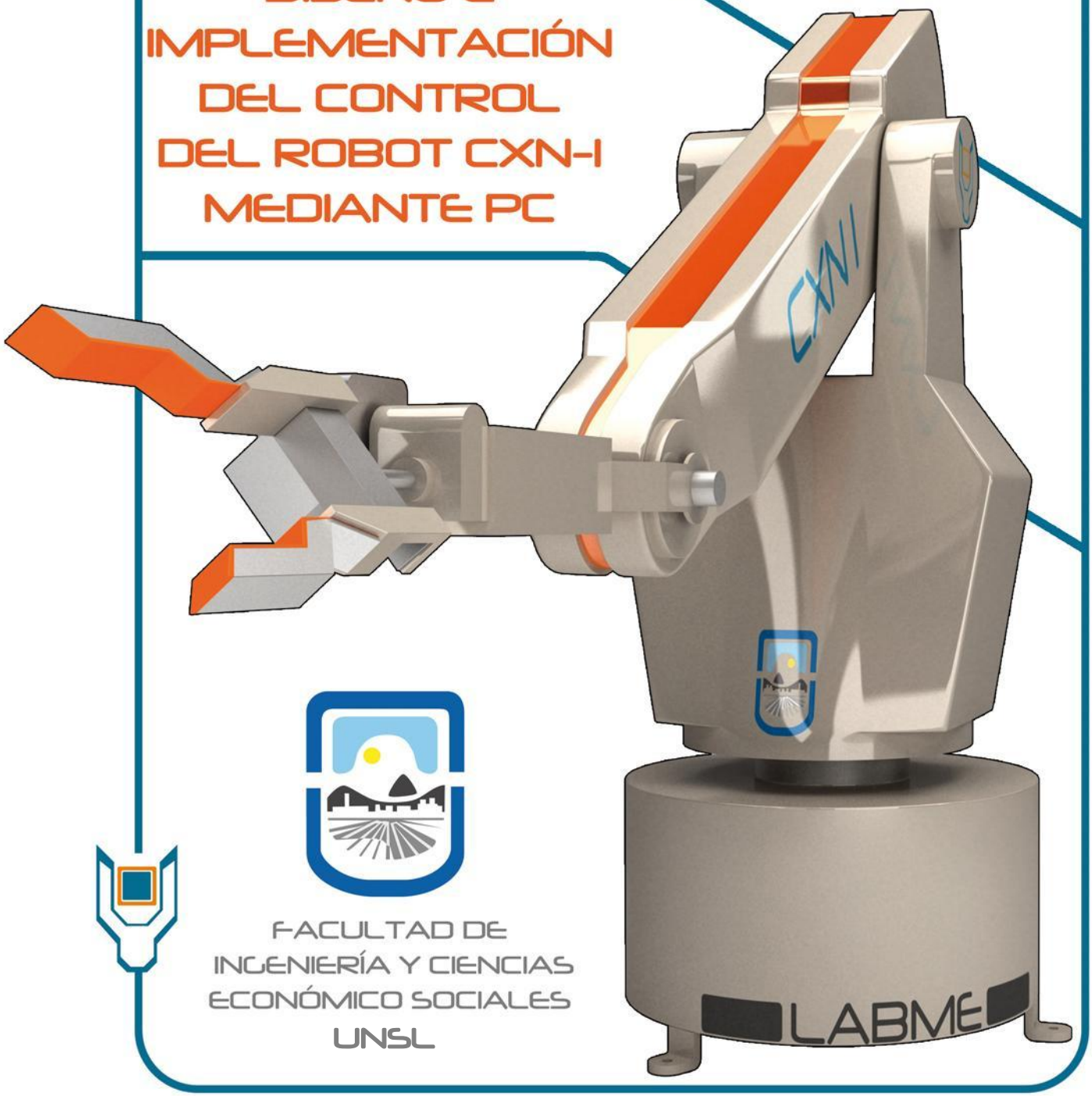
CENTRO ARGENTINO DE INGENIEROS

Premio PRE INGENIERÍA 2011

FEDERICO GERMÁN KÜNNING

MG. ING. OSCAR DANIEL MORÁN

DISEÑO E IMPLEMENTACIÓN DEL CONTROL DEL ROBOT CXN-I MEDIANTE PC



FACULTAD DE INGENIERÍA Y CIENCIAS ECONÓMICO SOCIALES
UNSL

VILLA MERCEDES - SAN LUIS - ARGENTINA - 2011

Universidad Nacional de San Luis

Facultad de Ingeniería y Ciencias Económico Sociales

Laboratorio de Mecatrónica

"DISEÑO E IMPLEMENTACIÓN DEL CONTROL DEL ROBOT CXN-I MEDIANTE PC"

TRABAJO FINAL DE CARRERA

Alumno:

Federico Germán KÜNNING

Director:

Mg. Ing. Oscar Daniel MORÁN

Carrera:

Ingeniería Electromecánica

Villa Mercedes - San Luis - Argentina

- 2011 -

Agradecimientos

A mis padres, quienes a través del ejemplo no sólo me inculcaron la cultura del estudio y del esfuerzo, sino también a ser una persona de moral y principios, y quienes fueron un apoyo incondicional y una fuente de aliento constante.

A toda mi familia, abuelos, hermanos, tíos y primos, quienes siempre se preocuparon por mí transcurrir en la vida académica y me brindaron su afectuoso aliento.

A Valeria, una de las fuentes de apoyo y aliento más importantes, y quien me dio fuerzas en los momentos de cansancio.

A mis amigos y compañeros, quienes condimentaron e hicieron más alegre este transcurrir, y quienes fueron una gran fuente de ayuda.

Al Mg. Ing. Daniel Morán y al Ing. José Cuello, quienes me brindaron su confianza y apoyo y me permitieron crecer en el mundo de la investigación y la docencia, lo que me brindó experiencias invaluables, y a quienes considero buenos amigos.

A los integrantes del LABME (Laboratorio de Mecatrónica de la Facultad de Ingeniería y Ciencias Económico-Sociales de la UNSL), quienes posibilitaron la construcción y desarrollo del robot CXN-I.

A todos, mi más profundo agradecimiento.

Germán

Índice de contenidos

1. Introducción

La robótica en general, y los robots industriales[1] en particular, han tomado una gran importancia debido a que el proceso de globalización ha obligado a las industrias a ser cada vez más competitivas y flexibles, a mejorar la calidad de sus productos y a reducir los costos y tiempos de producción, a mejorar las condiciones de trabajo de los empleados evitando que estos realicen tareas de riesgo, etc., requisitos para los cuales la robótica juega un papel primordial, especialmente gracias al avance de esta tecnología y a la reducción de los costos de instalación.

Los robots son empleados en todo tipo de tareas, pero sobre todo, en aquellas destinadas a la producción en serie, donde principalmente se utilizan robots industriales debido a las ventajas que poseen en cuanto a velocidad, precisión, repetibilidad, flexibilidad, reducción de costos, etc. Es por esto que son cada vez más utilizados en la industria, especialmente en los países más desarrollados, motivo por lo cual la investigación y desarrollo en estas tecnologías representa un avance muy importante en el progreso industrial de un país.

El control de robots es una de las áreas más importantes de la robótica debido a que en ella radica la inteligencia de los mismos, que constituye una de sus características fundamentales y sin la cual estos serían otra máquina herramienta más. Esta es un área de constante avance en robótica debido al permanente desarrollo de las tecnologías de hardware y software, el cual permite una mejoría continua de las capacidades de los robots.

El sistema de control de un robot es el que determina en gran medida lo que puede hacer y las tareas para las cual es más adecuado. De este sistema dependen, por ejemplo, el tipo de trayectorias que puede realizar, sus prestaciones dinámicas, su modo y posibilidades de programación, sus capacidades de comunicación, etc., haciéndolo más o menos apropiado para ciertas aplicaciones. En la actualidad, los robots industriales poseen diferentes tipos de control según la tarea que deban realizar, los cuales incluyen desde control punto a punto, control de trayectorias continuas, control adaptativo, etc.

Dentro de la robótica otra área en continuo desarrollo es el modo y las posibilidades de interacción del robot con el usuario y con el entorno. Actualmente los robots industriales realizan sus tareas siguiendo un programa establecido por el usuario y pueden ser programados mediante diferentes métodos, clasificándose en programación por guiado y programación textual, dependiendo del sistema empleado para indicar la secuencia de las acciones a realizar; pudiendo algunos robots conjugar ambos métodos.

[1] Robot manipulador industrial (ISO): Manipulador de 3 o más ejes, con control automático, reprogramable, multiaplicación, móvil o no, destinado a ser utilizado en aplicaciones de automatización industrial. Incluye al manipulador (sistema mecánico y accionadores) y al sistema de control (software y hardware de control y potencia).

"La flexibilidad en la aplicación del robot y, por tanto, su utilidad dependerá en gran medida de las características de su sistema de programación." (Barrientos, y otros, 2007)

"La programación por guiado o aprendizaje consiste en hacer realizar al robot, o a una maqueta del mismo, la tarea (llevándolo manualmente, por ejemplo) al tiempo que se registran las configuraciones adoptadas, para su posterior repetición de manera automática." (Barrientos, y otros, 2007)

"El método de programación textual permite indicar la tarea al robot mediante el uso de un lenguaje de programación específico. Un programa se corresponde ahora, como en el caso de un programa general, con una serie de órdenes que son editadas y posteriormente ejecutadas. Existe, por tanto, un texto para el programa.

El texto del programa es editado en un sistema informático que puede ser independiente del robot,…" (Barrientos, y otros, 2007)

Los programas para programar robots incorporan día a día más herramientas, como pueden ser el modelado en 3D de los robots y su entorno, y la posibilidad de simular los programas antes de cargarlos en el robot real, pero el modo más utilizado para crear los programas sigue siendo la programación mediante un lenguaje textual.

"Los escasos intentos de unificar en cierta medida los procedimientos de programación de robots no han tenido hasta la fecha el reconocimiento y la aceptación necesarios, encontrándose que cada fabricante ha desarrollado su método particular, válido únicamente para sus propios robots." (Barrientos, y otros, 2007)

Es debido a todo lo anterior que en el marco del proyecto de investigación *"Desarrollo y aplicación eficiente de sistemas mecatrónicos"* que se desarrolla en el en el *Laboratorio de Mecatrónica* (LABME) de la *Facultad de Ingeniería y Ciencias Económico-Sociales* (FICES) de la *Universidad Nacional de San Luis* (UNSL) se ha construido un robot antropomorfo de 4 GDL (Grados De Libertad) del tipo industrial, para fines didácticos y experimentales, denominado CXN-I, y este trabajo tiene como objetivo realizar el sistema de control mediante PC y Placas de Adquisición y Control de Datos de este robot. Este trabajo abarca el diseño e implementación de la arquitectura y estrategias de control de los motores, el desarrollo de un control cinemático punto a punto, de una interfaz de usuario y de un lenguaje de programación textual del robot. El sistema desarrollado se utilizará para operar el robot y programar tareas de manipulación dentro de su espacio de trabajo, controlando posición, velocidad y aceleración de las articulaciones.

En el capítulo 1 se presentan, una introducción a los temas abordados, la motivación y el marco en el que se desarrolla este trabajo. En este capítulo también se presentan, el planteo del problema, los objetivos generales y específicos, y los alcances, límites y criterios establecidos.

En el capítulo 2 se describe la morfología del robot CXN-I, las dimensiones y área de trabajo del mismo. Se presentan los modelos de Denavit-Hartenberg[2] y geométrico, y de este último las ecuaciones de la cinemática directa e inversa. También se describen los mecanismos de propulsión y la unidad de control.

En el capítulo 3 se presenta una introducción al control de motores Paso a Paso (PaP) y la descripción de la arquitectura y de la estrategia de control de motores diseñados.

En el capítulo 4 se presenta una introducción al control cinemático de robots industriales; se describe el control cinemático punto a punto (PTP) desarrollado, explicando detalladamente el interpolador utilizado (interpolador trapezoidal) y el cálculo del mismo en los distintos tipos de movimientos; por último, se describe la implementación del mismo de acuerdo a la arquitectura y estrategias de control de motores expuestos en el capítulo 3.

En el capítulo 5 se describe la interfaz de usuario desarrollada, denominada *RobotLab*, el sistema operativo soporte, el lenguaje informático y el entorno utilizado en su desarrollo, y aspectos del funcionamiento de la misma.

En el capítulo 6 se describe el lenguaje de programación desarrollado, sus comandos y palabras reservadas, y por último se muestra un programa de ejemplo para una tarea de manipulación.

En el capítulo 7 se expresan las conclusiones y trabajos futuros.

Debido a la extensión y complejidad del código fuente del software desarrollado, este se encuentra en formato digital en el disco compacto (CD) adjunto a este informe, y no en formato impreso.

1.1. Planteo del problema

El problema consiste en controlar, mediante una PC y Placas de Adquisición y Control de Datos y a través de una interfaz de control amigable y fácil de usar, el robot CXN-I, construido en el *Laboratorio de Mecatrónica de la Facultad de Ingeniería y Ciencias Económico-Sociales* de la *Universidad Nacional de San Luis*, para que ubique su extremo en cualquier punto del espacio dentro de su volumen de trabajo, controlando posición, velocidad y aceleración de sus articulaciones.

1.2. Objetivos generales

El objetivo general de este trabajo es lograr un sistema de control mediante PC y Placas de Adquisición y Control de Datos para que el robot antropomorfo CXN-I construido en el *Laboratorio de Mecatrónica* de la FICES ubique su extremo en cualquier punto del espacio

[2] Modelo para describir la estructura cinemática de una cadena articulada mediante un procedimiento sistemático que utiliza sistemas de coordenadas y matrices de transformación homogéneas.

comprendido dentro de su volumen de trabajo, controlando la posición, velocidad y aceleración de sus articulaciones.

1.2.1. Objetivos específicos

Además de los objetivos generales se plantearon los siguientes objetivos específicos:

- Desarrollar una estrategia de control de los motores paso a paso que cumpla los siguientes requisitos:
  - Los motores deben arrancar y parar en forma suave para evitar oscilaciones.
  - Todos los motores deben arrancar y parar en el mismo instante (control cinemático coordinado).
  - Se debe poder controlar la aceleración y velocidad de los ejes de las articulaciones.
- Desarrollar un control cinemático punto a punto para el posicionamiento del extremo del robot en cualquier punto de su espacio de trabajo.
- Desarrollar una interfaz de usuario para el control del robot.

1.3. Alcances, límites y criterios.

El alcance de este trabajo es desarrollar un sistema de control para que el robot CXN-I pueda alcanzar cualquier punto dentro de su espacio de trabajo. Para lograr esto, es necesario realizar como mínimo un control cinemático del tipo Punto a Punto, siendo este el tipo de control al que se limitó el desarrollo de este trabajo.

Para cumplir con el objetivo específico de que los motores arranquen y paren en forma suave de tal manera de evitar oscilaciones, es necesario utilizar algún tipo de interpolador de las trayectorias articulares que implemente rampas de aceleración y desaceleración. Para lograr esto, se estableció la utilización de un interpolador de segmento lineal con transiciones parabólicas, también conocido como interpolador trapezoidal, como ley de movimiento de las articulaciones.

Para el desarrollo de este trabajo se estableció la utilización del lenguaje de programación C++, ya que este es un lenguaje ampliamente difundido y utilizado en el mundo entero en el desarrollo de proyectos de esta índole y de muchas otras. También se estableció la utilización de un sistema operativo *Windows*, debido a que estos son muy populares y muy amigables para usuarios no expertos.

Dentro de los alcances de este trabajo también se encuentra el de desarrollar una interfaz de usuario que permita la operación del robot de manera sencilla, segura y práctica, para que el mismo pueda ser operado por una persona que posea conocimientos mínimos de robótica.

En este trabajo también se desarrolló un lenguaje textual de programación del robot, utilizado en la interfaz de usuario para la operación y la programación de tareas del mismo. El lenguaje desarrollado se limitó a un lenguaje de comandos y parámetros, sin contar con sentencias condicionales ni llamadas a subrutinas.

También se desarrollo, dentro de la interfaz de usuario, una interfaz para la edición, ejecución y depuración de programas del robot, que utiliza el lenguaje textual desarrollado.

2. Descripción del robot CXN-I

2.1. Descripción general, dimensiones y espacio de trabajo

El robot CXN-I, construido en el *Laboratorio de Mecatrónica* (LABME) de la FICES, es un robot antropomorfo del tipo industrial de 4 GDL (Grados De Libertad) de características didácticas y para experimentación (figura 2.1). Los 4 GDL están dados por cuatro articulaciones rotativas, una cintura, un hombro, un codo y la rotación del extremo, el cual está equipado con un aprehensor (o pinza) para la manipulación de pequeños objetos.

Figura 2.1. CXN-I.

En la figura 2.2 se indican los movimientos de cada una de las articulaciones.

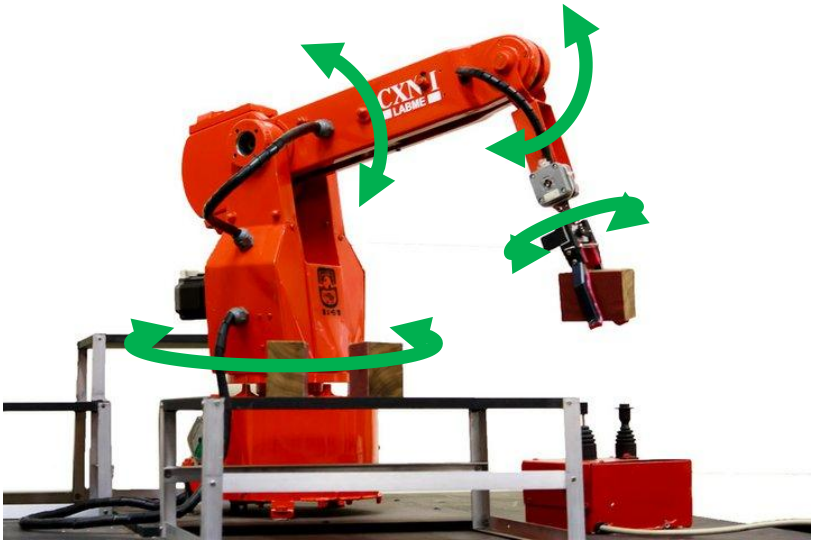


Figura 2.2. Direcciones de giro de las articulaciones.

En la figura 2.3 se pueden apreciar las dimensiones principales, los límites de giro de cada articulación y el espacio de trabajo del robot CXN-I.

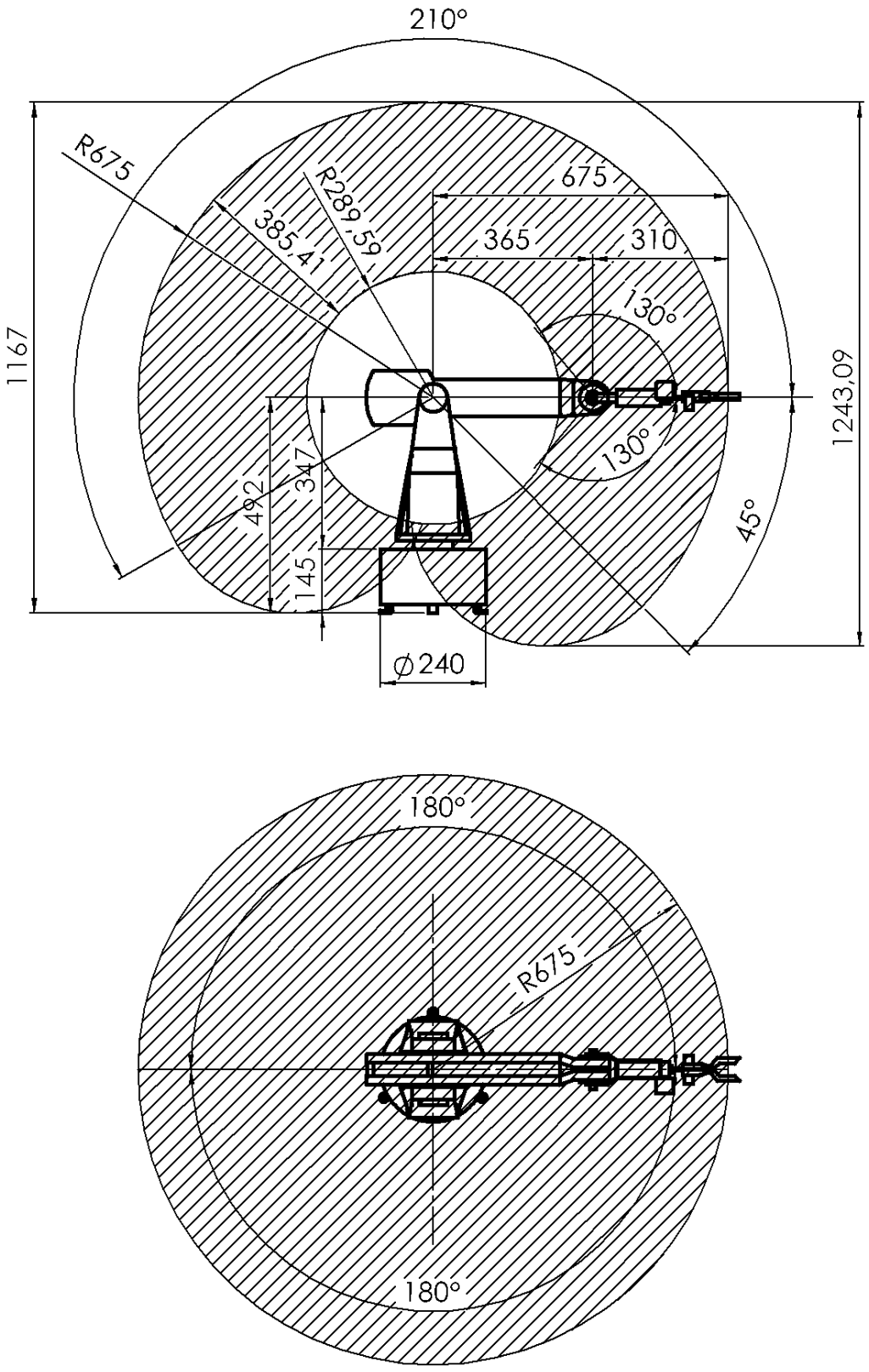


Figura 2.3. Dimensiones principales, límites de giro y espacio de trabajo.

2.2. Modelo de Denavit-Hartenberg

En la figura 2.4 se observa el modelo de Denavit-Hartenberg del robot CXN-I y en la tabla 2.1 los parámetros del mismo.

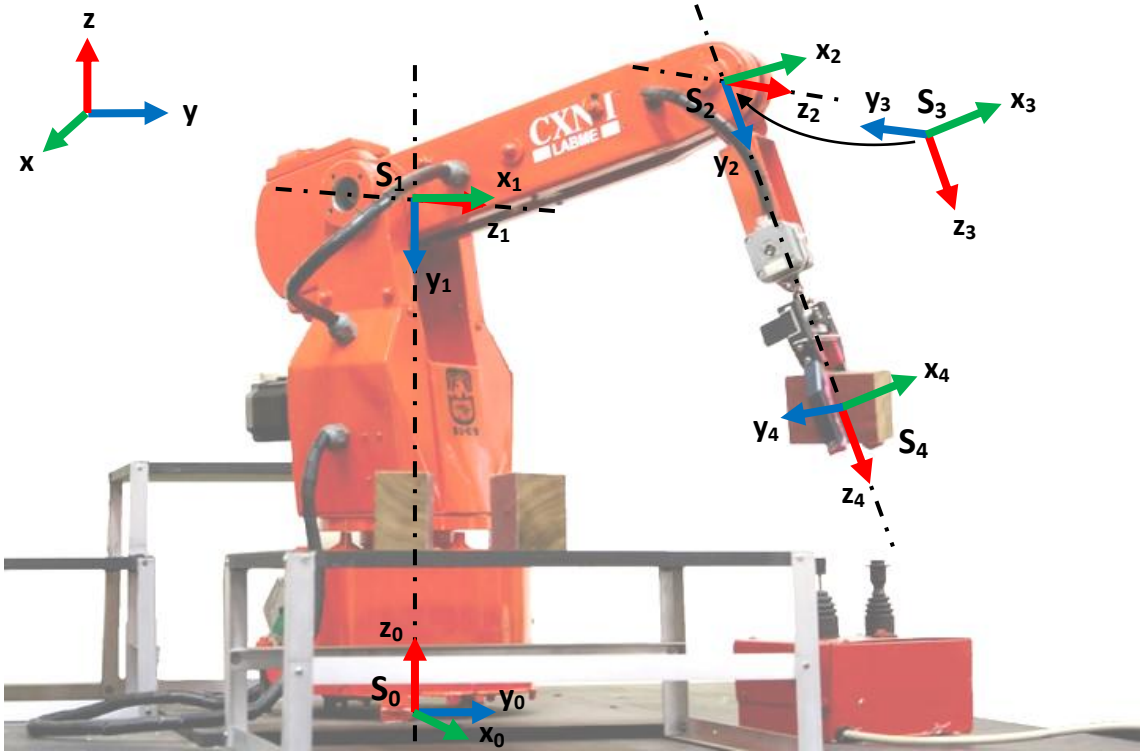


Figura 2.4. Robot CXN-I con el modelo D-H superpuesto.

	θ (tita)	d*	a*	α** (alfa)
1	θ_1	492	0	-90
2	θ_2	0	365	0
3	θ_3	0	0	-90
4	θ_4	310	0	0

*: Medidas en mm.
**: Ángulos en grados.

Tabla 2.1. Parámetros de D-H del CXN-I.

El método de Denavit-Hartenberg permite establecer de manera sistemática un sistema de coordenadas $\{Si\}$ ligado a cada eslabón i de una cadena articulada, pudiéndose determinar a continuación las ecuaciones cinemáticas de la cadena completa.

Desde el punto de vista computacional, debido a que este método utiliza matrices homogéneas (4x4), la complejidad de los algoritmos es del orden n^4, o sea que el número de operaciones crece con la cuarta potencia del número de grados de libertad.

Pese a su costo computacional, al obtenerse soluciones de manera sistemática, este método es conveniente en robots de geometría compleja o de elevado número de grados de libertad, siendo el más frecuentemente utilizado en estos casos.

2.3. Modelo geométrico

Debido a las características geométricas del robot CXN-I, el modelo geométrico (figura 2.5) del mismo es simple y en muchos casos más útil que el modelo de Denavit-Hartenberg debido a su menor costo computacional.

A continuación se presenta el modelo geométrico del robot CXN-I.

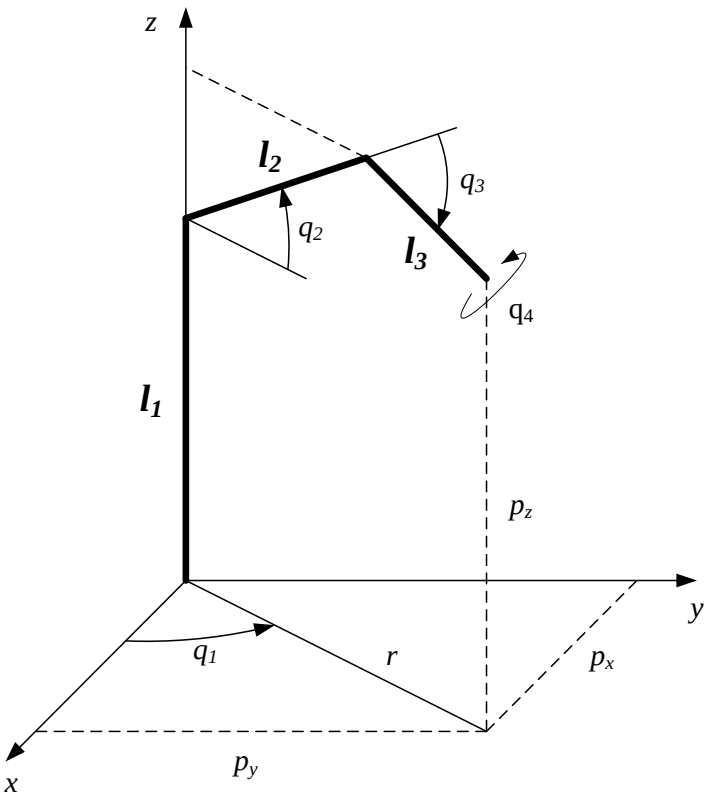


Figura 2.5. Modelo geométrico del robot CXN-I.

Donde:

$$l_1 = 492\,mm$$
$$l_2 = 365\,mm$$
$$l_3 = 310\,mm$$

2.3.1. Cinemática Directa

A partir del modelo geométrico de la figura 2.5, se pueden obtener las siguientes ecuaciones de la cinemática directa:

$$\begin{aligned} x &= \left[l_2 \cos q_2 + l_3 \cos\left(q_2 - q_3\right) \right] \cos q_1 \\ y &= \left[l_2 \cos q_2 + l_3 \cos\left(q_2 - q_3\right) \right] \operatorname{sen} q_1 \\ z &= l_1 + l_2 \operatorname{sen} q_2 + l_3 \operatorname{sen}\left(q_2 - q_3\right) \end{aligned} \qquad [2.1]$$

2.3.2. Cinemática Inversa

Debido a que el robot CXN-I tiene un modelo geométrico simple, se pueden obtener a partir del mismo las ecuaciones de la cinemática inversa.

De la figura 2.5 se puede obtener el valor de q1, como:

$$q_1 = \text{arctg}\left(\frac{p_y}{p_x}\right) \qquad [2.2]$$

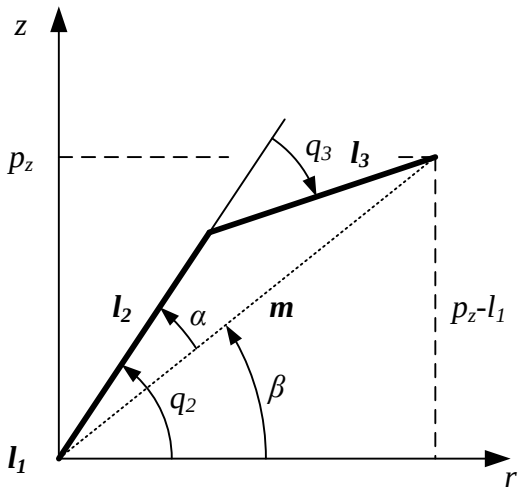


Figura 2.6. Modelo geométrico, plano z-r.

Luego, utilizando el modelo geométrico de la figura 2.5 y la proyección del mismo sobre el plano *z-r*, que se observa en la figura 2.6, se obtienen las siguientes ecuaciones:

$$r^2 = p_x{}^2 + p_y{}^2$$
$$m^2 = r^2 + \left(p_z - l_1\right)^2 = p_x{}^2 + p_y{}^2 + \left(p_z - l_1\right)^2$$

Aplicando el teorema del coseno al triángulo formado por los segmentos l_2,l_3 *y m, se obtiene:*

$$m^2 = l_2{}^2 + l_3{}^2 + 2l_2l_3\cos q_3$$
$$p_x{}^2 + p_y{}^2 + \left(p_z - l_1\right)^2 = l_2{}^2 + l_3{}^2 + 2l_2l_3\cos q_3$$

De donde se puede despejar cos q_3, obteniendo:

$$\cos q_3 = \frac{p_x{}^2 + p_y{}^2 + \left(p_z - l_1\right)^2 - l_2{}^2 - l_3{}^2}{2l_2l_3} \qquad [2.3]$$

De esta expresión se puede despejar el valor de q_3, pero, como se menciona en numerosos libros de robótica, debido a ventajas computacionales, es más conveniente utilizar la expresión de la arcotangente que la del arcocoseno.

Por ende, como

$$\operatorname{tg} q_3 = \frac{\operatorname{sen} q_3}{\cos q_3}$$

y

$$\operatorname{sen} q_3 = \pm\sqrt{1-\cos^2 q_3}$$

se tendrá

$$\boxed{q_3 = \operatorname{arctg}\left(\frac{\pm\sqrt{1-\cos^2 q_3}}{\cos q_3}\right)} \qquad [2.4]$$

donde $\cos q_3$ se calcula mediante la ecuación [2.3].

Como se observa en la ecuación [2.4], el valor de q_3 tiene dos soluciones posibles según se tome el signo positivo o negativo de la raíz, esto se debe a que el robot puede tomar la configuración codo arriba o codo abajo para el mismo punto. En este trabajo se tomó por defecto la configuración codo arriba, con lo cual la ecuación [2.4] se utiliza con el signo positivo, ya que la misma fue despejada del modelo en esa configuración.

Por último, para obtener el valor de q_2 se deben calcular α y β, ver figura 2.6.

$$q_2 = \alpha + \beta$$

De la figura 2.6, se observa que los valores de α y β son:

$$\alpha = \operatorname{arctg}\left(\frac{l_3 \operatorname{sen} q_3}{l_2 + l_3 \cos q_3}\right)$$

$$\beta = \operatorname{arctg}\left(\frac{p_z - l_1}{r}\right) = \operatorname{arctg}\left(\frac{p_z - l_1}{\sqrt{{p_x}^2 + {p_y}^2}}\right)$$

La raíz en el denominador de la última expresión no presenta el singo ± ya que *r* es una distancia, por lo cual su valor debe ser siempre mayor o igual a cero.

Entonces q_2 queda

$$\boxed{q_2 = \operatorname{arctg}\left(\frac{l_3 \operatorname{sen} q_3}{l_2 + l_3 \cos q_3}\right) + \operatorname{arctg}\left(\frac{p_z - l_1}{\sqrt{{p_x}^2 + {p_y}^2}}\right)} \qquad [2.5]$$

En esta ecuación se observa que el valor de q_2 es función de q_3, por ende el cálculo de q_3 para configuración codo arriba o codo abajo hará que el valor de q_2 sea el adecuado para esa configuración.

Mediante las ecuaciones [2.2], [2.3], [2.4] y [2.5] se resuelve el problema cinemático inverso para las tres primeras articulaciones (q_1, q_2, q_3) del robot CXN-I. Para la cuarta articulación, el valor del ángulo de orientación de la pinza (q_4) es definido directamente por el operador.

2.4. Propulsión

El robot CXN-I está impulsado por motores eléctricos paso a paso Sanyo Denki (RTA, 2008) (figura 2.7) en sus cuatro articulaciones y está equipado por un servomotor de corriente continua para el accionamiento de la pinza.

Las tres primeras articulaciones cuentan con motores modelo 103-H7123-1740 y la cuarta con uno modelo 103-H548-04500.

Los motores paso a paso con los que está equipado el robot CXN-I son motores bipolares que permiten la utilización de los mismos en lógica de medio paso, dando como resultado una resolución de 400 pasos por vuelta, es decir, un ángulo de 0,9° por cada paso.

Figura 2.7. Motores Paso a Paso Sanyo Denki.

El acoplamiento de los motores a los ejes de las articulaciones se realiza a través de cajas reductoras a tonillo sinfín-corona (figura 2.8). Las cajas reductoras utilizadas son irreversibles, de tal manera que cuando los motores están desenergizados el robot mantiene su posición, prescindiendo entonces de mecanismos de freno. En la tabla 2.2 se observan los valores de relación de reducción de las articulaciones.

Figura 2.8. Cajas reductoras.

Articulación	Reducción
1	30:1
2	30:1
3	55:1
4	35:1

Tabla 2.2. Relaciones de reducción de las articulaciones.

Los motores Paso a Paso son impulsados a través de accionamientos (drives) RTA SAC 26 (RTA, 2008) (figura 2.9) que permiten el control de los motores en lógica de medio paso. En la figura 2.10 se observan los accionamientos y los transformadores de alimentación para los mismos.

Figura 2.9. Accionamientos RTA SAC 26.

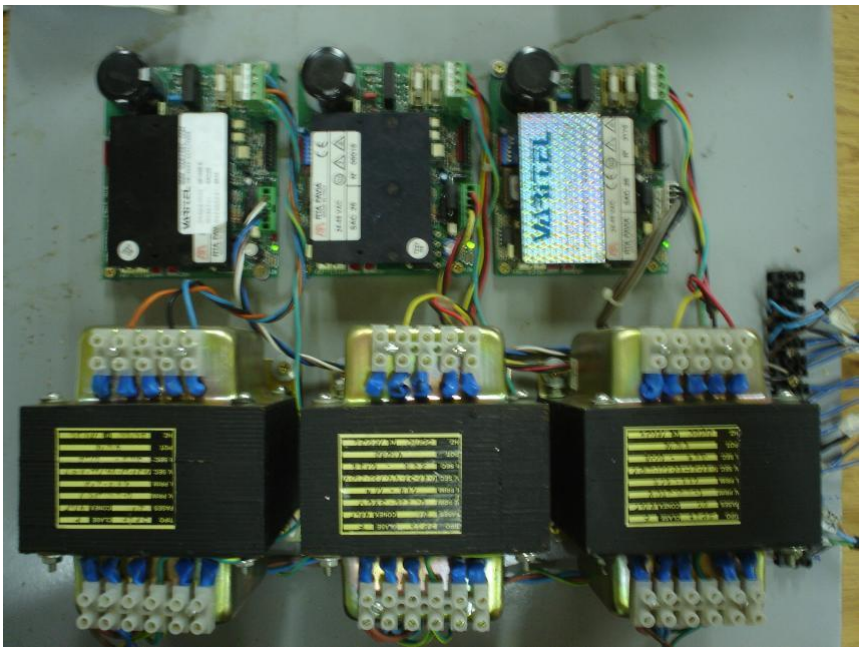


Figura 2.10. Accionamientos y transformadores de alimentación.

2.5. Unidad de control

En este trabajo, la configuración de control utilizada es la que se muestra en la figura 2.11. En esta configuración, la PC utilizada para el control se encuentra equipada con 2 Placas de Adquisición y Control de Datos PCX I/O de Axial Electrónica a través de las cuales se envían y realimentan las señales de control. Los accionamientos RTA SAC 26 transforman las señales de control en ondas eléctricas de potencia que impulsan los motores y posibilitan el movimiento del robot.

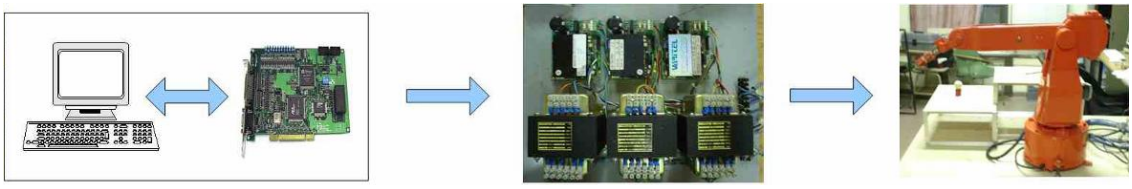


Figura 2.11. Esquema de la configuración de control.

En la figura 2.12 se observa la unidad de control, donde se encuentra la PC utilizada para el control equipada con las placas de Adquisición y Control de Datos. También se encuentran allí los accionamientos de los motores y los transformadores para su alimentación.

Figura 2.12. Gabinete de control.

3. Control de motores

3.1. Introducción al control de motores PaP

Los motores Paso a Paso son motores eléctricos capaces de girar en incrementos de un valor angular determinado, este valor depende de sus características constructivas y de las características del controlador utilizado. Los valores más comunes de estos incrementos son 1,8 grados, lo cual se conoce como lógica de paso completo, y 0,9 grados, que se conoce como lógica de medio paso. Estos motores tienen las ventajas de tener alta precisión y repetibilidad en cuanto al posicionamiento y un alto torque a bajas velocidades, lo cual los hace idóneos para su uso en robótica; pero su potencia nominal es baja, por lo cual se los emplea en robots pequeños (educacionales).

"Su principal ventaja con respecto a los servomotores tradicionales es su capacidad para asegurar un posicionamiento simple y exacto. Pueden girar además en forma continua, con velocidad variable, como motores síncronos, ser sincronizados entre sí, obedecer a secuencias complejas de funcionamiento, etc. Se trata al mismo tiempo de motores muy ligeros, fiables y fáciles de controlar, pues al ser estable cada estado de excitación del estator, el control se realiza en bucle abierto, sin la necesidad de sensores de realimentación." (Barrientos, y otros, 2007)

Los motores Paso a Paso están constituidos por un estator bobinado y un rotor que puede ser de imanes permanentes, de reluctancia variable o una combinación de ambos. El bobinado del estator está compuesto por dos bobinados independientes, llamados típicamente bobinado A y bobinado B, cuyas bobinas están dispuestas alternadamente y distribuidas uniformemente a lo largo del estator, de tal manera que mediante diferentes configuraciones de excitaciones, el rotor gira a determinadas posiciones a lo largo del estator, y al conmutar en una secuencia adecuada entre estas configuraciones el rotor del motor girará de a pasos. De esta manera, al controlar la conmutación de la excitación se puede controlar la posición, velocidad, aceleración y el sentido de giro del rotor.

El accionamiento de los motores Paso a Paso se debe realizar mediante accionamientos (Drives), que son los encargados de realizar la conmutación de la excitación de las bobinas. Generalmente los motores y los controladores se diseñan de manera que el motor se pueda mantener en una posición fija y también para que se los pueda hacer girar en un sentido y en el otro. Por lo general, estos accionamientos reciben 3 señales (TTL[3]) de entrada (figura 3.1):

- Un tren de pulsos
- Sentido de giro

[3] TTL, Transistor-Transistor Logic (Lógica Transistor a Transistor), es una tecnología de construcción de circuitos electrónicos digitales. Su tensión de alimentación característica debe estar entre 4,75V y 5,25V.

- Habilitación

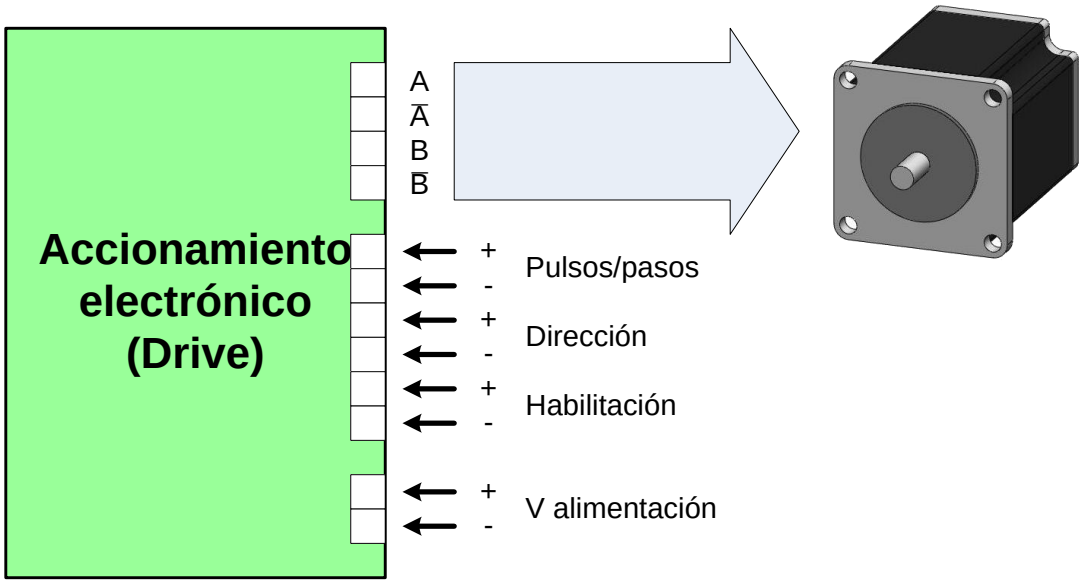


Figura 3.1. Esquema de señales de un accionamiento.

Por cada pulso del tren de pulsos enviado, el motor gira un paso (*p*), por ende el ángulo girado ($\Delta\theta$) es igual a la cantidad de pulsos enviados (*n*) por el paso (*p*) (ecuación [3.1]) y la velocidad de giro (ω) del mismo es proporcional a la frecuencia (*f*) del tren de pulsos enviado por el paso (*p*) (ecuación [3.2]):

$$\Delta\theta = p \cdot n \qquad [3.1]$$

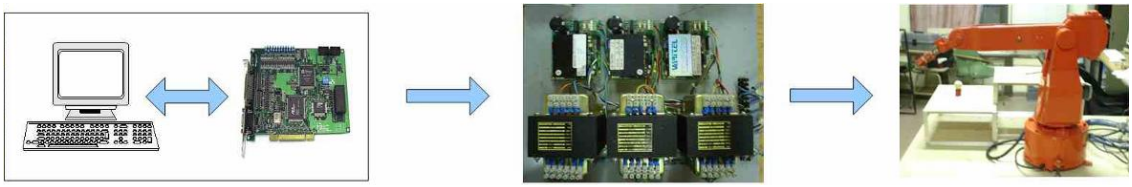
$$\omega = p \cdot f \qquad [3.2]$$

Por lo tanto, para controlar motores Paso a Paso se debe controlar el número de pulsos y la frecuencia del tren de pulsos enviado al accionamiento, como así también las señales de sentido de giro y de habilitación.

3.2. Arquitectura de control

En este trabajo el control de los motores del robot CXN-I se realiza mediante una PC equipada con placas de Adquisición y Control de Datos (figura 3.2).

En la PC, a través de la interfaz de usuario, se introducen los comandos y parámetros de control deseados. El programa de control desarrollado calcula las señales de control correspondientes, trenes de pulsos, sentidos de giro y habilitaciones. Las señales de control son enviadas a los accionamientos de los motores a través de las placas de adquisición y control de datos. Los accionamientos generan las ondas eléctricas necesarias para el movimiento de los motores y por ende del robot.



PC + Placas DAQ | Accionamientos y transformadores | Robot

Figura 3.2. Arquitectura de control.

3.3. Estrategia de control

Cuando se inicia el programa de control desarrollado, este configura convenientemente las placas de adquisición y control de datos. Luego, según los comandos y parámetros introducidos por el operador, el programa calcula los valores necesarios a cargar en las placas para generar las señales de control adecuadas para producir los movimientos deseados.

Los trenes de pulsos necesarios para comandar los motores, se generan utilizando los temporizadores (timers) de las unidades 8253 con las que cuentan las placas PCX-I/O, como se muestra en la figura 3.3. Por cada motor se utiliza un temporizador y, debido a que cada placa cuenta con una unidad 8253, y cada uno de estos contiene tres temporizadores, es necesario utilizar dos placas para controlar los cuatro motores del robot.

Para generar los trenes de pulsos, cada temporizador es configurado en modo generador de onda cuadrada. Los temporizadores pueden funcionar con la frecuencia interna de la placa o con una frecuencia externa, en este trabajo se optó por utilizar la frecuencia interna, que es de 1 MHz. En este modo, cada temporizador funciona como un divisor de frecuencia, donde la frecuencia del tren de pulsos (*f*) de salida es igual al valor de la frecuencia interna (*F*) de la placa dividido por un valor entero de 16 bits (*D*) que se carga por software en el registro de cada temporizador (ecuación [3.3]).

$$f = \frac{F}{D} \qquad [3.3]$$

Aprovechando la ventaja de los motores Paso a Paso de control a lazo abierto y debido a que, como se vio en el sección 3.1, el ángulo de giro de cada motor es proporcional al número de pulsos enviados a su controlador y la velocidad es proporcional a la frecuencia de ese tren de pulsos, para controlar el ángulo y la velocidad de giro de cada motor, es necesario controlar el número de pulsos enviados y la frecuencia de los mismos. Para esto, los trenes de pulsos son realimentados a través de los puertos digitales de las unidades 8255A con las que cuenta cada placa, de tal manera de conocer el número de pulsos enviados a cada controlador y también para ir variando la frecuencia de cada pulso enviado y así controlar a cada paso la velocidad de giro de cada motor y realizar la curva de desplazamiento calculada para cada motor de la manera más exacta posible.

Las señales de sentido de giro y de habilitación son enviadas utilizando los puertos de E/S digitales de las unidades 8255A, ver figura 3.3.

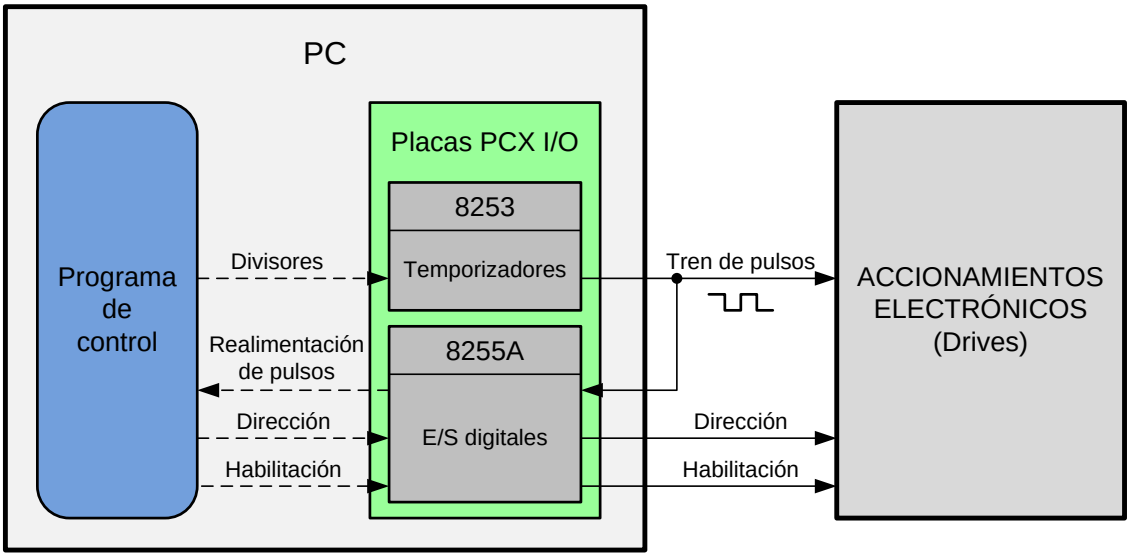


Figura 3.3. Estrategia de control de los motores PaP.

4. Control cinemático

En este capítulo se presenta una introducción al control cinemático de robots industriales, el control cinemático punto a punto (PaP) desarrollado y la implementación del mismo.

En la introducción al control cinemático de robots industriales se describen sus funciones generales, los tipos de trayectorias más usadas, haciendo hincapié en las trayectorias punto a punto y los tipos de movimientos de las articulaciones de este último (secuencial, simultáneo y coordinado), y por último se describe la interpolación de trayectorias.

Se describe después el control cinemático punto a punto desarrollado, explicando detalladamente el interpolador utilizado (interpolador trapezoidal) y el cálculo del mismo para cada articulación en los distintos tipos de movimientos.

Por último se describe la implementación del control cinemático de acuerdo a la arquitectura y estrategias de control de motores expuestos en el capítulo anterior.

4.1. Introducción al control cinemático

El control cinemático es el encargado de establecer la trayectoria a lo largo del tiempo que debe seguir cada articulación del robot para lograr los objetivos establecidos por el usuario, punto de destino, trayectoria cartesiana del efector final, duración de la incursión, etc. Estas trayectorias dependerán de las limitaciones físicas del robot y de sus actuadores (motores), del tipo de trayectoria (o movimiento) requerida por el usuario, de parámetros como aceleración y velocidad de las articulaciones o del efector final, tiempo de duración de la incursión, y en algunos casos también de criterios de calidad de la trayectoria, como suavidad o precisión de la misma.

El control cinemático recibe como variables de entrada los datos del movimiento especificado por el usuario (punto de destino, tipo de trayectoria cartesiana, aceleraciones, velocidades o tiempos de incursión, precisión, etc.) y, utilizando el modelo cinemático del robot, es decir, las ecuaciones de la cinemática directa e inversa, calcula las trayectorias articulares en función del tiempo.

4.1.1. Funciones del control cinemático

En general, el control cinemático realiza las siguientes funciones:

- Calcular la trayectoria cartesiana del efector final en función del tiempo de acuerdo a la especificación de movimiento dada por el usuario.
- Muestrear la trayectoria cartesiana.

- Utilizando las ecuaciones de la cinemática inversa, calcular para cada punto muestreado las correspondientes coordenadas articulares.
- Interpolar los puntos articulares obtenidos, de tal manera de obtener para cada articulación una trayectoria realizable por los actuadores y que dé cómo resultado una trayectoria cartesiana lo más próxima a la especificada por el usuario.
- Por último, muestrear la trayectoria articular obtenida para generar las referencias necesarias para el control dinámico.

En ciertos casos algunas de estas funciones pueden omitirse, como en el caso de que el usuario especifique una trayectoria ingresando las sucesivas configuraciones articulares que debe adoptar el robot a lo largo de la misma y no la trayectoria cartesiana para alcanzarlo, como por ejemplo en la programación por guiado o en un control punto a punto. En estos casos se omiten las tres primeras funciones mencionadas.

4.1.2. Tipos de trayectorias

Para alcanzar el punto de destino, los robots lo podrían hacer por infinitas trayectorias espaciales, pero, de todas estas, los robots comerciales realizan aquellas trayectorias que presentan facilidad de implementación o utilidad y aplicación a diversas tareas o a tareas específicas.

En general, debido a su utilidad y facilidad de implementación, los robots comerciales cuentan con la capacidad de realizar trayectorias punto a punto y algunos también trayectorias continuas, siendo las más comunes la línea recta y el arco de círculo.

4.1.2.1. Trayectoria punto a punto

La trayectoria punto a punto consiste en alcanzar el punto de destino estableciendo la trayectoria de las articulaciones, sin importar la trayectoria cartesiana realizada por el extremo del robot.

En este tipo de trayectoria el movimiento de las articulaciones se realiza en el menor tiempo posible, por esto se utiliza para el posicionamiento (o movimiento) rápido del extremo del robot, cuando no hay peligro de colisión del robot.

En este tipo de trayectoria al no tener importancia la trayectoria espacial del extremo, tampoco se realiza el muestreo correspondiente, solo se calcula la cinemática inversa del punto de destino para obtener sus coordenadas articulares y en cada articulación se utiliza un interpolador para unir las coordenadas articulares inicial y final.

Las distintas articulaciones pueden moverse de manera secuencial (eje a eje) o simultanea, y dentro de esta última, de manera coordinada o no.

4.1.2.1.1. Movimiento secuencial (eje a eje)

En este caso, las articulaciones se van moviendo de a una a la vez, es decir, una vez terminado el movimiento de una articulación comienza el movimiento de la siguiente. Por lo general el movimiento se realiza desde la base hacia el extremo del robot.

Este tipo de movimiento consume más tiempo que los demás, por lo cual no es muy utilizado. Este fue uno de los primeros controles realizados, debido a su sencillez y a que permitía la multiplexación del controlador.

4.1.2.1.2. Movimiento simultáneo

En este caso, todas las articulaciones comienzan a moverse en el mismo instante, pero cada una lo hace en el menor tiempo posible de acuerdo a sus parámetros de aceleración y velocidad, por lo cual pueden terminar de moverse en distintos momentos.

Este tipo de movimiento es más rápido que el movimiento secuencial (eje a eje).

El movimiento termina cuando se alcanza el punto de destino, lo cual acurre cuando todas las articulaciones hayan acabado de moverse, por ende el tiempo total del movimiento será igual al tiempo empleado por la articulación que más tarde.

Como el tiempo total del movimiento es igual al tiempo empleado por la articulación que más tarde, las articulaciones restantes pueden estar siendo sometidas a valores de aceleración y velocidad innecesariamente elevados, con los correspondientes esfuerzos dinámicos en las partes mecánicas, pudiendo estos valores ser disminuidos. Es por esto que este tipo de movimiento generalmente no es implementado y el que se utiliza es el movimiento coordinado.

4.1.2.1.3. Movimiento coordinado (Trayectorias coordinadas o isócronas)

En este tipo de movimiento todas las articulaciones comienzan a moverse en el mismo instante y también terminan en el mismo instante, es decir, todos los movimientos de las distintas articulaciones tienen la misma duración.

Esto se logra haciendo que los tiempos de incursión empleados por cada una de las articulaciones sean iguales al tiempo empleado por la articulación que más se tarde. Para aumentar el tiempo empleado por las articulaciones más rápidas se disminuyen los parámetros de aceleración o velocidad, o ambos, hasta que el tiempo de la incursión iguale al de la articulación que más tarde. Al disminuir las aceleraciones y velocidades de las articulaciones se disminuyen también los esfuerzos mecánicos experimentados por las mismas.

Debido a sus ventajas respecto a los otros tipos de movimientos, este es el tipo de movimiento comúnmente realizado por robots comerciales cuando se utilizan trayectorias punto a punto.

4.1.2.2. Trayectorias continuas

En este caso, el usuario especifica la trayectoria cartesiana que debe seguir el extremo del robot hasta alcanzar su punto de destino. Por lo general, las trayectorias cartesianas son definidas analíticamente y, por su simplicidad de implementación y utilidad práctica, las más habituales son la línea recta y el arco de círculo.

Debido a que, en general, no es factible obtener a partir de la descripción analítica de la trayectoria cartesiana la correspondiente descripción analítica de las trayectorias articulares,

para realizar este tipo de trayectorias, el control cinemático debe realizar todos los pasos mencionados anteriormente, muestrear la trayectoria espacial, calcular las coordenadas cartesianas para cada uno de los puntos muestreados y luego interpolar los puntos articulares para obtener trayectorias articulares continuas.

En general, las trayectorias articulares obtenidas presentarán una compleja evolución temporal, con cambios de dirección y velocidad y sin coordinación con el resto de las articulaciones. Sin embargo, el resultado conjunto será que el extremo del robot describirá la trayectoria deseada.

4.1.3. Interpolación de trayectorias

Una vez obtenidas las coordenadas articulares de los puntos muestreados de la trayectoria espacial del extremo del robot, esta sucesión de puntos articulares se debe unir para formar una trayectoria articular continua que pueda ser ejecutada por lo motores, respetando los límites de aceleración y velocidad de los mismos.

Para unir todos los puntos con una única función interpoladora, por lo general, se utilizaría un polinomio de grado elevado, pero esto presentaría múltiples inconvenientes desde el punto de vista computacional, por lo cual, se utilizan conjuntos de funciones más simples, que interpolan localmente la secuencia de puntos, esto es, unen unos pocos puntos consecutivos de un intervalo, empalmándose unas con otras para garantizar la continuidad.

Según el tipo de trayectoria cartesiana requerida y factores como la cantidad de puntos muestreados y el grado de precisión y suavidad requerida, se utilizarán diferentes interpoladores. Los más frecuentemente utilizados son interpoladores lineales, splin cúbico, splin quíntico y trapezoidales.

Cabe indicar que los interpoladores mencionados también son aplicables a las trayectorias espaciales del extremo del robot.

4.2. Control cinemático Punto a Punto desarrollado

De acuerdo a los objetivos y alcances de este trabajo, se desarrolló un control cinemático del tipo punto a punto, ya que se requiere que el robot sea capaz de alcanzar cualquier punto dentro de su volumen de trabajo, sin importar la trayectoria espacial utilizada para tal fin.

Como se mencionó anteriormente, una trayectoria punto a punto consiste en alcanzar el punto de destino estableciendo la trayectoria de las articulaciones y no la trayectoria espacial realizada por el extremo del robot.

En este tipo de trayectorias, al no tener importancia la trayectoria espacial del extremo, tampoco se realiza el muestreo correspondiente, solo se calcula la cinemática inversa del punto de destino para obtener sus coordenadas articulares y por cada articulación se utiliza un interpolador para unir las coordenadas iniciales y finales. En este trabajo se estableció como

criterio la utilización de un interpolador de segmento lineal con transiciones parabólicas, también conocido como interpolador trapezoidal.

Debido a que este trabajo se desarrolla en el marco de un proyecto de investigación y a que el robot CXN-I es para fines didácticos y experimentales, se implementaron los 3 tipos de movimientos posibles para un control punto a punto mencionados anteriormente, estos son, movimiento secuencial (eje a eje), movimiento simultáneo, y movimiento coordinado.

4.2.1. Interpolador Trapezoidal

Para cumplir con el objetivo específico de que los motores arranquen y paren en forma suave de tal manera de evitar oscilaciones, se hace necesario utilizar algún tipo de interpolador de las trayectorias articulares que implemente rampas de aceleración y desaceleración. Para lograr esto se estableció la utilización de un interpolador de segmento lineal con transiciones parabólicas, también conocido como interpolador trapezoidal, ver figura 4.1.

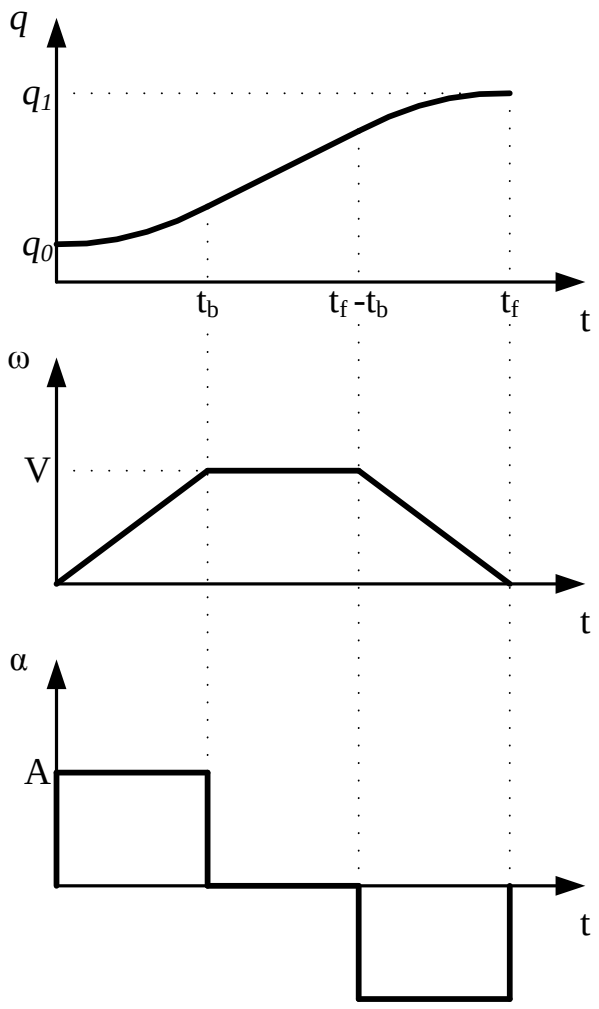


Figura 4.1. Interpolador trapezoidal.

La utilización de este interpolador se debe a que es el que permite realizar el movimiento en el menor tiempo posible para los parámetros de aceleraciones y velocidades articulares establecidos. Esto se debe a que las articulaciones se aceleran hasta alcanzar la velocidad de régimen lo más rápido posible, es decir, con aceleración constante e igual a los valores establecidos, por ende, estas velocidades se alcanzan en el menor tiempo posible. Las velocidades se mantienen el tiempo necesario y luego se desacelera también, tan rápidamente como se puede, es decir, con desaceleración constante e igual a los valores establecidos, por ende, se desacelera en el menor tiempo posible. Todo esto da como resultado, que el movimiento se realice en el menor tiempo posible. Para este trabajo se estableció usar iguales valores de aceleración y de desaceleración, haciendo la curva simétrica.

En el caso general, este interpolador está compuesto por tres tramos, en el primer tramo la articulación se acelera con aceleración constante (figura 4.1) hasta alcanzar la velocidad de régimen establecida, en el segundo tramo esta velocidad se mantiene constante, es decir, la articulación no es acelerada ni desacelerada, y en el tercer tramo la articulación se desacelera con desaceleración constante.

Puede ocurrir que la incursión no sea lo suficientemente grande como para que se alcance la velocidad de régimen, en cuyo caso la curva queda constituida por dos tramos, el tramo de aceleración e inmediatamente después, el de desaceleración.

Utilizando las ecuaciones de la cinemática para movimientos con aceleración uniforme, el desplazamiento angular necesario para que la articulación alcance la velocidad de régimen (Δq_v) está dado por la siguiente expresión:

$$\Delta q_V = \frac{V^2}{2A} \qquad [4.1]$$

Debido a que para este trabajo se han establecido valores de aceleración y desaceleración iguales, los tiempos de aceleración y de desaceleración también lo serán, por ende, si la incursión es menor o mayor al doble del desplazamiento angular necesario para que la articulación alcance la velocidad de régimen (ecuación [4.1]), el interpolador estará formado por dos o tres tramos respectivamente (expresión [4.2]).

$$Si\ (q_1 - q_0) \begin{cases} > \dfrac{V^2}{A} & \Rightarrow\ 3 tramos \\ \\ \leq \dfrac{V^2}{A} & \Rightarrow\ 2 tramos \end{cases} \qquad [4.2]$$

4.2.1.1. Interpolador con 3 tramos

En este caso, la incursión es lo suficientemente grande para que se alcance la velocidad de régimen (V), por ende se cumple la ecuación [4.3].

$$(q_1 - q_0) > \frac{V^2}{A} \qquad [4.3]$$

A continuación se presentarán las ecuaciones para cada tramo del interpolador.

4.2.1.1.1. Primer tramo

Este tramo va desde $t=0$ hasta $t=t_b$, ver figura 4.1. Al comienzo de este tramo la articulación se encuentra en reposo y es acelerada con aceleración constante (A) hasta alcanzar la velocidad de régimen (V).

Las ecuaciones de la posición (q), velocidad (ω) y aceleración (α) angular para este tramo serán:

$$\boxed{0 \le t \le t_b \quad \begin{cases} q_{(t)} = q_0 + \frac{1}{2} A t^2 \\ \omega_{(t)} = A t \\ \alpha_{(t)} = A \end{cases}} \qquad [4.4]$$

Sabiendo que en el tiempo t_b se alcanza la velocidad de régimen (V), y utilizando las ecuaciones [4.4], se tiene:

$$V = A t_b \qquad [4.5]$$

De la cual se puede despejar el valor de t_b, que será:

$$\boxed{t_b = \frac{V}{A}} \qquad [4.6]$$

4.2.1.1.2. Segundo tramo

En este tramo la velocidad se mantiene constante, es decir, que la aceleración es nula. Las ecuaciones para este tramo serán:

$$t_b \le t \le t_f - t_b \quad \begin{cases} q_{(t)} = q_{(t_b)} + V\left(t - t_b\right) \\ \omega_{(t)} = V \\ \alpha_{(t)} = 0 \end{cases} \qquad [4.7]$$

Reemplazando con el valor de t_b por el de la ecuación [4.6] y con el valor de q para el tiempo t_b, obtenido de las ecuaciones [4.4]

$$q_{(t_b)} = q_0 + \frac{1}{2} A {t_b}^2 \qquad [4.8]$$

Las ecuaciones para este tramo quedan:

$$t_b \le t \le t_f - t_b \quad \begin{cases} q_{(t)} = q_0 - \dfrac{V^2}{2A} + V\,t \\ \omega_{(t)} = V \\ \alpha_{(t)} = 0 \end{cases} \qquad [4.9]$$

4.2.1.1.3. Tercer tramo

En este tramo la articulación se desacelera con desaceleración contante desde la velocidad de régimen hasta detenerse en el punto de destino. Las ecuaciones para este tramo serán:

$$t_f - t_b \le t \le t_f \quad \begin{cases} q_{(t)} = q_1 - \dfrac{1}{2} A\left(t - t_f\right)^2 \\ \omega_{(t)} = -A\left(t - t_f\right) \\ \alpha_{(t)} = -A \end{cases} \qquad [4.10]$$

El valor de t_f puede ser despejado de la siguiente ecuación:

$$q_1 = q_0 + \frac{1}{2} A t_b{}^2 + V\left[\left(t_f - t_b\right) - t_b\right] + \frac{1}{2} A\left[\left(t_f - t_b\right) - t_f\right]^2 \qquad [4.11]$$

De donde se obtiene:

$$t_f = \frac{q_1 - q_0}{V} + t_b \qquad [4.12]$$

Y reemplazando t_b por el de la ecuación [4.6], se obtiene:

$$\boxed{t_f = \frac{q_1 - q_0}{V} + \frac{V}{A}} \qquad [4.13]$$

Utilizando este valor de t_f en las ecuaciones [4.10], las ecuaciones para el tramo 3 quedan:

$$t_f - t_b \le t \le t_f \quad \begin{cases} q_{(t)} = q_1 - \dfrac{1}{2} A\left[t - \left(\dfrac{q_1 - q_0}{V} + \dfrac{V}{A}\right)\right]^2 \\ \omega_{(t)} = -A\left(t - \dfrac{q_1 - q_0}{V}\right) + V \\ \alpha_{(t)} = -A \end{cases} \qquad [4.14]$$

4.2.1.2. Interpolador con 2 tramos.

Cuando la incursión no es lo suficientemente grande para que se alcance la velocidad de régimen (V), o para que esta pueda ser mantenida durante un tiempo, el interpolador constará de dos tramos, un tramo de aceleración, e inmediatamente después un tramo de desaceleración, ver figura 4.2. Para este caso se cumple la siguiente ecuación:

$$(q_1 - q_0) \leq \frac{V^2}{A} \qquad [4.15]$$

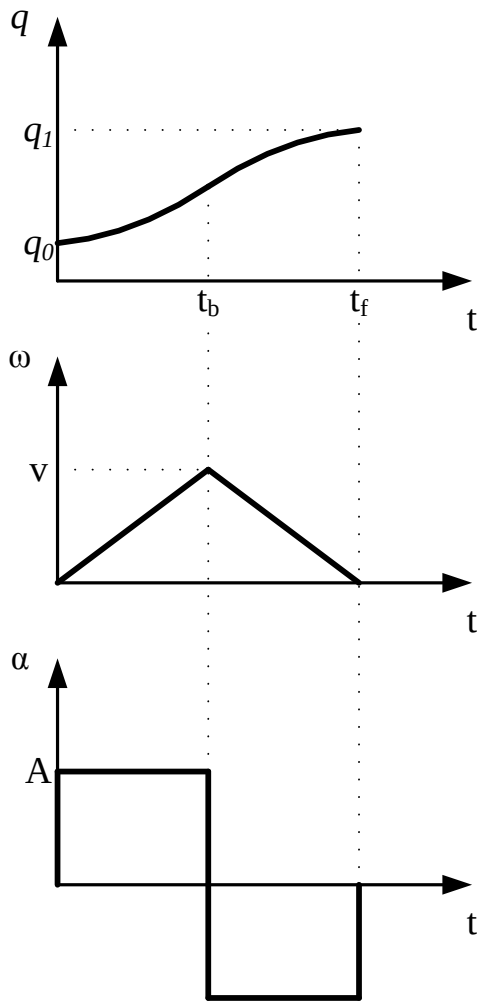


Figura 4.2. Interpolador trapezoidal con 2 tramos.

Nuevamente, mediante la utilización de las ecuaciones de la cinemática para un cuerpo que se acelera con aceleración uniforme, se puede encontrar el valor del tiempo t_b en el que la articulación realiza la mitad de su recorrido angular.

$$\frac{q_1 - q_0}{2} = \frac{1}{2} A t_b^{\,2} \qquad [4.16]$$

Despejando de esta ecuación el valor de t_b, se obtiene:

$$t_b = \sqrt{\frac{q_1 - q_0}{A}} \qquad [4.17]$$

Y como

$$t_f = 2t_b \qquad [4.18]$$

El valor de t_f será:

$$t_f = 2\sqrt{\frac{q_1 - q_0}{A}} \qquad [4.19]$$

A continuación se presentarán las ecuaciones para cada tramo del interpolador.

4.2.1.2.1. Primer tramo

Este tramo va desde $t=0$ hasta $t=t_b$, que en este caso será igual a la mitad del tiempo de incursión, ver figura 4.2. Al igual que en el caso del interpolador con tres tramos, al comienzo de este tramo la articulación se encuentra en reposo y es acelerada con aceleración constante (A) hasta alcanzar la mitad del recorrido.

Las ecuaciones de la posición (q), velocidad (ω) y aceleración (α) angular para este tramo serán:

$$0 \le t \le t_b \begin{cases} q_{(t)} = q_0 + \frac{1}{2}At^2 \\ \omega_{(t)} = At \\ \alpha_{(t)} = A \end{cases} \qquad [4.20]$$

Estas ecuaciones son iguales a las ecuaciones [4.4], para el primer tramo del caso del interpolador con tres tramos.

4.2.1.2.2. Segundo tramo

En este tramo la articulación se desacelera con desaceleración contante hasta detenerse en el punto de destino. Las ecuaciones para este tramo serán:

$$t_b \le t \le t_f \begin{cases} q_{(t)} = q_1 - \frac{1}{2}A\left(t - t_f\right)^2 \\ \omega_{(t)} = -A\left(t - t_f\right) \\ \alpha_{(t)} = -A \end{cases} \qquad [4.21]$$

Reemplazando t_f por el valor obtenido en la ecuación [4.19]:

$$t_b \le t \le t_f \begin{cases} q_{(t)} = q_1 - \frac{1}{2} A \left[t - 2\sqrt{\frac{q_1 - q_0}{A}} \right]^2 \\ \omega_{(t)} = -A\left(t - 2\sqrt{\frac{q_1 - q_0}{A}} \right) + V \\ \alpha_{(t)} = -A \end{cases} \quad [4.22]$$

4.2.2. Movimientos Secuencial, Simultáneo y Coordinado

Como se mencionó anteriormente, debido a que este trabajo se desarrolla en el marco de un proyecto de investigación y a que el robot CXN-I es para fines didácticos y experimentales, en este trabajo se implementaron los 3 tipos de movimientos posibles para un control punto a punto, estos son, movimiento secuencial (eje a eje), movimiento simultáneo, y movimiento coordinado.

4.2.2.1. Movimiento Secuencial (eje a eje)

En este caso, las articulaciones se van moviendo de a una a la vez, es decir, una vez terminado el movimiento de una articulación comienza el movimiento de la siguiente. Por lo general el movimiento se realiza desde la base hacia el extremo del robot.

Este tipo de movimiento consume más tiempo que los demás, por lo cual no es muy utilizado en la práctica. Este fue uno de los primeros controles realizados, debido a su sencillez y a que permitía la multiplexación del controlador.

En este tipo de movimiento cada articulación realizará la incursión con su correspondiente aceleración y velocidad, dando como resultado diferentes tiempos de incursión para cada articulación.

El tiempo total del movimiento será la suma de los tiempos de incursión de cada articulación.

4.2.2.2. Movimiento simultáneo

En este caso, todas las articulaciones comienzan a moverse en el mismo instante, pero cada una lo hace en el menor tiempo posible de acuerdo a sus parámetros de aceleración y velocidad, por lo cual pueden terminar de moverse en distintos momentos (figura 4.3).

El movimiento termina cuando se alcanza el punto de destino, lo cual acurre cuando todas las articulaciones hayan acabado de moverse, por ende el tiempo total del movimiento será igual al tiempo empleado por la articulación que más tarde.

Como el tiempo total del movimiento es igual al tiempo empleado por la articulación que más se tarde, las articulaciones restantes pueden estar siendo sometidas a valores de aceleración y velocidad innecesariamente elevados, con los correspondientes esfuerzos

dinámicos en las partes mecánicas, pudiendo estos valores ser disminuidos. Es por esto que este tipo de movimiento generalmente no es implementado y el que se utiliza es el movimiento coordinado.

A modo de ejemplo, en la figura 4.3 se muestran cuatro trayectorias articulares de un movimiento simultáneo. Las cuatro articulaciones realizan incursiones de diferente magnitud.

$$\Delta q_1 > \Delta q_2 > \Delta q_3 > \Delta q_4 \qquad [4.23]$$

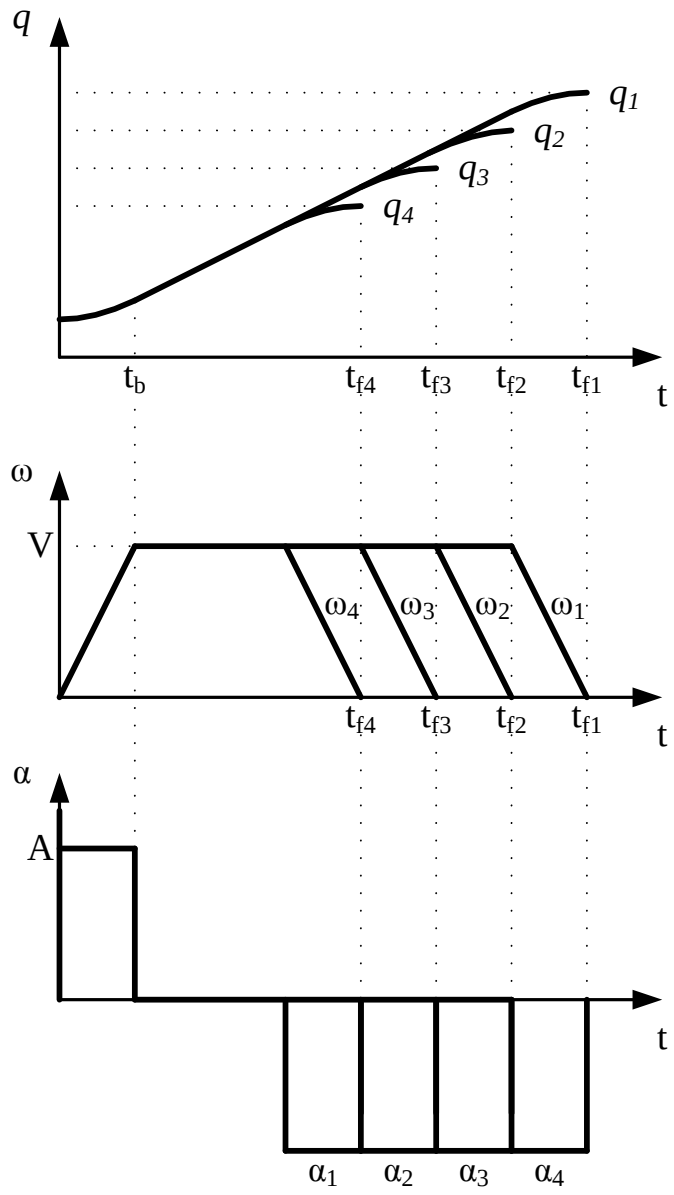


Figura 4.3. Trayectorias articulares en un movimiento simultáneo.

Para simplificar el gráfico todas las incursiones parten de la misma posición articular. También, por el mismo motivo, se tomaron iguales valores de aceleración y velocidad para las cuatro articulaciones.

$$A_1 = A_2 = A_3 = A_4 = A \quad [4.24]$$

$$V_1 = V_2 = V_3 = V_4 = V \quad [4.25]$$

Como se observa en el gráfico, para los parámetros establecidos, al tener las diferentes articulaciones diferentes magnitudes de desplazamientos angulares, estas terminan de moverse en distintos instantes de tiempo.

$$t_{f1} > t_{f2} > t_{f3} > t_{f4} \quad [4.26]$$

El tiempo total de la incursión t_f será igual al mayor de los tiempos de cada articulación, en este caso:

$$t_f = t_{f1} \quad [4.27]$$

4.2.2.3. Movimiento coordinado (Trayectorias coordinadas o isócronas)

En este tipo de movimiento todas las articulaciones comienzan y terminan sus movimientos en el mismo momento, es decir, los movimientos de las distintas articulaciones tienen la misma duración.

Esto se logra calculando cual sería la articulación que más tarde en realizar su incursión y haciendo que las demás tarden lo mismo. Para aumentar los tiempos empleados por las articulaciones más rápidas se pueden disminuir los valores de aceleración o velocidad, o ambos, hasta que igualen al de la articulación que más tarde. Al disminuir las aceleraciones y velocidades de las articulaciones se disminuyen también los esfuerzos mecánicos experimentados por las mismas.

En este trabajo se optó por aumentar el tiempo de incursión empleado por las articulaciones más rápidas mediante la disminución de los valores de aceleración de tal manera de disminuir los esfuerzos dinámicos sobre los elementos mecánicos. Haciendo esto para el ejemplo utilizado en el movimiento secuencial (figura 4.3), se obtienen las trayectorias articulares que se muestran en la figura 4.4.

Para realizar este tipo de movimiento, primero se calculan los tiempos de las incursiones de las articulaciones mediante las ecuaciones [4.13] o [4.19], según el caso correspondiente y se determina el mayor de ellos. Para el ejemplo de la figura 4.3, este estará dado por el valor t_{f1}.

Una vez obtenido el mayor tiempo de incursión (t_m), se calcula, para las articulaciones cuyo tiempo de incursión es menor, los valores de aceleración para que los tiempos de incursión sean iguales al tiempo obtenido (t_m).

Sin embargo, al utilizar un interpolador trapezoidal, para cada valor de incursión (Δ_q) y de velocidad de régimen (V), existe un tiempo de incursión límite (t_l) por encima del cual la articulación no alcanzará la velocidad de régimen, es decir, la trayectoria articular pasará de

estar compuesta por tres tramos a estar compuesta por dos. Esto es lo que ocurre en la articulación número 4 del ejemplo, ver figuras 4.3 y 4.4.

También podría decirse que existe un valor de aceleración límite (A_l) por debajo del cual, para una incursión dada (Δq), no se alcanzará la velocidad de régimen (V) y la trayectoria articular estará compuesta por dos tramos.

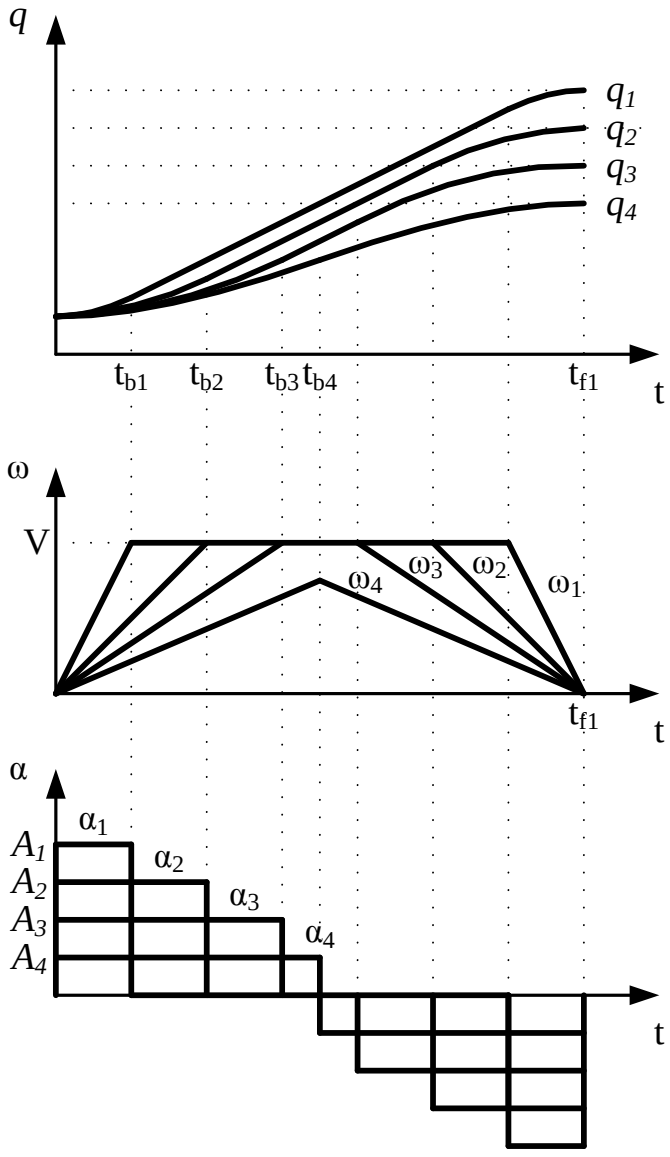


Figura 4.4. Trayectorias articulares para movimiento coordinado.

Utilizando las ecuaciones [4.2], [4.13] y [4.19], se obtiene:

$$t_l = 2\frac{\Delta q}{V} \qquad [4.28]$$

y

$$A_l = \frac{V^2}{\Delta q} \qquad [4.29]$$

Es por esto que para calcular la aceleración necesaria para que cada articulación realice la incursión en el tiempo t_m primero hay que calcular el tiempo límite (t_l) de dicha incursión y compararlo con t_m para así saber si la incursión estará formada por dos o tres tramos. Los valores de aceleración se obtienen despejando de las ecuaciones [4.13] y [4.19], para el caso de tres y dos tramos respectivamente.

$$Si\ \ t_m \begin{cases} \geq t_l \quad \Rightarrow \quad 2\,tramos \quad \Rightarrow \quad \boxed{A = 4\frac{\Delta q}{t_m^{\,2}}} \\ < t_l \quad \Rightarrow \quad 3\,tramos \quad \Rightarrow \quad \boxed{A = \frac{V}{t_f - \frac{\Delta q}{V}}} \end{cases} \qquad [4.30]$$

Con los valores de aceleración necesarios para cada incursión, calculados utilizando las ecuaciones [4.30], se determinan las trayectorias articulares para realizar el movimiento coordinado.

Debido a sus ventajas respecto a los otros tipos de movimientos, este es el tipo de movimiento realizado comúnmente por robots comerciales cuando se utilizan trayectorias punto a punto.

4.3. Implementación del control cinemático

De acuerdo a la arquitectura y a la estrategia de control de los motores descriptas en el capítulo 3, para implementar el control cinemático se deben calcular los valores de divisores de frecuencia que generen los trenes de pulsos adecuados para que los motores produzcan en las articulaciones las trayectorias articulares calculadas por el control cinemático.

Debido a que el robot CXN-I está impulsado por motores paso a paso, y que estos giran en incrementos finitos (pasos), se aproximan las coordenadas articulares de destino a las más próximas realizables por los motores. Con estas coordenadas articulares se procede al cálculo de las trayectorias articulares, según los parámetros establecidos para el control cinemático (aceleraciones, velocidades, tipo de movimiento, etc.).

Una vez obtenidas las trayectorias articulares, a estas se les aplica un factor de escala para obtener las trayectorias angulares que deben seguir los motores. El factor de escala es igual a la relación de transmisión de los motores a los ejes de las articulaciones. Por ende, la posición (θ), velocidad (ω_m) y aceleración (α_m) de los motores se obtiene multiplicando, respectivamente, la posición (q), velocidad (ω) y aceleración (α) de las articulaciones por el valor de la relación de transmisión (R) correspondiente, ecuaciones [4.31].

$$\begin{aligned} \theta_{(t)} &= R\, q_{(t)} \\ \omega_{m(t)} &= R\, \omega_{(t)} \\ \alpha_{m(t)} &= R\, \alpha_{(t)} \end{aligned} \qquad [4.31]$$

La relación de transmisión (R) es igual al número de vueltas del motor por cada vuelta del eje de la articulación. Los valores de relación de transmisión para el robot CXN-I se encuentran en la tabla 2.2.

De las ecuaciones [4.31] se desprende que el desplazamiento angular ($\Delta\theta_T$) y los parámetros de velocidad de régimen (V_m) y de aceleración angular (A_m) de los motores, son R veces mayores al desplazamiento angular (Δq), velocidad de régimen (V) y aceleración (A) en el eje de las articulaciones respectivamente, ecuaciones [4.32].

$$\begin{aligned} \Delta\theta_T &= R\left(q_1 - q_0\right) \\ V_m &= R V \\ A_m &= R A \end{aligned} \qquad [4.32]$$

Una vez obtenidos los parámetros de las trayectorias que deben realizar los motores, se deben calcular los trenes de pulsos, para lo cual es necesario calcular la cantidad de pulsos a enviar para que cada motor realice el correcto desplazamiento angular y el tiempo de duración de cada uno de estos pulsos para que los motores sigan las trayectorias que se obtuvieron del control cinemático.

La cantidad total de pulsos (n) es igual al desplazamiento angular ($\Delta\theta_T$) dividido por el ángulo de giro por paso (p) del motor.

$$n = \frac{\Delta\theta_T}{p} \qquad [4.33]$$

Debido a que en este trabajo los motores se utilizan en lógica de medio paso, el ángulo de giro por paso del motor es de 0,9 grados.

Para calcular el tiempo de duración de cada pulso se debe calcular el tiempo en el que los motores deben dar cada uno de sus pasos a lo largo de la trayectoria angular, ver figura 4.5.

Primero se debe determinar si la trayectoria estará compuesta por 2 o 3 tramos, ver expresión [4.2]. Luego con los valores de t_b y t_f se calculan los desplazamientos angulares ($\Delta\theta_b$) en los que se produce el cambio de tramo, siendo:

$$\Delta\theta_b = \frac{1}{2} A_m t_b^{\,2} \qquad [4.34]$$

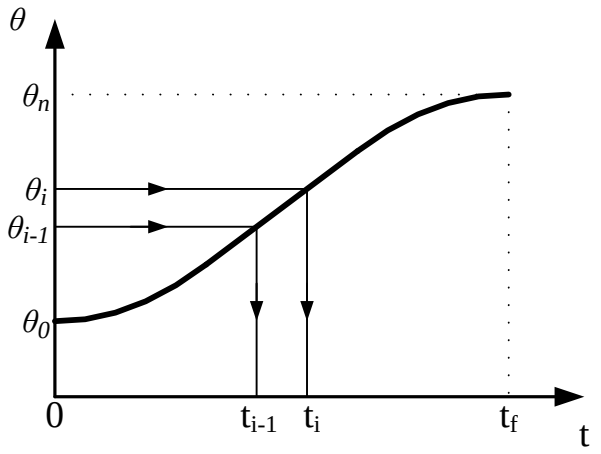


Figura 4.5. Cálculo de tiempos.

Utilizando las ecuaciones para el cálculo de las trayectorias articulares expuestas anteriormente, junto con las [4.31] y [4.32], el tiempo para un valor de desplazamiento angular dado ($\Delta\theta$) será:

$$t=\begin{cases}\sqrt{\dfrac{2\Delta\theta}{A_m}} & 0<\Delta\theta\le\Delta\theta_b \\ \dfrac{\Delta\theta}{V_m}+\dfrac{t_b}{2} & \Delta\theta_b<\Delta\theta<\Delta\theta_T-\Delta\theta_b \\ \sqrt{\dfrac{2\left(\Delta\theta-\Delta\theta_T\right)}{A_m}}+t_f & \Delta\theta_T-\Delta\theta_b\le\Delta\theta\le\Delta\theta_T\end{cases} \qquad [4.35]$$

Y como el desplazamiento angular para cada paso ($\Delta\theta_i$) está dado por:

$$\Delta\theta_i = p\cdot i \qquad i=1,2,3,\ldots,n \qquad [4.36]$$

Utilizando las ecuaciones [4.35] y [4.36], el tiempo (t_i) para cada paso será:

$$t_i=\begin{cases}\sqrt{\dfrac{2\Delta\theta_i}{A_m}} & 0<\Delta\theta_i\le\Delta\theta_b \\ \dfrac{\Delta\theta_i}{V_m}+\dfrac{t_b}{2} & \Delta\theta_b<\Delta\theta_i<\Delta\theta_T-\Delta\theta_b \\ \sqrt{\dfrac{2\left(\Delta\theta_i-\Delta\theta_T\right)}{A_m}}+t_f & \Delta\theta_T-\Delta\theta_b\le\Delta\theta_i\le\Delta\theta_T\end{cases} \qquad i=1,2,3,\ldots,n \qquad [4.37]$$

Obtenidos los tiempos (t_i) para cada paso de los motores, se calcula la frecuencia del pulso (f_i) a enviar para el correspondiente paso.

$$f_i = \frac{1}{t_i - t_{i-1}} \qquad i = 1, 2, 3, \ldots, n \qquad [4.38]$$

Siendo para $i = 1$, $t_{i-1} = 0$.

Con los valores de frecuencia (f_i) de cada paso, despejando el valor de divisor de frecuencia (D) de la ecuación [3.3], se calculan los valores de divisores de frecuencia teóricos para cada paso (D_i), ecuación [4.39].

$$D_i = \frac{F}{f_i} \qquad i = 1, 2, 3, \ldots, n \qquad [4.39]$$

Como los valores de divisores de frecuencia (D_i) deben ser valores enteros de 16 bits, los valores teóricos calculados mediante la ecuación [4.39] se redondean a los valores enteros de 16 bits más próximos. Al redondear se comete un error, es por esto que el cálculo se realiza mediante un bucle (*for*) donde los divisores de frecuencia se van calculando sucesivamente, desde el primero al último, teniendo en cuenta el error cometido al redondear el valor anterior, de tal manera de que estos errores no se vayan acumulando.

Obtenidos todos los vectores de divisores de frecuencia, se inicia el movimiento mediante la carga de estos valores en los registros de los temporizadores (de las placas DAQ) correspondientes a cada motor. Los valores se van cargando sucesivamente a medida que se van ejecutando los pulsos, esto se logra gracias a la realimentación de los mismos a través de los puertos digitales (figura 3.3). De esta manera se producen las trayectorias más próximas posibles a las calculadas por el control cinemático, y se logra el movimiento requerido por el usuario.

5. Interfaz de usuario

5.1. Introducción

Para la operación del robot e implementación del control cinemático realizado, se desarrolló una interfaz de usuario denominada "*RobotLab*". De acuerdo a los criterios establecidos para este trabajo, se desarrolló una interfaz fácil de usar, pero práctica y a la vez segura, de tal manera que una persona con conocimientos mínimos de robótica puede aprender rápidamente a operar el robot sin poner en riesgo los componentes del sistema, por ejemplo, en el caso de que el usuario ingrese puntos fuera del área de trabajo, o aceleraciones o velocidades que superen los máximos permitidos por los componentes mecánicos.

5.2. Sistema operativo soporte

De acuerdo a los criterios establecidos, la interfaz (*RobotLab*) se desarrolló para correr bajo sistemas operativos *Windows*, ya que estos están ampliamente difundidos y son muy amigables para usuarios no expertos.

La interfaz puede ser utilizada en modo fuera de línea en cualquier sistema operativo *Windows*, como se verá más adelante, pero en la PC de control del robot se utiliza el sistema operativo *Windows 98*, ya que, a diferencia de los sistemas operativos *Windows* posteriores, es menos restrictivo en cuanto al manejo de puertos y permite la deshabilitación y habilitación de interrupciones por programas en modo usuario[4], requisito fundamental para el correcto funcionamiento del control de motores desarrollado.

5.3. Lenguaje informático y entorno de desarrollo

Para el desarrollo de este proyecto se estableció como criterio la utilización del lenguaje de programación *C++*, ya que este es un lenguaje potente, flexible y ampliamente difundido y utilizado en el mundo entero en proyectos de desarrollo tecnológico de esta índole.

Para la creación de la interfaz de usuario, se utilizó el programa *C++ Builder 6* de *Borland*, debido a su facilidad de utilización y a que permite la programación en lenguaje de programación *C++*.

[4] El **modo usuario** es un modo menos privilegiado de funcionamiento de los programas, sin el acceso directo al hardware. El código que corre en este modo sólo actúa en su propio espacio de dirección. Este usa las APIs (System Application Program Interfaces) para pedir los servicios del sistema.

5.4. RobotLab

El control del robot a través de *RobotLab* se basa en la utilización de un lenguaje textual de comandos y parámetros, especialmente desarrollado para este trabajo y con esa finalidad.

La interfaz desarrollada cuenta con una ventana principal, que se abre al ejecutar el programa (*RobotLab.exe*) y una ventana para la edición de programas.

5.4.1. Ventana principal

La ventana principal, ver figura 5.1, sirve para introducción y ejecución individual de comandos, así como también para la apertura de programas en el editor de programas.

En la ventana principal se observan, una sub-ventana llamada "*Ventana de comandos*", adonde se introducen y ejecutan los comandos de manera individual, una sub-ventana llamada "*Historial*", adonde se van almacenando los comandos ejecutados, y un menú que cuenta con las opciones "*Archivo*" y "*Ayuda*".

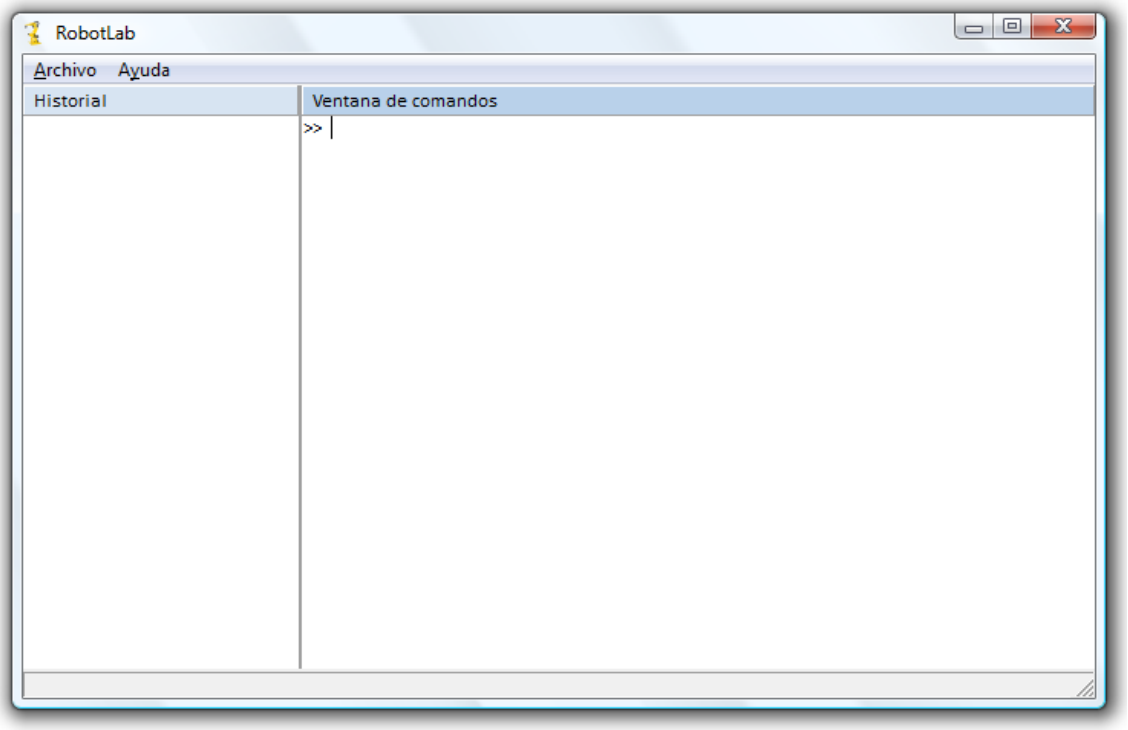


Figura 5.1. Ventana Principal.

Menú Archivo

Mediante el menú "*Archivo*" se accede a las opciones "*Nuevo*", "*Abrir*" y "*Salir*", ver figura 5.2.

Figura 5.2. Menú "*Archivo*" desplegado.

Estas opciones realizan las siguientes acciones:

Nuevo Abre un nuevo programa en una nueva ventana del editor de programas.

Abrir Abre un programa previamente guardado en una nueva ventana del editor de programas.

Salir Termina la ejecución de *RobotLab*.

Menú Ayuda

Mediante el menú "*Ayuda*" se accede a la opción "*Ayuda*", ver figura 5.3.

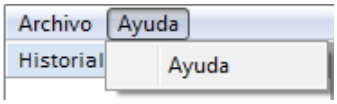


Figura 5.3. Menú "*Ayuda*" desplegado.

A través de esta opción se despliega la ayuda del programa, donde se describen la interfaz, el lenguaje y los comandos del mismo.

5.4.2. Editor de programas

En el Editor de Programas, ver figura 5.4, se pueden crear, abrir, ejecutar y depurar programas escritos en el lenguaje textual desarrollado.

En la ventana del Editor se observa un gran área blanca destinada a la escritura del programa y a la izquierda de esta se observan los números de línea.

En el menú se observan las opciones "*Archivo*", "*Programa*" y "*Ayuda*".

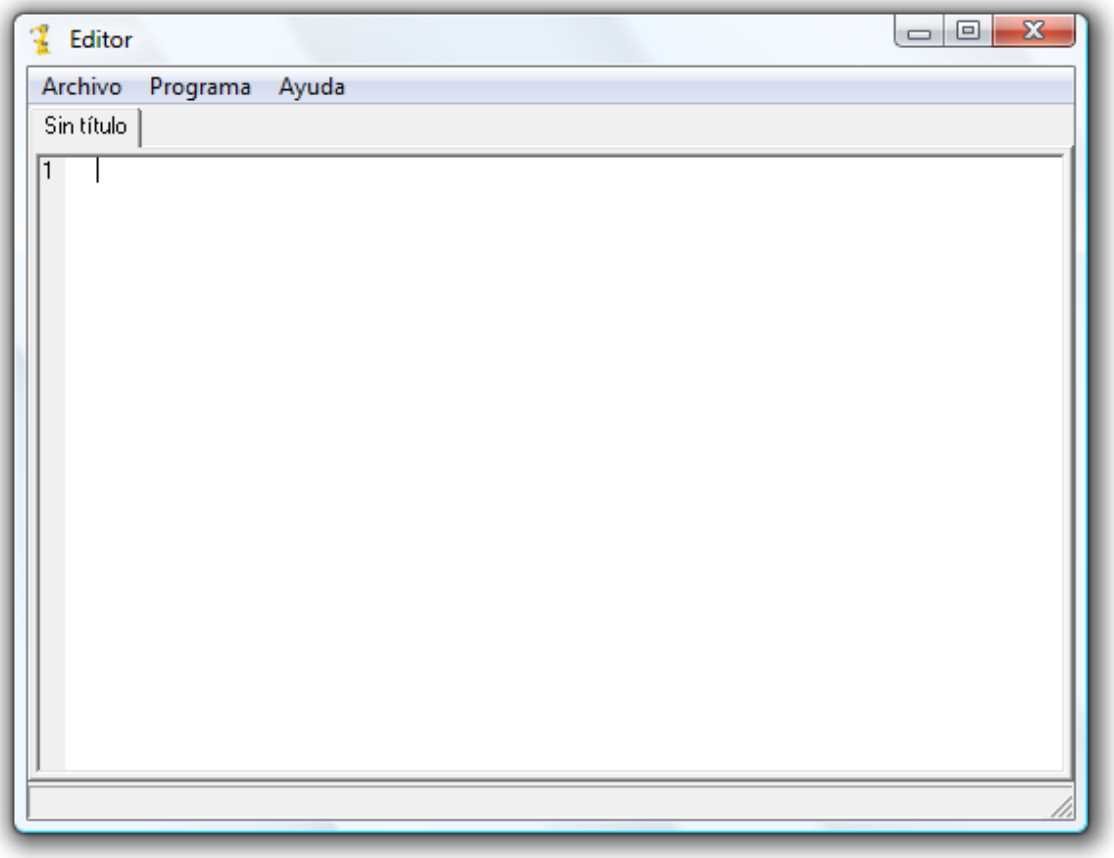


Figura 5.4. Editor de programas.

Menú Archivo

Dentro del menú "*Archivo*", ver figura 5.5, se encuentran las siguientes opciones:

Abrir	Abre un programa previamente creado.
Guardar	Guarda el programa actual.
Guardar Como…	Abre una ventana de diálogo para guardar el programa actual con un nuevo nombre.
Cerrar	Cierra el programa actual, dejando uno nuevo en blanco.
Salir	Cierra el programa actual y la ventana del Editor.

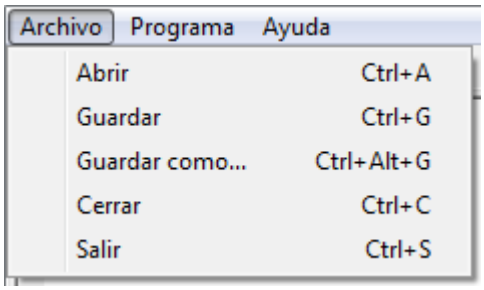


Figura 5.5. Menú "*Archivo*", ventana Editor.

Menú Programa

El menú "*Programa*", ver figura 5.6, sirve para ejecutar y depurar programas, y en este se encuentran las siguientes opciones:

Dar paso — *Para ejecutar el programa línea a línea. Ejecuta la línea señalada por el indicador ">>".*

Ejecutar — *Ejecuta el programa de corrido hasta la última línea.*

Detener — *Detiene la ejecución del programa y vuelve el indicador a la primera línea.*

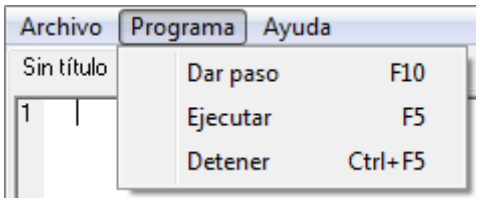


Figura 5.6. Menú "Programa", ventana Editor.

Menu Ayuda

Mediante el menú "*Ayuda*" se accede a la opción "*Ayuda*", ver figura 5.7.

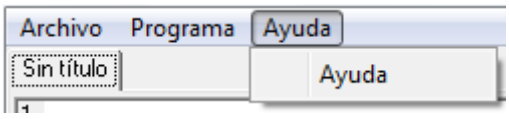


Figura 5.7. Menú "*Ayuda*" desplegado.

A través de esta opción se despliega la ayuda, donde se describen la interfaz, el lenguaje y los comandos del mismo.

5.4.3. Archivo de configuración

Al ejecutar *RobotLab* se carga un archivo de configuración llamado "*config.txt*" donde se definen parámetros tales como frecuencia interna de las placas, direcciones base de las mismas, reducciones de las articulaciones, longitudes de brazos, límites de incursión, velocidades y aceleraciones máximas, datos del conexionado de los motores, etc…

Por seguridad estos valores no pueden ser modificados a través de la interfaz, por lo cual, para modificarlos se debe modificar el archivo y reiniciar *RobotLab* para que los cambios tomen efecto.

5.4.4. Modo fuera de línea

RobotLab puede ser utilizado en modo fuera de línea, esto significa que los comandos de movimiento, en lugar de producir movimientos del robot, muestran un mensaje indicando que el programa se encuentra en modo simulación y que en ese momento se realizaría el movimiento, ver figura 5.8.

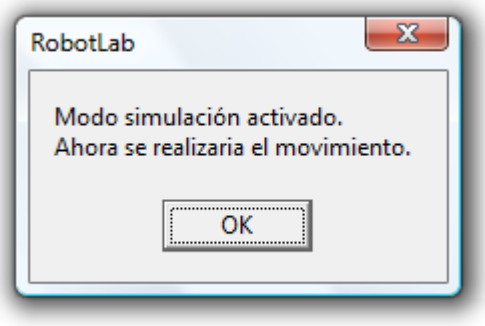


Figura 5.8. Mensaje de simulación de movimiento.

El modo fuera de línea permite que *RobotLab* sea ejecutado en computadoras que no poseen la electrónica de control del robot y que por ende no se encuentren conectadas al mismo, así como también en la propia computadora conectada al robot, pero sin producir movimientos. Esto facilita el aprendizaje y la práctica de la utilización del programa.

La activación del modo fuera de línea se realiza mediante el comando "*simulación*", ver Anexo A.

6. Lenguaje de programación

6.1. Introducción

Para la operación del robot CNX-I se desarrollo un lenguaje textual de programación debido a que esta es una de las formas más utilizadas para la operación de robots industriales.

El lenguaje desarrollado es del tipo de comandos y parámetros. Este es utilizado en la interfaz de usuario *RobotLab* tanto para la ejecución de comandos individuales en la ventana principal, como para la escritura de programas en el editor de programas.

El formato general de cada línea es

comando (lista de parámetros)

donde el comando es obligatorio y la lista de parámetros es opcional, dependiendo del comando.

El sistema de programación es, como para la mayoría de los robots, del tipo interpretado, es decir, que el programa se analiza y ejecuta línea por línea. Esto es muy conveniente ya que programar las acciones de un manipulador requiere de una continua interacción con el entorno, en un continuo proceso de prueba y error, y mediante este sistema se puede realizar un seguimiento línea por línea del programa a medida que se lo va ejecutando o depurando, evitándose de esta manera el laborioso ciclo de editar, compilar y ejecutar, que normalmente es muy costoso en tiempo.

En el editor de programas de *RobotLab* a cada línea de programa se le asigna un número de línea, que se ajusta automáticamente cuando se insertan o borran líneas.

6.2. Comandos

La tabla 6.1 muestra un listado de los comandos con una breve descripción de los mismos. Para una descripción detallada de cada comando ver el Anexo A.

Comando	Descripción
Coordenadas / Calibración	
pos	Muestra o establece la posición del extremo del robot.
ang	Muestra o establece la posición del robot en coordenadas articulares.
Posición de reposo	
home	Muestra las coordenadas de la posición de home.

Comando	Descripción
sethome	Establece la posición de home.
sethomeq	Establece la posición de home en coordenadas angulares.
Variables	
punto	Crea una variable Punto.
num	Crea una variable numérica.
borrar	Borra variables.
Parámetros Cinemáticos	
acel	Muestra o establece la aceleración angular de las articulaciones en [°/s²].
vel	Muestra o establece la velocidad angular de las articulaciones en [°/s].
acel%	Muestra o establece la aceleración angular de las articulaciones en un porcentaje de la aceleración máxima.
vel%	Muestra o establece la velocidad angular de las articulaciones en un porcentaje de la velocidad máxima.
acelmax	Muestra las aceleraciones máximas de las articulaciones.
velmax	Muestra las velocidades máximas de las articulaciones.
Movimientos	
mover	Mueve el robot a la posición requerida.
movimiento	Muestra o establece el tipo de movimiento.
girar	Mueve las articulaciones según los ángulos ingresados.
girarsec	Mueve las articulaciones según los ángulos ingresados con movimiento secuencial.
girarsim	Mueve las articulaciones según los ángulos ingresados con movimiento simultáneo.
girarcoord	Mueve las articulaciones según los ángulos ingresados con movimiento coordinado.
gohome	Mueve el robot a la posición de home.
ir	Mueve el robot a la posición requerida. Ídem *mover*.
ir_q	Mueve el robot a la posición requerida, expresada en coordenadas articulares.
Pinza	
pinza	Muestra o establece el estado de la pinza (abierta o cerrada).
abrirPinza	Abre la pinza.
cerrarPinza	Cierra la pinza.
Pantalla	

Comando	Descripción
mensaje	Despliega una ventana con el mensaje deseado.
mostrar	Muestra texto en la ventana de comandos.
clc	Borra todo el contenido de la ventana de comando.
Acceso a puertos	
in	Lee el puerto deseado.
out	Escribe en el puerto deseado
Extras	
%	Introducción de comentarios.
//	Introducción de comentarios.
detallado	Muestra o establece el modo detallado.
simulacion	Muestra o establece el modo simulación.
simulador	Muestra o establece el estado del simulador.
pausa	Pausa temporalmente la ejecución del programa.
salir	Termina la ejecución de RobotLab.

Tabla 6.1. Breve descripción de los comandos.

6.3. Palabras reservadas[5]

En este lenguaje son palabras reservadas todos los nombres de los comandos y también los nombres de parámetros especiales utilizados por algunos ellos, ver tabla 6.2.

si
no
abrir
cerrar
todo

Tabla 6.2. Parámetros especiales.

6.4. Programa de ejemplo

A continuación, se presenta un programa de ejemplo en el que el robot ha de realizar operaciones de manipulación de un objeto, ver programa 6.1. El robot debe recoger un objeto

[5] Una palabra reservada es una palabra que tiene un significado gramatical especial para un lenguaje y no puede ser utilizada como un identificador.

de la posición A de coordenadas espaciales [35; -10; 18,2; 0] y depositarlo en la posición B de coordenadas espaciales [35; 10; 18,2; 0].

```
% Programa ejemplo de operaciones de manipulación

clc         % Se limpia la ventana de comandos para visualizar mensajes
borrar todo % Se borran todas las variables existentes

% Se establecen los parámetros y datos
num vel_rap = 80
num acel_rap = 100
num vel_len = 20
num acel_len = 40

punto pA          = 35 -10 18.2 0
punto pA_arriba   = 35 -10 28.2 0
punto pB          = 35  10 18.2 0
punto pB_arriba   = 35  10 28.2 0

sethome 0 0 0 0 ang

% Se establecen parámetros cinemáticos.
vel vel_rap
acel acel_rap
movimiento coordinado

% Se avisa que se va a mover el robot.
mensaje ¡Cuidado! Se va a iniciar el movimiento.

% Al inicio se lleva el robot a la posición de home y se abre la pinza
abrirPinza
gohome
pausa 0.5          % Pequeña pausa al alcanzar la posición

% Movimiento rápido para situar el robot sobre el objeto.
mover pA_arriba
pausa 0.5

% Se bajan la aceleración y la velocidad para el acercamiento.
vel vel_len
acel acel_len
mover pA
pausa 0.5

% Se toma el objeto
cerrarPinza
pausa 0.5

% Alejamiento
mover pA_arriba
pausa 0.5

% Ya tomado el objeto se aumentan la aceleración y la velocidad para
% un movimiento rápido hasta la posición sobre el punto de depósito.
vel vel_rap
acel acel_rap
mover pB_arriba
pausa 0.5

% Se vuelven a bajar la velocidad y  aceleración para el acercamiento
vel vel_len
```

```
acel acel_len
mover pB
pausa 0.5

% Se suelta el objeto
abrirPinza
pausa 0.5

% Alejamiento
mover pB_arriba
pausa 0.5

% Tarea terminada, movimiento rápido al home
vel vel_rap
acel acel_rap
gohome

% Mensaje de tarea finalizada
mensaje Tarea finalizada.
```

Programa 6.1. Programa de ejemplo.

En el programa de ejemplo se observa la utilización de valores de aceleración y velocidad elevados para movimientos de posicionamiento rápido y valores reducidos para movimientos de acercamiento. También se observa que antes de iniciar la tarea el robot se moverá desde donde se encuentre a la posición de reposo, y que terminada la misma vuelve a esa posición.

7. Conclusiones y trabajos futuros

7.1. Conclusiones

Se ha logrado un primer sistema de control completo para el robot CXN-I, el mismo incluye el diseño e implementación de la arquitectura y de la estrategia de control, el desarrollo del software de control cinemático, de una interfaz de usuario (*RobotLab*) y de un lenguaje textual de programación muy fáciles de aprender y usar, y con características similares a los utilizados en robots comerciales.

Se ha logrado un control punto a punto que cumple los objetivos establecidos además de contar con la posibilidad de realizar movimientos secuenciales, simultáneos y coordinados.

Gracias a la facilidad de aprendizaje y de uso, tanto de la interfaz de usuario como del lenguaje de programación desarrollados, y a su similitud con los utilizados por robots comerciales, estos son adecuados para su uso didáctico, como un primer acercamiento a la programación de robots.

El sistema desarrollado permite ser adaptado o tomado como base para el control de otros robots o inclusive máquinas de varios ejes.

El software se ha desarrollado de tal manera que mediante relativamente simples modificaciones se le puede dar mayor funcionalidad, implementar nuevos tipos de controles, agregar nuevos comandos y ampliar la funcionalidad de los ya existentes.

7.2. Trabajos futuros

Del desarrollo de este trabajo y de las conclusiones obtenidas surgen propuestas de trabajos futuros tendientes a mejorar el desempeño, la funcionalidad y la seguridad del sistema, a continuación se mencionan algunas de ellas.

Una propuesta de trabajo futuro es el desarrollo de un control de los motores con hardware dedicado (microcontroladores, DSP, FPGA, etc.) con comunicación con el software de control en la PC, mejorando de esta manera el desempeño y la precisión del control de los motores y haciéndolo independiente del sistema operativo o la plataforma utilizada.

La precisión y seguridad podrían ser mejoradas equipando al robot con sensores de posición (encoders o resolvers) y finales de carrera en cada articulación.

Aprovechando la flexibilidad del software desarrollado, podría aumentarse la funcionalidad del robot mediante el diseño e implementación de nuevos tipos de controles cinemáticos, como por ejemplo, de trayectorias continuas y otros.

La interfaz podría ser mejorada mediante la incorporación de barras de herramientas con botones de acceso rápido, la ejecución de comandos mediante ventanas de diálogo, la visualización de información útil en la barra de estado (coordenadas instantáneas, tipo de movimiento, estado de la pinza, etc.), el desarrollo de un simulador gráfico y de un teach pendant (comando manual) virtual, etc.

La funcionalidad del lenguaje desarrollado podría ser aumentada mediante la incorporación de nuevos comandos y la adición de mayor funcionalidad a los ya existentes. También, mediante la incorporación de sentencias condicionales (if-else), de estructuras de control de bucle (for-next, do-while, etc), de llamadas a subrutinas, módulos y tareas.

Bibliografía

ANGULO, J.M. (1986). "*Robótica Práctica*". España: Ed. Paraninfo.

AXIAL electrónica. "*PCX I/O Adquisición de Datos & Control – Manual Técnico*".

BARRIENTOS, A.; PEÑIN, L.; BALAGUER, C.; ARACIL, R. (2007). "*Fundamentos de Robótica, 2.ª Edición*". España: Ed. McGraw-Hill.

BRONSON, G.J. (2007). "*C++ para ingeniería y ciencias. Segunda edición*". México: Ed. Thomson.

CHARTE OJEDA, F. (2006). "*Programación con C++ Builder 2006*". España: Ed. Anaya Multimedia.

CRAIG, J.J. (1989). "*Introduction to Robotics Mechanics and Control*". USA: Ed. Addison-Wesley.

FU, K.S.; GONZÁLES, R.C.; LEE, C.S.G. (1990). "*Robótica: Control, Detección, Visión e Inteligencia*". Madrid: Ed. McGraw-Hill.

Intel (1986). Hoja de datos: "*8253/8253-5 Programmable Internal Timer*".

Intel (1991). Hoja de datos: "*8255A/8255A-5 Programmable Peripheral Interface*".

KELLY, R.; SANTIBÁNEZ, V. (2003). "*Control de Movimiento de Robots Manipuladores*". Madrid: Ed. Pearson Educación, S. A.

KÜNNING, F.G.; CUELLO, J.A.; MORÁN, O.D. "*Desarrollo de un sistema de control para los motores de un robot*". En: Actas del XX° Congreso Argentino de Control Automático. (Buenos Aires, 28 al 30 de Agosto de 2006).

RTA (2008). "*RTA Stepping Motor Drives Catalogue*".

RTA (2008). "*Stepping Motors Catalogue*".

Anexo A. Comandos

A continuación se presentan los comandos del lenguaje desarrollado, sus formas de sintaxis y una descripción de las mismas.

Las unidades utilizadas por el programa son, para coordenadas cartesianas el cm (centímetro) y para coordenadas articulares el grado.

A.1. Coordenadas / Calibración

A.1.1. pos

Muestra o establece la posición del extremo del robot.

Sintaxis

pos

pos *x y z* ϑ_4

pos ϑ_1 ϑ_2 ϑ_3 ϑ_4 *ang*

Descripción

pos muestra la posición actual del extremo del robot.

pos *x y z* ϑ_4 establece la posición actual del extremo del robot en las coordenadas (*x y z* ϑ_4) ingresadas.

pos ϑ_1 ϑ_2 ϑ_3 ϑ_4 *ang* establece la posición actual del extremo del robot en las coordenadas angulares (ϑ_1 ϑ_2 ϑ_3 ϑ_4) ingresadas.

A.1.2. ang

Muestra o establece la posición del extremo del robot en coordenadas angulares.

Sintaxis

ang

ang ϑ_1 ϑ_2 ϑ_3 ϑ_4

Descripción

ang muestra la posición actual del extremo del robot.

ang ϑ_1 ϑ_2 ϑ_3 ϑ_4 establece la posición actual del extremo del robot en las coordenadas angulares (ϑ_1 ϑ_2 ϑ_3 ϑ_4) ingresadas.

A.2. Posición de reposo

A.2.1. home

Muestra las coordenadas de la posición de home.

Sintaxis

home

Descripción

home muestra las coordenadas de la posición de home.

A.2.2. sethome

Establece la posición de home.

Sintaxis

sethome

sethome $x\ y\ z\ \vartheta_4$

sethome $\vartheta_1\ \vartheta_2\ \vartheta_3\ \vartheta_4\ ang$

Descripción

sethome establece la posición de home en la posición actual del robot.

sethome $x\ y\ z\ \vartheta_4$ establece la posición de home en las coordenadas ($x\ y\ z\ \vartheta_4$) ingresadas. $x\ y\ z$ se deben ingresar en cm.

sethome $\vartheta_1\ \vartheta_2\ \vartheta_3\ \vartheta_4\ ang$ establece la posición de home en las coordenadas angulares ($\vartheta_1\ \vartheta_2\ \vartheta_3\ \vartheta_4$) ingresadas.

A.2.3. sethomeq

Establece la posición de home en coordenadas angulares.

Sintaxis

sethomeq

sethomeq $\vartheta_1\ \vartheta_2\ \vartheta_3\ \vartheta_4$

Descripción

sethomeq establece la posición de home en la posición actual del robot.

sethomeq $\vartheta_1\ \vartheta_2\ \vartheta_3\ \vartheta_4$ establece la posición de home en las coordenadas angulares ($\vartheta_1\ \vartheta_2\ \vartheta_3\ \vartheta_4$) ingresadas.

A.3. Variables

A.3.1. punto

Crea una variable "punto".

Sintaxis

punto P1

punto P1 = x y z ϑ_4

punto P1 = ϑ_1 ϑ_2 ϑ_3 ϑ_4 ang

Descripción

punto P1 crea la variable *P1*, del tipo *punto.*

punto P1 = x y z ϑ_4 crea la variable *punto P1* con coordenadas espaciales *x y z ϑ_4.*

punto P1 = ϑ_1 ϑ_2 ϑ_3 ϑ_4 ang crea la variable *punto P1* con coordenadas angulares ϑ_1 ϑ_2 ϑ_3 ϑ_4.

A.3.2. num

Crea una variable numérica.

Sintaxis

num N1

num N1 = valor

Descripción

num N1 crea la variable *N1*, del tipo *num.*

num N1 = valor crea la variable *N1*, del tipo *num* y le asigna el valor "*valor*".

A.3.3. borrar

Borra variables.

Sintaxis

borrar var1 var2 … varN

borrar todo

Descripción

borrar var1 var2 … varN borra las variables *var1, var2, …, varN.* Las variables pueden ser "*num*" o "*punto*".

borrar todo borra todas las variables.

A.4. Parámetros cinemáticos

A.4.1. acel

Muestra o establece la aceleración angular de las articulaciones en [°/s²].

Sintaxis

acel

acel valor

acel eje valor

acel valor1 valor2 valor3 valor4

Descripción

acel muestra las aceleraciones angulares de las articulaciones, en [°/s²].

acel valor establece las aceleraciones angulares de todas las articulaciones en el "*valor*" ingresado, en [°/s²].

acel eje valor establece la aceleración angular de la articulación "*eje*" en el "*valor*" ingresado, en [°/s²].

acel valor1 valor2 valor3 valor4 establece las aceleraciones angulares de las articulaciones en los valores ingresados, en [°/s²].

A.4.2. vel

Muestra o establece la velocidad angular de las articulaciones en [°/s].

Sintaxis

vel

vel valor

vel eje valor

vel valor1 valor2 valor3 valor4

Descripción

vel muestra las velocidades angulares de las articulaciones, en [°/s].

vel valor establece las velocidades angulares de todas las articulaciones en el "*valor*" ingresado, en [°/s].

vel eje valor establece la velocidad angular de la articulación "*eje*" en el "*valor*" ingresado, en [°/s].

vel valor1 valor2 valor3 valor4 establece las velocidades angulares de las articulaciones en los valores ingresados, en [°/s].

A.4.3. acel%

Muestra o establece la aceleración angular de las articulaciones en un porcentaje de la aceleración máxima.

Sintaxis

acel%

acel% valor

acel% eje valor

acel% valor1 valor2 valor3 valor4

Descripción

acel% muestra las aceleraciones angulares de las articulaciones en valores porcentuales de las aceleraciones máximas.

acel% valor establece las aceleraciones angulares de todas las articulaciones en un valor porcentual, dado por *valor*, de las aceleraciones máximas correspondientes.

acel% eje valor establece la aceleración angular de la articulación "*eje*" en un valor porcentual, dado por *valor*, de la aceleración máxima de ese eje.

acel% valor1 valor2 valor3 valor4 establece las aceleraciones angulares de las articulaciones en los valores porcentuales (*valor1, valor2, valor3, valor4*) ingresados, en [°/s²].

A.4.4. vel%

Muestra o establece la velocidad angular de las articulaciones en un porcentaje de la velocidad máxima.

Sintaxis

vel%

vel% valor

vel% eje valor

vel% valor1 valor2 valor3 valor4

Descripción

vel% muestra las velocidades angulares de las articulaciones en valores porcentuales de las velocidades máximas.

vel% valor establece las velocidades angulares de todas las articulaciones en un valor porcentual, dado por *valor*, de las velocidades máximas correspondientes.

vel% eje valor establece la velocidad angular de la articulación "*eje*" en un valor porcentual, dado por *valor*, de la velocidad máxima de ese eje.

vel% valor1 valor2 valor3 valor4 establece las velocidades angulares de las articulaciones en los valores porcentuales (*valor1*, *valor2*, *valor3*, *valor4*) ingresados, en [°/s²].

A.4.5. acelmax

Muestra las aceleraciones máximas de las articulaciones.

Sintaxis

acelmax

Descripción

acelmax muestra las aceleraciones máximas de las articulaciones.

A.4.6. velmax

Muestra las velocidades máximas de las articulaciones.

Sintaxis

velmax

Descripción

velmax muestra las velocidades máximas de las articulaciones.

A.5. Movimientos

A.5.1. mover

Mueve el robot a la posición requerida.

Sintaxis

mover punto

mover x y z ϑ_4

mover ϑ_1 ϑ_2 ϑ_3 ϑ_4 *ang*

Descripción

mover punto mueve el robot a la posición de la variable punto.

mover x y z ϑ_4 mueve el robot a las coordenadas espaciales *x y z* ϑ_4.

mover ϑ_1 ϑ_2 ϑ_3 ϑ_4 *ang* mueve el robot a las coordenadas angulares ϑ_1 ϑ_2 ϑ_3 ϑ_4.

A.5.2. movimiento

Muestra o establece el tipo de movimiento.

Sintaxis

movimiento

movimiento secuencial

movimiento simultaneo

movimiento coordinado

Descripción

movimiento muestra el tipo de movimiento establecido.

movimiento secuencial establece el tipo de movimiento en secuencial.

movimiento simultaneo establece el tipo de movimiento en simultaneo.

movimiento coordinado establece el tipo de movimiento en coordinado.

A.5.3. girar

Mueve las articulaciones según los ángulos ingresados.

Sintaxis

girar $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$

girar $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel acel*

girar $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel1 vel2 vel3 vel4 acel1 acel2 acel3 acel4*

Descripción

girar $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ mueve las articulaciones del robot en los ángulos ingresados.

girar $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel acel* mueve las articulaciones del robot en los ángulos ingresados, con la velocidad (*vel*) y aceleración (*acel*) ingresadas.

girar $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel1 vel2 vel3 vel4 acel1 acel2 acel3 acel4* mueve las articulaciones del robot en los ángulos ingresados, con las velocidades (*vel1, vel2, vel3, vel4*) y las aceleraciones (*acel1, acel2, acel3, acel4*) ingresadas para cada motor.

A.5.4. girarsec

Mueve las articulaciones según los ángulos ingresados, con movimiento secuencial.

Sintaxis

girarsec $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$

girarsec $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel acel*

girarsec $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel1 vel2 vel3 vel4 acel1 acel2 acel3 acel4*

Descripción

girarsec $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ mueve las articulaciones del robot en los ángulos ingresados, con movimiento secuencial.

girarsec $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel acel* mueve las articulaciones del robot en los ángulos ingresados, con la velocidad (*vel*) y aceleración (*acel*) ingresadas, y con movimiento secuencial.

girarsec $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel1 vel2 vel3 vel4 acel1 acel2 acel3 acel4* mueve las articulaciones del robot en los ángulos ingresados, con las velocidades (*vel1*, *vel2*, *vel3*, *vel4*) y las aceleraciones (*acel1*, *acel2*, *acel3*, *acel4*) ingresadas para cada motor, y con movimiento secuencial.

A.5.5. girarsim

Mueve las articulaciones según los ángulos ingresados con movimiento simultáneo.

Sintaxis

girarsim $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$

girarsim $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel acel*

girarsim $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel1 vel2 vel3 vel4 acel1 acel2 acel3 acel4*

Descripción

girarsim $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ mueve las articulaciones del robot en los ángulos ingresados, con movimiento simultáneo.

girarsim $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel acel* mueve las articulaciones del robot en los ángulos ingresados, con la velocidad (*vel*) y aceleración (*acel*) ingresadas, y con movimiento simultáneo.

girarsim $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel1 vel2 vel3 vel4 acel1 acel2 acel3 acel4* mueve las articulaciones del robot en los ángulos ingresados, con las velocidades (*vel1*, *vel2*, *vel3*, *vel4*) y las aceleraciones (*acel1*, *acel2*, *acel3*, *acel4*) ingresadas para cada motor, y con movimiento simultáneo.

A.5.6. girarcoord

Mueve las articulaciones según los ángulos ingresados con movimiento coordinado.

Sintaxis

girarcoord $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$

girarcoord $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel acel*

girarcoord $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel1 vel2 vel3 vel4 acel1 acel2 acel3 acel4*

Descripción

girarcoord $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ mueve las articulaciones del robot en los ángulos ingresados, con movimiento coordinado.

girarcoord $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel acel* mueve las articulaciones del robot en los ángulos ingresados, con la velocidad (*vel*) y aceleración (*acel*) ingresadas, y con movimiento coordinado.

girarcoord $\Delta\vartheta_1$ $\Delta\vartheta_2$ $\Delta\vartheta_3$ $\Delta\vartheta_4$ *vel1 vel2 vel3 vel4 acel1 acel2 acel3 acel4* mueve las articulaciones del robot en los ángulos ingresados, con las velocidades (*vel1*, *vel2*, *vel3*, *vel4*) y las aceleraciones (*acel1*, *acel2*, *acel3*, *acel4*) ingresadas para cada motor, y con movimiento coordinado.

A.5.7. gohome

Mueve el robot a la posición de home.

Sintaxis

gohome

Descripción

gohome mueve el robot a la posición de home.

A.5.8. ir

Mueve el robot a la posición requerida. Ídem mover.

Sintaxis

ir *punto*

ir *x y z* ϑ_4

ir ϑ_1 ϑ_2 ϑ_3 ϑ_4 *ang*

Descripción

ir *punto* mueve el robot a la posición de la variable punto.

ir *x y z* ϑ_4 mueve el robot a las coordenadas espaciales *x y z* ϑ_4.

ir ϑ_1 ϑ_2 ϑ_3 ϑ_4 *ang* mueve el robot a las coordenadas angulares ϑ_1 ϑ_2 ϑ_3 ϑ_4.

A.5.9. ir_q

Mueve el robot a la posición requerida, expresada en coordenadas articulares.

Sintaxis

ir_q ϑ_1 ϑ_2 ϑ_3 ϑ_4

Descripción

ir_q ϑ_1 ϑ_2 ϑ_3 ϑ_4 mueve el robot a las coordenadas angulares ϑ_1 ϑ_2 ϑ_3 ϑ_4.

A.6. Pinza

A.6.1. pinza

Muestra o establece el estado de la pinza ("*abierta*" o "*cerrada*").

Sintaxis

pinza

pinza *abrir*

pinza *cerrar*

Descripción

pinza muestra el estado de la pinza ("*abierta*" o "*cerrada*").

pinza *abrir* abre la pinza.

pinza *cerrar* cierra la pinza.

A.6.2. abrirPinza

Abre la pinza.

Sintaxis

abrirPinza

Descripción

abrirPinza abre la pinza.

A.6.3. cerrarPinza

Cierra la pinza.

Sintaxis

cerrarPinza

Descripción

cerrarPinza cierra la pinza.

A.7. Pantalla

A.7.1. mensaje

Despliega una ventana con el mensaje deseado.

Sintaxis

mensaje *mensaje a mostrar*

Descripción

mensaje *mensaje a mostrar* despliega una ventana con el mensaje "*mensaje a mostrar*". La ejecución del programa se detiene hasta que el usuario acepte el mensaje.

A.7.2. mostrar

Muestra texto en la ventana de comandos.

Sintaxis

mostrar *mensaje a mostrar*

Descripción

mostrar *mensaje a mostrar* muestra el mensaje "*mensaje a mostrar*" en la ventana de comandos. La ejecución del programa no se detiene.

A.7.3. clc

Borra todo el contenido de la ventana de comando.

Sintaxis

clc

Descripción

clc borra todo el contenido de la ventana de comandos.

A.8. Acceso a puertos

A.8.1. in

Lee el puerto deseado.

Sintaxis

in puerto

Descripción

in puerto muestra en la ventana de comandos el valor leído del puerto ingresado.

A.8.2. out

Escribe en el puerto deseado

Sintaxis

out puerto dato

Descripción

out puerto dato escribe el dato en el puerto ingresado.

A.9. Extras

A.9.1. %

Introducción de comentarios. Ídem //.

Sintaxis

% comentarios

comando argumentos % comentarios

Descripción

Utilizando el signo *%* se pueden introducir comentarios en el código de los programas. A partir del signo *%* la línea no es analizada en busca de comandos o argumentos. Ídem //.

A.9.2. //

Introducción de comentarios. Ídem *%.*

Sintaxis

// comentarios

comando argumentos // comentarios

Descripción

Utilizando // se pueden introducir comentarios en el código de los programas. A partir de // la línea no es analizada en busca de comandos o argumentos. Ídem *%*.

A.9.3. detallado

Muestra o establece el modo detallado.

Sintaxis

detallado

detallado *si*

detallado *no*

Descripción

detallado muestra el estado del modo detallado (“*si*” o “*no*”).

detallado *si* activa el modo detallado.

detallado *no* desactiva el modo detallado.

El modo detallado, cuando está activado, despliega convenientemente ventanas de diálogo mostrando valores relevantes del funcionamiento del programa.

A.9.4. simulacion

Muestra o establece el modo simulación.

Sintaxis

simulacion

simulacion *si*

simulacion *no*

Descripción

simulacion muestra el estado del modo detallado (“*si*” o “*no*”).

simulacion *si* activa el modo detallado.

simulacion *no* desactiva el modo detallado.

En modo simulación los comandos de movimiento, en lugar de producir movimientos reales del robot, muestran un mensaje indicando que el programa se encuentra en modo simulación y que en ese momento se realizaría el movimiento, ver figura A.1.

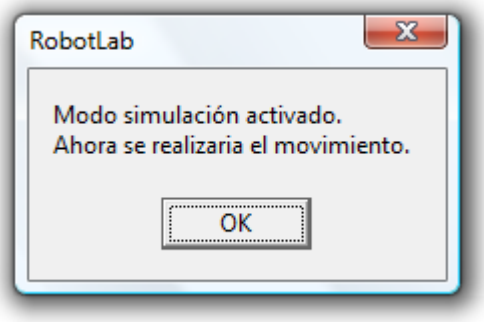


Figura A.1. Mensaje de simulación de movimiento.

A.9.5. simulador

Muestra o establece el estado del simulador.

Sintaxis

simulador

simulador *si*

simulador *no*

Descripción

simulador muestra el estado del modo simulador (*si* o *no*).

simulador *si* activa el modo simulador.

simulador *no* desactiva el modo simulador.

Mediante este comando se le indica a *RobotLab* si el simulador del robot CXN-I, realizado en *Matlab*, se está ejecutando o no. Cuando el estado de *simulador* es *si*, *RobotLab* en lugar de leer y escribir los puertos de las placas de adquisición y control de datos, lee y escribe a un archivo compartido con el programa del simulador, utilizado por este último para simular el comportamiento del robot CXN-I.

En el caso de que el simulador se encuentre inactivo, el estado de *simulador* debe ser *no*.

A.9.6. pausa

Pausa temporalmente la ejecución del programa.

Sintaxis

pausa

pausa *n*

Descripción

pausa pausa la ejecución del programa y despliega una ventana de diálogo indicativa. El programa permanece pausado hasta que se presiona aceptar en la ventana de diálogo.

pausa n pausa la ejecución del programa durante *n* segundos.

A.9.7. salir

Termina la ejecución de *RobotLab*.

Sintaxis

salir

Descripción

salir termina la ejecución de RobotLab.

También se puede terminar la ejecución de *RobotLab* seleccionando **Archivo > Salir**.

Anexo B. Disco Compacto

A este documento se adjunta un CD con el programa *RobotLab* y archivos necesarios para su funcionamiento. También se encuentran en el CD los archivos del código fuente y archivos necesarios para el desarrollo del programa en el entorno *C++ Builder 6*.