SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de La Laguna

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

“Planificación de Movimientos Óptimos de un Robot Industrial para la Ejecución de Tareas con una Mesa Cooperante de Rotaciones Secuenciales”

POR

Ing. Jesús De Anda Mijares

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

DIRECTOR DE TESIS

Dr. José Alfonso Pámanes García

CODIRECTOR DE TESIS

M.C. Edmundo Javier Ollervides Vázquez

ISSN: 0188-9060

RIITEC: (07)-TMCIE-2018

Torreón, Coahuila. México,
Junio 2018

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de La laguna

Torreón, Coah., 13/Junio/2018
Dependencia: DEPI/CPCIE
Oficio: DEPIJ/CPCIE/070/2018
Asunto: Autorización de impresión de tesis.

C. Jesús de Anda Mijares
CANDIDATO AL GRADO DE MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA.
PRESENTE

Después de haber sometido a revisión su trabajo de tesis titulado:

"Planificación de Movimientos Secuenciales de un Robot Experimental para Tareas con la Asistencia de una Mesa Cooperante"

Habiendo cumplido con todas las indicaciones que el jurado revisor de tesis hizo, se le comunica que se le concede la autorización con número de registro **RIITEC: (07)-TMCIE-2018**, para que proceda a la impresión del mismo.

ATENTAMENTE
EDUCACIÓN TECNOLÓGICA FUENTE DE INNOVACIÓN

SECRETARIA DE EDUCACIÓN PUBLICA
INSTITUTO TECNOLÓGICO de la laguna
División de Estudios de Posgrado e Investigación

P.A.

DR. ARMANDO LONGORIA DE LA TORRE
Jefe de la División de Estudios de Posgrado e Investigación del Instituto Tecnológico de la Laguna

ALT/JIHJ

Torreón, Coah., 23/Mayo/2018

DR. ARMANDO LONGORIA DE LA TORRE
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

Por medio de la presente, hacemos de su conocimiento que después de haber sometido a revisión el trabajo de tesis titulado:

"Planificación de Movimientos Secuenciales de un Robot Experimental para Tareas con la Asistencia de una Mesa Cooperante"

Desarrollado por el **C. Jesús de Anda Mijares**, con número de control **M1613009** y habiendo cumplido con todas las correcciones que se le indicaron, estamos de acuerdo que se le conceda la autorización de la fecha de examen de grado para que proceda a la impresión de la misma.

ATENTAMENTE
EDUCACIÓN TECNOLÓGICA FUENTE DE INNOVACIÓN

Dr. J. Alfonso Pámanes García
Asesor/Director de Tesis

M.C. Edmundo J. Ollervides Vázquez
Comité Tutorial

Dr. Jaime González Sierra
Comité Tutorial

Dr. José Luis Meza Medina
Comité Tutorial

Dedicatoria

A mis padres por apoyarme siempre y motivarme para alcanzar las metas que me he trazado a lo largo de mi vida.

A mi hijo por darme las fuerzas para superarme día a día.

Agradecimientos

Al Dr. José Alfonso Pámanes García por su apoyo durante todo el desarrollo de esta tesis, por sus consejos y conocimientos que me ayudaron a sacar adelante este trabajo.

Al M.C. Edmundo Javier Ollervides Vázquez por su disponibilidad, apoyo y atención durante el desarrollo de esta tesis.

A todos mis maestros de maestría por los conocimientos impartidos en cada una de las materias cursadas durante mis estudios.

A todos mis compañeros del área de posgrado por su ayuda y consejos a lo largo de mis estudios de posgrado.

Al TecNM y al CONACYT por el apoyo recibido y las facilidades otorgadas para el desarrollo de este trabajo.

Planificación de Movimientos Óptimos de un Robot Industrial para la Ejecución de Tareas con una Mesa Cooperante de Rotaciones Secuenciales

Resumen

En esta tesis se estudia el problema de la planificación de movimientos óptimos de un robot industrial de soldadura para la ejecución de movimientos óptimos en cooperación con una mesa posicionadora. Se considera que se deben efectuar una secuencia de movimientos de la mesa y del robot hasta la conclusión de una tarea de soldadura propuesta. El método propuesto permite determinar los sucesivos emplazamientos de la mesa cooperante de tal manera que el desempeño cinemático del robot sea óptimo durante la ejecución de la tarea. El método se aplica en tres casos de estudio con movimientos 3D de la herramienta del robot. Los resultados de las simulaciones en computadora permiten apreciar la eficacia del método propuesto.

Optimal Motion Planning of a Welding Robot for Cooperative Tasks with a Sequential Positioning Table

Abstract

This thesis deals with the motion planning problem of an industrial robot to achieve welding tasks working in cooperation with a positioning table. We consider that sequential displacements must be carried out by the table and the robot until finish the whole task. Thus, a method is proposed to find the successive placements that must be attained by the cooperative table in order to locate the workpiece in such a way that the kinematic performance of the robot be optimized during the accomplishment of the task. The method is applied in three case study for motion planning of a welding robot during the tracking of a 3D path. Simulation studies show the efficacy of the proposed method.

Índice general

Índice de figuras

Índice de tablas

1. Introducción

A lo largo de las últimas décadas, los manipuladores robóticos han tomado relevancia en el desarrollo tecnológico de las industrias modernas [1], [2]. Los excelentes resultados proporcionados por los robots en términos de calidad, productividad y flexibilidad, los hacen económicamente atractivos para la automatización de procesos industriales. Actualmente, el campo de los robots industriales es una de las áreas económicas más confiables y con mayor crecimiento. La Federación Internacional de Robótica (IFR) estima que la cantidad de robots operacionales que existen actualmente en el mundo es de aproximadamente 1,800,000 [1]. Entre las aplicaciones de esta población, una de las más importantes es la de la soldadura por arco. Asimismo, con base en los registros de la IFR, en los últimos años, México ha mostrado la mayor tasa de crecimiento de robots operativos en todo el continente americano.

En la industria metalmecánica, la introducción de dispositivos de posicionamiento auxiliares en estaciones de trabajo robotizadas se está convirtiendo en una alternativa atractiva para mejorar el desempeño de los robots manipuladores en dichas estaciones. Dichos dispositivos consisten en mecanismos manipuladores automatizados que sujetan y desplazan las piezas de trabajo de un robot, cooperando con éste, de tal manera que le faciliten el acceso a las tareas y permitan aumentar su productividad [3]. Un dispositivo de posicionamiento puede ser una mesa articulada de uno o más grados de libertad, o un mecanismo con una cadena cinemática de mayor complejidad, hasta llegar a ser otro robot manipulador. En la figura 1.1 se muestra un robot industrial de soldadura ejecutando una tarea en cooperación con un dispositivo posicionador de 2 grados de libertad.

Figura 1.1 Robot industrial de soldadura ejecutando una tarea en cooperación con un dispositivo posicionador de 2 grados de libertad. Cortesía de ABB.

Como consecuencia de lo expuesto en el párrafo anterior, la planificación de movimientos de los robots que trabajan en cooperación con dispositivos de posicionamiento se ha vuelto hoy en día en un tema relevante para las empresas que utilizan robots. En esta tesis, se estudiará la planificación de movimientos de un robot manipulador industrial que trabajará en coordinación con un dispositivo cooperante de un grado de libertad (gdl), al que le llamaremos en lo sucesivo *mesa cooperante*.

Desde hace algunas décadas, la planificación de movimientos de robots con mesas cooperantes ha sido estudiada bajo diferentes enfoques. En [4] se realiza un análisis interesante sobre la evolución de los tratamientos para los diversos aspectos de la planificación de rutas de robots. Los problemas de planificación de movimientos se pueden plantear bajo diferentes niveles de complejidad, teniendo en cuenta diversos aspectos del problema. Por ejemplo, se puede considerar que la mesa cooperante posee un solo grado de libertad [5], [6], o que la pieza de trabajo es movida por otro robot de 6 gdl [7], [8], y asumir que el robot y el dispositivo posicionador se mueven de manera simultánea o que lo hacen de manera secuencial. De la misma manera es posible suponer que el robot es cinemáticamente redundante [9].

En las estaciones de trabajo de robots industriales en pequeñas y medianas empresas generalmente se usan mesas posicionadoras de 1 gdl pero, en contraste con las grandes industrias, sólo se aplican para cambiar de pieza de trabajo una vez que se ha concluido parcial o totalmente la tarea en una pieza. No se aprovecha el potencial que poseen estas mesas para mejorar la accesibilidad del robot a sus tareas ni para mejorar el desempeño cinemático y la productividad de los robots. Lo anterior es una consecuencia del alto costo que implica un paquete de software profesional con capacidad para el tratamiento del correspondiente problema de planificación de movimientos.

1.1 Objetivos de la tesis

Teniendo en consideración lo anterior, se plantea como objetivo general del presente trabajo la formulación de un método para la planificación de movimientos de robots industriales de 6 gdl, para la ejecución de tareas con la asistencia de una mesa cooperante de rotaciones secuenciales de 1 gdl. Como objetivo específico se propone el desarrollo de una aplicación de software en el lenguaje computacional de Matlab© para la planificación de los movimientos óptimos del robot y de su mesa cooperante, así como para la simulación gráfica y la programación fuera de línea de dichos movimientos. Se pretende, asimismo, que la aplicación de software permita brindar servicio y capacitación a empresas de la localidad en el tema de planificación de movimientos de robots auxiliados por mesas cooperantes de 1 gdl.

1.2 Organización de la tesis

En el capítulo 2 se hace una revisión de la literatura relacionada con los temas de esta tesis, se describen los escenarios, los métodos, así como las contribuciones principales y limitaciones de los enfoques considerados en cada trabajo revisado. En el capítulo 3 se presenta una descripción de los conceptos básicos aplicados en la tesis: modelado cinemático de manipuladores, cálculo de la matriz jacobiana, los índices de desempeño de

las manipulabilidades traslacional y rotacional, así como la descripción del paquete SYMORO, utilizado para el modelado simbólico de manipuladores. En el capítulo 4 se describen las características técnicas del robot *7Bot* y de su estación de trabajo. Asimismo, se determinan los modelos cinemáticos directo e inverso de posición del manipulador considerado, se obtiene su matriz jacobiana, así como las manipulabilidades traslacional y rotacional. En el capítulo 5 se formula el procedimiento de resolución del problema de la optimización de movimientos del robot estudiado, y se describe el algoritmo de optimización que se aplica para la resolución del problema planteado. En el capítulo 6 se presenta la aplicación del método propuesto para la resolución de un caso de estudio y se analizan los resultados obtenidos. En la conclusión de la tesis se evalúa la eficacia del método propuesto y de la aplicación de software realizada. Se destaca la importancia de su aplicación en pequeñas y medianas industrias usuarias de robots. Finalmente se proponen diversos temas a ser considerados para su estudio en futuros trabajos.

2. Revisión bibliográfica

El problema de la optimización del desempeño de robots manipuladores ha sido estudiado desde hace varias décadas bajo diferentes enfoques, y teniendo en consideración diversos criterios para la medición del desempeño. En este capítulo se efectúa una breve revisión de algunos trabajos que han sido publicados sobre este tema en la literatura, según se reporta en [10], así como en fuentes diversas.

Desde el punto de vista de la productividad de un robot, el criterio que sirve de manera natural para evaluar su desempeño, con respecto a una tarea, es el tiempo necesario para ejecutar ésta. Scheinman y Roth [11] fueron de los primeros autores que trataron el problema de la optimización del desempeño de un robot considerando la minimización del tiempo de ejecución de su tarea. El método propuesto se aplicó para un robot con arquitectura SCARA, con resultados interesantes. Sin embargo, el criterio considerado sólo se puede aplicar para tareas en las que no existen restricciones para las trayectorias que debe seguir el órgano terminal para ir del punto inicial al punto final, por ejemplo, en tareas del tipo *pick and place*.

En tareas con trayectorias fijas, como en las de soldadura de arco eléctrico, no es posible utilizar el método de Schienman y Roth; en tales casos es conveniente medir el desempeño de los robots con base en otro tipo de criterios, por ejemplo, basados en la eficiencia cinemática, o en la facilidad de control, o en la aptitud para ejercer fuerzas en direcciones específicas.

En el proceso de diseño de una mano articulada robotizada, Salisbury y Craig [12] propusieron aplicar el número de condicionamiento de la matriz jacobiana de la cadena cinemática de cada dedo de la mano para medir el desempeño de dicha cadena. La minimización de este índice permite reducir la propagación de errores de control de los pares torsores aplicados en las articulaciones de un dedo hacia la fuerza ejercida en la punta del dedo.

Por otra parte, diversos autores aplican el índice de la manipulabilidad como criterio de desempeño cinemático. Como se verá en un capítulo posterior el índice de manipulabilidad, propuesto por T. Yoshikawa [13], mide la eficacia de la conversión de desplazamientos articulares de la cadena cinemática de un robot en movimiento de su órgano terminal.

Aparte del índice de manipulabilidad, otros autores han introducido distintos índices de para evaluar el desempeño de un robot bajo diversos criterios. Chiu [14] estableció el *índice de compatibilidad*, el cual incorpora en una sola función, de manera coherente, las relaciones de transmisión de fuerza y de velocidad. Para una postura dada de la cadena cinemática de un robot, este índice mide su capacidad para transmitir fuerza o para desplazarse en direcciones específicas en el espacio operacional.

Además de usar múltiples criterios para medir el desempeño, en la literatura se han considerado diversos escenarios con diferentes niveles de complejidad para la planificación de movimientos óptimos. El caso mencionado líneas arriba, referente a la minimización del tiempo de ciclo en tareas del tipo pick and place, es uno de los primeros que se trató en la bibliografía científica. Sin embargo, existe una gran cantidad de trabajos en los que se aplicaron criterios de carácter cinemático para la optimización del desempeño de robots. Por ejemplo, en el estudio de Nelson et al [15], orientado a la optimización de la manipulabilidad del robot durante la ejecución de tareas de trayectoria fija, se determina el posicionamiento de la ruta deseada dentro del espacio de trabajo de un robot industrial no redundante. No se considera desplazamiento alguno de la pieza de trabajo.

En contraste, Hemmerle y Prinz [8] estudiaron el problema de la ejecución de una tarea de trayectoria fija de un robot de 6 gdl, en una pieza de trabajo que tiene un movimiento general en el espacio, y es desplazada por dos robots de 6 gdl. Es decir, que se tienen tres robots cooperando durante la ejecución de una tarea. En este trabajo se plantea alejar las variables articulares de sus valores límite para todos los robots y minimizar todos los desplazamientos articulares. Además, se requiere resolver la redundancia cinemática del sistema de tres robots debido a que las tareas requieren, en el caso más general, un robot de 6 gdl mientras que aquí se utilizan 18 gdl.

Para tareas de trayectoria fija, Pámanes y Zeghloul [16] proponen un método de optimización multicriterio para determinar el emplazamiento de robots no redundantes en su estación de trabajo. Los criterios que se pueden aplicar para la optimización del desempeño son la manipulabilidad, las relaciones de transmisión de fuerza y de velocidad, y el número de condicionamiento de la matriz jacobiana. El método incluye restricciones para la elusión de obstáculos y para la prevención de posturas singulares. En un trabajo posterior, Pámanes, Cuan y Zeghloul [17] extendieron este método para su aplicación en robots cinemáticamente redundantes.

Otros estudios relativos a la optimización del desempeño de robots son los presentados en [18] y [19]. En el primer trabajo se resuelve el problema de la redundancia cinemática de un robot de 6 gdl en la ejecución de tareas de soldadura que demandan sólo 5 grados de movilidad de la antorcha de soldadura. En el segundo se diseña una mesa de trabajo fija para un robot industrial de soldadura, con dimensiones óptimas.

En las referencias [16] a [19] no se considera la cooperación de dispositivos posicionadores para la ejecución de la tarea y, en consecuencia, no existen desplazamientos de la pieza de trabajo.

Por otra parte, en [20] se estudió la optimización de trayectorias con algoritmos para la planificación de rutas sin colisiones en un caso industrial. El problema de planificación del

proceso investigado consiste en secuenciar los puntos individuales de soldadura y calcular una trayectoria correspondiente, de tal manera que se minimice el tiempo del ciclo de soldadura. El algoritmo propuesto comienza con una solución parcial que consta de dos cordones de soldadura cuyos puntos medios son los más alejados entre sí. El camino correspondiente a estos dos cordones de soldadura es una línea recta que conecta los puntos más cercanos de acceso. Entonces, en cada iteración, para cada punto aún no secuenciado, el algoritmo busca la mejor posición para insertar el punto dado en el trayecto actual, llamando a un algoritmo para la evaluación exhaustiva de las posibles posiciones de inserción, incluyendo al principio o al final del camino. La aleatorización se consigue perturbando el coste de cada inserción candidata.

En [21], se presenta un enfoque basado en redes neuronales con una implementación exitosa para el problema de secuenciación de tareas de un robot. En ese trabajo se considera una secuenciación de tareas en un sistema de flexible de manufactura (FMS), el cual consta de múltiples máquinas, múltiples sitios intermedios de almacenamiento y un único robot de alcance libre que transporta piezas entre máquinas y también entre máquinas y sitios de almacenamiento. En cada periodo de muestreo se examina el sistema y se genera un conjunto de peticiones de movimiento (tareas). Se asume que el conjunto de tareas en cada instante de tiempo se conoce a priori, por lo que se puede considerar que el robot tiene un conjunto de tareas a ejecutar en cada punto de planificación. Cada tarea se genera por la necesidad de cargar una pieza en una máquina o descargar una pieza de la máquina. Las tareas a secuenciar también pueden ser necesarias para cumplir con algunas restricciones de precedencia. El problema de optimización se resuelve construyendo una red neuronal recurrente apropiada y la función de energía para la red, e interpretando las salidas de red como la solución del problema de optimización. La función de energía se construye de tal manera que el estado de la red que alcanza el valor energético más bajo corresponde a la solución óptima (o casi óptima) del problema.

De los trabajos mencionados en los párrafos precedentes solamente en [8] se estudia la ejecución de tareas de un robot principal trabajando simultáneamente con dos robots cooperantes. Se trata de un escenario interesante que, sin embargo, no es frecuente en las pequeñas y medianas empresas (*pymes*) toda vez que éstas utilizan generalmente un robot por cada tarea robotizada.

Los dispositivos cooperantes que se suele aplicar en las *pymes*, específicamente en tareas de soldadura de arco, se reducen generalmente a una mesa de rotaciones secuenciales que se programa para facilitar el montaje de las piezas de trabajo que serán tratadas por el robot sobre la mesa, y el desmontaje de dichas piezas una vez que las operaciones han sido concluidas. Infortunadamente, ese tipo de dispositivos no se aprovechan para propiciar un mejor desempeño cinemático del manipulador. El software comercial con capacidad para obtener soluciones para aprovechar las mesas cooperantes en ese sentido no es económicamente interesante para las *pymes*. Así, el principal motivo del desarrollo del presente trabajo de tesis consiste en proponer un método, y el software correspondiente, que permita brindar soluciones a las *pymes* usuarias de robots de soldadura en relación a la optimización del desempeño cinemático de dichas máquinas a un costo reducido.

3. Fundamentos para el modelado cinemático de robots

En este capítulo se efectúa una revisión de las principales herramientas teóricas aplicadas en la formulación del problema estudiado de planificación de movimientos óptimos.

3.1 Modelo cinemático directo de posición

El modelo cinemático directo de posición de un robot consiste en un conjunto de m funciones escalares que determinan las coordenadas operacionales $[x_1, x_2, \ldots, x_m]^T$, las cuales definen la pose, $\boldsymbol{x_p}$, del órgano terminal del robot, a partir de la postura, $\boldsymbol{q}$, cuyos elementos son las variables articulares $[q_1, q_2, \ldots, q_n]^T$ del robot. Las coordenadas operacionales describen la pose (posición y orientación) del órgano terminal del robot con respecto a un marco ortogonal unido a la base de éste, mientras que las variables articulares definen la posición relativa de cada eslabón móvil de la cadena cinemática del robot con respecto al eslabón precedente. En los robots industriales seriales generalmente $n=6$ y $m=6$. Teniendo en cuenta la definición del modelo cinemático directo de posición de un robot, éste se puede expresar como:

$$\boldsymbol{x_p} = \boldsymbol{f}(\boldsymbol{q}) \quad , \tag{3.1}$$

donde $\boldsymbol{f}(\boldsymbol{q})$ es una función vectorial de la postura $\boldsymbol{q}$.

Un procedimiento sistematizado que simplifica la descripción de la arquitectura de un robot, y el cálculo del modelo directo de posición, se basa en el uso de los parámetros modificados de Denavit-Hartenberg [22]. El método considera que el manipulador está compuesto por $n+1$ eslabones idealmente rígidos (uno de los cuales es fijo) y n articulaciones, que pueden ser prismáticas (tipo P) o de revolución (tipo R), designando a los eslabones como $C_0, C_1, \dots, C_n$ y a las articulaciones como $a_1, a_2, \dots, a_n$ [23]. Nótese que si el robot posee n articulaciones, entonces éste tendrá $n+1$ eslabones. En consecuencia, si se utiliza el índice j para designar un número de eslabón, entonces $j = 0, 1, \dots, n$, y si el índice j designa un número de articulación o de variable articular, entonces $j = 1, \dots, n$

En el proceso de descripción de la cadena cinemática del robot se asigna un marco de referencia ortonormal Σ_j a cada eslabón C_j $(j = 0, 1, \dots, n)$ y se definen los parámetros que especifican la pose de cada marco con respecto al precedente. Los vectores unitarios que definen el j-ésimo marco de referencia se definen conforme a la siguiente convención:

- El vector $\boldsymbol{z}_j$ se define a lo largo del eje de la articulación a_j. Este eje es el de rotación en el caso de una articulación tipo R, o de deslizamiento en el caso de una articulación P.
- El vector $\boldsymbol{x}_j$ se define a lo largo de la línea perpendicular común a $\boldsymbol{z}_j$ y a $\boldsymbol{z}_{j+1}$. Si estos dos vectores son paralelos, entonces $\boldsymbol{x}_j$ no se puede definir de manera única. El punto de intersección de $\boldsymbol{x}_j$ y $\boldsymbol{z}_j$ define al origen O_j del marco Σ_j.
- El vector $\boldsymbol{y}_j$ se define a partir de los vectores $\boldsymbol{x}_j$ y $\boldsymbol{z}_j$, de tal manera que se complete un marco de mano derecha.

Una vez asignado un marco a cada eslabón, se definen los parámetros geométricos que especifican la posición y orientación de cada marco con respecto al precedente:

- α_j Es el ángulo de $\boldsymbol{z}_{j-1}$ a $\boldsymbol{z}_j$, medido con respecto a $\boldsymbol{x}_{j-1}$ conforme a la regla de la mano derecha.
- d_j Es la distancia entre $\boldsymbol{z}_{j-1}$ y $\boldsymbol{z}_j$, a lo largo de $\boldsymbol{x}_{j-1}$.

- θ_j Es el ángulo de $\boldsymbol{x}_{j-1}$ a $\boldsymbol{x}_j$, medido con respecto a $\boldsymbol{z}_j$ conforme a la regla de la mano derecha.
- r_j Es la distancia entre $\boldsymbol{x}_{j-1}$ y $\boldsymbol{x}_j$ a lo largo de $\boldsymbol{z}_j$.

A partir de los parámetros geométricos, se tiene que la variable articular q_j , asociada a la articulación *j*, se puede identificar mediante:

$$q_j = \bar{\sigma_j}\theta_j + \sigma_j r_j \tag{3.2}$$

Donde:

$$\bar{\sigma_j} = 1 - \sigma_j \tag{3.3-a}$$

y

$$\sigma_j = \begin{cases} 0, & si\ la\ articulacion\ j\ es\ tipo\ R \\ 1, & si\ la\ articulacion\ j\ es\ tipo\ P \end{cases} \tag{3.3-b}$$

Es posible determinar una matriz de transformación homogénea general que defina, en función de los parámetros geométricos, la posición y la orientación del marco Σ_j con respecto al Σ_{j-1} $(j = 1, \dots, n)$. Esta matriz está dada por:

$${}^{j-1}_{j}T = \begin{bmatrix} c\theta_j & -s\theta_j & 0 & d_j \\ c\alpha_j s\theta_j & c\alpha_j c\theta_j & -s\alpha_j & -r_j s\alpha_j \\ s\alpha_j s\theta_j & s\alpha_j c\theta_j & c\alpha_j & r_j c\alpha_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

donde $c\theta \equiv cos\,\theta$ y $s\theta \equiv sen\,\theta$.

Sustituyendo en esta matriz los parámetros geométricos correspondientes a cada eslabón de un robot específico, se obtienen n matrices de transformación homogéneas, las cuales son llamadas matrices elementales del robot.

De acuerdo a la convención modificada de Denavit-Hartenberg, se tiene que al eslabón fijo de un robot se le asigna un marco de referencia Σ_0, y que al órgano terminal (eslabón n) se le asigna un marco Σ_n. Por lo tanto la matriz $\boldsymbol{{}^0_n T}$ define la pose del órgano terminal del robot con respecto al marco Σ_0. Se sabe que esa matriz se puede obtener mediante:

$$ {}^0_n T = {}^0_1 T \ {}^1_2 T \ \dots \ {}^{n-2}_{n-1} T \ {}^{n-1}_n T \quad . \tag{3.5} $$

Obsérvese que, mientras que la matriz del lado izquierdo de la Ec. (3.5) contiene implícitamente las coordenadas operacionales de la pose del marco Σ_0 con respecto al Σ_n , el producto de las matrices del lado derecho determina implícitamente las componentes de la función vectorial $\boldsymbol{f}$ de la Ec. (3.1). En consecuencia, el desarrollo de este producto equivale a la resolución del modelo directo de posición.

3.2 Modelo cinemático inverso de posición

El modelo cinemático inverso de posición de un robot es la función inversa $\boldsymbol{f}^{-1}$ que, si existe, para un robot dado permite obtener la configuración $\boldsymbol{q}$ necesaria para ubicar su órgano terminal en una pose deseada $\boldsymbol{x}$ [23]:

$$ \boldsymbol{q} = \boldsymbol{f}^{-1}(\boldsymbol{x_p}) \ . \tag{3.6} $$

La resolución del modelo inverso consiste en identificar las funciones escalares que integran a $\boldsymbol{f}^{-1}$. Recordando que $\boldsymbol{q} \in \boldsymbol{R}^n$, y que $\boldsymbol{x_p} \in \boldsymbol{R}^m$, se tienen los siguientes casos para la resolución del modelo inverso de un robot:

a) Si $n < m$, no existe solución para $\boldsymbol{f^{-1}}$.

b) Si $n = m$, y la arquitectura del robot es compatible con la pose $\boldsymbol{x}$ propuesta, los siguientes sub-casos son posibles:

   b.1 Existe un número finito de posturas solución no singulares.

   b.2 Existe un número infinito de posturas solución singulares.

   Es conveniente advertir que un robot, bajo una postura singular, pierde al menos un grado de libertad, por lo que se reduce su capacidad para desplazar arbitrariamente al órgano terminal en el espacio.

c) Si $n > m$, y la arquitectura del robot es compatible con la pose $\boldsymbol{x_p}$ propuesta, existe un número infinito de soluciones. En este caso, el robot tiene más grados de libertad que los estrictamente necesarios para realizar su tarea.

3.3 Planteamiento del problema del modelado inverso de posición

Considérense los marcos de referencia ortonormales Σ_e fijo en la estación de trabajo de un manipulador de n grados de libertad, Σ_h unido a la herramienta del órgano terminal en el eslabón n del manipulador, y Σ_n unido también al eslabón n, pero definido de acuerdo a la convención modificada de Denavit-Hartenberg. Estos marcos se muestran en el esquema de la figura 3.1. La pose deseada de la herramienta en el espacio es especificada con respecto al marco Σ_e, mediante una matriz homogénea ${}^{e}_{h}\boldsymbol{T}$. Se tiene entonces que las poses de los marcos Σ_0, Σ_h, y Σ_n se definen con respecto a los marcos Σ_e, Σ_n, y Σ_0, respectivamente, mediante las matrices ${}^{e}_{0}\boldsymbol{T}$, ${}^{n}_{h}\boldsymbol{T}$, y ${}^{0}_{n}\boldsymbol{T}$, respectivamente. En consecuencia, para la pose deseada de la herramienta, se debe satisfacer la siguiente ecuación:

$$ {}^{e}_{h}\boldsymbol{T} = {}^{e}_{0}\boldsymbol{T}\ {}^{0}_{n}\boldsymbol{T}\ {}^{n}_{h}\boldsymbol{T} \tag{3.7} $$

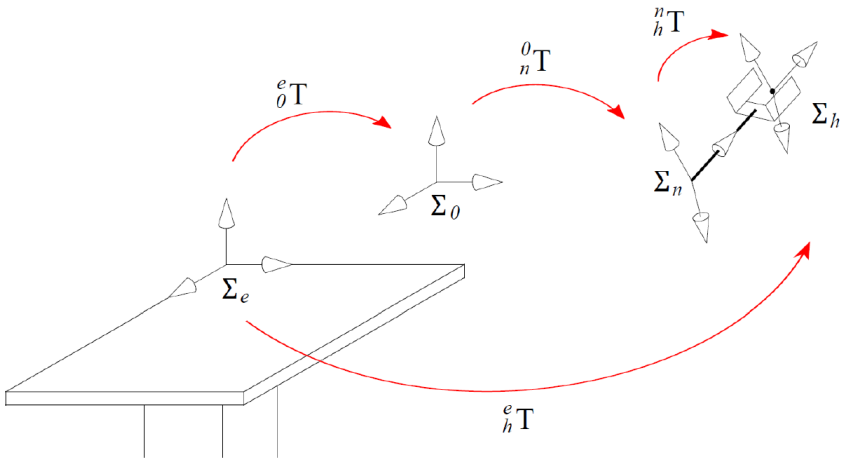


Figura 3.1 Pose del marco Σ_h, unido a la herramienta de un robot, con respecto al marco Σ_e de la estación de trabajo.

Asumiendo que se especifican tanto el emplazamiento del manipulador en la estación de trabajo como la geometría de la herramienta del órgano terminal, se tiene que las matrices ${}^{e}_{0}\boldsymbol{T}$ y ${}^{n}_{h}\boldsymbol{T}$ son conocidas. Agrupando entonces del lado izquierdo de la ecuación anterior todos los términos conocidos se tiene:

$$ {}^{e}_{0}\boldsymbol{T}^{-1}\ {}^{e}_{h}\boldsymbol{T}\ {}^{n}_{h}\boldsymbol{T}^{-1} = {}^{0}_{n}\boldsymbol{T} \tag{3.8}$$

Definiendo ahora la matriz $\boldsymbol{U}_0$ como

$$ \boldsymbol{U}_0 \equiv {}^{e}_{0}\boldsymbol{T}^{-1}\ {}^{e}_{h}\boldsymbol{T}\ {}^{n}_{h}\boldsymbol{T}^{-1} \tag{3.9}$$

entonces se puede escribir la Ec. (3.8) de la siguiente manera:

$$ \boldsymbol{U}_0 = {}^{0}_{n}\boldsymbol{T} \tag{3.10}$$

Obsérvese que la matriz $\boldsymbol{U}_0$ describe la pose deseada del marco Σ_n con respecto al Σ_0, por lo que ésta está definida de manera implícita por el vector $\boldsymbol{x_p}$ de coordenadas operacionales deseadas del manipulador. Por su parte, ${}^{0}_{n}\boldsymbol{T}$ depende de la postura $\boldsymbol{q}$, cuyos valores de sus variables articulares son desconocidos. Así, el problema del modelo inverso de posición del manipulador consiste en determinar el vector $\boldsymbol{q}$ cuyas componentes, contenidas en ${}^{0}_{n}\boldsymbol{T}$, satisfagan la Ec. (3.10).

3.4 Método de Paul

Este método ha sido propuesto [24] para el modelado de robots de 6 grados de libertad ($n = 6$), y se basa en el uso de las matrices elementales del robot considerado, las cuales permiten obtener sucesivamente las variables articulares a partir de la Ec. (3.10). Si bien existen otros métodos para la resolución del modelo cinemático inverso de posición, el método de Paul resulta el más apropiado considerando que la arquitectura del robot estudiado en este trabajo tiene una muñeca de ejes concurrentes. En [25] se pueden consultar algunos métodos alternativos.

Recordando que la matriz ${}^{0}_{n}\boldsymbol{T}$ resulta del producto de las matrices elementales del manipulador se tiene que la Ec. (3.10) se puede escribir como:

$$\boldsymbol{U}_0 = {}^{0}_{1}\boldsymbol{T}\ {}^{1}_{2}\boldsymbol{T} \dots {}^{n-2}_{n-1}\boldsymbol{T}\ {}^{n-1}_{n}\boldsymbol{T} \quad , \tag{3.11}$$

donde los elementos de la matriz conocida $\boldsymbol{U}_0$ se definen de la siguiente manera:

$$\boldsymbol{U}_0 = \begin{bmatrix} s_x & n_x & a_x & p_x \\ s_y & n_y & a_y & p_y \\ s_z & n_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.12}$$

Agrupando los elementos de cada columna en vectores, la matriz (3.12) se puede escribir de la siguiente manera:

$$\boldsymbol{U}_0 = [\boldsymbol{s} \quad \boldsymbol{n} \quad \boldsymbol{a} \quad \boldsymbol{p}] \tag{3.13}$$

Se observa que de la Ec. (3.11) resultan 12 ecuaciones escalares no triviales, conteniendo como incógnitas las variables del manipulador. Estas ecuaciones son altamente no lineales, lo cual complica significativamente su resolución. El enfoque de Paul

permite simplificar el aislamiento de cada variable articular mediante el principio que se ilustra para q_1 como sigue:

Premutiplíquense ambos miembros de la Ec. (3.11) por ${}^{1}_{0}\boldsymbol{T}$. Así se obtiene:

$$ {}^{1}_{0}\boldsymbol{T}\,\boldsymbol{U}_0 = {}^{1}_{2}\boldsymbol{T} \ldots {}^{n-2}_{n-1}\boldsymbol{T}\ {}^{n-1}_{\boldsymbol{n}}\boldsymbol{T} \tag{3.14}$$

En esta ecuación se observa que el producto del lado derecho está en función de las incógnitas $q_2,\ q_3 \ldots, q_6,$ mientras que el lado izquierdo contiene solamente a q_1, quedando así aislada esta incógnita. Entonces, de las 12 ecuaciones escalares que resultan, se busca una que facilite la determinación de q_1.

Obsérvese que aplicando de manera reiterada el procedimiento anterior es posible obtener ecuaciones para determinar todas las variables articulares. Esto se puede realizar de la siguiente manera:

- Para obtener q_j (con $j = 1, 2, \ldots, 5$) se premultiplica la siguiente ecuación por ${}^{j}_{j-1}\boldsymbol{T}$:

$$\boldsymbol{U}_{j-1} = \prod_{k=j}^{n} {}^{k-1}_{k}\boldsymbol{T} \tag{3.15}$$

De la ecuación que resulte se igualan los elementos de las matrices de ambos lados y se busca resolver para q_j.

- Finalmente, la variable q_6 se obtiene de las ecuaciones escalares que resultan de:

$$\boldsymbol{U}_5 = {}^{5}_{6}\boldsymbol{T} \tag{3.16}$$

En la primera iteración $(j = 1)$, la matriz del lado izquierdo de la ecuación (3.15) está definida por los elementos de los vectores $\boldsymbol{s}, \boldsymbol{n}, \boldsymbol{a}, \boldsymbol{p}$, correspondientes a la pose deseada del órgano terminal. En las siguientes iteraciones la matriz $\boldsymbol{U}_{j-1}$ para $j = 2, 3, .., 6$, que se requiere en las ecuaciones (3.15) y (3.16), se obtiene mediante:

$$\boldsymbol{U}_{j-1} = {}^{j-1}_{j-2}\boldsymbol{T}\,\boldsymbol{U}_{j-2} \quad , \tag{3.17}$$

donde la matriz ${}^{j-1}_{j-2}\boldsymbol{T}$ depende de q_{j-1}, que se determina en la iteración precedente, lo mismo que $\boldsymbol{U}_{j-2}$.

En cada iteración del procedimiento de Paul, es posible que se requiera resolver ecuaciones del tipo:

$$a\ cos\,\theta + b\,sin\,\theta = c \quad , \tag{3.18}$$

verificando $a^2 + b^2 \geq c^2$.

Se puede probar [23] que la solución de la Ec. (3.18) está dada por:

$$\sin\theta = \frac{bc+\varepsilon a\sqrt{a^2+b^2-c^2}}{a^2+b^2} \quad , \qquad \cos\theta = \frac{ac-\varepsilon b\sqrt{a^2+b^2-c^2}}{a^2+b^2} \tag{3.19}$$

Donde $\varepsilon = \pm 1$.

3.5 Modelado cinemático de velocidad

El modelo cinemático directo de velocidad del robot se obtiene derivando la Ec. (3.1) con respecto al tiempo:

$$\dot{\boldsymbol{x}}_{\boldsymbol{p}} = \boldsymbol{J}\dot{\boldsymbol{q}} \quad , \tag{3.20}$$

donde $\dot{\boldsymbol{x}}_{\boldsymbol{p}}$ es el vector del estado de velocidad del órgano terminal del robot ($\dot{\boldsymbol{x}}_{\boldsymbol{p}} \in \boldsymbol{R}^m$), expresado en términos de las derivadas de las coordenadas operacionales de posición con respecto al tiempo:

$$\dot{\boldsymbol{x}}_{\boldsymbol{p}} = [\dot{x}_1, \dot{x}_2, \ldots, \dot{x}_m]^T \quad , \tag{3.21}$$

mientras que $\dot{\boldsymbol{q}}$ es el vector de velocidades articulares del robot ($\dot{\boldsymbol{q}} \in \boldsymbol{R}^n$):

$$\dot{\boldsymbol{q}} = [\dot{q}_1, \dot{q}_2, \dots, \dot{q}_n]^T \quad , \tag{3.22}$$

y $\boldsymbol{J} \in \boldsymbol{R}^{m \times n}$ es la matriz jacobiana del robot, cuyos elementos J_{ij} (i=1,2,..., m; j=1, 2,..., n) son:

$$J_{ij} = \frac{\partial f_i}{\partial q_j} \quad , \tag{3.23}$$

donde claramente f_i es la i-ésima función del modelo cinemático directo de posición del robot.

Recordemos que el número de coordenadas operacionales, m, del órgano terminal de un robot depende del tipo de coordenadas operacionales que se utilicen. En el caso del robot considerado en esta tesis, se aplicarán 3 coordenadas cartesianas para especificar la posición del órgano terminal, y 3 ángulos para definir su orientación. Por lo tanto m será igual a 6. El marco de referencia de ambos tipos de coordenadas será el marco Σ_0. Por otra parte, el robot es de 6 gdl; es decir que posee 6 articulaciones actuadas. En consecuencia, n también será igual a 6. Así, la matriz jacobiana básica será una matriz de 6×6.

Por otra parte, de la Ec. (3.20) se consigue fácilmente el modelo cinemático inverso de velocidad del robot:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}\dot{\boldsymbol{x}}_{\boldsymbol{p}} \quad . \tag{3.24}$$

La obtención de los elementos de la matriz jacobiana puede resultar complejo mediante el uso de la Ec. (3.23). Esto se debe al carácter no lineal de las funciones escalares f_i, por lo cual se han desarrollado métodos alternativos más eficientes para el cálculo de $\boldsymbol{J}$ [23].

3.6 Modelo de velocidad basado en la matriz jacobiana *básica*

El estado de velocidad del órgano terminal de un robot de 6 grados de libertad también se puede expresar como:

$$\boldsymbol{t_n} = \boldsymbol{J_n}\,\dot{\boldsymbol{q}} \qquad , \tag{3.25}$$

donde $\boldsymbol{t_n}$ es el estado de velocidad (o *twist*, en inglés) del marco unido al último eslabón de la cadena cinemática del robot (eslabón *n*), cuyos elementos se dan en la siguiente ecuación:

$$\boldsymbol{t_n} = [v_{Onx} \quad v_{Ony} \quad v_{Onz} \quad \omega_{nx} \quad \omega_{ny} \quad \omega_{nz}]^T \tag{3.26}$$

En la Ec. (3.26) v_{Onx} , v_{Ony} , v_{Onz} son las componentes cartesianas de la velocidad del origen O_n del marco unido al eslabón *n* de la cadena cinemática, mientras que ω_{nx} , ω_{ny} , ω_{nz} son las componentes cartesianas del vector de velocidad angular de ese marco. $\boldsymbol{J_n}$ es la matriz jacobiana básica del robot [25] , y $\dot{\boldsymbol{q}}$ es la matriz columna 6-dimensional que contiene las velocidades articulares del manipulador.

La matriz jacobiana básica del manipulador es la matriz de 6×6 dada por:

$$\boldsymbol{J_n} = \begin{bmatrix} \sigma_1\boldsymbol{z}_1 + \overline{\sigma}_1\boldsymbol{z}_1 \times \boldsymbol{p}_{1,6} & \vdots & \dots & \vdots & \sigma_6\boldsymbol{z_6} + \overline{\sigma}_6\boldsymbol{z}_6 \times \boldsymbol{p}_{6,6} \\ \overline{\sigma}_1\boldsymbol{z}_1 & \vdots & \dots & \vdots & \overline{\sigma}_6\boldsymbol{z}_6 \end{bmatrix} \ , \tag{3.27}$$

donde σ_i y $\overline{\sigma}_i$ están definidas en las ecuaciones (3.3 *a*) y (3.3 *b*), $\boldsymbol{p}_{i,n}$ es el vector de posición que parte del origen del marco *i* y llega al origen del marco *n*, y $\boldsymbol{z}_i$ es el vector $\boldsymbol{z}$ del marco ortonormal unido al eslabón *i*, definido de acuerdo a la convención modificada de Denavit-Hartenberg *(i=1, 2,…., 6)*, que se presentó en la sección 3.1 de este documento.

La relación entre los estados de velocidad del órgano terminal del robot dados por $\dot{\boldsymbol{x}}$ y $\boldsymbol{t_n}$, se expresa mediante la siguiente ecuación:

$$\dot{\boldsymbol{x}} = \boldsymbol{M}(\boldsymbol{x})\, \boldsymbol{t_n} \tag{3.28}$$

Donde $\boldsymbol{M}(\boldsymbol{x})$ es una matriz de 6×6 que permite efectuar la transformación indicada, y tiene la siguiente forma:

$$\boldsymbol{M}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{M_P}(\boldsymbol{x_P}) & \boldsymbol{0_3} \\ \boldsymbol{0_3} & \boldsymbol{M_R}(\boldsymbol{x_R}) \end{bmatrix} \quad , \tag{3.29}$$

donde:

$\boldsymbol{M_P}(\boldsymbol{x_P})$ es una matriz de 3×3 que permite efectuar la transformación de las componentes de $\boldsymbol{t_n}$ correspondientes a la velocidad del punto O_6 del eslabón 6.

$\boldsymbol{M_R}(\boldsymbol{x_R})$ es una matriz de 3×3 que permite efectuar la transformación de las componentes de $\boldsymbol{t_n}$ correspondientes a la velocidad angular del órgano terminal.

$\boldsymbol{0_3}$ es la matriz nula de 3×3.

3.7 Manipulabilidades traslacional y rotacional de un robot

Dos de los índices que más se han utilizado, para la medición del desempeño cinetostático en problemas de emplazamiento óptimo, planificación de movimientos o de diseño de manipuladores, son la manipulabilidad, y el número de condicionamiento de la matriz jacobiana. Teniendo en cuenta que el robot industrial estudiado es de 6 gdl, es de mayor interés medir la capacidad de producir desplazamientos traslacionales y desplazamientos rotacionales, por separado, de una postura del robot. Así, resulta más conveniente la aplicación de los índices de manipulabilidad traslacional y de manipulabilidad rotacional formulados por T. Yoshikawa [26]. En esta sección se efectúa una revisión de los conceptos propuestos por dicho autor.

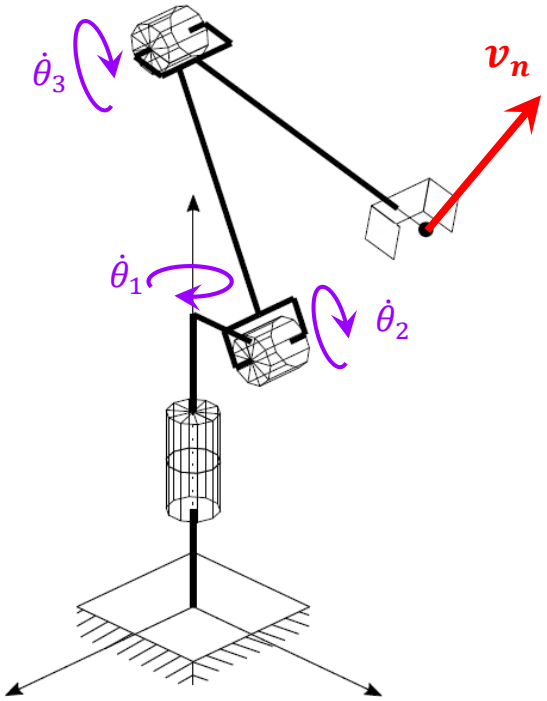


Figura 3.2. Esquema de un robot de tres grados de libertad 3R. Se indican las velocidades articulares y la velocidad del órgano terminal.

3.7.1 Manipulabilidad de un robot

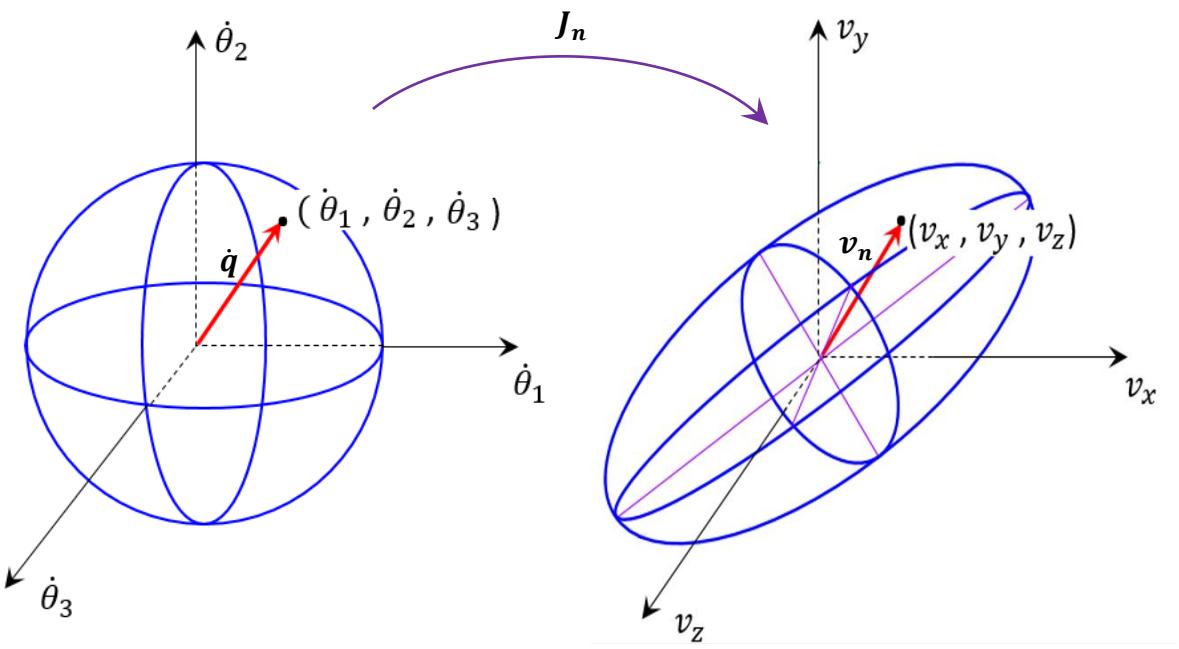
Consideremos un robot de 3 gdl *3R*, con la arquitectura mostrada en la figura 3.2, y asumamos que las velocidades articulares del manipulador se restringen de tal manera que:

$$\|\dot{\boldsymbol{q}}\| \leq 1 \tag{3.30}$$

Es posible probar que, bajo la condición (3.30), el dominio de velocidades realizables del órgano terminal de este manipulador está de restringido por la siguiente expresión [26]:

$$\boldsymbol{v_n}^T\left(\boldsymbol{J_n}\boldsymbol{J_n}^T\right)^{-1}\boldsymbol{v_n} \leq \mathbf{1} \quad , \tag{3.31}$$

donde $\boldsymbol{v_n}$ es el vector de velocidad lineal del punto de interés del órgano terminal y $\boldsymbol{J_n}$ es la matriz jacobiana básica de 3×3 del manipulador.



(*a*) Dominio de velocidades articulares (*b*) Elipsoide de manipulabilidad

Figura 3.3. Representaciones de los dominios de velocidad articular (*a*) y de velocidad operacional (*b*) del manipulador *3R* definidos por las restricciones (3.30) y (3.31).

La expresión (3.31) describe un elipsoide en el espacio Euclidiano de dimensión 3, al cual se define como el *elipsoide de manipulabilidad* [13] del robot. En la figura 3.3 se representan los dominios de velocidades articulares y de velocidad operacional definidos por las restricciones (3.30) y (3.31), en los espacios de velocidades articulares y de velocidad operacional del manipulador, respectivamente.

Se observa que, si el volumen del elipsoide de manipulabilidad es grande, las normas de los vectores de velocidad del órgano terminal, $\boldsymbol{v_n}$, también serán grandes en cualquier dirección. Esto significa que existe una conversión eficiente de las velocidades articulares en velocidad del órgano terminal. Inversamente, un elipsoide pequeño obtenido a partir de las mismas velocidades articulares implica que la mencionada conversión es ineficiente. La forma y tamaño del elipsoide dependen de los elementos de la matriz jacobiana y por lo tanto de la postura instantánea del manipulador. En consecuencia, el volumen del elipsoide

de manipulabilidad puede interpretarse como una medida de la eficacia con la cual la postura del manipulador transforma las velocidades articulares en velocidad del órgano terminal en cualquier dirección. Dicha transformación es óptima cuando la postura del manipulador agranda el volumen del elipsoide tanto como sea posible.

En general, para un robot de *n* grados de libertad, el elipsoide de manipulabilidad es de dimensión n. El volumen de dicho elipsoide en general se puede calcular mediante:

$$V = kw \quad , \tag{3.32}$$

donde k es un escalar que depende de la dimensión de la tarea del manipulador, mientras que la variable *w* se calcula mediante la siguiente ecuación:

$$w = \sqrt{det\, \boldsymbol{J_n J_n}^{\boldsymbol{T}}} \quad . \tag{3.33}$$

Debido a que k es constante, se tiene que la maximización del volumen del elipsoide depende de la variable *w*, que es definida por Yoshikawa como el índice de manipulabilidad. Considerando lo anterior, la manipulabilidad permite evaluar globalmente la facilidad de una postura de un manipulador para convertir velocidades articulares en velocidad del órgano terminal.

Yoshikawa, por otra parte, propuso índices para cuantificar de manera independiente las manipulabilidades traslacional y rotacional en robots de 6 gdl, como el mostrado en la figura 3.4. Bajo este enfoque, si se desea evaluar tales índices en un manipulador, éste debe satisfacer las siguientes condiciones:

- El manipulador es considerado como integrado por dos sub-cadenas cinemáticas: la primera asociada al brazo y la otra a la muñeca. El brazo contiene n_A articulaciones $(n_A \geq 3)$ y la muñeca contiene n_W articulaciones $(n_W \geq 3)$. Todas las articulaciones de la muñeca son rotacionales, y sus ejes son concurrentes.

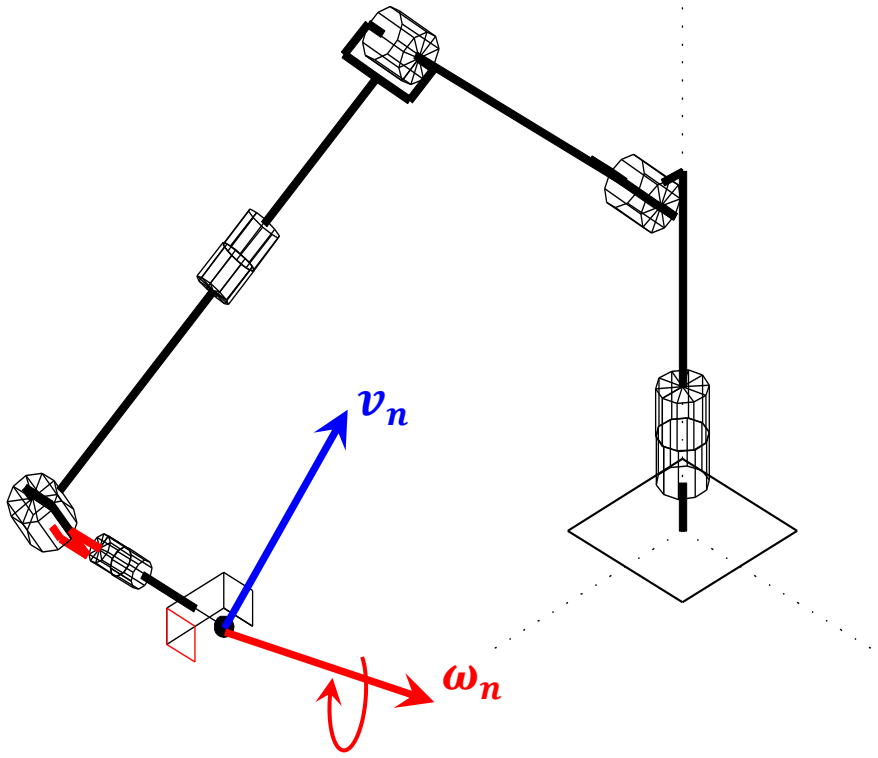


Figura 3.4. Esquema cinemático de un robot *6R*, de 6 gdl. Se representan los vectores de velocidad lineal y velocidad angular en el órgano terminal.

- Adicionalmente, a cada eslabón del manipulador se le asigna un marco de referencia ortonormal. Los marcos y eslabones se numeran del 0 (en la base) al n (en el órgano terminal), y se verifica que $n \geq 6$ ($n = n_A + n_W$). El eslabón 0 se conecta al 1 mediante la articulación 1, y así sucesivamente. El vector unitario z_i del marco i ($i = 1, \ldots, n$) coincide con el eje de la articulación i si ésta es rotacional (tipo R), y es paralelo a ese eje si la articulación es prismática (tipo P). El vector x_i del mismo marco se define alineado sobre la perpendicular común a los ejes de las articulaciones i e $i+1$.

Para manipuladores que satisfacen las condiciones precedentes, la manipulabilidad traslacional (W_A) del brazo y la manipulabilidad rotacional (W_W) de la muñeca del manipulador, se definen, respectivamente, como:

$$W_A = \sqrt{det(\boldsymbol{J_{TA}} \boldsymbol{J_{TA}^T})} \quad , \tag{3.34}$$

$$W_W = \sqrt{det(\boldsymbol{J_{RW}} \boldsymbol{J_{RW}^T})} \quad , \tag{3.35}$$

donde $\boldsymbol{J_{TA}}$ y $\boldsymbol{J_{RW}}$ son submatrices de la matriz jacobiana básica, definidas como:

$$\boldsymbol{J_{TA}} = [\boldsymbol{J_{TA1}} \quad \boldsymbol{J_{TA2}} \quad \dots \quad \boldsymbol{J_{TAnA}}] \quad , \tag{3.36}$$

$$\boldsymbol{J_{RW}} = [\boldsymbol{z}_{nA+1} \quad \boldsymbol{z}_{nA+2} \quad \dots \quad \boldsymbol{z}_n] \quad , \tag{3.37}$$

con:

$$\boldsymbol{J_{TAi}} = \begin{cases} \boldsymbol{z_i} \times \boldsymbol{p_{iw}} & \text{si la articulacion es de tipo R} \\ \boldsymbol{z_i} & \text{si la articulacion es de tipo P} \end{cases} \tag{3.38}$$

En la figura 3.5 se muestra esquemáticamente la ubicación de las matrices $\boldsymbol{J_{TA}}$ y $\boldsymbol{J_{RW}}$ dentro de una matriz jacobiana $\in \boldsymbol{R}^{6\times 6}$. En la ecuación (3.38), $\boldsymbol{p_{iw}}$ $(i = 1, \dots, n_A)$ es un vector con su origen en el punto O_i (el origen del marco $\boldsymbol{i}$ unido al eslabón $\boldsymbol{i}$) y extremo final en el punto de control O_w (el punto de concurrencia de los ejes de las articulaciones de la muñeca).

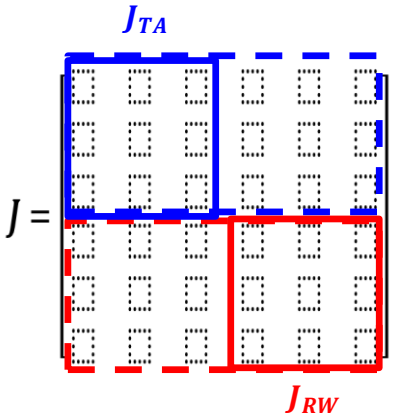


Figura 3.5. Sub-matrices $\boldsymbol{J_{TA}}$ y $\boldsymbol{J_{RW}}$ de la matriz Jacobiana.

4. Formulación para la planificación de movimientos óptimos

En este capítulo se presenta una formulación para resolver el problema de la planificación de movimientos óptimos de un robot industrial para la ejecución de una tarea con el auxilio de una mesa cooperante de rotaciones secuenciales.

4.1 Descripción del robot *7Bot*

El robot industrial que se utiliza en este trabajo para la aplicación del método propuesto de planificación de movimientos óptimos es el *7Bot*. Se trata de un robot de arquitectura *6R* (6 gdl), con una arquitectura idéntica y con las mismas características de programación que las un robot industrial, pero a escala reducida, para su utilización a nivel de laboratorio. Los eslabones del *7Bot* están hechos de aluminio. Sus actuadores son servomotores metálicos de alto par que proporcionan una fiabilidad robusta y un rendimiento excepcional en precisión y velocidad [27]. En la figura 4.1 se muestra una fotografía del robot, y en la tabla 4.1 se proporcionan algunas de sus características técnicas. En la tabla 4.2 se presentan las principales características técnicas de los servomotores. En la figura 4.2 se aprecian las secciones longitudinal y transversal del espacio de trabajo del robot.

Figura 4.1. Fotografía del robot *7Bot* (cortesia de 7Bot Robotics).

Tabla 4.1 Características del robot 7Bot

Característica	*Valor*
Peso	2.5 kg
Capacidad de carga	500 g
Alcance	434 mm
Alimentación	7.5V 6A DC
Potencia	60 W
Grados de libertad	6

Tabla 4.2 Características de los servomotores del *7Bot*

Motor	*Rango de trabajo*	*Velocidad Máxima*	*Torque*
1	180°	0.17 seg/60°	42Kg.cm 7.4V
2	180°	0.17 seg/60°	42Kg.cm 7.4V
3	180°	0.17 seg/60°	42Kg.cm 7.4V
4	180°	0.17 seg/60°	42Kg.cm 7.4V
5	180°	0.09 seg/60°	4.5Kg.cm 7.4V
6	180°	0.09 seg/60°	4.5Kg.cm 7.4V

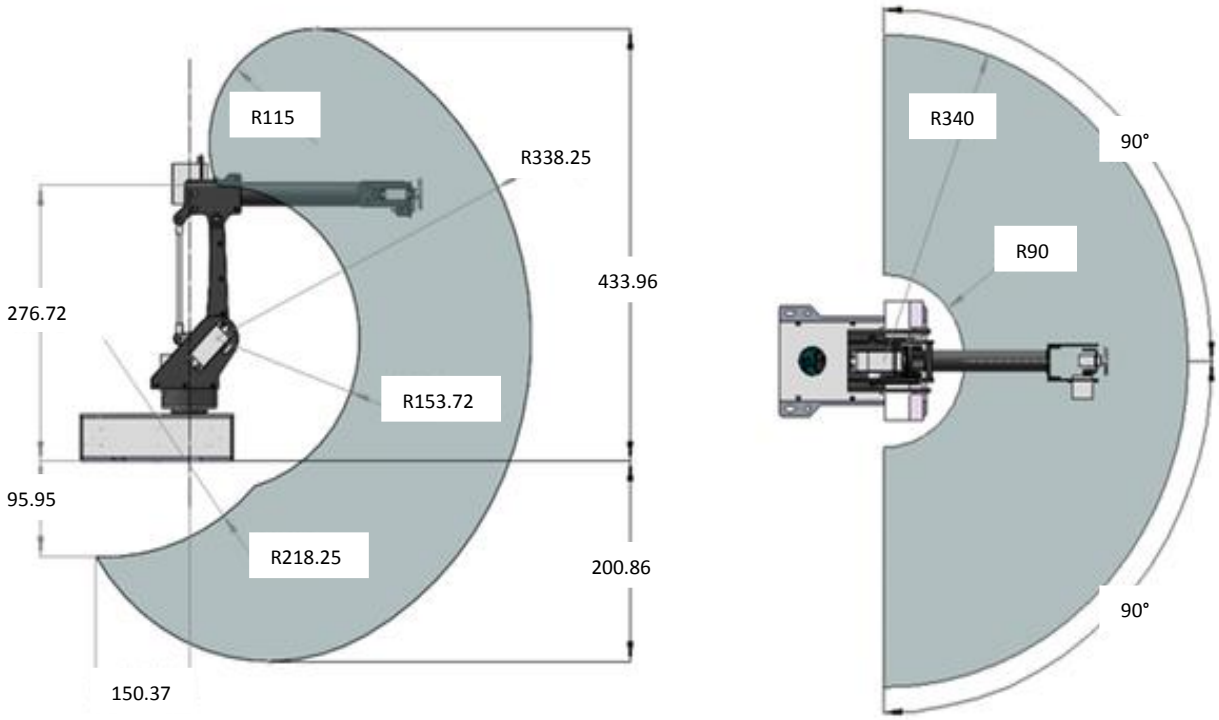


Figura 4.2. Secciones transversal y longitudinal del espacio de trabajo del robot *7Bot* (cortesia de *7Bot* Robotics).

El *7Bot* cuenta con una es una tarjeta de desarrollo *Arduino Due* basada en un microcontrolador de 32 bit CortexM3 ARM el cual puede ser programado mediante el IDE de Arduino. En la tabla 4.1 se listan las características de la tarjeta Arduino [28].

4.2 Estación de trabajo del *7Bot*

La estación de trabajo del robot *7Bot* está constituida por el robot mismo y una mesa cooperante rotatoria, circular, de 200 mm de diámetro y 80 mm de altura. La ubicación relativa entre la base del robot y la mesa se aprecia en la figura 4.3. Esta figura muestra modelos de alambre del robot y de la mesa generados en Matlab©.

Tabla 4.3 Características de la tarjeta *Arduino Due*

Parámetro	*Valor*
Microcontrolador	AT91SAM3X8E
Voltaje de operación	3.3V
Voltaje recomendado de entrada	(pin Vin) 7-12V
Pines de entrada y salida digitales	54 pines I/O, 12 salidas PWM
Pines de entrada análogos	12
Pines de salida análogos	2
Corriente de salida total en los pines I/O	130mA
Corriente DC máxima en el pin de 3.3V	800mA
Corriente DC maxima en el pin de 5V	800mA
Memoria Flash	512 KB
SRAM	96 KB (64KB + 32KB)
Velocidad de reloj	84 MHz

En la figura 4.3 se considera una antorcha de soldadura de arco como herramienta del robot. Asimismo, se representa un modelo de una pieza metálica cilíndrica a la que deberá de aplicarse un cordón de soldadura en forma de hélice circular sobre su superficie externa. En esta figura se consideró arbitrariamente el emplazamiento de la pieza en el centro de la mesa. En la figura 4.4 se muestra la ruta deseada del cordón de soldadura así como los marcos ortonormales de referencia Σ_w, Σ_t, y Σ_p unidos rígidamente a la estación de trabajo, a la mesa cooperante y a la base del tubo, respectivamente. El origen del marco Σ_w se define en algún punto conveniente de la estación de trabajo, por lo que a dicho marco se le llamará *marco de la estación de trabajo*. El origen del marco Σ_t se define en el centro del círculo de la mesa y el origen del marco Σ_p se define en el centro de la base del tubo.

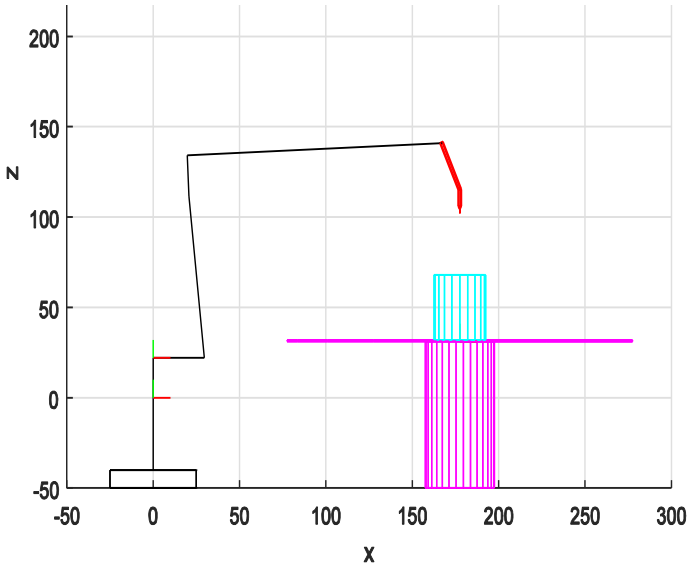


Figura 4.3. Ubicación relativa del robot y la mesa de posicionamiento en la estación de trabajo. Escala en mm.

En la figura 4.5 se indican los parámetros que permiten definir la ubicación del cordón de soldadura con respecto al marco de la mesa cooperante.

Las dimensiones del tubo están definidas por su longitud h_p y su radio exterior r_p. Este radio es el mismo de la hélice circular. Las coordenadas que definen la ubicación del tubo con respecto al marco auxiliar Σ_w, (figura 4.5) son la distancia r_e y el ángulo ψ_t. El marco Σ_w, tiene la misma orientación del marco Σ_w de la estación de trabajo, pero el origen de Σ_w, coincide con el de Σ_t. El marco Σ_w se hace coincidir con el Σ_0 , el cual está unido a la base del robot, conforme a la convención de Denavit-Hartenberg modificada. Como se observa en la figura 4.5, el ángulo ψ_t define la rotación de la mesa y el ángulo ψ_p establece el punto de inicio de la soldadura del cordón. Asumiendo que el emplazamiento relativo robot / mesa es conocido, la ubicación de la soldadura con respecto al marco Σ_0 de la base del robot sólo depende de las coordenadas r_e, ψ_t y ψ_p, las cuales se deben calcular de manera que se optimice un índice de desempeño cinemático del robot durante la ejecución de la tarea.

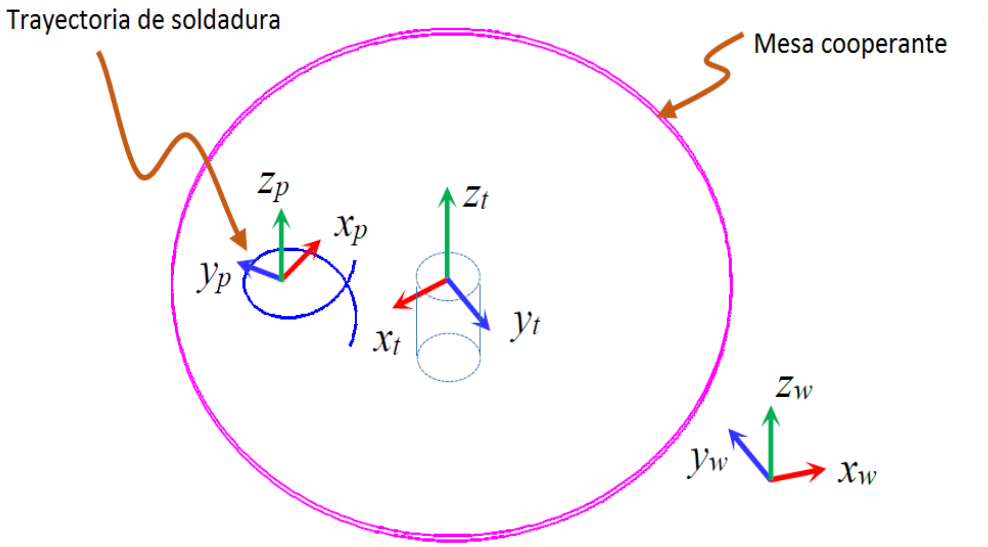


Figura 4.4. Vista en perspectiva de la trayectoria helicoidal deseada para la antorcha de soldadura en la mesa cooperante.

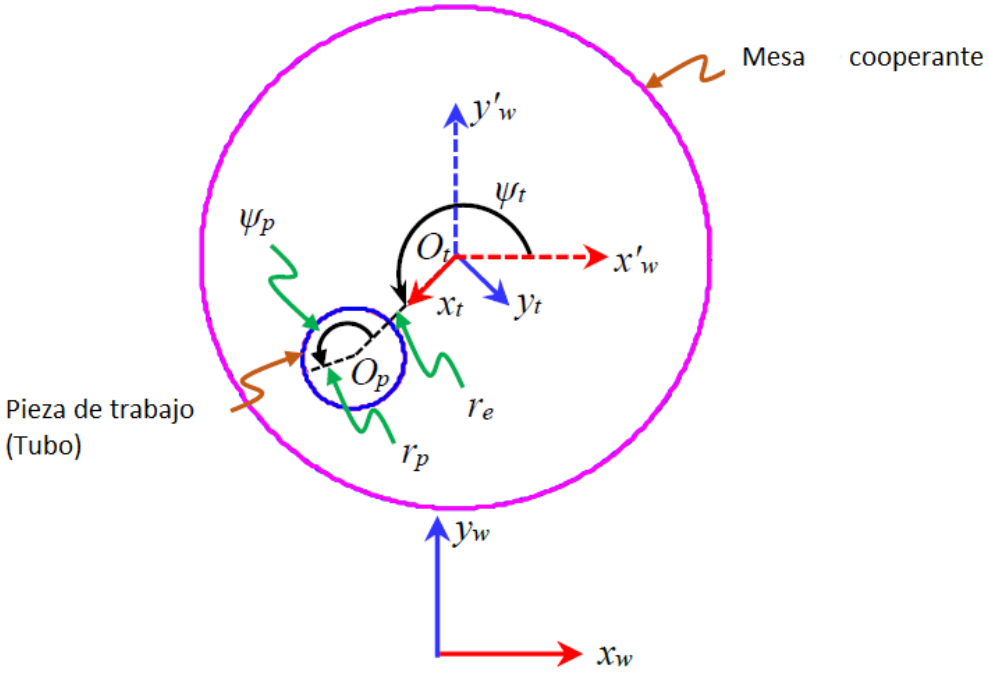


Figura 4.5. Vista superior de la estación de trabajo. Ubicación de la pieza de trabajo en la mesa cooperante.

4.3 Modelado cinemático del robot

4.3.1 Modelo directo de posición

El esquema cinemático del robot *7bot* se aprecia en la figura 4.6 y sus parámetros geométricos se presentan en la tabla 4.4. En la tabla 4.5 se dan los valores numéricos de los parámetros.

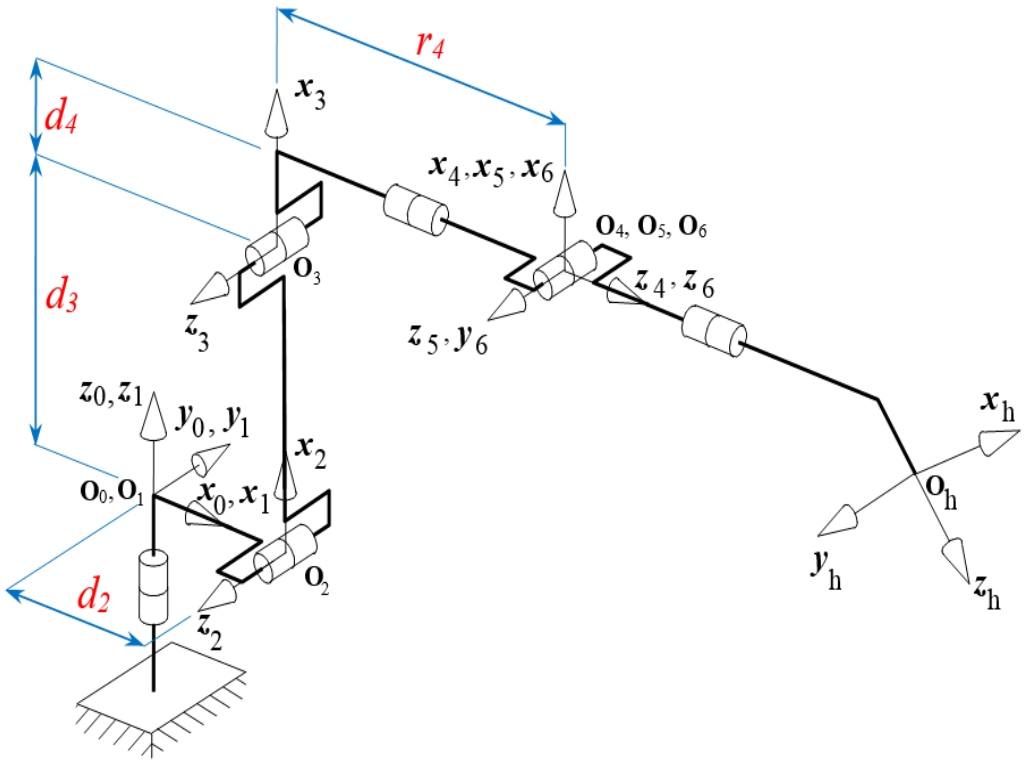


Figura 4.6. Esquema cinemático del robot *7bot*.

Tabla 4.4 Parámetros geométricos del *7Bot*. Convención modificada de Denavit-Hartenberg

j	σ	α	d	θ	r (cm)
1	0	0°	0	θ_1	0
2	0	90°	d_2	θ_2	0
3	0	0°	d_3	θ_3	0
4	0	90°	d_4	θ_4	r_4
5	0	-90°	0	θ_5	0
6	0	90°	0	θ_6	0

Tabla 4.5 Valores numéricos de los parámetros geométricos del *7Bot*

Parámetro	*7Bot escala real*	*7Bot escala industrial*
d_2	4 cm.	30 cm
d_3	12 cm.	90 cm
d_4	3 cm.	22.5 cm
r_4	19.85 cm.	148.9 cm

Para propósitos de las simulaciones consideradas en este trabajo, se incrementarán los parámetros d y r a una escala de 1:7.5, de tal manera que se obtengan las dimensiones de un robot equiparable a un manipulador industrial mediano. Así, se podrá trabajar el problema como si se tratara el de un robot industrial. Las dimensiones del robot escalado a nivel industrial se muestran también en la tabla 4.5.

Las ecuaciones del modelo directo de posición se obtienen mediante el uso del paquete SYMORO+ (acrónimo de *SYmbolic MOdelling of RObots*), que se describe en el Apéndice A. Las matrices elementales del robot que calcula el SYMORO son las siguientes:

$$ {}^{0}_{1}T = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^{1}_{2}T = \begin{bmatrix} c_2 & -s_2 & 0 & d_2 \\ 0 & 0 & -1 & 0 \\ s_2 & c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^{2}_{3}T = \begin{bmatrix} c_3 & -s_3 & 0 & d_3 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(4.1)

$$ {}^{3}_{4}T = \begin{bmatrix} c_4 & -s_4 & 0 & d_4 \\ 0 & 0 & -1 & -r_4 \\ s_4 & c_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^{4}_{5}T = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_5 & -c_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^{5}_{6}T = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

donde $c_1 \equiv \cos\theta_1$ y $s_1 \equiv \sin\theta_1$.

Por otra parte, teniendo en cuenta la geometría de la antorcha industrial utilizada en el robot, se sabe que la matriz de transformación homogénea del marco de la articulación 6 al marco de la herramienta es:

$$ {}^{6}_{h}T = \begin{bmatrix} 0.9239 & 0 & -0.3827 & -5 \\ 0 & 1 & 0 & 0 \\ 0.3827 & 0 & 0.9239 & 40 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2} $$

Para calcular la matriz de $\mathbf{{}^{0}_{h}T}$, que permite determinar las coordenadas operacionales del marco Σ_h con respecto al marco Σ_0, se realiza la siguiente multiplicación:

$$ {}^{0}_{h}T = {}^{0}_{1}T \ {}^{1}_{2}T \ {}^{2}_{3}T \ {}^{3}_{4}T \ {}^{4}_{5}T \ {}^{5}_{6}T \ {}^{6}_{h}T \quad . \tag{4.3} $$

4.3.2 Modelo inverso de posición

Para el cálculo del modelo inverso de posición se utiliza el paquete SYMORO+. A este paquete se le suministran los parámetros geométricos del robot, y se asume que se conocen todos los elementos de la matriz $\boldsymbol{snap}$ o $\boldsymbol{U_0}$, la cual describe la pose deseada del marco 6 del robot con respecto a su marco Σ_0, unido a la base. Esta matriz se calcula mediante el siguiente producto:

$$ \mathbf{U_0} = {}^{0}_{w}T \ {}^{w}_{t}T \ {}^{t}_{p}T \ {}^{p}_{h}T \ {}^{h}_{6}T \quad , \tag{4.4} $$

y sus elementos son designados de la siguiente manera para el cálculo de las ecuaciones del modelo inverso de posición:

$$ \mathbf{U_0} = \begin{bmatrix} s_x & n_x & a_x & p_x \\ s_y & n_y & a_y & p_y \\ s_z & n_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad . \tag{4.5} $$

Como resultado de la aplicación del método de Paul en el SYMORO, para la resolución del modelo inverso de posición, se obtienen las ecuaciones que se listan en la siguiente página. En esas ecuaciones, las variables t_i ($i=1,2,\ldots,6$) se usan para designar a las variables articulares θ_i ($i=1,2,\ldots,6$).

$$
\begin{aligned}
e &= 1;\\
t_1 &= atan2(p_y, p_x);\\
Z_1 &= -d_2 + p_x \cos t_1 + p_y \sin t_1 ;\\
B_1 &= 2(-(d_4 P_z) - r_4 Z_1);\\
B_2 &= 2(p_z r_4 - d_4 Z_1);\\
B_3 &= d_3^2 - d_4^2 - p_z^2 - r_4^2 - Z_1^2;\\
SQ &= \frac{B_1 B_3 + B_2 \sqrt{B_1^2 + B_2^2 - B_3^2}\, e}{B_1^2 + B_2^2};\\
CQ &= \frac{B_2 B_3 - B_1 \sqrt{B_1^2 + B_2^2 - B_3^2}\, e}{B_1^2 + B_2^2};\\
t_2 &= atan2\left[\frac{(p_z + r_4 \cos(atan2(SQ, CQ)) - d_4 \sin(atan2(SQ, CQ)))}{d_3},\right.\\
&\left.\frac{(Z_1 - d_4 \cos(atan2(SQ, CQ)) - r_4 \sin(atan2(SQ, CQ)))}{d_3}\right);\\
t_3 &= atan2(SQ, CQ) - t_2;\\
X &= -(a_y \cos(t_1)) + a_x \sin(t_1);\\
Y &= -(a_x \cos(t_1) \cos(t_2 + t_3)) - a_y \cos(t_2 + t_3) \sin(t_1) - a_z \sin(t_2 + t_3)\\
t_4 &= atan2(-X, Y);\\
Y12 &= \cos(t_4)\left[a_x \cos(t_1) \cos(t_2 + t_3) + a_y \cos(t_2 + t_3) \sin(t_1)\right.\\
&\qquad \left. + a_z \sin(t_2 + t_3)\right] + \left(a_y \cos(t_1)\right) - a_x \sin(t_1) \sin(t_4);\\
Y1 &= a_z \cos(t_2 + t_3) - a_x \cos(t_1) \sin(t_2 + t_3) - a_y \sin(t_1) \sin(t_2 + t_3);\\
t_5 &= atan2(-Y12, -Y1);\\
Y22 &= \cos(t_4)\left[s_y \cos(t_1) - s_x \sin(t_1)\right] + (s_x \cos(t_1) \cos(t_2 + t_3)\\
&\qquad + s_y \cos(t_2 + t_3) \sin(t_1) + s_z \sin(t_2 + t_3)) \sin(t_4);\\
Y21 &= \cos(t_4)\left[n_y \cos(t_1) - n_x \sin(t_1)\right] + (n_x \cos(t_1) \cos(t_2 + t_3)\\
&\qquad + n_y \cos(t_2 + t_3) \sin(t_1) + n_z \sin(t_2 + t_3)) \sin(t_4);\\
t_6 &= atan2(-Y22, -Y21);
\end{aligned}
$$

(4.6)

Los elementos de la matriz jacobiana también se determinan mediante la ayuda del paquete SYMORO +. Las siguientes son las expresiones obtenidas:

$J(1,1) = -(d_2\, sin(t_1)) - d_3\, cos(t_2)\, sin(t_1) - d_4\, cos(t_2 + t_3)\, sin(t_1) - r_4\, sin(t_1)\, sin(t_2 + t_3)$;
$J(2,1) = d_2\, cos(t_1) + d_3\, cos(t_1)\, cos(t_2) + d_4\, cos(t_1)\, cos(t_2 + t_3) + r_4\, cos(t_1)\, sin(t_2 + t_3)$;
$J(3,1) = 0$;
$J(4,1) = 0$;
$J(5,1) = 0$;
$J(6,1) = 1$;
$J(1,2) = r_4\, cos(t_1)\, cos(t_2 + t_3) - d_3\, cos(t_1)\, sin(t_2) - d_4\, cos(t_1)\, sin(t_2 + t_3)$;
$J(2,2) = r_4\, cos(t_2 + t_3)\, sin(t_1) - d_3\, sin(t_1)\, sin(t_2) - d_4\, sin(t_1)\, sin(t_2 + t_3)$;
$J(3,2) = d_3\, cos(t_2) + d_4\, cos(t_2 + t_3) + r_4\, sin(t_2 + t_3)$;
$J(4,2) = sin(t_1)$;
$J(5,2) = -\, cos(t_1)$;
$J(6,2) = 0$;
$J(1,3) = r_4\, cos(t_1)\, cos(t_2 + t_3) - d_4\, cos(t_1)\, sin(t_2 + t_3)$;
$J(2,3) = r_4\, cos(t_2 + t_3)\, sin(t_1) - d_4\, sin(t_1)\, sin(t_2 + t_3)$;
$J(3,3) = d_4\, cos(t_2 + t_3) + r_4\, sin(t_2 + t_3)$;
$J(4,3) = sin(t_1)$;
$J(5,3) = -\, cos(t_1)$;
$J(6,3) = 0$;
$J(1,4) = 0$;
$J(2,4) = 0$;
$J(3,4) = 0$;
$J(4,4) = cos(t_1)\, sin(t_2 + t_3)$;
$J(5,4) = sin(t_1)\, sin(t_2 + t_3)$;
$J(6,4) = -\, cos(t_2 + t_3)$;
$J(1,5) = 0$;
$J(2,5) = 0$;
$J(3,5) = 0$;
$J(4,5) = cos(t_4)\, sin(t_1) - cos(t_1)\, cos(t_2 + t_3)\, sin(t_4)$;
$J(5,5) = -(cos(t_1)\, cos(t_4)) - cos(t_2 + t_3)\, sin(t_1)\, sin(t_4)$;
$J(6,5) = -(sin(t_2 + t_3)\, sin(t_4))$;
$J(1,6) = 0$;
$J(2,6) = 0$;
$J(3,6) = 0$;
$J(4,6) = cos(t_1)\, cos(t_5)\, sin(t_2 + t_3) + cos(t_1)\, cos(t_2 + t_3)\, cos(t_4)\, sin(t_5) + sin(t_1)\, sin(t_4)\, sin(t_5)$;
$J(5,6) = cos(t_5)\, sin(t_1)\, sin(t_2 + t_3) + cos(t_2 + t_3)\, cos(t_4)\, sin(t_1)\, sin(t_5) - cos(t_1)\, sin(t_4)\, sin(t_5)$;
$J(6,6) = -(cos(t_2 + t_3)\, cos(t_5)) + cos(t_4)\, sin(t_2 + t_3)\, sin(t_5)$;

Estos elementos se incorporan en la matriz jacobiana como se indica en la siguiente expresión:

$$\boldsymbol{J} = \begin{bmatrix} J(1,1) & J(1,2) & J(1,3) & J(1,4) & J(1,5) & J(1,6) \\ J(2,1) & J(2,2) & J(2,3) & J(2,4) & J(2,5) & J(2,6) \\ J(3,1) & J(3,2) & J(3,3) & J(3,4) & J(3,5) & J(3,6) \\ J(4,1) & J(4,2) & J(4,3) & J(4,4) & J(4,5) & J(4,6) \\ J(5,1) & J(5,2) & J(5,3) & J(5,4) & J(5,5) & J(5,6) \\ J(6,1) & J(6,2) & J(6,3) & J(6,4) & J(6,5) & J(6,6) \end{bmatrix} \tag{4.7}$$

4.4 Índices de desempeño a optimizar

El criterio de desempeño que se establece en el presente trabajo de tesis, para su optimización mediante la planificación de movimientos del robot y su mesa cooperante, se refiere a la facilidad de la cadena cinemática del robot para convertir velocidades articulares en velocidades lineal y angular del órgano terminal en todas direcciones. En la medida en que el robot tenga una buena facilidad para efectuar esa conversión de velocidades, todos los motores trabajarán de manera eficiente, con lo que se consiguen las mejores condiciones de funcionamiento de éstos, mejorando con ello su vida útil. Los índices de desempeño que miden esa facilidad de conversión de velocidades articulares en velocidad del órgano terminal son las manipulabilidades traslacional y rotacional, que se presentaron en el Capítulo 3.

Para calcular las manipulabilidades traslacional y rotacional, se considera que el robot *7bot* está compuesto por dos sub-cadenas cinemáticas: la primera asociada al *brazo* y la otra asociada a la *muñeca*. El brazo tiene n_A articulaciones ($n_A = 3$) y la muñeca tiene n_W articulaciones ($n_W = 3$), el número de articulaciones actuadas es $n = n_A + n_W$. Así, la manipulabilidad traslacional (w_A) del brazo y la manipulabilidad rotacional (w_W) de la muñeca, se determinan respectivamente mediante:

$$W_A = \sqrt{\det(\boldsymbol{J_{TA}} \boldsymbol{J_{TA}^T})} \tag{4.8}$$

$$W_W = \sqrt{\det(\boldsymbol{J_{RW}} \boldsymbol{J_{RW}^T})} \tag{4.9}$$

En estas dos últimas ecuaciones, $\boldsymbol{J_{TA}}$ y $\boldsymbol{J_{RW}}$ son las submatrices de la matriz jacobiana básica $\boldsymbol{J}$ que se indican en la ecuación (4.10).

$$\boldsymbol{J} = \begin{bmatrix} J(1,1) & J(1,2) & J(1,3) & J(1,4) & J(1,5) & J(1,6) \\ J(2,1) & J(2,2) & J(2,3) & J(2,4) & J(2,5) & J(2,6) \\ J(3,1) & J(3,2) & J(3,3) & J(3,4) & J(3,5) & J(3,6) \\ J(4,1) & J(4,2) & J(4,3) & J(4,4) & J(4,5) & J(4,6) \\ J(5,1) & J(5,2) & J(5,3) & J(5,4) & J(5,5) & J(5,6) \\ J(6,1) & J(6,2) & J(6,3) & J(6,4) & J(6,5) & J(6,6) \end{bmatrix} \tag{4.10}$$

$\boldsymbol{J_{TA}}$ (primera submatriz 3×3) — $\boldsymbol{J_{RW}}$ (última submatriz 3×3)

El índice w_W está acotado de tal manera que $0 \leq w_W \leq 1$. Sin embargo, los valores de w_A dependen de las longitudes de los eslabones del manipulador. Por lo tanto, para hacer que ambos índices tengan la misma ponderación en un proceso de optimización, los valores de w_A son normalizados de la siguiente manera:

$$w_A^* = \frac{w_A}{w_{A\,max}} \tag{4.11}$$

donde $w_{A\,max}$ es el valor máximo de w_A que puede alcanzar el robot. Así, w_A^* también estará acotado de tal manera que $0 \leq w_A^* \leq 1$. Entre más grandes sean los valores de los índices w_W y w_A^* para una postura del robot, mayor será la facilidad que tenga éste para producir movimientos de traslación y rotación en la antorcha de soldadura. Si alguno de estos índices

es cero, entonces el manipulador tendrá una postura singular y perderá la capacidad de controlar arbitrariamente el movimiento del efector final. Claramente, en el proceso de planificación de movimientos se deberán determinar los sucesivos emplazamientos relativos robot / tarea que permitan alcanzar valores de w_W y w_A^* tan grandes como sea posible.

4.5 Descripción del algoritmo de optimización

El método propuesto aquí para determinar los sucesivos emplazamientos relativos robot / tarea que permitan mejorar los valores de w_W y w_A^* tanto como sea posible durante la ejecución de una tarea del robot, se basa en la optimización de una función de las variables r_e, ψ_t y ψ_p , descritas en la sección 4.2. Éstas, como se vio, determinan el emplazamiento del cordón de soldadura (y por lo tanto de la pieza de trabajo). La función objetivo que se propone para ser minimizada en el proceso de optimización se define como:

$$f = -(\overline{w} - w_\sigma) \quad , \tag{4.12}$$

donde $\overline{w}$ es la media de las manipulabilidades mixtas combinadas correspondientes a una muestra de n_p puntos de la trayectoria deseada de la antorcha de soldadura, y w_σ es la desviación estándar de la misma muestra. Se advierte que la diferencia $\overline{w} - w_\sigma$ representa una manipulabilidad mixta típicamente pequeña en el conjunto de las manipulabilidades mixtas de la muestra. En consecuencia, al minimizar la función de la Ec. (4.12), maximizamos globalmente todo el conjunto de manipulabilidades mixtas correspondientes a la tarea.

Nótese que las manipulabilidades dependen de las posturas requeridas del robot para lograr la tarea, y tales posturas están determinadas por la colocación de la pieza de trabajo definida por r_e, ψ_t y ψ_p. Así, la función f depende implícitamente de estas variables.

La manipulabilidad mixta asociada con el i-ésimo punto de la ruta deseada de la antorcha de soldadura se define como:

$$w_i \equiv \overline{w_i^*} - w_{\sigma i}^* \quad , \tag{4.13}$$

donde $\overline{w_i^*}$ es el promedio de la manipulabilidad traslacional normalizada w_{Ai}^* y la manipulabilidad rotacional w_{Wi} correspondientes a la postura en el i-ésimo punto de la trayectoria. Además, $w_{\sigma i}^*$ es la desviación estándar de ambas manipulabilidades en el mismo punto. Por lo tanto, la manipulabilidad mixta representa una manipulabilidad típicamente pequeña (cualquiera de ellas) del robot en el punto de la i-ésima ruta.

Para minimizar la función de la Ec. (4.12), el usuario propone una secuencia finita de rotaciones de la mesa cooperante para realizar la tarea. Es decir, se definen diferentes valores de ψ_t manteniendo fijos de r_e y ψ_p. Luego, se debe lograr un segmento del cordón de soldadura para cada rotación hasta completar la ruta deseada.

Teniendo en cuenta la función objetivo a minimizar, se propone el siguiente algoritmo para efectuar el proceso numérico de optimización aplicando un programa de optimización en Matlab©, para la minimización de funciones no lineales sujeta a restricciones no lineales:

1. Proponer un conjunto inicial de valores de las variables independientes r_e, ψ_t y ψ_p que definen el emplazamiento de la tarea con respecto a la base del robot.
2. Para el emplazamiento propuesto de la tarea, resolver el problema inverso de posición para una muestra suficientemente grande de $\boldsymbol{n_p}$ puntos de la ruta deseada de la antorcha del robot, y determinar las posturas del robot requeridas en cada uno de dichos puntos.
3. Calcular las manipulabilidades traslacional y rotacional del robot para cada postura correspondiente a los puntos de la muestra considerada.
4. Calcular la función objetivo a minimizar.
5. Valorar el emplazamiento propuesto y efectuar una prueba de convergencia.

6. Si el emplazamiento actual propuesto cumple el criterio de convergencia de la función de Matlab©, terminar el proceso de optimización; si no, proponer un emplazamiento mejorado y regresar al punto 2 del algoritmo.

El programa para la minimización de la función objetivo se basa en el método de región de confianza basado en técnicas de puntos interiores para programación no lineal [29].

5. Casos de estudio

En el problema de planificación de movimientos que se plantea aquí, se considera que la mesa cooperante y el robot deben efectuar movimientos secuenciales alternos hasta terminar toda la tarea. Durante la intervención del robot al ejecutar parcial o talmente el cordón de soldadura, la mesa permanece fija en una posición óptima. Al concluir el robot el cordón de soldadura, parcial o totalmente, la mesa rota a la siguiente posición óptima o se finaliza la tarea. El algoritmo de optimización es el que se describió en el capítulo precedente. El método se aplica en 3 casos de estudio en los que se deberá ejecutar una tarea que consiste en aplicar un cordón de soldadura siguiendo una ruta 3D.

5.1 Descripción de la tarea a realizar

La ruta a describir, como quedó establecido en el capítulo 4, es una hélice circular que deberá recorrer la antorcha de soldadura en un tiempo total de 40 segundos moviéndose a una velocidad uniforme. La orientación de la antorcha está definida por los ángulos de Euler φ, θ y ψ . Tales ángulos corresponden a las rotaciones sucesivas en el orden $\boldsymbol{z}-\boldsymbol{y}-\boldsymbol{z}$ que debe efectuar el marco Σ_h unido rígidamente a la antorcha de soldadura (figura 4.6), y cuya orientación coincide originalmente con la del marco Σ_p (figura 4.4) unido a la base del tubo, para llegar a la orientación deseada requerida por la tarea, con respecto al marco Σ_p. Los valores de estos ángulos para la pose de la antorcha en el

punto de partida son $\varphi = 45°,\ \theta = 135\ °$ y $\psi = 0°$. Los ángulos θ y ψ no cambian durante la tarea. Solo φ aumenta uniformemente hasta $405°$.

5.2 Caso 1

El caso 1 se propone como referencia para los otros dos casos. Se especifica el emplazamiento de la pieza de trabajo bajo un criterio intuitivo. Esta pieza se coloca en el centro de la mesa ($\boldsymbol{r_e} = \mathbf{0}$) con $\boldsymbol{\psi_t} = \mathbf{0}$, y comienza la trayectoria en un punto arbitrario definido por $\boldsymbol{\psi_p} = \mathbf{0}$. No se lleva a cabo ninguna optimización. Solo se efectúa un análisis de las manipulabilidades para la ejecución de la tarea con un solo cordón soldadura.

En la figura 5.1 se muestra una secuencia de 6 posturas del robot obtenidas durante la simulación de la ejecución de la tarea mediante Matlab© en el caso 1. En esa figura se observan sucesivas poses de la antorcha a través de la ruta helicoidal sobre la pieza de trabajo, y también se aprecian las posturas del robot correspondientes a diferentes momentos durante la ejecución de la tarea.

El comportamiento de las manipulabilidades rotacional y traslacional correspondiente a este caso se muestra en la figura 5.2, y las curvas que representan la historia de las variables articulares se aprecian en la figura 5.3.

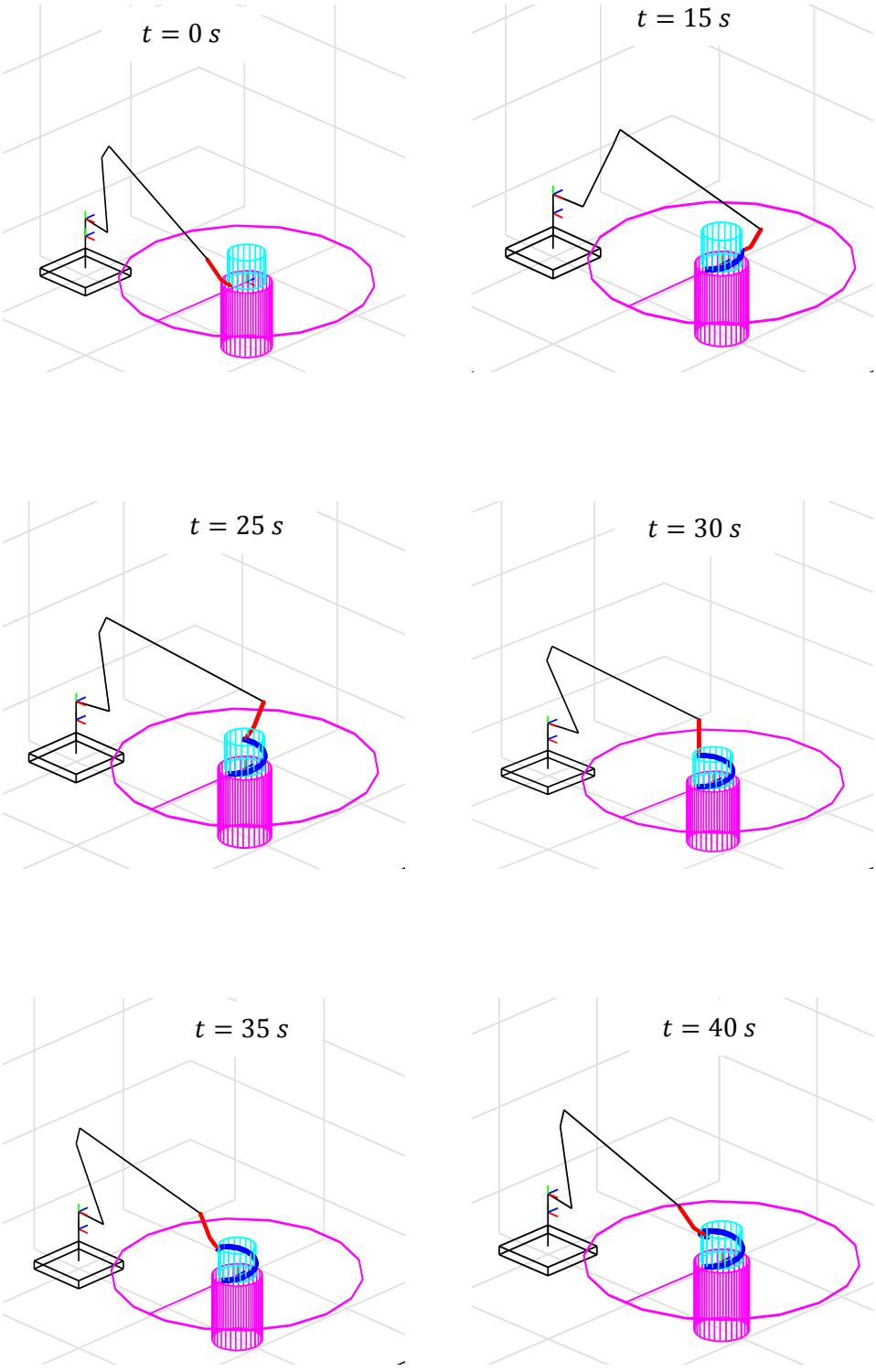


Figura 5.1. Caso 1. Secuencia de posturas del robot al ejecutar la tarea.

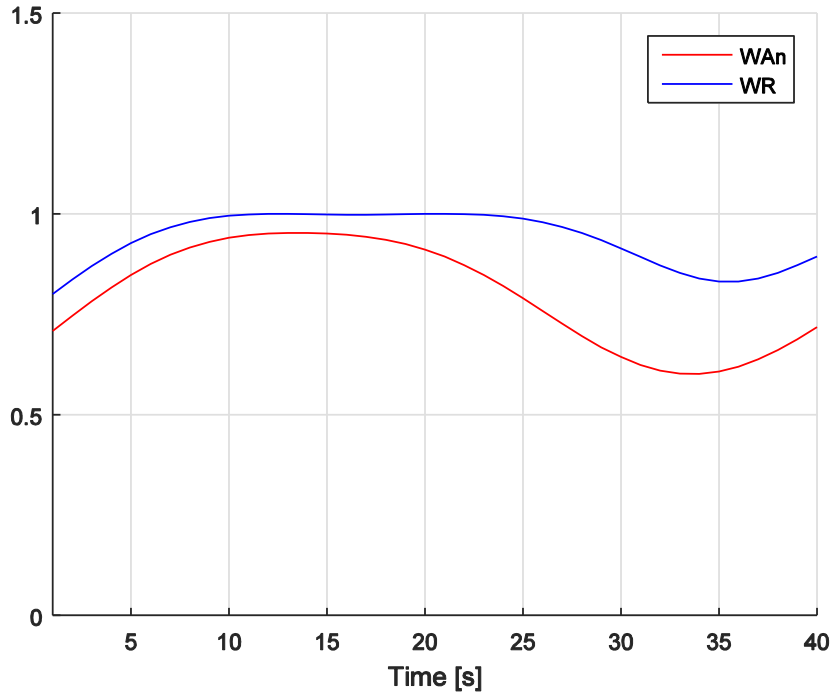


Figura 5.2. Caso 1. Manipulabilidades rotacional y traslacional normalizada.

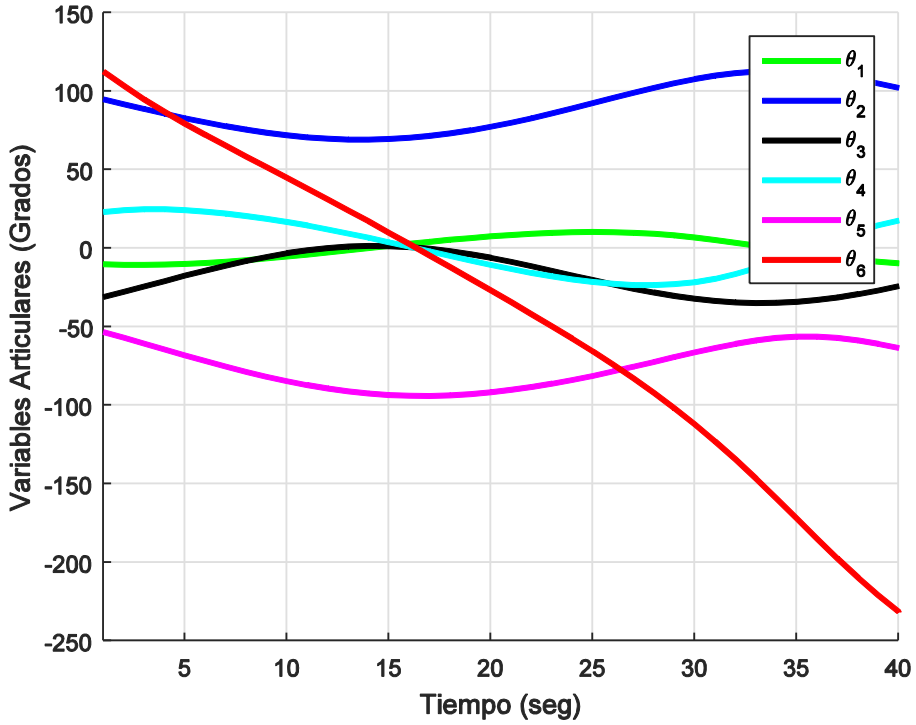


Figura 5.3. Caso 1. Historia de las variables articulares.

5.3 Caso 2

En el caso 2, se optimizan los valores de las variables independientes r_e, ψ_t y ψ_p para un solo cordón de soldadura. Como resultado del proceso de minimización de la función objetivo se obtienen los siguientes valores óptimos para estas variables: $r_e = 31.2785$ cm, $\psi_t = 4.6372°$ y $\psi_p = 22.1812°$. En la figura 5.4 se observa una muestra de posturas del robot durante la ejecución de la tarea, con la mesa cooperante en la posición óptima, obtenidas mediante la simulación del movimiento del robot en el paquete Matlab©. En esta figura se observan las poses correspondientes de la antorcha en diferentes puntos del cordón helicoidal de soldadura sobre la pieza de trabajo.

El comportamiento de las manipulabilidades rotacional y traslacional correspondiente a este caso se muestra en la figura 5.5, y las curvas que describen la historia de las velocidades articulares se aprecian en la figura 5.6.

5.4 Caso 3

En el Caso 3 se considera que el cordón de soldadura completo a aplicar se compone de dos sub-cordones, y que cada uno de ellos deberá ejecutarse con una posición óptima distinta de la mesa. Así, las variables a optimizar son r_e y ψ_p para los dos sub-cordones, además de ψ_{t1} para el primer sub-cordón, y ψ_{t2} para el segundo sub-cordón. Nótese que r_e y ψ_p definen la posición de la pieza de trabajo sobre la mesa, mientras que $\psi_{t\,i}$ *(i=1, 2)* determina la posición angular de la mesa durante la ejecución del sub-cordón i.

Mediante el proceso de minimización de la función objetivo se obtienen los siguientes valores óptimos para estas variables: $r_e = -40$ *cm* y $\psi_p = -54\,°$, $\psi_{t1} = 120\,°$ y $\psi_{t2} = -103.5\,°$. Después de aplicar el primer cordón, el robot tiene que esperar la rotación de la mesa y luego continuar con la tarea de soldadura para el segundo cordón.

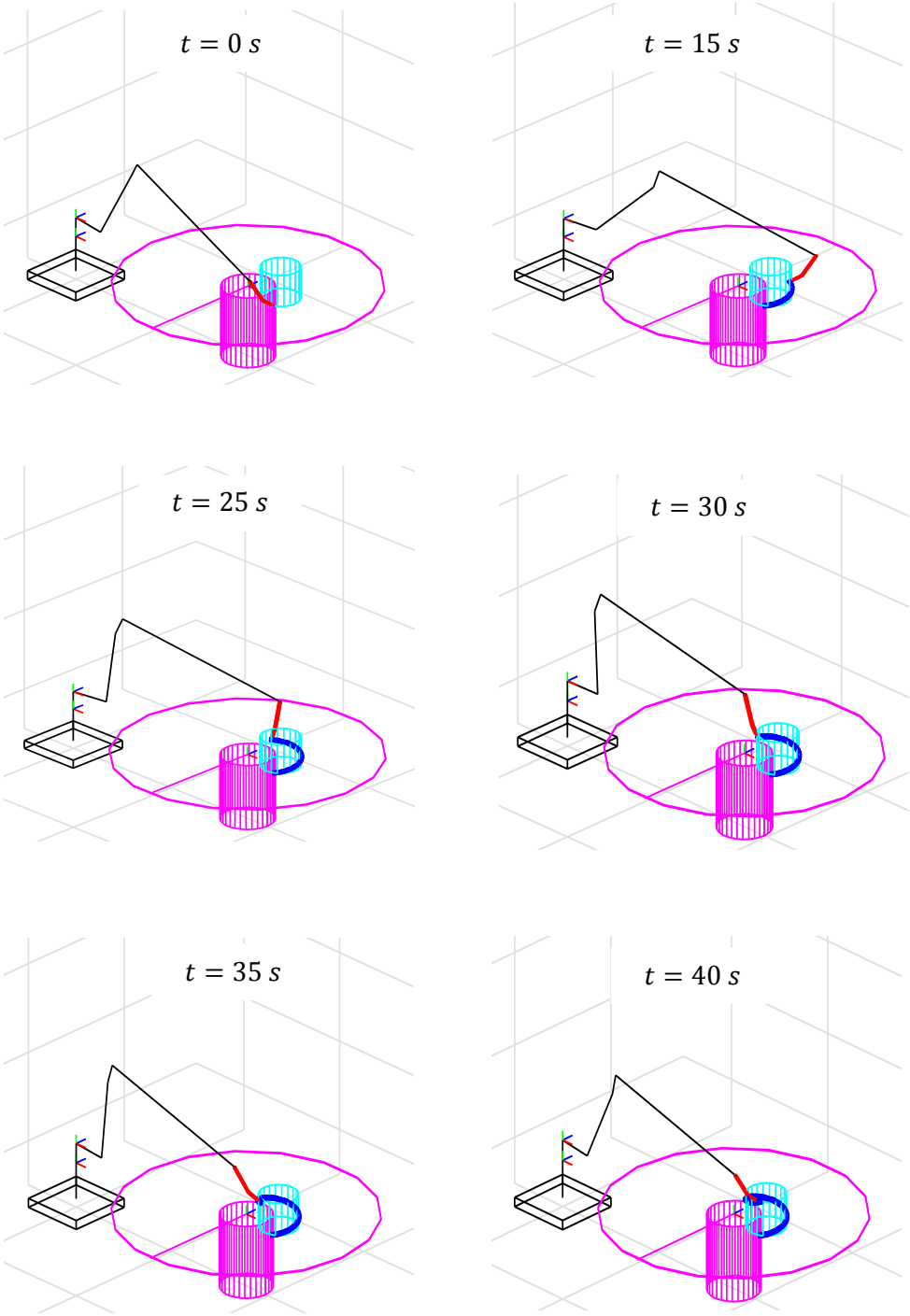


Figura 5.4. Caso 2. Secuencia de posturas del robot al ejecutar la tarea.

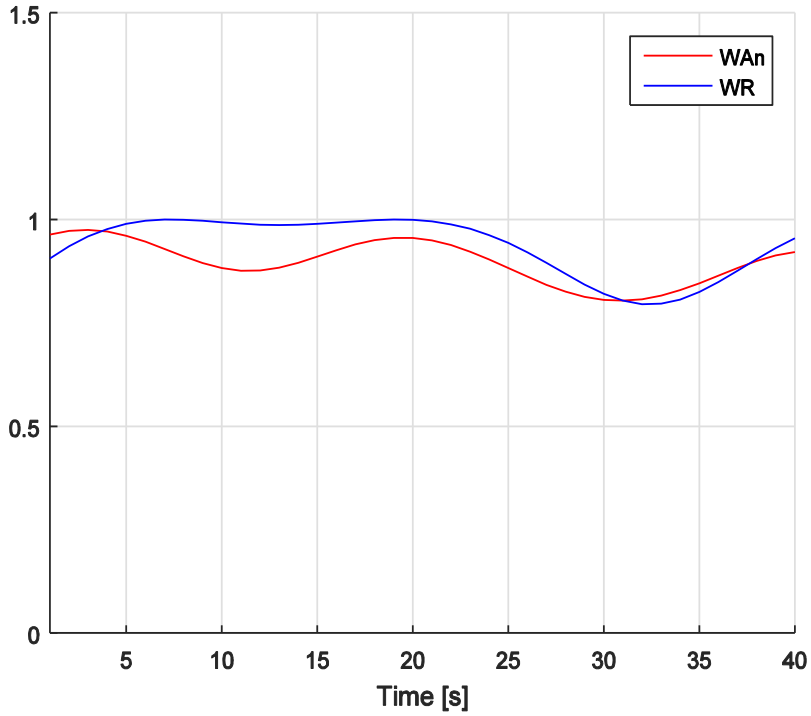


Figura 5.5. Caso 2. Manipulabilidades rotacional y traslacional normalizada.

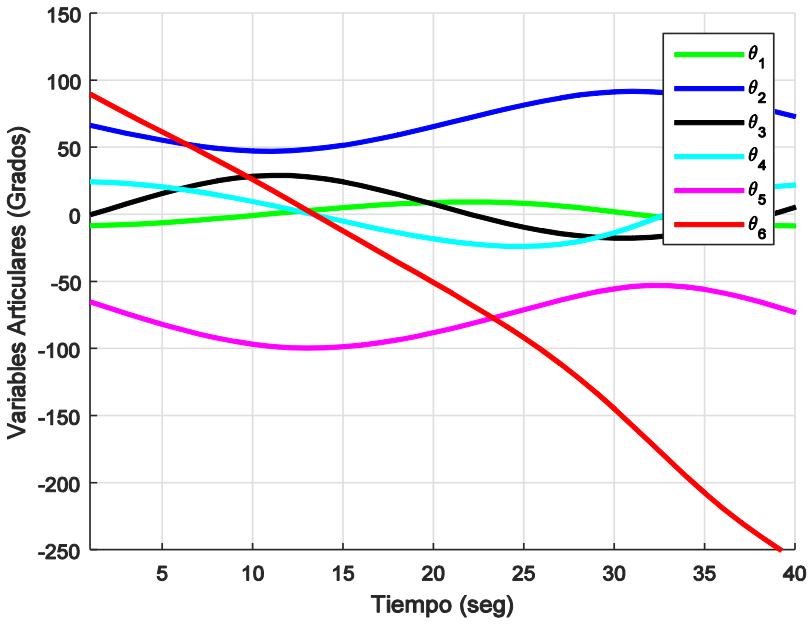


Figura 5.6. Caso 2. Historia de las variables articulares.

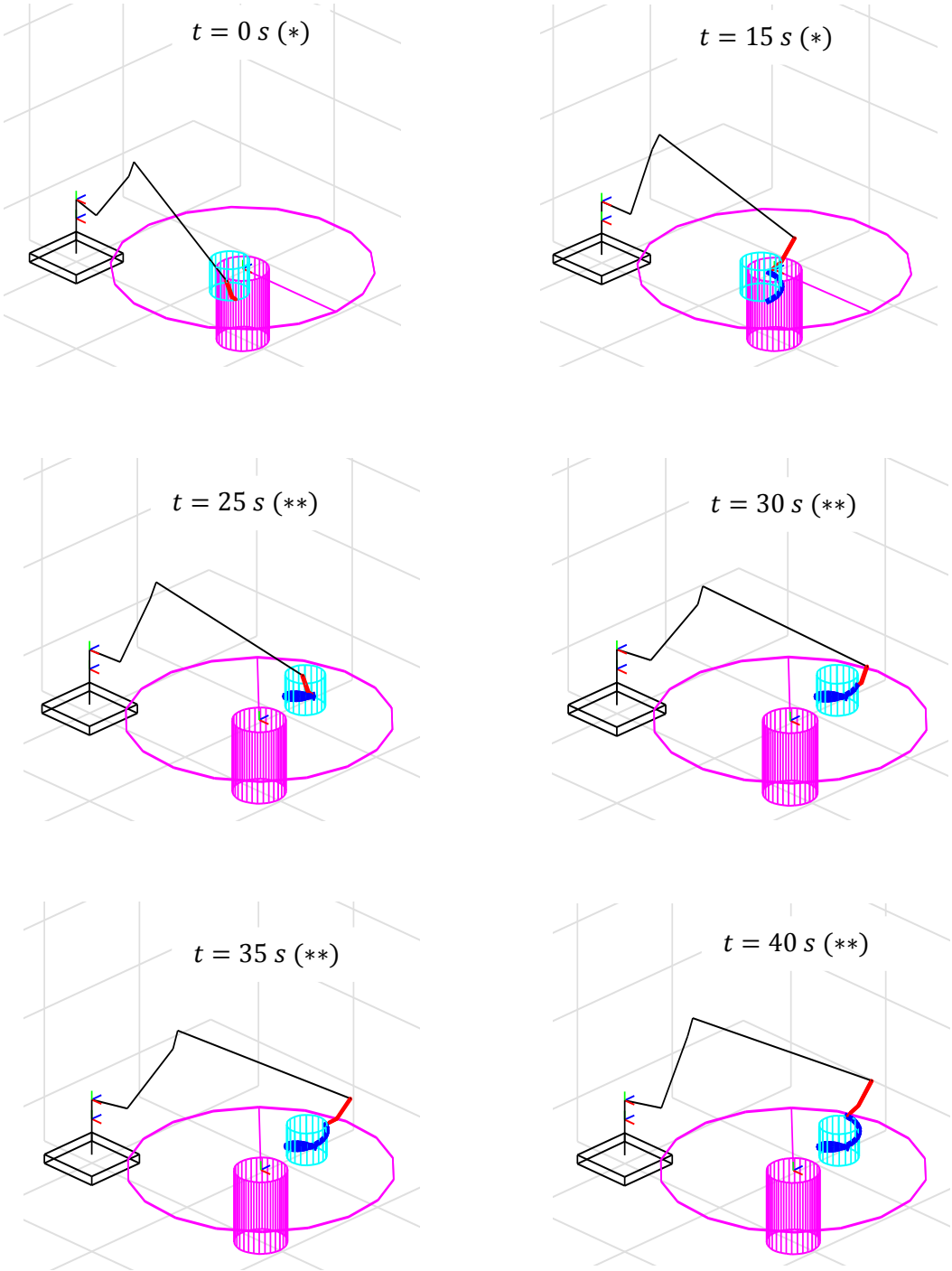


Figura 5.7. Caso 3. Secuencia de posturas del robot y de la mesa al ejecutar la tarea. (*) primer cordón, (**) segundo cordón.

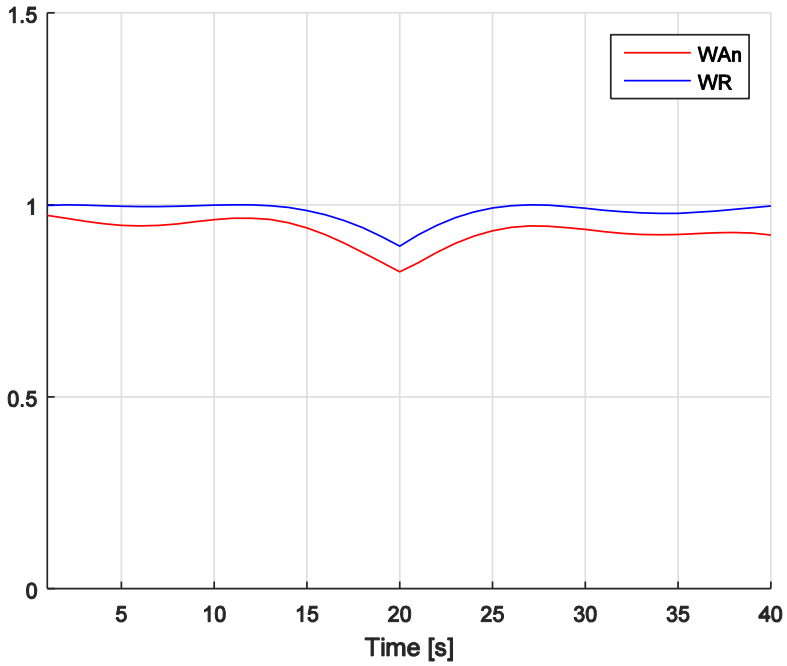


Figura 5.8. Caso 3. Manipulabilidades rotacional y traslacional normalizada.

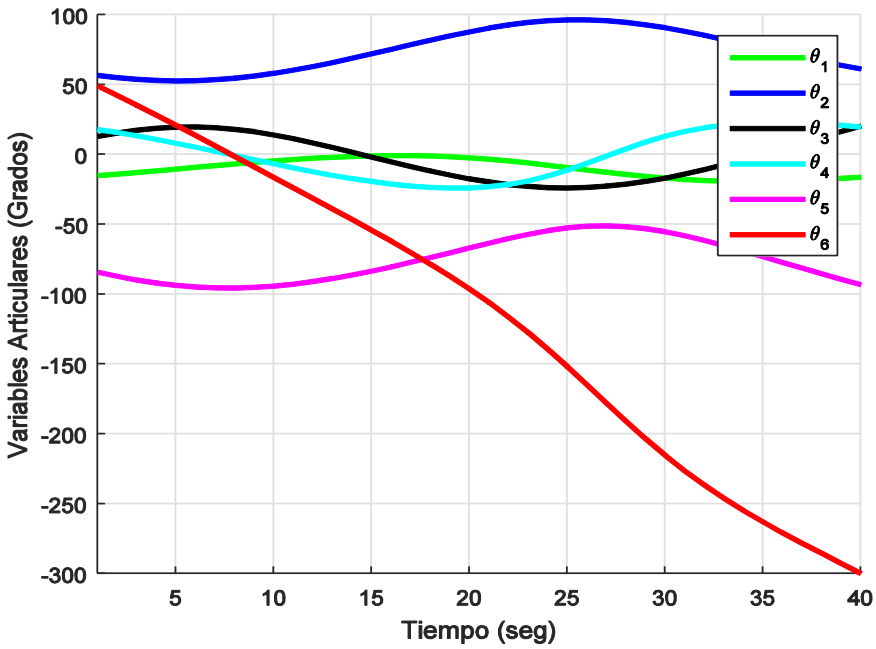


Figura 5.9. Caso 3. Historia de las variables articulares.

Al comparar los resultados obtenidos en el caso 2 observamos que el rendimiento cinemático del robot se mejoró en comparación con el del caso 1, esto se puede apreciar en la figura 5.5 donde se muestran las manipulabilidades rotacional y traslacional correspondientes a este caso. En particular se puede observar que la manipulabilidad traslacional (W_{An}) normalizada mejoró notablemente.

En la figura 5.7 se presenta la simulación de la tarea realizada en el caso 3, donde se observa la transición de la antorcha a través de la trayectoria helicoidal alrededor de la pieza de trabajo realizando los dos cordones de soldadura a diferencia de los dos casos anteriores que utilizan un solo cordón. También se aprecian las distintas posturas del robot para llevar a cabo la tarea de soldadura y las diferentes orientaciones de la mesa cooperante de acuerdo a los ángulos $\psi_t = 120\,°$ y $\psi_{t2} = -103.5\,°$. En la figura 5.8 se puede constatar la mejora tanto en la manipulabilidad traslacional como en la rotacional. En comparación con los dos casos anteriores, este caso es el que tiene proporciona los mejores resultados.

5.5 Análisis de resultados

Los resultados obtenidos después de realizar la optimización y las simulaciones correspondientes a cada caso se muestran en la Tabla 5.1.

Tabla 5.1 Resultados de las variables optimizadas y la función objetivo.

Caso	$r_e\ (cm)$	ψ_t (°)	ψ_p (°)	ψ_{t2} (°)	f
1	0	0	0	-	-0.6764
2	31.2785	4.6372	22.1812	-	-0.8585
3	-40	120	-54	-103.5	-0.8955

De acuerdo a los resultados obtenidos se logró minimizar la función objetivo hasta un valor muy cercano a -1, lo que representa una muy buena optimización tomando en cuenta que se incrementó en el rendimiento del robot en un 22% con respecto al valor inicial sin optimización.

Los estudios de simulación muestran la eficacia del método propuesto.

6. Conclusión

En este trabajo se propuso un método para la planificación de movimientos de un robot de soldadura para tareas ejecutadas con una mesa rotacional cooperante. El método se basa en la colocación óptima de la pieza de trabajo sujeta a la mesa. La colocación se define por la ubicación de la pieza en la mesa y la posición angular de la mesa. Los valores de las variables que definen la ubicación de la pieza de trabajo se encuentran de tal manera que se maximizan los índices de manipulabilidades traslacional y rotacionales del robot. El método propuesto y la aplicación asociada en Matlab© brindan a los usuarios de las industrias medianas y pequeñas una herramienta interesante para la planificación del movimiento de robots de soldadura asistidos por mesas cooperantes.

Para los casos de estudio considerados, no se detectaron colisiones del robot con objetos del entorno. Sin embargo, para entornos con obstáculos, se debe modificar la formulación para incorporar restricciones que permitan evitar colisiones del robot con los objetos del entorno. En trabajos futuros, el método se mejorará para incorporar algoritmos para la prevención de colisiones. Del mismo modo, se considerarán en el método otros criterios de desempeño a optimizar (v.gr. el número de condición de la matriz jacobiana, o las relaciones de transmisión de fuerza y de velocidad), y se agregarán más grados de libertad a la mesa cooperante.

Referencias Bibliográficas

[1] International Federation of Robotics, “Executive Summary World Robotics 2016 Industrial Robots", http://www.worldrobotics.org , 2017.

[2] Engelberger JF, *Robotics in Practice. Management and Applications of Industrial Robots*; AMACOM, (1980).

[3] Wilson M, *Implementation of Robot Systems*, Elsevier, 2015.

[4] Alatartsev S, Stellmacher S and Ortmeier F, “Robotic Task Sequencing Problem: A Survey; Journal of Intelligent & Robotic Systems", Vol. 80, Issue 2, pp. 279-298, 2015.

[5] Alatartsev S., Stellmacher S. and Ortmeier F.; “Robotic Task Sequencing Problem: A Survey"; Journal of Intelligent & Robotic Systems", Vol. 80, Issue 2, pp. 279-298, November 2015.

[6] Lin Y, Zhao H and Ding H, “Posture optimization methodology of 6R industrial robots for machining using performance evaluation indices"; Robotics and Computer-Integrated Manufacturing, 48, pp 59-72 (2017).

[7] H. Chen, Z. Liu, Offline Programming for an Arc Welding Robot with Redundant DOF; Applied Mechanics and Materials, Vols. 184-185, pp. 1623-1627 (2012).

[8] Hemmerle JS and Prinz FB, “Optimal path placement for kinematically redundant manipulators"; Proceedings of the 1991 International Conference on Robotics and Automation, pp. 1234-1243, 1991.

[9] Chen H and Liu Z, “Offline Programming for an Arc Welding Robot with Redundant DOF"; Applied Mechanics and Materials, Vols. 184- 185, pp. 1623-1627, 2012.

[10] Pamanes JA, “Contibution à l’ étude de l’ accessibilité aux tâches et à la detérmination du placement optimal de robots manipulateurs”, tesis doctoral de la Université de Poitiers, Francia, julio 8, 1992.

[11] Scheinman Y, Roth B., “On the optimal selection and placement of manipulators”, proceeding of the fifth CISM-IFToMM Symposium on theory and practice of robots and manipulators, pp. 39-45, 1984.

[12] Salisbury J.K., Craig J.J., “Articulated hands: force control and kinematic issues”, The International Journal of Robotics Research”, Vol. 1, pp. 4-17, 1982.

[13] Yoshikawa T., “Manipulability of robotic mechanisms”, Robotic Research: The Second International Symposium, pp. 439-446, 1984.

[14] Chiu S.L., “Task compatibility of manipulator postures”, The International Journal of Robotic Research, Vol. 7, pp. 13-21, 1988.

[15] Nelson B, Pedersen K, Donath M, “Locating assembly tasks in a manipulator’s workspace”, proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 1987.

[16] Zeghloul S., Pamanes JA; “Multi-criteria optimal placement of robots in constrained environments; revista Robotica, Vol. 11, pp. 105-110, Cambridge, Inglaterra, 1993.

[17] Pamanes JA, Cuan E. and Zeghloul S.; “Single and Multi-Objective Optimization of Path Placement for robotic Manipulators”; Revista INGENIERÍA Investigacion y Tecnología, Vol. IX, No. 3, ISSN 1405-7743; pp. 231-257; México, 2008.

[18] Pámanes JA, Blásquez JR, Llama MA; “Análisis de la redundancia cinemática de robots industriales en tareas de soldadura"; Memorias del XII Congreso Internacional Anual de la SOMIM (Sociedad Mexicana de Ingeniería Mecánica), pp. 571-579, ISSN 2448-5551, Mérida, México, 2016

[19] Pámanes JA, Soto D, Llama MA, Reveles D; “Optimización del diseño de una estación robotizada experimental de soldadura”, Memorias del XVII Congreso Mexicano de Robótica (COMRob 2016) de la AMRob, pp. 179-188, Mazatlán, México, 2016.

[20] Kovács, A. (2016). Integrated task sequencing and path planning for robotic remote laser welding. International Journal of Production Research, 54(4), 1210-1224.

[21] Maimon, O., Braha, D., & Seth, V. (2000). A neural network approach for a robot task sequencing problem. Artificial intelligence in engineering, 14(2), 175-189.

[22] Khalil W, Kleinfinger J-F, “A new geometric notation for open and closed loop robots”, Proc. IEEE Conf. on Robotics and Automation, San Francisco, pp. 1174-1180, 1986.

[23] Pámanes J.A., *Robótica: Introducción al modelado de manipuladores*, Instituto Tecnológico de la Laguna, 2015.

[24] Paul R.P, Renaud M., Stevenson C., "A systematic approach for obtaining the kinematics of recursive manipulators based on homogeneous transformations", First International Symposium on Robotics Research, 1983.

[25] Dombre E., Khalil W, *Robot Manipulators. Modelling, Performance, Analysis and Control*, ISTE ltd, London, 2007.

[26] Yoshikawa T, "Translational and rotational manipulability of robotic manipulators", Proc. of the American Control Conference, pp. 228-233 (1990).

[27] https://www.kickstarter.com/projects/1128055363/7bot-a-powerfuldesktop-robot-arm-for-future-inven?lang=es

[28] http://arduino.cl/arduino-due/

[29] Byrd, R.H. and Gilbert J.C. and Nocedal J.: A Trust Region Method Based on Interior Point Techniques for Nonlinear Programming, Mathematical Programming, Vol 89, No. 1, pp. 149-185 (2000).

[30] Khalil W., Creusot D., "SYMORO+: a system for the symbolic modelling of robots", Robotica, Vol. 15, 1997, p. 153-161.

APÉNDICES

Apéndice A

SYMORO+: UN SISTEMA PARA EL MODELADO SIMBÓLICO DE ROBOTS [30]

Este paquete permite generar el modelo geométrico directo, el modelo geométrico inverso, el modelo cinemático directo, el modelo cinemático inverso, el modelo dinámico y los modelos de identificación de parámetros inerciales de robots de arquitectura arbitraria. La estructura de los robots puede ser en serie, estructura de árbol o contener lazos cerrados.

El paquete se ha desarrollado bajo MATHEMATICA y en lenguaje C. El SYMORO+ genera los modelos simbólicos de robots, necesarios en problemas de simulación, control, identificación y diseño. Este paquete es el resultado del trabajo de investigación del equipo de robótica del *Laboratoire d'Automatique de Nantes*, de Francia, en el campo del modelado, control e identificación de robots.

La especificación del robot se lleva a cabo a partir de los parámetros de Denavit-Hartenberg modificados del robot, indicando el número de enlaces móviles, el número de articulaciones y el tipo de estructura (serial, serial-arborescente o cerrado). Los parámetros son definidos numéricamente o simbólicamente.

El modelo geométrico directo del robot consiste en el conjunto de ecuaciones que definen la posición y orientación del órgano terminal en función de las variables de articulares. Se puede obtener multiplicando las matrices de transformación a lo largo de la cadena cinemática entre la base del robot y del efector final.

El modelo geométrico inverso se refiere al conjunto de ecuaciones que proporcionan todas las posturas del robot correspondientes a una ubicación determinada del órgano final.

En el SYMORO+ se usan tres métodos para su cálculo:

- El primer método proporciona la solución de un robot de seis grados de libertad siempre que se cumpla las siguientes condiciones:
  - El robot contiene tres articulaciones traslacionales.
  - El robot tiene tres articulaciones rotacionales que definen una articulación esférica (sus ejes se cruzan).
- El segundo método puede proporcionar la solución de la mayoría de los robots industriales actuales.
- En el tercer método la solución de una variable se da como una ecuación polinómica de grado 16 como máximo, luego se pueden obtener las otras variables.

Por otra parte, el SYMORO+ calcula la matriz jacobiana de un robot utilizando el método desarrollado por M. Renaud [30].

Apéndice B

Código de programas elaborados en Matlab©

B.1 Programa para la simulación de los movimientos del robot. Casos 1 y 2.

```
%Se limpia el area de trabajo
clc, clear, close all;

%Se declaran los parametros del robot con escala de 7.5
d2 = 30;
d3 = 90;
d4 = 22.5;
r4 = 148.875;

%Radio de el cilindro
rc = 15;

%numero de puntos por trayectoria
np = 20;

%Tiempo para el cordon de soldadura
T  = 18;

%Tiempo para las trayectorias de acercamiento y alejamiento
T3 = 5;

%Coordenadas del centro de la mesa.
xc = 177.5;
yc = 0;
zc = 32.065;

%Linea de Radio en la mesa
xr = 177.5;
yr = -100;

%% Variables a optimizar

%caso 1
% %Psi Ángulo para girar la Mesa
% psit = deg2rad(0);
% %Psit Ángulo para girar el cilindro
% psi    = deg2rad(0);
% %Distancia entre los centros de la mesa y el cilindro.
% re = 0;
```

```
% %caso 2
%Psi Ángulo para girar la Mesa
psi    = deg2rad(4.6372);

%Psit Ángulo para girar el cilindro
psit = deg2rad(22.1812);

%Distancia entre los centros de la mesa y el cilindro.
 re = 31.2785;


%% Ángulos de la antorcha:
phieini = deg2rad(45);

phiefin = deg2rad(225);

thetaeini = deg2rad(135);

psieini = deg2rad(0);

%%% Coordenadas de Home:
phieh = deg2rad(0);
thetaeh = deg2rad(-180);
psieh = deg2rad(180);

xih = 0;
yih = -25;
zih = 70;

%%
%Matriz de Emplazamiento e Inversa
dx=0;
dy=0;
dz=22.065;

Te0 = [1  0  0  dx;
       0  1  0  dy;
       0  0  1  dz;
       0  0  0  1];

 T0e = inv(Te0);

%Matriz del Marco de la Herramienta e Inversa
T6H = [0.9239  0  -0.3827  -5;
        0      1    0       0;
       0.3827  0  0.9239   40;
        0      0    0       1];

TH6=inv(T6H);

%Matriz del Cilindro a la Mesa
TMC =  [cos(psit)  -sin(psit)  0  re;
        sin(psit)  cos(psit)   0  0;
        0              0       1  0;
        0              0       0  1];

%Matriz de la Mesa con respecto a ala estación de Trabajo
TeM =  [cos(psi)  -sin(psi)  0   xc;
        sin(psi)   cos(psi)  0   yc;
           0          0      1   zc;
```

```
           0          0      0   1];

%Base del Robot
a = [-25;  25; -62.065; 1];
b = [ 25;  25; -62.065; 1];
c = [ 25; -25; -62.065; 1];
d = [-25; -25; -62.065; 1];

A = Te0 * a;
B = Te0 * b;
C = Te0 * c;
D = Te0 * d;

Bx = [A(1) B(1) C(1) D(1) A(1)];
By = [A(2) B(2) C(2) D(2) A(2)];
Bz = [A(3) B(3) C(3) D(3) A(3)];

Bz2 = [A(3)-10 B(3)-10 C(3)-10 D(3)-10 A(3)-10];

%Cilindro
[x, y, z] = cylinder(15,20);

%Variables para girar y posicionar el cilindro
TeC = TeM * TMC;

x = x + TeC(1,4);
y = y + TeC(2,4);

z = z + zc;
z(2,:) = z(2,:) + 35;
xt = x;
yt = y;


% Mesa y base de la mesa
[xm, ym, zm] = cylinder(100,18);
[xb, yb, zb] = cylinder(20,30);

%Variables para girar y posicionar la mesa
xm = xm + xc;
xb = xb + xc;
xtm = xm;
ytm = ym;

%% Ejes X,Y y Z %%

% Ejes del Marco de la Estacion de Trabajo:
p1x = [0  10];
p1y = [0  0 ];
p1z = [0  0 ];

p2x = [0  0 ];
p2y = [0  10];
p2z = [0  0 ];

p3x = [0  0 ];
p3y = [0  0 ];
p3z = [0  10];

%Ejes del Marco del robot:
h1xyz =  Te0  * [0  0  0  10;
```

```
                0  0  0  0;
                0  0  0  0;
                0  0  0  1];

h2xyz =  Te0  * [0  0  0  0;
                 0  0  0  10;
                 0  0  0  0;
                 0  0  0  1];

h3xyz =  Te0  * [0  0  0  0;
                 0  0  0  0;
                 0  0  0  10;
                 0  0  0  1];

%% Calculo de Thetas para las trayectorias angulares

%%%%Primer Trayectoria Angular de home a primer punto de acercamiento
%Parametros de posición y orientación de la Herramienta
phie   = phieh;
thetae = thetaeh;
psie   = psieh;

xi = xih;
yi = yih;
zi = zih;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1h1,t2h1,t3h1,t4h1,t5h1,t6h1]=mcip(Tsnap);

%Parametros de posición y orientación de la Herramienta
phie   = phieini;
thetae = thetaeini;
psie   = psieini;

xi = 0;
yi = -rc - 7.07;
zi = 7.07;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1a1,t2a1,t3a1,t4a1,t5a1,t6a1]=mcip(Tsnap);

%%%%Segunda Trayectoria Angular del punto de escape a al punto intermedio
% Parametros de posición y orientación de la Herramienta
phie   = phieini;
thetae = thetaeini;
psie   = psieini;

xi =  0;
yi =  -rc - 7.07;
zi =  43.07;
```

```
% Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

% Se calcula la cinematica Inversa
[t1h2,t2h2,t3h2,t4h2,t5h2,t6h2]=mcip(Tsnap);

% Parametros de posición y orientación de la Herramienta
phie   = phieh;
thetae = thetaeh;
psie   = psieh;

xi = xih;
yi = yih;
zi = zih;

% Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

% Se calcula la cinematica Inversa
[t1a2,t2a2,t3a2,t4a2,t5a2,t6a2]=mcip(Tsnap);

%% Declaracion de figura
figure(1)

%% Ciclos for
for ii=0:1   %Recorre el numero de trayectorias
for i=0:np;   %Recorre el tiempo

%Se definen variables
t      = T*i/np;  %tiempo de 18 seg.
tar    = T3*i/np; %tiempo de 5 seg.

fcart  = (tar/T3-(0.5/pi)*sin(2*pi*tar/T3));   %funcion cicloidal de 5 seg.

phi    = deg2rad(-90)+(deg2rad(10)*t);         %phi para Ángulo helicoidal

zzp = 1; %Velocidad en z de la antorcha

%%Trayectorias %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

phie   = phieini;
thetae = thetaeini;
psie   = psieini;

%P1,1 a P1,2
if ii==0
phie = phieini + deg2rad(10*t);

xi = rc*cos(phi);
yi = rc*sin(phi);
zi = zzp*t;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;
```

```
%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

%P1,1 a P1,2
if ii==1

phie = phiefin + deg2rad(10*t);

xi = -rc*cos(phi);
yi = -rc*sin(phi);
zi = 18+zzp*t;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);

if t6 > 0
    t6 = t6 - deg2rad(360);
end

end

phie = phieini;

%P1,2 a P1,e
if ii==4

xi = 0;
yi = -rc - 7.07 * fcart;
zi = 36 + 7.07 * fcart;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

% % %P1,e a P12i
if ii==5

dt1 = t1a2 - t1h2;
t1 = t1h2 + dt1 * fcart;

dt2 = t2a2 - t2h2;
t2 = t2h2 + dt2 * fcart;

dt3 = t3a2 - t3h2;
t3 = t3h2 + dt3 * fcart;

dt4 = t4a2 - t4h2;
t4 = t4h2 + dt4 * fcart;

dt5 = t5a2 - t5h2;
```

```
t5 = t5h2 + dt5 * fcart;

dt6 = t6a2 - t6h2;
t6 = t6h2 + dt6 * fcart;

end

%% Matrices Modelo directo
%Base a la primera articulacion
TBT0 = [1  0  0     0;
        0  1  0     0;
        0  0  0  -62.5;
        0  0  0     1];

T0T01 = Te0 * TBT0;

T0T014 = T0T01(1,4);
T0T024 = T0T01(2,4);
T0T034 = T0T01(3,4);

%Matriz de 0 a 1
T0T11 = [cos(t1)  -sin(t1)  0  0;
         sin(t1)   cos(t1)  0  0;
            0         0     1  0;
            0         0     0  1];

T0T011 = Te0 * T0T11;

T0T114 = T0T011(1,4);
T0T124 = T0T011(2,4);
T0T134 = T0T011(3,4);

%Matriz de 0 a 2
T12 =  [cos(t2)  -sin(t2)  0   d2;
           0         0    -1    0;
        sin(t2)  cos(t2)   0    0;
             0       0     0   1];

T0T21 = T0T11 * T12;

T0T211 = Te0 * T0T21;

T0T214 = T0T211(1,4);
T0T224 = T0T211(2,4);
T0T234 = T0T211(3,4);

%Matriz de 0 a 3
T23=[cos(t3) -sin(t3) 0  d3;
     sin(t3)  cos(t3) 0   0;
        0        0    1   0;
        0        0    0  1];

T0T3 = T0T21 * T23;

T0T31 = Te0 * T0T3;

T0T314 = T0T31(1,4);
T0T324 = T0T31(2,4);
T0T334 = T0T31(3,4);

%Vector de 3 a codo
```

```
M_C=  [22.5   0   0   1]';

%Matriz de 0 a codo
M_0C = Te0 * T0T3 * M_C;

%Matriz de 0 a 4
T34=[cos(t4)  -sin(t4)  0   d4;
       0          0    -1  -r4;
     sin(t4)  cos(t4)   0    0;
        0        0      0   1];

T0T41 = T0T3 * T34;

T0T411 = Te0 * T0T41;

T0T414 = T0T411(1,4);
T0T424 = T0T411(2,4);
T0T434 = T0T411(3,4);

%Matriz de 0 a 5
T45=[cos(t5)  -sin(t5)  0  0;
        0        0      1  0;
    -sin(t5)  -cos(t5)  0  0;
        0        0      0  1];

T0T51 = T0T41 * T45;

T0T511 = Te0 * T0T51;

% %Matriz de 0 a 6
T56 = [cos(t6)  -sin(t6)  0  0;
          0        0     -1  0;
       sin(t6)  cos(t6)   0  0;
          0        0      0  1];

T0T56 = T0T51 * T56;

T0T6 = Te0 * T0T56;

T0TH = Te0*T0T11*T12*T23*T34*T45*T56*T6H;

%Coordenadas para unir los puntos del diagrama de alambre
A1x = [T0T014 T0T114];
A1y = [T0T024 T0T124];
A1z = [T0T034 T0T134];

A2x = [T0T114 T0T214];
A2y = [T0T124 T0T224];
A2z = [T0T134 T0T234];

A3x = [T0T214 T0T314];
A3y = [T0T224 T0T324];
A3z = [T0T234 T0T334];

Acx = [T0T314 M_0C(1)];
Acy = [T0T324 M_0C(2)];
Acz = [T0T334 M_0C(3)];

A4x = [M_0C(1) T0T414];
A4y = [M_0C(2) T0T424];
A4z = [M_0C(3) T0T434];
```

```
Ahx = [T0T414 T0T6(1,4)];
Ahy = [T0T424 T0T6(2,4)];
Ahz = [T0T434 T0T6(3,4)];

%% % Parametros para la antorcha
alp = deg2rad(22.5);
d   = 2;
l1  = 2;
l2  = 2;
l3  = 9;
l4  = 28;

salp = sin(alp);
calp = cos(alp);

%Vertices para dibujar Hexágono
H_B = [d/4  -d/4  -d/2  -d/4   d/4  d/2  d/4;
       d/2   d/2   0    -d/2  -d/2   0   d/2;
        0     0    0     0      0    0     0;
        1     1    1     1      1    1    1];

%Origen
H0 = [0;0;0;1];

% Matrices para rotar y trasladar los elementos de la antorcha %%

T_0_1 = T0TH *  [1  0  0   0;
                 0  1  0   0;
                 0  0  1  -l1;
                 0  0  0   1];

 T_0_2 = T0TH *  [1  0  0     0;
                  0  1  0     0;
                  0  0  1  -(l1+l2);
                  0  0  0     1];


T_0_3 = T0TH *  [1  0  0      0;
                 0  1  0      0;
                 0  0  1  -(l1+l2+l3);
                 0  0  0      1];


T_0_4 = T0TH * [calp   0   salp      -d/2+d/2*calp;
                 0      1    0            0;
                -salp   0  calp  -(l1+l2+l3+d/2*salp);
                 0      0    0            1];



T_0_5 = T0TH * [calp   0   salp     d/2-d/2*calp-l4*salp;
                 0      1    0                0;
                 -salp  0  calp  -(l1+l2+l3+d/2*salp+l4*calp);
                 0      0     0                1];

% Operaciones para trasladar los puntos y los Hexágonos
H1 = T_0_1 * H0;
H2 = T_0_2 * H_B;
H3 = T_0_3 * H_B;
H4 = T_0_4 * H_B;
```

```
H5 = T_0_5 * H_B;
H0 = T0TH * H0;

% Operaciones para unir los puntos con lineas
HL1 = [H0,H1,H2(:,1),H3(:,1),H4(:,1),H5(:,1)];
HL2 = [H0,H1,H2(:,2),H3(:,2),H4(:,2),H5(:,2)];
HL3 = [H0,H1,H2(:,3),H3(:,3),H4(:,3),H5(:,3)];
HL4 = [H0,H1,H2(:,4),H3(:,4),H4(:,4),H5(:,4)];
HL5 = [H0,H1,H2(:,5),H3(:,5),H4(:,5),H5(:,5)];
HL6 = [H0,H1,H2(:,6),H3(:,6),H4(:,6),H5(:,6)];



%% Se guardan thetas para Graficas

if ii>=0 && ii<=1
    qp1(i+1+np*ii)=rad2deg(t1);
    qp2(i+1+np*ii)=rad2deg(t2);
    qp3(i+1+np*ii)=rad2deg(t3);
    qp4(i+1+np*ii)=rad2deg(t4);
    qp5(i+1+np*ii)=rad2deg(t5);
    qp6(i+1+np*ii)=rad2deg(t6);

end
% if ii==3
%     qqp1(i+1+np*ii)=rad2deg(t1);
%     qqp2(i+1+np*ii)=rad2deg(t2);
%     qqp3(i+1+np*ii)=rad2deg(t3);
%     qqp4(i+1+np*ii)=rad2deg(t4);
%     qqp5(i+1+np*ii)=rad2deg(t5);
%     qqp6(i+1+np*ii)=rad2deg(t6);
% end
%% Graficacion

%Se limpia la figura y se indica que se quede dibujada
clf
hold on

%Dibuja el alambre
plot3(A1x,A1y,A1z,'k');
plot3(A2x,A2y,A2z,'k');
plot3(A3x,A3y,A3z,'k');
plot3(Acx,Acy,Acz,'k');
plot3(A4x,A4y,A4z,'k');
plot3(Ahx,Ahy,Ahz,'k');

%Dibuja la base
plot3(Bx,By,Bz,'k');
plot3(Bx,By,Bz2,'k');
plot3([A(1) A(1)],[A(2) A(2)],[A(3) A(3)-10],'k');
plot3([B(1) B(1)],[B(2) B(2)],[B(3) B(3)-10],'k');
plot3([C(1) C(1)],[C(2) C(2)],[C(3) C(3)-10],'k');
plot3([D(1) D(1)],[D(2) D(2)],[D(3) D(3)-10],'k');

%% Antorcha
%Hexagonos y puntos
plot3(H1(1,:),H1(2,:),H1(3,:),'r');
plot3(H2(1,:),H2(2,:),H2(3,:),'r');
plot3(H3(1,:),H3(2,:),H3(3,:),'r');
plot3(H4(1,:),H4(2,:),H4(3,:),'r');
plot3(H5(1,:),H5(2,:),H5(3,:),'r');
```

```
%Lineas
plot3(HL1(1,:),HL1(2,:),HL1(3,:),'r');
plot3(HL2(1,:),HL2(2,:),HL2(3,:),'r');
plot3(HL3(1,:),HL3(2,:),HL3(3,:),'r');
plot3(HL4(1,:),HL4(2,:),HL4(3,:),'r');
plot3(HL5(1,:),HL5(2,:),HL5(3,:),'r');
plot3(HL6(1,:),HL6(2,:),HL6(3,:),'r');

%Cilindros
% Dibuja cilindro
plot3(x, y, z, 'c');
plot3(x(1,:), y(1,:), z(1,:), 'c');
plot3(x(2,:), y(2,:), z(2,:), 'c');

% Dibuja mesa
plot3(xm, ym, zm+31.065,'m');
plot3(xm(1,:), ym(1,:), zm(1,:)+31.065,'m');
plot3(xm(2,:), ym(2,:), zm(2,:)+31.065,'m');

% Dibuja base de la mesa
plot3(xb, yb, (zb*-81.065)+31.065,'m');
plot3(xb(1,:), yb(1,:), zb(1,:)+31.065,'m');
plot3(xb(2,:), yb(2,:), (zb(2,:)*-71.065)+21.065,'m');

% Radio
plot3([177.5 xr],[0 yr],[31.065 31.065],'m');

%% Trayectorias
% if i>=1 && ii<2
% Vx(i+np*ii) = H0(1);
% Vy(i+np*ii)  = H0(2);
% Vz(i+np*ii)  = H0(3);
% plot3(Vx,Vy,Vz,'c','LineWidth',2);
% end

if ii>=2 && ii<=3
Vx1(i-(2*np-1)+ np*ii) = H0(1);
Vy1(i-(2*np-1)+np*ii)  = H0(2);
Vz1(i-(2*np-1)+np*ii)  = H0(3);
% plot3(Vx,Vy,Vz,'c','LineWidth',2);
plot3(Vx1,Vy1,Vz1,'b','LineWidth',2);
end

if ii>3
% Vx2(i-(4*np-1)+ np*ii) = H0(1);
% Vy2(i-(4*np-1)+np*ii)  = H0(2);
% Vz2(i-(4*np-1)+np*ii)  = H0(3);
% plot3(Vx,Vy,Vz,'c','LineWidth',2);
plot3(Vx1,Vy1,Vz1,'b','LineWidth',2);
% plot3(Vx2,Vy2,Vz2,'c','LineWidth',2);
end

%% Ejes

% Graficación de los ejes de Matlab
plot3(p1x,p1y,p1z, 'r',p2x,p2y,p2z, 'b',p3x,p3y,p3z, 'g');

% Graficación de los ejes del robot:
plot3([T0T011(1,4) h1xyz(1,4)],[T0T011(2,4) h1xyz(2,4)],[T0T011(3,4)
h1xyz(3,4)],'r');
```

```
plot3([T0T011(1,4) h2xyz(1,4)],[T0T011(2,4) h2xyz(2,4)],[T0T011(3,4)
h2xyz(3,4)],'b');
plot3([T0T011(1,4) h3xyz(1,4)],[T0T011(2,4) h3xyz(2,4)],[T0T011(3,4)
h3xyz(3,4)],'g');

% Graficación de los ejes de la Mesa
plot3(p1x+177.5,p1y,p1z+31.065, 'r',p2x+177.5,p2y,p2z+31.065,
'b',p3x+177.5,p3y,p3z+31.065, 'g');

%% Propiedades de la figura
%Vista
% view([0,0,1])
view([1,-1,1])
%Área de dibujo
axis([-50 300 -175 175 -50 300]);

%Etiquetas
xlabel('x');
ylabel('y');
zlabel('z');

%Cuadricula
grid on;

%Pausa en la animacion
pause(.01)

end
end

figure('Name','Velocidades Angulares caso 1','NumberTitle','off')
    hold on
    grid on
    plot(qp1,'g','linewidth',2);
    plot(qp2,'b','linewidth',2);
    plot(qp3,'k','linewidth',2);
    plot(qp4,'c','linewidth',2);
    plot(qp5,'m','linewidth',2);
    plot(qp6,'r','linewidth',2);
    legend('\theta_1','\theta_2','\theta_3','\theta_4','\theta_5', ...
        '\theta_6','Location','northeast','Orientation','vertical')
        xlabel('Tiempo (seg)')
    ylabel('Variables Articulares (Grados)')
    axis([1,40,-250,150])
```

B.2 Programa para la simulación de los movimientos del robot. Caso 3.

```
%Se limpia el area de trabajo
clc, clear, close all;

%Se declaran los parametros del robot con escala de 7.5
d2 = 30;
d3 = 90;
d4 = 22.5;
r4 = 148.875;

%Radio de el cilindro
rc = 15;

%numero de puntos por trayectoria
np = 20;

%Tiempo para el cordon de soldadura
T  = 18;

%Tiempo para las trayectorias de acercamiento y alejamiento
T3 = 5;

%Coordenadas del centro de la mesa.
xc = 177.5;
yc = 0;
zc = 32.065;



 %% Variables a optimizar caso 3
% % Psi Ángulo para girar la Mesa
% psi    = deg2rad(0);
%
% % Psit Ángulo para girar el cilindro
% psit = deg2rad(0);
%
% % Distancia entre los centros de la mesa y el cilindro.
%  re = 0;
%
%
%  psi2  = deg2rad(180);


%% Variables a optimizar caso 4
%Psi Ángulo para girar la Mesa
```

```
psi    = deg2rad(120.7207);

%Psit Ángulo para girar el cilindro
psit = deg2rad(-54.0510);

%Distancia entre los centros de la mesa y el cilindro.
 re = -39.9995;

%%
psi2 = deg2rad(-103.5746);

%Linea de Radio en la mesa
xr = 100*cos(psi)+177.5;
yr = 100*sin(psi);

xrt = xr;
yrt = yr;

%% Ángulos de la antorcha:
phieini = deg2rad(45);

phieint = deg2rad(135);

phiefin = deg2rad(225);

thetaeini = deg2rad(135);

psieini = deg2rad(0);

%%% Coordenadas de Home:
phieh = deg2rad(0);
thetaeh = deg2rad(-180);
psieh = deg2rad(180);

xih = 0;
yih = -25;
zih = 70;

%%
%Matriz de Emplazamiento e Inversa
dx=0;
dy=0;
dz=22.065;

Te0 = [1  0  0  dx;
       0  1  0  dy;
       0  0  1  dz;
       0  0  0  1];

 T0e = inv(Te0);

%Matriz del Marco de la Herramienta e Inversa
```

```
T6H = [0.9239  0  -0.3827  -5;
        0      1    0       0;
       0.3827  0  0.9239   40;
        0      0    0       1];

TH6=inv(T6H);

%Matriz del Cilindro a la Mesa
TMC =  [cos(psit)  -sin(psit)  0  re;
        sin(psit)  cos(psit)   0  0;
        0              0       1  0;
        0              0       0  1];

%Matriz de la Mesa con respecto a ala estación de Trabajo
TeM =  [cos(psi)  -sin(psi)  0   xc;
        sin(psi)   cos(psi)  0   yc;
           0          0      1   zc;
           0          0      0   1];

%Base del Robot
a = [-25;  25; -62.065; 1];
b = [ 25;  25; -62.065; 1];
c = [ 25; -25; -62.065; 1];
d = [-25; -25; -62.065; 1];

A = Te0 * a;
B = Te0 * b;
C = Te0 * c;
D = Te0 * d;

Bx = [A(1) B(1) C(1) D(1) A(1)];
By = [A(2) B(2) C(2) D(2) A(2)];
Bz = [A(3) B(3) C(3) D(3) A(3)];

Bz2 = [A(3)-10 B(3)-10 C(3)-10 D(3)-10 A(3)-10];

%Cilindro
[x, y, z] = cylinder(15,20);

%Variables para girar y posicionar el cilindro
TeC = TeM * TMC;
xt2 = x;
yt2 = y;
x = x + TeC(1,4);
y = y + TeC(2,4);

z = z + zc;
z(2,:) = z(2,:) + 35;
xt = x;
yt = y;


% Mesa y base de la mesa
```

```
[xm, ym, zm] = cylinder(100,18);
[xb, yb, zb] = cylinder(20,30);

%Variables para girar y posicionar la mesa
xm = xm + xc;
xb = xb + xc;
xtm = xm;
ytm = ym;

%% Ejes X,Y y Z %%

% Ejes del Marco de la Estacion de Trabajo:
p1x = [0  10];
p1y = [0  0 ];
p1z = [0  0 ];

p2x = [0  0 ];
p2y = [0  10];
p2z = [0  0 ];

p3x = [0  0 ];
p3y = [0  0 ];
p3z = [0  10];

%Ejes del Marco del robot:
h1xyz =  Te0  * [0  0  0  10;
                 0  0  0  0;
                 0  0  0  0;
                 0  0  0  1];

h2xyz =  Te0  * [0  0  0  0;
                 0  0  0  10;
                 0  0  0  0;
                 0  0  0  1];

h3xyz =  Te0  * [0  0  0  0;
                 0  0  0  0;
                 0  0  0  10;
                 0  0  0  1];

%% Calculo de Thetas para las trayectorias angulares

%%%%Primer Trayectoria Angular de home a primer punto de acercamiento
%Parametros de posición y orientación de la Herramienta
phie   = phieh;
thetae = thetaeh;
psie   = psieh;

xi = xih;
yi = yih;
zi = zih;

%Se forma la Matriz TCA
```

```
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1h1,t2h1,t3h1,t4h1,t5h1,t6h1]=mcip(Tsnap);

%Parametros de posición y orientación de la Herramienta
phie   = phieini;
thetae = thetaeini;
psie   = psieini;

xi = 0;
yi = -rc - 7.07;
zi = 7.07;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1a1,t2a1,t3a1,t4a1,t5a1,t6a1]=mcip(Tsnap);

%%%%Segunda Trayectoria Angular del punto de escape a al punto intermedio
% Parametros de posición y orientación de la Herramienta
phie   = phiefin;
thetae = thetaeini;
psie   = psieini;

xi =  0;
yi =  rc + 7.07;
zi =  25.07;

% Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

% Se calcula la cinematica Inversa
[t1h2,t2h2,t3h2,t4h2,t5h2,t6h2]=mcip(Tsnap);

% Parametros de posición y orientación de la Herramienta
phie   = phieint;
thetae = thetaeini;
psie   = psieini;

xi = -40;
yi = 0;
zi = 50;

% Se forma la Matriz TCA
```

```
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

% Se calcula la cinematica Inversa
[t1a2,t2a2,t3a2,t4a2,t5a2,t6a2]=mcip(Tsnap);

%%%%Tercer Trayectoria Angular del punto intermedio a al punto de
%%%%acercamiento
% Parametros de posición y orientación de la Herramienta
%Matriz de la Mesa con respecto a ala estación de Trabajo


phie   = phieint;
thetae = thetaeini;
psie   = psieini;

xi = -40;
yi = 0;
zi = 50;

% Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

% Se calcula la cinematica Inversa
[t1h3,t2h3,t3h3,t4h3,t5h3,t6h3]=cinematicainver(Tsnap);

% Parametros de posición y orientación de la Herramienta

TeM =  [cos(psi2)  -sin(psi2)  0   xc;
        sin(psi2)   cos(psi2)  0   yc;
           0           0       1   zc;
           0           0       0   1];

phie   = phiefin;
thetae = thetaeini;
psie   = psieini;

xi =  0;
yi =  rc + 7.07;
zi =  25.07;

% Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

% Se calcula la cinematica Inversa
[t1a3,t2a3,t3a3,t4a3,t5a3,t6a3]=cinematicainver(Tsnap);
```

```
%%%%Cuarta Trayectoria Angular del punto de escape a Home
% Parametros de posición y orientación de la Herramienta
phie   = phieini;
thetae = thetaeini;
psie   = psieini;

xi =  0;
yi =  -rc - 7.07;
zi =  43.07;

% Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

% Se calcula la cinematica Inversa
[t1h4,t2h4,t3h4,t4h4,t5h4,t6h4]=mcip(Tsnap);

% Parametros de posición y orientación de la Herramienta
phie   = phieh;
thetae = thetaeh;
psie   = psieh;

xi = xih;
yi = yih;
zi = zih;

% Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

% Se calcula la cinematica Inversa
[t1a4,t2a4,t3a4,t4a4,t5a4,t6a4]=mcip(Tsnap);


%Matriz de la Mesa con respecto a ala estación de Trabajo
TeM =  [cos(psi)  -sin(psi)  0   xc;
        sin(psi)   cos(psi)  0   yc;
           0           0     1   zc;
           0           0     0   1];

%% Declaracion de figura
figure(1)

%% Ciclos for
for ii=0:1;   %Recorre el numero de trayectorias
for i=0:np;   %Recorre el tiempo

%Se definen variables
t      = T*i/np;  %tiempo de 18 seg.
tar    = T3*i/np; %tiempo de 5 seg.
```

```
fcart  = (tar/T3-(0.5/pi)*sin(2*pi*tar/T3));   %funcion cicloidal de 5
seg.

phi    = deg2rad(-90)+(deg2rad(10)*t);         %phi para Ángulo
helicoidal

zzp = 1; %Velocidad en z de la antorcha

%%Trayectorias %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%
phie   = phieini;
thetae = thetaeini;
psie   = psieini;
%
%P1,1 a P1,2
if ii==0
phie = phieini + deg2rad(10*t);

xi = rc*cos(phi);
yi = rc*sin(phi);
zi = zzp*t;


%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end
phie = phiefin;

%P1,2 a P1,e
if ii==3

xi = 0;
yi = rc + 7.07 * fcart;
zi = 18 + 7.07 * fcart;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

% % %P1,e a P12i
```

```
if ii==4

dt1 = t1a2 - t1h2;
t1 = t1h2 + dt1 * fcart;

dt2 = t2a2 - t2h2;
t2 = t2h2 + dt2 * fcart;

dt3 = t3a2 - t3h2;
t3 = t3h2 + dt3 * fcart;

dt4 = t4a2 - t4h2;
t4 = t4h2 + dt4 * fcart;

dt5 = t5a2 - t5h2;
t5 = t5h2 + dt5 * fcart;

dt6 = t6a2 - t6h2;
t6 = t6h2 + dt6 * fcart;

end


% %P12,i a P2,a
if ii==5

%Matriz de la Mesa con respecto a ala estación de Trabajo
TeM =  [cos(psi2)  -sin(psi2)  0   xc;
        sin(psi2)   cos(psi2)  0   yc;
           0           0       1   zc;
           0           0       0   1];

%Se realiza el giro de Psi2.
psim    = (0 + psi2 *t/18)- (psi *t/18);

%Se gira el Radio de la mesa
% xr = 100*cos(psi2)+177.5;
% yr = 100*sin(psi2);
xr = (xrt-xc) * cos(psim) + xc - yrt * sin(psim);
yr = (xrt-xc) * sin(psim) + yrt * cos(psim);


%Se gira el cilindro
x = (xt-xc) * cos(psim) + xc - yt * sin(psim);
y = (xt-xc) * sin(psim) + yt * cos(psim);

%Se gira la mesa
xm = (xtm-xc) * cos(psim) + xc - ytm * sin(psim);
ym = (xtm-xc) * sin(psim) + ytm * cos(psim);

%Se ejecuta la trayectoria angular
dt1 = t1a3 - t1h3;
```

```
t1 = t1h3 + dt1 * fcart;

dt2 = t2a3 - t2h3;
t2 = t2h3 + dt2 * fcart;

dt3 = t3a3 - t3h3;
t3 = t3h3 + dt3 * fcart;

dt4 = t4a3 - t4h3;
t4 = t4h3 + dt4 * fcart;

dt5 = t5a3 - t5h3;
t5 = t5h3 + dt5 * fcart;

dt6 = t6a3 - t6h3;
t6 = t6h3 + dt6 * fcart;
end

phie   = phiefin;
thetae = thetaeini;
psie   = psieini;

%P1,a a P1,1
if ii==6
xi = 0;
yi = rc + 7.07 - 7.07 * fcart;
zi = 25.07 - 7.07 * fcart;

% Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

% P1,1 a P1,2
if ii==1

phie = phiefin + deg2rad(10*t);

xi = -rc*cos(phi);
yi = -rc*sin(phi);
zi = 18+zzp*t;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
```

```
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
if t6 > 0
    t6 = t6 - deg2rad(360);
end
end

phie   = phieini;

%P1,2 a P1,e
if ii==8

xi = 0;
yi = -rc - 7.07 * fcart;
zi = 36 + 7.07 * fcart;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

%P2,e a P23i
if ii==9

dt1 = t1a4 - t1h4;
t1 = t1h4 + dt1 * fcart;

dt2 = t2a4 - t2h4;
t2 = t2h4 + dt2 * fcart;

dt3 = t3a4 - t3h4;
t3 = t3h4 + dt3 * fcart;

dt4 = t4a4 - t4h4;
t4 = t4h4 + dt4 * fcart;

dt5 = t5a4 - t5h4;
t5 = t5h4 + dt5 * fcart;

dt6 = t6a4 - t6h4;
t6 = t6h4 + dt6 * fcart;

end

%% Matrices Modelo directo
%Base a la primera articulacion
TBT0 = [1   0   0     0;
        0   1   0     0;
        0   0   0  -62.5;
```

```
        0  0  0     1];

T0T01 = Te0 * TBT0;

T0T014 = T0T01(1,4);
T0T024 = T0T01(2,4);
T0T034 = T0T01(3,4);

%Matriz de 0 a 1
T0T11 = [cos(t1)  -sin(t1)  0  0;
         sin(t1)   cos(t1)  0  0;
            0         0     1  0;
            0         0     0  1];

T0T011 = Te0 * T0T11;

T0T114 = T0T011(1,4);
T0T124 = T0T011(2,4);
T0T134 = T0T011(3,4);

%Matriz de 0 a 2
T12 =  [cos(t2)  -sin(t2)  0   d2;
           0         0    -1    0;
        sin(t2)  cos(t2)   0    0;
             0       0     0   1];

T0T21 = T0T11 * T12;

T0T211 = Te0 * T0T21;

T0T214 = T0T211(1,4);
T0T224 = T0T211(2,4);
T0T234 = T0T211(3,4);

%Matriz de 0 a 3
T23=[cos(t3) -sin(t3) 0  d3;
     sin(t3)  cos(t3) 0   0;
        0        0    1   0;
        0        0    0  1];

T0T3 = T0T21 * T23;

T0T31 = Te0 * T0T3;

T0T314 = T0T31(1,4);
T0T324 = T0T31(2,4);
T0T334 = T0T31(3,4);

%Vector de 3 a codo
M_C=  [22.5   0   0   1]';

%Matriz de 0 a codo
```

```
M_0C = Te0 * T0T3 * M_C;

%Matriz de 0 a 4
T34=[cos(t4)  -sin(t4)  0   d4;
       0          0    -1  -r4;
     sin(t4)  cos(t4)   0    0;
        0        0      0   1];

T0T41 = T0T3 * T34;

T0T411 = Te0 * T0T41;

T0T414 = T0T411(1,4);
T0T424 = T0T411(2,4);
T0T434 = T0T411(3,4);

%Matriz de 0 a 5
T45=[cos(t5)  -sin(t5)  0  0;
        0        0      1  0;
    -sin(t5)  -cos(t5)  0  0;
        0        0      0  1];

T0T51 = T0T41 * T45;

T0T511 = Te0 * T0T51;

% %Matriz de 0 a 6
T56 = [cos(t6)  -sin(t6)  0  0;
          0        0     -1  0;
       sin(t6)  cos(t6)   0  0;
          0        0      0  1];

T0T56 = T0T51 * T56;

T0T6 = Te0 * T0T56;

T0TH = Te0*T0T11*T12*T23*T34*T45*T56*T6H;

%Coordenadas para unir los puntos del diagrama de alambre
A1x = [T0T014 T0T114];
A1y = [T0T024 T0T124];
A1z = [T0T034 T0T134];

A2x = [T0T114 T0T214];
A2y = [T0T124 T0T224];
A2z = [T0T134 T0T234];

A3x = [T0T214 T0T314];
A3y = [T0T224 T0T324];
A3z = [T0T234 T0T334];

Acx = [T0T314 M_0C(1)];
```

```
Acy = [T0T324 M_0C(2)];
Acz = [T0T334 M_0C(3)];

A4x = [M_0C(1) T0T414];
A4y = [M_0C(2) T0T424];
A4z = [M_0C(3) T0T434];

Ahx = [T0T414 T0T6(1,4)];
Ahy = [T0T424 T0T6(2,4)];
Ahz = [T0T434 T0T6(3,4)];

%% % Parametros para la antorcha
alp = deg2rad(22.5);
d   = 2;
l1  = 2;
l2  = 2;
l3  = 9;
l4  = 28;

salp = sin(alp);
calp = cos(alp);

%Vertices para dibujar Hexágono
H_B = [d/4  -d/4  -d/2  -d/4   d/4  d/2  d/4;
       d/2   d/2   0    -d/2  -d/2   0   d/2;
        0     0    0     0      0    0     0;
        1     1    1     1      1    1    1];

%Origen
H0 = [0;0;0;1];

% Matrices para rotar y trasladar los elementos de la antorcha %%

T_0_1 = T0TH *  [1  0  0   0;
                 0  1  0   0;
                 0  0  1  -l1;
                 0  0  0   1];

 T_0_2 = T0TH *  [1  0  0     0;
                  0  1  0     0;
                  0  0  1  -(l1+l2);
                  0  0  0     1];


T_0_3 = T0TH *  [1  0  0      0;
                 0  1  0      0;
                 0  0  1  -(l1+l2+l3);
                 0  0  0      1];


T_0_4 = T0TH * [calp   0   salp      -d/2+d/2*calp;
                 0      1    0            0;
```

```
                -salp   0  calp  -(l1+l2+l3+d/2*salp);
                 0      0    0            1];



T_0_5 = T0TH * [calp   0   salp     d/2-d/2*calp-l4*salp;
                 0      1    0                0;
                 -salp  0  calp  -(l1+l2+l3+d/2*salp+l4*calp);
                 0      0     0                1];

% Operaciones para trasladar los puntos y los Hexágonos
H1 = T_0_1 * H0;
H2 = T_0_2 * H_B;
H3 = T_0_3 * H_B;
H4 = T_0_4 * H_B;
H5 = T_0_5 * H_B;
H0 = T0TH * H0;

% Operaciones para unir los puntos con lineas
HL1 = [H0,H1,H2(:,1),H3(:,1),H4(:,1),H5(:,1)];
HL2 = [H0,H1,H2(:,2),H3(:,2),H4(:,2),H5(:,2)];
HL3 = [H0,H1,H2(:,3),H3(:,3),H4(:,3),H5(:,3)];
HL4 = [H0,H1,H2(:,4),H3(:,4),H4(:,4),H5(:,4)];
HL5 = [H0,H1,H2(:,5),H3(:,5),H4(:,5),H5(:,5)];
HL6 = [H0,H1,H2(:,6),H3(:,6),H4(:,6),H5(:,6)];



%% Se guardan thetas para Graficas
if ii<3
    qp1(i+1+np*ii)=rad2deg(t1);
    qp2(i+1+np*ii)=rad2deg(t2);
    qp3(i+1+np*ii)=rad2deg(t3);
    qp4(i+1+np*ii)=rad2deg(t4);
    qp5(i+1+np*ii)=rad2deg(t5);
    qp6(i+1+np*ii)=rad2deg(t6);

end
if ii==7
    qqp1(i+1+np*ii)=rad2deg(t1);
    qqp2(i+1+np*ii)=rad2deg(t2);
    qqp3(i+1+np*ii)=rad2deg(t3);
    qqp4(i+1+np*ii)=rad2deg(t4);
    qqp5(i+1+np*ii)=rad2deg(t5);
    qqp6(i+1+np*ii)=rad2deg(t6);
end

%% Graficacion

%Se limpia la figura y se indica que se quede dibujada
clf
hold on
```

```
%Dibuja el alambre
plot3(A1x,A1y,A1z,'k');
plot3(A2x,A2y,A2z,'k');
plot3(A3x,A3y,A3z,'k');
plot3(Acx,Acy,Acz,'k');
plot3(A4x,A4y,A4z,'k');
plot3(Ahx,Ahy,Ahz,'k');

%Dibuja la base
plot3(Bx,By,Bz,'k');
plot3(Bx,By,Bz2,'k');
plot3([A(1) A(1)],[A(2) A(2)],[A(3) A(3)-10],'k');
plot3([B(1) B(1)],[B(2) B(2)],[B(3) B(3)-10],'k');
plot3([C(1) C(1)],[C(2) C(2)],[C(3) C(3)-10],'k');
plot3([D(1) D(1)],[D(2) D(2)],[D(3) D(3)-10],'k');

%% Antorcha
%Hexagonos y puntos
plot3(H1(1,:),H1(2,:),H1(3,:),'r');
plot3(H2(1,:),H2(2,:),H2(3,:),'r');
plot3(H3(1,:),H3(2,:),H3(3,:),'r');
plot3(H4(1,:),H4(2,:),H4(3,:),'r');
plot3(H5(1,:),H5(2,:),H5(3,:),'r');

%Lineas
plot3(HL1(1,:),HL1(2,:),HL1(3,:),'r');
plot3(HL2(1,:),HL2(2,:),HL2(3,:),'r');
plot3(HL3(1,:),HL3(2,:),HL3(3,:),'r');
plot3(HL4(1,:),HL4(2,:),HL4(3,:),'r');
plot3(HL5(1,:),HL5(2,:),HL5(3,:),'r');
plot3(HL6(1,:),HL6(2,:),HL6(3,:),'r');

%Cilindros
% Dibuja cilindro
plot3(x, y, z, 'c');
plot3(x(1,:), y(1,:), z(1,:), 'c');
plot3(x(2,:), y(2,:), z(2,:), 'c');

% Dibuja mesa
plot3(xm, ym, zm+31.065,'m');
plot3(xm(1,:), ym(1,:), zm(1,:)+31.065,'m');
plot3(xm(2,:), ym(2,:), zm(2,:)+31.065,'m');

% Dibuja base de la mesa
plot3(xb, yb, (zb*-81.065)+31.065,'m');
plot3(xb(1,:), yb(1,:), zb(1,:)+31.065,'m');
plot3(xb(2,:), yb(2,:), (zb(2,:)*-71.065)+21.065,'m');

% Radio
plot3([177.5 xr],[0 yr],[31.065 31.065],'m');

%% Trayectorias
if i>=1 && ii<2
```

```
Vx(i+np*ii) = H0(1);
Vy(i+np*ii)  = H0(2);
Vz(i+np*ii)  = H0(3);
% plot3(Vx,Vy,Vz,'c','LineWidth',2);
end

if ii==2
Vx1(i-(2*np-1)+ np*ii) = H0(1);
Vy1(i-(2*np-1)+np*ii)  = H0(2);
Vz1(i-(2*np-1)+np*ii)  = H0(3);
% plot3(Vx,Vy,Vz,'c','LineWidth',2);
plot3(Vx1,Vy1,Vz1,'b','LineWidth',2);
end

if ii>2 && ii<5
Vx2(i-(3*np-1)+ np*ii) = H0(1);
Vy2(i-(3*np-1)+np*ii)  = H0(2);
Vz2(i-(3*np-1)+np*ii)  = H0(3);
% plot3(Vx,Vy,Vz,'c','LineWidth',2);
plot3(Vx1,Vy1,Vz1,'b','LineWidth',2);
% plot3(Vx2,Vy2,Vz2,'c','LineWidth',2);
end

if ii==5

Vxt  = (Vx-xc)  * cos(psim) + xc - Vy  * sin(psim);
Vyt  = (Vx-xc)  * sin(psim) + Vy  * cos(psim);
Vx1t = (Vx1-xc) * cos(psim) + xc - Vy1 * sin(psim);
Vy1t = (Vx1-xc) * sin(psim) + Vy1 * cos(psim);
Vx2t = (Vx2-xc) * cos(psim) + xc - Vy2 * sin(psim);
Vy2t = (Vx2-xc) * sin(psim) + Vy2 * cos(psim);


Vx3(i-(5*np-1)+ np*ii) = H0(1);
Vy3(i-(5*np-1)+np*ii)  = H0(2);
Vz3(i-(5*np-1)+np*ii)  = H0(3);

% plot3(Vxt,Vyt,Vz,'c','LineWidth',2);
plot3(Vx1t,Vy1t,Vz1,'b','LineWidth',2);
% plot3(Vx2t,Vy2t,Vz2,'c','LineWidth',2);
% plot3(Vx3,Vy3,Vz3,'c','LineWidth',2);
end

if ii==6

Vx4(i-(6*np-1)+ np*ii) = H0(1);
Vy4(i-(6*np-1)+np*ii)  = H0(2);
Vz4(i-(6*np-1)+np*ii)  = H0(3);

% plot3(Vxt,Vyt,Vz,'c','LineWidth',2);
plot3(Vx1t,Vy1t,Vz1,'b','LineWidth',2);
% plot3(Vx2t,Vy2t,Vz2,'c','LineWidth',2);
% plot3(Vx3,Vy3,Vz3,'c','LineWidth',2);
```

```
% plot3(Vx4,Vy4,Vz4,'c','LineWidth',2);
end

if ii==7

Vx5(i-(7*np-1)+ np*ii) = H0(1);
Vy5(i-(7*np-1)+np*ii)  = H0(2);
Vz5(i-(7*np-1)+np*ii)  = H0(3);

% plot3(Vxt,Vyt,Vz,'c','LineWidth',2);
plot3(Vx1t,Vy1t,Vz1,'b','LineWidth',2);
% plot3(Vx2t,Vy2t,Vz2,'c','LineWidth',2);
% plot3(Vx3,Vy3,Vz3,'c','LineWidth',2);
% plot3(Vx4,Vy4,Vz4,'c','LineWidth',2);
plot3(Vx5,Vy5,Vz5,'b','LineWidth',2);
end

if ii>7

Vx6(i-(8*np-1)+ np*ii) = H0(1);
Vy6(i-(8*np-1)+np*ii)  = H0(2);
Vz6(i-(8*np-1)+np*ii)  = H0(3);

% plot3(Vxt,Vyt,Vz,'c','LineWidth',2);
plot3(Vx1t,Vy1t,Vz1,'b','LineWidth',2);
% plot3(Vx2t,Vy2t,Vz2,'c','LineWidth',2);
% plot3(Vx3,Vy3,Vz3,'c','LineWidth',2);
% plot3(Vx4,Vy4,Vz4,'c','LineWidth',2);
plot3(Vx5,Vy5,Vz5,'b','LineWidth',2);
% plot3(Vx6,Vy6,Vz6,'c','LineWidth',2);
end


%% Ejes

% Graficación de los ejes de Matlab
plot3(p1x,p1y,p1z, 'r',p2x,p2y,p2z, 'b',p3x,p3y,p3z, 'g');

% Graficación de los ejes del robot:
plot3([T0T011(1,4) h1xyz(1,4)],[T0T011(2,4) h1xyz(2,4)],[T0T011(3,4)
h1xyz(3,4)],'r');
plot3([T0T011(1,4) h2xyz(1,4)],[T0T011(2,4) h2xyz(2,4)],[T0T011(3,4)
h2xyz(3,4)],'b');
plot3([T0T011(1,4) h3xyz(1,4)],[T0T011(2,4) h3xyz(2,4)],[T0T011(3,4)
h3xyz(3,4)],'g');

% Graficación de los ejes de la Mesa
plot3(p1x+177.5,p1y,p1z+31.065, 'r',p2x+177.5,p2y,p2z+31.065,
'b',p3x+177.5,p3y,p3z+31.065, 'g');

%% Propiedades de la figura
%Vista
% view([0,0,1])
```

```
view([0,-1,0])
%Área de dibujo
axis([-50 300 -175 175 -50 300]);

%Etiquetas
xlabel('x');
ylabel('y');
zlabel('z');

%Cuadricula
grid on;

%Pausa en la animacion
pause(.01)

end
end
figure('Name','Velocidades Angulares Primer Cordon','NumberTitle','off')
    hold on
    grid on
    plot(qp1,'g','linewidth',2);
    plot(qp2,'b','linewidth',2);
    plot(qp3,'k','linewidth',2);
    plot(qp4,'c','linewidth',2);
    plot(qp5,'m','linewidth',2);
    plot(qp6,'r','linewidth',2);
    legend('\theta_1','\theta_2','\theta_3','\theta_4','\theta_5', ...
        '\theta_6','Location','northeast','Orientation','vertical')
        xlabel('Tiempo (seg)')
    ylabel('Variables Articulares (Grados)')
    axis([1,40,-300,100])
```

B.3 Código de la función objetivo de *fmincon* para la optimización. Caso 2

```
%Se limpia el area de trabajo
clc, clear, close all;

%Se declaran los parametros del robot con escala de 7.5
d2 = 30;
d3 = 90;
d4 = 22.5;
r4 = 148.875;

%Radio de el cilindro
rc = 15;

%numero de puntos por trayectoria
np = 20;

%Tiempo para el cordon de soldadura
T  = 18;

%Tiempo para las trayectorias de acercamiento y alejamiento
T3 = 5;

%Coordenadas del centro de la mesa.
xc = 177.5;
yc = 0;
zc = 32.065;

%Linea de Radio en la mesa
xr = 177.5;
yr = -100;

% %% Variables a optimizar
% Psi Ángulo para girar la Mesa
psi    = deg2rad(4.6372);
```

```
% %Psit Ángulo para girar el cilindro
psit = deg2rad(22.1812);

% %Distancia entre los centros de la mesa y el cilindro.
 re = 31.2785;


%% Variables a optimizar
% %Psi Ángulo para girar la Mesa
% psi    = deg2rad(0);
%
% %Psit Ángulo para girar el cilindro
% psit = deg2rad(0);
%
% %Distancia entre los centros de la mesa y el cilindro.
%  re = 0;
%
%% Ángulos de la antorcha:
phieini = deg2rad(45);

phiefin = deg2rad(225);

thetae = deg2rad(135);

psie = deg2rad(0);

%%
%Matriz de Emplazamiento e Inversa
dx=0;
dy=0;
dz=22.065;

Te0 = [1  0  0  dx;
       0  1  0  dy;
       0  0  1  dz;
       0  0  0  1];

 T0e = inv(Te0);

%Matriz del Marco de la Herramienta e Inversa
T6H = [0.9239  0  -0.3827  -5;
        0      1    0       0;
       0.3827  0  0.9239   40;
        0      0    0       1];

TH6=inv(T6H);

%Matriz del Cilindro a la Mesa
TMC =  [cos(psit)  -sin(psit)  0  re;
        sin(psit)  cos(psit)   0  0;
        0              0       1  0;
        0              0       0  1];
```

```
%Matriz de la Mesa con respecto a ala estación de Trabajo
TeM =  [cos(psi)  -sin(psi)  0   xc;
        sin(psi)   cos(psi)  0   yc;
           0          0      1   zc;
           0          0      0   1];

%% Ciclos for
for ii=0:2   %Recorre el numero de trayectorias
for i=0:np;   %Recorre el tiempo

%Se definen variables
t      = T*i/np;  %tiempo de 18 seg.
tar    = T3*i/np; %tiempo de 5 seg.

fcart  = (tar/T3-(0.5/pi)*sin(2*pi*tar/T3));   %funcion cicloidal de 5
seg.

phi    = deg2rad(-90)+(deg2rad(10)*t);         %phi para Ángulo
helicoidal

zzp = 1; %Velocidad en z de la antorcha

%%Cordones %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%P1,1 a P1,2
if ii==0
phie = phieini + deg2rad(10*t);

xi = rc*cos(phi);
yi = rc*sin(phi);
zi = zzp*t;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

%P1,1 a P1,2
if ii==1

phie = phiefin + deg2rad(10*t);

xi = -rc*cos(phi);
yi = -rc*sin(phi);
zi = 18+zzp*t;

%Se forma la Matriz TCA
```

```
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

J(1,1) = -(d2*sin(t1)) - d3*cos(t2)*sin(t1) - d4*cos(t2 + t3)*sin(t1) -
r4*sin(t1)*sin(t2 + t3);

J(2,1) = d2*cos(t1) + d3*cos(t1)*cos(t2) + d4*cos(t1)*cos(t2 + t3) +
r4*cos(t1)*sin(t2 + t3);

J(3,1) = 0;

J(4,1) = 0;

J(5,1) = 0;

J(6,1) = 1;

J(1,2) = r4*cos(t1)*cos(t2 + t3) - d3*cos(t1)*sin(t2) - d4*cos(t1)*sin(t2
+ t3);

J(2,2) = r4*cos(t2 + t3)*sin(t1) - d3*sin(t1)*sin(t2) - d4*sin(t1)*sin(t2
+ t3);

J(3,2) = d3*cos(t2) + d4*cos(t2 + t3) + r4*sin(t2 + t3);

J(4,2) = sin(t1);

J(5,2) = -cos(t1);

J(6,2) = 0;

J(1,3) = r4*cos(t1)*cos(t2 + t3) - d4*cos(t1)*sin(t2 + t3);

J(2,3) = r4*cos(t2 + t3)*sin(t1) - d4*sin(t1)*sin(t2 + t3);

J(3,3) = d4*cos(t2 + t3) + r4*sin(t2 + t3);

J(4,3) = sin(t1);

J(5,3) = -cos(t1);

J(6,3) = 0;

J(1,4) = 0;

J(2,4) = 0;
```

```
J(3,4) = 0;

J(4,4) = cos(t1)*sin(t2 + t3);

J(5,4) = sin(t1)*sin(t2 + t3);

J(6,4) = -cos(t2 + t3);

J(1,5) = 0;

J(2,5) = 0;

J(3,5) = 0;

J(4,5) = cos(t4)*sin(t1) - cos(t1)*cos(t2 + t3)*sin(t4);

J(5,5) = -(cos(t1)*cos(t4)) - cos(t2 + t3)*sin(t1)*sin(t4);

J(6,5) = -(sin(t2 + t3)*sin(t4));

J(1,6) = 0;

J(2,6) = 0;

J(3,6) = 0;

J(4,6) = cos(t1)*cos(t5)*sin(t2 + t3) + cos(t1)*cos(t2 +
t3)*cos(t4)*sin(t5) + sin(t1)*sin(t4)*sin(t5);

J(5,6) = cos(t5)*sin(t1)*sin(t2 + t3) + cos(t2 +
t3)*cos(t4)*sin(t1)*sin(t5) - cos(t1)*sin(t4)*sin(t5);

J(6,6) = -(cos(t2 + t3)*cos(t5)) + cos(t4)*sin(t2 + t3)*sin(t5);

Jac = [J(1,1)  J(1,2)  J(1,3)  J(1,4) J(1,5)  J(1,6);
       J(2,1)  J(2,2)  J(2,3)  J(2,4) J(2,5)  J(2,6);
       J(3,1)  J(3,2)  J(3,3)  J(3,4) J(3,5)  J(3,6);
       J(4,1)  J(4,2)  J(4,3)  J(4,4) J(4,5)  J(4,6);
       J(5,1)  J(5,2)  J(5,3)  J(5,4) J(5,5)  J(5,6);
       J(6,1)  J(6,2)  J(6,3)  J(6,4) J(6,5)  J(6,6)];

JJA = [ J(1,1)  J(1,2)  J(1,3);
        J(2,1)  J(2,2)  J(2,3);
        J(3,1)  J(3,2)  J(3,3)];

JJR = [ J(4,4) J(4,5)  J(4,6);
        J(5,4) J(5,5)  J(5,6);
        J(6,4) J(6,5)  J(6,6)];

WA   = sqrt(det(JJA*JJA'));
```

```
fnor = 2.9381e+06;
WAn  = WA/fnor;

WR = sqrt(det(JJR*JJR'));


wan(i+1+np*ii)= WAn;
wr(i+1+np*ii)= WR;


ww = [WAn WR]';

[mean,stdev] = stat(ww);
Wmix = mean - stdev;

wmix(i+1+np*ii)= Wmix;



end
end

[mean,stdev] = stat(wmix);
Funw = mean - stdev

fk = -Funw;


% figure();
%     hold on
%     grid on
%     plot(wan,'r');
%     plot(wr,'b');
%     legend('WAn','WR','Location','northeast','Orientation','vertical')
%      xlabel('Time [s]')
%     axis([1,40,0,1.5])
%
```

B.4 Código de la función objetivo de *fmincon* para la optimización. Caso 3.

```
%Se limpia el area de trabajo
clc, clear, close all;

%Se declaran los parametros del robot con escala de 7.5
d2 = 30;
d3 = 90;
d4 = 22.5;
r4 = 148.875;

%Radio de el cilindro
rc = 15;

%numero de puntos por trayectoria
np = 20;

%Tiempo para el cordon de soldadura
T  = 18;

%Tiempo para las trayectorias de acercamiento y alejamiento
T3 = 5;

%Coordenadas del centro de la mesa.
xc = 177.5;
yc = 0;
zc = 32.065;

%Linea de Radio en la mesa
xr = 177.5;
yr = -100;

%% Variables a optimizar
% %Psi Ángulo para girar la Mesa
% psi    = deg2rad(0);
%
% %Psit Ángulo para girar el cilindro
% psit = deg2rad(0);
%
% %Distancia entre los centros de la mesa y el cilindro.
%  re = 0;
%
```

```
%   %%
%  psi2  = deg2rad(0);


%% Variables a optimizar
%Psi Ángulo para girar la Mesa
psi    = deg2rad(120.7207);

%Psit Ángulo para girar el cilindro
psit = deg2rad(-54.0510);

%Distancia entre los centros de la mesa y el cilindro.
 re = -39.9995;

%%
psi2 = deg2rad(-103.5746);

%% Ángulos de la antorcha:
phieini = deg2rad(45);

phiefin = deg2rad(225);

thetae = deg2rad(135);

psie = deg2rad(0);

%%
%Matriz de Emplazamiento e Inversa
dx=0;
dy=0;
dz=22.065;

Te0 = [1  0  0  dx;
       0  1  0  dy;
       0  0  1  dz;
       0  0  0  1];

 T0e = inv(Te0);

%Matriz del Marco de la Herramienta e Inversa
T6H = [0.9239  0  -0.3827  -5;
        0      1    0       0;
       0.3827  0  0.9239   40;
        0      0    0       1];

TH6=inv(T6H);

%Matriz del Cilindro a la Mesa
TMC =  [cos(psit)  -sin(psit)  0  re;
        sin(psit)  cos(psit)   0  0;
        0              0       1  0;
        0              0       0  1];
```

```
%Matriz de la Mesa con respecto a ala estación de Trabajo
TeM =  [cos(psi)  -sin(psi)  0   xc;
        sin(psi)   cos(psi)  0   yc;
           0          0      1   zc;
           0          0      0   1];

%% Ciclos for
for ii=0:1   %Recorre el numero de trayectorias
for i=0:np;   %Recorre el tiempo

%Se definen variables
t      = T*i/np;  %tiempo de 18 seg.
tar    = T3*i/np; %tiempo de 5 seg.

fcart  = (tar/T3-(0.5/pi)*sin(2*pi*tar/T3));   %funcion cicloidal de 5
seg.

phi    = deg2rad(-90)+(deg2rad(10)*t);         %phi para Ángulo
helicoidal

zzp = 1; %Velocidad en z de la antorcha

%%Cordones %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%P1,1 a P1,2
if ii==0
phie = phieini + deg2rad(10*t);

xi = rc*cos(phi);
yi = rc*sin(phi);
zi = zzp*t;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

% P1,1 a P1,2
if ii==1
%Matriz de la Mesa con respecto a ala estación de Trabajo
TeM =  [cos(psi2)  -sin(psi2)  0   xc;
        sin(psi2)   cos(psi2)  0   yc;
           0           0       1   zc;
           0           0       0   1];

phie = phiefin + deg2rad(10*t);
```

```
xi = -rc*cos(phi);
yi = -rc*sin(phi);
zi = 18+zzp*t;

%Se forma la Matriz TCA
[TCA]=Matriz_TCA(phie, thetae, psie, xi, yi, zi);

Tsnap = T0e * TeM * TMC * TCA * TH6;

%Se calcula la cinematica Inversa
[t1,t2,t3,t4,t5,t6]=mcip(Tsnap);
end

J(1,1) = -(d2*sin(t1)) - d3*cos(t2)*sin(t1) - d4*cos(t2 + t3)*sin(t1) -
r4*sin(t1)*sin(t2 + t3);

J(2,1) = d2*cos(t1) + d3*cos(t1)*cos(t2) + d4*cos(t1)*cos(t2 + t3) +
r4*cos(t1)*sin(t2 + t3);

J(3,1) = 0;

J(4,1) = 0;

J(5,1) = 0;

J(6,1) = 1;

J(1,2) = r4*cos(t1)*cos(t2 + t3) - d3*cos(t1)*sin(t2) - d4*cos(t1)*sin(t2
+ t3);

J(2,2) = r4*cos(t2 + t3)*sin(t1) - d3*sin(t1)*sin(t2) - d4*sin(t1)*sin(t2
+ t3);

J(3,2) = d3*cos(t2) + d4*cos(t2 + t3) + r4*sin(t2 + t3);

J(4,2) = sin(t1);

J(5,2) = -cos(t1);

J(6,2) = 0;

J(1,3) = r4*cos(t1)*cos(t2 + t3) - d4*cos(t1)*sin(t2 + t3);

J(2,3) = r4*cos(t2 + t3)*sin(t1) - d4*sin(t1)*sin(t2 + t3);

J(3,3) = d4*cos(t2 + t3) + r4*sin(t2 + t3);

J(4,3) = sin(t1);

J(5,3) = -cos(t1);
```

```
J(6,3) = 0;

J(1,4) = 0;

J(2,4) = 0;

J(3,4) = 0;

J(4,4) = cos(t1)*sin(t2 + t3);

J(5,4) = sin(t1)*sin(t2 + t3);

J(6,4) = -cos(t2 + t3);

J(1,5) = 0;

J(2,5) = 0;

J(3,5) = 0;

J(4,5) = cos(t4)*sin(t1) - cos(t1)*cos(t2 + t3)*sin(t4);

J(5,5) = -(cos(t1)*cos(t4)) - cos(t2 + t3)*sin(t1)*sin(t4);

J(6,5) = -(sin(t2 + t3)*sin(t4));

J(1,6) = 0;

J(2,6) = 0;

J(3,6) = 0;

J(4,6) = cos(t1)*cos(t5)*sin(t2 + t3) + cos(t1)*cos(t2 +
t3)*cos(t4)*sin(t5) + sin(t1)*sin(t4)*sin(t5);

J(5,6) = cos(t5)*sin(t1)*sin(t2 + t3) + cos(t2 +
t3)*cos(t4)*sin(t1)*sin(t5) - cos(t1)*sin(t4)*sin(t5);

J(6,6) = -(cos(t2 + t3)*cos(t5)) + cos(t4)*sin(t2 + t3)*sin(t5);

Jac = [J(1,1)  J(1,2)  J(1,3)  J(1,4) J(1,5)  J(1,6);
       J(2,1)  J(2,2)  J(2,3)  J(2,4) J(2,5)  J(2,6);
       J(3,1)  J(3,2)  J(3,3)  J(3,4) J(3,5)  J(3,6);
       J(4,1)  J(4,2)  J(4,3)  J(4,4) J(4,5)  J(4,6);
       J(5,1)  J(5,2)  J(5,3)  J(5,4) J(5,5)  J(5,6);
       J(6,1)  J(6,2)  J(6,3)  J(6,4) J(6,5)  J(6,6)];

JJA = [ J(1,1)  J(1,2)  J(1,3);
        J(2,1)  J(2,2)  J(2,3);
        J(3,1)  J(3,2)  J(3,3)];
```

```
JJR = [ J(4,4) J(4,5)  J(4,6);
        J(5,4) J(5,5)  J(5,6);
        J(6,4) J(6,5)  J(6,6)];

WA   = sqrt(det(JJA*JJA'));
fnor = 2.9381e+06;
WAn  = WA/fnor;

WR = sqrt(det(JJR*JJR'));


wan(i+1+np*ii)= WAn;
if i>=1
wr(i+np*ii)= WR;
end




ww = [WAn WR]';

[mean,stdev] = stat(ww);
Wmix = mean - stdev;

wmix(i+1+np*ii)= Wmix;



end
end
% wr(20)=wr(21)-0.02;
[mean,stdev] = stat(wmix);
Funw = mean - stdev

fk = -Funw;


% figure();
%     hold on
%     grid on
%     plot(wan,'r');
%     plot(wr,'b');
%     legend('WAn','WR','Location','northeast','Orientation','vertical')
%      xlabel('Time [s]')
%     axis([1,np*2,0,1.5])
```