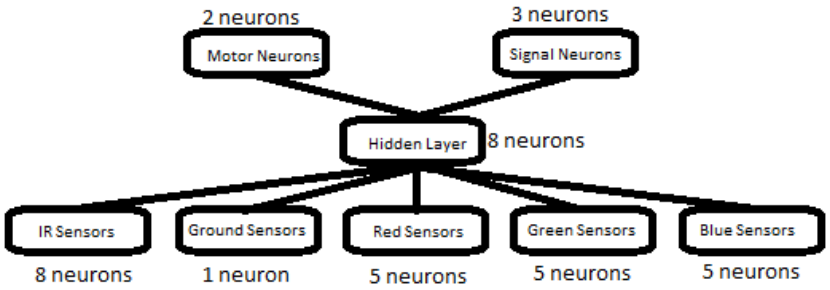


Figure 5: The configuration of the feed-forward ANN used for individuals in hcomplex-group in the multiple signals experiment. The evolutionary process optimized the weights of the neural network composed of 37 neurons: 24 neurons at the input layer, 8 in the hidden layer, and 5 at the output layer.

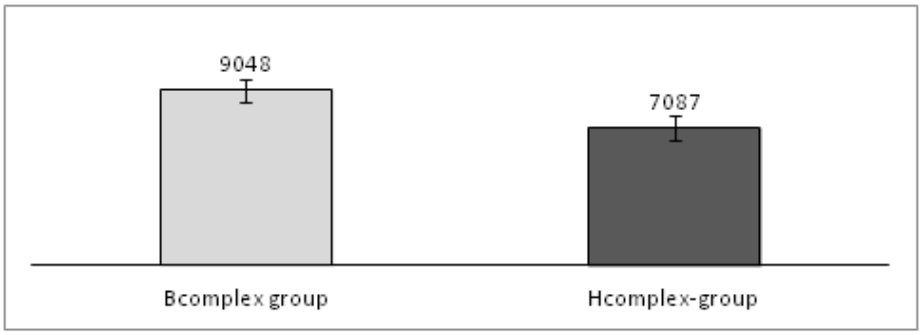


Figure 6: The median fitness and standard error of groups for experiment 2 - multiple signals.

could emit signals (food, poison, other robot, and no signal) and the number of defined-signals that emerged (a defined-signal is an emerged cue with lexical value, or a cue with a meaning which may be related in a particular situation, i.e. when the robot was on the food or poisoned zones). The experiments were registered for 10 minutes through an instantaneous scan sampling of the best individual at the last generation of all replications in the experimental groups.

The results of the Mann-Whitney Rank Sum analysis showed that there were statistical differences between experimental groups ($p=0.018$, $n=10$) based on fitness levels, and the highest fitness corresponded to the basic complexity group (see figure 6). The statistical test took the fitness as the dependent variable, and the level of environmental complexity as the independent variable.

Data obtained from the instantaneous scan sampling of the best individual revealed that in the basic complexity group, signals emerged on the white line (56.26%), black line

(12.5%), another robot presence (25%), and few robots did not produced signals (6.24%) with averaged defined-signals of 1.9. On the other hand, the high complexity group produced signals in the food zone (40.9%), poisoned zone (18.18%), and another robot presence (36.36%); all with averaged defined-signals of 3.0 (4.56% did not produce signals).

2.3 The Signals in a Complex Scenario Experiment

In this experiment we intended to prove that a communication system with the capacity of producing more signals was more effective in a complex environment. We have the hypothesis that a communication system with more potential signals has better performance than a simple communication system.

The first and second experiments were based on the experiment of (Floreano et al., 2008), which facilitated the emergence of signal communication because the environment set an evolutionary pressure to produce signals with contextual values like those signals produced for robots on both food and poisoned zones. The simple environment used in previous experiments did not promote complex control and communication systems that associated signals with lexical values under different stimulus situations. As a result a basic communication system was developed with limited the number of potential situations where signals emerged.

Therefore, the initial scenario was modified in order to facilitate the emergence of signals under different situations and obtain more cues with lexical values. First, the size of the arena was increased and included two delimiting perpendicular walls next to target areas. For this experiment, the target areas were used to notify individuals that food and poisoned areas were nearby. However, they were not the actual food and poisoned zones; thus adding new potential situation where robots emitted signals (see figure 7). The shape of the target areas was modified from circular to rectangular to add the target areas. Food and poisoned areas were located after the target areas and were colored in gray (RGB = 127, 127, 127), and also was the rest of the arena.

The distribution of food, poisoned, and target areas, was designed to facilitate the emergence of four different signals: two on both targets areas, two for food and poisoned zones. Also, the environment favored the emergence of signals when robots were next to walls or another robot.

In order to solve the task, of finding food areas and avoid poisoned zones, 20 individuals were produced for each generation, and then these individuals were tested in teams of 10 robots. As in the last two experiments, robot populations were configured with identical genetic individuals.

Four experimental groups with different communication systems were configured for this experiment. The first group was named control-group. It consisted of robots having a control system without a hidden layer and a communication system based on two signals.

The neural network used for controlling the robots in the control group was composed of 66 weights and 24 neurons (see figure 8). The input layer had 21 neurons (eight groups of infrared sensors; three ground sensors for the identification of white, black, and gray colors; five green and five blue component sensors for the linear camera). Then, three neurons at the output layer with a bias input (two motor neurons and one neuron for signal codification).

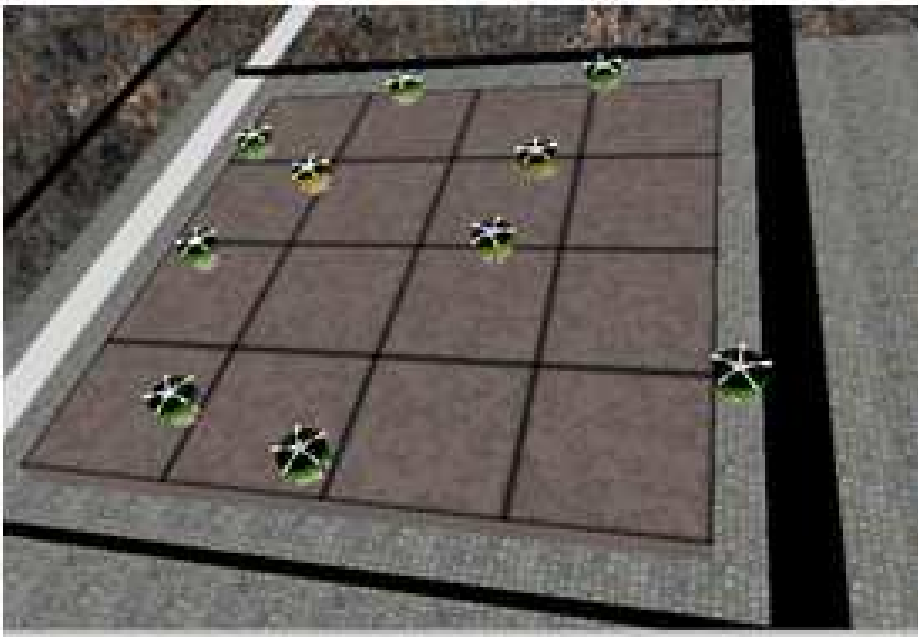


Figure 7: The virtual environment, set in FARSA simulator, of the poisoned and food task for signals for the complex scenario experiment. Ten robots were located in random central positions of the arena and have to find a food area situated beyond the white target area. Then the team has to avoid a poisoned zone situated across the black zone. The rest of the arena was colored in gray. Robots used a communication system based on cameras and led rings.

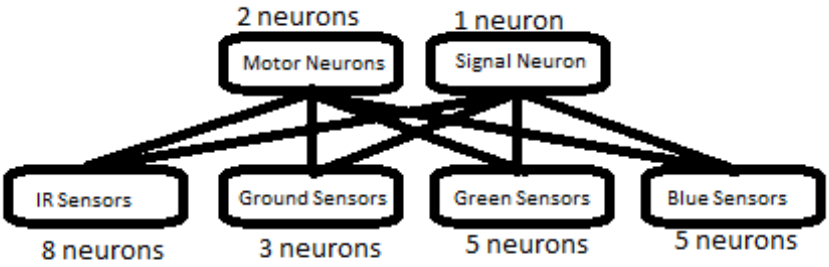


Figure 8: The configuration of the neural network used in the control-group of the experiment named signals in a complex scenario experiment. It was composed in total of 24 neurons: 21 inputs and 3 outputs.

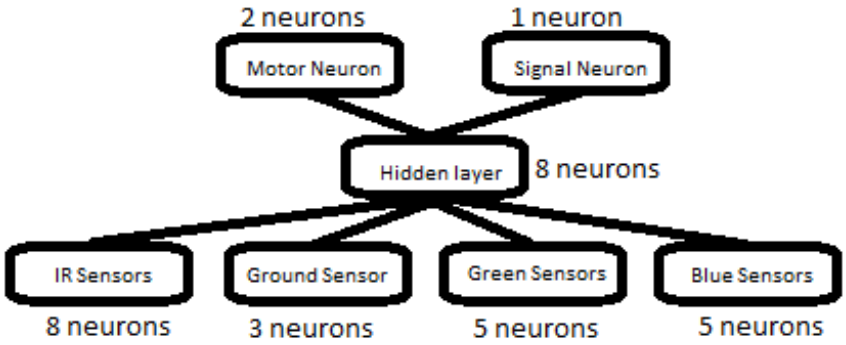


Figure 9: The configuration of the ANN used in group-1of the experiment: signals in a complex scenario. The neural network was composed by 32 neurons: 21 at the input layer, 8 at the hidden layer, and 3 at the output layer. Neurons at the hidden and output layer included bias connections.

Signal neurons worked in a binary way, when the activation exceeded a threshold, it was classified as high and produced a blue signal, and otherwise robots emitted a black signal.

The second experimental group was named group-1. The neural network had a hidden layer and the communication system was based on two signals. The neural network used for controlling robots was composed by 224 weights and 32 neurons (see figure 9). In the input layer 21 neurons were set (eight groups of infrared sensors; three ground sensors for the identification of white, black and gray colors; five green, and five blue component sensors for the linear camera). In the next layers neurons were set as follows, eight neurons at the hidden layer with a bias input, and three at output layer with a bias input (two motor neurons and one neuron for signal codification).

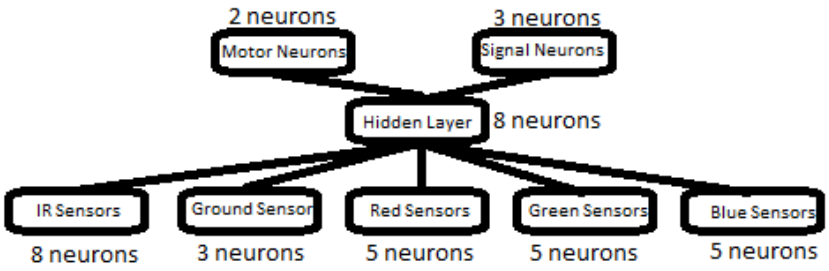


Figure 10: The configuration of the ANN used in group-2 and group-3 for the experiment signals in a complex scenario. The neural network was configured in a feed-forward architecture and composed by 39 neurons: 26 at the input layer, 8 at the hidden layer, and 5 at the output layer.

As for the third experimental group, group-2, robots used a hidden layer in the neural network and the communication system was based on three binary neurons that combined produced eight signals. The ANN was composed by 287 weights and 39 neurons (see figure 10). In the input layer, 26 neurons in total, eight groups of infrared sensors; three ground sensors for identification white, black, and gray colors; five red, five green and five blue component sensors for the linear camera. At the hidden layer eight neurons were set with a bias input, and five neurons at the output layer with a bias input (two motor neurons and three neurons for signal codification. Communication systems operated with binary neurons. A combination of activations was used to classify the emitted signal: white when all neurons had high levels at the output, and black for low levels.

In the fourth experimental group, group-3, robots were configured with a communication system based on a RGB continuous scale. The neural controller was formed by 287 weights and 39 neurons (see figure 10). Neurons were distributed as follows: 26 at the input layer (eight groups of infrared sensors; three ground sensors for identification white, black and gray colors; five red, five green, and five blue component sensors for the linear camera); 8 at the hidden layer with a bias input; and 5 at the output layer with a bias input (two motor neurons and three neurons for signal codification).

The ANN on this experimental group was the same as in group-2. The main difference was related to the way in which robots produced signals. Robots had three signal neurons that represented one RGB component. All neurons produced outputs in a range of 0 to 1 with a resolution of 0.00392; and signals were produced in the RGB scale with continuous values.

The complexity level of the communication systems was used as independent variable. Three dependent variables were measured: the fitness function level of the last 200 best individuals, the place where robots emitted signals, and the number of emerged defined-signals (a defined-signal is a cue that have a lexical value, which means that is emitted in a particular situation). In terms of fitness, a statistical analysis was used for finding statistical differences between experimental groups through a One Way ANOVA on ranks

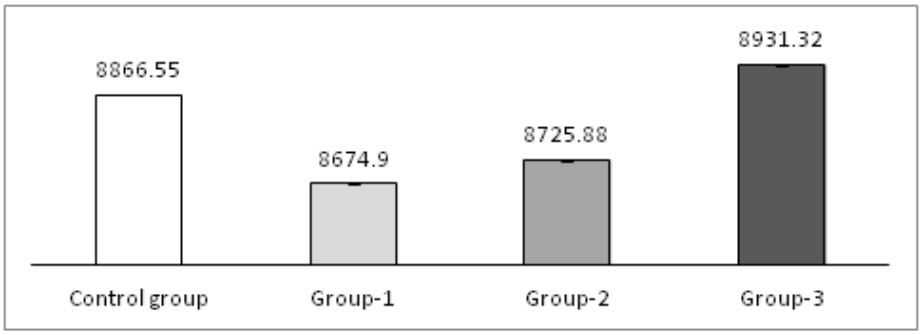


Figure 11: The median fitness and standard error of groups for experiment 3 - signals in a complex scenario.

and complemented with a post-hoc test Fisher LSD Method (=0.95). Fitness scores as the dependent variable was quantified with the average of the last two hundred best individuals of each replication. In relation to the places where robots emitted signals, there were five places or situations where robots could emit signals (no signal; a black line next to a poisoned zone; a white line next to the food zone; another robot detected; and a nearby wall). Signals were registered for 10 minutes through an instantaneous scan sampling of the best individual of the last generation for all replications in the experimental groups.

Our analysis of the fitness scores between groups showed that the best fitness level corresponded to groups in which signals were produced with a RGB continuous level and with two signals without a hidden layer, which corresponds to group-3 and control-group (see figure 11). As for rest of the groups, group-1 and group-2 had the worst performance in terms of fitness scores.

The statistical analysis of the fitness function as the dependent variable and the complexity of the communication systems as independent variable showed differences between experimental groups (p=0.026, n=12).

The post-hoc test proved that the experimental group-3 had the best performance compared to group-1 and control-group (p=0.008) and group-2 (p=0.03). A comparison between groups with 2 signals, control-group and group-1, showed statistical differences (p=0.042,) with a better performance of two signals and no hidden layer group (control-group).

In relation to the analysis of the place where signals and defined-signals were produced, robots of the control group tended to produce signals on the white line and food zones (55.55%), followed by signals produced near walls (16.67%), another robot (16.67%), and on the black line and poisoned zone (11.11%). The defined-signals average was 1.2.

In contrast, in group-1 signals emerged, for most of the seeds, on the white line and the food zone (23.09%), near other robots (15.38%), walls (16.9%), on the black line and the poisoned zone (12.23%), and then individuals that did not produce signals (32.4%). The average of emerged defined-signals was 0.5.

In group 2, robots developed signals on the black line and the poisoned zone (26.64%), the white line and the food zone (22.22%); near walls (25.92%), and robots (18.52%); all

with 1.5 defined-signals average (6.7% which did not produced signals). Finally, robots in group-3, signals were emitted on the black line and the poisoned zone (30.79%); on the white line and the food zone (23.07%); near another robot (26.92%), and walls (15.38%). All of them with a 2.1 average of defined-signals (3.84% which did not produce signals).

3. Discusion

The Blocking Task Experiment results showed that robots, which located food using signals without a blocking condition, suppressed this strategy because the evolutionary advantage associated with food signalization was reduced through the blocking condition. Therefore, the results on this experiment demonstrated that signal emergence changes in response to a change in the environment that ultimately reduces the evolutionary advantage of signaling some situation (e.g. food zones). We noticed that there is a relationship between the place where signals emerge and their adaptive advantages. When a signal emerges under controlled conditions and it does not have a specific function on a group of robots, as a consequence the signal tends to disappear (Cangelosi, 2001). As a consequence, the emergence depends on their real value; signals have to develop lexical values on the population of robots, otherwise the evolutionary process forces robots to change their signalization strategy.

It has been showed that the evolutionary importance of signals in the task is based on the attraction of robots towards the food areas under similar scenarios (Wischmann, Floreano, & Keller, 2012). Thus, robots tend to travel around food areas while emitting signals. However, when robots were blocked as soon as they arrived to the food zones, the evolutionary importance of emitting signals was reduced. For this reason, robots produce more signals under different situations or places (e.g. poisoned and food zones, and also to prevent a collision). When robots in a blocking group used signals in poisoned areas, they tried to prevent their teammates to reach these zones. The adaptive advantage of signaling poisoned zones is to avoid decaying fitness levels in the entire population; also to preserve their chromosomes throughout the evolutionary process (as some form of elitism).

When an experiment on evolutionary robotics is configured which involves elements such as a communication system and homogeneous populations; we have to consider what kind of situations produce a set of conditions that may boost results of robots producing signals to improve their fitness through evolution. Hence, as demonstrated in the Blocking Task Experiment, a developing communication system has an important relationship with the evolutionary importance of signals. In general, communication systems allow robots to efficiently solve collective tasks and evolutionary advantages promote information interchange.

Another important factor that have an impact on the emergence of signals is the environmental complexity level. A change of this element during the Multi Signal Experiment, caused signals to emerge in different places. We can safely assume that the complexity of environment has a relation with the place where signals emerge. When the complexity is high, the evolutionary process in this experiment developed robots with less capacity for producing food signals. This is related to the difficulty that robots have to solve the task when the environment and the search space were more complex to explore. In a high complexity scenario, it is less probable that robots can find food areas. Additionally, the evolutionary importance of signals was reduced in comparison to a basic complexity sce-

nario. Then the importance of producing signals outside of the food zones was improved through evolution.

The most important result of the dependent variables manipulation in the Multiple Signals Experiment, was the relation between the complexity levels and emerged defined-signals. As a consequence, a more elaborated scenario created the conditions for the emergence of more defined-signals that represent meaningful cues with a lexical value. This can be explained as a complex scenario that favored the emergence of complex behavior, which in turn represented more complex communication systems. Such systems associate more signals to different environmental stimulus (De Greeff & Nolfi, 2010). On the contrary, fitness levels decreased when the complexity of the environment increased. The relation between fitness and the manipulation of world complexity is the attempt of robot populations for solving the task using more complex tools.

As for the Signals in a Complex Scenario Experiment we noticed some important features that regulated the emergence of signals in the evolutionary communication systems. There was a trend associated to the fitness performance in groups with a hidden layer in their control system that was showed in the fitness of group-3. Thus, individuals in group-3 showed a better evolutionary adaption with a multi-signal communication system than those individuals with a simple communication system and a hidden layer (i.e. group-1, and group-2).

It is important to mention that control systems with a hidden layer (in group-1 and group-2) decreased their performance compared to control systems without a hidden layer (control-group). However, the performance also incremented when the number of potential signals was increased by the control system with a hidden layer (group-4). The presence of a hidden layer in a control system increases the number of weights to evolve, and consequently the complexity of the search space increases; thus making difficult to establish reliable communication system. In turn, for robots in the control group, evolution produced individuals with the ability to solve the main task. After artificial evolution ended for the optimized robots in group-1, fitness was decreased because the ANN and the search space were more complex than in the control group.

We observed that as soon as communication systems became more complex, the evolutionary processes produced new tools to improve the search in more complex search spaces. Thus, complex tools provide additional mechanisms for the evolutionary process to navigate through complex search spaces. This improvement was observed in the statistical test of group-3. The use of complex tools is necessary when both the control system and search space are complex. Some of these tools are developed as potential signals that establish different lexical values to different events.

In the Signals in a Complex Scenario Experiment the systems used in the groups, control-group and group-1, represent a basic complexity scenario because robots used at most two signals to establish communication. In contrast, robots in group-3, in the same experiment, had the highest level of complexity in their communication systems. The latter because the use of a multi-signal communication system that produced multiple signals associated with multiple lexical values. We may conclude that simple communication systems have better results in terms of fitness on simple environments. On the contrary, complex communication systems have better results on more complex environments.

In this experiment we noticed that when the complexity of the communication systems increased, the number of robots using signals near the black line also increased. An increase of signals related to the black line corresponds to high levels of fitness. In contrast in the control-group when the communication system was composed of two signals, the evolutionary process produced robots that signaled food zones. Similarly to the first experiment, robots used this strategy for attracting other robots to the food zones; thus, evolution developed these behavior to solve the task. In this way, when robots are equipped with a communication system with more signals, they can associate more situations with different signals. In other words, a complex communication system allows robots to use more signals under specific situations.

The communication system in group-3, which had more potential signals, produced more defined-signals. The communication system of this group demonstrated that a number of potential signals are an important feature, also an evolutionary advantage for evolving robust individuals. Thus, the trend observed in the results of the Signals in a Complex Scenario Experiment is based on variables such as the number of emerged signals , the number of defined-signals, and the replications that did not produce a communication system. The use of hidden layers increases the discrimination capability for the evolving individuals; however the use of a hidden layer does not guarantee establishing a reliable communication system.

In group-1 the use of a simple communication system did not produce individuals that associated signals with multiple situations. This means that less elaborated communication tools were insufficient and inappropriate in terms of fitness levels. This feature supports the idea that complex search spaces can be better explored with the appropriate motor sensory tools. When the number of potential signals was increased, the probability of finding meaningful signals also increased. For this reason, the number of defined-signals and places where signals emerged augmented with an increase of potential signals.

4. Conclusions

In our experiments we identified some important environmental features that contributed to the emergence of signals using the evolutionary robotics approach. Variations on the evolutionary values of the task to be resolved produced changes in the location where signals emerged. Additionally, changes in the complexity levels of the environmental configurations forced evolution to use complex communications systems. The number of potential signals in a communication system increases the performance of robots and the possibility of obtaining a robust communication system with extra signals associated with lexical values. A robust communication system helps to improve results of the evolutionary processes; hence increasing the capacity of individuals to solve the task, set in the last experiment, of discriminating poison and food.

As future work, we believe is important to study specific communication features, their syntaxes and semantics. Furthermore, in order to find out whether the identified features in this paper present interaction with each other (e.g. the complexity of the environment and the communication system). Finally, we aim to discover if there are more factors that regulate the emergence of evolutionary communication systems.

Acknowledgments

This work was supported by the Sistema Nacional de Investigadores (Exp. 30026 [FMMG]), and F. Aldana-Franco received fellowship from CONACyT (Reg. 377475)

References

Ampatzis, C., Tuci, E., Trianni, V., & Dorigo, M. (2008). Evolution of signaling in a multi-robot system: Categorization and communication. *Adaptive Behavior*, *16*(1), 5–26.

Bongard, J. C. (2013). Evolutionary robotics. *Communications of the ACM*, *56*(8), 74–83.

Cangelosi, A. (2001). Evolution of communication and language using signals, symbols, and words. *IEEE Transactions on Evolutionary Computation*, *5*(2), 93–101.

De Greeff, J., & Nolfi, S. (2010). Evolution of implicit and explicit communication in mobile robots.. *Evolution of communication and language in embodied agents*, *1*, 179.

Floreano, D., Mitri, S., Magnenat, S., & Keller, L. (2007). Evolutionary conditions for the emergence of communication in robots. *Current biology*, *17*(6), 514–519.

Floreano, D., Mitri, S., Perez-Uribe, A., & Keller, L. (2008). Evolution of altruistic robots. *Computational intelligence: Research frontiers*, 232–248.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.

Loula, A., Gudwin, R. R., & Queiroz, J. (2010). On the emergence of indexical and symbolic interpretation in artificial creatures, or what is this i hear?. In *ALIFE*, pp. 862–868.

Marocco, D., Cangelosi, A., & Nolfi, S. (2003). The emergence of communication in evolutionary robots. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, *361*(1811), 2397–2421.

Massera, G., Ferrauto, T., Gigliotta, O., & Nolfi, S. (2013). Farsa: An open software tool for embodied cognitive science.. In *ECAL*, pp. 538–545.

Mitri, S., Floreano, D., & Keller, L. (2009). The evolution of information suppression in communicating robots with conflicting interests. *Proceedings of the National Academy of Sciences*, *106*(37), 15786–15790.

Mitri, S., Floreano, D., & Keller, L. (2010). Relatedness influences signal reliability in evolving robots. *Proceedings of the Royal Society of London B: Biological Sciences*, rspb20101407.

Montes-Gonzalez, F., & Aldana-Franco, F. (2011). The evolution of signal communication for the e-puck robot. *Advances in Artificial Intelligence*, 466–477.

Nolfi, S. (1998). Evolutionary robotics: Exploiting the full power of self-organization. *Connection science*, *10*(3-4), 167–184.

Nolfi, S. Emergence of communication and language in evolving robots..

Rasheed, N., & Amin, S. H. (2016). Developmental and evolutionary lexicon acquisition in cognitive agents/robots with grounding principle. *Computational intelligence and neuroscience*, *2016*, 16.

Schwehm, M. (1996). Parallel population models for genetic algorithms. *Universität Erlangen-Nürnberg*.

Scott-Phillips, T. C., Blythe, R. A., Gardner, A., & West, S. A. (2012). How do communication systems emerge?. *Proceedings of the Royal Society of London B: Biological Sciences*, *279*(1735), 1943–1949.

Solé, R. V., Corominas-Murtra, B., Valverde, S., & Steels, L. (2010). Language networks: Their structure, function, and evolution. *Complexity*, *15*(6), 20–26.

Steels, L. (2003). Evolving grounded communication for robots. *Trends in cognitive sciences*, *7*(7), 308–312.

Steyven, A., Hart, E., & Paechter, B. (2015). The cost of communication: Environmental pressure and survivability in medea. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pp. 1239–1240. ACM.

Trianni, V., Labella, T. H., & Dorigo, M. (2004). Evolution of direct communication for a swarm-bot performing hole avoidance. *Lecture notes in computer science*, *3172*, 130–141.

Waibel, M., Keller, L., & Floreano, D. (2009). Genetic team composition and level of selection in the evolution of cooperation. *IEEE Transactions on Evolutionary Computation*, *13*(3), 648–660.

Wischmann, S., Floreano, D., & Keller, L. (2012). Historical contingency affects signaling strategies and competitive abilities in evolving populations of simulated robots. *Proceedings of the National Academy of Sciences*, *109*(3), 864–868.

Anexo 5.

Implementation of a simulation model for ROV in a decision task problem.

Implementation of a simulation model for ROV in a decision task problem.

Fernando Aldana-Franco[1], Fernando Montes-González[1], Alberto Ochoa-Zezzatti[2]
[1]Artificial Intelligence Research Center, University of Veracruz
[2]Applied Computing Master. Juarez City University

faldana@uv.com

Abstract . A multiplayer version of SimROV, a simulator of Virtual Reality and Artificial Intelligence, is presented. A ROV was tested in an underwater scenario composed by oil wells. SimROVincludes a console model for the operator and it is used to train more than one students or operators. All participants can control their own ROV, and all ROVs interact between them. There is a role of an instructor which introduces change on environments as ocean currents and fuel leakage. Students have to solve all problems in the virtual scenario. All solutions are detected and complemented by a guide that uses an intelligent guide based on artificial intelligence. SimROV is a virtual training system with educational purpose.

Keywords: ROV, Simulator, Multiplayer, Artificial Intelligence, Virtual Reality.

1 Introducction

Teleoperation in robotics is a field in where human operators done control on robots remotely (Iannuzzi, Grant, Corriveau, Boissy & Michaud [5]). In this area, robots are provided by electronic devices which allow to connect to computers that are at a distance point. The most popular way to connect robots and computers is using a Wireless Local Area Network (Miyagusuku, Yamishita & Asama [10]). But it is possible to use different techniques as radiofrequency point to point connection, or specific communication protocols as ZigBee. Teleoperative robots have complex architectures, and are dotted by a robust sensorial and motorize system (Coad, Okomura, Wren, Mintz, Lendvay, Jarc & Nisky [1]).

In this manner, robots depend in humans to complete a task. Operators use control systems base on consoles with the aim of manipulate robot components. And an essential component is a real time vision system that provides information about the environment in where the robot is situated.

Teleoperative robots are used in environments and task that result dangerous to humans. Deactivation of explosive, handling of chemical, surgery, mining tasks, aerospatial tasks, exploration, and tasks of oil industry are some example of application. In all field of application, humans have to train to improve their abilities to control

this robots (Shukla, & Karki [15]). And in many cases this robots are expensive, and operators have to train in simulators and virtual environments (Shukla, Karki, Bahera & Jamshidi [17]).

Our propose is to design and test a multiplayer version of SimROV, a simulator for ROV (Remote Operated Vehicle) used with educational purposes. The final users are operators or students in training for oil industry. Virtual ROVs are controlled by users that have to drive them into specific situations like ocean currents. In this version, an instructor and students use a network connection to share the same scenario and interact. The system includes artificial intelligent algorithm used to know the actions of participants and to guide operators in order to meet the goal proposed by instructor. In that way, trajectories of ROVs are verified and compared with the intention of known if students follows the correct route to solve the task assigned. When a trajectories is wrong, the system warning operators to return to the correct way. The optimal trajectory is calculated using Dijkstra algorithm.

ROVs are equipped by two arms with the purpose of manipulate situations, and two cameras for capturing video in real time. These vehicles are tested for students. One of the most difficult aspects is to control all movements and tasks of ROV in an underwater scenario which includes: movements of arms, stability of ROV and camera control. For that reason it is important to detect the ROV position in order to create an intelligent system that helps students to improve their capabilities to drive vehicles.

Thus we design and test an intelligent system which is used to help students, instructors and companies in oil industry to train personal that have to acquire abilities to drive underwater vehicles. Our simulator is a smart tutor and is used by multiplayer or multi-operators scenarios. Also SimROV has a modular architecture that facilitates the organization and design of the simulator.

The article is organized as follows. In section 2 some simulators are presented. In section 3 is presented or SimROV version. In section 4 is showed the discussion about our SimROV version, and in section 5 the conclusion of our development.

2 Simulators

A simulator is a computational system used to represent characteristics of real world in computers. About training in industry, a simulator is a system that allow to acquire specific abilities (Gopinath & Sawyer [3]). In that way, it is possible consider two types of simulators (Merino et al. [9]):

- Oriented to Installation Design Simulators (OIDS).
- Training Simulators (TS).

At the same time, Training Simulators are divided into:

- Full-Scale Simulators (FSS). Reproduce all operative environment, in where computational details of control console is high and reproduce functionallity and apperance.

- Process Reporduction Simulators (PRS). In this case simulators do not reproduce all details of control console, but reproduce operating times.

SimROV is a TS and FSS simulator. It includes a virtual environment and all controls for operate ROVs. Environmental comprise details of ROVs operation as oil wells exploration and task and the button of sea complement. About ROV control consoles, SimROV uses keyboard and mouse for controlling position and arm movements. In the real robot, all controls are based on joysticks.

It is possible to personalize the instruction for students. This because of Artificial Intelligence Algorithms that are the core of the intelligent system for training personal. This simulator monitors and guide students in their training processes. Additionally, it is possible to modify situations when instructors produce a change in the environment.

3 SimROV Architecture

SimROV architecture is based in requirements of the International Marine Contractors Association (IMCA). This association defines a simulator as a process in which conditions are created in an appropriated virtual and computational model (IMCA [4]). Also IMCA classifies two version of simulators:

- Class A: Uses a realistic reproduction of physical model that includes a visual representation.
- Class B: Uses a semi-realistic reproduction of physical model and includes a visual representation.

SimROV is considered a class A simulator (Olguín-Tolentino, Ingram-Ramírez, Pérez-Ramírez, Nava-Ayala & Hernández-Pérez [14]) because it reproduce the control console of ROVs, and some operations that are done by real operators in industry as change and repair pipes in oil wells in underwater scenario. Figure 1 shows the graphical representation of its architecture. It is a multiplayer modular simulator, in where students and the instructor have their own interface. Interfaces are different because the role is different, but environmental is the same for both roles.

The instructor uses an interface to control and observe all situations and processes done by operators. There is a data base which store processes and manipulation of students, and the optimal route calculated by our intelligent system. This data base is available to instructor who access and verify processes.

In the other hand, students control a ROV using an interface based on keyboard and mouse. All operations done by operators are stored in the data base. The intelligent system is a guide that feedback operators with the intention of correct a wrong route. Operators and instructor interact using a network that works using client-server model. This characteristic allow to users interact and share the same virtual environments.

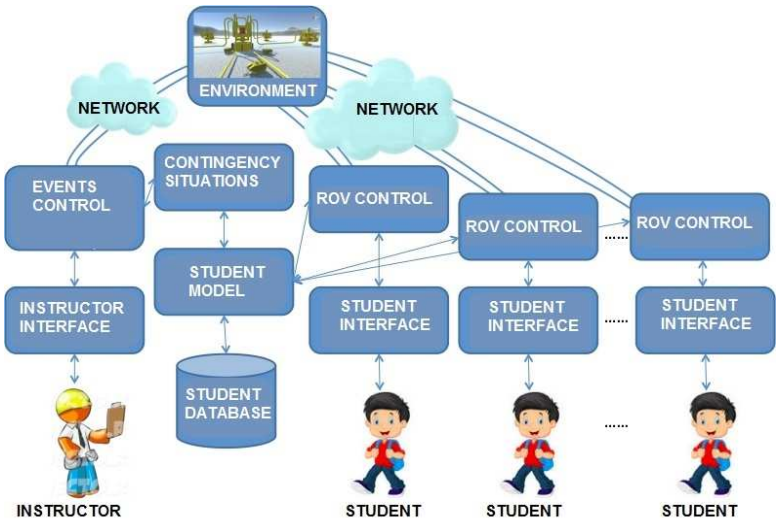


Fig. 1. Multiplayer SimROV architecture. Students interact in a virtual world between them. Instructor change environmental factors in order to test students during training process.

Architecture is composed by different components or modules:

1. *Interface*: This module allows to access to virtual world in a network. A user access in two roles: student (operator) and instructor (See figure 2).

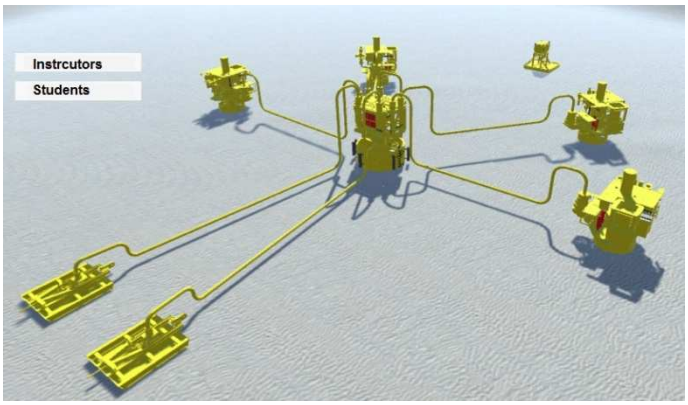


Fig. 2. Multiplayer SimROV Interface: Virtual environmental. It is possible to access as an operator (student) or as an instructor.

Instructor role has permissions to modify virtual environment, accepts to begin operations and observes students training. Interface for instructors includes two different cameras: student camera (first person) that allow to observe all manipulation of each student in a real time, and a global camera in where all process is showed. A data base

is included and related to instructors and students. So data base is a component in where all actions of all students are stored and are accessed by the instructor with the intention of verify individual actions.

Student interface allows to control ROVs. Motor motion that changes the position of robots. Also control arms: left arm is used as a help tool and right arm is used in manipulation processes and task assigned by instructors.

It is implemented an Artificial Intelligence algorithm in order to calculate the shortest route. This is the core of the intelligent system. It uses the data base that stored all routes of each student with the intention of create a guide which is used by students to improve routes real time. In that way, when a student deflects his way, the intelligent system sends a message for change the way.

2. *Same virtual environment for all users*: Both interfaces reproduce the same environmental for the instructor and operators. In that way it is used the multiplayer Network Manager method (Unity [23]). It uses an intranet or Internet for interchange data. An important characteristic is that environments reproducing an underwater world in where ROVs work and interact. Each ROV in the aquatic world is controlled just by one student. Animation of ocean currents and movements of ROVs are not pre-defined. It depends on movements of ROVs and changes in environments. Arms and translation of ROVs are originated with the control of students.
3. *ROV control*: Operators have the total control of ROVs, changing the direction of vehicles using the keyboard arrows. Also it is possible to control the right arm that allows to hold a hook and reduce the influence of currents. Then the left arm is free and is used to complete different tasks requested by the instructor.
4. *Events control*: The Instructor has the permissions in the simulator to change the environmental conditions. It is represented by contingency situations, open and close valves, change the direction of currents, create leaks.
5. *Contingency situations*: This module contains information about contingency situations, as they was defined previously. Thus information about how each student solve contingency situations are stored in the data base. It means that all procedures done by each user is stored at the data base. These procedures comprise when a valve is closed after a leak, or when a student change a section pipeline in the same situation. This module are affected by the artificial intelligence motor, and tries to guide operators to improve their solutions in a particular contingency situation.
6. *Student Model*: Contingency situations allow to evaluate two aspects: a) operators capability to move a ROV, and b) operation of arms. In that way, both aspects are registered in the data base as movements of ROVs and control of arms for each student. The idea is bring an efficient tool to instructors for evaluate operators and their processes and actions.

4 Develop.

4.1 Virtual Environment.

Objects are created using 3D models. They are imported to Unity, and in order to obtain a real appearance, objects are assigned with Rigidbodie and Mesh Collider properties. Regidbodie adds a physical and mathematical model. Thus objects react in a real way (Unity [22]). Also objects are affected by gravity. Interaction between objects are modeled by Physx, a physics motor of NVIDIA (NVIDIA [12] Craighead, Murphy, Burke & Goldiez [2]).

On the other hand, Mesh Collider represents a computational model for collision between objects (Unity [24]). Thus it is not used primitive forms, instead it is created a mesh with the physical form of the object. Then collision are enable regardless of object shape.

The stage is broad with the intention of promote the interaction between users. Environment is based in an underwater world defined using Marine Well Company Criteria (see figure 3).

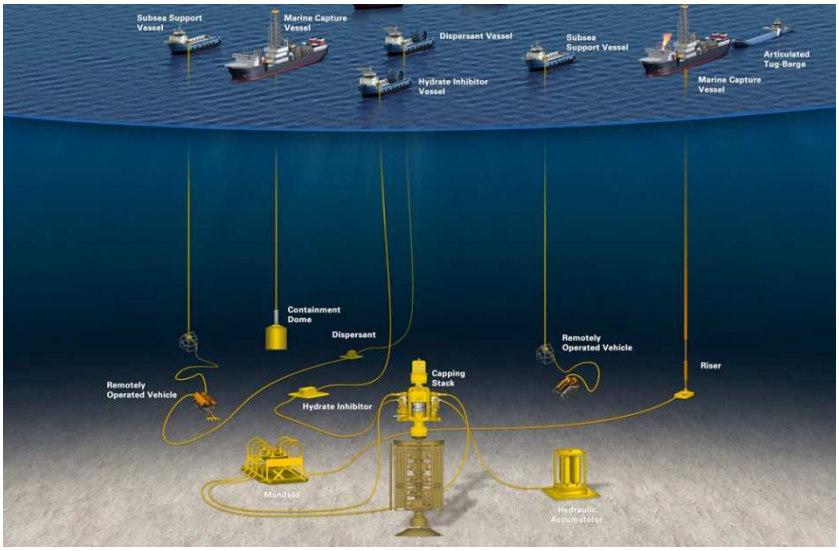


Fig. 3. Underwater virtual environment (Marine Well Containment Company [8]).

4.2 Multiplayer.

Students and instructor share the same underwater virtual environment. But each user has different properties and permissions. Students operates ROVs movements, arm control and actions. Instructor observes all students movements and procedures, and change the conditions of virtual environments. For that reason, two different consoles are designed. In the console of students, it is available the control of ROV movements and arms control. Also tasks assigned are observed by instructors. In the other hand,

Instructor console allow to assignee particular task to each operator. Also it is possible to observe in real time operators processes using cameras.

As it is used a network connection, the client-server model is used. The instructor console is configures as server, and operators are configured as clients (see figure 4). As it is necessary that the instructor is present in the simulation, the instructor is configured as a client too. Thus, the host is a client in the same process. As the host (instructor) is a special case, it is considered a Local Client. The rest of clients (operators) are considered Remote Clients (Unity [23]).

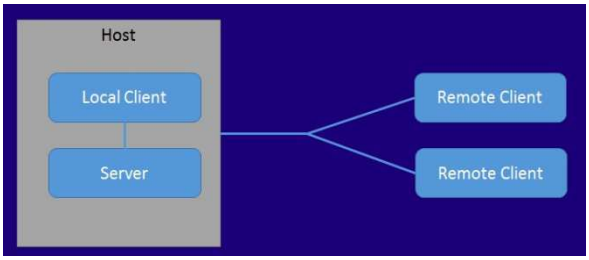


Fig. 4. Network configuration.

All students have a replication of the same console. For that, it is necessary to add an identifier. In this case is represented by a "NetworkIdentify" that represents the control identification of operators as a network object. That means each operator has just one ROV and one identification associated.

4.3 Algorithm.

The structure of simulator and intelligent system trace reference points that allow to identify the position of all oil wells or places in where students applied a particular process. All the routes and ways are stored in the data base of the system. When the instructor assignee a particular task to operators, the intelligent system calculates the shortest way form the first position of ROV to the final point in where operators have to solve a task or to apply a specific process with the intention of solve a contingency situation. In that way, the intelligent system guides operators in the use route calculated. This is a support system that allow to operators to improve their abilities and to solve the assigned task.

Dijkstra algorithm is used when it is necessary to obtain the way with lowest cost associated within a set of previously defined ways (see figure 5). This algorithm is used to calculate the shortest route. Thus Dijkstra algorithm is implemented and codified in C#.

```
G = (V,E) donde V es el conjunto de vértices y
    E el de arcos.
S   es el conjunto de vértices cuyos caminos
    más cortos al origen han sido ya
    determinados.
V-S es el resto de vértices.
d:  ARRAY de estimaciones de caminos más
    cortos a dichos vértices.
pr: ARRAY de predecesores para cada
    vértice.
<<Inicializar d y pr>>
<<Poner S = ∅>> // aún no hemos estudiado
ningún vértice
    While {V-S} ≠ ∅ // mientras queden
    nodos sin determinar su camino mínimo al
    origen
        <<Ordenar los vértices en V-S y
        analizar de acuerdo a la menor
        distancia al origen>>
        <<Añadir u, el vértice más cercano en
        V-S, a S>> // S = S + {u}
        <<Recalcular la distancia a todos los
        vértices todavía en V-S adyacentes a
        u>>
```

Fig. 5. Pseudocode for Dijkstra algorithm for the shortest route (Torrubia & Terrazas [20]).

In that way, as each ROV is in movement, the intelligent system detects change in the ideal route and informs to users that have to correct their way. This is possible because we use reference points included in Dijkstra algorithm. Figure 6 shows this reference points and their operation ranks.

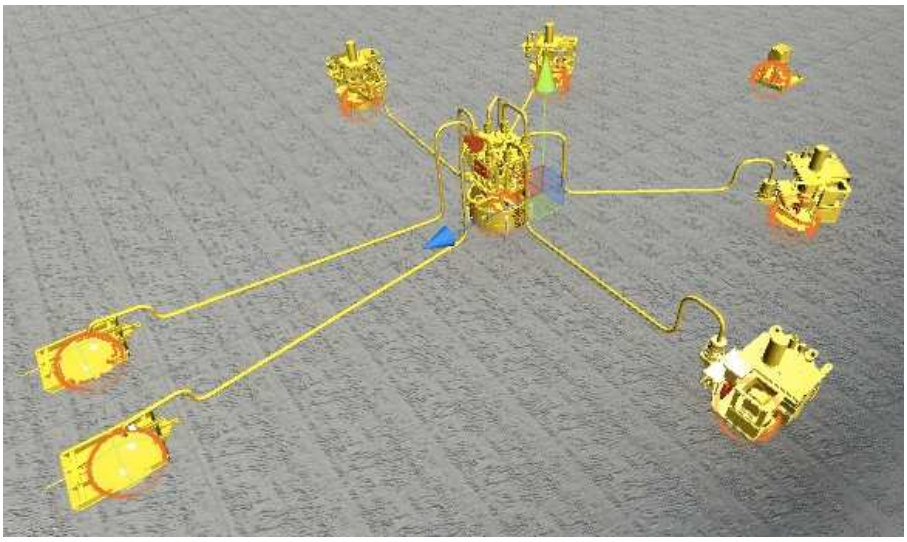


Fig. 6. Reference points produced by Dijkstra algorithm.

4.4 Software.

"3DS Max" software is used to obtain the 3D design models in simulation. This software produce virtual models which animates and renders the virtual environments

On the other hand, "Unity3D" is used to develop the application and to codify all scrips. This is a core for videogames.

For dynamical animations to obtain interaction between users, it is used Ageia PhysX (Tumult [21], Craighead, Murphy, Burt & Goldiez [2]). It is the physical information core of UNITY.

5 Discussion

SimROV is a simulator that involves different computational tools. The purpose of this simulator is improve educational processes implicated in training processes for teleoperative robots (Tanwani & Calinon [19]), specially to train operator for ROVs in oil industry (Shukla, & Karki [16]). It is a complete simulator that belongs to Class A system (IMCA). In addition to its simulation power and its applications, an intelligent system is included. For that reason, it is possible to say that SimROV is a learning tool. Thus, our intelligent system is a smart tutor that guides the learning process of users. Also uses Virtual Reality and operates in real time. It means that this simulator tries to reduce the cost associated to training personal in real ROVs. Virtual reality uses dynamical animation, which means that movements are not predefined and are produced in real time through the interaction of users and the virtual environment.
This simulator can be incorporated to oil industry. It is a virtual platform to prepare and train individuals that have to work in the deep of the ocean. Our simulator are

prepare to train people who have to do: a) Installations in oil wells, b) process like repair and change pipes, c) cleaning oil spilled in the sea, that allow to avoid important damage in ocean, d) inspection, e) maintenance, and f) reparation. Mexico is a country with an important oil industry, and nowadays the national company needs to extract oil in deep sea. Our intention is to create a tool that can be used by national and international. Thus, costs are reduced in oil industry produced by acquisition of software (Kohlbrecher & Von Stryk [6]), which could represent an increase of foreign investment in this industry in Mexico (Ochoa-Zezzatti, Sánchez, Hernández-Aguilar & Pérez [13]).
Our simulator has a modular structure, which means that the work is divided and it is possible to add new components in an easier way. This characteristic allow to add new characteristics, properties and components (Leitner, Harding, Föster & Corke [7]). Finally, is important to show that simulators can be design for a particular problem. It is not necessary to depend on commercial simulators, which prices are higher. Open source and open access simulators are been developed around the world with the intention of reduces cost and favor the development of technology in many fields of knowledge.

6 Conclusiones

We presented a multiplayer simulator based on virtual reality and artificial intelligence. It is applied to oil industry, especially for ROV and underwater activities.. Virtual environments include computational models of oil wells, multiple ROVs. Each ROV represents an operator that is in training. We use a client-server model in order to provide inter-connection to users through a LAN (Local Area Network). Thus each operator has a ROV which allows to them to solve a task. This task is generated by an instructor. The instructor can modify the environmental and follow the performance of students. As our simulator is a smart tutor, it is integrated an artificial intelligence algorithm. This calculate the shortest route for solving a particular task. Also guide operators from the initial point of the task to the final. When an operator change the calculated route, the intelligent system notifies that the user has to modify the route. Thus, time for solving a task is optimized and allow to operators to learn how to operate a ROV in a real situation.

Next step in our development is to increase the number of simulated task, particular situations, and to reach the computational model that represents SimROV. It is necessary to know how experts in oil industry can improve the situation included through their experience. In the other hand, the intelligence system included in SimROV can add new algorithms. It is necessary to include more intelligence power, as Evolutionary Computing algorithms that increase the capacity of optimization of routes. Also includes new mechanisms to guide the learning and training processes like Artificial Neural Networks that can be incorporated to SimROV to improve routes in real time and to support a guide for arm movement process (Yusoff, Zain, Sharif, Sallehuddin & Ngadiman [25]).

Additionally it is possible to use Evolutionary Robotics techniques (Silva, Duarte, Correia, Oliveira & Christensen [18], Montes-Gonzales & Aldana-Franco [11]) with the aim of simulate prediction for solutions that are in process by operators. In this manner, simulator tends to prevent errors in solutions through a simulations that evaluates potential solutions.

7 References

1. Coad, M.M., Okomura, A.M., Wren, S., Mintz, Y., Lendvay, T.S., Jarc, A.M., & Nisky, I. (2017). Training in Divergent and Convergent Force Fields During 6-DOF Teleoperation with a Robot-Assisted Surgical System. In World Heptics Conference, 2017, IEEE, 195-200.
2. Craighead, J., Murphy, R., Burke, J., & Goldiez, B. (2007) A survey of commercial & open source unmanned vehicle simulators. In Robotics and Automation, 2007 IEEE International Conference on, 852-857. IEEE.
3. Gopinath, C. & Sawyer, J. (1999). Exploring the Learning from an Enterprise Simulation. Journal of Management Development, 18(5), 477-489.
4. IMCA. (2016). International Marine Contractors Association. London, England: Guidance on the Use of Simulators. https://www.imca-int.com/?s=simulator
5. Iannuzzi, D., Grant, A., Corriveau, H., Boissy, P., & Michaud, F. (2016). Specification of an integrated information architecture for a mobile teleoperated robot for home telecare. Informatics for health & social care, 41(4), 350.361.
6. Kohlbrecher, S., & Von Stryk, O. (2016). From RoboCup Rescue to Supervised Autonomous Mobile Robots for Remote Inspection of Industrial Plants. Kl-Künstliche Intelligenz, 30(3-4), 311-314.
7. Leitner, J., Harding, S., Föster, A., & Corke, P. (2016). A modular software framework for eye-hand coordination in humanoid robots. Frontiers in Robotics and AI, 3, 26.
8. Marine Well Contaiment Company. (2012). Containment System Overview. MWCC's Marine Well Contaiment System, 1, 1.
9. Merino A., Pelayo S., Rueda A., Alves R., García A., Acebes F., de Prada C., Gutiérrez G. & García M. (2017). Un simulador de alcance total para la formación de los operarios de sala de control de factorias azucareras. In 2ª Reunión de Usuarios de EcosimPro, UNED.
10. Miyagusuku, R., Yamishita, A., & Asama, H. (2016). Distributed algorithm for robotic network self-deployment in indoor enviroments using wirless signal strength. In Intelligent Autonomous Systems 13, 1491-1502. Springer.
11. Montes-Gonzalez, F., & Aldana-Franco, F. (2011). The evolution of Signal Communication for the e-puck Robot. Advances in Artificial Intelligence, 466-477.
12. NVIDIA. (2016). NVIDIA Corporation. USA: Tecnologías NVIDIA PHYSX. http://www.nvidia.es/object/nvidia-physx-es.html PhysX
13. Ochoa-Zezzatti, C.A.,Hernández-Aguilar, A., & Pérez, R. (2016). Improving an Industrial problem optimizing the material in car seats. International Journal of Combinatorial Optimization Problems and Informatics, 7(1), 54.
14. Olguín-Tolentino N. D., Ingram-Ramírez J.W., Pérez-Ramírez M. (IEEE Member), Nava-Ayala E. R., Hernández-Pérez M. Y. (2016). Prototype of a ROV simulator using dynamic animation and Virtual Reality. In CIIDET-Congreso Internacional sobre Innovación y Desarrollo Tecnológico 2016, 68.
15. Shukla, A., & Karki, H. (2016). Application of robotics in offshore oil and gas industry- A review Part II. Robotics and Autonomous Systems, 75, 508-524.

16. Shukla, A., & Karki, H. (2016). Application of robotics in onshore oil and gas industry- A review Part I. Robotics and Autonomous Systems, 75, 490-507.
17. Shukla, A., Karki, H., Behera, L., & Jamshidi, M.M. (2016). Teleoperation by Using Nonisomorphic Mechanisms in the Master-Slave Configuration for Speed Control. IEEE Systems Journal.
18. Silva, F. Duarte, M., Correia, L., Oliveira, S.M., & Christensen, A.L. (2016). Open issues in evolutionary robotics. Evolutionary computation, 24(2), 205-236.
19. Tanwani, A.K., & Calinon, S. (2017). A generative model for intetntion recognition and manipulation assistence in teleoperation. In IEEE/RSJ Intenational Conference on Intelligent Robots and Systems.
20. Torrubia, G. S., & Terrazas, V. L. (2012). Algoritmo de Dijkstra. Un tutorial interactivo. In VII Jornadas de Enseñanza Universitaria de la Informática (JENUI 2001).
21. TUMULT. (2016). Tumult Inc. USA: Tumult Hype Documentation. http://tumult.com/hype/documentation/3.0/#physics
22. Unity. (2016). Unity Technologies . USA: 3D physics reference. https://docs.unity3d.com/es/current/Manual/class-Rigidbody.html
23. Unity. (2016). Unity Technologies. USA: Documentation NetworkManager. https://docs.unity3d.com/es/current/Manual/UNetManager.html
24. Unity. (2016). Unity Technologies . USA: Mesh Collider. https://docs.unity3d.com/es/current/Manual/class-MeshCollider.html
25. Yusoff, Y., Zain, A.M., Sharif, S., Sallehuddin, R., & Ngadiman, M.S. (2016). Potential ANN prediction model for multiperformances WEDM on Inconel 718. Neural Computing and Applications, 1-15.

Anexo 6.

Acetyl-Modulated Architecture for Evolutionary Robotics.

Acetyl-modulated architecture for evolutionary robotics

Aldana-Franco F.[1], Montes-González F.[1], Ochoa-Zezzatti A.[2]
1 Artificial Intelligence Research Center. University of Veracruz, Xalapa, Mexico.
2. City of Juarez Autonomous University, City of Juarez, Mexico.
faldana@uv.mx

Abstract. Control modularity is an important tool for improving the organization and potential task resolution in Evolutionary Robotics. Neuro-controllers are divided into many sections or many networks that are coordinated in order to solve a particular task. In this paper, we present a model for control evolutionary robots inspired in the effects of acetylcholine neurotransmitter, chemical synapse, renshaw cells, and based on artificial neural networks. The performance of our model is compared to other two implementations: a) a system controlled by one single neural network and b) a control system based in one neural network divided in different sections. A garbage collection and room swap tasks were used for the experimentation in order to validate the proposed model. As for the simulator, we decided to use a commercial platform as Webots, which includes a virtual e-puck robot in a 3D environment.

Keywords: Evolutionary Robotics, Neural Networks, Modularity, Acetyl-Coline, chemical synapses, E-puck.

This work was supported by the Sistema Nacional de Investigadores (Exp. 30026 [FMMG], Exp. 200825 [BLG]). F. Aldana-Franco (Reg. 377475) received a fellowship from CONACyT.

1 Introduction

Evolutionary Robotics (ER) is a Robotics field where control and morphological components are set under evolutionary artificial pressure that optimizes the desired components. Hence, ER is a field where software and hardware components are modeled. Robust solutions are obtained after the optimization process, and the worst solutions are discarded.

The most popular object of optimization in this field are Artificial Neural Networks (ANNs) neural network controllers or neuro-controllers (Montes & Aldana [47], Loula, Gudwin & Queiroz [37], Mitri, Floreano & Keller [44] & [45]). Amongst other solutions that can be combined using this approach, is possible to find tools such as: Fuzzy Logic (Mendel [42], Jamil, Jalani & Ahmad [31]); Based Rules Classification Systems (Kodjabachian & Meyer [35]); and Bayesian Networks (Inamura, Inaba e Inoue [28], Calandra, Seyfarth, Peters & Deisenroth [9]).

The weights of the ANNs are codified in a straightforward manner in order to be optimized using Evolutionary Computation (EC) algorithms; a commonly used algorithm is a Genetic Algorithms (GA) (Nolfi, Bongard, Husbands & Floreano [50]). Therefore, robots sensors are the inputs of the neural controller and actuators are linked to neurons at the output layer. Mainly optimization works on neural weights but is possible to evolve morphological components of the ANNs.

When a neuro-controller is optimized, the work realized by the EC algorithm is to search a robust configuration, which facilitates robots to solve a particular task. Depending of the complexity of the task, neuro-controllers can be simple or elaborated selection mechanisms. This add complexity to the search space of potential solutions.

It is widely known that when an ANN has to solve more than two classification or control tasks the search stalls because of the complexity of search space. In other words, suppose we have two control tasks: A and B. The optimal weights to solve task A

and task B are in different sections of the complete search space. The algorithm adjusts weights to find an optimal solution for task A, but solution for task B is missed. Therefore, the solution for task A blocks the task B solution. In Artificial Vision, this problem is identified as Neuronal Blocking (Jordan, Jordan & Barto [30]). In ER, a similar problem is named Search Space Blocking. In the literature (Calabretta, Nolfi, Parisi & Wagner [8]), it has been identified that the task of cleaning environments produces this problem when robots have to deposit and collect garbage. Robots tend to collect garbage but are stuck depositing garbage when one single ANN controls both sub-tasks (collection and depositing).

Proposed solutions to this problem consider differentiating ANNS mechanisms to employ a divide and conquer strategy, which is similar to living beings dividing functions through organs, apparatus, and systems (Newman [48]). In particular, this solution can be roughly compared to division in the human brain, which solves different tasks in a cooperative way through different specific and neighbor regions. In humans, modularity is the product of evolution that has specialized regions that are able to interact with each other (Cho & Shimohara [12]).

Modularity in the human brain is the product of large number of neurons compared to a limited number of connections, 100,000 millions of neurons having 7,000 connections for neuron (Ballard [3]). Focused specialization in particular regions of the human brain is the consequence of interactions between neurons and neighbor regions (Geschwind & Galaburda [21]). This propriety is fundamental for successful multi-task systems (Potter, Meeden & Schultz [53]).

In biology, modularity can be used as a form of organization to describe simple organisms (Bolker [4]). In computer sciences, modularity is considered an operation that facilitates decomposition of the search space where each individual part is as complex to explore as the set of the altogether components (Hart, Kammeyer & Belew [24]). Therefore, an ER a module is considered a part of a whole solution (Khare, Yao, Sendhoff, Jin & Wersing [34]) where interaction mechanisms are a basic tool for improving artificial evolution.

The modularization of a control system in ER depends on the complexity of tasks (Di Fernando, Calabretta & Parisi [14]). When robots have to solve a simple task, a modular control system may not be necessary. On the other hand, when a complex task is evolved thus a complex control system has to be employed. The possibility of finding a locking condition in the search space is increased with complex tasks and environments (Jacobs & Jordan [29]). It has been identified that this kind of interference happens at the beginning of the search process (Calabretta, Di Fernando, Parisi & Keil [6]).

Modularity in ER is based in the implementation of fixed ANNs as blocks that process sensorial information together with system values whereas other mechanisms work with action selection arbitration between different modules (Floreano, Dürr & Mattiussi [17]). Thus, we identified two possible approaches: internal and external modularity. Then, we divide ANNs in an internal way where groups of neurons of one ANN are forced to specialize for attending a specific part of the whole problem (internal modularity). On the other hand, more than one ANN control system can be used to solve a global task divided into sub-tasks (external modularity).

The proposal of our research is to test an external modules architecture that offers a solution for the locking search space problem. Our architecture was based in bio-inspired elements such as the neuro-modulation carried out by acetylcholine, chemical synapses, and Renshaw cells. In order to validate the proposed architecture, we developed two experiments for evolutionary robots where we compared three different architectures.

This paper is structured as follows. Section 1 introduce the problem and presents the purpose of our research. In section 2, some work related to the ANNs modular architecture is presented. Section 3 presents all bio-inspired concepts where our architecture is based. The details around our model are explained in section 4. In section 5, the methodology and experiments are described in section 5. Section 6 contains the results of the experimentation. Next, the discussion of our findings is presented in section 7. Finally, Section 8 shows the conclusion of our research.

2 Related Work.

Initial research in modularity and ANNs optimizes morphology and modules weights in the same evolutionary process (Gruau [22]). In this case, the architecture is based on parallelism, and results show that, as in nature, the evolutionary process tend to generate regularity in the topology. Later research show that less complex structures have better performance rather than complex structures (NourAshrafoddin, Vahdat & Ebadzadeh [51]).

In terms of internal modularity, one of the initial models uses inhibitory and excitatory connections in order to produce a competition between regions of one single ANN. At the beginning of the evolutionary processes complex structures appear, but tend to disappear at the end (Cho [11]).

One of the most important research in internal modularity is proposed by Nolfi [49]. Five architectures for control systems of evolutionary robots are tested and compared in this work. The setup is based in an experiment for cleaning a squared arena using the Khepera robots. Each robot has to collect cylinders and deposit them outside the arena. In this work five different ANNs topologies are tested (see figure 1): two layered ANN; three layered ANN; two layered ANN with retro-connections; external modules ANNs (two ANNs in parallel); and the Emergent Modular Architecture (EMA), which is an internal modules ANN.

For instance, the external modules ANNs is a divided control system where two nets compete for accessing the actuators every step of the simulation. In this case, no explicit mechanism mediates the competition between modules. On the other hand, the EMA is composed by two modules for each actuator; as a result, output-modules are groups of two neurons at the output layer. One neuron is used to codify motor velocity, and the other is the activation neuron. Therefore, two modules compete to access the associated actuator, and control is granted to the highest activation level module.

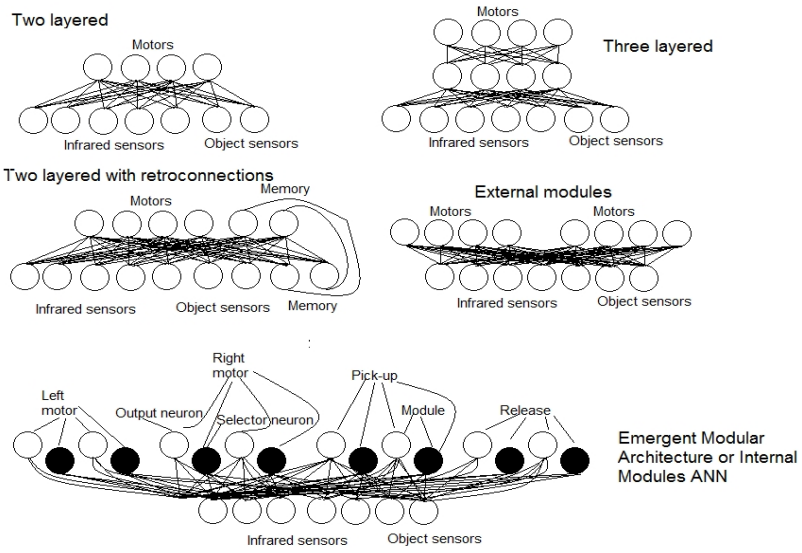


Fig. 1. Architectures tested by Nolfi [49]: A two layered ANN, a three-layered ANN, a two layered ANN with retro-connections, the external modules ANNs, and the internal modules ANN. The retro-connected ANN uses memory neurons to remember the last state. The external modules control is composed by two ANNs that compete to solve the task. The internal modules ANN is formed by two modules that compete for accessing the actuators.

Results show that the EMA has the best performance in terms of fitness and cylinders collected. The reason of these results is associated to the design of the ANN and the competition between modules. In addition, it has been showed that just one pair of modules is always active for selection. This constrains the variety of potential solutions to one group of neurons and does not reproduce what happens in the human brain where module monopolization does not occur. Module specialization is the consequence of the evolutionary pressure exercised by competition between modules. External module representation has average results, because two ANNs compete to access the resources and do not collaborate to achieve the general solution. In order to alleviate this situation, is necessary to include a mechanism that regulates competition and promotes cooperative process.

For the first three architectures (a two layered ANN, a three-layered ANN, and a two-layered ANNS with retro-connections), evolution does not optimize module selection and individuals are not capable of collecting and depositing garbage. This evidences that the cleaning task presents a blocking in search space where the ANN has to be optimized to collect and deposit objects.

Further research supports the claim of the EMA having a better performance compared to architecture based on the original EMA where a hidden layer and retro-connections are included (Ziemke, Carlsonn & Bodén [65]). The best competitor of the different implementations corresponds to the model with retro-connections and the reason is the capability of these connections to maintain information from previous steps (Ziemke [63]).

An important variation of the EMA is proposed using new evolutionary components (Calabretta *et al.* [7], which is named the Emergent Modular Architecture Based on Duplications (EMABD). Additionally, it has been used to explain evolutionary mechanisms for the evolution of modules. This architecture considers that modularity is included in the genetic information. Therefore, there are many available evolutionary methods for the optimization of module morphology. One method employs assembler Encoding (Praczyk [54]), which tries to concentrate efforts on evolving module morphology instead of neural weights. Their results show that this methodology produces both simple and complex modules (Praczyk [55]).

As for the external modularity, one of the first implementations used sequential components, which means that the output one ANN is the input for a next stage ANN (Cao, Ahmadu & Shridhar [10]). A later approach implemented a control system for a four-legged robot based in external and sequential modularity (Manoonpong, Pasemann & Fischer [38]). Thus, two independent ANN modules work as the input for a third module that controls most of the actuators. As a result, a robust system controls an avoiding robot that has to explore the environment ((Manoonpong, Paseman & Roth [39]). Furthermore, the model is also implemented for multi legged robots (Manoonpong, Paseman & Wörgötter [40]). Additionally, incremental neural evolution is used for an external modular system for controlling helicopters (Rice [56]).

3 Bio-inspiration in Evolutionary Robotics.

Clearly, from its conception, ER is a highly bio-inspired algorithm. Computational optimization processes tried to mimic evolution in nature. Therefore, solutions that are tested and do not have enough abilities to solve a task, tend to disappear and they are replaced with new solutions. Then, solutions that have better adaption to the environment, and the task to be solved, have the highest possibilities to remain in the evolutionary process (Darwinian Evolution).

The use of ANNs adds an extra path to the research of bio-inspired methods that are highly related to neurosciences. In this way, an evolved neuro-controller is a representation of an artificial nervous system (Gurkiewicz & Kornegreen [23]). Thereby, ER is considered a standard platform that facilitates proving and conducting experiments in areas such as neuroscience, and computation and biology (computational neurobiology), and their implementation is analogous to the evolution of the nervous systems (Floreano, Husbands & Nolfi [18]).

ER has the capability to generate computational models of neural components (Webb [61]). These models try to imitate the way nature solve problems that are optimized by billions of years through evolution (Mayer [41], Pasemann [52], Jeason & White [32], Fernando, Szathmary & Husbands [16]). For this reason, is very important to study and apply evolutionary mechanisms to ER (Husbands *et al.* [26]).

In terms of external modularity, the main problem consists in creating inter-modulator systems that provide an efficient solution in terms of which specialized module can solve a problem after is presented. In nature, chemical synapses and neurotransmitters are the solution to this problem.

Neurons have two different ways for transmitting information: electrical and chemical. Electrical synapses are replicated by all models of ANNs. Electrical synapses, in the brain, happens when two physically proximal neurons (pre-synaptic and post-synaptic) transmit an electrical current that generates a rapid voltage in form of a spike. On the other hand, chemical synapses happen when pre-synaptic neurons liberate a chemical substance (neurotransmitter) close to a post-synaptic neuron. There is not direct contact between axons, and there is a space between neurons named the synaptic cleft. Post-synaptic neuron captures molecules of neurotransmitter using receptors and then produces a depolarization. Chemical synapses do not produce an instantaneous effect on the post-synaptic neuron because chemical substances take time to travel between neurons. Neurotransmitters amplify their effects through an increasing number of receptors at the post-synaptic cells. In addition, a special mechanism recovers neurotransmitter molecules, which are not captured by the post-synaptic neurons. The aforementioned mechanism allows recovering the neuro-transmitter to convey further chemical processes in the post-synaptic neurons. Consequently, neurotransmitters regulate interactions between neurons and brain regions (Katz [33] & Spitzer [60]).

Different types of neurotransmitters are available in the brain: acetylcholine, dopamine, serotonin, noradrenalin, endorphin, and glutamate. Each of these has a particular effect under different brain regions and interact between them in particular cases. Many neurotransmitters are produced in the basal ganglia (neuronal groups that belong to primitive brain); and the transportation mechanisms from production zones to stock zones are named pathways. When a neurotransmitter arrives to pre-synaptic neurons, specialized vesicles store them for posterior release purposes.

In the nervous system, the aforementioned substances have four characteristics that distinguish them from other similar substances (Seiggelbaum & Hudspeth [58]):

- Neurons produced them.
- They are stored in the pre-synaptic neurons, and then they are liberated for post-synaptic purposes creating a specific effect.
- In most of the cases, their effect can be replicated administrating them, in reasonable concentrations, near the synaptic cleft (except for acetylcholine).
- There is a mechanism in the synaptic cleft for their recollection.

Under these conditions, is possible to create virtual neurotransmitters using ANNs models. Thus, virtual neurotransmitters can be used for communicating internal states, process status, or promoting specific outputs in the ANNs (Smith & Philippides [59]). However, the most important application is the modulation of complex control systems for evolutionary robots (Husbands *et al.* [27]). A neurotransmitter that deserves particular attention is the Acetylcholine (Ach), which modulates the action of motor neurons. Additionally, is considered a primitive neurotransmitter because is contained in all organisms around the world.

The composition of ACh is based in Choline and Acetylcoenzyme type A that is acquired from glycolysis consuming energy molecules. Catalization is made by choline acetyltransferase. Production centers are located in Renshaw neurons, Pedunculopontine nucleus, Meynert Basal Nucleus, and other basal ganglia nucleus in the cholinergic pathway. Acetylcholine molecules are captured by two types of receptors: Muscarinic and Nicotinic. The first type of receptors is associated to the inhibition of the parasympathetic system. The second type is associated to excitation of neurons in the muscular and skeletal apparatus, and involuntary movements of the heart. Nicotinic N1 receptors regulate synapses in voluntary movement muscles. When acetylcholine is recaptured, a substance named Acetylcholinesterase assimilates those molecules of neurotransmitter that are not used in chemical synapses.

ACh is present in all organisms, but is produced exclusively by neurons. After humans eat, butryrylcholinesterase decomposes acetylcholine molecules in order to disallow excitatory effects in the human body that include involuntary movements. For this reason, acetylcholine is not based in the food intake and cannot be ingested. Deficiencies of ACh are associated to Alzheimer and Parkinson. Nicotine is an agonist substance of ACh, which produces strong addiction in the human brain.

In renshaw neurons, ACh acts in an inhibitory way that promotes right movements and avoids excitation from motor neurons in the contrary muscles. When a motor neuron is excited, renshaw neurons, which are considered inter-neurons of the spinal cord (Aznavurian & Aguilar Rebolledo [2]), send inhibitory signals to lateral motor neurons and produce the desired movement. This mechanism is named lateral inhibition (Alvarez & Fyffe [1]). In other words, renshaw neurons allow muscle contraction when another one is extended (Kravos & Malesic [36]).

4 Proposed Architecture.

The Acetyl-Modulated Architecture (AChMA) is composed by two blocks (see figure 2). The first block corresponds to all modules or Artificial Neural Networks in which is decomposed or divided the proposed general problem. The second block is represented by an ANN that produces artificial levels of acetylcholine to other modules and is named network producer (similar to the basal ganglia).

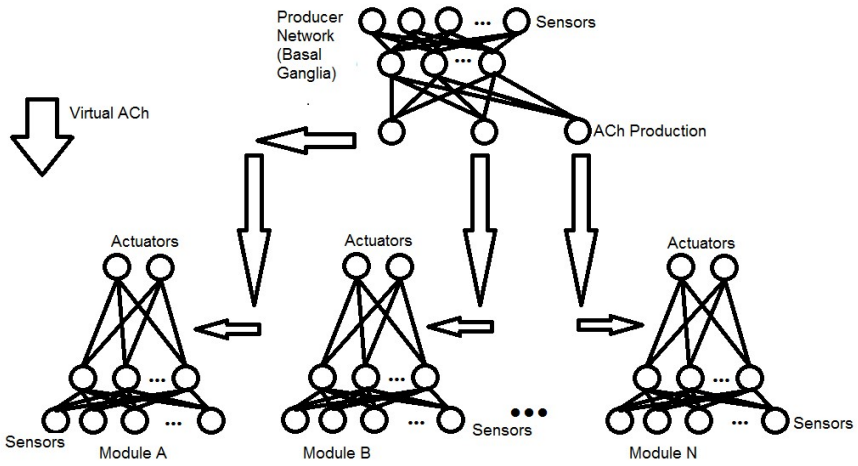


Fig. 2. Acetyl-Modulated Architecture. Ganglia groups of modules compete for accessing the motor system of an evolutionary robot. A producer network sends levels of virtual ACh to the inter-neurons of each module to produce synapses between neurons of the hidden layer and neurons of the output layer. Consequently, resource access and participation in the solution are regulated.

The network producer senses all variable of the environments where robots are situated. Additionally, this ANN receives the information of which module was binary activated in the previous step (internal state sensor). Hence, when a module was active in the previous step, the internal state sensor receives a binary value of one. When the module is not active, the virtual sensor has a binary value of zero.

In order to improve discrimination in the producer network, a hidden layer was included. The number of neurons in this layer is less than neurons in the input layer, and more than the number at the output layer. Next, the output layer of the net is a virtual value of acetylcholine. This value is associated to each producer neuron, which is an activation neuron. Moreover, its function is to inhibit the Ach production of other modules by means of inhibitory connections. In this way, when a producer neuron reaches the maximum value of activation, the neuron sends an inhibitory signal to other producer neurons in an off-center off-surround manner. Therefore, only one of all producer networks send their artificial neuro-transmitter to their representative module. This mechanism is inspired by renshaw cells.

Virtual levels of ACh are sent to the interneurons (hidden layer) of each module. When a module receives the virtual substance, neurons in the hidden layer produce synapses (replicating a chemical synapsis) with the output neurons and produce changes on robot actuators. If inter-neurons do not receive virtual ACh, they cannot produce synapses between hidden and output layers.

On the other hand, modules networks are composed by sensor neurons at the input, hidden layer, and finally actuator neurons in the output layer. Each module is evolved independently and each one solves a part of the general problem (a divide and conquer strategy). In order to split the original problem into sub-problems, we used a division of the fitness function to reduce the influence of human design in the evolutionary process. After all modules are evolved, a final evolutionary process adjusts just the producer network leaving the rest of the modules intact. This evolutionary process uses all fitness function terms, which results in an alternative for reducing the influence of human designers during evolution (Frenken, Marengo & Valente [20], Frenken [19], and Weitzenfield [62]).

5 Methodology.

In order to validate the proposed architecture, we designed two experiments where the performance of the architecture was compared with two control systems: a Single Net Based Control (SNBC), which is one ANN that controls all robots operations, and the Emergent Modular Architecture (Nolfi [49]). The first experiment consists of a cleaning arena task where a single robot has to collect and deposit trash. The second experiment is a simple maze. Robots has to interchange the room where they are situated. The arena, right in the middle, has a door that is open before changing room.

In all cases, a generational Genetic Algorithm was used for weight optimization of the artificial neural networks employed as robot controllers. The parameters utilized in the evolutionary process were 100 generations of 20 individuals, a mutation rate of 2%, one random crossover point, tournament selection, 80% of substitution, 20% individuals from the previous generation, and finally elitism (the best individual of the previous generation). Evolution run in a virtual environment modeled in Webots (Michel [43]). This simulator includes 3D models of various commercial robots and is possible to program controllers using programming tools such as Matlab, Java, C, and C++ among others.

The e-puck robot (Mondada et al [46]) was selected as the virtual platform for our experimentation. This robot is an open hardware platform, built for educational purposes; thus, users can add new hardware components. In its basic version, the robot is equipped with several actuators such as infrared sensors, light sensors, a VGA camera (640x480 pixels), step motors, a ring of LEDs, one microphone and speaker, gyroscope, Bluetooth communication, and a serial port. All devices are controlled by a TTL component, the dsPic30F6014A, which can be programmed using C or assembler language.

5.1 Experiment 1: An Environmental Cleaning Task

In this experiment, a robot has to clean a square arena delimited by four walls. In the original experiment, a Khepera robot uses its gripper to collect cylinders that represent trash. Thus, the robot deposits the cylinder outside of arena. Taking into consideration the limitations and differences of the e-puck robot, we adapted the original experiment.

In our experiment cylinders were replaced by virtual garbage contained in a circular white target and walls were colored in blue (see figure 3). The e-puck robot was rewarded with one garbage unit for each step that it remained in the white zone. Therefore, one point was accumulated every step that the robot spent at the white zone until the maximum load capacity of 200 units was reached. Additionally, for each step that the robot stayed in the black zone, it was equivalent to depositing one garbage unit. Furthermore, the two target areas included, in the center, a cylinder that was used as an additional visual aid for the robot.

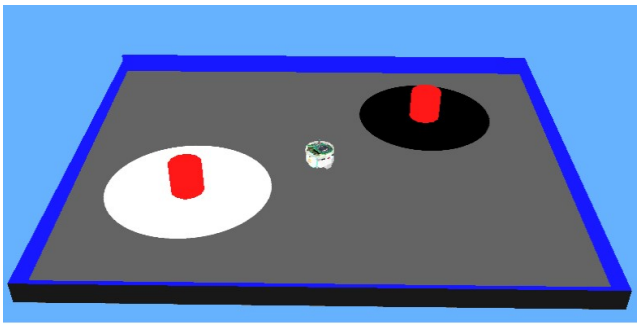


Fig. 3. Environmental setup for experiment 1. A robot is placed in a square arena delimited by four blue walls. There is a white circular target where robot collect garbage and a black circular target where the virtual garbage is deposited.

In relation to the sensory system, the robot uses eight infrared sensors to avoid obstacles. Additionally, three infrared sensors located at bottom of robot are used for locating target areas. When an obstacle are detected, IR sensors send a binary value of 1 (target detected); otherwise the IR sends an output value of 0 (undetected target). These kind of sensors were used for detecting color variations on the floor, which are white, gray, and black. When white color is detected, the first sensor has a binary value of 1 and the other two sensors remain inactive. In the case that the gray color is detected, the second sensor has an active status and the rest of sensors have an inactive value. Finally, when black color is detected the third sensor is active and the rest are inactive. Next, a garbage virtual sensor was created to acquire information about the status of the garbage load. This sensor is a coefficient of the actual load of garbage divided into the maximum amount of garbage (200 U). Additionally, one point was added to the fitness function for each step that robot remained at the black target until the minimum of zero units was reached.

The red and blue components of the VGA camera were used, and four sectors of 72 pixels were delimited. Thus, four binary virtual sensors were generated by each color component. When the color component was detected within a color segment, the virtual color sensor sent and active value; otherwise sensors outputted an inactive value.

Three different types of control were evolved for this experiment. The first control system was a Single Network Based Control (SNBC). It was represented by one ANN formed by 27 neurons, 20 neurons at the input layer, eight for the infrared sensors, three for the ground sensors, one for the virtual garbage sensor, four for the red VGA component, and four for the blue VGA

component. In this experiment, and the rest of the experiments, the green VGA component was not used to avoid influence from synaptic connections related to the green color. Particularly, because nothing is colored in green, in the 3D scene, and this extra connections may unbalance the evolution of the control system. As for the neural topology, we added five neurons at the hidden layer, and two neurons in the output layer for controlling motors that encoded the speed of the robot. The ANN had a feed-forward architecture and the sigmoid transfer function.

The second control was an example of internal modularity named the Emergent Modular Architecture and was composed in total by 38 neurons. A the input layer 20 neurons, eight infrared sensors, three ground sensors, one garbage sensor, four red color components, and four blue color components; next, ten at the hidden layer; and eight at the output layer. Four modules were built at the output layer, and each one was composed by two neurons. One output neuron was used to codify motor speed and the other was used as an activation neuron. A pair of modules competed to control the right wheel; and the other pair of modules controlled the left wheel. The maximum level of the activation neurons was used to select, at every simulation step, which pair gain motor control. The implemented ANN had a feed forward architecture and the sigmoid transfer function.

The third control system was a representation of external modularity (ACh modulated) composed by three ANNs: two modules and one producer. Modules were composed in total by 32 neurons. The input layer was formed by 20 neurons: eight infrared sensors, three ground sensors, one garbage sensor, four VGA red components, and four VGA blue components; ten neurons at the hidden layer; and two neurons at the output layer. Next, the producer network was composed by 36 neurons in total. The input layer was formed by 22 neurons as follows: eight infrared sensors, three ground sensors, one garbage sensor, four VGA red components, four VGA blue components; and as a form of short memory, one neuron for the previous binary state of module A, and an extra one for the previous binary state of neuron B. Next, ten neurons at the hidden, and four at the output layer. The first module (module A) was optimized for controlling the collection task, and the second module (module B) was optimized for the depositing task. The fitness function is composed by two terms that correspond to modular division, and respectively reward the individual with one point for each unit of garbage collected and deposited.

Each control system was represented as an experimental group. Hence, each group had 18 replications of the evolutionary process where the best individual of each replication was used for experimentation in a post-evolution test. In order to find statistical differences between the experimental groups, we used a one-way ANOVA for ranks (Kruskal Wallis) and a post-hoc Student Newman-Keuls test. As the dependent variable, we used the amount of trash collected and deposited for a post-evolution test of 6 trails of 1000 steps. Next, the best chromosome of the last generation of the current evolutionary process was used. As for the independent variable, the type of control system was used.

5.2 Experiment 2: Interchange Room Task.

The interchange room task is an experiment developed for robots to acquire the ability to solve mazes. Also, can be used to test the architecture with more modules. The arena was built with two square rooms delimited by a red wall with a door in the center (see figure 4). There was a rectangular black target area in the first room. This target was used as a switch that allowed opening the door between rooms. The door opened, after the robot stayed at least one-step at the black rectangle. A surrounding white area was set below the red wall to add extra information about the proximity of the border wall to help and as a visual aid for finding the aisle between rooms.

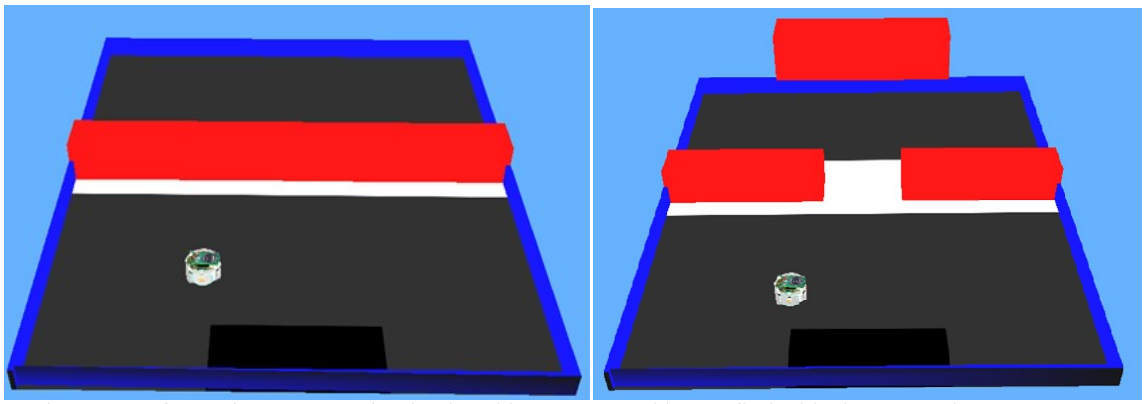


Fig. 4. Environment of experiment 2. A robot is placed in a room, and has to find a black rectangular target area to open the door. Then, the robot has to find aisle between rooms to move to the next room.

A robot with an initial random position had to find the black target, open the door and change to other room. The fitness function rewarded with 3.0 points when the robot opened the door, 11.0 points when robot found the aisle and 29.0 points when robot changed room.

The sensory system was integrated by eight infrared sensors, three ground sensors, four VGA red color component, and four VGA blue color component sensors that represented four sectors of 72 pixels. Additionally, three virtual sensors were created representing the actual state of the task (door open, aisle find, and change room). These sensors operated in a binary way and had an active value when the robot completed a partial task. Next, the motors were used as actuators. All ANNs used as control system had a feed-forward architecture and the sigmoid transfer function.

Three ANNs control systems were tested as in the experiment 1. A single control network was configured with one ANN composed by 35 neurons: 22 at the input layer (eight infrared sensors, three ground sensors, three task status sensors, four VGA red color component, and four VGA blue component sensors); 11 neurons at the hidden layer; and 2 at the output layer that controlled the motor speeds.

The Emergent Modular Architecture (internal modularity representation) was integrated by one ANN with 41 neurons: 22 at the input layer (eight infrared sensors, three ground sensors, three task status sensors, four VGA red color component, and four VGA blue color component sensors); 11 neurons at the hidden layer; and eight at the output layer. Four output modules were built, two competing modules for controlling the right wheel and two competing modules for the left wheel.
The Acetyl-modulated control system (external modularity representation) was composed by four ANNs: three modules and one producer. Each module was composed by 35 neurons: 22 at the input layer (eight infrared sensors, three ground sensors, three task status sensors, four VGA red color component, and four VGA color blue component sensors); 11 neurons at the hidden layer; and two neurons at output layer. The producer network was composed by 42 neurons: 22 neurons at the input layer, eight infrared sensors, three ground sensors, three task status sensors, four VGA red color component, four VGA blue color component, one neuron for storing the previous binary state of module A, one for storing the previous state of neuron B, and one for the previous state of neuron C. Next, 11 neurons at the hidden layer, and six neurons at the output layer.

The third control system is an example of external modularity (ACh modulated) composed by three ANNs: two modules and one producer. Modules were composed by 32 neurons: 20 at the input layer (eight infrared sensors, three ground sensors, one garbage sensor, four VGA red color component, four VGA blue color component); ten neurons at the hidden layer; and two neurons at the output layer. The producer network was composed by 36 neurons: 22 neurons at the input layer (eight infrared sensors, three ground sensors, one garbage sensor, four VGA red color component, four VGA blue color component, one neuron for storing the previous binary state of module A, and one for the previous state of neuron B). Next, ten neurons at the hidden layer, and four neurons at the output layer. The first module (module A) was optimized for controlling the collection task, whereas the second module (module B) was optimized for the depositing task. The fitness function was designated focusing on modular division by means of its fitness function components. Thus, module A was optimized for finding and opening the door, module B was optimized for finding the aisle, and module C was optimized for changing to the next room.

Three experimental groups represent the proposed control architectures. All experimental groups were integrated by 18 replications of the evolutionary process. The best individual of the last generation for each replication was tested. A one-way

ANOVA for ranks (Kruskal Wallis) was used to find statistical differences, complemented with a post-hoc Student Newman Keuls test. A post-evolution test of six trials of 1000 steps measured fitness levels of the individuals in each experimental group.

6 Results

6.1 Environmental Cleaning Task

The one-way ANOVA for Ranks analysis (see table 1) showed significance differences between all groups ($H_{(13.622)}$, p=0.001, 2df). Next, the post-hoc test revealed that all groups (ACh = 307.66 ± 33.812, Control = 133.33 ± 21.737, EMA = 204.00 ± 37.660) were different from each other ($p<0.05$).

Table 1. One-way ANOVA for ranks (Kruskal Wallis) of experiment 1.

	AChMA	SNBC	EMA
Median	307.660 *	133.330 *	204.00*
Mean + standard error	318.617± 33.812	133.579± 21.737	250.553 ± 37.660
Standard Deviation	143.453	92.223	159.777
H=13.622, p=0.001, n=18 *SNK comparison between groups			

The maximum expected fitness was 700 unities of collected (350 unities) and deposited garbage (350 unities). The ACh experimental group had the best fitness of the post evolution test (see figure 5). Individuals that belonged to this group had enough abilities for solving the task. In contrast, the fitness of individuals in the EMA group (able to solve the task) was lower with respect to the ACh group but greater than the control group. The latter corresponds to the findings in the work of Nolfi [49] where the control group obtained a lower fitness compared with the EMA. The fitness levels also showed that during the evolutionary process, the control group could not complete the garbage collection task. These individuals tended to learn how to collect garbage, but not to deposit it. The collection sub-task locked the posterior optimization of the depositing sub-task.

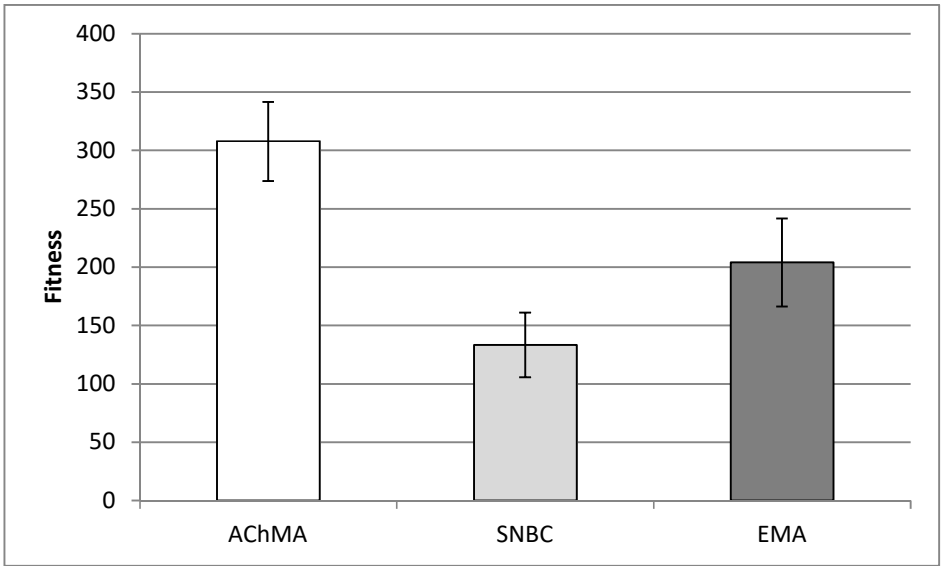


Fig. 5. The Median values and standard error in experiment 1. AChMA had the best performance, followed by the EMA and SNBC. The level of the SNBC group shows that robots are more efficient for collecting and depositing garbage.

6.2 Interchange Room Task

The one-way ANOVA on ranks (see table 2) exhibited significant between groups differences ($H_{(13.890)}$, $p<=0.001$, 2df) for the interchange room task. This was verified with the post-hoc test where was showed that the ACh group (14 ± 4.693) was statistically different ($p<0.05$) than the EMA (3 ± 2.238) and control (0 ± 0.79) groups.

Table 2. The one-way ANOVA for ranks (Kruskal Wallis) for experiment 2.

	AChMA	SNBC	EMA
Median	14*	0	3
Mean + standard error	21.994 ± 4.693	1.944 ± 0.79	5.056 ± 2.238
Standard deviation	19.910	3.351	9.496
H=13.890, p≤0.001, n=18 *SNK comparison between groups			

The highest possible score for this task was 43. The ACh experimental group had the best score, producing individuals able to open the door, find the corridor, and change room (see figure 6). Individuals in the EMA group were able to open the door and in some cases move to the next room. Finally, individuals in the control group were not able to open the door, and consequently were not able to change to the next room. In other words, the evolutionary process locked the search space for the developing of later sub-tasks.

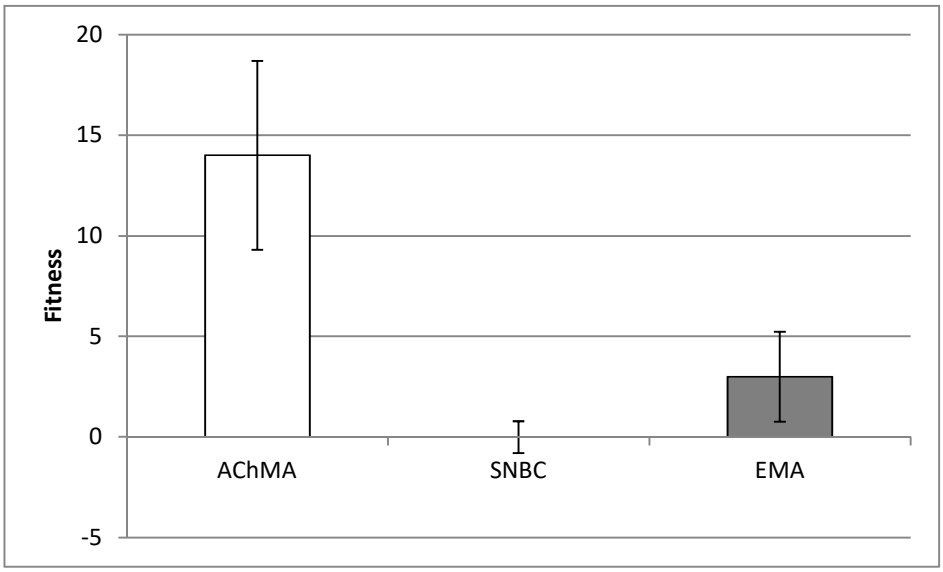


Figure 6. The Median and standard error values for experiment 2. In this experiment, the number of modules of the AChMA group increased. The best performance correspond to the AChMA group.

7 Discussion.

7.1 The environmental Cleaning Task

The first experiment had the purpose to test and compare the optimization of a task with a lock in the search space when using artificial evolution such as the environmental cleaning task. Therefore, we propose the development of the AChMA and two more control systems.

Our results confirm that cleaning environmental task is a valid platform to study the problem of locked search spaces (Ziemke *et al.* [64]). A Single Network Based Control has the worst performance with a median of 133 points. The SNBC had the best performance with robots that can collect an average number of garbage and deposit it in a lesser proportion. However, in the

worst scenario the robot neither collects nor deposits any garbage unit. The argument for such a poor performance is related to the complexity of the search space where the optimal weights for collection may be located in a different space in comparison to those of the depositing task (Jacobs *et al.* [30]).

On the other hand, the EMA produces individuals that are specialized for a single task with a median of 200 units. In this case, the evolutionary process obtains better individuals than in the SNBC. These results are consistent with reports from previous research (Nolfi [49]). The reason for this improvement is related to internal modularity.

Because each ANN is divided into internal modules, the discrimination level for each neuron at output layer is relaxed. As a result, some competitive modules in the EMA monopolize decisions participating in all steps of the solution tasks. However, this characteristic leads to a locking condition of the search space that reduces the variety of the solutions.

The AChMA is a highly bio-inspired model and its most important component is the ANN producer module. This module allows gradually controlling other modules. In other words, the virtual acetylcholine production avoids the monopolization, of a particular resource, by a single module. Virtual ACh mimics the behavior of ACh in nature; thus, its main effect is to control actuators similar to the movement of volunteer muscles. To facilitate a good performance of the external modularity control systems is necessary, but not sufficient, to have a good inter-modular mechanism optimized by evolution. As shown in experiment 1, the evolutionary process of the AChMA model is decomposed in two parts: the first one where modules are optimized independently and the second one where the ANN producer is optimized. The module search space is affected by the environmental configuration and sensory motor system (Bongard [5]), and in general by the dimensionality and complexity of the search space (Huizinga, Mouret & Clune [25]). The simpler the modules, the less convoluted the search space is.

Hence, the AChMA models the external modularity function. It is common in the human brain that different regions or zones cooperate in the solution of a general problem. In our work, this characteristic facilitates searching subspaces of the overall problem. As a result, we are adding robustness to the space search using artificial evolution. Particularly, we use this characteristic as a way for producing better solutions and complex control systems through evolution pressure (Corucci [13]). These modules, in ER, have the possibility to produce, organize, and represent thought, reasoning, and ultimately memory elements (Scheper, Tijmons, De Visser & De Croon [57]).

Next, the lateral inhibition used in the neural net, which is similar to the effect of Renshaw cells, is another important component of our architecture. This kind of inhibition guarantees that just one of the modules takes part at every step of the evolution. The latter, is combined with the model of chemical synapses that inhibits, in the absence of virtual Ach, synapses from neurons of the hidden layer into neurons of the output layer.

The virtual substance in the model acts as a virtual neurotransmitter. Next, this substance is produced in specialized neurons and transported to pre-synaptic neurons (inter neurons at the hidden layer). At certain levels of the neuro transmitter, the pre-synaptic neurons affect the post-synaptic neurons. At the end of each step, all levels of neurotransmitter are eliminated. In nature, ACh cannot be induced to an organism because is related to muscles and can cause noxious effects. The modeled Virtual ACh is exempt of these effects.

Our results confirmed that all the properties and characteristics of the bio-inspired AChM are combined to produce a solution that unlocks the search space. The performance of this architecture shows the best solution for this experiment. Robots are ready to collect and deposit garbage units during all the evolutionary process. The use of modules facilitates the division of the general search space into simple search spaces. The two independent evolutionary processes optimize the weights of the two ANNs that execute the collection and depositing sub-tasks. The optimization of the network producer becomes a decision problem where it acts as a switch for the modules participating in the solution of the general problem. In order to improve the performance, the producer has to choose the right action from the behavioral repertoire (collection, depositing, or no action).

7.2 The Interchange Room Task

With the aim to know if the AChMA is expandable to more than three modules, the experiment 2 was developed. The purpose of this experiment is to analyze the modular capability of reducing the effects of locking the search space when the number of modules increases.

The interchange room task was discomposed into three modules related to three different sub-tasks: Open the door, look for the corridor, and swap rooms. Although, we use evolution for the optimization, actions have to be executed in a specific sequence

that adds an extra parameter for searching the space problem. The sequence is as follows, the robot has to find the black rectangular target area for opening the door, then to explore the initial room until it finds the white target area and locates the door, finally to cross the door and move to the next room. This sequence produces a sequential interference that in some cases causes the evolutionary algorithm to fail.

Our results show that the best performance is associated to the AChM, followed by the EMA and then the SNBC. In the latter, the evolutionary process produced individuals that were able to solve, at least partially, the global task.

This result is significant because the interchanging room task presents a locking of the search space because the added complexity and dimensionality of the ANN. An extra complexity is added because of the sequential nature of the task. Thus, individuals explore the arena, but cannot cross the corridor because the door is closed. As a result, an optimal point of the search space is reached when individuals are able to open the door, though this does not imply that they are near the rest of the optimal points for completing the general task.

The interchange room task is a good experimental platform to study modular architectures and the locking search space problem. A single ANN did not produce adequate results; hence, to improve the results the control system is divided and used as a modular architecture. In the case of a modular control system, is important to have the proper module selection for solving the general task.

The EMA produced better fitness results than the SNBC, but the evolutionary processes did not produced fitted individuals for solving the general task. Evolution spans individuals that are able to open the door but not to complete the rest of global task. The latter, is the effect of module monopolization that causes one particular module to take over the motor control most of the time. In the case of the cleaning environment task, the internal modules had more than one choice. Although in the interchange room task there are three selection options, sequential interference complicates the evolution of selection. Moreover, the EMA is affected by sequential interference. A potential solution for this problem is the use of incremental evolution where internal modules are evolved at different stages to facilitate robots learning the right sequential order. Therefore, at later evolution stages, new fitness terms are added (Faíña, Jacobsen & Risi [15]).

The AChMA results showed that most of the individuals are able to solve two thirds of the task. Additionally, there are individuals that are able to solve the whole task. This improvement is due to the external modules architecture, particularly the mentioned AChMA, which is able to avoid both the locking condition and sequential interference in the search space. Independent processes isolate chunks of the sequence that are evolved to solve in a partial way the general task. In experiment 2, individuals in the first module are evaluated for their abilities to find the black target area and open the door. The initial conditions for the second module start with the door open, thus evolution focuses in evolving the task of locating the door. As for the third module, individuals focus on moving across the nearby corridor. Then, the producer network is optimized for module selection in both the SNBC and EMA systems.

Finally, the AChMA demonstrated that is able to overcome the locking search space problem, and avoid sequential interference. Furthermore, this expandable architecture allows the addition of extra modules.

8 Conclusions

The Acetyl-Modulated Architecture (AChMA) offers a good solution for the search space-locking problem. This solution uses a divide and conquer strategy, and modular division derived from equation terms of the global fitness function. The AChMA roughly replicates living organisms in the sense that regions, organs, apparatus, and systems can be considered as modules. Control on this architecture facilitates searching the general search focusing on smaller sub-spaces guided through the included elements in the fitness function. In other words, the AChMA represents a re-dimension of the search space. Therefore, each module represents a part of the general problem. Furthermore, an important component is the producer network that regulates the interaction between modules in order to solve the main task. A virtual neurotransmitter is included to regulate module selection.

As for the Environmental cleaning task, our experiments presented a comparison between the SNBC, EMA, AChMA architectures. This task presented a locking in the search space produced by the collection and garbage deposit tasks. The AChMA architecture overcame the locking condition using modular components and a robust selection system.

Finally, the interchange room experiments showed that the AChMA preserved its functionality even though the number of modules increased. A decompositional task methodology facilitated the optimization of independent modules, and at a later stage, the producer network was evolved for producing the right module interaction.

References

1. Alvarez, F.J., & Fyffe, R.E. (2007). The continuing case for the Renshaw cell. The Journal of physiology, 584(1), 31-45.
2. Aznavurian, A., & Aguilar-Rebolledo, F. (2006). Espasticidad ¿Qué es y qué no es? Nuevos Horizontes en la Restauración Neurológica. Plasticidad & Restauración Neurológica, 5(2).
3. Ballard, D.H. (1986). Cortical Connections and Parallel Processing. Structure and Function. The Behavioral and Brain Science, 9(1). 67-90.
4. Bolker, J.A. (2000). Modularity in Development and Why It Matters to Evo-Devo. American Zoologist, 40(5), 770-776.
5. Bongard, J. (2015). Using robots to investigate the evolution of adaptive behavior. Current Opinion in Behavioral Sciences, 6, 168-173.
6. Calabretta, R., Di Fernando, A., Parisi, D., & Keil, F.C. (2008). How to learn multiple tasks. Biological Theory, 3(1), 30-41.
7. Calabretta, R., Nolfi, S., Parisi, D., & Wagner, G.P. (1998). Emergence of Functional Modularity in Robots. From Animals to Animats, 5, 497-504.
8. Calabretta, R., Nolfi, S., Parisi, D., & Wagner, G.P. (1998). A case of the evolution of modularity: towards a bridge between evolutionary biology, artificial life, neuro- and cognitive science. Proceedings of Sixth International Conference on Artificial Life, (6), 275-284.
9. Calandra, R., Seyfarth, A., Peters, J., & Deisenroth, M.P. (2016). Bayesian optimization for learning gaits under uncertainty. Annals of Mathematics and Artificial Intelligence, 76(1-2), 5-23.
10. Cao, J., Ahmadi, M., & Shridhar, M. (1997). A Hierarchical Neural Network Architecture for Handwritten Numeral Recognition. Pattern Recognition, 30(2), 289-294.
11. Cho, S. (1997). Combining Modular Neural Networks Developed by Evolutionary Algorithm. (1997). In IEEE International Conference on Evolutionary Computation 1997, 647-650.
12. Cho, S.B., & Shimohara, K. (1997). Emergence of Structure and Function in Evolutionary Modular Neural Networks. In Proceedings of Fourth European Conference on Artificial Life, 197-204.
13. Corucci, F. (2017). Evolutionary Developmental Soft Robotics: towards adaptive and intelligence soft machines following nature's approach to design. In Soft Robotics: Trends, Applications and Challenges, 111-116. Springer International Publishing.
14. Di Fernando A., Calabretta, R., & Parisi, D. (2001). Evolving Modular Architectures for Neural Networks. Proceedings of the Sixth Neural Computation and Psychology Workshop: Evolution Learning and Development, 253-262. Springer Verlag.
15. Faíña, A., Jacobsen, L.T., & Risi, S. (2017). Automating the incremental Evolution of Controllers for Physical Robots. Artificial Life.
16. Fernando, C.T., Szathmary, E., & Husbands, P. (2012). Selectionist and evolutionary approaches to brain function: a critical appraisal. Frontiers in computational neuroscience, 6, 24.
17. Floreano, D., Dürr, P., & Mattiussi, C. (2008). Neuroevolution: From architectures to learning. Evolutionary Intelligence, 1(1), 47-62.
18. Floreano, D., Husbands, P., & Nolfi, S. (2008). Evolutionary Robotics. In Springer handbook of robotics, 1432-1451. Springer Berlin Heidelberg.
19. Frenken, K. (2006). A fitness landscape approach to technological complexity, modularity, and vertical disintegration. Structural Change and Economic Dynamics, 17(3), 288-305.
20. Frenken, K., Marengo, L., & Valente, M. (1999). Interdependencies near-decomposibility and adaption. Computational techniques for modeling learning in economics, 209-244.
21. Gecshwind, N., & Galaburda, A.M. (1985). Cerebral Lateralization: Biological Mechanism, Association, and Pathology: I.A. hypothesis and a program for search. Archives of neurology, 42(5), 428-459.
22. Gruau, F. (1994). Automatic Definition of Modular Neural Networks. Adaptive Behavior, 3(2), 151-183.
23. Gurkiewicz, M., & Kornegreen, A. (2007). A numerical approach to ion channel modelling using whole-cell voltage-clamp recordings and genetic algorithm. PLoS Comput Biol, 3(8), e169.
24. Hart, W.E., Kammeyer, T.E., & Belew, R.K. (1994). The role of development in genetic algorithms. Doundations of genetic algorithms, 3, 315-332.
25. Huizinga, J., Mouret, J.B., & Clune, J. (2016). Does Aligning Phenotypic and Genotypic Modularity Improve the Evolution of Neural Networks? In Proceedings of the 2016 on Genetic and Evolutionary Computation Conference, 125-132. ACM.
26. Husbands, P., Moioli, R., Shim, Y., Philippides, A., Vargas, P., & O'shea, M. (2014). Evolutionary robotics and neuroscience. The horizons of evolutionary robotics, 17-63.
27. Husbands, P., Smith, T., O'shea, M., Jakobi, N., Anderson J., & Philippides A. (1998). Brains, Gases and Robots. In Proceeding ICANN, (98).
28. Inamura, T., Inaba, M., e Inoue, H. (2000) User adaption of human-robot interaction model based on Bayesian network and introspection of interaction experience. In Proceedings of the 2000 IEE/JRS International Conference on Intelligent Robots and Systems, (3), 2139-2144.
29. Jacobs, R. & Jordan, M. (1992). Computational Consequences of a bias toward short connections. Journal of Cognitive Neuroscience, 4(4), 323-335.
30. Jacobs, R., Jordan, M., & Barto, A. (1991). Task decomposition through competition in a modular connectionist architecture: The what and where vision task. Cognitive Science, 15(2), 219-250.

31. Jamil, M.F.A., Jalani, J., & Ahmad, A. (2016). A new approach of active compliance control via fuzzy logic control for multifingered robot hand. In First International Workshop on Pattern Recognition, (10011), id 1001111 9.
32. Jeason, F., & White, A. (2012). Evolving Axonal Delay Neural Networks for Robot Control. In Proceedings of the 14th annual conference on Genetic and Evolutionary Computation, 121-128. ACM.
33. Katz, B. (1969). The release of neural transmitter substances. Liverpool University Press.
34. Khare, V., Yao, X., Sendhoff, B., Jin, Y., & Wersing, H. (2005). Co-evolutionary Modular Neural Network for Automatic Problem Decomposition. In Proceedings of the 2005 IEEE congress on Evolutionary Computation, (3), 2691-2698.
35. Kodjabachian, J., & Meyer, J. (1995). Evolution and Development of Control Architectures in Animats. Robotics and Autonomous Systems, 16(2-4), 161-182.
36. Kravos, M., & Malesic, I. (2017). The Role of Glutamate Dehydrogenase Activity in Development of Neurodegenerative Disorders. World Journal of Neuroscience, 7, 181-192.
37. Loula, A., Gudwin, R., & Queiroz, J. (2010). On the emergence of indexical and symbolic interpretation in artificial creatures, or What Is this I Hear? In ALIFE 2010, 862-868.
38. Manoonpong, P., Pasemann, F., & Fischer, J. (2005). Modular Neural Control for a Reactive Behavior of Walking Machines. IEEE International Symposium on Computational Intelligence in Robotics and Automation 2005, 403-408.
39. Manoonpong, P., Paseman, F., & Roth, H. Modular Reactive Neurocontrol for Biologically-Inspired Walking Machines. The International Journal of Robotics Research, 26(3), 301-331. 2007.
40. Manoonpong, P., Paseman, F. & Wörgötter, F. (2008). Sensor-driven neural control for omnidirectional locomotion and versatile reactive behaviors of walking machines. Robotics and Autonomous Systems, 56(3), 265-288.
41. Mayer, H. (2011). Evolution of Robotic Neurocontrollers with Intrinsic Noise and their Behavior in Noisy Environments. In the 2011 International Joint Conference on Neural Networks (IJCNN), 1975-1980. IEEE.
42. Mendel, J. Fuzzy Logic Systems for Engineering: a tutorial. (1995). Proceedings of the IEEE Transactions on Fuzzy Systems, 83(3), 345-377.
43. Michel, O. (1998). Webots: Symbiosis between virtual and real mobile robots. In International Conference on Virtual Works, 254-263. Springer, Berlin, Heidelberg.
44. Mitri, S., Floreano, D., & Keller, L. (2009). The evolution of information suppression in communicating robots with conflict of interests. Proceedings of the National Academy of Sciences, 106(37), 15786-15790.
45. Mitri, S., Floreano, D., & Keller, L. (2010). Relatedness influences signal reliability in evolving robots. Proceedings of The Royal Society of London B: Biological Science, (278), 378-383.
46. Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.C., Floreano, D., & Martinoli, A. (2009). The e-puck robot, designed for education in engineering. In Proceedings of the 9th conference on autonomous robot systems and competitions, (1), 59-65. IPCB: Instituto Politectino de Castelo Branco.
47. Montes-Gonzalez, F., & Aldana-Franco, F. (2011). The evolution of signal communication for the e-puck Robot. Advances in Artificial Intelligence, 466.477.
48. Newman, M. (2006). Modularity and community structure in networks. Proceedings of the National Academy of Science, 103(23), 8577-8582.
49. Nolfi, S. (1997).Using emergent modularity to develop control systems for mobile robots. Adaptive Behavior, 5(3-4), 343-363.
50. Nolfi, S., Bongard, J., Husbands, P., & Floreano, D. (2016). Evolutionary Robotics. In Springer Handbook of Robotics, 2035-2068. Springer International Publishing. Springer-Verlag.
51. NourAshrafoddin, N., Vahdat, A., & Ebadzadeh, M. (2007). Automatic Design of Modular Neural Networks Using Genetic Programming. Artificial Neural Networks-ICANN 2007, 788-798.
52. Pasemann, F. (2013). Self-regulating Neurons in the Sensor motor Loop. In International Work-Conference on Artificial Neural Networks, 481.491. Springer Berlin Heidelberg.
53. Potter, M., Meeden, L., & Schultz, A. (2001). Heterogeneity in the Coevolved Behaviors of Mobile Robots: The Emergent of Specialists. In Proceedings of International joint conference on artificial intelligence, (17), 1337-1343. Lawrence Erlbaum Associates LTD.
54. Praczyk, T. (2008). Modularity networks in Assembler Encoding. Computational Methods in Science and Technology, 14(1), 27-38.
55. Praczyk, T. (2013). Modularity and Regularity in Neural Networks Produced with Assembler Code. Computational Methods in Science and Technology, 19(3), 145-155.
56. Rice, S. (2000). The evolution of developmental interactions: epitasis, canalization and integration. In Epitasis and The Evolutionary Process, 82-98.
57. Scheper, K. Y., Tijmons, S., De Visser, C. C., & De Croon, G.C. (2016). Behavior trees for evolutionary robotics. Artificial life.
58. Seiggelbaum, S.A., & Hudspeth, A.J. (2000). Principles of neural science. E.R. Kandel, J.H. Schwart, & T.M. Jessell (eds). New York: McGraw-Hill.
59. Smith, T. & Philippides, A. (2000). Nitric Oxide in Real and Artificial Neural Networks. BT Technology Journal, 18(4), 140-149.
60. Spitzer, N.C. (2017). Neurotransmitter Switching in the Developing and Adult Brain. Annual Review of Neuroscience, (0).
61. Webb, B. (2009). Animals versus animats: or why not model the real iguana? Adaptive Behavior, 17 (4), 269-286.
62. Weitzenfield, A. (2000). From Schemas to Neural Networks: A Multi-level Modeling Approach to Biologically-Inspired Autonomous Robotic Systems. Robotics and Autonomous Systems, 56(2), 177-197.
63. Ziemke, T. (2000). On 'Parts' and 'Wholes' of Adaptive Behavior: Functional Modularity and Diachronic Structure in Recurrent Neural Robot Controllers. From Animals to Animats, 6, 171-180.
64. Ziemke, T., Bergfeldt, N., Buason, G., Susi, T., & Svensson, H. (2004). Evolving Cognitive Scaffolding and Environment Adaption: A new research direction for Evolutionary Robotics. Connection Science, 16(4), 339-350.

65. Ziemke, T., Carlsonn, J., & Bodén, M. (1999). An Experimental Comparison of Weight Evolution in Neural Control Architectures for a 'Garbage-Collecting' Khepera Robot. In Proceedings of the First International Khepera Workshop, 31-40.