ISBN: 978-628-96613-1-6. ISSN: 2414-6390. Digital Object Identifier: https://dx.doi.org/10.18687/LACCEI2025.1.1.2244

Trajectory planning of a 5 DOF collaborative robot using numerical methods with quaternions

Cesar Ciriaco-Martinez, M. Sc[1]; Bruno Trujillo-Miranda, B.Sc[1]; Jhorkaef Pena-Chavez, B.Sc[1]
[1] Universidad Tecnológica del Perú, Perú, C14110@utp.edu.pe, 1410705@utp.edu.pe, U19207387@utp.edu.pe

***Abstract*– This work presents a quaternion-based methodology for trajectory control of a 5-degree-of-freedom (DOF) collaborative robot. The mathematical representation of quaternions and their application in trajectory interpolation are addressed, avoiding issues such as singularities and discontinuities found in other representations like Euler angles and rotation matrices. The interpolation method with a 4-1-4 parabolic blend is implemented, prioritizing position and velocity continuity while maintaining minimal acceleration variation. The method is validated through simulations in MATLAB/Simulink, comparing the results obtained using homogeneous transformation matrices and quaternions. The results show that using quaternions reduces trajectory tracking errors and optimizes the system's computational performance. Finally, the numerical computation results of both methods are discussed.**

***Keywords*-- Quaternions, MATLAB/Simulink, Trajectory Interpolation, Parabolic Blend, Computational Optimization.**

Planificación de trayectorias de un robot colaborativo de 5 DOF haciendo uso de métodos numéricos con cuaternios

Cesar Ciriaco-Martinez, M. Sc[1]; Bruno Trujillo-Miranda, B.Sc[1]; Jhorkaef Pena-Chavez, B.Sc[1]
[1] Universidad Tecnológica del Perú, Perú, C14110@utp.edu.pe, 1410705@utp.edu.pe, U19207387@utp.edu.pe

Resumen*– En este trabajo se presenta una metodología basada en cuaternios para el control de trayectorias de un robot colaborativo de 5 grados de libertad (DOF). Se aborda la representación matemática de los cuaternios y su aplicación en la interpolación de trayectorias, evitando problemas como singularidades y discontinuidades que presentan otras representaciones como los ángulos de Euler y las matrices de rotación. Se implementa el método de interpolación con enlace parabólico 4-1-4 donde prioriza la continuidad de la posición y la velocidad, pero con una mínima variación en la aceleración. La validación del método se realiza mediante simulaciones en MATLAB/Simulink, donde se comparan los resultados obtenidos con matrices de transformación homogéneas y cuaternios. Los resultados muestran que el uso de cuaternios permite reducir el error en el seguimiento de trayectorias y optimizar el desempeño computacional del sistema. Finalmente, se discuten los valores resultantes de los dos métodos de cálculos numéricos.

***Palabras clave:** Cuaternios, MATLAB/Simulink, Interpolación de trayectorias, Enlace parabólico, Optimización computacional.*

I. Introducción

En el ámbito de la robótica colaborativa, la planificación de trayectorias es un desafío clave para garantizar que los robots puedan moverse de manera precisa, eficiente y segura en entornos compartidos. Dado que los robots colaborativos (cobots) están diseñados para interactuar y colaborar con los humanos de manera directa, se necesitan métodos novedosos implementados para la planificación de trayectorias que proporcionarían formas de controlar los movimientos de los robots de manera precisa y, al mismo tiempo, evitar colisiones, maximizar la eficiencia operativa y minimizar el tiempo y consumo energético[1]. Asimismo, a efectos de garantizar la eficiencia operativa máxima de los robots industriales, la elección de los métodos y los enfoques para colisiones, enfoques de rapidez y optimización del movimiento de las articulaciones para alcanzar el punto designado con una trayectoria óptima puede ser diferente[2]. Uno de los enfoques más prometedores para la representación y el control de los movimientos de robots con varios grados de libertad es el uso de cuaternios, se utiliza usa para representar rotaciones ha demostrado ser una solución eficaz para la descripción de orientaciones en sistemas robóticos, ya que evita problemas y proporciona una forma más eficiente de manejar las interpolaciones entre configuraciones de rotación. Esta propiedad es especialmente relevante cuando se busca optimizar el cálculo de trayectorias en robots colaborativos, donde los movimientos deben ser no solo precisos, sino también rápidos en tiempo de respuesta y energéticamente eficientes[3]. Así mismo, es una extensión matemática de los números complejos que proporciona una forma eficiente y robusta de modelar y manipular rotaciones en el espacio tridimensional, dado que los cuaternios evitan las singularidades que pueden surgir con otras representaciones de rotaciones, como las matrices de rotación o los ángulos de Euler, han ganado relevancia en la planificación de trayectorias de robots con alta complejidad de movimiento. Es crucial para maximizar la eficiencia operativa, reducir el impacto ambiental de las operaciones industriales y lograr una optimización de trayectoria, en términos de tiempo y consumo de energía [4].

Uno de los aspectos clave en la planificación de trayectorias, análisis de cinemática y dinámica de los robots de múltiples grados de libertad (DOF) es la representación adecuada de las rotaciones, traslaciones y posicionamiento de las articulaciones. Tradicionalmente, las rotaciones en el espacio tridimensional se han modelado utilizando ángulos de Euler, matrices homogéneas. Sin embargo, las primeras presentan desventajas como las singularidades (conocidas como "bloqueos de gimbal"), mientras que las matrices homogéneas que utilizan matrices de rotación, aunque son más estables, son computacionalmente costosas y no son tan intuitivas debido a que usan una gran cantidad de recursos para realizar el cálculo inicial del robot. En este contexto, los cuaternios se destacan como una herramienta eficiente para representar rotaciones en tres dimensiones, ya que no sufren de singularidades y tienen un coste computacional reducido, lo que los hace adecuados para robots con un número elevado de grados de libertad (DOF)[5].

El uso de cuaternios en la robótica ha sido ampliamente explorado en los últimos años. Estos proporcionan una forma compacta y robusta de representar rotaciones y permiten una interpolación fluida entre posiciones y orientaciones. Además, su capacidad para evitar los problemas de gimbal lock hace que sean una opción atractiva para la planificación de trayectorias en robots con rotaciones complejas, como los robots colaborativos. Diversos estudios han demostrado cómo los cuaternios pueden ser utilizados para representar y planificar trayectorias en robots, avanzando en el análisis de las trayectorias y de la cinemática de distintos tipos de robots

manipuladores individuales, de doble brazo o robots colaborativos para profundizar en el estudio de este método, con el objetivo de ahorrar tiempo de cálculo a los controladores de estos robots[6], [7].

La planificación de trayectorias en robots colaborativos presenta varios desafíos, debido a la necesidad de que los robots trabajen en espacios compartidos con humanos. En particular, la precisión y la seguridad son factores determinantes en estos entornos, por lo tanto es importante que el tiempo de cálculo se minimice para mejorar un tiempo de respuesta en los robots[8]. La literatura existente sobre la planificación de trayectorias de robots se enfoca en diversos métodos, como algoritmos de optimización, métodos de interpolación, y enfoques basados en aprendizaje automático.

En esta documentación se encuentran métodos como la aplicación de redes neuronales artificiales, en estos estudios se mencionan como se incluyeron redes neuronales artificiales en el cual se utilizaron la red de función de base radial en los 2 casos se utilizaron las redes neuronales para exclusivamente la planificación y seguimiento de la trayectoria programada, se utiliza este método exclusivamente para el control de la salida de cada grado de libertad y su posición, su optimización se basa en que el robot mantenga la trayectoria establecida con seguridad[9], [10].

Como otros de los métodos a destacar son aquellos que están orientados para que los robot cooperativos o aquellos con 2 brazos se mantengan sin colisiones, o que logre evitar los obstáculos para no dañar a un operador físico que realiza procesos de manufactura o aquellos manipuladores de los robots, aquí se destaca que el objetivo es la prevención de colisiones y respuestas rápidas por partes de los algoritmos, 2 de estos estudios se basan en polinomios de Legendre para conseguir un algoritmo en la planificación de trayectoria[11], [12]. Por otro lado, el otro estudio sobre evasión de obstáculos se basa en un controlador de fuerza para la planificación de trayectoria en el que optimizan el consumo[13].

En otra investigación también se tocan métodos genéticos que se basan en una estructura variable para la planificación de trayectorias[14]. También, se tiene otros métodos como el método de Monte Carlo, método de interpolación de polinomios y métodos híbridos que utilizan una combinación de nuevos métodos[15]–[18].

A partir de esto se hizo una revisión de la investigación sobre cuaternios para hacer la planificación de trayectorias con el nuevo método de cálculo y tener una mejora en eficiencias de tiempo y consumo ya que además de tener esta optimización de tiempo de cálculo se le podría añadir nuevos algoritmos como los mencionados anteriormente para mejorar la eficiencia de la planificación de la trayectoria aún más.

En los estudios sobre los métodos numéricos que emplean cuaternios para la interpolación de trayectorias se han utilizado en diversas aplicaciones, como la manipulación de objetos o la navegación en entornos complejos. Estos métodos permiten generar trayectorias suaves y continuas, cruciales para evitar movimientos erráticos o peligrosos en el entorno de trabajo[19].

La integración de cuaternios en los algoritmos de optimización es particularmente útil para los robots colaborativos, ya que asegura una transición suave entre las posiciones y orientaciones sin introducir perturbaciones indeseadas en el movimiento [20].

El estado actual de la planificación de trayectorias para robots colaborativos de utilizando cuaternios y métodos numéricos ha avanzado considerablemente ya que se analiza la cinemática y dinámica, pero aún existen retos a resolver como la comparación de tiempos cuando se realiza una trayectoria programada[21], [22]. A través de la optimización de cuaternios y la mejora de los métodos numéricos, los robots colaborativos tienen el potencial de alcanzar un rendimiento más eficiente, seguro y adaptable en entornos industriales complejos.

En este trabajo, se presenta un enfoque de planificación de trayectorias para un robot colaborativo de 5 grados de libertad (DOF), utilizando cuaternios para la representación de las rotaciones. A través de métodos numéricos avanzados, se aborda el problema de calcular trayectorias suaves y seguras que permitan al robot realizar tareas complejas en entornos dinámicos, teniendo en cuenta las restricciones del espacio de trabajo y las características del sistema. La solución propuesta se basa en un algoritmo de optimización que, a partir de condiciones iniciales y finales predefinidas, genera una secuencia de movimientos que minimizan el error de trayectoria y aseguran la estabilidad del sistema.

Este enfoque tiene como objetivo mejorar la precisión y la fiabilidad de los robots colaborativos en tareas industriales, así como proporcionar una base para futuras investigaciones en la interacción entre robots y humanos, especialmente utilizar nuevos métodos de optimización de trayectoria junto a este método de optimización de cálculo, esto beneficiaria a las aplicaciones que requieran alta precisión y seguridad en entornos de trabajo compartidos.

II. METODOLOGÍA

A. Modelo cinemático mediante la teoría de cuaternios

De forma análoga a las matrices de transformación homogénea, se desarrolla el análisis de la cinemática directa de un robot articular a través de la teoría de cuaternios, desarrollando en primer lugar, los conceptos básicos del álgebra de cuaternios para posteriormente realizar el modelo cinemático del robot propuesto.

Un cuaternio se encuentra definido por la siguiente expresión:

$$Q = [s\ ,\ v] \tag{1}$$

Donde **s** representa la parte escalar (real) conformada por el parámetro q_0 y **v** representa la parte vectorial conformada por las componentes q_1, q_2 y q_3 tal y como se expresa a continuación:

$$Q = [q_0, q_1, q_2, q_3] \quad (2)$$

Así mismo, es posible denotar el cuaternio en el sistema complejo considerando los términos i, j y k que representan las unidades imaginarias tal y como sigue:

$$Q = q_0 + iq_1 + jq_2 + kq_3 \quad (3)$$

1) álgebra de cuaternios: De forma similar a las operaciones matriciales, el conjunto de cuaternios, con las operaciones de suma y multiplicación forman un sistema matemático conocido como anillo de división no conmutativo y, además, satisface que para cada elemento diferente de cero existe un inverso multiplicativo [23].

La Tabla I resume las relaciones de producto entre las unidades básicas de los cuaternios, proporcionando una representación clave en el plano complejo.

TABLA I
MULTIPLICACIÓN VECTORIAL DE CUATERNIOS EN EL PLANO COMPLEJO

	1	***i***	***j***	***k***
1	1	i	j	k
i	i	-1	k	$-j$
j	j	$-k$	-1	i
k	k	j	$-i$	-1

Se destaca la aplicación de la teoría de cuaternios para describir el movimiento rotacional y traslacional de un cuerpo rígido, facilitando la transformación de orientación y posición en el espacio tridimensional.

Sean los siguientes cuaternios Q_1 , Q_2 y Q_3 cuyos componentes vienen dados por sus índices identificados según el número del cuarternión, se tiene:

$$Q_1 = [s_1, v_1] = [q_{10}, q_{11}, q_{12}, q_{13}]$$
$$Q_2 = [s_2, v_1] = [q_{20}, q_{21}, q_{22}, q_{23}]$$
$$Q_3 = [s_3, v_3] = [q_{30}, q_{31}, q_{32}, q_{33}]$$

El complejo conjugado de un cuaternio viene dado por:

$$Q^* = [s, -v]$$

Como resultado de la expresión anterior, Q^* queda expresado como:

$$Q = [q_0, -q_1, -q_2, -q_3]$$

La norma de un cuaternio representa su magnitud, similar al módulo de un vector. Su normalización se denota como $\|Q\|$.

$$\|Q\| = \sqrt{Q \cdot Q^*} = \sqrt{{q_0}^2 + {q_1}^2 + {q_2}^2 + {q_3}^2} \quad (4)$$

El cuaternio inverso representa Q con los ejes de rotación en direcciones opuestas, permitiendo revertir rotaciones de manera eficiente[24].

$$Q^{-1} = \frac{Q^*}{\|Q\|^2} \quad (5)$$

El producto entre un cuaternio y su inversa da como resultado la unidad.

$$Q \cdot Q^{-1} = Q^{-1} \cdot Q = I = [1,0] \quad (6)$$

2) Rotación y traslación mediante cuaternios: A diferencia de las matrices de rotación, los cuaternios permiten representar rotaciones utilizando cualquier vector como eje, lo que simplifica las operaciones y mejora la eficiencia del cálculo de orientación. Para ello, el vector debe ser unitario, conocido como eje de Euler, como se ilustra en la Fig. 1.

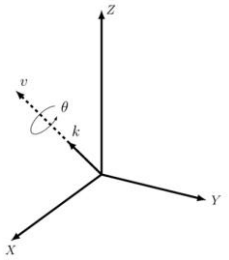

Fig. 1 Movimiento rotacional con respecto a un eje cualquiera.

A partir de la teoría de cuaternios establecida por Hamilton (1866), el cuaternio que describe la rotación de un cuerpo rígido debe ser de Norma 1, el cual es expresado como:

$$Q = \left(\cos\left(\frac{\theta}{2}\right), \vec{k} \cdot sin\left(\frac{\theta}{2}\right)\right) \quad (7)$$

Donde $\vec{k}$ corresponde al vector unitario del vector $\vec{v}$:

$$\vec{k} = \frac{1}{|\vec{v}|} \cdot \vec{v}$$

Además, el vector $\vec{k}$ puede ser expresado en función del sistema de coordenadas X, Y y Z, quedando la notación anterior expresada como:

$$Q_x = \left(\cos\left(\frac{\theta}{2}\right), sin\left(\frac{\theta}{2}\right), 0, 0\right)$$
$$Q_y = \left(\cos\left(\frac{\theta}{2}\right), 0, sin\left(\frac{\theta}{2}\right), 0\right)$$
$$Q_z = \left(\cos\left(\frac{\theta}{2}\right), 0, 0, sin\left(\frac{\theta}{2}\right)\right) \quad (8)$$

3) Cinemática inversa mediante cuaternios. El análisis de la cinemática inversa se desarrolla mediante un algoritmo evolutivo basado en el cuaternio dual, estableciendo la relación entre la postura del efector final y las variables articulares del robot [25]. Se emplea la matriz Jacobiana, formulada con el método de Newton-Raphson, para relacionar los cambios en el espacio cartesiano y articular. En caso de que la matriz Jacobiana no sea cuadrada o invertible, se recurre a la pseudoinversa [26].

En el espacio hipercomplejo R^4, la matriz Jacobiana se define por las derivadas parciales del cuaternio dual respecto a las variables articulares [27]. Si la matriz es rectangular, se emplea la pseudoinversa por la izquierda para garantizar la solución [28]. La relación entre la velocidad del efector y la velocidad articular se expresa como $\hat{v} = J_I \cdot \dot{\theta}$, permitiendo la implementación de control por realimentación basado en el error entre la postura deseada y la postura actual [29].

El esquema de control en lazo cerrado se basa en la integración numérica utilizando la pseudo-inversa del Jacobiano, donde la velocidad angular se ajusta mediante la matriz de ganancia K. La actualización de las coordenadas articulares se realiza iterativamente mediante $\theta_{k+1} = \theta_k + J_I^{\dagger} \cdot K \cdot e(t)$, asegurando la convergencia al estado deseado [30].

Para transformar la posición y orientación en coordenadas cartesianas al espacio, se utiliza el cuaternio de rotación derivado de la matriz de rotación y los ángulos de Euler.

B. Planeación de trayectorias en el espacio articular.

Tras el desarrollo de la cinemática del robot, es fundamental implementar un modelo de control de trayectoria eficiente, ya sea en el espacio articular o cartesiano. En robótica, destacan las trayectorias punto a punto y continuas, cuya variación angular suave se logra mediante interpolaciones polinómicas de alto grado o con enlace parabólico.

Aunque el control en el espacio articular suele emplearse en modelos dinámicos para regular el torque del actuador, esta investigación analiza la eficiencia de modelos cinemáticos mediante matrices de transformación homogénea y cuaternios [31].

La Fig. 2 muestra el diagrama general del control de trayectoria en el espacio articular.

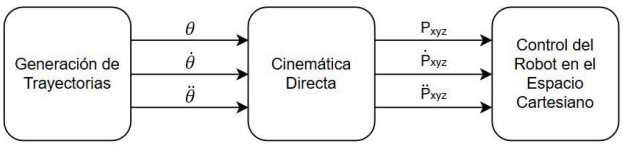

Fig. 2 Vectores unitarios en el sistema de coordenada XYZ

Las interpolaciones en el ámbito de la robótica son esenciales para el control de trayectorias, especialmente cuando se requiere un movimiento suave y sin discontinuidades. Existen varios métodos de interpolación, entre ellos la interpolación de grado 3, grado 5 y las interpolaciones con enlaces parabólicos, que permiten generar trayectorias entre puntos de manera eficiente y controlada.

1) Interpolación de Grado 3. La interpolación de grado 3, también conocida como polinómica cúbica, es ampliamente utilizada debido a su simplicidad y capacidad para generar trayectorias suaves. En esta interpolación, se utiliza un polinomio de tercer grado para relacionar la posición de un robot con el tiempo, garantizando que el movimiento sea continuo en términos de posición, velocidad y aceleración. Su principal ventaja es evitar cambios bruscos en la velocidad angular y asegurar una transición suave entre los estados inicial y final [32].

2) Interpolación de Grado 5. La interpolación de grado 5 es más compleja que la de grado 3 y permite una mayor precisión en la trayectoria, ya que el polinomio de quinto grado proporciona una mayor flexibilidad en la modelización de la aceleración angular y la velocidad. Esta interpolación asegura que tanto la posición, la velocidad como la aceleración sean continuas, resultando en trayectorias más suaves y adaptadas a movimientos más exigentes [33]. La ventaja principal es su capacidad de controlar de manera precisa los cambios en la aceleración.

3) Interpolación con enlace parabólico 4-3-4. En el caso de las interpolaciones con enlace parabólico, se busca suavizar las transiciones entre trayectorias, especialmente cuando se trata de un cambio entre movimientos rápidos o cuando se presentan cambios abruptos en la velocidad. La interpolación 4-3-4 utiliza un enlace parabólico que conecta dos trayectorias en el espacio articular o cartesiano mediante curvas suaves de cuarta y tercera orden. Este método es útil para mantener una transición continua y evitar movimientos bruscos que puedan ser perjudiciales para la precisión del robot [34].

4) Interpolación con enlace parabólico 3-5-3. La interpolación 3-5-3 es una variante más compleja en la cual se emplean enlaces parabólicos de tercer y quinto orden para conseguir una transición más fluida entre los movimientos. Este tipo de interpolación es ideal para trayectorias que requieren un mayor control de la aceleración, siendo especialmente efectiva en aplicaciones donde se necesitan movimientos precisos y controlados [35].

5) Interpolación con enlace parabólico 4-1-4. Finalmente, la interpolación 4-1-4 combina un enlace parabólico de orden cuatro, con un enlace lineal en el medio (orden 1), lo que permite generar trayectorias con un cambio de velocidad controlado en la transición. Este método es particularmente útil cuando se quiere evitar movimientos bruscos, pero manteniendo una velocidad constante entre los puntos intermedios [36]. Es común en aplicaciones donde se prioriza la continuidad de la posición y la velocidad, pero con una mínima variación en la aceleración. Como se muestra en la Fig. 3.

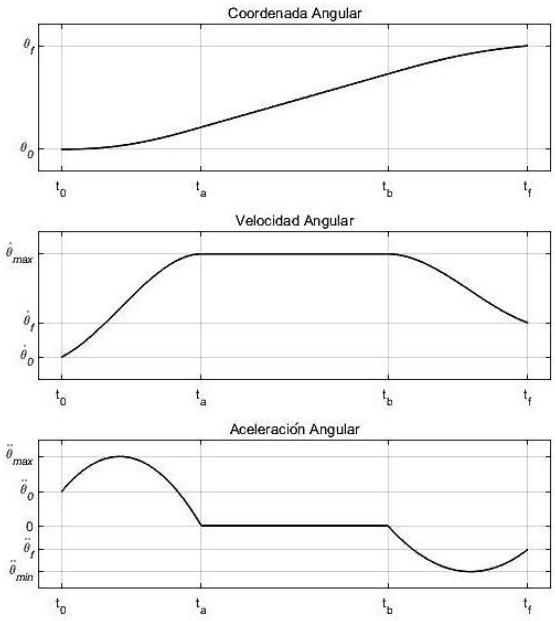

Fig. 3 Representación de la posición, velocidad y aceleración articular mediante interpolaciones polinómicas 4-1-4.

C. Planeación de trayectorias en el espacio cartesiano

El movimiento de un robot se define generalmente en el espacio cartesiano a través de su posición y orientación en los ejes X, Y y Z, así como la velocidad del efector final. Para obtener este movimiento, se emplea la cinemática inversa mediante un algoritmo iterativo que determina las variables articulares necesarias para cada posición del efector, lo que aumenta el costo computacional a medida que se incrementa la precisión en la trayectoria. Este proceso de planificación y control de movimiento se ilustra en un diagrama que integra la generación de trayectorias en el espacio cartesiano con el control en el espacio articular.

El movimiento del efector final se describe utilizando funciones matemáticas en función del tiempo o por medio de coordenadas cartesianas, lo que asegura una trayectoria suave si la cinemática inversa se calcula correctamente. Es crucial imponer restricciones en las variables articulares, teniendo en cuenta el espacio de trabajo del robot y sus posibles singularidades. Esta metodología permite controlar la posición y orientación del robot desde el inicio hasta el final, generando trayectorias continuas, especialmente en trayectorias lineales y splines, que no pueden ser logradas únicamente mediante la planificación en el espacio articular.

III. DESARROLLO DE LA SOLUCIÓN

Para analizar la cinemática del robot propuesto, se define primero su configuración geométrica, que simula la morfología de un robot colaborativo de cinco grados de libertad, como se muestra en la Fig. 4.

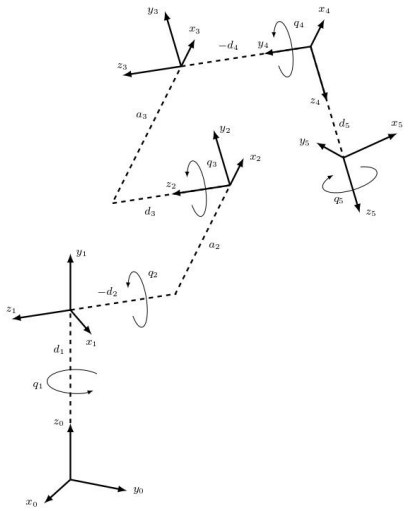

Fig. 4 Geometría y definición de los parámetros de Denavit Hartenberg del robot colaborativo de 5 grados de libertad.

Se establecen los parámetros Denavit-Hartenberg para desarrollar la cinemática directa mediante matrices de transformación homogénea, los cuales se detallan en la Tabla II.

TABLA II
PARÁMETROS DE DENAVIT HARTENBERG

i	a_k	α_k	d_k	θ_k
1	0	90°	d_1	θ_1
2	a_2	0°	$-d_2$	θ_2
3	a_3	0°	d_3	θ_3
4	0	90°	$-d_4$	θ_4
5	0	0°	d_5	θ_5

A. Cinemática directa mediante la teoría de cuaternios

Mediante el producto de los cuaternios duales, que vinculan la rotación y traslación entre el sistema de coordenadas de un eslabón y el eslabón previo, se obtiene la cinemática directa del robot articular. Esto es posible gracias a la compatibilidad con los parámetros Denavit-Hartenberg especificados en la Tabla II. Al reemplazar dichos parámetros en la ecuación 9.

$$_{k-1}DQ^{k} = Q_{z0} \otimes Q_{x0} + \varepsilon(Q_{z0} \otimes Q_{x\varepsilon} + Q_{z\varepsilon} \otimes Q_{x0}) \qquad (9)$$

De ello se obtiene el cuaternio dual que relaciona el sistema de coordenadas del eslabón 1 con el sistema base del robot.

$$_{0}DQ^{1} = \frac{\sqrt{2}}{2}cos\left(\frac{\theta_1}{2}\right) + \frac{\sqrt{2}}{2}cos\left(\frac{\theta_1}{2}\right)i + \frac{\sqrt{2}}{2}sin\left(\frac{\theta_1}{2}\right)j + \frac{\sqrt{2}}{2}sin\left(\frac{\theta_1}{2}\right)k \; +$$

$$\varepsilon\left(-d1\frac{\sqrt{2}}{4}sin\left(\frac{\theta_1}{2}\right)-d1\frac{\sqrt{2}}{4}sin\left(\frac{\theta_1}{2}\right)i+d1\frac{\sqrt{2}}{4}cos\left(\frac{\theta_1}{2}\right)j+d1\frac{\sqrt{2}}{4}cos\left(\frac{\theta_1}{2}\right)k\right)$$

De manera similar, para obtener el cuaternio dual que vincula el sistema de coordenadas del eslabón 2 con el del eslabón 1, se utiliza el mismo procedimiento y lo mismo con el 3 al 2, del 4 al 3, del 5 al 4 respectivamente:

$${}_1DQ^2=cos\left(\frac{\theta_2}{2}\right)+0i+0j+sin\left(\frac{\theta_2}{2}\right)k+\varepsilon\left(\frac{d2}{2}sin\left(\frac{\theta_2}{2}\right)+\frac{a2}{2}cos\left(\frac{\theta_2}{2}\right)i+\frac{a2}{2}sin\left(\frac{\theta_2}{2}\right)j-\frac{d2}{2}cos\left(\frac{\theta_2}{2}\right)k\right)$$

$${}_2DQ^3=cos\left(\frac{\theta_3}{2}\right)+0i+0j+sin\left(\frac{\theta_3}{2}\right)k+\varepsilon\left(-\frac{d3}{2}sin\left(\frac{\theta_3}{2}\right)+\frac{a3}{2}cos\left(\frac{\theta_3}{2}\right)i+\frac{a3}{2}sin\left(\frac{\theta_3}{2}\right)j+\frac{d3}{2}cos\left(\frac{\theta_3}{2}\right)k\right)$$

$${}_3DQ^4=\frac{\sqrt{2}}{2}cos\left(\frac{\theta_4}{2}\right)+\frac{\sqrt{2}}{2}cos\left(\frac{\theta_4}{2}\right)i+\frac{\sqrt{2}}{2}sin\left(\frac{\theta_4}{2}\right)j+\frac{\sqrt{2}}{2}sin\left(\frac{\theta_4}{2}\right)k+\varepsilon\left(d4\frac{\sqrt{2}}{4}sin\left(\frac{\theta_4}{2}\right)+d4\frac{\sqrt{2}}{4}sin\left(\frac{\theta_4}{2}\right)i-d4\frac{\sqrt{2}}{4}cos\left(\frac{\theta_4}{2}\right)j-d4\frac{\sqrt{2}}{4}cos\left(\frac{\theta_4}{2}\right)k\right)$$

$${}_4DQ^5=cos\left(\frac{\theta_5}{2}\right)+0i+0j+sin\left(\frac{\theta_5}{2}\right)k+\varepsilon\left(-\frac{d5}{2}sin\left(\frac{\theta_5}{2}\right)+0i+0j+\frac{d5}{2}cos\left(\frac{\theta_5}{2}\right)k\right)$$

Así, la relación entre el sistema de coordenadas del efector final y el sistema de coordenadas base del robot se determina mediante la multiplicación sucesiva de los cuaternios duales previamente calculados. El cuaternio dual resultante como se indica en la ecuación 10 describe la rotación y traslación del efector final con respecto al origen del robot.

$${}_0DQ^5=(q_0+q_1i+q_2j+q_3k)+\varepsilon(q_{0\varepsilon}+q_{1\varepsilon}i+q_{2\varepsilon}j+q_{3\varepsilon}k) \quad (10)$$

Con los parámetros de los cuaternios duales de rotación y traslación obtenidos, se puede expresar de manera explícita en el espacio R^3 la orientación y posición, proporcionando una descripción directa y compacta de las transformaciones espaciales en la cinemática del robot.

Finalmente, la combinación de estos cálculos genera la matriz de transformación homogénea ${}_0T^5$, que representa de manera consolidada y explícita la postura del efector final del robot.

B. *Cinemática inversa mediante cuaternios*

El cálculo de la cinemática directa mediante cuaternios duales permite obtener de manera sencilla las coordenadas que definen la orientación y posición del efector final, en contraste con el complicado cálculo de las coordenadas articulares a través de métodos algebraicos. Para la cinemática inversa, se utiliza un algoritmo evolutivo que aproxima el valor deseado en cada iteración, asegurando la convergencia del sistema.

Primero, se calcula el Jacobiano del cuaternio dual que relaciona el efector final con el origen del sistema de coordenadas, obteniendo una matriz de tamaño 8x5. Luego, se determina la pseudo inversa de esta matriz utilizando Matlab. A continuación, se define la postura inicial del robot y se calcula el cuaternio dual inicial, que depende de la ubicación del efector final. La matriz de ganancia, inicialmente definida como la matriz identidad, puede ser ajustada según las características del sistema.

El algoritmo evalúa el error entre el cuaternio dual deseado y el inicial, y mediante iteraciones actualiza los valores articulares hasta que el error es suficientemente pequeño (0.0001°). Una vez alcanzada la precisión deseada, el algoritmo concluye la cinemática inversa.

Aunque el método numérico basado en cuaternios es preciso, puede fallar en algunos casos debido a singularidades en el sistema, por lo que es crucial delimitar el espacio de trabajo del robot y establecer restricciones para evitar estas zonas de singularidades.

C. *Planeación de trayectorias para el robot propuesto.*

Se establece inicialmente la planeación de trayectoria en el espacio articular para evaluar el tiempo de ejecución de movimientos punto a punto, considerando solo la cinemática directa del robot. Se emplea la interpolación polinómica 4-1-4, que garantiza una velocidad constante en el tramo intermedio del movimiento articular, permitiendo definir los parámetros de posición, velocidad y aceleración angular al inicio y final de la trayectoria.

Además, se define la trayectoria que seguirá el robot en el espacio cartesiano, manteniendo constante la posición en el eje Z y la orientación del efector final perpendicular al plano XY, debido a la configuración geométrica del robot. Esta orientación se describe mediante ángulos de Euler $R(\emptyset)=[\pi \quad 0 \quad 0]$, resultando en la siguiente matriz de rotación:

$${}_0R^5=\begin{bmatrix}1&0&0\\0&-1&0\\0&0&-1\end{bmatrix} \quad (11)$$

IV. Resultados

Para el desarrollo de las simulaciones y el análisis de la eficiencia del modelo cinemático del robot y el control de trayectoria, se define la geometría del robot de cinco grados de libertad utilizando los parámetros Denavit–Hartenberg, los cuales se detallan en la Tabla III.

TABLA III
DEFINICIÓN DE LOS PARÁMETROS DE DENAVIT HARTENBERG DEL ROBOT

i	θ_i	$d_i\ (m)$	$a_i\ (m)$	α_i
1	θ_1	0.12	0	$\pi/2$
2	θ_2	−0.09	0.215	0°
3	θ_3	0.075	0.180	0°
4	θ_4	−0.075	0	$\pi/2$
5	θ_5	0.055	0	0°

A. *Resultados de la cinemática directa.*

Se desarrolló un algoritmo en Matlab para calcular la cinemática directa utilizando matrices de transformación homogénea y cuaternios. El tiempo de ejecución de cada método se midió mediante las funciones de temporización del programa. Los resultados, presentados en tablas, comparan los tiempos de cálculo en 10 ejecuciones consecutivas para una configuración angular previamente definida, lo que permite evaluar la eficiencia de ambos enfoques en términos computacionales. El primer muestreo utiliza la siguiente configuración articular: $q_1 = 70°$, $q_2 = 70°$, $q_3 = -50°$, $q_4 = 30°$, $q_5 = 15°$. La comparación de tiempos de ejecución se muestra en la Tabla IV.

TABLA IV
TABLA COMPARATIVA DE LOS TIEMPOS OBTENIDOS PARA LA CINEMATICA DIRECTA (MUESTREO 1)

Muestras	Matrices T.H. (µs)	Cuaternios (µs)	Diferencia (µs)
1	340	191	149
2	207	144	63
3	143	99	44
4	205	143	62
5	396	158	238
6	154	133	21
7	290	140	150
8	168	110	58
9	148	217	-69
10	155	100	55
Promedio	220.6	143.5	77.1

Según los resultados de la Tabla IV, se observa que el tiempo requerido para calcular la cinemática directa utilizando cuaternios es un 34.9% más rápido que al usar matrices de transformación homogénea.

En el segundo muestreo, se define la siguiente configuración articular: $q_1=130°$, $q_2=20°$, $q_3=70°$, $q_4=-20°$, $q_5=50°$. Al igual que en el primer caso, se ejecuta el algoritmo y se presentan los resultados en la Tabla V.

TABLA V
TABLA COMPARATIVA DE LOS TIEMPOS OBTENIDOS PARA LA CINEMATICA DIRECTA (MUESTREO 2)

Muestras	Matrices T.H. (µs)	Cuaternios (µs)	Diferencia (µs)
1	169	103	66
2	176	99	77
3	132	108	24
4	189	110	79
5	242	151	91
6	183	115	68
7	168	131	37
8	144	89	55
9	165	103	62
10	166	92	74
Promedio	173.4	110.1	63.3

Los datos obtenidos muestran una mejora significativa al utilizar la teoría de cuaternios como método alternativo en la cinemática directa del robot, ya que ofrece un tiempo de cálculo en 36.5% más rápido que el método clásico. Esta reducción en el costo computacional se logra al obtener de manera implícita la rotación y posición de la cadena cinemática del robot, relacionando los sistemas de coordenadas de los eslabones de forma compacta en el espacio R^4. Además, proporciona una representación más estable de la orientación, evitando singularidades y errores acumulativos.

Se compara el tiempo consumido al ejecutar de manera iterativa la cinemática directa durante la planificación de una trayectoria en el espacio articular. Se definen los ángulos de las articulaciones para trazar figuras geométricas en el plano tridimensional usando interpolaciones polinómicas 4-1-4. El algoritmo, desarrollado en Matlab, visualiza el movimiento del robot a través del Robotics Toolbox de Peter Corke, lo que permite simular y validar trayectorias. Para evaluar la eficiencia, se mide el tiempo de ejecución de 10 iteraciones consecutivas, donde el robot sigue trayectorias punto a punto. La primera simulación muestra en la Fig. 4 una trayectoria rectangular dentro del espacio de trabajo esférico del robot.

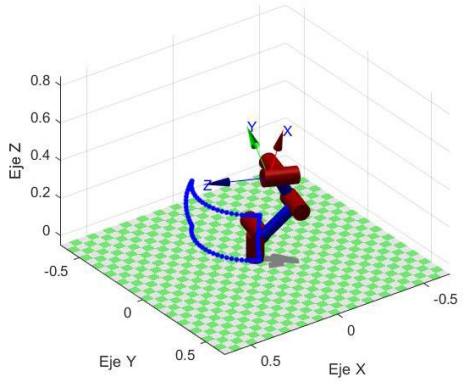

Fig. 5 Simulación gráfica de la trayectoria de un rectángulo.

Los tiempos empleados para el desarrollo de este movimiento se presentan en la Tabla VI a continuación.

TABLA VI
TABLA COMPARATIVA DE LOS TIEMPOS OBTENIDOS PARA LA TRAYECTORIA DE UN RECTANGULO.

Muestras	Matrices T.H. (s)	Cuaternios (s)	Diferencia (s)
1	13.600725	13.723576	-0.122851

2	13.738345	13.640602	0.097743
3	13.642143	13.530019	0.112124
4	13.757194	13.77436	-0.017166
5	13.768716	13.635513	0.133203
6	13.579537	13.573696	0.005841
7	13.577181	13.461509	0.115672
8	13.673837	13.52893	0.144907
9	13.539761	13.571599	-0.031838
10	13.557284	13.464604	0.09268
Promedio	13.643472	13.590441	0.0530315

Como se observa en la tabla VI, el uso de matrices de transformación y cuaternios para la generación de trayectorias muestra tiempos de ejecución similares, con una diferencia de 53031.5 µs en microsegundos. La aplicación con cuaternios es ligeramente más rápida, con una ventaja del 0.39% en la escala temporal de ejecución.

En la segunda simulación, se ejecuta el algoritmo de trayectoria utilizando interpolaciones polinómicas para graficar la silueta de un triángulo en el espacio cartesiano, como se muestra en la Fig. 6.

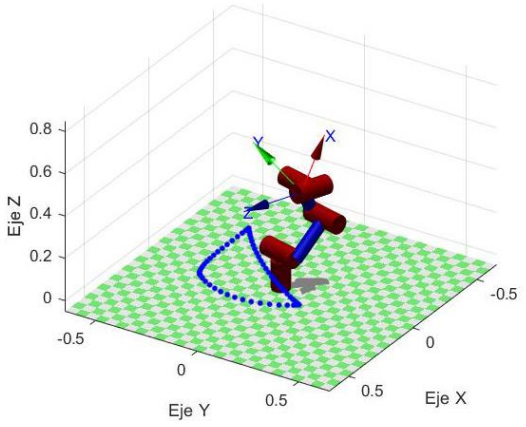

Fig. 6 Simulación gráfica de la trayectoria de un triángulo.

Los tiempos de ejecución para el desarrollo de este movimiento se presentan en la Tabla VII a continuación.

TABLA VII
TABLA COMPARATIVA DE LOS TIEMPOS OBTENIDOS PARA LA TRAYECTORIA DE UN TRIANGULO

Muestras	Matrices T.H. (s)	Cuaternios (s)	Diferencia (s)
1	9.464876	9.491721	-0.026845
2	9.461173	9.42302	0.038153
3	9.361766	9.358552	0.003214
4	9.53777	9.542913	-0.005143
5	9.539279	9.544866	-0.005587
6	9.591253	9.580252	0.011001
7	9.519216	9.466862	0.052354
8	9.470235	9.382347	0.087888
9	9.422917	9.366565	0.056352
10	9.550292	9.559202	-0.00891
Promedio	9.4918777	9.47163	0.0202477

Los resultados de esta trayectoria muestran una diferencia mínima respecto a la simulación anterior, con una variación de 20247.7 µs. Se observa una pequeña ventaja de la aplicación de cuaternios, que es del 0.21%, manteniendo una ejecución similar en términos de tiempo.

B. Resultados de la cinemática inversa

En este análisis, se define una trayectoria cuadrada en el espacio cartesiano para calcular los ángulos articulares a partir de la cinemática inversa, manteniendo la orientación del efector final perpendicular al plano XY a lo largo del recorrido. La orientación del robot se define mediante los ángulos de Euler ZYX indicado en el punto 3 del desarrollo de la solución y el cuaternio $Q_0 = [0 \quad 1 \quad 0 \quad 0]$, para la rotación del efector final. Se especifican las posiciones de los vértices del cuadrado en metros:

$$P_1(x,y,z) = [0.37 \quad 0.12 \quad 0.15]$$
$$P_2(x,y,z) = [0.37 \quad -0.10 \quad 0.15]$$
$$P_3(x,y,z) = [0.15 \quad -0.10 \quad 0.15]$$
$$P_4(x,y,z) = [0.15 \quad 0.12 \quad 0.15]$$

Y utilizando el Robotics Toolbox de Peter Corke en Matlab, se mide el tiempo de ejecución del algoritmo de cinemática inversa al recorrer la trayectoria 10 veces. La Fig. 7 ilustra la trayectoria seguida por el robot.

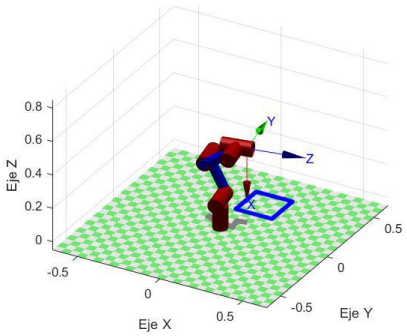

Fig. 7 Simulación gráfica de la trayectoria de un cuadrado.

La Tabla VIII muestra las coordenadas cartesianas obtenidas de los ángulos calculados mediante la cinemática inversa, comparando los resultados del método geométrico y el

método numérico, con una precisión entre 0.0001 y 0.001 decimales.

TABLA VIII
TABLA COMPARATIVA DE LAS VARIABLES ARTICULARES

Posiciones	Método Geométrico	Método Numérico
$P_1(x,y,z)$	[0.37 0.12 0.15]	[0.3691 0.1199 0.1497]
$P_2(x,y,z)$	[0.37 −0.10 0.15]	[0.3685 −0.0992 0.1497]
$P_3(x,y,z)$	[0.15 −0.10 0.15]	[0.1504 −0.1008 0.15]
$P_4(x,y,z)$	[0.15 0.12 0.15]	[0.1492 0.1196 0.1498]

La Tabla IX muestra los tiempos medidos para la simulación de este movimiento.

TABLA IX
TABLA COMPARATIVA DE LOS TIEMPOS OBTENIDOS PARA EL CÁLCULO DE LA CINEMÁTICA INVERSA

Muestras	Matrices T.H. (s)	Cuaternios (s)	Diferencias (s)
1	10.409112	10.332619	0.076493
2	10.352812	10.184559	0.168253
3	10.428634	10.365963	0.062671
4	10.441311	10.358532	0.082779
5	10.383657	10.343386	0.040271
6	10.317494	10.318836	-0.001342
7	10.35397	10.279914	0.074056
8	10.283792	10.307498	-0.023706
9	10.408471	10.372315	0.036156
10	10.354314	10.329199	0.025115
Promedio	10.373357	10.319282	0.0540746

Los valores nos muestran que la cinemática inversa utilizando cuaternios es un 0.52% más rápida, con una diferencia de 54074 µs en la ejecución, aunque presenta un mayor error en la trayectoria, alcanzando un error máximo de 0.0009 m.

V. CONCLUSIONES

En conclusión, las simulaciones en Matlab muestran que los cuaternios mejoran la eficiencia en un 35.7% al promediar 34.9% y 36.5% de los dos muestreos del cálculo de la cinemática directa del robot de 5 grados de libertad. aunque las matrices de transformación y los cuaternios presentan tiempos de ejecución muy similares para la generación de trayectorias, el uso de cuaternios ofrece una ligera ventaja en términos de velocidad, con una mejora del 0.39% en el tiempo de procesamiento.

La cinemática inversa mediante cuaternios es ligeramente más rápida (0.52%) en comparación con otros métodos, pero presenta un pequeño sacrificio en la precisión de la trayectoria, con un error máximo de 0.0009 m.

La precisión es clave según los requisitos de la aplicación, lo que permite decidir entre optimizar la velocidad o priorizar la exactitud. Aunque los métodos iterativos requieren varias repeticiones, pueden ser muy eficientes cuando se optimizan, especialmente cuando se usan junto con técnicas como el Jacobiano de mínimos cuadrados, que mejora la estabilidad y precisión.

El control de movimiento con cuaternios es más eficiente que el de matrices de transformación, aunque, al operar en un espacio R^4, es necesario convertir los resultados a R^3 para expresar la posición y orientación del efector final en el espacio cartesiano. A pesar de este paso adicional, los cuaternios son una buena alternativa en simulaciones.

Finalmente, aunque las simulaciones ofrecen una aproximación de la velocidad de respuesta, no garantizan resultados exactos. Sin embargo, el modelo propuesto proporciona una visión clara del comportamiento cinemático y dinámico del robot, facilitando su desarrollo y alcanzando los objetivos de la investigación.

AGRADECIMIENTO/RECONOCIMIENTO

En esta sección puede agradecer a las personas e instituciones que contribuyeron con el desarrollo del estudio.

REFERENCIAS

[1] M. P. Yagüe, J. E. S. García, M. Santos, M. P. Yagüe, J. E. S. García, and M. Santos, “Hacia la optimización de trayectorias de robots colaborativos mediante algoritmos genéticos,” *XIX Simposio CEA de Control Inteligente: libro de actas*, pp. 17–20, 2025.

[2] S. Chen and J. T. Wen, “Industrial Robot Trajectory Tracking Control Using Multi-Layer Neural Networks Trained by Iterative Learning Control,” *Robotics 2021, Vol. 10, Page 50*, vol. 10, no. 1, p. 50, Mar. 2021, doi: 10.3390/ROBOTICS10010050.

[3] M. X. Kong, C. Ji, Z. S. Chen, and R. F. Li, “Application of orientation interpolation of robot using unit quaternion,” *2013 IEEE International Conference on Information and Automation, ICIA 2013*, pp. 384–389, 2013, doi: 10.1109/ICINFA.2013.6720328.

[4] Y. Aydm and S. Kucuk, “Quaternion based inverse kinematics for industrial robot manipulators with euler wrist,” *2006 IEEE International Conference on Mechatronics, ICM*, pp. 581–586, 2006, doi: 10.1109/ICMECH.2006.252591.

[5] J. Ramírez-Gordillo, E. A. Merchán-Cruz, E. Lugo-González, R. G. Rodríguez-Cañizo, R. Ponce-Reynoso, and G. Urriolagoitia-Sosa, “Desarrollo de una Nueva Solución Compacta a la Cinemática de Manipuladores Robóticos basada en Cuaterniones Duales,” *Revista Iberoamericana de Automática e Informática Industrial RIAI*, vol. 8, no. 4, pp. 334–344, Oct. 2011, doi: 10.1016/J.RIAI.2011.09.012.

[6] J. Z. Vidakovic, M. P. Lazarevic, V. M. Kvrgic, Z. Z. Dancuo, and G. Z. Ferenc, “Advanced quaternion forward kinematics algorithm including overview of different methods for robot kinematics,” *FME Transactions*, vol. 42, no. 3, pp. 189–199, 2014, doi: 10.5937/FMET1403189V.

[7] F. Thomas, “Approaching dual quaternions from matrix algebra,” *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1037–1048, Oct. 2014, doi: 10.1109/TRO.2014.2341312.

[8] X. Guan and J. Wang, “Trajectory planning theory and method of industrial robot,” *ICCRD2011 - 2011 3rd International Conference on Computer Research and Development*, vol. 2, pp. 340–343, 2011, doi: 10.1109/ICCRD.2011.5764146.

[9] Q. Qingwen, W. Jixiang, and S. Xiujun, “Trajectory planning of a 6-DOF robot based on RBF neural networks,” *2007 IEEE International Conference on Robotics and Biomimetics, ROBIO*, pp. 324–329, 2007, doi: 10.1109/ROBIO.2007.4522182.

[10] X. Y. Wang, Z. K. Zhang, and B. Zhou, “Application of RBF neural network in trajectory planning of robot,” *2009 International Conference on Artificial Intelligence and Computational Intelligence, AICI 2009*, vol. 2, pp. 493–496, 2009, doi: 10.1109/AICI.2009.305.

[11] Y. Wen and P. Pagilla, “Path-Constrained and Collision-Free Optimal Trajectory Planning for Robot Manipulators,” *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 2, pp. 763–774, Apr. 2023, doi: 10.1109/TASE.2022.3169989.

[12] Y. Wen and P. R. Pagilla, “Path-constrained optimal trajectory planning for robot manipulators with obstacle avoidance,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 1421–1426, 2021, doi: 10.1109/IROS51168.2021.9636674.

[13] C. Tiseo, V. Ivan, W. Merkt, I. Havoutis, M. Mistry, and S. Vijayakumar, “A Passive Navigation Planning Algorithm for Collision-free Control of Mobile Robots,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, pp. 8223–8229, 2021, doi: 10.1109/ICRA48506.2021.9561377.

[14] M. Pang and Z. F. Chen, “Trajectory planning based on variable structure GA of a three-limbed robot,” *2010 International Conference on Logistics Systems and Intelligent Management, ICLSIM 2010*, vol. 1, pp. 611–614, 2010, doi: 10.1109/ICLSIM.2010.5461348.

[15] H. Yang and B. Xiao, “Kinematics Analysis of Delta Parallel Robot Based on Numerical Solutions Method,” *Proceedings - 2024 International Conference on Artificial Intelligence and Digital Technology, ICAIDT 2024*, pp. 294–297, 2024, doi: 10.1109/ICAIDT62617.2024.00070.

[16] Z. He and J. Li, “Six-degree-of-freedom Robot Trajectory Planning Based on MATLAB,” *2022 International Conference on Automation, Robotics and Computer Engineering, ICARCE 2022*, 2022, doi: 10.1109/ICARCE55724.2022.10046483.

[17] B. Shi and H. Zeng, “Time-Optimal Trajectory Planning for Industrial Robot based on Improved Hybrid-PSO,” *Chinese Control Conference, CCC*, vol. 2021-July, pp. 3888–3893, Jul. 2021, doi: 10.23919/CCC52363.2021.9549441.

[18] G. Singh and V. K. Banga, “Kinematics and trajectory planning analysis based on hybrid optimization algorithms for an industrial robotic manipulators,” *Soft Computing*, vol. 26, no. 21, pp. 11339–11372, Nov. 2022, doi: 10.1007/S00500-022-07423-Y/METRICS.

[19] K. Bai, Y. Zhang, J. Cheng, and Q. Zhao, “Kinematic analysis of the 6R serial robot based on double quaternions,” *2015 12th International Computer Conference on Wavelet Active Media Technology and Information Processing, ICCWAMTIP 2015*, pp. 482–486, Jun. 2016, doi: 10.1109/ICCWAMTIP.2015.7494036.

[20] W. Ge, L. Chen, X. Wang, E. Xing, and T. Zielinska, “Kinematics Modeling and Analysis of Manipulator Using the Dual Quaternion,” *Proceedings of 2019 IEEE International Conference on Mechatronics and Automation, ICMA 2019*, pp. 750–755, Aug. 2019, doi: 10.1109/ICMA.2019.8816603.

[21] E. Sariyildiz and H. Temeltas, “Solution of inverse kinematic problem for serial robot using quaterninons,” *2009 IEEE International Conference on Mechatronics and Automation, ICMA 2009*, pp. 26–31, 2009, doi: 10.1109/ICMA.2009.5246684.

[22] J. Liang *et al.*, “Dual quaternion based kinematic control for Yumi dual arm robot,” *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2017*, pp. 114–118, Jul. 2017, doi: 10.1109/URAI.2017.7992899.

[23] Cepeda Gómez, R. (2010). Sistema de Control Robusto, Basado en Cuaternios para un Satélite de Órbita Baja [Tesis de Maestría, Pontifica Universidad Javeriana]

[24] Hernández Vergara, R., Magaña Mendez, M., Ramos Fernández, J., y Hernández Cortes, T. (2023), Análisis cinemático utilizando cuaternios duales. Pädi Boletín Científico De Ciencias Básicas E Ingenierías Del ICBI, 10(20), 52-60. https://doi.org/10.29057/icbi.v10i20.9297

[25] J. Craig, Introduction to Robotics: Mechanics and Control, 3rd ed. Pearson, 2005

[26] P. Gouasmi, et al., "Pseudoinverse Methods in Robotics," Mechanics and Robotics Journal, vol. 45, no. 1, pp. 98-112, 2012

[27] H. Pham, et al., "Dual Position Control Scheme Using the Jacobian Pseudo-Inverse," International Conference on Robotics and Automation (ICRA), pp. 1045-1052, 2010.

[28] M. Hughes, "Numerical Stability in Quaternion Computation," Computational Mechanics, vol. 33, no. 7, pp. 890-905, 2019.

[29] P. Fraisse, et al., "Quaternion-Based Motion Control in Robotics," IEEE Transactions on Robotics, vol. 41, no. 1, pp. 102-114, 2020.

[30] A. Ponce, et al., "Inverse Kinematics Using Dual Quaternion State Representation," Journal of Robotics, vol. 38, no. 2, pp. 125-137, 2022.

[31] B. Siciliano, L. Sciavicco, L. Villani, y G. Oriolo, Robotics: Modelling, Planning and Control, Springer, 2010.

[32] J. Craig, Introduction to Robotics: Mechanics and Control, 3rd ed. Upper Saddle River, NJ, USA: Pearson, 2005.

[33] M. Spong, S. Hutchinson, and M. Vidyasagar, Robot Modeling and Control. Hoboken, NJ, USA: Wiley, 2006.

[34] R. Paul, Robot Manipulators: Mathematics, Programming, and Control. Cambridge, MA, USA: MIT Press, 1981.

[35] B. Siciliano and O. Khatib, Springer Handbook of Robotics, 2nd ed. Cham, Switzerland: Springer, 2016.

[36] R. Kelly, V. Santibáñez, and A. Loria, Control of Robot Manipulators in Joint Space. London, UK: Springer, 2005.