UNIVERSIDAD AUTÓNOMA DE SAN LUIS POTOSÍ

Facultad de Ingeniería

CENTRO DE INVESTIGACIÓN Y ESTUDIOS DE POSGRADO

Maestría en Ingeniería Eléctrica

"Desarrollo de Software para Control de Impedancia de Robots Industriales"

TESIS

Que para obtener el grado de

Maestra en Ingeniería Eléctrica

Presenta:

Ing. Lina Nohemí Rojas García

Asesor:

Dr. Emilio Jorge González Galván

San Luis Potosí, S.L.P., Febrero de 2012

UNIVERSIDAD AUTÓNOMA DE SAN LUIS POTOSÍ

Facultad de Ingeniería

CENTRO DE INVESTIGACIÓN Y ESTUDIOS DE POSGRADO

Maestría en Ingeniería Eléctrica

Tesis:

"Desarrollo de Software para Control de Impedancia de Robots Industriales"

Presenta:

Ing. Lina Nohemí Rojas García

Sinodales:

Dr. Emilio Jorge González Galván
Asesor

Dra. Elvia Ruth Palacios Hernández
Revisor

Dr. Mauro Eduardo Maya Méndez
Revisor

Dr. Raúl Eduardo Balderas Navarro
Revisor

Agradecimientos

*A **Dios**, por la vida, el amor y la hermosa familia que me ha dado.*

*A mi esposo **Héctor Daniel**, por todo lo que inspira en mí y por ser siempre el aire bajo mis alas.*

*A mi hija **Dánae**, por ser mi razón de vivir y de superarme cada día.*

*A mis padres **Ramona y Jesús**, por hacer de mí la persona que soy ahora.*

*A mis hermanas **Silvia y Azucena**, y a mi hermano **Omar**, por brindarme su apoyo incondicional, su amor, su amistad y su compañía sincera.*

*A mi tíos **Catalina y Juan**, y a mi suegros **Carmen y Gerardo**, por siempre cuidar de mi, como si fuera su hija.*

*A mis **Amigas** y **Amigos**, por los momentos de alegría, por acompañarme en los de tristeza, por su confianza, sus consejos y estar ahí siempre que los necesito.*

*A los **Doctores del CIEP**, por compartir sus conocimientos.*

*A los Doctores **Emilio González** y **Ambrocio Loredo**, por su inspiración, conocimiento, talento y paciencia. Sin lugar a dudas, ésta tesis es para ustedes.*

*A **CONACyT**, por la beca otorgada.*

*Con amor, respeto y admiración...**Gracias**.*

Resumen

Algunos enfoques de control de impedancia han sido propuestos y desarrollados en robots industriales utilizando software que no se ejecuta directamente en el controlador del robot. Sin embargo, en la implementación de un controlador de impedancia en un robot industrial con arquitectura cerrada, los retrasos de comunicación entre éste y el equipo de cómputo donde el control es implementado, pueden dar lugar a desempeños inaceptables en tareas industriales o aquellas que involucran la interacción con humanos.

Con el fin de reducir el retraso en la comunicación entre dispositivos, se considera viable el desarrollo de algoritmos de control dentro del controlador del robot, utilizando el lenguaje de programación propio de este dispositivo. Esta afirmación está basada en el conocimiento de que el controlador tiene la capacidad suficiente para poder ejecutar dichos algoritmos. El desarrollo de este trabajo se realiza para ser utilizado en el controlador R30iA Mate de un robot FANUC modelo LR Mate-200iC, que ejecuta programas de tipo p-code provenientes del lenguaje de programación FANUC KAREL. Este lenguaje es suficientemente potente para el manejo de manipulaciones robóticas y sus características reducen substancialmente el desarrollo, prueba y tiempo de implementación del algoritmo de control, debido a que no existe comunicación con dispositivos externos.

Este trabajo genera grandes expectativas para ser enfocado a tareas de rehabilitación que implican interacción humano-robot, las cuales presentan actualmente requerimientos de tecnología robótica a bajos costos, aspirando a atender dicha demanda con los recursos que ofrecen los sistemas robóticos comerciales.

Índice general

Índice de figuras

Índice de tablas

Introducción

Automatizar ha venido a significar, de manera cada vez más generalizada, la introducción de robots en diferentes áreas de desarrollo. En la actualidad el uso de robots representa la forma más avanzada de automatización, no obstante, no sólo en esta área se ha desarrollado la investigación en robótica. La incursión de los robots se ha extendido a diversas áreas, incluida entre ellas el área médica, ya sea en cirugía, rehabilitación, o tareas de ayuda para personas con capacidades diferentes. Todo esto debido a que proveen ventajas como repetibilidad, precisión, además del mejoramiento significativo de la habilidad humana para desempeñar tareas importantes.

Desde hace tres décadas, el denominado ***control de interacción*** ha presentado un constante desarrollo en cuanto a ambientes no estructurados se refiere; surgiendo de la creciente complejidad de tareas que requieren contacto del manipulador con su entorno, como por ejemplo la interacción física entre los robots y los seres humanos. Inicialmente la investigación en este ámbito se enfocó en la operación de objetos a distancia, por medio de técnicas maestro-esclavo. Posteriormente se desarrolló el control de interacción entre un manipulador y un objeto, debido a la presión que ejerce el primero sobre el segundo al realizar una maniobra para sujetarlo.

A la par, se desarrollaron investigaciones tales como el ***control de impedancia***, el cual pretende caracterizar al robot con un comportamiento igual al de una impedancia mecánica, y especificar los parámetros de comportamiento de dicha impedancia según la tarea o superficie de operación sobre la cual el robot será manipulado. En este ámbito se han desarrollado ya diversos mecanismos enfocados al área de rehabilitación, área en la cual se pretende tenga su principal desarrollo el presente proyecto de tesis.

La tendencia actual de incluir a los mecanismos robóticos en terapias de rehabilitación está en aumento. Sin embargo, es necesario un constante trabajo en el desarrollo de dicha investigación para generar nuevas soluciones de hardware y software en esta área. La evidencia clínica ha mostrado que la terapia robótica es efectiva y proporciona beneficios duraderos, esto se ha demostrado en

múltiples estudios clínicos que ya han sido reportados por ejemplo en [13] y [16].

Existen diversas áreas de investigación al respecto, tales como las presentadas por [12], donde se rehabilita la muñeca, en [22] se da rehabilitación al brazo utilizando un robot WREX, reportando avances significativos en cuanto a movilidad. En [17], se rehabilita el tobillo por medio del modelo robótico *MIT's ankle robot system*, basado en el modelo MIT-MANUS desarrollado para extremidades superiores y en [6] se genera una estrategia de rehabilitación basada en el funcionamiento del *Lokoman*. Una desventaja que presentan los robots creados específicamente para tareas en el área de rehabilitación, es que son sistemas con costos elevados, debido a la especialización en la que se enfocan.

Actualmente, en el mercado existen diferentes compañías dedicadas al diseño, ensamble y venta de robots industriales logrando que, aunque el precio de mercado de estos brazos robóticos es considerable, en este momento su costo es menor a los costos de cualquier robot creado con fines médicos. Es sobre los robots industriales donde se busca el desarrollo de nuevos enfoques de utilización de los mismos ya que, debido a su versatilidad, pueden realizar más tareas de las que imaginaron sus propios creadores, penetrando así con las adaptaciones necesarias en áreas de rehabilitación a menores costos y con terapia robótica efectiva que proporcione beneficios duraderos.

Lenguaje de Programación FANUC-KAREL

Un sistema robótico FANUC consiste de un robot, un controlador y un sistema de software. KAREL es el lenguaje de programación de la marca FANUC, en el cual pueden ser desarrolladas diferentes tareas industriales, tales como generación de movimientos, control, comunicación con equipos externos, e incluso interacción con el operador. KAREL contiene características de lenguajes de alto nivel, e incorpora además aplicaciones especialmente desarrolladas para manipulaciones robóticas. Algunas de las principales son:

- Tipos de datos simples y estructurados.
- Aritmética, operadores relacionales y booleanos.
- Estructuras de control para bucles y selecciones.
- Controladores de estado.
- Rutinas de funciones y procedimientos.
- Instrucciones de control de movimiento.
- Operaciones de entrada y salida.
- Multi-programación y apoyo del movimiento simultáneo.

Este lenguaje es suficientemente potente para manejar prácticamente casi cualquier proceso de manipulación, aplicación o montaje robótico y sus características incorporadas reducen substancialmente el desarrollo, prueba y tiempo de implementación. KAREL es recomendado para aplicaciones avanzadas que requieren comunicación por medio de ethernet, procesamiento lógico avanzado, codificación y decodificación de cadenas de caracteres ASCII a binario o decimales, y la manipulación de los datos de archivo, incluido el cálculo de la posición de los robots suministrado por el usuario o por archivos de datos externos.

Planteamiento del problema

En el laboratorio de robótica de la Universidad Autónoma de San Luis Potosí, se implementó un esquema de control de impedancia basado en retroalimentación de posición [3]; el cual fue desarrollado también en un robot industrial de arquitectura cerrada de la marca FANUC, modelo M-16iB/20T. En el desarrollo de la investigación, se utilizó un sensor de fuerza/par ATI FT Gamma, el cual está conectado a una tarjeta de adquisición de datos que envía las señales de fuerza y torque a la computadora para ser procesadas. Posteriormente, por medio del lenguaje de programación de alto nivel C++, se genera la nueva posición del manipulador la cual será comandada al robot por medio de sockets.

El trabajo anterior se desarrolló exitosamente, arrojando resultados alentadores en el área de interacción humano-robot, principalmente en el aspecto terapéutico. Sin embargo en esta implementación existe un retardo de tiempo, debido a la comunicación entre el equipo de cómputo y el robot, calculado en 400 ms aproximadamente. Dicho tiempo de retardo podría ser despreciable para tareas que no estuvieran directamente relacionadas con la interacción con un humano, en algún tipo particular de aplicación como la rehabilitación robótica. Sin embargo, debido a la percepción de los mismos, este retardo se vuelve considerable. Se pretende que, usando el lenguaje de programación KAREL, se disminuya este tiempo de retraso al máximo, para poder realizar tareas que involucren la interacción humano-robot.

Objetivo

El objetivo es realizar una adaptación del control de impedancia basado en posición e implementarlo en un robot industrial, usando el lenguaje de programación KAREL propio del robot. Se buscará habilitar este tipo de control y el uso de un sensor de fuerza instalado en el mismo controlador, reduciendo al máximo los retrasos de tiempo en la comunicación computadora-robot.

Contribuciones

Se espera reducir el tiempo de muestreo del sistema de control de impedancia y, entre las contribuciones generadas directamente del trabajo de investigación, se pueden mencionar las siguientes:

- Desarrollar un algoritmo de control dentro del controlador de un robot industrial, tomando ventaja de las características del lenguaje de programación KAREL.
- Generar un algoritmo de control capaz de reducir el tiempo de muestreo.
- Sintonizar los valores de los parámetros asociados a la impedancia, adecuados para el desarrollo de las pruebas experimentales presentadas.

Organización del documento

La estructura del documento se presenta de la siguiente manera:

En el *Capítulo 1* se describen brevemente los conceptos fundamentales de la cinemática de un robot manipulador. Se caracterizan las ecuaciones que describen el comportamiento del la cinemática directa e inversa de los robots industriales, usados en el proyecto.

En el *Capítulo 2* se describe el estado del arte del control de impedancia, y se presenta el desarrollo del control de impedancia basado en posición realizado para esta investigación. Se presentan también las propuestas desarrolladas para el filtrado de fuerza y un diagrama general del algoritmo de control utilizado.

En el *Capítulo 3* se presenta la plataforma experimental, así como los resultados obtenidos a partir de la simulación y experimentación de los algoritmos de control desarrollados.

Posteriormente se detallan las conclusiones de este trabajo y una visión acerca del trabajo futuro. Para finalizar, se presentan algunos apéndices que ayudarán a complementar información a lo largo de este documento, tales como las especificaciones del sensor de fuerza utilizado, y la de los robots industriales utilizados en la plataforma experimental.

Capítulo 1

Cinemática del Robot Manipulador

1.1. Conceptos Cinemáticos

Se llama *cinemática* al estudio del movimiento un cuerpo sin estudiar las causas que lo producen. Debido a que un mecanismo robótico esta diseñado esencialmente para el movimiento de herramientas y partes, la cinemática del mismo se vuelve fundamental para el diseño, análisis, control y simulación. Es necesario poder predecir, de manera confiable, tanto la posición como la orientación del efector final en términos del valor de los pares cinemáticos. Estos pueden ser de revoluta o prismáticos y se relacionan con cada uno de los grados de libertad del manipulador, para poder tener conocimiento del modelo descriptivo del comportamiento cinemático del manipulador.

Un *mecanismo robótico* puede definirse como un sistema de cuerpos rígidos conectados por medio de juntas, las cuales definen la cinemática de un robot. Ésta puede describir la posición, velocidad y aceleración de cada punto de dicho mecanismo.

1.1.1. Posición y Orientación de un Mecanismo Robótico

La cinemática de un cuerpo rígido es la manera de representar la posición de un cuerpo en un espacio determinado. El mínimo número de coordenadas necesarias para localizar a un cuerpo en el espacio Euclidiano son seis. Es necesario el uso de un marco de referencia coordenado fijo $x_0y_0z_0$ y otro en movimiento $x_iy_iz_i$ para expresar la posición de cada uno de los eslabones del brazo robótico.

1.1.1.1. Posición

La *posición* del origen del marco de referencia $x_i y_i z_i$ relativo al marco de referencia $x_0 y_0 z_0$ pueden ser denotado por un vector $\boldsymbol{P}$, tal que

$$\begin{bmatrix} P_{0x} \\ P_{0y} \\ P_{0z} \end{bmatrix} = \boldsymbol{T}_0^i \begin{bmatrix} P_{ix} \\ P_{iy} \\ P_{iz} \end{bmatrix} \tag{1.1}$$

donde la matriz $T_0^i \in \Re^{3\times 3}$ representa la matriz de transformación de coordenadas del punto $\boldsymbol{P}$ del $x_i y_i z_i$ con respecto al sistema $x_0 y_0 z_0$.

Por otra parte, una *traslación* es un desplazamiento en el cual ningún punto del cuerpo rígido permanece en su posición inicial, y los ejes $x_0 y_0 z_0$ permanecen paralelos a sus orientaciones iniciales.

1.1.1.2. Orientación

La descripción de la orientación de un cuerpo requiere un desarrollo más exhaustivo. Una *rotación* es un desplazamiento en el que al menos un punto del cuerpo rígido permanece en su posición inicial y no todas las alineaciones en el cuerpo permanecerán paralelas a sus orientaciones iniciales. Por ejemplo, un cuerpo en una órbita circular gira alrededor de un eje que pasa por el centro de su trayectoria circular, y cada punto del eje de rotación es un punto en el cuerpo que permanece en su posición inicial. Como en el caso de la posición y la traslación, toda representación de la orientación se puede utilizar para crear una representación de la rotación, y viceversa [19]. Es necesario, para la representación de este movimiento, el uso de matrices de rotación que describan la orientación del marco de coordenado $x_i y_i z_i$ denotado en función del marco de referencia $x_0 y_0 z_0$, y cuya representación es una matriz $\boldsymbol{R} \in \Re^{3\times 3}$.

Existen tres diferentes casos que describen la rotación sobre cada eje como se muestra en la Figura 1.1, y se expresan de la siguiente manera:

- Rotación elemental del marco $x_i y_i z_i$ alrededor del eje x_0 a través de un ángulo θ

$$\boldsymbol{R}_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{1.2}$$

- Rotación elemental del marco $x_i y_i z_i$ alrededor del eje y_0 a través de un ángulo ϕ

$$\boldsymbol{R}_{y,\phi} = \begin{bmatrix} \cos(\phi) & 0 & \sin(\phi) \\ 0 & 1 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) \end{bmatrix} \tag{1.3}$$

- Rotación elemental del marco $x_i y_i z_i$ alrededor del eje z_0 a través de un ángulo ψ

$$\boldsymbol{R}_{z,\psi} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.4}$$

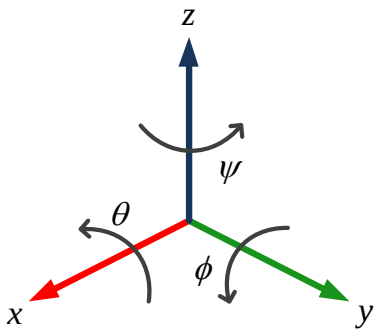


Figura 1.1. Rotaciones sobre los ejes x,y y z

Las matrices anteriores tiene la propiedad de que su inversa es simplemente su transpuesta ya que son matrices ortonormales.

1.1.2. Transformaciones Homogéneas

Una matriz de *transformación homogénea* puede representar tanto la posición como la orientación de un cuerpo con una sola notación. Dicha transformación puede ser representada por medio de una matriz $\boldsymbol{T}$, tal que

$$\boldsymbol{T} = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_y & a_y & d_y \\ n_z & s_z & a_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.5}$$

donde $\boldsymbol{T} \in \Re^{4\times4}$, $\boldsymbol{n} = (n_x, n_y, n_z)^T$ es el vector que representa la dirección del eje x_i referido al sistema $x_0 y_0 z_0$, $\boldsymbol{s} = (s_x, s_y, s_z)^T$ representa la dirección de y_i con respecto a $x_0 y_0 z_0$ y $\boldsymbol{a} = (a_x, a_y, a_z)^T$ representa la dirección de z_i. El vector $\boldsymbol{d} = (d_x, d_y, d_z)^T$ es el vector dirigido desde el origen x_0, y_0, z_0 al origen del sistema $x_i y_i z_i$.

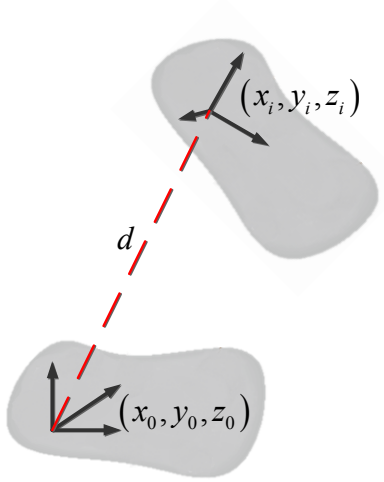


Figura 1.2. Posición inicial y final de un punto arbitrario que experimenta un desplazamiento d

1.1.3. Cinemática Directa

El problema de la *cinemática directa* de un manipulador de cadena serial es encontrar la posición y orientación del efector final relativo a un marco de referencia fijo. Prácticamente, este problema es resuelto calculando la transformación entre el marco de referencia fijo en el efector final y el marco de referencia fijo en la base del manipulador. Esto se desarrolla adjuntando un sistema coordenado de referencia a cada una de los eslabones del robot y calculando las transformaciones homogéneas subsecuentes.

El valor de las variables de junta en un robot de arquitectura cerrada se obtiene de la lectura de los encoders, los cuales pueden estar colocados directamente sobre los ejes de las juntas o en el eje del motor que las acciona. Así, la posición y orientación de la herramienta colocada al extremo del manipulador puede expresarse en función de dichas variables.

Existen métodos para encontrar la cinemática directa, tal como el procedimiento de *Denavit-Hartenberg* [21]. En esta convención, cada transformación homogénea $\boldsymbol{A}_i$ está representada como un producto de 4 transformaciones básicas

$$
\begin{aligned}
\boldsymbol{A}_i &= Rot_{z,\theta_i} Trans_{z,d_i} Trans_{x,a_i} Rot_{x,\alpha_i} \\
&= \begin{bmatrix} c_{\theta i} & s_{\theta i} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha i} & -s_{\alpha i} & 0 \\ 0 & s_{\alpha i} & c_{\alpha i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} c_{\theta i} & -s_{\theta i} c_{\alpha i} & s_{\theta i} s_{\alpha i} & a_i c_{\theta i} \\ s_{\theta i} & c_{\theta i} c_{\alpha i} & -c_{\theta i} s_{\alpha i} & a_i s_{\theta i} \\ 0 & s_{\alpha i} & c_{\alpha i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{1.6}
$$

donde los parámetros θ_i,d_i,a_i y α_i, llamados generalmente: *ángulo*, *distancia*, *longitud* y *giro*, están relacionados directamente al eslabón i. Sin embargo, dicho planteamiento presenta restricciones en ciertos manipuladores por lo que, de ser necesario, se genera una convención propia para expresar la cinemática directa.

1.1.4. Cinemática Inversa

El problema de la *cinemática inversa* para manipuladores de cadena cinemática serial es encontrar el valor de la posición de las juntas, dadas la posición y orientación del manipulador. Puede definirse una matriz $\boldsymbol{H} \in \Re^{4\times 4}$ tal que

$$
\boldsymbol{H} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{d} \\ 0 & 1 \end{bmatrix} \tag{1.7}
$$

donde se busca encontrar todas las soluciones de la ecuación

$$
\boldsymbol{T}_0^n (q_1, \ldots, q_n) = \boldsymbol{H} \tag{1.8}
$$

donde

$$
\boldsymbol{T}_0^n (q_1, \ldots, q_n) = \boldsymbol{A}_1 \ldots \boldsymbol{A}_n \tag{1.9}
$$

y $\boldsymbol{A}_i$, para $i = 1 \ldots n$ está representado en la ecuación (1.6).

La ecuación (1.8) da como resultado un conjunto de 12 ecuaciones no lineales con n variables desconocidas que se pueden expresar como

$$\boldsymbol{T}_i^j\left(q_1,\ldots,q_n\right)=\boldsymbol{H}_i^j \tag{1.10}$$

para $i=1,2,3$, $j=1,2,3,4$, y donde $\boldsymbol{T}_i^j$ y $\boldsymbol{H}_i^j$ representan respectivamente las componentes no triviales de las matrices $\boldsymbol{T}_0^n$ y $\boldsymbol{H}$ en el $i^{ésimo}$ renglón y $j^{ésima}$ columna.

La solución puede encontrarse de manera explícita, requiriendo encontrar una relación definida por medio de

$$q_k=f_k=(\boldsymbol{H}_{11},\ldots,\boldsymbol{H}_{34}) \tag{1.11}$$

para $k=1,\ldots,n$.

Existen diferentes métodos para la dar solución a la misma, algunos de ellos son métodos numéricos como el de Newton-Raphson o el de mínimos cuadrados y existen métodos geométricos que implican desacoplamiento cinemático.

Con la finalidad de disminuir el tiempo de ejecución del algoritmo de programación, se recurre al método de desacoplamiento cinemático.

1.1.4.1. Desacoplamiento Cinemático

Este método involucra la identificación de puntos del manipulador en los cuales la posición u orientación pueden ser expresadas como función de un conjunto reducido de variables de junta. Esto implica la descomposición del problema espacial en planos separados. Las ecuaciones resultantes son resueltas utilizando métodos algebraicos. Las condiciones para la existencia de la solución de un manipulador de seis grados de libertad es la descomposición del problema en *cinemática de posición inversa y cinemática de orientación inversa*, lo cual puede llevarse a cabo si el robot tiene una muñeca de configuración esférica. El primero de ellos consiste en encontrar la posición del punto de intersección de los ejes de la muñeca, llamado centro de la muñeca, para posteriormente encontrar la orientación. La ecuación (1.8) puede representarse como un conjunto de dos ecuaciones que representan la condición de rotación $\boldsymbol{R}$, y de posición $\boldsymbol{d}$ del marco de referencia en el extremo del manipulador, como se expresa en la ecuación (1.12).

$$\boldsymbol{R}_0^6(q_1,\ldots,q_6) = \boldsymbol{R} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$
$$\boldsymbol{d}_0^6(q_1,\ldots,q_6) = \boldsymbol{d} = \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix} \tag{1.12}$$

La posición del centro de la muñeca depende sólo de las tres primeras variables de junta θ_1, θ_2 y θ_3. Para los casos que aquí se presentan, la posición de la muñeca esférica es simplemente una traslación desde el punto final de la herramienta, hasta la intersección de las juntas θ_4, θ_5 y θ_6

Encontrando las tres primeras variables de junta, se puede determinar la matriz de rotación $\boldsymbol{R}_0^3$, y a partir de este valor, determinar la orientación del extremo del manipulador usando la siguiente expresión

$$\boldsymbol{R} = \boldsymbol{R}_0^3 \boldsymbol{R}_3^6 \tag{1.13}$$

donde se obtiene

$$\boldsymbol{R}_3^6 = \left(\boldsymbol{R}_0^3\right)^{-1} \boldsymbol{R} = \left(\boldsymbol{R}_0^3\right)^T \boldsymbol{R} \tag{1.14}$$

1.2. Cinemática Directa de los Manipuladores Industriales

Los brazos robóticos utilizados para este trabajo de tesis son:

- Robot FANUC, Modelo M-16iB/20T, que se detalla en el Apéndice A.
- Robot FANUC, Modelo LR Mate 200-iC, cuyas principales características se incluyen en el Apéndice B.

Una característica de los robots FANUC, atribuida al fabricante, es su comportamiento en paralelogramo, mostrado en la Figura 1.3. Debido a esta propiedad fue necesario, en el caso del manipulador M-16iB/20T, generar una convención propia para el desarrollo de la cinemática directa y, para el manipulador LR Mate-200iC, modificar la convención de Denavit-Hartenberg.

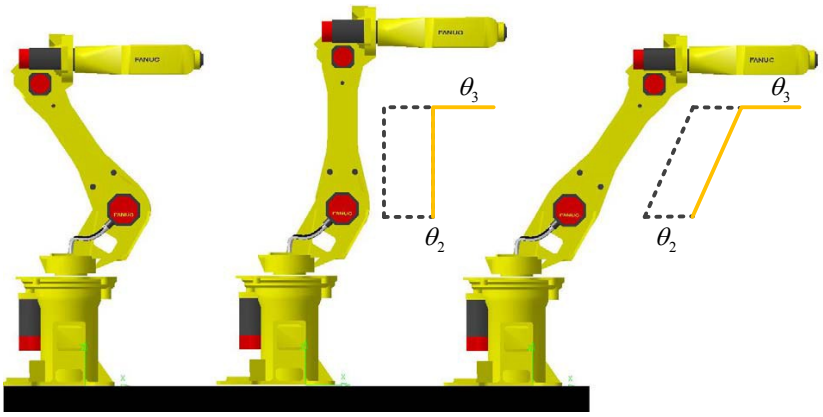


Figura 1.3. Configuración paralelogramo de un robot FANUC.

1.2.1. Cinemática Directa del Manipulador Industrial M-16iB/20T

Con ayuda de la Figura 1.4 puede demostrarse que la cinemática directa del robot M16iB-20T está dada por las siguientes matrices de transformación

$$\begin{aligned}
\boldsymbol{A}_1 &= Trasl_{y,d_1} \\
\boldsymbol{A}_2 &= Trasl_{x,770\cos\theta_2} Trasl_{z,-770\sin\theta_2} \\
\boldsymbol{A}_3 &= Rot_{y,-\theta_3} Trasl_{x,100} \\
\boldsymbol{A}_4 &= Rot_{z,\theta_4} Trasl_{z,-740} \\
\boldsymbol{A}_5 &= Rot_{y,-\theta_5} \\
\boldsymbol{A}_6 &= Rot_{z,\theta_6} Trasl_{z,-100} \\
\boldsymbol{A}_7 &= Rot_{z,\pi} Trasl_{x,-Tx} Trasl_{y,-Ty} Trasl_{z,Tz}
\end{aligned} \tag{1.15}$$

Donde cada una de las matrices son representadas por una matriz homogénea equivalente a los siguientes valores

$$\boldsymbol{A}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_2 = \begin{bmatrix} 1 & 0 & 0 & 770\cos(\theta_2) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -770\sin(\theta_2) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_3 = \begin{bmatrix} \cos(\theta_3) & 0 & -\sin(\theta_3) & 100\cos(\theta_3) + 740\sin(\theta_3) \\ 0 & 1 & 0 & 0 \\ \sin(\theta_3) & 0 & \cos(\theta_3) & 100\sin(\theta_3) - 740\cos(\theta_3) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_4 = \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & 0 \\ \sin(\theta_4) & \cos(\theta_4) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.16}$$

$$\boldsymbol{A}_5 = \begin{bmatrix} \cos(\theta_5) & 0 & -\sin(\theta_5) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\theta_5) & 0 & \cos(\theta_5) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_6 = \begin{bmatrix} \cos(\theta_6) & \sin(\theta_6) & 0 & 0 \\ \sin(\theta_6) & -\cos(\theta_6) & 0 & 0 \\ 0 & 0 & -1 & -100 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_7 = \begin{bmatrix} 1 & 0 & 0 & -399 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 152.4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La matriz homogénea de transformación total está dada por

$$\boldsymbol{H} = \boldsymbol{A}_1\boldsymbol{A}_2\boldsymbol{A}_3\boldsymbol{A}_4\boldsymbol{A}_5\boldsymbol{A}_6\boldsymbol{A}_7 \tag{1.17}$$

1.2.2. Cinemática Directa del Manipulador Industrial LR Mate-200iC

Para el robot industrial LR Mate-200iC, la cinemática directa es obtenida por medio del diagrama auxiliar mostrado en la Figura 1.5. Para este caso, se utiliza la convención de Denavit-Hartenberg. La Tabla 1.1, muestra los valores correspondientes. Es necesario mencionar que, para el caso de la matrices $\boldsymbol{A}_3$ y $\boldsymbol{A}_4$, no se puede tener la representación original del método de Denavit-Hartenberg, el cual define $\boldsymbol{A}_i$ en función de θ_i. Esto debido al comportamiento en paralelogramo mencionado en la sección 1.2.

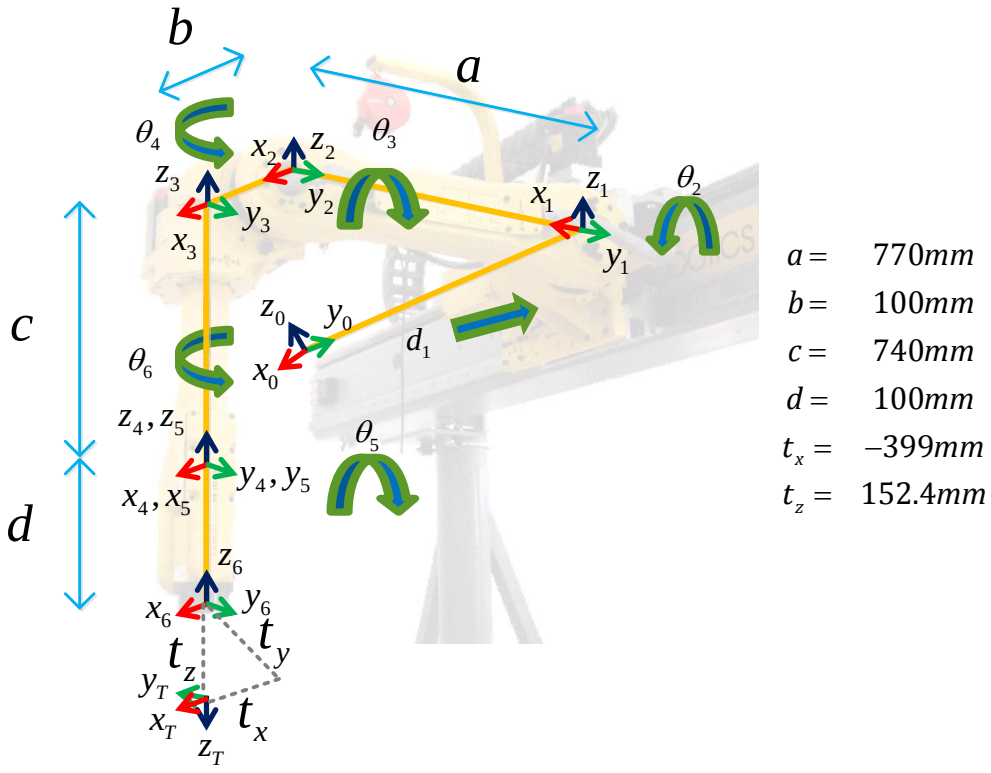


Figura 1.4. Diagrama auxiliar para el cálculo de la cinemática directa del robot M16iB-20T

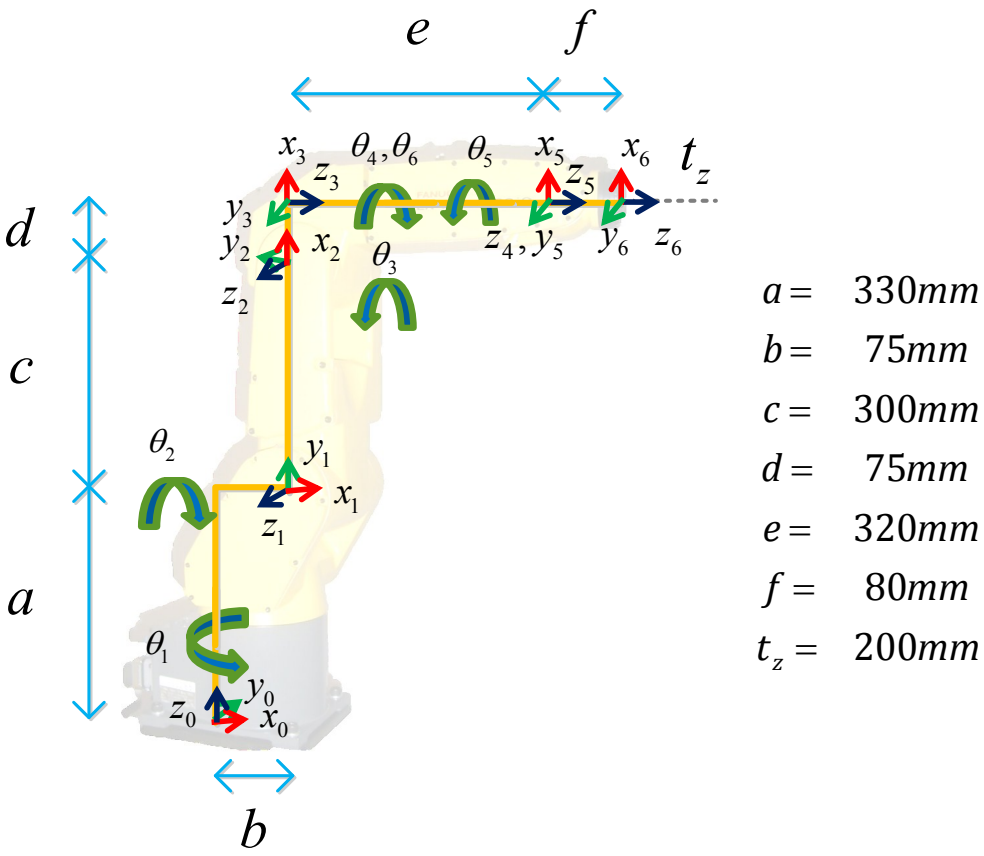


Figura 1.5. Diagrama para el cálculo de la cinemática directa del robot LRMate-200iC.

θ	d_z	a_z	α_x
θ_1	0	b	$\frac{\pi}{2}$
$-\theta_2 + \frac{\pi}{2}$	0	c	0
$\theta_3 + \theta_2$	0	d	$\frac{\pi}{2}$
$-\theta_4$	e	0	$-\frac{\pi}{2}$
θ_5	0	0	$\frac{\pi}{2}$
$-\theta_6$	f	0	$-\frac{\pi}{2}$

Tabla 1.1. Parámetros de Denavit-Hartenberg para robot LR Mate-200iC

Las matrices de transformación asociadas al modelo cinemático del robot se indican a continuación:

$$\boldsymbol{A}_1 = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 75\cos(\theta_1) \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 75\sin(\theta_1) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_2 = \begin{bmatrix} \sin(\theta_2) & -\cos(\theta_2) & 0 & 300\sin(\theta_2) \\ \cos(\theta_2) & \sin(\theta_2) & 0 & 300\cos(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_3 = \begin{bmatrix} \cos(\theta_2+\theta_3) & 0 & \sin(\theta_2+\theta_3) & 75\cos(\theta_2+\theta_3) \\ \sin(\theta_2+\theta_3) & 0 & -\cos(\theta_2+\theta_3) & 75\cos(\theta_2+\theta_3) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_4 = \begin{bmatrix} \cos(\theta_4) & 0 & \sin(\theta_4) & 0 \\ -\sin(\theta_4) & 0 & \cos(\theta_4) & 0 \\ 0 & -1 & 0 & 320 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.18}$$

$$\boldsymbol{A}_5 = \begin{bmatrix} \cos(\theta_5) & 0 & \sin(\theta_5) & 0 \\ \sin(\theta_5) & 0 & -\cos(\theta_5) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_6 = \begin{bmatrix} \cos(\theta_6) & 0 & \sin(\theta_6) & 0 \\ -\sin(\theta_6) & 0 & \cos(\theta_6) & 0 \\ 0 & -1 & 0 & 280 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_7 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La matriz homogénea de transformación total está dada por

$$\boldsymbol{H} = \boldsymbol{A}_1\boldsymbol{A}_2\boldsymbol{A}_3\boldsymbol{A}_4\boldsymbol{A}_5\boldsymbol{A}_6\boldsymbol{A}_7 \tag{1.19}$$

1.3. Cinemática Inversa del Manipulador Industrial

La *cinemática inversa* no es un problema trivial, como se mencionó en la sección 1.1.4. Siguiendo el procedimiento descrito en dicha sección se obtienen los siguientes resultados.

1.3.1. Cinemática Inversa del Manipulador Industrial M-16iB/20T

Como se mencionó con anterioridad, se obtienen los valores de θ_1, θ_2, y θ_3 por medio de la representación de la geometría del robot de la Figura 1.6. Se hacen algunas consideraciones debido al tipo de manipulador, tales como que la primer variable de junta d_1 es una junta prismática y se expresa la matriz de rotación $\boldsymbol{R}_3^6$ del sistema de la forma

$$\boldsymbol{R}_3^6(\theta_4, \theta_5, \theta_6) = \boldsymbol{R} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{1.20}$$

Haciendo las operaciones matemáticas necesarias, se obtiene la cinemática inversa del manipulador como a continuación se indica, donde las variables utilizadas se indican esquemáticamente en las Figuras 1.4 y 1.6.

$$
\begin{aligned}
&\mathrm{d}_1{=}\mathrm{y}_m \\
&\theta_2{=}\alpha+\beta \\
&\theta_3=\pi-\gamma-\theta_2+\delta \\
&\theta_4=atan2\left(R_{23},R_{13}\right) \\
&\theta_5=atan2\left(\left(1-{R_{33}}^2\right),-R_{33}\right) \\
&\theta_6=atan2\left(R_{32},R_{31}\right)
\end{aligned}
\tag{1.21}
$$

El uso de la función $atan2$ o segunda tangente, permite obtener valores en cualquier cuadrante.

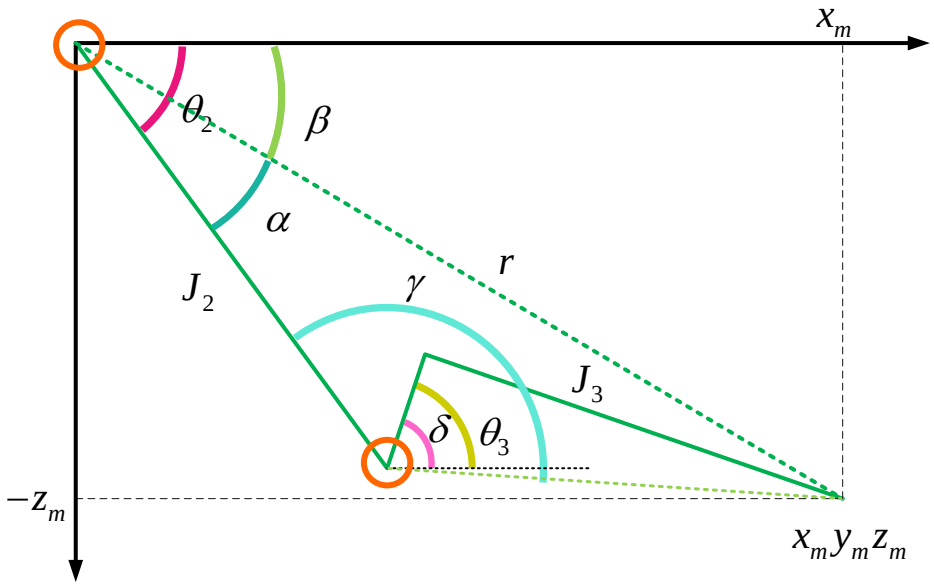


Figura 1.6. Diagrama auxiliar para el cálculo de la cinemática inversa del manipulador M16iB-20T.

1.3.2. Cinemática Inversa del Manipulador Industrial LR Mate-200iC

El cálculo de la cinemática inversa para este manipulador es similar al anterior. De igual manera, se tienen consideraciones propias del comportamiento del mismo, como el hecho de que cuenta con 6 juntas de revoluta. El diagrama que representa los tres primeros ángulos del robot se muestra en la fig. 1.7:

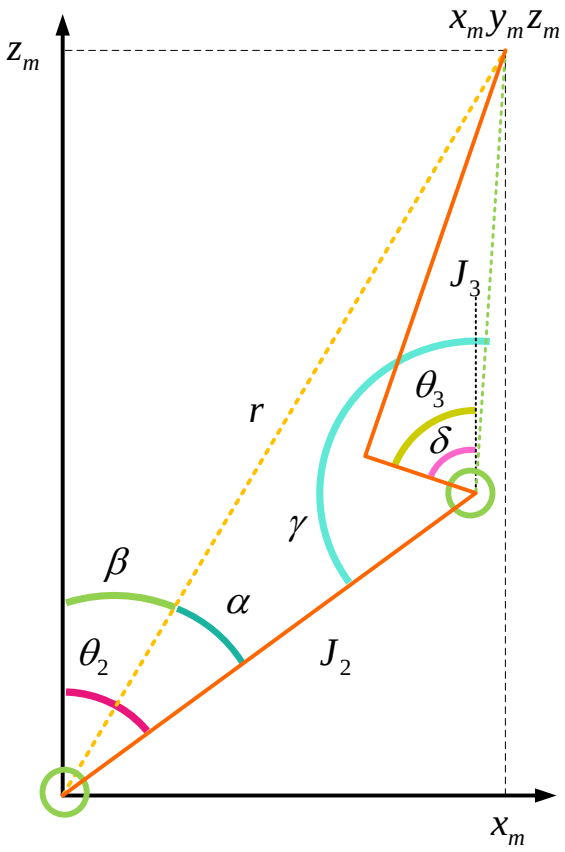


Figura 1.7. Diagrama auxiliar para el cálculo de la cinemática inversa del manipulador LRMate-200iC

Retomando la ecuación (1.20), el desarrollo matemático de la aplicación del desacoplamiento cinemático, conduce al siguiente resultado

$$\begin{aligned}
\theta_1 &= atan2\left(y_m, x_m\right) \\
\theta_2 &= -\left(\alpha + \beta\right) \\
\theta_3 &= -\left(\gamma - \delta\right) \\
\theta_4 &= atan2\left(R_{22}, R_{12}\right) \\
\theta_5 &= atan2\left(\left(1 - {R_{32}}^2\right), -R_{32}\right) \\
\theta_6 &= atan2\left(R_{33}, R_{31}\right)
\end{aligned} \tag{1.22}$$

La resultados cinemáticos obtenidos a lo largo de este capítulo constituyen la herramienta utilizada para generar la simulación de los movimientos de los manipuladores robóticos. Debido a las características internas de programación del lenguaje KAREL, las ecuaciones para las transformaciones cinemáticas en el algoritmo de control de impedancia no fueron necesarias.

Capítulo 2

Control de Impedancia Basado en Posición

2.1. Estado del Arte

En una tarea de manipulación en donde existe contacto físico entre el robot y su entorno, el control de movimiento como tal resulta insuficiente. Una solución a este problema es especificar con exactitud ambos modelos, tanto del robot manipulador como el del entorno, lo cual se torna una tarea muy compleja y poco factible si se toma en cuenta la diversidad de ambientes en los que puede desarrollarse una manipulación robótica.

El control de la interacción física entre el robot y su entorno es crucial para la exitosa ejecución de un gran número de tareas prácticas donde el efector final del robot tiene que manipular un objeto o desarrollar alguna operación sobre una superficie [19]. Durante el contacto, el ambiente puede contener restricciones dentro de la trayectoria a seguir por el efector final, denotadas como *restricciones cinemáticas*. Esta situación, correspondiente al contacto con una superficie rígida, se conoce como *restricción de movimiento*. Las tareas de contacto pueden tener características inerciales, disipativas o elásticas y en cualquier caso, el uso del control de movimiento puro es propenso a fallas, por lo que surge el concepto de ***control de impedancia***.

Los dos principales enfoques de control de impedancia desarrollados son el control híbrido fuerza/posición [11] y el control de impedancia. Este último fue propuesto por Hogan en 1985 y, en base a sus múltiples interpretaciones, es posible generar nuevos enfoques de control de impedancia, tales como el control de impedancia no lineal robusto [10] y el control de impedancia híbrido [1].

Posterior a estos trabajos, en [7] desarrollaron un controlador de impedancia manejado por fuerza que tiene un óptimo intercambio entre robustez y desempeño. Han sido propuestos con éxito

controles de impedancia no lineales, incluyendo control adaptativo con interacción en presencia de incertidumbres [20], control de aprendizaje, robustez en modo deslizante y control descentralizado [9].

Retomando los inicios del control de impedancia, en [8] se establece que, cuando un manipulador encuentra fuerzas de interacción presentes en su entorno que no pueden ser despreciadas, es decir ($\boldsymbol{F} \neq 0$), es necesario controlarlas. Dicho control se conoce actualmente como *control de fuerza*. Aunque modelar el comportamiento dinámico del robot, modificando su retroalimentación de posición y velocidad, es una solución a este problema, se pueden también explotar las propiedades intrínsecas del robot como el camino más viable para la interacción mecánica. Hogan pretende el desarrollo de un modelo dinámico que represente la interacción entre una admitancia, la cual es representada por el entorno de trabajo y una impedancia, comportamiento que será adquirido por el robot. Este enfoque de manipulación, denominado *control de impedancia*, pretende que la interacción dinámica entre el manipulador y su entorno sea modulada, regulada y controlada por la variación de dicha impedancia.

Si se tienen los sensores adecuados, a partir de una posición de entrada se puede obtener una fuerza de salida, cuya relación estática está dada por una rigidez, que es la relación entre fuerza y posición, un amortiguamiento, que es la relación entre fuerza-velocidad y si además se incluye la relación de la inercia, se obtendrá una ecuación que representa la fuerza de interacción de las siguiente manera:

$$\boldsymbol{F}_{\text{int}} = \boldsymbol{K}\left[X_0 - X\right] + \boldsymbol{B}\left[\dot{X}_0 - \dot{X}\right] + \boldsymbol{M}\left[\ddot{X}_0 - \ddot{X}\right] \tag{2.1}$$

donde $\boldsymbol{F}_{int} \in \Re^3$ es la fuerza de interacción, $X_0 \in \Re^3$ es la posición comandada, $X \in \Re^3$ es la posición del efector final, $\boldsymbol{K}_d = diag\left[\begin{array}{ccc} k_d & k_d & k_d \end{array}\right]$ es la relación de rigidez, $\boldsymbol{B}_d = diag\left[\begin{array}{ccc} b_d & b_d & b_d \end{array}\right]$ de amortiguamiento y $\boldsymbol{M}_d = diag\left[\begin{array}{ccc} m_d & m_d & m_d \end{array}\right]$ de inercia.

A partir de esta ecuación, en [14], se presentan dos enfoques de control de impedancia, por retroalimentación de torque y por retroalimentación de posición. A diferencia del control de impedancia simple, en este último enfoque se necesita de una retroalimentación de fuerzas para generar comandos de posición en el controlador interno del robot. Esta forma de control se basa en el control de posición preciso del manipulador. Al control de impedancia propuesto en [8] se agrega un lazo de control adicional a la posición controlada por el manipulador. Considerando la siguiente relación entre esfuerzo y movimiento de la trayectoria nominal del efector final:

$$\boldsymbol{F} = \boldsymbol{K}\left(X - X_0\right) + \boldsymbol{B}\left(\dot{X} - \dot{X}_0\right) + \boldsymbol{M}\left(\ddot{X} - \ddot{X}_0\right) \tag{2.2}$$

donde $\boldsymbol{F}$ es un vector de fuerzas y torques sobre el manipulador debido al contacto con el entorno, y $\left(X - X_0, \dot{X} - \dot{X}_0, \ddot{X} - \ddot{X}_0\right)$ representa la diferencia entre la posición comandada por el robot y la posición del efector final. La ecuación (2.2) describe la dinámica de la trayectoria de perturbación. La impedancia especificada por la ecuación (2.2) está dada en el dominio de frecuencia por la matriz

$$\boldsymbol{Z} = \boldsymbol{K} + \boldsymbol{B}s + \boldsymbol{M}s^2 \tag{2.3}$$

El parámetro $\boldsymbol{K}$ es la matriz de rigidez, $\boldsymbol{B}$ es la matriz de amortiguamiento y $\boldsymbol{M}$ es la matriz de inercia. Para tareas particulares, el acoplamiento entre ejes de impedancias puede ser útil, pero se considerarán solo *impedancias desacopladas*, es decir $\boldsymbol{K}$, $\boldsymbol{B}$ y $\boldsymbol{M}$ diagonales. Para el enfoque basado en posición, las fuerzas y torques son sensados explícitamente, y los comandos de posición son expedidos para el controlador de lazo interno. Se crea un vector de ajuste de posición mediante el filtrado de las medidas de interacción de fuerzas y torques para satisfacer:

$$\boldsymbol{F} = \boldsymbol{K}X_a + \boldsymbol{B}\dot{X}_a + \boldsymbol{M}\ddot{X}_a \tag{2.4}$$

obteniendo como resultado el vector de ajuste de posición $X_a\,(s)$

$$X_a\,(s) = \left[\boldsymbol{K} + \boldsymbol{B}s + \boldsymbol{M}s^2\right]^{-1}\boldsymbol{F}\,(s) \tag{2.5}$$

Con la simplificación obtenida al considerar las matrices $\boldsymbol{K}$, $\boldsymbol{B}$, y $\boldsymbol{M}$ como diagonales, se reducirá a segundo orden el filtro pasa bajas para cada componente de $\boldsymbol{F}$, generando el respectivo componente de X_a. El ajuste de X_a es agregado a la trayectoria nominal comandada X_0, para generar un comando de posición global X_e

$$X_e = X_0 + X_a \tag{2.6}$$

Si $\boldsymbol{F} \equiv 0$ (sin contacto con el entorno) entonces $X_e = X_0$. Si el manipulador tiene la habilidad para aplicar precisión a este comando, por ejemplo $X \equiv X_c$, entonces:

$$X_a = X - X_0 \tag{2.7}$$

La ley de control en las ecuaciones (2.5) y (2.6) satisfacen la especificación de impedancia original de la ecuación (2.2). Así, el enfoque de control de impedancia basado en posición se fundamenta en el control preciso de posición del manipulador, dentro del lazo de control de impedancia actual.

El tiempo de retardo en el lazo de control de impedancia es a menudo significativo porque es un lazo de control cartesiano y las transformaciones entre espacio articulado y espacio cartesiano son necesarias para proveer comandos adecuados para el actuador. El lazo de control de impedancia puede, por lo tanto, consistir del filtro de la ecuación (2.2) seguido de un cambio de marco coordenado, seguido por la ecuación (2.6), seguida por la transformación cinemática inversa para llegar al comando de posición de unión.

Basados en el desarrollo de [14], se genera la propuesta de control de impedancia basada en posición por [3], involucrando el control de retroalimentación de posición como eje principal de la misma, el cual es el modelo del enfoque presentado en esta investigación. Un diagrama representativo del trabajo de tesis desarrollado puede ser representado por la Figura 2.1. Cada uno de los conceptos en la misma se detalla en las siguientes secciones.

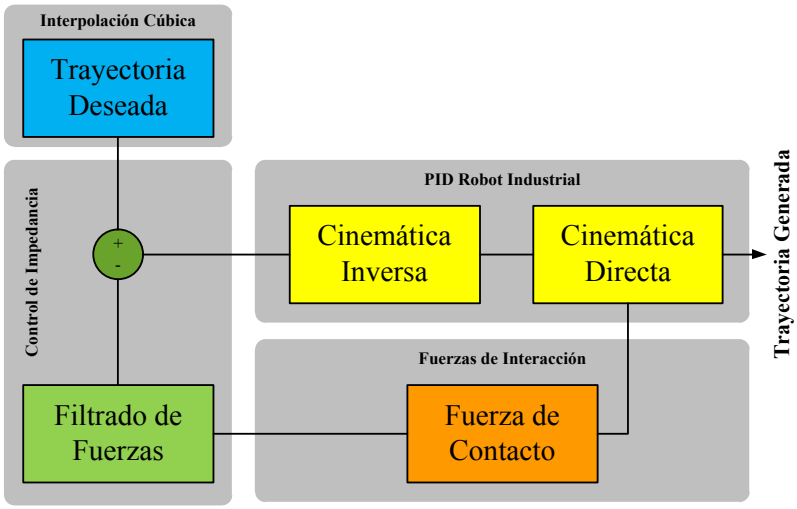


Figura 2.1. Esquema de Control de Impedancia por Retroalimentación de Posición

2.2. Trayectoria Deseada

La mayoría de los robots industriales son utilizados para desarrollar tareas repetitivas. Todo lo que realiza un robot en este tipo de tareas es ir a puntos asignados previamente por el programador, pero sin interactuar con su entorno. La trayectoria de los manipuladores, en estos casos, depende únicamente de la tarea a la que son asignados para realizar y, debido a esto, existen muchas trayectorias posibles por realizar por un robot industrial. Algunos ejemplos prácticos son: pintura, soldadura, ensamble, embalaje y manufactura, entre otras.

2.2.1. Interpolación Cúbica

La *interpolación* es un proceso para estimar valores que se encuentran entre puntos conocidos; cuando se conocen dos de ellos se pueden unir mediante una línea. Las curvas de tercer grado empleadas para unir pares de datos se llaman segmentarias cúbicas, y utilizan polinomios de segundo o tercer grado para interpolar entre todos los pares de puntos, mientras que los coeficientes de los polinomios se calculan utilizando datos adicionales provenientes de puntos adyacentes a los dos seleccionados. Estas funciones se pueden construir de tal forma que las conexiones entre ecuaciones cúbicas adyacentes resulten visiblemente suaves. De esta manera, las curvas de tercer grado proporcionan una mejor aproximación al comportamiento de las funciones que tienen cambios locales abruptos. La fórmula general para un polinomio de *n-ésimo* grado es

$$f(t) = a_0 + a_1 t + a_2 t^2 + \ldots + a_n t^n \tag{2.8}$$

Dados n+1 puntos, hay un solo polinomio de grado n que pasa a través de ellos. La *interpolación polinomial* consiste en determinar el polinomio único de *n-ésimo* grado que se ajuste a n+1 puntos.

Existe una gran variedad de formas matemáticas en las cuales puede expresarse un polinomio, sin embargo, en la segmentarias cúbicas se pretende obtener un polinomio de tercer grado para cada intervalo entre los nodos, por medio de la siguiente expresión

$$f(t) = a_i t^3 + b_i t^2 + c_i t + d_i \tag{2.9}$$

Existen así, 4 incógnitas a evaluar y se requieren 4 condiciones para evaluarlas. Éstas son:

$$\begin{aligned} q_i &= q(t=0) = q_i \\ q_f &= q(t=t_f) = q_f \\ \dot{q}_i &= q(t=0) = \dot{q}_i = 0 \\ \dot{q}_f &= q(t=t_f) = \dot{q}_f = 0 \end{aligned} \tag{2.10}$$

Donde q_i corresponde a la posición inicial, q_f es la posición final, mientras que t y t_f son el tiempo actual y el tiempo final, respectivamente.

Realizando las operaciones algebraicas correspondientes, se obtienen los siguientes valores para las incógnitas de la ecuación (2.9).

$$
\begin{aligned}
a_i &= \frac{-2\,(q_f - q_i)}{t_f^3} \\
b_i &= \frac{3\,(q_f - q_i)}{t_f^2} \\
c_i &= 0 \\
d_i &= q_i
\end{aligned}
\tag{2.11}
$$

Estos valores se introducen en la ecuación (2.9), para obtener los puntos intermedios entre la posición de inicio del manipulador y la posición final. El número de puntos intermedios estará dado por el número los valores de t, muestreados entre q_i y q_f.

2.3. Filtrado de Fuerzas

Las fuerzas involucradas en el esquema de control de impedancia, se obtienen de las mediciones proporcionadas por el sensor de fuerza utilizado en el experimento. Las señales de fuerza provenientes del sensor son muestreadas y expresadas de manera discreta para ser transformadas en el vector de ajuste de posición que permite generar la nueva trayectoria del manipulador. Existen varios métodos para la discretización de señales. Para el filtrado de fuerzas en el desarrollo actual, se presentan dos casos, los cuales son: *Retenedor de Orden Cero y Método de Resolución Numérica de Runge-Kutta de segundo orden*. Cada uno de ellos se describe en las secciones subsecuentes.

2.3.1. Retenedor de Orden Cero (ZOH)

2.3.1.1. Muestreo de Señales en Tiempo Continuo

El muestreo de señales es utilizado en sistemas que serán controlados de manera computarizada y sus señales se presentan en tiempo continuo. La idea principal es que un sistema en tiempo continuo puede ser *transformado* en un sistema en tiempo discreto, tomando en consideración el comportamiento en los instantes de muestreo. En este proyecto, el controlador del robot recibe medidas del proceso en tiempos continuos y transmite nuevas señales de control en tiempos discretos. El objetivo es describir el cambio de la señal entre muestra y muestra e ignorar el comportamiento entre ellas.

En su forma más general, muestrear significa: "el acto o proceso de tomar una pequeña parte o cantidad de algo, para realizar pruebas o análisis". En el contexto de comunicación, significa que una

señal en tiempo continuo será reemplazada por una secuencia de números, los cuales representan el valor de las señales en tiempos específicos.

Una situación común en control computarizado es que el convertidor D-A (Digital-Analógico) está construido de modo que mantiene la señal análoga constante hasta que una nueva conversión es ordenada. Esto es conocido como *retenedor de orden cero*, donde es natural seleccionar instantes de muestreo iguales a los cambios de tiempo en el controlador.

La relación entre las variables del sistema, en tiempos de muestreo, puede ser determinada a partir de la ecuación de espacio de estados del sistema en tiempo continuo, el cual se representa por la expresión general:

$$\begin{aligned} \frac{dx}{dt} &= \boldsymbol{A}x\left(t\right) + \boldsymbol{B}u\left(t\right) \\ y\left(t\right) &= \boldsymbol{C}x\left(t\right) + \boldsymbol{D}u\left(t\right) \end{aligned} \tag{2.12}$$

dicho sistema tiene r entradas, p salidas, y es de orden n.

Dado el estado del sistema para un tiempo t_{k-1}, el estado en el tiempo t_k es obtenido resolviendo la ecuación (2.12). La señal de control es representada por la señal muestreada $\{u\left(t_{k-1}\right) : k = \ldots, -1, 0, 1, \ldots\}$. El estado en t_k, donde $t_{k-1} \leq t_k$ está dado por:

$$\begin{aligned} x(t_k) &= e^{\boldsymbol{A}(t_k - t_{k-1})}x\left(t_{k-1}\right) + \int_{t_{k-1}}^{t} e^{\boldsymbol{A}(t_k - s')}\boldsymbol{B}u\left(s'\right)ds' \\ &= e^{\boldsymbol{A}(t_k - t_{k-1})}x\left(t_{k-1}\right) + \int_{t_{k-1}}^{t_k} e^{\boldsymbol{A}(t_k - s')}ds'\boldsymbol{B}u\left(t_{k-1}\right) \\ &= e^{\boldsymbol{A}(t_k - t_{k-1})}x\left(t_{k-1}\right) + \int_{0}^{t_k - t_{k-1}} e^{\boldsymbol{A}s}ds\boldsymbol{B}u\left(t_{k-1}\right) \\ &= \boldsymbol{\Phi}\left(t_k, t_{k-1}\right)x\left(t_{k-1}\right) + \boldsymbol{\Gamma}\left(t_k, t_{k-1}\right)u\left(t_{k-1}\right) \end{aligned} \tag{2.13}$$

El vector en el tiempo t_k es una función lineal de $x\left(t_{k-1}\right)$ y $u\left(t_{k-1}\right)$, y la conversión de tiempo es despreciable, entonces la entrada u y la salida y pueden ser consideradas como la muestra en los instantes mismos [2]. La ecuación del sistema muestreado en cada instante es

$$\begin{aligned} x\left(t_k\right) &= \boldsymbol{\Phi}\left(t_k, t_{k-1}\right)x\left(t_{k-1}\right) + \boldsymbol{\Gamma}\left(t_k, t_{k-1}\right)u\left(t_{k-1}\right) \\ y\left(t_{k-1}\right) &= \boldsymbol{C}x\left(t_{k-1}\right) + \boldsymbol{D}u\left(t_{k-1}\right) \end{aligned} \tag{2.14}$$

donde

$$\begin{aligned} \boldsymbol{\Phi}\left(t_k, t_{k-1}\right) &= e^{\boldsymbol{A}(t_k - t_{k-1})} \\ \boldsymbol{\Gamma}\left(t_k, t_{k-1}\right) &= \left(\int_0^{t_k - t_{k-1}} e^{\boldsymbol{A}s} ds\right) \boldsymbol{B} \end{aligned} \tag{2.15}$$

Para periodos de tiempo τ se tiene que $t_k = (k\tau)$ y el modelo de la ecuación (2.14) se simplifica a un sistema invariante en el tiempo [2]:

$$\begin{aligned} x\left(k\tau + \tau\right) &= \boldsymbol{\Phi} x\left(k\tau\right) + \boldsymbol{\Gamma} u\left(k\tau\right) \\ y\left(k\tau\right) &= \boldsymbol{C} x\left(k\tau\right) + \boldsymbol{D} u\left(k\tau\right) \end{aligned} \tag{2.16}$$

donde

$$\begin{aligned} \boldsymbol{\Phi}\left(\tau\right) &= e^{\boldsymbol{A}\tau} \\ \boldsymbol{\Gamma}\left(\tau\right) &= \int_0^{\tau} e^{\boldsymbol{A}s} ds \boldsymbol{B} \end{aligned} \tag{2.17}$$

Con las ecuaciones (2.17) se obtienen las expresiones en tiempo discreto para el sistema de control de impedancia. La ecuación del sistema masa-resorte-amortiguador, que describe el comportamiento del robot de manera continua, se representa con base en las características de relación de amortiguamiento ξ, frecuencia natural no amortiguada ω_n, y amortiguamiento deseado b_d, de acuerdo a la siguiente relación:

$$\begin{aligned} \omega_n &= \sqrt{\frac{k_d}{m_d}} \\ \xi &= \frac{b_d}{2\omega_n m_d} \end{aligned} \tag{2.18}$$

la ecuación del sistema se expresa entonces como:

$$\ddot{x} + 2\xi\omega_n \dot{x} + \omega_n^2 x = \frac{2\xi\omega_n}{b_d} F \tag{2.19}$$

y de forma matricial está dada por:

$$\begin{aligned}
\boldsymbol{A} &= \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\xi\omega_n \end{bmatrix} \\
\boldsymbol{B} &= \begin{bmatrix} 0 \\ \frac{2\xi\omega_n}{b_d} \end{bmatrix} \\
\boldsymbol{C} &= \begin{bmatrix} 1 & 0 \end{bmatrix} \\
\boldsymbol{D} &= 0
\end{aligned} \tag{2.20}$$

Al sustituir los valores anteriores en la ecuación (2.17) se obtiene:

$$\begin{aligned}
\boldsymbol{\Phi} &= e^{\boldsymbol{A}\tau} = e^{\lambda\tau} = \sum_{k=0}^{n-1} \alpha_k \lambda^k \\
\boldsymbol{\Gamma} &= \int_0^{\tau} e^{\boldsymbol{A}\tau} \boldsymbol{B} d\tau
\end{aligned} \tag{2.21}$$

Dando como resultado, de manera general, las siguientes expresiones para $\boldsymbol{\Phi}$ y $\boldsymbol{\Gamma}$, que son la representación en sistema discreto del sistema masa-resorte-amortiguador:

$$\begin{aligned}
\boldsymbol{\Phi} &= \begin{bmatrix} \alpha_0 & \alpha_1 \\ -\alpha_1\omega_n^2 & \alpha_0 - 2\alpha_1\xi\omega_n \end{bmatrix} \\
\boldsymbol{\Gamma} &= \int_0^{\tau} \boldsymbol{\Phi} \boldsymbol{B} d\tau
\end{aligned} \tag{2.22}$$

Las variables α_0 y α_1 obtendrán valores diferentes para cada caso de amortiguamiento, ya que dependen de los eigenvalores de la matriz característica. Dichos valores propios se obtienen de la siguiente ecuación:

$$\lambda_1, \lambda_2 = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1} \tag{2.23}$$

Caso sub amortiguado Para este caso, la relación de amortiguamiento puede tomar valores mayores de cero, pero menores a la unidad. Además, los eigenvalores de la matriz característica del sistema son conjugados, y pueden expresarse como

$$\lambda_1, \lambda_2 = a \pm bi \tag{2.24}$$

Los valores de α_0 y α_1 se especifican a continuación

$$\begin{aligned}\alpha_0 &= e^{a\tau}\left(\cos b\tau - \frac{a\sin b\tau}{b}\right)\\ \alpha_1 &= \frac{e^{a\tau}\sin b\tau}{b}\end{aligned} \tag{2.25}$$

Caso críticamente amortiguado Cuando la relación de amortiguamiento tiene un valor unitario, y los eigenvalores de la matriz A son iguales y reales, las variables α_0 y α_1 están dadas entonces por:

$$\begin{aligned}\alpha_0 &= e^{\lambda\tau}\left(1-\lambda\tau\right)\\ \alpha_1 &= \tau e^{\lambda\tau}\end{aligned} \tag{2.26}$$

Caso sobre amortiguado Para este último caso, la relación de amortiguamiento presenta un valor superior a la unidad y los valores propios del sistema son reales y diferentes de cero. Las variables α_0 y α_1, presentan los siguiente valores

$$\begin{aligned}\alpha_0 &= e^{\lambda_1\tau} - \frac{e^{\lambda_1\tau}-e^{\lambda_2\tau}}{\lambda_1+\lambda_2}\lambda_1\\ \alpha_1 &= \frac{e^{\lambda_1\tau}-e^{\lambda_2\tau}}{\lambda_1+\lambda_2}\end{aligned} \tag{2.27}$$

2.3.2. Método de Resolución Numérica Runge-Kutta

Los métodos de resolución numérica de Runge-Kutta logran la exactitud del procedimiento de la serie de Taylor, sin necesitar el cálculo de derivadas de orden superior. La fórmula generalizada de la ecuación representativa, para esta metodología de resolución, está dada por:

$$y_{i+1} = y_i + \phi\left(x_i, y_i, \tau\right)\tau \tag{2.28}$$

donde $\phi\left(x_i, y_i, \tau\right)$ se conoce como una función incremento, que se interpreta como una pendiente representativa en un intervalo y su función incremento se escribe como

$$\phi = a_1k_1 + a_2k_2 + ... + a_nk_n \tag{2.29}$$

donde $a_1, \ldots, a_n$ son constantes y $k_1, \ldots, k_n$ se obtienen a partir de:

$$\begin{aligned} k_1 &= f(x_i, y_i) \\ k_2 &= f\left(x_i + p_1\tau, y_i + q_{11}k_1\tau\right) \\ k_3 &= f\left(x_i + p_2\tau, y_i + q_{21}k_1\tau + q_{22}k_2\tau\right) \\ &\vdots \\ k_n &= fx_i + p_{n-1}h, y_i + q_{n-1,1}k_1\tau + q_{n-1,2}k_2\tau + ... + q_{n-1,n-1}k_{n-1}\tau \end{aligned} \tag{2.30}$$

donde p y q son constantes y, como puede observarse, las variables k son relaciones de recurrencia, lo que hace eficientes los métodos de Runge-Kutta para cálculos computacionales.

Es posible variar el orden de aproximación del método de Runge-Kutta, empleando diferente número de términos en la función incremento especificada por n. Para $n = 1$ se genera el procedimiento de Runge-Kutta de primer orden el cual, básicamente es el Método de Euler referido en la Figura 2.2. Para orden mayor, n indicará el orden de aproximación.

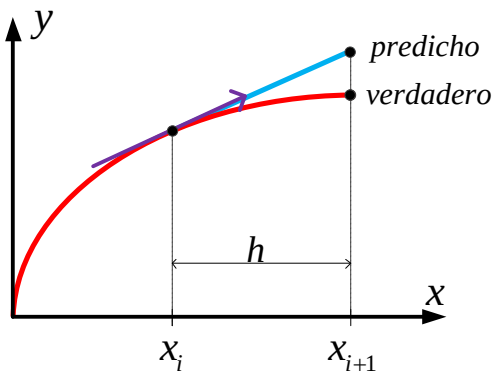


Figura 2.2. Método de Euler

2.3.2.1. Método de Runge-Kutta de Segundo Orden

La versión de segundo orden la ecuación de Runge-Kutta, es la siguiente

$$y_{i+1} = y_i + (a_1k_1 + a_2k_2)\,\tau \tag{2.31}$$

donde

$$\begin{aligned} k_1 &= f\left(x_i, y_i\right) \\ k_2 &= f\left(x_i + p_1\tau, y_i + q_{11}k_1\tau\right) \end{aligned} \tag{2.32}$$

Los valores de a_1, a_2, p_1 y q_{11} se evalúan al igualar la ecuación (2.31) con la expresión de la serie de Taylor hasta el término de segundo orden. Al hacerlo, se desarrollan tres ecuaciones para evaluar cuatro constantes desconocidas. Al resolverlas de manera simultánea se obtiene que:

$$\begin{aligned} a_1 &= 1 - a_2 \\ p_1 &= q_{11} = \frac{1}{2a_2} \end{aligned} \tag{2.33}$$

Se observa que a_2 puede ser asignado con infinito número de valores, por lo que existen también infinitos métodos de Runge-Kutta de segundo orden [5].

Las versiones más utilizadas son

- *Método de Heun* con un solo corrector ($a_2 = \frac{1}{2}$)
- El *método del punto medio* ($a_2 = 1$)
- El *método de Ralston* ($a_2 = \frac{2}{3}$)

El método del punto medio es una alternativa viable en el desarrollo de nuestro algoritmo de control. Este método supone a_2=1 y, a partir de esta condición, se obtienen los valores de a_1=0 y $p_1 = q_{11} = \frac{1}{2}$. Al sustituir los valores anteriores en la ecuación (2.31), se obtiene

$$y_{i+1} = y_i + k_2\tau \tag{2.34}$$

donde

$$\begin{aligned} k_1 &= f\left(x_i, y_i\right) \\ k_2 &= f\left(x_i + \frac{1}{2}\tau, y_i + \frac{1}{2}k_1\tau\right) \end{aligned} \tag{2.35}$$

Se desarrolla el método anterior para el sistema representado por la ecuación:

$$F = m\ddot{x} + b\dot{x} + kx \tag{2.36}$$

obteniendo así los siguientes resultados matemáticos:

$$\begin{aligned} x(\tau) &= x_0 + \dot{x}_0\tau + \frac{\tau^2}{2m}(F_0 - b\dot{x}_0 - kx_0) \\ \dot{x}(\tau) &= \dot{x}_0 + \frac{F_0}{m}\tau - \frac{b}{m}\left(\dot{x}_0\tau + \frac{\tau^2}{2m}(F_0 - b\dot{x}_0 - kx_0)\right) - \frac{k}{m}\left(\dot{x}_0\tau + \frac{\tau^2}{2}\dot{x}_0\right) \end{aligned} \tag{2.37}$$

donde x y $\dot{x}$ representan la posición y velocidad, las variables x_0 y $\dot{x}_0$ representan las condiciones iniciales de posición y velocidad respectivamente; m, b y k representan el valor de la masa, amortiguamiento y rigidez deseada, F_0 es la fuerza de interacción y τ el tiempo de muestreo del sistema.

Capítulo 3

Resultados Experimentales

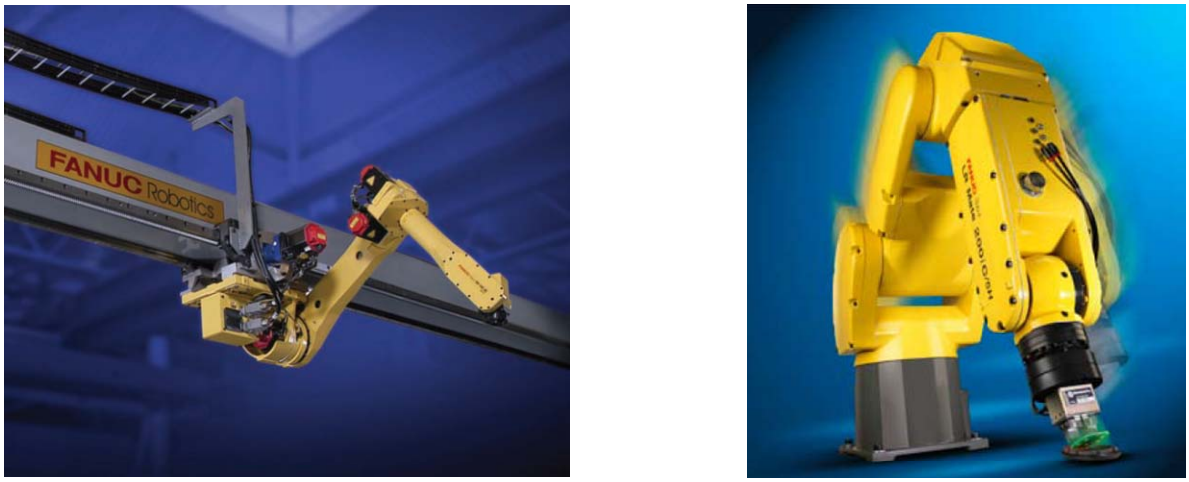
3.1. Laboratorio de Robótica de la UASLP

Esta investigación se llevó a cabo dentro de las instalaciones del Laboratorio de Robótica de la Universidad Autónoma de San Luis Potosí. Fue necesario, para su desarrollo, el manejo y programación de 2 robots industriales de la marca FANUC. Los modelos M-16iB/20T, Figura 3.1(a) y LR Mate-200iC, Figura 3.1(b), son manipuladores de 6 grados de libertad. Se utilizó además un sensor de fuerza de la misma marca, modelo FS-10iA, mostrado en la Figura 3.7 (ver Apéndices A,B y C para mayor referencia).

Como primer instancia, se pretendía adaptar un sensor de fuerza al modelo M-16iB/20T, sin embargo, esto no fue posible ya que este manipulador no tiene compatibilidad con ningún sensor de fuerza de la misma marca, debido a restricciones impuestas por el mismo fabricante. El lenguaje de programación KAREL, por otro lado, es igual para cualquier modelo robótico FANUC. Por este motivo, al inicio de esta investigación fue necesario trabajar con el robot M-16iB/20T y simular las fuerzas de interacción presentes en el entorno. Todo esto se llevaba a cabo mientras el sensor de fuerza estaba en trámites de entrega y aún no se contaba con el modelo LR Mate-200iC en el Laboratorio de Robótica, el cual es totalmente compatible al sensor FS-10iA.

3.2. Robot M-16iB/20T

Como se mencionó en la sección 2.2, existen múltiples tareas de manipulación asignadas a diversos tipos de robots. Para el caso que aquí compete, se pretende que el robot siga una trayectoria



(a) M-16iB/20T (b) LR Mate-200iC

Figura 3.1. Brazos robóticos de 6 grados de libertad.

arbitraria y sea perceptible el momento en que una fuerza de interacción se presenta, desviando al robot de su trayectoria original. Con la intención de visualizar gráficamente este desplazamiento, se sugiere el uso de una trayectoria en línea recta, para poder visualizar el cambio de posición con respecto de ella.

3.2.1. Programación de la Trayectoria Deseada

Se desarrolló un programa en lenguaje KAREL para designar la trayectoria deseada por medio de una interpolación cúbica, como la indicada en la sección 2.2.1. Se programó al robot para generar un movimiento desde un punto inicial designado por las coordenadas:

x_i	y_i	z_i
1081.796	1500.000	-978.469

hasta el punto final situado en

x_f	y_f	z_f
1081.796	2200.000	-978.469

Por medio de MATLAB, se realizó una simulación de dicho comportamiento para obtener una gráfica del los resultados esperados, obteniendo la trayectoria presentada en la Figura 3.2.

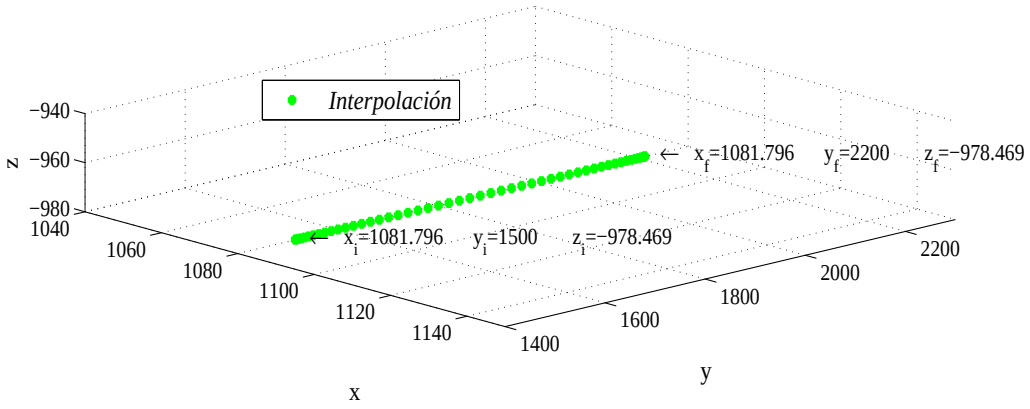


Figura 3.2. Interpolación cúbica, simulación en MATLAB

Puede observarse que, debido a la naturaleza del cálculo de los puntos intermedios, la distancia que se presenta entre ellos no es constante. Además, el número de puntos interpolados dependerá del programador. En este experimento, se establece un total de 50 puntos intermedios para poder visualizar su comportamiento.

Desarrollando el procedimiento anterior en KAREL, con los datos proporcionados por el manipulador, se obtiene la Gráfica 3.3. En esta figura, debido a la superposición de la trayectoria real que realiza el robot, no es posible identificar los puntos interpolados con la misma facilidad que en la gráfica anterior. Sin embargo, la condición que pretende apreciarse, es que el robot sigue la misma trayectoria que la que se designó por medio de simulación.

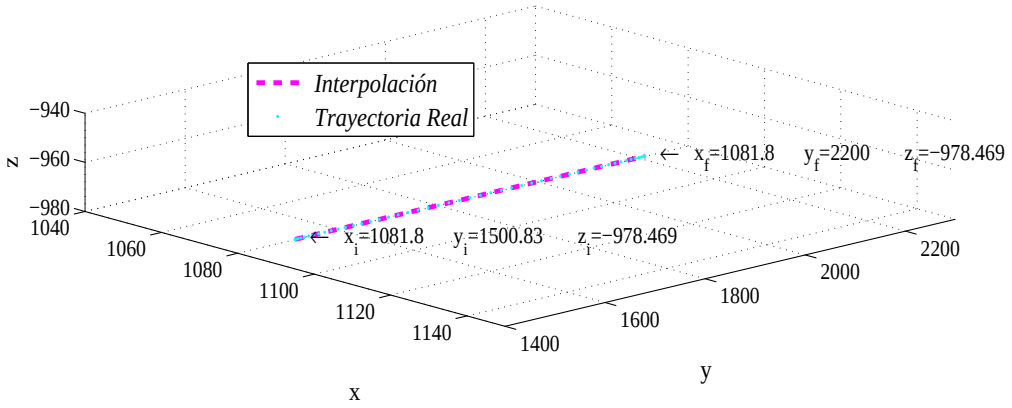


Figura 3.3. Simulación y trayectoria generada por el robot M 16iB/20T sin fuerzas de interacción presentes

3.2.2. Programación de Filtrado de Fuerzas

Se desarrolló también un algoritmo de control para la discretización de fuerzas, con base en los resultados obtenidos en la sección 2.3.1. En el manipulador M 16iB/20T, únicamente fue programado y compilado el algoritmo para el caso sobreamortiguado. A las ecuaciones (2.22) y (2.26) se asignan los valores de las matrices de inercia, amortiguamiento y rigidez, indicadas en la Tabla 3.1.

Parámetro	Valor	Unidades
m_d	2	Kg
b_d	200	$N \cdot s/m$
k_d	20	N/m

Tabla 3.1. Tabla de parámetros para el caso sobreamortiguado en el robot M-16iBT/20T

Con los parámetros anteriores, se obtuvo el modelo en simulación por medio de MATLAB, para el sistema descrito por la ecuación (3.1), con una entrada escalón de 1 Kgf. El resultado se muestra en la Figura 3.4.

$$F = m\ddot{x} + b\dot{x} + kx \tag{3.1}$$

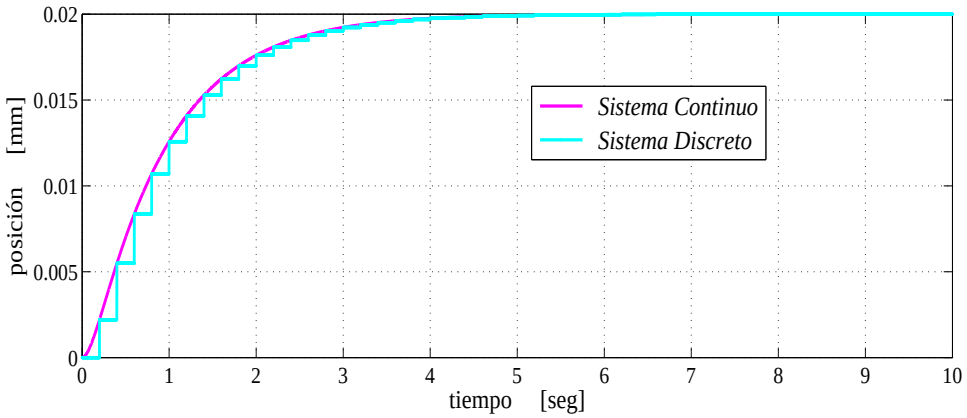


Figura 3.4. Simulación del sistema sobre amortiguado para el robot M-16iB/20T

Para este caso, al ser programada la fuerza dentro del robot, se toma un tiempo de muestreo constante de $200ms$. Dicho tiempo es estimado de los datos de muestreo arrojados por el robot, los cuales oscilan alrededor de este valor. Es apreciable, en la Figura 3.4, que el sistema tiene la respuesta esperada; no presenta oscilaciones para alcanzar el punto máximo y es necesario resaltar que, una entrada de fuerza de 1 Kgf, genera una salida de desviación de la posición de 0.02 mm.

Los parámetros de la Tabla 3.1, son referencia del trabajo presentado en [4], calibrados para su uso en este robot industrial.

3.2.2.1. Fuerzas de Interacción

Las fuerzas de interacción necesarias para el desarrollo de esta técnica de control, fueron procesadas, como se ha mencionado, dentro del mismo programa. Se proponen las ecuaciones (3.2) las cuales, al ser funciones senoidales, tienen un comportamiento suave, para proporcionar un movimiento similar al del robot. Por otro lado, son fuerzas relativamente pequeñas, para que no se aleje demasiado de la trayectoria deseada.

$$\begin{aligned} Fx &= 2\sin(1.5t)\,, \\ Fy &= 2\sin(2t)\,, \\ Fz &= 2\sin(2.5t)\,. \end{aligned} \tag{3.2}$$

En la Gráfica 3.5 puede verse como el manipulador, al percibir una fuerza externa, modifica su trayectoria de acuerdo a ésta, con un movimiento definido por las ecuaciones de retroalimentación que proporciona el retenedor de orden cero. Un diagrama de flujo de este algoritmo se muestra en la Figura 3.6. La *trayectoria real*, está generada con datos provenientes del controlador del robot.

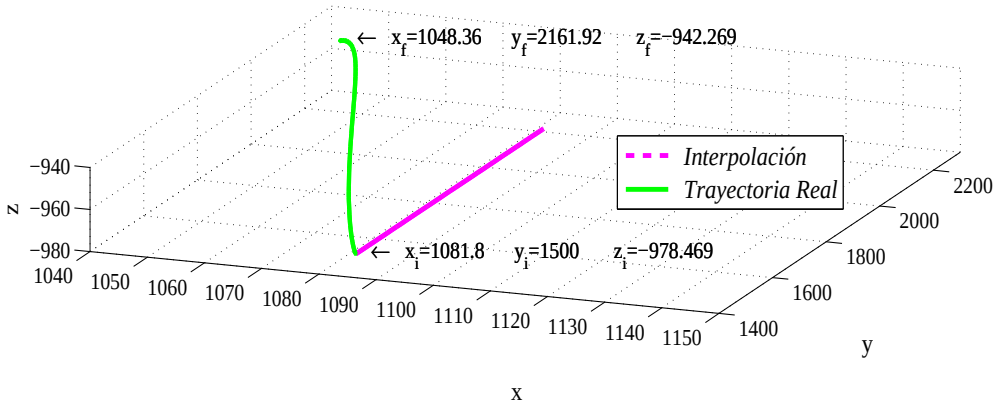
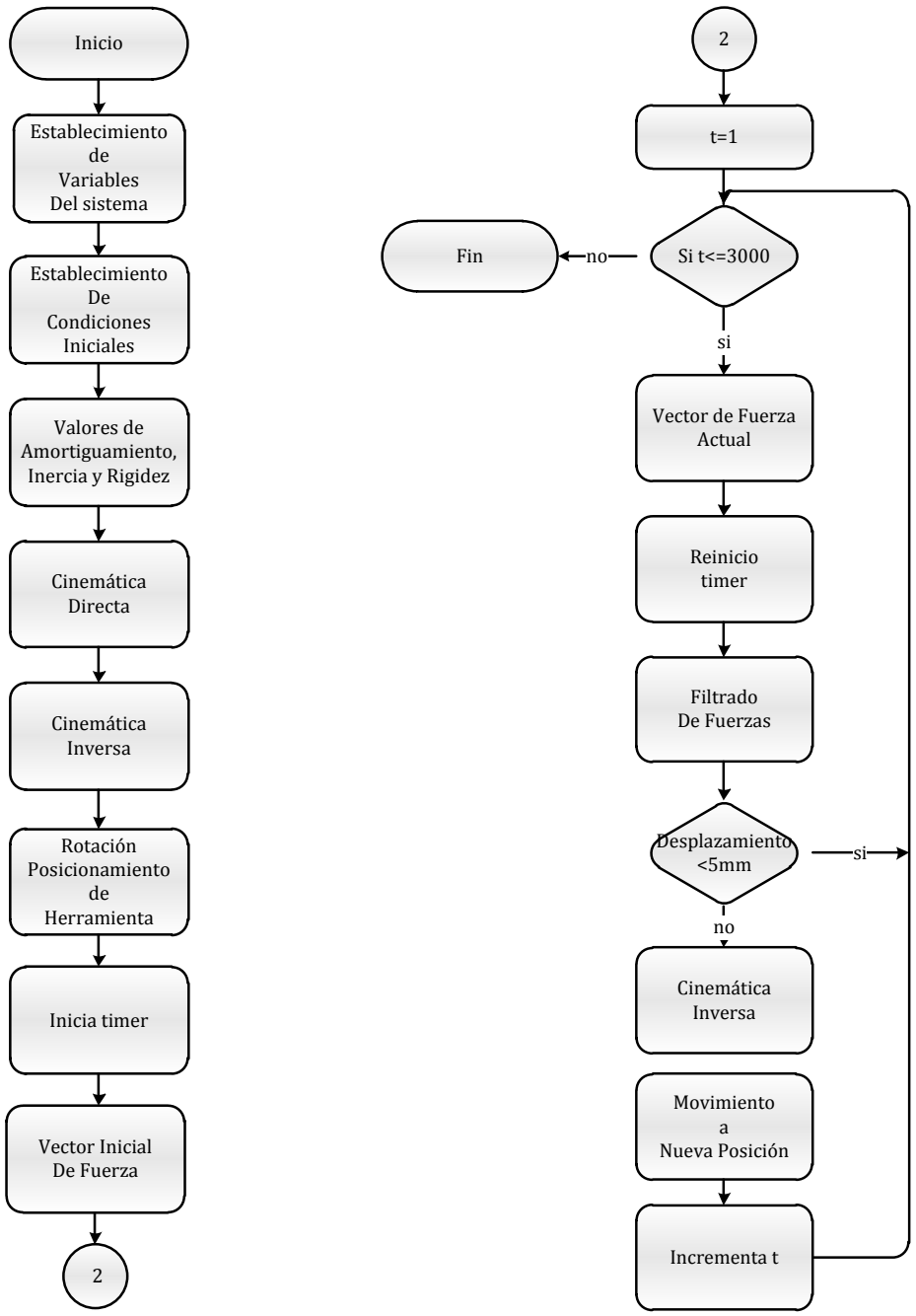


Figura 3.5. Simulación y trayectoria real generada por el robot M-16iB/20T con fuerzas de interacción programadas.



(a) Condiciones iniciales (b) Control de impedancia

Figura 3.6. Diagrama de flujo de programa de control de impedancia

3.3. Robot LR Mate-200iC

3.3.1. Plataforma Experimental

A diferencia del robot anterior, en este manipulador ya se cuenta con las variables que genera el sensor de fuerza FS-10iA. Estas variables están representadas como un vector de 6 elementos, expresado de la siguiente manera:

$$V_f = \begin{bmatrix} F_x \\ F_y \\ F_z \\ M_x \\ M_y \\ M_z \end{bmatrix} \tag{3.3}$$

Donde $[F_x, F_y, F_z]^T$ son las fuerzas detectadas con el sensor que se asocian con la posición de la herramienta, mientras que $[M_x, M_y, M_z]^T$ representa los torques medidos, relacionados con la orientación del efector final.

(a) Sensor y Aditamento (b) Robot, sensor y aditamento integrado.

Figura 3.7. Sensor de fuerza FS-10iA y aditamento para transmisión de fuerzas integrado.

Para un mejor manejo del sensor de fuerza, fue necesaria la construcción de un aditamento al sensor, el cual fue manufacturado en la misma UASLP. Gracias a este aditamento, es posible transmitir la fuerza de una mano humana, al sensor. La Figura 3.7 muestra tanto al sensor de fuerza, como al citado aditamento.

En las siguientes subsecciones se presentan los resultados obtenidos a lo largo de esta

investigación de tesis. Para corroborar los experimentos, se creó una plataforma experimental que consiste en una rampa, mostrada en la Figura 3.8, y se programa al robot para que siga una trayectoria mostrada en la misma figura. El control de impedancia debe lograr un seguimiento de la rampa mostrada, y posteriormente continuar con la trayectoria designada por interpolación.

Figura 3.8. Plataforma experimental para control de impedancia.
Trayectoria comandada en línea verde.

Para los casos de retenedor de orden cero y método numérico de Runge-Kutta se presenta cada punto mencionado a continuación:

- Tipo de Sistema.
- Comentarios del experimento.
- Tabla con los parámetros de impedancia.
- Simulación en MATLAB del sistema propuesto.
- Fuerzas y torques medidos en la interacción.
- Tiempo de muestreo.
- Trayectoria real y trayectoria deseada.

3.3.2. Retenedor de Orden Cero

Se implementó un algoritmo de control de impedancia en el manipulador **LR Mate-200iC**, basado en la plataforma experimental propuesta. Si se percibe una fuerza en el sensor, entonces la herramienta desviará su posición, a la nueva calculada por la diferencia de posiciones entre la trayectoria deseada y el vector de posición generado por el retenedor de orden cero. Es necesario el uso de tres diferentes casos de amortiguamiento, como se especificó en la sección 2.3.1. Además, este enfoque depende de los valores de la frecuencia natural no amortiguada del sistema ω_n y de

la relación de amortiguamiento del sistema ξ, esto con la finalidad de utilizar parámetros más intuitivos. La ecuación que describirá el sistema para los siguientes tres casos es la ecuación (3.4):

$$\ddot{x} + 2\xi\omega_n\dot{x} + \omega_n^2 x = \frac{2\xi\omega_n}{b_d}F \tag{3.4}$$

3.3.2.1. Caso sub amortiguado

Se seleccionan los parámetros adecuados para este amortiguamiento y se programan las ecuaciones de la sección 2.3.1.1, con los valores impedancia de la Tabla 3.2. La simulación del comportamiento para este caso puede apreciarse en la Figura 3.9, la cual representa la salida del sistema ante una entrada escalón de 1 Kgf. De esta figura se puede establecer que, ante la entrada de una unidad de fuerza, se produce una salida de 10 mm de desplazamiento.

Para el caso experimental del seguimiento de la rampa, este tipo de sistema genera desplazamientos muy alejados al contacto con la superficie, por lo que no es la selección más adecuada para una tarea de este tipo. Sin embargo, para el caso de interacción humano robot, se percibe un comportamiento muy dócil.

Los tiempos de muestreo que se presentaron en la ejecución del programa, se muestran en la Figura 3.10(c). La media y la desviación estándar de estos tiempos de muestreo, están contenidas en la Tabla 3.3. Se puede observar en ésta, que la media para este experimento tiene un valor de 82.0 ms.

La Figura 3.10(a) muestra los tres instantes en que se presentó fuerza de contacto con la superficie, con un valor máximo de 16 Kgf , mientras que la Figura 3.10(b) muestra un torque máximo de -24 Kgf-cm, generando tanto vector de ajuste de posición en el sistema, como de orientación.

La Gráfica 3.10(d) muestra la trayectoria comandada al robot por medio de interpolación y el comportamiento real del robot, ante las cargas mencionadas.

Parámetros	Valor	Unidades
ξ	0.790	$-$
ω_n	3.162	rad/s
b_d	0.050	$Kgf \cdot s/m$

Tabla 3.2. Parámetros de impedancia, caso sub amortiguado. Retenedor de orden cero.

$\bar{X}$	σ
0.0820 s	0.0793 s

Tabla 3.3. Media y desviación estándar del tiempo de muestreo, caso sub amortiguado. Retenedor de orden cero.

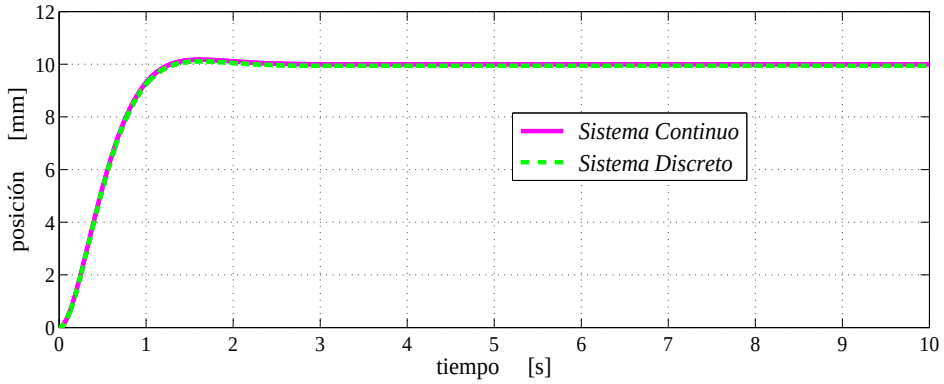
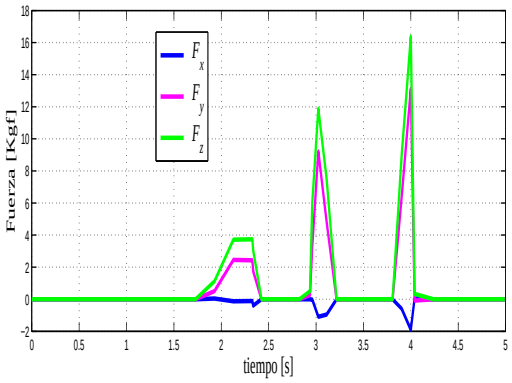
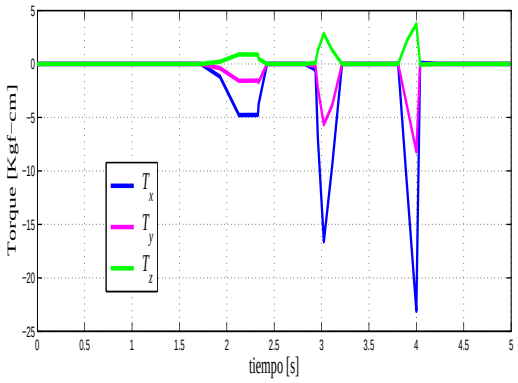


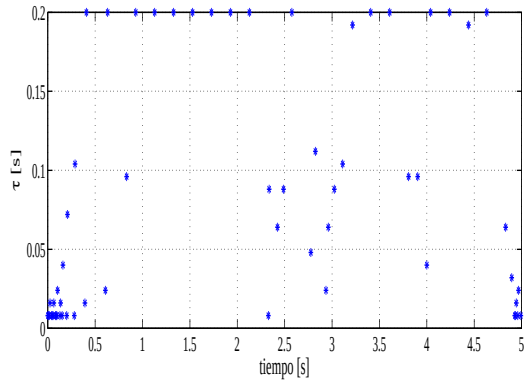
Figura 3.9. Simulación del sistema, $\tau = 10ms$



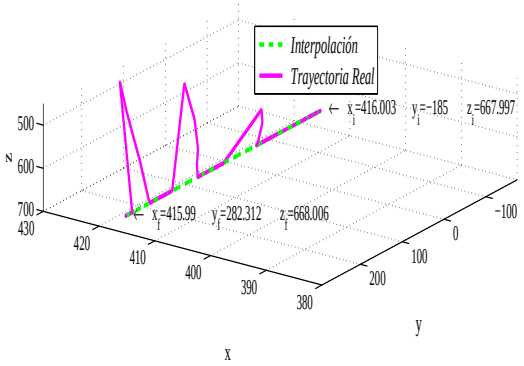
(a) Fuerzas medidas en la interacción



(b) Torques medidos en la interacción



(c) Tiempo de muestreo



(d) Interpolación y trayectoria real

Figura 3.10. Caso sub amortiguado.
Retenedor de orden cero.

3.3.2.2. Caso críticamente amortiguado

Se programan las ecuaciones de la sección 2.3.1.1, con los parámetros de la Tabla 3.4. El comportamiento en simulación de este caso, está dado por la Gráfica 3.11. Para este tipo de sistema, una entrada de 1 Kgf proporciona una salida de 0.5 mm, por lo que para mover la misma distancia que el caso anterior, es decir 10 mm, se tendrían que aplicar 20 Kgf.

Las fuerzas y torques presentes se muestran en las Figuras 3.12(a) y 3.12(b) respectivamente. La fuerza máxima presente es de 10 Kgf ya que, al ser un caso críticamente amortiguado, responde más rápido al contacto, evitando fuerzas mayores.

Con la misma cantidad de fuerza que el caso anterior, la distancia de salida es menor y se percibe un sistema rígido al contacto humano de manera experimental. El recorrido sobre la rampa lo realiza alejándose menos de la superficie de contacto. Los tiempos de muestreo del sistema, para este caso, se presenta en la Figura 3.12(c). La media y desviación estándar se pueden ver en la Tabla 3.5. Como puede apreciarse en la misma, el valor medio del tiempo de muestreo es de 67.6 ms.

La interpolación programada y la trayectoria real seguida en el experimento por el robot, se presentan en la Figura 3.12(d). Nuevamente se tienen tres contactos, pero en la trayectoria real, se aprecia una disminución del valor máximo de desplazamiento del sistema contra la trayectoria deseada. También disminuye el valor máximo de fuerza y torque de contacto.

Parámetros	Valor	Unidades
ξ	1.000	–
ω_n	3.162	rad/s
b_d	1.264	$Kgf \cdot s/m$

Tabla 3.4. Parámetros de impedancia, caso críticamente amortiguado.
Retenedor de orden cero.

$\bar{X}$	σ
0.0676 s	0.0749 s

Tabla 3.5. Media y desviación estándar del tiempo de muestreo, caso críticamente amortiguado.
Retenedor de orden cero.

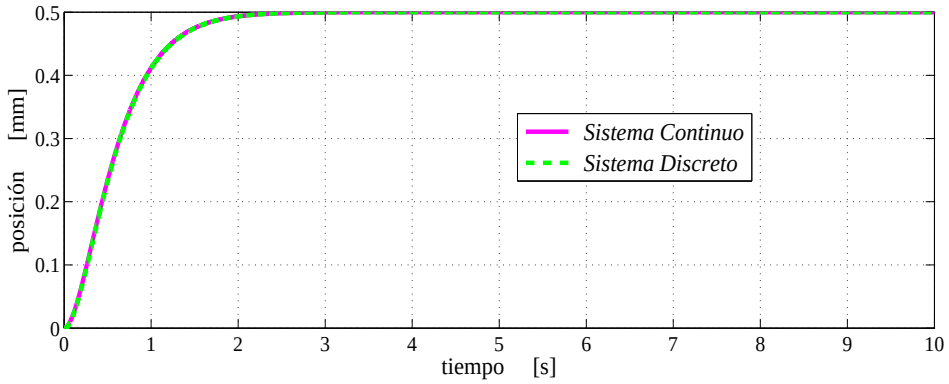
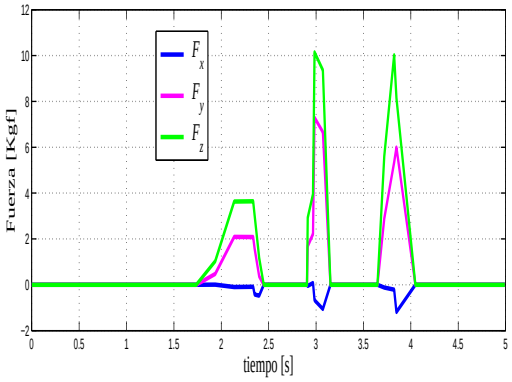
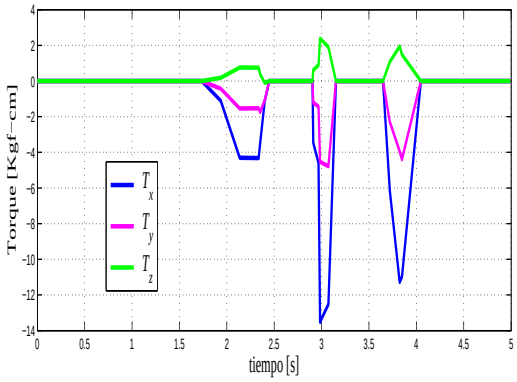


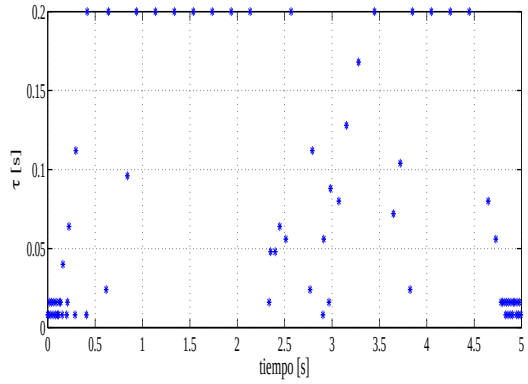
Figura 3.11. Simulación del sistema, $\tau = 10ms$



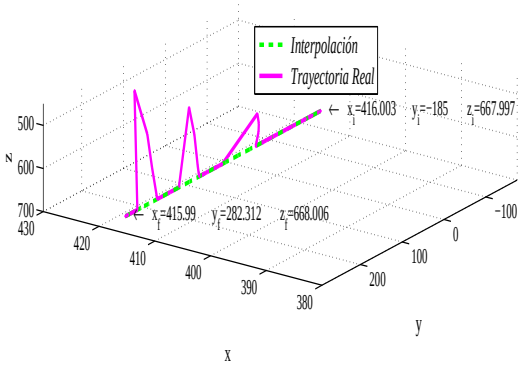
(a) Fuerzas medidas en la interacción



(b) Torques medidos en la interacción



(c) Tiempo de muestreo



(d) Interpolación y trayectoria real

Figura 3.12. Caso críticamente amortiguado.
Retenedor de orden cero.

3.3.2.3. Caso sobre amortiguado

Para el algoritmo de control de este tipo de sistema, se programan las ecuaciones de la sección 2.3.1.1. Se obtiene un comportamiento de sistema sobre amortiguado en simulación, con el tiempo de restablecimiento mayor a los casos anteriores. Puede apreciarse que se espera un desplazamiento de 2 mm, para una unidad de fuerza de entrada al sistema.

Los parámetros de impedancia de la Tabla 3.6 generan los tiempos de muestreo de la Figura 3.7, de manera experimental. Las fuerzas y torques medidos en esta prueba se pueden ver en las Figuras 3.14(a) y 3.14(b). En este caso, las fuerzas y torques del experimento presentan valores más elevados que los casos anteriores, esto se atribuye al sobre amortiguamiento del sistema, que provoca una resistencia al movimiento del robot.

Finalmente, con los datos anteriores, la trayectoria del sistema y su interpolación pueden verse en la Gráfica 3.14(d), donde se aprecian las cuatro interacciones de fuerza. Para el caso del seguimiento de la rampa, en la misma figura se puede apreciar que las distancias de la trayectoria real y la interpolación se ven disminuidas, es decir, el robot se aleja menos de la superficie a seguir, comportamiento que es adecuado a este tipo de tarea de interacción. Se puede apreciar en la Figura 3.14(c), que los tiempos de muestreo también se reducen. El tiempo de muestreo se encuentra en un valor medio de 33.3 ms. Sin embargo, para el caso de contacto humano, el sistema se percibe rígido.

Parámetros	Valor	Unidades
ξ	2.236	–
ω_n	2.236	rad/s
b_d	1.000	$Kgf \cdot s/m$

Tabla 3.6. Parámetros de impedancia, caso sobre amortiguado.
Retenedor de orden cero.

$\bar{X}$	σ
0.0333 s	0.0538 s

Tabla 3.7. Media y desviación estándar del tiempo de muestreo, caso sobre amortiguado.
Retenedor de orden cero.

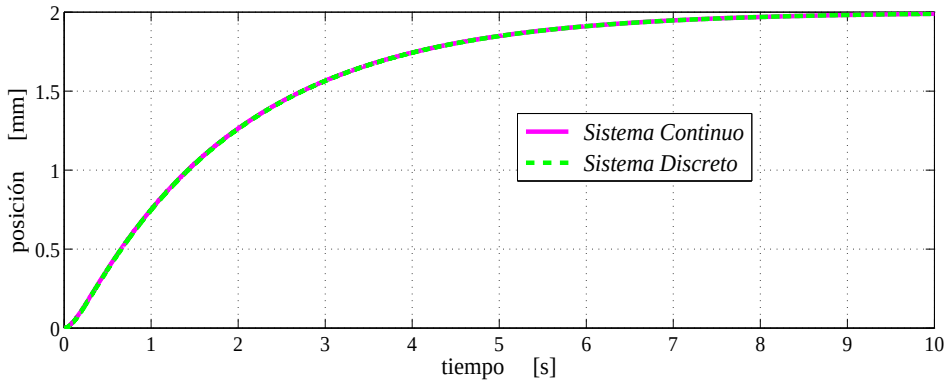
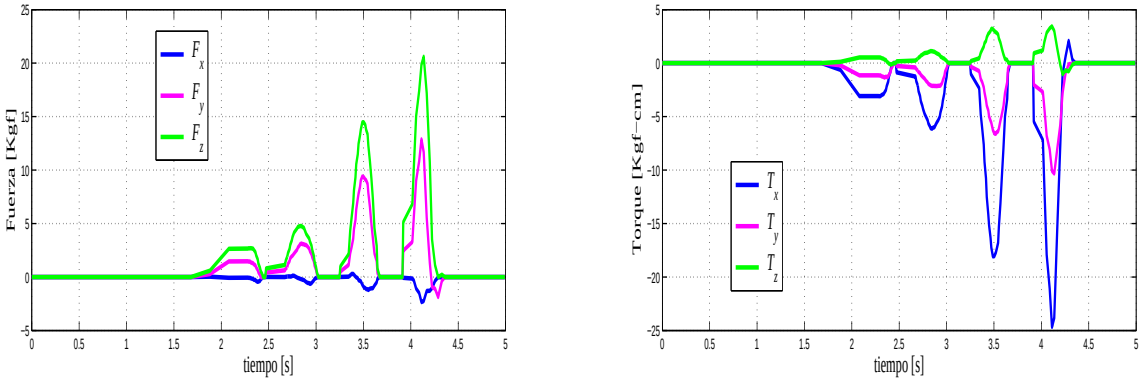
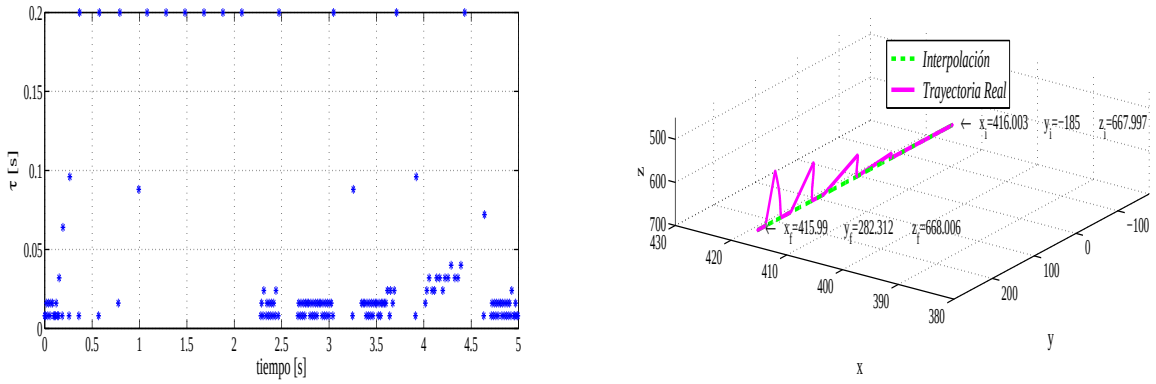


Figura 3.13. Simulación del sistema, $\tau = 10ms$



(a) Fuerzas medidas en la interacción

(b) Torques medidos en la interacción



(c) Tiempo de muestreo

(d) Interpolación y trayectoria real

Figura 3.14. Caso sobre amortiguado.
Retenedor de orden cero.

3.3.3. Método Numérico de Runge-Kutta de Segundo Orden

Se desarrollaron los tres casos de amortiguamiento para este tipo de filtrado, con la finalidad de tener un estudio comparativo entre el comportamiento del retenedor de orden cero, y este método. Para este último, no es necesario desarrollar ecuaciones para cada tipo de sistema, ya que existe una solución única para todo caso. Esto mejora el tiempo de ejecución del algoritmo en el desarrollo experimental. Además, se utilizan variables en función del amortiguamiento b_d, rigidez k_d e inercia m_d deseados.

3.3.3.1. Caso sub amortiguado

La simulación de este sistema tiene el mismo comportamiento que la del retenedor de orden cero, como puede observarse en la Figura 3.19. Los tiempos de muestreo se presentan en la Gráfica 3.16(c) y las estadísticas de la media y desviación estándar en la Tabla 3.9. Mientras que el retenedor de orden cero tiene estadísticamente una media en el tiempo de muestreo $\overline{X} = 82$ ms, el método de Runge-Kutta presenta, para este mismo caso, un valor de $\overline{X} = 32.6$ ms.

Para el caso de Runge-Kutta, el comportamiento sub amortiguado es el que presenta un mejor desempeño en el seguimiento de la rampa de manera experimental. Las gráficas de fuerza y torque, Figuras 3.16(a) y 3.16(b) respectivamente, muestran que la fuerza presente en los experimentos es mayor a la que se presentó en el retenedor de orden cero. Al reducirse el tiempo de muestreo, se aprecia que se logran 4 interacciones para este caso. La desviación de la interpolación y la trayectoria real es la más baja de todos los experimentos, esto puede apreciarse en la Figura 3.16(d). Este sistema tiene un comportamiento dócil en la interacción humano-robot, por lo que se emplean los mismos valores de impedancia para el desarrollo del experimento denominado "punto fijo".

Parámetros	Valor	Unidades
m_d	0.010	Kg
k_d	0.100	Kgf/m
b_d	0.050	$Kgf \cdot s/m$

Tabla 3.8. Parámetros de impedancia, caso sub amortiguado.
Método numérico Runge-Kutta de segundo orden.

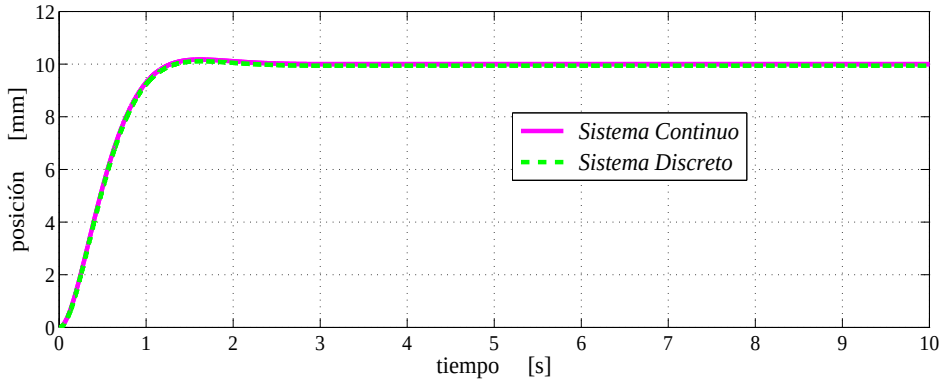
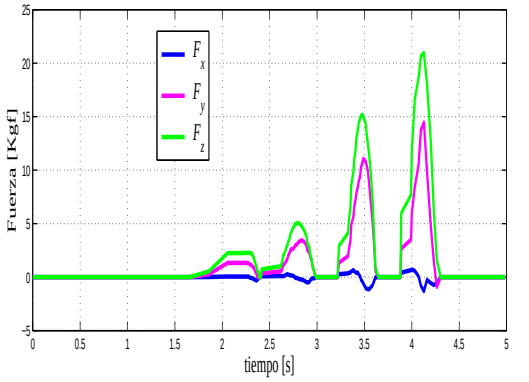
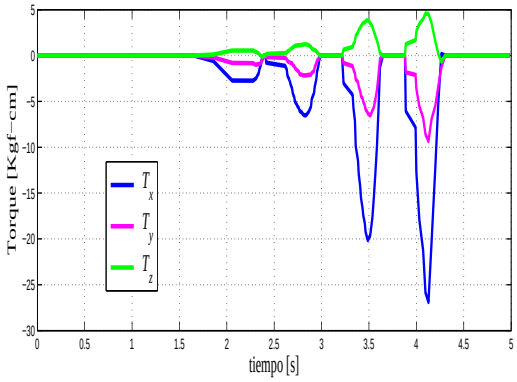


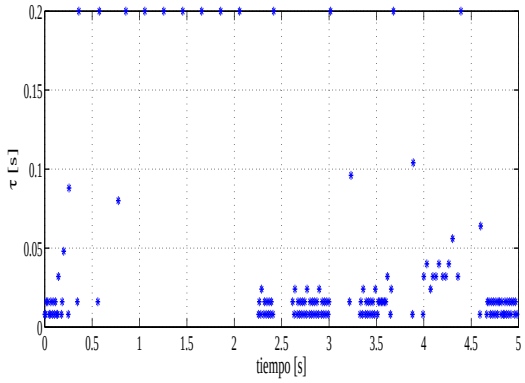
Figura 3.15. Simulación del sistema, $\tau = 10ms$



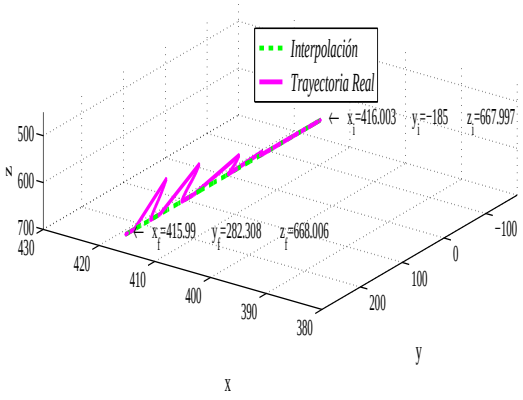
(a) Fuerzas medidas en la interacción



(b) Torques medidos en la interacción



(c) Tiempo de muestreo



(d) Interpolación y trayectoria real

Figura 3.16. Caso sub amortiguado.
Método numérico Runge-Kutta de segundo orden.

$\bar{X}$	σ
0.0337 s	0.0541 s

Tabla 3.9. Media y desviación estándar del tiempo de muestreo, caso sub amortiguado. Método numérico Runge-Kutta de segundo orden.

3.3.3.2. Caso críticamente amortiguado

La gráfica de la simulación de este sistema se presenta en la Figura 3.19. Para el caso experimental, las fuerzas medidas en la interacción se muestran en la Gráfica 3.18(a) y los torques en la Figura 3.18(b). Como puede observarse en ellas, las fuerzas y torques presentes son altos y se presentan además 4 interacciones. El comportamiento del tiempo de muestreo durante la experimentación se puede observar en la Gráfica 3.18(c), mientras que la media y desviación estándar del mismo, están contenidas en la Tabla 3.11.

Retomando el retenedor de orden cero, caso críticamente amortiguado, el valor medio del tiempo de muestreo $\overline{X}$ presentaba un valor de $67.6ms$. Para el caso de Runge-Kutta, $\overline{X} = 32.6ms$. Nuevamente, el tiempo de muestreo con el método numérico se reduce significativamente, mejorando a la vez el tiempo de ejecución del algoritmo de control. Se presenta como última figura de este caso, la gráfica de la trayectoria real y la interpolación de este experimento, aunque se alejan más de la trayectoria de interpolación que el caso anterior sub amortiguado, la respuesta al seguimiento sigue siendo superior a los casos del retenedor.

Parámetros	Valor	Unidades
m_d	0.200	Kg
k_d	2.000	Kgf/m
b_d	1.264	$Kgf \cdot s/m$

Tabla 3.10. Parámetros de impedancia, caso críticamente amortiguado.
Método numérico Runge-Kutta de segundo orden.

$\bar{X}$	σ
0.032.6 s	0.0531 s

Tabla 3.11. Media y desviación estándar del tiempo de muestreo, caso críticamente amortiguado.
Método numérico Runge-Kutta de segundo orden.

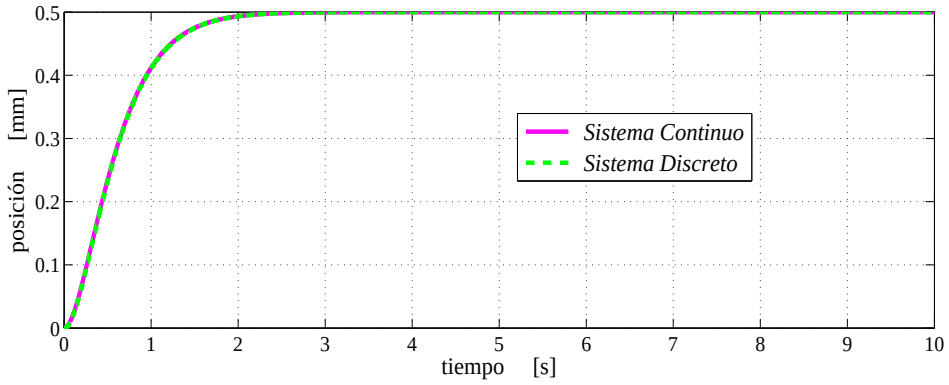
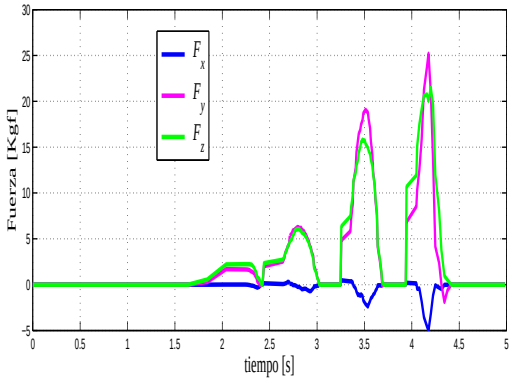
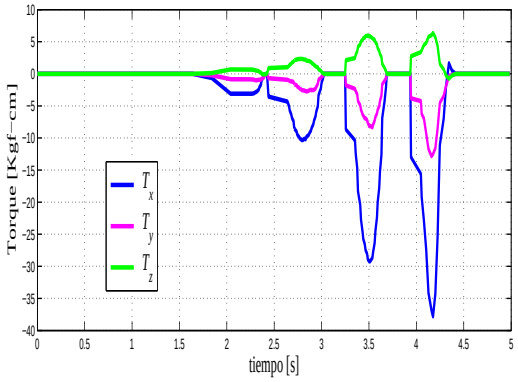


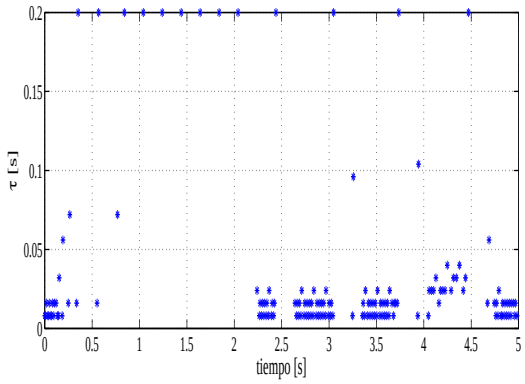
Figura 3.17. Simulación del sistema, $\tau = 10ms$



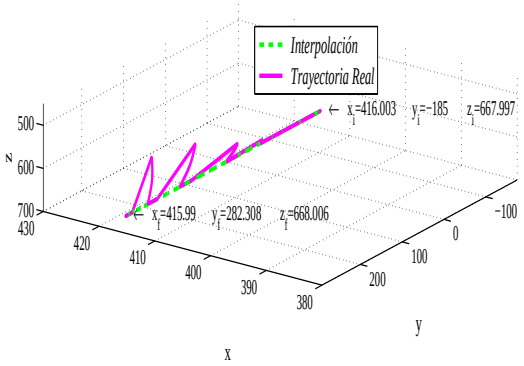
(a) Fuerzas medidas en la interacción



(b) Torques medidos en la interacción



(c) Tiempo de Muestreo



(d) Interpolación y trayectoria real

Figura 3.18. Caso críticamente amortiguado.
Método numérico Runge-Kutta de segundo orden.

3.3.3.3. Caso sobre amortiguado

Para este último caso, se presentan de igual manera las gráficas características del sistema. La simulación del sistema, en la Figura 3.19, nos permite ver el comportamiento esperado de 2 mm de desplazamiento por unidad de fuerza aplicada en la entrada. Las Figuras 3.20(a) y 3.20(b), presentan fuerzas máximas de aproximadamente 18 Kgf y torques de -25 Kgf-cm. Se presentan tres contactos con la superficie, lo que nos sugiere que la rigidez del sistema se percibirá alta en la interacción humano-robot. La trayectoria real y la estimada por interpolación, se muestran en la Figura 3.20(d); debido al aumento de fuerza, los desplazamientos del sistema se alejan más de la rampa.

El tiempo de muestreo para este caso, está graficado en la Figura 3.20(c). Los valores de la media y la desviación estándar del mismo están presentes en la Tabla 3.13. Comparando con el retenedor de orden cero, cuyo valor medio $\overline{X}$ está establecido en 33.3 ms en el caso sobre amortiguado, se tiene que el método de Runge-Kutta presenta un tiempo de muestreo mayor, de $\overline{X} = 59.5ms$. Como puede verse, este es el único caso para el cual el retenedor de orden cero ofrece un mejor tiempo de muestreo promedio.

De manera práctica, esta prueba genera fuerzas de contacto altas, pero realiza un seguimiento adecuado de la rampa, aunque su comportamiento con interacción humano-robot se percibe rígido con estos parámetros.

Parámetros	Valor	Unidades
m_d	0.100	Kg
k_d	0.500	Kgf/m
b_d	1.000	$Kgf \cdot s/m$

Tabla 3.12. Parámetros de impedancia, caso sobre amortiguado.
Método numérico Runge-Kutta de segundo orden.

$\bar{X}$	σ
0.0595 s	0.0708 s

Tabla 3.13. Media y desviación estándar del tiempo de muestreo, caso sobre amortiguado. Método numérico Runge-Kutta de segundo orden.

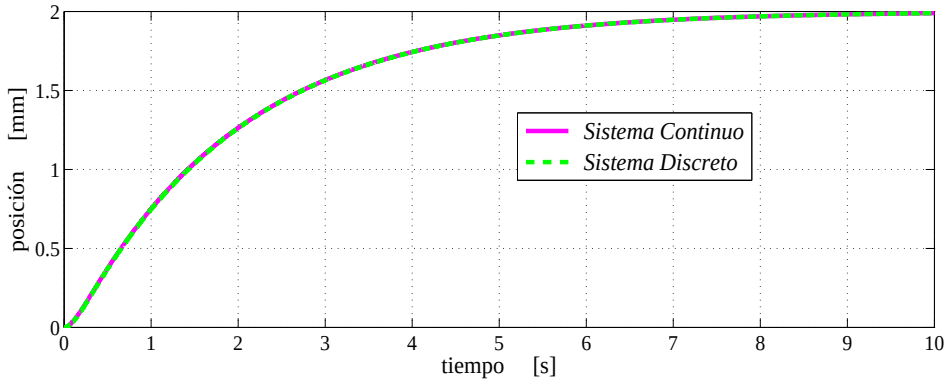
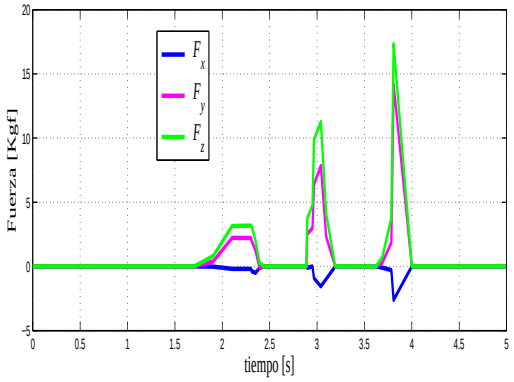
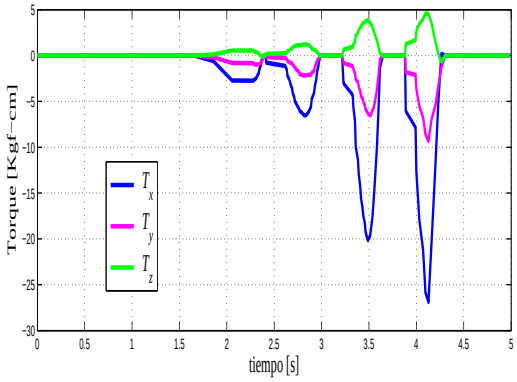


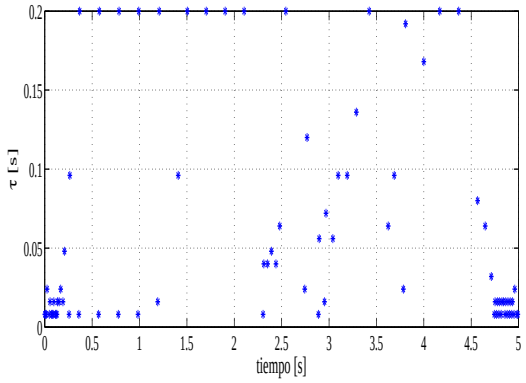
Figura 3.19. Simulación del sistema, $\tau = 10ms$



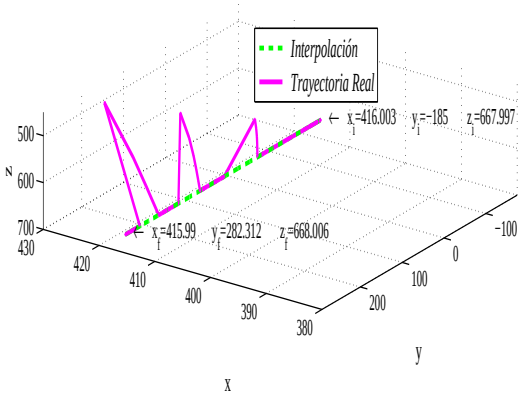
(a) Fuerzas medidas en la interacción



(b) Torques medidos en la interacción



(c) Tiempo de muestreo



(d) Interpolación y trayectoria real

Figura 3.20. Caso sobre amortiguado.
Método numérico Runge-Kutta de segundo orden.

3.3.3.4. Robot fijo

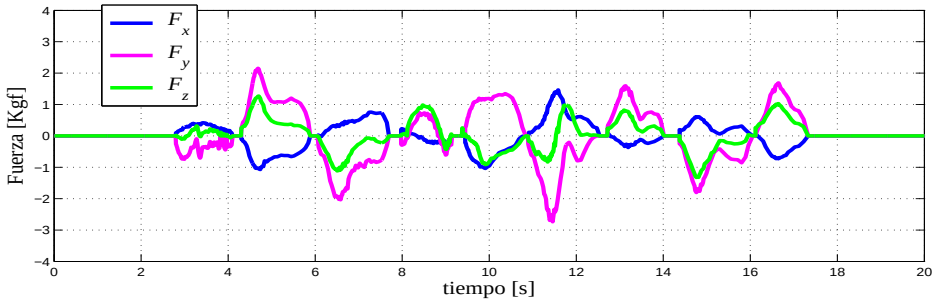
Para este caso, se realiza un segundo experimento en el cual se mantiene al robot en un punto fijo, y solo cambia de posición cuando una fuerza es aplicada. Este procedimiento ayuda a verificar que el programa envía al controlador las señales de fuerzas adecuadas, y que el robot genera movimientos en la dirección correcta. Se selecciona, de los casos anteriores, el que genera un movimiento más suave al tacto ante la presencia de una interacción, el cual es el caso subamortiguado. Se establece como punto fijo, las coordenadas del extremo del aditamento en la posición cartesiana: $(416, -185, 668)$

Al aplicarse las fuerzas mostradas en la Gráfica 3.21(a), se obtiene un desplazamiento de la herramienta, indicado por la Gráfica 3.21(d). El tiempo de muestreo se puede apreciar en la Figura 3.21(c). La Tabla 3.9 muestra los valores de la media y la desviación estándar de este tiempo de muestreo; para este caso, $\overline{X} = 7.1ms$. Este tiempo es más bajo que cualquiera de los mostrados en los casos anteriores, lo que se atribuye al hecho de que el algoritmo no genera ecuaciones adicionales para el cálculo de una posición deseada ya que, como se comentó anteriormente, esto genera tiempos de ejecución más elevados, incrementando a su vez el tiempo de muestreo.

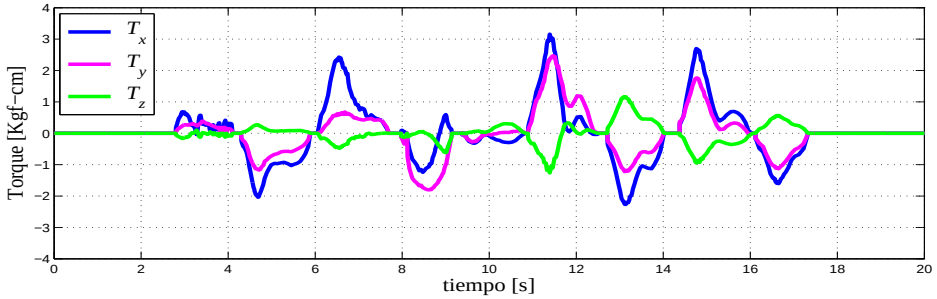
$\bar{X}$	σ
0.0071 s	0.0141 s

Tabla 3.14. Media y desviación estándar del tiempo de muestreo, caso sobre amortiguado. Método numérico Runge-Kutta de segundo orden.

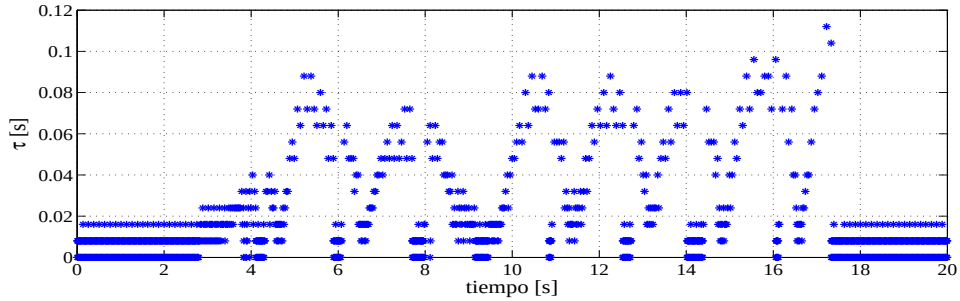
Se pueden apreciar claramente los movimientos guiados por el robot, pues al no existir ninguna trayectoria definida, todo movimiento proviene del filtrado de fuerzas proporcionado por el algoritmo de control. Las gráficas tanto de las fuerzas y torques de interacción se presentan en las Figuras 3.21(a) y 3.21(b), de aquí se puede tener una noción de lo que un humano sano, puede generar en fuerza para una tarea de rehabilitación, referido a los parámetros aquí establecidos. La trayectoria realizada por el robot se grafica en 3.21(d) la cual, como puede observarse, es una trayectoria aleatoria.



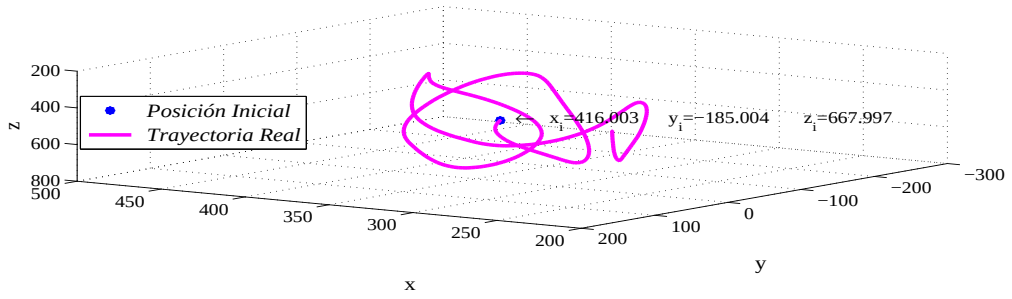
(a) Fuerzas medidas en la interacción



(b) Torques medidos en la interacción



(c) Tiempo de muestreo



(d) Posición inicial y trayectoria real

Figura 3.21. Método numérico Runge-Kutta de segundo orden.
Robot con posición inicial fija

3.3.4. Conclusiones Filtrado de Fuerzas

La comparación de resultados experimentales entre los dos tipos de filtrado, arrojan algunos puntos importantes por mencionar. Para una mejor apreciación del comportamiento de los tiempos de muestreo, se puede consultar la Tabla 3.15 que muestra los resultados totales.

Método	Condición de Amortiguamiento	$\bar{X}$	σ
ZOH	Sub amortiguamiento	0.0820 s	0.0793 s
ZOH	Amortiguamiento crítico	0.0676 s	0.0749 s
ZOH	Sobre amortiguamiento	0.0333 s	0.0538 s
RK	Sub amortiguamiento	0.0337 s	0.0541 s
RK	Amortiguamiento crítico	0.0326 s	0.0531 s
RK	Sobre amortiguamiento	0.0595 s	0.0708 s
PF	Sub amortiguamiento	0.0071 s	0.0141 s

Tabla 3.15. Tabla comparativa de los resultados obtenidos en los experimentos.

donde las abreviaciones, significan lo siguiente

Tipo de experimento:

- ZOH: Retenedor de orden cero.
- RK: Método genérico de Runge-Kutta.
- PF: Punto Fijo.

Medidas estadísticas:

- $\bar{X}$: Media.
- σ: Desviación estándar.

El retendor de orden cero es una buena estrategia, pero debido al cálculo de numerosas ecuaciones y a la naturaleza secuencial del lenguaje KAREL, genera tiempos de muestreo que, son del orden de los milisegundos, pero no son adecuados para la tarea de interacción que se presenta. Además, se buscaba justamente desarrollar las medidas necesarias para reducir el tiempo de muestreo del programa.

El método numérico de Runge-Kutta, por otro lado, reduce las ecuaciones programadas en el controlador, ofreciendo tiempos de muestreo menores al retenedor, y con el mismo nivel de precisión que éste. Esto es observable en el comportamiento de ambos sistemas ante los mismos valores de

impedancia. De la Tabla 3.15 se puede apreciar que de los 6 experimentos sobre la rampa, el mejor tiempo de muestreo lo generó el caso sub amortiguado. En el caso del punto fijo, el cual se toma como experimento único, se presenta una disminución del tiempo de muestreo aunque las características de impedancia son las mismas que para el caso de Runge-Kutta sub amortiguado. Esto se atribuye nuevamente a la disminución de ecuaciones debido a que no se programa una trayectoria, y a la característica de programación secuencial del lenguaje KAREL.

De las gráficas de fuerza y torque presentadas, se puede concluir que la variabilidad de rangos obtenidos obedece al tipo de movimiento ejecutado por el controlador, inherente al sistema y no al tipo de amortiguamiento programado. Se debe hacer referencia también a que la velocidad utilizada en las pruebas no es la velocidad máxima de operación del robot. El uso del mismo se limita al 10 % de su valor nominal. Es perceptible, debido a los experimentos realizados, que al incrementar la velocidad, disminuye el tiempo de muestreo. La limitación en velocidad es debida a razones de seguridad. También es importante mencionar que la velocidad de operación de un robot que interactúa con humanos, está restringida por normas, las cuales varían en cada país [18].

Conclusiones

Existen limitaciones al programar un manipulador de arquitectura abierta, debido a las características propias del diseño del robot. Algunas tales como el tipo de control, la rigidez natural del sistema, las limitaciones en velocidad, los tiempos mínimos de ejecución y la precisión. Pero por otro lado, los manipuladores industriales tienen también particularidades que, aunque son conocidas, no han sido explotadas al máximo y pueden mejorar significativamente el desarrollo de enfoques tecnológicos actuales, puesto que pueden ser usadas para tareas totalmente diferentes para las que originalmente fueron desarrolladas. Un caso particular de la amplia área de enfoques de la robótica es la terapia de rehabilitación que, como ya se muestra en [15], puede ofrecer resultados muy benéficos para pacientes afectados por una enfermedad vascular cerebral u otros tipos de apoplejía.

La ventaja que ofrece la programación de terapias de rehabilitación en un robot industrial es que, los robots médicos son tan especializados que generalmente están dedicados a la rehabilitación de solo una parte del cuerpo. En cambio, un robot industrial con los aditamentos necesarios, puede extrapolar sus funciones a otras extremidades.

Algo que es muy importante destacar, es que el comportamiento del sistema depende en gran medida de que los parámetros de impedancia sean seleccionados adecuadamente. En estos experimentos es perceptible que, en la selección de los mismos, intervienen diversos factores tales como: la superficie sobre la que se desarrolla la tarea, la tarea a desarrollar en cuestión, la calibración del sensor, e incluso la temperatura del controlador, parámetro que tiene influencia directa sobre el comportamiento del sensor de fuerza. Dentro de los objetivos del presente trabajo de investigación no se contempló la sintonización de dichos parámetros; sin embargo, se obtuvieron resultados que ilustran el efecto de los parámetros de impedancia en una tarea específica.

Una de las características importantes que se deben tener en cuenta para la programación en lenguaje KAREL, es que este lenguaje es de tipo secuencial y, debido a esta característica,

los tiempos de muestreo estarán directamente relacionados con el número de ecuaciones que se desarrollen dentro del programa. Si se requiere minimizar aún más los tiempos, se deben buscar alternativas que disminuyan los cálculos matemáticos o, por otro lado, generar nuevas soluciones de programación que permitan interrupciones en caso de presentarse ciclos que incrementen los tiempos de retardo.

Se desarrolló e implementó un algoritmo de control de impedancia dentro de un controlador industrial, con la finalidad de mejorar el tiempo de muestreo estimado en 400 ms, discutido en el enfoque desarrollado en [3]. Este objetivo fue alcanzado, reduciendo este tiempo de muestreo por medio de la técnica de resolución numérica de Runge-Kutta, hasta tiempos de muestreo variables con valores medios de 33.8 ms. Además, pudo comprobarse que el controlador R30iA del robot LR Mate 200iC tiene la capacidad necesaria para desarrollar procedimientos matemáticos extensos en tiempos de ejecución mínimos, ampliando las oportunidades para desarrollar programas que requieran cálculos de este tipo. Con esto se incrementan las posibilidades de extender este trabajo a otros controladores robóticos, que tengan capacidades de procesamiento similares o superiores a las que tiene el equipo empleado en el presente proyecto.

Trabajo a futuro

Se proponen las siguiente puntos, como continuación al trabajo presentado:

- Sintonización de parámetros de impedancia para cada tarea asignada. Un procedimiento para poder sintonizar estos parámetros sería muy conveniente para este tipo de control.
- Se requiere ubicar el área de trabajo del robot de acuerdo a la tarea de interacción, para evitar que, con los desplazamientos generados, llegue a áreas inalcanzables, deteniendo así el programa en ejecución.
- Se encuentra en desarrollo una tarea de rehabilitación específica para pacientes con enfermedad vascular cerebral, tomando como referencia este trabajo de investigación.
- Se requiere ampliar la seguridad para tareas que implican interacción con humanos, tanto por software como por hardware.
- Se sugiere una investigación más detallada de las características ofrecidas por el lenguaje de programación del robot industrial. Se pudieron anexar al presente algoritmo, variables de sistema que lo mejoraron substancialmente sin embargo, existe un gran número aún sin ser estudiadas o utilizadas.

Apéndices

Apéndice A

Robot FANUC M-16iB/20T

El robot **M-16iB/20T** es un robot fijo, montado sobre un riel, construido con seis grados de libertad y una muñeca esférica. Debido a su tamaño, tiene un gran espacio de trabajo disponible. Esta diseñado para operaciones a alta velocidad y configuración amigable al usuario. Entre sus características destacan, su *i*Pendant a color, carga máxima de 20 Kg y ± 0.10 mm de repetibilidad. Presenta un sistema de seguridad de fallas en cada junta. La Tabla A.1, muestra algunas de las especificaciones de este modelo, mientras que sus dimensiones son especificadas en la Figura A.2.

Figura A.1. Laboratorio de robótica UASLP. Robot M 16iB/20T

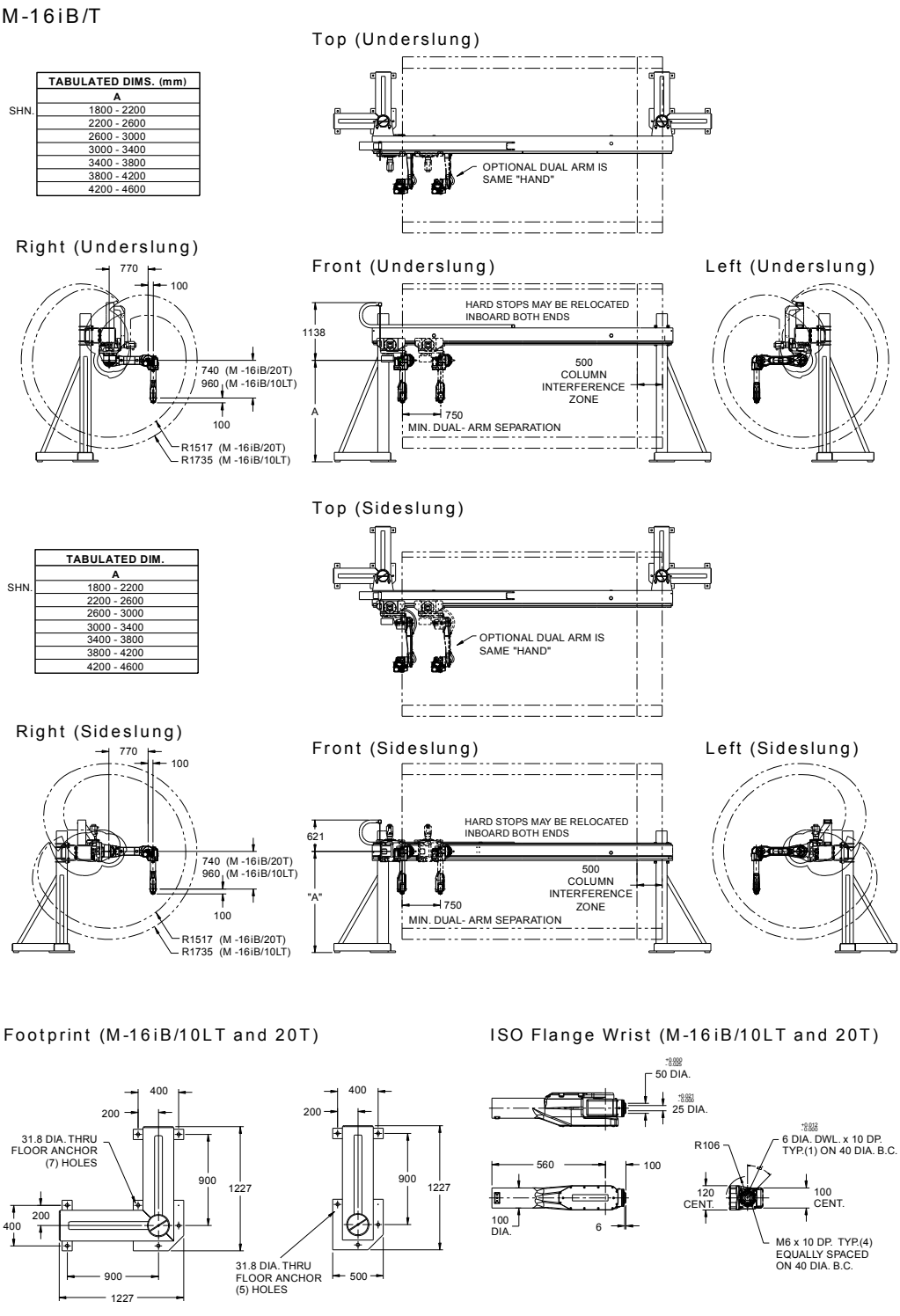


Figura A.2. Dimensiones del Manipulador M-16iB/20T. (Especificaciones en mm)

Item		
Ejes	6	
Carga Máxima	20 Kg	
Repetibilidad	± 0.10 mm	
Rango de movimiento	$J1$	$2.4 - 32$ m
	$J2$	$300°$
	$J3$	$460°$
	$J4$	$400°$
	$J5$	$280°$
	$J6$	$900°$
Velocidad de movimiento	$J1$	2750 mm/s
	$J2$	$165°/s$
	$J3$	$175°/s$
	$J4$	$350°/s$
	$J5$	$340°/s$
	$J6$	$520°/s$

Tabla A.1. Especificaciones del robot FANUC M-16iB/20T

Apéndice B

Robot FANUC LR Mate-200iC

El robot **LR Mate-200iC** es un robot que ofrece un excelente desempeño en cuanto a eficiencia, ligereza, precisión y agilidad. Sus dimensiones se pueden apreciar en la Figura B.1. Las características principales de este manipulador son: su brazo esbelto, peso ligero, gran destreza, gran velocidad y mayor precisión de posicionamiento. Es una herramienta útil para innumerables aplicaciones comerciales e industriales. Algunas tareas previas asignadas al robot son: manejo de materiales, ensamble, sujeción y colocación de objetos y remoción o distribución de material.

Cuenta con 6 grados de libertad, con muñeca esférica y presenta ± 0.02 mm de repetibilidad con carga máxima, y a la mayor velocidad que el robot puede alcanzar. La carga máxima es de 5 Kg. Tiene conexiones eléctricas y neumáticas para dispositivos externos ubicadas en la junta 4. Su diseño mecánico cerrado elimina cables y mangueras visibles, esto lo hace un diseño atractivo y a la vez práctico si se pretende utilizarlo con interacción con humanos. Presenta alta rigidez y servo tecnología, que permite movimientos finos sin vibraciones. Tiene opción a comunicación vía Ethernet y serial.

Su alimentación eléctrica es de 110 VAC monofásica. Permite movimiento de hasta 360 grados de liberad en la junta 1. Presenta un sistema de seguridad de fallas en cada junta. Su *i*pendant es a color, además de incorporar conección a internet y via celular usando los diseños de interface apropiados y es totalmente programable en lenguaje KAREL.

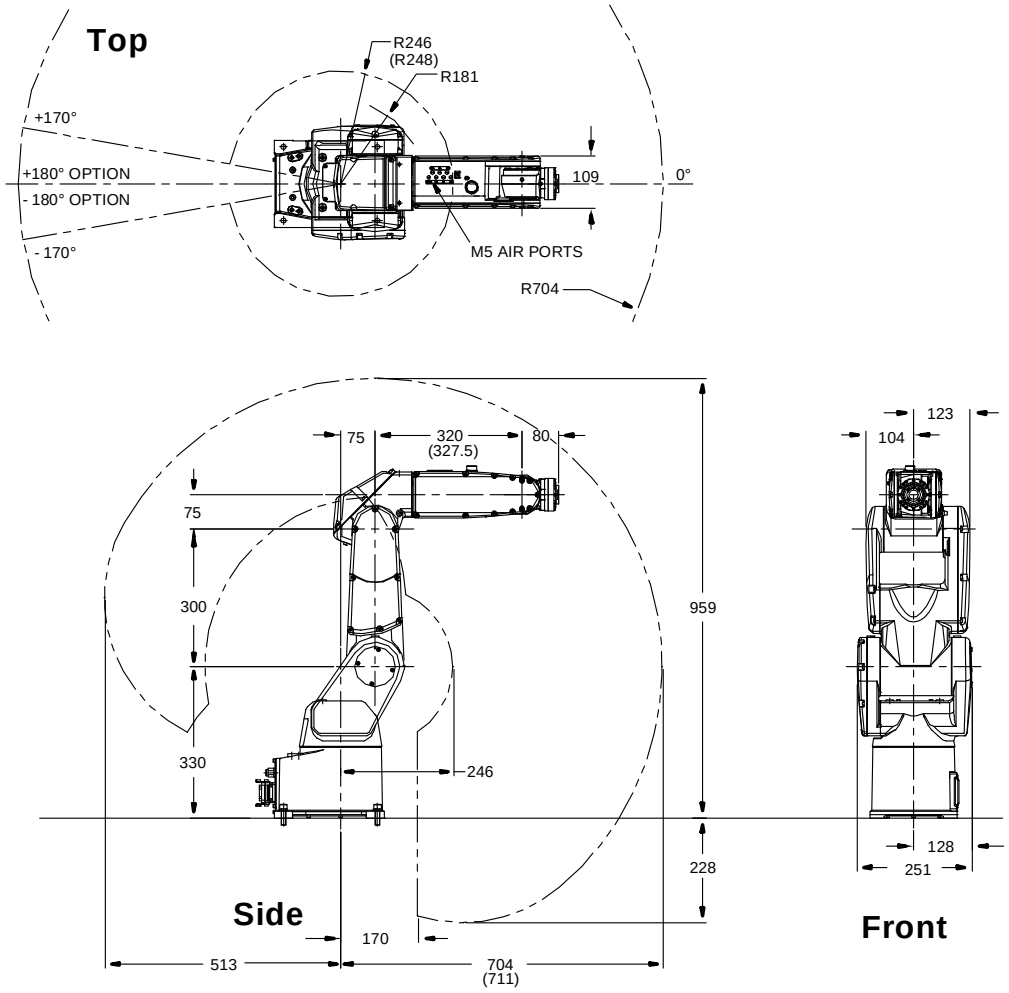


Figura B.1. Dimensiones del Manipulador LR Mate-200iC. (Especificaciones en mm)

Figura B.2. Laboratorio de robótica UASLP. Robot LR Mate-200iC

Item		
Ejes	6	
Carga Máxima	5 Kg	
Repetibilidad	± 0.020 mm	
Rango de movimiento	$J1$	$340° - 360°$
	$J2$	$200°$
	$J3$	$388°$
	$J4$	$380°$
	$J5$	$240°$
	$J6$	$720°$
Velocidad de movimiento	$J1$	$350°/s$
	$J2$	$350°$/s
	$J3$	$400°$/s
	$J4$	$450°$/s
	$J5$	$450°$/s
	$J6$	$720°$/s

Tabla B.1. Especificaciones del robot FANUC LR Mate-200iC

Apéndice C

Sensor de Fuerza FANUC FS-10iA

Aunque este sensor fue pensado para uso industrial originalmente, para manejo de objetos en 3-D, su uso puede ir mas allá de la manufactura. Las especificaciones del sensor de fuerza FANUC, pueden apreciarse en la Tabla C.1. Una imagen del mismo sensor, se muestra en la Figura C.1.

Item	
Dimensiones (mm)	90×43
Peso (Kg)	0.56
Carga nominal F_x, F_y, F_z	10 Kgf
Carga nominal T_x, T_y, T_z	80 Kgf-cm
Tolerancia a sobrecarga estática F_x, F_y, F_z	160 Kgf
Tolerancia a sobrecarga estática T_x, T_y, T_z	1280 Kgf-cm
Resolución F_x, F_y, F_z	0.040 Kgf
Resolución T_x, T_y, T_z	0.160 Kgf-cm
Precisión F_x, F_y, F_z	2 % o menos
Precisión T_x, T_y, T_z	2 % o menos

Tabla C.1. Especificaciones del sensor de fuerza FANUC FS-10iA

Figura C.1. Sensor de Fuerza FANUC FS-10iA

Apéndice D

Algoritmo de Control de Impedancia KAREL

Todos los comentarios en el programa presentado a continuación, tienen el siguiente formato: *'–Comentarios.'* Para poder compilar el programa, los apóstrofes deben ser eliminados.

```
'--Nombre del programa.'
PROGRAM TrkTD
'--Librerías utilizadas.'
%COMMENT = 'Orientación'
%RWACCESS
%STACKSIZE = 4000
%NOLOCKGROUP
%NOPAUSE=ERROR+COMMAND+TPENABLE
%ENVIRONMENT uif
%ENVIRONMENT sysdef
%ENVIRONMENT memo
%ENVIRONMENT kclop
%ENVIRONMENT bynam
%ENVIRONMENT fdev
%ENVIRONMENT flbt
%ENVIRONMENT STRNG
%INCLUDE klevccdf
%INCLUDE klevkeys
%INCLUDE klevkmsk
'--Variables globales del algoritmo.'
VAR
xyzd, xyzg ,xyzi, xyzf:XYZWPR
STATUS, q, p, o, entry, entry1,tf,num2,clock:INTEGER
value0, value1,vn:ARRAY[6] OF REAL
jpos:JOINTPOS
Fx1, Fy1, Fz1, Gx1, Gy1, Gz1, k, ko, umbral, or_umbral, t:REAL
```

```
27 limiteo, limite, limiteos, limites, xn, yn, zn, wn, pn, rn, abxn, abyn, abzn,
     abwn, abpn, abrn:REAL
28 graf:FILE
29 A7, Rh, rt:ARRAY [3] OF ARRAY [3] OF REAL
30 xa_1, vx_1, ya_1, vy_1, za_1, vz_1, wa_1, wx_1, pa_1, py_1, ra_1, rz_1:REAL
31 Md, Bd, Kd, Fx, Fy, Fz, Gx, Gy, Gz:REAL
32 Xa, Vx, Ya, Vy, Za, Vz, Wa, Wx, Pa, Py, Ra, Rz:REAL
33 var_x, var_y, var_z, var_w, var_p, var_r:BOOLEAN
34 '--Subrutina de intepolación, requiere la posición inicial, la posición final y
      el tiempo actual; genera la posición deseada'
35 ROUTINE interpol
36   BEGIN
37  xyzd.x=xyzi.x+(3*((xyzf.x-xyzi.x)/(tf*tf))*(t)*(t))-(2*((xyzf.x-xyzi.x)/(tf*
       tf*tf))*(t)*(t)*(t))
38  xyzd.y=xyzi.y+(3*((xyzf.y-xyzi.y)/(tf*tf))*(t)*(t))-(2*((xyzf.y-xyzi.y)/(tf*
       tf*tf))*(t)*(t)*(t))
39  xyzd.z=xyzi.z+(3*((xyzf.z-xyzi.z)/(tf*tf))*(t)*(t))-(2*((xyzf.z-xyzi.z)/(tf*
       tf*tf))*(t)*(t)*(t))
40  xyzd.w=xyzi.w+(3*((xyzf.w-xyzi.w)/(tf*tf))*(t)*(t))-(2*((xyzf.w-xyzi.w)/(tf*
       tf*tf))*(t)*(t)*(t))
41  xyzd.p=xyzi.p+(3*((xyzf.p-xyzi.p)/(tf*tf))*(t)*(t))-(2*((xyzf.p-xyzi.p)/(tf*
       tf*tf))*(t)*(t)*(t))
42  xyzd.r=xyzi.r+(3*((xyzf.r-xyzi.r)/(tf*tf))*(t)*(t))-(2*((xyzf.r-xyzi.r)/(tf*
       tf*tf))*(t)*(t)*(t))
43  xyzg=xyzd
44   END interpol
45 '--Subrutina de Runge-Kutta, genera el vector de ajuste de posición [Xa,Ya,Za,
     Wa,Pa,Ra].'
46 ROUTINE rungek
47 BEGIN
48 Vx=vx_1+(num2/1000)*(Fx/Md-Bd*(vx_1+1/2*(num2/1000)*(Fx/Md-Bd*vx_1/Md-Kd*xa_1/
     Md))/Md-Kd*(xa_1+1/2*(num2/1000)*vx_1)/Md)
49 Vy=vy_1+(num2/1000)*(Fy/Md-Bd*(vy_1+1/2*(num2/1000)*(Fy/Md-Bd*vy_1/Md-Kd*ya_1/
     Md))/Md-Kd*(ya_1+1/2*(num2/1000)*vy_1)/Md)
50 Vz=vz_1+(num2/1000)*(Fz/Md-Bd*(vz_1+1/2*(num2/1000)*(Fz/Md-Bd*vz_1/Md-Kd*za_1/
     Md))/Md-Kd*(za_1+1/2*(num2/1000)*vz_1)/Md)
51 Wx=wx_1+(num2/1000)*(Gx/Md-Bd*(wx_1+1/2*(num2/1000)*(Gx/Md-Bd*wx_1/Md-Kd*wa_1/
     Md))/Md-Kd*(wa_1+1/2*(num2/1000)*wx_1)/Md)
52 Py=py_1+(num2/1000)*(Gy/Md-Bd*(py_1+1/2*(num2/1000)*(Gy/Md-Bd*py_1/Md-Kd*pa_1/
     Md))/Md-Kd*(pa_1+1/2*(num2/1000)*py_1)/Md)
53 Rz=rz_1+(num2/1000)*(Gz/Md-Bd*(rz_1+1/2*(num2/1000)*(Gz/Md-Bd*rz_1/Md-Kd*ra_1/
     Md))/Md-Kd*(ra_1+1/2*(num2/1000)*rz_1)/Md)
54 Xa=xa_1+(num2/1000)*(vx_1+1/2*(num2/1000)*(Fx/Md-Bd*vx_1/Md-Kd*xa_1/Md))
55 Ya=ya_1+(num2/1000)*(vy_1+1/2*(num2/1000)*(Fy/Md-Bd*vy_1/Md-Kd*ya_1/Md))
56 Za=za_1+(num2/1000)*(vz_1+1/2*(num2/1000)*(Fz/Md-Bd*vz_1/Md-Kd*za_1/Md))
57 Wa=wa_1+(num2/1000)*(wx_1+1/2*(num2/1000)*(Gx/Md-Bd*wx_1/Md-Kd*wa_1/Md))
58 Pa=pa_1+(num2/1000)*(py_1+1/2*(num2/1000)*(Gy/Md-Bd*py_1/Md-Kd*pa_1/Md))
```

```
Ra=ra_1+(num2/1000)*(rz_1+1/2*(num2/1000)*(Gz/Md-Bd*rz_1/Md-Kd*ra_1/Md))
END rungek
'--Subrutina limite inferior, limita el desplazamiento mínimo tanto de posición
      como de orientación'
ROUTINE limitei
BEGIN
  limite=0
 IF abxn<limite THEN
       xyzd.x=xyzd.x
 else
  xyzd.x=xn
 END IF
  IF abyn<limite THEN
       xyzd.y=xyzd.y
 else
  xyzd.y=yn
 END IF
  IF abzn<limite THEN
       xyzd.z=xyzd.z
 else
  xyzd.z=zn
  END IF
  limiteo=2
 IF abwn<limiteo THEN
       xyzd.w=xyzd.w
 else
  xyzd.w=wn
 END IF
  IF abpn<limiteo THEN
       xyzd.p=xyzd.p
 else
  xyzd.p=pn
 END IF
  IF abrn<limiteo THEN
       xyzd.r=xyzd.r
 else
  xyzd.r=rn
  END IF
END limitei
'--Subrutina limite superior, limita el desplazamiento máximo de la herramienta
     .'
ROUTINE limitesup
BEGIN
  limites=200
 IF abxn>limites THEN
       xyzd.x=xyzd.x
 else
```

```
104   xyzd.x=xn
105  END IF
106   IF abyn>limites THEN
107         xyzd.y=xyzd.y
108  else
109   xyzd.y=yn
110  END IF
111   IF abzn>limites THEN
112         xyzd.z=xyzd.z
113  else
114   xyzd.z=zn
115   END IF
116   limiteos=5
117  IF abwn>limiteos THEN
118         xyzd.w=xyzd.w
119  else
120   xyzd.w=wn
121  END IF
122   IF abpn>limiteos THEN
123         xyzd.p=xyzd.p
124  else
125   xyzd.p=pn
126  END IF
127   IF abrn>limiteos THEN
128         xyzd.r=xyzd.r
129  else
130   xyzd.r=rn
131   END IF
132 END limitesup
133 '--Inicia programa principal.'
134 BEGIN
135 '--Variable que establece el límite de velocidad de operación del robot a la
      velocidad máxima alcanzable.'
136 $GROUP[1].$SPEED = $PARAM_GROUP[1].$SPEEDLIMJNT
137 '--Variable para minimizar el desplazamiento de giro'
138 $USE_TURNS=FALSE
139 '--Variable para alcanzar la máxima aceleración de un punto a otro.'
140 $GROUP[1].$USEMAXACCEL=TRUE
141 '--Variable para seleccionar la herramienta.'
142 $GROUP[1].$UTOOL = $MNUTOOL[1,1]
143 '--Establecimiento de las condiciones iniciales.'
144 var_x=FALSE;var_y=FALSE; var_z=FALSE
145 var_w=FALSE;var_p=FALSE; var_r=FALSE
146 xa_1=0; vx_1=0; ya_1=0; vy_1=0; za_1=0; vz_1=0    --
147 Fx=0;   Fy=0;   Fz=0      --
148 wa_1=0; wx_1=0; pa_1=0; py_1=0; ra_1=0; rz_1=0    --
149 Gx=0;   Gy=0;   Gz=0;     --
```

```
Xa=0;    Ya=0;    Za=0;    Vx=0;    Vy=0;    Vz=0       --
Wa=0;    Pa=0;    Ra=0;    Wx=0;    Py=0;    Rz=0       --
clock=8;num2=0   ;fs=0;
k=1;ko=1/70;  --kes dos
t=0;tf=5
FOR q=1 TO 3  DO
  FOR p=1 TO 3 DO
    Rh[q,p] =0; rt[q,p]=0;A7[q,p]=0
    END FOR
END FOR
'--Parámetros de impedancia.'
Md=0.01
Bd=0.05
Kd=0.1
'--Características del archivo de lectura-escritura (RW).'
SET_FILE_ATR(graf, ATR_IA)
'--Apertura del archivo RW.'
OPEN FILE graf ('RW', 'RK.kl')
'--Almacena el valor del vector inicial de fuerzas en la variable value0.'
GET_VAR(entry,'*SYSTEM*','$CCC_GRP[1].$FS_FORCE', value0, STATUS)
'--Escribe a archivo los valores de impedancia.'
WRITE graf ('Md=',Md,'Bd=',Bd,'Kd=',Kd,CR)
'--Asigna a la variable jpos el valor del arreglo de la posición actual del
    robot.'
jpos=CURPOS(0,0,1)
'--Arreglo que contiene la posición inicial del robot en coordenadas
    cartesianas.'
vn[1]=-13.134
vn[2]=-7.567
vn[3]=9.567
vn[4]=-21.091
vn[5]=58.203
vn[6]=-152.156
'--Convierte el arreglo a variable tipo jointpos.'
CNV_REL_JPOS(vn,jpos,STATUS)
'--Mueve el robot a la posición inicial.'
MOVE TO jpos
'--Guarda la posición anterior como posición inicial y posición deseada.'
xyzd=jpos
xyzi=xyzd
'--Arreglo que contiene la posición final a la que se moverá el robot.'
vn[1]=49.816
vn[2]=30.530
vn[3]=17.171
vn[4]=-29.418
vn[5]=80.507
vn[6]=-89.928
```

```
'--Convierte el arreglo a variable tipo jointpos.'
CNV_REL_JPOS(vn,jpos,STATUS)
'--Guarda la posición anterior como posición final.'
xyzf=jpos
'--Conecta el timer del sistema a la variabe clock.'
CONNECT TIMER TO clock
'--Etiquetas para saltos en el programa.'
fuerza::
aqui::
'--Reinicia los valores de las matrices que indican la rotación final de la
    herramienta.'
FOR q=1 TO 3  DO
  FOR p=1 TO 3 DO
    Rh[q,p] =0; rt[q,p]=0;A7[q,p]=0
    END FOR
END FOR
'--Matriz de rotación roll, pitch yaw hasta el plato.'
rt[1,1]=COS(xyzd.r)*COS(xyzd.p);  rt[1,2]=-(SIN(xyzd.r)*SIN(xyzd.w)+COS(xyzd.r)
    *SIN(xyzd.p)*COS(xyzd.w))
rt[1,3]= -SIN(xyzd.r)*COS(xyzd.w)+COS(xyzd.r)*SIN(xyzd.p)*SIN(xyzd.w);  rt
    [2,1]=SIN(xyzd.r)*COS(xyzd.p)
rt[2,2]=-(-COS(xyzd.r)*SIN(xyzd.w)+SIN(xyzd.r)*SIN(xyzd.p)*COS(xyzd.w))
rt[2,3]= COS(xyzd.r)*COS(xyzd.w)+SIN(xyzd.r)*SIN(xyzd.p)*SIN(xyzd.w);
rt[3,1]=-SIN(xyzd.p)
rt[3,2]= -(COS(xyzd.p)*COS(xyzd.w))
rt[3,3]=COS(xyzd.p)*SIN(xyzd.w)
'--Matriz de rotación del plato al final de la herramienta.'
A7[1,1]=1;  A7[1,2]=0;  A7[1,3]=0;
A7[2,1]=0;  A7[2,2]=0;  A7[2,3]=-1;
A7[3,1]=0;  A7[3,2]=1;  A7[3,3]=0;
'--Matriz de rotación total.'
FOR q=1 TO 3 DO
  FOR p=1 TO 3 DO
    FOR o=1 TO 3 DO
    Rh[q,p] = Rh[q,p] + rt[q,o]*A7[o,p]
    END FOR
  END FOR
END FOR
'--Obtiene el arreglo de fuerzas actual, y se asigna a la variable value1.'
GET_VAR(entry1, '*SYSTEM*','$CCC_GRP[1].$FS_FORCE', value1, STATUS)
'--Valor actual del timer se asigna a la variable num2.'
    num2=clock
'--Guarda en archivo el valor de la variable num2=tiempo de muestreo.'
WRITE graf (num2)
'--Limita el tiempo de muestreo mínimo y máximo para evitar errores en el
    sistema.'
      IF num2=0 THEN
```

```
238       num2=8
239     end if
240       IF num2>200 THEN
241       num2=200
242       end if
243'--Guarda en archivo el valor el tiempo actual.'
244WRITE graf (t)
245       t=t+(num2*.001)
246'--Reinicia el reloj para iniciar un nuevo ciclo de muestreo.'
247     clock=0
248'--Se calculan las fuerzas y torques.'
249      Fx1=(value0[1]-value1[1])
250      Fy1=(value0[2]-value1[2])
251      Fz1=(value0[3]-value1[3])
252      Gx1=(value0[4]-value1[4])*(1/10)
253      Gy1=(value0[5]-value1[5])*(1/10)
254      Gz1=(value0[6]-value1[6])*(1/10)
255'--Se asignan al eje correspondiente de acuerdo a la matriz de rotación total.'
256     Fx=Rh[1,1]*Fx1+ Rh[1,2]*Fy1+Rh[1,3]*Fz1
257     Fy=Rh[2,1]*Fx1+ Rh[2,2]*Fy1+Rh[2,3]*Fz1
258     Fz=Rh[3,1]*Fx1+ Rh[3,2]*Fy1+Rh[3,3]*Fz1
259     Gx=Rh[1,1]*Gx1+ Rh[1,2]*Gy1+Rh[1,3]*Gz1
260     Gy=Rh[2,1]*Gx1+ Rh[2,2]*Gy1+Rh[2,3]*Gz1
261     Gz=Rh[3,1]*Gx1+ Rh[3,2]*Gy1+Rh[3,3]*Gz1
262'--Se asigna un umbral para la detección de fuerzas.'
263    umbral=0.25
264    or_umbral=1
265'--Se verifica si existen o no fuerzas de interacción.'
266     IF ABS(Fx)>umbral THEN
267       var_x=TRUE
268     end if
269     IF ABS(Fy)>umbral THEN
270       var_y=TRUE
271     end if
272     IF ABS(Fz)>umbral THEN
273       var_z=TRUE
274     end if
275     IF ABS(Gx)>or_umbral THEN
276       var_w=TRUE
277     end if
278     IF ABS(Gy)>or_umbral THEN
279       var_p=TRUE
280     end if
281     IF ABS(Gz)>or_umbral THEN
282       var_r=TRUE
283     end if
284     IF var_x OR var_y OR var_z OR var_w OR var_p OR var_r THEN
```

```
285 WRITE TPERROR(CHR(128),CHR(137),'FUERZA!',Fx,Fy,Fz)
286 '--Se hace un llamado a la subrutina de Runge-Kutta.'
287 rungek
288 '--Se calcula la nueva posición de acuerdo al vector de ajuste.'
289  xn=xyzd.x-k*(Xa)
290  yn=xyzd.y-k*(Ya)
291  zn=xyzd.z-k*(Za)
292  wn=xyzd.w-ko*(Wa)
293  pn=xyzd.p-ko*(Pa)
294  rn=xyzd.r-ko*(Ra)
295 '--Se obtiene el valor absoluto del desplazamiento total.'
296  abxn=ABS(xyzd.x-xn)
297  abyn=ABS(xyzd.y-yn)
298  abzn=ABS(xyzd.z-zn)
299  abwn=ABS(xyzd.w-wn)
300  abpn=ABS(xyzd.p-pn)
301  abrn=ABS(xyzd.r-rn)
302 '--Se limita el desplazamiento.'
303 limitei
304 limitesup
305 '--Se asigna la nueva posición.'
306 jpos=xyzd
307 '--Tipo de movimiento sin desaceleración.'
308 $TERMTYPE=NODECEL
309 '--Comando que mueve al robot a la posición deseada.'
310 MOVE TO jpos NOWAIT
311 '--Se guarda en archivo las fuerzas, torques y posición deseada.'
312 WRITE graf (Fx)
313 WRITE graf (Fy)
314 WRITE graf (Fz)
315 WRITE graf (Gx)
316 WRITE graf (Gy)
317 WRITE graf (Gz)
318 WRITE graf (xyzd.x)
319 WRITE graf (xyzd.y)
320 WRITE graf (xyzd.z)
321 WRITE graf (xyzd.w)
322 WRITE graf (xyzd.p)
323 WRITE graf (xyzd.r,CR)
324 xa_1=Xa; vx_1=Vx; ya_1=Ya; vy_1=Vy; za_1=Za; vz_1=Vz;
325 wa_1=Wa; wx_1=Wx; pa_1=Pa; py_1=Py; ra_1=Ra; rz_1=Rz;
326 '--Se reinician los valores de fuerza en cero.'
327 var_x=FALSE;var_y=FALSE; var_z=FALSE
328 var_w=FALSE;var_p=FALSE; var_r=FALSE
329 '--Salto a la etiqueta fuerza:: en caso de existir interaccion.'
330 GOTO fuerza
331 else
```

```
'--Condición para detener la tarea.'
 IF t>tf THEN
'--Si se cumple la condición anterior, salto a etiqueta termina:: al final del programa.'
  GOTO termina
  end if
'--De otra manera, regresa a interpolación.'
interpol
'--Se propone un promedio de distancias cuando existe fuerza, para evitar contactos bruscos con la superficie.'
xyzg.x=(xyzg.x+xyzd.x)/2
xyzg.y=(xyzg.y+xyzd.y)/2
xyzg.z=(xyzg.z+xyzd.z)/2
xyzg.w=(xyzg.w+xyzd.w)/2
xyzg.p=(xyzg.p+xyzd.p)/2
xyzg.r=(xyzg.r+xyzd.r)/2
'--Se genera la posición deseada por interpolacion.'
xyzd=xyzg
'--Escribe al teach pendant la leyenda "no hay fuerza."'
WRITE TPERROR(CHR(128),CHR(137),'NO HAY FUERZA!',Fx,Fy,Fz)
'--Cinemática inversa: convierte coordenadas a juntas.'
jpos=xyzd
'--Mueve al robot a la posición deseada.'
MOVE TO jpos NOWAIT
'--Guarda en archivo los datos cuando no existe fuerza de interacción presente.'
WRITE graf (0)
WRITE graf (0)
WRITE graf (0)
WRITE graf (0)
WRITE graf (0)
WRITE graf (0)
WRITE graf (xyzd.x)
WRITE graf (xyzd.y)
WRITE graf (xyzd.z)
WRITE graf (xyzd.w)
WRITE graf (xyzd.p)
WRITE graf (xyzd.r,CR)
'--Salto a la  etiqueta aqui:: para continuar sensando fuerza.'
GOTO aqui
end if
'--Escribe a teach pendant la leyenda "termina programa".'
WRITE TPERROR(CHR(128),CHR(137),'termina programa.')
'--Se cierra archivo (RW).'
CLOSE FILE graf
termina::
'--Fin del programa.'
```

```
376 END TrkTD
```

Bibliografía

[1] R. Anderson and M. Spong. Hybrid impedance control of robotic manipulators. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 1073 – 1080, mar 1987.

[2] K.J. Åström and B. Wittenmark. *Computer-controlled systems: theory and design*. Prentice-Hall information and system sciences series. Prentice Hall, 1997.

[3] I. Bonilla, E.J. González-Galván, C. Chavez-Olivares, M. Mendoza, A. Loredo-Flores, F. Reyes, and Biao-Zhang. A vision-based, impedance control strategy for industrial robot manipulators. In *Automation Science and Engineering (CASE), 2010 IEEE Conference on*, pages 216 –221, aug. 2010.

[4] Isela Bonilla-Gutiérrez. *Control de interacción de robots manipuladores en tareas industriales y de rehabilitación*. PhD thesis, Universidad Autónoma de San Luis Potosí, Ago 2011.

[5] S.C. Chapra and R.P. Canale. *Métodos numéricos para ingenieros*. McGraw-Hill, 2007.

[6] A. Duschau-Wicke, J. von Zitzewitz, A. Caprez, L. Lunenburger, and R. Riener. Path control: A method for patient-cooperative robot-aided gait rehabilitation. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 18(1):38 –48, feb. 2010.

[7] J.J. Gonzalez and G.R. Widmann. A force commanded impedance control scheme for robots with hard nonlinearities. *Control Systems Technology, IEEE Transactions on*, 3(4):398 –408, dec 1995.

[8] N. Hogan. Impedance control - An approach to manipulation. I - Theory. II - Implementation. III - Applications. *ASME Transactions Journal of Dynamic Systems and Measurement Control B*, 107:1–24, mar 1985.

[9] N Hogan and S P Buerger. Impedance and interaction control 19.1. *Regulation*, pages 1–22, 2005.

[10] H. Kazerooni. Robust, non-linear impedance control for robot manipulators. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 741 – 750, mar 1987.

[11] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *Robotics and Automation, IEEE Journal of*, 3(1):43 –53, february 1987.

[12] H.I. Krebs, B.T. Volpe, D. Williams, J. Celestino, S.K. Charles, D. Lynch, and N. Hogan. Robot-aided neurorehabilitation: A robot for wrist rehabilitation. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 15(3):327 –335, sept. 2007.

[13] Gert Kwakkel, Boudewijn J Kollen, and Hermano I Krebs. Effects of robot-assisted therapy on upper limb recovery after stroke: a systematic review. *Neurorehabilitation and Neural Repair*, 22(2):111–121, 2008.

[14] D.A. Lawrence. Impedance control stability properties in common implementations. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, volume 2, pages 1185 –1190, apr 1988.

[15] Marco-Octavio Mendoza-Gutiérrez. *Control de un sistema de teleoperación para su aplicación en terapias robóticas de rehabilitación.* PhD thesis, Universidad Autónoma de San Luis Potosí, Ago 2011.

[16] Gerdienke B Prange, Michiel J A Jannink, Catharina G M Groothuis-Oudshoorn, Hermie J Hermens, and Maarten J IJzerman. Systematic review of the effect of robot-aided therapy on recovery of the hemiparetic arm after stroke. *The Journal of Rehabilitation Research and Development*, 43(2):171, 2006.

[17] A. Roy, H.I. Krebs, D.J. Williams, C.T. Bever, L.W. Forrester, R.M. Macko, and N. Hogan. Robot-aided neurorehabilitation: A novel robot for ankle rehabilitation. *Robotics, IEEE Transactions on*, 25(3):569 –582, june 2009.

[18] Rolf Dieter Schraft, Christian Meyer, Christopher Parlitz, and Evert Helms. Powermate - a safe and intuitive robot assistant for handling and assembly tasks. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005, April 18-22, 2005, Barcelona, Spain*, pages 4074–4079. IEEE, 2005.

[19] B. Siciliano and O. Khatib. *Springer handbook of robotics.* Springer, 2008.

[20] S.K. Singh and D.O. Popa. An analysis of some fundamental problems in adaptive control of force and impedance behavior: theory and experiments. *Robotics and Automation, IEEE Transactions on*, 11(6):912 –921, dec 1995.

[21] M.W. Spong and M. Vidyasagar. *Robot Dynamics And Control*. Wiley India Pvt. Ltd., 2008.

[22] E.T. Wolbrecht, V. Chan, D.J. Reinkensmeyer, and J.E. Bobrow. Optimizing compliant, model-based robotic assistance to promote neurorehabilitation. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 16(3):286 –297, june 2008.