Jornadas de Automática

Optimización inteligente de trayectorias en manipuladores industriales mediante algoritmo PSO

Peñacoba Yagüe, Mario[a,*], Sierra García, Jesús Enrique[a], Santos Peñas, Matilde[b]

[a] *Departamento de Digitalización, Universidad de Burgos, Campus Río Vena, Avda. Cantabria, s/n, 09006 Burgos, España.*
[b] *Instituto de Tecnología del Conocimiento, Universidad Complutense de Madrid, 28040-Madrid, España.*

To cite this article: Peñacoba Yagüe, Mario, Sierra García, Jesús Enrique, Santos Peñas, Matilde. 2025. Intelligent optimization of trajectories in industrial manipulators using PSO algorithm: first approach. Jornadas de Automática, 46. https://doi.org/10.17979/ja-cea.2025.46.12051

Resumen

Este trabajo presenta un primer enfoque sobre una metodología inteligente para la optimización de trayectorias en manipuladores industriales, basada en el algoritmo metaheurístico Particle Swarm Optimization (PSO). La evaluación se realiza en un entorno de simulación desarrollado en MATLAB que incorpora un modelo dinámico del robot, incluyendo masa, volumen e inercia de los eslabones, así como la presencia de obstáculos y detección de colisiones. Se proponen y comparan tres funciones de coste que consideran distintas métricas físicas: el tiempo total de trayectoria, el esfuerzo articular, estimado mediante los pares medios y máximos aplicados, y una combinación ponderada de ambos criterios. El proceso de optimización sigue una estrategia escalonada, que primero garantiza la viabilidad geométrica mediante la eliminación de trayectorias con colisiones, y posteriormente refina el resultado optimizando las métricas físicas. Los resultados evidencian que la técnica de optimización propuesta permite obtener trayectorias más eficientes y realistas, con mayor potencial de implementación en entornos exigentes.

Palabras clave: Robótica industrial, Optimización de trayectorias, Algoritmos metaheurísticos, Planificación de movimiento, Control cinemático.

Intelligent trajectory optimization in industrial manipulators by PSO algorithm

Abstract

This work presents an initial approach toward an intelligent methodology for trajectory optimization in industrial manipulators, based on the metaheuristic algorithm Particle Swarm Optimization (PSO). The evaluation is carried out in a MATLAB simulation environment that incorporates a dynamic model of the robot, including mass, volume, and inertia of the links, as well as obstacle representation and collision detection. Three cost functions are proposed and compared, each considering different physical performance metrics: (i) total trajectory time, (ii) joint effort, estimated through the average and maximum torques applied, and (iii) a weighted combination of both criteria. The optimization follows a staged strategy: it first ensures geometric feasibility by eliminating trajectories with collisions, and then refines the solution by optimizing the physical metrics. The results show that the proposed optimization technique leads to more efficient and realistic trajectories, with higher potential for deployment in demanding industrial environments.

Keywords: Industrial Robotics, Trajectory Optimization, Metaheuristic algorithms, Motion planning, Kinematic control.

1. Introducción

En la automatización industrial, la planificación de trayectorias juega un papel crucial para garantizar la eficiencia, seguridad y sostenibilidad de los procesos que involucran manipuladores robóticos. Tradicionalmente, la generación de trayectorias ha estado dominada por métodos geométricos o cinemáticos que, si bien aseguran el cumplimiento de ciertas restricciones espaciales, descuidan en muchos casos aspectos físicos relevantes como el esfuerzo

*Mario Peñacoba Yagüe: mpenacoba@ubu.es

requerido por las articulaciones del robot o el consumo energético asociado al movimiento.

En este contexto, los algoritmos de optimización inspirados en la inteligencia colectiva, como el Particle Swarm Optimization (PSO), ofrecen una alternativa prometedora para abordar el problema de forma más global e inteligente (Yiyang et al, 2021). Su capacidad para explorar espacios de búsqueda complejos y adaptarse a distintos criterios de evaluación permite diseñar trayectorias más realistas y mejor adaptadas a las limitaciones físicas del sistema robótico (Sadegian et al, 2021), (Nonoyama et al, 2022).

Este trabajo presenta una metodología de optimización de trayectorias basada en un algoritmo de optimización PSO que tiene en cuenta tanto la viabilidad geométrica como el rendimiento físico del robot. Para ello, se desarrolla un entorno de simulación en MATLAB que integra el modelo dinámico del manipulador, incluyendo masa, volumen e inercia de sus eslabones, así como la presencia de obstáculos con los que podría colisionar (Grosz and Borzan, 2025). Sobre este entorno se definen y comparan tres funciones de coste distintas, centradas en:

I. Minimizar el tiempo total de ejecución de la trayectoria.
II. Reducir el esfuerzo articular mediante la evaluación de los pares medios y máximos.
III. Reducir tanto el esfuerzo articular como el tiempo total de ejecución de la trayectoria mediante una combinación ponderada de ambas métricas.

Además, se plantea una estrategia escalonada en un primer paso, el algoritmo descarta soluciones que implican colisiones y, posteriormente optimiza las trayectorias válidas conforme a los criterios físicos establecidos. Los resultados obtenidos muestran que este enfoque no solo mejora la eficiencia y realismo de las trayectorias generadas, sino que también incrementa su aplicabilidad en entornos exigentes donde las limitaciones dinámicas y de seguridad son críticas (Peñacoba et al, 2023). Este estudio se enmarca dentro del proyecto europeo MANiBOT, cuyo objetivo es desarrollar soluciones avanzadas de manipulación bimanual colaborativa en escenarios industriales y logísticos, como aeropuertos y supermercados (MANiBOT, 2023).

Este artículo se organiza como sigue: en la Sección 2 se describe la técnica de optimización empleada. La Sección 3 define el problema de optimización y los criterios considerados. La Sección 4 presenta las distintas funciones de coste que se han propuesto. En la Sección 5 se presenta el modelo del sistema robótico y su entorno. La Sección 6 expone y analiza los resultados obtenidos. Finalmente, en la Sección 7 se extraen las principales conclusiones del trabajo y se proponen posibles líneas de investigación futuras.

2. Técnica de Optimización

En este trabajo, el proceso de optimización de trayectorias del sistema robótico se basa en el algoritmo metaheurístico Particle Swarm Optimization (PSO), una técnica de optimización poblacional inspirada en el comportamiento colectivo de sistemas naturales, como las bandadas de aves o los bancos de peces. En este enfoque, cada partícula representa una solución candidata en el espacio de búsqueda y adapta su posición en función de su mejor experiencia personal y de la mejor solución conocida por todo el enjambre (Shami et al 2022).

La evolución de cada partícula se rige por la actualización iterativa de su posición y velocidad. La posición en el instante $t+1$ se obtiene mediante la ecuación (1).

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{1}$$

Donde v_i^{t+1} es la nueva velocidad de la partícula, calculada con la ecuación 2.

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (p_i^t - x_i^t) + c_2 r_2 (g^t - x_i^t) \tag{2}$$

En esta expresión, ω es el peso de inercia, que controla la influencia de la velocidad previa sobre el nuevo movimiento, c_1 y c_2 son los coeficientes de atracción cognitiva y social**,** respectivamente; r_1 y r_2 son números aleatorios en el intervalo [0,1], que introducen variabilidad estocástica y fomentan la exploración del espacio de soluciones; p_i^t representa la mejor posición alcanzada por la partícula i hasta el instante t, y g^t es la mejor posición encontrada por todo el enjambre (Peñacoba et al, 2024), (Liu et al, 2021).

Este mecanismo logra un equilibrio eficaz entre exploración global (mediante el componente aleatorio y la dispersión inicial del enjambre) y explotación local (convergencia hacia soluciones óptimas conocidas), lo que permite abordar con éxito problemas complejos de planificación de trayectorias en entornos con restricciones geométricas, cinemáticas o dinámicas (Priyadarshi and Kumar, 2025), (Ekrem and Aksoy, 2023).

3. Definición del problema de optimización

El presente trabajo aborda el problema de generación automática de trayectorias para un brazo manipulador industrial en un entorno tridimensional con obstáculos. El objetivo principal es que el robot sea capaz de aprender a evitar colisiones mediante la optimización de trayectorias en coordenadas cartesianas, garantizando en primer lugar la viabilidad geométrica, y en segundo, la eficiencia física del movimiento.

Además, la optimización de la trayectoria se realiza en presencia de elementos fijos que actúan como restricciones espaciales (Figura 1).

Figura 1: Representación del entorno de simulación: obstáculos y robot CRB15000.

3.1. Variables de decisión del problema.

La trayectoria a generar está compuesta por cuatro puntos en el espacio tridimensional $[x_i, y_i, z_i]$. De estos, los puntos iniciales $[x_0, y_0, z_0]$ y finales $[x_n, y_n, z_n]$. son fijos y corresponden a tareas definidas (por ejemplo, recogida y colocación de un objeto). El resto de puntos se consideran libres y su posición se ajusta mediante el algoritmo de optimización. Estos puntos libres constituyen las variables de decisión del problema; representar los parámetros sobre los que actúa el algoritmo para mejorar la trayectoria generada (Rascón et al, 2024).

En concreto, cada solución candidata codificada por una partícula del algoritmo se define mediante las coordenadas cartesianas de los puntos de control libres, lo que configura un espacio de búsqueda de tres dimensiones. En este trabajo, al ser una primera aproximación a esta metodología, no se han comprendido los ángulos cartesianos.

3.2. Control del manipulador: cinemática inversa

Dado que el control del manipulador se formula en el espacio cartesiano, resulta necesario convertir cada trayectoria candidata —definida mediante posiciones sucesivas del extremo del efector— en configuraciones articulares equivalentes. Este proceso se lleva a cabo mediante la resolución de la cinemática inversa, que consiste en determinar los valores de las variables articulares que permiten alcanzar una posición y orientación concretas del extremo del robot (Peñacoba et al, 2025).

Aunque en la mayoría de arquitecturas robóticas no se dispone de una solución analítica general, el problema de la cinemática inversa puede expresarse conceptualmente como un conjunto de funciones no lineales que relacionan la posición y orientación cartesianas $[x, y, z, \phi, \theta, \psi]$ con las variables articulares $[q_1, q_2, \dots q_n]$ siendo n el número de grados de libertad del manipulador (Abdor-Sierra et al, 2022). Para un brazo de un total de 6 grados de libertad, se puede expresar el cálculo de la cinemática inversa de forma conceptual según las ecuaciones (3-8).

$$q_1 = g_1(x, y, z, \phi, \theta, \psi) \quad (3)$$

$$q_2 = g_2(x, y, z, \phi, \theta, \psi) \quad (4)$$

$$q_3 = g_2(x, y, z, \phi, \theta, \psi) \quad (5)$$

$$q_4 = g_2(x, y, z, \phi, \theta, \psi) \quad (6)$$

$$q_5 = g_2(x, y, z, \phi, \theta, \psi) \quad (7)$$

$$q_6 = g_4(x, y, z, \phi, \theta, \psi \quad (8)$$

La cinemática inversa es un problema no trivial, ya que puede presentar múltiples soluciones, soluciones inexistentes o configuraciones que resultan indeseables desde el punto de vista del control (Singh et al, 2021). En particular, uno de los principales desafíos es la aparición de puntos singulares, situaciones en las que el Jacobiano del robot pierde rango y se produce una pérdida de grados de libertad efectiva, generando movimientos no controlables o incrementos desproporcionados en los pares requeridos (Lu et al 2022), (Yime et al, 2023).

4. Función de coste

La evaluación de cada trayectoria candidata se realiza mediante funciones de coste diseñadas para reflejar distintos criterios físicos y operativos relevantes en el contexto industrial. En este trabajo se han definido tres formulaciones independientes, cada una centrada en un objetivo de optimización específico: (i) minimizar el tiempo total de ejecución, (ii) minimizar el esfuerzo articular, y (iii) alcanzar un equilibrio entre ambos criterios.

En todos los casos, el planteamiento de la función de coste sigue una estrategia escalonada: inicialmente se descartan las trayectorias que presentan colisiones, y posteriormente se optimizan las soluciones geométricamente viables. Es decir, todas las funciones de coste comparten una fase inicial de validación geométrica, en la que se penalizan aquellas trayectorias que colisionan con el entorno o consigo mismas. Esta penalización se implementa mediante un término aditivo proporcional al tiempo total en colisión ($t_{colisión}$), tal como se muestra en la ecuación (9).

$$f_c = 1 + t_{colisión} \quad (9)$$

Este planteamiento permite descartar tempranamente soluciones que no respetan las restricciones espaciales del entorno de trabajo. Como resultado, todas las trayectorias con colisión presentan valores de $f_c > 1$, siendo penalizadas proporcionalmente a su grado de invasión del espacio ocupado. Esta fase es común a todas las formulaciones descritas a continuación, que se aplican únicamente a las trayectorias viables.

4.1. Optimización basada en el tiempo de ejecución de la trayectoria.

La primera formulación está diseñada para favorecer trayectorias que minimicen el tiempo total de ejecución. Una vez descartadas las trayectorias con colisiones, se evalúa el tiempo total necesario para recorrer la trayectoria, normalizado respecto a un umbral máximo $t_{Máx}$. Esta normalización asegura que la función de coste se mantenga en un rango de valores $f_c < 1$, pues $t_{Máx}$ se corresponde con el tiempo máximo de ejecución de la trayectoria comprendida entre el punto inicial y el punto final. La función de coste en esta segunda fase viene definida por la ecuación 10.

$$f_c = \frac{t_{trayectoria}}{t_{Máx}} \quad (10)$$

Esta formulación permite una comparación coherente entre diferentes trayectorias desde el punto de vista de su eficiencia temporal, favoreciendo aquellas que alcanzan el objetivo en el menor tiempo posible sin comprometer la integridad física del entorno.

4.2. Optimización basada en el esfuerzo articular.

La segunda función de coste busca reducir el esfuerzo mecánico requerido por el manipulador. Para trayectorias

viables, se consideran dos métricas: el par medio $\bar{T}$ y el par máximo $T_{Máx}$ aplicados en las articulaciones a lo largo del movimiento. Ambas se normalizan respecto a un umbral de seguridad T_{Lim}, de tal manera que $T_{Lim} > max(\bar{T}, T_{Máx})$

La función resultante aplica un promedio ponderado equitativo entre ambas métricas normalizadas (11).

$$f_c = \alpha * \frac{\bar{T}}{T_{Lim}} + \beta * \frac{T_{Máx}}{T_{Lim}} \quad (11)$$

En este trabajo se ha escogido $\alpha = \beta = 0{,}5$. Esta formulación favorece trayectorias que reducen el desgaste mecánico de los actuadores, mejoran la eficiencia energética y evitan operar cerca de los límites estructurales del sistema.

4.3. Optimización híbrida tiempo de ejecución – esfuerzo articular.

La tercera función de coste implementa una estrategia híbrida que combina criterios temporales y de esfuerzo mecánico con el objetivo de obtener trayectorias físicamente eficientes y rápidas, sin, como en las anteriores, comprometer la viabilidad geométrica. Para trayectorias viables, se calcula una función de coste compuesta que pondera tanto el término temporal como el esfuerzo articular (12):

$$f_c = \alpha \left(0{,}5 * \frac{\bar{T}}{T_{Lim}} + 0{,}5 * \frac{T_{Máx}}{T_{Lim}} \right) + \beta * \frac{t_{trayectoria}}{t_{Máx}} \quad (12)$$

Donde α y β son los factores de ponderación empleados para equilibrar ambos términos. En este caso, como en el anterior, se ha escogido $\alpha = \beta = 0{,}5$.

Esta formulación promueve soluciones de compromiso entre trayectorias rápidas y de bajo esfuerzo, permitiendo que el proceso de optimización explore soluciones con un equilibrio funcional adecuado para aplicaciones exigentes.

5. **Modelado del sistema**

El entorno de simulación se ha desarrollado íntegramente en MATLAB 2024b, aprovechando el soporte nativo para la importación de modelos URDF, la cinemática inversa y el análisis dinámico de robots mediante la "Robotics System Toolbox". El objetivo del entorno es evaluar trayectorias generadas por algoritmos de optimización bajo criterios físicos y geométricos, incluyendo detección de colisiones, análisis del esfuerzo articular y cálculo de métricas temporales.

El modelo del robot utilizado corresponde al ABB GOFA CRB 15000-95, importado desde un archivo URDF enriquecido con información dinámica de los eslabones, con el fin de garantizar un análisis dinámico realista. Véase el robot y el entorno representados gráficamente en la Figura 1.

4.4. Interpolación de las trayectorias.

Las trayectorias se interpolan en el espacio cartesiano utilizando un interpolador de tipo quintico, lo cual garantiza la continuidad de posición, velocidad y aceleración. Esta suavidad es esencial para asegurar que las trayectorias puedan ser seguidas de forma realista por el manipulador sin provocar cambios bruscos en los actuadores. El polinomio viene descrito en la ecuación 13.

$$p(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \quad (13)$$

Donde t es el tiempo y los coeficientes $a_0, \dots, a_5$ se calculan a partir de las condiciones de posición, velocidad y aceleración en los puntos inicial y final del segmento. Esto permite generar una trayectoria suave entre cada par de puntos consecutivos, manteniendo la coherencia temporal a lo largo de toda la trayectoria.

Esta formulación garantiza que el manipulador pueda seguir las consignas generadas sin generar discontinuidades cinemáticas, facilitando una evaluación precisa del comportamiento dinámico y una ejecución más robusta en un entorno físico real.

4.5. Control cinemático del robot

A lo largo de la trayectoria interpolada, se resuelve la cinemática inversa en cada instante mediante la función "*inverseKinematics()*". Para cada objetivo cartesiano, se generan múltiples soluciones utilizando semillas aleatorias cercanas a la configuración anterior. De entre todas las soluciones obtenidas, se selecciona la que minimiza la distancia euclídea en el espacio articular respecto a la configuración anterior. Esta estrategia busca garantizar la continuidad cinemática del movimiento, evitando saltos innecesarios entre configuraciones muy distintas que podrían generar trayectorias no realistas o difíciles de seguir para el manipulador.

Esta aproximación incremental basada en continuidad local resulta especialmente útil para minimizar los efectos de ambigüedad inherentes a la cinemática inversa, mejorar la estabilidad del movimiento y reducir la probabilidad de alcanzar configuraciones cercanas a singularidades.

La solución finalmente seleccionada es aquella cuya distancia en el espacio articular respecto a la configuración previa es mínima, con el objetivo de mantener la continuidad y estabilidad en el movimiento del robot.

4.6. Detección de colisiones

La detección de colisiones se realiza mediante la función "*checkCollision*()", que evalúa si una configuración articular del robot interseca con algún objeto del entorno definido o consigo mismo. Esta función opera sobre el modelo cinemático del robot y un conjunto de obstáculos definidos en el escenario simulado.

El entorno incluye obstáculos fijos generados previamente en función del escenario. Además, se consideran explícitamente las colisiones internas, es decir, posibles auto colisiones entre los diferentes eslabones del robot. Esto garantiza una evaluación más rigurosa de la viabilidad geométrica de las trayectorias, evitando configuraciones físicamente inviables.

La comprobación se aplica punto a punto a lo largo de toda la trayectoria, permitiendo detectar de forma precisa cualquier violación del espacio libre, ya sea con el entorno externo o entre partes del propio manipulador. Esta información se utiliza posteriormente para penalizar las soluciones inválidas dentro del proceso de optimización.

4.7. *Cálculo de los pares*

De forma paralela al análisis geométrico, se evalúa el esfuerzo articular requerido para ejecutar cada trayectoria utilizando la función "*inverseDynamics()*". Esta función permite calcular el vector de torques necesario considerando las velocidades y aceleraciones articulares, estimadas por diferencias finitas entre configuraciones consecutivas. A partir de esta información, se obtienen dos métricas fundamentales: el par medio ($\bar{T}$) y el par máximo ($T_{Máx}$) aplicado en cada articulación durante toda la trayectoria.

6. **Resultados**

Los resultados obtenidos evidencian el impacto de cada función de coste sobre la forma y calidad de las trayectorias generadas. En primer lugar, la trayectoria inicial, mostrada en línea discontinua negra en la Figura 2, no es viable, ya que colisiona con el obstáculo (mesa), tal como se recoge en la Tabla 1. Por el contrario, las tres trayectorias optimizadas (FC1, FC2 y FC3) logran evitar la colisión con éxito, adaptando su geometría al entorno (Figura 2).

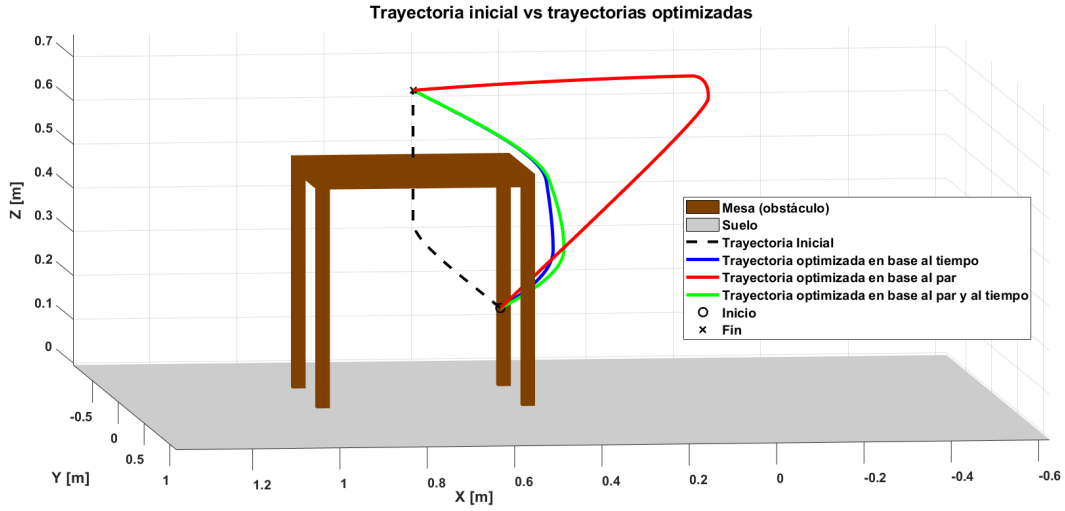

Figura 2: Comparación entre trayectoria inicial y las trayectorias optimizadas en 3D.

La Tabla 1 resume las métricas obtenidas para cada trayectoria, evaluando el tiempo total de ejecución, el par medio, el par máximo medio y la existencia de colisiones.

Tabla 1: Métricas

Trayectoria	Tiempo	Par Medio	Par Máximo Medio	Colisión
Inicial	5,87	-	-	Sí
FC1	7,22	38,77	106,63	No
FC2	14,50	21,76	64,74	No
FC3	7,47	38,20	105,7	No

De estos resultados se puede apreciar que la trayectoria FC1, optimizada con respecto al tiempo, consigue la menor duración (7,22 s), pero presenta un esfuerzo articular elevado, con un par máximo medio de 106,63 Nm. Esta solución puede resultar adecuada en tareas que prioricen la rapidez de ejecución, aunque penaliza el consumo energético y el desgaste mecánico.

Por otro lado, la trayectoria FC2, optimizada en función del par, consigue reducir notablemente el esfuerzo aplicado, con un par medio de 21,76 Nm y un par máximo medio de tan solo 64,74 Nm. No obstante, esta suavidad se logra a costa de una mayor duración del movimiento (14,50 s), más del doble que en FC1. En la Figura 2 puede observarse que esta trayectoria (en rojo) adopta un recorrido amplio y alejado del obstáculo.

La trayectoria FC3, por último, representa un compromiso entre ambas métricas, combinando tiempo y par en su función de coste. Esta solución alcanza un tiempo total de 7,47 s, muy próximo al de FC1, con un par medio de 38,20 Nm y un par máximo medio de 105,70 Nm, similares también a los de la trayectoria más rápida. Visualmente (Figura 2, curva verde), su forma se posiciona entre la agresividad de FC1 y la suavidad de FC2, logrando un equilibrio eficaz entre seguridad, rapidez y eficiencia física.

Finalmente, la Figura 3 presenta la evolución del par máximo articular por instante de tiempo para cada trayectoria.

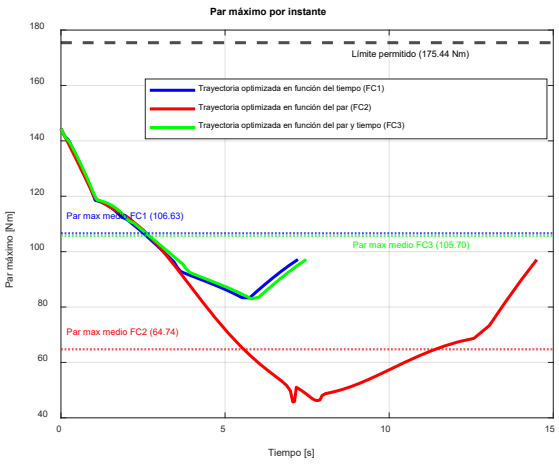

Figura 3: Pares máximos.

En esta gráfica se observa cómo la solución FC2 (en rojo) mantiene valores consistentemente bajos a lo largo de todo el movimiento, mientras que FC1 y FC3 (azul y verde, respectivamente) muestran picos más elevados. En todos los casos, se respeta el límite permitido de par, establecido en 175,44 Nm (línea discontinua negra), garantizando la viabilidad dinámica de las trayectorias.

7. Conclusiones y trabajos futuros

Los resultados obtenidos en este estudio demuestran la eficacia del enfoque propuesto para la generación automática de trayectorias en manipuladores industriales mediante optimización basada en Particle Swarm Optimization (PSO). El sistema ha sido capaz de evitar colisiones con obstáculos presentes en el entorno, ajustando dinámicamente la forma de la trayectoria en función de diferentes objetivos de optimización.

Se han evaluado tres funciones de coste distintas, centradas en minimizar el tiempo de ejecución, el esfuerzo articular y una combinación de ambas. Los resultados permiten extraer las siguientes conclusiones principales:

La optimización basada en el tiempo (FC1) produce trayectorias rápidas y libres de colisiones, pero a costa de elevados valores de par articular. Esta estrategia es adecuada para aplicaciones donde la velocidad del ciclo de trabajo sea prioritaria, aunque puede generar un mayor desgaste mecánico a largo plazo.

Por el contrario, la optimización centrada en el par articular (FC2) genera trayectorias más suaves y eficientes desde el punto de vista energético, reduciendo significativamente el esfuerzo requerido. No obstante, esta ganancia se traduce en trayectorias más lentas y de mayor longitud, lo que podría no ser viable en entornos con restricciones de tiempo.

La función de coste combinada (FC3) ha demostrado ser la opción más equilibrada, logrando un compromiso adecuado entre velocidad y esfuerzo. Esta trayectoria presenta tiempos similares a FC1, pero con menores niveles de par, lo que la convierte en una solución robusta.

Además, se ha verificado que todas las trayectorias optimizadas respetan el límite de par permitido por el fabricante, lo cual garantiza la viabilidad física de su ejecución.

Como línea de trabajo futura, se propone profundizar en la optimización individualizada del par articular, abordando cada articulación del manipulador por separado. El objetivo sería minimizar el estrés mecánico no solo en promedio, sino asegurando que todas las articulaciones trabajen de forma sostenida lejos de su límite máximo de par. Esta estrategia permitiría una distribución más equilibrada del esfuerzo entre motores, lo que contribuiría a mejorar la fiabilidad operativa del sistema y alargar la vida útil del robot. Asimismo, se prevé la integración de nuevos criterios de optimización, como el consumo energético acumulado o la sensibilidad frente a perturbaciones externas, así como la extensión del sistema a entornos dinámicos con obstáculos en movimiento.

Agradecimientos

Este trabajo ha sido realizado parcialmente gracias al apoyo de la Comisión Europea en el marco del proyecto europeo MANiBOT, número de referencia 101120823.

Referencias

Abdor-Sierra, J. A., Merchán-Cruz, E. A., & Rodríguez-Cañizo, R. G. (2022). A comparative analysis of metaheuristic algorithms for solving the inverse kinematics of robot manipulators. Results in Engineering, 16, 100597.

Ekrem, Ö., & Aksoy, B. (2023). Trajectory planning for a 6-axis robotic arm with particle swarm optimization algorithm. Engineering Applications of Artificial Intelligence, 122, 106099.

Grosz, E. A., & Borzan, M. (2025). Modular Robotics Configurator: A MATLAB Model-Based Development Approach. Applied System Innovation, 8(1), 21.

Liu, F., Huang, H., Li, B., & Xi, F. (2021). A parallel learning particle swarm optimizer for inverse kinematics of robotic manipulator. International Journal of Intelligent Systems, 36(10), 6101-6132.

Lu, J., Zou, T., & Jiang, X. (2022). A neural network based approach to inverse kinematics problem for general six-axis robots. Sensors, 22(22), 8909.

MANiBOT. (2023). *MANiBOT project*. https://manibot-project.eu/

Nonoyama, K., Liu, Z., Fujiwara, T., Alam, M. M., & Nishi, T. (2022). Energy-efficient robot configuration and motion planning using genetic algorithm and particle swarm optimization. Energies, 15(6), 2074.

Peñacoba, M., Bayona, E., Sierra-García, J. E., & Santos, M. (2024). Route Optimization for UVC Disinfection Robot Using Bio-Inspired Metaheuristic Techniques. Biomimetics, 9(12), 744.Able, B. C., 1956.

Peñacoba, M., Sierra-García, J. E., Santos, M., & Mariolis, I. (2023). Path Optimization Using Metaheuristic Techniques for a Surveillance Robot. Applied Sciences, 13(20), 11182

Peñacoba-Yagüe, M., Sierra-García, J.E., Santos-Peñas, M., Ruano, A. (2025). Enhancing Robotic Control Efficiency with MLP-Based Inverse Kinematics: First Approach. In: Aguiar, A.P., Rocha Malonek, P., Pinto, V.H., Fontes, F.A.C.C., Chertovskih, R. (eds) CONTROLO 2024. CONTROLO 2024. Lecture Notes in Electrical Engineering, vol 1325. Springer, Cham. https://doi.org/10.1007/978-3-031-81724-3_51

Priyadarshi, R., & Kumar, R. R. (2025). Evolution of Swarm Intelligence: A Systematic Review of Particle Swarm and Ant Colony Optimization Approaches in Modern Research. Archives of Computational Methods in Engineering, 1-42.

Rascón, R., Flores-Mendoza, A., Moreno-Valenzuela, J., & Aguilar-Avelar, C. (2024). Control para seguimiento de trayectorias cartesianas en robots manipuladores. Revista Iberoamericana de Automática e Informática industrial, 21(3), 252-261.

Sadeghian, Z., Akbari, E., Nematzadeh, H., & Motameni, H. (2025). A review of feature selection methods based on meta-heuristic algorithms. Journal of Experimental & Theoretical Artificial Intelligence, 37(1), 1-51.

Shami, T. M., El-Saleh, A. A., Alswaitti, M., Al-Tashi, Q., Summakieh, M. A., & Mirjalili, S. (2022). Particle swarm optimization: A comprehensive survey. Ieee Access, 10, 10031-10061.

Singh, R., Kukshal, V., & Yadav, V. S. (2021). A review on forward and inverse kinematics of classical serial manipulators. Advances in Engineering Design: Select Proceedings of ICOIED 2020, 417-428.

Yime, E., Saltarén, R. J., & Mckinley, J. A. R. (2023). Análisis dinámico inverso de robots paralelos: Un tutorial con álgebra de Lie. Revista Iberoamericana de Automática e Informática Industrial, 20(4), 327-346.

Yiyang, L., Xi, J., Hongfei, B., Zhining, W., & Liangliang, S. (2021). A general robot inverse kinematics solution method based on improved PSO algorithm. Ieee Access, 9, 32341-32350.