CEA Comité Español de Automática
XLVI JORNADAS AUTOMÁTICA CARTAGENA 2025
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

Cinemática inversa rápida de robots blandos utilizando el algoritmo TRR

García-Samartín, Jorge Francisco[a,*], Martín-Díaz, Mar[a], del Cerro, Jaime[a], Barrientos, Antonio[a]

[a] *Centro de Automática y Robótica, Universidad Politécnica de Madrid, C/ José Gutiérrez Abascal, nº 2, 28006, Madrid, España.*

To cite this article: García-Samartín, Jorge Francisco, Martín-Díaz, Mar, del Cerro Jaime, Barrientos Antonio. 2025. Fast Inverse Kinematics for Soft Manipulators Using TRR Algorithm. Jornadas de Automática, 46. https://doi.org/10.17979/ja-cea.2025.46.12253

Resumen

A la hora de abordar el modelo cinemático inverso de robots blandos, los algoritmos de optimización son la mejor opción a nivel de precisión, facilidad de implementación y manejo de restricciones, pero no suelen lograr soluciones en tiempos suficientemente reducidos. Este artículo propone la utilización del algoritmo TRR (*Trust Region Reflective*), metodología de minimización basada en aproximar la función objetivo en regiones de tamaño variable. El desarrollo se prueba sobre un simulador del robot PETER y, posteriormente, sobre el propio manipulador, comparándose en ambos casos con una red neuronal entrenada para modelar la cinemática inversa del robot. Los errores promedio obtenidos en ambas situaciones son de 0,22 mm y 14 mm para el TRR y de 1,27 mm y 15 mm para la red. Aunque la precisión de las pruebas sobre el robot real tiene aún margen de recorrido, se demuestra cómo el algoritmo presentado es capaz de ofrecer soluciones precisas en bajos tiempos de ejecución.

Palabras clave: Tecnología robótica, Robots manipuladores, Robótica y mecatrónica, Mecatrónica, Metodologías de diseño, Redes neuronales.

Fast Inverse Kinematics for Soft Manipulators Using TRR Algorithm

Abstract

When addressing the inverse kinematic model of soft robots, optimisation algorithms are the best option in terms of precision, ease of implementation and constraint management, but they do not usually achieve solutions in sufficiently short times. This article proposes the use of the TRR (Trust Region Reflective) algorithm, a minimisation methodology based on approximating the objective function in regions of variable size. The development is tested on a PETER robot simulator and, subsequently, on the manipulator itself, comparing it in both cases with a neural network trained to model the inverse kinematics of the robot. The average errors obtained in both situations are 0,22 mm and 14 mm for the TRR and 1,27 mm and 15 mm for the network. Although the accuracy of the tests on the real robot still has room for improvement, it demonstrates how the presented algorithm is capable of offering accurate solutions in low execution times.

Keywords: Robotics technology, Robots manipulators, Robotics and mechatronics, Mechatronics, Design methodologies, Neural networks.

1. Introducción

El campo de aplicaciones de la Robótica Blanda se ha extendido considerablemente en los últimos años. A los usos más tradicionales, como grippers (Navas et al., 2021; Shan et al., 2024) o rehabilitación (Terrile et al., 2021), se le están sumando otros tan diversos como pueden ser la exploración de tuberías (Blanco et al., 2024), la locomoción por entornos desestructurados (Seyidoğlu and Rafsanjani, 2024), la manipulación submarina (Liu et al., 2024) o el apoyo en tareas quirúrgicas (Kaviri et al., 2023).

Modelar analíticamente estos robots no es sencillo debido a la no linealidad de sus ecuaciones, la impredecibilidad de sus deformaciones y la dificultad de determinar paráme-

*Autor para correspondencia: jorge.gsamartin@upm.es

tros propios de sus materiales constitutivos (Wu et al., 2023; García-Samartín et al., 2024c). La cinemática inversa, además, es especialmente complicada de resolver ya que muchas veces estos dispositivos son altamente modulares (Wan et al., 2023) con lo que su resolución implica un proceso secuencial y/o de optimización. Si no se dispone de un modelo rápido y preciso de cada módulo, no se puede abordar la cinemática inversa en tiempo real.

Como se ha mencionado, una primera opción es recurrir al modelado analítico, como se hace en (Keyvanara et al., 2023), donde se consigue resolver el problema inverso para un robot genérico de siete grados de libertad. Estas metodologías requieren de poco coste computacional y son precisas, siempre que se tenga un modelo convenientemente bueno del actuador, lo cual se da principal y casi exclusivamente en robots de cables. Sin embargo, llevan a ecuaciones complejas de difícil manejo en muchas ocasiones y, cuando el número de grados de libertad es muy alto, y las restricciones, pocas, presentan soluciones que dan lugar a movimientos poco naturales.

Otras soluciones se basan en el uso de técnicas de aprendizaje, que han demostrado manejar con éxito este tipo de manipuladores, altamente no lineales y con abundantes singularidades en la mayoría de las ocasiones. Las redes neuronales de tipo *feedforward* (Bianchi et al., 2024), aunque permiten modelar con sencillez sistemas complejos, no permiten gestionar redundancias. Aunque esto se soluciona con arquitecturas más complejas, como pueden ser las redes recurrentes (Sun et al., 2022; Wagaa et al., 2023), se precisa entonces de una gran cantidad de datos reales. Un problema similar ocurre cuando se recurren a técnicas de aprendizaje por refuerzo (Thuruthel et al., 2019) o por imitación (Seleem et al., 2023).

Provenientes de la robótica hiperredundante y del mundo de los videojuegos, los algoritmos iterativos, como FABRIK (Aristidou and Lasenby, 2011), SLInKi (Chiang et al., 2021) o CCD (Martín et al., 2018), son apreciados por sus bajos costos computacionales. Sin embargo, no es fácil gestionar en ellos las redundancias y, además, cuando existen restricciones angulares estrictas, algo que ocurre en robots blandos con mucha más frecuencia que en sistemas hiperredundantes, no logran convergir a una solución.

Las metodologías basadas en optimización son extremadamente versátiles y logran combinar las ventajas de los distintos enfoques anteriores. En base a modelos, analíticos (Bhalkikar et al., 2024), numéricos (Bern and Rus, 2021) o de aprendizaje (Bern et al., 2020), de la cinemática directa, iteran hasta obtener una solución. Funcionan muy bien tanto sin restricciones como con un número elevado de ellas y, aunque pueden llegar a ser lentos, los últimos modelos son capaces de ofrecer alternativas en el orden de milisegundos. Su principal problema es la dificultad de poder posicionar, además del extremo final del robot, puntos intermedios del mismo.

Un trabajo previo (García-Samartín et al., 2024a) resuelve el mismo haciendo uso de un algoritmo genético. Aunque dicha alternativa es precisa y flexible, los tiempos de cálculo son, en general, demasiado elevados. Para paliar ese problema, se introduce, en el presente artículo, una nueva metodología para la obtención de la cinemática inversa de manipuladores blandos modulares.

Este se basa en el algoritmo *Trust Region Reflective* (TRR). Se trata de un método avanzado de optimización de mínimos cuadrados no lineales diseñado específicamente para resolver problemas con restricciones en los parámetros. Opera bajo el concepto de "región de confianza", ajustando los parámetros dentro de un área limitada y utilizando una característica de "reflectividad" para asegurar que se mantengan dentro de los límites predefinidos. Esto permite el establecimiento de restricciones en el espacio de las articulaciones, garantizando así el cumplimiento de ciertos límites de seguridad, y de condicionamientos, por medio de la función objetivo, sobre la forma adoptada por el manipulador, a la vez que se logran tiempos de ejecución muy reducidos.

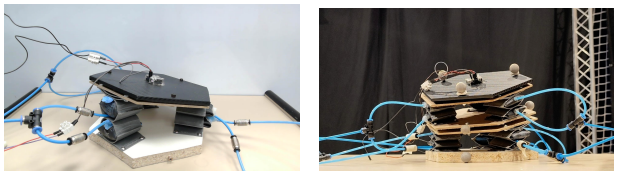
Todo el algoritmo se ha implementado sobre PETER, acrónimo de *Pneumatic and Elastomeric TPU-made Extensible Robot*, un robot serial blando modular desarrollado por el grupo de investigación en Robótica y Cibernética de la Universidad Politécnica de Madrid (García-Samartín et al., 2024b). Para estudiar la validez de la metodología desarrollada, se comparan los resultados con los de una red neuronal *feedforward* encargada también del modelado del robot.

El artículo se ha estructurado de la siguiente manera: en primer lugar, se describe, en la Sección 2, al manipulador sobre el que se desarrolla el algoritmo. Posteriormente, la Sección 2 introduce los fundamentos matemáticos del TRR, así como la red neuronal que se usa para estudiar su validez. Las pruebas a las que se somete el sistema y los resultados de las mismas se muestran en la Sección 4, mientras que la Sección 5 expone las conclusiones y líneas futuras.

2. El manipulador neumático PETER

2.1. *Diseño mecánico*

PETER, acrónimo de *Pneumatic and Elastomeric TPU-made Extensible Robot* y presentado en la Figura 1, es un robot modular blando de actuación neumática. Se caracteriza por una gran capacidad de extensión, pues puede prácticamente duplicar su altura, y una capacidad de carga más que notable, pues soporta masas de hasta 500 g sin necesidad de modificar su modelo cinemático (García-Samartín et al., 2024b).



(a) Con un módulo (b) Con dos módulos

Figura 1: Manipulador neumático modular PETER.

El robot consta de distintos bloques, cada uno con tres patas extensibles, separadas 120°. Montando únicamente un módulo, por tanto, PETER alcanza tres grados de libertad cartesianos. Montando dos o más, alcanza cinco, puesto que no es posible lograr torsión. Esto dota de redundancia a prácticamente cualquier configuración que se disponga, haciendo que la cinemática inversa no sea un problema trivial. Para las pruebas llevadas a cabo, se han empleado dos módulos.

A nivel constructivo, cada pata consta de una estructura de TPU con forma de fuelle, y de unas cámaras de caucho,

que son las que hinchan. Esta combinación de materiales permite combinar una alta capacidad de extensión, dada por la geometría de fuelle, con un hinchado libre de pinchazos accidentales. Hinchar directamente el TPU hubiese conducido a menor capacidad de extensión, pues hubiese sido necesario hacer paredes más gruesas para evitar fugas, perdiéndose flexibilidad. El hinchado de las cámaras se realiza abriendo y cerrando válvulas durante una cantidad determinada de tiempo, empleando el banco neumático descrito en (García-Samartín et al., 2024d).

2.2. *Modelado cinemático directo*

El modelado cinemático de PETER se realizó, como es habitual en robótica blanda, en dos etapas. En la primera, se estableció el modelo dependiente, que relaciona las variables directamente actuadas (el tiempo de hinchado en este caso) con las variables articulares del robot (la longitud de cada una de sus patas). Posteriormente, el modelo independiente relaciona estas con la posición $\boldsymbol{x} = (x, y, z)$ alcanzada por el punto medio de la plataforma superior. Dada la capacidad de carga de PETER, que es superior al peso de un módulo, no se tiene en cuenta el efecto de los módulos superiores sobre los inferiores. Las ecuaciones aquí presentadas se aplican secuencialmente a cada módulo.

2.2.1. *Modelado dependiente*

La relación entre tiempos de hinchado y longitud de los actuadores se modeló experimentalmente. Para ello, se recogieron 968 muestras del comportamiento del actuador ante distintos tiempos de hinchado. La relación, que puede verse en la Figura 2, se aproximó por mínimos cuadrados a una recta, cuya expresión se muestra en (1):

$$l(t) = 42{,}926 + 0{,}010 \cdot t \tag{1}$$

estando l expresada en mm y t, en ms.

El valor del término independiente de la expresión, 43 mm, es aproximadamente igual al valor real de la longitud de los actuadores de PETER (44 mm), lo que indica la validez de la aproximación.

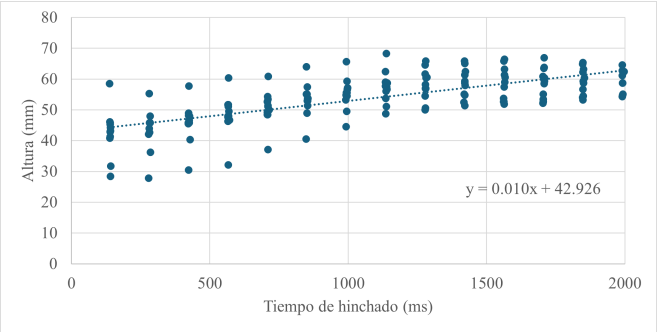


Figura 2: Altura alcanzada por uno de los actuadores de PETER en función del tiempo de hinchado.

2.2.2. *Modelado independiente*

En una segunda fase, se procedió a relacionar la longitud de los actuadores con la posición del centro de la base superior del módulo, que es lo que se ha considerado que determina la posición de esa sección. Para ello, se ha considerado que cada actuador, aún estando dentro del módulo, se expande y se contrae linealmente y son estas deformaciones las que provocan el cambio de orientación y posición de la plataforma superior. Aunque el movimiento de los actuadores es, debido a que están restringidos superior e inferiormente, curvo, al ser el radio de las bases mucho mayor que la altura de los mismos, dicha curva puede aproximarse sin mucha pérdida de precisión a una recta.

Se parte de esta hipótesis y se conoce, para el módulo i, la posición inferior A_{ij} del j-ésimo actuador y su longitud l_{ij}. Se conoce también la dirección de alargamiento del módulo como la perpendicular a la base n_i, que es el vector $(0, 0, 1)$ en la primera iteración y, para el resto de módulos, el calculado, usando (4), al final de la iteración anterior.

Con ello, la posición de su extremo superior B_{ij} viene dada por:

$$B_{ij} = \boldsymbol{A}_{ij} + l_{ij}\boldsymbol{n}_i \tag{2}$$

La posición del centro de su base superior, $\boldsymbol{x}_i = (x_i, y_i, z_i)$ es simplemente el baricentro de sus $\boldsymbol{B}_{ij}$, tal como refleja (3):

$$\boldsymbol{x}_i = \frac{1}{3}\sum_{j=1}^{3} \boldsymbol{B}_{ij} \tag{3}$$

Finalmente, se calcula la posición de los actuadores inferiores siguiente módulo. Para ello, se calcula el vector normal a la base superior, $\boldsymbol{m}_i$, y se desplaza $\boldsymbol{x}_i$ una distancia d, de 17 mm, que se corresponde con la separación entre la base superior del módulo i y la inferior del módulo $i + 1$. Esto se muestra en (4) y (5):

$$\boldsymbol{n}_i = \frac{(\boldsymbol{B}_{i2} - \boldsymbol{B}_{i1}) \times (\boldsymbol{B}_{i3} - \boldsymbol{B}_{i1})}{\|(\boldsymbol{B}_{i2} - \boldsymbol{B}_{i1}) \times (\boldsymbol{B}_{i3} - \boldsymbol{B}_{i1})\|} \tag{4}$$

$$\boldsymbol{A}_{(i+1)j} = \boldsymbol{B}_{ij} + d\boldsymbol{n}_i \tag{5}$$

Finalmente, se hace $\boldsymbol{n}_{i+1} = \boldsymbol{n}_i$.

3. Modelo cinemático inverso desarrollado

3.1. *El algoritmo Trust Region Reflective*

El algoritmo *Trust Region Reflective* (TRR) es una metodología de optimización pensada para resolver problemas no lineales de mínimos cuadrados con restricciones. A diferencia de otros métodos, como el descenso del gradiente o Levenberg-Marquardt, que considera en cada iteración todo el espacio de trabajo, TRR minimiza una aproximación de segundo orden de la función en una región de confianza (Branch et al., 1999).

El tamaño de la misma se ajusta en cada iteración, creciendo si la variación en la función objetivo es alta, para explorar un espacio más amplio de soluciones y disminuyendo si la variación es baja, buscando centrar la búsqueda en un área más pequeña. Si una solución intenta salir del rango de restricciones, el algoritmo la "refleja" en la frontera del conjunto, es decir, aplica una transformación lineal tal que la trayectoria que une dicha solución con la anterior es la que tendría un rayo de luz que hubiese rebotado en los límites del dominio permitido.

El método, además de ser rápido y muy recomendado cuando se manejan restricciones, es robusto ante jacobianos mal condicionados o dispersos. Tanto la Global Optimization Toolbox MATLAB como SciKit de Python tienen implementado este algoritmo.

Sus pasos son:

1. **Definición del problema de optimización:** Dado un conjunto de variables $\boldsymbol{s} = (s_1, \ldots, s_n) \in \mathbb{R}^n$ y una función objetivo $f : \mathbb{R}^n \to \mathbb{R}$ se define el problema de optimización según (6):

$$\min_{\boldsymbol{s}} f(\boldsymbol{s}) \quad \boldsymbol{s} \in \Omega \tag{6}$$

   siendo $\Omega = \{\boldsymbol{s} : l_i \leq s_i \leq u_i\}$ el conjunto impuesto por las restricciones sobre las variables de entrada. Dicho conjunto puede ser cerrado o estar abierto parcial, o incluso, totalmente.
   Desde un punto de vista robótico, las variables $(s_1, \ldots, s_n)$ se corresponden con las coordenadas articulares del manipulador, mientras que la función objetivo es habitualmente una medida de la distancia entre la posición alcanzada y la deseada, incluyendo, si se quisiese hacer un control de forma, la posición de los elementos intermedios.
2. **Condiciones iniciales:** se definen los valores iniciales de las variables, el vector $\boldsymbol{s}_0 = (s_{10}, \ldots, s_{n0})$ y el radio inicial de la región de confianza, Δ_0. Esta será una elipsoide centrada en $\boldsymbol{s}_0$ dada por la expresión $\|\boldsymbol{D}\boldsymbol{s}_0\| \leq \Delta_0$, siendo $\boldsymbol{D}$ una matriz de ponderación. Si $\boldsymbol{D} = \boldsymbol{I}$, la región de confianza es una esfera.
3. **Minimización de la función:** en cada iteración, dada la configuración actual $\boldsymbol{s}_k$, se busca $\boldsymbol{p}_k$ que minimice la aproximación por el desarrollo de Taylor de segundo orden de la función en la región de confianza.
   Matemáticamente, se busca resolver, en la k-ésima iteración, la expresión (7):

$$\min_{\boldsymbol{p}} m_k(\boldsymbol{p}) = f_k + \nabla f_k^T \boldsymbol{p} + \frac{1}{2}\boldsymbol{p}_k^T \boldsymbol{H}_k \boldsymbol{p}_k \quad \left\|\boldsymbol{D}\boldsymbol{p}_k\right\| \leq \Delta_k \tag{7}$$

   siendo $\boldsymbol{H}_k$, la matriz de hessianos en la iteración k.
   Para ello, puede emplearse el descenso del gradiente o cualquier método de minimización. La precisión no tiene por que ser muy alta, por lo que bastan unas pocas iteraciones. La rapidez de este paso, llamado resolución del subproblema, pues implica una optimización dentro del proceso de optimización, es por tanto rápida, máxime teniéndose en cuenta que se está aproximando una función de segundo grado.
4. **Cálculo de la reducción:** el cociente de (8), ρ_k mide cuánta mejora supone el nuevo punto encontrado:

$$\rho_k = \frac{f(\boldsymbol{s}_k) - f(\boldsymbol{s}_k + \boldsymbol{p}_k)}{f(\boldsymbol{s}_k) - m_k(\boldsymbol{p}_k)} \tag{8}$$

   En concreto, ρ_k evalúa cuánto se ha reducido la función objetivo (numerador) frente a la reducción esperada (denominador). Mientras que esta última siempre es positiva, el numerador puede ser positivo o negativo. Valores negativos indican empeoramiento de la solución encontrada.
5. **Actualización del radio de la región de confianza:** en base al valor de ρ_k obtenido en (8), se actualiza el valor de Δ_k. Si ρ_k es pequeño, inferior a cierto valor μ, la mejora ha sido pequeña y el radio de la región de confianza debe reducirse. Si, por el contrario, $\rho_k > \eta$, se incrementa Δ_k. En situaciones intermedias, este valor se mantiene constante.
6. **Actualización del centro de la región:** si $\rho_k > \sigma$, se considera que la función ha encontrado un valor mejor y se actualiza el centro de la región de confianza, $\boldsymbol{s}_{k+1} = \boldsymbol{s}_k$. El algoritmo para cuando $\|\boldsymbol{s}_{k+1} - \boldsymbol{s}_k\|$ es suficientemente pequeño o si se ha cumplido el número máximo de iteraciones.
7. **Transformación reflectiva:** en cada iteración, se comprueba si $bsys_k$ se encuentra dentro de los límites de Ω. Si no es así, se le aplica la transformación lineal descrita en las ecuaciones (9) y (10):

$$\boldsymbol{v} = \quad \text{mód } \{\boldsymbol{s}_k - \boldsymbol{l}, 2 \cdot (\boldsymbol{u} - \boldsymbol{l})\} \tag{9}$$

$$\boldsymbol{s}_{k+1} = \boldsymbol{l} + \text{mín}\{\boldsymbol{v}, 2 \cdot (\boldsymbol{u} - \boldsymbol{l}) - \boldsymbol{v}\} \tag{10}$$

3.2. Implementación del algoritmo TRR sobre PETER

A la hora de implementar el algoritmo sobre el manipulador blando, se empleó la librería SciKit de Python. Las variables de entrada $\boldsymbol{s}$ se corresponden con los tiempos de hinchado de los actuadores del manipulador, $\boldsymbol{s} = (t_1, \ldots, t_6)$, mientras que la función objetivo es la distancia entre la posición deseada $\boldsymbol{x}_d$ y el valor devuelto por el modelo cinemático directo, $\boldsymbol{x}_m$. Esto viene dado por:

$$f(\boldsymbol{x}) = \|\boldsymbol{x}_d - \boldsymbol{x}_m(\boldsymbol{t})\| \tag{11}$$

Si se quisiera condicionar la forma del robot se podría, como se hace en (García-Samartín et al., 2024a), añadir un segundo término a (11) que midiese la distancia entre las posiciones deseadas en los módulos intermedios y las realmente alcanzadas, ponderado por un factor de importancia, quedando la función objetivo como $f(\boldsymbol{x}) = \|\boldsymbol{x}_d - \boldsymbol{x}_m(\boldsymbol{t})\| + \alpha \left\|\boldsymbol{x}_d^{int} - \boldsymbol{x}_m^{int}(\boldsymbol{t})\right\|$.

Como restricciones, se impuso en todos los actuadores un tiempo de hinchado mínimo de 0 ms y uno máximo de 2000 ms, para evitar roturas debidas a una presión excesiva. SI el robot está realizando una trayectoria, se toma como punto de partida el punto anterior. En caso contrario, la posición en la que todas las válvulas están completamente deshinchadas. Se estableció en 2000 el número máximo de iteraciones. Como parámetros del algoritmo, se tomaron $\mu = \frac{1}{4}$, $\eta = \frac{3}{4}$ y $\sigma = 10^{-4}$.

3.3. Implementación de modelo cinemático por aprendizaje

Con el objetivo de contrastar los resultados obtenidos con otros modelos existentes, se desarrolló también un modelo basado en una red neuronal de tipo *feedforward*. Se optó por un método de aprendizaje debido a su amplia capacidad de generalización, empleándose redes prealimentadas por ser las que menos cantidad de datos necesitan y que ofrecen buenos resultados en manipuladores relativamente sencillos, como PETER.

La red constó de 4 capas, de 256, 128, 64 y 32 neuronas, respectivamente, cada una de ellas con función de activación GeLU (*Gaussian Error Linear Unit*) y regularización L^2 para

prevenir el sobreajuste. Del mismo modo, con el fin de aumentar las capacidades de generalización del modelo, se incluyó una capa de *dropout* con una tasa del 30 % entre las capas densas. El modelo se compiló con el optimizador Nadam y se empleó el error cuadrático medio (MSE) como función de pérdida.

Se utilizaron para el entrenamiento 10000 datos, generados a partir del modelo cinemático, reservándose 15 % para validación interna y un 10 % como conjunto de prueba. Se estableció un máximo de 250 épocas, con tamaños de lote de 32 muestras. El entrenamiento paraba también si durante 20 épocas no había mejoría en el valor de la función de pérdida.

4. Experimentos y Resultados

Los experimentos se llevaron a cabo en dos fases. En primer lugar, se validaron ambos algoritmos sobre un simulador del robot, con el objetivo de ajustar sus parámetros de forma rápida y desprovista de riesgos de pinchazos o fugas por un uso excesivo del manipulador. Posteriormente, se probaron ambos algoritmos sobre el propio PETER.

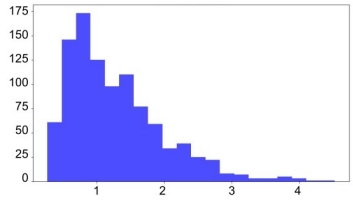
4.1. Pruebas en simulación

Las pruebas se desarrollaron sobre un simulador previamente desarrollado. Dado que este no tiene limitaciones físicas, se evaluó el comportamiento de los algoritmos en dispositivos de hasta cuatro módulos.

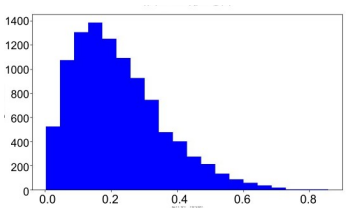
Se mandó el robot a un total de 1000 posiciones aleatorias dentro de su campo de trabajo, en situaciones de 2, 3 y 4 módulos. Los resultados se muestran en la Tabla 1 y la Figura 3.

Tabla 1: Estadísticas de error y tiempos de ejecución de los dos modelos desarrollados en el artículo.

Algoritmo	Redes Neuronales	TRR
Error medio (mm)	1.27	0.22
Error mediano (mm)	1.10	0.19
Error máximo (mm)	4.5	1.11
Tiempo promedio (ms)	92	87



(a) Red Neuronal



(b) Algoritmo TRR

Figura 3: Errores (en mm) de los dos modelos desarrollados en el artículo.

En lo que a duración de la ejecución se refiere, ambos modelos tardan prácticamente lo mismo en dar una respuesta, y lo hacen en tiempos suficientemente cortos para que sea viable emplearlos en tareas como la planificación de trayectorias en tiempo real. Esto es muy meritorio para el algoritmo TRR, pues, como se ha comentado, los algoritmos de optimización pueden llegar a ser muy lentos.

Las diferencias surgen, sin embargo, cuando se comparan sus precisiones. Si bien ambos dan valores dentro de rangos aceptables para lo que es la robótica blanda, el TRR es casi diez veces más preciso. Unido a ello, a la vista de la gráfica se observa cómo acumula una menor cantidad de elementos con errores altos, lo que garantiza una mejor fiabilidad en cualquier situación.

4.2. Validación sobre el robot real

Finalmente, se llevó a cabo una validación sobre el propio PETER, equipado con dos módulos. Se llevó el manipulador a cinco posiciones distintas de su espacio de trabajo y, mediante un sistema de cámaras infrarrojas, se capturó la posición de su base superior. Los resultados se muestran en la Figura 4.

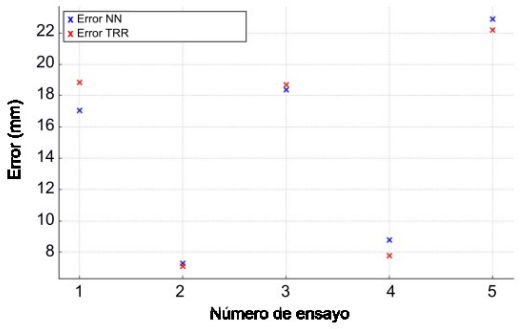


Figura 4: Errores (en mm) de los dos modelos desarrollados en el artículo al ser validados sobre PETER.

La principal diferencia que surge al comparar con los resultados obtenidos en el simulador es el error mucho mayor existente, de 14 mm para el TRR y de 15 mm para la red. Sin embargo, se aprecia cómo ambos modelos presentan peores resultados en los mismos puntos, lo cual puede indicar que el modelo cinemático desarrollado no es del todo preciso. Esto puede deberse a que la suposición de movimiento rectilíneo de los actuadores no es adecuada o a la existencia de pinchazos o fugas dentro del robot. Debe tenerse en cuenta, además, que el sistema se encuentra en cadena abierta.

Entrando a comparar los algoritmos, aunque las diferencias son menores que las observadas en el simulador, sí se puede decir que el algoritmo TRR mejora claramente o, al menos, iguala el rendimiento de la red en todas las situaciones, exceptuando la primera. Esto da idea de que, con un mejor modelo del robot, este algoritmo puede perfectamente modelar en tiempo real robots blandos modulares con elevada redundancia.

5. Conclusiones

En este artículo se ha presentado una metodología para el modelado cinemático inverso de robots blandos. Esta se basa en técnicas de optimización, pues aseguran encontrar una solución, no necesitan un entrenamiento muchas veces difícil de lograr y permiten establecer restricciones en el espacio articular, así como condicionar el movimiento de los elementos intermedios.

En concreto, en este trabajo se ha empleado el algoritmo TRR, del que no se conocen usos previos para el modelado de robots blandos. A diferencia de tantas otras metodologías de minimización, es extremadamente rápida y logra resultados muy precisos. En concreto, se ha probado el modelo sobre un simulador del robot blando PETER arrojando, en menos de 100 ms, resultados con errores de 0,22 mm, muy inferiores a los presentados por un modelo cinemático basado en una red neuronal.

Posteriormente, se ha validado el algoritmo sobre el manipulador real. Aunque la precisión alcanzada ha sido menor y las diferencias con la red neuronal, menos significativas, se espera que, con un mejor modelo del robot, puedan lograrse resultados significativos.

Agradecimientos

Este trabajo ha sido realizado en las instalaciones del Centro de Automática y Robótica (UPM-CSIC), en la sede de la ETSI Industriales de la Universidad Politécnica de Madrid, gracias a las "Ayudas para contratos predoctorales para la realización del doctorado con mención internacional en sus escuelas, facultad, centros e institutos de I+D+i", financiadas por el "Programa Propio I+D+i 2022 de la Universidad Politécnica de Madrid" y es parte del proyecto "CollaborativE Search And Rescue robots (CESAR)" (PID2022-142129OB-I00) financiado por MCIN/AEI/10.13039/501100011033 y "FEDER, UE"

Referencias

Aristidou, A., Lasenby, J., 2011. FABRIK: A fast, iterative solver for the Inverse Kinematics problem. Graphical Models 73, 243–260. URL: http://dx.doi.org/10.1016/j.gmod.2011.05.003, doi:10.1016/j.gmod.2011.05.003.

Bern, J.M., Rus, D., 2021. Soft IK with stiffness control. 2021 IEEE 4th International Conference on Soft Robotics, RoboSoft 2021 , 465–471doi:10.1109/RoboSoft51838.2021.9479195.

Bern, J.M., Schnider, Y., Banzet, P., Kumar, N., Coros, S., 2020. Soft Robot Control with a Learned Differentiable Model. 2020 3rd IEEE International Conference on Soft Robotics, RoboSoft 2020 , 417–423doi:10.1109/RoboSoft48309.2020.9116011.

Bhalkikar, A., Lokesh, S., Ashwin, K.P., 2024. Kinematic models for Cable-driven Continuum Robots with multiple segments and varying cable offsets. Mechanism and Machine Theory 200, 105701.

Bianchi, D., Campinoti, G., Comitini, C., Laschi, C., Rizzo, A., Sabatini, A.M., Falotico, E., 2024. SoftSling: A Soft Robotic Arm Control Strategy to Throw Objects with Circular Run-ups. IEEE Robotics and Automation Letters PP, 1–8. doi:10.1109/LRA.2024.3442535.

Blanco, K., Navas, E., Rodriguez-Nieto, D., Emmi, L., Fernandez, R., 2024. Soft Bellow-Based 3D Printed Robot for in-Pipe Inspection Applications. 2024 7th Iberian Robotics Conference, ROBOT 2024 doi:10.1109/ROBOT61475.2024.10797416.

Branch, M.A., Coleman, T.F., Li, Y., 1999. A Subspace, Interior, and Conjugate Gradient Method for Large-Scale Bound-Constrained Minimization Problems. SIAM Journal on Scientific Computing 21, 1–23. URL: https://doi.org/10.1137/S1064827595289108, doi:10.1137/S1064827595289108.

Chiang, S.S., Yang, H., Skorina, E., Onal, C.D., 2021. SLInKi: State Lattice based Inverse Kinematics - A Fast, Accurate, and Flexible IK Solver for Soft Continuum Robot Manipulators, in: IEEE International Conference on Automation Science and Engineering, IEEE. pp. 1871–1877. doi:10.1109/CASE49439.2021.9551686.

García-Samartín, J.F., del Cerro, J., Barrientos, A., 2024a. Inverse kinematic modelling with shape control of a soft robot using genetic algorithms, in: Jornadas de Automática. URL: https://doi.org/10.17979/ja-cea.2024.45.10968, doi:10.17979/ja-cea.2024.45.10968.

García-Samartín, J.F., Charles, M., del Cerro, J., Barrientos, A., 2024b. PETER : a Soft Pneumatic Manipulator with High Resistance and Load Capacity, in: 2024 7th Iberian Robotics Conference (ROBOT), Madrid. pp. 1–6. doi:10.1109/ROBOT61475.2024.10796857.

García-Samartín, J.F., Molina-Gómez, R., Barrientos, A., 2024c. Model-Free Control of a Soft Pneumatic Segment. Biomimetics 9. doi:https://doi.org/10.3390/biomimetics9030127.

García-Samartín, J.F., Rieker, A., Barrientos, A., 2024d. Design, Manufacturing, and Open-Loop Control of a Soft Pneumatic Arm. Actuators 13. URL: https://www.mdpi.com/2076-0825/13/1/36, doi:10.3390/act13010036.

Kaviri, M., Fesharaki, A.J., Sadeghnejad, S., 2023. Soft robotics in medical applications: State of the art, challenges, and recent advances, in: Boubaker, O. (Ed.), Medical and Healthcare Robotics. Academic Press. Medical Robots and Devices: New Developments and Advances. chapter 2, pp. 25–61. URL: https://www.sciencedirect.com/science/article/pii/B9780443184604000093, doi:https://doi.org/10.1016/B978-0-443-18460-4.00009-3.

Keyvanara, M., Goshtasbi, A., Kuling, I.A., 2023. A Geometric Approach towards Inverse Kinematics of Soft Extensible Pneumatic Actuators Intended for Trajectory Tracking. Sensors 23, 1–16. doi:10.3390/s23156882.

Liu, J., Song, Z., Lu, Y., Yang, H., Chen, X., Duo, Y., Chen, B., Kong, S., Shao, Z., Gong, Z., Wang, S., Ding, X., Yu, J., Wen, L., 2024. An Underwater Robotic System with a Soft Continuum Manipulator for Autonomous Aquatic Grasping. IEEE/ASME Transactions on Mechatronics 29, 1007–1018. doi:10.1109/TMECH.2023.3321054.

Martín, A., Barrientos, A., Del Cerro, J., 2018. The natural-CCD algorithm, a novel method to solve the inverse kinematics of hyper-redundant and soft robots. Soft Robotics 5, 242–257. doi:10.1089/soro.2017.0009.

Navas, E., Fernández, R., Sepúlveda, D., Armada, M., Gonzalez-De-santos, P., 2021. Soft grippers for automatic crop harvesting: A review. Sensors 21. doi:10.3390/s21082689.

Seleem, I.A., El-Hussieny, H., Ishii, H., 2023. Imitation-based Path Planning and Nonlinear Model Predictive Control of a Multi-Section Continuum Robots. Journal of Intelligent and Robotic Systems: Theory and Applications 108. doi:10.1007/s10846-023-01811-8.

Seyidoğlu, B., Rafsanjani, A., 2024. A textile origami snake robot for rectilinear locomotion. Device 2. doi:10.1016/j.device.2023.100226.

Shan, Y., Zhao, Y., Wang, H., Dong, L., Pei, C., Jin, Z., Sun, Y., Liu, T., 2024. Variable stiffness soft robotic gripper: design, development, and prospects. Bioinspiration and Biomimetics 1. doi:10.1088/1748-3190/ad0b8c.

Sun, W., Akashi, N., Kuniyoshi, Y., Nakajima, K., 2022. Physics-Informed Recurrent Neural Networks for Soft Pneumatic Actuators. IEEE Robotics and Automation Letters 7, 6862–6869. doi:10.1109/LRA.2022.3178496.

Terrile, S., Miguelañez, J., Barrientos, A., 2021. A Soft Haptic Glove Actuated with Shape Memory Alloy and Flexible Stretch Sensors. Sensors 21. doi:10.3390/s21165278.

Thuruthel, T.G., Falotico, E., Renda, F., Laschi, C., 2019. Model-Based Reinforcement Learning for Closed-Loop Dynamic Control of Soft Robotic Manipulators. IEEE Transactions on Robotics 35, 127–134. doi:10.1109/TRO.2018.2878318.

Wagaa, N., Kallel, H., Mellouli, N., 2023. Analytical and deep learning approaches for solving the inverse kinematic problem of a high degrees of freedom robotic arm. Engineering Applications of Artificial Intelligence 123, 106301. URL: https://doi.org/10.1016/j.engappai.2023.106301, doi:10.1016/j.engappai.2023.106301.

Wan, Z., Sun, Y., Qin, Y., Skorina, E.H., Gasoto, R., Luo, M., Fu, J., Onal, C.D., 2023. Design, Analysis, and Real-Time Simulation of a 3D Soft Robotic Snake. Soft Robotics 10, 258–268. doi:10.1089/soro.2021.0144.

Wu, M., Zhang, Y., Wu, X., Li, Z., Chen, W., Hao, L., 2023. Review of Modeling and Control Methods of Soft Robots Based on Machine Learning. Chinese Control Conference, CCC 2023-July, 4318–4323. doi:10.23919/CCC58697.2023.10240787.