Memorias del 2022 Congreso Nacional de Control Automático
12-14 de Octubre, 2022. Tuxtla Gutiérrez, México.

Evolutionary Algorithm to Auto-tuning Hyperbolic Tangent Sliding-Mode Controllers

Francsico-David Hernandez * **Domingo Cortes** *
Maria-Margarita Vargas *

* *Instituto Politécnico Nacional, ESIME Culhuacan.*
(e-mail: <fcod.hdez@gmail.com>, <domingocortes@gmail.com>,
<maria.vargasm.08@gmail.com>).

Abstract
In recent years, control engineers has started to explore new control, optimization and tune paradigms. And they have found that Evolutionary Algorithms (EA) are good alternatives for traditional methods. Thanks to the power to explore high search-spaces with multicombinational problems and a lot of potential solutions. For this reason, in this work an Evolitive Algorithm is planned as a heuristic auto tunning algorithm for Sliding-Mode Controllers. Using an approach by Hyperbolic Tangents. In order to obtain easily implementable control laws with a desirable performance to linear and non-linear systems.

Keywords: Applied control, Genetic Programming, Evolutionary Algorithms, Genetic Algortihm, Genetic Programming, Control, Sliding-Mode Control, Hyperbolic-Tangent

1. INTRODUCCIÓN

Los algoritmos evolutivos (EA, por sus siglas en inglés), son algoritmos computacionales donde se emula la evolución que organismo vivos sufren para adaptarse al medio ambiente donde se desarrollan. Esta caracteristica ha permitido que desde las primeras propuestas con los algortimos genéticos, se hayan adaptado a múltiples problemas computacionales. Uno de los enfoques que más ha mostrado el poder de los EA, es la optimización metaheurística. Este tipo de optimización, generalmente se aplica a problemas donde encontrar una solución analítica es complicada, para aquellos problemas donde se conoce pocas condiciones o bien cuando no es posible implementar un método de óptimización. En este tipo de enfoques, cada individuo codifican una solución potencial, el ambiente y las restricciones bajo la cuales los individuos se desarrollan representan la problemática a solucionar (Slowik and Kwasnicka, 2020). El objetivo es encontrar: nuevos y mejores individuos a travez de la propagación y recobinación de los más adaptados. Lo que se traduce como soluciones cada vez mejores a la problemática.

Para generar las nuevas poblaciones, se codifican mecanismos que imitan la evolución biológica y selección natural de la teoría de Darwing. Estos procesos incluyen formas particulares de combinar, reconstruir o modificar a los individuo. Generalmente se conocmen como operaciones genéticas.

Para imitar, los proceso biológicos, la primera población de individuos, así como la selección de operaciones genéticas se llevan a cabo de manera pseudo aletoria. La califiación de los individuos está dada por una función de aptitud o de costo, con la cual se evalua cuantitativamente el desempeño de cada solución. Siendo ambas funciones recíprocas. A mayor aptitud, menor costo y viceversa. Por lo que el algoritmo se puede visualizar con un método de optimización que se enfoca a maximizar la función de aptitud o minimizar el valor de costo.

Recientemente este tipo de algoritmos, ha generado interés en el área de ingeniería en control, por dar solución con éxito relativo a sistemas en los cuales se tiene poca infomación del modelo (Joseph et al., 2022), con dinámicas complicadas o no modeladas (Alfaro-Cid et al., 2008) y en problemas combitorios. Por ejemplo, en (Hui et al., 2020) se propone que cada individuo codifique un controlador PID, seleccionando de manera inicial valores aleatorios y libres para las constantes K_p,K_i, y K_d. A través de la recombinación de cada una de estas, se elige a los controladores estabilizantes y se descarta a los que inestabilizan al sistemas. En (Samakwong and Assawinchaichote, 2016), de igual forma se codifican controladres tipo PID, ademas abordan una comparativa entre métodos de sintonización tradicional contra el algoritmos genéticos estandar propuesto. Los autores de (Lazarevic et al., 2013), también codifican controladores PID, agregando restricciones en las combinaciones de las constantes K_p,K_i, K_d. Y Describen la importancia de la funcion de costo para evaluar

el desempeño de cada controlador y su importancia en la convergencia del algoritmo. Además de mencionar algunas de las problemáticas y problemas que aun existen para este tipo sistemas de optimización.

Los trabajos antes mencionados, tienen la particularidad de que los individuos codifican estructuras fijas (Controladores tipo PID) y el algortimo evolutivo, solo genera combinaciones de las constantes K_p,K_i, K_d. Manteniendo estructura fija que se mantiene durante toda la búsqueda, por lo que no se pueden evaluar otros tipos de estructuras de control. Una caracteristica de los algortimos evolutivos bajo el enfoque de programación genética, es que permite evolucionar el tipo de estructuras, además de los parámetros. En (Duriez et al., 2015), se plantea que cada individuo codifique una expresión algebraica como ley de control. Quedando la expresión completamente abierta, únicamente restringida por la cantidad de operaciones anidadas, el tipo de funciones que puede generar y las variables independientes: estados del sistema. Esto tiene como consecuencia un espacio de búsquqeda demasiado amplio, generando un alto costo computacional, la realentización o inlcusive la nula convergencia del algoritmo a una solución aceptable.

En este trabajo se considera al algortimo evolutivo bajo un enfoque intermedio, donde se pueda modificar tanto la estructura del controlador, como los parámetros del mismo. Generando un algoritmo que sea capaz de generar leyes de control orientadas a un tipo inicial de técnica de control reportada en literatura, pero que durante el proceso de búsqueda, esta pueda evolucionar con el objetivo de encontrar nuevas propuestas. Por lo que para este trabajo, se elgió una técnica de control donde se puedan construir expresiones muy complejas, con un costo computacional relativamente bajo. El control por Modos Deslizante (SMC, por sus siglas en inglés), cuya idea fundamental es mantener al sistema en un espacio de control en donde los estados se mantienen dentro de un rango aceptable.

1.1 Control por Modos Deslizantes

En este trabajo, se eligió al SMC por ser uno de los métodos de control más robustos, tanto para sistemas lineales como no lineales. Proporcionan una solución directa para la entrada de control. Una de las principales ventajas del método es la robustez ante variaciones paramétrica, asi como ante perturbaciones externas. Además del tratamiento de plantas de alto orden y no lineales. Por lo que es una de la técnicas más poderosas para diseños prácticos (Utkin and Chang, 2002).

Sin embargo, estas expresiones generan cambios de muy alta frecuencia en el controlador, propagándose como oscilaciones en la salida (Prieto et al., 2015). Con el objetivo de obtener controladores implementables y disminuir las consecuencias que los SMC acarrean, se implementó una técnica de suavizado reportado en la literatura. La aproximación de funciones de conmutación por funciones continuas, mediante tangentes hiperbólicas (Ding et al., 2018). Esta técnica se basa en la semejanza que tiene la función signo con la tangente hiperbólica. La idea central del control por modos deslizantes por tangentes hiperbólicas (HTSMC, por sus siglas en inglés), es crear una superficie deslizante adecuada para que los errores de seguimiento y las desviaciones de salida se reduzcan a un nivel satisfactorio o incluso cero.

En este sentido, en este trabajo se propone utilizar el algoritmo evolutivo de manera general. El cual, inicialmente crea una población de individuos, donde cada uno codifica una superficies deslizantes, como solución candidatos para el sistema a controlar. Siendo el objetivo: obtener controladores estáticos con un gran margen de robustez frente a efectos no lineales, sencillos de implementar y que puedan realizar seguimiento de referencias.

El presente trabajo se distribuye de la siguiente manera, en la Sección 2, se describe el algoritmo evolutivo como método de búsqueda de controladores. En la Sección 3, se describe la codificación de la planta para utilizarse en conjunto con el algoritmo. En la Sección4 se describe la codificación de los controladores y de la población inicial. En la Sección 5 se describen las operaciones genéticas, así como las restricciones propuestas para cada uno. En la Sección 7 se muestran los resultados en simulación para un péndulo simple y finalmente en la última sección se dan las conclusiones.

2. ALGORITMO EVOLUTIVO PROPUESTO

El diagrama a bloques de la Figura 1 resume el funcionamiento del algoritmo evolutivo como auto sintonizador de leyes de control. Y en la Figura 2 se muestra como opera en conjunto con una planta. El algortimo comienza con la creación psdeudo aletoria de una familia de controladores.

Se evealuan cada uno de los individuos, y si alguno presenta un desempeño mayor o igual al buscado, el algoritmo se detiene y arroja a ese individuo como la solución optimizada para el sistema, en caso contrario, se genera una nueva población, intentando que en la siguiente generación se cumpla la condición anterior. De no cumplirse, el algoritmo se ejecuta hasta alcanzar un máximo de generaciones.

El objetivo es mejorar cada controlador hasta encontrar y sintonizar una ley de control u, cuya estructura y parámetros mantengan una mínima diferencia entre el valor deseado y_d y el valor real de salida del sistema y. La relación entre ambas señales $e = y_d - y$ se denomina error. Por lo que el objetivo del algoritmo es encontrar un controlador del tipo que haga la función de costo $J(e)$ mínima.

En este trabajo, el criterio para evaluar el desempeño de cada controlador está constituido por el cuadrado de la norma-2 (NORM) y la función absoluta de tiempo integrado (ITA). Las cuales están dadas por (1) y (2), respectivamente. La función de costo $J(e)$ se implementa

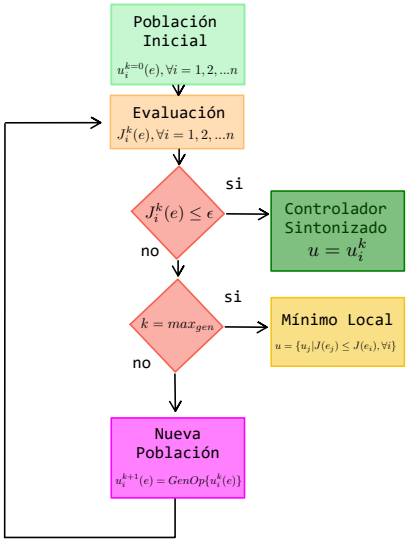


Figura 1. Algoritmo evolutivo propuesto.

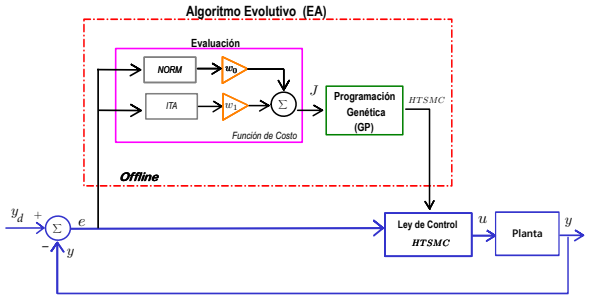


Figura 2. Algoritmo evolutivo Propuesto.

como una combinación ponderada de ambas (3), la cual se utiliza como función objetivo para obtener la mejor estructura y parámetros de la ley de control a través del algoritmo evolutivo.

$$NORM(e) = \int_{t_0}^{t_f} |e(t)|^2 dt \tag{1}$$

$$ITA(e) = \int_{t_0}^{t_f} t|e(t)|dt \tag{2}$$

$$J(e) = w_0 \cdot NORM(e) + w_1 \cdot ITA(e) \tag{3}$$

La morfología de los controladores, será una expresión algebraica y estática en función del error. Cuya forma inicial se especifica en (4). La forma final de la expresión será el resultado de las múltiples mutaciones y recombinaciones que esta pueda sufrir.

$$u_n(e) = [k_b \cdot \tanh(k_a \cdot f_1(e))] \otimes f_2(e) \tag{4}$$

Donde: $\otimes$ es un operador binario elegido aleatoriamente, k_a y k_b constantes aleatorias y $f_1(e), f_2(e)$ funciones algebraicas del error.

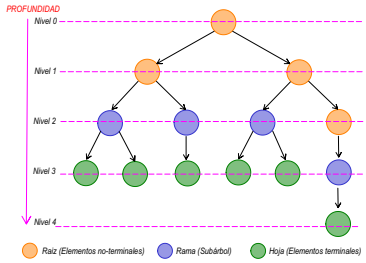


Figura 3. Representación de un árbol de profundidad 4.

3. DESCRIPCIÓN DE LA PLANTA

Si bien, el algoritmo opera bajo un principio heurístico, donde la única información para evaluar el comportamiento del sistema sería analizando la señal de salida $y(t)$, para evalaur computacionalmente al sistema junto con el controlador, es necesario realizar una simulación numérica. Dentro del algoritmo, la planta debe estar modelada en una representación en variables de estado, la cual se describe en un objeto llamado 'rdynsys', quien recupera la siguiente información:

- Ecuaciones de estado.
- Expresione de salida.
- Símbolos empleados para la definición de los estados.
- Condiciones iniciales de cada estado.
- Valores de los parámetros auxiliares.
- Símbolos de las señales de entrada.

Con esta información, un módulo del algoritmo, traduce este objeto, asi como el controlador en un archivo de simulación ODE en lenguaje XPPAUT. El cual realiza la simulación numérica y guarda en otro archivo de datos, las series de tiempo de las señales de estado, salida, error y referencia. Posteriormene el archivo es tratado para evaluar la función de costo.

4. CONTROLADORES

El programa genera las superficies deslizantes en forma de funciones algebraicas, las cuales se representan en estructuras de tipo árbol y se codifican en forma de listas anidadas. Esta representación es útil, ya que permite la total manipulación de las expresiónes, pudiendo así modificar, eliminar o agregar cualquier parte de la expresión, simplemente con modificar elementos de las listas que lo representan.

En la Figura 3, se aprecian los elementos principales que conforman a un árbol: nodos, ramas (sub-árboles) y hojas (elementos de terminación). Así como, la profundidad del árbol (máxima cantidad de elementos anidado desde el nodo raíz hasta la hoja del último sub-árbol).

La funciones algebraicas que se pueden generar, se restringieron por dos conjuntos. OP y S. El primero es un conjunto con operadores unarios y binarios. En el segundo se encuentran los simbolos y elementos constantes.

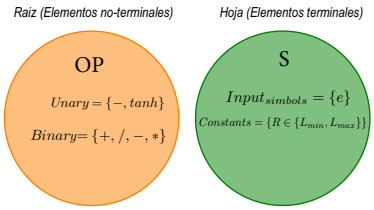
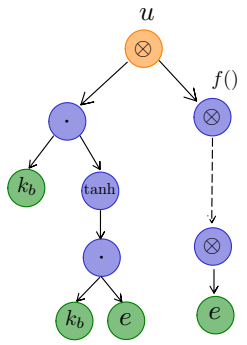


Figura 4. Elementos terminales y no-terminales.

Los símbolos denotan las variables independientes de las funciones. Para este trabajo, solo se trataron funciones del error, por lo que el único simbolo definido es e. Sin embargo modificando este rubro, facilmente su pueden generar expresiones dependientes de los estados, el tiempo y/o de la salida, etc. La Figura 4 ilustra los elementos que continen ambos conjuntos.

4.1 Población Inicial

La familia de controladores iniciales, se conforman por dos partes principales. La primera es una expresión algebraica totalmente aleatoria, solo restringida por la longitud máxima definida en el algoritmo evolutivo. La segunda es una propuesta aleatoria de superficie deslizante. Donde se tiene una tangente hiperbólica escalada por una constante y como argumento de esta, una función algebraica libre. Ambas funciones son combinadas por un operador binario, seleccionado aleatoriamente. Es decir cada individuo de la población inicial, está codificado como un árbol con dos ramas. En la Figura 5 se observa que esta morfología permite un número infinito de leyes de control u, lo que sigue definiendo un amplio espacio de búsqueda pero ahora bajo un enfoque de exploración definido.



$$u = (k_b \cdot \tanh(k_a \cdot e)) \otimes f(e)$$

Figura 5. Representación de una expresión algebraica a estructura árbol.

5. OPERACIONES GENÉTICAS Y RESTRICCIÓN EN MUTACIONES

Se plantearon tres operaciones genéticas y algunas variantes de estas:

1. Réplica: Esta función no modifica en absoluto al individuo, esta operación propaga al individuo sin tener en cuenta su desempeño. Este proceso emula la reproducción asexual.
2. Cruza: Crea un nuevo individuo como la mezcla de dos individuos llamados padres.
   a) Mutua: Selecciona una rama a cada uno de los padres, las cuales son reemplazadas mutuamente. Esta genera dos individuos, de los cuales se selecciona aleatoriamente uno.
   b) Ponderada: Las ramas de los padres se combinan a través de un porcentaje $r_new = \{(1-P)\cdot r_A\} + \{P \cdot r_B\}$ con $P \in [0, 1]$. El índice de porcentaje se elige aleatoriamente.
3. Mutación: Esta operación modifica solo una parte específica del individuo.
   a) Reemplazo: Se sustituye una rama por otra totalmente nueva.
   b) Truncamiento: Una rama es reemplazada por alguna de sus hojas.
   c) Herencia: La rama de un árbol se convierte en el árbol principal.
   d) Reparametrización: Alguna hoja de una rama es modificadas.
   e) Escalamiento: Una rama del árbol se multiplica por un factor.

En este sentido, la cruza sirve para explorar a lo largo del espacio de búsqueda, pudiendo encontrar así un mínimo global, mientras que la función de mutación ayuda a sintonizar las leyes de control.

Uno de los principales retos en la aplicación del EA como auto-sintonizador de controladores fue evitar la generación espontanea. Por lo que se optó por cambios suaves en las expresiones, la cuales se combinan bajo las siguientes reglas: Siguiendo el diagrama de la Figura 6. El procedimiento para realizar cualquier operación de mutación es el siguiente: 1) Seleccionar aleatoriamente la ramificación del tipo: discontinua o expresión algebraica AoB. Si la rama elegida corresponde a la expresión discontinua, las modificaciones solo se llevan a cabo en las ramas C o D. Por el contrario, si la rama seleccionada corresponde a B, esta se modifica libremente. En las operaciones de cruza, solo se recombinan entre grupos, por ejemplo no es posible combinar la D de un padre con la B de otro.

6. POBLACIÓN NUEVA

La siguiente población se crea a partir de la evaluación de una familia de controladores, en función del desempeño cuantificado por (3). La nueva población se conforma por: La élite de la población. Un porcentaje de los individuos con mejor desempeño. Los cuales son propagados sin recibir modificación alguna. Y el resto de la población se completa con los individuos resultantes de un subproceso de selección llamado torneo, donde se crean aletoriamente grupos más pequeños de la población restante y se elige al mejor evaluado para aplicarle una operación genética.

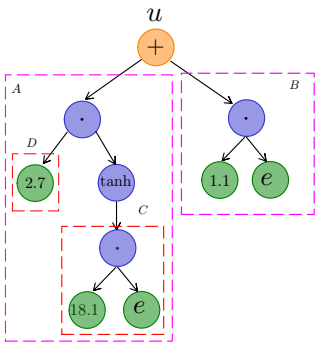


Figura 6. Representación de una expresión algebraica a estructura árbol.

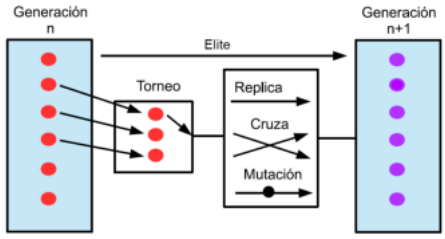


Figura 7. Población Nueva

Cada torneo selecciona de manera aleatoria una operación genética (Réplica, Cruza o Mutación).

La Réplica y Mutación asemejan la reproducción asexual, donde no existe intercambio de material genético. Mientras que la Cruza, la reproducción sexual donde el resultado es el intercambio de información y material genético de dos individuos. El proceso para la obtención de una nueva familia de controladores a partir de la evaluación de una anterior se observa en la Figura 7.

7. RESULTADOS

Para verificar la convergencia del algortimo así como el enfoque de control propuesto, se probó con el modelo no lineal de un péndulo simple, un caso de estudio ampliamanete utilizado en el área de control. El cuyo modelo está dado por (5).

Se probaron dos condiciones: 1) Un cambio de referencia de $\frac{\pi}{2}$ a un inclremento del 10 %. y 2) Una perturbación en la señal de salida, ambas a un tiempo de simulación igual a $t = 7.5[s]$. El algoritmo se evaluó bajo los parámetros de simulación y evolución mostrado en la Tabla 7. Luego de evaluar las 100 generaciones y con un tiempo de ejecución de $54[min]$, se obtuvieron los resultados siguientes.

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{g}{l} sin(x_1) - (b * x_2) + \frac{u}{(m * l^2)} \end{aligned} \tag{5}$$

Con $m = 0.5, g = 9.81, l = 1.0, b = 1.0$;

Tabla 1. Parámetros de Simulación y Ejecución

Población	
Parametro	***Valor***
Población	100 indvs
Max_{Gen}	30
$Prob_{replica}$	10 %
$Prob_{mutación}$	60 %
$Prob_{cruza}$	0.30 %
Pob. Élite	10 %
Ponderaciones	$w_1 = 0.5, w_2 = 0.5$
Simulación	
Referencia	$r(t) = \{\pi/2\}$
Condiciones iniciales	$x_1(0) = 0.25, x_2(0) = 0.0$
Tiempo de Simulación	$t_0 = 0.0$, $t_f = 20s$, $dt = 0.001s$

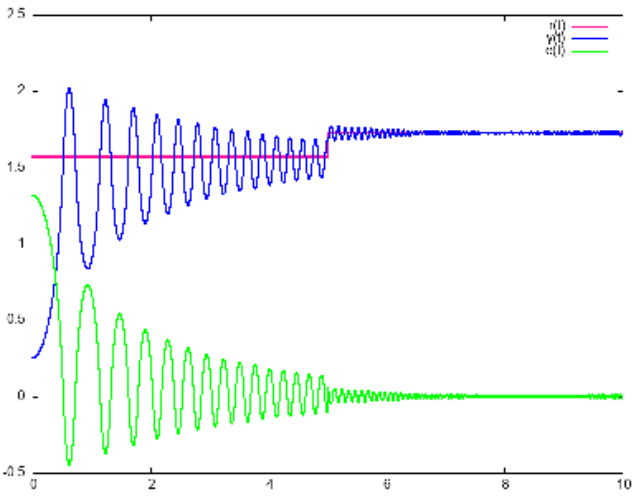


Figura 8. Respuesta a un cambio de referencia

El mejor controlador obtenido fue (6) con un valor de costo de 3.391587147015008 unidades.

$$u_{best} = \frac{f_1(e)}{f_2(e)} \tag{6}$$

Con:

$$f_1(e) = -4.26883 \cdot \tanh\left(\frac{-1.36097 + 0.9872e}{\frac{1.1899e}{3.79551+e}}\right) \tag{7}$$

$$f_2(e) = \frac{-6.62256e^2 - 2.6816}{-1.4386e^3 + 4.8147e^2 - 9.3302e + 31.2263} \tag{8}$$

Como se puede observar en las Figuras 8 y 9. En ambos experimentos, la señal de salida converge hacia la señal de referencia. Mostrando en el primer caso, que puede realizar un seguimiento de la referencia pese a incrementar el valor en un 10 %. En la siguiente figura, se observa la respuesta del sistema bajo una perturbación de tipo impulso unitario, en donde se observa la afectación en la salida pero la recuperación del mismo y la convergencia nuevamente a la señal de referencia. Con lo que se muestra que el controlador obtenido por el algortimo evolutivo cumple con los criterios de búsqueda.

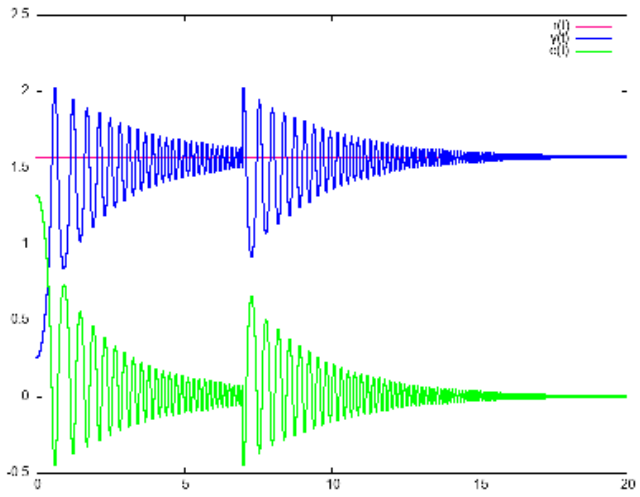


Figura 9. Respuesta con perturbación en la salida a $t = 7.5s$

8. CONCLUSIONES

En este artículo se desarrolló un algoritmo evolutivo con técnicas de programación genética como herramienta para auto sintonizar controladores de forma semi-dirigida. El cual, tiene la capacidad de restringir la morfología de búsqueda para explorar leyes de control por HTSMC bajo un proceso heurísticamente. Para lograr lo anterior, se implementó el algoritmo evolutivo, así como las bibliotecas y bloques de programación que: Codifican un sistema dinámico, leyes de control como estructuras de tipo árbol y su codificación en forma de listas anidadas. Se desarrolló el traductor a archivos tipo ODE para simular con el software XPPAU y agilizar el tiempo de respuesta. Se propusieron variantes a las operaciones genéticas convencionales con el objetivo de evitar la generación espontanea y realizar mutaciones suaves. El EA, cumple con dos objetivos. Permite utilizar un algoritmo heuriscico como método de búsqueda y evaluación de controladores. Por otra parte permite direccionar la búsqueda para orientar hacia un tipo de estrategia de control, sin embargo con pocos ajustes se puede adaptar a diferentes estrategias de control.

Los resultados presentados muestras la convergencia del algoritmo, así como el correcto funcionamiento de las funciones de restricción para la búsqueda de controladores por modos deslizantes. Por tanto, se tiene un algoritmo capaz de generar un conjunto de controladores estáticos que aproximan leyes discontinuos mediante la aproximación con tangentes hiperbólicas, que se van mejorando generación tras generación mediante operaciones genéticas hasta encontrar leyes de control robustas, estabilizante y que permitan el seguimiento de referencias.

El algoritmo, sirven como base para experimentar con sistemas mas complejos y probar nuevas estructuras de control. Lo cual permite usar el programa como una herramienta para la educación e investigación en áreas como programación, control, sistemas dinámicos, etc.

Cada una de los subsistemas que componen al algoritmo son perfectibles. Los métodos de combinación, la función de costos, las restricciones, el simulador de sistemas dinámicos, incluso el tipo de sistemas que se utilizan como planta. Por lo que el programa se puede ampliar y atacar enfoques mas específico por ejemplo el manejo de sistemas con retardo, etc. Esto agregando bibliotecas que complementen el desarrollo realizado.

REFERENCIAS

Alfaro-Cid, E., McGookin, E., Murray-Smith, D., and Fossen, T. (2008). Genetic programming for the automatic design of controllers for a surface ship. *IEEE Transactions on Intelligent Transportation Systems*, 9, 311–321. doi:10.1109/tits.2008.922932.

Ding, Y., Liu, C., Lu, S., and Zhu, Z. (2018). Hyperbolic sliding mode trajectory tracking control of mobile robot. In *Proceedings of the 2018 3rd International Conference on Electrical, Automation and Mechanical Engineering (EAME 2018)*. Atlantis Press. doi:10.2991/eame-18.2018.49.

Duriez, T., Parezanović, V., and Brunton, S.L. (2015). Feedback control of turbulent shear flows by genetic programming. doi:10.48550/ARXIV.1505.01022.

Hui, T., Zeng, W., and Yu, T. (2020). Core power control of the ADS based on genetic algorithm tuning PID controller. *Nuclear Engineering and Design*, 370, 110835. doi:10.1016/j.nucengdes.2020.110835.

Joseph, S.B., Dada, E.G., Abidemi, A., Oyewola, D.O., and Khammas, B.M. (2022). Metaheuristic algorithms for pid controller parameters tuning: review, approaches and open problems. *Heliyon*, 8, e09399. doi: https://doi.org/10.1016/j.heliyon.2022.e09399.

Lazarevic, M.P., Batalov, S.A., and Latinovic, T.S. (2013). Fractional pid controller tuned by genetic algorithms for a three dofs robot system driven by dc motors. *IFAC Proceedings Volumes*, 46, 385–390. doi: 10.3182/20130204-3-fr-4032.00179.

Prieto, P., Cazarez-Castro, N., Aguilar, L., and Espinoza, D. (2015). Estudio de métodos de reducción de chattering en modos deslizantes.

Samakwong, T. and Assawinchaichote, W. (2016). Pid controller design for electro-hydraulic servo valve system with genetic algorithm. *Procedia Computer Science*, 86, 91–94. doi: https://doi.org/10.1016/j.procs.2016.05.023.

Slowik, A. and Kwasnicka, H. (2020). Evolutionary algorithms and their applications to engineering problems. *Neural Computing and Applications*, 12363–12379. doi: 10.1007/s00521-020-04832-8.

Utkin, V.I. and Chang, H.C. (2002). Sliding mode control on electro-mechanical systems. *Mathematical Problems in Engineering*, 8(4-5), 451–473. doi: 10.1080/10241230306724.