Performance Between Algorithm and Micro Genetic Algorithm to Solve the Robot Locomotion

F. Chávez-Estrada, J. Herrera-Lozada, J. Sandoval-Gutiérrez, and M. Cervantes-Valencia

***Abstract*—Gait robot refers to the locomotion platform coordinating the leg motions. Several studies are using genetic algorithms (GA) to solve the problem of gait learning. These studies coincide with the high computational cost and high energy consumption; this is mainly due to the handling of large numbers of individuals compromising the memory space in the system. To solve this problem, and with the intention of being implemented in hardware, it is proposed the use of micro genetic algorithms (μGA) that use populations of reduced size in this research twelve individuals. This article presents a comparison of the performance of both algorithms, μGA versus the standard GA, solving the problem of the generation of gait patterns for a quadruped robot. It demonstrated that the μGA has a rapid convergence to the solution and generate the robot locomotion. Implementation of the algorithms have performed in an embedded system with four cores to validate the results.**

***Index Terms*—Genetic Algorithm, micro Genetic Algorithm, Embedded System and fitness.**

I. Introducción

LOS algoritmos con inspiración biológica, también conocidos como algoritmos bio-inspirados, han adquirido gran importancia dentro del área de la inteligencia artificial debido a que han demostrado ser exitosos en la solución de ciertos problemas complejos de aprendizaje de máquina, reconocimiento de patrones, detección de fallas y optimización. Dentro de éstos, los algoritmos evolutivos (AE) son estrategias para la búsqueda de soluciones, generalmente con fines de optimización, inspirados en el mecanismo de la evolución biológica de acuerdo con De La Cruz, *et al*. [1]. La idea principal de este proceso de evolución artificial es mantener un conjunto de entidades (individuos) que representan posibles soluciones a un problema, las cuales se combinan, mutan y compiten entre sí, de esta forma las más aptas son capaces de prevalecer, evolucionando hacia mejores soluciones en cada generación. Los Algoritmos Genéticos (AG) son los algoritmos más representativos en la generalidad de los AE. Se trata de algoritmos poblacionales, es decir, generan un conjunto de posibles soluciones (individuos, codificados como cromosomas) del problema a resolver, y su aplicabilidad consiste en buscar las mejores dentro de esta población realizando cambios frecuentes a las características de los individuos. Los cambios son controlados aplicando sistemáticamente una serie de operadores genéticos: selección, cruza, mutación y remplazo, contribuyendo a mantener la diversidad de la población como indicó Golberg en [2]. Dentro de la población cada individuo es diferenciado de acuerdo con su valor de fitness (aptitud) que es obtenido usando algunas propuestas empíricas de acuerdo con el problema a resolver. La función fitness es la función objetivo del problema de optimización de acuerdo con Asteroth y Hagg [3] además de considerar el estudio de Burke *et al*. [4]. En la Figura 1 se muestra el funcionamiento de un AG simple.

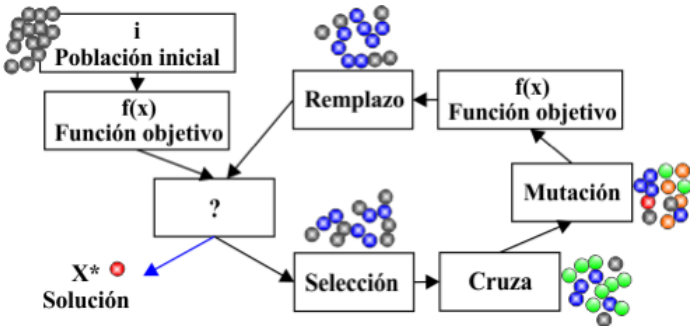


Fig. 1. Comportamiento de un AG simple.

II. Estado del Arte

Una de las principales características de funcionamiento en los AG es la dependencia de una población de individuos generada aleatoriamente. Existe [5] un costo computacional ligado al tamaño de la población, referido al uso de la memoria, al tiempo de procesamiento y al consumo de energía. La implementación de hardware de estos algoritmos de acuerdo a Ahmadi, *et al*. [5] es atractiva por los métodos y la utilización de los recursos internos de las arquitecturas embebidas para tratar con este tipo de procesos.

Los AG estándar (AGE) se han aplicado para resolver problemas de optimización de soluciones en donde los grados de libertad (GDL) de los sistemas se incrementan y es complicado aplicar los métodos tradicionales, tal es el caso de la locomoción de robots con patas estas investigaciones parten de los patrones de marcha aplicados a las extremidades determinan la manera en que éstos se desplazarán sobre una superficie. La sincronización de estos patrones permite el movimiento coordinado y estable del robot. Gumiel lo logra aplicando un AG en un robot Aibo [6], por su parte Suzuki en [7] realiza una optimización en la trayectoria de impulso de las patas aplicando un AG, sin embargo el robot utilizado se

F. A. Chávez-Estrada, Centro de Innovación y Desarrollo Tecnológico en Cómputo CIDETEC, Instituto Politécnico Nacional, México frchaveze1400@alumno.ipn.mx / Docente del Instituto Tecnológico de Tlalnepantla del TecNM, en el Depto. de Eléctrica y Electrónica, Estado de México fachavez@ittla.edu.mx.

J. C. Herrera-Lozada, Centro de Innovación y Desarrollo Tecnológico en Cómputo CIDETEC, Instituto Politécnico Nacional, México jlozada@ipn.mx.

J. Sandoval-Gutiérrez, Universidad Autónoma Metropolitana Unidad Lerma, Estado de México j.sandoval@correo.ler.uam.mx.

M. I. Cervantes-Valencia, Instituto Tecnológico de Tlalnepantla del TecNM, Estado de México macervantes@ittla.edu.mx.

ha dejado de producir desde el 2009, Parker y Torino en [8] aplican un AG cíclico para generar una secuencia de marcha.

El interés de mejorar los robots cuadrúpedos ha permitido desarrollos en la dinámica de las patas y dotarlos de propulsión como expone Elijah [9], Jiang y Cai en su investigación [10] simulan cuatro AE en la optimización de la marcha, por su parte Oliveira en [11] combina un generador central de patrones (GCP) bio-inspirados y AE multi-objetivos para optimizar la marcha al disminuir las vibraciones, incrementar la velocidad, la estabilidad y el comportamiento todo en una plataforma de simulación. Otros investigadores como Santos enfocan sus investigaciones [12] en la optimización de energía al mejorar las trayectorias del robot al caminar usando un AG.

Marín *et al*. en su investigación [13], muestran la variedad de aplicaciones al sintonizar un control PID usando con un AG y lo robusto que resultan las soluciones. Teng *et al*. trabajaron el problema de la marcha en robots con patas a partir de un GCP [14]. Varios de los trabajos citados sólo se limitan a la simulación por el alto costo computacional que requieren para encontrar las soluciones, por lo mismo, en esta investigación se ha enfocado en un método alterno al diseñar algoritmos con poblaciones reducidas en tamaño para coadyuvar en la construcción física, garantizando un desempeño similar a las versiones estándar. A estos AG con una población de individuos reducida en tamaño, que incorporan modificaciones para funcionar adecuadamente, se les nombra micro Algoritmo Genético (μAG) [2]. La hipótesis de nuestra investigación se centra en utilizar un μAG que resuelvan de forma autónoma la marcha de un robot cuadrúpedo al generar de manera evolutiva los patrones de marcha que establecen la posición de cada articulación de las patas de un robot cuadrúpedo, su principal característica son micro poblaciones, que cambian los operadores genéticos para encontrar la solución al problema de la marcha y que permite implementarlo en un sistema embebido (SE) con la opción de utilizar probabilidad estocástica para mantener diversidad al buscar las soluciones, completar la población con la clonación de los mejores individuos; lo anterior decrece la producción de los individuos, mejora la diversidad genética y crece la velocidad de convergencia de acuerdo a los resultados de trabajos previos de Herrera *et al*. [15] y en consecuencia se disminuye el tiempo de procesamiento.

El presente trabajo está organizado en las siguientes secciones: En la sección 2 se presenta la relación entre los algoritmos evolutivos y la marcha en robots, a través del estado del arte; en la sección 3 se describe la plataforma robótica utilizada para la experimentación, la sección 4 se utiliza para presentar el micro-Algoritmo Genético diseñado, la sección 5 detalla la implementación en el sistema embebido, en la sección 6 se describen los experimentos y pruebas realizadas, en la sección 7 se realiza la discusión de los resultados, y la sección 8 contiene las conclusiones del trabajo.

III. Robot Cuadrupedo

La locomoción de los robots con cuatro patas, imitan el modo de caminar de los animales como indican Chávez *et al*. [16] principalmente es de dos tipos: En el tipo reptil, el cuerpo es más próximo al terreno, su estabilidad es mayor, el consumo de energía es mayor al mamífero por el peso del robot, el torque en las articulaciones son mayores al mamífero y el desplazamiento es menor. En el tipo mamífero, el cuerpo es más alto, por tanto, el polígono de apoyo es más estrecho y el centro de gravedad es más alto, la estabilidad del robot es menor y el consumo de energía depende de la cantidad de eslabones habilitados al mismo tiempo.

El tipo de locomoción mamífero se ha utilizado en la investigación y se ha diseñado el robot cuadrúpedo prototipo es una estructura rectangular que aloja el SE y cuatro patas; las piezas se han fabricado con plástico ABS en una impresora 3D. Cada pata consta de tres articulaciones y un servomecanismo por articulación por tanto el robot tiene 12 GDL, el espesor de las piezas es de 0.004 m y para las piezas que se requiere mayor rigidez mecánica tienen un espesor de 0.006 m y en algunas piezas se reforzaron las esquinas con ángulos y soleras. En la Tabla I se muestran las dimensiones de las piezas principales indicando las que se reforzaron.

TABLA I
Dimensiones de Piezas del robot prototipo

Dimensiones	Largo (m)	Ancho (m)	Espesor (m)	Centros (m)
Cuadro o cuerpo del robot [a]	0.19	0.16	0.004	0.11
Eslabón1 (d)	0.06	0.045	0.003	0.035
Eslabón2 (a_2) [b]	0.095	0.0047	0.014	0.07
Eslabón3 (a_3)	0.06	0.03	0.004	0.045
Servomotor [c]	0.029	0.014	0.038	NA

[a] Soleras de 0.002 m en la orilla para lograr un espesor total de 0.006 m
[b] Ángulos de refuerzos en las esquinas
[c] Dimensiones del fabricante

En la Figura 2 se muestra el robot prototipo.

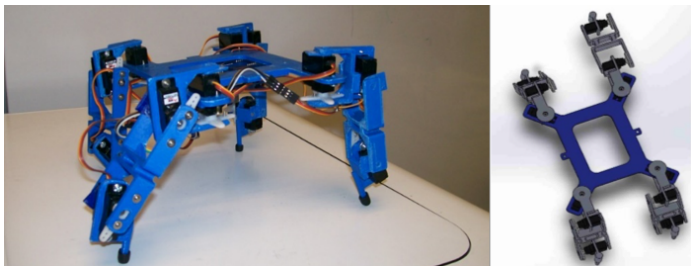


Fig. 2. Robot cuadrúpedo prototipo.

Se determinó los ángulos válidos o restricciones (movimientos no restringidos) y no válidos en cada articulación, variando los ángulos de las articulaciones, considerando el desplazamiento angular máximo del servomecanismo (180º) y las interferencias mecánicas entre cada eslabón, estas condiciones deben ser consideradas al formar el cromosoma del individuo determinan las restricciones del sistema. En la Tabla II se muestran los resultados de los ángulos válidos.

Se utilizó el método de Denavit –Hatenberg para representar la geometría espacial de los elementos de una cadena cinemática respecto a un sistema de referencia, en este caso los tres eslabones que forma una pata del robot, se establece un sistema de coordenadas (SC) a cada articulación que forman

TABLA II
ÁNGULOS VÁLIDOS Y NO VÁLIDOS DE CADA ARTICULACIÓN

Eslabón	Reptil Ángulos válidos	Mamífero Ángulos válidos
Eslabón 1 (cadera)	θ_1 = 0 – 70º	θ_1 = Posición fija
Eslabón 2 (pierna)	θ_2 = 0 – 30º	θ_2 = 0 – 20º
Eslabón 3 (tobillo)	θ_3=Posición fija	θ_3 = 0 - 40º

una cadena abierta en donde la punta de la pata es el final de la cadena, se considera las dimensiones de los eslabones y sus desplazamientos con estos se determinan los parámetros, tal como se muestra en la Tabla III.

TABLA III
PARÁMETROS DE COORDENADAS DE LA PATA DEL ROBOT

Articulaciones	α_{i-1}	a_{i-1}	d_i	θ_i
1	$-\pi/2$	0	D	$\theta_1 - \pi/2 = \phi$
2	0	a_2	0	θ_2
3	0	a_3	0	θ_3

$$ {}^{i-1}T_i = \begin{pmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1) $$

$$ {}^{i-1}T_i = \begin{pmatrix} {}^{i-1}R_i & {}^{i-1}P_i \\ 0\ 0\ 0 & 1 \end{pmatrix} \quad (2) $$

${}^{i-1}T_i$: Matriz que transforma los vectores del SC_i al SC_{i-1}, para su descripción.
${}^{i-1}R_i$: Matriz de rotación, describe la rotación del SC_i con respecto al SC_{i-1}.
${}^{i-1}P_i$: Vector de posición, describe la rotación del SC_i con respecto al SC_{i-1} .

La pata tiene 3 GDL por tanto la ecuación (3) tiene 3 matrices de transformación homogéneas donde:

$$ {}^0T_3 = {}^0\ A_1\ {}^1A_2\ {}^2A_3 \quad (3) $$

Se realiza la multiplicación de matrices en el orden que se indica, dado que el producto de matrices no es conmutativo.

$$ {}^0A_1 = \begin{pmatrix} \cos\theta_1 & 0 & -\sin\theta_1 & d\cos\theta_1 \\ \sin\theta_1 & 0 & \cos\theta_1 & d\sin\theta_1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (4) $$

$$ {}^1A_2 = \begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_i & \cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5) $$

$$ {}^2A_3 = \begin{pmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_3\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & a_3\sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (6) $$

Con la ecuación 7, se obtienen el vector de posición (x, y, z) de la pata final con respecto al SC de referencia.

$$ p_0 = {}^0\ T_i\, p_i \quad (7) $$

Considerando a s como sin y c como cos se puede escribir 8.

$$ {}^0p_3 = \begin{pmatrix} a_3c\theta_1c\theta_2c\theta_3 - a_3c\theta_1s\theta_2s\theta_3 + a_2c\theta_1c\theta_2 + ds\theta_1 \\ a_3s\theta_1c\theta_2c\theta_3 - a_3s\theta_1s\theta_2s\theta_3 + a_2s\theta_1c\theta_2 + ds\theta_1 \\ -a_3s\theta_2c\theta_3 - a_3c\theta_2s\theta_3 - a_2s\theta_2 \\ 1 \end{pmatrix} \quad (8) $$

IV. EL MICRO ALGORITMO GENÉTICO (μAG)

Con base en las características de los AGE se debe generar una población aleatoria de individuos, por tanto inicialmente se define al individuo, cada individuo o cromosoma es una solución que contiene los valores requeridos de cada servomecanismo para generar el patrón de marcha del robot (PMR). El individuo o cromosoma está formado por una cadena binaria y en ella se codifica el PMR. El individuo está formado por genes y son representados por bits los cuales forman los valores requeridos en cada servomecanismo para cada posición estática en la marcha de acuerdo a la Figura 3.

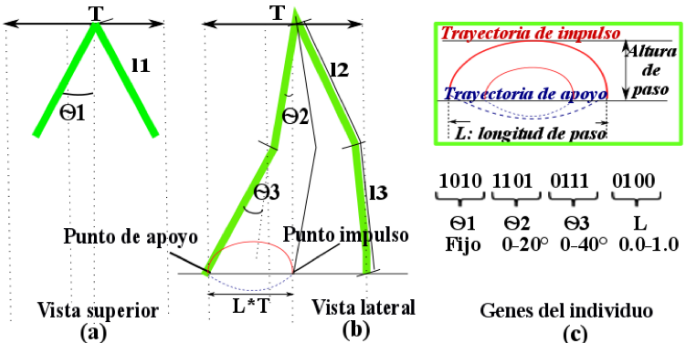


Fig. 3. Estructura binaria del individuo.

Se observa que el desplazamiento de una pata depende de los desplazamientos de los eslabones y son determinados por θ_1, θ_2 y θ_3, los cambios de velocidad lo determina la longitud de paso L, por tanto son cuatro parámetros por cada pata θ_1, θ_2, θ_3 y L que determina la marcha del robot. Se busca el máximo desplazamiento y esto será cuando θ_1=90º fijo, θ_2=20º, θ_3=40º. Por tanto el Individuo solución o función fitness es una cadena binaria que contiene θ_1, θ_2, θ_3 y L que deben alcanzar el máximo valor binario considerando las restricciones del sistema.

Se utilizan los conceptos descritos por Goldberg [2] respecto a los μAG y se realizaron varios experimentos utilizando individuos con representación binaria. Se probó una población de sólo doce individuos, asegurando que esta cantidad son suficientes para la convergencia del algoritmo sin importar la dimensión del cromosoma. Además se aplicaron los operadores genéticos hasta alcanzar una convergencia nominal. Esta convergencia nominal es un ciclo interno que finaliza

cuando los individuos son muy similares entre sí o cuando se alcanza cierto número predefinido de iteraciones. Al finalizar esta convergencia limitada, se obtiene un nuevo individuo (el de mejor aptitud), para posteriormente generar de manera aleatoria los otros once individuos que completarán la nueva población. Bajo este esquema, se precisó un criterio para reinicializar el algoritmo al concluir la convergencia nominal, manteniendo al menos al mejor individuo (elitismo) y generando ruido estocástico al completar la nueva población con individuos concebidos aleatoriamente. De esta manera se incrementa una iteración del ciclo externo del algoritmo y se utiliza la nueva población. En estos resultados, se aseguró que manteniendo elitismo y con el ruido estocástico habría convergencia aunque el número de generaciones sea muy grande de acuerdo con Herrera [15]. Por tanto, el algoritmo básico planteado se puede resumir como muestra en el algoritmo de la Figura 4

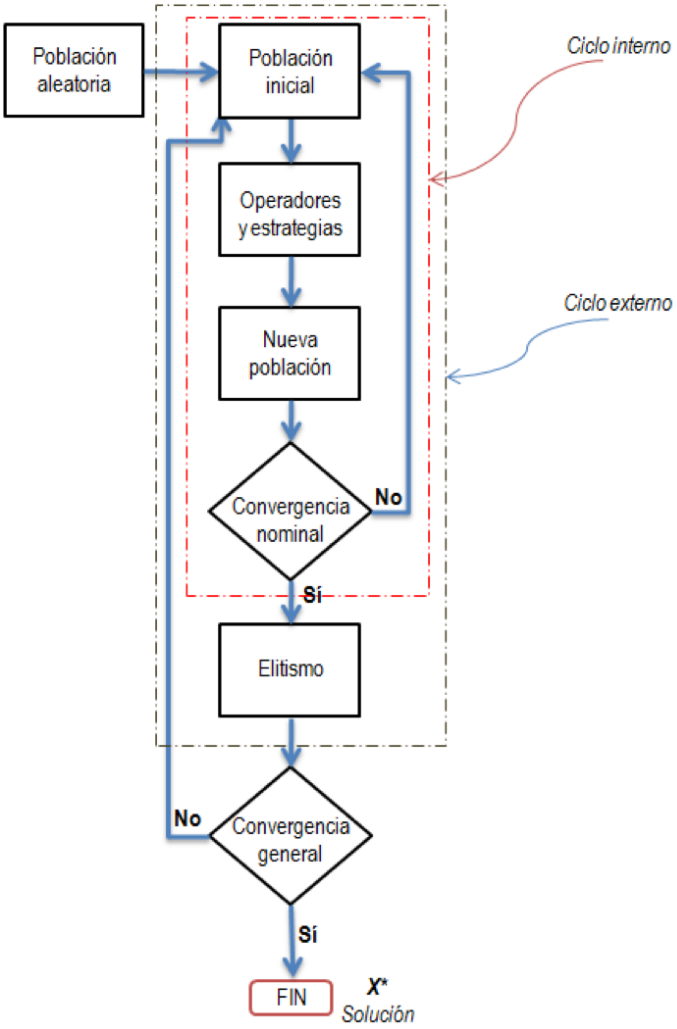


Fig. 4. Diagrama de flujo μAG.

En la Figura 4 existe una población aleatoria de doce cromosomas que se copia a la población inicial. Después de haber aplicado los operadores genéticos se itera el procedimiento de evolución de acuerdo a la convergencia nominal. Si no se ha concluido la convergencia nominal, la población de trabajo siguen siendo los doce cromosomas que están evolucionando, si ya concluyó la convergencia, el mejor cromosoma evolucionado se copia a la población inicial y ésta se completa con once cromosomas generados aleatoriamente. De esta manera, se aprecian los ciclos internos y externos del μAG; el ciclo interno acaba cuando se alcanza la convergencia nominal y el ciclo externo concluirá hasta alcanzar la convergencia general.

Las características que muestran los μAG con poblaciones pequeñas es el interés de nuestra investigación, diseñar micro algoritmos para el aprendizaje de marcha en robots o sistemas más complejos con mayor cantidad de GDL y que se ejecuten en un sistema embebido y no en una computadora.

El Individuo solución es la función fitness, es una cadena binaria de unos que representa el máximo valor de desplazamiento que tiene la marcha del robot, por tanto cada individuo se evalúa con esta función y se eligen las mejores soluciones y tendrán dos destinos:

(i) El mejor individuo se mantiene para generar las siguientes generaciones.
(ii) El mejor individuo se decodifica para obtener los ángulos de desplazamiento de los servomotores y generar la marcha del robot.

Los movimientos de los servomecanismos son controlados por señales periódicas por modulación de ancho de pulso (MAP) generadas por el sistema de control embebido en el cual corre el μAG creando todas las señales requeridas para los servomecanismos de las patas y se generar el patrón de marcha del robot.

Una señal periódica MAP de 20 ms genera el movimiento angular en cada servomecanismo el tiempo TA determina el máximo ángulo a desplazarse y en el tiempo TB, el servomecanismo regresa a su posición original.

V. Implementación del SE

Se diseñó un ambiente de simulación por lo que se determinó obtener la cinemática directa generando las matrices de transformación homogénea para cada articulación a partir de las ecuaciones (1 – 7), estas matrices describieron la geometría espacial, la rotación, traslación de las patas del robot al caminar, las matrices se utilizaron para realizar una simulación en MATLAB, la aplicación se utilizó de referencia para la implementación en la marcha del robot. El AG controló el movimiento de los 12 servomecanismos. Es una tarea complicada que el AG es capaz de realizar. En la Figura 5 se muestra el gráfico de simulación del robot cuadrúpedo.

A partir de la simulación orientó y generó objetivos de la investigación:

(i) Tener un modelo de marcha de referencia dirigido a la implementación en el SE.
(ii) Identificar las áreas de oportunidad para optimizar el algoritmo sin dañar el robot prototipo.

Se generó el patrón de marcha en donde dos patas se desplazan a la vez. El desplazamiento de una pata se reproduce en las tres patas restantes, se desfasada el patrón de marcha en cada pata, primero se logró con la simulación gráfica en MATLAB y después se codificó el programa en Lenguaje C, finalmente se implementó en el SE, las especificaciones

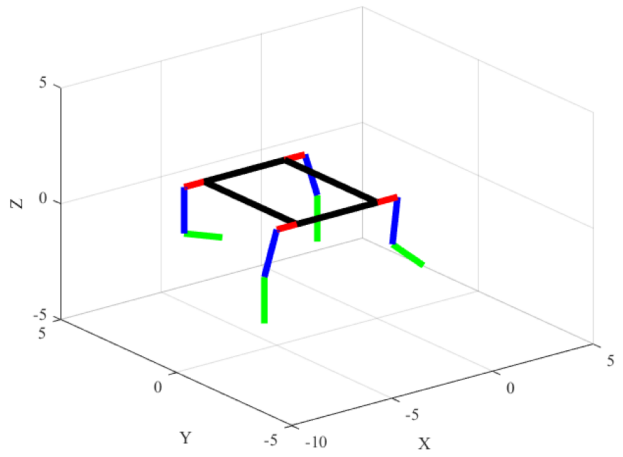


Fig. 5. Gráfico de simulación del robot en MATLAB.

técnicas del SE son: una arquitectura de 8 cores, 32 KB EEPROM, 32 KB RAM, un oscilador de 5 MHz, 32 puertos de entrada/salida, dimensiones de 10.2x7.75x1.6 cm, alimentación de 3.3 VCD, tiene periféricos implementados como CAD, CDA, USB y otros. El sistema y la aplicación se configuraron para programar en lenguaje C. El sistema requiere 12 puertos de salida para los servomecanismos y se tiene 20 puertos más para sensores o cualquier otro dispositivo de entrada/salida. El sistema está limitado por la memoria de programa de 32 Kbytes.

VI. Experimentos

En una primera etapa de experimentos se inicia implementando el AGE con poblaciones de 2000 individuos el programa demandó un espacio de 29.5 Kb y el programa no corrió en el SE. Con el objetivo de evaluar el AGE y el μAG diseñados para la marcha del robot prototipo en la parte de experimentación se establece la siguiente configuración que aplica para las pruebas de ambos algoritmos lo que permito comparar el desempeño del AGE versus el μAG:

(i) Población inicial de 12 individuos por generación generados aleatoriamente.
(ii) Individuos de un byte.
(iii) Poblaciones que van de 10 a 100 generaciones.
(iv) Selección por elitismo del mejor individuo por generación
(v) Un cruce en cada byte del individuo
(vi) Mutación de un bit en cada byte

A. Prueba1

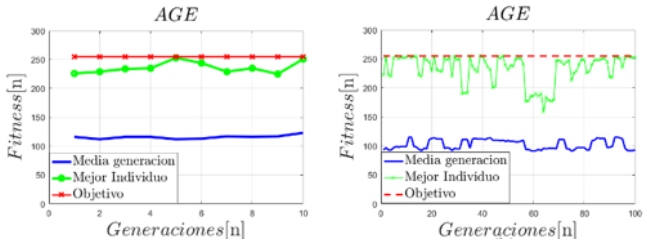
Desempeño del algoritmo, para cada experimento se corre el algoritmo 10 veces, se selecciona el mejor individuo, se calcula el fitness medio de cada generación y se hace una gráfica con la información generada para los algoritmos: AGE, μAG y μAG variando operadores genéticos.

B. Prueba2

Identificar el criterio de paro, para cada experimento se corre el algoritmo 10 veces, en este caso se identifica la generación en que se alcanza el valor fitness de 255 y el tiempo, utilizando una población de 12 individuos y una longitud del individuo de 8 bits. Realizar la prueba para los algoritmos de la prueba 2 y mostrar los datos en una tabla.

C. Resultados de Prueba

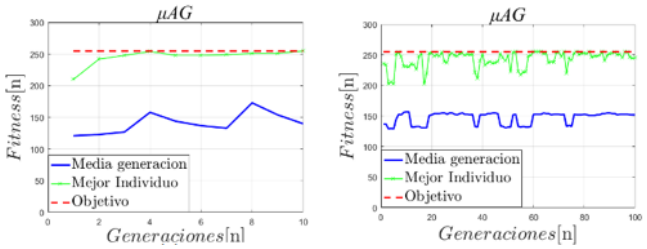
En la Figura 6, se muestran las gráficas de los experimentos del AGE para 10 y 100 generaciones.



(a) 10 generaciones (b) 100 generaciones

Fig. 6. Resultados de experimentos AGE.

En cada generación se gráfica la media de la generación cuyo valor es de 100 y el mejor individuo en cada generación. Se corre el μAG, se aplica elitismo en conservar el mejor individuo de cada generación y se generan de manera aleatoria el resto de la población para completar 10 y 100 generaciones, en la Figura 7 se muestra el comportamiento.



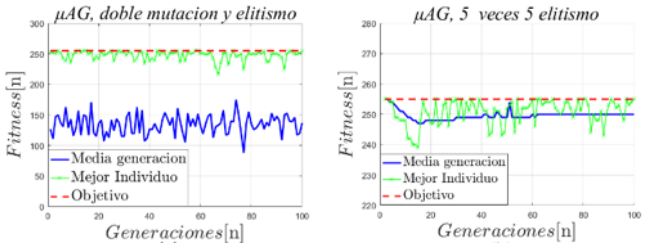
(a) 10 generaciones (b) 100 generaciones

Fig. 7. Resultados del experimento con μAG.

Para las 10 generaciones la media de la generación alcanza un valor de 140 y se alcanza el fitness buscado, para las 100 generaciones, la media de la generación se incrementa a 147 y también se alcanza el objetivo, se muestra en la Figura 7.

En la Figura 8a, se muestra el comportamiento del μAG, variando los operadores genéticos.

La media de cada generación alcanza un valor de 130 y el mejor individuo se acerca más al valor del objetivo, mejorando el experimento anterior. En la Figura 8b se muestran los resultados del μAG5x5, este algoritmo tiene un ciclo interno donde realiza cinco ciclos, esto permite sintonizar el algoritmo para que la convergencia sea rápida al fitness buscado y se selecciona los mejores individuos solución en cada ciclo, una vez que termina se ejecuta el ciclo externo el cual reinicializa el algoritmo para asegurar la diversidad en la búsqueda de



(a) Doble mutación (b) 5 veces elitismo

Fig. 8. Resultados de con μAG, variando los operadores genéticos.

nuevas soluciones, el ciclo externo se realiza 5 veces, por tanto el algoritmo se ejecuta 25 veces para encontrar la solución, este algoritmo tiene la mejor media por generación de 200 y el mejor individuo de cada generación es muy cercano al objetivo.

D. Resultados de Prueba2

Para los casos AGE y el AGE solo elitismo se ejecutan 10 veces, en cada corrida se producen 100 generaciones y se muestran resultados en la Tabla IV.

TABLA IV
DATOS DEL AGE

Tabla I No	AGE Generación	Tiempo (s)	AGE solo elitismo Generación	Tiempo (s)
1	21	0.143	10	0.111
2	15	0.101	11	0.102
3	6	0.033	11	0.1
4	56	0.32	46	0.484
5	68	0.329	9	0.079
6	10	0.067	8	0.058
7	7	0.02	38	0.328
8	5	0.014	16	0.177
9	10	0.068	7	0.0624
10	87	0.483	11	0.112
Valor medio	28.5	0.1578	16.7	0.16134

Para el μAG 5x5 con elitismo y en una segunda etapa el μAG con doble Mutación con elitismo se realizan las pruebas y se muestra los resultados en la Tabla V.

En el μAG los tiempos de procesamiento son favorables de 0.053-0.056s y 9.5-8.5 generaciones, encontró la solución. Las versiones actuales de μAG se implementan en el SE generando una marcha de dos patas a la vez, el programa ocupó un espacio en memoria de programa de 18 Kbytes y se codificaron en Lenguaje C. En la última prueba se implementó el AGE con 10 generaciones y el μAG5x5, para evaluar la marcha del robot. Los valores en azul muestran el desempeño del AGE y se observó una velocidad media de 0.043 m/s, además la marcha del robot es lenta, recta y pausada. Los valores en rojo muestra el desempeño del robot con el μAG5x5 la velocidad media de marcha es tres veces más rápido con respecto al AGE, además la marcha es recta y continúa, se graficaron datos en la Figura 9.

TABLA V
DATOS DE μAG

Tabla II No	μAG 5 x 5 elitismo Ciclo	Tiempo (s)	μAG doble Mutación y elitismo Ciclo	Tiempo (s)
1	4	0.147	2	0.069
2	1	0.008	2	0.069
3	3	0.118	1	0.037
4	2	0.06	1	0.037
5	2	0.06	2	0.069
6	1	0.008	3	0.081
7	1	0.008	3	0.093
8	1	0.008	1	0.037
9	2	0.06	1	0.037
10	2	0.06	1	0.037
Valor medio	1.9X5=9.5	0.0537	1.7X5=8.5	0.0566

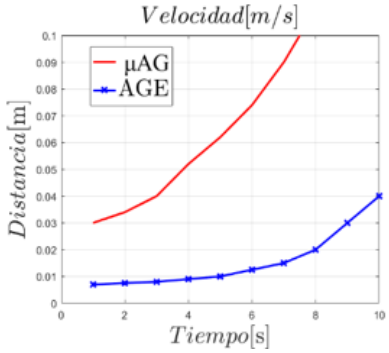


Fig. 9. Velocidad del Robot.

VII. DISCUSIÓN Y RESULTADOS

En la primera etapa de pruebas se identificó que el AGE demanda mayor espacio de memoria de programa, lo anterior se debe a las poblaciones de miles de individuos que maneja el AGE son dos poblaciones que requieren espacio en memoria para la población inicial y la población mejorada. Se redujo la población hasta determinar que el programa se ejecutó en el SE con poblaciones de 500 individuos y encontró la solución al sistema.

Al analizar las gráficas de la Figura 6, la media de la generación del AGE se mantiene y se alcanza el fitness (objetivo). En el caso del μAG desde poblaciones de una decena, la media de la generación se incrementó un 40% alcanza el fitness buscado, y de acuerdo a lo que se observa en las gráficas de la Figura 6 el mejor individuo de cada generación se encuentra a un valor más próximo del objetivo conforme el programa corre y el número de generaciones se incrementa. Se trabaja en el μAG incrementando la cruza y la mutación, se encuentra una mejor respuesta manteniendo el elitismo de los dos mejores individuos de cada generación y se incrementa la mutación al doble obteniendo una mejor convergencia al fitness buscado y se muestra en la Figura 8a.

Al analizar el μAG5x5 por medio de las gráficas de la Figura 8b se concluyó que este último tiene el mejor desempeño al obtener la mayor media en cada generación y la mejor convergencia al fitness en un tiempo de procesamiento menor como se describe a continuación.

En la prueba 2 se identifica en que generación y en qué tiempo se obtiene la mejor solución para el AGE y el μAG5x5, con este último algoritmo se obtiene la solución en menor un menor número de generaciones y en consecuencia un menor tiempo de procesamiento al tener como base poblaciones reducidas para encontrar la solución. Con estos resultados el robot prototipo tiene generación de marcha continua y recta como muestran las tablas y gráficas de resultados.

VIII. Conclusiones

Al inicio de las pruebas se concluye que una optimización del algoritmo es reducir el número de individuos de la población. Los μAG son una opción adecuada para resolver el problema de marcha de robots con patas o movimientos definidos en brazos manipuladores donde se tiene más de 10 GDL, la característica primordial es que generan poblaciones menores a una decena de individuos lo que permite encontrar las soluciones válidas en un menor tiempo. Utilizando los operadores genéticos adecuadamente se logra una rápida convergencia a la solución durante la ejecución del algoritmo propiciando la marcha continua en el robot prototipo cuadrúpedo. El desempeño del μAG es mejor con respecto al AGE, se encuentran las soluciones en menos generaciones, máximo en 10 generaciones y el tiempo de procesamiento es mucho menor como se observa en las gráficas y tablas de los experimentos. Los μAG se implementaron en un SE evitando usar una computadora. El μAG se implementó en el robot prototipo cuadrúpedo en donde se obtuvo una marcha recta y continua, debido a que las soluciones se realizan en un menor tiempo de procesamiento con respecto al AGE y las poblaciones no son mayores a 12 individuos. La investigación de los μAG se observa promisorio por la factibilidad de ser implementados en SE y en trabajos futuros se tiene la opción de enfocarnos en mejorar su rendimiento al estandarizar los operadores genéticos, el tamaño de las poblaciones y generaciones, además si se realiza un algoritmo estructurado en lenguaje ensamblador, se asegura reducir el tiempo de procesamiento de dichos algoritmos.

Referencias

[1] C. De La Cruz, Héctor D. Patiño, and R. Carelli. "New Evolutionary Algorithm based on the Mathematical Modeling of the Evolution of a Species", IEEE Latin America Transactions, Vol. 3, No. 4, pp. 8-14, 2005.

[2] D. E. Goldberg. "Genetic Algorithms in search Optimization e Machine Learning", USA: Addison-Wesley Publishing Company Inc., 1989.

[3] A. Asteroth and A. Hagg. "How to successfully apply genetic algorithms in practice: representation and parametrization", IEEE, 2015 International Symposium On Innovations In Intelligent Systems And Applications (INISTA), pp. 390-395, DOI:10.1109/INISTA.2015.7276778, 2005.

[4] E. Burke, M. Gendreau, M. Hyde and *et al*. "Hyper-heuristics: a survey of the state of the art", Journal of the Operational Research Society, Vol. 64, No. 12, pp. 1695-1724, DOI:10.1057/jors.2013.71, 2013.

[5] F. Ahmadi, R. Tati , S. Ahmdi and V. Hossaini. "New Hardware Engine for Genetic Algorithms", IEEE, Genetic and Evolutionary Computing, International Conference on, pp. 122-126, DOI: 10.1109/ICGEC.2011.37, 2011.

[6] P. Gumiel-Moreno, Y. Sáez-Achaerandio and D. Quintana-Montero, "Implementación de técnicas de computación evolutiva a la programación automática de un robot", Madrid España: Universidad Carlos III de Madrid, Obtenido de http://hdl.handle.net/10016/6581, 2009.

[7] H. Suzuki, H. Nishi, A. Aburadani and S. Inoue, "Animal Gait Generation for Quadrupedal Robot", Innovative Computing, Information and Control, IEEE, pp. 20, DOI:10.1109/ICICIC.2007.169, 2007.

[8] G. B. Parker and William, T. Tarimo, "Using Cyclic Genetic Algorithms to Learn Gaits for an Actual Quadruped", IEEE International Conference onsystems, pp. 1938-1943, DOI: 10.1109/ICSMC.2011.6083871, 2011.

[9] J. Elijah-cKenzie, "Design of Robotic Quadruped legs", Massachusetts, USA: MIT, DOI:785729031-MIT, 2012.

[10] C. Cai and H. Jiang, "Performance Comparisons of Evolutionary Algorithms for Walking Gait Optimization", International Conference on (IEEE), pp. 129-134, DOI:10.1109/ISCC-C.2013.100, 2013.

[11] M. Oliveira, Cristina P. Santos, and Costa Lino, "Sensitivity analysis of multi-objective optimization of CPG parameters for quadruped robot locomotion", AIP Publishing, Vol. 1479, No. 1, pp. 495-498, DOI:10.1063/1.4756175, 2012.

[12] P. Gonzalez de Santos, E. García, R. Ponticelli et. al., "Minimizing energy consumption in hexapod robots". Advanced Robotics, Vol. 23, No. 6, pp. 681-704, DOI:10.1163/156855309X431677, 2009.

[13] A. Marín, J. A. Hernández R., J. A. Jiménez, "Tuning Multivariable Optimal PID Controller for a Continuous Reactor Using an Evolutionary Algorithm", IEEE Latin America Transactions, Vol. 16, No. 2, pp. 422-427, 2018.

[14] L. Teng, X. Wu and W. Chen, "Center of gravity balance approach based on CPG algorithm for locomotion control of a quadruped robot", IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pp. 325-329, DOI: 10.1109/AIM.2013.6584112, 2013.

[15] J. C. Herrera–Lozada, H. Calvo, H. Taud y A. Portilla, "Metodología Generalizada para Diseñar Micro Algoritmos Bioinspirados", CIC-IPN, Vol. 43, pp. 189-199, ISBN:978-607-00-1970-8, 2009.

[16] F. A. Chávez, J. C. Herrera, J. Sandoval *et al*., "La corriente eléctrica como un parámetro clave en la locomoción de un robot cuadrúpedo", Vol. 3, DOI:10.6036/NT8074, 2016.

Francisco Alejandro Chávez Estrada was born in México City, México. He is electronic engineering at the Universidad Autónoma Metropolitana Azcapotzalco (UAMA) in 1990. In 2005 obtained a Master degree in Computer Engineering in Digital Systems at the Centro de Investigación en Cómputación (CIC) from Instituto Politécnico Nacional (IPN) and Ph D. in Robotics Engineering and Mechatronic Systems at the Centro de Innovación y Desarrollo Tecnológico en Cómputo (CIDETEC) from IPN in 2018. He is a professor at the Instituto Tecnológico de Tlalnepantla del TecNM; his research is focused on the application of evolutionary algorithms applied in robotics applying embedded systems.

Juan Carlos Herrera Lozada He received the Ph.D in Computer Science and M.Sc. in Computer Engineering from Centro de Investigación en Computación (CIC) of the Instituto Politecnico Nacional (IPN), in Mexico City. He has authored several articles and speaker at national and international conferences. He has participated in different research projects and technological developments. His research interest include: design of reconfigurable logic devices, practical applications with microcontrollers and embedded systems design.

Jacobo Sandoval Gutiérrez graduated at the Escuela Superior de Ingeniería Mecánica y Eléctrica (ESIME) from Instituto Politécnico Nacional (IPN) in Mexico City, Engineer Industrial Robotics in 2004 and later as Master and Doctor of Advanced Technology at the Centro de Investigación en Ciencia Aplicada y Tecnología Avanzada (CICATA) from IPN in 2006 and 2010 respectively. He is a research professor at the Universidad Autónoma Metropolitana Campus Lerma, Estado de Mexico. He has published several articles related to the field of robotics, as well as patents in the area of Engineering.

María Isabel Cervantes Valencia was student of Master degree of Industrial Engineering at Instituto Tecnológico de Celaya. Engineer at Escuela Superior de Ingeniería Química e Industrias Extractivas (ESIQIE) from Instituto Politécnico Nacional, She is a professor at the Instituto Tecnológico de Tlalnepantla del TecNM, coauthor of Planning, Design and Layout Facilities competency approach.