

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES
Departamento de Electrónica

Materia: Robótica
Trabajo Práctico 1

Docente: Ing. Hernán Gianetta

Ayudante de TP: Ing. Damián Granzella

Grupo N°:

Alumnos :

	Apellido y Nombre	Legajo
1	Resquín, Fernando Emilio	112316-6

Tabla de contenido

1. Introducción Teórica	2
1.1 Cinemática – Definición:	2
1.2 Robot Manipulador:	2
1.2.1 Problema cinemático directo	3
1.2.2 Problema cinemático inverso	4
1.2.3 Modelo diferencial (Matriz Jacobiana)	4
1.3 Robot Móvil:	5
1.3.1 Restricciones cinemáticas	5
1.3.2 Tipos de robots móviles con ruedas	6
1.3.3 Centro instantáneo de Rotación (CIR) o centro instantáneo de curvatura (CIC)	6
1.3.4 Teorema de Kennedy	7
2. Descripción y desarrollo del robot elegido	7
2.1 Manipulador – Descripción del modelo cinemático directo	8
2.2 Manipulador – Descripción del modelo cinemático inverso	10
2.3 Plataforma móvil – Desarrollo del modelo cinemático	12
2.3.1 Cinemática directa	14
2.3.2 Estimación de la posición	15
2.3.3 Cinemática inversa	16
3. Simulación en Matlab	17
3.1 Manipulador	18
3.2 Plataforma Móvil	22
4. Implementación del modelo cinemático en DSP	27
4.1 Manipulador	28
4.2 Plataforma Móvil	33
5. Conclusiones	34
6. Referencias	35

1. Introducción Teórica

1.1 Cinemática – Definición:

La cinemática es la parte de la mecánica clásica que estudia las leyes del movimiento de los cuerpos sin tener en cuenta las causas que lo producen, limitándose esencialmente, al estudio de la trayectoria en función del tiempo.

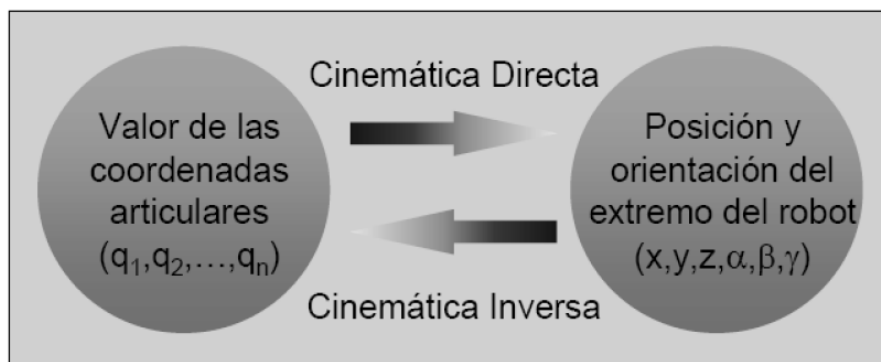
En la cinemática se utiliza un sistema de coordenadas para describir las trayectorias y se le llama sistema de referencia.

La cinemática del robot consiste en el estudio de su movimiento con respecto a un sistema de referencia y la descripción analítica del movimiento espacial en función del tiempo.

1.2 Robot Manipulador:

Para el caso de la robótica de manipuladores, la cinemática se ocupa de la relación entre la localización del extremo del robot y los valores de sus articulaciones, por lo tanto implica la resolución de los siguientes problemas:

- **Problema cinemático directo:** Determinar la posición y orientación del extremo final del robot, con respecto a un sistema de coordenadas de referencia, conocidos los ángulos de las articulaciones y los parámetros geométricos de los elementos del robot.
- **Problema cinemático inverso:** Determinar la configuración que debe adoptar el robot para alcanzar una posición y orientación del extremo conocidas.
- **Modelo diferencial (Matriz Jacobiana):** Relaciones entre las velocidades de las articulaciones y las del extremo del robot, en sentido directo e inverso.



En la obra de Barrientos {1} se podrá encontrar toda la teoría respecto de los robots manipuladores.

1.2.1 Problema cinemático directo:

Uno de los métodos para encararlo es a partir de relaciones geométricas (trigonometría). Es un método no sistemático y sólo es útil para robots con pocos grados de libertad.

El otro método se basa en matrices de transformación homogéneas. Básicamente se asocia una matriz de transformación homogénea a cada eslabón, que permite pasar del sistema de coordenadas de ese eslabón al sistema del anterior. Las matrices contienen translaciones y rotaciones convenientemente

elegidas. Denavit y Hartenberg propusieron un método sistemático para obtener estas matrices. El producto de esas matrices permite obtener la transformación entre las coordenadas articulares y las del extremo.

$$T = {}_0A_{1.1}A_{2.2}A_3$$

Por otro lado, la cinemática del robot trata también de encontrar las relaciones entre las velocidades del movimiento de las articulaciones y las del extremo. Esta relación viene dada por el modelo diferencial expresado mediante la matriz Jacobiana.

1.2.2 Problema cinemático inverso:

No existe un método sistemático para abordar el problema cinemático inverso. Además, el procedimiento de obtención de las ecuaciones es fuertemente dependiente de la configuración del robot.

A la hora de resolver el problema cinemático inverso es mucho más adecuado encontrar una solución cerrada de la forma:

$$q_k = f_k(x, y, z, \alpha, \beta, \gamma) \\ k=1 \dots n(\text{GDL})$$

Con este método se obtiene una función por cada coordenada articular. Si bien los cálculos requieren un poder computacional importante, se realizan en tiempo real. Cabe aclarar que muchas veces las soluciones no son únicas, por lo que hay que aplicar algún criterio para elegir la solución adecuada. Existen dos métodos para obtener estas funciones. El método geométrico consiste en obtener las relaciones geométricas y trigonométricas, a partir de los triángulos formados por los elementos. Sirve sólo para pocos grados de libertad.

El otro método es el de la matriz homogénea, en el cual se forman ecuaciones matriciales a partir de :

$$T = {}_0A_{1.1}A_{2.2}A_3$$

Se despeja alguna de las sub-matrices usando la matriz inversa. Luego se forma un sistema de ecuaciones alineaes a partir de comparar elementos, que permitan despejar las variables articulares.

1.2.3 Modelo diferencial (Matriz Jacobiana):

Permite conocer las relaciones de velocidades entre las variables articulares y las del extremo. La Matriz Jacobiana Directa se obtiene derivando las ecuaciones correspondientes al modelo cinemático directo. La Matriz Jacobiana Inversa se obtiene invirtiendo la jacobiana directa sea simbólicamente o numéricamente.

1.3 Robot Móvil:

Para el sgte trabajo nos basaremos en el marco teórico para los robots móviles con ruedas, propuesto por Olleros {2}, el cual supone las siguientes hipótesis:

- a) El robot se mueve sobre una superficie plana (x, y).
- b) Los ejes de guiado son perpendiculares al suelo.
- c) Se supone que las ruedas se mueven por rodadura pura, es decir que el deslizamiento es despreciable en el período de control.
- d) El robot no tiene partes flexibles.
- e) Durante un período de tiempo suficientemente pequeño en el que se mantiene constante la consigna de dirección, el robot se moverá de un punto al siguiente a lo largo de un arco de circunferencia.
- f) El robot se comporta como un sólido rígido, de forma que si existen partes móviles (rueda de dirección), éstas se situarán en la posición adecuada mediante el sistema de control.

También utilizaremos conceptos de los sgtes papers {3} {7} {8}.


En un sistema físico, en este caso el robot, sus partes están sometidas a restricciones cinemáticas que determinan su movimiento. Las mismas se obtienen como parte del modelo cinemático del robot. Las restricciones pueden ser holónomas y no holónomas.

Las holónomas son aquellas que no dependen de las velocidades. En cambio las no holónomas dependen de las velocidades y además no pueden deducirse mediante la integración de otra restricción. Los robots manipuladores se caracterizan porque sus restricciones son holónomas. En cambio, en los robots móviles con ruedas, considerando la rodadura de las ruedas sobre un plano, se suelen presentar restricciones no holónomas.

1.3.1 Restricciones cinemáticas:

Matemáticamente: $G(p, \dot{p}, t) = 0$

R. Holónoma no depende de \dot{p}


$$\dot{x} = R \cdot \dot{\theta}$$

↓

$$\int dt \rightarrow x - R \cdot \theta = cte$$

**R. No Holónoma depende de \dot{p}
y no es integrable**

Una rueda puede rodar hacia adelante y atrás pero no es la idea que se pueda mover de costado, en la dirección del eje de la rueda, porque habría deslizamiento. Por lo tanto, el movimiento de las ruedas del robot posee restricciones cinemáticas.

1.3.2 Tipos de robots móviles con ruedas:

Para encarar este estudio el autor utiliza el Modelo Jacobiano, el cual está basado en velocidades. Dicho modelo vincula la velocidad angular de cada rueda con la velocidad de translación y la de giro del cuerpo del robot. Existen varias configuraciones típicas de robots móviles, los cuales poseen cada uno diferente modelo jacobiano:

- Locomoción diferencial
- Locomoción tipo Triciclo
- Locomoción tipo Ackerman
- Otros: Síncrona, Omnidireccional, etc.

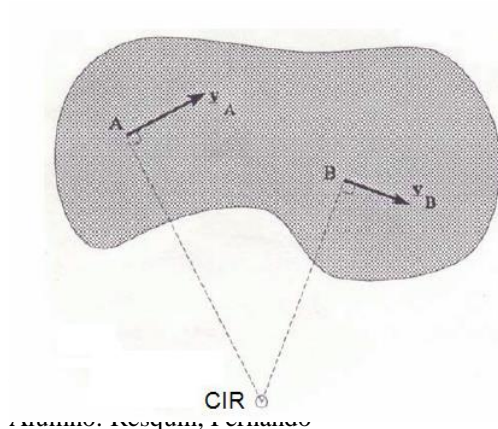
Algunos conceptos previos antes de encarar el estudio del robot sujeto de este trabajo:

Tipos de ruedas:

- Rueda motriz: La que proporciona fuerza de tracción al robot.
- Rueda directriz: Ruedas de direccionamiento de orientación controlable.
- Ruedas fijas: Sólo giran en torno a su eje sin tracción motriz.
- Ruedas locas o ruedas de castor: Ruedas orientables no controladas.

1.3.3 Centro instantáneo de Rotación (CIR) o centro instantáneo de curvatura (CIC):

El centro instantáneo de rotación, también llamado centro instantáneo, es el punto fijo a un cuerpo sometido a movimiento planar que tiene velocidad cero en un instante de tiempo determinado, o sea que está sometido a rotación, pero no a translación.



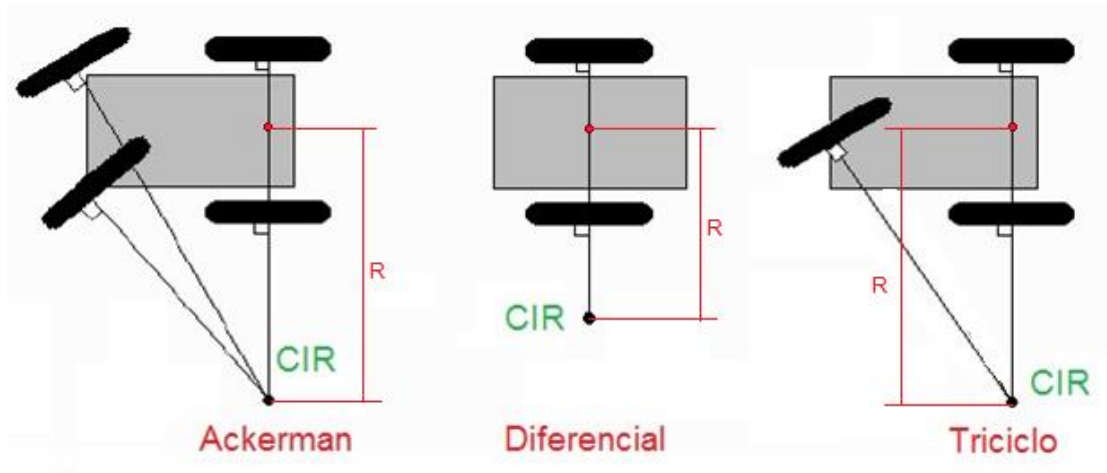
En un cuerpo en rotación, las velocidades de cada punto del mismo tienen un radio de giro asociado, el cual siempre es perpendicular al vector de velocidad asociado al punto. Tomando dos puntos A y B, con sus vectores de velocidad V_A y V_B , el CIR queda determinado por el punto de cruce de los radios de giro.

En caso de que el movimiento del cuerpo sea de pura translación el CIR se encuentra en un punto en el infinito.

1.3.4 Teorema de Kennedy:

Si tres cuerpos tienen movimiento plano uno respecto del otro, entonces hay tres centros instantáneos de rotación, y los 3 centros instantáneos se encuentran en la misma línea.

El concepto de CIR y el teorema de Kennedy son aplicables a los robots móviles con ruedas:



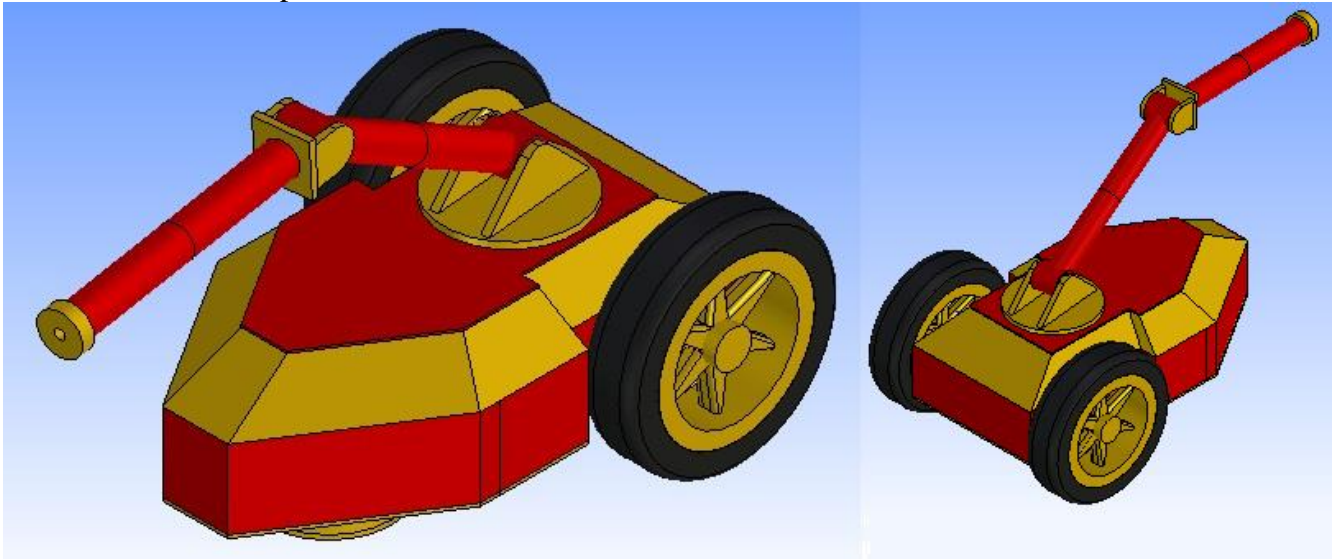
Se indica con un punto rojo el radio de curvatura de la trayectoria del robot, de valor R . En ese eje se cumple el teorema de Kennedy.

2. Descripción y desarrollo del robot elegido

El sujeto de este trabajo es un robot móvil con ruedas en configuración diferencial, el cual puede ser utilizado para tareas varias en lugares peligrosos para el ser humano, como ser plantas industriales, centros urbanos, o lugares donde el suelo sea relativamente plano, dentro de las posibilidades de movimiento de las ruedas. El robot no posee tracción por orugas ni flaps para trepar escaleras o subirse a obstáculos.

El sistema incorpora un brazo manipulador de 3 grados de libertad que posiciona el extremo, al cual se le coloca un accesorio que incorpora la herramienta necesaria para la tarea a realizar y además la muñeca que determina la orientación de la misma. Se utiliza entonces un enfoque de desacoplo

cinemático en el manipulador.



Como se ve en la figura, el robot tiene dos ruedas de tracción en configuración diferencial ubicados en la parte trasera de la plataforma móvil. Los motoredutores y las baterías están ubicadas también en la parte trasera, para aumentar la tracción. Para soportar la parte delantera se utiliza una rueda loca tipo “castor” que facilita los giros del robot. El movimiento de la plataforma se basa en el seguimiento de trayectorias en base a una sucesión de puntos interpolados, las cuales son determinadas por un compilador externo o en los niveles superiores del control.

El presente trabajo está enfocado en la parte de robótica móvil del proyecto. Si bien se incluye el modelo cinemático del manipulador, el mismo fue tomado de una publicación de otro autor, sin desarrollarlo completamente. Sólo se adaptó el desarrollo al robot en cuestión.

El modelo cinemático de la parte móvil se desarrolló a partir de publicaciones y libros de otros autores para obtener el marco teórico necesario o las referencias para el desarrollo.

Objetivos:

- Descripción del modelo cinemático del manipulador. Determinación del espacio de trabajo.
- Desarrollo del modelo cinemático de la plataforma móvil.
- Simulación y validación de ambos modelos cinemáticos en Matlab.
- Programación de los modelos cinemáticos de ambas partes del robot en DSP, usando Codewarrior, con la posterior verificación contra los resultados de la simulación.

2.1 Manipulador – Descripción del modelo cinemático directo

Como se mencionó anteriormente, este trabajo no tiene por objetivo el estudio completo de un robot manipulador, sino que se enfoca en la robótica móvil. El modelo cinemático directo de esta sección del robot se obtuvo de la publicación de Arias y Fonseca {6}. Se transcribe a continuación los resultados del método de Denavit y Hartenberg, modificándolo brevemente para mantener en valores simbólicos la longitud de los eslabones.

Articulación	Θ	d	a	α
1	Θ_1	l_1	0	$\pi/2$
2	Θ_2	0	l_2	0
3	Θ_3	0	0	l_3

$$H_0A_1 = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 0 \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_1A_2 = \begin{bmatrix} \cos(\theta_2) & -\sin(\theta_2) & 0 & l_2 \cdot \cos(\theta_2) \\ \sin(\theta_2) & \cos(\theta_2) & 0 & l_2 \cdot \sin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_2A_3 = \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & l_3 \cdot \cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3) & 0 & l_3 \cdot \sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_0A_3 = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix}$$

$$A_{11} = \cos(\Theta_1) \cdot \cos(\Theta_2) \cdot \cos(\Theta_3) - \cos(\Theta_1) \cdot \sin(\Theta_2) \cdot \sin(\Theta_3)$$

$$A_{12} = -\cos(\Theta_1) \cdot \cos(\Theta_2) \cdot \sin(\Theta_3) - \cos(\Theta_1) \cdot \sin(\Theta_2) \cdot \cos(\Theta_3)$$

$$A_{13} = \sin(\Theta_1)$$

$$A_{14} = \cos(\Theta_1) \cdot \cos(\Theta_2) \cdot l_3 \cdot \cos(\Theta_3) - \cos(\Theta_1) \cdot \sin(\Theta_2) \cdot l_3 \cdot \sin(\Theta_3) + \cos(\Theta_1) \cdot l_2 \cdot \cos(\Theta_2)$$

$$A_{21} = \sin(\Theta_1) \cdot \cos(\Theta_2) \cdot \cos(\Theta_3) - \sin(\Theta_1) \cdot \sin(\Theta_2) \cdot \sin(\Theta_3)$$

$$A_{22} = -\sin(\Theta_1) \cdot \cos(\Theta_2) \cdot \sin(\Theta_3) - \sin(\Theta_1) \cdot \sin(\Theta_2) \cdot \cos(\Theta_3)$$

$$A_{23} = -\cos(\Theta_1)$$

$$A_{24} = \sin(\Theta_1) \cdot \cos(\Theta_2) \cdot l_3 \cdot \cos(\Theta_3) - \sin(\Theta_1) \cdot \sin(\Theta_2) \cdot l_3 \cdot \sin(\Theta_3) + \sin(\Theta_1) \cdot l_2 \cdot \cos(\Theta_2)$$

$$A_{31} = \sin(\Theta_2) \cdot \cos(\Theta_3) + \cos(\Theta_2) \cdot \sin(\Theta_3)$$

$$A_{32} = -\sin(\Theta_2) \cdot \sin(\Theta_3) + \cos(\Theta_2) \cdot \cos(\Theta_3)$$

$$A_{33} = 0$$

$$A_{34} = \sin(\Theta_2) \cdot l_3 \cdot \cos(\Theta_3) + \cos(\Theta_2) \cdot l_3 \cdot \sin(\Theta_3) + l_2 \cdot \sin(\Theta_2) + l_1$$

$$A_{31} = A_{32} = A_{33} = 0$$

$$A_{34} = 1$$

Es necesario confeccionar una matriz de transformación homogénea para pasar del sistema fijo de referencia (tierra) a las coordenadas locales de la plataforma móvil, sobre la cual se ubica el manipulador.

$$Rot_H = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) & 0 & 0 \\ \sin(\varphi) & \cos(\varphi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Transl_H = \begin{bmatrix} 1 & 0 & 0 & Xg \\ 0 & 1 & 0 & Yg \\ 0 & 0 & 1 & Zg \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Tg = Transl_H * Rot_H = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) & 0 & Xg \\ \sin(\varphi) & \cos(\varphi) & 0 & Yg \\ 0 & 0 & 1 & Zg \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.2 Manipulador – Descripción del modelo cinemático inverso

En esta sección se busca obtener el modelo cinemático inverso del manipulador que es útil para obtener las coordenadas articulares del manipulador a partir de las coordenadas del extremo expresadas en el sistema local de la plataforma móvil. Nuevamente nos basamos en la publicación de Arias y Fonseca [6], pero en este caso hubo que deducir nuevamente las funciones para que la longitud de los eslabones queden expresados en forma simbólica. Todas las consideraciones y aproximaciones indicadas en el paper siguen siendo válidas.

$$P_Z \cdot \sin(\Theta_2) - l_2 - l_1 \cdot \sin(\Theta_2) + P_X \cdot \cos(\Theta_1) \cdot \cos(\Theta_2) + P_Y \cdot \sin(\Theta_1) \cdot \cos(\Theta_2) = l_3 \cdot \cos(\Theta_3) \quad (17)$$

$$P_Z \cdot \cos(\Theta_2) - l_1 \cdot \cos(\Theta_2) - P_X \cdot \cos(\Theta_1) \cdot \sin(\Theta_2) - P_Y \cdot \sin(\Theta_1) \cdot \sin(\Theta_2) = l_3 \cdot \sin(\Theta_3) \quad (18)$$

$$P_X \cdot \sin(\Theta_1) - P_Y \cdot \cos(\Theta_1) = 0 \quad (19) \Rightarrow P_X \cdot \sin \Theta_1 = P_Y \cdot \cos \Theta_1 \Rightarrow \Theta_1 = \tan^{-1} \frac{P_Y}{P_X}$$

De (17):

$$\sin(\Theta_2) \cdot [P_Z - l_1] + \cos(\Theta_2) [P_X \cdot \cos(\Theta_1) + P_Y \cdot \sin(\Theta_1)] - l_2 = l_3 \cdot \cos(\Theta_3) \quad (23)$$

De (18):

$$-\sin(\Theta_2) \cdot [P_X \cdot \cos(\Theta_1) + P_Y \cdot \sin(\Theta_1)] + \cos(\Theta_2) \cdot [P_Z - l_1] = l_3 \cdot \sin(\Theta_3) \quad (25)$$

Defino:

$$a = P_X \cdot \cos(\Theta_1) + P_Y \cdot \sin(\Theta_1) \quad (26) \quad ; \quad b = P_Z - l_1 \quad (27)$$

De (23):

$$\sin(\Theta_2) \cdot b + \cos(\Theta_2) \cdot a - l_2 = l_3 \cdot \cos(\Theta_3) \quad (28)$$

De (25):

$$-\sin(\Theta_2) \cdot a + \cos(\Theta_2) \cdot b = l_3 \cdot \sin(\Theta_3) \quad (29)$$

Dividiendo la ecuación (29) por la (28):

$$\tan \Theta_3 = \frac{-\sin \Theta_2 \cdot a + \cos \Theta_2 \cdot b}{\sin \Theta_2 \cdot b + \cos \Theta_2 \cdot a - l_2}$$

$$\Rightarrow \Theta_3 = \tan^{-1} \left[\frac{-\sin \Theta_2 \cdot a + \cos \Theta_2 \cdot b}{\sin \Theta_2 \cdot b + \cos \Theta_2 \cdot a - l_2} \right]$$

Para determinar la expresión para Θ_2 el autor utiliza una aproximación polinomial para aproximar una ecuación trascendente:

$$\Theta_2 = 2 \cdot \tan^{-1} \left[\frac{b \pm \sqrt{(b^2 + a^2 - c^2)}}{a + c} \right]$$

Determinación de la expresión de “c”:

Se toman las ecuaciones (28) y (29), se las eleva al cuadrado y luego se las suma

De (28):

$$\sin^2(\Theta_2). b^2 + [\cos(\Theta_2). a - l_2]^2 + 2. \sin(\Theta_2). b[\cos(\Theta_2). a - l_2] = l_3^2 \cdot \cos^2(\Theta_3)$$

$$\Rightarrow \sin^2(\Theta_2). b^2 + \cos^2(\Theta_2). a^2 + l_2^2 - 2.\cos(\Theta_2).a.l_2 + 2. \sin(\Theta_2).\cos(\Theta_2).b.a - 2.\sin(\Theta_2).b.l_2 = l_3^2 \cdot \cos^2(\Theta_3) \quad (28')$$

De (29):

$$\sin^2(\Theta_2).a^2 + \cos^2(\Theta_2).b^2 - 2.\sin(\Theta_2).\cos(\Theta_2).b.a = l_3^2 \cdot \sin^2(\Theta_3) \quad (29')$$

Sumando miembro a miembro (28') y (29'):

$$\sin^2(\Theta_2). b^2 + \sin^2(\Theta_2).a^2 + \cos^2(\Theta_2). a^2 + \cos^2(\Theta_2).b^2 + l_2^2 - 2.\cos(\Theta_2).a.l_2 - 2.\sin(\Theta_2).b.l_2 = l_3^2$$

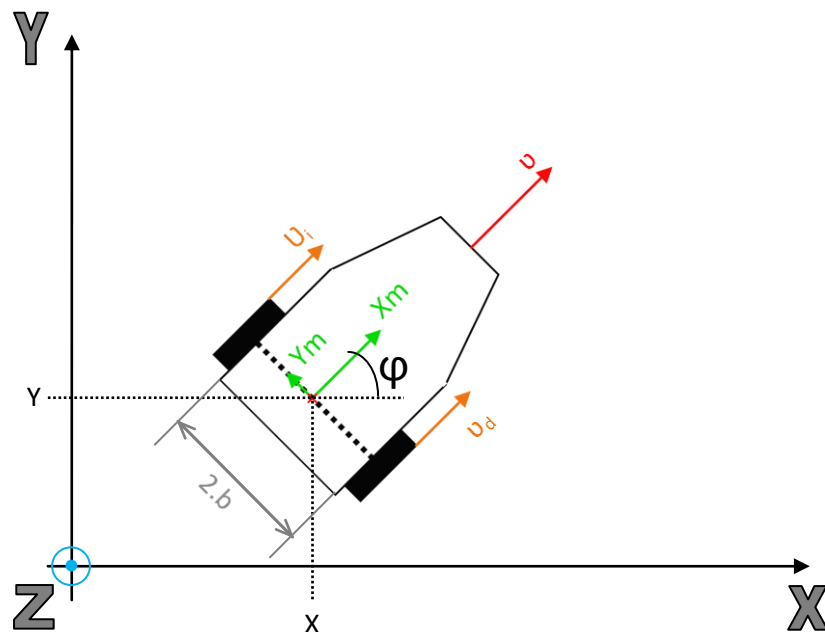
$$\Rightarrow \sin^2(\Theta_2).(b^2 + a^2) + \cos^2(\Theta_2). (a^2 + b^2) - 2. l_2.[\cos(\Theta_2).a. + 2.\sin(\Theta_2).b] + l_2^2 = l_3^2$$

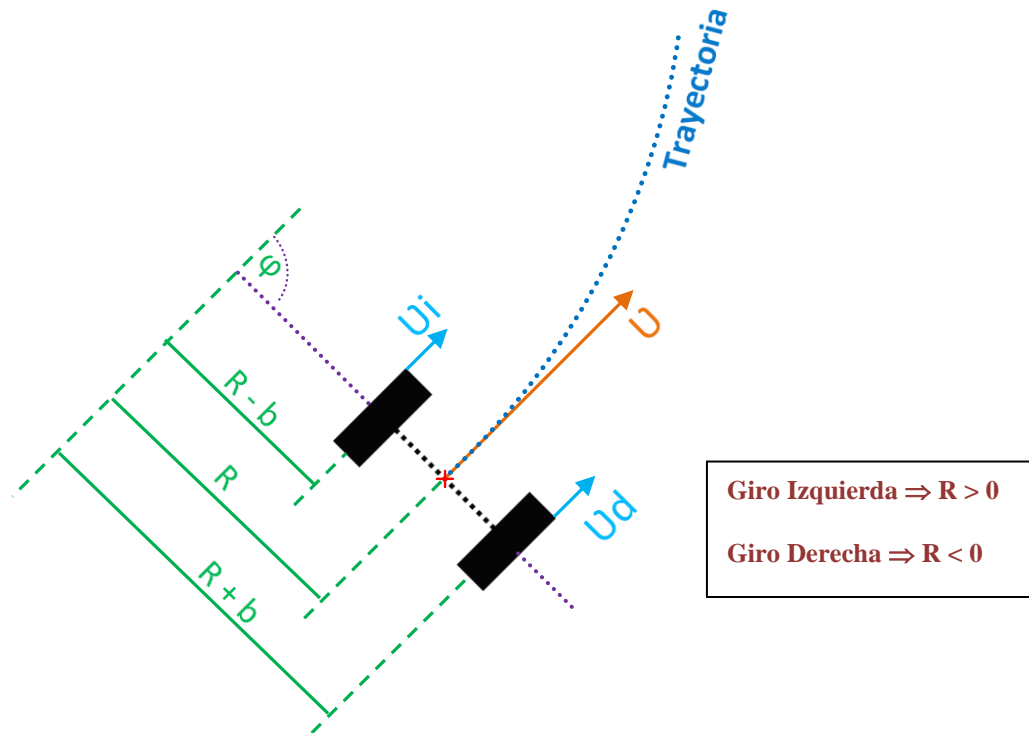
$$\Rightarrow \frac{[\sin^2(\Theta_2) + \cos^2(\Theta_2)]}{=1} .(a^2 + b^2) - 2. l_2.[\cos(\Theta_2).a. + 2.\sin(\Theta_2).b] + l_2^2 = l_3^2$$

$$\Rightarrow c = a. \cos \Theta_2 + b. \sin \Theta_2 = \frac{(a^2 + b^2) - l_3^2 + l_2^2}{2. l_2}$$

$c = a. \cos \Theta_2 + b. \sin \Theta_2$ forma parte de la función de Θ_2 por lo tanto se buscó encontrar otra ecuación que dependa de variables de valores conocidos o de Θ_1 .

2.3 Plataforma móvil – Desarrollo del modelo cinemático





ω : Velocidad angular del robot referida a su eje de giro. Vinculada a ϕ .

v : Velocidad lineal de la plataforma.

v_i : Velocidad lineal de la rueda izquierda.

v_d : Velocidad lineal de la rueda derecha.

R : Radio de curvatura instantáneo de la trayectoria que está siguiendo el robot.

Θ_i : Posición angular de la rueda de tracción izquierda.

Θ_d : Posición angular de la rueda de tracción derecha.

ω_i : Velocidad angular de la rueda de tracción izquierda.

ω_d : Velocidad angular de la rueda de tracción derecha.

r : Radio de las ruedas de tracción.

A partir de la publicación de Jaime Jiménez Cuesta [\[5\]](#), el libro de Olleros [\[2\]](#) y tomando conceptos de [\[3\]](#) [\[7\]](#) [\[8\]](#) desarrollamos la matriz cinemática de la plataforma móvil.

2.3.1 Cinemática directa:

$$v = \frac{v_i + v_d}{2} \quad ; \quad v_d = \omega \cdot (R + b) \quad ; \quad v_i = \omega \cdot (R - b)$$

Velocidad angular del robot en la trayectoria: $\omega = \frac{(v_d - v_i)}{2 \cdot b}$

Velocidad angular del robot en la trayectoria: $R = b \cdot \frac{(v_d + v_i)}{(v_d - v_i)}$

$$\begin{cases} v = \frac{1}{2} \cdot v_i + \frac{1}{2} \cdot v_d \\ \omega = -\frac{1}{2 \cdot b} \cdot v_i + \frac{1}{2 \cdot b} \cdot v_d \end{cases} \Rightarrow \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2 \cdot b} & \frac{1}{2 \cdot b} \end{bmatrix} \cdot \begin{bmatrix} v_i \\ v_d \end{bmatrix}$$

$$\begin{cases} \dot{x} = v \cdot \cos \varphi \\ \dot{y} = v \cdot \sin \varphi \\ \dot{\varphi} = \omega \end{cases} \Rightarrow \begin{cases} \dot{x} = \frac{1}{2} \cdot v_i \cdot \cos \varphi + \frac{1}{2} \cdot v_d \cdot \cos \varphi \\ \dot{y} = \frac{1}{2} \cdot v_i \cdot \sin \varphi + \frac{1}{2} \cdot v_d \cdot \sin \varphi \\ \dot{\varphi} = -\frac{1}{2 \cdot b} \cdot v_i + \frac{1}{2 \cdot b} \cdot v_d \end{cases}$$

$$\mathbf{v}_i = \dot{\theta}_i \cdot r \quad ; \quad \mathbf{v}_d = \dot{\theta}_d \cdot r$$

$$\begin{cases} \dot{x} = \frac{1}{2} \cdot \dot{\theta}_i \cdot r \cdot \cos \varphi + \frac{1}{2} \cdot \dot{\theta}_d \cdot r \cdot \cos \varphi \\ \dot{y} = \frac{1}{2} \cdot \dot{\theta}_i \cdot r \cdot \sin \varphi + \frac{1}{2} \cdot \dot{\theta}_d \cdot r \cdot \sin \varphi \\ \dot{\varphi} = -\frac{1}{2 \cdot b} \cdot \dot{\theta}_i \cdot r + \frac{1}{2 \cdot b} \cdot \dot{\theta}_d \cdot r \end{cases} \Rightarrow \begin{cases} \dot{x} = \frac{1}{2} \cdot r \cdot \cos \varphi \cdot (\dot{\theta}_i + \dot{\theta}_d) \\ \dot{y} = \frac{1}{2} \cdot r \cdot \sin \varphi \cdot (\dot{\theta}_i + \dot{\theta}_d) \\ \dot{\varphi} = \frac{1}{2 \cdot b} \cdot r \cdot (\dot{\theta}_d - \dot{\theta}_i) \end{cases}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} \frac{r}{2} \cdot \cos \varphi & \frac{r}{2} \cdot \cos \varphi \\ \frac{r}{2} \cdot \sin \varphi & \frac{r}{2} \cdot \sin \varphi \\ -\frac{r}{2 \cdot b} & \frac{r}{2 \cdot b} \end{bmatrix} \cdot \begin{bmatrix} \dot{\theta}_i \\ \dot{\theta}_d \end{bmatrix}$$

2.3.2 Estimación de la posición:

La estimación de la posición y orientación del robot puede hacerse integrando en el tiempo la componente de velocidad en X, la componente de velocidad en Y, y además la velocidad angular ω de la plataforma móvil al seguir la trayectoria.

$$\begin{cases} x = X_0 + \int_0^t \frac{1}{2} \cdot r \cdot \cos \varphi \cdot (\dot{\theta}_i + \dot{\theta}_d) \cdot dt \\ y = Y_0 + \int_0^t \frac{1}{2} \cdot r \cdot \sin \varphi \cdot (\dot{\theta}_i + \dot{\theta}_d) \cdot dt \\ \varphi = \phi_0 + \int_0^t \frac{1}{2 \cdot b} \cdot r \cdot (\dot{\theta}_d - \dot{\theta}_i) \cdot dt \end{cases}$$

$$\begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \\ \phi_0 \end{bmatrix} + \begin{bmatrix} \int_0^t \frac{1}{2} \cdot r \cdot \cos \varphi \cdot (\dot{\theta}_i + \dot{\theta}_d) \cdot dt \\ \int_0^t \frac{1}{2} \cdot r \cdot \sin \varphi \cdot (\dot{\theta}_i + \dot{\theta}_d) \cdot dt \\ \int_0^t \frac{1}{2 \cdot b} \cdot r \cdot (\dot{\theta}_d - \dot{\theta}_i) \cdot dt \end{bmatrix}$$

2.3.3 Cinemática inversa:

En los casos en que la matriz no es cuadrada se calcula la matriz pseudoinversa, definida como:

$$A^+ = (A^T \cdot A)^T$$

Defino: $A = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2.b} & \frac{1}{2.b} \end{bmatrix}$; Por Matlab: $A^{-1} = \begin{bmatrix} 1 & -b \\ 1 & b \end{bmatrix}$

Sabiendo que:

$$\begin{cases} \dot{x} = v \cdot \cos \varphi \\ \dot{y} = v \cdot \sin \varphi \\ \dot{\varphi} = \omega \end{cases} \Rightarrow \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} \cos \varphi & 0 \\ \sin \varphi & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v \\ \omega \end{bmatrix}$$

Defino $B = \begin{bmatrix} \cos \varphi & 0 \\ \sin \varphi & 0 \\ 0 & 1 \end{bmatrix}$

Por Matlab, Pseudoinversa (B) = $\begin{bmatrix} \cos \varphi & \sin \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$\text{Luego: } \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \cos \varphi & \sin \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix} \quad (1)$$

$$\text{Además: } \begin{bmatrix} v_i \\ v_d \end{bmatrix} = \begin{bmatrix} 1 & -b \\ 1 & b \end{bmatrix} \quad (2)$$

De (2):

$$\begin{cases} v_i = v - b \cdot \omega \\ v_d = v + b \cdot \omega \end{cases} \quad (4)$$

De (1):

$$\begin{cases} v_i = \dot{x} \cdot \cos \varphi + \dot{y} \cdot \sin \varphi \\ \omega = \dot{\varphi} \end{cases} \quad (3)$$

Reemplazando (3) en (4):

$$\begin{cases} \mathbf{v}_i = \dot{x} \cdot \cos \varphi + \dot{y} \cdot \sin \varphi - \mathbf{b} \cdot \dot{\varphi} \\ \mathbf{v}_d = \dot{x} \cdot \cos \varphi + \dot{y} \cdot \sin \varphi + \mathbf{b} \cdot \dot{\varphi} \end{cases} \quad (5)$$

$$\begin{bmatrix} \mathbf{v}_i \\ \mathbf{v}_d \end{bmatrix} = \begin{bmatrix} \cos \varphi & \sin \varphi & -b \\ \cos \varphi & \sin \varphi & b \end{bmatrix} \cdot \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix}$$

Tomando en cuenta que $\mathbf{v}_i = \dot{\theta}_i \cdot r$ y $\mathbf{v}_d = \dot{\theta}_d \cdot r$ y reemplazando en (5)

$$\begin{cases} \dot{\theta}_i = \dot{x} \cdot \frac{\cos \varphi}{r} + \dot{y} \cdot \frac{\sin \varphi}{r} - \frac{b}{r} \cdot \dot{\varphi} \\ \dot{\theta}_d = \dot{x} \cdot \frac{\cos \varphi}{r} + \dot{y} \cdot \frac{\sin \varphi}{r} + \frac{b}{r} \cdot \dot{\varphi} \end{cases} \quad (5)$$

$$\begin{bmatrix} \dot{\theta}_i \\ \dot{\theta}_d \end{bmatrix} = \begin{bmatrix} \frac{\cos \varphi}{r} & \frac{\sin \varphi}{r} & -\frac{b}{r} \\ \frac{\cos \varphi}{r} & \frac{\sin \varphi}{r} & \frac{b}{r} \end{bmatrix} \cdot \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix}$$

3. Simulación en Matlab

Se llevó a cabo la simulación de la cinemática directa e inversa para el Manipulador y para la Plataforma Móvil. Se generaron varios scripts de Matlab con rutinas, algoritmos y funciones, para que sirvan de guía en la implementación de la matriz cinemática en el DSP.

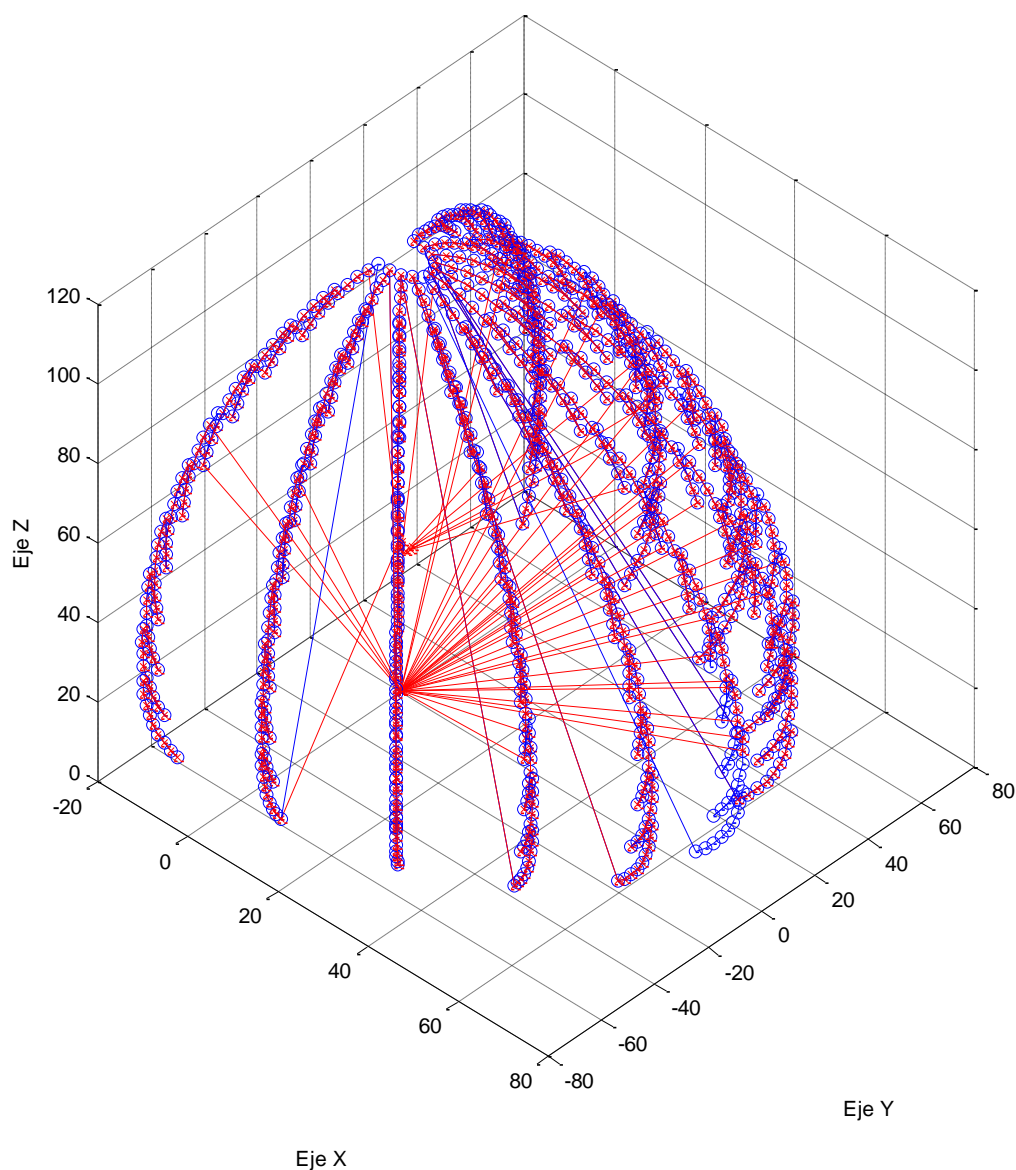
Tanto para el Manipulador como la plataforma, el script permite comparar las variables de interés, expresadas en el sistema de coordenadas de terreno o referencia, producto de la cinemática directa, y compararlas con los valores que resultan de la cinemática inversa. Con este enfoque se buscó darle

a la matriz cinemática una estructura adecuada para facilitar la conformación del lazo cerrado de control en capas superiores del sistema. Además, permite validar el desarrollo teórico descripto anteriormente.

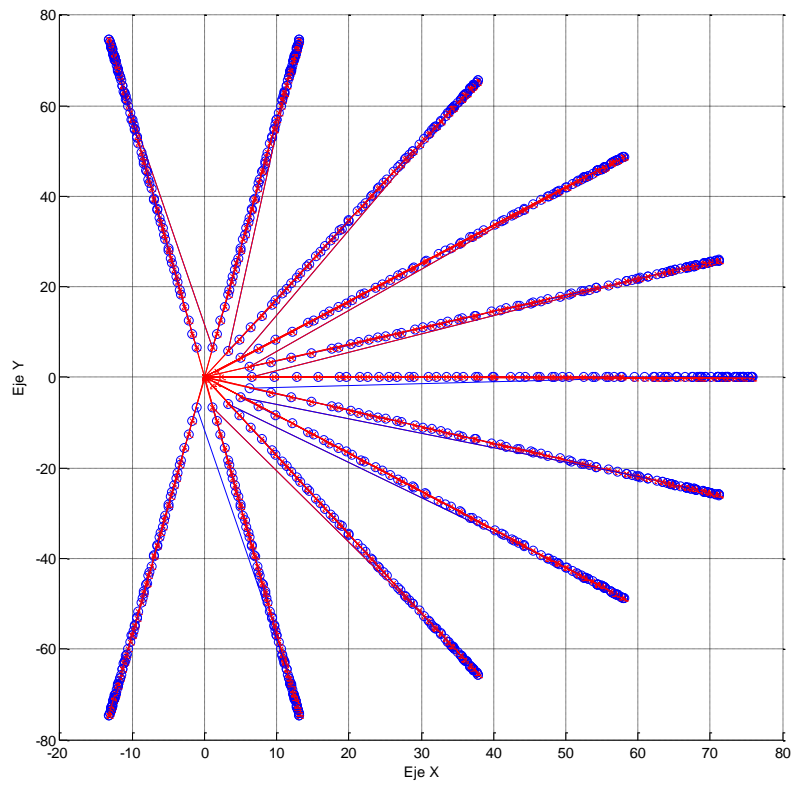
3.1 Manipulador:

En el caso del manipulador, se grafica el espacio de trabajo del mismo a través de su cinemática directa, pero afectado por la transformación de coordenadas que representa a la plataforma móvil. En definitiva, lo que se está graficando es el espacio de trabajo del robot completo para un set de valores dados de posición y orientación de la plataforma. Se solicitan dichos valores por teclado.

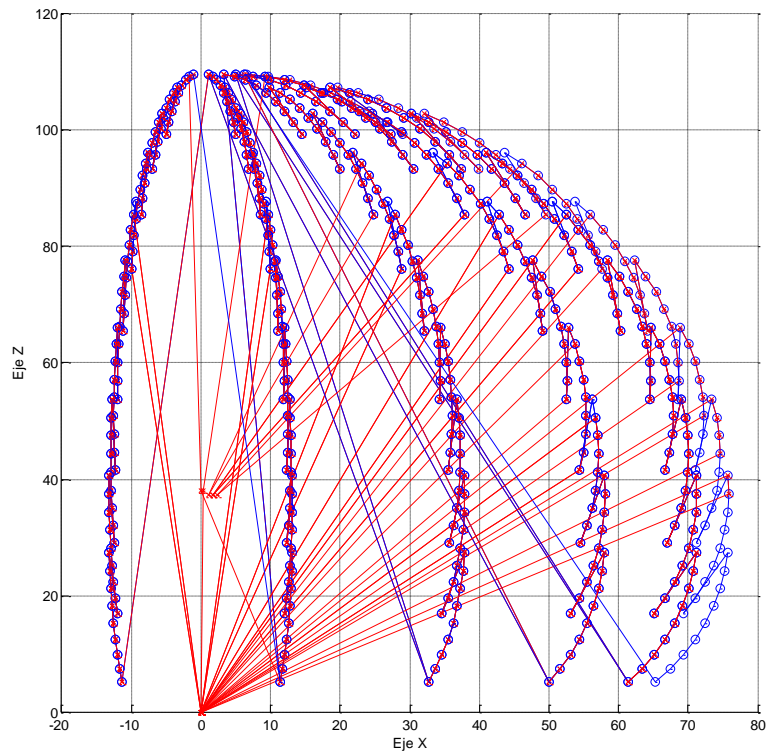
Dentro del mismo ciclo de iteración, se ingresan los puntos del espacio de trabajo en la cinemática inversa, previamente transformados a coordenadas locales. Luego se utilizan las coordenadas articulares obtenidas para, mediante la cinemática directa, generar un segundo espacio de puntos para comparar visualmente con el espacio de trabajo.



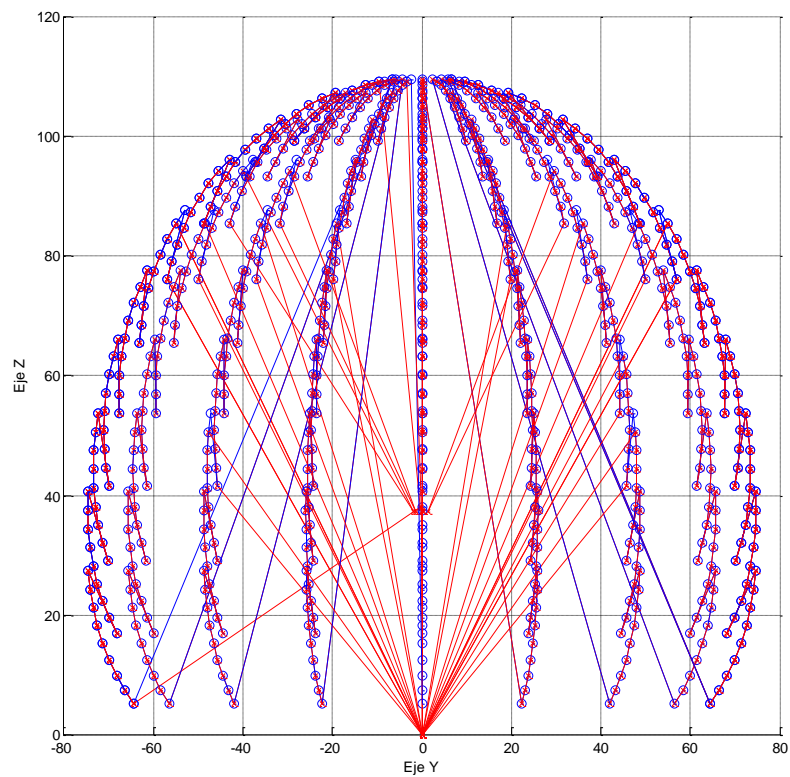
Espacio de trabajo - Perspectiva



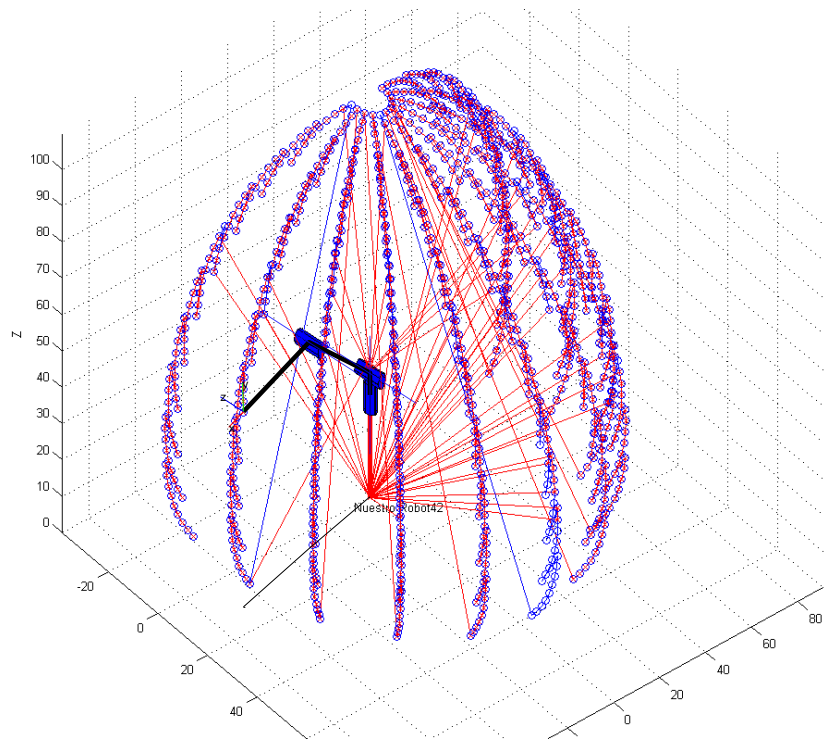
Espacio de trabajo – Plano XY



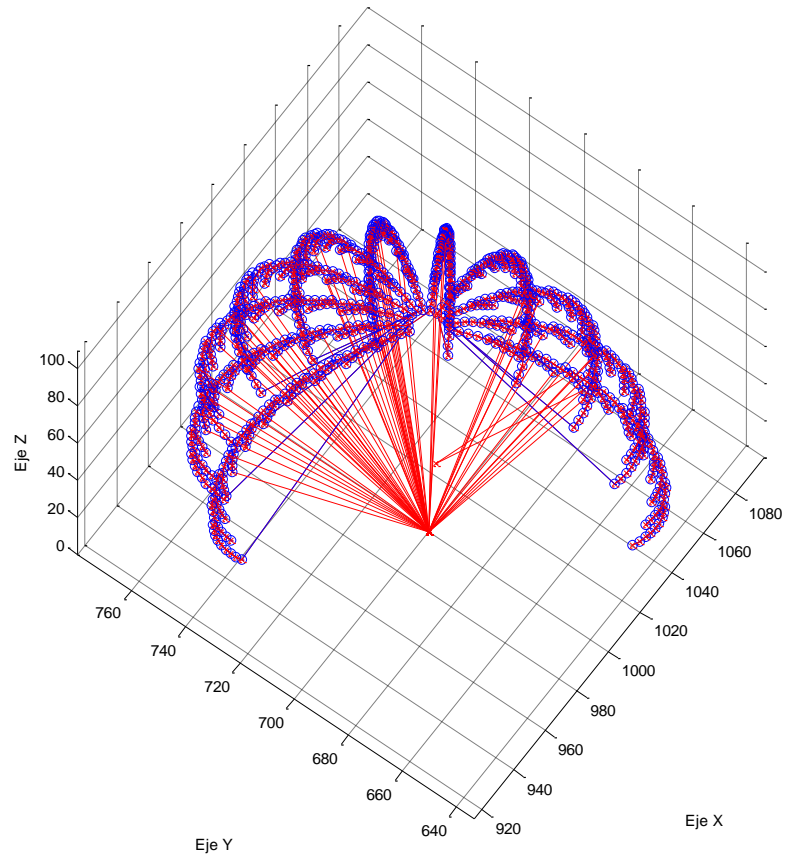
Espacio de trabajo – Plano XZ



Espacio de trabajo – Plano YZ



Espacio de trabajo con el Manipulador



Espacio de trabajo para $X=1000\text{cm}$, $Y=700\text{cm}$, $\phi=45^\circ$

Se utilizó el Toolbox de Peter Corke para generar la animación del manipulador {4}

Respecto de las coordenadas articulares utilizadas, se expresa en la sgte tabla los límites establecidos para las mismas y el incremento de ángulo para cada ciclo de iteración:

Articulación	θ_{MIN}	θ_{MAX}	$\Delta\theta$
1	-100°	100°	20°
2	-5°	90°	10°
3	-40°	0°	5°

3.2 Plataforma Móvil:

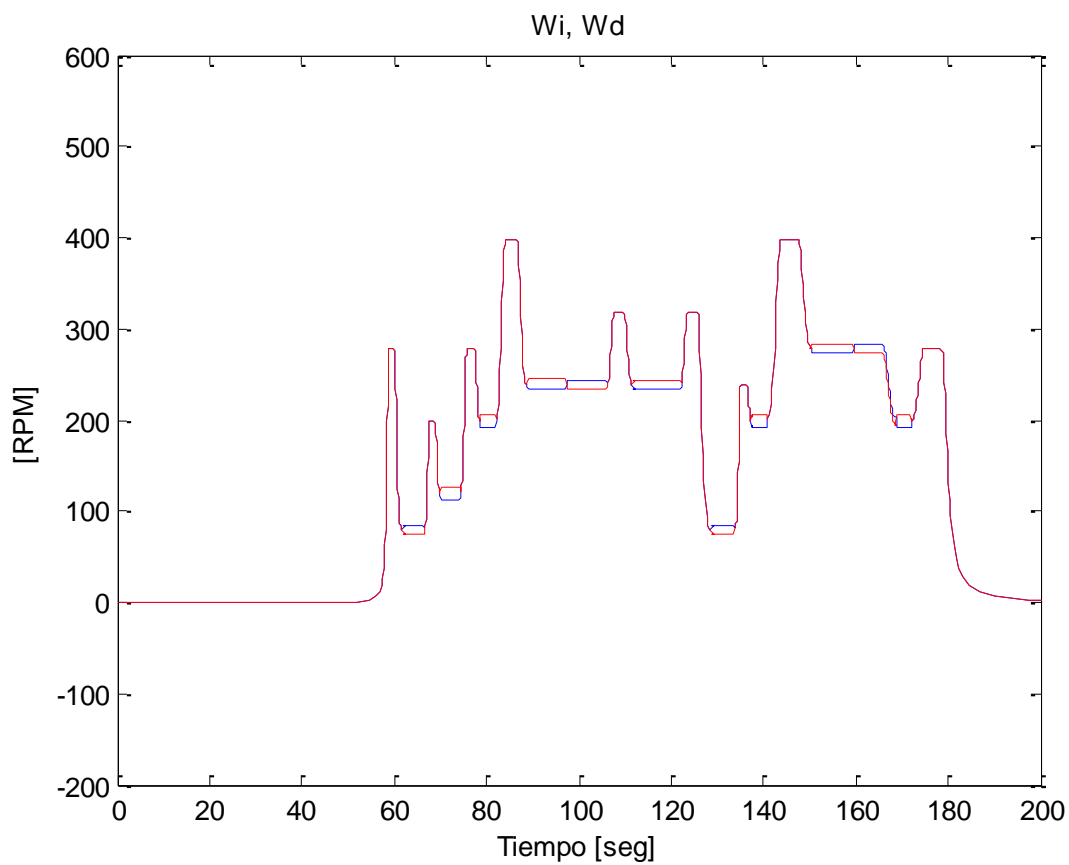
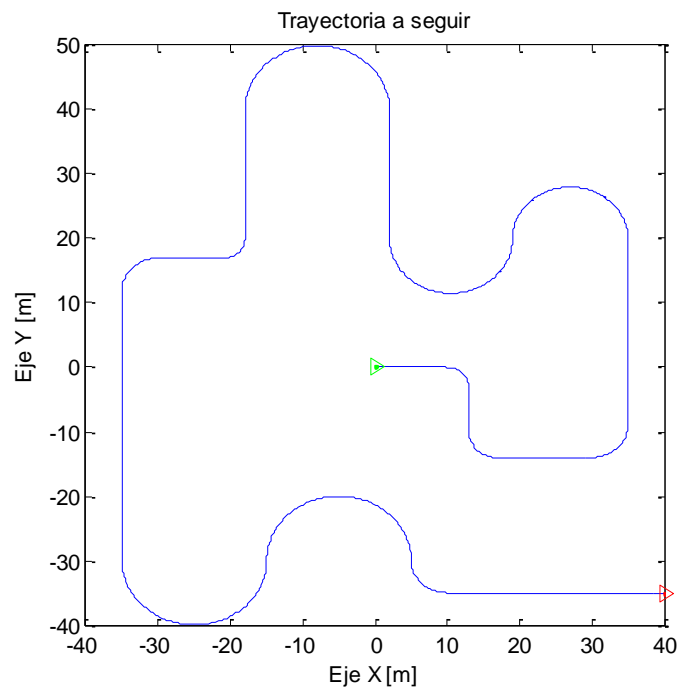
En el caso de la plataforma móvil el concepto de espacio de trabajo no es aplicable de la misma manera que en los manipuladores. Podría decirse que en principio las variables articulares Θ_i , Θ_d no están restringidas en un determinado rango, como en el caso anterior. La única restricción podría ser la autonomía de la batería.

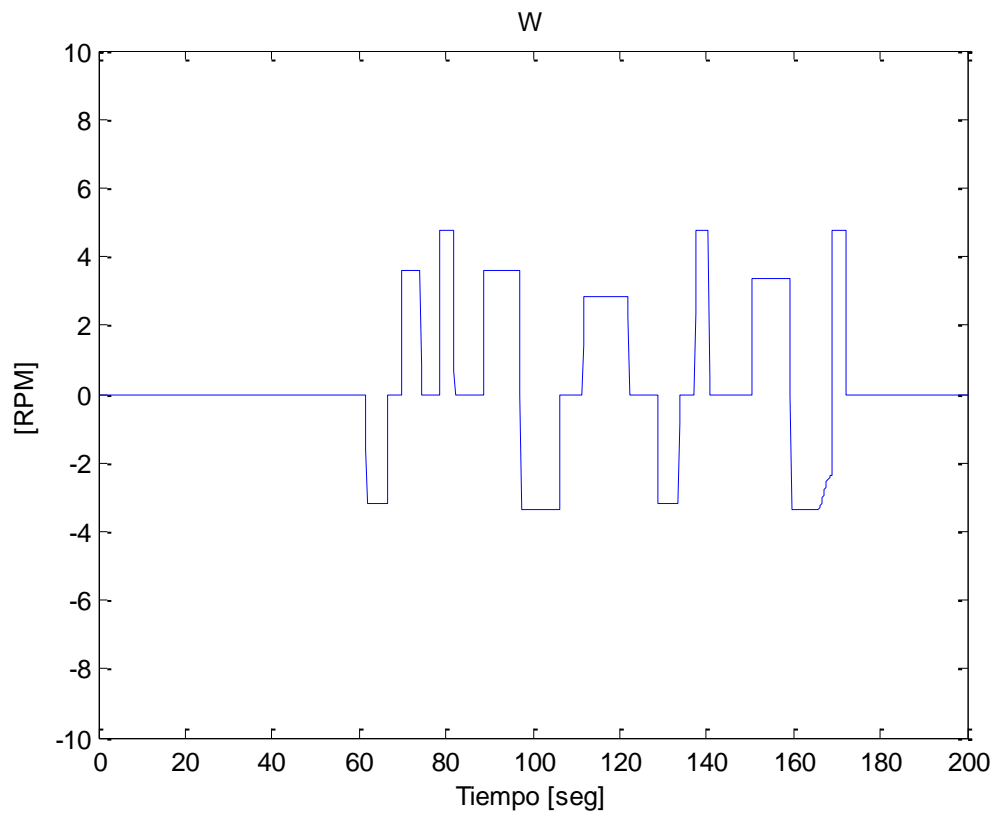
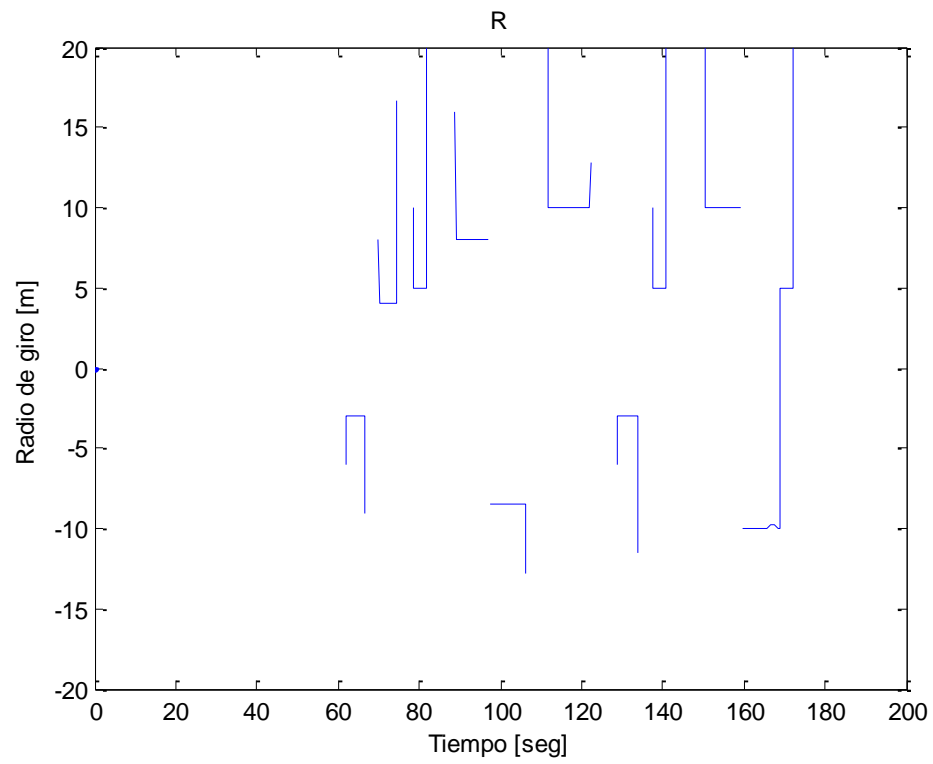
En lugar de eso se optó por generar una trayectoria o camino a seguir por el robot, para el cual se definen las coordenadas (x, y) en el plano de cada punto que lo conforma. A su vez se define una velocidad instantánea para cada punto de la trayectoria. A partir de la distancia entre cada punto y el sucesivo se determina un salto temporal “dt” que permite derivar o integrar las variables de manera discreta. Para generar dicha trayectoria se creó un script adicional, el cual no formaría parte de la implementación en el DSP.

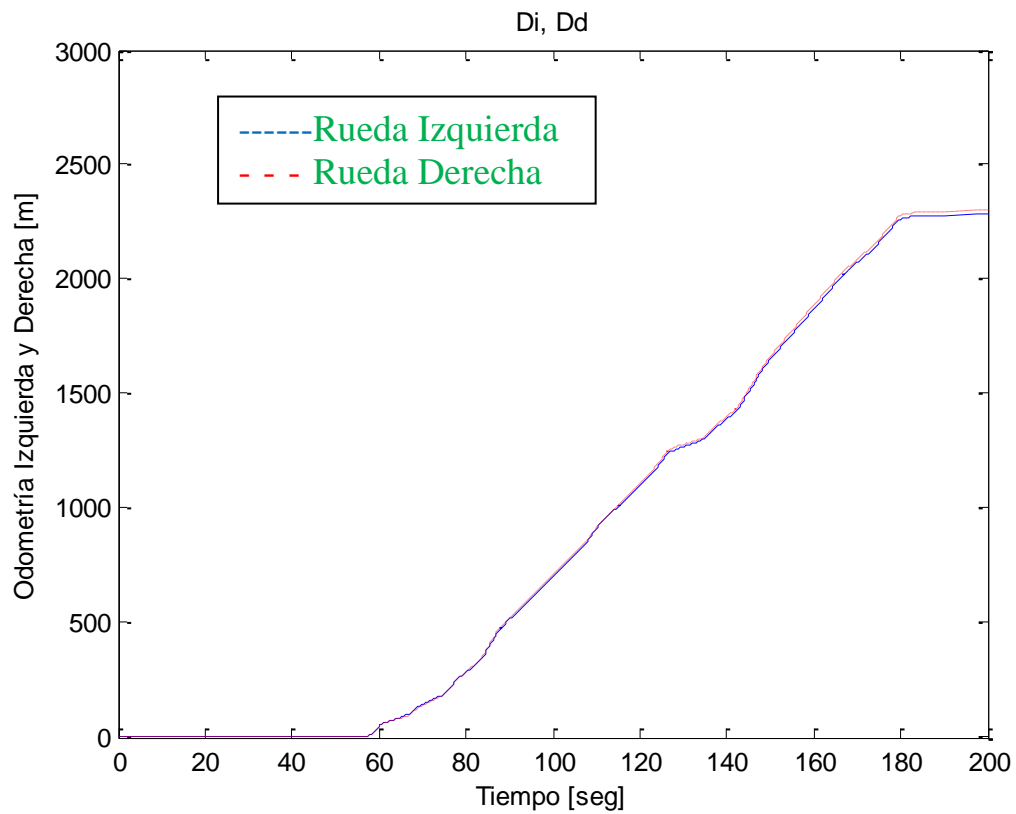
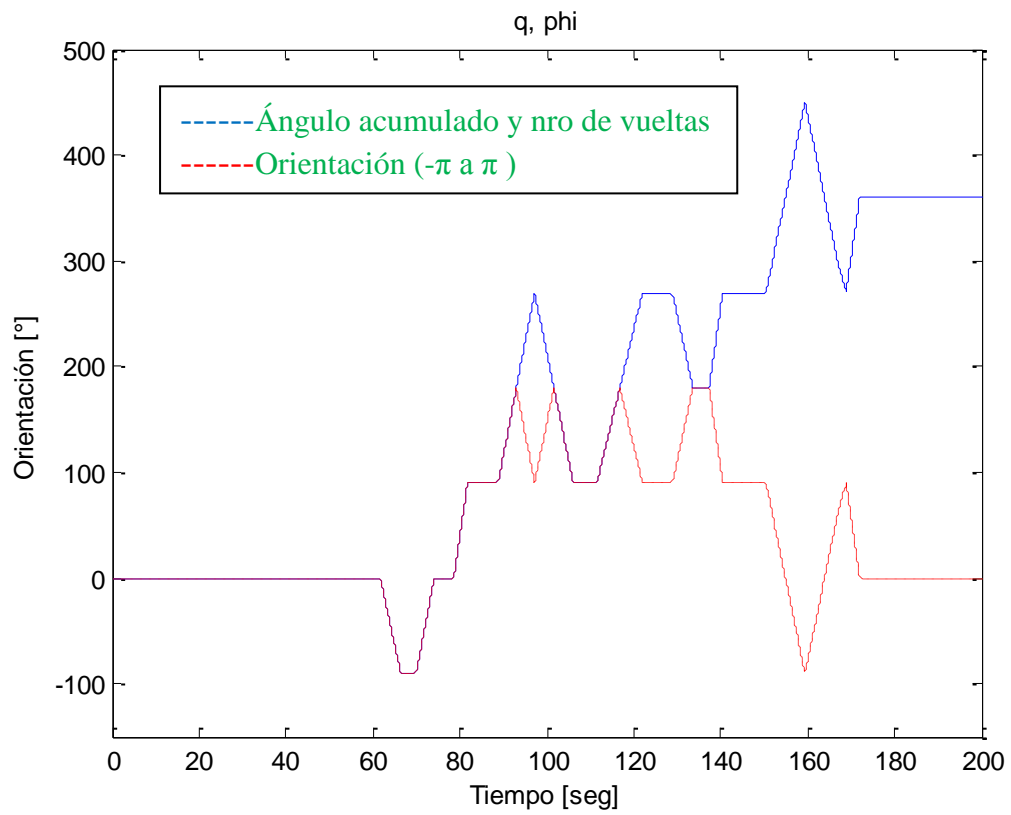
En este caso el proceso consiste en barrer punto a punto un vector que representa la trayectoria y, mediante el modelo jacobiano inverso, obtener las variables articulares. Dichas variables ingresan en el modelo jacobiano directo, cuya salida se compara gráficamente con la trayectoria en seguimiento. Básicamente el vector de camino a seguir representa el setpoint de un sistema de control de velocidad. Integrando la velocidad de giro de las ruedas se obtiene la odometría del robot, la cual podría utilizarse para determinar la posición del mismo en el plano. En el robot real esta integración se realizaría mediante encoders asociados a dichas ruedas.

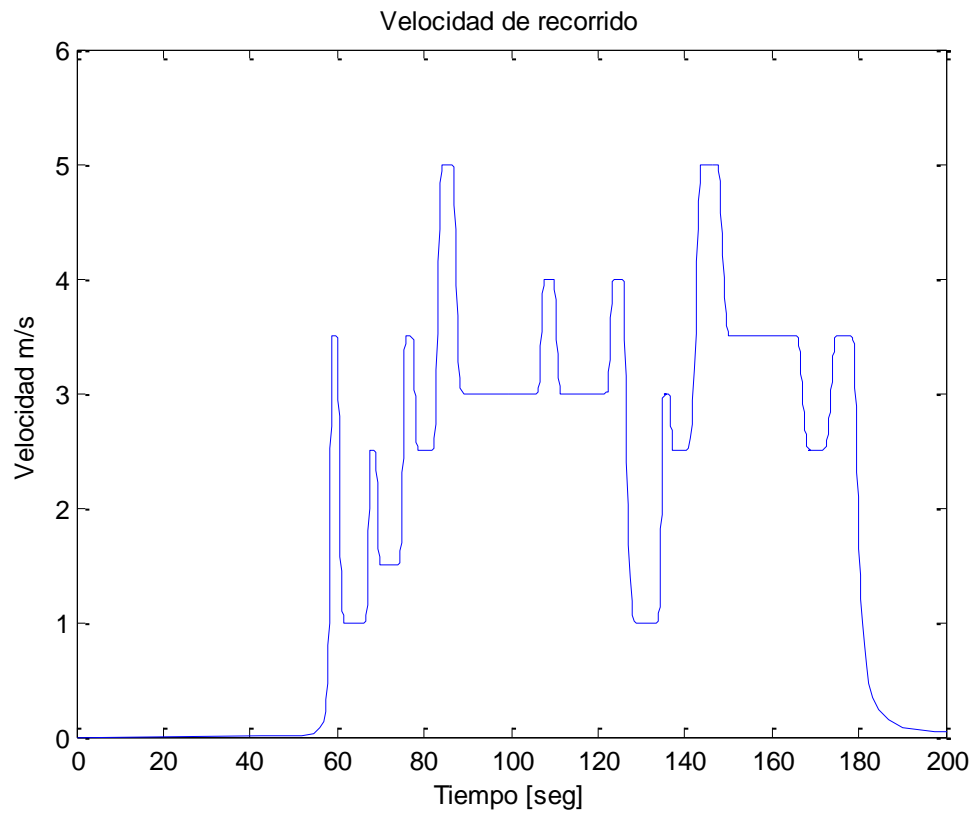
Como resultado del algoritmo implementado se obtiene un juego de variables en función del tiempo que representan completamente el movimiento del robot, dentro de la validez del modelo cinemático desarrollado. Estas variables serán utilizadas en las capas superiores de software para el sistema de control.

El método odométrico sólo es útil para determinar la posición y orientación del robot para intervalos de tiempo reducidos, pues los errores de posicionamiento acumulados en el tiempo lo vuelven inutilizable para trayectos largos. Se hace necesario complementar el método odométrico con otro tipo de sistemas y sensores que sirvan para determinar la posición y orientación del robot, como ser GPS, sensores de orientación basados en acelerómetros o en las líneas de campo magnético terrestre. Se utilizó la herramienta de Corke para algunas funciones.

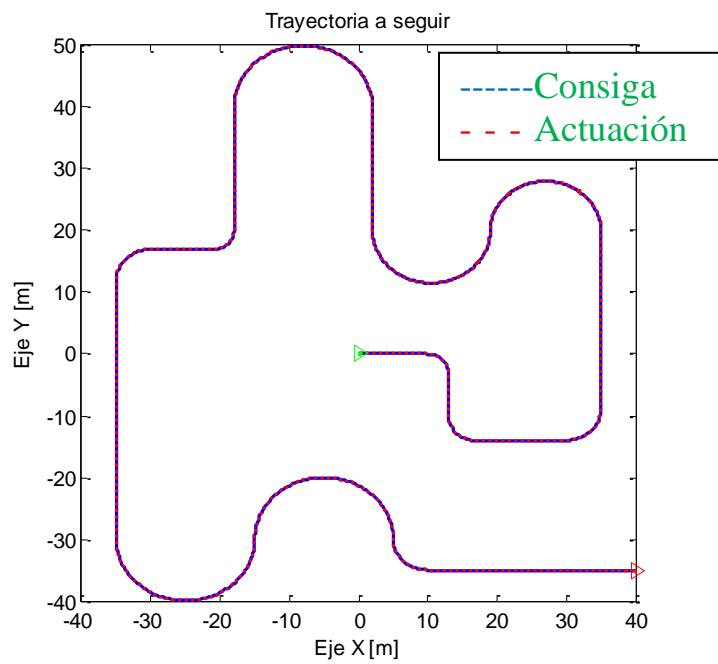


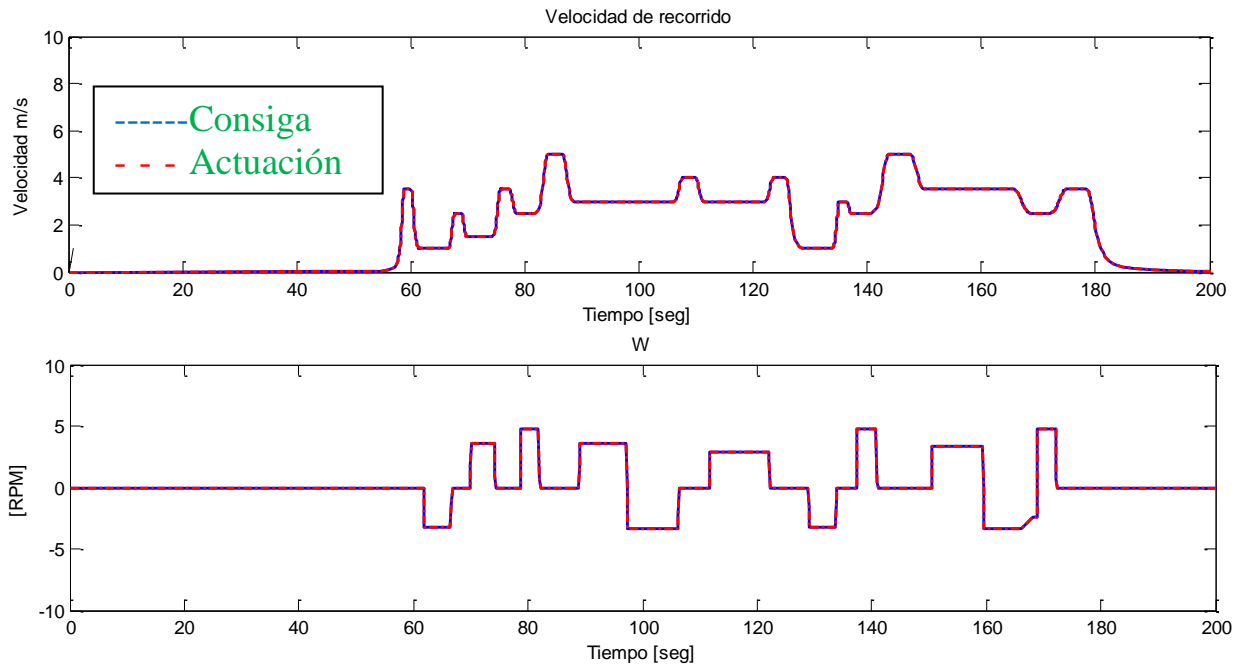






A continuación se grafica la comparación entre la consigna y la salida del modelo cinemático directo:





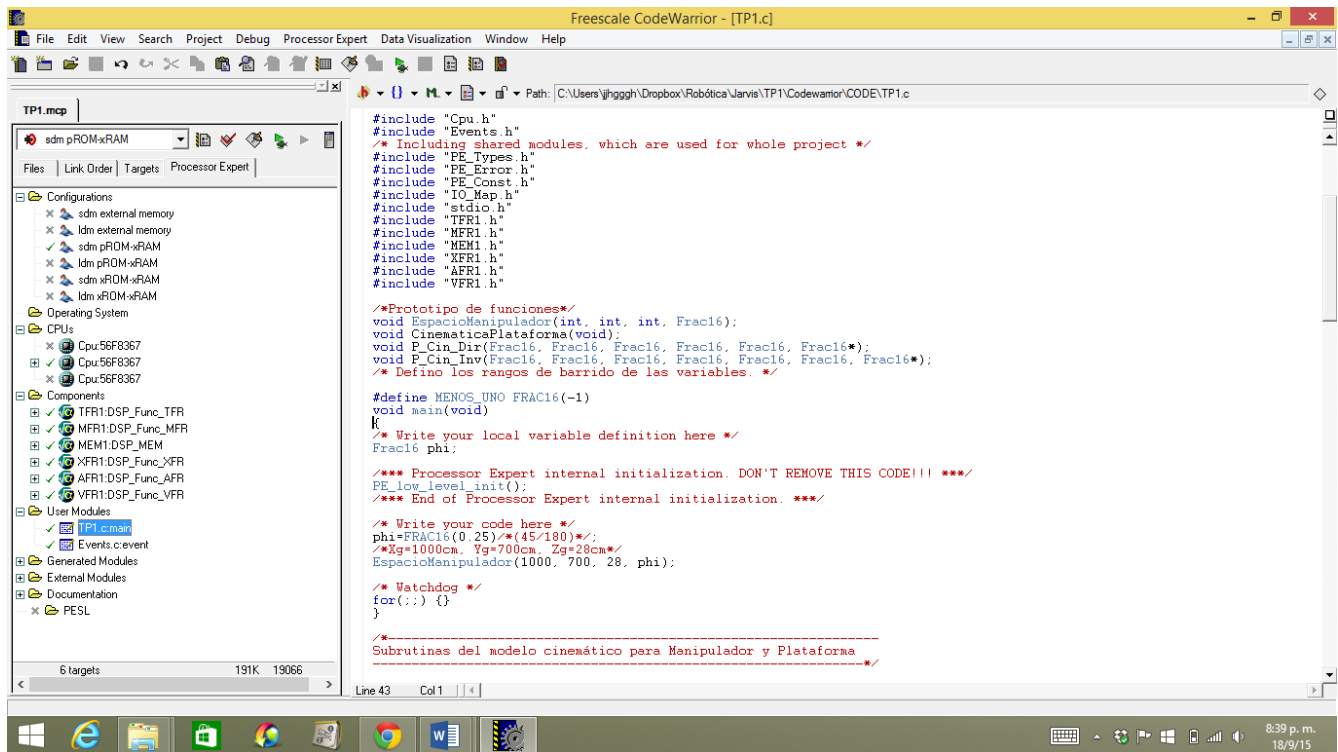
Se observa que las curvas se superponen totalmente, verificando la validez de ambos modelos.

4. Implementación del modelo cinemático en DSP

Obtenido el modelo cinemático anterior, se procede a implementarlo en el elemento que luego controlará al robot. Se observa que la tarea de controlar un manipulador requiere de un gran procesamiento matemático y además que ese procesamiento sea muy veloz, es por ello que generalmente se utilizan DSP para tal tarea. En el caso de un robot móvil con ruedas lo que se implementaría en el DSP serían las funciones que representan a la matriz jacobiana directa e inversa, y además todos los cálculos en tiempo real de velocidades, aceleraciones y posición derivados del seguimiento de la trayectoria.

Se considerò el uso del procesador 56F8367 de la familia 56800E de Freescale. El fabricante provee el software Codewarrior para realizar el programa y debugearlo.

A continuación se muestra una imagen de cómo se ve el entorno de desarrollo del CodeWarrior.



4.1 Programa en Codewarrior

Se presenta a continuación el código en C que representa la matriz cinemática directa del brazo manipulador que posee el robot. Dicho código es sólo una pequeña parte del necesario para implementar todo el robot. Sólo cubre la parte que se ocupa de determinar el espacio de trabajo del manipulador.

```

/** #####
** Filename : TPN1.C
** Project : TPN1
** Processor : 56F8367
** Version : Driver 01.14
** Compiler : Metrowerks DSP C Compiler
** Date/Time : 10/06/2012, 08:06 p.m.
** Abstract :
** Main module.
** This module contains user's application code.
** Settings :
** Contents :
** No public methods
**
** #####*/
/* MODULE TPN1 */
/* Including needed modules to compile this module/procedure */
#include "Cpu.h"

```

```

#include "Events.h"
/* Including shared modules, which are used for whole project */
#include "PE_Types.h"
#include "PE_Error.h"
#include "PE_Const.h"
#include "IO_Map.h"
#include "stdio.h"
#include "TFR1.h"
#include "MFR1.h"
#include "MEM1.h"
#include "XFR1.h"
#include "AFR1.h"
#include "VFR1.h"

/*Prototipo de funciones*/
void EspacioManipulador(int, int, int, Frac16);

/* Defino los rangos de barrido de las variables. */

#define MENOS_UNO FRAC16(-1)
void main(void)
{
/* Write your local variable definition here */
Frac16 phi;

/**/ Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ***/
PE_low_level_init();
/**/ End of Processor Expert internal initialization. ***/

/* Write your code here */
phi=FRAC16(0.25)/(45/180)*/;
/*Xg=1000cm, Yg=700cm, Zg=28cm*/
EspacioManipulador(1000, 700, 28, phi);

/* Watchdog */
for(;;) { }
}

/*-----
Subrutinas del modelo cinemático
-----*/

void EspacioManipulador(int Xg, int Yg, int Zg, Frac16 q0)
{
#define Q1_INI FRAC16(-0.5555555)//-100/180
#define Q1_STEP FRAC16(0.1111111)//20/180
#define Q1_FIN FRAC16(0.5555555)//100/180
#define Q2_INI FRAC16(-0.0277777777)//(-5/180)

```

```

#define Q2_STEP FRAC16(0.02777777777)//10/180
#define Q2_FIN FRAC16(0.4722222)//85/180
#define Q3_INI FRAC16(-0.2222222)//-40/180
#define Q3_STEP FRAC16(0.02777777777)//5/180
#define Q3_FIN FRAC16(0)
/* Defino los elementos fijos del ROBOT */
Frac16 l1=FRAC16(0.06)/(6/100)*;
Frac16 l2=FRAC16(0.4)/(40/100)*;
Frac16 l3=FRAC16(0.36)/(36/100)*;

/* Write your local variable definition here */
Frac16 q1, q2, q3;
Frac16 M_0A1[4][4] = {
{0*cos(q1)/,FRAC16(0),0*sin(q1)/,FRAC16(0)},
{0*sin(q1)/,FRAC16(0),0*-cos(q1)/,FRAC16(0)},
{FRAC16(0),FRAC16(1),FRAC16(0),0*l1*},
{FRAC16(0),FRAC16(0),FRAC16(0),FRAC16(1)}
};

Frac16 M_1A2[4][4] = {
{0*cos(q2)/,0*-sin(q2)/,FRAC16(0),0*l2*cos(q2)/},
{0*sin(q2)/,0*cos(q2)/,FRAC16(0),0*l2*sin(q2)/},
{FRAC16(0),FRAC16(0),FRAC16(1),FRAC16(0)},
{FRAC16(0),FRAC16(0),FRAC16(0),FRAC16(1)}
};

Frac16 M_2A3[4][4] = {
{0*cos(q3)/,0*-sin(q3)/,FRAC16(0),0*l3*cos(q3)/},
{0*sin(q3)/,0*cos(q3)/,FRAC16(0),0*l3*sin(q3)/},
{FRAC16(0),FRAC16(0),FRAC16(1),FRAC16(0)},
{FRAC16(0),FRAC16(0),FRAC16(0),FRAC16(1)}
};

Frac16 ROT_G[4][4] = {
{0*cos(q0)/,0*-sin(q0)/,FRAC16(0),FRAC16(0)},
{0*sin(q0)/,0*cos(q0)/,FRAC16(0),FRAC16(0)},
{FRAC16(0),FRAC16(0),FRAC16(1),FRAC16(0)},
{FRAC16(0),FRAC16(0),FRAC16(0),FRAC16(1)}
};

Frac16 M_0A2[4][4],M_0A3[4][4], H[4][4];
Frac16 senq0, senq1, senq2, senq3, cosq0, cosq1, cosq2, cosq3;

long x, y, z;

senq0 = tfr16SinPIx(q0);

```

```

cosq0 = tfr16CosPIx(q0);
ROT_G[0][0] = cosq0;
ROT_G[0][1] = mult(MENOS_UNO,senq0);
ROT_G[1][0] = senq0;
ROT_G[1][1] = cosq0;
printf("Puntos del Manipulador\n");

/* Código propio para calcular los límites del espacio de trabajo. */
for(q1=Q1_INI;q1<=Q1_FIN;q1+=Q1_STEP) {
senq1 = tfr16SinPIx(q1);
cosq1 = tfr16CosPIx(q1);
M_0A1[0][0] = cosq1;
M_0A1[0][2] = senq1;
M_0A1[1][0] = senq1;
M_0A1[1][2] = mult(MENOS_UNO,cosq1);
M_0A1[2][4] = l1;

for(q2=Q2_INI;q2<=Q2_FIN;q2+=Q2_STEP) {
senq2 = tfr16SinPIx(q2);
cosq2 = tfr16CosPIx(q2);
M_1A2[0][0] = cosq2;
M_1A2[0][1] = mult(MENOS_UNO,senq2);
M_1A2[0][3] = mult(l2,cosq2);
M_1A2[1][0] = senq2;
M_1A2[1][1] = cosq2;
M_1A2[1][3] = mult(l2,senq2);

for(q3=Q3_INI;q3<=Q3_FIN;q3+=Q3_STEP) {
senq3 = tfr16SinPIx(q3);
cosq3 = tfr16CosPIx(q3);
M_2A3[0][0] = cosq3;
M_2A3[0][1] = mult(MENOS_UNO,senq3);
M_2A3[0][3] = mult(l3,cosq3);
M_2A3[1][0] = senq3;
M_2A3[1][1] = cosq3;
M_2A3[1][3] = mult(l3,senq3);

xfr16Mult((Frac16 *)M_0A1,4,4,(Frac16 *)M_1A2,4,(Frac16 *)M_0A2);
xfr16Mult((Frac16 *)M_0A2,4,4,(Frac16 *)M_2A3,4,(Frac16 *)M_0A3);
xfr16Mult((Frac16 *)ROT_G,4,4,(Frac16 *)M_0A3,4,(Frac16 *)H);

x=(long) H[0][3];
y=(long) H[1][3];
z=(long) H[2][3];
x=(long) x*100/32768;
y=(long) y*100/32768;
z=(long) z*100/32768;

```

```

printf("%ld %ld %ld ;\n", x + Xg, y + Yg, z + Zg);
}

}
}
}

/* END TPN1 */
/*
** #####
**
** This file was created by Processor Expert 3.00 [04.35]
** for the Freescale 56800 series of microcontrollers.
**
** #####
**/

```

Básicamente se adaptó parte del algoritmo realizado en Matlab para poder implementarlo en el DSP. Dicho dispositivo utiliza un tipo de datos fraccionario de punto fijo normalizado.

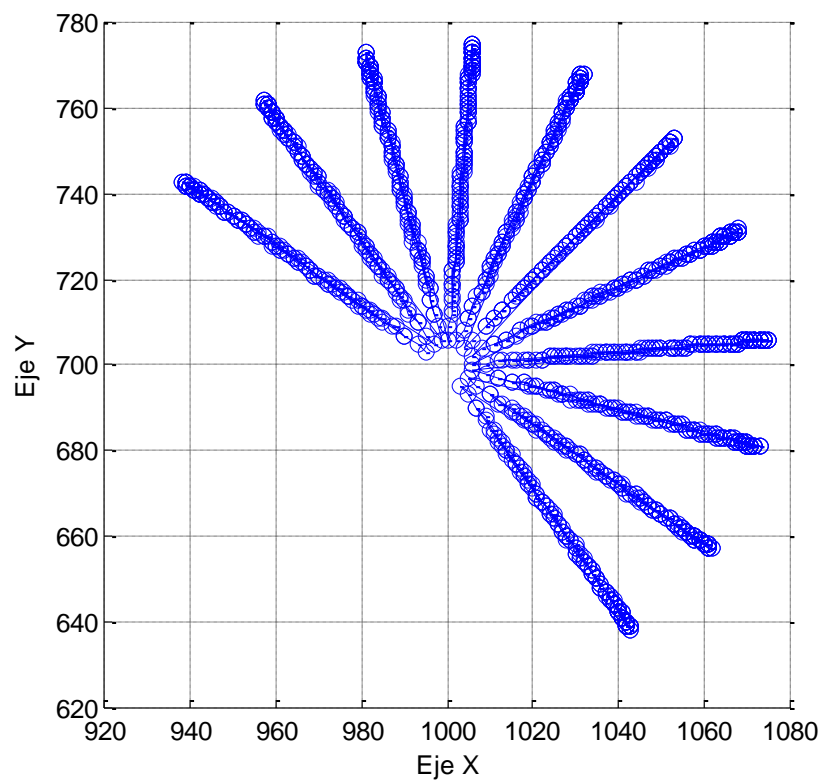
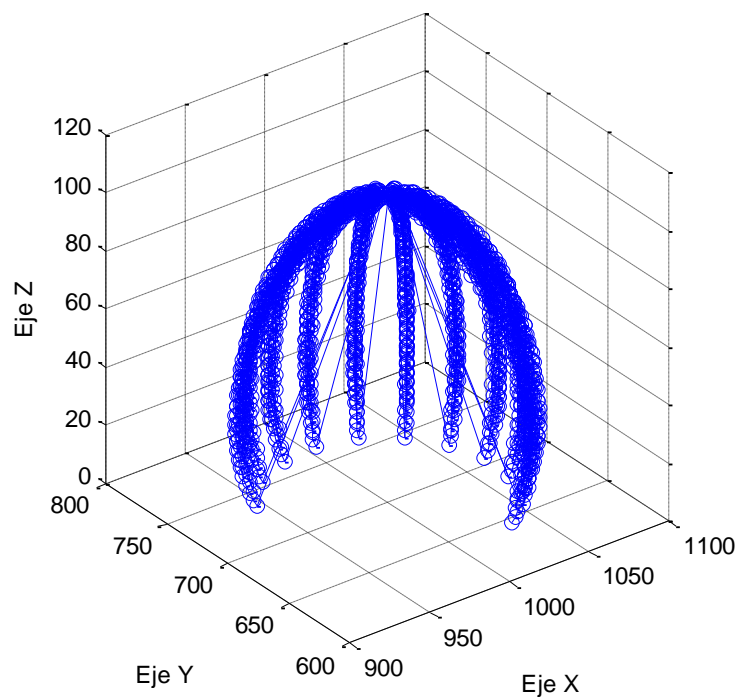
Frac16 Representa **-1 a $(1 - 2^{-15})$**

Frac32 Representa **-1 a $(1 - 2^{-31})$**

Adaptando todas las variables a este tipo de datos permite utilizar las funciones nativas del DSP, como ser suma, resta, multiplicación y producto de matrices, las cuales se resuelven directamente por hardware. Dicho hardware es la razón de ser del DSP y es la principal ventaja que posee este dispositivo respecto a un microcontrolador convencional, en lo que a matemática se refiere. Sin embargo, el principal problema a resolver es cómo normalizar y desnormalizar las variables para poder aprovechar estas funciones. Se utilizaron variables en Frac16.

En el caso de los ángulos de las articulaciones, las propias funciones trigonométricas ya establecen que la escala para el Frac16 es $-\pi$ a π . Las dimensiones de los segmentos se normalizan con respecto a las dimensiones máximas del espacio de trabajo observadas en la simulación en Matlab.

4.2 Espacio de trabajo generado por el DSP



5. Conclusiones

- **En líneas generales se observa que el presente desarrollo dio los resultados esperados dentro de las restricciones propuestas. Los desarrollos matemáticos pudieron verificarse totalmente al menos en la simulación de Matlab, y parcialmente en la simulación del código escrito en Codewarrior.**
- En la simulación de Matlab del manipulador se observó que, al comparar los puntos del espacio de trabajo con los puntos generados a partir de intercalar las funciones de cinemática inversa, alrededor de un 5% de estos puntos no coincidían. El resto de los puntos coincidían perfectamente, por lo que se atribuye la diferencia a un problema de precisión matemática del Matlab en las funciones trigonométricas y los cálculos realizados en el modelo cinemático inverso. Se asumió que si el modelo cinemático inverso fuera incorrecto la cantidad de errores sería más elevada. Lamentablemente no se pudo solucionar. No fue objeto del presente trabajo verificar los cálculos del Matlab utilizando el Codewarrior.
- Durante la etapa de desarrollo de los script de Matlab para la plataforma móvil, se determinó que no puede haber variaciones bruscas de pendiente en la recta tangente a cada punto de la trayectoria programada, ya que aparecían picos en las señales de velocidad generadas, ya que se calculan derivando señales de posición. Este problema sólo se evidenciaba en las transiciones entre un tramo recto oblicuo de la trayectoria y una curva. Para este trabajo se optó entonces por generar una trayectoria donde todos los tramos rectos son horizontales o verticales, con curvas circulares que empiezan y terminan en orientaciones múltiplo de $\pi/2$. En etapas posteriores del desarrollo del robot se deberá tener en cuenta este problema y evitar puntos angulosos en las trayectorias generadas por el sistema de control.
- Todos los algoritmos generados para la implementación se realizaron utilizando el Matlab, por su gran flexibilidad y poder de cálculo, a sabiendas que su verificación no requiere que los cálculos sean en tiempo real. Sin embargo, al querer adaptar esos algoritmos al Codewarrior, el principal problema fue la naturaleza del tipo de datos Frac16. Si bien está fuera del alcance de este trabajo, si se quiere implementar todas las funciones desarrolladas en el DSP, el principal desafío será el correcto manejo de las normalizaciones y desnormalizaciones requeridas, para lograrlo. Como pudo observarse, en el caso del algoritmo del manipulador las funciones necesarias están disponibles en el entorno de desarrollo y los cambios de escala se realizaron al principio, en la declaración de variables, y al final en la “impresión” del espacio de trabajo. En el caso de la plataforma el trabajo sería más arduo, ya que a diferencia del primer caso es necesario realizar divisiones entre variables Frac16, que exigen que el dividendo sea siempre menor en módulo que el divisor.

6. Referencias:

1. Barrientos, A, Peñin, L, Balaguer, C y Aracil, R. (2001). *Fundamentos de robótica*, Mc Graw Hill
2. Baturone, AO. (2001). *Robótica: manipuladores y robots móviles*. Marcombo
3. Burgard W, Stachniss C, Bennewitz M, Arras K. *Introduction to mobile robotics. Wheeled locomotion*
Disponible en
<http://ais.informatik.uni-freiburg.de/teaching/ss13/robotics/slides/03-locomotion.pdf>
4. Corke, PI (2002). *Robotics Toolbobox for Matlab*
5. Jiménez Cuesta J. (2012) *Simulación de vehículos eléctricos ligeros*. Valencia: Universidad Politécnica
6. Ramírez Arias, JL y Rubiano Fonseca, A. (2012). *Modelamiento matemático de la cinemática directa e inversa de un robot manipulador de tres grados de libertad*. Revista Ingeniería solidaria, vol. 8, no. 15, p. 46-52
7. Rocheleau, D. EMC Kinematics.
Disponible en:
<http://www.me.sc.edu/fs/rochelea/EMCH-332/handout04.pdf>
8. *Sistemas de locomoción de robots móviles*
Disponible en:
http://www.esi2.us.es/~vivas/ayr2iaei/LOC_MOV.pdf