# Implementación de Micro Algoritmos Genéticos en Robots Manipuladores

*F. A. Chávez-Estrada* *Marco Antonio Quintero-Mercado* *Leslie Janet Carboney Palafox*♣
*Juan Carlos Herrera Lozada , Dr.* *Jacobo Sandoval Gutiérrez, Dr.*

*Instituto Politécnico Nacional*
*CIDETEC*

falexchavez@hotmail.es leslie_carboney@hotmail.com

Resumen

El avance tecnológico demanda el uso de robots en los procesos y tareas repetitivas, lo anterior permite el aumento de la productividad y aumenta la confiabilidad en los procesos y tareas, la especialización de los ingenieros y científicos que desarrollan dichos sistemas se incrementa y han generado nuevas técnicas de solución con los algoritmos genéticos (AG) y han evolucionado dichas soluciones a los micro AG (μAG ), son algoritmos con poblaciones reducidas con las mismas características del AG, demandan menos recursos computacionales , el tiempo de ejecución se reduce notablemente, encuentra las soluciones y con estas nuevas características permite implementarlos en sistemas embebidos (SE) que resuelvan problemas sobre la posición, velocidad y aceleración de los robots manipuladores (RM), robots móviles con ruedas (RMR), robots con patas (RP). En esta investigación se elige un μAG para determinar la posición final de un RM por medio de un SE y se comprueba las ventajas que ofrecen y demuestra que los μAG es un tipo de solución para ayudar en la autonomía de un robot y darles aplicación en los RM por medio de un SE y no un computador.
***Palabras clave:*** *Algoritmos genéticos, micro algoritmos genéticos, sistema embebido, robot manipulador, función fitness.*

## I. Introducción

Los algoritmos genéticos (AG) se consideran métodos adaptativos para resolver problemas de optimización en un sistema de control de un robot, tiene por objetivo mejorar posición, velocidad, torque, aceleración del sistema, es un método alternativo en comparación de las técnicas de control clásico, con la ventaja adicional que el sistema aprende y esto le permita mejorar su desempeño [1], sea una función fitness nos genera las soluciones y seleccionando las mejores; la implementación se lleva a cabo a un sistema de control para observar y comprobar experimentalmente la mejora de su desempeño, tomando en cuenta las restricciones [2] del sistema. Este antecedente es descrito como la clave de los AE [3]. Los ingenieros e investigadores adoptan los AG cuando la complejidad del sistema aumenta, como es el caso en la robótica, donde se incrementan los grados de libertad (GDL) complicando la solución por los métodos de matrices convencionales. El AG no requiere conocimiento previo del sistema a solucionar, se adapta y genera

soluciones, por esta razón tiene una variedad de aplicaciones en Ingeniería, se describe a continuación algunas investigaciones: En los robots manipuladores que tiene menos GDL se aplican las técnicas evolutivas, Vaca y Peña resuelven la cinemática inversa de un robot manipulador serial de 4 GDL aplicando un AG a partir del modelo cinemático directo screws [4] en un ambiente simulado. Estas técnicas se destacan por su amplia gama de aplicación, además se destacan por ser robustas, se aplican donde no existen técnicas especializadas, no requieren conocimiento del problema y mejoran las técnicas tradicionales. Chaohong y Hong en sus investigaciones [5], aborda la optimización de marcha con cuatro diferentes AG y solo compara las soluciones encontradas por estos métodos evaluando la velocidad, estabilidad y flexibilidad, mostrando resultados en simulación. Considerando las fortalezas de estos algoritmos, se decide utilizar como base un AG y se toma de referencia los μAG como expone Herrera Lozada en su investigación [6]. Las características que muestran los μAG con poblaciones menores a una decena y encuentran las soluciones de los sistemas, es aquí donde nace el interés de nuestra investigación, diseñar micro algoritmos para la autonomía de marcha en robots, se ejecute en un SE y no en una computadora, desarrollar un robot prototipo que beneficie al usuario en sus investigaciones de marcha o aplicarlo en la exploración en zonas de riesgo para el ser humano, aplicar los μAG para resolver problemas en determinar la posición final de: brazos manipuladores [7], robots bípedos [8] [9], cuadrúpedos [5] y hexápodos [10].

El artículo se ha organizado en secciones como se indica: I Introducción, II Micro Algoritmos Genéticos (μAG), III Desarrollo y resultados, IV La implementación del SE y V Conclusiones.

## II. Micro Algoritmos genéticos (μAG)

Los μAG son una versión de los AG, tienen las mismas bases y características, con la excepción sobre las poblaciones que genera pueden ser de una a dos decenas de individuos solución o incluso menos a una decena de Individuos, Goldberg [3] en su investigación pone las bases en estos algoritmos. Krishnakumar en [11] utiliza algoritmos genéticos simples (AG) con representación binaria y poblaciones reducidas, de igual manera Herrra Lozada en su investigación trabaja los μAG e indican que la clave es aplicar los operadores genéticos hasta alcanzar una convergencia nominal. ***Convergencia nominal,*** es un ciclo interno que concluye cuando los individuos son muy similares entre sí o cuando se alcanza cierto número predefinido de iteraciones. Se obtiene el individuo de mejor aptitud (función fitness), para posteriormente generar de manera aleatoria los otros individuos que completarán la nueva población es la etapa de inicialización del algoritmo de Goldberg, lo que permitirá diversidad en la soluciones y evitar la **convergencia sin diversidad**. Se muestra el diagrama de flujo del μAG en la figura 1.

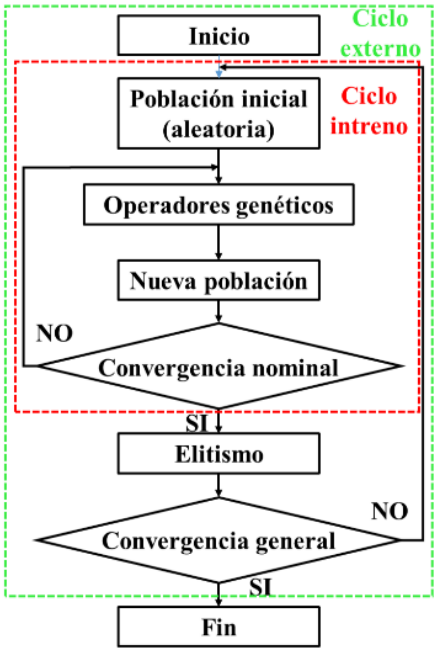


Figura1. Diagrama de flujo del μAG

- Población aleatoria de cromosomas.
- Aplicando los operadores genéticos, se itera el método de evolución de acuerdo a la convergencia nominal (ciclo interno).
- La **convergencia nominal**, se ejecutan los operadores genéticos para evolucionar la población.
- Concluida la **convergencia**, por elitismo se conserva el mejor cromosoma y el resto de población inicial se completa con cromosomas generados aleatoriamente.
- El **ciclo externo concluirá hasta que se haya alcanzado la convergencia general** (ciclo externo)**.**

La principal característica de los AG que encuentran soluciones en distintas áreas de la ingeniería donde las soluciones son duras y complejas, donde los métodos tradicionales demandan grandes recursos de cómputo en la búsqueda de las soluciones tal es el caso de encontrar la posición final de los robots manipuladores por medio de la cinemática inversa, por medio de matrices de rotación y transformación, la solución se complica si aumentan los GDL. Utilizar un AG nos permite encontrar la posición final del RM con mayor facilidad sin embargo debido a que estos algoritmos son poblacionales, es decir se deben generar miles de individuos solución demandan un computador para ejecutar el algoritmo, se elige utilizar un μAG, tiene las características del anterior, nos permite encontrar la posición final del RM debido a que utiliza poblaciones menores a una decena de individuos soluciones y el algoritmo demanda menos recurso de cómputo, el μAG se puede ejecutar en un sistema embebido (SE) en un menor tiempo. En estos últimos algoritmos radica nuestro interés de la investigación, comprobar que utilizando un μAG en un SE, se determina la posición final de un RM.

**III. Desarrollo y resultados**

El algoritmo debe cumplir las siguientes características:

- Individuos de un byte. La función fitness es alcanzar el valor de 255.
- Población inicial de 16 individuos generados aleatoriamente.
- Generaciones que van de 10 y 100 unidades.
- Selección por elitismo de los dos mejores individuos.
- Un cruce en cada byte del individuo
- Mutación de un bit en cada byte.

1. Desarrollar un µAG 5x5 (5 ciclos externos X 5 ciclos internos= 25 generaciones). Cada generación de este µAG se realizan 25 generaciones. El µAG 5x5 se ejecuta para 100 generaciones y se analizan los datos.
2. Desarrollar un µAG 5x5 (5 ciclos externos X 5 ciclos internos= 25 generaciones). Cada generación de este µAG se realizan 25 generaciones. El µAG 5x5 se ejecuta para 10 generaciones y se analizan los datos con respecto a una generación aleatoria de individuos.
3. Exponer los resultados y analizando la información.

**Resultados**

1. El µAG de 100 generaciones se ejecuta y genera las soluciones en cada generación y se grafican sus soluciones, se muestran en la figura 2.

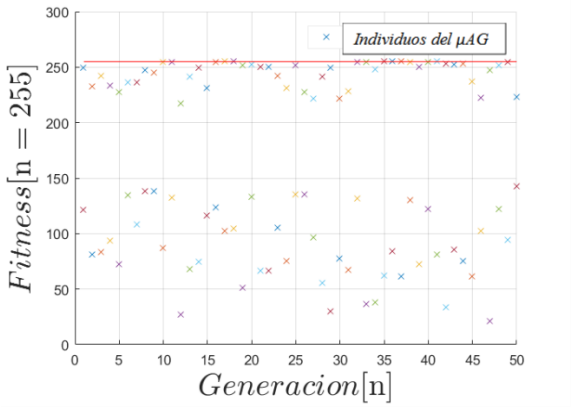


Figura 2. Soluciones del µAG

2. Se ejecuta µAG 5x5 de 100 generaciones y se observa que el algoritmo encuentra la solución en la generación 10 a partir de este valor alcanza la mejor optimización, los tres mejores cromosomas de cada generación tienden alcanzar el valor de a 255. Se observa en la figura 3, en cada generación se tiene diversidad en los individuos tienen valores de 20 hasta 255 y siempre se obtienen individuos optimizados.
   Se genera una población aleatoria del mismo tamaño y se grafican sus soluciones en la figura 3.

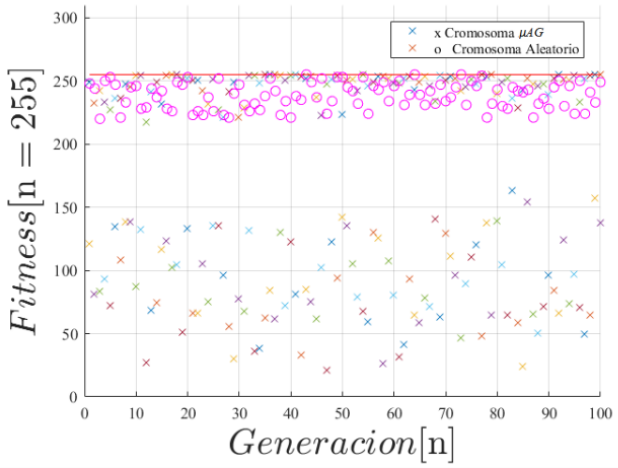


Figura 3. Soluciones μAG 5x5 vs población aleatoria

La población aleatoria (círculos) no tiene diversidad, incluso pocos cromosomas alcanzan el valor máximo de 255 con respecto al μAG 5x5 (X).

**IV. La implementación del SE**

Se evalúan las trayectorias válidas en el prototipo variando los ángulos de las articulaciones y se determinan las restricciones de cada articulación, resultado del desplazamiento angular del servo, máximo 180˚ de cada eslabón. Tabla I

| **Restricciones** | **RM 3 GDL** |
|---|---|
| **Eslabón** | **Ángulos válidos** |
| Eslabón 1 | θ1 = 0 – 180˚ |
| Eslabón 2 | θ2 = 0 – 180˚ |
| Eslabón 3 | θ3=0 – 180˚ |

Tabla I. Rango angular de cada eslabón

Cada ángulo estará representado por un byte por tanto se tiene un cromosoma de 3 bytes. La caracterización del desplazamiento angular de cada articulación es de acuerdo a la Tabla II.

| Ángulo | Decimal | Binario |
|---|---|---|
| Mínimo | 0 | 0000 0000 |
| Máximo | 255 | 1111 1111 |

Tabla II. Caracterización del desplazamiento angular

Investigación: Obtener la matriz de transformación de un brazo manipulador de 3 GDL y por medio del vector de posición comprobar la ecuación (1) obtenida por medio del método Denavit Hartenberg.

$$X= a3*\cos(\Theta_1)*\cos(\Theta_2)*\cos(\Theta_3)-a3*\cos(\Theta_1)*\sin(\Theta_2)*\sin(\Theta_3)+a2*\cos(\Theta_1)*\cos(\Theta_2)+d2*\cos(\Theta_1)$$

$$Y= a3*\sin(\Theta_1)*\cos(\Theta_2)*\cos(\Theta_3)-a3*\sin(\Theta_1)*\sin(\Theta_2)*\sin(\Theta_3)+a2*\sin(\Theta_1)*\cos(\Theta_2)+d2*\sin(\Theta_1)$$

$$Z= -a3*\sin(\Theta_2)*\cos(\Theta_3)-a3*\cos(\Theta_2)*\sin(\Theta_3)-a2*\sin(\Theta_2) \quad \text{---------(1)}$$

***Es el vector de posición que determina la posición final del RM, las soluciones se obtienen a partir de la cinemática directa.***

Obtener el desplazamiento del RM al sustituir los ángulos válidos en las ecuaciones del vector de posición obtenido a partir de la matriz de transformación del RM, se ejecuta en el SE sin embargo los resultados tardan en generarse y en ocaciones no se ejecuta. Se decide ejecutarlo en la computadora y se genera un modelado en MATLAB, se muestra en figura 4. Al evaluar la ecuación (1), se obtendrán las coordenadas en $R^3$ con respecto al origen del RM: x=2.3381, y=0 y z= -1.4870

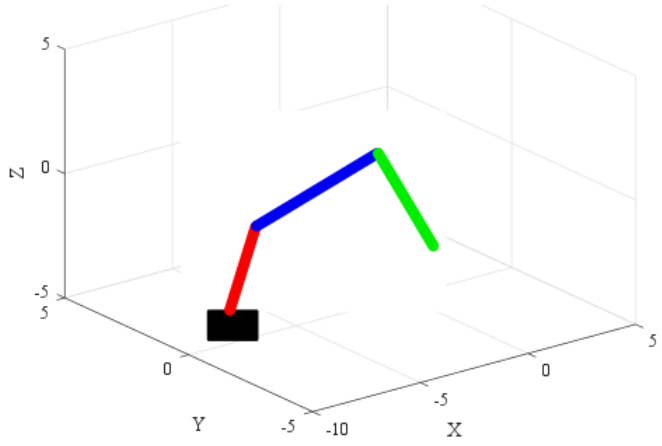


Figura 4. Modelado en MATLAB del RM

Se implementa el μAG 5x5 de 100 y se ejecuta sin problemas en el SE, se generan las poblaciones de individuos solución en cada generación, se requieren 3 individuos optimizados para generar el desplazamiento deseado del RM a la posición deseada, se muestra prototipo del RM en figura 5.

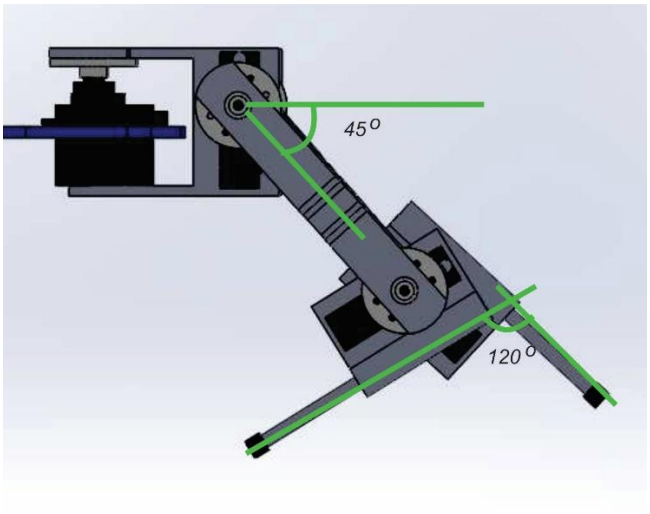


Figura 5. Prototipo del RM con 3 GDL

**V. Conclusiones**

Los µAG nos permiten determinar la posición final del último eslabón en un RM de *n* GDL e implementarlo en un SE lo anterior es posible a que los µAG encuentran las soluciones buscadas en poblaciones menores a una decena de individuos, menor número de individuos menores recursos de cómputo, se reduce el algoritmo y el tiempo de ejecución, por tanto esto demanda menos recursos y no se requiere una computadora. Estas nuevas características nos permite aplicarlo a RM, RMR y RP, para dotarlos de una mayor autonomía al trabajar el sistema con un SE, el cual demanda menos energía y baterías de menor tamaño.

Referencias

[1] L. Araujo, Algoritmos Evolutivos un Enfoque Práctico, Alfaomega, 2009.

[2] J. H. Holland, « Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and artificial intelligence,» MIT press, 1992.

[3] D. Golberg, «Genetic algorithms in search, optimization and machine learning,» Addion Wesley, 1989.

[4] J. J. V. a. C. C. A. P. a. G. H. V. González, «Cinemática inversa de robot serial utilizando algoritmo genético basado en MCDS,» Revista Tecnura, vol. 19, nº 44, pp. 33--46, 2015.

[5] C. a. J. H. Cai, «Performance Comparisons of Evolutionary Algorithms for Walking Gait Optimization,» International Conference on(IEEE), pp. 129--134, 2013.

[6] J. C. Herrera Lozada, «Sistema inmune artificial con población reducida para optimización numérica,» CIC-IPN, 2011.

[7] L. F. a. D.-T. E. a. C.-D. G. Giraldo, «Cinemática Inversa de un Brazo Robot Utilizando Algoritmos Genéticos,» RASI, vol. 3, nº 1, pp. 29--34, 2006.

[8] G. a. P. V. a. K. P. D. Dip, «Genetic algorithm-based optimal bipedal walking gait synthesis considering tradeo between stability margin and speed,» Cambridge Univ Press(Robotica), vol. 27, nº 03, pp. 355--365, 2009.

[9] F. B. J. G. M. R. a. A. Peregrón, «Optimización Evolutiva de la Locomoción de un Robot Bípedo».

[10] C.-F. a. C. Y.-H. a. J. Y.-H. Juang, «Wall-following control of a hexapod robot using a data-driven fuzzy controller learned through diferential evolution,» IEEE Transactions on Industrial Electronics, vol. 62, nº 1, pp. 611--619, 2015.

[11] K. Krishnakumar, «Micro-genetic algorithms for stationary and non-stationary function optimization,» SPIE Proceddings: Intelligent Control and Adaptive systems, pp. 289-296, 1989.