UNIVERSIDAD POLITÉCNICA DE PUEBLA

MAESTRÍA EN INGENIERÍA EN AUTOMATIZACIÓN DE PROCESOS INDUSTRIALES

TÍTULO DE LA TESIS

DISEÑO DE UN CONTROL DE POSICIÓN PD CON SINTONIZACIÓN AUTOMÁTICA PARA UN MANIPULADOR DE 2GDL USANDO REDES NEURONALES DE BASE RADIAL

Presenta:

Ing. Donaciano Coyotecatl Cuautle

Asesores:

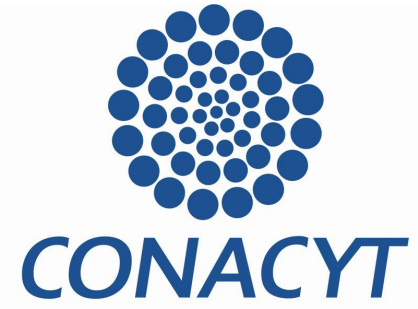
Dr. Carlos Muñiz Montero

M.C. Juan Antonio Arízaga Silva

Juan C. Bonilla, Puebla. 30 de Noviembre de 2015.

El presente trabajo fue realizado en el Laboratorio de Investigación y Posgrado de la Universidad Politécnica de Puebla, ubicada en Tercer carril del Ejido "Serrano" S/N, San Mateo Cuanalá, Municipio Juan C. Bonilla, Puebla CP 72640.

Apoyo del CONACYT, Beca No. 366051, Programa de Maestría perteneciente al Programa Nacional de Posgrados de Calidad (PNPC-CONACYT). Apoyo del CONACYT a través del Proyecto Ciencia Básica 181201.



Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En este capítulo se plantea la necesidad de diseñar un control de posición tipo PD con sintonización automática de las ganancias proporcional y derivativa, en función de la posición deseada para mejorar el desempeño del control de posición PD propuesto por Takegaki y Arímoto en 1981. La principal ventaja del controlador propuesto consiste en la posibilidad de adaptar las ganancias cuando ocurran cambios abruptos en el seguimiento de trayectorias, además de no requerir del conocimiento de los parámetros que definen el modelo dinámico del manipulador.

1.1. Robots manipuladores

La robótica se ocupa del estudio de un tipo de máquinas que pueden sustituir a los seres humanos en la ejecución de una tarea específica, en lo que respecta tanto actividades físicas como en la toma de decisiones (Siciliano et al., 2009). Con la creación de autómatas enfocados a la manipulación de objetos surgen los llamados robots manipuladores, los cuales han sido diseñados basándose en la estructura física de un brazo humano, a los que también se les llama robots antropomórficos debido a esta semejanza (W.Spong et al., 2004).

La robótica industrial fue desarrollada a principios del año 1960, influenciada por dos tipos de tecnologías: las máquinas de control numérico por un lado, utilizadas para tener mayor precisión en los procesos de manufactura y teleoperación por el otro, para el manejo remoto de materiales radioactivos. Comparado con sus precursores, el primer manipulador de acuerdo a (Siciliano et al., 2009) contemplaba las siguientes ventajas:

1.- Versatilidad para emplear diferentes tipos de efectores finales.
2.- Uso de sensores para obtener adaptación previa a situaciones desconocidas.
3.- Exactitud de posicionamiento utilizando técnicas de control por realimentación.
4.- Repetibilidad en la ejecución utilizando programación.

Un robot manipulador se considera como una serie de eslabones rígidos interconectados por medio de articulaciones (servomotores) rotacionales o prismáticas en forma de cadena cinemática abierta, formando un brazo y una mano a la cual comúnmente se le

llama efector final. Es utilizado para colocar objetos y herramientas, logrando realizar una amplia gama de operaciones físicas dentro de su espacio de trabajo (W.Spong et al., 2004).

De acuerdo con la definición adoptada por la Federación Internacional de Robótica bajo la norma $ISO/TR8373$, un robot manipulador se define de la siguiente manera: Un robot manipulador industrial es una máquina manipuladora con varios grados de libertad controlada automáticamente, reprogramable y de múltiples usos, pudiendo estar en un lugar fijo o móvil para su empleo en aplicaciones industriales (Kelly and Santibañez, 2003).

La capacidad que tiene un robot manipulador de realizar diversos trabajos mediante una herramienta adjunta a su efector final permite distinguir dos formas en las que puede interactuar con su entorno: (i) si la interacción es despreciable, conviene implementar un control de posición; (ii) Si la interacción es significativa, como es el caso de la aplicación de fuerzas sobre una superficie, se deben plantear esquemas de control más sofisticados, tal como el referido a un control de fuerza (Patarinski and Botev, 1992).

Para que un robot manipulador pueda realizar una tarea de forma adecuada y precisa se requiere de un control coordinado para cada articulación, esto con la finalidad de producir el movimiento deseado en el efector final. Cabe mencionar que para implementar algún tipo de control en el robot se necesita conocer de manera precisa su modelo dinámico.

El modelo dinámico intenta describir la mayor parte de los fenómenos físicos presentes en el robot manipulador y se obtiene aplicando las ecuaciones de movimiento de Euler-Lagrange. Para un manipulador con n grados de libertad esta se expresa como (Kelly and Santibañez, 2003, L.Lewis et al., 2004):

$$\tau = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(\dot{q}) \tag{1.1}$$

Siendo:
$M(q) \in \mathbb{R}^{n\times n}$: Matriz de inercias.
$C(q,\dot{q}) \in \mathbb{R}^{n\times n}$: Matriz de fuerzas centrípetas y de Coriolis.
$g(q) \in \mathbb{R}^{n\times 1}$: Vector de par gravitacional.
$f(\dot{q}) \in \mathbb{R}^{n\times n}$: Matriz fricción viscosa y de Coulomb.

La dinámica de un robot manipulador proporciona una descripción completa entre los pares τ aplicados a los servomotores y el movimiento de la estructura mecánica. La ecuación descrita en (1.1) considera a los eslabones del robot como rígidos, es decir, no incluyen el fenómeno de elasticidad.

1.2. Control de movimiento

Control de movimiento sin restricciones

El control de movimiento sin restricciones o simplemente control de movimiento se realiza cuando los robots manipuladores se desplazan libremente dentro de su espacio

Figura 1.1: Control de posición: pintado de piezas mecánicas

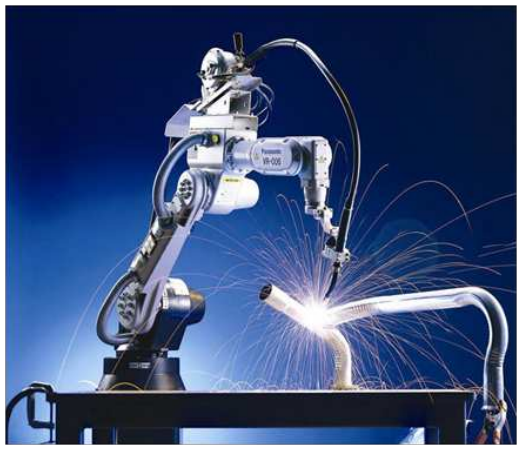


Figura 1.2: Interacción del robot con su entorno de trabajo

de trabajo sin tener interacción con el ambiente que los rodea. Para tal situación el efector final no es obstruido por ningún objeto y la implementación de un control de posición o trayectoria es adecuado para el buen desempeño de la tarea (L.Lewis et al., 2004). La Figura 1.1 muestra un ejemplo de este tipo de control donde el efector final del manipulador es utilizado para pintar piezas mecánicas.

Control de movimiento restringido

El control de movimiento restringido se utiliza cuando el efector final del manipulador presenta interacción o contacto con el ambiente que lo rodea. En este caso, el efector final es obstruido por algún objeto, su movimiento está restringido, por lo cual se generan fuerzas de contacto que no pueden ser despreciadas. En otras palabras, si se consideran tanto al robot manipulador como al objeto constituidos de materiales rígidos, entonces movimientos pequeños provocarán grandes fuerzas de interacción (Raibert and Craig, 1981, Patarinski and Botev, 1992). Algunos ejemplos de tareas que el robot manipulador puede realizar cuando su efector final presenta interacción con el ambiente son los siguientes: acabado y pulido de superficies, desbastado de materiales, perforación, doblado de materiales, ensamble de alta precisión, etc., las cuales son tareas que precisan de una aplicación de fuerza significativa en su ejecución. En la Figura 1.2 se muestra un robot manipulador realizando un proceso de soldadura en la fabricación de piezas.

1.3. Estado del arte en el control de movimiento

El control de movimiento en un robot manipulador corresponde no sólo al posicionamiento de piezas dentro de su espacio de trabajo (control de posición) sino también al seguimiento de trayectorias deseadas, el cual comúnmente se resuelve con esquemas de control de trayectoria. Con la finalidad de realizar un estudio bajo diferentes enfoques de los esquemas de control implementados para resolver el problema de seguimiento, a continuación se enumeran en orden cronológico algunos trabajos relevantes.

1. Es efectiva la ley de control lineal y local PD con realimentación para el seguimiento de trayectorias en el movimiento de un robot?. Autores: S. Kawamura, F. Miyazaki y S. Arimoto (1988)
2. Una clase de controladores no lineales tipo PD para robots manipuladores. Autores: R. Kelly y R. Carelli (1996)
3. Control PD modificado de robots manipuladores usando un compensador de red neuronal. Autores: J. A. Heredia Huerta y W. Yu (1999)
4. Control PD con compensación hacia adelante para robots manipuladores: análisis y experimentación. Autores: V. Santibánez y R. Kelly (2001)
5. Control lineal y no lineal tipo PD de robots manipuladores para seguimiento de trayectorias. Autores: L. Fan y E. Meng Joo (2009)
6. Control de trayectoria PD de robots manipuladores con ganancias variables: análisis de estabilidad y simulaciones. Autores: F. G. Salas, V. Santibáñez y M. A. Llama (2012)
7. Control de red neuronal para compensar la fricción basado en energía en el movimiento de subida de un Pendubot . Autores: D. Xia, L. Wang y T. Chai (2014)

En el trabajo de (Kawamura et al., 1988) se prueba que el control de posición PD puede ser implementado en el seguimiento de una trayectoria deseada, la cual es descrita como una función del tiempo. Sin embargo, para que el seguimiento se realize de manera adecuada se requiere que la velocidad de realimentación de las ganancias sea suficientemente rápida. La demostración de estabilidad indica que el sistema es asintóticamente estable pero sólo de manera local.

En (Kelly and Carelli, 1996) se presenta una clase de controladores tipo PD para posición y control de movimiento en donde de acuerdo al análisis de estabilidad por el método directo de Lyapunov se demuestra estabilidad asintótica global. La importancia de este trabajo radica en que las ganancias proporcional y derivativa pueden ser funciones no lineales dependientes de los estados del robot, con lo cual dejan de ser constantes. También introducen la estructura de matriz diagonal para las ganancias proporcional y derivativa y el regulador propuesto esta restringido a entregar torques dentro de límites preestablecidos. Una de las desventajas de esta clase de controladores de movimiento es el uso de un control PD+, el cual requiere de la realimentación de la dinámica del robot, y por lo tanto, es un control basado en el modelo dinámico.

En (Huerta and Yu, 1999) se propone un control tipo PD modificado, el cual es utilizado para compensar términos no lineales de la dinámica de un robot planar con dos grados de libertad usando redes neuronales RBF. Las ventajas que se tiene con este enfoque es que no se requiere entrenamiento fuera de línea de las redes y no esta basado en el modelo dinámico, sin embargo, para que exista convergencia de la red neuronal (compensación correcta de los parámetros) se requiere de condiciones iniciales ideales.

La demostración de estabilidad asintótica global para el control PD con realimentación hacia adelante es realizada en (Santibañez and Kelly, 2001), ya que antes de este trabajo nadie lo había considerado, este tipo de estabilidad se garantiza al prevenir adecuadamente la selección de las ganancias k_p y k_v, sin embargo la naturaleza propia de este control requiere para su implementación de los parámetros nominales del robot (control basado en modelo).

Como una continuación del trabajo realizado en (Kelly and Carelli, 1996), en (Fan and Joo, 2009) se plantea un control modificado tipo PD con ganancias tanto lineales como no lineales y se demuestra estabilidad asintótica global para el problema de control de trayectoria, sin embargo solamente en el caso no lineal se consideran ganancias variables. Cabe mencionar que las ganancias en ambos casos dependen de los estados del sistema (control basado en modelo).

En el trabajo presentado en (Salas et al., 2012) se aborda la solución al problema de control de trayectoria utilizando un esquema PD con ganancias k_p y k_v variables, las cuales son dependientes del error de seguimiento. Esta característica mantiene la ventaja de simplicidad del control PD ya que sólo contempla dos ganancias, además de no necesitar compensación de la dinámica del manipulador. La sintonización de las ganancias proporcional y derivativa se realiza mediante un método de control difuso, estas ganancias deben ser además funciones acotadas del error de trayectoria, lo que complica su selección.

En años recientes se ha optado por la implementación de los así llamados controles inteligentes (Salas et al., 2012). Por ejemplo en (Xia et al., 2014) se propone un controlador basado en el estudio de la energía en el movimiento de subida de un péndulo con la finalidad de alcanzar su punto de equilibrio más inestable. En este esquema se utilizan redes neuronales del tipo RBF (Radial Basis Function) para compensar el fenómeno de fricción. Además, no requiere el conocimiento del modelo dinámico para su implementación. Sin embargo el algoritmo de control resultante es complicado y no se demuestra estabilidad del sistema.

1.4. Planteamiento del Problema

Después de realizar la revisión del estado del arte se elaboró la Tabla 1.1 en donde se presentan los factores más relevantes para la implementación del control de movimiento de robots manipuladores.

En primer lugar se observó que para resolver el problema de seguimiento de trayectorias no se implementa el control de posición, sino que en su lugar se utiliza el control de trayectoria. Sin embargo este tipo de esquemas tiene la desventaja de requerir explíci-

Referencia	**Tipo de control**	**Depende del modelo**	**Utiliza RBFNN**	**Demostración de estabilidad**	**Ganancias variables**
1	Trayectoria	no	no	si (local)	no
2	Pos/Trayec	si	no	si (global)	si
3	Trayectoria	no	si	si (local)	no
4	Trayectoria	si	no	si (global)	no
5	Trayectoria	si	no	si (global)	si
6	Trayectoria	no	no (fuzzy)	si (local)	si
7	Basado en energía	no	si	no	no

Tabla 1.1: Principales características en el estado del arte

tamente de los valores para los parámetros del modelo dinámico con lo cual se tiene un problema de implementación ya que no son fáciles de calcular. Cabe mencionar que existen métodos recursivos de *identificación paramétrica* como el presentado en (Reyes, 2012) que permiten aproximar los valores del modelo dinámico, pero requieren de un tiempo de procesamiento considerable.

Otra característica a considerar es que los controladores del tipo PD existentes para el seguimiento de trayectorias no contemplan sintonización automática de sus ganancias y que solamente en (Kelly and Carelli, 1996), (Fan and Joo, 2009) y (Salas et al., 2012) se proponen a las ganancias como variables ya sea en función de los estados del sistema o del error de seguimiento.

Cabe señalar que las redes neuronales de base radial (RBF) fueron utilizadas en los trabajos de (Huerta and Yu, 1999) y (Xia et al., 2014) pero sólo para compensar fenómenos no lineales presentes en la dinámica del manipulador tal como lo es la fricción y el par gravitacional. Además en (Salas et al., 2012) se utiliza un método difuso para sintonizar las ganancias del controlador con lo cual se da pauta para proponer un proceso de sintonización automática utilizando diferentes técnicas.

De acuerdo a lo anterior en el presente trabajo se plantea el diseño de un controlador de posición del tipo PD que permita realizar el seguimiento de trayectorias punto a punto sin la necesidad del conocimiento completo de su modelo dinámico, además de obtener la sintonización automática de sus ganancias proporcionales y derivativas las cuales deben ser variables usando para ello redes neuronales de base radial.

1.5. Objetivos

Objetivo general

Proponer una ley de control con sintonización automática basada en redes neuronales de base radial (RBFNN) para el seguimiento de trayectorias y construir un prototipo de robot manipulador con dos grados de libertad que incorpore control de posición.

Objetivos específicos

1. Proponer una ley de control de posición tipo PD con sintonización automática de las ganancias en función de la posición deseada a partir de redes neuronales de base radial.

2. Implementar control de posición tipo PD en un prototipo de robot manipulador con 2 gdl y arquitectura abierta.

Metas para el objetivo específico 2

a).- Diseño y validación por simulación en Matlab de un regulador de posición tipo PD auto-sintonizable a partir de redes neuronales de base radial.
b).- Comparar el desempeño de la ley de control propuesta con otras leyes de control de trayectoria con ganancias variables o constantes.

Metas para el objetivo específico 1

a).- Construcción del prototipo.
b).- Implementación del control de posición PD en los actuadores.

1.6. Justificación

Como ya se ha mencionado anteriormente, para resolver el problema de seguimiento de trayectorias se implementan esquemas de control de trayectoria que traen como desventaja el requerir del conocimiento del modelo dinámico.

Si se parte de la facilidad de implementación que se tiene con el control de posición PD y de su aplicación inmediata al denominado control punto a punto, el cual consiste en mover el efector final del manipulador a un conjunto de posiciones deseadas qd_i (Reyes, 2011). De esta forma es posible seguir trayectorias tales como círculos, elipses y figuras parametrizadas en función de un conjunto de puntos sin la necesidad de implementar un control de trayectoria.

Si se tiene auto-sintonización de las ganancias en un algoritmo de control entonces para cada posición deseada se tendrá un valor diferente de estas ganancias, logrando con ello la disminución del error en estado estable y del sobretiro, además de mejorar el desempeño en el seguimiento de la trayectoria deseada, incluso si existen cambios abruptos de dirección.

Con la finalidad de permitir la experimentación de diversos algoritmos de control ya sea de posición o trayectoria, se pretende proporcionar un prototipo didáctico con arquitectura abierta, de manera que pueda ser programado por el usuario para alcanzar posiciones deseadas dentro de su espacio de trabajo. Con la realización de este proyecto se pretende poner en práctica los conceptos teóricos de cinemática, modelado dinámico y control.

1.7. Metodología de trabajo

La metodología utilizada para la realización de esta tesis se basa de acuerdo a las siguientes etapas:

1.- Modelado y simulación del sistema: En esta etapa partimos de un modelo cinemático y dinámico parametrizado de un robot manipulador propuesto en (Reyes, 2011, Reyes and Kelly, 2001). El objetivo principal es observar la respuesta dinámica de algoritmos de control propuestos en la literatura tanto de posición como de trayectoria y poder compararlos tomando en cuenta para su desempeño parámetros tales como tiempo de estabilización, sobretiro y error en estado estable.

2.- Proponer algoritmo de control de posición tipo PD auto-sintonizable: La propuesta de la ley neuronal PD se realiza bajo el diseño de redes neuronales del tipo de base radial (RBF), debido a que tienen características de localidad consiguiendo con ello que se puedan especializar en una región determinada del espacio de trabajo del robot. Para mejorar el desempeño de control de posición PD propuesto por (Arimoto and Takegaki, 1981) en el seguimiento de trayectorias, el nuevo controlador considera a las ganancias proporcionales y derivativas como variables, las cuales deben ser funciones de la posición deseada. Este proceso se llevará a cabo realizando un entrenamiento manual fuera de línea de la red neuronal generando como consecuencia una red de interpolación.

3.- Construcción del robot manipulador: Para la construcción del robot fue necesario contar con un diseño de acceso libre, el cual fue desarrollado en el software de diseño SolidWorks. Las ventajas del diseño de acceso libre fue la obtención de los planos de cada una de las piezas que forman el prototipo, lo cual fue de gran ayuda para la rápida fabricación del mismo.

4.- Instrumentación del prototipo: En esta etapa se seleccionaron las tarjetas de control, potencia y actuadores. La tarjeta de control se encargará de procesar las mediciones del encoder acoplado a los motorreductores para calcular su posición y velocidad necesarias para la implementación del control PD, así mismo debe enviar las señales de control a la tarjeta de potencia como una señal PWM. La tarjeta de potencia recibe la señal de control PWM y lo interpreta como un valor de voltaje el cual será aplicado a los actuadores como un valor de torque. Los actuadores se seleccionaron de acuerdo al máximo torque ofrecido a rotor bloqueado.

5.- Pruebas con el prototipo: En esta etapa se realizaron diversas pruebas de posicionamiento para el control PD implementado en el prototipo, además de la programación del control neuronal propuesto en la tarjeta de desarrollo Freescale y la obtención de resultados experimentales.

1.8. Contribuciones

- Estructura mecánica del prototipo y de arquitectura abierta para la implementación de controladores.

- Resultados de simulación en MatLab para el control de posición PD de un sistema parametrizado.
- Resultados de simulación de la propuesta de control tipo PD auto-sintonizable usando RBFNN.
- Resultados experimentales de la implementación del control de posición PD en los actuadores del prototipo.

1.9. Estructura de la tesis

En el capítulo 1 se describe el planteamiento del problema comenzando con una breve introducción a los robots manipuladores y los tipos de control de acuerdo a las restricciones en el efector final. El estado del arte realizado sugiere la necesidad de proponer una ley de control con sintonización automática y de la construcción de un prototipo de arquitectura abierta para la implementación de algoritmos de control. Además se tratan los aspectos de la justificación del tema de tesis, la metodología seguida y las contribuciones esperadas.

En el capítulo 2 se presenta el marco teórico como fundamento en el desarrollo de la tesis. Los temas tratados hacen referencia a la historia, definición y clasificación de la robótica, modelado cinemático y dinámico para un robot manipulador con dos grados de libertad y una sección dedicada a las generalidades de las redes neuronales del tipo base radial (RBFNN's), resaltando su proceso de diseño y entrenamiento.

En el capítulo 3 se propone una ley neuronal del tipo PD denominada PD-N y de una metodología de diseño. El controlador propuesto permite sintonización automática de sus ganancias proporcional y derivativa usando el entrenamiento fuera de línea de las redes neuronales RBF para conseguir tal objetivo. Además se presenta la demostración de estabilidad de la ley de control neuronal propuesta aplicando el método directo de Lyapunov y se presentan los resultados de simulación tanto para resolver el problema de posición como de seguimiento de trayectorias.

En al capítulo 4 se desarrolla el proceso de fabricación de un prototipo robótico con dos grados de libertad considerado de arquitectura abierta debido a la posibilidad de variar los parámetros del controlador, para tal objetivo se implemento un control de posición tipo PD en los actuadores. Se realizan diversas pruebas y se presentan los resultados de experimentación.

Finalmente el capítulo 5 esta dedicado a las conclusiones y discusión de resultados obtenidos, así como de las perspectivas y trabajos futuros que pueden desarrollarse a partir de este trabajo de tesis.

Capítulo 2

Marco Teórico

En este capítulo se presentan los fundamentos teóricos necesarios para sustentar esta investigación. En la sección 2.1 se presenta una breve introducción a la robótica, historia y los tipos de robots utilizados en la industria. Posteriormente en las secciones 2.2 y 2.3 se aborda el modelado cinemático y dinámico de robots manipuladores y se presentan algunos ejemplos. Finalmente, en la sección 2.4 se discute el tema de las redes neuronales de base radial como aproximadores de funciones no lineales.

2.1. Introducción a la robótica

A lo largo de los años, la humanidad ha intentado reproducir algunas de sus características, tales como el poder moverse o interactuar con el medio que lo rodea a través de objetos inanimados. Para conseguir tal objetivo, fue necesario implementar estructuras mecánicas constituidas por una serie de mecanismos simples como ruedas, poleas e incluso engranes. Sin embargo, los avances en la electrónica y los circuitos integrados han favorecido la creación de sistemas mecánicos cada vez más complejos, los cuales no sólo pueden reproducir movimientos parecidos a los de los humanos sino que además han logrado sustituirle en procesos inteligentes como lo es la toma de decisiones.

2.1.1. Historia, definición y clasificación

Historia de la robótica

En el año de 1920, el escritor checo Karel Capek escribe una obra teatral titulada *Rossum's Universal Robots* (R.U.R), en la cual se narra la historia de creaturas hechas de material orgánico, esclavizadas y destinadas únicamente a la realización de trabajos no deseados por los humanos. En esta obra se emplea por primera vez la palabra "robot", que en el idioma nativo significa "fuerza de trabajo" o "servidumbre". Posteriormente, otro dramaturgo de origen ruso y dedicado a la escritura de ciencia ficción, llamado Isaac Asimov, concibe al robot como un artefacto mecánico de apariencia humana pero con ausencia de sentimientos (Jazar, 2010).

Definición de robótica

En la literatura existen muchas definiciones correspondiente a esta área del conocimiento. A continuación se presentan tres propuestas seleccionadas. Consultando en (Jazar, 2010), se propone lo siguiente. "La Robótica se ocupa del estudio de un tipo de máquinas que pueden reemplazar seres humanos en la ejecución de una tarea específica en lo que respecta tanto actividades físicas como en la toma de decisiones".

Una definición interesante se puede encontrar en (Siciliano et al., 2009), donde se define a la robótica como "la ciencia que estudia las conexiones inteligentes entre percepción a través de sensores y acción por medio de actuadores".

En (Reyes, 2011), la robótica es considerada como un sistema complejo, representado funcionalmente por subsistemas múltiples, con lo cual define lo siguiente "La naturaleza multidisciplinaria de la robótica permite involucrar una gran cantidad de áreas del conocimiento tales como matemáticas, física, mecánica, electrónica, eléctrica, ciencias de la computación e inteligencia artificial".

De acuerdo a lo anterior se puede concluir que la Robótica es un tipo de sistema multidisciplinario en donde la realimentación se realiza a través de sensores donde sus actuadores permiten realizar acciones o movimientos y además puede reemplazar a los seres humanos para realizar actividades peligrosas o repetitivas.

Clasificación de la Robótica

Para su estudio, la robótica se puede clasificar en dos categorías (Siciliano et al., 2009, Ollero, 2007):

i) Tipo de base que presenta el robot.

En esta categoría se contempla que el robot puede ser de base fija o móvil. Para el primer caso, la base se encuentra anclada a uno de sus extremos o a una parte fija del espacio de trabajo, que generalmente es donde se intenta depositar el mayor peso posible del robot.

En el segundo caso, la base del robot puede contener ruedas o cuerpos rígidos implementadas como algún tipo de piernas, con la finalidad de darle movimiento a la base y que pueda moverse libremente en su ambiente de operación. El movimiento libre implica además que el robot móvil debe contemplar cierta autonomía en dicho movimiento, de manera que es indispensable generar trayectorias y que el robot puede decidir qué opción es la mejor.

ii) Tipo de aplicación que tiene el robot.

Esta categoría se lleva a cabo de acuerdo al área de aplicación en particular para la cual el robot fue diseñado, por ejemplo, para cada especialización se puede contemplar a la robótica como: médica, doméstica, asistencial, industrial, etc.

En la Figura 2.1 se muestra la clasificación de la robótica, de acuerdo a su área de aplicación y la referente a su estructura mecánica, la cual puede ser de base fija o móvil.

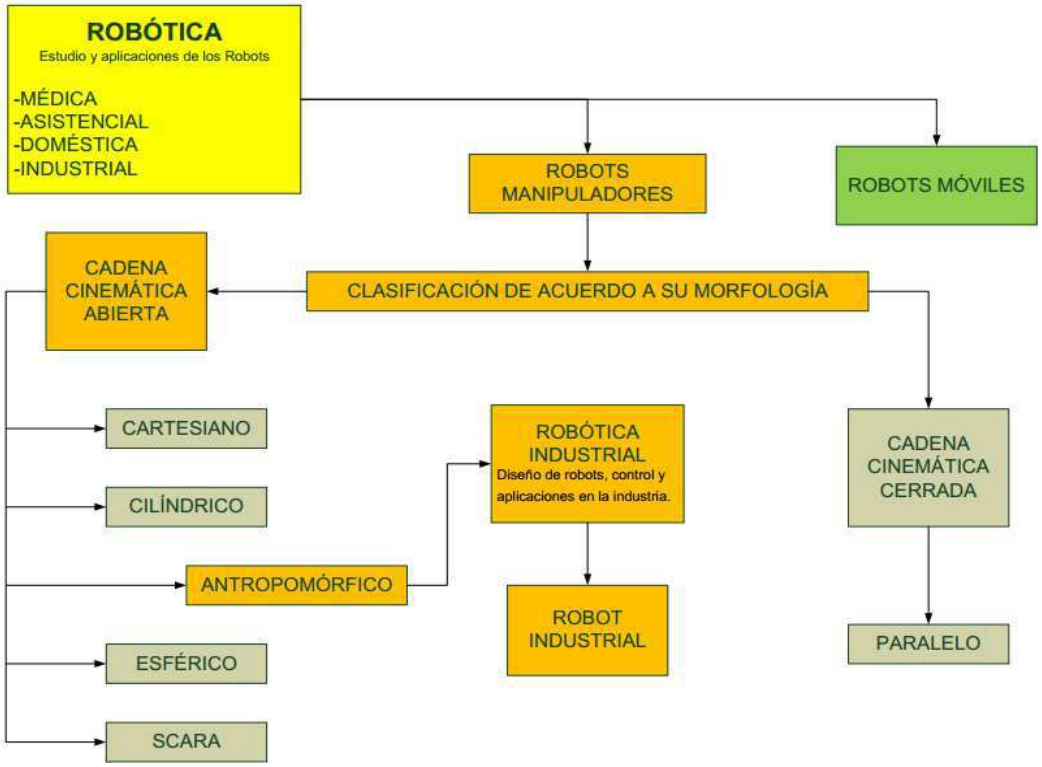


Figura 2.1: Clasificación de acuerdo al tipo de base y morfología

Los robots manipuladores pueden presentar una cadena cinemática cerrada o abierta, dependiendo del último eslabón si está o no unido a la base. La morfología representa el espacio de trabajo en donde el manipulador puede moverse, lo cual implica sus limites máximos y mínimos de alcance para realizar una tarea.

En este trabajo sólo serán tratados temas relacionados con robots de base fija que son llamados comúnmente robots manipuladores, además de la implementación de un control de posición el cual permita al usuario posicionar el efector final del manipulador en puntos deseados dentro de su rango de operación.

2.1.2. Robots manipuladores

La estructura mecánica de un robot manipulador consiste de la secuencia de eslabones rígidos interconectados por articulaciones. Un manipulador se caracteriza por tener un brazo que asegura movilidad, una muñeca que brinda destreza y un efector final que permite ejecutar la tarea requerida impuesta al robot (Siciliano et al., 2009).

La estructura fundamental de un manipulador se encuentra clasificada por ser de cadena cinemática abierta o cerrada. Para que un robot manipulador pueda ejecutar una tarea específica, se requiere en primera instancia posicionar una articulación correspondiente a la muñeca, por lo cual a través de su movimiento, el efector final podrá orientarse y posicionarse de manera adecuada.

Tipos de robots manipuladores

Para determinar el tipo de robot manipulador, es necesario considerar la secuencia de los grados de libertad que presenta. Los grados de libertad referidos a la movibilidad que debe tener el sistema, pueden clasificarse de acuerdo al tipo de articulación utilizada, cuando se tienen articulaciones rotacionales, el movimiento producido es giratorio, en el caso de contemplar articulaciones prismáticas, el movimiento efectuado será un desplazamiento lineal. Si se consideran a los grados de libertad comenzando desde una articulación base, esto permite obtener una clasificación de los robots manipuladores como cartesianos, cilíndricos, esféricos , SCARA y antropomórficos (Siciliano et al., 2009, Kelly and Santibañez, 2003, Ollero, 2007, Barrientos et al., 1996).

a) Cartesiano.- La geometría de un robot manipulador tipo cartesiano está formada por tres articulaciones lineales o también denominadas prismáticas (PPP), cuyos ejes son mutuamente ortogonales. El número de articulaciones determinan los grados de libertad del sistema los cuales corresponden a variables dentro de un espacio cartesiano (x, y, z).

La estructura cartesiana ofrece bastante rigidez mecánica debido a que las articulaciones son independientes, esto es, ninguna articulación carga el peso de las otras articulaciones. Debido a esta rigidez, la exactitud en el posicionamiento de la muñeca en donde se encuentra adjunto el efector es constante en cualquier lugar dentro del espacio de trabajo. Los manipuladores cartesianos son empleados frecuentemente para manipulación de materiales y ensamblado de piezas.

b) SCARA.- El manipulador tipo SCARA (*Selective Compliance Assembly Robot Arm*) tiene una geometría especial compuesta por dos articulaciones rotacionales para la base y hombro, además de una articulación prismática para el efector final localizadas de tal forma que todos los ejes de movimiento son paralelos.

La estructura mecánica es de alta rigidez para cargas en forma perpendicular al plano formado por las dos articulaciones rotacionales y flexibilidad para cargas paralelas a dicho plano. La configuración SCARA es adecuada para tareas de ensamble con objetos pequeños.

c) Antropomórfico.- La geometría antropomórfica es caracterizada por tener 3 articulaciones rotacionales, en donde el eje rotacional de la primera es ortogonal a los ejes de las otras dos articulaciones las cuales son paralelas. La estructura antropomórfica es la más popular debido a que todas las articulaciones son rotacionales y permite mayor número de aplicaciones.

d) Manipulador antropomórfico con geometría paralela.- Este tipo de manipuladores son considerados de cadena cinemática abierta, sin embargo, cuando se requiere mayor capacidad de carga, la estructura mecánica tendrá mayor rigidez para garantizar precisión de posicionamiento. En tal caso, es recomendable recurrir a una cadena cinemática parcialmente cerrada. Por ejemplo, para una estructura antropomórfica se adopta una geometría paralela entre la articulación del hombro y el codo para crear una cadena cinemática cerrada.

2.1.3. Robótica industrial y sus ventajas

La robótica industrial es la disciplina sobre el diseño de robots, control y aplicaciones en la industria. La relación de un robot manipulador con aplicaciones industriales se refiere a que deben operar en ambientes estructurados, lo cual significa que las características geométricas y físicas del entorno son conocidas previamente.

Los factores principales que han determinado la propagación de la tecnología robótica en un amplio rango de aplicaciones para la industria manufacturera son: la reducción de costos, incremento de la productividad, mejoras en los estándares de calidad del producto y la posibilidad de eliminar trabajos desagradables y peligrosos para los operadores humanos en los sistemas de manufactura (Siciliano et al., 2009, Craig, 2005).

Un robot industrial se considera como una máquina que posee características de versatilidad y flexibilidad. De acuerdo a la Federación Internacional de Robótica (IFR), por sus siglas en inglés, la implementación de robots manipuladores de manera predominante está en la industria automotriz. Los robots industriales presentan capacidades fundamentales que son muy útiles en los procesos de manufactura como lo es el manejo de materiales, manipulación de piezas y la medición.

En procesos de manufactura, cada objeto o pieza tiene que ser transferido de un lugar a otro con la finalidad de ser almacenado, maquinado, ensamblado y empaquetado. Durante la transferencia las características físicas del objeto, no deben someterse a ninguna alteración. Es por ello que la capacidad de un robot manipulador para cargar un objeto, moverlo en su espacio de trabajo bajo posiciones o trayectorias definidas y soltarlo, hacen que sea ideal para realizar este tipo de tareas.

2.2. Cinemática de robots manipuladores

En esta sección se estudian los modelos cinemáticos para un robot manipulador tipo cartesiano y rotacional, ambos con dos grados de libertad. Los modelos se basan en el empleo de transformaciones entre sistemas de referencia.

Para encontrar la cinemática directa de un robot manipulador en particular, es importante mencionar que esta depende de la configuración geométrica, grados de libertad y del mismo modo el tipo de articulación ya sea rotacional (R) o prismática (P) con los que estén constituidos. Lo anterior da origen a diferentes tipos de robots, entre los más importantes se encuentran los robots antropomórficos con 3 articulaciones rotacionales (RRR), configuración SCARA con 2 articulaciones rotacionales y una prismática (RRP), configuración esférica (RRP), configuración cilíndrica (RPP) y configuración cartesiana (PPP).

La cinemática del robot estudia el movimiento del mismo con respecto a un sistema de referencia fijo sin considerar las fuerzas que lo producen. En otras palabras, se interesa por la descripción analítica del movimiento espacial del robot como función del tiempo y en particular por las relaciones entre la posición y orientación del extremo final del robot de acuerdo a los valores que toman sus coordenadas articulares (Barrientos et al.,

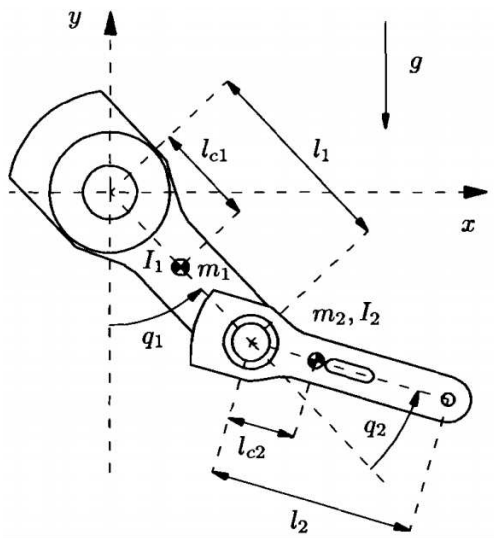


Figura 2.2: Marco de referencia base x-y del robot

1996). Existen dos problemas fundamentales a resolver en la cinemática del robot, los cuales son referidos como problema de cinemática directa e inversa.

2.2.1. Cinemática directa y convención Denavit-Hartenberg

La cinemática directa consiste en determinar cuál es la posición y orientación del extremo final del robot respecto a un sistema de coordenadas que se toma como referencia y los parámetros de los elementos geométricos del robot.

La cinemática directa es un vector que relaciona las coordenadas articulares con las coordenadas cartesianas del robot $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, donde n son los grados de libertad y m es la dimensión de las coordenadas cartesianas.

Para obtener el modelo de cinemática directa en robots manipuladores se debe considerar a un robot manipulador con n grados de libertad (gdl) colocado en una superficie fija. Además el usuario define un marco de referencia también fijo en algún lugar de la superficie; dicho marco de referencia suele denominarse marco referencial de base o inercial.

El problema de la determinación del modelo cinemático directo del robot consiste en expresar la posición y orientación (si procede) de un marco de referencia colocado sólidamente en la parte terminal del último eslabón del robot referida al marco referencial de base en término de las coordenadas articulares del robot. La solución de este planteamiento se reduce, desde un punto de vista matemático, a la solución de un problema geométrico (Kelly and Santibañez, 2003).

Con relación a un robot manipulador de 2 gdl como el mostrado en la Figura 2.2 se define en primera instancia el marco referencial de base como un sistema cartesiano de 2 dimensiones cuyo origen se localiza exactamente en la primera articulación del robot. Las coordenadas cartesianas x e y denotan la posición del extremo final del segundo

eslabón con respecto al marco referencial de base.

Para el caso en donde se tiene a un robot manipulador en el espacio tridimensional con movimientos traslacionales y diversas orientaciones, el modelo cinemático directo expresado como función vectorial se presenta en la ecuación (2.1)(Reyes, 2011), en la cual se convierten coordenadas articulares del robot a coordenadas cartesianas.

La relación entre estas define al modelo cinemático directo propiamente dicho:

$$[x\ y\ z\ \theta\ \phi\ \psi]^T = f_R(q) \tag{2.1}$$

donde: $q \in \mathbb{R}^n$, representa el vector de posiciones articulares del robot, θ, ϕ, $\psi \in \mathbb{R}$ representan la orientación de la herramienta final del robot, f_R es una función continúa y diferenciable de estado q y $[x, y, z]^T$ son las coordenadas cartesianas.

En este trabajo se contempla el uso de un manipulador de 2 *gdl*. De esta manera la relación que define al modelo cinemático directo se presenta en la ecuación (2.2).

$$\begin{bmatrix} x \\ y \end{bmatrix} = f(q_1, q_2) \tag{2.2}$$

donde: $f : \mathbb{R}^2 \to \mathbb{R}^2$.

El modelo de cinemática directa expresado en (2.2) y que corresponde a la Figura 2.2, muestra que el manipulador sólo responde a dos coordenadas cartesianas x e y, esto debido a que sus movimientos están limitados a un plano, en este sentido no existen orientaciones (movimientos rotacionales) en el extremo final del robot. Además, el posicionamiento del robot se encuentra en función de las coordenadas articulares q_1 y q_2.

Cabe mencionar que existen dos maneras de obtener la cinemática directa, la primera se refiere a un método geométrico en el cual se considera precisamente las relaciones geométricas que se mantienen de acuerdo a distintos marcos de referencia y el segundo método es el más conocido llamado de Denavit-Hartenberg (D-H).

El método geométrico es muy sencillo de implementar, ya que solo se necesita de conocimientos elementales de funciones trigonométricas y las ya conocidas leyes de senos y cosenos. Sin embargo a medida que los grados de libertad aumentan en un robot manipulador esté método incrementará su dificultad de aplicación, para este caso el algoritmo D-H dará mejores resultados.

Convención Denavit-Hartenberg

Una convención comúnmente usada para seleccionar marcos de referencia en aplicaciones robóticas es la denominada convención D-H. Este método permite expresar las coordenadas del efector final descritas en un marco de referencia adjunto a éste, en un marco de referencia fijo, el cual en la mayoría de los casos, se coloca en la base del manipulador. En esta convención cada transformación homogénea H_i es representada como el producto de cuatro transformaciones básicas (Ollero, 2007, Barrientos et al., 1996),

$$H_i = Rot_{z,\theta_i}\ Trans_{z,d_i}\ Trans_{x,a_i}\ Rot_{x,\alpha_i} \tag{2.3}$$

donde Rot_{z,θ_i} y Rot_{x,α_i} son las matrices de rotación medidas alrededor del eje z con el ángulo θ y con respecto al eje x_i para el ángulo α respectivamente. $Trans_{z,d_i}$ y $Trans_{x,a_i}$ son las matrices de traslación tanto si se trata de articulaciones prismáticas o medidas respecto a las longitudes de cada eslabón. En forma matricial la ecuación (2.3) se expresa como:

$$H_i = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.4)$$

donde C y S representan las funciones trigonométricas *coseno* y *seno* calculadas para el ángulo θ o α según corresponda. El parámetro d_i corresponde a las articulaciones lineales del robot y a_i es la longitud del eslabón i-ésimo. Realizando la multiplicación matricial se obtiene la matriz de transformación H_i de la ecuación (2.4):

$$H_i = \begin{bmatrix} C\theta_i & -S\theta_i C\alpha_i & S\theta_i S\alpha_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\alpha_i & -C\theta_i S\alpha_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.5)$$

La matriz de transformación homogénea nos permite encontrar las coordenadas del efector final respecto a un marco de referencia fijo adjunto a la base del manipulador independientemente de las rotaciones o traslaciones que pueda sufrir dicho efector.

Algoritmo D-H

Con el algoritmo D-H se puede deducir la cinemática directa para cualquier manipulador, sin importar los grados de libertad que contenga. Cabe mencionar que la elección de la referencia determinara la configuración del manipulador y de la misma manera su modelo cinemático. A continuación se presentan los pasos a seguir para la aplicación del método D-H (W.Spong et al., 2004).

1. Localizar y etiquetar los ejes articulares $z_0, \cdots, z_{n-1}$, como se muestra en la Figura 2.3.

2. Establecer el sistema de referencia base. Configurar el origen en cualquier lugar sobre el eje z_0. Los ejes x_0, y_0 son determinados de acuerdo a la regla de la mano derecha. Para $i = 0, ..., n-1$, realizar los pasos 3-5.

3. Localizar el origen 0_i donde la normal común a z_i y z_{i-1} intercepta z_i. Si z_i intercepta z_{i-1} localizar 0_i en esta intersección. Si z_i y z_{i-1} son paralelos, localizar O_i en cualquier posición conveniente a lo largo de z_i.

4. Establecer x_i a lo largo de la normal común entre z_{i-1} y z_i a través de 0_i o en la dirección normal al plano $z_{i-1} - z_i$ si z_{i-1} y z_i se interceptan.

5. Establecer y_i para complementar con la regla de la mano derecha.

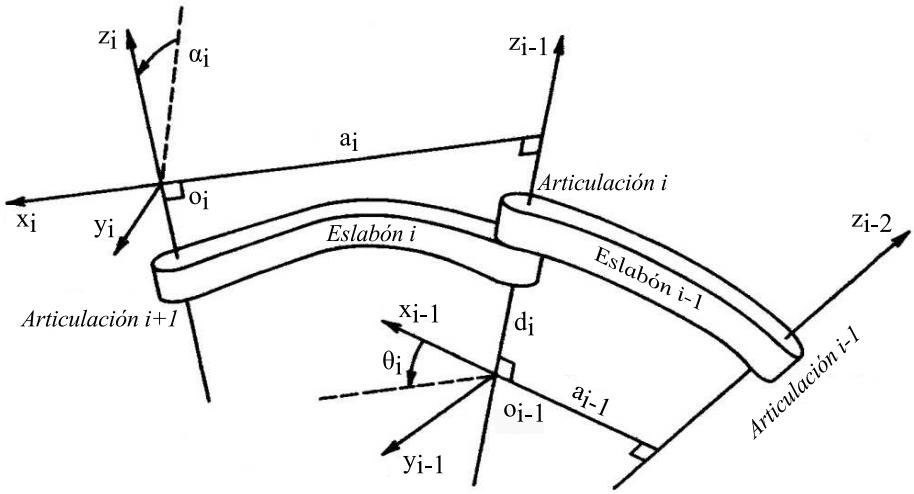


Figura 2.3: Asignación de los marcos de referencia

Parámetros	Características
l_i	Longitud del eslabón i-ésimo
d_i	Articulación lineal o prismática
α_i	Ángulos entre los ejes z_{i-1} y z_i medidos con respecto al eje x_i
θ_i	Articulaciones rotacionales que representan el ángulo entre los ejes x_{i-1} y x_i medido alrededor del eje z_i

Tabla 2.1: Parámetros utilizados en el método D-H

6. Establecer el marco del efector final $0_n x_n y_n z_n$.

7. Crear una tabla a partir de los parámetros de los eslabones $l_i, d_i, \alpha_i, \theta_i$.

En la Tabla 2.1 se presentan los principales parámetros que son utilizados para la aplicación del metodo Denavit-Hartemberg con la finalidad de encontrar la cinemática directa de robots manipuladores.

Cinemática directa para un robot rotacional de *2 gdl*

A continuación se considera un robot manipulador con dos articulaciones rotacionales el cual se mueve en el plano vertical x_0, y_0 como el que se muestra en la Figura 2.4.

En la Tabla 2.2 (Reyes, 2011) se presentan los parámetros Denavit-Hartenberg, considerando que las dos articulaciones que forman al robot son del tipo rotacional.

Al emplear la matriz de transformación (2.5) se encuentra la cinemática directa del manipulador rotacional. Por lo tanto para el movimiento de las articulaciones referidas

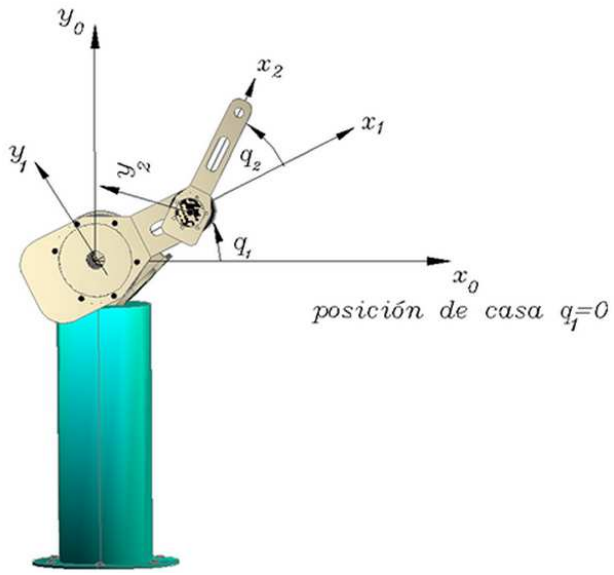


Figura 2.4: Robot planar 2 gdl con posición de casa en el eje x_0. Reyes (2011)

Eslabón	$\mathbf{l_i}$	$\boldsymbol{\alpha}$	$\mathbf{d_i}$	$\boldsymbol{\theta_i}$
1	l_1	0	0	q_1
2	l_2	0	0	q_2

Tabla 2.2: Parámetros D-H para un manipulador rotacional de 2gdl

a la base del robot se tiene que:

$$H_0^2 = H_1 H_2 \tag{2.6}$$

Las matrices H_1 y H_2 que se calculan en (2.7) y (2.8), corresponden a los movimientos traslacionales y rotacionales del efector final del manipulador con respecto a un marco de referencia fijo.

$$H_0^1 = \begin{bmatrix} cos(q_1) & -sen(q_1) & 0 & l_1\ cos(q_1) \\ sen(q_1) & cos(q_1) & 0 & l_1\ sen(q_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

$$H_1^2 = \begin{bmatrix} cos(q_2) & -sen(q_2) & 0 & l_2\ cos(q_2) \\ sen(q_2) & cos(q_2) & 0 & 1_2\ sen(q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

Al sustituir en la ecuación (2.6) se obtiene la matriz de transformación homogénea para un manipulador rotacional de 2 gdl:

$$H_0^2 = \begin{bmatrix} cos(q_1+q_2) & -sen(q_1+q_2) & 0 & l_1cos(q_1)+l_2cos(q_1+q_2) \\ 0 & 0 & 1 & l_1sen(q_1)+l_2sen(q_1+q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

La ecuación (2.9) es conocida como matriz de transformación homogénea, donde la matriz cuadrada interna formada por las tres primeras filas y columnas representa al movimiento de rotación del efector final y a la cuarta columna formada por las tres primeras filas representa al vector de traslación del efector referido al marco de referencia anclado en la base del manipulador. De acuerdo a lo anterior la cinemática directa para un robot planar de dos grados de libertad expresada en forma matricial es (Reyes, 2011):

$$\begin{bmatrix} x \\ y \end{bmatrix} = f(q) = \begin{bmatrix} l_1\ cos(q_1) + l_2\ cos(q_1 + q_2) \\ l_1\ sen(q_1) + l_2\ sen(q_1 + q_2) \end{bmatrix} \tag{2.10}$$

2.2.2. Cinemática inversa

La posición del extremo final de un robot manipulador se puede calcular de acuerdo a los movimientos traslacionales y rotacionales realizados dentro de su área de trabajo, este problema referido a la cinemática directa se resuelve al encontrar sus coordenadas cartesianas $[x, y, z]^T$ y orientaciones $[\psi, \theta, \phi]$, ambos con respecto a un sistema de referencia fijo $\Sigma_0(x_0, y_0, z_0)$, sin embargo con ello sólo se obtiene la posición actual del robot en función de las coordenadas articulares y por lo tanto se dice que la tarea a desarrollar esta planteada desde el espacio articular (Ollero, 2007).

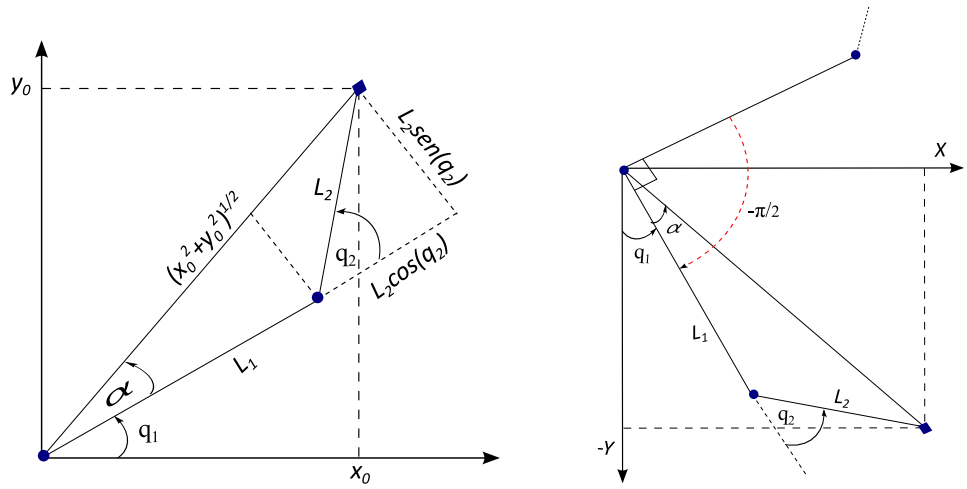
En problemas de aplicación de robots suele ser más conveniente definir sus movimientos desde el espacio cartesiano, tal como sucede con algunas máquinas de control numérico o CNC, debido a la familiaridad que se tiene con las coordenadas cartesianas. Si se requiere seguir alguna trayectoria o tarea en específico, existe la posibilidad de poder planificarla-diseñarla a partir de coordenadas x, y. Este tipo de planteamientos lo resuelve la cinemática inversa, ya que en función de coordenadas cartesianas se obtienen las coordenadas articulares para los motores, con lo cual se indica cuantas unidades en grados o radianes deben girar para alcanzar los puntos o posiciones deseadas.

En otras palabras, el modelo cinemático inverso de robots manipuladores permite obtener las posiciones articulares q_i en términos de la posición y orientación del extremo final del último eslabón referido a un marco de referencia cartesiano base. En la ecuación (2.11) se expresa el modelo cinemático inverso para un robot con 2 grados de libertad, sin considerar la orientación de su efector final (Kelly and Santibañez, 2003).

$$\begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = f^{-1}(x, y) \tag{2.11}$$

donde: $f : \mathbb{R}^2 \to \mathbb{R}^2$.

A diferencia de la cinemática directa, el modelo de cinemática inversa representa una mayor complejidad de solución, ya que en primera instancia se pueden encontrar diferentes soluciones y configuraciones en el robot que definan a la posición deseada, por lo cual se dice que no existe unicidad de solución. Otro inconveniente se debe a la posibilidad de no encontrar solución analítica, en tal situación se tienen algunos métodos posibles, entre las propuestas están las redes neuronales, los iterativos, numéricos y geométricos (Reyes, 2011).



(a) Robot 1^{er} cuadrante (b) Robot 4^{to} cuadrante

Figura 2.5: Análisis de la cinemática inversa por método geométrico

Cinemática inversa para un robot rotacional de *2 gdl*

En este apartado se realiza el cálculo de la cinemática inversa mediante el método geométrico para un manipulador de 2 grados de libertad (Figura 2.5). Para este caso la posición articular $[q_1, q_2]^T$ es calculada en función de las coordenadas cartesianas $[x, y]^T$. El análisis presentado inicia en el primer cuadrante, por lo cual se puede formar un triángulo con lados $L_1 + L_2cos(q_2)$, $L_2sen(q_2)$ y el valor de la hipotenusa $\sqrt{x_0^2 + y_0^2}$, que une al efector final con la base del robot, tal como se observa en la Figura 2.5a. Al aplicar el teorema de pitágoras se obtiene la solución para la coordenada articular q_2 como se presenta a continuación,

$$\begin{aligned} x_0^2 + y_0^2 &= (l_1 + l_2cos(q_2))^2 + (l_2sen(q_2))^2 \\ &= l_1^2 + 2l_1l_2cos(q_2) + l_2^2cos^2(q_2) + l_2^2sen^2(q_2) \\ &= l_1^2 + l_2^2 + 2l_1l_2cos(q_2) \end{aligned} \tag{2.12}$$

De (2.12) se despeja la variable q_2, con lo que finalmente se tiene:

$$q_2 = acos\left(\frac{x_0^2 + y_0^2 - l_1^2 - l_2^2}{2l_1l_2}\right) \tag{2.13}$$

Para obtener el valor de la variable articular q_1 se utiliza el triángulo con lados x_0 y y_0, en donde el ángulo formado por el cateto adyacente y la hipotenusa corresponde a la suma de $\alpha + q_1$, por lo tanto,

$$\alpha + q_1 = atan\left(\frac{y_0}{x_0}\right) \tag{2.14}$$

Despejando q_1 de (2.14) y tomando en cuenta que $\alpha = atan\left(\frac{l_2sen(q_2)}{l_1+l_2cos(q_2)}\right)$ se tiene

$$q_1 = atan\left(\frac{y_0}{x_0}\right) - atan\left(\frac{l_2sen(q_2)}{l_1 + l_2cos(q_2)}\right) \tag{2.15}$$

Es importante mencionar que la ecuación (2.15) representa a la cinemática inversa del manipulador posicionado en el primer cuadrante, sin embargo, el modelo de simulación utilizado para este trabajo se encuentra en el cuarto cuadrante. Para dar solución a este problema, se hace uso de una matriz de rotación (Reyes, 2012) girando $\frac{-\pi}{2}$ en sentido contrario a las manecillas del reloj, con lo cual q_1 ahora se mide desde el eje negativo $-y$ hacia el eje de las abscisas x, como se muestra en la Figura 2.5b. De acuerdo a lo anterior la expresión para q_1 es la siguiente:

$$q_1 = \frac{\pi}{2} + atan\left(\frac{y_0}{x_0}\right) - atan\left(\frac{l_2 sen(q_2)}{l_1 + l_2 cos(q_2)}\right) \tag{2.16}$$

Con las ecuaciones (2.13) y (2.16) se tiene el modelo cinemático inverso para un manipulador de dos grados de libertad, con lo cual no solo se conoce la posición actual del robot en un plano cartesiano, sino además se podrán programar puntos deseadas desde el espacio de tareas.

2.3. Modelo dinámico de robots manipuladores

Para obtener las ecuaciones de movimiento, esto es, el modelo dinámico de los robots manipuladores, un método bastante utilizado es el propuesto por Euler-Lagrange (Kelly and Santibañez, 2003, W.Spong and M.Vidyasagar, 2006, Ollero, 2007). A continuación se presenta la manera de obtener el *modelo dinámico* de un robot manipulador de n grados de libertad *(gdl)*, formado de eslabones rígidos conectados por articulaciones no elásticas dispuestos en una cadena cinemática abierta (Loria and Panteley, 1999). Las ecuaciones de Euler-Lagrange se pueden expresar de la siguiente manera:

$$\tau_i = \frac{d}{dt}\left[\frac{\partial \mathcal{L}(q,\dot{q})}{\partial \dot{q}_i}\right] - \frac{\partial \mathcal{L}(q,\dot{q})}{\partial q_i} \qquad i = 1, 2, 3 \ldots n \tag{2.17}$$

donde:
τ_i son los pares i-ésimos aplicados en los actuadores (servomotores). El lado derecho de la ecuación (2.17) intenta representar todos los fenómenos físicos presentes en el sistema, los cuales están en función de las posiciones, velocidades y aceleraciones articulares.

El lagrangiano $\mathcal{L}(q,\dot{q})$ de un robot manipulador de n grados de libertad expresado en la ecuación (2.17) se define como:

$$\mathcal{L}(q,\dot{q}) = \mathcal{K}(q,\dot{q}) - \mathcal{U}(q) \tag{2.18}$$

donde: $\mathcal{K}(q,\dot{q})$ es la energía cinética del robot y $\mathcal{U}(q)$ es la energía potencial.

La energía cinética $\mathcal{K}(q,\dot{q})$ del robot manipulador tiene una estructura matemática cuadrática bien definida, la cual está en función de la posición y velocidad articular.

$$\mathcal{K}(q,\dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} \tag{2.19}$$

donde: $M(q) \in \mathbb{R}^{n\times n}$ es la matriz de inercias del manipulador y es una matriz definida positiva y simétrica.

Para el caso de la energía potencial presente en el robot manipulador, no se tiene una forma específica, sin embargo, se conoce que tiene una dependencia exclusivamente del vector de posiciones articulares q, ya que su presencia se considera debido a campos conservativos como lo es la gravedad.

Al sustituir la ecuación (2.19) en (2.18), se tiene lo siguiente:

$$\mathcal{L}(q,\dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} - \mathcal{U}(q) \tag{2.20}$$

A partir del lagrangiano expresado en la ecuación (2.20), se pueden obtener la derivadas parciales de acuerdo a las ecuaciones de movimiento de Euler-Lagrange descritas en (2.17), en consecuencia:

$$\frac{\partial \mathcal{L}(q,\dot{q})}{\partial \dot{q}} = \frac{\partial}{\partial \dot{q}}\left[\frac{1}{2}\dot{q}^T M(q)\dot{q}\right] = M(q)\dot{q} \tag{2.21}$$

$$\frac{d}{dt}\left[\frac{\partial \mathcal{L}(q,\dot{q})}{\partial \dot{q}}\right] = \frac{d}{dt}\left[\frac{\partial}{\partial \dot{q}}\left[\frac{1}{2}\dot{q}^T M(q)\dot{q}\right]\right] = M(q)\ddot{q} + \dot{M}(q)\dot{q} \tag{2.22}$$

$$\frac{\partial \mathcal{L}(q,\dot{q})}{\partial q} = \frac{\partial}{\partial q}\left[\frac{1}{2}\dot{q}^T M(q)\dot{q}\right] - \frac{\partial \mathcal{U}(q)}{\partial q} \tag{2.23}$$

Al sustituir (2.22) y (2.23) en (2.17) se obtienen las ecuaciones de movimiento de Euler-Lagrange para un robot de n grados de libertad, las cuales pueden ser expresadas de la siguiente manera:

$$\tau = M(q)\ddot{q} + \dot{M}(q)\dot{q} - \frac{1}{2}\frac{\partial}{\partial q}\left[\dot{q}_T M(q)\dot{q}\right] + \frac{\partial \mathcal{U}(q)}{\partial q} \tag{2.24}$$

El modelo dinámico de un robot manipulador proporciona una descripción completa entre los pares aplicados a los servomotores y el movimiento de la estructura mecánica. Con la formulación de las ecuaciones de Euler-Lagrange, las ecuaciones de movimiento pueden ser obtenidas de manera sistemática independientemente del sistema coordenado (Reyes, 2011).

En forma compacta y para efectos de estudio, el modelo dinámico de un robot manipulador con n *gdl* se expresa en la ecuación (2.25):

$$\tau = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(\dot{q}) \tag{2.25}$$

Siendo: $C(q,\dot{q})\dot{q} = \dot{M}(q)\dot{q} - \frac{1}{2}\frac{\partial}{\partial q}\left[\dot{q}^T M(q)\dot{q}\right]$ y $g(q) = \frac{\partial \mathcal{U}(q)}{\partial q}$

donde $q, \dot{q}, \ddot{q} \in \Re^n$ son vectores de posición, velocidad y aceleración articular, $\tau \in R^n$ es el vector de torques aplicados o ley de control, $f(\dot{q}) \in R^n$ es el vector de fenómenos de fricción (en este trabajo se considera únicamente la fricción viscosa $f(\dot{q}) = b\dot{q}$, $M(q) \in R^{n\times n}$ es la matriz de inercia del manipulador, simétrica y definida positiva, $C(q,\dot{q}) \in R^n$ es el vector de fuerzas centrípetas y de Coriolis y $g(q) \in R^n$ es el vector de torques gravitacionales calculado como el gradiente de la energía potencial $u(q)$ del manipulador.

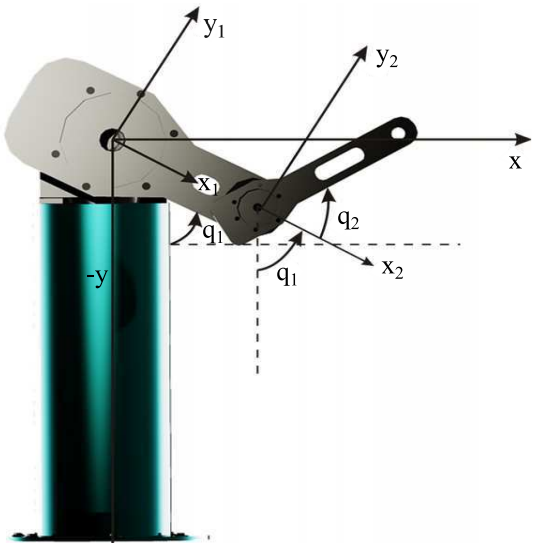


Figura 2.6: Robot manipulador ROTRADI de 2 gdl. Reyes (2011)

Dinámica de un robot rotacional con *2 gdl*

Para obtener el modelo dinámico del manipulador de dos grados de libertad como el mostrado en la Figura 2.6, se debe calcular el lagrangiano $\mathcal{L}(q,\dot{q})$ del sistema expresado en (2.18) y de este modo resolver las derivadas parciales de (2.17).

La energía cinética total del sistema es la sumatoria correspondiente a cada uno de los eslabones,

$$\mathcal{K} = \mathcal{K}_1(q,\dot{q}) + \mathcal{K}_2(q,\dot{q}) \tag{2.26}$$

donde: K_1 y K_2 , son las energías cinéticas asociadas a las masas m_1 y m_2. Tomando en cuenta las inercias debidas al movimiento giratorio de la masas que constituyen a los eslabones,

$$\mathcal{K}_1(q,\dot{q}) = \frac{1}{2}m_1 v_1^2 + \frac{1}{2}I_1\dot{q}_1^2 \tag{2.27}$$

$$\mathcal{K}_2(q,\dot{q}) = \frac{1}{2}m_2 v_2^2 + \frac{1}{2}I_2[\dot{q}_1+\dot{q}_2]^2 \tag{2.28}$$

La cinemática directa para el manipulador ubicado en el cuarto cuadrante como el de la Figura 2.6 está dada por:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1 sen(q_1) + l_2 sen(q_1+q_2) \\ -l_1 cos(q_1) - l_2 cos(q_1+q_2) \end{bmatrix} \tag{2.29}$$

Las velocidades v_1 y v_2 se calculan derivando (2.29) para los eslabones 1 y 2,

$$\mathcal{K}_1(q,\dot{q}) = \frac{1}{2}m_1 l_{c1}^2\dot{q}_1^2 + \frac{1}{2}I_1\dot{q}_1^2 \tag{2.30}$$

$$\begin{aligned}\mathcal{K}_2(q,\dot{q}) &= \frac{1}{2}m_2 l_1^2\dot{q}_1^2 + \frac{1}{2}m_2 l_{c2}^2[\dot{q}_1^2 + 2\dot{q}_1\dot{q}_2 + \dot{q}_2^2] \\ &+ m_2 l_1 l_{c2}[\dot{q}_1^2 + \dot{q}_1\dot{q}_2]cos(q_2) + \frac{1}{2}I_2[\dot{q}_1+\dot{q}_2]^2\end{aligned} \tag{2.31}$$

El cálculo de la energía potencial total $\mathcal{U}(q) = \mathcal{U}_1 + \mathcal{U}_2$ de los dos eslabones:

$$\mathcal{U}_1(q) = mgh = -m_1 g\ l_{c1} cos(q_1) \tag{2.32}$$
$$\mathcal{U}_2(q) = mgh = -m_2 g\ l_1 cos(q_1) - m_2 g\ l_{c2} cos(q_1 + q_2) \tag{2.33}$$

Al sustituir la energía cinética y potencial total del sistema en (2.18) se obtiene:

$$\begin{aligned} \mathcal{L}(q, \dot{q}) &= \frac{1}{2}[m_1 l_{c1}^2 + m_2 l_1^2]\dot{q}_1^2 + \frac{1}{2} m_2 l_{c2}^2 [\dot{q}_1^2 + 2\dot{q}_1 \dot{q}_2 + \dot{q}_2^2] \\ &+ m_2 l_1 l_{c2} [\dot{q}_1^2 + \dot{q}_1 \dot{q}_2] cos(q_2) + [m_1 l_{c1} + m_2 l_1] g cos(q_1) \\ &+ m_2 g\ l_{c2} cos(q_1 + q_2) + \frac{1}{2} I_1 \dot{q}_1^2 + \frac{1}{2} I_2 [\dot{q}_1 + \dot{q}_2]^2 \end{aligned} \tag{2.34}$$

Empleando el método de Euler-Lagrange de (2.17), se obtienen las ecuaciones dinámicas del robot manipulador de 2 grados de libertad de la forma dada en (2.25).

$$M(q) = \begin{bmatrix} m_1 l_{c1}^2 + m_2 l_1^2 + m_2 l_{c2}^2 + 2 m_2 l_1 l_{c2} cos(q_2) + I_1 + I_2 & m_2 l_{c2}^2 + m_2 l_1 l_{c2} cos(q_2) + I_2 \\ m_2 l_{c2}^2 + m_2 l_1 l_{c2} cos(q_2) + I_2 & m_2 l_{c2}^2 + I_2 \end{bmatrix}$$

$$C(q, \dot{q}) = \begin{bmatrix} -2 m_2 l_1 l_{c2} sen(q_2) \dot{q}_2 & -m_2 l_1 l_{c2} sen(q_2) \dot{q}_2 \\ m_2 l_1 l_{c2} sen(q_2) \dot{q}_1 & 0 \end{bmatrix}$$

$$g(q) = g \begin{bmatrix} (m_1 l_{c1} + m_2 l_1) sen(q_1) + m_2 l_{c2} sen(q_1 + q_2) \\ m_2 l_{c2} sen(q_1 + q_2) \end{bmatrix}$$

Definidos los parámetros anteriores y sin considerar los efectos de fricción en el manipulador el modelo dinámico representado matricialmente queda como sigue:

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = M(q) \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + C(q, \dot{q}) \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + g(q) \tag{2.35}$$

donde: τ_1 y τ_2 son los pares que actúan en las uniones 1,2 respectivamente.

2.4. Dinámica y control PD del manipulador

Para diseñar algún tipo de control ya sea para posición o seguimiento de trayectorias, se debe partir de la ecuación en lazo cerrado del sistema estudiado. Esta ecuación representa los efectos del algoritmo de control implementado dentro de su modelo dinámico. Para el caso de estudio propuesto en este trabajo, se tiene el modelo dinámico parametrizado de un manipulador de dos grados de libertad (Reyes and Kelly, 2001, Reyes, 2011), al cual se le debe implementar un control del tipo PD propuesto en (Arimoto and Takegaki, 1981).

Si despejamos la aceleración $\ddot{q}$ de la ecuación (2.25) correspondiente al modelo dinámico generalizado del manipulador descrito en la sección 2.3 y considerando la derivada del error de posición $\frac{d}{dt}\tilde{q}$ como la diferencia entre la posición deseada q_d, la cual es constante en el tiempo y la posición q medida desde los actuadores, se obtiene la ecuación en lazo cerrado del manipulador como se muestra a continuación,

$$\frac{d}{dt}\begin{bmatrix} \tilde{q} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} -\dot{q} \\ M^{-1}(q)\tau - C(q,\dot{q})\dot{q} - g(q) - f(\dot{q}) \end{bmatrix} \tag{2.36}$$

donde $q_d = [q_{d1}, q_{d2}, \ldots, q_{dn}]^T \in \Re^n$ es el vector de posiciones deseadas y $\tilde{q} = q_d - q \in \Re^n$ es el vector correspondiente a los errores de posición de ambos grados de libertad. El objetivo del control de posición será entonces satisfacer que tanto el error como la velocidad sean cero a medida que el tiempo evoluciona, lo cual se expresa como sigue:

$$\lim_{t\to\infty}\begin{bmatrix} \tilde{q} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{2.37}$$

La ecuación (2.37) garantiza que el efector final del manipulador se localice, conforme avanza el tiempo, en una posición deseada fija y constante con velocidad $\dot{q} = 0$. Para satisfacer dicha ecuación, Takegaki y Arimoto proponen la ley de control proporcional-derivativa PD (Arimoto and Takegaki, 1981) siguiente:

$$\tau = k_p\tilde{q} - k_v\dot{q} + g(q) \tag{2.38}$$

donde $K_p, K_v \in \Re^{n\times n}$ son matrices diagonales definidas positivas de ganancias proporcionales y derivativas. El valor de K_p y K_v puede aproximarse teóricamente, pero en la práctica se elige por prueba y error para satisfacer especificaciones tales como sobretiro o tiempo de establecimiento. Además, si se cambia la posición deseada es necesario sintonizar nuevamente K_p y K_v, lo que limita la precisión del regulador PD para seguimiento de trayectorias punto a punto (Salas et al., 2012, Reyes, 2011).

2.5. Redes neuronales de base radial (RBFNNs)

Las redes neuronales de base radial son redes multicapa con conexiones hacia adelante, las cuales se caracterizan por estar formadas de una sola y única capa oculta. Cada neurona de ésta capa posee un carácter local, en el sentido de que cada neurona se activa para una región diferente, dependiente del espacio de los patrones de entrada. La salida de estas redes son formadas por una combinación lineal de funciones de activación, sin embargo, obteniéndose una relación no lineal entre las variables de entrada y de salida. Es por ello que sus aplicaciones son recomendadas en procesos de estimación con parámetros no lineales (Yu and Li, 2001).

2.5.1. Generalidades

Las redes de neuronas artificiales como también se les conoce son esencialmente un medio de predicción que utilizan técnicas computacionales, las cuales intentan reprodu-

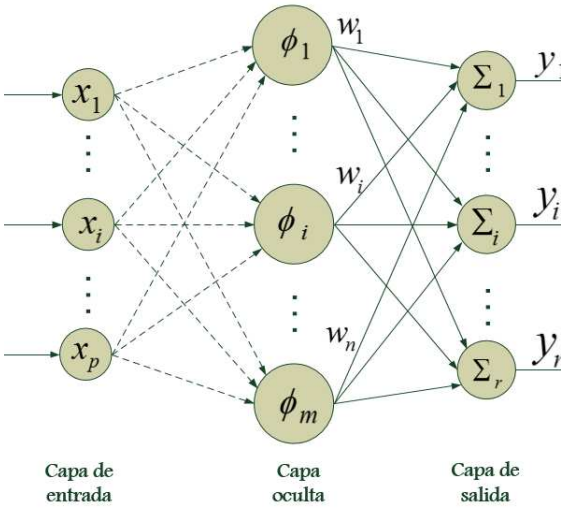


Figura 2.7: Estructura de una red neuronal de base radial

cir a los procesos humanos de pensamiento y razonamiento. Dichas técnicas de neurocomputación han existido desde la década de 1950, sin embargo, debido a que consumían demasiados recursos computacionalmente hablando, no fue sino hasta la década de 1990 cuando se convirtieron en una alternativa de solución para este tipo de problemas (Love, 2007).

De lo anterior se desprende que las redes neuronales son más útiles cuando en el problema se tienen relaciones desconocidas y de forma particular, son buenos en el manejo de relaciones no lineales y dinámicas. La capacidad de predicción de redes neuronales se presta a una variedad de aplicaciones como modelado, estimación, optimización y clasificación de procesos (Haykin, 1999).

Arquitectura

Las redes neuronales de base radial están formadas por tres capas: de entrada, oculta y de salida. De acuerdo con la Figura 2.7, se tiene una red de neuronas de base radial con p neuronas en la capa de entrada, m neuronas en la capa de oculta y r neuronas en la capa de salida. Es importante mencionar que la capa de entrada se compone por un conjunto de neuronas que reciben las señales del exterior, transmitiéndolas a la siguiente capa sin realizar ningún tipo de proceso sobre ellas. Las neuronas de la capa oculta reciben las señales de la capa de entrada, con lo cual realizan una transformación no lineal y local sobre dichas neuronas. Esta capa es la única que incluye componentes no lineales en las redes de base radial. La capa de salida realiza una combinación lineal de las activaciones correspondientes a las neuronas ϕ en la capa oculta, que actúa además como salida de la red. Las activaciones de las neuronas de salida para el patrón de entrada n, $X(n) = (x_1(n), x_2(n), ..., x_p(n))$, denotadas como y_k, vienen dadas por la siguiente expresión (Isasi and Galván, 2004):

$$y_k(n) = u_k + \sum_{i=1}^{m} \omega_{ik}\phi_i(n) \quad para\ k = 1, 2, \ldots, r \tag{2.39}$$

Donde ω_{ik} representa el peso de la conexión de la neurona de la capa oculta i a la neurona de salida k, u_k es el umbral de salida y $\phi_i(n)$ son las activaciones de las neuronas ocultas para el patrón de entradas $X(n)$.

Cabe señalar que este tipo de redes tiene como característica principal que las conexiones de la capa de entrada a la capa oculta no llevan asociado ningún peso, sin embargo, en las conexiones de la capa oculta a la capa de salida sí llevan asociado un número real o peso de la conexión denotado en la Figura 2.7 como w. Además, este tipo de redes poseen un umbral como una conexión más de las neuronas en la capa de salida cuyo valor puede ser constante e igual a 1, aunque para efectos prácticos también puede ser despreciado (Isasi and Galván, 2004).

Funciones de activación

Las funciones de base radial son aquellas que van a determinar las activaciones de las neuronas en la capa oculta de la red, en función del vector de entrada $X(n)$ y vienen dadas por:

$$\phi_i(n) = \phi\left(\frac{\|X(n) - C_i\|}{d_i}\right) \quad para\ i = 1, 2, \ldots, m \tag{2.40}$$

donde ϕ es una función de base radial, $C_i = (c_{i1}, \ldots, c_{ip})$ son vectores que representan los centros, d_i son los números reales que representan la anchura de la función de base radial y $\|\cdot\|$ es la distancia euclídea del vector de entrada $X(n)$ al centro C_i, definida por:

$$\|X(n) - C_i\| = \sqrt{\sum_{j=1}^{p}(x_j(n) - c_{ij})^2} \tag{2.41}$$

Como se ha mencionado, la activación de una neurona oculta en las redes de base radial depende de la distancia del patrón de entrada $X(n)$ al centro C_i de dicha función, si la distancia entre una de las neuronas y el valor de entrada es grande, su nivel de activación será mínimo.

La función de base radial ϕ puede adoptar diferentes formas y expresiones, algunas de ellas se presentan en las ecuaciones (2.42),(2.43) y (2.44), sus gráficas se muestran en la Figura 2.8.

- Función gaussiana: $$\phi(r) = e^{\frac{-r^2}{2}} \tag{2.42}$$
- Función inversa cuadrática: $$\phi(r) = \frac{1}{1 + r^2} \tag{2.43}$$
- Función inversa multicuadrática: $$\phi(r) = \frac{1}{\sqrt{1 + r^2}} \tag{2.44}$$

En el contexto de las redes neuronales de base radial, la función de activación gaussiana en (2.42) es una de las más utilizadas (Bishop, 1995).

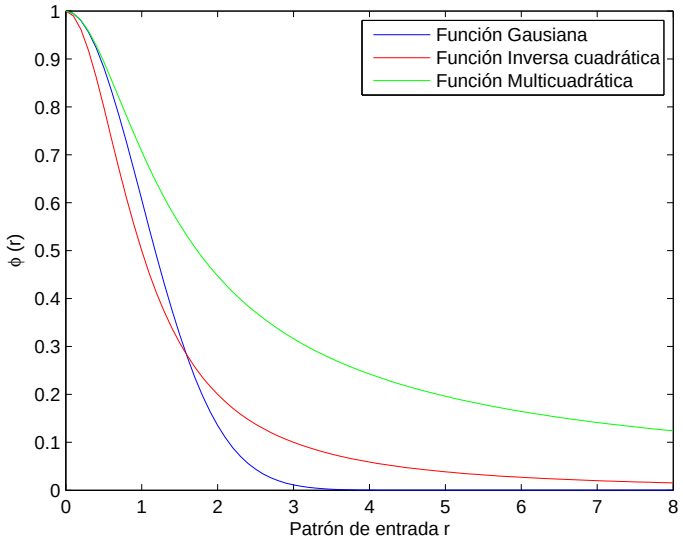


Figura 2.8: Nivel de activación en funciones de base radial

Carácter local de las redes de base radial

Como se observa en la Figura 2.8, las funciones de base radial se caracterizan porque poseen un nivel máximo de activación para valores de entrada cercanos a cero. El carácter local se debe a que cada una de las funciones se activa para una determinada región del espacio definida por los patrones de entrada. Así, si el patrón de entrada a la red $x(n)$ se encuentra en la vecindad del centro C_i, la neurona oculta i alcanzará un valor alto de activación. A medida que el patrón se aleja del centro "dependiendo de la desviación", la activación de la neurona disminuye en tanto que puede activarse otra neurona oculta de la red (Isasi and Galván, 2004).

Esta característica de localidad se refiere a que dado un patrón de entrada a la red, solo aquellas neuronas ocultas cuyos centros se encuentren en la vecindad de dicho patrón se van a activar, el resto de las neuronas ocultas permanecerán inactivas o con un menor nivel de activación.

2.5.2. Proceso de Diseño

A continuación se presenta de manera simplificada el proceso de diseño de una red neuronal de base radial con una salida simple (Figura 2.9). Las distancias de cada una de las neuronas de la capa oculta medida entre los valores de las entradas y los centros de las funciones de base radial pueden calcularse de acuerdo a

$$d_k = \sqrt{([x - C_k]^T [x - C_k])} \tag{2.45}$$

donde x es el vector columna de las entradas actuales, es decir $x = [x_1\ x_2 \cdots x_n]^T$, y C_k es el vector de centros, correspondientes a las coordenadas de los centros de las neuronas en la capa oculta. Al sustituir la distancia d_k la función gaussiana de activación resulta

$$g_k = exp\left(\frac{-d_k^2}{\sigma_k^2}\right) \tag{2.46}$$

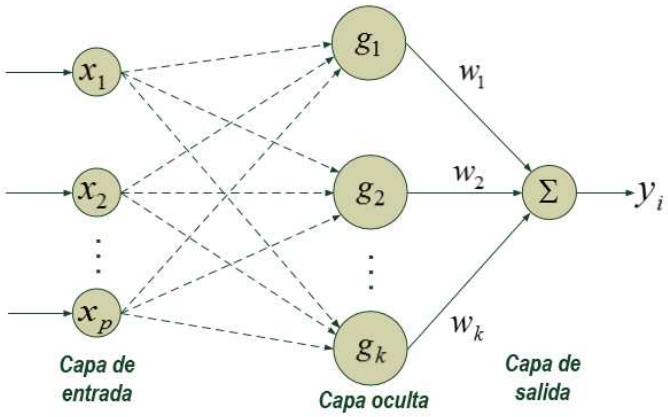


Figura 2.9: Red neuronal con una sola salida

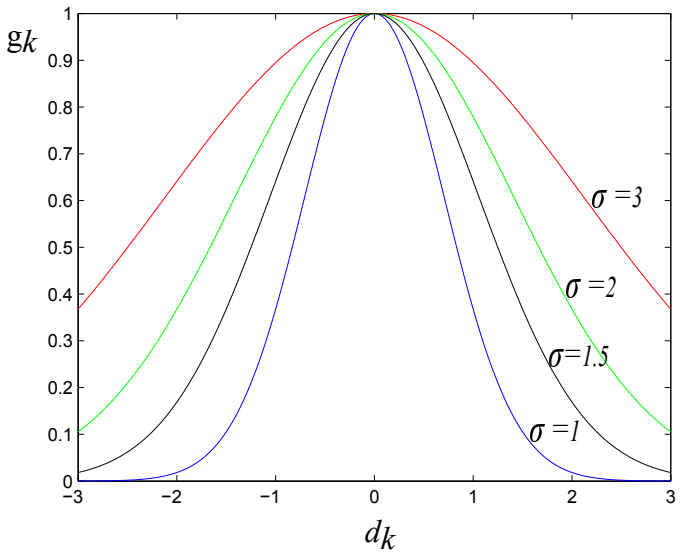


Figura 2.10: Variación de la desviación en una función gaussiana

donde el parámetro σ_k controla el ancho de la función gaussiana (Figura 2.10). La salida $\hat{y}$ de la red neuronal se calcula como

$$\hat{y} = W_0 + \sum_{k=1}^{h} W_k \cdot g_k \tag{2.47}$$

En la Figura 2.9, se puede observar que mientras el vector de entrada x es común a todas las neuronas en la capa oculta, el vector de centros C, el parámetro σ y los pesos de salida W son únicos en cada neurona localizada en la capa oculta.

En la Figura 2.10 se presentan diferentes valores para el parámetro σ. En la medida en que σ se incrementa, la anchura de la función gaussiana también crece.

2.5.3. Proceso de Optimización

El proceso de aprendizaje de las redes neuronales de base radial consiste en establecer apropiadamente los valores de todos sus parámetros: centros y desviaciones σ (anchuras) de las funciones gaussianas en la capa oculta, los pesos que unen a la capa oculta con la capa de salida y los umbrales de activación en la capa de salida, en donde la optimización para este caso se basa en las salidas deseadas para la red neuronal.

Centros

Un método común para establecer el vector de centros es el algoritmo "k-medias", sin embargo, el procedimiento tiene un grado de aleatoriedad asociado, debido a la selección de naturaleza aleatoria de los vectores de centros provisionales o iniciales. Si se conoce el espacio de los valores de entrada, el grado de aleatoriedad disminuye haciendo que con éste método se obtengan buenos resultados.

Desviaciones

Con la finalidad de establecer el parámetro de anchuras σ_k óptimo de la red neuronal, el cual corresponde al nivel de activación de la funciones gaussianas en la capa oculta es típicamente utilizado el llamado método de "vecinos cercanos"(Love, 2007).

Cada neurona en la capa oculta tiene una función de base radial cuyo vector de centros C_k es conocido. De esta manera la distancia entre los centros de cualquier función de base radial o gaussiana que esté en las vecindades puede ser calculada.

Pesos

Los pesos de la sinápsis entre la capa oculta y de salida son normalmente entrenadas a través de múltiples regresiones lineales (Haykin, 1999). Suponiendo que el valor real de la salida y es conocida, entonces el error en la red de predicción esta dada por:

$$y - \hat{y} = \epsilon \tag{2.48}$$

Sustituyendo la ecuación (2.47) en (2.48) se obtiene lo siguiente.

$$y = W_0 + \sum_{k=1}^{h} W_k \cdot g_k + \epsilon \tag{2.49}$$

En la ecuación (2.49) se tiene la salida de una red neuronal optimizada para reducir el error en el sistema. Por tanto, para un conjunto n de datos de entrada con $j = 1 \to n$ y h neuronas en la capa oculta con $(k = 1 \to h)$, en forma matricial se tiene que:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & g_{11} & g_{12} & \cdots & g_{1h} \\ 1 & g_{21} & g_{22} & \cdots & g_{2h} \\ 1 & \vdots & \vdots & \ddots & \vdots \\ 1 & g_{n1} & g_{n2} & \cdots & g_{nh} \end{bmatrix} \begin{bmatrix} W_0 \\ W_1 \\ W_2 \\ \vdots \\ W_h \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \tag{2.50}$$

La ecuación (2.50) representa un proceso de predicción, sin embargo si se conocen las salidas esperadas y y el valor de la matriz G constituida de funciones gaussianas g, sin considerar una optimización (reducción del error) se puede obtener un proceso de entrenamiento calculando el valor de los pesos W como sigue (Haykin, 1999):

$$W = G^{-1}y \tag{2.51}$$

Como se puede deducir (2.51) es obtenida despejando W de (2.50). Es importante mencionar que el entrenamiento de las redes neuronales de base radial para este trabajo se basa en esta ecuación.

Una manera de calcular la diferencia entre la salida deseada y estimada es a través del error cuadrático medio (ECM):

$$ECM = \frac{1}{2}\sum_{j=1}^{4}(y_j - \hat{y}_j)^2 \tag{2.52}$$

donde: y_j son las salidas deseadas y $\hat{y}_j$ son las estimadas. El valor del error cuadrático medio es una medida de ponderación del error promedio correspondiente a la estimación realizada por la red neuronal de base radial.

Capítulo 3

Regulador de posición tipo PD autosintonizable

En este capítulo se propone una ley de control de posición tipo PD para robots manipuladores que incorpora auto-sintonización de las ganancias proporcionales y derivativas en función de la posición deseada. Esta característica mejora el desempeño del regulador PD en el seguimiento de trayectorias aún y cuando no se tenga conocimiento de los parámetros del modelo dinámico del manipulador. Para ello, cada ganancia proporcional o derivativa es sustituida por una red neuronal de base radial entrenada fuera de línea. Se demuestra estabilidad en el sentido de Lyapunov y se valida la ley de control a partir de los resultados de simulación en Matlab R2012b para un manipulador de dos grados de libertad que sigue una trayectoria circular y otra en forma de flor de ocho pétalos, con lo cual se observa una mejora del 74.81 % y del 60.64 % respectivamente, estos valores son calculados a partir del índice de desempeño $\mathcal{L}_2$ en relación al regulador PD convencional de Takegaki y Arimoto (Arimoto and Takegaki, 1981).

3.1. Ley de control PD-N

A continuación se propone una variante del control de posición PD que modifica de forma automática las matrices de ganancias K_p y K_v en función de la posición deseada, desde el espacio cartesiano al articular, a través de la cinemática inversa. El objetivo es mejorar el desempeño de este controlador en el seguimiento de trayectorias punto a punto aún y cuando no se cuente con los parámetros del modelo dinámico (2.25).

La propuesta consiste en sustituir las matrices de ganancias constantes K_p y K_v en (2.38), por matrices diagonales $diag\{k_{pi}(q_d)\}$ y $diag\{k_{vi}(q_d)\}$, para $i = 1, 2, \ldots, n$, donde n es el número de grados de libertad (gdl) del sistema. Las ganancias variables $k_{pi}(q_d)$ y $k_{vi}(q_d)$ son funciones escalares de la variable vectorial q_d que sintonizan las ganancias proporcionales y derivativas en función de la posición deseada. Cada $k_{pi}(q_d)$ o $k_{vi}(q_d)$ corresponde a la salida k_i de una red neuronal de base radial (RBFNN por sus siglas en inglés), como la que se muestra en la Figura 3.1. Estas redes poseen una capa de entrada, una capa oculta de funciones de activación gaussianas $\phi_{i,j}^{P,V}(q_d)$ para $j = 1, 2, ..., m$,

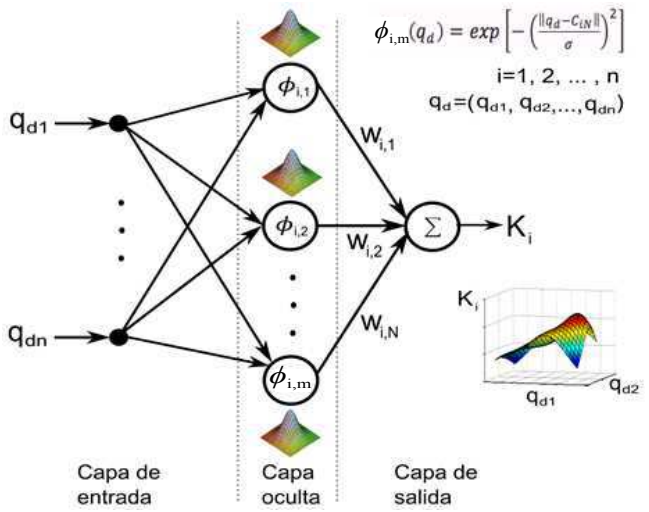


Figura 3.1: Red neuronal de n-ésimo grado de libertad

donde m es el número de neuronas en la capa oculta de cada red y una capa de salida que suma las funciones de activación multiplicadas por factores de ponderación $\omega_{ij}^{P,V}$ (Love, 2007). La notación P, V de $\phi_{i,j}^{P,V}$ hace referencia a funciones de activación para la red asociada a las ganancias proporcionales y derivativas, respectivamente.

Considerando a cada ganancia proporcional y derivativa como una salida aproximada k_i por medio de funciones de base radial, y sustituyendo $diag\{k_{pi}(q_d)\}$ y $diag\{k_{vi}(q_d)\}$ en (2.38) se obtiene la ley de control PD neuronal propuesta, dada por

$$\tau = diag\{K_{pi}(q_d)\}\tilde{q} - diag\{K_{vi}(q_d)\}\dot{q} + g(q) \tag{3.1}$$

donde:

$$k_{pi}(q_d) = \sum_{j=1}^{m} \omega_{ij}^{P} exp\left[-\left(\frac{\|q_d - C_j\|}{\sigma}\right)^2\right] > 0, \quad i = 1, \ldots n \tag{3.2}$$

$$k_{vi}(q_d) = \sum_{j=1}^{m} \omega_{ij}^{V} exp\left[-\left(\frac{\|q_d - C_j\|}{\sigma}\right)^2\right] > 0, \quad i = 1, \ldots n \tag{3.3}$$

Aquí, ω_{ij}^{P} y ω_{ij}^{V} son los pesos de la capas ocultas de las redes neuronales $k_{pi}(q_d)$ y $k_{vi}(q_d)$, respectivamente, $C_j \in \Re^n$ son los centros de las funciones gaussianas, σ es una constante que controla el ancho de las funciones gaussianas y $\|.\|$ denota distancia euclidiana.

En la Figura 3.2 se presenta el diagrama a bloques del sistema en lazo cerrado para el controlador propuesto, denominado control PD-N. El sistema de control propuesto parte de las coordenadas en el espacio cartesiano $[x_d, y_d]$ y como salidas se obtienen las variables de estado que para este caso son la posición y velocidad articular del modelo q y $\dot{q}$ respectivamente. La compensación de gravedad esta determinada en función de la posición del robot. El torque τ necesario para mover a cada una de las articulaciones del manipulador se encuentra representado en la ecuación (3.1). Es importante mencionar que para lograr que las ganancias del control neuronal PD-N sean auto sintonizables las redes neuronales de base radial RBFNNś son entrenadas fuera de línea, es decir cuando el sistema no se encuentra en funcionamiento.

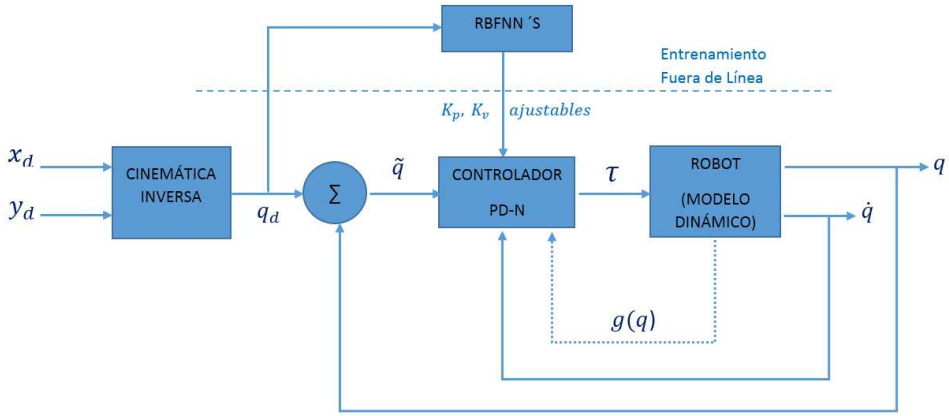


Figura 3.2: Diagrama a bloques del controlador neuronal PD-N

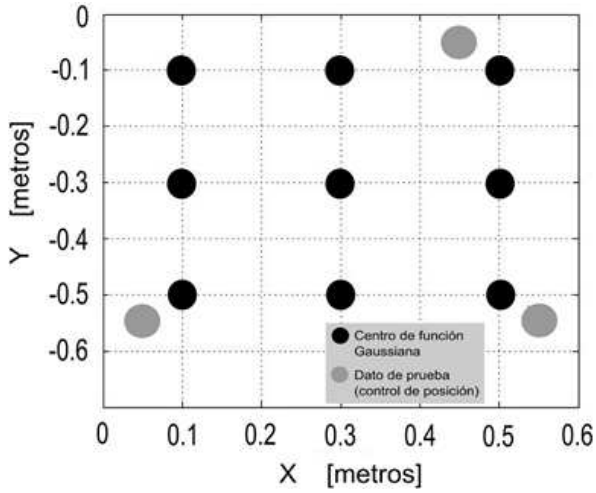


Figura 3.3: Distribución de los 9 puntos de entrenamiento o posiciones deseadas.

Metodología Propuesta

El procedimiento para entrenar las RBFNNś se describe a continuación:

a) Se utilizan $2n$ redes neuronales RBF, donde n es el número de grados de libertad, una para cada ganancia $K_p(q_d)$ y $K_v(q_d)$, tal como se muestra en la Figura 3.1.

b) Seleccione m posiciones deseadas distribuidas uniformemente dentro del espacio de trabajo del manipulador tal como se muestra en la Figura 3.3. Estas posiciones representan los centros de las funciones gaussianas en coordenadas cartesianas, mismas que son convertidas a coordenadas articulares a partir de la Cinemática Inversa del manipulador (2.13) y (2.16)(Reyes, 2012). Estas muestras forman el conjunto de datos de entrenamiento de entrada $C = \{C_1, C_2, \ldots, C_m\}$, donde $C_j \in \Re^n$.

c) Sintonice manualmente las ganancias K_p y K_v del controlador (2.38) para cada posición del conjunto C. El conjunto de datos obtenidos corresponden a los valores

de entrenamiento asignados a la capa de salida de las $2n$ redes neuronales, esto es, $K_i^P = \{K_{i1}^P, K_{i2}^P, \ldots, K_{nm}^P\}$ y $K_i^V = \{K_{i1}^V, K_{i2}^V, \ldots, K_{nm}^V\}$, donde $K_{ij}^{P,V}$ representa a las ganancias proporcional (P) o derivativa (V) de la i-ésima articulación para el j-ésimo dato de entrenamiento.

d) Asigne el mismo valor σ para todas las funciones de activación, procurando que la activación de las funciones gaussianas en la capa oculta sea menor al 50 % entre neuronas adyacentes, esto es:

$$\phi_{i,kj}^{P,V} = exp\left[-\left(\frac{\|Q_k - Q_j\|}{\sigma}\right)^2\right] \leq 0{,}5 \quad \forall k \neq j \tag{3.4}$$

con $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots m$ y $k = 1, 2, \ldots m$, donde m es el número de neuronas en la capa oculta. Este valor de σ permite que las neuronas en la capa oculta se especialicen en regiones del plano xy con centros en cada coordenada C_i. El máximo nivel de activación de estas neuronas se alcanzará cuando el valor de entrada se encuentre muy cercano a su centro.

e) Calcule los pesos ω_{ij}^P y ω_{ij}^V de cada red neuronal a partir del correspondiente sistema de ecuaciones:

$$\begin{bmatrix} K_{i1}^{P,V} \\ K_{i2}^{P,V} \\ \vdots \\ K_{im} \end{bmatrix} = \begin{bmatrix} \phi_{i,11}^{P,V} & \phi_{i,12}^{P,V} & \cdots & \phi_{i,1m}^{P,V} \\ \phi_{i,21}^{P,V} & \phi_{i,22}^{P,V} & \cdots & \phi_{i,2m}^{P,V} \\ \vdots & \vdots & \cdots & \vdots \\ \phi_{i,m1}^{P,V} & \phi_{i,m2}^{P,V} & \cdots & \phi_{i,mm}^{P,V} \end{bmatrix} \begin{bmatrix} \omega_{i1}^{P,V} \\ \omega_{i2}^{P,V} \\ \vdots \\ \omega_{im}^{P,V} \end{bmatrix} \tag{3.5}$$

f) Sustituya en (3.2) y (3.3) los valores obtenidos de σ, C_j, ω_{ij}^P y ω_{ij}^V para obtener $k_{pi}(q_d)$ y $k_{vi}(q_d)$ como funciones de la posición deseada q_d.

Con esta metodología se obtienen $2n$ redes entrenadas fuera de línea a las que se les ha denominado "redes de interpolación". Alternativamente, se pueden optimizar los parámetros σ, C_j, ω_{ij}^P y ω_{ij}^V con métodos de búsqueda, como el descenso del gradiente (Aved'yan, 1995).

3.2. Demostración de estabilidad

La demostración de estabilidad con la ley de control (3.1) por el método directo de Lyapunov es similar a la presentada en (Arimoto and Takegaki, 1981) para el control PD convencional, debido a que $diag\{k_{pi}(q_d)\}$ y $diag\{k_{vi}(q_d)\}$ no son funciones del tiempo o de los estados del sistema. Sea entonces la siguiente función candidata de Lyapunov,

$$V(\dot{q}, \tilde{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} + \frac{1}{2}\tilde{q}^T diag\{k_{pi}(q_d)\}\tilde{q} > 0 \tag{3.6}$$

la cual es una función de energía que combina la energía cinética y potencial. Esta función es definida positiva, si se cumple que $k_{pi} > 0, \forall i = 1, 2, \ldots, n$. Derivando con

respecto al tiempo y aplicando la propiedad vectorial $x^T y \equiv y^T x$, se tiene:

$$\dot{V}(\dot{q}, \tilde{q}) = \dot{q}^T M(q)\ddot{q} + \frac{1}{2}\dot{q}^T \dot{M}(q)\dot{q} - \tilde{q}^T diag\{k_{pi}(q_d)\}\dot{q} \tag{3.7}$$

Sustituyendo la aceleración $\ddot{q}$ de (2.36) en (3.7) y considerando a τ como en (3.1), y luego de aplicar la propiedad de anti-simetría $\dot{q}^T \dot{M}(q)\dot{q} - 2\dot{q}^T C(q, \dot{q})\dot{q} = 0$ se obtiene:

$$\dot{V}(\dot{q}, \tilde{q}) = -\dot{q}^T diag\{k_{vi}(q_d)\}\dot{q} - \dot{q}^T B\dot{q} \leq 0 \tag{3.8}$$

Esta función semidefinida negativa indica que la energía disminuirá a medida en que el manipulador se acerque al punto de equilibrio conforme el tiempo evolucione. De manera análoga que en el caso de la ganancia proporcional, lo anterior es cierto si $k_{vi}(q_d)\ 0, \forall i = 1, 2, \ldots, n$. Así, se demuestra estabilidad global del punto de equilibrio $[\tilde{q}\ \dot{q}]^T = [0\ 0]^T$. Además, dado que (2.36) es una ecuación diferencial autónoma, es posible demostrar la estabilidad asintótica del punto de equilibrio mediante el teorema de Barbashin-Krasovskii-LaSalle (Kelly and Santibañez, 2003).

3.3. Resultados de simulación

En esta sección se presentan los resultados obtenidos después de realizar el entrenamiento de la redes neuronales de base radial.

Con la finalidad de validar el control de posición propuesto PD-N en el problema de seguimiento de trayectorias punto a punto, en primera instancia se realizan simulaciones cuando la posición deseada es fija (problema de regulación (Reyes and Rosado, 2005)). Posteriormente se simulan tres trayectorias y se comparan los resultados obtenidos correspondientes al seguimiento de las mismas trayectorias con las leyes de control PD (Arimoto and Takegaki, 1981) y TANH (Reyes, 2011), bajo las mismas condiciones. Las simulaciones se realizaron en MatLab 2012b con los valores paramétricos del modelo dinámico presentado en la ecuación (2.25) correspondientes al robot manipulador reportado en (Reyes and Kelly, 2001, Reyes, 2011).

Entrenamiento de las redes neuronales

Al aplicar la metodología presentada en la sección 3.1 se obtuvieron los siguientes resultados:

a).- En el primer paso se determina el número de redes neuronales que deben ser creadas. Debido a que el manipulador contempla dos grados de libertad y sustituyendo en $2n$ para el proceso de diseño, el resultado obtenido son cuatro redes neuronales RBF como las mostradas en la Figura 3.1, las cuales se utilizan para aproximar a $k_{p1}(q_d)$, $k_{p2}(q_d)$, $k_{v1}(q_d)$ y $k_{v2}(q_d)$.

b).- Los datos de entrada para el proceso de entrenamiento son tomados desde el espacio de trabajo del robot manipulador en coordenadas cartesianas mostrado en la Figura 3.3 y los cuales son transformados al espacio articular a través de la cinemática inversa como se observa en la Figura 3.5. De acuerdo a lo anterior se eligieron nueve posiciones

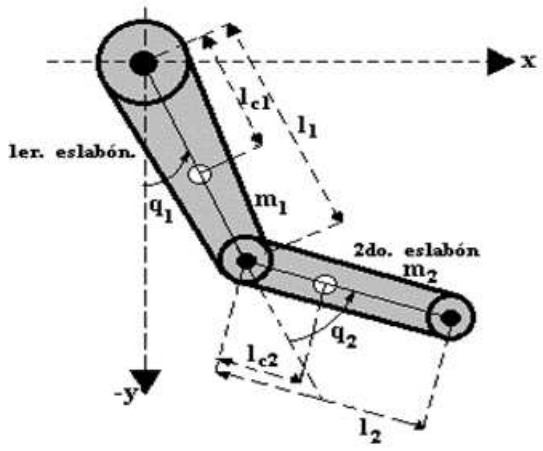


Figura 3.4: Manipulador con dos grados de libertad

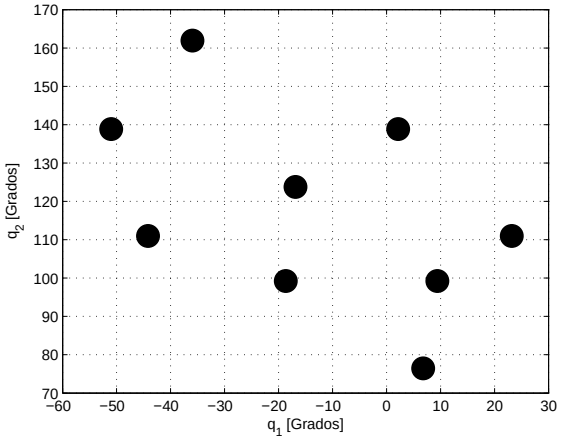


Figura 3.5: Puntos de entrenamiento en el espacio articular

deseadas ($m = 9$) distribuidas uniformemente, estas posiciones representan a los centros de las neuronas correspondientes a la capa oculta y cuya finalidad es que cada una de ellas se especialice en una región determinada. En la Figura 3.5 se muestran las coordenadas articulares del robot manipulador que representan posiciones deseadas en grados, las cuales deben ser adoptadas por los actuadores localizados en cada una de las articulaciones q_1 y q_2.

c).- En este tercer paso se debe sintonizar a las $2nm = 36$ ganancias tanto proporcionales como derivativas K_{1j}^P, K_{2j}^P, K_{1j}^V, K_{2j}^V, donde $n = 2$ son los grados de libertad, $m = 9$ corresponden a los datos de entrenamiento para $j = 1, 2, \ldots, m$. La sintonización se realiza de forma manual y bajo las siguientes condiciones: sobreimpulso $\leq 1\,\%$, tiempo de respuesta ≤ 1 seg, $\tau_1 < 150$ Nm y $\tau_2 \leq 15$ Nm, estos valores de torque son seleccionados con el objetivo de evitar saturación en los actuadores (Santibanez et al., 1998).

d).- Para establecer el valor del parámetro σ se realizaron simulaciones a prueba y error con el objetivo de obtener los mejores resultados tanto para el control de posición como de trayectoria. Debido a que el valor de las funciones de activación depende de las

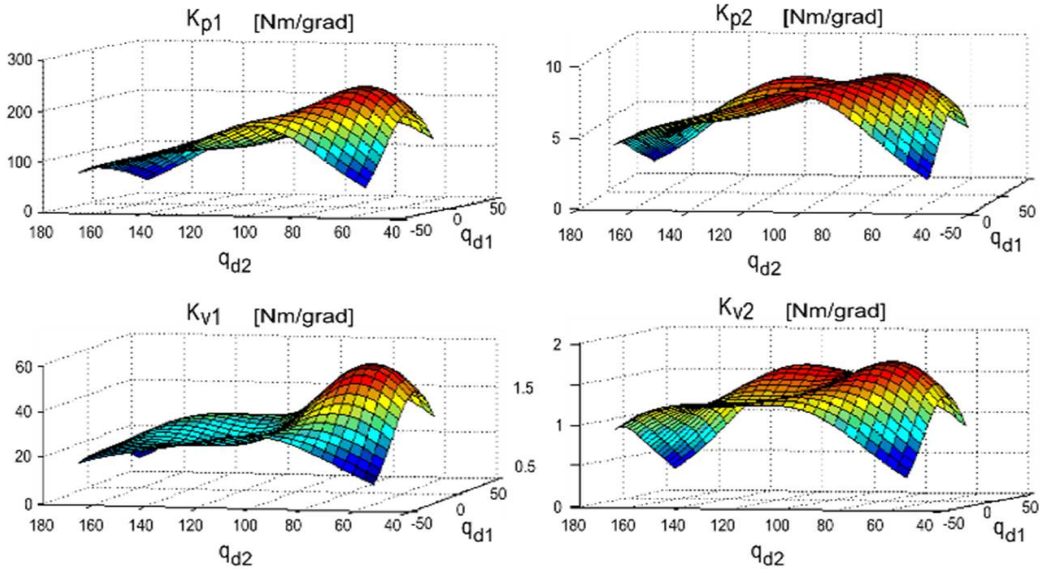


Figura 3.6: Resultado del entrenamiento de la red neuronal

distancias entre los centros de las neuronas de la capa oculta y los centros de los datos de entrenamiento, dependiendo del dato de entrada la activación será mayor para las neuronas que se encuentren cercanas y menor para las alejadas. Si el valor del parámetro σ se encuentra por encima de 1 la mayoría de neuronas serán activadas al 100 % y con ello la existencia de traslape entre neuronas adyacentes. Con la finalidad de evitar un posible sobre-entrenamiento se elige para este proceso un valor de activación menor o igual al 50 % (0.5) para las neuronas que se encuentran cerca del dato de entrada. De esta manera el valor de anchura establecido para la red neuronal es de σ=0.6 con lo cual se garantiza la condición dada en la ecuación (3.4).

e).- En este paso se obtiene el valor de los pesos ω o salidas de cada red neuronal de base radial, las cuales son calculadas a partir de una matriz de funciones gaussianas y un vector de ganancias ya sean proporcionales o derivativas según sea el caso. El valor de ganancias es conocido ya que corresponden a la sintonización manual realizada en el paso del inciso c), la matriz de funciones es calculada desde la ecuación (2.40) presentada en la sección 2.5.1. Al resolver cuatro sistemas de ecuaciones de la forma presentada en la ecuación (3.5), se consiguen los valores para ω_{1j}^{P}, ω_{2j}^{P}, ω_{1j}^{V} y ω_{2j}^{V}.

f).- Finalmente para obtener el valor deseado de las ganancias $k_{p1}(q_d)$, $k_{p2}(q_d)$, $k_{v1}(q_d)$ y $k_{v2}(q_d)$ como resultado del entrenamiento de cada red neuronal, se sustituyen los valores de los pesos ω calculados en el inciso e), el valor del parámetro σ obtenido en el inciso d) y los centros seleccionados en el inciso b), con lo cual el resultado obtenido es el conjunto de cuatro redes neuronales cuyos modelos matemáticos permiten la sintonización automática de las ganancias proporcionales y derivativas para el control tipo PD y tienen la forma de (3.2) y (3.3).

En la Figura 3.6 se observa el comportamiento de las ganancias proporcional y derivativa de las articulaciones 1 y 2 del robot manipulador presentado en la Figura 3.4, las cuales son aproximadas mediante redes neuronales RBF y sus valores son calculados en función de las posiciones deseadas articulares $q_d = [q_{d1}, q_{d2}]^T$.

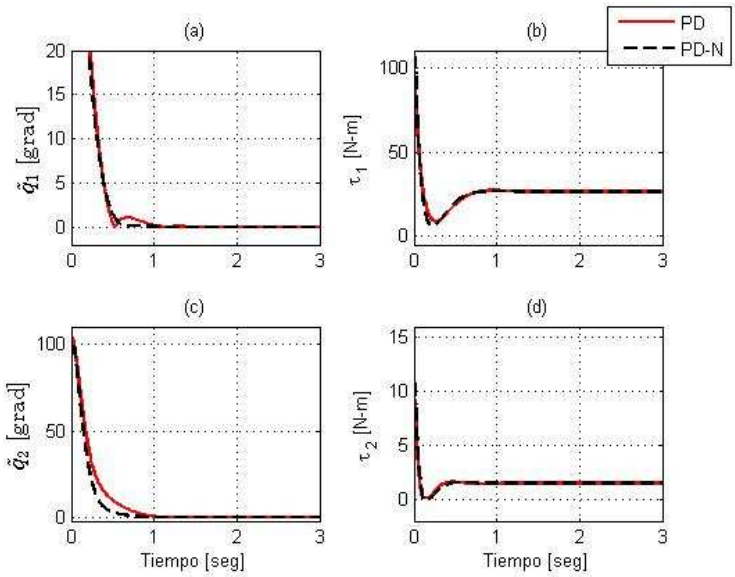


Figura 3.7: Gráfica del error y torques articulares para el punto 1

3.3.1. Control de posición

Después de realizar el entrenamiento de las redes neuronales, se procede a validar el control PD-N para el caso en donde las posiciones deseadas son consideradas como puntos fijos. De la Figura 3.3 correspondiente a la distribución de los centros de las funciones gaussianas dentro del espacio de trabajo del manipulador, se seleccionan tres puntos extremos dados como un par de coordenadas $[x_d, y_d]$, los cuales encuentran su correspondencia en las coordenadas articulares como $[qd_1, qd_2]$, estos datos se especifican en la Tabla 3.1.

Puntos	**Coordenadas Cartesianas (metros)**	**Coordenadas Articulares (grados)**
1	0.05, -0.55	-46.9530, 104.2949
2	0.45, -0.05	23.8636, 119.5924
3	0.55, -0.54	14.4427, 62.1659

Tabla 3.1: Puntos límite como datos de prueba

En las Figuras 3.7, 3.8, 3.9 se presentan los resultados de auto-sintonización del controlador neuronal PD-N para los puntos de prueba 1,2 y 3, en donde se puede observar que se cumple con los criterios de sintonización utilizados para el proceso de entrenamiento descrito en el inciso c) del apartado anterior. El tiempo de respuesta del sistema $t \leq 1\ seg$ y el valor de los torques en las articulaciones no excede los limites de saturación (Santibanez et al., 1998).

Es importante mencionar que cuando el valor de la ganancia proporcional k_p aumenta, el tiempo de respuesta del sistema mejora y además se reduce el error en estado estable, sin embargo si la sintonización de la ganancia k_v no es la correcta el fenómeno de oscilación

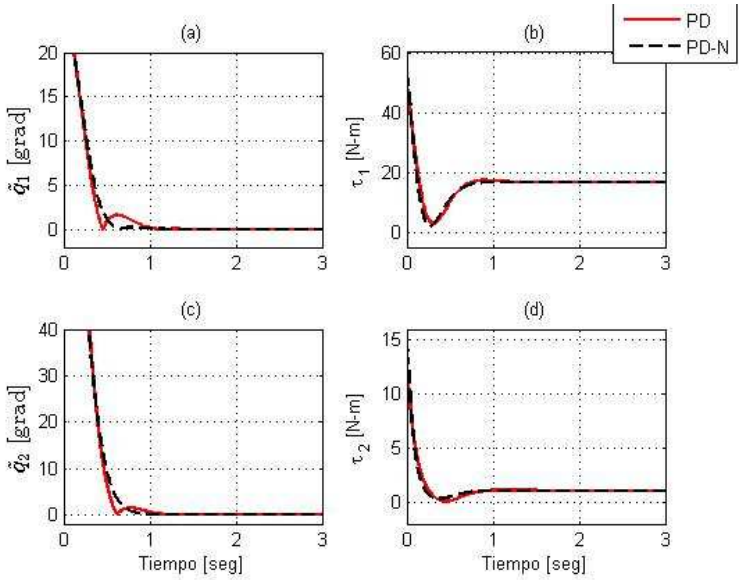


Figura 3.8: Gráfica del error y torques articulares para el punto 2

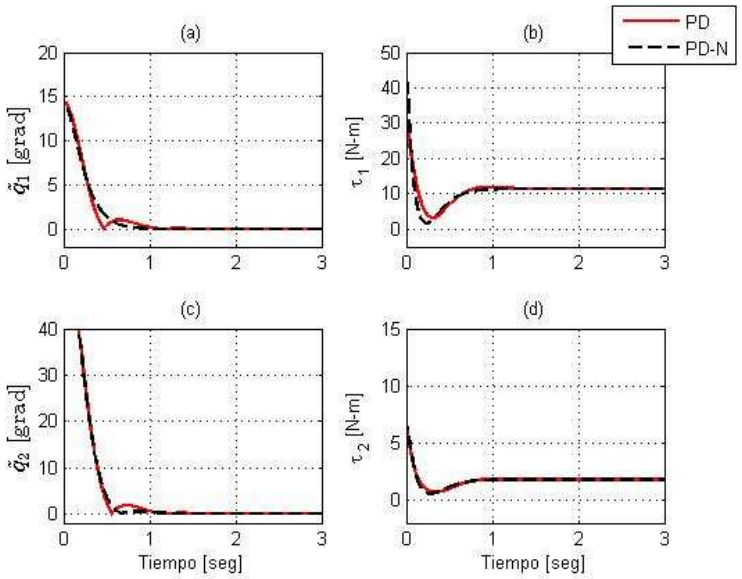
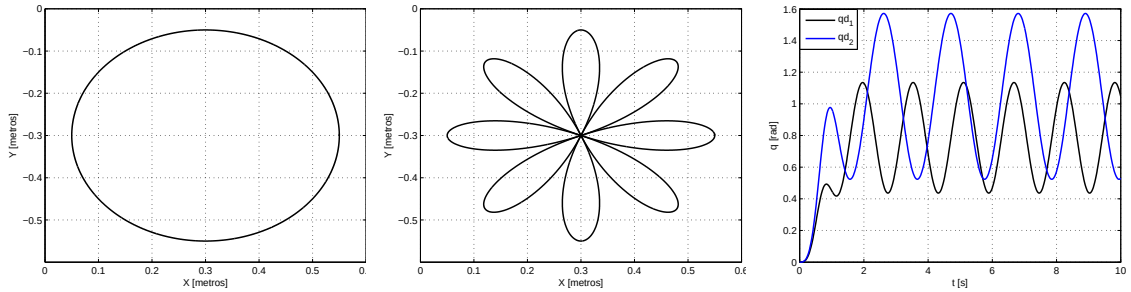


Figura 3.9: Gráfica del error y torques articulares para el punto 3

o sobretiro se incrementará y con ello el error de posición articular, esto es debido a que la ganancia derivativa k_v funciona como una especie de freno mecánico para este tipo de controladores.

En la Figura 3.7 se muestra que el control PD tiene una respuesta más lenta en comparación con el control PD-N, con lo cual el sobretiro es nulo para la articulación 2, sin embargo en las Figuras 3.8 y 3.9 se observa el incremento en el valor de sobretiro debido a que el sistema intenta responder de manera más rápida.

Los resultados obtenidos para el control PD-N como se observa en las Figuras 3.7, 3.8, 3.9 es la reducción casi en su totalidad del valor de sobretiro, esto debido a la generación de valores adecuados para las ganancias k_p y k_v ofrecidas a la salida de las



(a) Circular (b) Flor de ocho Pétalos (c) Articular

Figura 3.10: Trayectorias deseadas o ideales

redes neuronales de base radial después de su proceso de entrenamiento. Con lo anterior se observa que el control PD-N presenta un mejor desempeño comparado con el control PD clásico para el caso del problema de regulación.

3.3.2. Seguimiento de Trayectorias

Como ya se ha mencionado al inicio de este capítulo el control de posición en robots manipuladores permite que su efector final se pueda posicionar en un punto constante en el tiempo y permanecer ahí de forma indefinida. Para propósitos prácticos o industriales una vez que se alcanza el punto deseado o set-point deberán pasar uno o más períodos de muestreo para cambiar el valor de dicho punto deseado, de tal forma que el actual punto tomará el papel de condición inicial y el extremo final del robot se moverá al nuevo punto deseado y así sucesivamente. Este concepto da la posibilidad de interpolar curvas para que el robot pueda seguir trayectorias basándose en un esquema de control de posición con puntos cercanos entre sí, por lo cual en control automático se le denomina control punto a punto (Reyes, 2011).

A continuación se presentan los resultados de simulación para un esquema de control de posición punto a punto implementando tres algoritmos de control PD-N, PD y TANH (Reyes, 2012). Además, se plantea el seguimiento de las tres trayectorias que se muestran en la Figura 3.10. En la Tabla 3.2 se presentan las ecuaciones paramétricas para el seguimiento de las trayectorias deseadas.

Cabe mencionar que para realizar las simulaciones del seguimiento de estas trayectorias con el brazo robot y el control neuronal propuesto PD-N se utilizaron las ecuaciones (3.1), (3.2) y (3.3), en donde los valores para las ganancias proporcionales y derivativas son obtenidos de la Figura 3.6. Los parámetros de simulación y sintonización de ganancias para el control proporcional-derivativo (PD) y tangente hiperbólico (TANH) se presentan en la Tabla 3.3.

Con la finalidad de calcular el intervalo de muestreo mínimo para completar una vuelta alrededor de las trayectorias propuestas, se eligió que el tiempo de simulación fuera de 63 [seg], con lo cual el período de muestreo toma el valor de h=0.01 [seg], lo que corresponde a 6300 muestras.

Trayectorias	**Ecuaciones Paramétricas**
Circular	$x = 0{,}3 + 0{,}25cos(0{,}1t)$ $y = -0{,}3 + 0{,}25sin(0{,}1t)$
Flor de ocho pétalos	$x = 0{,}3 + 0{,}25cos(0{,}4t)cos(0{,}1t)$ $y = -0{,}3 + 0{,}25cos(0{,}4t)sin(0{,}1t)$
Articular	$q_{d1} = 0{,}7854[1 - e^{-2t^3}] + 0{,}3490[1 - e^{-2t^3}]sin(0{,}5t)$ $q_{d2} = 1{,}0472[1 - e^{-2t^3}] + 0{,}5236[1 - e^{-2t^3}]sin(0{,}375t)$

Tabla 3.2: Seguimiento de trayectorias parametrizadas

Articulación	**PD** $\tau_{PD} = kp\tilde{q} - kv\dot{q} + g(q)$	**Tanh** $\tau_{Tanh} =$ $kptanh\{\lambda\tilde{q}\} - kvtanh\{\gamma\dot{q}\} + g(q)$
$\tau_1^{max} = 150Nm$ (Hombro)	$kp_1 = 72\frac{Nm}{grados}$ $kv_1 = 0{,}25\frac{Nm-seg}{grados}$	$kp_1 = 100\ Nm$ $kv_1 = 100\ Nm$ $\lambda_1 = \frac{180}{\pi}$ $\gamma_1 = 1$
$\tau_2^{max} = 15Nm$ (Codo)	$kp_2 = 0{,}085\frac{Nm}{grados}$ $kv_2 = 0{,}25\frac{Nm-seg}{grados}$	$kp_2 = 100\ Nm$ $kv_2 = 100\ Nm$ $\lambda_2 = \frac{180}{\pi}$ $\gamma_2 = 1$

Tabla 3.3: Parámetros de sintonización en el control PD y TANH

La ecuación que describe las relaciones anteriores se presenta como:

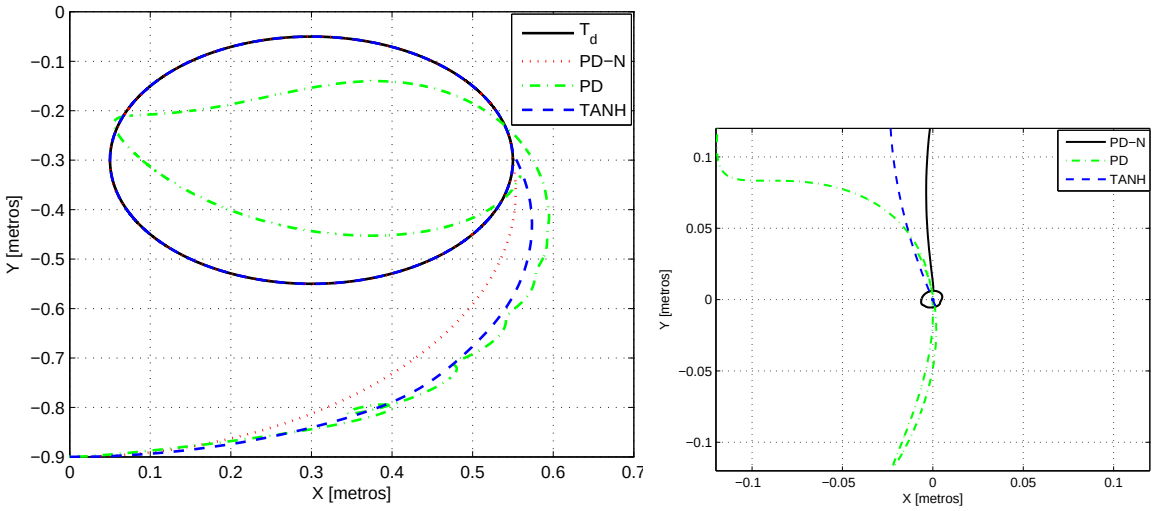
$$n = \frac{tf}{h} \tag{3.9}$$

donde n corresponde al número de puntos necesarios para describir correctamente la trayectoria y es definida por el usuario. El valor para el tiempo final t_f depende de la los ajustes en las ganancias de los controladores y de la respuesta del sistema. El parámetro h representa el valor para el intervalo de muestreo.

Resultados para la trayectoria circular

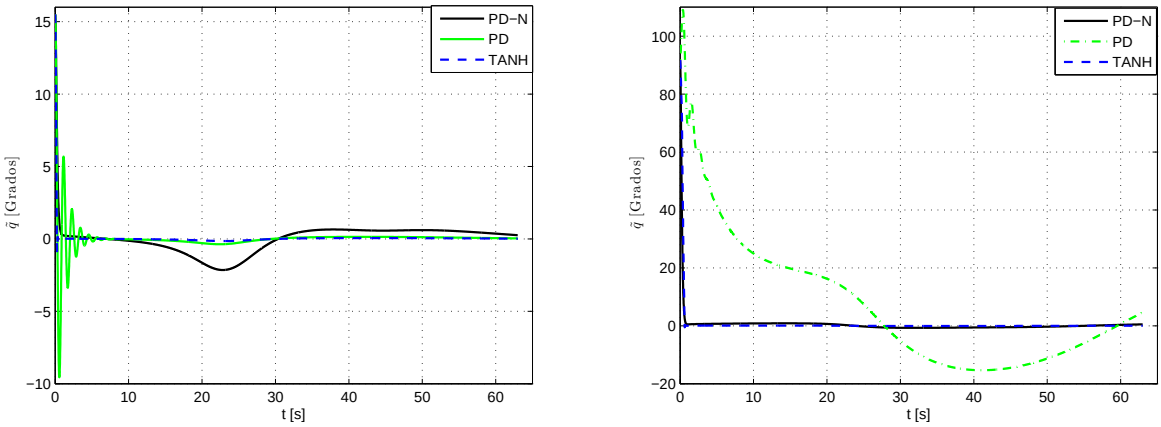
La Figura 3.11a muestra el seguimiento de la trayectoria circular y la respuesta del sistema modelado para los tres algoritmos de control. Se observa que el control PD no sigue de manera correcta la trayectoria deseada T_d, tal como lo hacen los controladores PD-N y TANH. En la Figura 3.11b se muestra la magnitud de los errores en el espacio cartesiano con lo cual se prueba que el control saturado TANH tiene un mejor desempeño comparado con el control PD y PD-N.

En la Figura 3.12 se presentan los errores articulares en la implementación de las tres leyes de control, observándose que el error para el control PD-N en la articulación 1 es mayor que la calculada en la articulación 2, esto debido a que la primera articulación u hombro del robot debe cargar a la segunda, con lo cual de acuerdo a su dinámica debe vencer la fuerza en la posición de reposo de ambos eslabones. Sin embargo, en



(a) Trayectoria circular (b) Error cartesiano

Figura 3.11: Seguimiento de una trayectoria circular

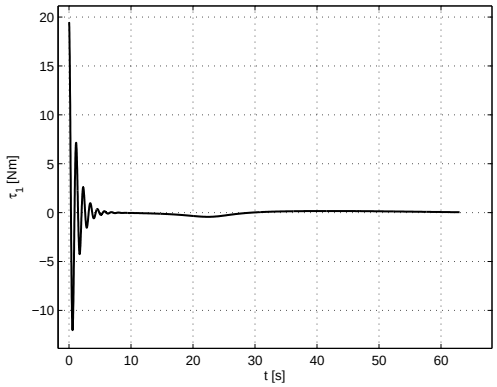


(a) Articulación 1 (b) Articulación 2

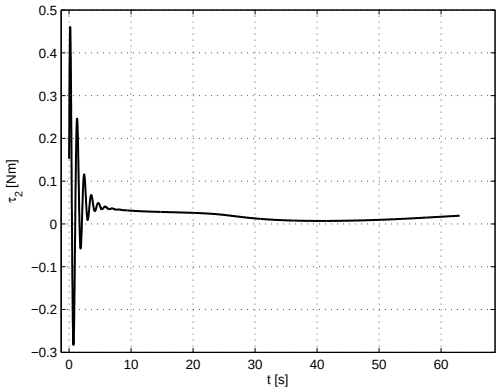
Figura 3.12: Errores articulares con las tres leyes de control

comparación con el control PD el control PD-N mantiene una mejora del 28.58 % en el error de seguimiento respecto a los valores calculados del error cuadrático medio (ECM).

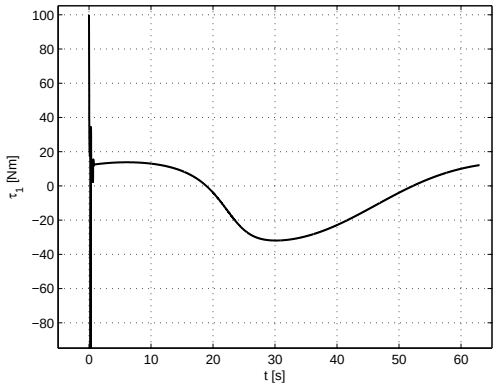
El comportamiento de los pares articulares de un manipulador de dos grados de libertad en el seguimiento de una trayectoria circular para los controladores PD, TANH y PD-N se presentan en la Figura 3.13. Como se muestra el control PD presenta una mayor variación (oscilaciones) en el valor de los torques en comparación con los otros dos controladores. Cabe señalar que tanto para el control PD y PD-N los torques se acotan por debajo de los límites de saturación de los motores ($\tau_1 = 150$ [N-m] y $\tau_2 = 15$ [N-m]), sin embargo, para el control TANH el torque correspondiente a la articulación 2 alcanza el límite de saturación de 15 [N-m].



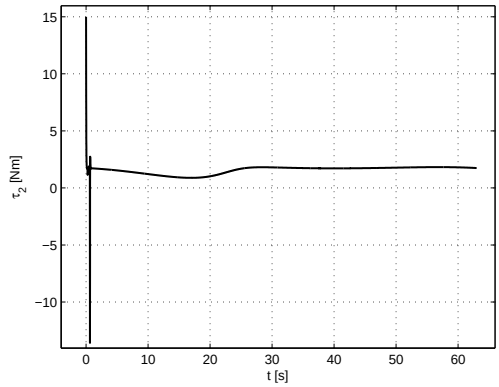
(a) Par articulación 1 -PD



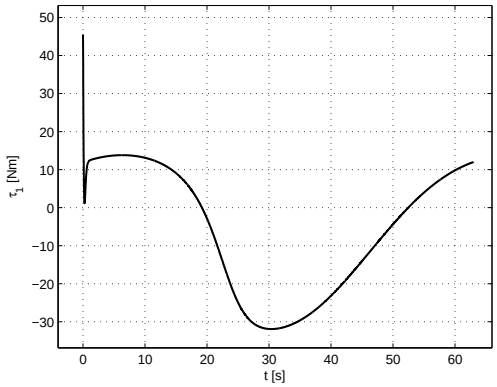
(b) Par articulación 2 -PD



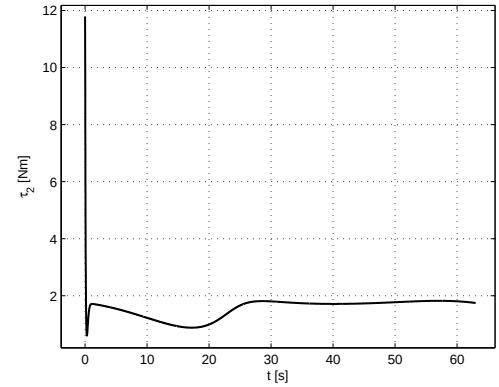
(c) Par articulación 1 -TANH



(d) Par articulación 2 -TANH



(e) Par articulación 1 PD-N

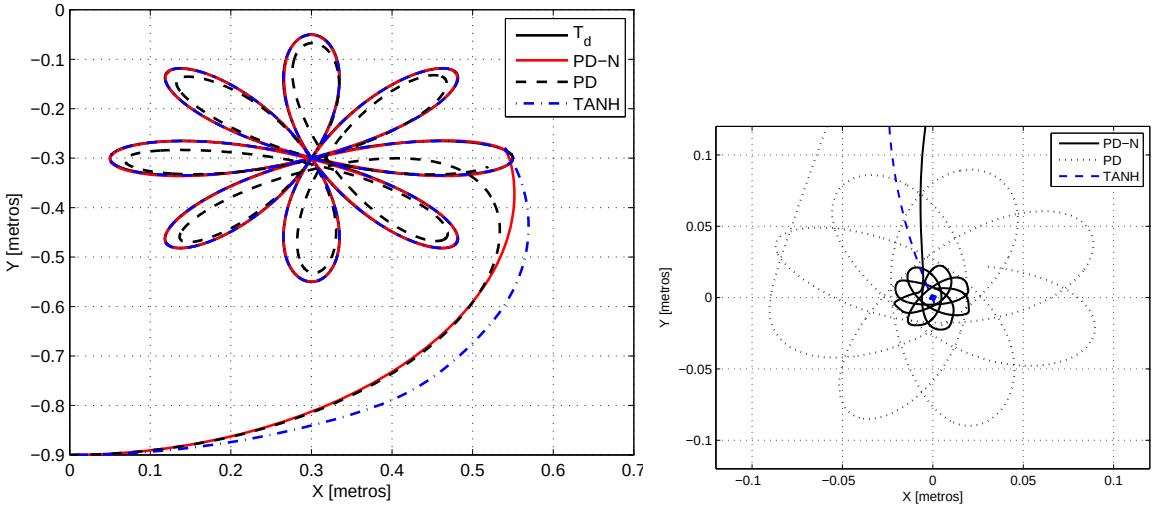


(f) par articulación 2 PD-N

Figura 3.13: Pares articulares con el control PD, TANH, PD-N T-Circular

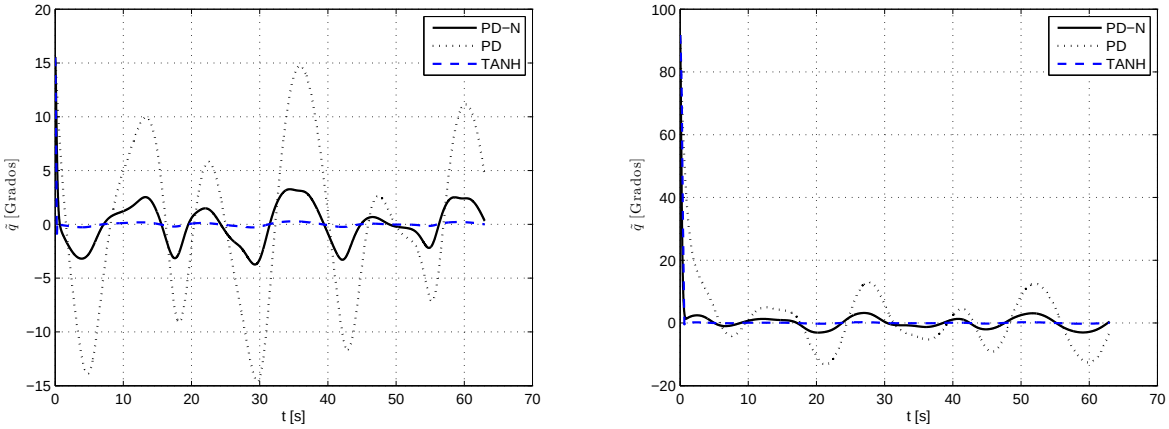
Trayectoria flor de ocho pétalos

En la Figura 3.14 se presenta el seguimiento de la trayectoria "flor de 8 pétalos" con los controladores PD-N, PD y TANH. Se observa que el control PD-N sigue la trayectoria deseada con un mejor desempeño que el control PD en los puntos donde la inercia del manipulador se incrementa, esto es, cuando la trayectoria cambia abruptamente de



(a) Flor de ocho pétalos (b) Error cartesiano

Figura 3.14: Seguimiento de la trayectoria flor de ocho pétalos

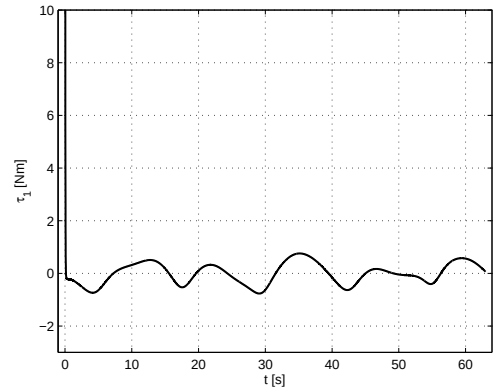


(a) Articulación 1 (b) Articulación 2

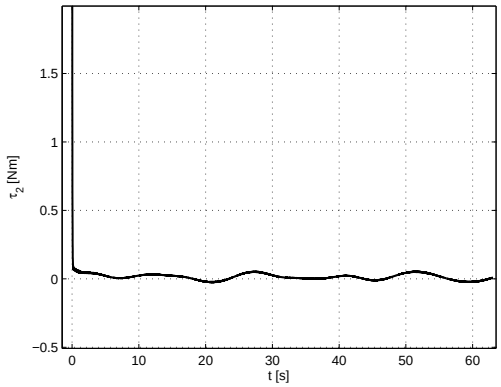
Figura 3.15: Errores articulares en el control PD-N, PD y Tanh T-Flor

dirección. Además como se muestra la Figura 3.14b el error cartesiano se encuentra acotado a una región cuadrada de 9 [cm], 2.5 [cm] y 0.06 [cm] de largo para el control PD, PD-N y TANH respectivamente. De acuerdo a lo anterior el control PD es el que mayor error contempla en el seguimiento de esta trayectoria. Además, es preciso notar que el control TANH es el que tiene mejor desempeño comparado con los otros dos, sin embargo este no contempla el tipo de ganancias variables tal como lo hace el controlador PD-N.

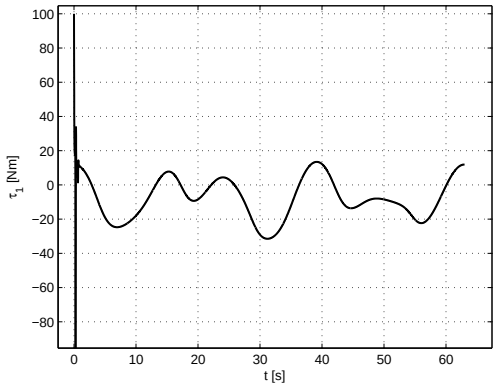
En la Figura 3.15 se observa el comportamiento del error articular para el primero y segundo grado de libertad del manipulador. En la gráfica el control PD mantiene un mayor error comparado con los controladores PD-N Y TANH, por lo que el mejor desempeño lo alcanza nuevamente el control TANH.



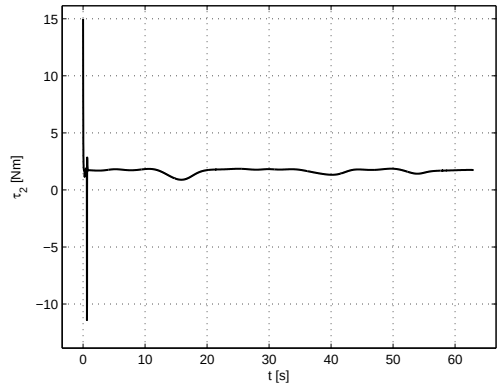
(a) Par articulación 1 -PD



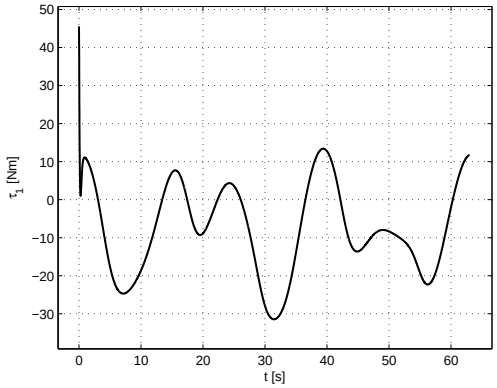
(b) Par articulación 2 -PD



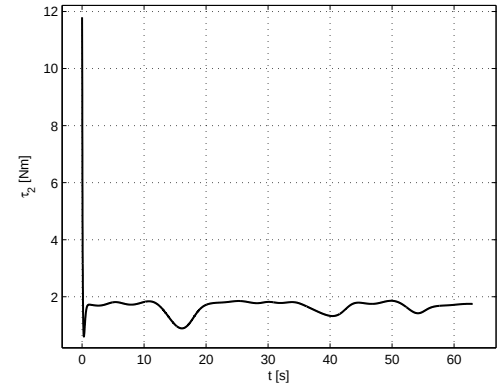
(c) Par articulación 1 -TANH



(d) Par articulación 2 -TANH



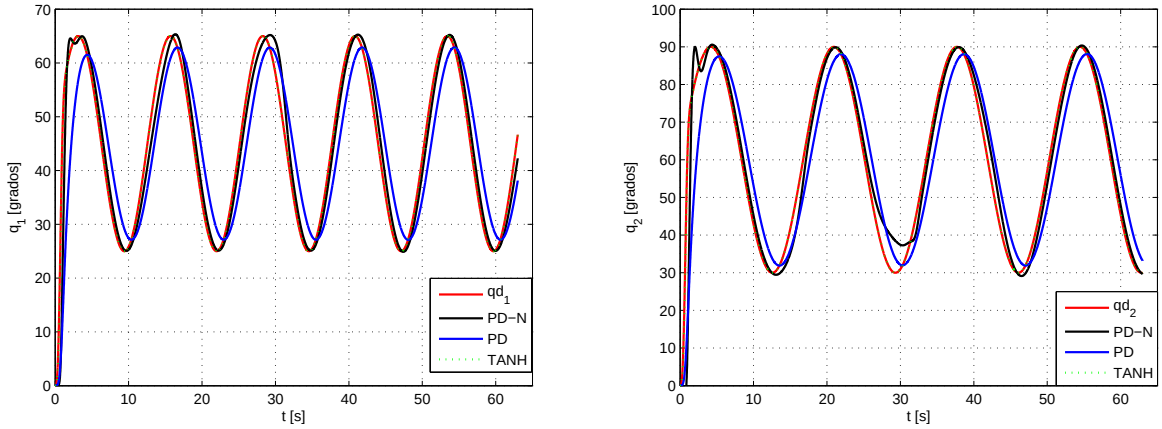
(e) Par articulación 1 PD-N



(f) Par articulación 2 PD-N

Figura 3.16: Pares articulares con el control PD, TANH, PD-N T-Flor

Los pares articulares para el seguimiento de la trayectoria de flor de ocho pétalos se presentan en la Figura 3.16, en donde para el caso del controlador TANH se alcanza el límite de saturación para la articulación 2, sin embargo este no es superado. Cabe señalar que el valor de par en la articulación 1 continúa siendo mayor que en la articulación 2, esto como ya se ha explicado antes se debe a la dinámica propia del manipulador.



(a) Trayectoria en articulación 1

(b) Trayectoria en articulación 2

Figura 3.17: Trayectorias de referencia paramétricas

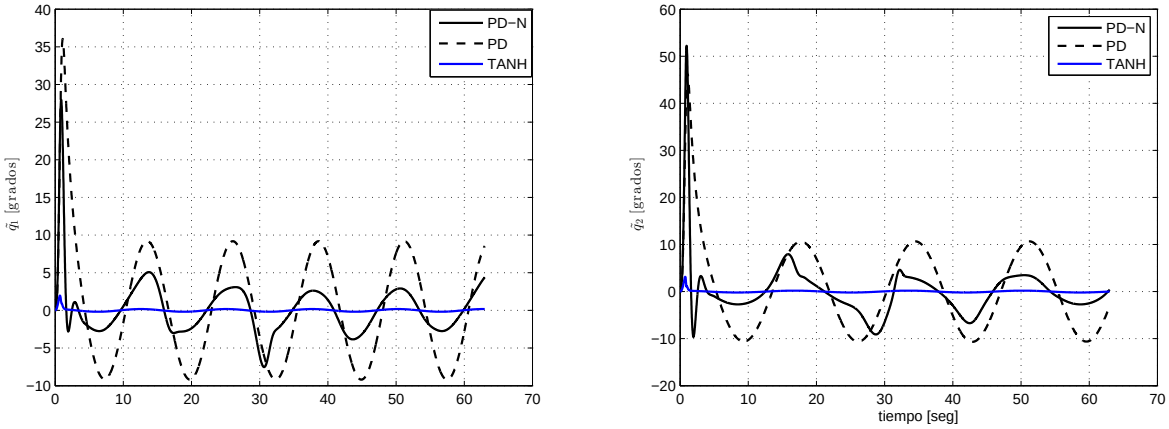
Trayectoria de referencia en el espacio articular

La trayectoria que se presenta a continuación denominada como de "referencia" se ha implementado directamente en el espacio articular por lo que no se necesita el uso de la cinemática inversa tal como fue el caso en las dos trayectorias anteriores circular y flor de ocho pétalos. Cabe mencionar que esta trayectoria ha sido utilizada como una referencia para diversos trabajos, tanto para el control de posición como de trayectoria (Kelly et al., 2005, Salas et al., 2012), esto se debe a que el valor de sus parámetros son conocidos, es decir que se encuentra parametrizada.

En la Figura 3.17 se presenta el comportamiento de la trayectoria de referencia en el espacio articular, para el manipulador de dos grados de libertad con las tres leyes de control PD-N, PD y TANH. Se observa que con el control PD-N se tienen "pequeños" sobretiros al inicio de la trayectoria, ya que se necesita de un mayor torque para poder vencer la inercia que se tiene en la posición de reposo. El control PD no presenta sobretiros, sin embargo su respuesta se encuentra desfasada considerablemente de la trayectoria de referencia por lo que el error de seguimiento se incrementa. Nuevamente el control saturado TANH es el que pasa el mayor tiempo de simulación (60 [seg]) sobre la trayectoria deseada.

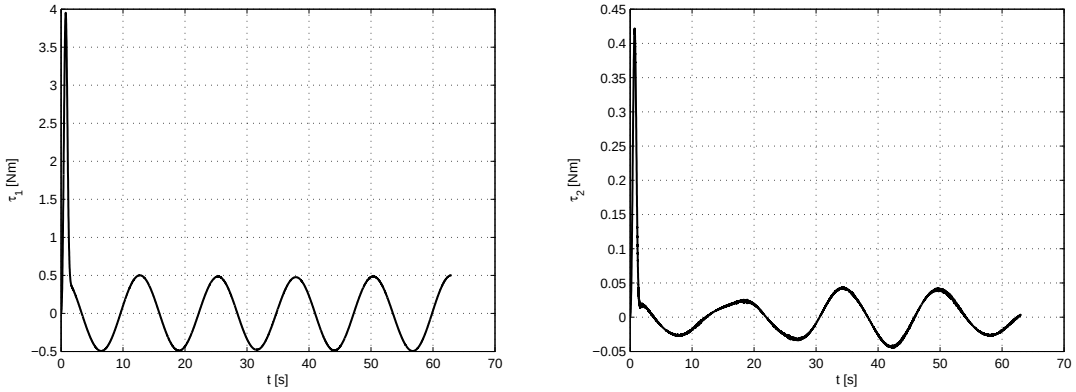
Los errores articulares se presentan en la Figura 3.18, donde se observa claramente mejor desempeño del controlador TANH en el seguimiento de la trayectoria deseada con un error que se aproxima a cero. Cabe mencionar que el control PD-N mejora el error por encima del 60 % en el control de trayectoria comparado con el control PD, respecto del cálculo realizado para el error cuadrático medio.

En la Figura 3.19 se muestran los pares articulares para cada una de las articulaciones del robot manipulador tanto para el control PD como para el TANH. Los valores de torque para ambos controladores no superan los limites de saturación de los actuadores alcanzando sólo un máximo de 12 [N-m] para la articulación 1 con el control TANH.

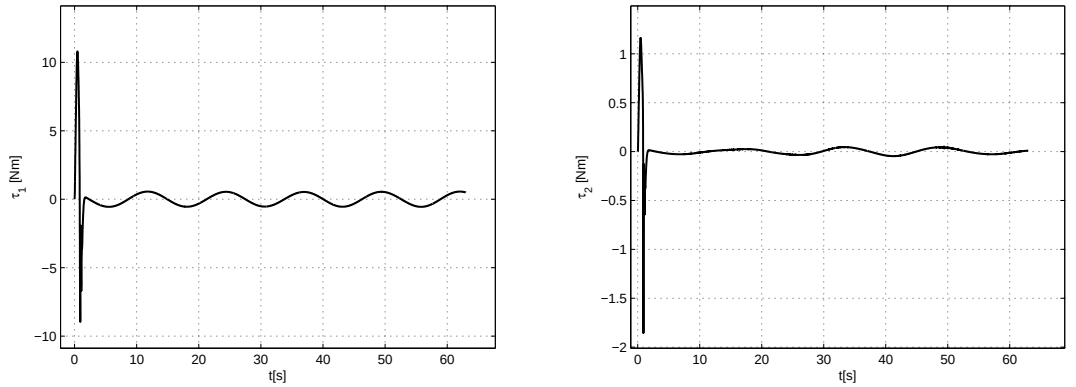


(a) Articulación 1 (b) Articulación 2

Figura 3.18: Gráfica de errores en la trayectoria de referencia articular



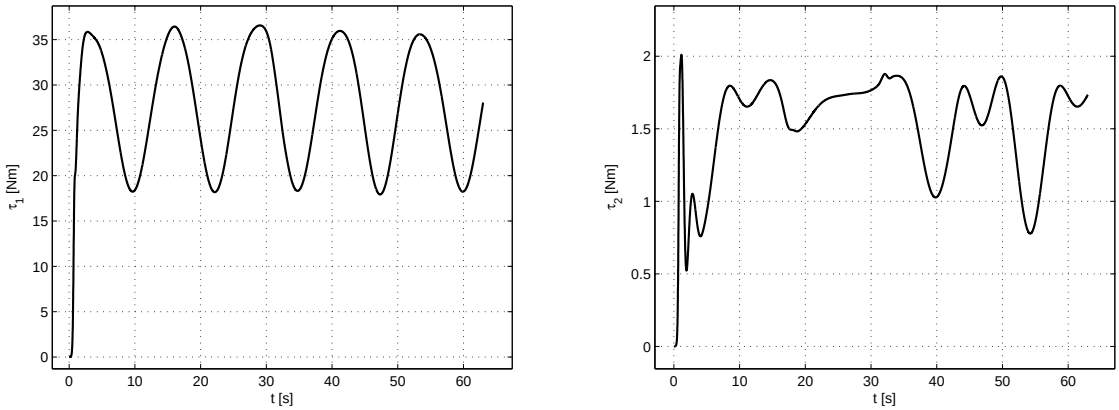
(a) Par en articulación 1 -PD (b) Par en articulación 2 -PD



(c) Par en articulación 1 -TANH (d) Par en articulación 2 -TANH

Figura 3.19: Pares articulares con el control PD y TANH

Para el caso del controlador PD-N el comportamiento de los pares articulares para las dos articulaciones del robot se presentan en la Figura 3.20. En la primera articulación,



(a) Par en articulación 1 PD-N (b) Par en articulación 2 PD-N

Figura 3.20: Pares articulares con el control PD-N

se observa que existe una oscilación de los valores de torque de 18 a 37 [N-m] y para la segunda articulación de 0.5 a 2 [N-m]. Como se muestra los torques se mantienen por debajo de los 37 y 2 [N-m], esto se debe a la metodología que se siguió para realizar el entrenamiento de las redes neuronales con lo cual se garantiza que no exista saturación a los valores máximos de torque permitido por los actuadores.

Índice de desempeño

Un criterio académico aceptado por la comunidad científica de control y robótica para medir el desempeño de un algoritmo de control es la referida al cálculo del índice de desempeño determinado por la norma $\mathcal{L}_2$, el cual se refiere a la exactitud que debe tener la respuesta del robot manipulador. Una explicación más detallada del procedimiento matemático que se utiliza para implementar este índice se presenta en el apéndice A.

Los aspectos cualitativos de la estrategia de control son factores importantes que determinan este desempeño, como ejemplos se puede mencionar a las propiedades del gradiente de la energía potencial artificial en el cual se permita exhibir un corto transitorio, sobre impulsos y vibraciones mecánicas atenuadas a través de la función derivativa o amortiguamiento, facilidad en la sintonía de las ganancias proporcional y derivativa, robustez frente a incertidumbre paramétrica estas son algunas de las cualidades del esquema de control que se son reflejadas en el índice de desempeño (Reyes, 2011).

En la Figura 3.21 se presenta una gráfica de barras en donde se muestra el valor calculado para el índice de desempeño de las leyes de control PD-N, PD y TANH en el seguimiento de una trayectoria circular. Los resultados fueron 4.2113, 1.0611 y 0.8434 [grados] para los controladores PD, PD-N y TANH respectivamente. Se puede observar que el control PD-N tiene una mejora en el desempeño del 74.81 % y el control TANH del 79.97 % con respecto al controlador PD.

El valor calculado para el desempeño de la trayectoria flor de ocho pétalos se presenta en la Figura 3.22, los resultados obtenidos fueron de 2.3887, 0.9401 y 0.1203 [grados]

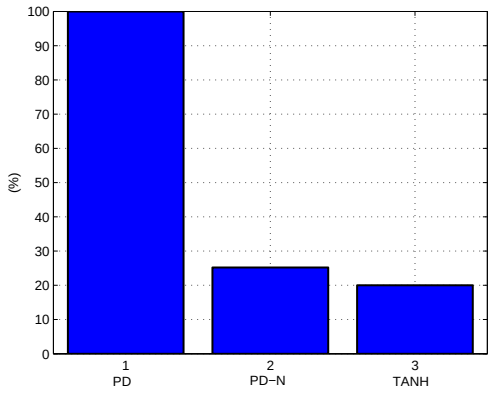


Figura 3.21: Índice de desempeño para la trayectoria circular

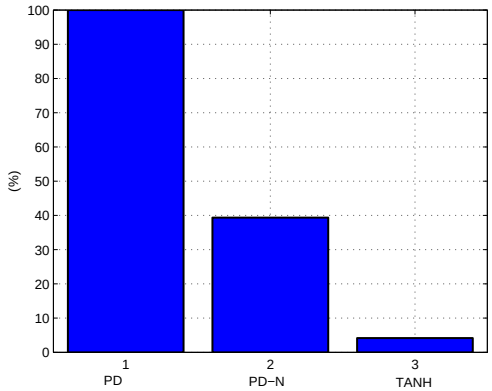


Figura 3.22: Índice de desempeño para la trayectoria flor

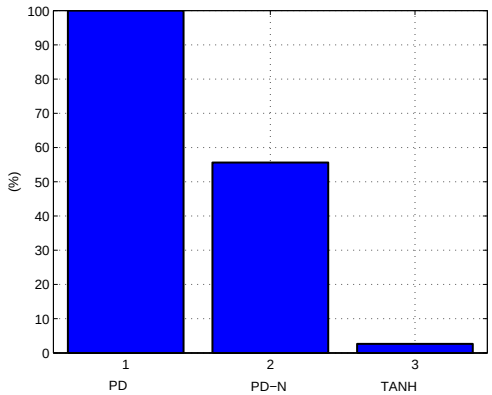


Figura 3.23: Índice de desempeño para la trayectoria articular

para los controladores PD, PD-N y TANH respectivamente. La mejora del controlador PD-N y el control TANH en comparación con el control PD fue del 60.65 % y del 94.96 % respectivamente para ambas leyes de control.

Para la trayectoria denominada como “articular” los resultados para el índice de desempeño fueron 2.1737 [grados] para el control PD, 1.2086 [grados] para el control PD-N

y 0.0575 [grados] en el control TANH, los cuales se presentan en la Figura 3.23. Los valores anteriores de desempeño equivalen a una mejora del 44.39 % y 97.35 % para los controladores PD-N y TANH respectivamente.

Es importante mencionar que debido a que el índice de desempeño representa a la suma de los errores articulares, a menor valor porcentual se considera un mejor desempeño por parte del controlador. Además, de acuerdo a los resultados obtenidos de la norma $\mathcal{L}_2$, en la medida en que las trayectorias deseadas son más complejas el control saturado TANH obtuvo un desempeño superior comparado con los esquemas PD y PD-N.

Capítulo 4

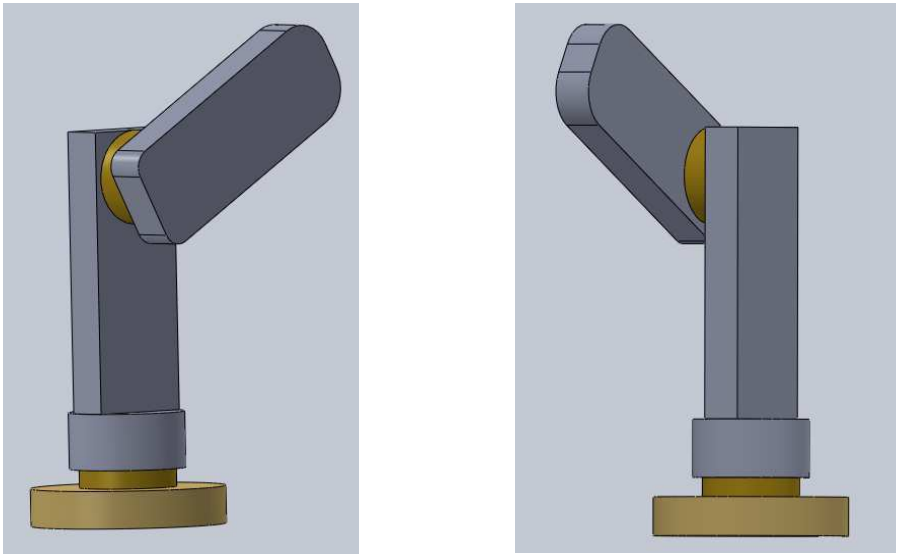
Construcción e implementación del control PD en el prototipo

En este capítulo se construye y fabrica un prototipo de robot manipulador de dos grados de libertad al cual se le incorpora un algoritmo de control PD a los actuadores mediante una tarjeta de desarrollo Freescale FRDM-KL25Z. A partir de este prototipo constituido tanto por la estructura mecánica como por la tarjeta, se valida experimentalmente el algoritmo de control neuronal PD-N presentado en el capítulo 3.

4.1. Selección del prototipo

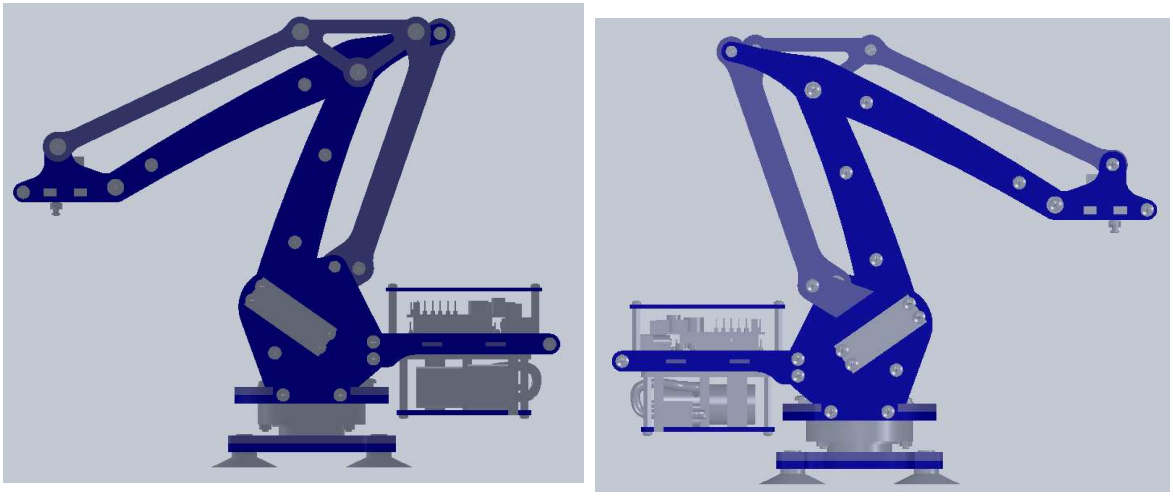
Se realizo una primera versión del robot manipulador mostrado en la Figura 4.1, en la cual se observa la vista frontal y trasera de un robot que consta de dos eslabones inter-conectados por medio de articulaciones. En estas dos articulaciones se instalan motores de transmisión directa que dan movimiento rotacional a la base del robot y entre eslabones. Sin embargo a medida que el diseño se vuelve más complejo se observó que el consumo de tiempo aumentaba, es por esta razón que se decidió proponer la construcción de un prototipo con un diseño ya elaborado.

En la Figura 4.2 se muestran las vistas lateral izquierda y derecha de un robot manipulador diseñado por la empresa Ufactory (EvolDesign, 2015), la cual fue creada a finales del 2014 y proporciona acceso libre a los diseños y planos de prototipos robóticos. La selección del diseño se llevo acabo considerando la facilidad de fabricación de las piezas y la sencillez con la que se realiza el montaje. Además dicho diseño fue creado en base a un robot industrial comercial que corresponde al modelo IRB 460 de la marca ABB, este robot es utilizado principalmente como paletizador de alta velocidad y algunas de sus características son su alta capacidad de carga y su rapidez de movimiento (ABB, 2015).



(a) Vista frontal (b) Vista trasera

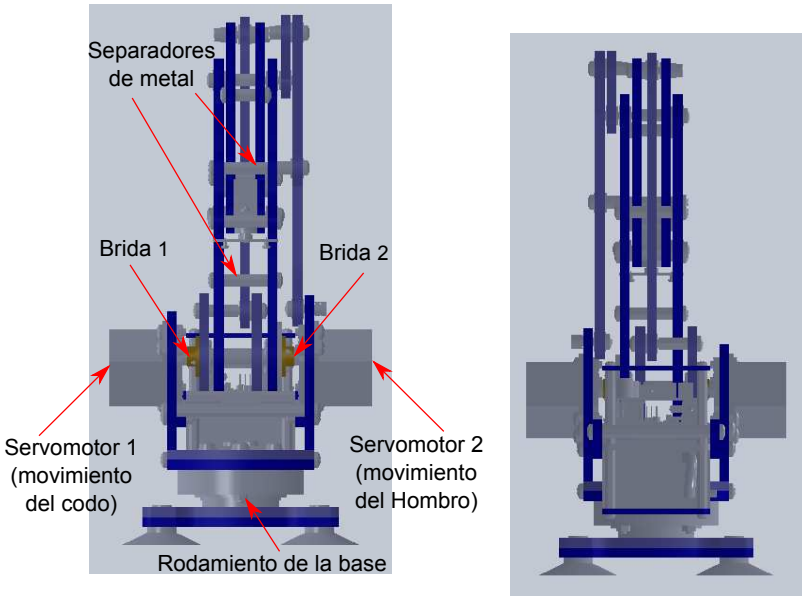
Figura 4.1: Primera versión del robot con dos grados de libertad



(a) Vista lateral izquierda (b) Vista lateral derecha

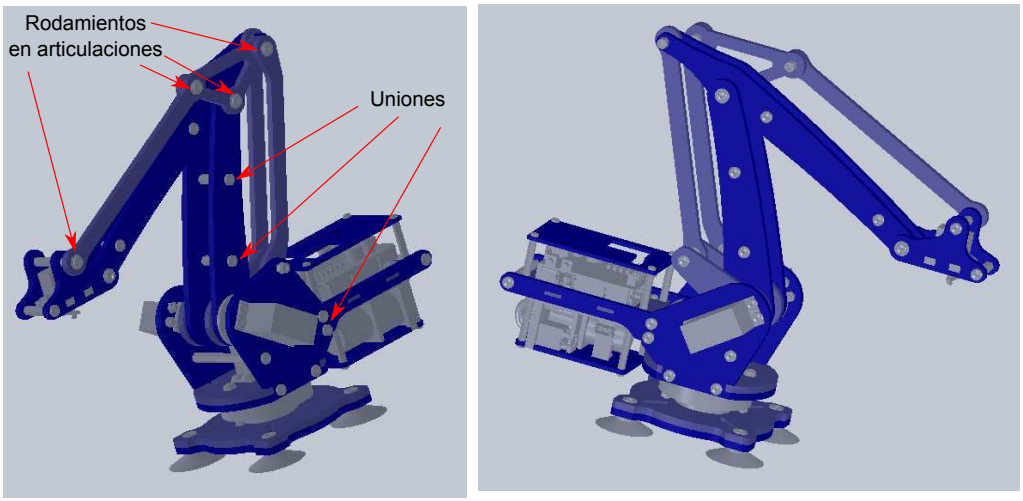
Figura 4.2: Vistas laterales del diseño de acceso libre

En la Figura 4.3a se muestran en color amarillo las bridas de acoplamiento para los servomotores. La estructura es compacta y los eslabones están unidos a través de separadores de metal. En las vistas presentadas en la Figura 4.3 se muestra la colocación de los servomotores en los costados de la estructura, con lo cual se intenta depositar el mayor peso posible en la base. Las uniones y articulaciones del robot (Figura 4.4), están diseñadas para proporcionar solidez y estabilidad a la estructura. Para reducir el fenómeno de fricción se instalan rodamientos tanto en la base como en las articulaciones.



(a) Vista frontal (b) Vista trasera

Figura 4.3: Vistas frontal y lateral del diseño de acceso libre



(a) Izquierda (b) Derecha

Figura 4.4: Vistas en perspectiva del diseño de acceso libre

4.2. Proceso de construcción y fabricación

Para llevar a cabo el proceso de construcción del manipulador presentado en la Sección 4.1, se guardó cada una de las piezas del ensamble que forman al robot con la extensión *.iges*. Este tipo de archivos permiten compatibilidad entre diversos programas de diseño asistido por computadora (CAD) y manufactura (CAM). La fabricación de las piezas se realizó con una máquina de control numérico o CNC. Los archivos en código *G*

Figura 4.5: Piezas para ensamble del robot manipulador

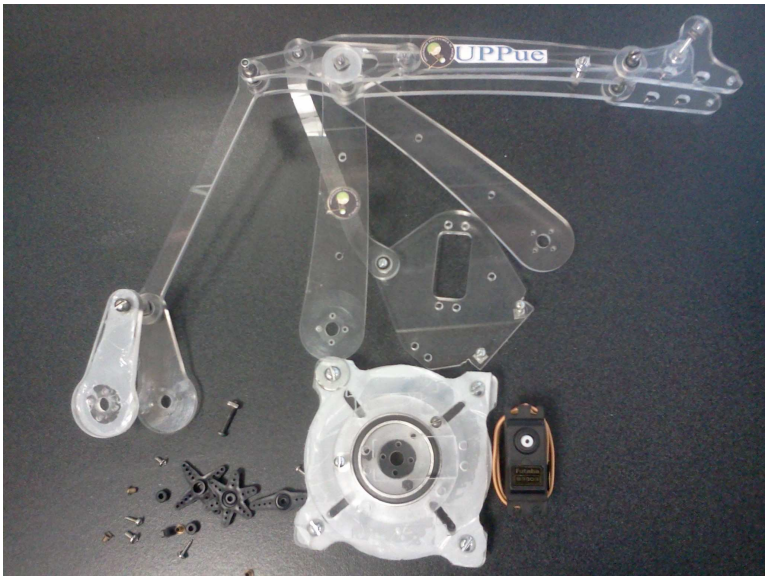


Figura 4.6: Montaje de la parte superior del prototipo

se obtuvieron a partir del programa Mastercam (CNC Software, 2015) y Rhinoceros (Robert McNeel, 2014). En la Figura 4.5 se muestran las piezas utilizadas para el montaje del prototipo robótico, herramientas y tornillería. Cabe mencionar que en el diseño original las dimensiones están dadas en unidades del sistema internacional [mm], sin embargo, para la construcción y montaje tanto de los barrenos como de los tornillos las medidas fueron cambiadas al sistema inglés [pulgadas]. Este cambio se debe a la dificultad de encontrar tornillos pequeños [m3] en comparación con el bajo costo y disponibilidad que presentan las medidas estándar [1/8 in].

El montaje de los eslabones para el primer y segundo grado de libertad del robot se muestran en la Figura 4.6. El material utilizado es acrílico transparente el cual presenta alta flexibilidad antes de la ruptura y es fácil de maquinar en comparación con algún tipo de acero. Sin embargo el peso que puede soportar la estructura se encuentra limitado.

(a) Rodamiento de la base (b) Servomotor instalado en la base

Figura 4.7: Montaje de la base e instalación del servomotor

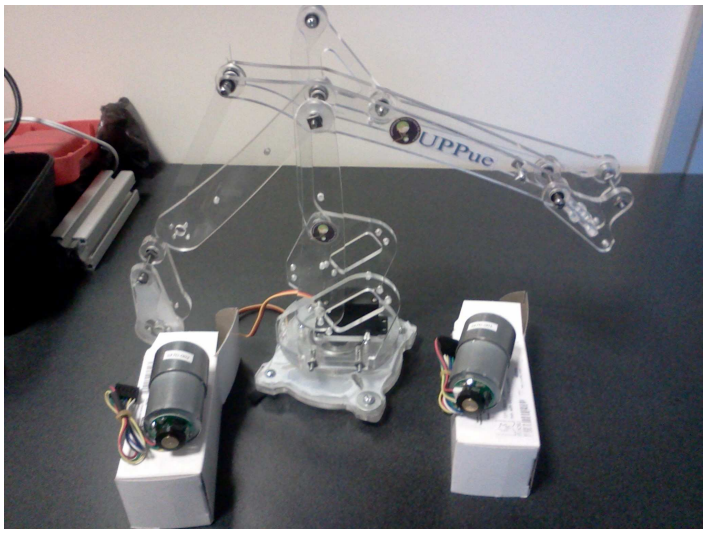


Figura 4.8: Motorreductores para el robot con dos grados de libertad

El armado de la base e instalación del servomotor se presentan en la Figura 4.7. Además, en esta figura también se muestra la colocación del rodamiento cuya finalidad es evitar fricción en el momento de girar y facilitar la aplicación de torque por parte del servomotor.

En la Figura 4.8 se muestran los motorreductores con modelo $37Dx57L$ de la empresa *Pololu* (Pololu Corporation, 2001-2015), los cuales van a ser instalados en la base del robot con el objetivo de mover los dos grados de libertad del manipulador. La selección de estos motores se realizó en función del máximo torque entregado a la salida del sistema de engranaje a rotor bloqueado como se muestra en la Figura 4.9. Este tipo de motorreductores de corriente directa (DC) con escobillas se alimenta con un voltaje de 12V y tiene una relación de transmisión 131.25:1, además de tener integrado un encoder en cuadratura que brinda una resolución de 64 cuentas por revolución (CPR) en el eje del motor, lo cual corresponde a 8400 cuentas por revolución en el eje a la salida de la caja de engranaje. Sus dimensiones se presentan en la Figura 4.10, algunos de sus

Relación de transmisión	*Velocidad sin carga a 12 V*	*Torque a rotor bloqueado a 12 V*	*Corriente a rotor bloqueado a 12 V*	*Motor con encoder*	*Motor sin encoder*
1:1	11,000 RPM	5 oz-in	5 A	Motor sin caja de reducción	
19:1	500 RPM	84 oz-in	5 A	37Dx52L mm	37Dx52L mm
30:1	350 RPM	110 oz-in	5 A	37Dx52L mm	37Dx52L mm
50:1	200 RPM	170 oz-in	5 A	37Dx54L mm	37Dx54L mm
70:1	150 RPM	200 oz-in	5 A	37Dx54L mm	37Dx54L mm
100:1	100 RPM	220 oz-in	5 A	37Dx57L mm	37Dx57L mm
131:1	80 RPM	250 oz-in	5 A	37Dx57L mm	37Dx57L mm

Figura 4.9: Criterios de selección

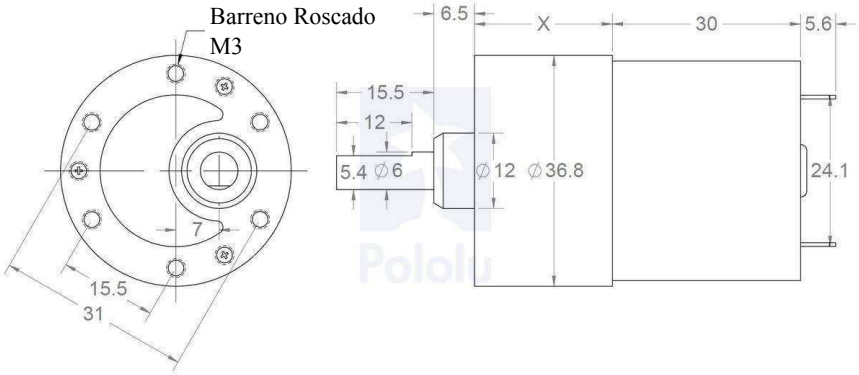


Figura 4.10: Dimensiones en mm del motorreductor 37D mm

aspectos importantes son su velocidad maxima a 12V de 80 RPM, con un consumo de 300mA y una capacidad de torque de 18 kg-cm con una corriente de 5A.

En la Figura 4.11 se muestra la colocación de las bridas a los ejes de los motorreductores con la finalidad de transmitir el movimiento hacia los eslabones del robot. Las bridas fabricadas se presentan en la Figura 4.12, las cuales fueron elaboradas con material de aluminio. Cabe señalar que esta parte fue una de las más difíciles de resolver ya que en el diseño original se utilizan servomotores donde solo se puede implementar control de lazo abierto, sin embargo, para lograr el objetivo de control en lazo cerrado se cambio a el uso de motorreductores.
El montaje completo del robot manipulador que consta de un servomotor para el movimiento de la base, y de dos motorreductores para la transmisión de movimiento al primero y segundo grado de libertad se presenta en la Figura 4.13.

(a) Brida acoplada al eje (b) Instalación de los actuadores

Figura 4.11: Montaje de los actuadores en las 3 articulaciones del robot

(a) Articulación 1 (b) Articulación 2

Figura 4.12: Bridas de Sujeción para los motores DC

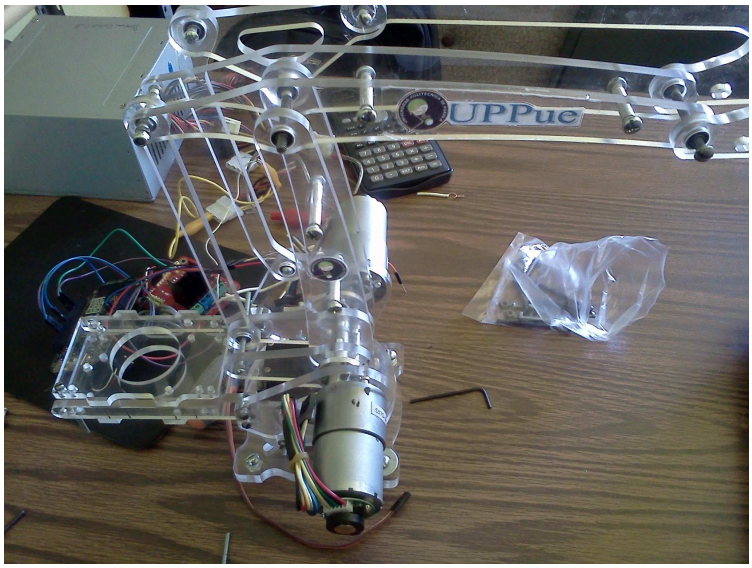


Figura 4.13: Arquitectura final del robot

4.3. Implementación del control de posición PD en los actuadores

Para la implementación del control PD en los actuadores es necesario realizar la lectura y medición de la posición en el eje del motorreductor. El motorreductor presentado en la Sección 4.2 cuenta con un encoder incremental con una resolución de 64 cuentas por revolución, lo que corresponde a 8400 cuentas por revolución a la salida del sistema de engranaje. El cálculo de la velocidad se realiza implementando el algoritmo de Euler. Con los parámetros de posición y velocidad es suficiente para implementar el control PD en el prototipo didáctico.

Tarjeta de control

En la Figura 4.14 se presenta una descripción general de la tarjeta de desarrollo Freescale FDRM-KL25Z, la cual se ha seleccionado para implementar el control PD en los motorreductores para el movimiento del prototipo. Además, se muestran las terminales utilizadas como puertos de entrada (Canales A,B del encoder) y como puertos de salida las direcciones para los motores (Dir) y las señales PWM. Esta tarjeta es una plataforma de bajo costo integrada por una familia de procesadores Kinetics L serie KL basado en el procesador ARM® Cortex™-M0+ y que ha sido diseñada por la empresa Freescale en colaboración con la plataforma *mbed* (Arm Ltd, 2014).

Cabe mencionar que *mbed* permite un rápido diseño de prototipos y de la experimentación correspondiente con microcontroladores ARM Cortex-M3 y ARM Cortex-M0 de 32 bits. Provee a los desarrolladores experimentados una plataforma productiva para realizar pruebas conceptuales y prototipos, mientras que a los principiantes sin experiencia previa les provee una forma accesible de realizar proyectos con microcontroladores de 32 bits mediante el acceso a las librerías, tutoriales y ejemplos, además de la comunidad online de mbed.

Algunas de sus principales características incluyen fácil acceso a los puertos de entrada y salida (I/O) del procesador tanto analógicas como digitales, salidas PWM, DAC para funcionar como tarjeta de adquisición y el uso de protocolos de comunicación I^2C. Su construcción tal como se muestra en la Figura 4.15 facilita el uso de interfaces para su expansión y posee una interfaz integrada de depuración para la programación de la flash y de control de gestión directamente desde la computadora. Cabe mencionar que su funcionamiento a baja energía permite la instalación de baterías. Características adicionales de la placa Freescale se enumeran a continuación (Arm Ltd, 2014):

1.-Procesador MKL25Z128VLK4 MCU – 48 MHz, 128 KB flash, 16 KB SRAM, USB OTG (FS), 80LQFP
2.-Alimentación 5V USB o 4.5-9V
3.-Conector mini USB tipo B con función de USB-host.
4.-Open SDA.
5.-Sensor capacitivo integrado.
6.-Acelerómetro MMA8451Q integrado.
7.-LED RGB integrado.
8.-Opciones de alimentación flexibles – USB, batería, fuente externa.

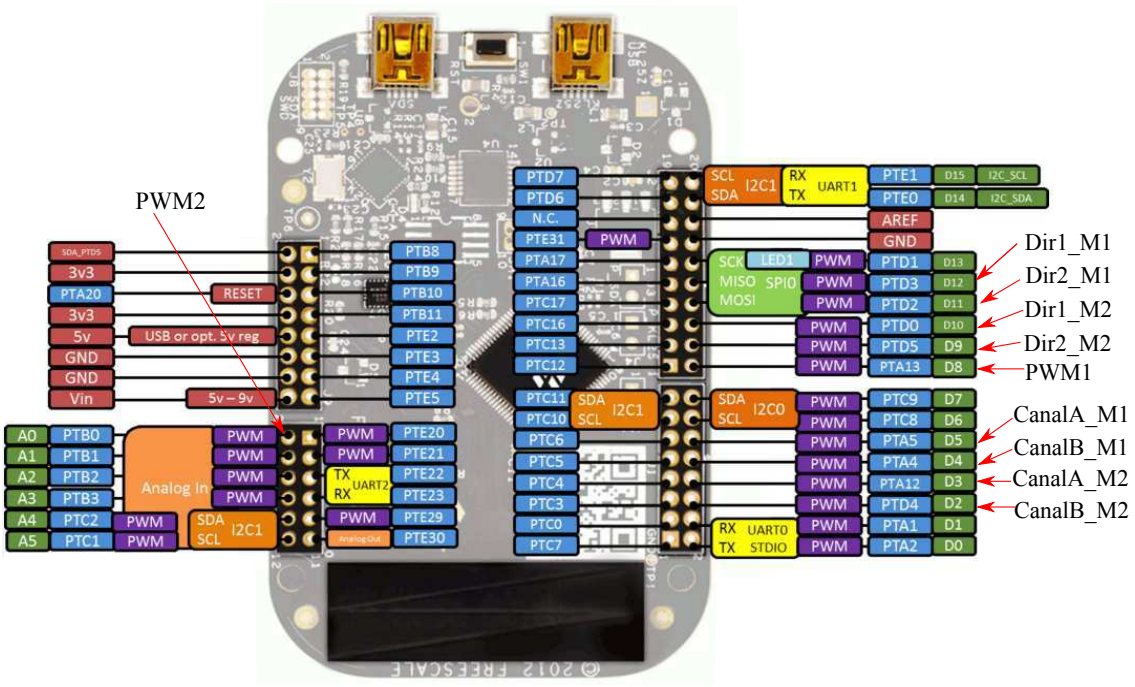


Figura 4.14: Diagrama general de los puertos y conexiones de la Tarjeta Freescale

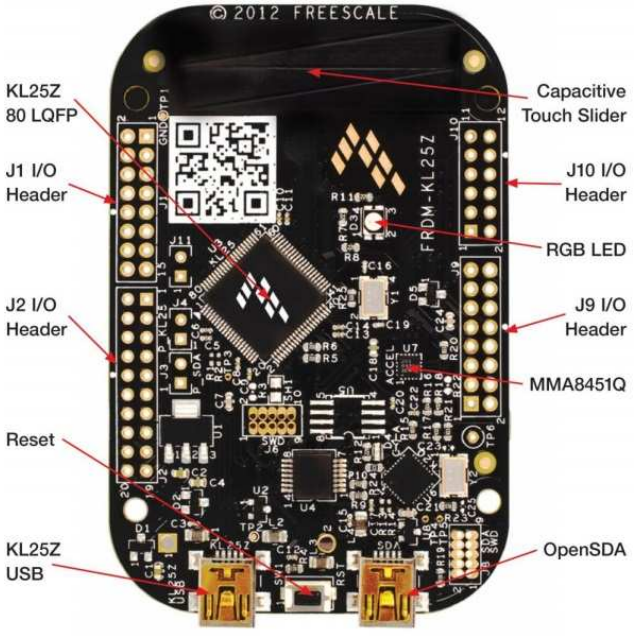


Figura 4.15: Tarjeta de desarrollo Freescale

9.-Fácil acceso a los puertos I / O del procesador a través de los conectores compatibles con el Arduino UNO R3.
10.-Interfaz de depuración programable OpenSDA con múltiples aplicaciones disponibles.
11.-Interfaz de programación de la flash del dispositivo de almacenamiento masivo.
12.-Interfaz de depuración P & E que provee control de ejecución de depuración y com-

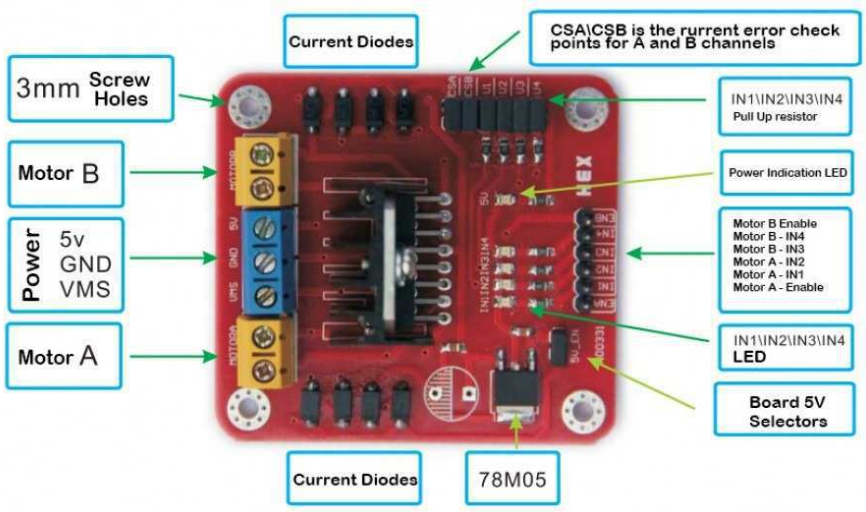


Figura 4.16: Tarjeta de potencia

patibilidad con herramientas IDE.
13.-Interfaz CMSIS-DAP.
14.-Aplicación de registro de datos.
15.-Mbed compatible.

Tarjeta de potencia

La tarjeta de potencia utilizada es la que se muestra en la Figura 4.16. Cuenta con un integrado L298N de doble puente H completo, diseñada para corrientes altas y voltaje para el control de cargas inductivas. Permite controlar hasta dos motores de corriente continua o un motor paso a paso bipolar, también permite controlar un motor paso a paso unipolar configurado como bipolar de forma muy sencilla y eficaz. El módulo puede controlar el sentido de giro y velocidad mediante señales TTL que se pueden obtener de microcontroladores y tarjetas de desarrollo como Arduino, Raspberry Pi y Lauchpads de Texas Instruments.

Algunas de sus características son el contar con un regulador de voltaje a 5V implementado con el integrado LM7805 para alimentación de la parte lógica, contiene dos entradas de habilitación pwm para el control de velocidad de los motores, el integrado posee un disipador de calor, el voltaje de alimentación puede ser hasta 46V y una corriente total de hasta 2A. Además, contiene diodos de protección y conectores tipo bloque de terminales para comodidad de conexión. Este módulo es útil para proyectos de robótica, para manejo de motores paso a paso, motores DC, relés, solenoides y Router CNC.

Conexiones del encoder

La medición de la posición es necesaria para calcular el error de posición $\tilde{q} = q_d - q$ y la velocidad,

$$\dot{q} = \frac{q_{actual} - q_{anterior}}{h} \tag{4.1}$$

donde h es el período de muestreo, para nuestra aplicación fue de 90 milisegundos. La

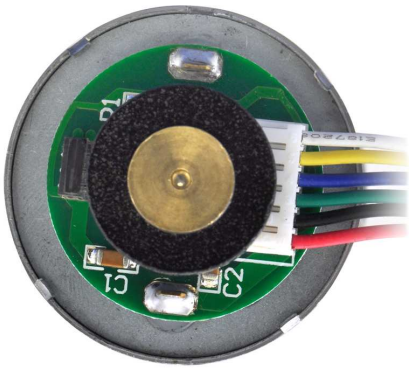


Figura 4.17: Encoder de efecto Hall con 32 CPR por canal

Color	Función
Rojo	1ra Terminal de conexión para potencia del motor
Negro	2da Terminal de conexión para potencia del motor
Verde	Terminal GND del encoder
Azul	Terminal Vcc del encoder (3.5 - 20 V)
Amarillo	salida del encoder (Canal A)
Blanco	salida del encoder (Canal B)

Tabla 4.1: Parámetros de conexión para el motor y encoder

posición y velocidad son utilizadas para implementar la ley de control de posición PD expresado como $\tau = kp\tilde{q} - kv\dot{q} + g(q)$ en la tarjeta electrónica. Para lograr este objetivo se utilizó el encoder incremental de efecto Hall mostrado en la Figura 4.17, el cual consta de dos canales para sensar la rotación del disco magnético. Si se desea utilizar el encoder en cuadratura se debe considerar que la resolución brindada por cada canal es de 32 cuentas por revolución (CPR). Por lo tanto, para calcular las CPR a salida de la caja de engranaje se debe multiplicar por un factor de 64 (Pololu Corporation, 2001-2015). En la Tabla 4.1 se presentan las especificaciones para la conexión de las líneas del encoder. Se puede observar que la alimentación se limita a 20V y los colores para las salidas del canal A, B son el amarillo y blanco, respectivamente.

Diagrama de Flujo

Con la finalidad de realizar la programación del control PD en la tarjeta Freescale FDRM-KL25Z bajo la plataforma de programación *mbed* se realizó el diagrama de flujo presentado en la Figura 4.18. En este diagrama se presenta el proceso de implementación para el control PD. Como se observa las variables se deben inicializar a cero y para calcular la velocidad en el eje del motor se generan interrupciones con un período de muestreo definido, de tal forma que se obtenga la diferencia entre la posición actual y la posición anterior. Es importante mencionar que los niveles de *pwm* se deben ajustar a los valores de torque calculados por el programa, de tal forma que las ganancias proporcional y derivativa se mantengan en un rango de variación. Este ajuste se debe

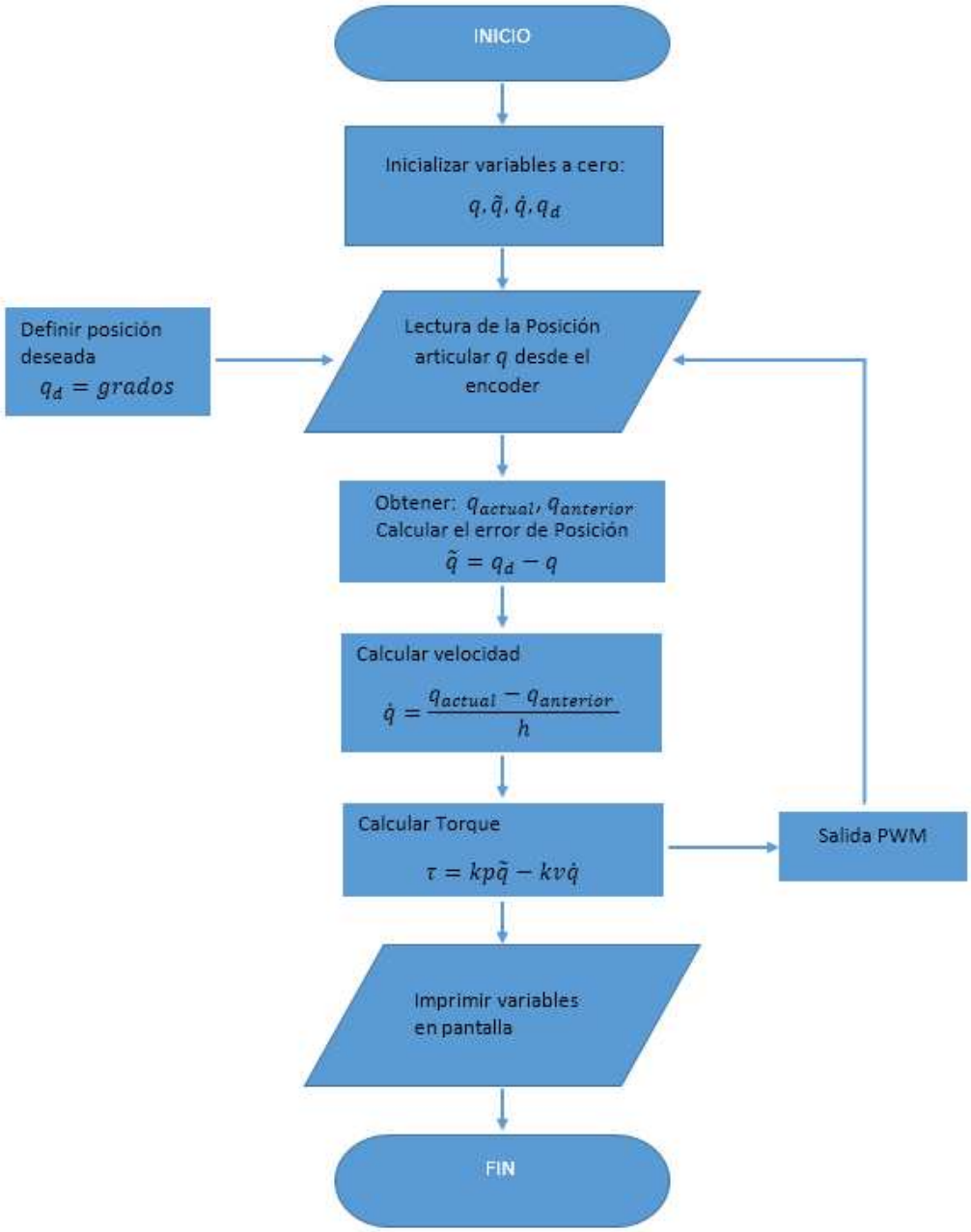


Figura 4.18: Proceso de implementación del control

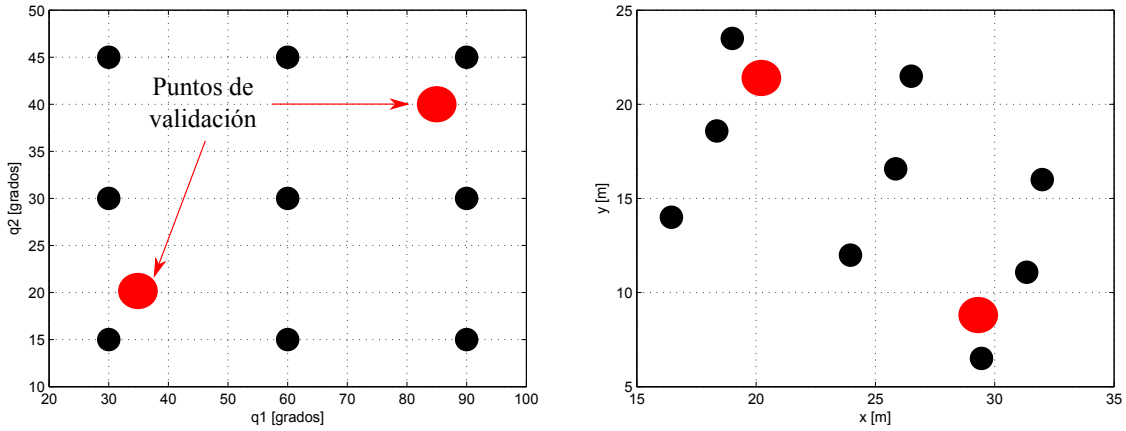
a que el rango para la señal de *pwm* es de $0 - 1$, sin embargo en las señales de torque calculados se obtienen valores superiores.

Programación del control de posición en la tarjeta

En las siguientes líneas se presenta el programa creado en la plataforma *mbed*, haciendo uso de las bibliotecas para el cálculo de las operaciones en tiempo real *rtos.h* y *cmsis_os.h* que es una actualización de la primera. El valor de torque "tau" en la línea 25 se obtiene desde la expresión del esquema PD y en donde el período de muestreo h declarado en la línea 9 es de 90 milisegundos.

```
/INTERRUP_2  ------ >PROGRAMA PARA 1 MOTOR
//REV ---> 13-07-2015
// Ing. Donaciano Coyotecatl Cuautle
//-------------------------------------
#include "mbed.h"
#include "rtos.h"
#include "cmsis_os.h"

//Variables Globales
1. Serial         pc(USBTX, USBRX);//Transmisión de datos puerto serial
2. InterruptIn    chA_(PTA5); //Pin de entrada canal A del encoder para M1
3. InterruptIn    chB_(PTA12);//Pin de entrada canal B del encoder para M1
4. PwmOut         pwm(PTA13); //Pin de salida PWM
5. DigitalOut     dir1(PTD3); //Pin de salida dirección 1 del motor1
6. DigitalOut     dir2(PTD2); //Pin de salida dirección 2 del motor1
7. int            pos_=0;     //Inicialización de la posición a cero
8. int            qd_cuentas = 0; //Inicialización de las CPR a cero
//Período de muestreo
9.  int             h=90;         //Período de muestreo en milisegundos
10. int            estadoA_;      //Variable para el conteo del canal A
11. int            estadoB_;      //Variable para el conteo del canal B
//constantes para el control PD
//Sintonización para 45 [grados] = 1050 CPR
12. float          kp=1.1;         //Valor para la ganancia proporcional
13. float          kv=0.6;         //Valor para la ganancia derivativa
//Parámetros del control PD [posición-velocidad]
14. int            qtilde;//Error de posición en cuentas por revolución (CPR)
15. float          qp;    //Velocidad en (CPR)
16. int            q_actual=0;  //Posición actual en CPR
17. int            q_anterior=0;//Posición anterior en CPR
//Par Articular
18. float         tau;          //Torque articular

//Métodos-->Funciones
//---------------------------------------
//Función de interrupción para conteo en el canal A
19. void interrupcionA(){
    estadoA_=chA_.read();
    estadoB_=chB_.read();
        if(estadoA_){(estadoB_)?pos_--:pos_++; }
        else{(estadoB_)?pos_++:pos_--;}
        }                 //Termina función
//------------
//Función de interrupción para conteo en el canal B
20. void interrupcionB(){
    estadoA_=chA_.read();
    estadoB_=chB_.read();
```

```
        if(estadoB_){(estadoA_)?pos_++:pos_--;}
        else{(estadoA_)?pos_--:pos_++;}
        }                //Termina función
//----------------------------------------
//Declaración de la función "controlar" para implementar control PD
21. void controlar(void const *args){
22.    q_actual = pos_;                //Posición actualizada en CPR
23.    qtilde = qd_cuentas - q_actual; //Cálculo del error de posición
24.    qp = (q_actual - q_anterior)/h*1000;//Velocidad en cuentas/seg

//---> Algoritmo de control
25.    tau =  ((kp * qtilde)/ qd_cuentas) - ((kv*qp)/9000);//Torque
//CLOCK  (Sentido horario)
26.    if(tau > 0 ) {
                dir1=1;dir2=0;pwm=tau;
                }
//CCLOCK (Sentido anti-horario)
27.    if(tau < 0) {
               dir1=0;dir2=1;pwm=-tau;
               }
//Muestra variables calculadas en pantalla
28.    pc.printf("pos:%d, pwm: %f, qd:%d, tau:%f, qtilde:%d,
       vel:%f \n\r",pos_,pwm.read(),qd_cuentas,tau,qtilde,qp);
       q_anterior = q_actual;
}


//---------INICIA PROGRAMA PRINCIPAL-----------
29. int main() {
30.    pc.printf(" Inician interrupciones!! \n \r");
    //Interrupciones
31.     chA_.rise(&interrupcionA);
32.     chA_.fall(&interrupcionA);
33.     chB_.rise(&interrupcionB);
34.     chB_.fall(&interrupcionB);
    //-------------------------
//rtosTIMER   Interrupción periódica cada h [ms]
35.     RtosTimer  control_h(&controlar, osTimerPeriodic);
36.     control_h.start(h);
//--------------------------------------------------------
   while(1) {
37.          pc.printf("\n\n\n  Rutina 1 \n \n \n ");
            // Posición deseada en cuentas
38.          qd_cuentas = 1050; //45 degrees
39.          wait(15);
            }
}//------TERMINA PROGRAMA PRINCIPAL---
```



(a) Articular (b) Cartesiano

Figura 4.19: Espacio de trabajo y puntos de validación del prototipo

4.4. Resultados experimentales

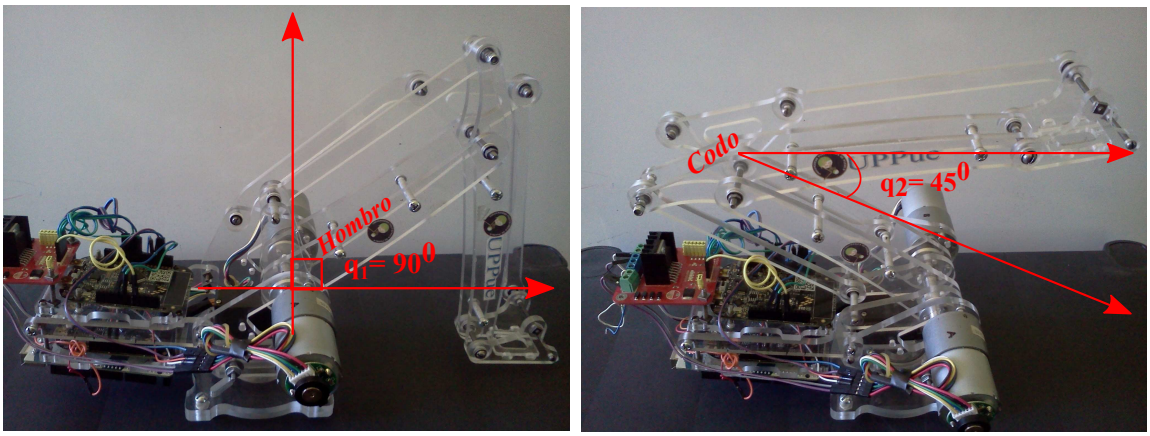
En esta sección se describe la implementación del control neuronal en la tarjeta de desarrollo Freescale FDRM-KL25Z, esto con la finalidad de que el prototipo pueda sintonizar automáticamente sus ganancias proporcional y derivativa. Los resultados experimentales son obtenidos desde el puerto serial de la computadora y las gráficas presentadas se realizan utilizando la interfaz de Matlab.

4.4.1. Entrenamiento de la red neuronal para el prototipo

Para el proceso de entrenamiento de la red neuronal en primera instancia se requiere de la selección de los puntos de entrenamiento distribuidos de manera uniforme a lo largo y ancho del espacio de trabajo del manipulador, mismo que se ilustra en la Figura 4.19. Para el entrenamiento del prototipo se eligieron 9 posiciones deseadas localizadas en el espacio articular, sin embargo, es preciso utilizar el modelo de cinemática directa del robot para poder representar dichos puntos en el espacio cartesiano de manera que se conozca el posicionamiento del efector final en coordenadas [x,y], En el apéndice B se presenta el método geométrico con el cual se obtuvo dicho modelo.

En la Figura 4.20 se presentan dos posiciones de restricción con lo cual se limita el movimiento del robot en cada una de las articulaciones, estas posiciones límite también fueron determinantes para la selección de los puntos de entrenamiento de la red. De este modo la posición angular para la articulación 1 u "hombro"del robot es de 0-90 [grados] y para la articulación 2 o "codo" de 0-45 [grados], esto con la finalidad de evitar colisiones entre los eslabones del prototipo robótico.

Para llevar a cabo el proceso de sintonización de las ganancias k_p y k_v correspondientes a cada uno de los puntos deseados, los cuales son considerados como datos de entrenamiento, se fijó una posición inicial o de "casa"para el prototipo que es mostrada en la



(a) Límite 1 (b) Límite 2

Figura 4.20: Restricciones de la arquitectura

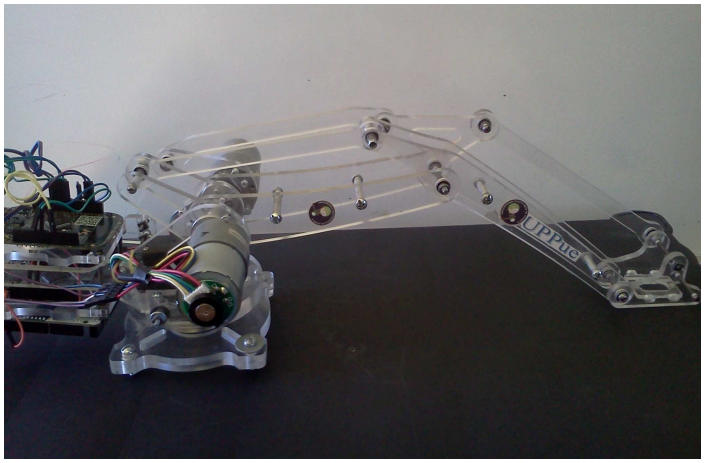


Figura 4.21: Posición de casa

Figura 4.21, esta posición también puede considerarse como de reposo del robot. Cada posición de la Figura 4.19 debe ser sintonizada manualmente en el prototipo hasta conseguir el mejor desempeño bajo las siguientes consideraciones: se establece un tiempo de respuesta menor a 1 segundo, se debe obtener el menor sobreimpulso posible y el valor de torque en los actuadores no debe sobrepasar sus límites nominales con la finalidad de evitar la saturación.

El resultado de las pruebas de control de posición realizadas para determinar el valor de las ganancias k_p y k_v en el prototipo se presentan en la Tabla 4.2. Los nueve puntos constan de sus posiciones angulares deseadas qd_1, qd_2 y la sintonización de cuatro ganancias que corresponden a dos por cada grado de libertad.

Resultados del entrenamiento

Los datos de entrada presentados en la Tabla 4.2 son ingresados al programa de entrenamiento desarrollado en Matlab considerando un ancho de las funciones gaussianas σ=0.6, el cual se presenta en el apéndice C. Este proceso de entrenamiento da como

Puntos	qd_1	qd_2	kp_1	kv_1	kp_2	kv_2
1	30	15	0.9	0.75	0.45	0.35
2	30	30	0.9	0.75	0.9	0.75
3	30	45	0.9	0.75	1.1	0.6
4	60	15	2	1	0.45	0.35
5	60	30	2	1	0.9	0.6
6	60	45	2	1	1.3	0.6
7	90	15	1	0.7	0.45	0.35
8	90	30	1	0.7	0.9	0.75
9	90	45	1	0.7	1.1	0.6

Tabla 4.2: Puntos de entrenamiento para la red neuronal

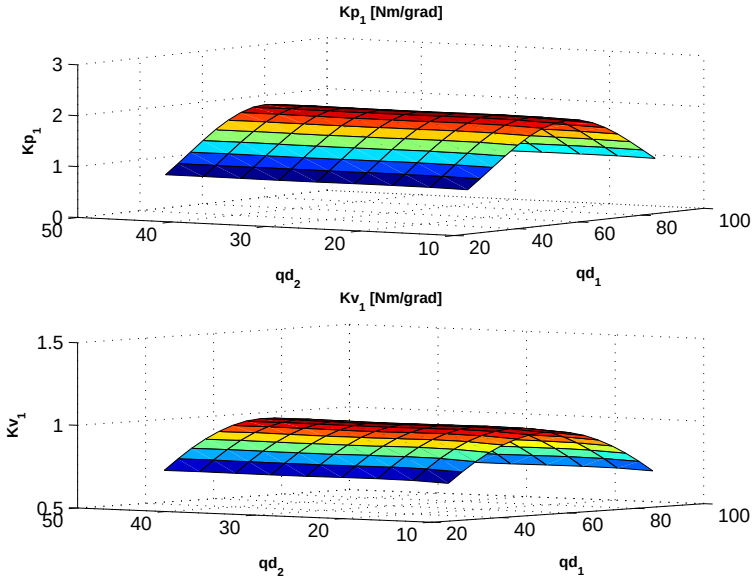


Figura 4.22: Salidas de la red neuronal para la articulación 1

resultado una red de interpolación gaussiana por cada variable kp_1, kp_2, kv_1 y kv_2 con lo cual se obtiene la sintonización automática para cualquier punto deseado por el usuario. La autosintonización como ya fue descrita en el capítulo 3 esta basada en función de la posición deseada y por consiguiente la salida de la red neuronal son estimaciones de los puntos de entrenamiento.

En la Figuras 4.22 y 4.23 se presentan las variaciones de las ganancias proporcional y derivativa k_p y k_v como funciones de las posiciones deseadas en el espacio articular. Para la articulación 1 con estimaciones kp_1 y kv_1 puede observarse la aproximación a una función cuadrática o de segundo orden, sin embargo para la articulación 2 la estimación kp_2 se presenta como una sábana casi lineal y para kv_2 las variaciones de los valores son menos predecibles con mayores protuberancias.

Cabe mencionar que las estimaciones obtenidas de las ganancias pueden variar en mayor grado produciendo una cantidad considerable de montes y valles como se presentó en la sección 3.5 de manera proporcional a la variación de las ganancias utilizadas en

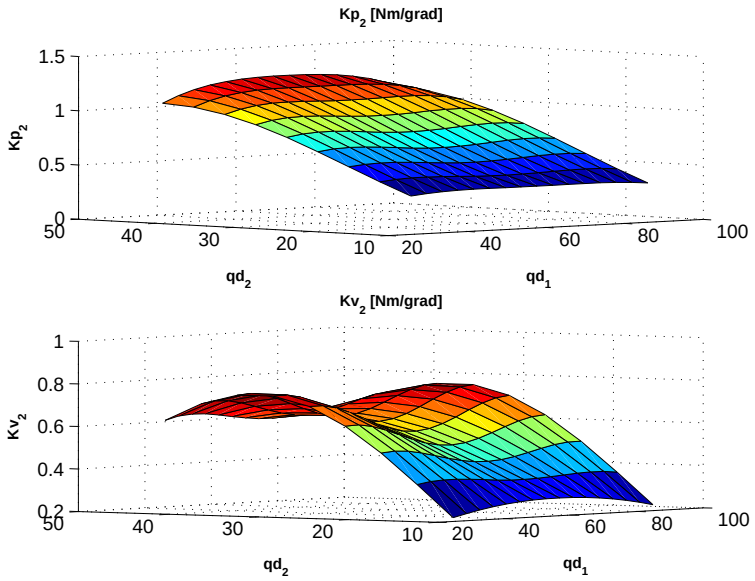


Figura 4.23: Salidas de la red neuronal para la articulación 2

el entrenamiento y por ende al espacio de trabajo utilizado por el manipulador. Las restricciones debidas a la estructura del prototipo son la razón por la cual las ganancias kp y kv son en su mayoría del mismo orden.

4.4.2. Programación de la red neuronal en la tarjeta de desarrollo

Con los resultados obtenidos en la Sección 4.4.1 se procede a implementar el control neuronal del tipo PD en la tarjeta de desarrollo, donde los valores para las ganancias k_p y k_v son calculados a partir de las redes de interpolación presentada en la Tabla 4.2. Dichos valores dependen de la posición deseada de cada articulación del robot manipulador, la cual puede ser cualquiera siempre y cuando este dentro de los límites de entrenamiento q1=(0-90) y q2=(0-45) [grados].

En el apéndice D se presenta el programa para implementar la red neuronal RBF en la tarjeta de desarrollo Freescale FDRM-KL25Z utilizando la plataforma "mbed" como la interfaz para realizar la programación y compilación. En la Figura 4.24 se muestran los resultados obtenidos al programar la tarjeta para obtener las salidas estimadas de la red neuronal basadas en los pesos calculados en el proceso de entrenamiento. Como se observa para el primer punto de entrenamiento P1=[30,15] la salida estimada corresponde con el punto 1 de la Tabla 4.2.

De manera análoga la salida estimada para el punto P9=[90,45] obtenida en la tarjeta se muestra en la Figura 4.25. Los resultados indican la correcta aproximación de la red de entrenamiento con nueve puntos, sin embargo la red de interpolación no esta restringida para aceptar cualquier valor angular de entrada como ya se ha mencionado.

En la Figura 4.26 se presenta el valor estimado de las ganancias proporcionales y de-

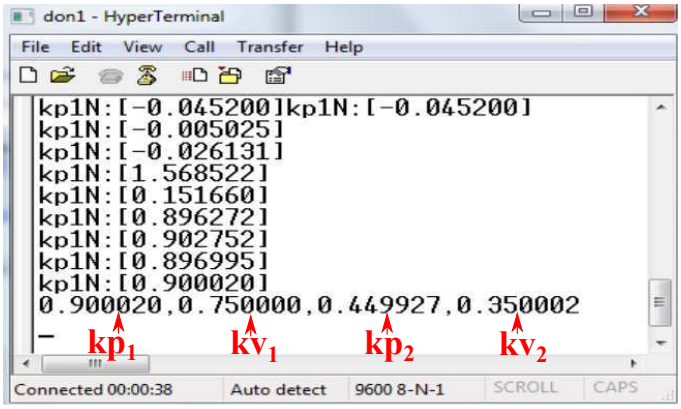


Figura 4.24: Posición deseada qd1=[30,15]

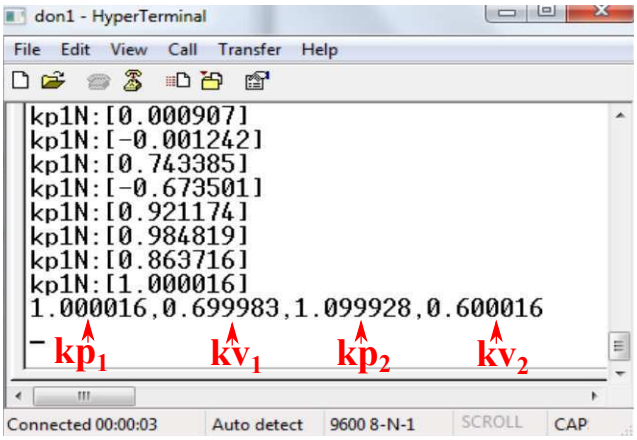


Figura 4.25: Posición deseada qd9=[90,45]

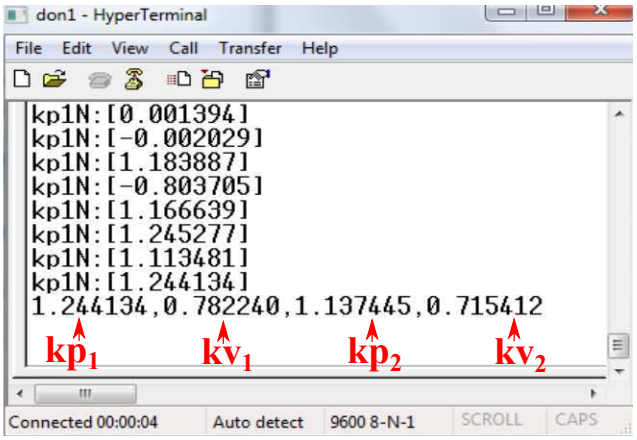
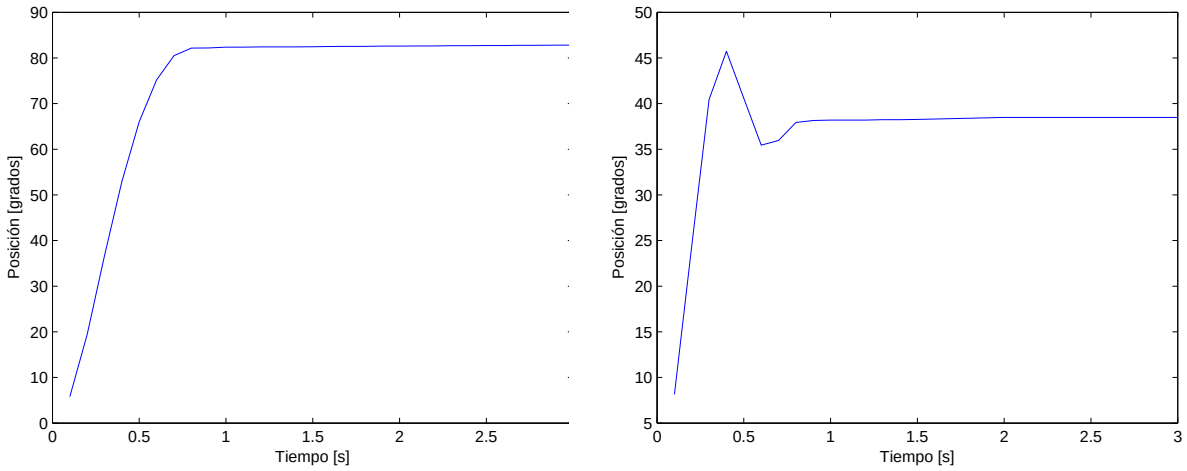


Figura 4.26: Posición deseada qd=[85,40]

rivativas para un punto de validación qd=[85,40] como la posición deseada de cada articulación. Los valores de las ganancias del primer grado de libertad kp_1, kv_1 y del segundo grado de libertad kp_2, kv_2 se presentan de izquierda a derecha respectivamente.



(a) Posición articulación 1 (b) Posición articulación 2

Figura 4.27: Posiciones articulares punto de validación V1=[85,40]

Pruebas en el prototipo robótico

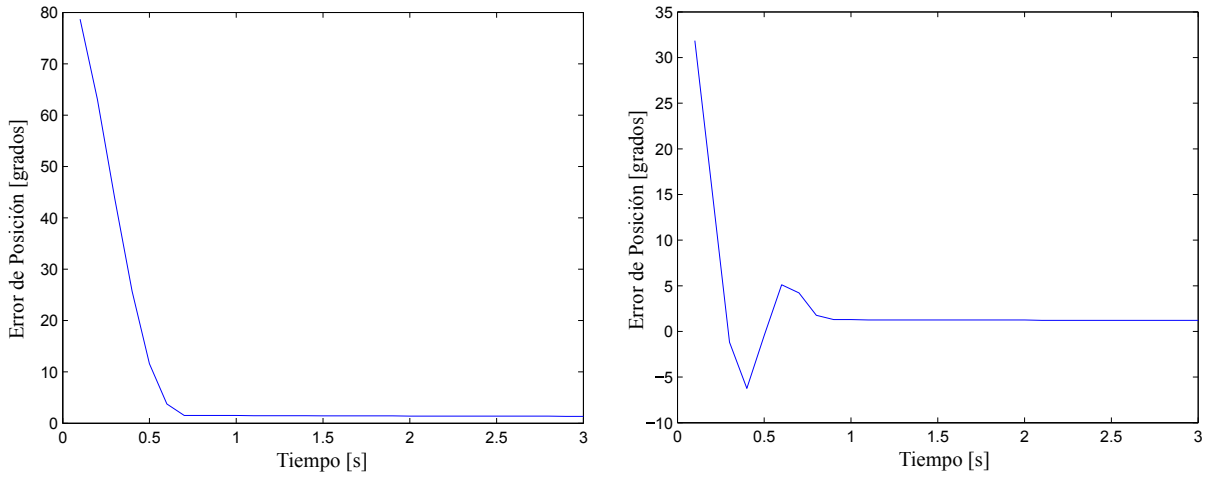
A continuación se presentan dos pruebas realizadas en el prototipo con la finalidad de validar el control de posición con sintonización automática de las ganancias kp y kv utilizando redes neuronales RBF programadas en la tarjeta. En el apéndice E se presenta el programa completo del control de posición para los dos motores del robot y las cuatro redes neuronales que son utilizadas para estimar cada una de las ganancias.

Para realizar el proceso de validación se eligieron dos posiciones extremas del manipulador V1=[85,40] y V2=[35,20] (Figura 4.19), las cuales no se encuentran explícitos en los datos de entrenamiento. Con la finalidad de graficar el comportamiento dinámico del robot a estos valores de entrada y el resultado de auto-sintonización se utilizó transmisión de datos por el puerto serial. El programa de graficación se escribió en Matlab y se describe en el apéndice F.

Punto de validación V1=[85,40]

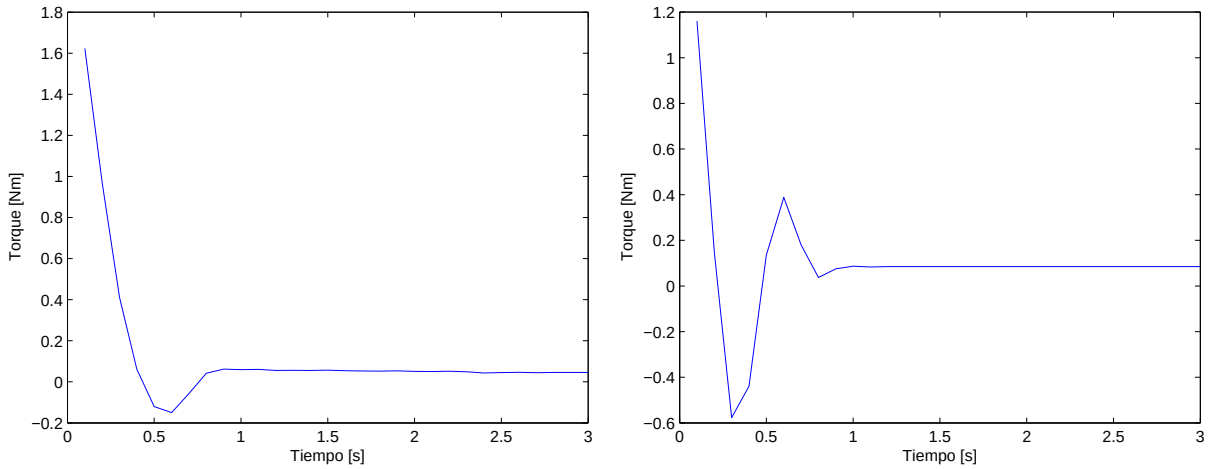
En la Figura 4.27 se muestra la visualización de la respuesta de posición para cada articulación del manipulador en el punto de validación 1. Como se observa en la Figura 4.27a la respuesta del eslabón 1 tiene un buen desempeño ya que no existe sobretiro, el tiempo de establecimiento es menor a 1 segundo y el error de posición (Figura 4.28a) tiene un valor casi igual a cero. De la gráfica mostrada en la Figura 4.27b para la articulación 2 con una posición deseada de 40 grados se observa un sobretiro considerable del 12.5 %, sin embargo, el tiempo de establecimiento a la posición deseada se realiza en menos de un segundo y el error de posición (Figura 4.27b) es de aproximadamente 2 grados. Cabe señalar que los resultados obtenidos de los valores de las ganancias para el punto de validación V1=[85,40] a la salida de las cuatro redes neuronales fueron las siguientes: kp_1=1.2441, kv_1=0.7823, kp_2=1.1375, kv_2=0.7154.

La visualización del valor de torque para la articulación 1 y 2 se muestra en la Figura 4.29. El torque articular aplicado por los motores al sistema alcanza un valor máximo de



(a) Articulación 1 (b) Articulación 2

Figura 4.28: Errores de posición punto 1



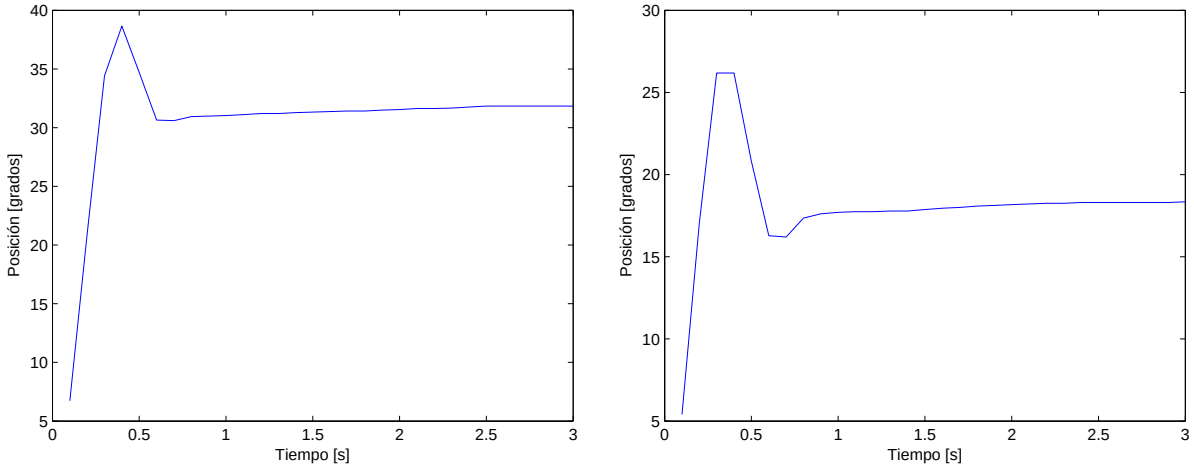
(a) Articulación 1 (b) Articulación 2

Figura 4.29: Torques articulares punto 1

1.6 [N-m] y mínimo de 0.05 [N-m] para el primer grado de libertad aproximadamente. Del mismo modo para el segundo grado de libertad el valor máximo y mínimo de 1.18 [N-m] y 0.1 [N-m] respectivamente. Con los valores anteriores de torque se comprueba que los actuadores no llegan a su límite de saturación que es de 1.765 [N-m], el cual es un dato proporcionado por el fabricante.

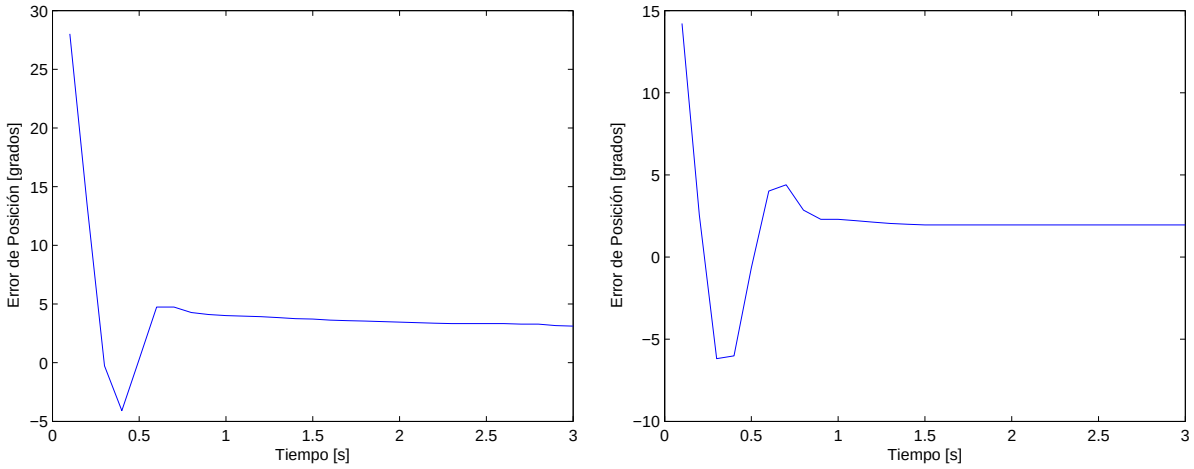
Punto de validación V2=[35,20]

La respuesta para el segundo punto de validación se presenta en la Figura 4.30. Como se observa el sobretiro en la primera articulación es mayor que el presentado en la articulación dos de aproximadamente 9 y 7 grados con respecto a los valores angulares deseados respectivamente, sin embargo, es importante notar que el sistema se intenta



(a) Posición articulación 1 (b) Posición articulación 2

Figura 4.30: Posiciones articulares punto 2 de validación

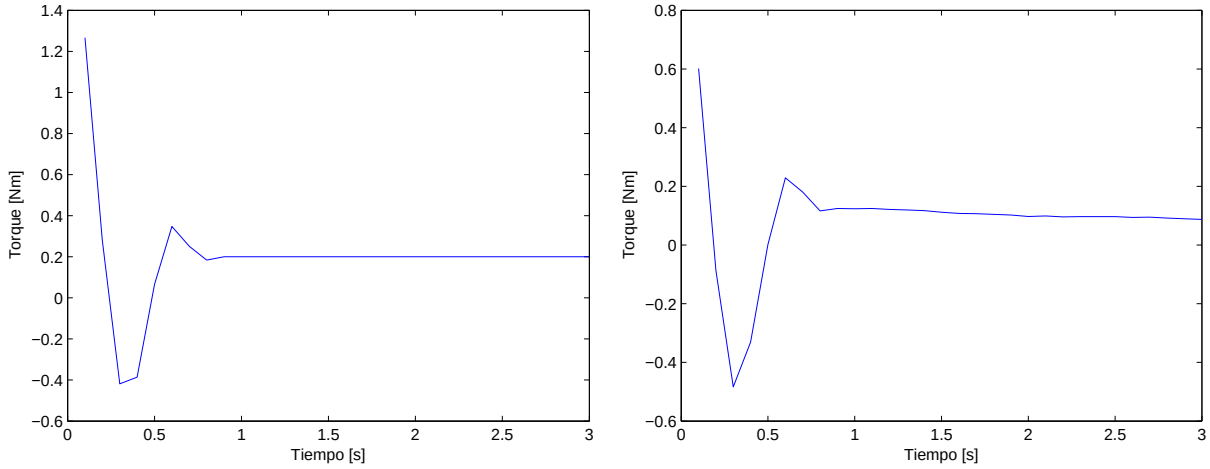


(a) Articulación 1 (b) Articulación 2

Figura 4.31: Errores de posición punto 2

estabilizar en un tiempo menor a 1 segundo como en el caso del primer punto de validación. Los errores articulares se muestran en la Figura 4.31 donde para la articulación uno oscila entre los valores de -5 y 5 grados y para la segunda articulación de -7 y 5 grados, cabe señalar que el error en estado estable es menor a 5 grados para ambos grados de libertad.

En la Figura 4.32 se presentan los torques articulares del robot, como se puede observar los límites de saturación en los actuadores no se alcanzan, además que el torque aplicado es mayor en comparación con los torques del punto de validación 1. Con la finalidad de que el efector final del manipulador pueda permanecer indefinidamente en la posición deseada el control de posición mantiene un torque constante de 0.2 [N-m] y de 0.1 [N-m] para la primera y segunda articulación respectivamente.



(a) Articulación 1 (b) Articulación 2

Figura 4.32: Torques articulares punto 2

Es importante mencionar que para este trabajo no se compensan los fenómenos de gravedad y fricción. En el primer caso debido a que los motores tienen instalados una caja de reducción de velocidad por medio de engranes, la gravedad no afecta considerablemente a la posición del efector final del manipulador, sin embargo la inercia debida a su movimiento si lo hace. La fricción podría afectar al control de posición siempre y cuando este algoritmo requiera de la realimentación del modelo dinámico del prototipo, lo cual no sucede con el control PD. Para el caso de seguimiento de trayectorias se puede evitar el uso de la dinámica del prototipo implementando el control neuronal PD-N del capítulo 3. De acuerdo a la validación realizada y a las gráficas obtenidas se puede concluir que el error en estado estable de al menos 5 grados se debe al fenómeno de backlash que es el debido al juego mecánico en la caja de engranajes, lo cual es palpable en el prototipo.

Comparación con la interpolación lineal y cuadrática

Debido a que las salidas de las redes neuronales para las ganancias kp_1 y kv_1 correspondientes al primer grado de libertad del prototipo adoptaron una forma aproximada de la función cuadrática (Figura 4.22) y la ganancia kp_2 a una función lineal (Figura 4.23), en este apartado se presenta una comparación entre los resultados obtenidos por el entrenamiento utilizando redes neuronales RBF (red de interpolación gaussiana) y los alcanzados por un proceso de interpolación lineal y cuadrática de los valores de las ganancias de la tabla 4.2.

Los valores de las ganancias calculadas por la red de interpolación gaussiana o neuronal se presenta en la Tabla 4.3, que como se ya ha mencionado reiteradamente son funciones de las posiciones deseadas qd_1 y qd_2. Los resultados de la interpolación lineal o cuadrática de datos de la Tabla 4.2 son obtenidos a través de las funciones $polyfit()$ y $polyval()$ implementados en el software de Matlab, dichos programas se presentan en el apéndice G.

Puntos	**qd$_1$**	**qd$_2$**	**kp$_1$**	**kv$_1$**	**kp$_2$**	**kv$_2$**
1	30	15	0.9	0.75	0.45	0.35
2	36	18	0.1990	0.8436	**0.5560**	**0.4624**
3	42	21	1.4953	0.9172	0.6460	0.5325
4	48	24	1.7505	0.9683	0.7253	0.5644
5	54	27	1.9276	0.9959	0.8065	0.5789
6	60	30	2	1	1.3	0.6
7	66	33	1.9571	0.9811	1.0041	0.6380
8	72	36	1.8068	0.9401	1.1015	0.6821
9	78	39	1.5736	0.8780	1.1650	0.7056
10	84	42	**1.2922**	**0.7969**	**1.1687**	**0.6819**
11	90	45	1	0.7	1.1	0.6

Tabla 4.3: Posiciones de validación calculadas por la red neuronal RBF

Para la *interpolación lineal* es necesario la selección de dos puntos consecutivos dentro de la Tabla 4.2 y cuyos valores tengan cierto grado de variación, es decir, que no se repitan consecutivamente. De este modo los puntos seleccionados son el 1 y 2 con valores correspondientes a las ganancias kp_2 y kv_2. El punto de referencia para poder realizar la comparación es P2=[36,18] de la Tabla 4.3.

Punto 1 $\Longrightarrow$ qd_2=15; kp_2=0.45; kv_2=0.35
Punto 2 $\Longrightarrow$ qd_2=30; kp_2=0.9; kv_2=0.75.

Como los valores para las ganancias kp_1 y kv_1 correspondientes a la posición deseada qd_1 no cambian (Tabla4.2), sólo se tomaron los valores de la posición deseada qd_2. De este modo los resultados alcanzados por el programa en Matlab para la interpolación lineal evaluados en qd_2=18 fueron los siguientes: $fkp_2(18)$=0.54, $fkv_2(18)$=0.43 y en la Tabla 4.3 se tiene 0.5560 y 0.4624, con lo cual se tiene una variación del 2.87 % y 7 % respectivamente en relación a los valores ofrecidos por la red neuronal.

Tomando los valores de las ganancias kp_1, kv_1, kp_2 y kv_2 de los puntos 6 y 7 en la Tabla 4.2 y considerando la posición de referencia a evaluar localizada en el punto P10=[84,42] de la Tabla 4.3, la interpolación lineal arroja los siguientes resultados: $fkp_1(84)$=2.6, $fkv_1(84)$=1.18, $fkp_2(42)$=1.81, $fkv_2(42)$=0.75, con variaciones respecto a la Tabla 4.3 de 50.3 %, 32.4 %, 35.4 % y 9 % respectivamente.

Para el proceso de *interpolación cuadrática* se seleccionan tres puntos consecutivos con distintas variaciones como principal criterio. Con respecto al punto a evaluar P2=[36,18] y considerando los puntos 1,2 y 3 de la Tabla 4.2 se tiene que, qd_2=[15,30,45], kp_2=[0.45,0.9,1.1] y kv_2=[0.35,0.75,0.6], con lo cual se obtiene:
$fkp_2(18)$=0.56, $fkv_2(18)$=0.4740 que representan el 0.7 % y 2.4 % de variación.

Los resultados para el punto P10=[84,42] con los mismos puntos 1,2 y 3 de la Tabla 4.2 fueron $fkp_2(42)$=1.08 $fkv_2(42)$=0.6740 de modo que la variación en relación a la red de interpolación gaussiana es del 7.6 % y 1.16 % para ambas ganancias kp_2 y kv_2.

Capítulo 5

Conclusiones

Finalmente en este capítulo se presentan las conclusiones referentes a la realización del trabajo de tesis cuyo principal problema a resolver fue el diseño de una ley de control que permitiera el seguimiento de trayectorias sin la necesidad de utilizar los parámetros del modelo dinámico del sistema que se esté estudiando. La solución presentada a esta demanda se logra mediante la sintonización automática de las ganancias en un control de posición tipo PD utilizando redes neuronales de base radial (RBF).

5.1. Resumen

Durante el desarrollo del presente trabajo de tesis uno de los problemas con los que nos enfrentamos fue la implementación del control de posición PD en los actuadores del prototipo didáctico con dos grados de libertad (GDL). En el proceso de elaboración del estado del arte se encontró que el control de posición sólo es un caso particular del control de movimiento de robots, por lo cual no sólo se requiere contemplar una posición deseada para el efector final del robot sino de toda una secuencia de posiciones con el objetivo de seguir trayectorias específicas y de esta manera poder realizar actividades de aplicación industrial.

El problema de seguimiento de trayectorias se le puede atacar bajo dos esquemas de control, punto a punto y de trayectoria puro. En el primer caso este tipo de control se basa en el control de posición PD con compensación de gravedad, los esquemas de control de trayectoria también son aplicables sin embargo surge un problema sistémico ya que la posición deseada deja de ser constante para convertirse en una variable de estado.

En base a lo anterior la implementación de un control de trayectoria como una solución al problema de seguimiento deja de ser trivial debido a que se requiere de la realimentación de los parámetros del modelo dinámico y esto en muchas ocasiones es difícil de calcular. Cabe mencionar que existen algunas técnicas de identificación paramétrica (Reyes, 2012) en donde se utilizan algoritmos recursivos para la aproximación de parámetros sin embargo el costo computacional es elevado.

Para resolver el problema que se tiene con el control de trayectoria y debido a que el control de posición sólo requiere calcular el valor del par gravitacional, en el capítulo 3 se presenta una metodología para el diseño de cuatro redes neuronales de base radial que permiten aproximar las ganancias proporcionales y derivativas de un control de posición del tipo PD a partir de un proceso de entrenamiento. Esta característica de las redes neuronales RBF calcula el valor de las ganancias k_p y k_v en función de la posición deseada y por tanto se dice que las ganancias en el control de posición son variables y con sintonización automática de sus valores.

Con la finalidad de probar mediante un proceso de simulación el algoritmo neuronal PD-N propuesto en el capítulo 3 y poder compararlo con otras leyes de control en el seguimiento de trayectorias se utilizó el modelo dinámico y cinemático de un sistema robótico parametrizado propuesto por (Reyes, 2011). Cabe mencionar que la dinámica se obtuvo de las ecuaciones de movimiento de Euler-Lagrange y que para la cinemática tanto directa como inversa se utilizaron métodos geométricos.

La construcción del prototipo robótico para fines didácticos deja como aportación la conjugación de distintos procesos, tal como el diseño, fabricación de piezas (maquinado) y construcción. La instrumentación tanto electrónica como de control permiten enfoque de un proceso global como lo es la automatización.

El trabajo de tesis fue planteado desde el problema de la sintonización automática para el control de posición y de la necesidad de conocer el modelo dinámico del robot para implementar control de trayectoria. La solución propuesta radica en la utilización de redes neuronales de base radial con la finalidad de conseguir auto-sintonización de las ganancias proporcional y derivativa en un control PD, todo ello para realizar el seguimiento correcto de trayectorias deseadas.

5.2. Discusión

Objetivo específico 1: Proponer una ley de control de posición tipo PD auto-sintonizable a partir de redes neuronales de base radial.

De acuerdo a la revisión de la literatura para la elaboración del estado del arte en el capítulo 1, se planteo la propuesta de un regulador de posición para el seguimiento de trayectorias mediante la sintonización automática de sus ganancias y haciendo uso de las redes neuronales de base radial como primer objetivo específico.

Esta idea surge debido a que en la mayoría de los trabajos de investigación no se contemplan ganancias variables en el control de posición y sólo en algunos casos se presentan controladores con variación de ganancias, sin embargo dependen directamente del error de posición o de los estados del sistema.

Cabe señalar que el control de trayectoria es implementado en la mayoría de los estudios, pero como ya se ha mencionado este tipo de esquemas requiere conocer en primera instancia los valores para todos los parámetros del modelo dinámico.

Es por ello que diferencia de otros autores las redes neuronales en este trabajo de tesis

no fueron tratadas para compensar fenómenos no lineales inherentes al sistema sino para obtener sintonización automática de las ganancias proporcional y derivativa en un control de posición.

Debido a que el proceso de auto-sintonización se encuentra basado en un control de posición donde sólo se requiere el valor del par gravitacional, se tiene como resultado prescindir del conocimiento del modelo dinámico para cualquier sistema o prototipo.

Por tales motivos la implementación del control neuronal PD-N propuesto en el capítulo 3 es posible y que además se obtuvieron buenos resultados de desempeño y exactitud para el seguimiento de trayectorias desde el espacio cartesiano. La siguiente información resalta las mejoras obtenidas con respecto al control de posición PD clásico propuesto por (Arimoto and Takegaki, 1981).

De acuerdo a los indices de desempeño calculados en el capitulo 3 para el seguimiento de una trayectoria circular, el control neuronal PD-N obtuvo una mejora del 74.81 % y para la trayectoria en forma de flor de ocho pétalos el 60.65 %. Sin embargo se observo que a medida que las trayectorias describen figuras más complejas el control saturado TANH obtuvo mejores resultados, el 94.96 % para la trayectoria en forma de flor de ocho pétalos y el 97.35 % para una trayectoria paramétrica (Salas et al., 2012) desde el espacio articular comparado con el 44.39 % obtenido con el control PD-N.

La novedad del nuevo enfoque de control neuronal propuesto PD-N es que no se requiere del conocimiento del modelo dinámico para su implementación en el seguimiento de trayectorias, además de mejorar su desempeño con respecto al control de posición PD clásico logrando que las ganancias se ajusten adecuadamente incluso cuando existen cambios abruptos de dirección.

Objetivo específico 2: Implementar control de posición tipo PD en un prototipo de robot manipulador con dos grados de libertad y arquitectura abierta.

Después de haber concluido la etapa de construcción del prototipo mecánico se procedió a cumplir con el segundo objetivo específico. Para lograrlo se desarrollo un programa bajo la plataforma "mbed"(Arm Ltd, 2014) con rutinas destinadas al conteo de los pulsos por revolución del encoder con la finalidad de obtener la posición y velocidad angular de los motores, calcular el torque adecuado para los actuadores desde el control de posición PD y la implementación de la red neuronal entrenada para alcanzar la sintonización automática en el prototipo.

Los resultados de entrenamiento muestran que si el espacio de trabajo del manipulador es muy reducido debido a las limitaciones que se presentan en la arquitectura mecánica, la redes neuronales resultantes pueden representar funciones casi lineales y cuadráticas. La comparación con un proceso de interpolación lineal y cuadrático reflejan una desviación mínima del 1.16 % y máxima de 50.3 % con respecto a los los valores obtenidos por la interpolación gaussiana o entrenamiento de la red, además cabe señalar que la interpolación no puede realizarse si no se presentan variaciones considerables en los datos de entrenamiento (ganancias calculadas manualmente), ya que se requieren al menos dos puntos con valores diferentes para lograr el proceso.

Como resultados experimentales se obtuvieron gráficas de posición, error y torque para

dos puntos de validación localizados en los extremos del espacio de trabajo, en ellas se muestra la existencia de sobretiros y errores de al menos 5 grados para la mayoría de los puntos, sin embargo es importante mencionar que la respuesta del sistema a la ley de control neuronal PD-N es muy buena con un tiempo menor a 1 segundo.

Para poder disminuir el fenómeno de sobretiro se debe sintonizar de mejor manera cada punto de entrenamiento. Con referencia al error en estado estable el principal problema es el backlash del reductor constituido por engranajes, además de la presencia de fenómenos inerciales debidos a la gravedad y velocidad de los eslabones, por lo cual es importante considerar la compensación de gravedad para lograr reducir dicho error.

Cabe aclarar que para el proceso de implementación de las redes neuronales en la tarjeta con la finalidad de obtener auto-sintonización del prototipo no se consideró la compensación de gravedad ya que se partió de la idea en que la caja de engranajes no permitiría el movimiento del eslabón cuando la posición deseada fuera alcanzada, sin embargo ya en la experimentación se observo que esto no sucede para cualquier posición.

5.3. Perspectivas

- Modelado dinámico del prototipo para simulación del sistema.
  En base a las ecuaciones de movimiento de Euler-Lagrange obtener el modelo dinámico del prototipo, con la finalidad de visualizar la respuesta del sistema mediante simulaciones por computadora bajo la implementación de diferentes algoritmos de control tanto de posición, trayectoria pura e incluso el propio control neuronal PD-N.

- Modelado de la cinemática inversa.
  Si se pretende realizar seguimiento de trayectorias desde el espacio cartesiano en coordenadas [x,y], es necesario obtener el modelo de la cinemática inversa la cual puede ser obtenida desde métodos geométricos o establecidos como el de Denavit-Hartenberg.

- Programar diferentes trayectorias en el prototipo.
  Además de las trayectorias desde el espacio cartesiano se pueden programar trayectorias directamente en el espacio articular tal como las presentadas en (Salas et al., 2012, Kelly et al., 2005) las cuales son trayectorias de referencia paramétrica.

- Simular la trayectoria de referencia paramétrica con el modelo del prototipo.
  Para desarrollar algoritmos de control más eficientes para el prototipo se deben simular trayectorias parametrizadas en el modelo dinámico del prototipo.

- Implementar el control de posición PD en los actuadores tomando en cuenta la compensación de gravedad.
  Con la finalidad de reducir el error en estado estable y para contrarrestar los efectos de la gravedad sobre los eslabones del prototipo se debe considerar el cálculo del factor de gravedad g(q).

- Creación de una interfaz de usuario.
  Como medio para la facilitar la interacción del robot con el usuario es importante la creación de una interfaz, en la cual se tenga como datos de entrada la posición deseada para las articulaciones del prototipo y como resultado se obtenga el movimiento deseado así como de su gráfica de posición en tiempo real.

Apéndice A

Cálculo del índice de desempeño

El índice de desempeño se utiliza para medir la norma $\mathcal{L}_2$ del error de posición. Un valor pequeño de $\mathcal{L}_2$ representa un error más pequeño y por lo tanto indica un mejor desempeño. Una función vectorial $\mathbb{R}^n \to \mathbb{R}^n \in \mathcal{L}_2$, si al evaluar:

$$\mathcal{L}_2 = \sqrt{\int_o^\infty \|f(t)\|^2 dt < \infty} \tag{A.1}$$

donde $\|f(t)\|$ es la norma euclidiana de la función sobre el intervalo y es un número escalar. Debido a que el tiempo de simulación es finito se debe aplicar el concepto de valor efectivo para calcular la desviación de la función entre cada intervalo de simulación, por lo cual la norma $\mathcal{L}_2$ para el error articular se define de la siguiente manera:

$$\mathcal{L}_2 = \sqrt{\frac{1}{T}\int_o^T \|\tilde{q}(t)\|^2 dt} \tag{A.2}$$

Con la finalidad de implementar el algoritmo para el índice de desempeño la representación de la norma $\mathcal{L}_2$ para el tiempo discreto se presenta como sigue (Sánchez, 2005):

$$\int \|\tilde{q}(t)\|^2 \, dt \to \; I_k = I_{k-1} + h\|\tilde{q}(t)\|^2 \tag{A.3}$$

$$\mathcal{L}_2 = \sqrt{\frac{1}{T} I_k}$$

donde h es el período de muestreo y T es el intervalo de simulación.

Apéndice B

Modelo de cinemática directa del prototipo

El método geométrico utilizado para obtener el modelo de cinemática directa para el prototipo con 2 grados de libertad (GDL) se presenta a continuación. En la Figura

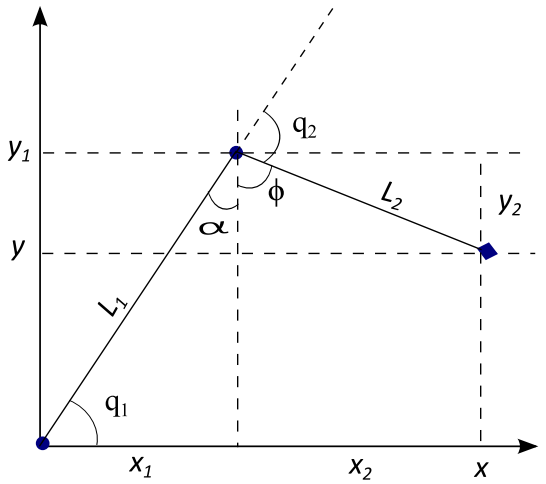


Figura B.1: Cinemática directa del prototipo con 2 GDL

B.1 se observa que las posiciones angulares q_1 y q_2 determinan la orientación de los eslabones 1 y 2 del prototipo, es por ello que la posición en el espacio cartesiano del efector final del manipulador este determinada en función de la posición angular q_1 y q_2.

A partir de la Figura B.1 se obtiene la posición del efector final del robot en coordenadas cartesianas [x,y] representada en las ecuaciones (B.1) y (B.2).

$$x = x_1 + x_2 \tag{B.1}$$

$$y = y_1 - y_2 \tag{B.2}$$

El cálculo para obtener los valores de x_1 y x_2 están relacionados con el ángulo q_1 y ϕ

de la siguiente manera,

$$x_1 = L_1 cos(q_1) \tag{B.3}$$
$$x_2 = L_2 sen(\phi) \tag{B.4}$$

Del mismo modo para y_1 y y_2 se tiene que,

$$y_1 = L_1 sen(q_1) \tag{B.5}$$
$$y_2 = L_2 cos(\phi) \tag{B.6}$$

Para calcular el valor del ángulo α se toman la relación de ángulos en el triángulo formado por L_1, y_1 y x_1,

$$\alpha + q_1 + 90 = 180 \tag{B.7}$$
$$\alpha = 90 - q_1 \tag{B.8}$$

De los ángulos complementarios α y q_2 se obtiene el valor de ϕ,

$$\alpha + \phi + q_2 = 180 \tag{B.9}$$

Despejando el ángulo ϕ de la ecuación (B.9) y sustituyendo el valor de α dado en la ecuación B.8 se tiene que,

$$\phi = 180 - \alpha - q_2 \tag{B.10}$$
$$= 180 - 90 + q_1 - q_2 = 90 + q_1 - q_2 \tag{B.11}$$
$$\phi = 90 - (q_2 - q_1)$$

Con lo anterior las ecuaciones (B.1) y (B.2) se pueden expresar como,

$$x = L_1 cos(q_1) + L_2 sen(90 - [q_2 - q_1]) \tag{B.12}$$
$$y = L_1 sen(q_1) - L_2 cos(90 - [q_2 - q_1]) \tag{B.13}$$

Finalmente al aplicar las identidades trigonométricas para $sen(u - v)$ y $cos(u - v)$ se obtiene la cinemática directa.

$$x = L_1 cos(q_1) + L_2 cos(q_2 - q_1) \tag{B.14}$$
$$y = L_1 sen(q_1) - L_2 sen(q_2 - q_1) \tag{B.15}$$

Apéndice C

Entrenamiento de la red neuronal en MatLab

```
%Programa de entrenamiento Redes neuronales RBF en Matlab
%Ing. Donaciano Coyotecatl Cuautle
%Revisión 10-10-2015

function [Gan]=SAuto(qd1,qd2)

%POSICIÓN DESEADA (ESPACIO ARTICULAR)
qd1rad=qd1*(pi/180);
qd2rad=qd2*(pi/180);

c1g=[30;30;30;60;60;60;90;90;90];
c2g=[15;30;45;15;30;45;15;30;45];
c1rad=c1g*(pi/180);
c2rad=c2g*(pi/180);
%--------------------

%Entrega pesos de entrenamiento
%****************************************
%ENTRADA vector/simple
[W0]=F9_RM2;
%****************************************
%Parámetros de la red neuronal
sigma=0.6;
den2=(sigma^2);
%---------------------------------
W0_interp=W0;
%-------------------------
%RECUPERA PESOS
W_kp1=W0_interp(:,1);
W_kv1=W0_interp(:,2);
```

```
W_kp2=W0_interp(:,3);
W_kv2=W0_interp(:,4);
%==================================0
%%%-SALIDASSS--GANANCIAS KP KV ESTIMADAS
%PARA 9 PUNTOS (X,Y)
%---KP1
k1_0=W_kp1(1)*exp((-1/den2)*((qd1rad-c1rad(1)).^2 +(qd2rad-c2rad(1)).^2));
k2_0=W_kp1(2)*exp((-1/den2)*((qd1rad-c1rad(2)).^2 +(qd2rad-c2rad(2)).^2));
k3_0=W_kp1(3)*exp((-1/den2)*((qd1rad-c1rad(3)).^2 +(qd2rad-c2rad(3)).^2));
k4_0=W_kp1(4)*exp((-1/den2)*((qd1rad-c1rad(4)).^2 +(qd2rad-c2rad(4)).^2));
k5_0=W_kp1(5)*exp((-1/den2)*((qd1rad-c1rad(5)).^2 +(qd2rad-c2rad(5)).^2));
k6_0=W_kp1(6)*exp((-1/den2)*((qd1rad-c1rad(6)).^2 +(qd2rad-c2rad(6)).^2));
k7_0=W_kp1(7)*exp((-1/den2)*((qd1rad-c1rad(7)).^2 +(qd2rad-c2rad(7)).^2));
k8_0=W_kp1(8)*exp((-1/den2)*((qd1rad-c1rad(8)).^2 +(qd2rad-c2rad(8)).^2));
k9_0=W_kp1(9)*exp((-1/den2)*((qd1rad-c1rad(9)).^2 +(qd2rad-c2rad(9)).^2));
kp1N=k1_0+k2_0+k3_0+k4_0+k5_0+k6_0+k7_0+k8_0+k9_0;

%----KV1
k1_0=W_kv1(1)*exp((-1/den2)*((qd1rad-c1rad(1)).^2 +(qd2rad-c2rad(1)).^2));
k2_0=W_kv1(2)*exp((-1/den2)*((qd1rad-c1rad(2)).^2 +(qd2rad-c2rad(2)).^2));
k3_0=W_kv1(3)*exp((-1/den2)*((qd1rad-c1rad(3)).^2 +(qd2rad-c2rad(3)).^2));
k4_0=W_kv1(4)*exp((-1/den2)*((qd1rad-c1rad(4)).^2 +(qd2rad-c2rad(4)).^2));
k5_0=W_kv1(5)*exp((-1/den2)*((qd1rad-c1rad(5)).^2 +(qd2rad-c2rad(5)).^2));
k6_0=W_kv1(6)*exp((-1/den2)*((qd1rad-c1rad(6)).^2 +(qd2rad-c2rad(6)).^2));
k7_0=W_kv1(7)*exp((-1/den2)*((qd1rad-c1rad(7)).^2 +(qd2rad-c2rad(7)).^2));
k8_0=W_kv1(8)*exp((-1/den2)*((qd1rad-c1rad(8)).^2 +(qd2rad-c2rad(8)).^2));
k9_0=W_kv1(9)*exp((-1/den2)*((qd1rad-c1rad(9)).^2 +(qd2rad-c2rad(9)).^2));
kv1N=k1_0+k2_0+k3_0+k4_0+k5_0+k6_0+k7_0+k8_0+k9_0;
%---KP2
k1_0=W_kp2(1)*exp((-1/den2)*((qd1rad-c1rad(1)).^2 +(qd2rad-c2rad(1)).^2));
k2_0=W_kp2(2)*exp((-1/den2)*((qd1rad-c1rad(2)).^2 +(qd2rad-c2rad(2)).^2));
k3_0=W_kp2(3)*exp((-1/den2)*((qd1rad-c1rad(3)).^2 +(qd2rad-c2rad(3)).^2));
k4_0=W_kp2(4)*exp((-1/den2)*((qd1rad-c1rad(4)).^2 +(qd2rad-c2rad(4)).^2));
k5_0=W_kp2(5)*exp((-1/den2)*((qd1rad-c1rad(5)).^2 +(qd2rad-c2rad(5)).^2));
k6_0=W_kp2(6)*exp((-1/den2)*((qd1rad-c1rad(6)).^2 +(qd2rad-c2rad(6)).^2));
k7_0=W_kp2(7)*exp((-1/den2)*((qd1rad-c1rad(7)).^2 +(qd2rad-c2rad(7)).^2));
k8_0=W_kp2(8)*exp((-1/den2)*((qd1rad-c1rad(8)).^2 +(qd2rad-c2rad(8)).^2));
k9_0=W_kp2(9)*exp((-1/den2)*((qd1rad-c1rad(9)).^2 +(qd2rad-c2rad(9)).^2));
kp2N=k1_0+k2_0+k3_0+k4_0+k5_0+k6_0+k7_0+k8_0+k9_0;
%---KV2
k1_0=W_kv2(1)*exp((-1/den2)*((qd1rad-c1rad(1)).^2 +(qd2rad-c2rad(1)).^2));
k2_0=W_kv2(2)*exp((-1/den2)*((qd1rad-c1rad(2)).^2 +(qd2rad-c2rad(2)).^2));
k3_0=W_kv2(3)*exp((-1/den2)*((qd1rad-c1rad(3)).^2 +(qd2rad-c2rad(3)).^2));
k4_0=W_kv2(4)*exp((-1/den2)*((qd1rad-c1rad(4)).^2 +(qd2rad-c2rad(4)).^2));
k5_0=W_kv2(5)*exp((-1/den2)*((qd1rad-c1rad(5)).^2 +(qd2rad-c2rad(5)).^2));
k6_0=W_kv2(6)*exp((-1/den2)*((qd1rad-c1rad(6)).^2 +(qd2rad-c2rad(6)).^2));
k7_0=W_kv2(7)*exp((-1/den2)*((qd1rad-c1rad(7)).^2 +(qd2rad-c2rad(7)).^2));
```

```
k8_0=W_kv2(8)*exp((-1/den2)*((qd1rad-c1rad(8)).^2 +(qd2rad-c2rad(8)).^2));
k9_0=W_kv2(9)*exp((-1/den2)*((qd1rad-c1rad(9)).^2 +(qd2rad-c2rad(9)).^2));
kv2N=k1_0+k2_0+k3_0+k4_0+k5_0+k6_0+k7_0+k8_0+k9_0;

Gan=[kp1N;kv1N;kp2N;kv2N];

end
%- - - - - - - - - - - - - - - - - -  - - - - - -  - - - - - -

%-- FUNCIÓN DE APRENDIZAJE POR INTERPOLACIÓN GAUSSIANA--%
%-- EN UNA RED NEURONAL  DE BASE RADIAL       ----------%
%-- 9 PUNTOS DEL PROTOTIPO MECÁNICO 2GDL        --------%
%Ing. Donaciano Coyotecatl Cuautle          ------------%
%---->02/09/2015

%clear all; clc;

function [W0]=F9_RM2(qd1rad,qd2rad,c1rad,c2rad)

 puntos=9; hc=9;

qd1g=[30;30;30;60;60;60;90;90;90];
qd2g=[15;30;45;15;30;45;15;30;45];
qd1rad=qd1g*(pi/180);
qd2rad=qd2g*(pi/180);

c1g=[30;30;30;60;60;60;90;90;90];
c2g=[15;30;45;15;30;45;15;30;45];
c1rad=c1g*(pi/180);
c2rad=c2g*(pi/180);

qin=[qd1rad,qd2rad];
cin=[c1rad,c2rad];
%-----------------------------------------------------------------
%Parámetros de la red neuronal
%*************
sigma=0.6;
den=(sigma^2);
den2=den;
%*************
%--------------%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%X-C
%AGRUPA EN UN VECTOR TOTAL DE PUNTOS DESEADOS[GRADOS]
for i=1:puntos
 qx=qin(i,:);
qxs(:,i)=qx;
end
```

```
%-----------------
for i=1:hc
  qa=cin(i,:);
  qas(:,i)=qa;
end
%************
var3=1;
for i=1:puntos
   for j=1:hc
       d1=qxs(1,i);
       d2=qxs(2,i);
       d3=qas(1,j);
       d4=qas(2,j);
       res(var3,j)=d1-d3;
       res(var3+1,j)=d2-d4;
   end
   var3=var3+2;
end
%res
%----------x^2
fila=1;
for i=1:puntos
    for j=1:hc
        d1=res(fila,j);
        d2=res(fila+1,j);
        de2(i,j)=[d1 d2]*[d1 d2]';
       %fprintf('\n e (%d,%d)=%d',i,j,mr(i,j));
    end
    fila=fila+2;
end
%de2
%*******************************
%Distancias (dk)Entradas(puntos)--centrosNN
for i=1:puntos
    for j=1:hc
        dist(i,j)=de2(i,j)^0.5;
   end
end
%FUNCIONES GAUSIANAS
for i=1:puntos
    for j=1:hc
        g(i,j)=exp((-de2(i,j))/den);
    end
end

%g  %Muestra nivel de activación de gaussianas
```

```
%-------------------------
%Solución de la ecuación lineal
%GANANCIAS DE APRENDIZAJE
%================================
 %MATRIZ 3X3=9PUNTOS
  kp1=[0.9,0.9,0.9,2,2,2,1,1,1]';
  kv1=[0.75,0.75,0.75,1,1,1,0.7,0.7,0.7]';
  kp2=[0.45,0.9,1.1,0.45,0.9,1.3,0.45,0.9,1.1]';
  kv2=[0.35,0.75,0.6,0.35,0.6,0.6,0.35,0.75,0.6]';
 %================================
%CALCULAR PESOS
%GANANCIAS--(vectores columna)
W_kp1=inv(g)*kp1;
W_kv1=inv(g)*kv1;
W_kp2=inv(g)*kp2;
W_kv2=inv(g)*kv2;

W0=[W_kp1,W_kv1,W_kp2,W_kv2];

%================================0
%%%-SALIDASSS--GANANCIAS KP KV ESTIMADAS
%PARA 9 PUNTOS (X,Y)
%---KP1
k1_0=W_kp1(1)*exp((-1/den2)*((qd1rad-c1rad(1)).^2 +(qd2rad-c2rad(1)).^2));
k2_0=W_kp1(2)*exp((-1/den2)*((qd1rad-c1rad(2)).^2 +(qd2rad-c2rad(2)).^2));
k3_0=W_kp1(3)*exp((-1/den2)*((qd1rad-c1rad(3)).^2 +(qd2rad-c2rad(3)).^2));
k4_0=W_kp1(4)*exp((-1/den2)*((qd1rad-c1rad(4)).^2 +(qd2rad-c2rad(4)).^2));
k5_0=W_kp1(5)*exp((-1/den2)*((qd1rad-c1rad(5)).^2 +(qd2rad-c2rad(5)).^2));
k6_0=W_kp1(6)*exp((-1/den2)*((qd1rad-c1rad(6)).^2 +(qd2rad-c2rad(6)).^2));
k7_0=W_kp1(7)*exp((-1/den2)*((qd1rad-c1rad(7)).^2 +(qd2rad-c2rad(7)).^2));
k8_0=W_kp1(8)*exp((-1/den2)*((qd1rad-c1rad(8)).^2 +(qd2rad-c2rad(8)).^2));
k9_0=W_kp1(9)*exp((-1/den2)*((qd1rad-c1rad(9)).^2 +(qd2rad-c2rad(9)).^2));
kp1N=k1_0+k2_0+k3_0+k4_0+k5_0+k6_0+k7_0+k8_0+k9_0;
%----KV1
k1_0=W_kv1(1)*exp((-1/den2)*((qd1rad-c1rad(1)).^2 +(qd2rad-c2rad(1)).^2));
k2_0=W_kv1(2)*exp((-1/den2)*((qd1rad-c1rad(2)).^2 +(qd2rad-c2rad(2)).^2));
k3_0=W_kv1(3)*exp((-1/den2)*((qd1rad-c1rad(3)).^2 +(qd2rad-c2rad(3)).^2));
k4_0=W_kv1(4)*exp((-1/den2)*((qd1rad-c1rad(4)).^2 +(qd2rad-c2rad(4)).^2));
k5_0=W_kv1(5)*exp((-1/den2)*((qd1rad-c1rad(5)).^2 +(qd2rad-c2rad(5)).^2));
k6_0=W_kv1(6)*exp((-1/den2)*((qd1rad-c1rad(6)).^2 +(qd2rad-c2rad(6)).^2));
k7_0=W_kv1(7)*exp((-1/den2)*((qd1rad-c1rad(7)).^2 +(qd2rad-c2rad(7)).^2));
k8_0=W_kv1(8)*exp((-1/den2)*((qd1rad-c1rad(8)).^2 +(qd2rad-c2rad(8)).^2));
k9_0=W_kv1(9)*exp((-1/den2)*((qd1rad-c1rad(9)).^2 +(qd2rad-c2rad(9)).^2));
kv1N=k1_0+k2_0+k3_0+k4_0+k5_0+k6_0+k7_0+k8_0+k9_0;
%---KP2
k1_0=W_kp2(1)*exp((-1/den2)*((qd1rad-c1rad(1)).^2 +(qd2rad-c2rad(1)).^2));
k2_0=W_kp2(2)*exp((-1/den2)*((qd1rad-c1rad(2)).^2 +(qd2rad-c2rad(2)).^2));
```

```
k3_0=W_kp2(3)*exp((-1/den2)*((qd1rad-c1rad(3)).^2 +(qd2rad-c2rad(3)).^2));
k4_0=W_kp2(4)*exp((-1/den2)*((qd1rad-c1rad(4)).^2 +(qd2rad-c2rad(4)).^2));
k5_0=W_kp2(5)*exp((-1/den2)*((qd1rad-c1rad(5)).^2 +(qd2rad-c2rad(5)).^2));
k6_0=W_kp2(6)*exp((-1/den2)*((qd1rad-c1rad(6)).^2 +(qd2rad-c2rad(6)).^2));
k7_0=W_kp2(7)*exp((-1/den2)*((qd1rad-c1rad(7)).^2 +(qd2rad-c2rad(7)).^2));
k8_0=W_kp2(8)*exp((-1/den2)*((qd1rad-c1rad(8)).^2 +(qd2rad-c2rad(8)).^2));
k9_0=W_kp2(9)*exp((-1/den2)*((qd1rad-c1rad(9)).^2 +(qd2rad-c2rad(9)).^2));
kp2N=k1_0+k2_0+k3_0+k4_0+k5_0+k6_0+k7_0+k8_0+k9_0;
%---KV2
k1_0=W_kv2(1)*exp((-1/den2)*((qd1rad-c1rad(1)).^2 +(qd2rad-c2rad(1)).^2));
k2_0=W_kv2(2)*exp((-1/den2)*((qd1rad-c1rad(2)).^2 +(qd2rad-c2rad(2)).^2));
k3_0=W_kv2(3)*exp((-1/den2)*((qd1rad-c1rad(3)).^2 +(qd2rad-c2rad(3)).^2));
k4_0=W_kv2(4)*exp((-1/den2)*((qd1rad-c1rad(4)).^2 +(qd2rad-c2rad(4)).^2));
k5_0=W_kv2(5)*exp((-1/den2)*((qd1rad-c1rad(5)).^2 +(qd2rad-c2rad(5)).^2));
k6_0=W_kv2(6)*exp((-1/den2)*((qd1rad-c1rad(6)).^2 +(qd2rad-c2rad(6)).^2));
k7_0=W_kv2(7)*exp((-1/den2)*((qd1rad-c1rad(7)).^2 +(qd2rad-c2rad(7)).^2));
k8_0=W_kv2(8)*exp((-1/den2)*((qd1rad-c1rad(8)).^2 +(qd2rad-c2rad(8)).^2));
k9_0=W_kv2(9)*exp((-1/den2)*((qd1rad-c1rad(9)).^2 +(qd2rad-c2rad(9)).^2));
kv2N=k1_0+k2_0+k3_0+k4_0+k5_0+k6_0+k7_0+k8_0+k9_0;

G=[kp1N,kv1N,kp2N,kv2N];

end
%- - - - - - - - - - - - - - - - - - -  - - - - - - -  - - - - - -
```

Apéndice D

Implementación de la red neuronal RBF en la tarjeta

```
//Programa: Implementación de los resultados de entrenamiento
//Ing. Donaciano Coyotecatl Cuautle
//Revisión: 10-10-2015

#include "mbed.h"
#include "rtos.h"


int main()
{
   const float PI=3.141592654;
     int i;
float w1[9]={-0.0452,0.0486,-0.0452,3.4151,-3.6707,3.4151,0.1363,-0.1465,0.1363};
float w2[9]={0.7750,-0.8330,0.7750,1.0474,-1.1258,1.0474,0.6842,-0.7354,0.6842};
float w3[9]={-0.8801,1.5977,-0.1036,1.3312,-2.7523,2.2006,-0.8801,1.5977,-0.1036};
float w4[9]={-2.4091,4.4133,-2.0002,1.8061,-3.1756,1.8932,-2.4091,4.4133,-2.0002};
float kp1[9],kv1[9],kp2[9],kv2[9],kp1N,kv1N,kp2N,kv2N;
float qd1g=30, qd2g=15  ;
float c1r[9]={0.5236,0.5236,0.5236,1.0472,1.0472,1.0472,1.5708,1.5708,1.5708};
float c2r[9]={0.2618,0.5236,0.7854,0.2618,0.5236,0.7854,0.2618,0.5236,0.7854};
float num=-1, den2=0.36, qd1r, qd2r;
          qd1r=qd1g*(PI/180);
          qd2r=qd2g*(PI/180);
            kp1N=0;
            kv1N=0;
            kp2N=0;
            kv2N=0;

            for(i=0; i<=8; i++){
kp1[i]=w1[i]*exp(   (num/den2)*( ((qd1r-c1r[i])* (qd1r-c1r[i]))
```

```
+ ((qd2r-c2r[i])*(qd2r-c2r[i]))  )   );
kv1[i]=w2[i]*exp(   (num/den2)*( ((qd1r-c1r[i])* (qd1r-c1r[i]))
+ ((qd2r-c2r[i])*(qd2r-c2r[i]))  )   );
kp2[i]=w3[i]*exp(   (num/den2)*( ((qd1r-c1r[i])* (qd1r-c1r[i]))
+ ((qd2r-c2r[i])*(qd2r-c2r[i]))  )   );
kv2[i]=w4[i]*exp(   (num/den2)*( ((qd1r-c1r[i])* (qd1r-c1r[i]))
+ ((qd2r-c2r[i])*(qd2r-c2r[i]))  )   );

            kp1N=kp1[i]+kp1N;
            kv1N=kv1[i]+kv1N;
            kp2N=kp2[i]+kp2N;
            kv2N=kv2[i]+kv2N;

            printf("kp1N:[%f]",kp1N);
            printf("\n\r");
         }
        printf("kp1:%f,kv1:%f,kp2:%f,kv2:%f\r\n",kp1N,kv1N,kp2N,kv2N);
    }
```

Apéndice E

Sintonización automática del control PD en el prototipo

```
//Ing. Donaciano Coyotecatl Cuautle
//REVISIÓN 21-09-2015
#include "mbed.h"
#include "rtos.h"
#include "cmsis_os.h"

//CONFIGURACIÓN TARJETA
Serial         pc(USBTX, USBRX); //Comunicación con puerto Serial
InterruptIn    chA1(PTA5);  //Pin de entrada canal A motor 1
InterruptIn    chB1(PTA4);  //Pin de entrada canal B motor 1
InterruptIn    chA2(PTA12); //Pin de entrada canal A motor 2
InterruptIn    chB2(PTD4);  //Pin de entrada canal B motor 2
PwmOut         pwm1(PTA13); //Pin de salida PWM motor 1
PwmOut         pwm2(PTB0);  //Pin de salida PWM motor 2
DigitalOut     dir1(PTD3);  //Pin de salida dirección1 de giro motor 1
DigitalOut     dir2(PTD2);  //Pin de salida dirección2 de giro motor 1
DigitalOut     dir3(PTD0);  //Pin de salida dirección1 de giro motor 2
DigitalOut     dir4(PTD5);  //Pin de salida dirección2 de giro motor 2
//-------------------------
int            pos1=0;
int            pos2=0;
//- -   -   -   -   -   -   -
//posición deseada del Robot [grados]
float           qd1g=35;
float           qd2g=20;
//- -   -   -   -   -   -   -
int             h=90;
//-------------------------
int            estadoA1;
```

```
int            estadoB1;
int            estadoA2;
int            estadoB2;
//-------------------------
//-------------- Ganancias neuronales
float kp1[9],kv1[9],kp2[9],kv2[9], kp1Ne,kv1Ne,kp2Ne,kv2Ne;
//--------------------------
//posición-velocidad PD
float           qtildeM1;
float           qtildeM2;
float           qpM1;
float           qpM2;
float           q_actualM1=0;
float           q_anteriorM1=0;
float           q_actualM2=0;
float           q_anteriorM2=0;
//Salidas Torque
float         tau1;
float         tau2;

//Métodos-->Funciones
//----------------------------------------
void interrupcionA1(){
    estadoA1=chA1.read();
    estadoB1=chB1.read();
        if(estadoA1){
            (estadoB1)?pos1--:pos1++;
            }
        else{
            (estadoB1)?pos1++:pos1--;
            }
                        }
void interrupcionB1(){
    estadoA1=chA1.read();
    estadoB1=chB1.read();
        if(estadoB1){
            (estadoA1)?pos1++:pos1--;
            }
        else{
            (estadoA1)?pos1--:pos1++;
            }
        }
//------------
void interrupcionA2(){
    estadoA2=chA2.read();
    estadoB2=chB2.read();
        if(estadoA2){
```

```
            (estadoB2)?pos2--:pos2++;
            }
        else{
            (estadoB2)?pos2++:pos2--;
            }
        }
void interrupcionB2(){
    estadoA2=chA2.read();
    estadoB2=chB2.read();
        if(estadoB2){
            (estadoA2)?pos2++:pos2--;
            }
        else{
            (estadoA2)?pos2--:pos2++;
            }
        }
//------------------------------
//Funciones  para implementar control  PD
void controlar(void const *args){
    //- -   -   -   -   -   -   -
    float kp1N, kv1N,  kp2N, kv2N;
    float qd_cuentasM1,  qd_cuentasM2, qt1g,qt2g;

    kp1N=kp1Ne; kv1N=kv1Ne; kp2N=kp2Ne; kv2N=kv2Ne;

    // conversión de grados-> cpr
    qd_cuentasM1=(qd1g*8400)/360;
    qd_cuentasM2=(qd2g*8400)/360;
    //- -   -   -   -   -   -   -
    q_actualM1 = pos1;
    q_actualM2 = pos2;
    qtildeM1 = qd_cuentasM1 - q_actualM1;
    qtildeM2 = qd_cuentasM2 - q_actualM2;
    qt1g=(qtildeM1*360)/8400;
    qt2g=(qtildeM2*360)/8400;

    qpM1 = (q_actualM1 - q_anteriorM1)/h*1000;
    qpM2 = (q_actualM2 - q_anteriorM2)/h*1000;

    //---> Algoritmo de control
    tau1 =  ((kp1N * qtildeM1)/ qd_cuentasM1) -  ((kv1N*qpM1)/9000);
    tau2 =  ((kp2N * qtildeM2)/ qd_cuentasM2) -  ((kv2N*qpM2)/9000);

    if(tau1 > 0 ) {
        //CLOCK
        dir1=0;
        dir2=1;
```

```
        pwm1=tau1;
                 }
    if(tau2 > 0 ) {
        //CLOCK
        dir3=0;
        dir4=1;
        pwm2=tau2;
                 }
    if(tau1 < 0) {
        //CCLOCK
        dir1=1;
        dir2=0;
        pwm1=-tau1;
                }
    if(tau2 < 0) {
        //CCLOCK
        dir3=1;
        dir4=0;
        pwm2=-tau2;
                 }
//pc.printf("\r C1:%f,C2:%f,kp1N:%f,kv1N:%f,kp2N:%f,kv2N:%f,t:%f\n\r",
//qd_cuentasM1,qd_cuentasM2,kp1N,kv1N,kp2N,kv2N,tau2);
pc.printf("%d\n\r",pos1);

     q_anteriorM1 = q_actualM1;
     q_anteriorM2 = q_actualM2;
}

 //----------------------------------------

int main() {
    //-------------------------
//PARÁMETROS DE LA RBFNN
const float PI=3.141592654;

int i;
float w1[9]={-0.0452,0.0486,-0.0452,3.4151,-3.6707,3.4151,0.1363,-0.1465,0.1363};
float w2[9]={0.7750,-0.8330,0.7750,1.0474,-1.1258,1.0474,0.6842,-0.7354,0.6842};
float w3[9]={-0.8801,1.5977,-0.1036,1.3312,-2.7523,2.2006,-0.8801,1.5977,-0.1036};
float w4[9]={-2.4091,4.4133,-2.0002,1.8061,-3.1756,1.8932,-2.4091,4.4133,-2.0002};
float c1r[9]={0.5236,0.5236,0.5236,1.0472,1.0472,1.0472,1.5708,1.5708,1.5708};
float c2r[9]={0.2618,0.5236,0.7854,0.2618,0.5236,0.7854,0.2618,0.5236,0.7854};
float num=-1, den2=0.36, qd1r, qd2r;

//SECCIÓN DE AUTO-SINTONIZACIÓN
          qd1r=qd1g*(PI/180);
          qd2r=qd2g*(PI/180);
```

```
            kp1Ne=0;
            kv1Ne=0;
            kp2Ne=0;
            kv2Ne=0;

    for(i=0; i<=8; i++){
kp1[i]=w1[i]*exp((num/den2)*(((qd1r-c1r[i])*(qd1r-c1r[i]))
+((qd2r-c2r[i])*(qd2r-c2r[i])) ));
kv1[i]=w2[i]*exp((num/den2)*(((qd1r-c1r[i])*(qd1r-c1r[i]))
+((qd2r-c2r[i])*(qd2r-c2r[i])) ));
kp2[i]=w3[i]*exp((num/den2)*(((qd1r-c1r[i])*(qd1r-c1r[i]))
+((qd2r-c2r[i])*(qd2r-c2r[i])) ));
kv2[i]=w4[i]*exp((num/den2)*(((qd1r-c1r[i])*(qd1r-c1r[i]))
+((qd2r-c2r[i])*(qd2r-c2r[i])) ));

            kp1Ne=kp1[i]+kp1Ne;
            kv1Ne=kv1[i]+kv1Ne;
            kp2Ne=kp2[i]+kp2Ne;
            kv2Ne=kv2[i]+kv2Ne;
            }
   //-------------------------
   //pc.printf("\n \r Primer programa!!  ");
    chA1.rise(&interrupcionA1);
    chA1.fall(&interrupcionA1);
    chB1.rise(&interrupcionB1);
    chB1.fall(&interrupcionB1);
    //-------------------------
    chA2.rise(&interrupcionA2);
    chA2.fall(&interrupcionA2);
    chB2.rise(&interrupcionB2);
    chB2.fall(&interrupcionB2);
//-------------------------------------------------
    RtosTimer  control_h(&controlar, osTimerPeriodic);
    control_h.start(h);
//---------------------->START
    while(1) {
        qd1g=30;
        qd2g=20;
        wait(2);
        }
//------------------>STOP
}
```

Apéndice F

Programa en MatLab para la comunicación con el puerto serial

```
%TRANSMISIÓN DE DATOS POR PUERTO SERIAL
%Ing. Donaciano Coyotecatl Cuautle
%Revisión: 10-10-2015

clear all; close all; clc;
s = serial('COM10','BaudRate',9600);
fopen(s);
fgets(s)

%---
figure(1)
%title('Resultados');
%xlabel('muestras');
%ylabel('posición [grados]')
%ylabel('Torque [Nm]')
hold on;
%---
var1=0;
m=30; %número de muestras recibidas
cont=1;

while cont<=m
   xlim([0 cont]);
var1=fscanf(s,'%f');
var2(cont)=(var1*360)/8400; %conversión a [grados]
%-----------------
%Calcular Torque
%var2(cont)=(var1*1.765)/1.1;
%var2(cont)=var1;
%------------------
```

```
plot(var2,'b');
drawnow
cont=cont+1;
end
%------------------------Final
fclose(s);
clear s
```

Apéndice G

Interpolación lineal y cuadrática en MatLab

```
%INTERPOLACIÓN POLINÓMICA DE ORNEN "n"
%Ing. Donaciano Coyotecatl Cuautle
%Revisión: 10-10-2015

clear all; clc; close all;

%Interpolación Lineal
%----------------------
%--->Primer punto de validación
% P=[36,18];         %Posición a interpolar
% qd=[15,30];        %Rango de posiciones qd2
% kp2=[0.45,0.9];    %Rango de valores ganancia kp2
% kv2=[0.35,0.75];    %Rango de valores ganancia kv2
% v=18;              %Valor que se desea interpolar
%----------------------
%--->Segundo punto de validación
%P=[82,42];  %Posición a interpolar
qd=[60,90];   %Rango de posiciones qd1 y qd2
kp1=[2,1];
kv1=[1,0.7];
kp2=[1.3,0.45];
kv2=[0.6,0.35];
v=42;
%-  -   -   -   -   -   -   -   -   -   -   -%
%---->Funciones para interpolación lineal
%Calcula coeficientes
p=polyfit(qd,kp1,1);
%Calcula valor del polinomio para un valor v
f=polyval(p,v)
%-  -   -   -   -   -   -   -   -   -   -   -%
```

```
%-----------------------------------------
%Interpolación cuadrática
%---------------------------
%--->Primer punto de validación
P=[36,18];       %Posición a interpolar
qd=[15,30,45];   %Rango de posiciones qd2 (3 puntos)
kp22=[0.45,0.9,1.1];
kv22=[0.35,0.75,0.6];
v2=18;
%--->Segundo punto de validación
% % P=[82,42];  %Posición a interpolar
% qd=[15,30,45];   %Rango de posiciones qd1 y qd2
% %kp1=no tiene variación
% %kv1=no tiene variación
% kp22=[0.45,0.9,1.1];
% kv22=[0.35,0.75,0.6];
% v2=42;
%-  -   -   -   -   -   -   -   -   -   -   -%
%---->Funciones para interpolación lineal
%Calcula coeficientes
p=polyfit(qd,kp22,2);
%Calcula valor del polinomio para un valor v
f=polyval(p,v2)
%-  -   -   -   -   -   -   -   -   -   -   -%
```

Bibliografía

ABB. http://new.abb.com/products/robotics/es/robots-industriales/irb-460, 2015. Accedido 16-08-2015.

S. Arimoto and M. Takegaki. A new feedback method for dynamic control of manipulators. *J. Dyn. Systems, Measurement Control*, 102:119–125, 1981.

Arm Ltd. http://www.mbed.org/, 2014. Accedido 17-08-2015.

E. Aved'yan. *Learning Systems*. Springer- Verlag, 1995.

A. Barrientos, L. F. Peñin, and C. Balaguer. *Fundamentos de Robótica*. McGraw-Hill, Madrid, ESPAÑA, 1996.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, USA, first edition, 1995.

CNC Software. https://www.mastercam.com/en-us/, 2015. Accedido 16-08-2015.

J. J. Craig. *Introduction To Robotics*. Prentice Hall, USA, tercera edition, 2005.

EvolDesign. http://www.ufactory.cc/uarm, 2015. Accedido 16-08-2015.

L. Fan and E. M. Joo. Linear and nonlinear pd-type control of robotic manipulators for trajectory tracking. In *Industrial Electronics and Applications, 2009. ICIEA 2009. 4th IEEE Conference on*, pages 3442–3447. IEEE, 2009.

S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, second edition, 1999.

J. A. H. Huerta and W. Yu. A modified pd control of robot manipulator using neural network compensator. In *Neural Networks, 1999. IJCNN '99. International Joint Conference on*, volume 3, pages 1999–2004, 1999. doi: 10.1109/IJCNN.1999.832691.

P. Isasi and I. M. Galván. *Redes de Neuronas Artificiales. Un Enfoque Práctico*. 2004.

R. N. Jazar. *Theory of Applied Robotics*. 2010.

S. Kawamura, F. Miyazaki, and S. Arimoto. Is a local linear pd feedback control law effective for trajectory tracking of robot motion? In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 1335–1340 vol.3, Apr 1988.

R. Kelly and R. Carelli. A class of nonlinear pd-type controllers for robot manipulators. *Journal of Robotic Systems*, 13(12):793–802, 1996.

R. Kelly and V. Santibañez. *Control de Movimiento de Robots Manipuladores*. Pearson-Prentice Hall, Madrid, ESPAÑA, 2003.

R. Kelly, V. S. Davila, and J. A. L. Perez. *Control of robot manipulators in joint space*. Springer Science & Business Media, 2005.

F. L.Lewis, D. M.Dawson, and C. T.Abdallah. *Robot Manipulator Control: Theory and Practice*. Marcel Dekker, New York, USA, 2004.

A. Loria and E. Panteley. Force /motion of constrained manipulators without velocity measurements. *IEEE*, 44:1407–1412, 1999. University of california, C.A, USA.

J. Love. *Process Automation Handbook. A Guide to Theory and Practice*. Springer-Verlag, first edition, 2007.

A. Ollero. *Robótica: Manipuladores y robots móviles*. Marcombo, Barcelona, ESPAÑA, 2007.

S. Patarinski and R. Botev. Robot force control: A review. *Pergamon Press Ltd*, 3: 377–398, 1992. Technical University of Sofia, BULGARIA.

Pololu Corporation. `https://www.pololu.com/product/1447`, 2001-2015. Accedido 17-08-2015.

M. Raibert and J. Craig. Hybrid position/force control of manipulators. *ASME, Journal of dynamic systems,measurement, and control*, 102:126–133, 1981. California Institute of Technology, C.A, USA.

F. Reyes. *Robótica: Control de Robots Manipuladores*. Alfaomega, México D.F, MÉXICO, 2011.

F. Reyes. *MatLab: Aplicado a la Robótica y Mecatrónica*. Alfaomega, México D.F, MÉXICO, 2012.

F. Reyes and R. Kelly. Experimental evaluation of model-based controllers on a direct-drive robot arm. *Mechatronics*, 11(3):267–282, 2001.

F. Reyes and A. Rosado. Polynomial family of pd-type controllers for robot manipulators. *Control Engineering Practice*, 13(4):441 – 450, 2005. ISSN 0967-0661. doi: http://dx.doi.org/10.1016/j.conengprac.2004.04.005.

Robert McNeel. `https://www.rhino3d.com/`, 2014. Accedido 16-08-2015.

F. G. Salas, V. Santibanez, and M. A. Llama. Variable gains pd tracking control of robot manipulators: Stability analysis and simulations. In *World Automation Congress (WAC), 2012*, pages 1–6, June 2012.

V. Santibañez and R. Kelly. Pd control with feedforward compensation for robot manipulators: analysis and experimentation. *Robotica*, 19:11–19, 1 2001. ISSN 1469-8668. doi: 10.1017/S0263574700002848.

V. Santibanez, R. Kelly, and F. Reyes. A new set-point controller with bounded torques for robot manipulators. *Industrial Electronics, IEEE Transactions on*, 45(1):126–133, Feb 1998. ISSN 0278-0046. doi: 10.1109/41.661313.

B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: Modelling, Planning and Control*. Springer-Verlag, LONDON, 2009.

P. Sánchez. Control cartesiano de robots manipuladores. Master's thesis, Benemérita Universidad Autónoma de Puebla, BUAP, MÉXICO, Mayo 2005.

M. W.Spong and M.Vidyasagar. *Robot Modeling and Control*. John Wiley & Sons,Inc, USA, 2006.

M. W.Spong, S. Hutchinson, and M.Vidyasagar. *Robot Dynamics and Control*. John Wiley & Sons,Inc, USA, 2004.

D. Xia, L. Wang, and T. Chai. Neural-network-friction compensation-based energy swing-up control of pendubot. *Industrial Electronics, IEEE Transactions on*, 61(3): 1411–1423, March 2014. ISSN 0278-0046. doi: 10.1109/TIE.2013.2262747.

W. Yu and X. Li. Pd control of robot with velocity estimation and uncertainties compensation. *International Journal of Robotics and Automation*, 21(1):1–9, 2001.