SOLUCIÓN DE LA PLANIFICACIÓN DE MOVIMIENTOS DE UN ROBOT MÓVIL PARALELO BINARIO MEDIANTE CINEMÁTICA INVERSA DE UN ROBOT HIPER-REDUNDANTE EQUIVALENTE

Mario Pérez-Checa, Adrián Peidró*, Luis Miguel Jiménez, Luis Payá, Óscar Reinoso
Grupo de Automatización, Robótica, y Visión por Computador, Universidad Miguel Hernández
03202 Elche (Alicante), Spain
*Autor de correspondencia. Emails: mario.perez08@alu.umh.es, apeidro@umh.es, o.reinoso@umh.es

Resumen

Este artículo presenta la resolución de la planificación de movimientos de un robot móvil paralelo y binario, determinando la secuencia de movimientos discretos necesarios para alcanzar cualquier posición final deseada. Este problema se reformula y resuelve como la cinemática inversa de un manipulador híper-redundante equivalente, formado por concatenación de tantos módulos binarios como posiciones intermedias debe ocupar el robot móvil a lo largo de su trayectoria. Esta reformulación permite aplicar eficientes técnicas de resolución de cinemática inversa de robots binarios híper-redundantes, resolviendo el problema con gran precisión, evitando métodos de fuerza bruta por enumeración exhaustiva que resultan imprácticos por requerir excesivos tiempos de cómputo y recursos de memoria. Se incluyen varios ejemplos en simulación para demostrar la validez y eficiencia del método de cálculo, que además puede aplicarse en tiempo real.

Palabras clave: robot móvil, robot binario, robot paralelo, robot híper-redundante, cinemática inversa

1 INTRODUCCIÓN

Las instalaciones industriales requieren tareas de inspección y mantenimiento periódicos para asegurar su correcto y seguro funcionamiento. Dichas tareas de inspección y mantenimiento suelen conllevar ciertos riesgos, como la caída de altura, el trabajo en zonas de difícil acceso, la presencia de atmósferas tóxicas o explosivas, radiación, etc. Por tanto, son tareas que se prestan para ser realizadas por robots, tanto de forma tele-operada como semi-automatizada, liberando así a los operarios humanos de tales riesgos.

Un tipo de instalaciones industriales de especial peligrosidad son las denominadas como ATEX, que son aquellas con ATmósferas EXplosivas, como pueden ser las plantas de tratamiento de aguas residuales (los gases de digestión o lodos secos son explosivos), talleres de pintura (los vapores y disolventes son explosivos), o las instalaciones de suministro de gas, por citar algunas. Este riesgo de explosión impone restricciones severas sobre el tipo de equipamiento que puede utilizarse en dichas instalaciones para minimizar los riesgos de explosión. En particular, el uso de robots movidos por actuadores eléctricos como servomotores resulta arriesgado, si no prohibitivo, debido a la posibilidad de que una chispa producida en dichos actuadores provoque una explosión. En entornos explosivos como los indicados, el tipo de actuación que resulta más apropiado es el neumático, debido a su menor riesgo de iniciar una explosión que la actuación eléctrica.

Existen diversos robots para inspección basados en actuadores neumáticos, como por ejemplo [1] o [2]. Uno de los principales inconvenientes de los actuadores neumáticos es que, debido a la alta compresibilidad del aire, exhiben un comportamiento altamente no-lineal que complica su control en posiciones intermedias. En otras palabras: resulta mucho más sencillo operar un actuador neumático de forma binaria o todo/nada (donde los pistones neumáticos del robot se extienden totalmente o se retraen totalmente), que controlarlo de forma que el pistón se detenga con precisión y de forma estable en posiciones intermedias de su carrera. Por tanto, los actuadores neumáticos resultan más apropiados cuando se desea operar al robot de forma binaria. La actuación binaria tiene la ventaja de un control mucho más simple, ya que no hay bucle de realimentación, pero presenta el inconveniente de que el robot binario ya no puede posicionarse de forma fina en su espacio de trabajo: queda limitado a ocupar posiciones discretas que sean múltiplos de su carrera de avance. De este modo, los robots binarios no suelen permitir su posicionamiento fino ni ofrecen una elevada maniobrabilidad, lo cual puede limitar su aplicación.

Para solucionar estas limitaciones de los robots binarios, en [9] se propuso un nuevo robot móvil binario basado en un mecanismo paralelo de dos grados de libertad. Dicho robot se llama Xrobin (de robot binario en forma de X), y posee la peculiaridad de poder alternar entre distintas configuraciones correspondientes a un mismo estado de sus actuadores

 https://doi.org/10.17979/spudc.9788497498418.0820

binarios, sin atravesar singularidades. Este fenómeno, denominado "transición no-singular", es posible en algunos robots paralelos denominados "cuspidales", y permite ampliar el rango de trabajo del robot [10]. En el caso particular del robot Xrobin, el resultado de explotar esas transiciones no-singulares es que su espacio de trabajo se vuelve mucho más denso de lo que se esperaría para ser un robot binario [9]. Esto le dota de una elevada precisión de posicionamiento y una elevada capacidad de maniobra pese al hecho de ser binario, solucionando así las mencionadas limitaciones de los robots binarios, y volviéndolo especialmente apto para realizar tareas de inspección en industrias ATEX de forma segura mediante actuadores neumáticos operados en modo todo/nada.

Teniendo todo esto en cuenta, en el presente artículo se aborda el problema de planificación de movimientos del robot binario Xrobin en el plano. Se pretende determinar la secuencia de movimientos discretos a realizar por el robot, teniendo en cuenta su actuación binaria, para alcanzar con la mayor precisión posible una posición de destino en su plano de trabajo. Al tratarse de un robot binario con espacio de trabajo discreto, un primer enfoque para atacar este problema podría ser un método de enumeración exhaustiva, en el que se genera a priori todo el espacio de trabajo discreto del robot, y luego se realiza una búsqueda entre todos los puntos de dicho espacio de trabajo, para elegir el más cercano a la posición deseada. No obstante, esto resulta inviable para el robot Xrobin estudiado en este artículo, ya que su espacio de trabajo puede llegar a estar formado por un número demasiado elevado de puntos, lo cual requiere tiempos prohibitivos de cálculo y reservas de memoria inviables, requiriéndose otro enfoque al problema.

La solución que se ha adoptado en este artículo es la siguiente. El movimiento del robot Xrobin en el plano, visto como robot móvil, puede entenderse, de forma equivalente, como un manipulador híper-redundante, de modo que dicho manipulador híper-redundante se forma concatenando en serie varios robots Xrobin. Cada módulo Xrobin de este manipulador híper-redundante coincide con cada una de las posiciones intermedias que ocuparía el robot móvil en su travesía por el plano hasta el punto de destino. De este modo, el problema de planificación de movimientos del robot móvil Xrobin se reformula como la cinemática inversa de un manipulador híper-redundante binario, permitiendo así explotar soluciones disponibles en la literatura para resolver su cinemática inversa.

Los robots híper-redundantes formados concatenando robots paralelos, sobre todo con actuadores binarios, han sido estudiados ampliamente en la literatura. En [7] se analizan robots híper-redundantes binarios formados concatenando robots paralelos planos de tipo 3RPR, mientras que en [8] se concatenaban robots espaciales de tipo 3RPS. En [3] se aborda el diseño óptimo de este tipo de robots. La cinemática inversa de los robots híper-redundantes es un problema complejo que se ha abordado mediante métodos de splines [15], divide-y-vencerás [12], el método modal [4], o métodos basados en la densidad del espacio de trabajo [5]. Otros trabajos han calculado el espacio de trabajo de este tipo de robots mediante métodos recursivos hacia atrás [6] o métodos de difusión [13].

Para resolver la planificación de trayectorias del robot móvil binario Xrobin, en este artículo se adoptará el método descrito en [5], que hace uso de la densidad del espacio de trabajo [6]. La elección de este método se debe a su generalidad y sencillez, pues únicamente necesita generar mallas de densidad del espacio de trabajo para ir bloqueando, uno a uno, módulos del robot híper-redundante en aquellas configuraciones que maximicen la densidad para la posición deseada.

Este artículo está organizado como sigue. En la Sección 2 se describe el robot móvil binario Xrobin. La Sección 3 reformula la planificación de movimientos de este robot como el problema cinemático inverso de un robot híper-redundante equivalente, y adapta el método presentado en [5] para resolver dicho problema. La Sección 4 muestra varias simulaciones que demuestran la eficacia del método. Por último, la Sección 5 presenta las conclusiones y sugiere trabajos futuros.

2 EL ROBOT XROBIN

Esta sección presenta al robot Xrobin, que es un robot móvil con estructura paralela de tipo 2RPR-PR y con dos actuadores binarios, preferentemente de tipo neumático. Este robot, mostrado en la Figura 1, se encuentra bajo protección por patente [11].

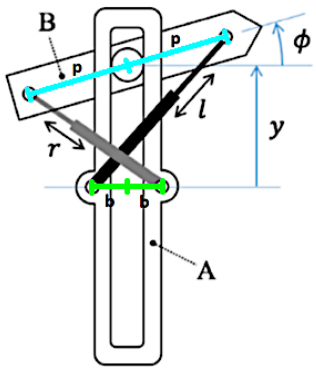

Figura 1: Representación esquemática de Xrobin

Este robot tiene dos cuerpos principales (A y B) que se mueven uno respecto al otro siguiendo la ranura definida en el cuerpo A, y lo que controla dicho movimiento son los pistones binarios *l* y *r*. Como

ilustra la Figura 1, dichos pistones se han dispuesto de forma cruzada (de ahí la X del nombre del robot), lo que le confiere una gran movilidad con tan solo dos actuadores binarios [9]. Por otra parte, b y p son parámetros de diseño del robot, mientras que ϕ e y definen la orientación y posición de B respecto a A. En este artículo: b = 18.59 mm y p = 101.31 mm.

Que este robot sea capaz de alcanzar una movilidad elevada estando gobernado solo por dos pistones binarios se debe a la disposición de dichos pistones en forma de X, la cual le permite alcanzar 8 configuraciones diferentes, las cuales se representan en la Figura 2. Normalmente, un robot movido por n actuadores binarios puede alcanzar 2^n configuraciones distintas. De este modo, un robot con n=2 actuadores binarios como Xrobin, debería alcanzar solo 2^2=4 configuraciones. Sin embargo, el robot Xrobin, al ensamblarlo con sus pistones cruzados, puede realizar transiciones entre distintas soluciones de la cinemática directa para un mismo valor de sus actuadores sin atravesar singularidades [10]. Esto implica que el número de configuraciones distintas alcanzables usando solo dos pistones binarios se duplica, pudiendo alcanzar 8 configuraciones, las cuales se muestran en la Figura 2 como ocho puntos 11U, 01V, 00V… en el plano (ϕ, y), donde la notación de cada punto representa lo siguiente: los dos primeros números indican el estado de sus dos actuadores binarios, siendo "0" un actuador totalmente retraído, y "1" un actuador totalmente extendido, mientras que U o V indican cuál de las dos soluciones simétricas de la cinemática directa se elige, para ese estado de los actuadores (solución U o V). La postura que adoptaría el robot para cada uno de esos ocho puntos se representa gráficamente en un pequeño recuadro con bordes a trazos, identificando dichas configuraciones mediante los números del 1 al 8 en la Figura 2.

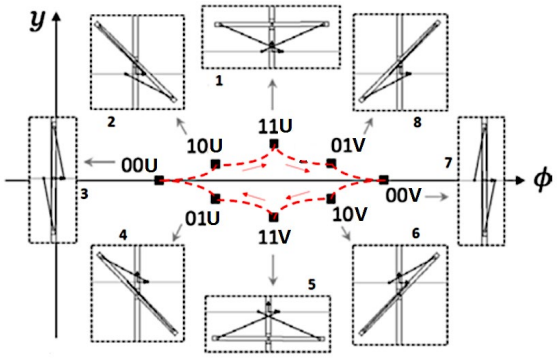

Figura 2: Configuraciones de Xrobin

Partiendo de las configuraciones definidas en la Figura 2, el robot Xrobin es capaz de desplazarse por el plano empleando el siguiente modo de locomoción. Cada uno de los cuerpos del robot (A y B) puede pegarse o despegarse del plano de movimiento de forma independiente al otro. Pegando solo uno de los cuerpos al plano, y accionando los actuadores, es posible desplazar el otro cuerpo a un nuevo punto de sujeción, adoptando el robot una de las ocho posibles configuraciones mostradas en la Figura 2, permitiendo así una locomoción de tipo oruga. Por ejemplo: en primer lugar, se fija el cuerpo A y se mueve B hasta la configuración deseada; a continuación, se fija B y se mueve A hasta la configuración deseada; y finalmente, se vuelve a fijar A para iniciar otro ciclo. La secuencia que se acaba de describir es un ciclo completo de movimiento del robot. Cada una de las mitades de este ciclo, se considera como un semiciclo: el primer semiciclo es aquél que mantiene fijo el cuerpo A y mueve el cuerpo B, mientras que el segundo semiciclo es el que mantiene fijo el cuerpo B y mueve el cuerpo A hasta volver a fijarlo y así completar un ciclo de movimiento. Para comprender los resultados que se expondrán en la Sección 4 de este artículo, es importante tener claro que, dentro de cada semiciclo, el robot adoptará una de las ocho configuraciones denotadas con números del 1 al 8 en la Figura 2. El objetivo será determinar cuál de éstas configuraciones deberá ser adoptada por el robot dentro de cada semiciclo para lograr alcanzar la posición deseada.

El método de locomoción de tipo oruga que se acaba de describir, permite al robot moverse a lo largo de un plano. Sin embargo, dado que los actuadores son binarios, el espacio de trabajo del robot (definido como el conjunto de posiciones que ocuparía el centro del cuerpo A tras N ciclos de movimiento como los descritos) es discreto. Por ejemplo, la Figura 3 muestra una nube de puntos que representa el espacio de trabajo del robot tras N=2 ciclos de movimiento, es decir, representa las posibles posiciones que podría ocupar el centro del cuerpo A tras dos ciclos, partiendo del origen. Este espacio de trabajo puede generarse de forma recursiva para un número N cualquiera de ciclos mediante el algoritmo descrito en [9].

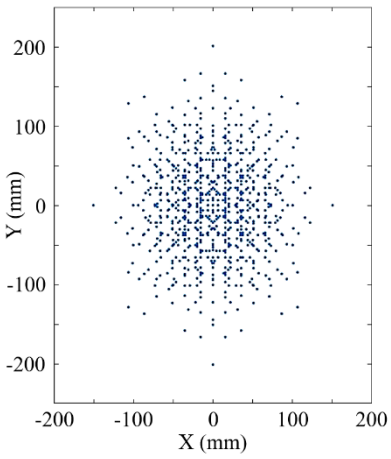

Figura 3: Espacio de trabajo para N=2 ciclos

El problema que abordamos en este artículo es la planificación de movimientos del robot Xrobin, para

https://doi.org/10.17979/spudc.9788497498418.0820

llevarlo desde el origen hasta una posición final deseada con la suficiente precisión. Es decir, determinaremos la configuración a adoptar por el robot dentro de cada semiciclo de movimiento para alcanzar la posición deseada. Dado que el espacio de trabajo de este robot es discreto, el método más directo para resolver este problema podría ser uno de fuerza bruta: primero se realiza una enumeración exhaustiva de todas las posiciones alcanzables (mediante el método de generación del espacio de trabajo descrito en [9]), y luego se realiza una búsqueda de aquella posición alcanzable más cercana a la deseada, de entre todas las generadas. Para posiciones deseadas que sean cercanas al origen, este enfoque es viable ya que, como muestra la Figura 3, las posiciones cercanas al origen (en un radio de aproximadamente 100 mm) pueden alcanzarse con una precisión decente tras únicamente N=2 ciclos. No obstante, para alcanzar puntos más lejanos, el robot tendrá que ejecutar más ciclos de movimiento, lo cual provocará un aumento exponencial del número de posiciones alcanzables y desbordará tanto la memoria del ordenador como los tiempos de ejecución, haciendo que este enfoque sea inviable, como se demostrará en los siguientes párrafos. De hecho, el número de puntos que integran dicha nube es $64 + 64^2 + \ldots + 64^N$, siendo N el número de ciclos, por lo que el aumento exponencial de la cantidad de puntos hace imposible tanto el almacenamiento de los mismos como la búsqueda de una solución entre ellos mediante un método basado en fuerza bruta (generación de todas las soluciones y búsqueda de la mejor entre ellas). Es decir, resulta problemático resolver la cinemática inversa para muchos ciclos de movimiento (más concretamente, 4 o más ciclos) mediante el enfoque de fuerza bruta.

A continuación, se mostrarán dos tablas que muestran la inviabilidad de dicho enfoque. La Tabla 1 muestra el aumento exponencial de tiempo de cómputo que necesita el método de fuerza bruta para obtener el espacio de trabajo, con N ciclos entre 1 y 4. Estos tiempos han sido medidos en Matlab R2018a, en Windows 10(64bit) con un procesador Intel(R) Core(TM) i7-8750H 2.20GHz y 16GB de RAM. Así, se puede comprobar que para 5 o más ciclos, el tiempo de cómputo es inviable, máxime si se desea resolver la planificación de movimientos en tiempo real.

Tabla 1: Tiempos de cómputo para N entre 1 y 4.

N (ciclos)	Tiempo de cómputo (s)
1	0.002
2	0.085
3	5.432
4	352.634

Por otra parte, la Tabla 2 muestra el tamaño de reserva de memoria que sería necesario para almacenar los puntos del espacio de trabajo para N ciclos entre 3 y 6. Como puede apreciarse, el aumento de la memoria necesaria también es exponencial, y de 5 ciclos en adelante ya resulta inviable, ya que pocos ordenadores comunes tienen más de 24 GB de memoria RAM.

Tabla 2: Memoria necesaria para la reserva.

N (ciclos)	Núm. de puntos generados	Reserva de memoria (GB)
3	266 304	0.00595236
4	17 043 520	0.380952
5	$1.0908 \cdot 10^9$	24.4
6	$6.9810 \cdot 10^{10}$	1 560.4

En resumen, la combinación entre elevados tiempos de cómputo y la necesidad de reserva excesiva de memoria para almacenar toda la nube de puntos, hacen que el algoritmo de fuerza bruta sea inviable a partir de N = 4 ciclos de movimiento. Es por ello que surge la necesidad de desarrollar un método más práctico, viable y eficiente para resolver la planificación de movimientos de este robot móvil binario.

3 PLANIFICACIÓN DE MOVIMIENTOS

En esta sección se abordará la planificación de movimientos del robot Xrobin descrito en la sección anterior, que consiste en calcular la secuencia de configuraciones a adoptar por el robot en cada semiciclo de movimiento, para alcanzar su destino.

A pesar de que la solución buscada se va a aplicar a un robot móvil con dos actuadores binarios, como ya se ha comentado anteriormente, dicha solución se basará en un algoritmo [5] diseñado para resolver la cinemática inversa de manipuladores híper-redundantes binarios. Para ello, se va a formular el problema de planificación de trayectorias del robot móvil Xrobin como la cinemática inversa de un manipulador híper-redundante binario, obtenido concatenando varios módulos binarios idénticos, de modo que dichos módulos coincidan con las posiciones intermedias que irá ocupando el robot móvil a lo largo de su movimiento.

En primer lugar es necesario transformar el problema de planificación de trayectorias del robot móvil Xrobin en la cinemática inversa de un robot híper-redundante equivalente. La transformación del robot móvil Xrobin en un robot híper-redundante se realiza de la siguiente manera. Cada módulo del robot híper-redundante está formado por dos robots móviles

https://doi.org/10.17979/spudc.9788497498418.0820

Xrobin conectados en serie, de forma que comparten el cuerpo B, tal y como se puede apreciar en la Figura 4. La justificación de que cada módulo esté formado por dos robots Xrobin conectados en serie, es que, de este modo, cada módulo equivaldrá a un ciclo completo de locomoción del robot Xrobin, ya que un ciclo comienza moviendo el cuerpo B estando fijo A, y termina moviendo el cuerpo A estando fijo B. De este modo, un robot híper-redundante de N módulos equivaldrá a la locomoción de un robot móvil Xrobin que realiza N ciclos de movimiento.

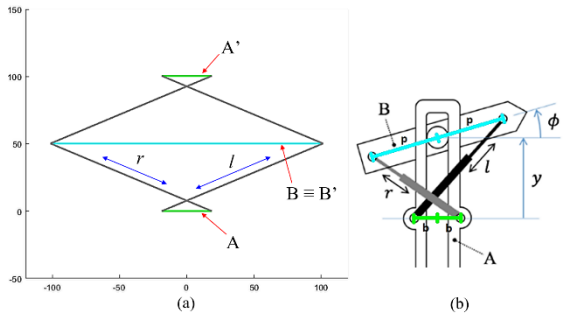

Figura 4: (a) Representación esquemática de un módulo individual del robot híper-redundante equivalente. Dicho módulo está formado por dos robots Xrobin conectados en serie que comparten el cuerpo B. (b) Representación detallada del robot Xrobin, para facilitar su comparación con (a).

La unión de varios módulos como el de la Figura 4(a) da como resultado un manipulador híper-redundante como el mostrado en la Figura 5, donde los cuerpos A se muestran en color verde lima, mientras que los cuerpos B se muestran en color azul cian. Este código de colores para representar los cuerpos del robot Xrobin se ha utilizado en todo el artículo.

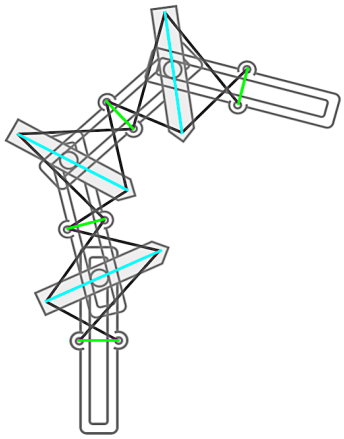

Figura 5: Robot híper-redundante de 3 módulos

En las siguientes subsecciones se describen las operaciones a realizar para resolver la cinemática inversa del robot híper-redundante equivalente.

3.1 CINEMÁTICA INVERSA DE UN ROBOT BINARIO HIPER-REDUNDANTE

Como se comentó en la Sección 2, la enumeración exhaustiva de todas las posibles posiciones alcanzables por un robot móvil binario tras N ciclos de movimiento resulta inviable si N es demasiado grande. Un problema similar se encontró en trabajos previos [5,6] al abordar la resolución de la cinemática inversa de robots híper-redundantes binarios mediante métodos de fuerza bruta, siendo necesario desarrollar métodos más computacionalmente eficientes, como [5]. En esta sección, aplicaremos el método [5] al robot híper-redundante de N módulos, que equivale a realizar N ciclos de movimiento con el robot móvil Xrobin. A continuación, se resumirá el método propuesto en [5], incluyendo alguna modificación introducida en el presente artículo.

El método propuesto en [5] se basa en la siguiente idea. En lugar de realizar una búsqueda exhaustiva variando todos los módulos del robot a la vez, se va fijando cada módulo en su configuración más "favorable", empezando por el módulo más cercano a la base del robot híper-redundante, y terminando en el más lejano. Comenzando por el primer módulo, se obtienen todas sus posibles configuraciones discretas teniendo en cuenta que sus actuadores son binarios. Para robots híper-redundantes como el de la Figura 5, cada módulo está formado por dos robots Xrobin conectados en serie compartiendo el cuerpo B. Como cada robot Xrobin puede adoptar 8 configuraciones distintas, cada módulo de nuestro robot híper-redundante podrá adoptar 8^2=64 configuraciones distintas. Por tanto, será necesario fijar el primer módulo del robot híper-redundante en cada una de esas posibles 64 configuraciones.

Fijando el primer módulo en cada una de esas 64 configuraciones, se superpone al final de dicho módulo el espacio de trabajo del sub-robot híper-redundante (formado por N-1 módulos) que quedaría después del primer módulo ya fijado, una vez excluido éste. Para que este método resulte eficiente, dicho espacio de trabajo no debe estar almacenado como una nube de puntos que contenga la totalidad de los puntos alcanzables por dicho sub-robot, porque eso seguiría exhibiendo el problema de tener que calcular y almacenar una elevadísima cantidad de puntos, que es el principal problema de los métodos de fuerza bruta que se desea evitar. Al contrario, el espacio de trabajo que se utiliza en este punto del método es una malla, con la resolución que se desee, que únicamente almacena el número (o densidad) de puntos que caerían dentro de cada celda de dicha malla. A esta malla, que puede calcularse mediante el método descrito en [6], la llamaremos "malla de trabajo" para distinguirla del "espacio de trabajo" convencional. De este modo, superponiendo esa malla al final del primer

https://doi.org/10.17979/spudc.9788497498418.0820

módulo fijado (en cada una de sus posibles 64 configuraciones), dicho módulo se deberá fijar en la configuración que maximice la densidad de puntos de la celda en la que caiga la posición de destino deseada. Este proceso se ilustra en la Figura 6.

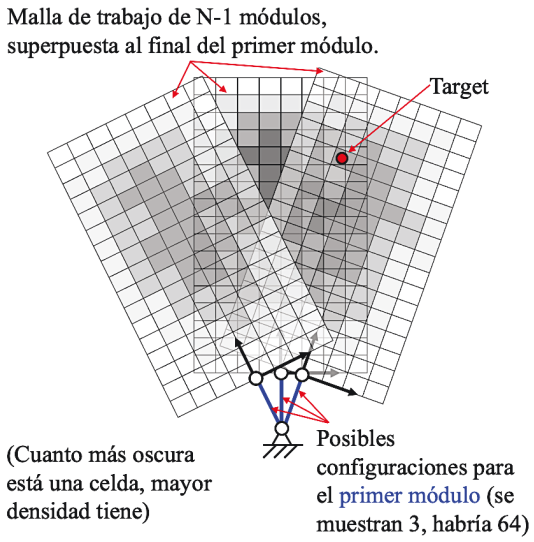

Figura 6: Malla de trabajo superpuesta al final del primer módulo, para todas las posibles configuraciones del primer módulo. En este ejemplo se fijaría la configuración situada más a la derecha, porque el target cae en una celda de mayor densidad que para las otras configuraciones.

Como modificación del algoritmo original presentado en [5], cabe destacar que, en este paso, fue necesario incluir la siguiente modificación. Cuando hay varias configuraciones distintas (de entre las 64 posibles en las que se puede fijar el primer módulo) que posicionan la malla de trabajo de forma tal que el punto deseado cae en una celda con igual máxima densidad, se debe elegir la configuración que minimice la distancia entre el extremo final del módulo fijado y la posición deseada de destino. Esta modificación fue necesaria porque, realizando experimentos, se comprobó que, si se fijaba el módulo en una configuración cualquiera de entre todas las que generan la misma densidad máxima, a veces ocurría que el robot no era capaz de alcanzar la posición final, a pesar de caer dentro de su alcance. Esta situación resulta bastante frecuente en nuestro robot, dada la simetría de su espacio de trabajo (ver la Figura 3).

Para terminar, este proceso se debe repetir de forma iterativa "hacia delante" en el robot híper-redundante. Es decir, una vez se ha elegido la configuración óptima en la que se fija el primer módulo para maximizar la densidad de la celda en la que cae el punto deseado, se procede a repetir el proceso con el siguiente módulo, así sucesivamente hasta que se fija el último módulo.

3.2 APLICACIÓN DE LA SOLUCIÓN AL ROBOT MÓVIL

Una vez resuelta la cinemática inversa del manipulador híper-redundante, la planificación de trayectorias del robot móvil se puede obtener de forma directa a partir de las configuraciones en las que se han fijado los sucesivos módulos de dicho manipulador.

El resultado obtenido del método descrito en la sección anterior es un vector de enteros que identifican las configuraciones que debe adoptar el robot en cada semiciclo de movimiento, de entre las ocho configuraciones ilustradas en la Figura 2. Dicho vector está compuesto por $2N$ valores (siendo N el número de ciclos de movimiento), de los cuales los primeros N valores corresponden con todos los primeros semiciclos de cada ciclo, y de N+1 hasta $2N$ son los segundos semiciclos. Esto se ilustrará mediante varios ejemplos en la siguiente sección.

4 RESULTADOS

A continuación, se mostrarán tres ejemplos de aplicación del algoritmo descrito. Los resultados se mostrarán como vector de configuraciones, y también se representará gráficamente el espacio de trabajo y el manipulador híper-redundante con todos sus módulos. Todos los ejemplos que se mostrarán a continuación son para un número de ciclos (o de módulos binarios) de N=7. En todos los ejemplos, el robot parte del origen (0,0).

El primer ejemplo se muestra en la Figura 7, y éste se ha obtenido imponiendo como objetivo la posición (400, 400) mm.

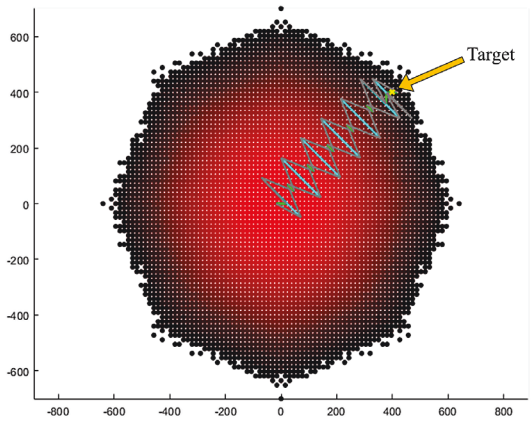

Figura 7: Resultado para target en (400, 400) mm

El vector de configuraciones obtenido como resultado de aplicar el método descrito en la Sección 3 es el siguiente: S = [2 1 1 1 1 1 8, 5 5 5 5 5 6 4]. Este vector se divide en dos subvectores mediante una coma, tal

https://doi.org/10.17979/spudc.9788497498418.0820

que S = [S_1, S_2], siendo: S_1 = [2 1 1 1 1 1 8], y S_2 = [5 5 5 5 5 6 4]. La interpretación de dichos vectores es la siguiente: el vector S_1 representa las configuraciones de la Figura 2 adoptadas en los primeros semiciclos de cada ciclo, y el vector S_2 representa las configuraciones adoptadas durante los segundos semiciclos. Así, se determina que, para alcanzar el objetivo, el robot debe adoptar primero la configuración 2 (primer valor de S_1), y después la 5 (primer valor de S_2). El siguiente paso sería adoptar la configuración 1, después la 5, y así sucesivamente hasta alcanzar el último valor de S_2 (configuración 4).

El segundo ejemplo se muestra en la Figura 8, y en éste, el objetivo se ha definido en (200, 200) mm. El vector de configuraciones obtenido es: S = [2 1 1 7 1 1 1, 5 5 5 2 1 1 1].

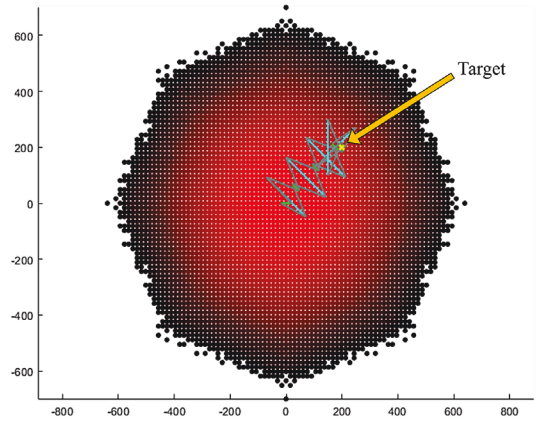

Figura 8: Resultado para target en (200, 200) mm

Para el tercer y último ejemplo, mostrado en la Figura 9, el target está en (1000, 1000) mm, que está fuera del espacio de trabajo alcanzable con N=7 ciclos. El resultado es: S = [1 1 2 1 1 1 1, 5 5 5 5 5 5 5].

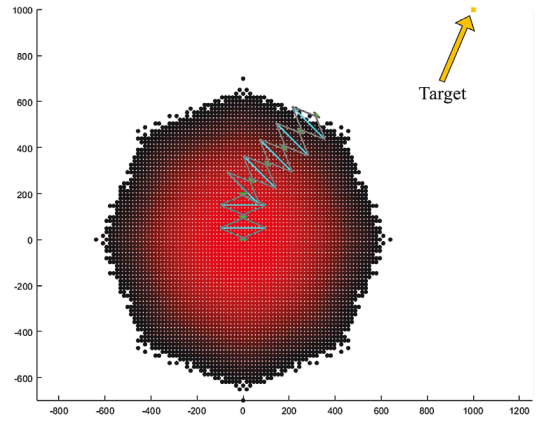

Figura 9: Resultado para target en (1000, 1000) mm

Como se observa en la Figura 8, existen situaciones en las que el robot necesita menos ciclos de movimiento que los establecidos (N=7) para llegar al objetivo. En esas situaciones, una vez el robot híper-redundante ha alcanzado el objetivo, el robot pliega los módulos uno sobre otro las veces que necesite para no moverse del objetivo ya alcanzado. Esta situación puede resolverse de forma sencilla interrumpiendo el algoritmo cuando el robot ya haya alcanzado su objetivo, evitando así movimientos innecesarios. También es destacable que, si el objetivo está fuera del espacio de trabajo (Figura 9), el robot intenta igualmente dirigirse hacia él. Partiendo de la posición final alcanzada, podría tomarse ésta como nuevo origen para ejecutar otra vez el algoritmo manteniendo N=7 ciclos, y solapar un nuevo espacio de trabajo en el punto donde terminaba el anterior, de modo que el nuevo espacio de trabajo sí contenga al target deseado.

Para interpretar correctamente las figuras anteriores, se debe resaltar una vez más que el robot sobre el que se está aplicando este método es un robot móvil con dos actuadores binarios, no un manipulador híper-redundante. De hecho, en el siguiente video puede verse la simulación del movimiento del robot móvil a lo largo del manipulador híper-redundante del ejemplo de la Figura 7, mostrando cómo el robot móvil va recorriendo todos los módulos de dicho manipulador hasta alcanzar la posición final deseada:
https://youtu.be/zTh4TrHcWiw

Por último, destacar que el tiempo medio de ejecución de los anteriores ejemplos es de 1.5 segundos, lo que sugiere la posible utilización de este método en tiempo real, al contrario que los métodos de fuerza bruta.

5 CONCLUSIONES Y TRABAJOS FUTUROS

En este artículo hemos resuelto la planificación de movimientos de un robot móvil paralelo que dispone de un espacio de trabajo muy denso, a pesar de estar gobernado únicamente por dos pistones binarios. El gran volumen de puntos de su espacio de trabajo impide un enfoque de fuerza bruta, por lo que el problema se ha resuelto reformulándolo como la cinemática inversa de un robot binario híper-redundante, para el cual existen métodos eficientes. La aplicación de dichos métodos a nuestro robot ha demostrado resolver satisfactoriamente el problema y en bajos tiempos, que permiten su implementación en tiempo real. En el futuro, incorporaremos la evasión de obstáculos al problema, y lo ensayaremos sobre un prototipo real como el presentado en [9].

Agradecimientos

Este trabajo es parte del proyecto PID2020-116418RB-I00 financiado por MCIN/AEI/ 10.13039/501100011033.

https://doi.org/10.17979/spudc.9788497498418.0820

English summary

SOLUTION OF MOTION PLANNING OF A BINARY PARALLEL MOBILE ROBOT THROUGH INVERSE KINEMATICS OF AN EQUIVALENT HYPER-REDUNDANT ROBOT

Abstract

This paper presents the solution of motion planning of a parallel binary mobile robot, in order to find the sequence of movements necessary to reach any desired target. This problem is reformulated as the inverse kinematics of an equivalent hyper-redundant manipulator built by stacking as many binary modules as the number of intermediate positions that the mobile robot must traverse along its trajectory. This allows us to apply efficient techniques for the inverse kinematics of binary hyper-redundant robots, solving the problem accurately, avoiding brute-force methods that enumerate all configurations and require long computation times and large memory allocation. The simulations show the validity and efficiency of the method, which can be used in real time.

Keywords: mobile robot, binary robot, parallel robot, hyper-redundant robot, inverse kinematics

Referencias

[1] La Rosa, G., Messina, M., Muscato, G., & Sinatra, R. (2002). A low-cost lightweight climbing robot for the inspection of vertical surfaces. Mechatronics, 12(1), 71-96.

[2] Chen, I. M., & Yeo, S. H. (2003). Locomotion of a two-dimensional walking-climbing robot using a closed-loop mechanism. The Intl. J. of Robotics Research, 22(1), 21-40.

[3] Chirikjian, G. S. (1994). A binary paradigm for robotic manipulators. In Proceedings of the 1994 IEEE Intl. Conf. on Robotics and Automation (pp. 3063-3069). IEEE.

[4] Chirikjian, G. S., & Burdick, J. W. (1994). A modal approach to hyper-redundant manipulator kinematics. IEEE Transactions on Robotics and Automation, 10(3), 343-354.

[5] Ebert-Uphoff, I., & Chirikjian, G. S. (1996). Inverse kinematics of discretely actuated hyper-redundant manipulators using workspace densities. In Proceedings of IEEE Intl. Conf. on Robotics and Automation (Vol. 1, pp. 139-145).

[6] Ebert-uphoff, I., & Chirikjian, G. S. (1995). Efficient workspace generation for binary manipulators with many actuators. Journal of Robotic Systems, 12(6), 383-400.

[7] Lees, D. S., & Chirikjian, G. S. (1996). An efficient method for computing the forward kinematics of binary manipulators. In Proceedings of IEEE Intl. Conf. on Robotics and Automation (Vol. 2, pp. 1012-1017). IEEE.

[8] Maeda, K., & Konaka, E. (2014). Ellipsoidal outer-approximation of workspace of binary manipulator for inverse kinematics solution. In 2014 IEEE/ASME Intl. Conf. on Advanced Intelligent Mechatronics (pp. 1331-1336). IEEE.

[9] Peidró, A., García-Martínez, A., Marín, J.M., Payá, L., Gil, A., & Reinoso, Ó. (2022). Design of a mobile binary parallel robot that exploits nonsingular transitions. Mech. Mach. Theory, 171, 104733.

[10] Peidró, A., Marín, J.M., Gil, A., & Reinoso, Ó. (2015). Performing nonsingular transitions between assembly modes in analytic parallel manipulators by enclosing quadruple solutions. J. Mech. Des., 137(12), 122302.

[11] Peidró, A., Marín, J.M., Ballesta, M., Reinoso O., Payá, L., & Jiménez, L.M. "Robot móvil desplazable en un plano". Patente ES2853898. Prioridad: 17-03-2020. Concesión: 04-02-2022. Titular: Universidad Miguel Hernández.

[12] Wang, Y., & Chirikjian, G. S. (2002). A divide-and-conquer method for inverse kinematics of hyper-redundant manipulators. In Advances in Robot Kinematics (pp. 407-414). Springer.

[13] Wang, Y., & Chirikjian, G. S. (2004). Workspace generation of hyper-redundant manipulators as a diffusion process on SE(N). IEEE Transactions on Robotics and Automation, 20(3), 399-408.

[14] Zanganeh, K. E., & Angeles, J. (1995). The inverse kinematics of hyper-redundant manipulators using splines. In Proceedings of 1995 IEEE International Conference on Robotics and Automation (Vol. 3, pp. 2797-2802). IEEE.

© 2022 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution CC-BY-NC-SA 4.0 license (https://creativecommons.org/licenses/by-nc-sa/4.0/deed.es).