ALGORITMOS EVOLUTIVOS IMPLEMENTADOS EN ROBÓTICA

Daniel Fernández Valderrama[1], Samuel Domínguez Cid[1], Javier Antonio Guerra Coronado[1], Diego Francisco Larios Marín[1], Alejandro Gallardo Soto[1], Félix Biscarri Triviño[1]

[1] *Departamento de Tecnología Electrónica, Escuela Politécnica Superior, Universidad de Sevilla.*

E-mail de correspondencia: dfvalderrama@us.es

RESUMEN

En este trabajo se presenta la programación de un algoritmo evolutivo realizado en el lenguaje C++ para la resolución de distintos problemas en los que se busca escoger los mejores valores para un conjunto de variables definidas de un problema. La idea principal consiste en que estas variables estarán contenidas en un vector para cada individuo que tengamos, llamado cromosoma. El programa comenzará inicializando estos vectores de forma que comience una simulación y se mida cómo de bueno es ese individuo a partir de una métrica definida que variará en función del tipo de problema que tengamos. Estos mejores individuos serán seleccionados para la generación de la siguiente población, generados por cruce y mutación de los padres, los mejores de la anterior generación. Este proceso se repetirá hasta llegar a un resultado que consideremos óptimo o en su defecto hasta que el programa finalice después de simular un cierto número de generaciones. Como prueba de concepto, esta metodología se ha implementado en el aprendizaje de un robot cuadrúpedo para caminar en el entorno Webots.

INTRODUCCIÓN

La gran mayoría de ramas de la ciencia tiene que lidiar con problemas de optimización, existiendo diversa metodología para abordarlos. Sin embargo, suelen presentar algunas trabas cuando se refiere a aplicarse a problemas del mundo real. En estos casos, se suele recurrir a aproximaciones que por lo general simplifican el problema, como puede ser la aproximación de problemas no lineales a lineales, que, aunque funcione bien a nivel local, no es aplicable al espacio de todo el problema.

En los últimos años la inteligencia artificial ha intentado asimilar distintos procedimientos utilizados por la naturaleza para resolver este tipo de problemas, como son el aprendizaje, la evolución o la adaptación. A partir de estas ideas, surge la rama basada en la evolución biológica llamada “Algoritmos Genéticos” (Zhang *et al.*, 2021).

DESARROLLO DEL ALGORITMO

El programa comienza con la inicialización de los parámetros de nuestro conjunto de individuos, de nuestra población. Un individuo o cromosoma, consiste simplemente en el conjunto de variables que posee el problema. Estas variables estarán contenidas en un vector para una cómoda manipulación. Seguidamente, se evalúa cómo de buenos son en función de una *fitness* o función objetivo. Esta función objetivo establecerá el método de búsqueda de nuestro algoritmo. En este caso, como queremos que el prototipo aprenda a caminar, la función *fitness* será simplemente la distancia recorrida de cada individuo.

Una vez evaluada la *fitness*, escogemos los dos individuos que mejor "puntuación" hayan alcanzado, es decir, los que han recorrido mayor distancia. Y a partir de estos, creamos la siguiente población.

Las principales funciones evolutivas para la generación de nuevos individuos son la mutación y el cruce. Por lo tanto, son estas las implementadas en nuestro programa.

Mutación

Teniendo como parámetro de entrada el cromosoma a mutar, se escoge en primer lugar el último valor del cromosoma, el cual es la varianza del ruido Gaussiano para las modificaciones. A este valor, se le realiza una modificación entre unos valores predeterminados. Y con este nuevo, se realiza a cada uno de los restantes parámetros una alteración con forma de campana de Gauss a partir de la fórmula de Box Muller (Ross, 1999).

Cruce

Esta función recibe los dos individuos a cruzar, de tal forma, que genera un número aleatorio entre cero y el número de variables que posee el cromosoma. De esta forma, se establece el intercambio de información para generar los nuevos individuos.

Condiciones para finalizar

El algoritmo itera continuamente para generar nuevos individuos y evaluarlos, hasta que finaliza por dos posibles motivos. El primero, considerar una solución como suficientemente buena, es decir, llegar a un valor objetivo preestablecido para finalizar el programa. En segundo lugar, y pueden ser complementarios, el algoritmo finaliza tras cumplir un número de épocas preestablecido (Zhou *et al.*, 2021).

IMPLEMENTACIÓN

Para la simulación de nuestro caso particular, en el cuál queremos que un robot cuadrúpedo aprenda a caminar, hemos realizado un prototipo mediante una estructura de cápsulas en el entorno Webots (Cyberbotics Ltd., s.f.). Se compone de cuatro eslabones para conformar la columna y otras cuatro más alargadas en cada extremo que nos servirán de patas.

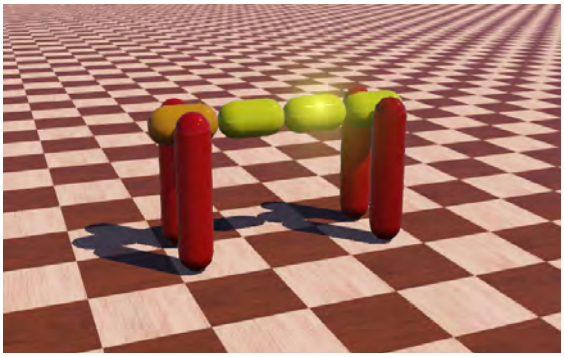

Figura 1. Prototipo.
Fuente: Imagen tomada del entorno Webots.

El movimiento de las patas estará asociado conjuntamente para las patas traseras y para las delanteras, tratándose de un movimiento senoidal; con amplitud fase y frecuencia, y posiciones máxima y mínima. Por lo cual, las variables quedan con cinco elementos para las patas delanteras y otros cinco para las traseras. Obteniéndose un cromosoma total de diez variables.

Como población se han establecido diez individuos, con 20 segundos de simulación para cada época, siendo el tiempo en el cual van a competir los individuos. Haciendo que el programa finalice tras 180 épocas, nos queda un entrenamiento total de una hora.

RESULTADOS

Para concluir, los resultados obtenidos en el estudio han resultado congruente, llegando a la convergencia en nuestro problema. En la gráfica se puede observar cómo evoluciona la *fitness* de los dos mejores individuos a medida que se va entrenando la simulación. Esta *fitness* por tanto pertenece siempre al mejor de los individuos, y permanece constante ya que si no hay un individuo mejor, mantendremos al anterior para asegurarnos que el predecesor es un buen candidato para la generación de nuevos individuos. Los individuos llegan a recorrer 9 metros en los 20 segundos establecidos en una época a lo largo del entrenamiento.

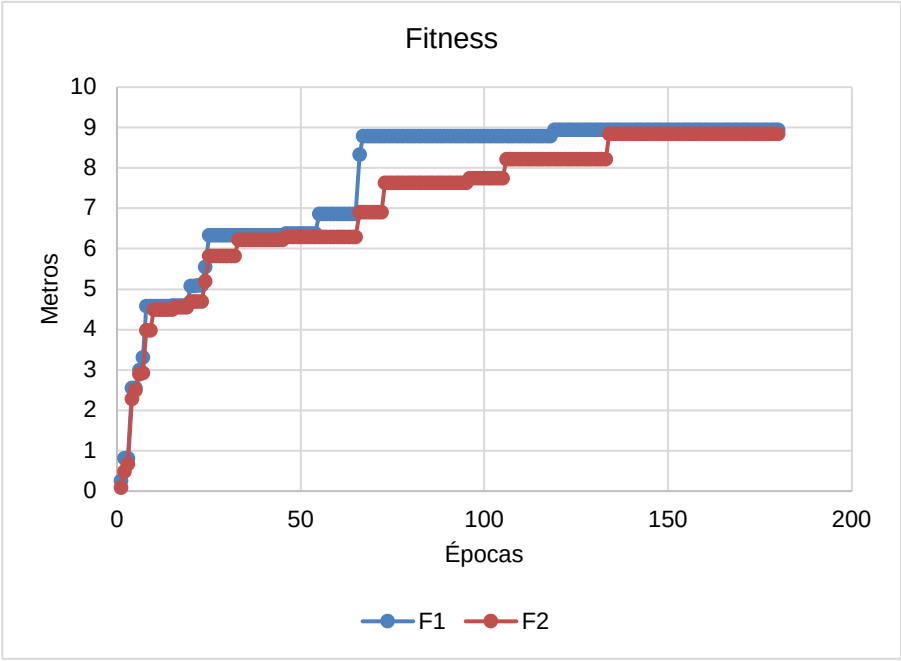

Figura 2. Evolución Fitness.
Fuente: elaboración propia a partir de la simulación.

REFERENCIAS BIBLIOGRÁFICAS

Cyberbotics Ltd. (s.f.). *Webots: robot simulator.* (último acceso: 26 de noviembre de 2020). https://cyberbotics.com/

Ross, S. M. (1999). *Simulación* (2ª ed.). Prentice Hall. pp. 72-75.

Zhang, H., Wang, Z., Chen, W., Heidari, A. A., Wang, M., Zhao, X., Liang, G., Chen, H., & Zhang, X. (2021). Ensemble mutation-driven salp swarm algorithm with restart mechanism: Framework and fundamental analysis. *Expert Systems with Applications, 165*, 113897. https://doi.org/10.1016/j.eswa.2020.113897

Zhou, Y., Zhang, W., Kang, J., Zhang, X., & Wang, X. (2021). A problem-specific non-dominated sorting genetic algorithm for supervised feature selection. *Information Sciences, 547*, 841-859. https://doi.org/10.1016/j.ins.2020.08.083