

Instituto Tecnológico de Costa Rica  
Área Académica de Ingeniería Mecatrónica



Centro Superior de Investigaciones Científicas (CSIC)

Instituto Cajal

Grupo de Neuro-Rehabilitación



**“Control en trayectoria multi-articular para una plataforma humanoide.”**

Proyecto para obtener el grado de:  
Licenciado en Ingeniería Mecatrónica.

Mario Alberto Ríos Mora

Carnet: 201281252

**Supervisores:**

Prof. José Luis Pons Rivera

Dr. José E. González Vargas

**Tribunal Evaluador:**

Juan Luis Crespo Mariño

Marta Vílchez Monge

Aníbal Coto Cortés

Madrid, España

2016

## **DECLARACIÓN DE AUTENCIDAD**

Declaro que el presente Proyecto de Graduación ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conceptos propios.

En los casos que he utilizado bibliografía, he procedido a indicar las fuentes mediante las respectivas citas bibliográficas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

Cartago, 17 de enero del 2017.



Mario Alberto Ríos Mora

1-1513-0872

# APROBACIÓN

**INSTITUTO TECNOLÓGICO DE COSTA RICA**  
**CARRERA DE INGENIERÍA MECATRÓNICA**  
**PROYECTO DE GRADUACIÓN**  
**ACTA DE APROBACIÓN**

Proyecto de Graduación defendido ante el presente Tribunal Evaluador como requisito para optar por el título de Ingeniero en Mecatrónica con el grado académico de Licenciatura, del Instituto Tecnológico de Costa Rica.


Miembros del Tribunal

  
M.Sc. Marta Eugenia Vilchez  
Monge.

Profesor lector

\_\_\_\_\_  
M.Sc. Aníbal Coto Cortés.

Profesor lector

  
\_\_\_\_\_  
Ph.D. Juan Luis Crespo Marino.  
Profesor asesor

Los miembros de este Tribunal dan fe de que el presente trabajo de graduación ha sido aprobado y cumple con las normas establecidas por la Carrera de Ingeniería Mecatrónica

Cartago, 17 de enero del 2017.

## DEDICATORIA

Ante todo, deseo dedicar esta tesis a mis padres Erick Rios Ruh e Ileana Mora González, gracias por su esfuerzo y dedicación en hacer de mí una persona de bien, por todo lo que han hecho por mí, les estaré por siempre agradecido.

A todos mi amigos y compañeros del ITCR, el camino fue largo y difícil, pero se disfrutó con cada paso. No puedo imaginar a un mejor grupo de personas con las que hubiera deseado pasar todos estos años de formación.

A mi tutor José González por darme la oportunidad de realizar esta tesis, por sus palabras de apoyo cuando creí que no lo lograría y por su guía a lo largo del desarrollo del proyecto.

Y, por último, a mi novia Kathy.

Uno pertenece donde:

Se le quiere

Se le ama

Se le desea

Se le siente...

Y en repetidas

Ocasiones, se le

Extraña pero

Sobre todo

Donde se le

Demuestra.

Te amo.

## RECONOCIMIENTOS

Quiero agradecer de corazón a todos aquellos que de alguna forma u otra me brindaron su apoyo, ayuda y consejo durante todo este proceso.

A mi tribunal evaluador: Juan Luis Crespo, Marta Vílchez y Aníbal Coto.

A Rosa Elena Leiva de la VIESA por su comprensión y deseos de ayudarme.

A Hannia Álvarez de la Escuela de Mantenimiento Industrial.

A mi Tío Pedro por su ayuda y por los buenos momentos que compartimos en Madrid.

A todos en el grupo de Neuro Rehabilitación del Instituto Cajal por recibirme con los brazos abiertos, por su amistad, por los buenos momentos y por ayudarme cuando lo necesité.

A todos ustedes, gracias infinitas.

## RESUMEN

El presente documento abarca el desarrollo de una arquitectura de comunicación y de control para los actuadores de rigidez variable. Dichos actuadores se encuentran en las articulaciones inferiores de un robot bípedo llamado Binocchio y desarrollado por el grupo de Neuro-rehabilitación del Instituto Cajal. El diseño de dicho robot aplica principios biológicos clave presentes en los seres humanos como lo es la elasticidad y rigidez. Estos fueron implementados por medio de actuadores de rigidez variable (VSA) que poseen un muelle como elemento elástico. El comportamiento del bípedo depende en gran medida de una buena comunicación y de un adecuado esquema de control que permita ejecutar las trayectorias deseadas en cada articulación. Sin embargo, problemas a nivel del bus de comunicación entre los dispositivos de control y la simplicidad de los controladores PID para el control de los VSA comprometen el correcto funcionamiento del robot. Por ello en una primera instancia se trabajó en encontrar el origen de los problemas de comunicación para luego enfocarse en adaptar un método bio-inspirado de control conocido como Feedback Error Learning (FEL) que utiliza una red neuronal para aprender el modelo inverso de una determinada planta y de esta forma mejorar el esquema de control. Del mismo modo se procedió a realizar pruebas de comunicación y de control para corroborar y validar las soluciones planteadas. Se determinó que existía una saturación sobre el único buffer de transmisión utilizado, por lo que se incorporó el uso de los buffers restantes. Pruebas de comunicación arrojaron una reducción importante tanto en la cantidad como en la magnitud de los errores antes existentes. Por otro lado, fue posible adaptar el FEL para el control de los VSA, lo que incidió en una mejora significativa en el rendimiento de los controladores de trayectoria. Pruebas de robustez y de estabilidad permitieron validar el uso del FEL como una alternativa viable para el control del bípedo.

## INDICE DE FIGURAS

Figura 1-1: Plataformas robóticas: ASIMO, Valkyrie, ATLAS y COMAN. ....	1
Figura 1-2: Actuadores en articulaciones del robot Binocchio [1] .....	3
Figura 1-3: Arquitectura distribuida de control del Binocchio[1] .....	4
Figura 2-1: Modelo CAD y resultado final[1]. ....	7
Figura 2-2: Miembros inferiores y grados de libertad para cada articulación[1]. ....	8
Figura 2-3: Actuador MACCEPA en la rodilla del bípodo[1]. ....	9
Figura 2-4: Modelo CAD de los SPU[1] .....	9
Figura 2-5: Concepto de funcionamiento del actuador MACCEPA[8]. ....	10
Figura 2-6: Diagrama esquemático del actuador MACCEPA[9] .....	11
Figura 2-7: Funcionamiento del actuador MACCEPA[8]. ....	12
Figura 2-8: Neurona artificial [10].....	13
Figura 2-9: Funciones de activación [10] .....	14
Figura 2-10: Perceptron Multicapa [11]. ....	15
Figura 2-11: Principio del algoritmo de gradiente descendente[11].....	16
Figura 2-12: Concepto biológico sobre el que se basa el FEL.....	18
Figura 2-13: Esquema de control FEL.[15] .....	18
Figura 2-14: Bus CAN[21] .....	20
Figura 2-15: Diagrama de flujo del protocolo CAN.[22] .....	21
Figura 2-16: Composición de un paquete de datos CAN [23]. ....	21
Figura 2-17: Arbitración en un bus CAN [21].....	22
Figura 2-18: Lógica inversa del bus CAN [21] .....	22
Figura 2-19: Velocidades de transmisión y longitud del bus [21] .....	23
Figura 2-20: Mascaras y filtros para recepción de mensajes[23].....	23
Figura 3-1: División de paquetes enviados[5] .....	24
Figura 3-2: Conformación de los identificadores[5]. ....	24
Figura 3-3: Identificadores para cada SPU[5]. ....	25
Figura 3-4: Prueba de comunicación realizada [25]. ....	26
Figura 3-5: Variaciones del encoder alrededor de la posición de equilibrio (0°)[25]. ....	26
Figura 3-6: Errores en señal senoidal del paquete 1[25]. ....	27
Figura 3-7: Errores en paquete 1 [25]. ....	27
Figura 3-8: Errores presentes en señal senoidal del paquete 2[25]. ....	28
Figura 3-9: Problema adicional presente en paquete 2[25].....	28
Figura 3-10: Criterio de clasificación de muestras [25]. ....	29
Figura 3-11: Porcentaje de error, ECM y variabilidad en paquetes 1 y 2. ....	30
Figura 3-12: Flujo del programa al transmitir datos [25]. ....	31
Figura 3-13: Resumen de tiempos de ejecución de tareas en el SPU[27].....	32
Figura 3-14: Diagrama de flujo de la función can_send () modificada [25]. ....	34
Figura 3-15: Porcentaje de error, ECM y variabilidad en paquetes 1 y 2 después de implementada la solución. ....	36
Figura 3-16: Porcentaje de error, ECM y variabilidad en paquetes 1 y 2 de cada SPU .....	39
Figura 4-1:Correa de Kevlar a nivel de la rodilla del Binocchio[25]. ....	40
Figura 4-2: Respuesta en lazo abierto del sistema.[25] .....	42
Figura 4-3:Mediciones para modelado empírico[25]. ....	42
Figura 4-4: Porcentaje de ajuste de modelos obtenidos[25]. ....	43
Figura 4-5: Función de activación en la capa oculta [10]. ....	45
Figura 4-6: Péndulo invertido construido [25]. ....	46
Figura 4-7: Controlador PID y compensación de gravedad [25]. ....	47
Figura 4-8: Torque debido al peso del péndulo[25]. ....	47
Figura 4-9: Control de trayectoria por PID [25]. ....	48
Figura 4-10: Control de trayectoria utilizando el FEL [25]. ....	49

Figura 4-11: Variación del error RMS en el control del péndulo invertido [25].	50
Figura 4-12: Trayectoria obtenida con FEL entrenado estocásticamente [25]	51
Figura 4-13: Variación del error luego de modificar el método de entrenamiento[25].	52
Figura 4-14: Señales de control antes y después del entrenamiento [25].	54
Figura 4-15: Trayectoria mediante PID [25].	55
Figura 4-16: Trayectoria del actuador mediante FEL [25].	56
Figura 4-17: Error RMS en la trayectoria efectuada por la rodilla [25].	57
Figura 4-18: Variación del valor medio de señal de control Ufb [25].	57
Figura 4-19: Variación de señales de control durante el entrenamiento[25].	59
Figura 4-21: Perturbación estática [25].	60
Figura 4-22: Variación del error RMS debida a carga de 3 kg [25].	61
Figura 4-23: Comportamiento del sistema ante perturbaciones transitorias [25].	63
Figura 4-24: Valor del error RMS durante prueba con perturbaciones [25].	64
Figura 4-25: Ruido en lectura del sensor[25].	64
Figura 4-26: Comparativa de control de trayectoria entre PID y FEL.	65
Figura 4-27: Variación del valor RMS del error debida a ruido en sensores [25].	66
Figura 4-28: Trayectoria senoidal con variación de amplitud.	67
Figura 4-29: Variación del error RMS en prueba de estabilidad [25].	67
Figura 4-30: Variación del aporte del PID y MLP durante prueba de estabilidad [25].	68
Figura 4-31: Trayectoria sagital de la rodilla durante la marcha [25].	68
Figura 4-32: Trayectoria sagital de la rodilla por PID[25].	69
Figura 4-33: Trayectoria sagital de la rodilla por FEL[25].	69
Figura 4-20: Oscilación en los valores de los pesos de la capa de salida[25]	71
Figura A-1 Código función can_send () Buffer 0 [25].	a
Figura A-2: Código función can_send () Buffer 1 [25].	a
Figura A-3: Código función can_send () Buffer 2[25].	b
Figura A-4: Inicialización de Variables [25]	b
Figura A-5: Calculo de % de aporte de señal Ufb [25].	c
Figura A-6: Función para generar trayectorias senoidales modificables [25].	c
Figura A-7: Red Neuronal MLP [25].	d
Figura A-8: Envío y recepción de datos para control del actuador MACCEPA [25].	e
Figura A-9: Control PID, Red neuronal y compensación de gravedad [25].	f
Figura A-10: Generación de trayectoria senoidal [25].	f
Figura A-11: Dimensiones del motor[30].	g
Figura A-12: Constantes eléctricas y mecánicas[30].	g
Figura A-13: Especificaciones del motor [30].	h
Figura A-14: Pines y componentes [5].	h
Figura A-15: Pines y conexiones [31].	i



## INDICE DE TABLAS

TABLA 3.1: INFORMACIÓN DE LOS PAQUETES ENVIADOS POR LA SPU [25] .....	25
TABLA 3.2: PORCENTAJE DE ERROR Y ERROR CUADRÁTICO MEDIO DE CADA PAQUETE[25]. ...	29
TABLA 3.3: ESTADÍSTICA DE LA PRUEBA REALIZADA[25].....	30
TABLA 3.4: PRIORIDAD DE BUFFERS DE TRANSMISIÓN [25].....	33
TABLA 3.5: DATOS OBTENIDOS LUEGO DE IMPLEMENTAR LA SOLUCIÓN[25].....	35
TABLA 3.6: ESTADÍSTICA DESPUÉS DE IMPLEMENTAR LA SOLUCIÓN[25]. .....	36
TABLA 3.7: IDENTIFICADORES PARA CADA NODO[25].....	37
TABLA 3.8: DATOS PARA CADA SPU EN PRUEBA DE MAYOR EXIGENCIA [25] .....	37
TABLA 3.9: ESTADÍSTICA DE PRUEBA DE MAYOR EXIGENCIA[25]. .....	38
TABLA 4.1: ZONA MUERTA SEGÚN A POSICIÓN DEL ACTUADOR (P=38.57MM)[25]. .....	41
TABLA 4.2: PARÁMETROS DEL MLP PARA EL CONTROL DEL PÉNDULO. ....	49
TABLA 4.3: COMPARACIÓN DEL ERROR Y EL RETRASO ENTRE PID Y FEL [25]. .....	52
TABLA 4.4: COMPARACIÓN ENTRE EL APOORTE DEL PID Y EL MLP A LA SEÑAL DE CONTROL PARA CADA ESQUEMA DE CONTROL [25]. .....	53
TABLA 4.5: PARÁMETROS DEL MLP PARA EL CONTROL DEL ACTUADOR [25].....	55
TABLA 4.6: ERROR Y EL RETRASO OBTENIDOS POR PID Y FEL PARA CONTROL DEL ACTUADOR [25]. .....	58
TABLA 4.7: APOORTE DEL PID Y EL MLP A LA SEÑAL DE CONTROL DEL ACTUADOR.....	59
TABLA 4.8: ERROR Y RETRASO EN PRUEBA DE CAMBIO DE DINÁMICA [25]. .....	62
TABLA 4.9: APOORTE DEL PID Y EL MLP A LA SEÑAL DE CONTROL LUEGO DEL CAMBIO DE DINÁMICA [25]. .....	62
TABLA 4.10: COMPARATIVA ENTRE PID Y FEL ANTE RUIDO EN LOS SENSORES [25] .....	65
TABLA 4.11: COMPARATIVA DE RENDIMIENTO ENTRE PID Y FEL[25]. .....	70

# TABLA DE CONTENIDO

DECLARACIÓN DE AUTENCIDAD .....	i
APROBACIÓN.....	ii
DEDICATORIA .....	iii
RECONOCIMIENTOS .....	iv
RESUMEN .....	v
1. CAPITULO 1: Introducción. ....	1
1.1. Contexto del proyecto. ....	1
1.2. Definición del problema.....	3
1.2.1 Generalidades.....	3
1.2.2 Síntesis del problema. ....	4
1.3. Justificación. ....	4
1.4. Enfoque de la solución.....	5
1.5. Objetivos .....	5
1.5.1 Objetivo general.....	5
1.5.2 Objetivos específicos. ....	5
2. CAPITULO 2: Marco Teórico.....	7
2.1. Robot bípedo Binocchio.....	7
2.2. Actuador MACCEPA .....	10
2.3. Redes neuronales (Perceptrón Multicapa) .....	12
2.4. Feedback Error Learning .....	17
2.5. Protocolo de comunicación CAN (Control Area Network) .....	20
3. CAPITULO 3: Estrategia de Comunicación.....	24
3.1 Configuración actual .....	24
3.2 Problemática. ....	25
3.3 Estudio del problema .....	29
3.4 Solución planteada. ....	31
3.5 Resultados.....	35
3.6 Conclusiones.....	38
4. CAPITULO 4: Estrategia de control.....	40
4.1 Caracterización del actuador.....	40
4.2 Solución planteada .....	44
4.3 Control de péndulo invertido .....	45
4.3.1 Descripción de la planta.....	45
4.3.2 Control Clásico. ....	46
4.3.3 Controlador FEL.....	48

4.3.4	Análisis de Resultados. ....	52
4.4	Control del actuador de rigidez variable. ....	54
4.4.1	Control Clásico. ....	54
4.4.2	Controlador FEL. ....	55
4.4.3	Análisis de Resultados. ....	58
4.5	Pruebas para validación del FEL. ....	59
4.5.1	Pruebas de robustez.....	60
4.5.2	Prueba de estabilidad. ....	66
4.5.3	Prueba extra: Trayectoria Sagital de la rodilla. ....	68
4.5.4	Limitaciones del FEL.....	70
5.	CAPITULO 5: Conclusiones y trabajo futuro. ....	71
5.1.1	Conclusiones. ....	71
5.1.2	Trabajo futuro. ....	72
6.	Referencias.....	74
A.	Anexos .....	a
A1.	Código en C y MATLAB.....	a
A1.1	Código función cand_send () modificada. ....	a
A1.2	Código Función Early Stop.....	b
A1.3	Código Función para generar trayectoria senoidal variable.....	c
A2.	Modelos de Simulink. ....	d
A2.1	Bloque: Red Neuronal.....	d
A2.2	Modelo Simulink: Control desde ordenador. ....	e
A2.3	Modelo Simulink: Control en PC-104. ....	f
A3.	Hojas de Datos. ....	g
A3.1	Motor del actuador del Binnochio.....	g
A4.	Diagramas de componentes .....	h
A4.1	Unidad Secundaria de Procesamiento.....	h
A4.2	Driver ESCON 50/5.....	i
A5.	Derivación de algoritmo de backpropagation para un MLP de 3 capas.....	j
A6.	Carta de aceptación .....	m
A7.	Hoja de proyecto .....	n

## 1. CAPITULO 1: Introducción.

Este capítulo presenta el contexto en el cual se enmarca el proyecto mostrando su importancia e impacto. Por otro lado, se presenta la problemática, el enfoque de la solución y se concluye con los objetivos planteados.

### 1.1. Contexto del proyecto.

La excepcional habilidad humana para caminar de forma eficiente y estable es el resultado de una compleja interacción de mecanismos biomecánicos, neuronales y cognitivos. La forma en que se produce esta interacción y la contribución de cada uno de estos mecanismos para un correcto desarrollo de la marcha humana suponen la base del desarrollo de robots biológicamente inspirados [1].

Los robots biomiméticos o biológicamente inspirados, como su nombre lo implica, imitan la estructura y movimiento de los seres humanos y animales. Esto se traduce en consideraciones biomecánicas y antropométricas como cantidad de grados de libertad (en los planos sagital, transversal y frontal), así como peso y dimensiones de las articulaciones. Del mismo modo los sistemas de actuación tienen una gran influencia en el comportamiento del robot, por lo que deben ser capaces de imitar el comportamiento dinámico de sus contrapartes biológicas, lo que a su vez supone toda una nueva serie de consideraciones desde el punto de vista de control y del tipo de actuador a utilizar.

En la actualidad se cuenta con plataformas robóticas muy avanzadas que incorporan en alguna u otra medida dichas consideraciones biomiméticas tales como ATLAS de Boston Dynamics, ASIMO de Honda, Valkyrie de la NASA o el COMAN del ITT (Instituto Italiano de Tecnología) por citar algunos de los ejemplos más representativos

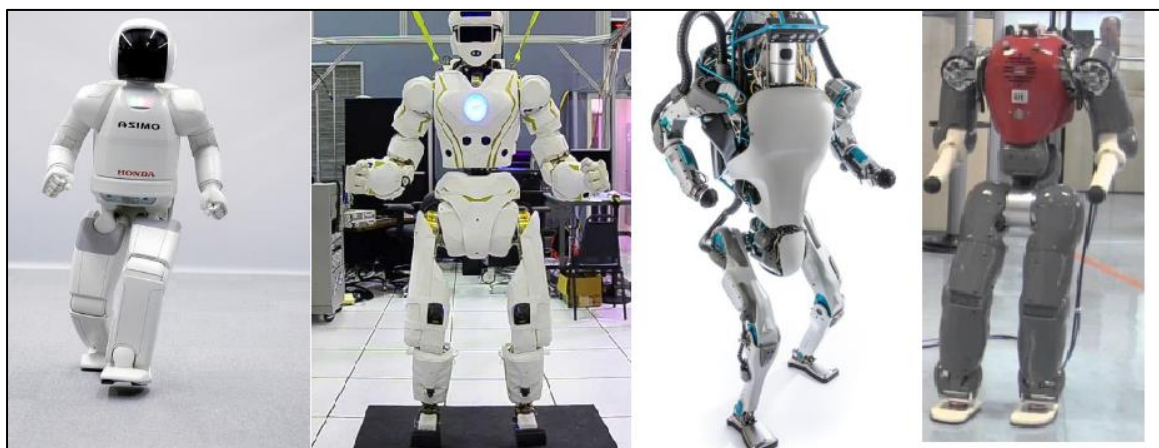


Figura 1-1: Plataformas robóticas: ASIMO, Valkyrie, ATLAS y COMAN.

Como se puede observar en la Figura 1-1 dichas plataformas cumplen desde el punto de vista antropomórfico con la definición de un robot humanoide. No obstante, cada uno incorpora diferentes tipos de actuadores, por ejemplo, ATLAS cuenta con actuadores hidráulicos mientras que ASIMO utiliza actuadores eléctricos [2]. Cada uno presenta sus ventajas y desventajas, sin embargo, la estructura de un actuador eléctrico es simple si la comparamos con la de los actuadores hidráulicos y neumáticos, ya que éstos sólo requieren de energía eléctrica como fuente de potencia [1].

No obstante, los actuadores eléctricos tienen un comportamiento rígido por lo que requieren de estrategias de control relativamente complejas para emular el comportamiento dinámico de los sistemas de actuación biológico, ya que la marcha humana es el resultado de la interacción de diferentes mecanismos biomecánicos, como por ejemplo la visco-elasticidad de los músculos. La especialización bípeda de los seres humanos con sus adaptaciones morfo-funcionales, ha estimulado nuevos enfoques basados en el comportamiento elástico de sus articulaciones.[3]

Considerando lo anterior se han desarrollado actuadores que pretenden emular la característica elástica de las articulaciones humanas [4]. El concepto básico radica en incluir un elemento elástico (generalmente un resorte) en serie con un actuador rígido. Dichos actuadores se denominan Series Elastic Actuators (SEA), cuando la rigidez de la articulación permanece constante (seleccionada en la fase de diseño), o Variable Stiffness Actuators (VSA) que permite la variación y el control de la rigidez de forma continua [4].

Dentro del proyecto Europeo H2R - “Integrative Approach for the Emergence of Human-Like Robotic Locomotion”[5] - se han desarrollado diferentes plataformas que aplican principios biológicos clave presentes en los humanos (diseño bio-inspirado) en un robot humanoide llamado Binocchio. El Binocchio está compuesto por 17 grados de libertad que conforman el tronco y los miembros inferiores del humanoide (Figura 1-2). El plano sagital está conformado por siete actuadores basados en el concepto del MACCEPA. Este concepto describe un actuador elástico de rigidez variable (VSA – Variable Stiffness Actuators)[4],el cual pretende replicar algunas propiedades de las articulaciones humanas mediante la variación de la rigidez. Por ejemplo, almacenamiento y liberación de energía en los músculos, o una instantánea reacción flexible en el caso de producirse impactos.

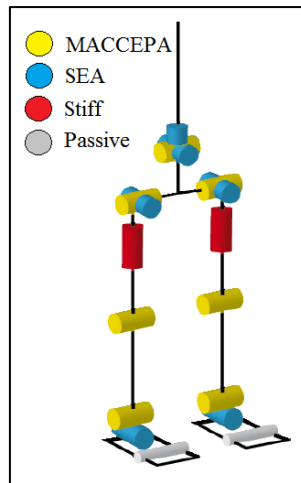


Figura 1-2: Actuadores en articulaciones del robot Binocchio [1]

Pese a sus ventajas biomecánicas, el uso de los actuadores MACCEPA en el Binocchio supone a su vez toda una nueva serie de retos desde el punto de vista de control, ya que el elemento elástico y su estructura mecánica aportan una mayor complejidad y no linealidad al actuador. Una primera versión de la plataforma ya ha sido desarrollada[1], pero actualmente se está trabajando en una optimización de los protocolos de comunicación y de control de los actuadores. Estas optimizaciones se están realizando a nivel de los dispositivos de control para garantizar un flujo y un control en trayectoria robusto y eficiente.

## 1.2 Definición del problema.

### 1.2.1 Generalidades.

Como se muestra en la Figura 1-3, el Binocchio presenta una arquitectura distribuida donde la unidad de procesamiento central (CPU) se encarga de realizar el procesamiento de alto nivel que se utiliza para generar las trayectorias deseadas de cada articulación del robot. Estas trayectorias se envían vía CAN-BUS a las unidades de procesamiento secundario (SPU) de cada articulación. Cada SPU tiene programado un control a lazo cerrado que permite seguir una trayectoria de posición. A su vez, las SPU envían datos de los sensores y los parámetros de control a la CPU, los cuales se utilizan para monitorizar el comportamiento de cada actuador. Debido a que el comportamiento del bípedo depende en gran medida de una buena comunicación, es importante garantizar que no haya errores o pérdidas en los datos transmitidos. Sin embargo, en el sistema actual, cuando se utilizan dos o más nodos, se han notado pérdidas de datos y ruido en el bus que influyen a la inestabilidad de los controladores de trayectoria actuales.

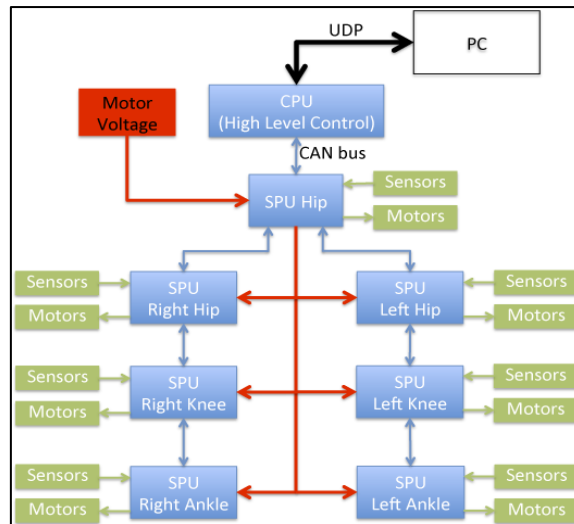


Figura 1-3: Arquitectura distribuida de control del Binocchio[1]

Por otro lado, para controlar los actuadores del Binocchio se están utilizando controladores PID clásicos con compensación de la gravedad, los cuales no son suficientes para realizar un control de trayectoria robusto y eficiente. Estas limitaciones se deben a que el PID es un control lineal que no es capaz de tomar en cuenta las no-linealidades intrínsecas del actuador debido a su estructura mecánica y a su naturaleza elástica. Esto resulta problemático ya que, pese a que la estructura mecánica del actuador puede compensar por algunas perturbaciones o diferentes condiciones que se puedan encontrar, es necesario que el esquema de control sea capaz de adaptarse y aprender cuando sea necesario.

### 1.2.2 Síntesis del problema.

Desarrollo de una arquitectura de comunicación y de control para los actuadores de rigidez variable de las articulaciones inferiores del robot bípedo Binocchio

### 1.3 Justificación.

El robot humanoide Binocchio sobre el que se trabajará constituye el estado del arte en el diseño de robots bípedos biológicamente inspirados. Debido al empleo de actuadores de rigidez variable, dicha plataforma robótica presenta propiedades elásticas en sus articulaciones con el fin de emular algunas de las propiedades de músculos en los seres humanos. De esta forma se puede estudiar la influencia de la variación de la rigidez de las articulaciones en el comportamiento dinámico de los miembros inferiores durante la marcha, en muy diversos escenarios y bajo diversas condiciones de operación[6].

Se espera que los resultados de dichos estudios, así como la implementación de actuadores de rigidez variable ayuden a la creación de nuevos robots con comportamientos dinámicos más naturales y seguros para la interacción humano-robot. El conocimiento desarrollado en robots bípedos, así como el uso de los actuadores de rigidez variable son aplicables en prótesis de pierna por citar un ejemplo[4]. Por otro lado, fomenta incluso la mejora en el diseño de sistemas de asistencia de la marcha para personas con problemas de movilidad, como exoesqueletos y aparatos de rehabilitación.

Por ello se debe asegurar el correcto funcionamiento de los diferentes actuadores ya que de esto depende el correcto comportamiento dinámico necesario para el desarrollo de dicha tarea. El desarrollo y la explotación de este tipo de actuadores dará paso a nueva generación de robots que pueden coexistir y cooperar con la gente y llegar a estar más cerca de lo que se esperaría en cuanto a manipulación y locomoción de los seres humanos.[1]

#### **1.4 Enfoque de la solución.**

Se deben desarrollar diferentes estrategias tanto en el CPU como en las SPU que compensen o detecten la pérdida de datos o el retraso en el envío de datos, especialmente cuando hay varios nodos enviando y recibiendo dentro del mismo bus de datos. Del mismo modo se debe mejorar el esquema de control actual implementando un control adaptativo robusto que sea capaz de aprender la dinámica del actuador en diferentes condiciones. Para esto, se propone utilizar el método bio-inspirado llamado Feedback Error Learning [7] utilizando una Red Neural Artificial, que a diferencia de estrategias de control no-lineal basadas en modelos matemáticos, aprenda la dinámica del actuador en tiempo real, adaptándose así a diferentes condiciones de trabajo.

#### **1.5 Objetivos**

##### **1.5.1 Objetivo general**

Optimizar los protocolos de comunicación y de control de los actuadores de rigidez variable de los miembros inferiores del robot bípedo Binocchio.

##### **1.5.2 Objetivos específicos.**

El presente trabajo se divide en dos áreas específicas. La primera consiste en optimizar los protocolos de comunicación actuales de la plataforma para garantizar un flujo de datos



robusto. La segunda consiste en implementar un control adaptativo para los actuadores de rigidez variable de los miembros inferiores del robot bípedo Binocchio. Por ello se han desglosado los siguientes objetivos específicos, dos para la parte de comunicación y dos para la parte de control:

- Desarrollar estrategias de comunicación entre las unidades de procesamiento secundario (SPU) y la unidad de procesamiento central (CPU).  
Contemplando el correcto funcionamiento del bus de comunicación cuando se tienen de dos a cinco nodos enviando y recibiendo datos en el mismo bus de comunicación.
- Diseñar algoritmos de arbitraje de datos.  
De manera a disminuir y/o compensar ruido y pérdida de datos que comprometen el rendimiento de los controladores de trayectoria.
- Mejorar el rendimiento de los controladores de trayectoria actuales.  
Mediante la adaptación del algoritmo de Feedback Error Learning para el actuador del Binocchio y comparado su rendimiento con respecto al control PID clásico.
- Ejecutar pruebas de rendimiento para el controlador propuesto.  
Por medio de pruebas (de robustez y estabilidad) que permitan validar la adaptación de la estrategia de control propuesta y comparando su rendimiento con respecto al control PID clásico.

## 2. CAPITULO 2: Marco Teórico.

Este capítulo sirve de punto de partida en la comprensión de los problemas anteriormente planteados. Se realiza una descripción general del robot Binocchio y haciendo especial énfasis en el actuador sobre el que se trabajará, así mismo se explica la teoría relacionada con redes neuronales y su uso en el algoritmo de Feedback Error Learning. Finalmente se realiza una pequeña introducción al protocolo CAN exponiendo sus principales características.

### 2.1. Robot bípedo Binocchio.

Como se mencionó anteriormente Binocchio es un robot bípedo con un diseño bioinspirado, está compuesto de miembros inferiores, pelvis y tronco superior, todos ellos actuados dada su importancia durante el proceso de la marcha humana mientras que los hombros, brazos y la cabeza se han añadido con propósitos estéticos. Con una altura de 1,70m y un peso de 40kg, el robot dispone de una morfología y distribución de pesos similar a la de los seres humanos, de manera que el comportamiento dinámico del robot al caminar se asemeje a estos. En la Figura 2-1 se puede apreciar al robot en cuestión.

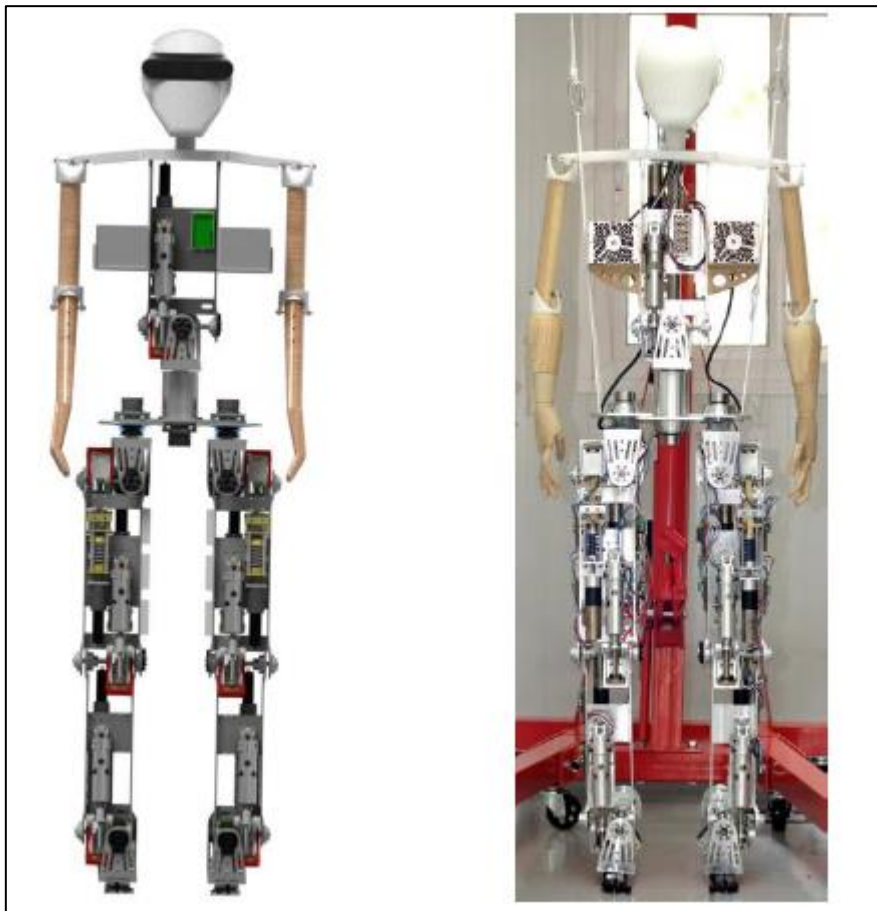
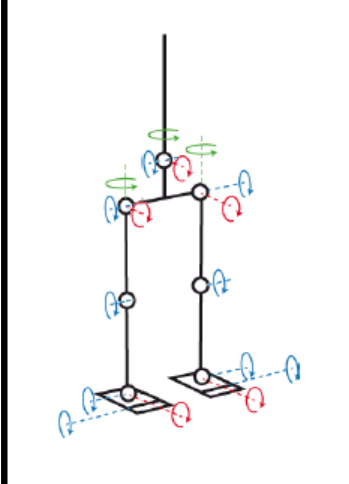


Figura 2-1: Modelo CAD y resultado final[1].

El bípedo cuenta con un total de 17 grados de libertad (DoFs por sus siglas en inglés) actuados por medio de 22 motores, de estos DoFs, 15 son activos, mientras que los 2 restantes son considerados pasivos. La Figura 2-2 resume los grados de libertad que se consideraron imprescindibles de implementar en el bípedo, así como sus rangos de movimientos y el papel que juegan en cuanto a variación de la rigidez.



Articulación	DoF	Rango de movimiento	Papel
Tobillo	Sagital	30°	Activo+Flexible
	Frontal	10°	Activo+Flexible
Rodilla	Sagital	70°	Activo+Flexible
Cadera	Sagital	40°	Activo+Flexible
	Frontal	10°	Activo+Flexible
	Transversal	8°	Activo
Cintura	Sagital	4°	Activo+Flexible
	Frontal	4°	Activo+Flexible
	Transversal	10°	Activo+Flexible
Pie	Sagital	55°	Pasivo

Figura 2-2: Miembros inferiores y grados de libertad para cada articulación[1].

Se constata que el plano sagital es el plano de mayor importancia (esencial para la marcha) por lo tanto, se encuentran presentes en la mayoría de robots bípedos[1]. La variación de la rigidez (comportamiento flexible) es una propiedad que poseen la mayoría de articulaciones biológicas y que permite un rápido almacenamiento y liberación de energía en los músculos, y una instantánea reacción flexible en el caso de producirse impactos. La variación de la rigidez para la locomoción de robots bípedos basados en la marcha en humanos permite emular la co-contracción de los músculos humanos y modificar la velocidad y la longitud del paso durante la marcha.

Por todos estos motivos y la naturaleza del diseño bio-inspirado se incluyeron estas propiedades en las articulaciones del bípedo. Para ello existen los denominados actuadores de rigidez variable (VSA)[4]. Éstos se diferencian de los actuadores rígidos (formados por un motor y una reductora), debido a la inclusión de un elemento elástico entre el motor y el eje de salida, añadiendo propiedades elásticas a las articulaciones donde se implementan, mejorando su comportamiento dinámico. Los actuadores MACCEPA (Figura 2-3) implementados (Mechanically Adjustable and Compliance Equilibrium Position Actuators) constituyen un subgrupo dentro de los VSA, y se basan en el control de la rigidez de forma mecánica [6].

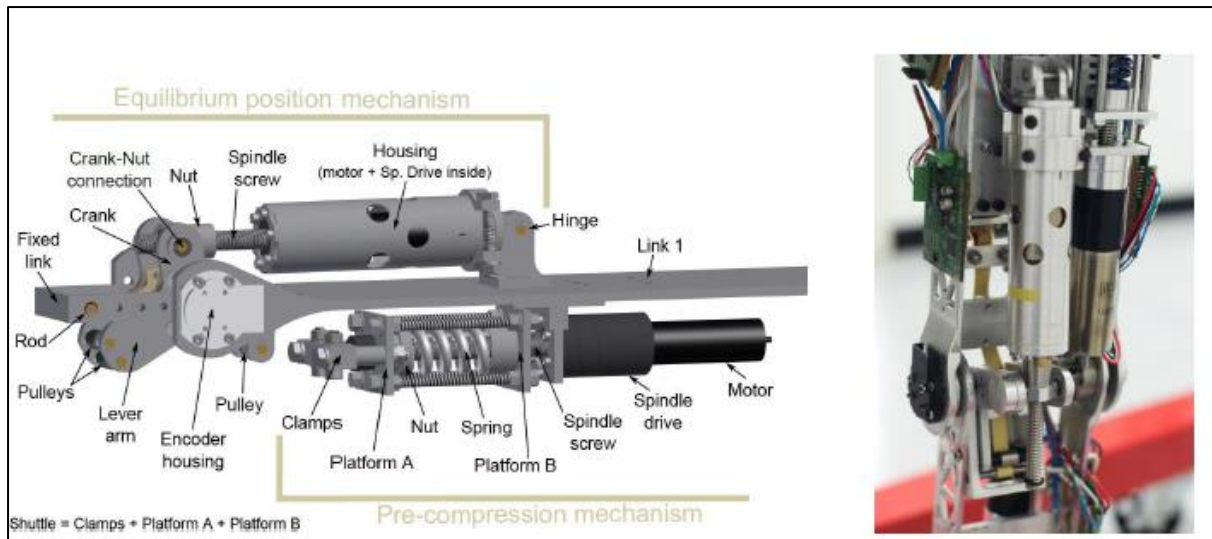


Figura 2-3: Actuador MACCEPA en la rodilla del bípedo[1].

Cada uno de estos actuadores tiene un controlador local (SPU) específicamente diseñado para cumplir con los requerimientos específicos de los mismos (Figura 2-4). Cada SPU es empleada para monitorizar un motor y sus sensores correspondientes, de manera que se encarga de generar las corrientes necesarias para actuar los motores (MAXON EC-4Pole 200W), así como leer los diferentes sensores del actuador. Los actuadores MACCEPA que utiliza el robot Binocchio requieren cada uno de 3 sensores: 2 encoders magnéticos rotatorios y 1 célula de carga (tanto para al actuador sagital como al frontal).



Figura 2-4: Modelo CAD de los SPU[1]

Los SPU están compuestos por un microcontrolador (Microchip DSPIC30F4011) y los drivers MAXON EC-50/5. Estos están dentro de una red distribuida comandados y coordinados por una unidad central de procesamiento (CPU) compuesta por un ordenador de mesa corriendo Matlab Simulink (Figura 1-3). La CPU se encarga de generar, coordinar y enviar los movimientos articulares que generan los diferentes comportamientos del robot. También recibe los datos de los sensores en cada una de las articulaciones.

El controlador de los actuadores está basado en el modelo PID (Proporcional-Integral-Derivativo). El modelo de controlador PID se implementó debido a su amplio uso en robótica y su relativa sencillez.

## 2.2. Actuador MACCEPA

El concepto sobre el cual se desarrollaron los actuadores MACCEPA (descrito con más detalle en [1], [6], [8]) se puede apreciar en la Figura 2-5. Se considera una barra conectada a una superficie fija mediante un pasador, por lo que esta puede rotar libremente alrededor de dicho elemento (eje de rotación). Si se fija una cuerda en un punto de la barra y se tira de ella hacia la derecha, se producirá un par que hará que la barra rote en sentido horario. Si, por el contrario, se tira hacia la izquierda, se producirá un par en sentido anti horario. El ángulo entre la cuerda y la barra determinará la relación entre la fuerza con la que se tira de la cuerda y el par producido. Cuanto menor sea ese ángulo, menor será el par para una fuerza dada. Cuando el ángulo sea nulo, es decir que la barra y la cuerda se encuentran alineadas, no se estará aplicando ningún par con la mano. Por lo tanto, el par aplicado, es función tanto de la posición de la mano como de la fuerza ejercida sobre la cuerda.

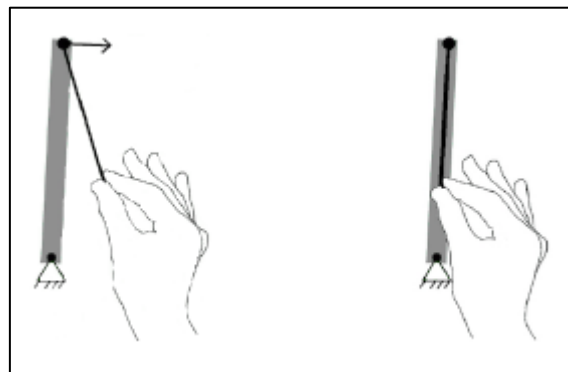


Figura 2-5: Concepto de funcionamiento del actuador MACCEPA[8].

Si se invierte el razonamiento anterior y en lugar de mover la mano, esta queda en una determinada posición y se procede a rotar la barra (asumiendo que la dirección de la cuerda no cambia y que la fuerza aplicada se mantiene constante) aparecerá un par en el sentido contrario. En este caso, la posición de la mano define la posición de equilibrio para el cuál no se genera ningún par.

Como ilustra la Figura 2-6, el MACCEPA está formado por cuatro cuerpos: Link 1 (eslabón 1), Link 2 (eslabón 2), lever arm (brazo de palanca) y Fixed link cuya rotación es igual a la del Link2. El Fixed link se encuentra rígidamente conectado con el eje de rotación de la articulación y actúa directamente sobre el eslabón de salida (Link 2). Una correa de kevlar se

conecta con el Fixed link y es guiada mediante poleas a través del lever arm hasta un resorte que se encuentra alojado en el Link 1. Este resorte puede ser pre-tensionado/pre-comprimido, para modificar las características de par y rigidez producidas por el actuador.

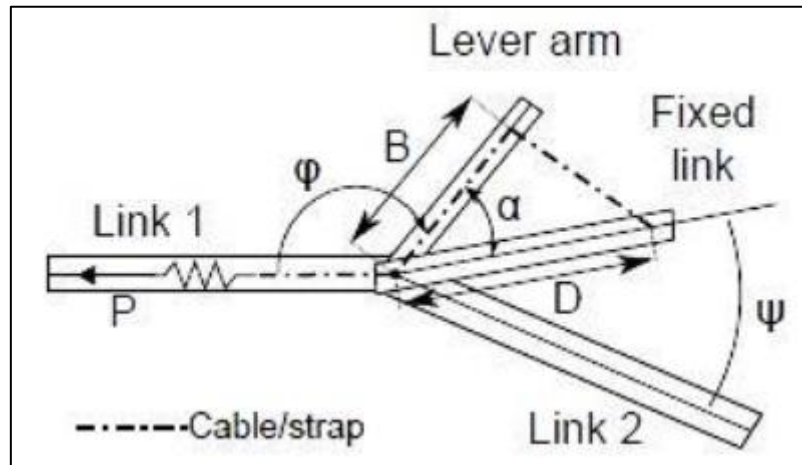


Figura 2-6: Diagrama esquemático del actuador MACCEPPA[9]

Si se considera el Link 1 como fijo y el lever arm se encuentra alineado con el Fixed Link (que está unido al Link 2), la fuerza ejercida por el resorte por medio de la correa, está alineada el Fixed Link, por tanto, no se ejerce ningún par en el Link 2. Si el lever arm gira, de forma que aparezca una desviación angular ( $\alpha$ ) entre éste y el Fixed Link la fuerza que el resorte ejerce sobre Fixed Link dejará de estar alineada con este. Ello produce un par de rotación tanto en el Fixed link como en el Link 2 (el ángulo  $\psi$  es constante), determinado por la componente perpendicular de la fuerza producida en el muelle, que tenderá a alinear el Fixed Link con el lever arm. La posición del lever arm,  $\varphi$ , es la posición de equilibrio del actuador MACCEPA, definida como la posición en la que el actuador no produce ningún par de rotación en la articulación.

El actuador necesita dos motores: uno para desplazar el brazo de palanca y por tanto de definir la posición de equilibrio y otra para alterar la pre-tensión/pre-compresión y así cambiar la rigidez del actuador. (aunque es posible prescindir de este si no se requiera de rigidez variable). Ambos motores se encuentren alojados en el mismo eslabón de la articulación (Link 1), de esta forma no se requieren de componentes que atraviesen o se extiendan a lo largo de la articulación, y por tanto se incrementa su posibilidad de implementación en el diseño de un actuador de, al menos, 2 grados de libertad.

En la Figura 2-7 se muestra la secuencia de funcionamiento del actuador. El ángulo  $\varphi$  corresponde a la desviación angular del Lever Arm (Brazo de palanca) respecto a la posición de equilibrio, mientras que  $\Theta$  indica la desviación entre esta posición y el Fixed Link (Brazo Fijo). Finalmente,  $\alpha$  es la variación angular entre el Fixed Link y el Lever Arm. En un inicio, se muestra el actuador en su posición de equilibrio, sin ángulo de desviación  $\alpha$  y con el resorte sin comprimir. Luego de modificar la posición de equilibrio, se muestra el mecanismo en una posición desviada. Todo esto hace que la correa se tense, haciendo que la plataforma de compresión se desplace comprimiendo el resorte contra el carro del husillo. La fuerza producida por la correa en el Fixed Link ya no está alineada con el Lever Arm y como consecuencia se produce un par en el brazo de salida. El par intentará alinear el Lever Arm con el Fixed Link. Ello produce la rotación de la articulación y que el resorte vuelva de nuevo a su posición de reposo.

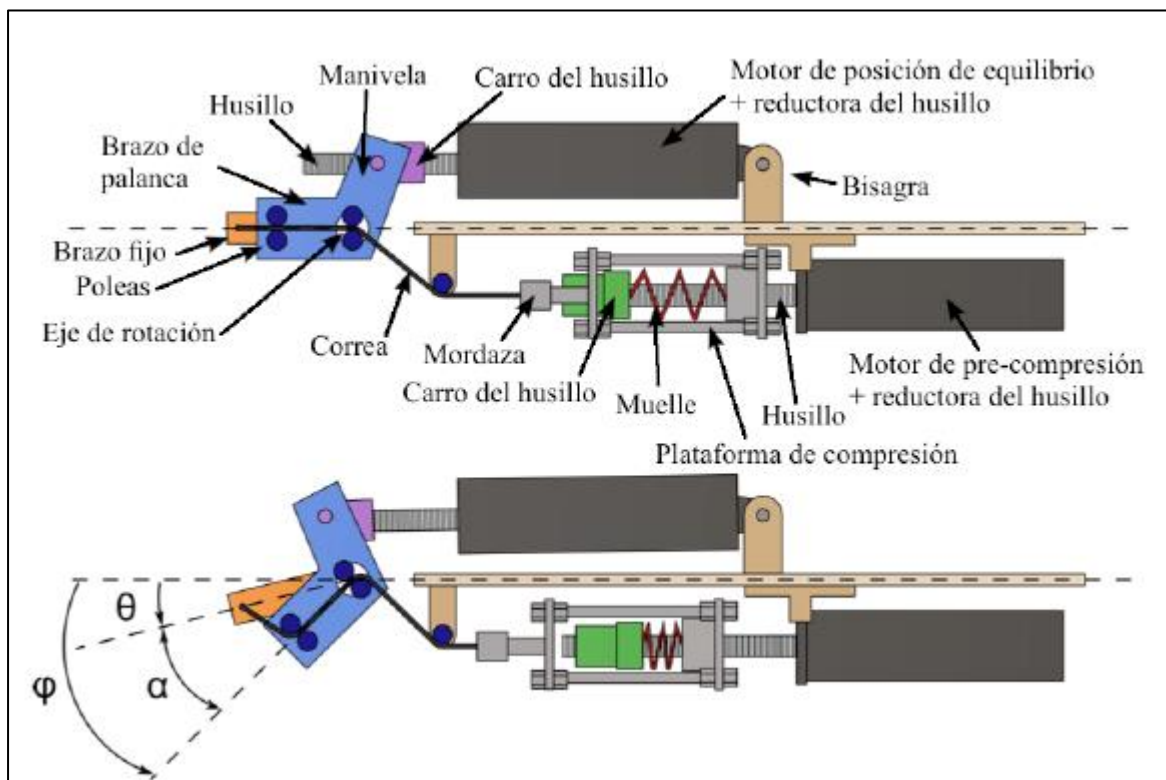


Figura 2-7: Funcionamiento del actuador MACCEPA[8].

### 2.3. Redes neuronales (Perceptrón Multicapa)

Las redes neuronales son modelos matemáticos que reproducen el comportamiento del cerebro humano, donde el principal objetivo es la obtención de un cierto comportamiento inteligente[10]. Esto implica la capacidad de aprender a realizar una determinada tarea. La neurona artificial constituye el elemento principal de una determinada red neuronal y estas se



combinan en estructuras denominadas capas. De esta forma son capaces de adaptar su funcionamiento a distintos entornos modificando sus conexiones con otras neuronas (sinapsis). En la Figura 2-8 se aprecia los elementos que componen una neurona artificial.

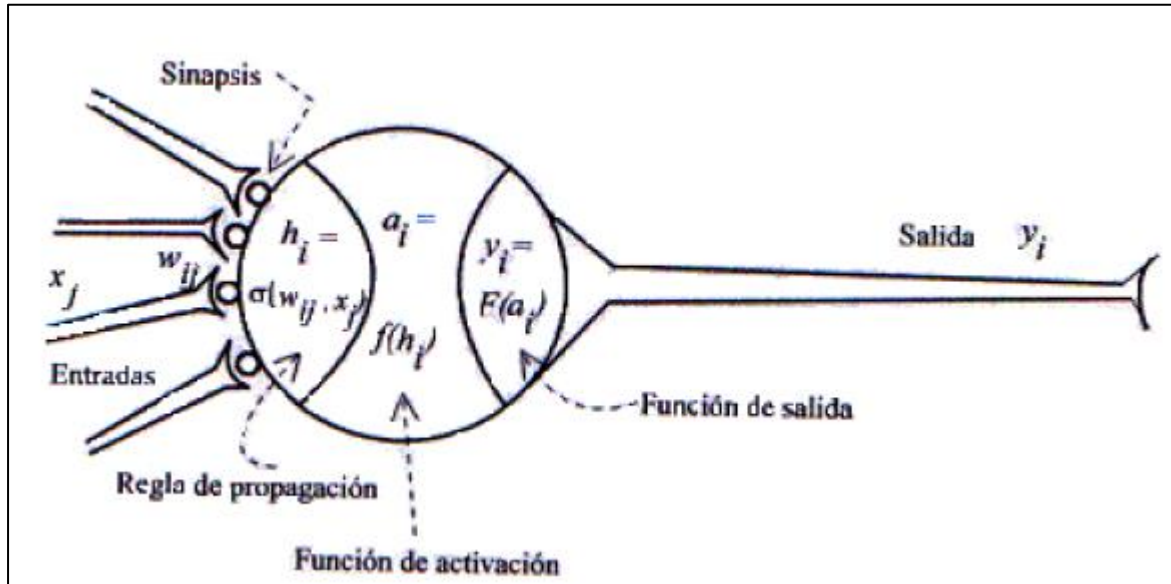


Figura 2-8: Neurona artificial [10].

**Entradas:** corresponde a la información de entrada que se le presenta al algoritmo. Se designan con la letra  $x$

**Pesos:** corresponden a las sinapsis o conexiones entre neuronas. Por medio del ajuste de estos pesos es que la red neuronal es capaz de aprender. Se designan con la letra  $w$ .

**Regla de propagación:** La regla de propagación determina el potencial resultante de la interacción de la neurona  $i$  con las  $N$  neuronas vecinas. La regla de propagación más simple y utilizada consiste en realizar una suma de las entradas ponderadas con sus pesos correspondientes.

$$h_i(t) = \sum_j w_{ij} * x_i \tag{2-1}$$

**Función de activación:** determina el estado de activación actual de la neurona en base al potencial resultante  $h_i(t)$ . En la Figura 2-9 se resumen algunas de las funciones de activación más comunes, así como sus características.

$$a_i(t) = f(h_i(t)) \tag{2-2}$$



Función	Formula	Rango
Identidad	$y = x$	$[-\infty, \infty]$
Escalón	$y = \begin{cases} +1 & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases}$	$[0, 1]$
	$y = \begin{cases} +1 & \text{si } x \geq 0 \\ -1 & \text{si } x < 0 \end{cases}$	$[-1, 1]$
Lineal a tramos	$y = \begin{cases} x & \text{si } -1 \leq x \leq 1 \\ +1 & \text{si } x > 1 \\ -1 & \text{si } x < -1 \end{cases}$	$[-1, 1]$
Sigmoidea	$y = \frac{1}{1 + e^{-x}}$	$[0, 1]$
	$y = \tanh(x)$	$[-1, 1]$
Sinusoidal	$y = \text{Sen}(\omega \cdot x + \varphi)$	$[-1, 1]$

Figura 2-9: Funciones de activación [10]

**Salidas:** La función de salida proporciona el valor de salida de la neurona, en base al estado de activación de la neurona. En general se utiliza la función identidad, es decir:

$$y_i(t) = F(a_i(t)) = a_i(t) \quad (2-3)$$

En general podemos encontrar tres tipos de neuronas artificiales:

- Neuronas de entrada: reciben información directamente desde el exterior.
- Neuronas ocultas: reciben información desde otras neuronas artificiales (de entrada, o de otras neuronas ocultas).
- Neuronas de salida: reciben la información procesada y las devuelven al exterior.

Las entradas y salidas de una neurona pueden ser clasificadas en, binarias o continuas. Las neuronas binarias sólo admiten dos valores posibles ( $\{0,1\}$  o  $\{-1,1\}$ ), mientras que las neuronas continuas admiten valores dentro de un determinado rango (en general se tiene  $[-1, 1]$ ).

Las neuronas suelen agruparse en unidades funcionales denominadas capas. Una red neuronal artificial está compuesta por una o más capas, las cuales se encuentran interconectadas entre sí. Cuando la red está compuesta por dos o más capas hablamos de redes multicapa. La topología de un perceptron multicapa (MLP) es un tipo de red neuronal definida por una o más capas ocultas, una capa de entrada y una de salida. En la Figura 2-10 se puede ver un ejemplo de un MLP.

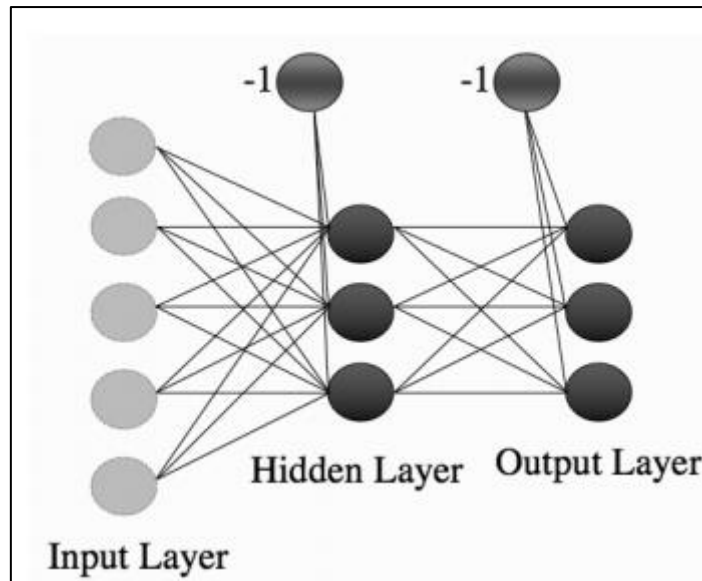


Figura 2-10: Perceptrón Multicapa [11].

No existen restricciones sobre la función de activación, aunque en general se suelen utilizar funciones sigmoideas[10]. Del mismo modo cabe destacar la presencia de una neurona adicional en las capas ocultas y, de entrada. Dicha neurona cuenta con una salida fijada en -1 y se conoce como neurona sesgada y nos evita la necesidad de modificar la función de activación de una determinada neurona [11].

Existen varios métodos de entrenamiento, el más utilizado corresponde al denominado aprendizaje supervisado en el cual se le presenta al algoritmo una serie de entradas para las cuales se sabe cuáles son las salidas correctas. Los pesos se van modificando de manera proporcional al error que se produce entre la salida real de la red y la salida. De esta forma la red aprende la relación existente entre los datos, adquiriendo la capacidad de generalizar, por lo que puede tratar más adelante con información que no le fue presentada durante de la fase de entrenamiento.

De esta forma se distinguen claramente dos fases: primero se aplica un patrón de entrada, el cual se propaga por las distintas capas que componen la red hasta producir la salida de la misma. Esta salida se compara con la salida deseada y se calcula el error cometido por cada neurona de salida. Por medio del método del gradiente descendente, se define una función  $E(W)$  que proporciona el error que comete la red en función del conjunto de pesos sinápticos  $W$ . El objetivo del aprendizaje será encontrar la configuración de pesos (mediante un proceso iterativo) que permita encontrar el mínimo global de dicha función de error.

Partiendo de un conjunto de pesos  $W(0)$  con valores aleatorios, para el instante inicial ( $t=0$ ), se calcula la dirección de máxima variación del error, dada por el gradiente  $\nabla E(W)$ . Estos errores se transmiten hacia atrás (backpropagation), partiendo de la capa de salida, hacia todas las neuronas de las capas intermedias. Luego, se actualizan los pesos siguiendo el sentido contrario al indicado por dicho gradiente. De este modo se va produciendo un descenso por la superficie de error hasta alcanzar un mínimo (Figura 2-11).

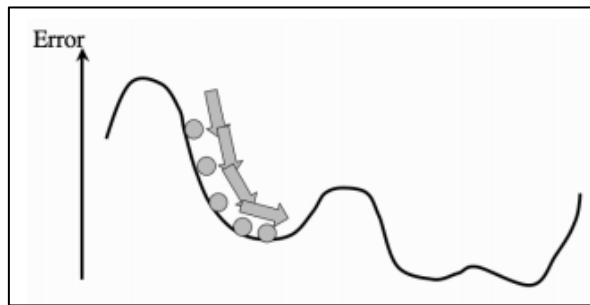


Figura 2-11: Principio del algoritmo de gradiente descendente[11]

A continuación, se resume los pasos a seguir para la implementación de un MLP para un desarrollo más detallado ver anexo A5:

1. Inicializar los pesos con valores aleatorizados. Se recomienda que dichos valores estén en un rango de manera que se cumpla la siguiente condición[11]:

$$\frac{-1}{\sqrt{n}} < w < \frac{1}{\sqrt{n}} \quad (2-4)$$

Donde  $n$  corresponde a la cantidad de nodos de entrada.

2. Para cada vector de entrada calcular el potencial correspondiente a cada neurona utilizando la suma ponderada de las entradas descrita en la ecuación 2-1.
3. Evaluar dicho potencial en la función de activación de la neurona correspondiente.
4. Repetir los pasos 2 y 3 hasta llegar a la capa de salida donde la función de activación puede ser por lo general la función identidad.
5. Calcular el error en la capa de salida (Se debe definir una función de error como, por ejemplo: error cuadrático medio).
6. Propagar error hacia atrás a través de cada una de las capas ocultas (depende de la función de activación utilizada en cada capa y esta debe ser derivable).
7. Actualizar el valor de los pesos según la siguiente fórmula:

$$w^t = w^{t-1} + \mu * \delta * a_i + \alpha * \Delta w^{t-1} \quad (2-5)$$

Donde  $w$  corresponde al valor del peso,  $\mu$  corresponde a la razón de aprendizaje (que tanto debe cambiar el valor del peso con cada iteración),  $a_i$  corresponde a la función de activación utilizada,  $\delta$  corresponde al error presente en la capa,  $\alpha$  corresponde al término de momento (evita que el algoritmo quede atrapado en un mínimo local) y  $\Delta w$  corresponde al gradiente de error.

8. Repetir hasta que la función de error sea prácticamente igual a cero.

Finalmente se debe considerar que MLP fue diseñado como un algoritmo tipo “batch” o “offline”. Este es considerado el enfoque de aprendizaje clásico en el que se utiliza un set de entrenamiento (compuesto por múltiples entradas). El cambio en los pesos se realiza una vez procesados todas las entradas. Esto implica que la red neuronal es entrenada antes de ser utilizada. Este método realiza un mejor estimado del gradiente del error por lo que es capaz de converger hacia los valores finales de los pesos más rápidamente.

Por otro lado, se tiene el diseño secuencial o “online” donde los pesos son cambiados cada vez que se procesa un ejemplo. Esta opción resulta atractiva en aplicaciones donde se tenga un set de entrenamiento sumamente grande (inclusive redundante) [11]. Por otro lado, este método es más robusto ya que errores, omisiones y redundancias pueden ser corregidos o simplemente ignorados. Este método es especialmente apropiado para aprender funciones variantes en el tiempo de manera que puede adaptarse constantemente ante condiciones cambiantes.

#### **2.4. Feedback Error Learning**

Desde el punto de vista biológico (Figura 2-12) el Feedback Error Learning (FEL) fue originalmente concebido como un modelo del cerebelo mediante el cual el sistema nervioso central adquiere un modelo inverso capaz controlar una determinada acción. Para que el cuerpo humano pueda realizar movimientos rápidos y coordinados, este no puede depender únicamente de un control por lazo de retroalimentación, ya que los lazos de retroalimentación biológicos son lentos y tienen ganancias pequeñas[12]. A nivel subcortical y cortical, el cerebelo juega un papel importante en el control del sistema motor. Una característica importante del sistema nervioso central (SNC) es su notable capacidad para adaptarse a los cambios, tanto en el medio ambiente y como en el interior del cuerpo. Esta capacidad indica que el SNC aprende y mantiene modelos internos de la cinemática y dinámica del medio ambiente y el cuerpo. Ante este panorama, una posibilidad es que el cerebelo contenga dichos

modelos internos de nuestro aparato locomotor [13][14]. Esta hipótesis supone que luego de una etapa de aprendizaje el SNC obtiene un modelo inverso del objeto controlado y mediante este se obtiene una señal de control. De esta forma pueden aprender y predecir las consecuencias de una determinada acción o trayectoria y así contrarrestar retrasos de la señal de retroalimentación por medio de acciones de compensación (activas o pasivas).

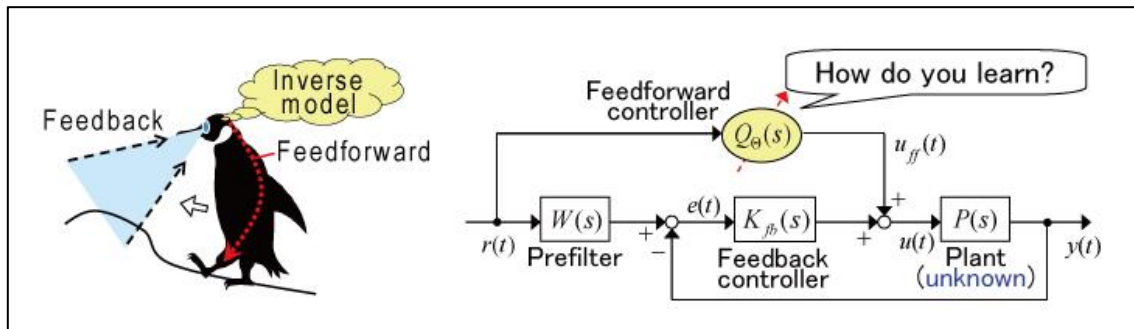


Figura 2-12: Concepto biológico sobre el que se basa el FEL.

Desde el punto de vista de control el FEL es un método de control adaptativo mediante el uso de un aprendizaje supervisado desarrollado para aprender el modelo inverso de una determinada planta. Se descubrió que era posible utilizar una señal de control por retroalimentación para aprender la dinámica inversa de una planta, del mismo modo se ha demostrado que dicho método es capaz de proveer un control estable tanto durante como después del aprendizaje[7]. Como se puede apreciar en la Figura 2-13 el Feedback Error Learning (FEL) es un esquema de control que integra control por retroalimentación (Feedback Control) con control por pre-alimentación (Feedforward Control). De esta forma el control está compuesto por dos subcontroladores, uno adaptativo que utiliza la señal de retroalimentación como parámetro de aprendizaje y un controlador clásico por retroalimentación.

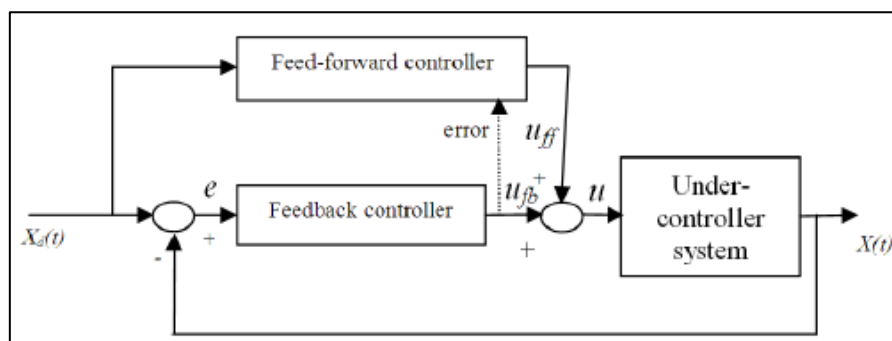


Figura 2-13: Esquema de control FEL.[15]

El FEL debe incorporar entonces un controlador por retroalimentación (por ejemplo, un controlador PID convencional), una planta controlada y su modelo inverso. Un modelo

inverso predice la señal de control necesaria para poder llevar el estado actual de la planta al siguiente [16]. Una red neuronal (por ejemplo, un MLP) se utiliza como modelo inverso y se entrena por medio de la señal de control proporcionada por el control PID (usada como señal de error). La señal de control total ( $U$ ) se obtiene al sumar las contribuciones de dicho controlador ( $U_{fb}$ ) y de la red neuronal ( $U_{ff}$ ).

La salida del control por retroalimentación es en esencia una medida del error existente entre la dinámica de la planta y la dinámica inversa calculada por la red neuronal. Esto debido a que si la señal de retroalimentación es nula implica que el error también lo es, de esta forma el sistema hace uso del error existente entre la salida deseada y la actual para guiar el entrenamiento[14]. Una vez finalizado el aprendizaje, si el modelo inverso es preciso la señal  $U_{ff}$  será capaz por si sola de controlar la planta, de manera que  $U_{fb} \approx 0$  [16]–[18]. Durante las etapas iniciales de entrenamiento la señal  $U_{ff}$  será pequeña en comparación con  $U_{fb}$ , sin embargo, a medida que los pesos de la red se van actualizando y convergiendo hacia su valor final, la señal  $U_{ff}$  será cada vez más dominante.

Los métodos convencionales de diseño de sistemas de control implican el desarrollo de un modelo matemático que describa la dinámica de la planta que se desea controlar y la aplicación de técnicas analíticas de manera a derivar un paradigma de control. Usualmente dicho modelo matemático consiste en una serie de ecuaciones lineales y/o no lineales en las que se han hecho alguna aproximación o simplificación. Estas técnicas convencionales resultan insuficientes cuando el modelo presenta parámetros con cierta incertidumbre o complejidad, adema de que el control por retroalimentación por lo general supone un compromiso entre la simplicidad de modelo y la exactitud a la hora de describir el comportamiento del sistema a controlar[19].

De esta forma la principal ventaja de utilizar dicha técnica radica en que la red neuronal es capaz de aprender la dinámica inversa sin ningún conocimiento a priori de los parámetros ni ecuaciones que describan la dinámica de la planta a controlar. Sin embargo, su implementación supone a su vez toda una serie de consideraciones tales como escoger las entradas adecuadas, estrategias de aprendizaje, arquitectura de la red y el valor de las ganancias en el lazo de retroalimentación para asegurar la estabilidad del FEL. [20]

## 2.5. Protocolo de comunicación CAN (Control Area Network)

El protocolo CAN fue especialmente diseñado para la conexión de sensores, actuadores y ECU's (Engine Control Unit) de un vehículo al reemplazar complejos cableados por un sencillo par trenzado. Sin embargo, CAN se ha convertido con el paso de los años en una opción atractiva para sistemas de control embebidos debido a su bajo coste, robustez, capacidad de detección y reparación de errores e inmunidad a ambientes ruidosos. En la Figura 2-14 se presenta un bus de comunicación CAN. Notar la presencia de resistencias terminales en ambos extremos del bus para evitar que la señal se refleje (se recomiendan resistencias de  $120\ \Omega$  que igualen la resistencia del cable [21]).

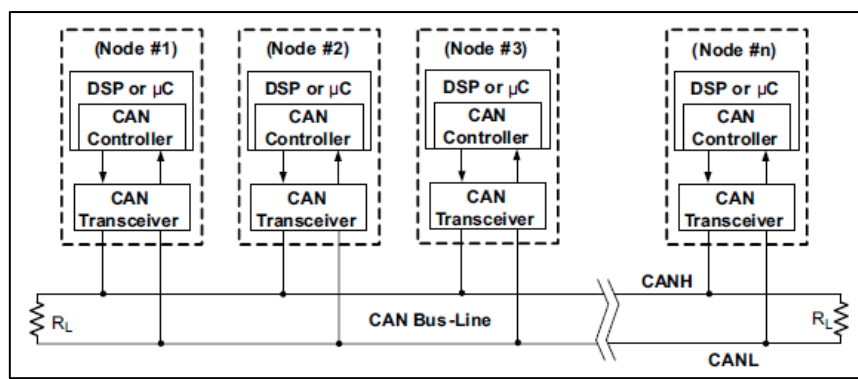


Figura 2-14: Bus CAN[21]

CAN es una red uno a uno, es decir que no existe un dispositivo maestro que controle el acceso de lectura y escritura. Esto implica que todos los nodos tienen el mismo privilegio en el bus (acceso multi-maestro). De esta forma cada nodo en el bus debe esperar por un periodo de inactividad antes de intentar enviar un mensaje. Cuando la red está ocupada (otro nodo se encuentra transmitiendo), la transmisión del paquete que contiene al mensaje debe retrasarse hasta que la transmisión finalice. Existe la posibilidad de que dos o más nodos intenten enviar sus respectivos paquetes al mismo tiempo lo que provoca una colisión de datos.

Dicha situación, como se aprecia en a la Figura 2-15 se resuelve por un proceso de arbitración (comparando los identificadores de los paquetes involucrados), el paquete con el identificador de mayor prioridad gana el derecho de utilizar la red, mientras que los demás nodos deben detener la transmisión de sus respectivos paquetes e intentar reenviarlos cuando el bus esté libre nuevamente.

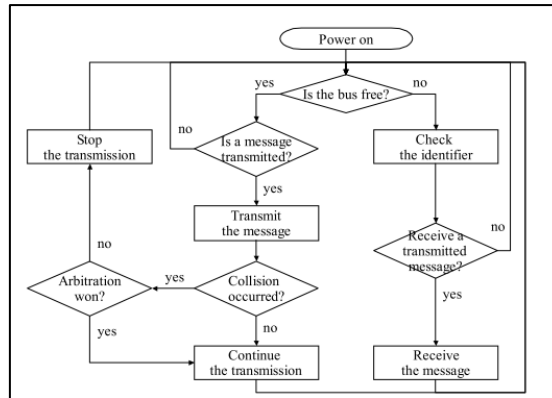


Figura 2-15: Diagrama de flujo del protocolo CAN.[22]

El identificador o ID puede tener dos tipos de formato el standard de 11 bits (+ 1 bit) y el extendido de 29 bits (+ 3 bits). Ambos tipos de formatos pueden ser transmitidos en el mismo bus por cualquier nodo, y el proceso de arbitración tiene los medios para procesar diferentes tipos de identificador al mismo tiempo. En la Figura 2-16 se aprecia la composición de un paquete de datos CAN donde el tamaño de cada parte se expresa en bits. Cada paquete empieza con un bit dominante que interrumpe el estado de inactividad del bus. Seguidamente se tiene el identificador que define tanto la prioridad como el tipo de mensaje. Los demás espacios corresponden a:

- Control: determina la longitud del mensaje.
- Data: contiene el mensaje a transmitir hasta un máximo de 8 bytes.
- Checksum: permite determinar si el mensaje se transmitió correctamente.
- Acknowledge: confirma a recepción del mensaje.
- End of frame: para indicar el fin del paquete.

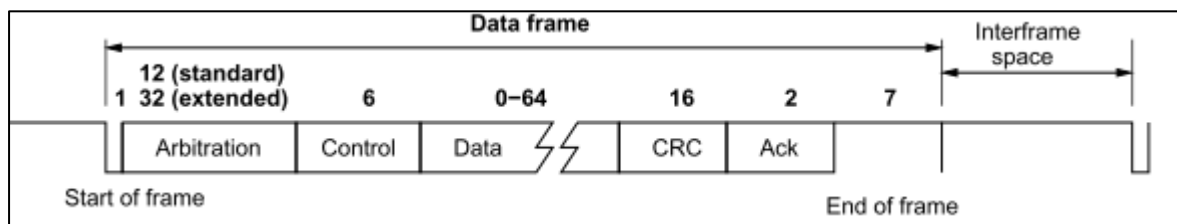


Figura 2-16: Composición de un paquete de datos CAN [23].

Como se puede apreciar el identificador se encuentra al inicio del paquete por lo que en caso de perder el proceso de arbitración la información contenida en el espacio de Data no se ve comprometida en ningún sentido. En la Figura 2-17 se muestra como se da el proceso de arbitración, de esta forma cuando el bit recesivo enviado por el nodo B es sobrescrito por el bit dominante del nodo C, B detecta que el estado del bus no corresponde al bit que transmitió. Por lo tanto, el nodo B detiene la transmisión mientras que el nodo C continua con



la suya. Finalmente, el nodo B reintentará la transmisión una vez que el nodo C libere el bus [21].

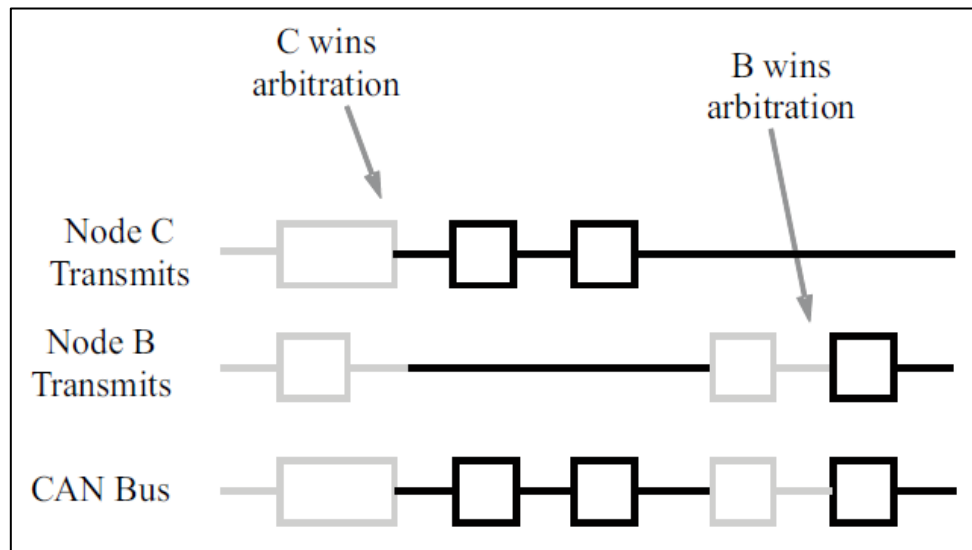


Figura 2-17: Arbitración en un bus CAN [21]

Cabe mencionar que CAN utiliza una lógica inversa en cuanto a lo que se considera un nivel lógico alto y un nivel lógico bajo. Normalmente un nivel alto se asocia con un 1 y un nivel lógico bajo con un 0. No obstante, en CAN este no es el caso, como se aprecia en la Figura 2-18 un 1 se asocia con un nivel recesivo y un 0 con uno dominante. De esta forma entre menor sea el número del identificador mayor prioridad tendrá el paquete correspondiente, por lo que el ID = 0 tiene la mayor prioridad al dominar el bus por más tiempo (un bit dominante siempre sobrescribe a uno recesivo). También recordar que un nodo al transmitir monitorea su propia transmisión, por lo que la salida CANH y CANL del transmisor están internamente conectadas a las entradas del receptor.

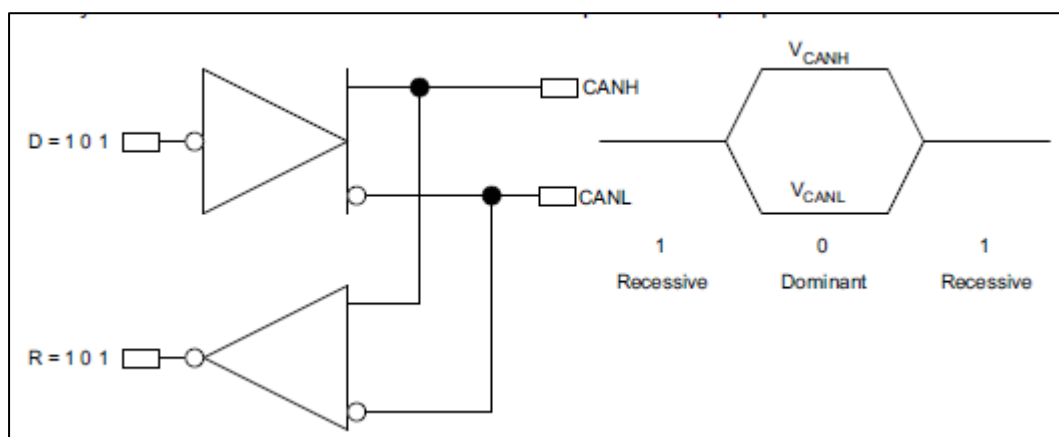


Figura 2-18: Lógica inversa del bus CAN [21]

También destacar que el hecho de que la señal en el bus sea diferencial es precisamente donde reside su resistencia al ruido y robustez. Al ser una señal balanceada (la tensión en cada línea

es igual, pero en sentido contrario) resulta en la cancelación de cualquier campo eléctrico o ruido que pueda afectar al bus. El estándar ISO 11898 especifica una velocidad de transmisión de hasta 1Mbps para un bus de 40 m. En la Figura 2-19 se especifican las velocidades de transmisión con respecto a la longitud del bus.

Bit rate	bit time	bus length
1 Mb/s	1 $\mu$ s	25m
800 kb/s	1.25 $\mu$ s	50m
500 kb/s	2 $\mu$ s	100m
250 kb/s	4 $\mu$ s	250m
125 kb/s	8 $\mu$ s	500m
62.5 kb/s	16 $\mu$ s	1000m
20 kb/s	50 $\mu$ s	2500m
10 kb/s	100 $\mu$ s	5000m

Figura 2-19: Velocidades de transmisión y longitud del bus [21]

Finalmente, para realizar la recepción de mensajes los controladores CAN poseen por lo general 1 o 2 registros de recepción (RX buffers). En cada nodo se puede definir uno o más filtros y máscaras que permiten definir los mensajes que se desean recibir en dicho nodo. Una máscara define en cuales bits del identificador se debe aplicar el filtro. De esta forma un 1 en la máscara indica que se deben comparar los bits del ID y del filtro, por otro lado, un bit en 0 en la máscara indica que no se debe aplicar el filtro para dicho bit del ID. En la Figura 2-20 se muestra un ejemplo donde el ID corresponde a 011101010 (0x3AA). Dada la configuración de la máscara, solo el primer, tercer, sexto, séptimo y octavo bit del ID deben ser comparados con los filtros de recepción. Luego de realizada la comparación, los bits del filtro RxObject1 coinciden con los del ID por lo que el mensaje es aceptado y guardado en el buffer correspondiente.

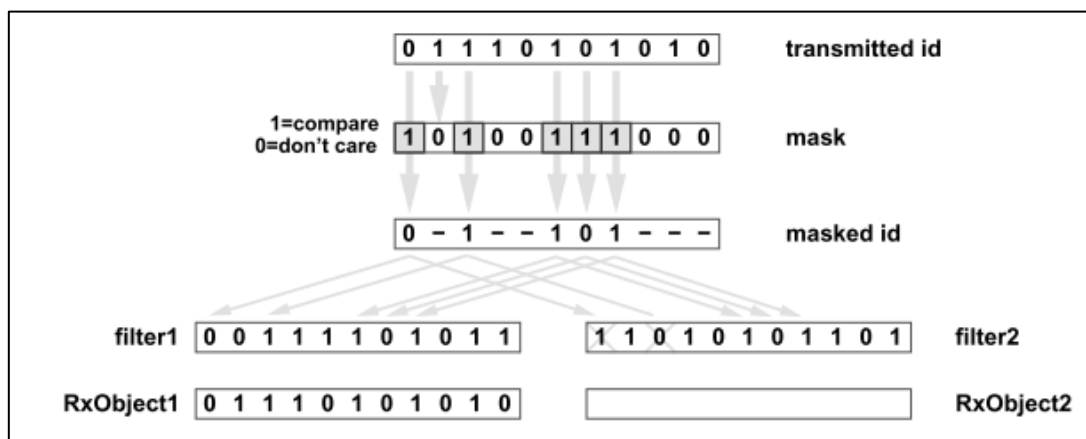


Figura 2-20: Máscaras y filtros para recepción de mensajes[23].

### 3. CAPITULO 3: Estrategia de Comunicación.

Este capítulo expone como estaba configurado el protocolo CAN en los dispositivos de control del robot bípedo Binocchio, del mismo modo se presentan tanto los problemas de comunicación existentes, así como las pruebas realizadas para determinar la causa de estos. Finalmente se expondrá la solución planteada y se procederá a determinar cualitativamente la mejoría en el rendimiento.

#### 3.1 Configuración actual

El modulo CAN del microcontrolador DSPIC30F4011 posee 3 buffers de transmisión con capacidad para 8 bytes de información cada uno[24]. El microcontrolador está configurado actualmente para la trasmisión de 6 bytes únicamente, por lo que los paquetes de información a enviar constan de 48 bits divididos en sub paquetes según convenga como se puede ver en la Figura 3-1

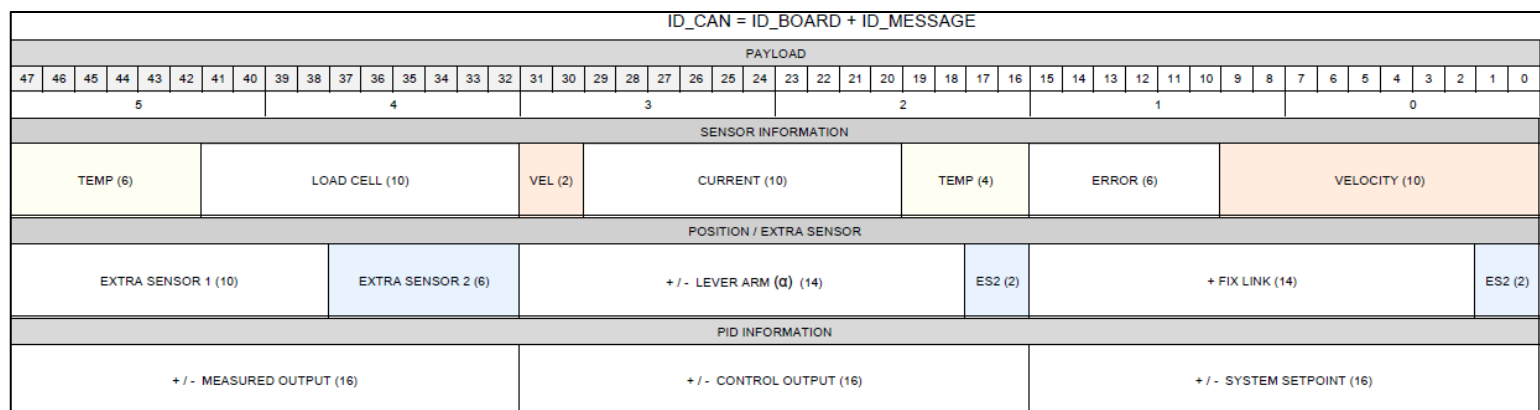


Figura 3-1: División de paquetes enviados[5]

Los paquetes de datos tienen identificadores o ID's de 11 bits (identificador standard). Los ID's están dispuestos de tal manera que se pueda identificar tanto la tarjeta a la que se le envía el mensaje, como el tipo de mensaje. En la Figura 3-2 se aprecia la conformación del identificador.

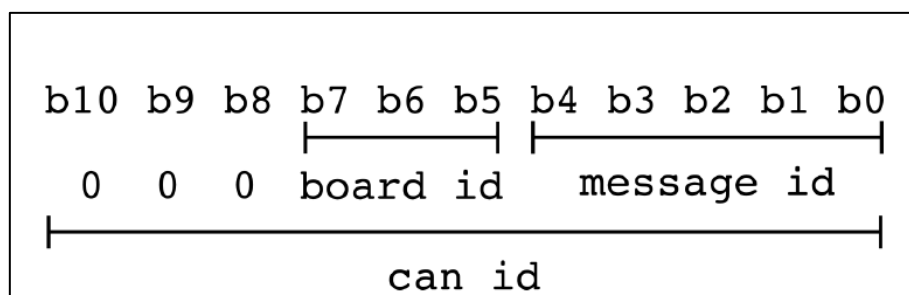


Figura 3-2: Conformación de los identificadores[5].

Por otro lado, para la recepción de datos el microcontrolador cuenta con dos buffers de recepción, se configuró el modulo CAN de manera que la recepción de datos utilice el buffer 0. Si dicho buffer está lleno y otro mensaje es aceptado se guardará en el buffer 1 siempre y cuando esté vacío, de lo contrario el mensaje es descartado. En cada SPU se utiliza la misma máscara (0XFE0) y los filtros se asocian con el “Board id” como se aprecia en la Figura 3-3:

BOARD IDENTIFIERS					
LEFT HIP	(LH)	=>	000 001 00000	=>	BOARD ID 32 (0x020)
LEFT KNEE	(LK)	=>	000 010 00000	=>	BOARD ID 64 (0x040)
LEFT ANKLE	(LA)	=>	000 011 00000	=>	BOARD ID 96 (0x060)
RIGHT HIP	(RH)	=>	000 100 00000	=>	BOARD ID 128 (0x080)
RIGHT KNEE	(RK)	=>	000 101 00000	=>	BOARD ID 160 (0x0A0)
RIGHT ANKLE	(RA)	=>	000 110 00000	=>	BOARD ID 192 (0x0C0)

Figura 3-3: Identificadores para cada SPU[5].

Finalmente, la velocidad de transmisión se ha configurado a 1 Mbps ya que la longitud del bus a lo largo de la pierna del robot lo permite (menor a 26 m).

### 3.2 Problemática.

Para identificar las posibles causas de los problemas de comunicación se procedió a realizar la siguiente prueba. Se utilizó únicamente una sola SPU ejecutando el firmware correspondiente al proyecto H2R para el control del robot bípedo Binocchio. Se procedió entonces a realizar únicamente la recepción y transmisión de datos por medio del bus CAN. La SPU recibiría de la PC-104 (CPU) una señal senoidal (Amplitud 50 / Frecuencia: 1 Hz) además de realizar la lectura de dos encoders magnéticos ubicados en la articulación de la rodilla del robot como se aprecia en la Figura 3-4. Hecho esto se procedió a enviar dos paquetes de datos (6 bytes) establecidos de la siguiente forma:

- Uno conteniendo la información de los encoders (2 valores de ángulo de 2 bytes cada uno) además reenviar la señal senoidal (2 bytes).
- El otro únicamente reenviaría la señal senoidal (2 bytes, el resto del paquete sería completado con 0's).

Byte	5	4	3	2	1	0
Paquete 1	Senoidal		Encoder Alpha Angle		Encoder Fixed Link Angle	
Paquete 2	0000	0000	0000	0000	Senoidal	

Tabla 3.1: Información de los paquetes enviados por la SPU [25]

Finalmente, los datos recibidos por la PC-104 junto con la señal original son enviados a una computadora por UDP para ser analizados por medio de Matlab Simulink. Notar que dicha

prueba da lugar a un bus de comunicación con 2 nodos (PC-104 y SPU) y de 3 mensajes diferentes es decir 3 ID's.

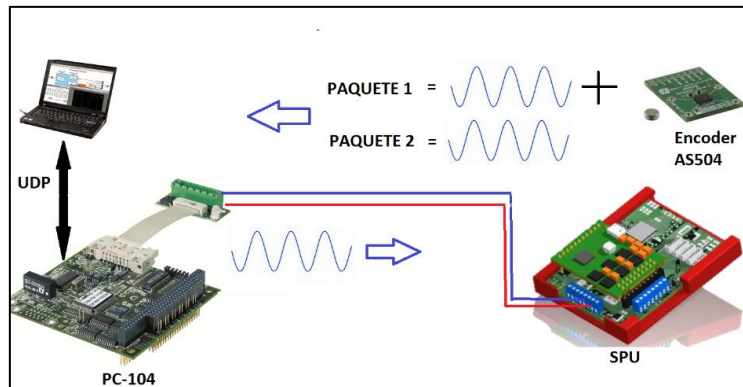


Figura 3-4: Prueba de comunicación realizada [25].

Inicialmente se creía que la comunicación presentaba problemas debido a ruido en el bus, sin embargo, esta teoría se fue descartando al analizar más detalladamente los problemas presentados. A continuación, se presentan los principales errores encontrados:

- La lectura del valor de los ángulos dada por los encoders magnéticos presentaba oscilaciones como se aprecia en la Figura 3-5. Sin embargo, las variaciones del valor del encoder son prácticamente despreciables si consideramos que el encoder tiene una precisión de 14 bits (puede tener valores de 0 a 16384 para representar de 0 a 360°)[26]. Se determinaría más tarde que las oscilaciones presentes en los encoders corresponden a pequeños desajustes entre el encoder y el imán a nivel de la articulación. Además, cabe destacar que ya se encontraba implementado un filtro digital a nivel del SPU para evitar problemas relacionados con interferencias electromagnéticas [27].

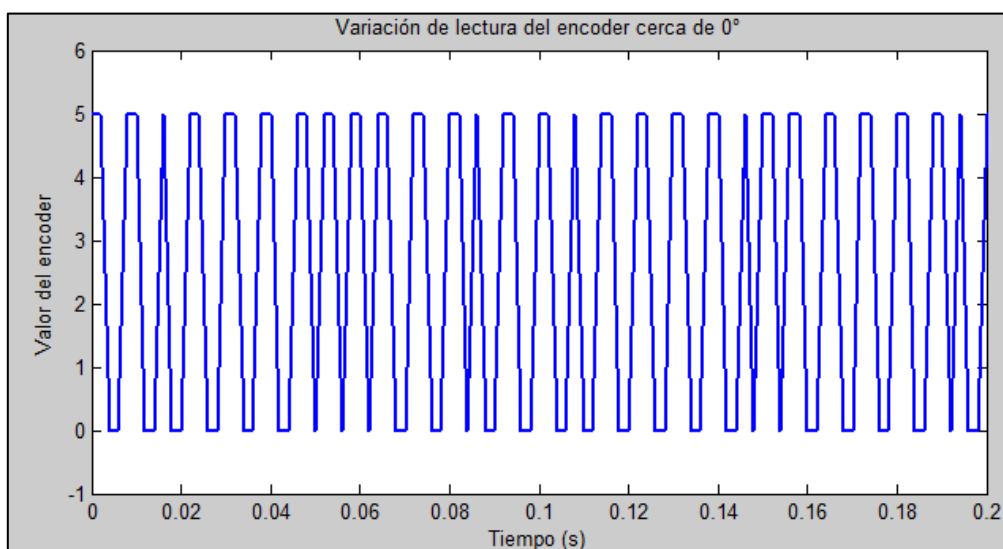


Figura 3-5: Variaciones del encoder alrededor de la posición de equilibrio (0°)[25].

- De manera aparentemente aleatoria la señal senoidal reenviada junto al valor de los ángulos comenzaba a presentar fuertes variaciones. Del mismo modo, dichas variaciones solo ocurrían en el primer paquete (ángulos + señal senoidal) y solo afectan la porción del paquete que corresponde a la señal senoidal (últimos 16 bits) como se aprecia en la Figura 3-6. Además, cabe mencionar que a la hora de realizar las pruebas de comunicación los motores no se encontraban alimentados, descartando así la principal y más obvia fuente del posible ruido que podría estar afectando al bus.

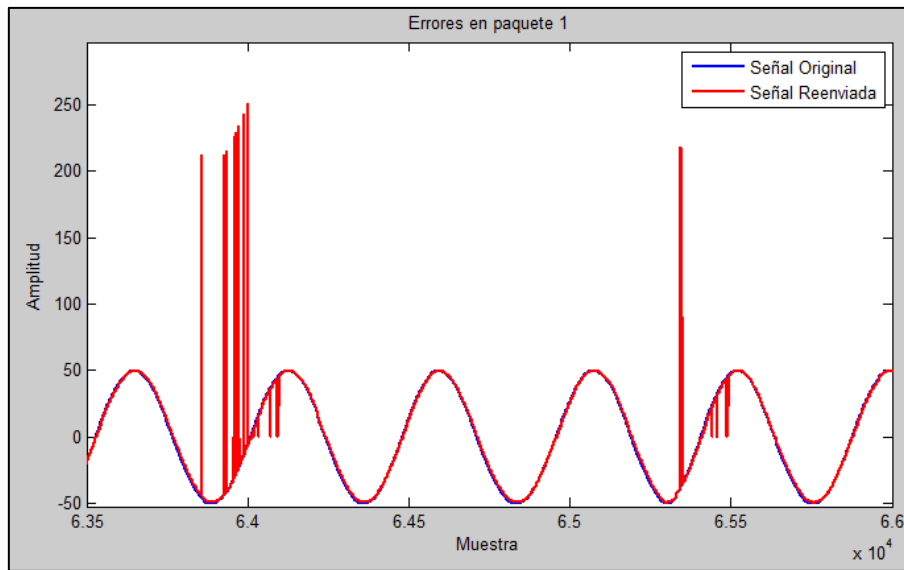


Figura 3-6: Errores en señal senoidal del paquete 1[25].

- Del mismo modo también se presentan errores como los descritos en la Figura 3-7 que como bien se evidencia comprometen completamente la señal del paquete 1.

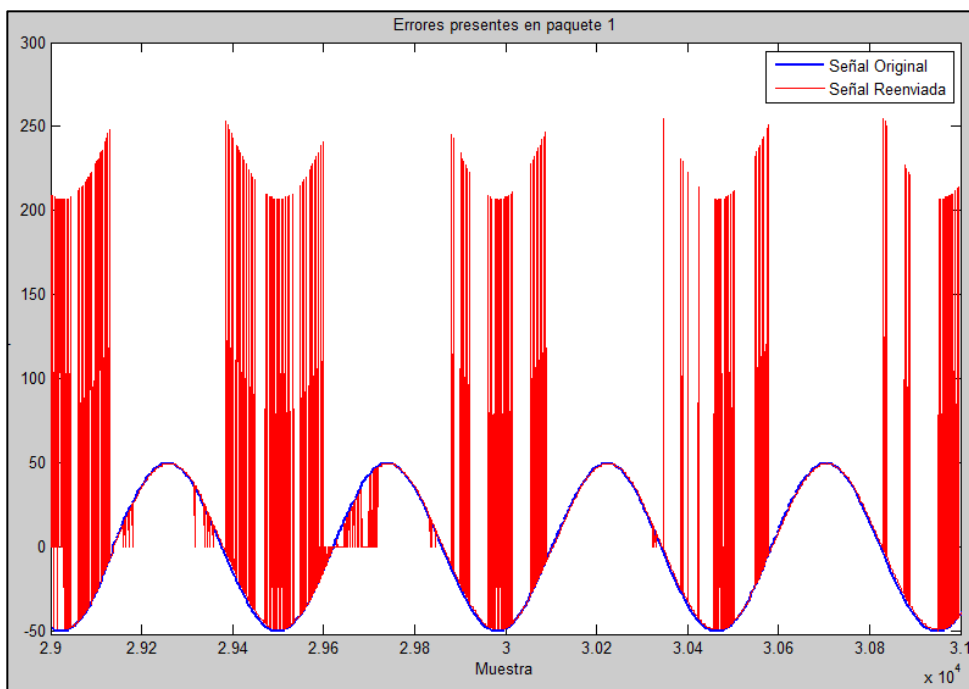


Figura 3-7: Errores en paquete 1 [25].

- Por otro lado, una inspección detallada de la señal proveniente del paquete 2 arrojó que si bien no existen variaciones como las encontradas en la señal del paquete 1 existen también problemas de comunicación que afectan la calidad de dicha señal como se aprecia en la Figura 3-8

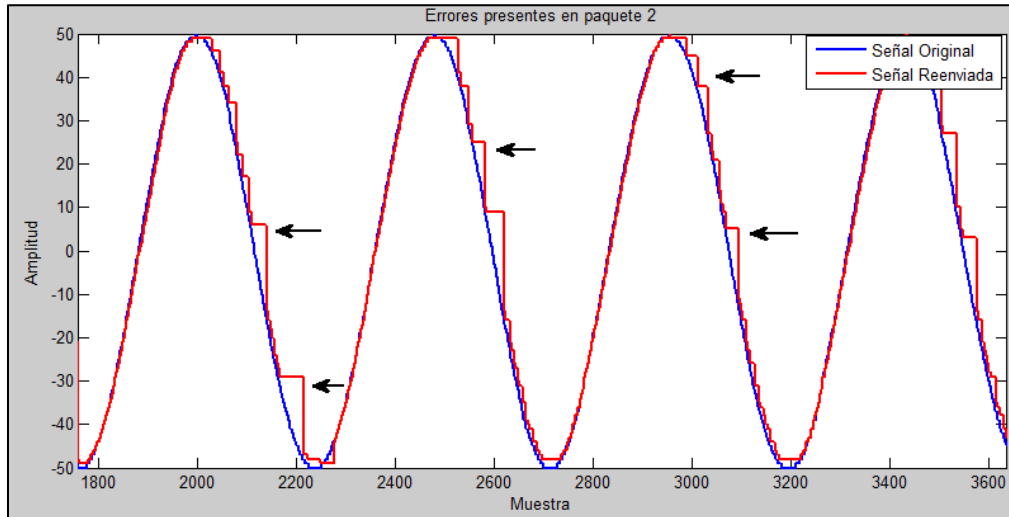


Figura 3-8: Errores presentes en señal senoidal del paquete 2[25].

- Adicionalmente también se presentan casos como el que se puede observar en la Figura 3-9 donde la señal senoidal del paquete 2 pierde por completo la forma de una senoidal. Cabe destacar que esta señal se mantiene en un rango de  $\pm 50$  igual al de la señal senoidal.

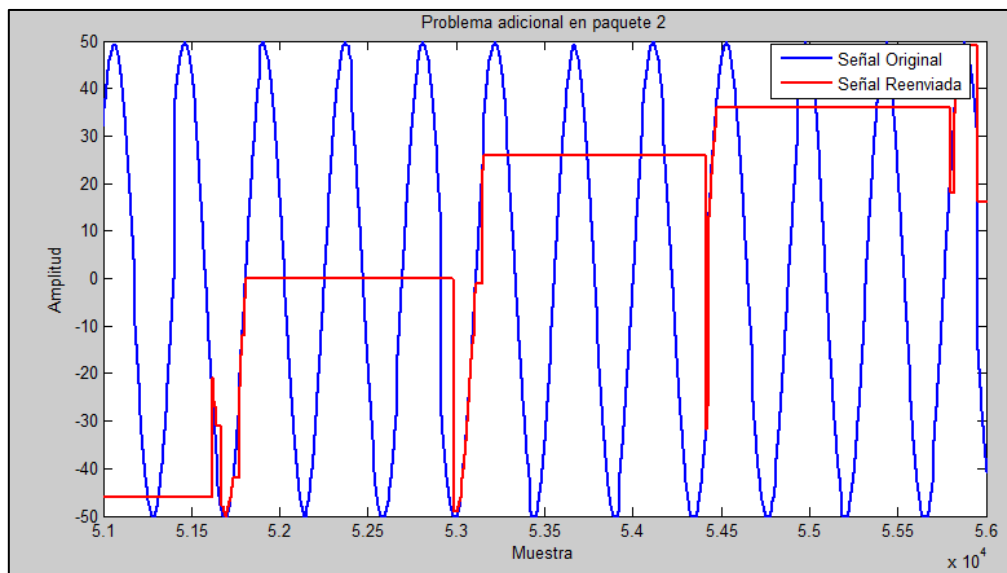


Figura 3-9: Problema adicional presente en paquete 2[25]

De este modo y considerando que los diferentes problemas difieren de un paquete a otro tanto en forma como magnitud se considera que la problemática obedece errores en la implementación del protocolo CAN más que a la presencia de ruido en el bus de comunicación.

### 3.3 Estudio del problema

Se realizaron 10 tomas de datos de 5 minutos cada una, siguiendo los lineamientos descritos en la sección anterior. Se guardaron los datos de la señal original, así como de las señales reenviadas en los dos paquetes. Posteriormente se realizó un análisis de los datos de manera a determinar el porcentaje de error en cada paquete, comparando el valor esperado de cada muestra en la señal original con el obtenido en la señal reenviada de cada paquete. El criterio para definir una muestra como errónea obedece al descrito en la Figura 3-10. De esta manera cuando analizamos una muestra de la señal enviada se debe determinar si el valor de dicha muestra está o no dentro del rango propuesto. En nuestro caso se determinó un rango equivalente al 10% de la amplitud de la señal es decir un rango de 5 (error absoluto menor a 2.5).

$$muestra(i) \begin{cases} \text{if} \left( original(i) - \frac{Rango}{2} < muestra(i) < original(i) + \frac{Rango}{2} \right) \rightarrow \text{correcta} \\ \text{else} \rightarrow \text{erronea} \end{cases}$$

Figura 3-10: Criterio de clasificación de muestras [25].

Adicionalmente se calculó el error cuadrático medio entre las señales de manera a obtener un indicador adicional que nos aporte información en cuanto a la magnitud del error o diferencia entre las señales. Dicha información se presenta en la Tabla 3.2

Medición	Paquete	Porcentaje de error %	Error cuadrático medio
1	1	4.33	242.96
	2	33.49	33.49
2	1	19.37	994.13
	2	42.51	420.73
3	1	4.30	254.17
	2	33.55	35.25
4	1	4.35	243.52
	2	34.38	36.46
5	1	24.27	1508.50
	2	52.72	737.36
6	1	23.43	1384.71
	2	38.74	727.56
7	1	3.98	183.36
	2	6.61	3.48
8	1	12.92	583.84
	2	14.59	198.30
9	1	11.08	647.06
	2	19.55	284.47
10	1	3.99	177.45
	2	6.54	3.43

Tabla 3.2: Porcentaje de error y error cuadrático medio de cada paquete[25].



Para un mejor análisis se procedió a realizar un estudio estadístico de los datos de la Tabla 3.2. Los resultados se encuentran tabulados en la Tabla 3.3. Como se puede apreciar el paquete 1 posee en promedio un 11.20 % de muestras erróneas pese a tener en promedio un error cuadrático medio de 621.97, esto nos indica que, aunque el paquete 1 tiene un menor porcentaje de muestras erróneas que el paquete 2 (con un 28.27%) cuando se presenta un error en el paquete 1, este error es de mayor magnitud que uno en el paquete 2. De este modo por un lado tenemos un paquete que falla de forma más significativa con respecto al valor deseado y otro que lo hace en menor medida, pero lo hace de manera mucho más frecuente. Por otro lado, los valores de máximo, mínimo, rango y % de variación confirman la variabilidad de los errores en ambos paquetes tanto en cantidad como en magnitud.

Indicador	Porcentaje de error		Error cuadrático medio	
	Paquete 1	Paquete 2	Paquete 1	Paquete 2
<b>Promedio</b>	11.20	28.27	621.97	248.05
<b>Desviación estándar</b>	8.41	15.66	507.76	290.35
<b>Máximo</b>	24.27	52.72	1508.5	737.36
<b>Min</b>	3.98	6.54	177.45	3.43
<b>Rango</b>	20.29	46.18	1331.05	733.93
<b>% Variación</b>	75.04	55.40	81.64	117.05

Tabla 3.3: Estadística de la prueba realizada[25].

Esto se puede apreciar visualmente en el diagrama de cajas y bigotes que se presenta en la Figura 3-11.

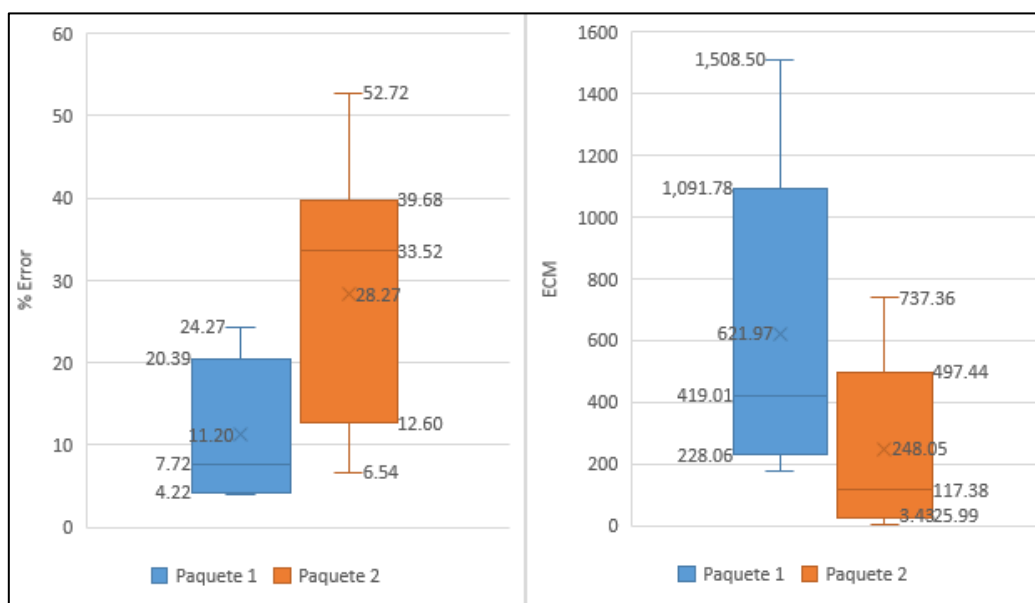


Figura 3-11: Porcentaje de error, ECM y variabilidad en paquetes 1 y 2.

### 3.4 Solución planteada.

De manera general en el dsPIC30F4011 para realizar una transmisión se debe cargar el mensaje a enviar en un buffer de transmisión, asignar una ID (identificación) en el registro correspondiente y, por último, asignar un 1 al bit TXREQ para solicitar la transmisión. En el firmware original se implementó una función denominada *can\_send ()* que toma como parámetros el mensaje a enviar y la ID. En la Figura 3-12 se puede apreciar el flujo del programa al ejecutar dicha función.

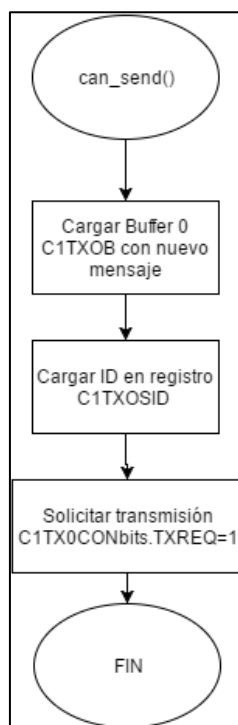


Figura 3-12: Flujo del programa al transmitir datos [25].

Entre cada llamada a la función *can\_send ()* el programa principal ejecuta un “delay” de 200  $\mu$ s, esto se debe a que al asignar un 1 al bit TXREQ no se inicia inmediatamente la transmisión del mensaje[28], ya que no solo se debe esperar a que el bus de comunicación esté disponible, en caso de que otro nodo este utilizándolo, sino que una vez disponible el bus, el mensaje a enviar debe ganar el proceso de arbitración.

Esto representa un problema ya que el tiempo entre dos llamadas a la función *can\_send ()* puede no ser lo suficientemente grande para contemplar situaciones que limiten la disponibilidad del bus como lo pueden ser:

- Uso del bus por otro nodo.
- Pérdida del proceso de arbitración.

- Errores en la transmisión del mensaje.
- Necesidad de retransmitir un mensaje.

En la Figura 3-13 se muestra una tabla de tiempos donde se puede apreciar el tiempo que le toma al SPU realizar las diferentes tareas como lectura de sensores, recepción de mensajes, envío de mensajes, etc. Se puede evidenciar lo expuesto anteriormente ya que el envío de datos puede tomar entre 200ms y 340ms, lo que significa que el tiempo necesario para enviar un mensaje por el bus de comunicación no es constante.

Procedure name	Time (us)
Fix link SPI data read	100
Fix link low pass filtering	36
Alpha SPI data read	100
Alpha low pass filtering	2.2
Load cell processing	120
CAN-BUS data receive	3.2
PID run	6
Apply control action	90
CAN-BUS send torque data	220
CAN-BUS send PID data	340
CAN-BUS send joint angles	200
CAN-BUS send other data	220

Figura 3-13: Resumen de tiempos de ejecución de tareas en el SPU[27]

De este modo si solo se utiliza un buffer de transmisión, este debe transmitir el mensaje antes de que el CPU reciba la instrucción de cargar un nuevo mensaje en el buffer. De lo contrario, el primer mensaje podría verse comprometido al ser modificado mientras ocurre la transmisión. En el programa original se utilizaron “delays” entre cada instrucción de envío de datos, de manera a garantizar que el mensaje tuviera suficiente tiempo para ser enviado. El valor de dicho delay se fijó en 200  $\mu$ s sin embargo como se ha visto anteriormente, dicho valor no garantiza un correcto funcionamiento.

Ante esta situación se exploran dos opciones:

- Aumentar el valor del “delay” entre cada envío de datos.
- Utilizar los buffers restantes.

Si bien la primera opción es más sencilla de implementar no es una solución viable si se planea transmitir una mayor cantidad de mensajes por ciclo ya que a mayor cantidad de mensajes transmitidos mayor cantidad de “delays” lo que podría comprometer el rendimiento del programa. Por otro lado, utilizando 3 buffers, uno puede estar transmitiendo, el segundo

puede estar esperando a transmitir tan pronto el primero complete la transmisión y finalmente el tercero puede estar disponible para cargar un nuevo mensaje. Esto repercute positivamente en el programa ya que disminuye la necesidad de que este mantenga sincronización con el bus y a su vez elimina el problema de sobre escritura en los buffers.

Por otro lado, el uso de 3 buffers permite un cierto grado de priorización de los mensajes a transmitir, ya que es posible asignar prioridades a cada buffer. De esta manera sin contemplar la prioridad dada por el ID del mensaje, el modulo CAN puede transmitir un mensaje de menor prioridad si este se encuentra cargado en un buffer de mayor prioridad. Por medio del registro C1TXnCON (bits TXPRI) es posible asignar 4 grados de prioridad como se aprecia en la Tabla 3.4.

Valor de bits TXPRI	Prioridad
11	Alta
10	Media Alta
01	Media Baja
00	Baja

Tabla 3.4: Prioridad de buffers de transmisión [25].

Considerando lo anterior se procedió a realizar las modificaciones necesarias para asegurar una correcta transmisión de los datos, en la Figura 3-14 se presenta el diagrama de flujo de la función *cand\_send ()* modificada.

Como se puede apreciar la función modificada contempla si el buffer donde se quiere cargar el nuevo mensaje ya ha realizado o no la transmisión del mensaje anterior. Esto se confirma por medio del valor del bit TXREQ, por lo que solo un buffer que no contenga un mensaje previo o que haya completado la transmisión del mensaje anterior (el CPU libera automáticamente este bit una vez completada la transmisión [28]) estará disponible evitando así una sobre escritura no deseada. Adicionalmente la asignación de prioridades de cada buffer se realiza de manera dinámica asegurándose así que los mensajes sean enviados en el orden en que fueron cargados en cada buffer.

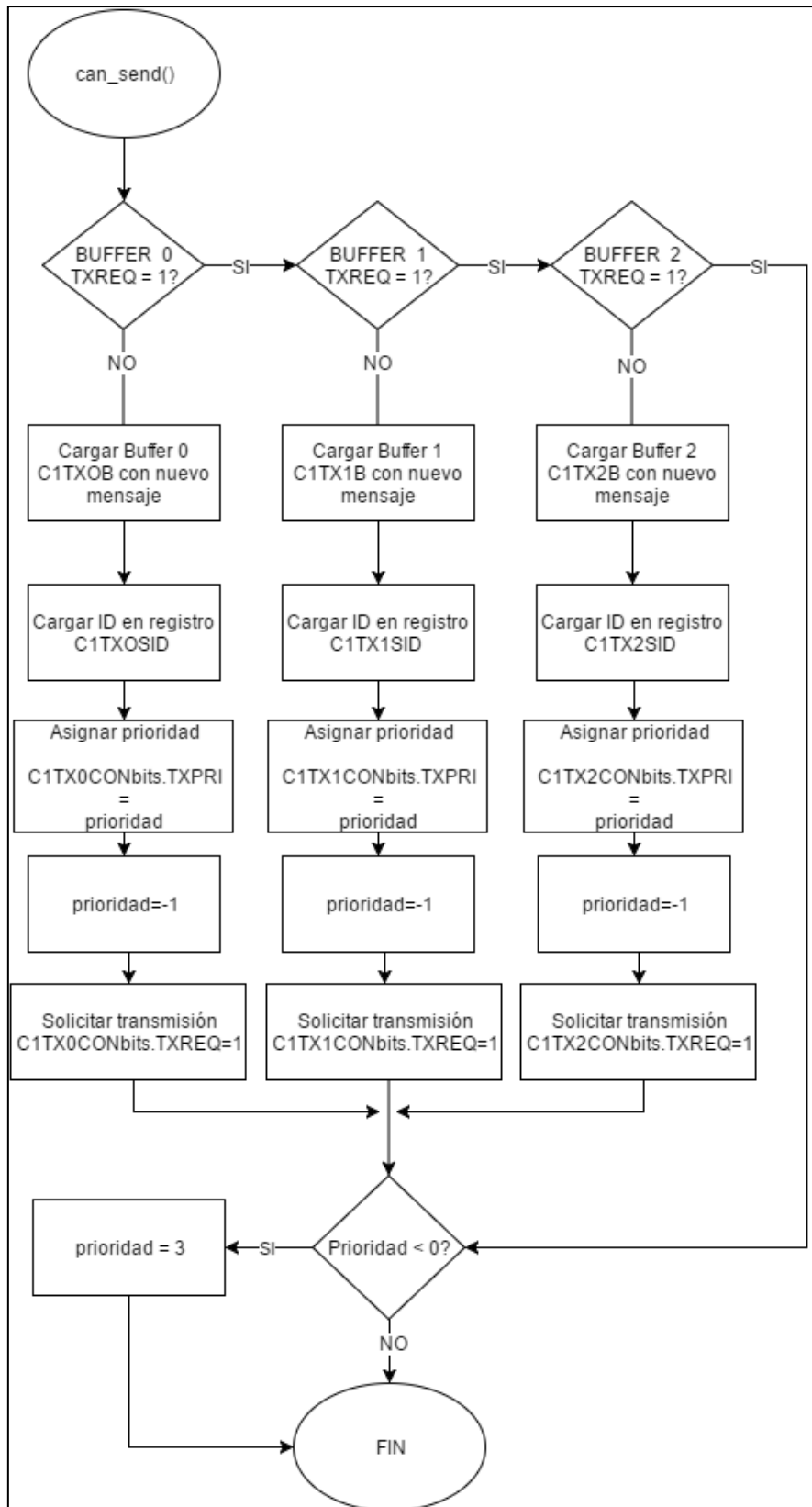


Figura 3-14: Diagrama de flujo de la función *can\_send()* modificada [25].

### 3.5 Resultados.

Una vez implementada la nueva función *can\_send()* se procedió a realizar una nueva toma de datos siguiendo los planteamientos ya descritos. En la Tabla 3.5 se presentan los datos recopilados:

Medición.	Paquete	Porcentaje de error %.	Error cuadrático medio
1	1	3.85	1.73
	2	3.91	1.74
2	1	3.69	1.69
	2	3.88	1.72
3	1	4.05	1.75
	2	3.89	1.73
4	1	3.67	1.70
	2	3.87	1.72
5	1	3.66	1.69
	2	3.77	1.70
6	1	3.90	1.72
	2	3.91	1.73
7	1	3.54	1.67
	2	3.74	1.70
8	1	3.94	1.73
	2	3.87	1.72
9	1	3.85	1.72
	2	4.04	1.75
10	1	3.76	1.70
	2	3.82	1.71

Tabla 3.5: Datos obtenidos luego de implementar la solución[25].

Del mismo modo se procedió a realizar un estudio estadístico de los datos de la Tabla 3.5. Los resultados se encuentran tabulados en la Tabla 3.6. Como se puede apreciar se da una notoria mejoría reduciendo el porcentaje de muestras erróneas a menos del 4% en ambos paquetes. Aún más relevante el error cuadrático medio en ambos paquetes se redujo considerablemente indicando que, si bien aún persisten muestras erróneas, el valor de estas no difiere significativamente del que se esperaría. Cabe mencionar que, si el criterio utilizado para clasificar las muestras se incrementa a un rango de 6 (error absoluto menor a 3), el porcentaje de error desaparece totalmente evidenciando nuevamente que la magnitud del error se redujo drásticamente. Los parámetros máximo, mínimo, rango y % de variación

confirman nuevamente que el grado de variabilidad de los errores tanto en cantidad como en magnitud se redujo notoriamente.

Indicador	Porcentaje de error		Error cuadrático medio	
	Paquete 1	Paquete 2	Paquete 1	Paquete 2
Promedio	3.79	3.87	1.71	1.72
Desviación estándar	0.15	0.08	0.02	0.02
Máximo	4.05	4.04	1.75	1.75
Mín	3.54	3.74	1.67	1.70
Rango	0.51	0.30	0.08	0.05
% Variación	4.07	2.14	1.41	0.94

Tabla 3.6: Estadística después de implementar la solución[25].

Esto se puede apreciar visualmente en el diagrama de cajas y bigotes que se presenta en la Figura 3-15 donde se ve claramente no solo una clara reducción del error (en cantidad y magnitud) sino también de la variabilidad. *Figura 3-11*

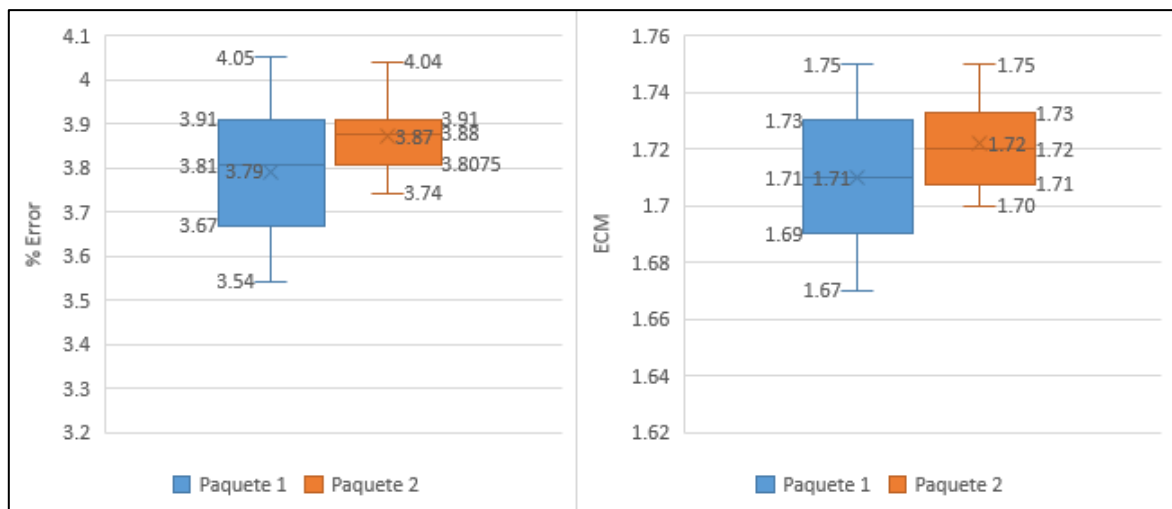


Figura 3-15: Porcentaje de error, ECM y variabilidad en paquetes 1 y 2 después de implementada la solución.

Posteriormente se procedió a comprobar la calidad de la comunicación considerando el caso de mayor demanda que se puede encontrar en el Binocchio. Dicho caso corresponde a 5 nodos de comunicación compuesto por una CPU (PC-104) y 4 SPU's. De esta forma se procedió a realizar una nueva prueba de comunicación, en la que cada SPU envía dos paquetes y recibe uno tal y como se tenía anteriormente. Tanto el paquete recibido como los enviados están conformados por la misma información que se estableció en secciones anteriores (ver Tabla 3.1). De esta forma se tiene entonces un bus de comunicación con 5 nodos y 12 ID's diferentes como se presenta en la Tabla 3.7.

Nodo	Recibe ID	Transmite ID	Filtro	ID Board
PC-104	39	42	-	-
	40			
	71	74		
	72			
	103	106		
	104			
	135	138		
136				
SPU 1	42	39	0X020	32
		40		
SPU 2	74	71	0X040	64
		74		
SPU 3	106	103	0X060	96
		104		
SPU 4	138	135	0X080	128
		136		

Tabla 3.7: Identificadores para cada nodo[25].

Nuevamente los datos obtenidos para cada SPU se guardan de manera a poder realizar el cálculo del porcentaje de muestras erróneas y del error cuadrático medio (ECM) para cada paquete de cada SPU. Se ha tabulado dicha información en la Tabla 3.8.

Medición. Paquete		SPU 1		SPU 2		SPU 3		SPU 4	
		% error	ECM	% error	ECM	% error	ECM	% error	ECM
<b>1</b>	1	5.00	1.87	5.05	1.88	5.15	1.90	5.82	1.99
	2	5.18	1.90	5.17	1.90	5.08	1.89	5.13	1.89
<b>2</b>	1	5.49	1.95	5.63	1.97	5.74	1.98	5.67	1.97
	2	5.09	1.90	4.90	1.87	5.48	1.94	6.11	2.03
<b>3</b>	1	5.04	1.88	5.64	1.96	5.69	1.96	5.67	1.96
	2	5.23	1.90	4.98	1.87	5.55	1.95	6.44	2.06
<b>4</b>	1	5.50	1.94	5.94	2.00	5.32	1.92	5.15	1.89
	2	5.17	1.89	5.04	1.88	5.46	1.93	5.25	1.90
<b>5</b>	1	5.13	1.89	5.22	1.90	5.39	1.92	5.77	1.97
	2	5.31	1.92	5.00	1.87	5.22	1.90	5.43	1.93
<b>6</b>	1	5.70	1.97	5.99	2.00	4.81	1.85	5.03	1.87
	2	5.15	1.90	5.27	1.91	4.90	1.86	5.17	1.89
<b>7</b>	1	5.48	1.93	5.17	1.89	5.23	1.90	5.63	1.95
	2	5.07	1.88	5.30	1.91	5.25	1.90	5.58	1.94
<b>8</b>	1	5.07	1.88	5.62	1.96	5.71	1.96	5.39	1.92
	2	5.22	1.90	5.33	1.92	5.67	1.96	5.52	1.93
<b>9</b>	1	5.46	1.93	5.04	1.88	5.15	1.90	5.82	1.98
	2	5.39	1.93	4.98	1.87	5.08	1.89	5.78	1.98
<b>10</b>	1	5.32	1.90	5.27	1.91	5.74	1.98	5.44	1.93
	2	5.29	1.90	5.32	1.92	5.48	1.94	5.59	1.95

Tabla 3.8: Datos para cada SPU en prueba de mayor exigencia [25]



Seguidamente se realiza un estudio estadístico para un mejor análisis de los datos obtenidos. Los resultados para cada paquete de cada SPU se encuentran tabulados en la Tabla 3.9. Como se puede observar la mayor exigencia al bus de comunicación se traduce por un leve aumento en la cantidad de muestras erróneas sin embargo cabe destacar que en todos los paquetes de cada SPU el porcentaje de muestras erróneas no supera el 6%, esto sumado al hecho que el error cuadrático medio sigue siendo muy pequeño permite deducir que las muestras erróneas no deberían diferir significativamente del valor de la señal original. De hecho, cabe mencionar que al aumentar el rango a 6 (error absoluto menor a 3) como se mencionó anteriormente reduce el porcentaje de error a prácticamente cero en la mayoría de los casos. Los parámetros de máximo, mínimo, rango y % de variación confirman a su vez que el grado de variabilidad de los errores tanto en cantidad como en magnitud es bajo y permite confirmar que pese a un mayor uso del bus de comunicación, este presenta un rendimiento aceptable, donde no se dan los problemas anteriormente expuestos (picos aleatorios, distorsión, etc...). De igual manera esto se puede apreciar visualmente en la Figura 3-15.

Valor	Paquete	SPU 1				SPU 2				SPU 3				SPU 4			
		% Error		ECM		% Error		ECM		% Error		ECM		% Error		ECM	
		1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
<b>Promedio</b>		5.32	5.21	1.91	1.90	5.46	5.13	1.94	1.89	5.30	5.26	1.91	1.91	5.54	5.60	1.94	1.95
<b>Desviación estándar</b>		0.24	0.10	0.03	0.01	0.35	0.17	0.05	0.02	0.34	0.27	0.04	0.04	0.28	0.42	0.04	0.06
<b>Máximo</b>		5.7	5.39	1.97	1.93	5.99	5.33	2	1.92	5.74	5.67	1.98	1.96	5.82	6.44	1.99	2.06
<b>Min</b>		5	5.07	1.87	1.88	5.04	4.9	1.88	1.87	4.81	4.9	1.85	1.86	5.03	5.13	1.87	1.89
<b>Rango</b>		0.7	0.32	0.1	0.05	0.95	0.43	0.12	0.05	0.93	0.77	0.13	0.1	0.79	1.31	0.12	0.17
<b>% Variación</b>		4.56	1.91	1.80	0.74	6.47	3.24	2.49	1.16	6.39	5.15	2.34	1.90	5.02	7.44	2.05	2.96

Tabla 3.9: Estadística de prueba de mayor exigencia[25].

### 3.6 Conclusiones

Se determinó la causa del problema de comunicación presentes (uso limitado de los recursos para realizar la transmisión de datos desde las SPU) y fue posible desarrollar un algoritmo de arbitraje para realizar el envío de datos de forma correcta. De esta forma se logró reducir no solo la cantidad de errores presentes sino también la magnitud de estos, asegurando de este modo una correcta comunicación entre los diferentes dispositivos conectados al bus de comunicación.

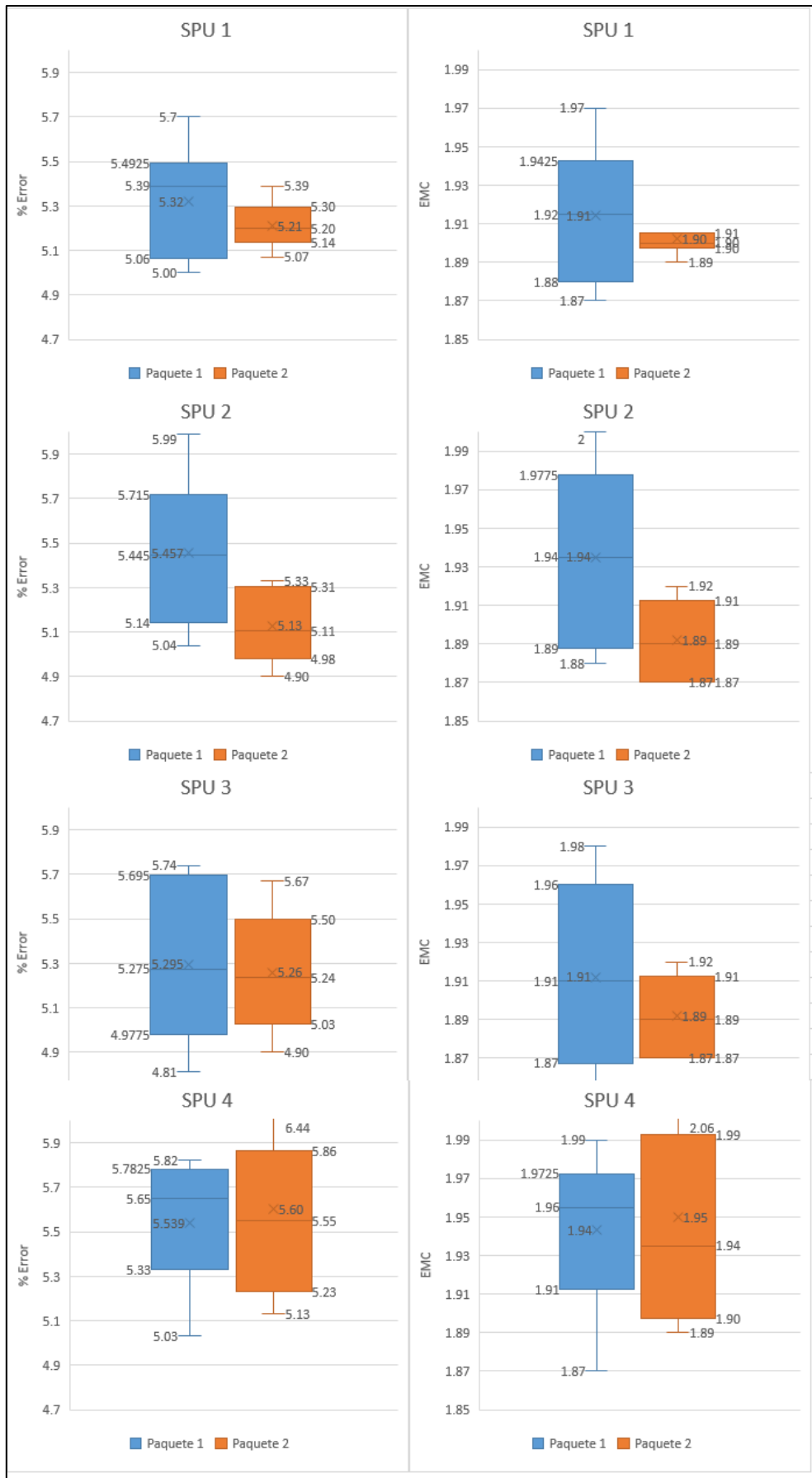


Figura 3-16: Porcentaje de error, ECM y variabilidad en paquetes 1 y 2 de cada SPU

## 4. CAPITULO 4: Estrategia de control.

Este capítulo presenta el trabajo realizado para desarrollar la estrategia de control propuesta. Se expone el proceso previo a la implementación del control FEL del actuador MACCEPA y los principales resultados obtenidos, así como su respectivo análisis. Finalmente se procede a validar el uso del control por medio de pruebas rendimiento que permitan comparar el control convencional y el control FEL.

### 4.1 Caracterización del actuador.

La estructura del actuador desarrollado para el Binnocchio supone todo un reto desde el punto de vista de control, ya que la inclusión de elementos elásticos trae consigo toda una serie de dificultades que se deben solventar. Por un lado, la capacidad del actuador para cambiar su rigidez implica que no se puede depender de un control estático ya que ante dicho cambio se modifica la dinámica del actuador requiriendo un reajuste de las ganancias. Adicionalmente la correa de kevlar utilizada para conectar el Fixed link al resorte que se encuentra alojado en el Link 1 (Figura 4-1 ) agrega características tales como:

- Variación de la tensión en la correa (debido al uso), modificando la dinámica del actuador al variar la rigidez.
- Zona muerta en la que un cambio en la posición de equilibrio del actuador no se traduce en movimiento de la articulación.

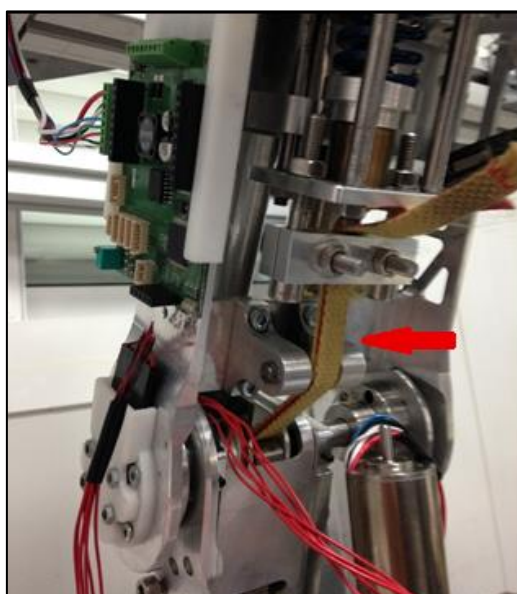


Figura 4-1:Correa de Kevlar a nivel de la rodilla del Binocchio[25].

Mediante una sencilla prueba en la que se procedió a enviar un valor de %PWM al actuador en lazo abierto se determinó que dicha zona muerta es variable y depende de la posición en la que se encuentre el actuador. En la Tabla 4.1, se resume la información obtenida. Como se puede observar la zona muerta puede llegar a ser de hasta  $\pm 7\%$  sin embargo por lo general es menor y puede también depender de la dirección en la que se quiera mover el actuador.

Angulo	Zona Muerta %PWM
0	4
5	$\pm 2.5$
10	-3.5 a +4
15	-2.5 a 3
20	-4 a +2.5
25	$\pm 2.5$
30	-3 a +2.5
35	$\pm 2.5$
40	$\pm 2.5$
45	-5 a +7
50	$\pm 7$

Tabla 4.1: Zona muerta según a posición del actuador ( $p=38.57\text{mm}^1$ )[25].

Por otro lado, se procedió a registrar la respuesta en lazo abierto del actuador, se le envió una señal que establece el valor de %PWM (traducido a corriente por el driver ESCON 50/5) que se desea aplicar en el motor esto con tal de obtener un modelo aproximado del actuador que relacione % PWM con el ángulo de la articulación. En la Figura 4-2 se puede apreciar la entrada (% PWM) y la salida obtenida (ángulo de la articulación) ambos valores se encuentran normalizados<sup>2</sup>. La entrada se escogió de manera que permitiera mover el actuador pero que de igual manera permitiera observar la presencia de la zona muerta y del retraso existente. Como se puede observar a medida que el valor de PWM aumenta (positivamente) no se obtiene ninguna respuesta (a excepción de un ligero movimiento cuando el PWM alcanza el +2%), no es sino hasta alcanzar el +4% que luego un pequeño lapso de tiempo (un retraso de alrededor de 0.25s) que el actuador logra moverse hasta alcanzar una determinada posición (alrededor de 12-13 °). Posteriormente se repite la entrada, pero esta vez con valores negativos. Nuevamente vemos la presencia de una zona muerta y del retraso, ya que no es sino hasta que se alcanza un valor de -2% que el actuador es capaz de moverse y de volver a

<sup>1</sup> Pre compresión del resorte que controla la rigidez del actuador.

<sup>2</sup> El valor de PWM se normaliza con respecto al 100% y el ángulo con respecto a 360°.

la posición inicial. Notar que el actuador llega a la posición inicial antes que se retire la señal de %PWM por lo que podría seguir moviéndose sin embargo dicha posición corresponde al límite mecánico del actuador. Del mismo modo implica que se requiere de menos corriente en una dirección que en la otra para un mismo rango de movimiento.

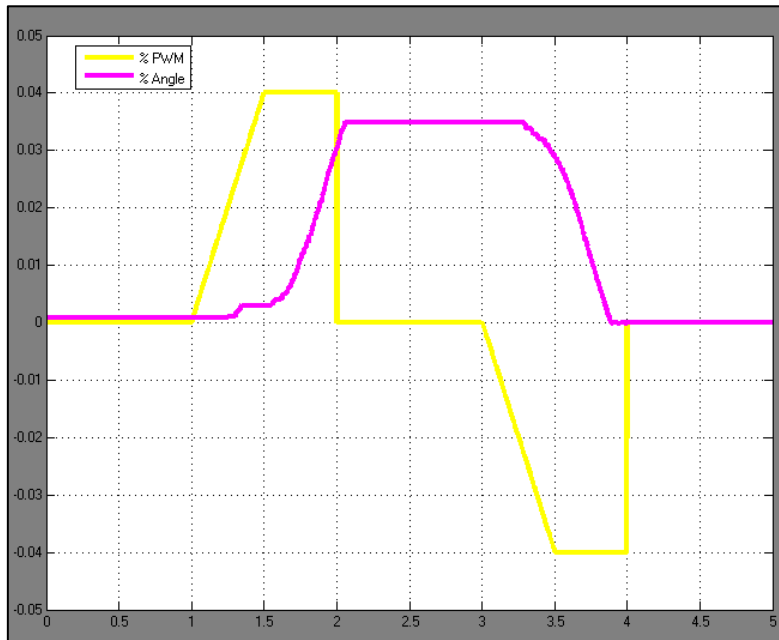


Figura 4-2: Respuesta en lazo abierto del sistema.[25]

Con tal de obtener un modelo aproximado del actuador que relacione el %PWM con el ángulo de la articulación (rodilla), se tomaron 10 mediciones aplicando la misma señal de control anterior. Como se puede observar en la Figura 4-3 la respuesta del sistema es distinta entre una medición y otra, ya que se pueden observar diferencias a nivel del retraso y de la posición alcanzada (se presenta una diferencia de hasta 5° entre los valores extremos).

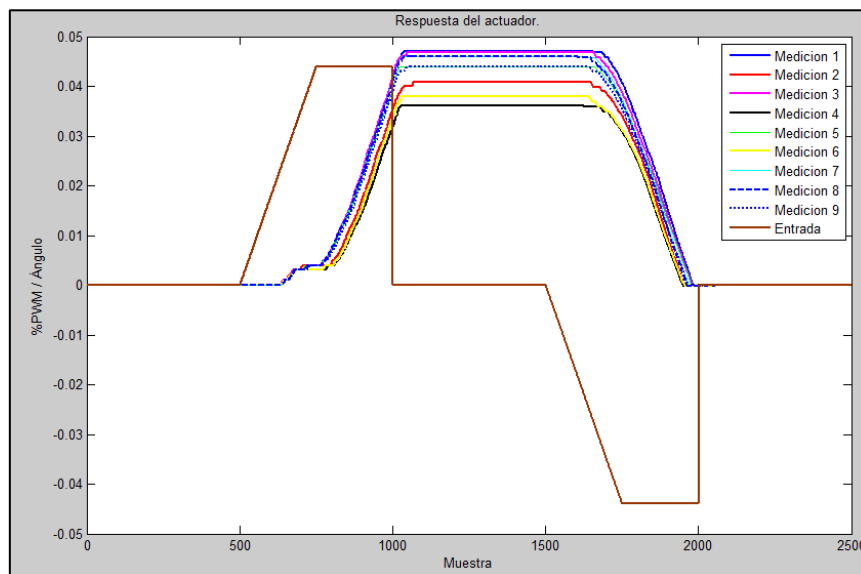


Figura 4-3: Mediciones para modelado empírico[25].

De modo a utilizar dicha información para obtener un modelo empírico se procede a obtener la respuesta promedio del sistema<sup>3</sup> y a procesarlos mediante el uso del System Identification Tool (ident) de Matlab. Mediante la prueba de varias opciones de modelado se obtuvieron varios modelos con distintos porcentajes de ajuste, sin embargo, se consideraron únicamente los 3 modelos con mejor ajuste, estos corresponden a una función de transferencia con un cero, 3 polos y un delay (P2DIZ) y dos modelos no lineales (nlhw1 y nlhw5)<sup>4</sup> que estiman la zona muerta y el retraso.

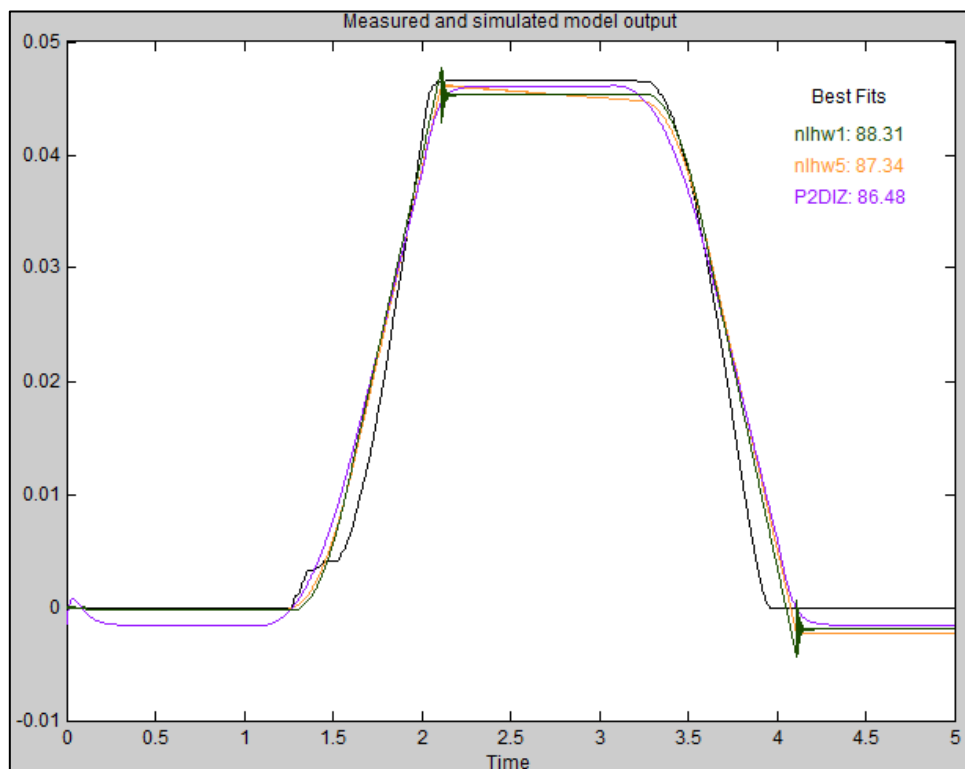


Figura 4-4: Porcentaje de ajuste de modelos obtenidos[25].

No obstante, pese al alto porcentaje de ajuste, dichos modelos no lograron representar adecuadamente la dinámica del sistema en las simulaciones realizadas, ya sea porque omiten por completo las no linealidades (P2DIZ) o porque son incapaces de caracterizar los cambios de estas (nlhw1 y nlhw5) al aumentar el rango de movimiento. Como se mencionó anteriormente (página 17), un modelado preciso implica una mayor complejidad y análisis

<sup>3</sup> Se obtiene al sumar todas las mediciones y dividir entre la cantidad de estas.

<sup>4</sup> Obtenidos por el método Hammerstein-Wiener que estima diferentes tipos de no-linealidades estáticas.

que escapen a los alcances del proyecto y supone en si una razón más a favor de la utilización de los algoritmos capaces de aprender automáticamente esta dinámica.

#### **4.2 Solución planteada**

Como se vio en el apartado anterior el actuador desarrollado para el robot Binnocchio presenta una dinámica no lineal y variante. El control clásico por retroalimentación pese a su sencillez, es por lo tanto insuficiente para un correcto funcionamiento, la implementación de métodos de control no lineal son una opción para afrontar dichas dificultades. El FEL constituye una opción de control no lineal adaptativo con el que se puede obtener un modelo inverso que considere dicha dinámica.

La tesis que se plantea es que usando la señal del control por retroalimentación el MLP pueda obtener el modelo inverso, dando paso a un control feedforward adaptativo. De esta forma, a medida que la señal del MLP vaya tomando mayor protagonismo en el control y por lo tanto mejorando la respuesta del actuador, el PID solo tendrá que hacer pequeñas correcciones que de paso ayudarán al MLP a mejorar, hasta el punto que la señal del PID será prácticamente nula. Considerando lo anterior se supone que, aunque el control por PID no sea totalmente correcto, las capacidades de generalización de la red neuronal permitirán de todas formas obtener un modelo inverso adecuado (probablemente a expensas de un mayor tiempo de aprendizaje).

Las capacidades universales de aproximación del MLP lo hacen una opción muy popular a la hora de modelar sistemas no lineales y/o para implementar controladores no lineales. La red neuronal a utilizar es entonces un perceptrón multicapa (MLP) compuesto por 3 capas (una de entrada, una oculta y una de salida) y para su entrenamiento se utiliza el algoritmo de backpropagation descrito en el anexo A5. Se considera que como máximo dos capas ocultas son suficientes para la mayor parte de aplicaciones, sin embargo según el Teorema de Aproximación Universal, una única capa oculta con suficiente neuronas ocultas nos daría el mismo resultado[11]. Por otro lado, debido a que se trata de una aplicación de control se procede a un entrenamiento “online” o secuencial.

Algunas consideraciones importantes para el correcto funcionamiento de la red neuronal utilizada en el controlador FEL son:

- Inicializar los pesos con valores pequeños (tanto positivos como negativos) siguiendo la regla estipulada en la ecuación (2-1). Esto se conoce como aprendizaje uniforme ya que de esta forma todos los pesos convergen prácticamente al mismo tiempo [11].
- Los valores de las entradas deben ser re-escalados, para nuestro caso se realiza de modo que los valores varíen en un rango de  $\pm 1$ .
- La función de activación en las neuronas ocultas corresponde a la función tangente hiperbólica Figura 4-5. Esta función sigmoidea varía entre  $\pm 1$  y resulta primordial en el correcto funcionamiento del MLP implementado, debido a que la señal de control puede tener tanto valores positivos como negativos y por lo tanto red debe ser capaz de representarlos.

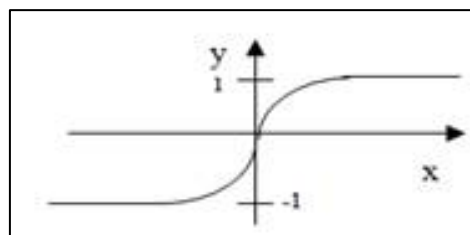


Figura 4-5: Función de activación en la capa oculta [10].

- La función de activación en la neurona de salida es la función identidad.
- Learning Rate dependerá de cada aplicación sin embargo se recomienda que el valor no sea muy alto ya que puede ocasionar oscilaciones.
- El valor del momentum también dependerá de cada aplicación sin embargo es preferible aumentar su valor antes que el del Learning Rate de manera a mantener la estabilidad durante el entrenamiento.

Finalmente, la cantidad de neuronas de entrada y de neuronas ocultas se deben determinar por medio de prueba y error considerando el coste computacional debido al cálculo de una mayor cantidad de pesos sinápticos al aumentar el número de neuronas, ya que entre más compleja sea la red, mayor cantidad de ejemplos y de tiempo serán necesarios para entrenarla.

### 4.3 Control de péndulo invertido

#### 4.3.1 Descripción de la planta.

Con tal de comprobar el correcto funcionamiento de la red neuronal en el controlador FEL y familiarizarse con su uso, se procedió a probar su funcionamiento en una planta más sencilla, previo a su uso en el control del actuador del Binnocchio. Debido a su naturaleza no



lineal e inestable, el péndulo invertido supone un sistema ideal para la prueba de métodos de control. Por ello se construyó rápidamente una planta que se puede ver en la Figura 4-6.



Figura 4-6: Péndulo invertido construido [25].

Como se puede apreciar en la Figura 4-6 la planta está compuesta por 5 componentes (enumerados en la imagen):

- 1- PC-104: CPU encargada del control por medio de Simulink Real Time, envía la señal de control a la SPU por medio de un bus CAN.
- 2- ESCON Module 50/5: Driver para el control del motor.
- 3- SPU: Unidad secundaria de procesamiento utilizada en el control del bípodo Binocchio, realiza la lectura del sensor, el envío y recepción de datos por CAN y la aplicación de la señal de control.
- 4- EC-4pole 22: Motor brushless de 24V, utiliza la misma tecnología que los utilizados en el actuador MACCEPA.
- 5- Péndulo: conectado directamente al eje del motor.

#### 4.3.2 Control Clásico.

Para el control de dicha planta se calibró un controlador PID clásico, las constantes se obtuvieron de manera empírica mediante su ajuste progresivo, hasta obtener un comportamiento aceptable. La idea es que el péndulo pueda seguir una trayectoria (por ejemplo, una senoidal) alrededor de su posición de equilibrio. Dada la forma en la que el

péndulo esta acoplado al motor, su control se hace más complicado ya que ante cualquier perturbación el error puede incrementarse fácilmente a altos valores haciendo que la planta entre en oscilación. Para mitigar esta situación se realiza una compensación de gravedad para facilitar la acción de control (también obtenida empíricamente). En la Figura 4-7 se puede apreciar el controlador en cuestión:

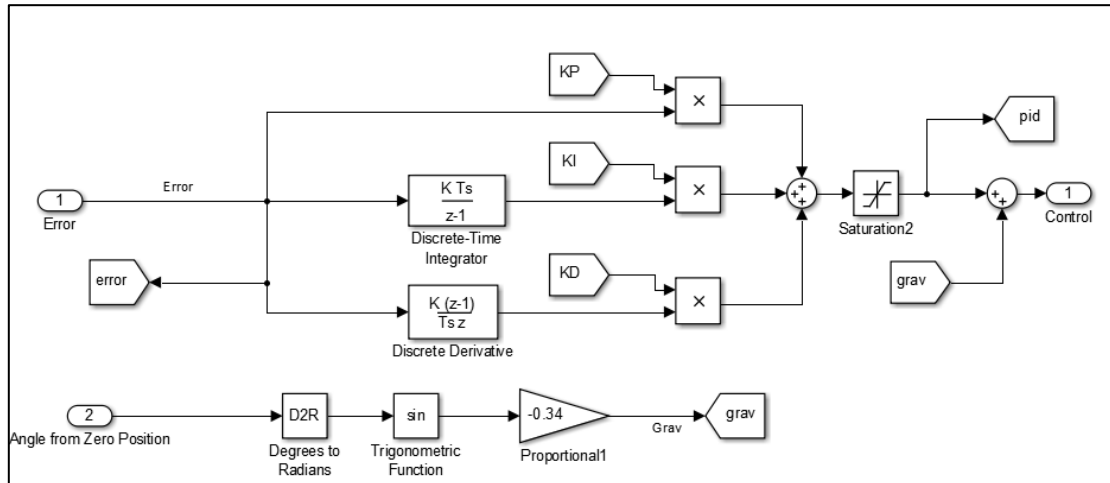


Figura 4-7: Controlador PID y compensación de gravedad [25].

La compensación de gravedad como su nombre lo implica proporciona el torque necesario para compensar el peso del péndulo cuando este difiere de su posición de equilibrio (función del ángulo) por lo que permite que el péndulo pueda permanecer en una determinada posición sin caer. Dicho torque se modela de la siguiente manera:

$$\tau = K * \sin \theta \tag{4-1}$$

Donde K es un constante que depende de parámetros como: la gravedad, la masa del péndulo, longitud del péndulo y demás constantes eléctricas en el motor, como se puede ver en la Figura 4-8

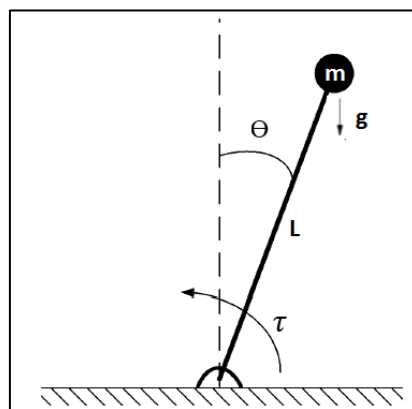


Figura 4-8: Torque debido al peso del péndulo[25].

De esta forma se calibró el PID de manera que el péndulo pudiera seguir una trayectoria determinada por una onda senoidal de amplitud 10 grados y frecuencia de 0.05 Hz (muestreo a 0.002s). El resultado se puede observar en la Figura 4-9.

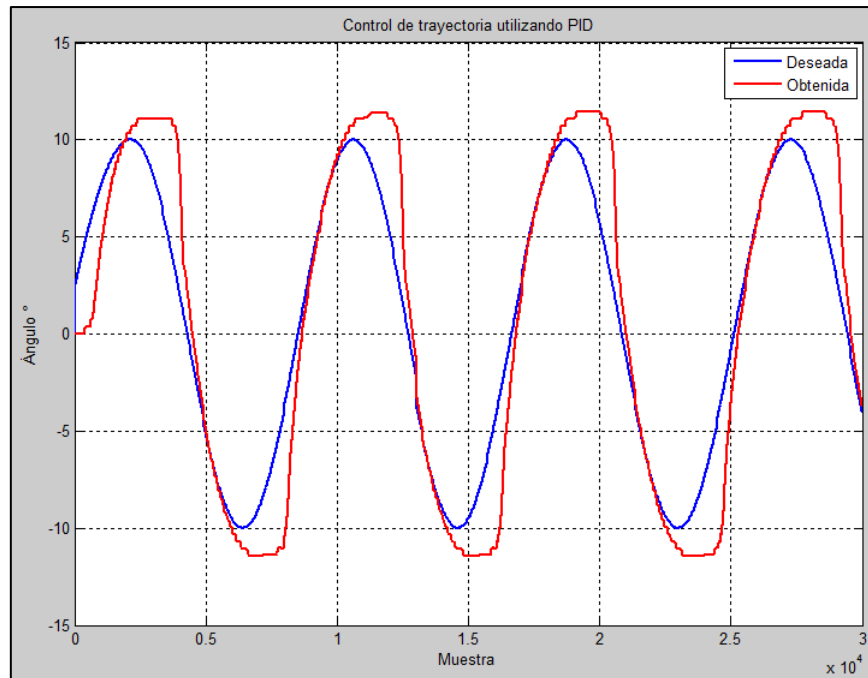


Figura 4-9: Control de trayectoria por PID [25].

Se puede observar que el control clásico no ejecuta una trayectoria totalmente satisfactoria, ya que se puede apreciar un desfase entre las señales además de un error significativo (ver Tabla 4.3), sin embargo, esto no supone un problema ya que nos permitirá más adelante apreciar más fácilmente el efecto de la red neuronal sobre el control de la planta. Por otro lado, a esta frecuencia se puede obtener un mejor desempeño del PID al aumentar más las ganancias, no obstante, esto resulta contraproducente ya que altas ganancias pueden provocar oscilaciones más fácilmente en caso de perturbaciones. Como se dijo anteriormente el control por retroalimentación no debería ser necesariamente perfecto (aunque es deseable) para que la red neuronal pueda obtener el modelo inverso, pero si debe asegurar cierta estabilidad de manera que la planta sea capaz de describir una trayectoria que, aunque no sea perfecta no implique inestabilidad.

### 4.3.3 Controlador FEL.

La red neuronal utilizada en el controlador FEL se implementó siguiendo los lineamientos expuestos en la sección 4.2 las entradas, cantidades de neuronas de entrada, la cantidad de neuronas ocultas, el Learning Rate y el Momentum se determinaron mediante prueba y error y se exponen en la Tabla 4.2.

Parámetro	Valor
Entradas	Posición, Velocidad y Aceleración deseadas
# Neuronas de entrada	9
# Neuronas ocultas	9
Learning Rate	0.3
Momentum	0.4
Periodo de muestreo (s)	0.002

Tabla 4.2: Parámetros del MLP para el control del péndulo.

Como se ve en la Tabla 4.2, se tienen únicamente 3 entradas, pero 9 nodos de entrada, este se debe a que por cada entrada se toman 3 puntos: el punto actual y los dos puntos anteriores. Por otro lado si bien no existe una teoría para determinar la cantidad de nodos en la capa oculta, se recomienda la cantidad sea igual o mayor al número de nodos de entrada [18]. En la Figura 4-10 se puede apreciar la trayectoria realizada por el péndulo utilizando el FEL. Se constata una mejoría tanto en términos de fase como en términos del error. Sin embargo, persisten algunos problemas como cuando se da el cambio de dirección (donde se ve el mayor error).

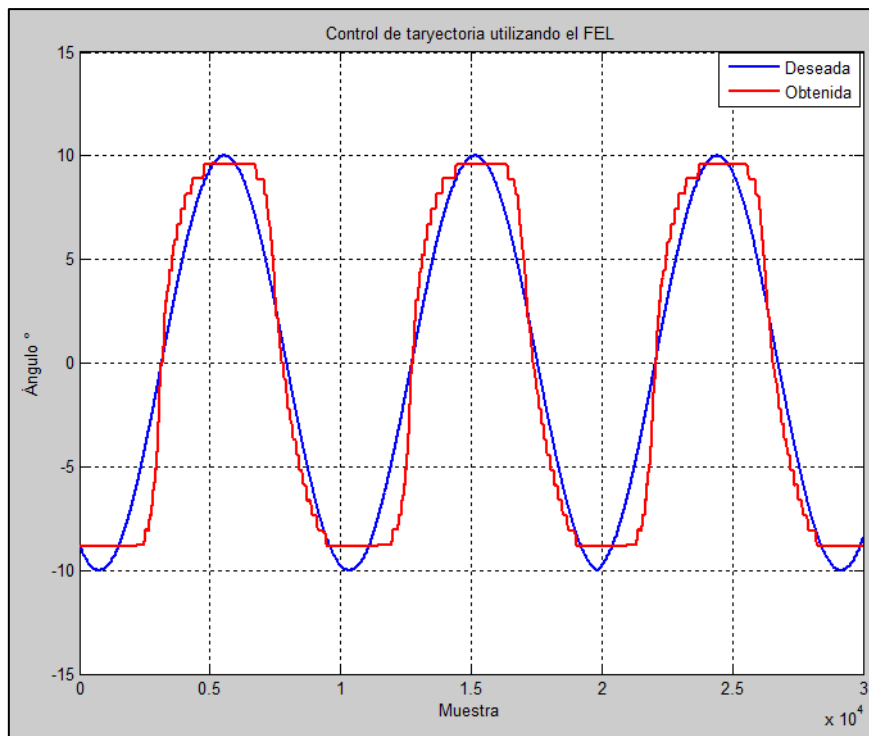


Figura 4-10: Control de trayectoria utilizando el FEL [25].

A continuación, se presenta la evolución del error eficaz o “RMS” a conforme la red neuronal aprende el modelo inverso (Figura 4-11). Como se puede ver el error RMS inicialmente alto se reduce conforme avanza el entrenamiento hasta un punto donde la reducción es cada vez menor, indicando que los pesos sinápticos están convergiendo o están cerca de hacerlo, para este punto la red neuronal no será capaz de aprender mucho más (si se mantienen las mismas condiciones). Del mismo modo debe notarse que el error eficaz o “RMS” sigue una tendencia decreciente a lo largo de los 30 ciclos de la trayectoria senoidal que se registraron.

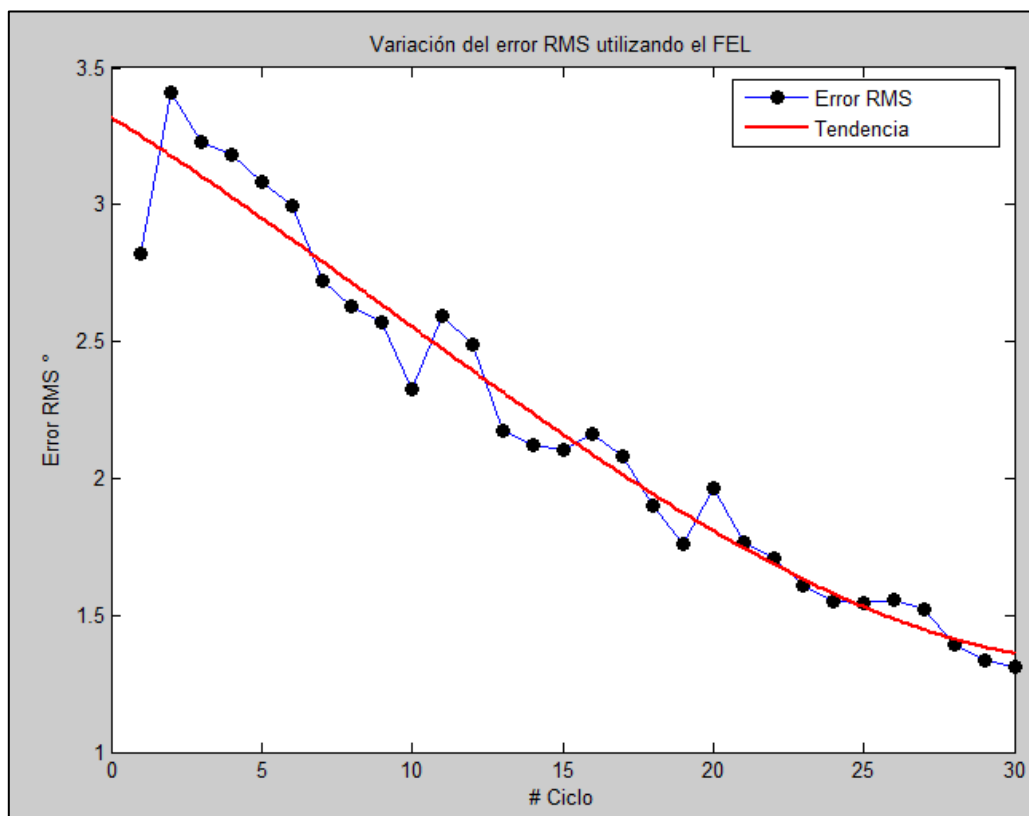


Figura 4-11: Variación del error RMS en el control del péndulo invertido [25].

Ante esta situación se buscaron formas de acelerar y mejorar el proceso de aprendizaje. La más obvia supone el aumento del Learning Rate, no obstante, esto puede resultar en oscilaciones, al hacer que los valores de los pesos diverjan. Se analizó la posibilidad de utilizar el método “batch” que converge más rápidamente, sin embargo, la necesidad de tener que separar la fase de entrenamiento de la fase de operación resulta poco atractiva para control no lineal de sistemas variantes en el tiempo (como lo puede ser el MACCEPA). En el método secuencial el orden en que se realizan los cambios a los valores de los pesos puede tener un efecto sobre el rendimiento de la red [11]. Por ello una solución radica en procesar la

información de manera aleatoria, esto se conoce como “gradiente descendiente estocástico”. Esta solución se resume en entrenar a la red de manera aleatoria esto puede tener efectos positivos tanto en el tiempo de aprendizaje como en la reducción del error RMS ya que las fluctuaciones aleatorias permiten al algoritmo escapar de los mínimos locales más fácilmente. Para su implementación se procedió a habilitar y deshabilitar la opción de aprendizaje de la red de forma estocástica. A continuación, se muestra la trayectoria obtenida al realizar el aprendizaje de manera aleatoria (Figura 4-12). Se puede apreciar una mejoría con respecto a la que se tenía con la red neuronal entrenada de manera totalmente secuencial.

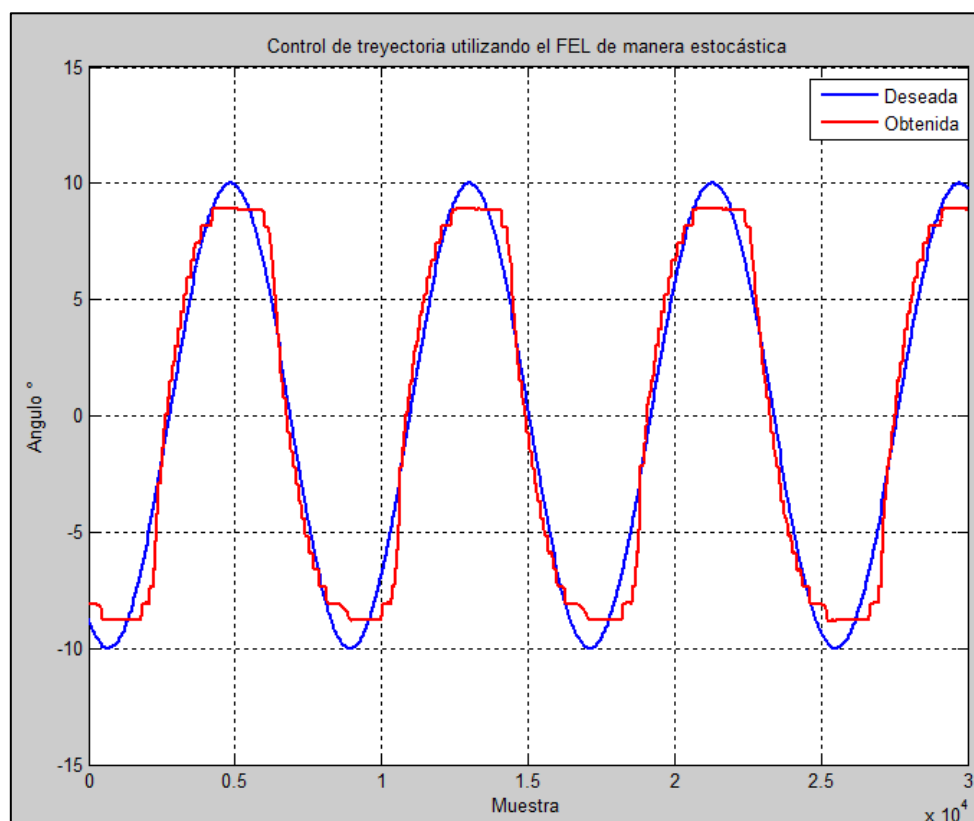


Figura 4-12: Trayectoria obtenida con FEL entrenado estocásticamente [25]

Del mismo modo al observar la variación del error RMS en la Figura 4-13 a medida que pasan los ciclos, se puede apreciar cómo el error RMS empieza a estabilizarse cerca del ciclo #12 (240 s) lo que representa una gran reducción del tiempo de aprendizaje ante los 30 ciclos que se tenían anteriormente. También mencionar que el error RMS desde el ciclo 12 al 30 es menor al que se puede observar en la Figura 4-11

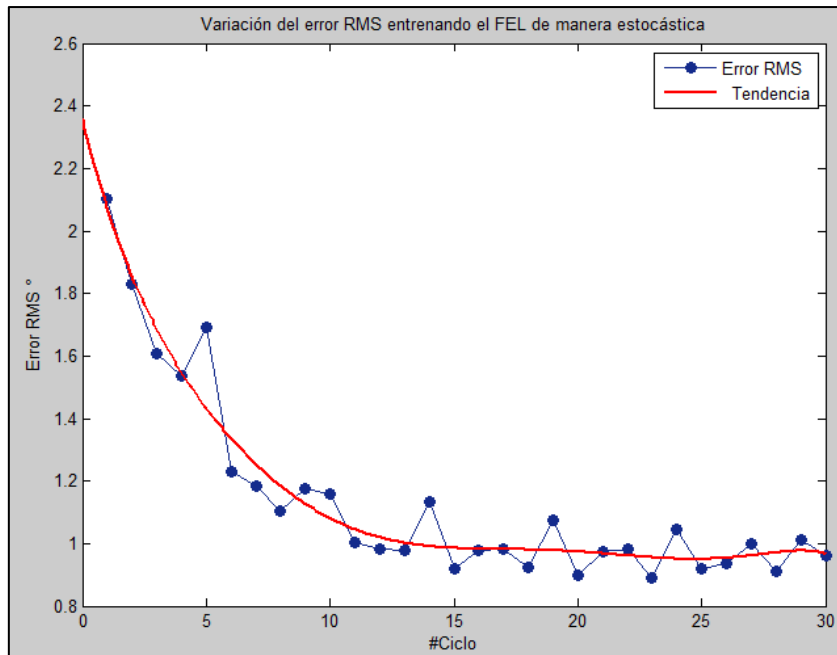


Figura 4-13: Variación del error luego de modificar el método de entrenamiento [25].

#### 4.3.4 Análisis de Resultados.

Con tal de demostrar cuantitativamente el correcto funcionamiento del FEL se procedió a comparar el rendimiento del control por PID<sup>5</sup> con el del FEL. Se analizarán el error RMS, el retraso existente entre la trayectoria deseada y la obtenida, así como el aporte de cada componente (PID y red neuronal) en la señal de control.

Frecuencia (Hz)	Control	Amplitud (°)	Error RMS (°)	Delay (s)
0.05	PID	5	2.6	0.96
		10	3.24	0.48
	FEL	5	0.6	0
		10	1.19	0

Tabla 4.3: Comparación del error y el retraso entre PID y FEL<sup>6</sup> [25].

<sup>5</sup> Las ganancias del PID se mantuvieron constantes e iguales a las utilizadas para realizar el entrenamiento de la red neuronal.

<sup>6</sup> Valores calculados utilizando los últimos 8 ciclos de cada toma de datos. Dicha metodología se conserva a lo largo del presente documento a menos que se indique lo contrario.

A partir de la Tabla 4.3, el uso del controlador FEL permite obtener un rendimiento superior. Se utilizó la función “finddelay” de Matlab para encontrar el “delay” o retraso entre la trayectoria deseada y la obtenida. Como se puede observar el uso del FEL eliminó el retraso que existía lo cual indica que es capaz de predecir correctamente los valores de control necesarios para llevar al péndulo a la posición siguiente. Del mismo modo vemos una reducción importante en el valor del error RMS, ya que para el caso de una amplitud de 5 grados tenemos una disminución<sup>7</sup> de 79.92%, mientras que para el caso de una amplitud de 10 grados la disminución es de 63.27%.

Frecuencia (Hz)	Control	Amplitud	RMS Control (%PWM)	RMS MLP (%PWM)	RMS PID (%PWM)	% RMS MLP	% RMS PID	Aprendizaje (# Ciclos)
0.05	PID	5	15.91	0	15.91	0	100	-
		10	20.06	0	20.06	0	100	-
	FEL	5	14.85	14.45	3.34	95	5	13
		10	18.76	18.26	4.34	94.7	5.3	12

Tabla 4.4: Comparación entre el aporte del PID y el MLP a la señal de control para cada esquema de control [25].

Como se mencionó en la sección 2.4, finalizado el entrenamiento el MLP debería de aportar la mayor parte de la acción de control. Dado que la señal de control es una señal oscilante con valores tanto positivos como negativos se requiere del valor RMS (Valor eficaz) para poder compararlas<sup>8</sup>. Como se puede ver en la Tabla 4.4 el PID deja de aportar el 100% del valor eficaz de la señal de control cuando este trabaja solo y pasa a aportar únicamente un 5% una vez entrenado el MLP. Esto se puede corroborar visualmente al comparar las señales de control provenientes del PID y el MLP antes y después del entrenamiento como se ve en la Figura 4-14.

<sup>7</sup> Porcentaje de reducción calculado como:  $\% = \left( \frac{Error_{PID} - Error_{FEL}}{Error_{PID}} \right) * 100$

<sup>8</sup> El valor eficaz de una combinación de ondas es:  $V_{rms} = \sqrt{V_1^2 + V_2^2 + \dots + V_n^2}$



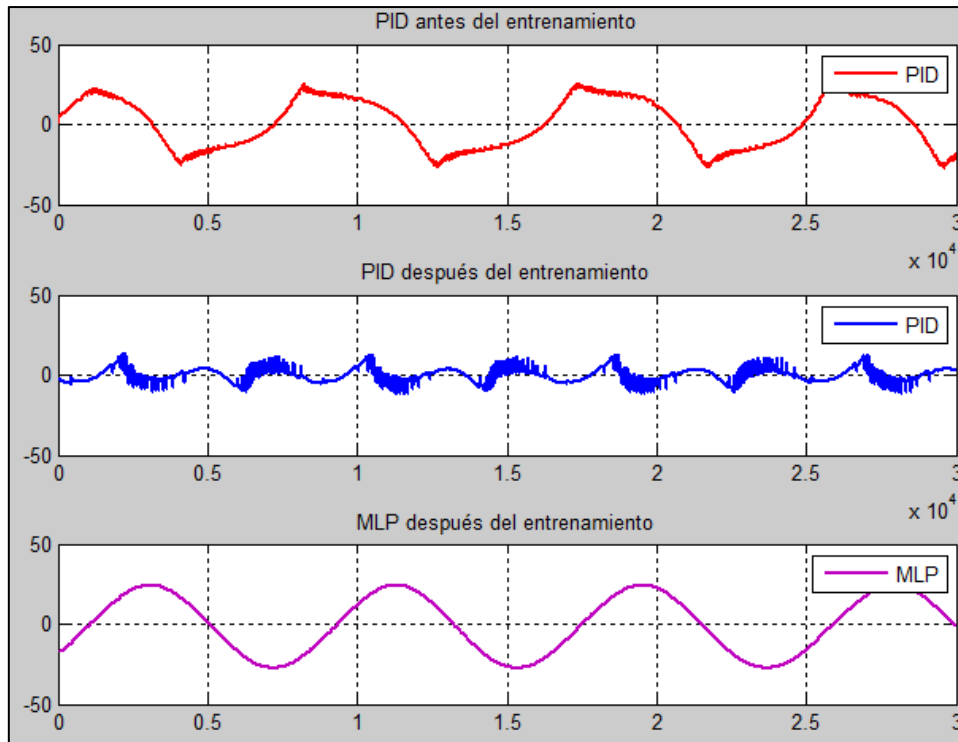


Figura 4-14: Señales de control antes y después del entrenamiento [25].

#### 4.4 Control del actuador de rigidez variable.

Una vez corroborado el correcto funcionamiento del MLP para el control de una planta no lineal se procedió a aplicar lo aprendido en el control del actuador del Binnocchio. A continuación, se presenta la comparación entre el PID y el FEL, así como las pruebas realizadas para validar su uso.

##### 4.4.1 Control Clásico.

Nuevamente se calibró un controlador PID clásico, las constantes se obtuvieron de manera empírica mediante su ajuste progresivo, hasta obtener un comportamiento aceptable. La idea es que el péndulo pueda seguir una trayectoria senoidal<sup>9</sup>. Se buscó evitar ganancias muy altas para evitar oscilaciones en caso de perturbaciones o durante la etapa de aprendizaje. Del mismo modo se calibró una compensación de gravedad ( $K=0.25$ ) que contrarreste el peso de la articulación, el controlador en cuestión se mantiene igual con respecto a lo expuesto en la Figura 4-7. Como se puede observar en la Figura 4-15 el control clásico no ejecuta una trayectoria satisfactoria, ya que se puede apreciar un desfase entre las señales además de un error significativo (Ver Tabla 4.6).

<sup>9</sup> Amplitud de 10 grados y frecuencia de 0.5 Hz.

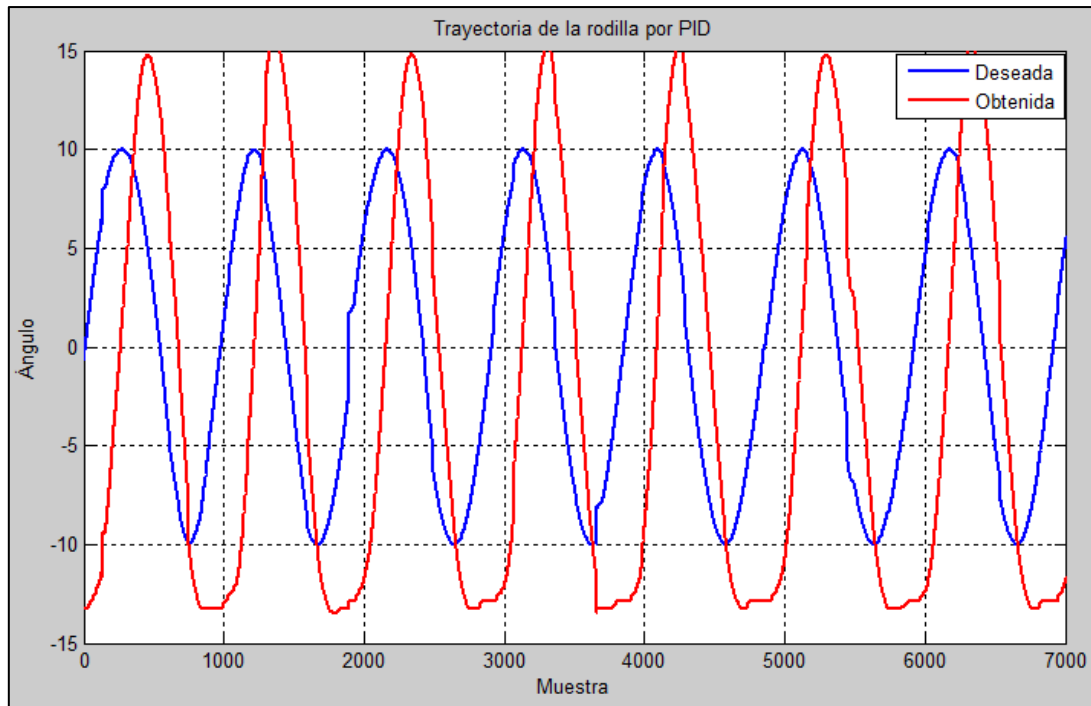


Figura 4-15: Trayectoria mediante PID [25].

#### 4.4.2 Controlador FEL.

La red neuronal se configuró siguiendo lo estipulado en la Tabla 4.5. Se determinó mediante prueba y error que era posible reducir la cantidad de entradas sin comprometer el aprendizaje ni la reducción del error, esto puede repercutir positivamente en el tiempo de aprendizaje ya que se deben calcular menos pesos sinápticos. Por otro lado, se descubrió que el valor asignado al “bias node” de la salida tiene un efecto sobre la forma en que la red neuronal inicia la etapa de aprendizaje, si esta conserva su valor recomendado (-1) el valor de la señal de control Uff dada por la red neuronal toma valores negativos muy altos, provocando una fuerte perturbación que puede desestabilizar la planta. Ante ello se decidió disminuir dicho valor a “-0.7” permitiendo que el aprendizaje inicie de forma más estable.

Parámetro	Valor
Entradas	Posición, Velocidad y Aceleración deseadas
# Neuronas de entrada	3
# Neuronas ocultas	9
Learning Rate	0.6
Momentum	0.8
Periodo de muestreo (s)	0.002
Bias node o	-0.7

Tabla 4.5: Parámetros del MLP para el control del actuador [25].

En la Figura 4-16 se puede apreciar una mejoría considerable al utilizar el FEL, tanto el error como el desfase se han reducido significativamente (Ver Tabla 4.6) y el actuador es capaz de seguir la trayectoria deseada.

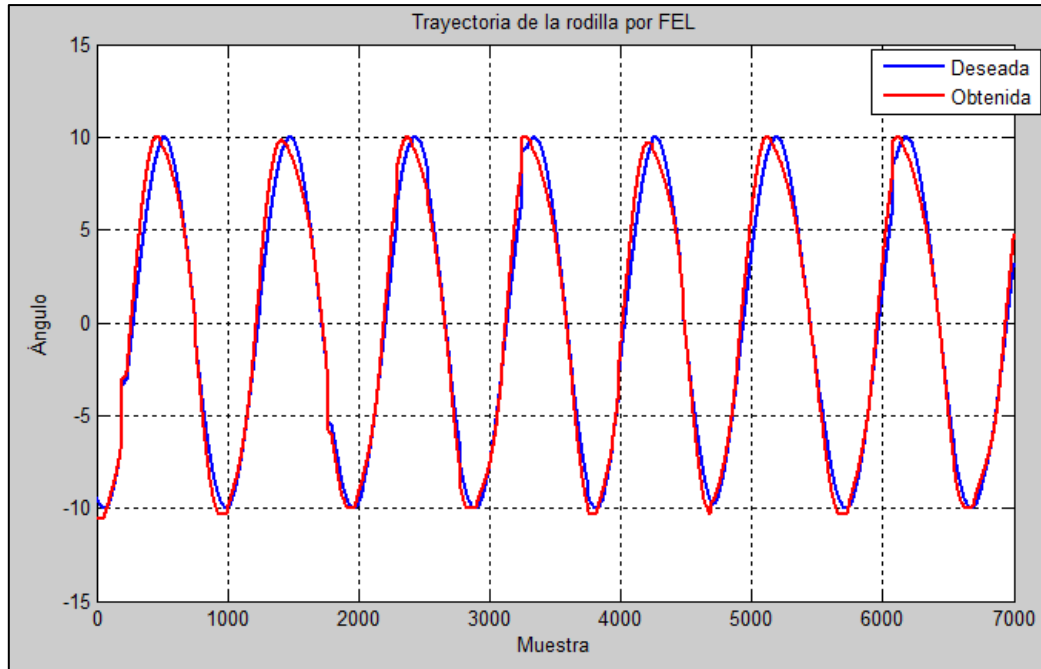


Figura 4-16: Trayectoria del actuador mediante FEL [25].

A continuación, se presenta la evolución del error RMS entre la trayectoria deseada y la obtenida a lo largo del entrenamiento (Figura 4-17). Se puede apreciar que la tendencia del error es decreciente hasta después de 50-60 ciclos (120s aproximadamente) donde el error se estabiliza por lo que se puede considerar que el entrenamiento ha finalizado.

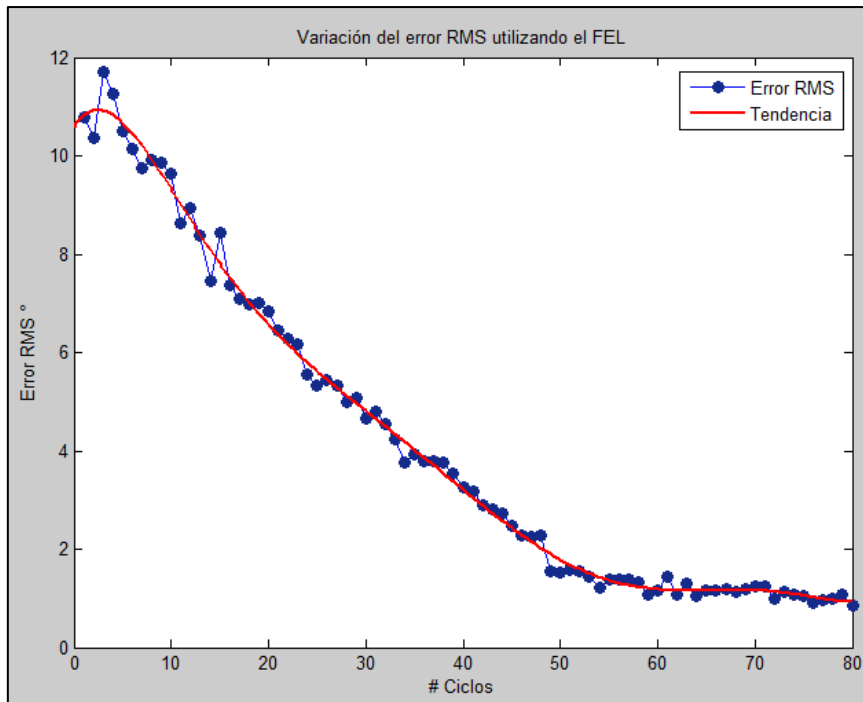


Figura 4-17: Error RMS en la trayectoria efectuada por la rodilla [25].

Resulta también interesante observar la variación del valor medio de la señal de control por retroalimentación “Ufb” a medida que se da el entrenamiento de la red neuronal. Como se puede observar en la Figura 4-18 dicha evolución sigue una tendencia similar a la del error RMS de la Figura 4-17 y se estabiliza cerca del mismo número de ciclos. Esto es de esperar ya que un error mínimo a la salida del actuador implica que el control por retroalimentación no debe generar ninguna acción correctiva.

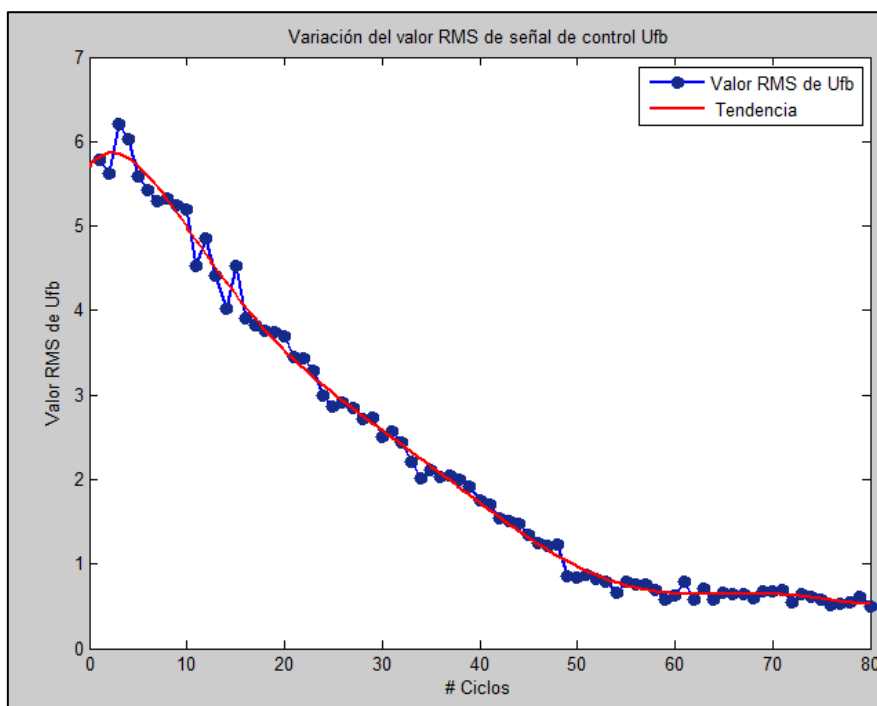


Figura 4-18: Variación del valor medio de señal de control Ufb [25].

### 4.4.3 Análisis de Resultados.

Nuevamente con tal de demostrar cuantitativamente el correcto funcionamiento del FEL se procedió a compararlo con el rendimiento del control por PID<sup>10</sup>. Se analizarán el error RMS, el retraso existente entre la trayectoria deseada y la obtenida, así como el aporte de cada componente (PID y MLP) en la señal de control.

Frecuencia (Hz)	Control	Amplitud (°)	Error RMS (°)	Delay (s)
0.5	PID	5	3.53	0.53
		10	7.06	0.37
		15	10.62	0.35
	FEL	5	0.72	0.01
		10	0.96	-0.02
		15	1.15	0

Tabla 4.6: Error y el retraso obtenidos por PID y FEL para control del actuador [25]

A partir de la Tabla 4.6, el uso del FEL permite obtener un rendimiento superior. Como se puede observar se elimina o reduce el retraso que existía entre la trayectoria deseada y la obtenida, notar que para el caso de una amplitud de 10 grados se presente un retraso negativo, lo que indica que el control hace que el actuador adelante a la trayectoria deseada, no obstante, su valor es pequeño si se compara con el retraso que existe al utilizar solo el PID (lo mismo aplica para el caso de la amplitud de 5 grados). Del mismo modo vemos una reducción importante en el valor del error RMS, ya que se tiene una reducción de 79.60%, 86.80 % y 89.17% para las trayectorias de 5 ,10 y 15 grados de amplitud respectivamente.

---

<sup>10</sup> Nuevamente las ganancias del PID se mantuvieron constantes e iguales a las utilizadas para realizar el entrenamiento de la red neuronal.

Frecuencia (Hz)	Control	Amplitud	RMS Control (%PWM)	RMS MLP (%PWM)	RMS PID (%PWM)	% RMS MLP	% RMS PID	Aprendizaje (# Ciclos)
0.5	PID	5	2.88	0	2.88	0	100	-
		10	5.52	0	5.52	0	100	-
		15	7.36	0	7.36	0	100	-
	FEL	5	2.83	2.80	0.39	98.59	1.91	55
		10	4.68	4.65	0.54	98.72	1.33	50
		15	6.03	6.00	0.62	99.00	1.00	40

Tabla 4.7: Aporte del PID y el MLP a la señal de control del actuador.

Como se puede ver en la Tabla 4.7 el PID deja de aportar el 100% del valor “RMS” de la señal de control cuando este trabaja solo y pasa a aportar menos del 2%, una vez entrenado el MLP. Esto se puede corroborar visualmente al comparar las señales de control provenientes del PID ( $U_{fb}$ ) y el MLP ( $u_{FF}$ ) después de 50 ciclos (100 s) como se ve en la Figura 4-19.

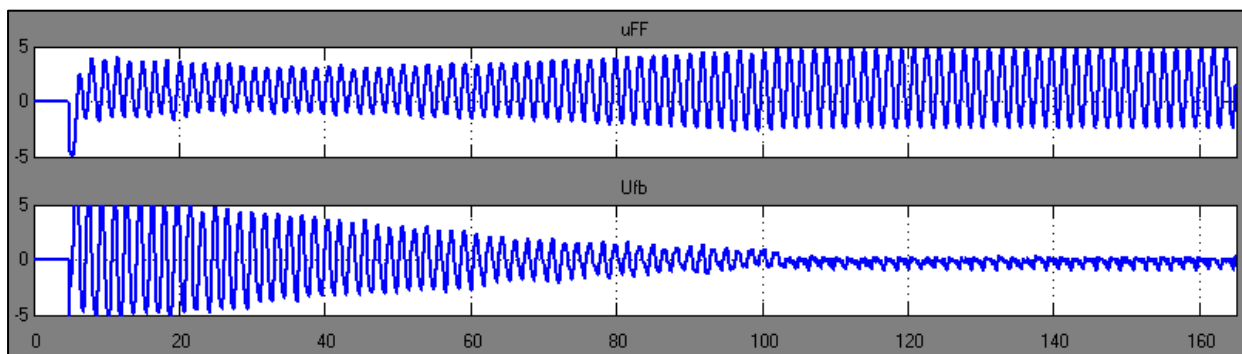


Figura 4-19: Variación de señales de control durante el entrenamiento[25].

#### 4.5 Pruebas para validación del FEL.

A partir de la información anterior es seguro afirmar que la red neuronal es capaz de aprender la a dinámica del sistema de manera a aportar la señal de control necesaria para controlar la trayectoria de la articulación de forma satisfactoria (error y desfase mínimos y/o nulos). Sin embargo, queda por corroborar si dicho aprendizaje responde a una especialización para una determinada trayectoria y/o condiciones de operación específicos o si por el contrario el modelo aprendido es capaz de generalizar para otras condiciones de operación. A continuación, se presentan pruebas realizadas para comprobar la robustez y estabilidad del controlador FEL.

#### 4.5.1 Pruebas de robustez

- **Cambio de dinámica por perturbación estática**

Para corroborar la robustez del control por medio del FEL se procedió a cambiar la dinámica del actuador al agregar peso adicional (de 2 y 3 kg)<sup>11</sup>. En la Figura 4-20 se puede apreciar el montaje realizado correspondiente a la carga adicional de 3 kg, esto resulta en un cambio en el peso del robot y de la localización de su centro de masa. Esto puede verse desde el punto de vista de operación como utilizar el robot para cargar un objeto o mochila, lo que modificaría el peso y la distribución de masas y por ende la dinámica del robot al caminar. Se ejecutaron las mismas trayectorias anteriores<sup>12</sup> y se utilizaron los valores de los pesos obtenidos durante el entrenamiento como punto de partida (red neuronal previamente entrenada).

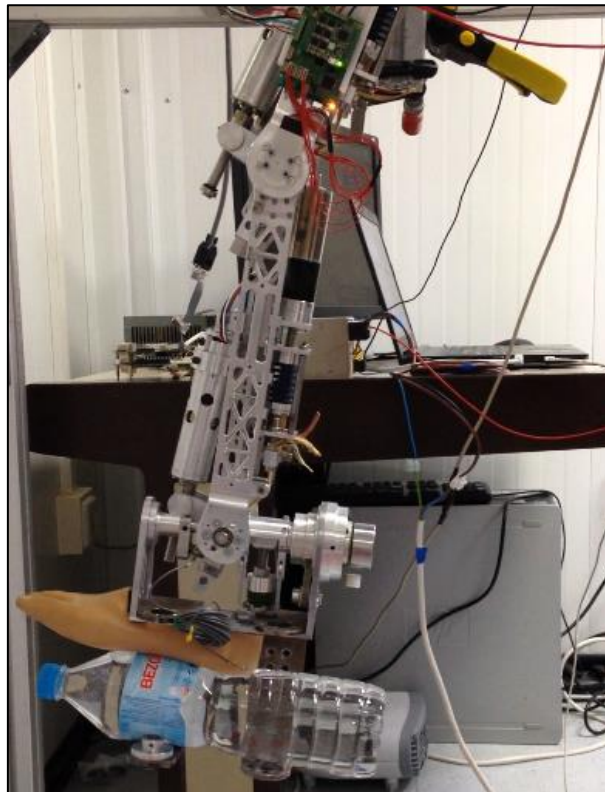


Figura 4-20: Perturbación estática [25].

Dicha prueba permitió corroborar por un lado la rapidez con la cual el sistema es capaz de adaptarse ante la nueva dinámica además de preservar el rendimiento obtenido anteriormente, luego de finalizado el aprendizaje de las nuevas características de la planta. Como se puede

---

<sup>11</sup> Las cargas representan 26% y 40% del peso total de la pierna (7.56 kg [1])

<sup>12</sup> Frecuencia de 0.5 Hz con amplitudes de 5°, 10° y 15°

ver en la Figura 4-21 la red neuronal es capaz de aprender la nueva dinámica y reducir el error, ya que el valor “RMS” inicialmente alto se reduce paulatinamente a medida que el MLP aprende las nuevas condiciones de operación. Cabe mencionar que para que esto ocurra el aprendizaje debe estar activado ya que, de lo contrario, el modelo inverso precargado inicialmente en el MLP no será capaz de reducir el error. Esto nos indica que se debe pasar por un nuevo periodo de entrenamiento que permita al MLP aprender la nueva dinámica, no obstante, dicho entrenamiento no requiere de tanto tiempo como el que se vio en la sección anterior debido a que se parte de una base previa que ya contempla gran parte de la dinámica del sistema.

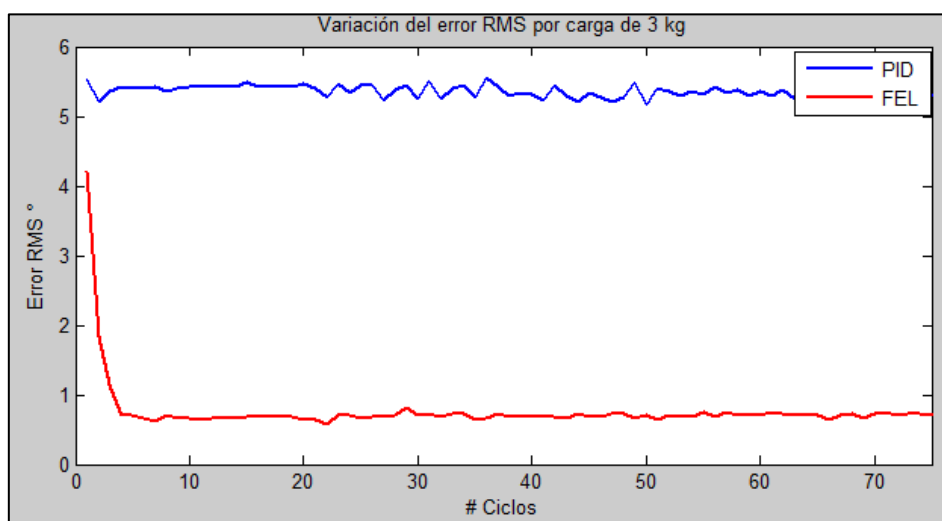


Figura 4-21: Variación del error RMS debida a carga de 3 kg [25].

Por otro lado, como es de esperar el PID (línea azul en Figura 4-21) por si solo resulta insuficiente para garantizar un buen rendimiento, ya que el valor “RMS” del error se mantiene relativamente constante. Por otro lado, es interesante resaltar que el error inicial obtenido con el FEL es menor al que se tiene con el PID. Esto nos indica que a pesar de que el modelo inicial en el MLP no es capaz de contemplar completamente la nueva dinámica, su rendimiento sigue siendo mejor que el del PID. Adicionalmente en la Tabla 4.8 se puede apreciar la diferencia entre el error inicial utilizando el FEL y el error con el PID. En todos los casos expuestos que puede corroborar que incluso inicialmente el FEL tiene un mejor rendimiento que el PID clásico. Como es de esperar luego del periodo de adaptación el error se ha reducido significativamente con respecto a su valor inicial. Del mismo modo pese a que existe un retraso entre la trayectoria deseada y la obtenida por el FEL, este tiene un valor pequeño si se compara con el que se tiene mediante un control por PID únicamente.



Frecuencia (Hz)	Carga	Amplitud (°)	PID		FEL			% Reducción del error inicial
			Error RMS (°)	Delay (s)	Error RMS Inicial (°)	Error RMS Final (°)	Delay (s)	
0.5	2 kg	5	4.14	-0.48	3.55	0.94	0.07	73.52
		10	8.27	-0.56	5.45	1.50	0	72.48
		15	11.02	0.35	8.59	1.71	0	80.09
	3g	5	4.11	0.48	3.58	0.80	-0.02	77.65
		10	7.24	-0.56	5.26	1.41	0.04	73.19
		15	9.92	-0.65	8.22	1.28	0.02	84.43

Tabla 4.8: Error y retraso en prueba de cambio de dinámica [25].

Finalmente, en la Tabla 4.9 se puede confirmar que la señal de control Uff (Red neuronal) domina la acción de control luego del proceso de adaptación ya que en ningún caso el PID aporta más del 4%

Frecuencia (Hz)	Carga	Amplitud °	RMS Control (%PWM)	RMS MLP (%PWM)	RMS PID (%PWM)	% RMS MLP	% RMS PID	# Ciclos De Adaptación
0.5	2 k	5	2.72	2.67	0.51	96.51	3.49	35
		10	4.60	4.52	0.84	96.64	3.36	20
		15	5.70	5.62	0.96	97.19	2.81	10
	3 kg	5	3.26	3.23	0.44	98.20	1.80	60
		10	3.87	3.80	0.76	96.18	3.82	20
		15	5.61	5.56	0.70	98.46	1.54	10

Tabla 4.9: Aporte del PID y el MLP a la señal de control luego del cambio de dinámica [25].

- **Resistencia ante perturbaciones transitorias.**

Se ejecutó la trayectoria de 15° de amplitud a 0.5 Hz con un peso extra de 3kg. La red neuronal ya se encuentra entrenada con base a la amplitud y frecuencia ya descritas, de esta forma se procede a esperar a que la red se adapte a la carga adicional y se procede a perturbar manualmente la articulación. En la Figura 4-22 se puede apreciar el comportamiento de las diferentes señales luego de realizar algunas perturbaciones en diferentes puntos (marcados con flechas rojas).

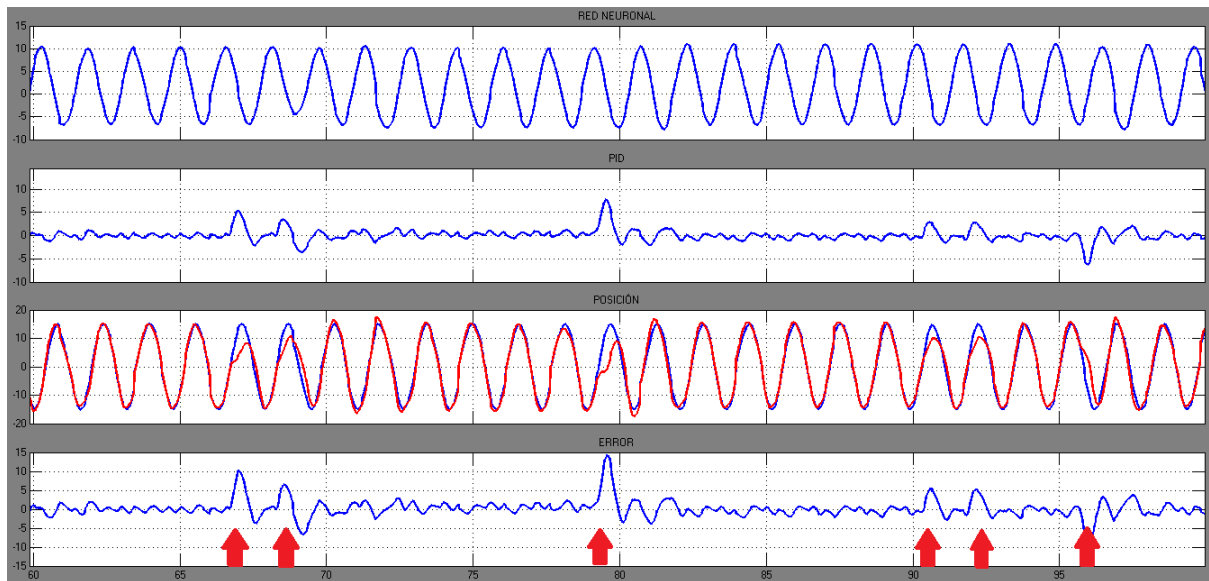


Figura 4-22: Comportamiento del sistema ante perturbaciones transitorias [25].

Los diferentes momentos en los que la articulación fue expuesta a perturbaciones se traducen en picos en la señal de error y de PID. Tomando por ejemplo la tercera flecha en la Figura 4-22 se puede apreciar que se requieren de cerca de 3s para corregir el error provocado por la perturbación. Cabe mencionar que en los puntos donde ocurren las perturbaciones no se nota ningún cambio significativo en la señal proveniente de la red neuronal (más allá de un pequeño cambio de amplitud a nivel de la segunda flecha). Esto es de esperar, ya que ante dichas perturbaciones el control por PID se hace necesario ya que la red neuronal una vez entrenada, no tiene manera de saber que ocurre en la salida según la topología con la que se está trabajando (la red neuronal funciona como un control feedforward).

Esto implica que la red neuronal mantiene el control de la trayectoria deseada mientras el PID aumenta su aporte a la señal de control de manera a corregir el error, de esta forma una vez corregido el error, el PID vuelve al reducir su contribución al mínimo. No obstante, se debe siempre monitorear la señal del PID de modo a activar el entrenamiento de la red neuronal en caso de que el aporte del PID aumente más allá del 5%, ya que al detectar un aumento en la señal del PID (por cambio de dinámica y/o perturbación no transitoria) la red neuronal se debe adaptar nuevamente y reducir eventualmente tanto el error como el aporte del PID.

En la Figura 4-23 se puede ver la variación del error RMS, una vez terminada la etapa de aprendizaje (rectángulo rojo), se procedió a perturbar manualmente la articulación en determinados momentos (círculos rojos). Como se puede observar pese a llegar a tener valores altos luego de las perturbaciones, el valor RMS del error se reduce rápidamente (2-3 ciclos). Debido a lo anterior es primordial que el control por retroalimentación sea capaz de

estabilizar la planta ante perturbaciones ya que de lo contrario el MLP no podrá hacer nada para contrarrestarlas (recordar que se trata de un control feedforward), el rendimiento del control por retroalimentación debe entonces ser lo suficientemente bueno para realizar dicha tarea, de manera que un mejor control por retroalimentación no solo incidirá en un entrenamiento más rápido y/o en un mejor modelo inverso sino también en una mejor respuesta ante perturbaciones.

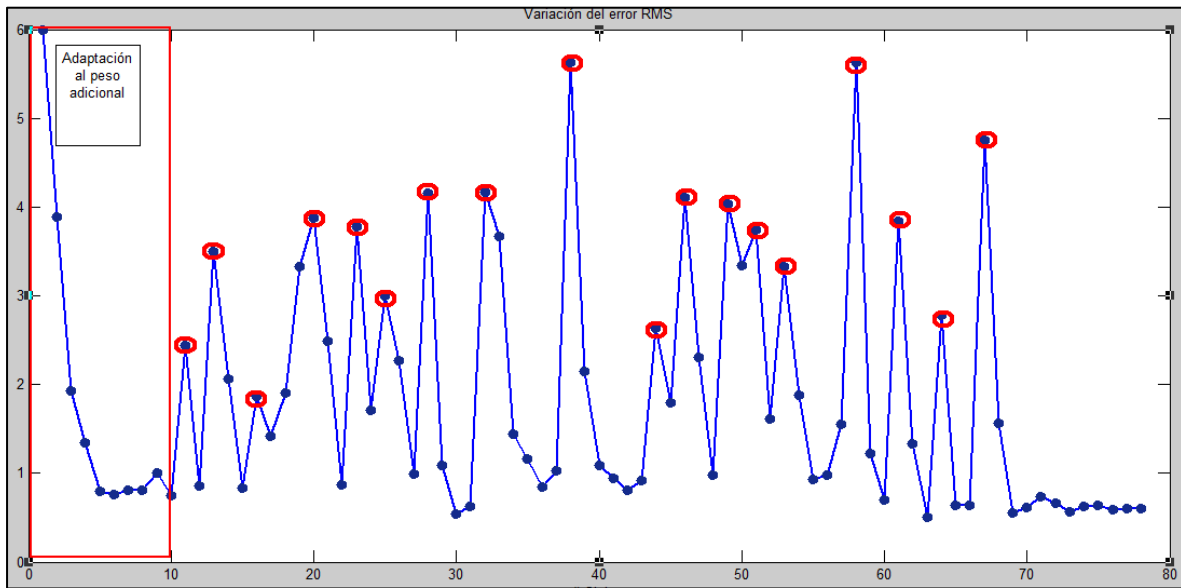


Figura 4-23: Valor del error RMS durante prueba con perturbaciones [25].

- **Resistencia ante ruido en señal de sensores.**

Se quiso comprobar la resistencia del control ante posibles interferencias en la lectura del valor del ángulo en la articulación. Se simuló dicha situación agregando un generador de ruido gaussiano y sumando su salida a la lectura del sensor (Figura 4-24).

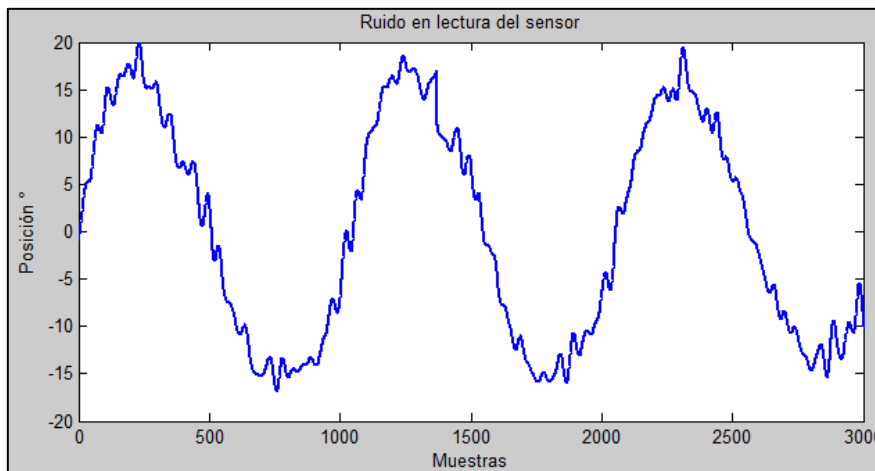


Figura 4-24: Ruido en lectura del sensor[25].

Observando la Figura 4-25, se puede apreciar que, a nivel de comportamiento, el control por PID y FEL no tiene mayor diferencia con respecto al que se vio en la sección 4.4, ya que como se puede observar el PID sigue teniendo un claro retraso y error mientras que el control por FEL permite reducirlos visiblemente (Ver Tabla 4.10). Lo realmente importante en este apartado es que, a pesar de la señal ruidosa, el MLP es capaz de obtener un modelo inverso capaz de mejorar el rendimiento.

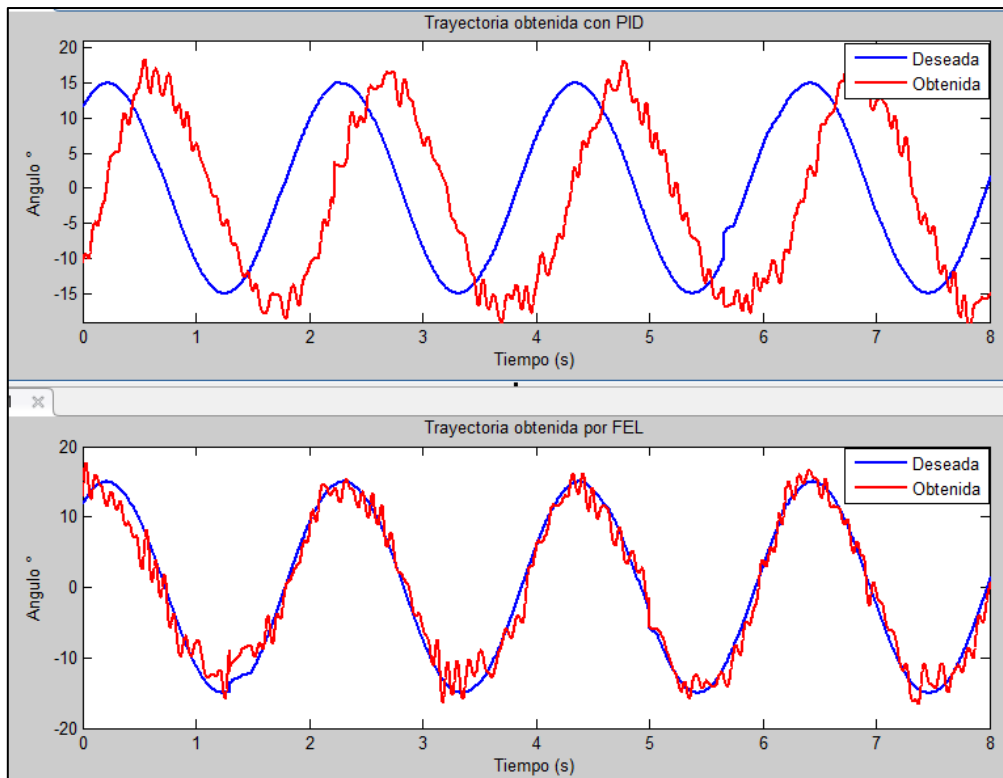


Figura 4-25: Comparativa de control de trayectoria entre PID y FEL.

Seguidamente se procedió a comparar cuantitativamente el rendimiento del control para una trayectoria senoidal de 0.5 Hz con una amplitud de 15 por medio del PID y por medio del FEL. Se pretende comprobar que las capacidades de generalización de la red neuronal sean capaces de contrarrestar la señal ruidosa. Dicha información se puede observar en la Tabla 4.10:

Frecuencia (Hz)	Amplitud (°)	Control	Error RMS (°)	Delay (s)	RMS Control (%PWM)	RMS MLP (%PWM)	RMS PID (%PWM)	% RMS MLP	% RMS PID
0.5	15	PID	13.73	0.42	6.21	0	6.21	0	100
		FEL	2.14	0.02	6.30	5.92	2.15	88.35	11.65

Tabla 4.10: Comparativa entre PID y FEL ante ruido en los sensores [25]

Se puede afirmar que el controlador FEL permite obtener un mucho mejor rendimiento ya que el valor RMS del error (85 % más pequeño con respecto al obtenido por el controlador PID clásico) y del desfase es mucho menor cuando se utiliza este método de control. Por otro lado, durante esta prueba el aporte del PID es mayor si lo comparamos con pruebas anteriores, lo que implica que este se encarga de estabilizar la planta debido al ruido. De hecho, si se observa la variación del valor RMS del error (Figura 4-26), se puede constatar que de manera general sigue una tendencia decreciente. Esto nos permite intuir que, si bien el PID mantiene la estabilidad, la red neuronal parece ser capaz de aprender y adaptarse a la señal ruidosa.

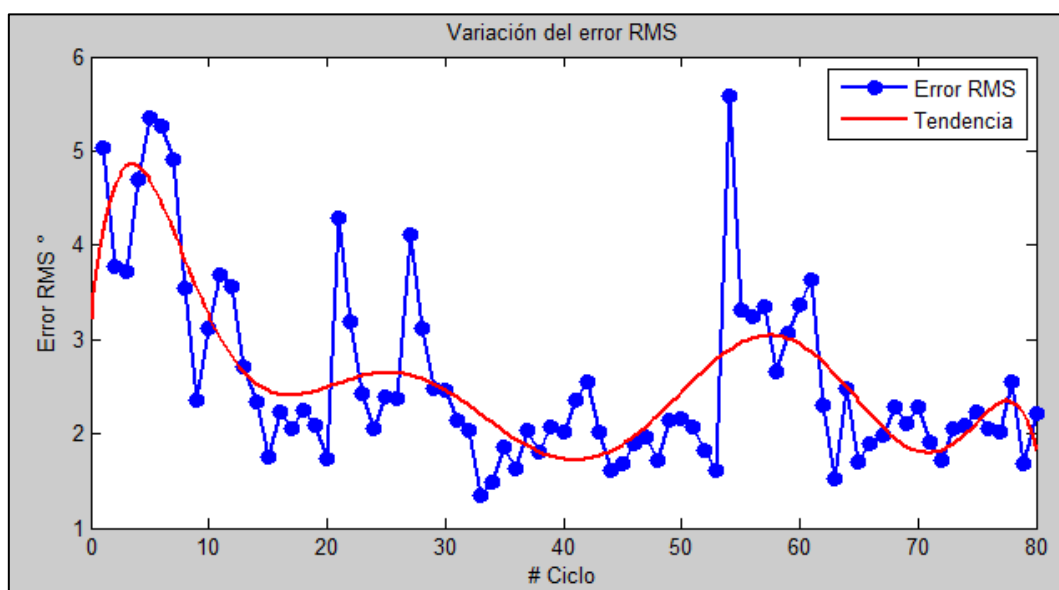


Figura 4-26: Variación del valor RMS del error debida a ruido en sensores [25].

#### 4.5.2 Prueba de estabilidad.

- **Variación de la amplitud**

Siempre conservando la misma frecuencia de operación y utilizando una red neuronal previamente entrenada con una amplitud de  $10^\circ$ , se procedió a variar la amplitud de la trayectoria de forma aleatoria cada 5 segundos de manera que pudiera tener cualquier valor en un rango de 5 a  $15^\circ$  (Figura 4-27).

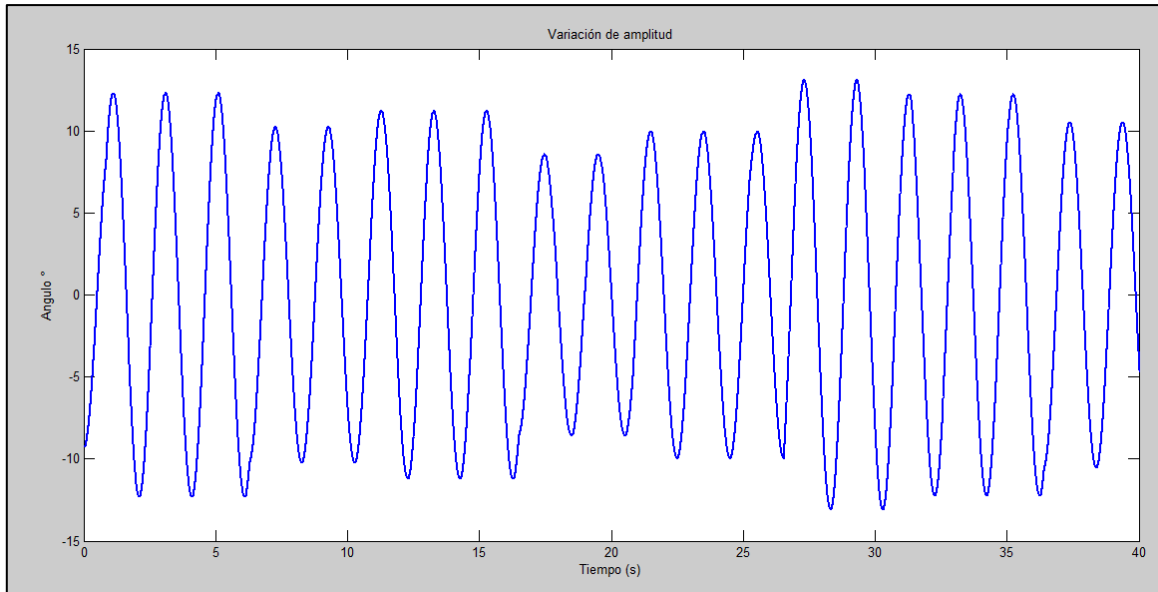


Figura 4-27: Trayectoria senoidal con variación de amplitud.

Se registró la información durante cada segundo de operación, como se puede observar en la Figura 4-28 pese a que se presentan picos momentáneos debido al cambio de amplitud, en promedio el valor RMS del error es de  $1.39^\circ$ , lo que implica que la red neuronal es capaz de adaptarse a estos manteniendo un buen rendimiento.

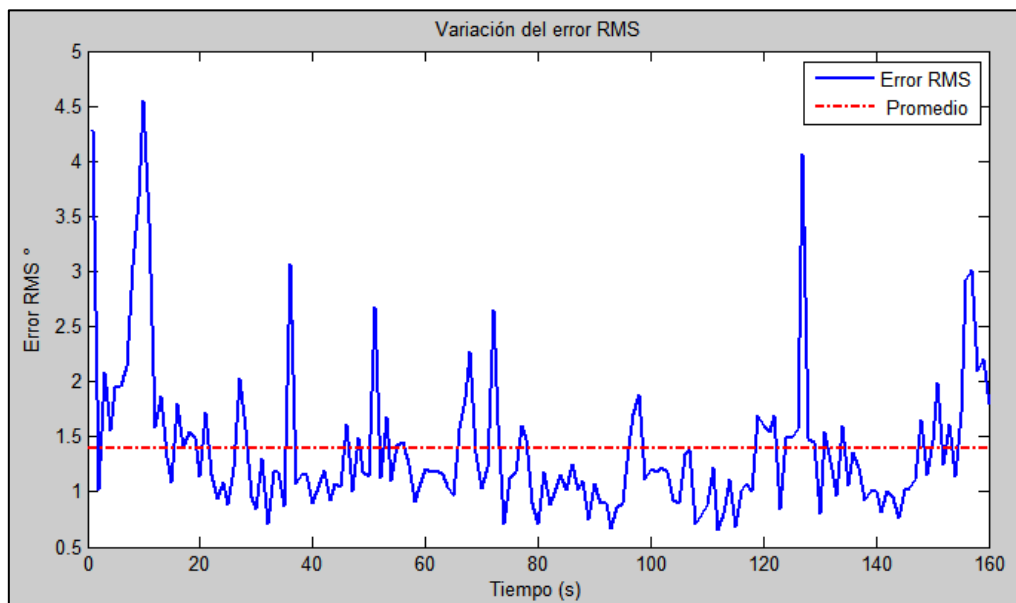


Figura 4-28: Variación del error RMS en prueba de estabilidad [25].

Por otro lado, en la Figura 4-29 podemos ver que la red neuronal mantiene el dominio sobre la señal del PID a lo largo del tiempo de ejecución ya que en promedio el valor RMS de la señal de control de la red neuronal es de  $4.99\% \text{PWM}$  mientras que el PID en promedio tiene un valor RMS de  $0.80\% \text{PWM}$ . El PID se hace necesario únicamente durante las etapas

iniciales de adaptación al haber un cambio de amplitud de manera a entrenar la red neuronal ante las nuevas condiciones.

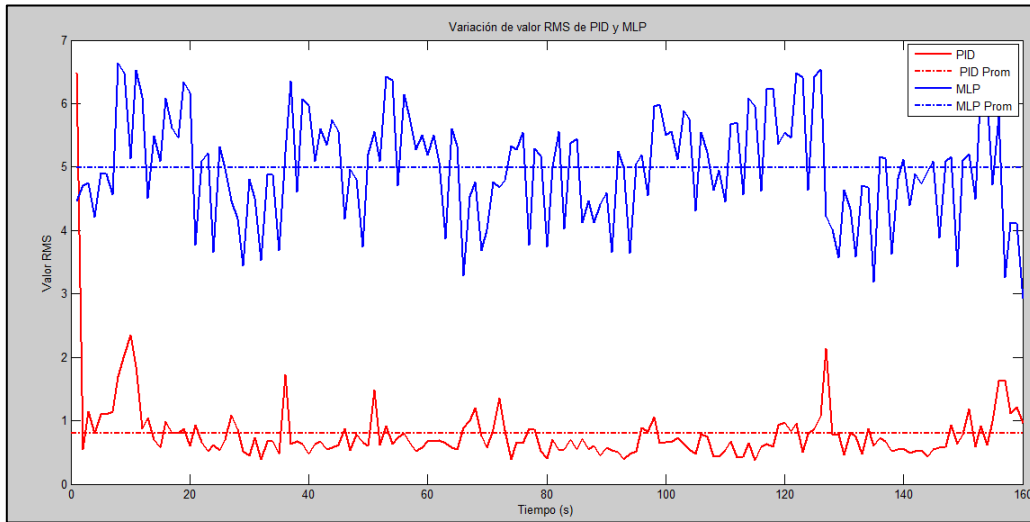


Figura 4-29: Variación del aporte del PID y MLP durante prueba de estabilidad [25].

### 4.5.3 Prueba extra: Trayectoria Sagital de la rodilla.

Como prueba final se quiso comprobar el rendimiento del controlador FEL para el control de la trayectoria real de la rodilla durante la marcha. Dicha trayectoria está disponible a partir del trabajo en biomecánica realizado por David A. Winter (PhD)[29] donde se puede encontrar las trayectorias sagitales para las articulaciones utilizadas durante la marcha (cadera, tobillo y rodilla) para diferentes cadencias de movimiento. En la Figura 4-30 se puede apreciar un ciclo de la trayectoria en cuestión, donde el rango de movimiento es de unos  $35^\circ$  a lo largo de un periodo de poco más de 6s.<sup>13</sup>

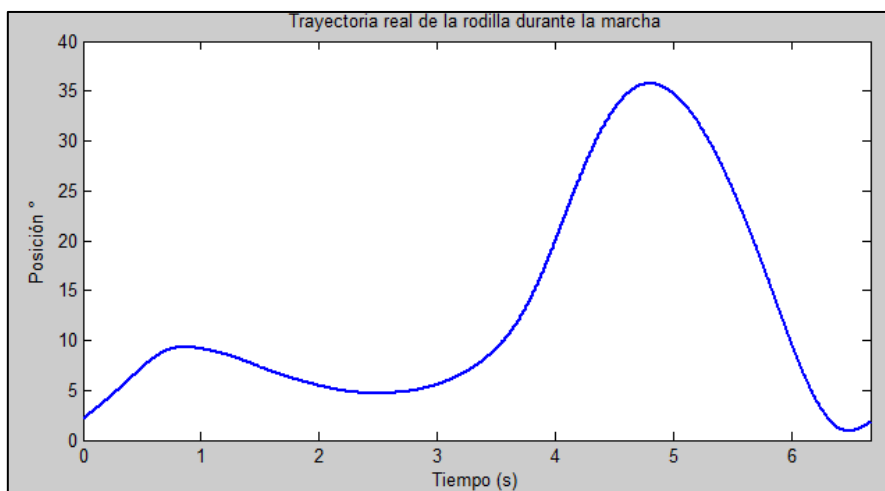


Figura 4-30: Trayectoria sagital de la rodilla durante la marcha [25].

<sup>13</sup> Cadencia de movimiento lenta y amplitud máxima escalada a  $35^\circ$  (a partir de  $70^\circ$ )

A continuación, se presenta la trayectoria obtenida mediante el control clásico por PID (Figura 4-31), como se puede observar existe un claro retraso entre las trayectorias, además de un significativo error (ver Tabla 4.11). Por lo tanto, es evidente que el esquema de control clásico es insuficiente para garantizar la correcta ejecución de la trayectoria deseada. Esto no se diferencia de lo observado en secciones anteriores bajo otras condiciones de operación.

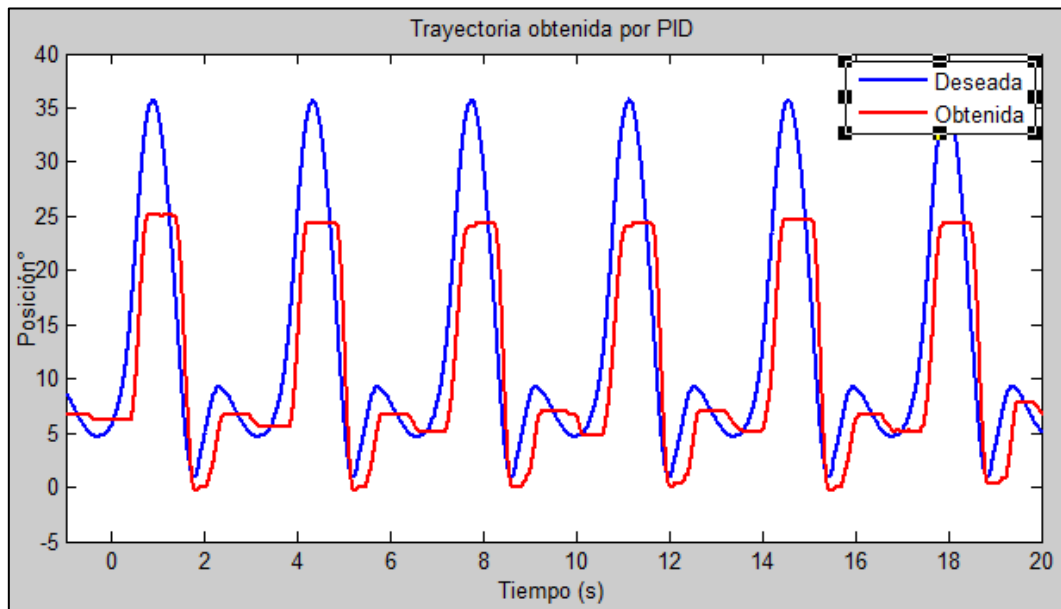


Figura 4-31: Trayectoria sagital de la rodilla por PID[25].

Por otro lado, en la Figura 4-32 se puede apreciar la trayectoria obtenida mediante el controlador FEL, se aprecia una significativa mejora ya que ambas trayectorias están ahora en fase y el error, aunque no se ha eliminado del todo, si se ha reducido (para más detalle ver Tabla 4.11).

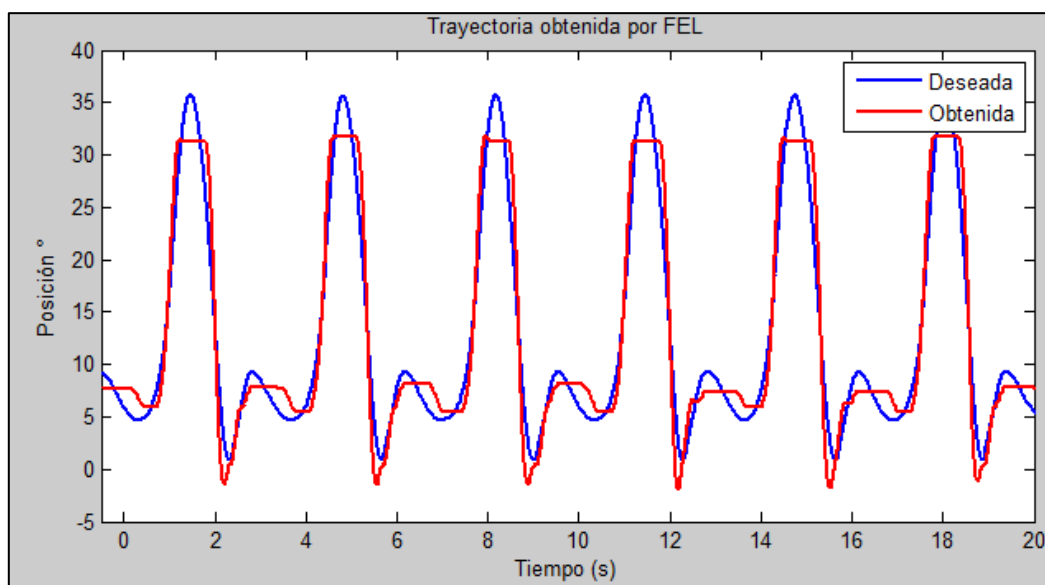


Figura 4-32: Trayectoria sagital de la rodilla por FEL[25].



Se comparó el rendimiento del PID clásico con el del controlador FEL. Dicha información se puede apreciar en la Tabla 4.11 y se puede confirmar que el uso del controlador FEL supone una mejora tanto en términos de error (reducción del error RMS en un 61.58%) como en términos de fase (eliminación del retardo existente en la mayor parte de la trayectoria). Por otro lado, se puede ver que pese a que el aporte del PID a la acción de control es mayor (19.25%) a lo que se tiene en secciones anteriores, sigue siendo menor al aporte de la red neuronal (80.75%). Se debe reconocer que pese a la mejora existente persisten algunos problemas, como lo es la incapacidad del control de lleva al actuador a posiciones máximas y el leve retraso que se da en la parte donde se tiene una pequeña oscilación. Todo ello implica que la dinámica aprendida no es completamente igual a la dinámica real (probablemente a raíz de que el PID por si solo presenta los mismos problemas), por lo que se debe explorar otros métodos de control por retroalimentación que consideren mejor la dinámica de la planta y permitan obtener un mejor rendimiento.

Control	Error RMS (°)	Delay (s)	RMS Control (%PWM)	RMS MLP (%PWM)	RMS PID (%PWM)	% RMS MLP	% RMS PID
<b>PID</b>	6.24	0.12	4.52	0	4.52	0	100
<b>FEL</b>	2.40	0	4.24	3.81	1.86	80.75	19.25

Tabla 4.11: Comparativa de rendimiento entre PID y FEL[25].

#### 4.5.4 Limitaciones del FEL.

No obstante, pese a que los resultados anteriores resultan alentadores, se debe reconocer que existen algunas limitantes en cuanto al funcionamiento del FEL:

- El valor del Learning Rate y del Momentum son sensibles a la frecuencia a la que se está entrenando la red neuronal. Por ello los valores de los parámetros que se observan en la Tabla 4.5 son solo válidos si se entrena a 0.5 Hz. En la Figura 4-33 se puede apreciar el efecto del valor de los parámetros utilizados (Tabla 4.5) a 0.7Hz. Como se puede ver el valor de los pesos no se estabiliza y permanecen fluctuando. Es posible evitar esto por medio del ajuste progresivo del Learning Rate y el momentum, sin embargo, el ajuste debe realizarse mediante prueba y error y son específicos de la frecuencia a la que se realiza el ajuste.

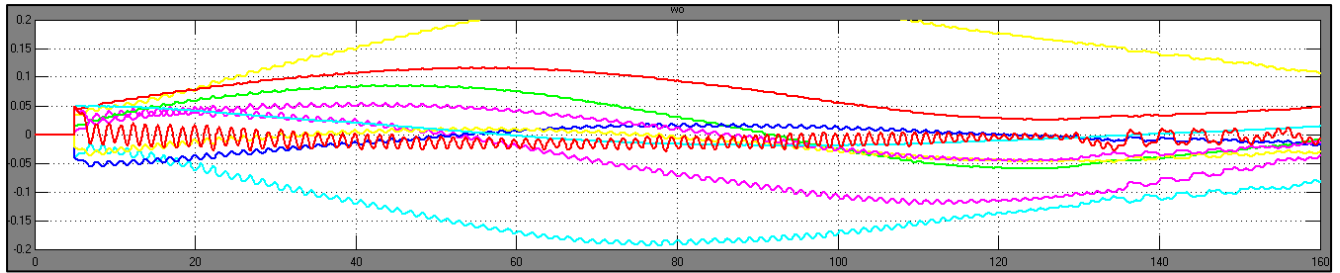


Figura 4-33: Oscilación en los valores de los pesos de la capa de salida[25]

- Si el entrenamiento se prolonga más allá del punto donde el valor RMS del error se estabiliza, el rendimiento de la red se deteriora debido a que la señal de control de la red neuronal se reduce progresivamente. Para evitar esto se implementó una función que monitorea cada segundo el valor “RMS” de la señal del PID. Cuando este representa un 5% o menos del valor RMS de la señal de control total ( $U_{ff}+U_{fb}$ ) entonces se detiene el entrenamiento (Ver Anexo A1.2).
- El tiempo de aprendizaje depende tanto de la frecuencia de la trayectoria (o duración del ciclo), como del rendimiento del control por retroalimentación utilizado. Esto debido a que la red neuronal no puede aprender en menos tiempo que el dictado por la duración de un ciclo de la trayectoria, además si el control por retroalimentación no es lo suficientemente bueno se requiere de más tiempo para obtener un modelo inverso adecuado. Adicionalmente el rendimiento del control por retroalimentación afecta el grado de aprendizaje de la dinámica del sistema por parte de la red neuronal y por ende su rendimiento final.
- El control por retroalimentación debe asegurar un funcionamiento estable en cualquiera de las condiciones de operación del actuador que se puedan presentar (diferentes frecuencias, amplitudes, perturbaciones estáticas y/o transitorias) de lo contrario la red neuronal no será capaz de aprender la dinámica inversa de la planta y/o de proveer una señal de control adecuada.

## 5. CAPITULO 5: Conclusiones y trabajo futuro.

### 5.1.1 Conclusiones.

1. Se desarrolló una estrategia de comunicación distribuida entre las unidades de procesamiento. De esta forma se contempla el uso de hasta cinco nodos de comunicación al mismo tiempo, todos ellos enviando y recibiendo información. Como se pudo observar en el capítulo 3, pese a una mayor demanda en el bus de

comunicación, el rendimiento de este se mantiene dentro de un margen aceptable (porcentaje de muestras erróneas menor al 6%) que garantiza un correcto funcionamiento y comunicación entre los dispositivos de control.

2. Se diseñó un algoritmo de arbitraje de datos. Esto permitió eliminar por completo los errores de comunicación que se presentaban inicialmente (picos aleatorios, distorsión, entre otros). Del mismo modo el algoritmo de transmisión de datos desarrollado a nivel del firmware de las unidades de procesamiento secundario contempla el uso de todos los recursos disponibles para la transmisión de datos, así como el orden en el que es enviada la información.
3. Se mejoró el rendimiento de los controladores de trayectoria. Mediante la adaptación del algoritmo del Feedback Error Learning para el control del actuador de rigidez variable. Como se pudo observar en el capítulo 4, el FEL supone una clara mejora en cuanto a rendimiento ya que reduce el error y el desfase existentes en el esquema de control original (PID clásico). Cuantitativamente la implementación del FEL supone una reducción del error de hasta el 90 % <sup>14</sup> en comparación al PID clásico.
4. Se realizaron pruebas de robustez y estabilidad para el controlador FEL. Dichas pruebas consisten en pruebas de resistencia a perturbaciones (estáticas y transitorias), ruido en la señal de los sensores y variación aleatoria de la amplitud de la trayectoria. Como se pudo observar en el capítulo 4 el FEL tiene en todos los casos un mejor rendimiento que su contraparte original por lo que supone una estrategia de control viable para el control de actuadores de rigidez variable.

### **5.1.2 Trabajo futuro.**

1. Investigar e implementar un algoritmo para adaptar los parámetros de la red neuronal a variaciones de frecuencia. Como se vio en el capítulo 4 pese a obtener un mejor rendimiento que el control por PID clásico, el FEL es sensible a variaciones de frecuencia. Esto debido a que los parámetros de aprendizaje de la red neuronal deben ajustarse con respecto a la frecuencia de operación. Este ajuste se realiza por medio de prueba y error por lo que resulta en un proceso lento y poco eficiente. Una alternativa

---

<sup>14</sup> Para el caso de una trayectoria senoidal de 0.5 Hz y 15 ° de amplitud.

puede ser el ajuste progresivo de dichos parámetros por métodos iterativos que tomen en consideración la frecuencia de operación y/o nivel de aprendizaje de la red neuronal.

2. Implementar el controlador FEL para el control de trayectoria de dos o más articulaciones operando al mismo tiempo, esto no se pudo probar debido a falta de tiempo. Considerando que la dinámica del actuador depende también de su posición relativa con respecto a las demás articulaciones, es importante validar el uso del FEL en dichas condiciones de operación.
3. Verificar que la red neuronal sea capaz de adaptarse a cambios de rigidez del actuador. Una de las principales características del actuador utilizado es el cambio de rigidez de manera mecánica, lo que conlleva a un cambio de la dinámica del actuador, por ello la red neuronal debe ser capaz de aprender la nueva dinámica y mantener un rendimiento aceptable.
4. Valorar otros tipos de control por retroalimentación para realizar el entrenamiento. Como se pudo observar a lo largo del proyecto el rendimiento del control por retroalimentación afecta la rapidez y la efectividad del entrenamiento. Se considera que al utilizar un mejor control por retroalimentación que considere mejor la dinámica del actuador, la red neuronal será capaz de aprender de manera más completa y rápida el modelo inverso del actuador.
5. Probar nuevas topologías de control y otros tipos de redes neuronales. Con tal de conservar el enfoque biomimético del FEL no se probaron otras topologías de control. Sin embargo, considerando que una vez entrenado la red neuronal no tiene forma de saber que ocurre a la salida<sup>15</sup>, resulta interesante utilizar información del estado actual de la planta (posición, velocidad y/o aceleración actual) para intentar mejorar el rendimiento de la red neuronal. Por otro lado, el uso de otros tipos de red neuronal (como, por ejemplo, radial basis function [11]) suponen a su vez alternativas a considerar de manera a aumentar la versatilidad del esquema de control.

---

<sup>15</sup> En el esquema implementado las entradas a la red neuronal son únicamente la trayectoria, velocidad y aceleración deseadas por lo que la red neuronal representa un control feedforward.

## 6. Referencias

- [1] D. Rodriguez, "DISEÑO Y CONSTRUCCIÓN DEL ROBOT BÍPEDO BINOCCHIO," Universidad Politecnica de Madrid, 2014.
- [2] D. Torricelli, J. Gonzalez, M. Weckx, R. Jiménez-Fabián, B. Vanderborght, M. Sartori, S. Dosen, D. Farina, D. Lefeber, and J. L. Pons, "Human-like compliant locomotion: state of the art of robotic implementations," *Bioinspir. Biomim.*, vol. 11, no. 5, p. 51002, 2016.
- [3] H. Witte, H. Hofmann, R. Hackert, C. Schilling, M. S. Fischer, and H. Preuschoft, "Biomimetic robotics should be based on functional morphology," *Journal of Anatomy*, vol. 204, no. 5. pp. 331–342, 2004.
- [4] B. Vanderborght, A. Albu-Schaeffer, A. Bicchi, E. Burdet, D. G. Caldwell, R. Carloni, M. Catalano, O. Eiberger, W. Friedl, G. Ganesh, M. Garabini, M. Grebenstein, G. Grioli, S. Haddadin, H. Hoppner, A. Jafari, M. Laffranchi, D. Lefeber, F. Petit, S. Stramigioli, N. Tsagarakis, M. Van Damme, R. Van Ham, L. C. Visser, and S. Wolf, "Variable impedance actuators: A review," *Robotics and Autonomous Systems*, vol. 61, no. 12, Elsevier B.V., pp. 1601–1614, 2013.
- [5] "H2R: Integrative Approach For The Emergence Of Human-Like Locomotion," *Spanish Council for Scientific Research (CSIC)*, 2016. [Online]. Available: <http://www.h2rproject.eu/>.
- [6] R. Van Ham, M. Van Damme, B. Vanderborght, B. Verrelst, and D. Lefeber, "MACCEPA, the Mechanically Adjustable Compliance and Controllable Equilibrium Position Actuator," *Proc. 10th Int. Conf. New Actuators*, vol. 43, no. 4, pp. 467–474, 2006.
- [7] M. Kawato, "Feedback-error-learning neural network for supervised motor learning," *Adv. neural Comput.*, vol. 6, no. 3, pp. 365–372, 1990.
- [8] C. M<sup>a</sup> Rodríguez Parrilla Tutor and D. Rafael del Pino Casado Dpto, "MODELO Y ANÁLISIS DE UN ACTUADOR DE RIGIDEZ VARIABLE Y SU IMPLEMENTACIÓN EN LA PELVIS DE UN ROBOT BÍPEDO," *Trab. fin grado.*, 2014.
- [9] M. Weckx, R. Van Ham, H. Cuypers, R. Jiménez-Fabián, D. Torricelli, J. L. Pons, B. Vanderborght, and D. Lefeber, "Prototype design of a novel modular two-degree-of-freedom variable stiffness actuator," *IEEE-RAS Int. Conf. Humanoid Robot.*, vol. 2015–Febru, pp. 33–38, 2015.
- [10] L. F. Bertona, "Entrenamiento de redes neuronales basado en algoritmos evolutivos," *Grado, Tesis Ing. Informática, Univ. BUENOS AIRES*, 2005.
- [11] S. Marsland, *Machine Learning: An Algorithmic Introduction*, Second. Boca Raton, FL: CRC Press, 2009.
- [12] E. R. Kandel, J. H. Schwartz, and T. M. Jessell, *Principles of Neural Science*, vol. 3. 2000.
- [13] D. M. Wolpert, R. C. Miall, and M. Kawato, "Internal models in the cerebellum," *Trends Cogn. Sci.*, vol. 2, no. 9, pp. 338–347, 1998.
- [14] S. Sharma, V. Kumar, and R. Kumar, "Supervised Online Adaptive Control of Inverted Pendulum System Using ADALINE Artificial Neural Network with Varying System Parameters and External Disturbance," *Int. J. Intell. Syst. Appl.*, vol. 4, no. 8, pp. 53–61, 2012.
- [15] F. Error, L. Technique, and D. T. Tracking, "Modification of Feedback Error Learning Technique for Desired Trajectory Tracking in the Presence of Disturbance," no. February,

- 2014.
- [16] N. Gopalan, “Feedback Error Learning for Gait Acquisition,” no. November, 2012.
  - [17] F. Resquín, J. Gonzalez-Vargas, J. Ibáñez, F. Brunetti, and J. L. Pons, “Feedback error learning controller for functional electrical stimulation assistance in a hybrid robotic system for reaching rehabilitation,” *20th Conf. Int. Funct. Electr. Stimul. Soc.*, 2016.
  - [18] Y. Koike, J. Gonzalez, J. Gomez, and W. Yu, “Implementing Feedback Error Learning for FES control,” *Proc. - 2011 4th Int. Conf. Biomed. Eng. Informatics, BMEI 2011*, vol. 3, pp. 1324–1328, 2011.
  - [19] M. M. Gupta, “Feedback-Error Learning Scheme Using Recurrent Neural Networks for Nonlinear Dynamic Systems,” pp. 175–180, 1994.
  - [20] Z. Hamavand and H. M. Schwartz, “Trajectory Control of Robotic Manipulators by Using a Feedback-Error-Learning Neural Network,” *Robotica*, vol. 13, no. 5, pp. 449–459, 1995.
  - [21] S. Corrigan, “Introduction to the Controller Area Network ( CAN ),” no. August 2002. pp. 1–15, 2008.
  - [22] K. C. L. K. C. Lee, H.-H. L. H.-H. Lee, K. C. Kyung Chang Lee, and H.-H. Hong-Hee Lee, “Network-based fire-detection system via controller area network for smart home automation,” *IEEE Trans. Consum. Electron.*, vol. 50, no. 4, pp. 1093–1100, 2004.
  - [23] M. Di Natale, H. Zeng, P. Giusto, and A. Ghosal, “Understanding and Using the Controller Area Network Communication Protocol: Theory and Practice,” *inst. eeecs. berkeley. edu/~ ee249/fa08/Lectures/ ...*, 2012.
  - [24] Microchip Technology Inc., “dsPIC30F4011/4012 Data Sheet.” pp. 1–2, 2005.
  - [25] M. Rios, “Elaboracion Propia,” 2016.
  - [26] ams, “AS5048A / AS5048B Magnetic Rotary Encoder (14-Bit Angular Position Sensor).” pp. 1–41, 2016.
  - [27] D. Torricelli, J. Gonzalez, S. Schuetz, Q. Liu, J. Zhao, T. Mergner, V. Lippi, M. Sartori, S. Dosen, and R. Jimenez, “Assessment of H2R locomotion abilities ( Validation of the integration tasks ).” 2016.
  - [28] Microchip Technology Inc., “dsPIC30F Family Reference Manual,” vol. 426, no. 2. pp. 1–34, 2006.
  - [29] D. a D. a. D. a. Winter, *BIOMECHANICS AND MOTOR CONTROL OF Fourth Edition*, vol. 2nd. 2009.
  - [30] Escon Maxon, “EC 4Pole 200W 48V,” no. April. p. 2014, 2014.
  - [31] H. Reference, “ESCON Module 50/5,” no. November, 2015.
  - [32] L. Eciolaza, T. Taniguchi, M. Sugeno, D. Filev, and Y. Wang, “Piecewise bilinear models for feedback error learning: On-line feedforward controller design,” *IEEE Int. Conf. Fuzzy Syst.*, 2013.

## A. Anexos

### A1. Código en C y MATLAB.

#### A1.1 Código función `can_send ()` modificada.

```
void can_send(uint16_t id, int16_t* mens,int* priority)
{
    //Buffer 0
    if(C1TX0CONbits.TXREQ == 0)
    {
        C1TX0B1 = mens[0];
        C1TX0B2 = mens[1];
        C1TX0B3 = mens[2];

        id &= 0x07FF; // identifier 11 bits
        id <<= 5;
        int16_t tx0sid = id;
        tx0sid &= 0xF800;
        id <<= 5;
        id >>= 8;
        tx0sid |= id;

        C1TX0SID = tx0sid;

        C1TX0CONbits.TXPRI = *priority;
        C1TX0CONbits.TXREQ = 1;
        *priority -- 1;
    }
}
```

Figura A-1 Código función `can_send ()` Buffer 0 [25].

```
//Buffer 1
else if (C1TX1CONbits.TXREQ == 0)
{
    C1TX1B1 = mens[0];
    C1TX1B2 = mens[1];
    C1TX1B3 = mens[2];

    id &= 0x07FF; // identifier 11 bits
    id <<= 5;
    int16_t tx1sid = id;
    tx1sid &= 0xF800;
    id <<= 5;
    id >>= 8;
    tx1sid |= id;

    C1TX1SID = tx1sid;
    C1TX1CONbits.TXPRI = *priority;
    C1TX1CONbits.TXREQ = 1;
    *priority -- 1;
}
```

Figura A-2: Código función `can_send ()` Buffer 1 [25].

```

//Buffer 2
else if (C1TX2CONbits.TXREQ == 0)
{
C1TX2B1 = mens[0];
C1TX2B2 = mens[1];
C1TX2B3 = mens[2];

id &= 0x07FF; // identifier 11 bits
id <<= 5;
int16_t tx2sid = id;
tx2sid &= 0xF800;
id <<= 5;
id >>= 8;
tx2sid |= id;

C1TX2SID = tx2sid;
C1TX2CONbits.TXPRI = *priority;
C1TX2CONbits.TXREQ = 1;
*priority -- 1;
}

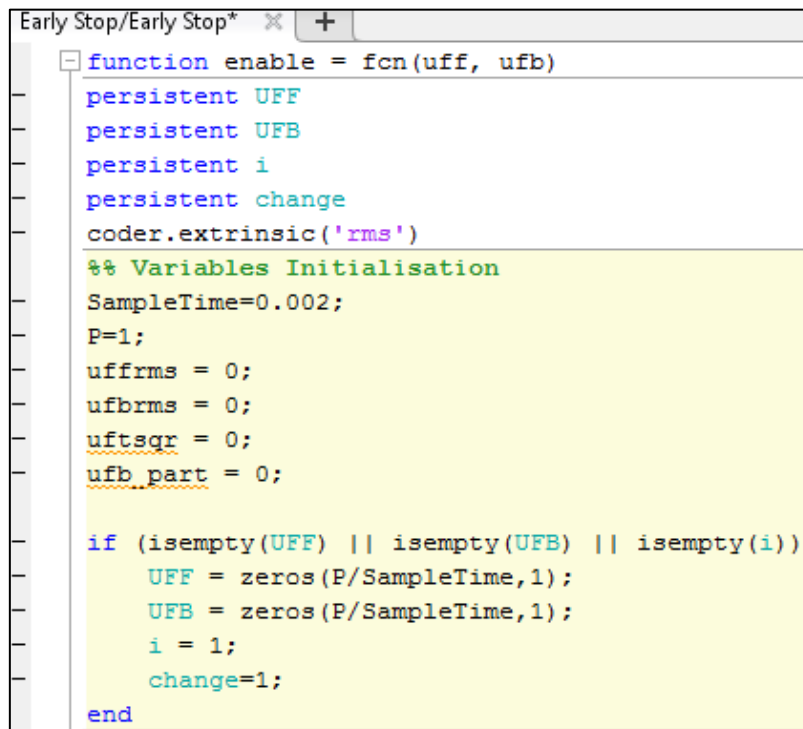
if(*priority < 0){
*priority = 3 ;
}

```

Figura A-3: Código función can\_send () Buffer 2[25].

### A1.2 Código Función Early Stop.

Dicha función detiene el entrenamiento cuando la señal UFB (proveniente del PID) representa un 5% o menos de la señal total de control. Se revisa el % de aporte de dicha señal cada ciclo de la trayectoria.



```

Early Stop/Early Stop* x +
function enable = fcn(uff, ufb)
persistent UFF
persistent UFB
persistent i
persistent change
coder.extrinsic('rms')
%% Variables Initialisation
SampleTime=0.002;
P=1;
uffrms = 0;
ufbrms = 0;
uftsqr = 0;
ufb part = 0;

if (isempty(UFF) || isempty(UFB) || isempty(i))
UFF = zeros(P/SampleTime,1);
UFB = zeros(P/SampleTime,1);
i = 1;
change=1;
end

```

Figura A-4: Inicialización de Variables [25]



```

if i>=(P/SampleTime)
    % UFF RMS Value
    uffrms = rms(UFF(1:(P/SampleTime),1));
    % UFB RMS Value
    ufbrms = rms(UFB(1:(P/SampleTime),1));
    % (UFF + UFB) RMS Value = Total control RMS Value
    uftsqr = uffrms*uffrms+ufbrms*ufbrms;
    % Percentage of UFB RMS Value
    ufb_part=100*ufbrms*ufbrms/uftsqr;
    % Learn Until UFB represents 5% of control signal
    if ufb_part <= 5
        change=0;
    else
        change=1;
    end
    UFF = zeros(P/SampleTime,1);
    UFB = zeros(P/SampleTime,1);
    i = 1;
else
    UFF(i)=uff;
    UFB(i)= ufb;
    i=i+1;
end
enable=change;
end

```

Figura A-5: Calculo de % de aporte de señal Ufb [25].

### A1.3 Código Función para generar trayectoria senoidal variable.

```

Control Loop/ Trajectory/Manually Variable Sine Signal * | +
function y = fcn(A,F,T,D)
persistent Fprev
persistent Aprev
persistent phi
%% Variables Initialization
if (isempty(Fprev) || isempty(Aprev))
    Fprev=F;
    Aprev=A;
    phi=0;
end
%% If the frequency and the amplitude stay the same keep using previous values
if (F==Fprev && A==Aprev)
    y=A*sin(2*pi*F*T+phi);
elseif (D>=0) % If there is a change in frequency or amplitude, change the signal during the positive semi cycle, c
    ynext=Aprev*sin(2*pi*Fprev*T+phi); % Calculate the next value with previous frequency and amplitude
    if abs(ynext) <= A % Verify that the next value is not greater than the new amplitude, otherwise keep using
        newphi=asin(ynext/A)-2*pi*F*T; % Calculate new phase to avoid disruption
        if newphi <=-pi % Keep the phase in between +- pi
            newphi=newphi+2*pi;
        end
        if newphi >= pi
            newphi=newphi-2*pi;
        end
        phi=newphi; %update the phase value
        y=A*sin(2*pi*F*T+phi); % Calculate next value using new frequency and amplitude
        Fprev=F; % Update previous values
        Aprev=A;
    else
        y=Aprev*sin(2*pi*Fprev*T+phi);
    end
else
    y=Aprev*sin(2*pi*Fprev*T+phi);
end
end
end

```

Figura A-6: Función para generar trayectorias senoidales modificables [25] .

## A2. Modelos de Simulink.

### A2.1 Bloque: Red Neuronal.

En la Figura A-7 se muestra el bloque de Simulink correspondiente a la red neuronal utilizada, a continuación, se describe su estructura:

- Enable: habilita el funcionamiento de la red.
- Enable Learning: habilita el entrenamiento de la red.
- Reset Weights: Resetea el valor de los pesos sinápticos en la capa oculta y de salida.
- Ufb: Señal de aprendizaje.
- Reference: vectores de entrada.
- Len\_Samples: Cantidad de muestras por ciclo.
- Uff: Salida de la red.
- Wo: pesos en la capa de salida.
- Wh: pesos en la capa oculta.

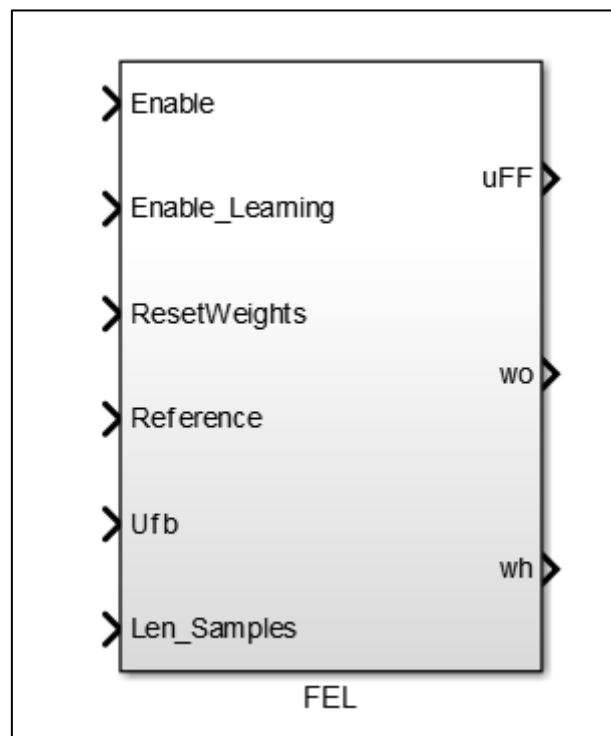


Figura A-7:Red Neuronal MLP [25].

## A2.2 Modelo Simulink: Control desde ordenador.

Este modelo permite realizar el control del modelo utilizado en la PC-104 desde un ordenador portátil o de escritorio, la comunicación se realiza por UDP y permite realizar las siguientes acciones:

- Calibración fina del punto correspondiente a  $0^\circ$ .
- Activación de la trayectoria senoidal.
- Encendido y apagado del control (PID+FEL).
- Escogencia de frecuencia y amplitud de trayectoria senoidal deseada.
- Calibración de constante de PID.
- Activación del FEL.
- Activación del aprendizaje de la red neuronal.
- Reset de valores de los pesos sinápticos.
- Recepción, visualización y almacenamiento de señales y pesos sinápticos.

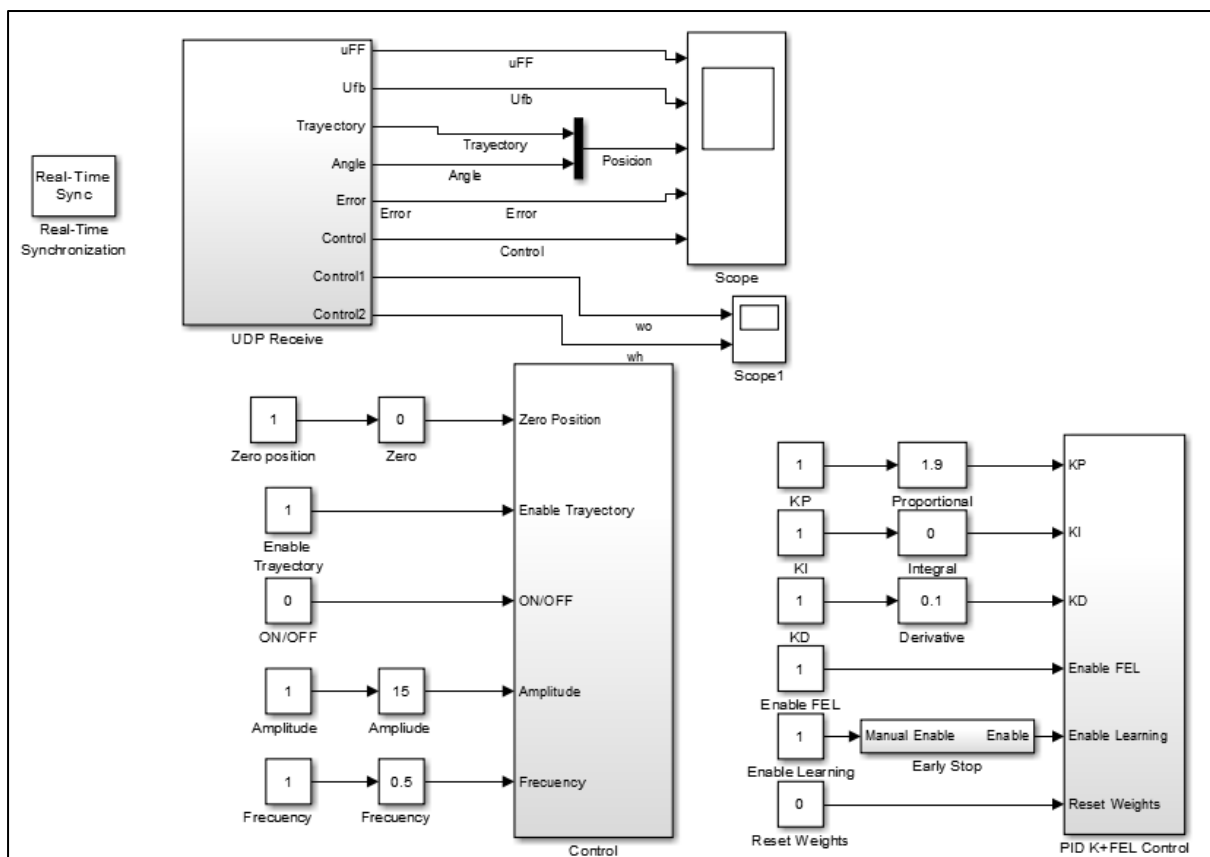


Figura A-8: Envío y recepción de datos para control del actuador MACCEPA [25].

### A2.3 Modelo Simulink: Control en PC-104.

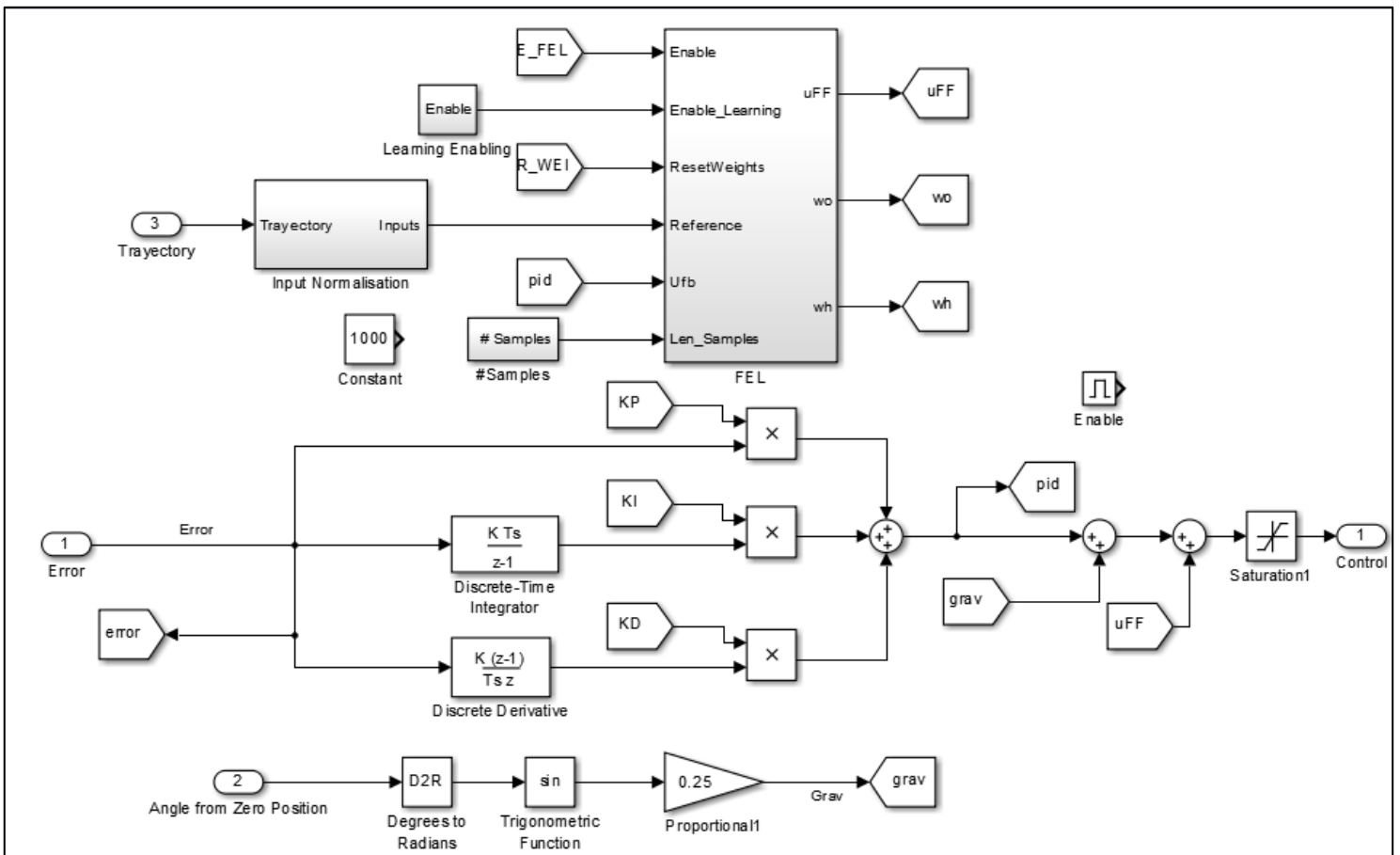


Figura A-9: Control PID, Red neuronal y compensación de gravedad [25].

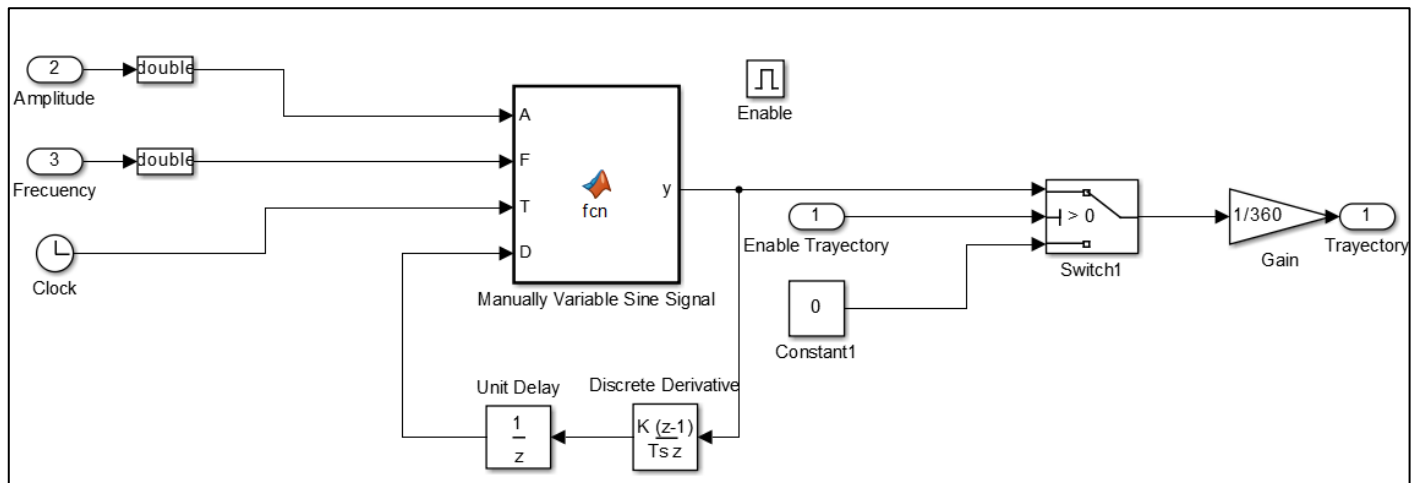


Figura A-10: Generación de trayectoria senoidal [25].

### A3. Hojas de Datos.

#### A3.1 Motor del actuador del Binnocchio.

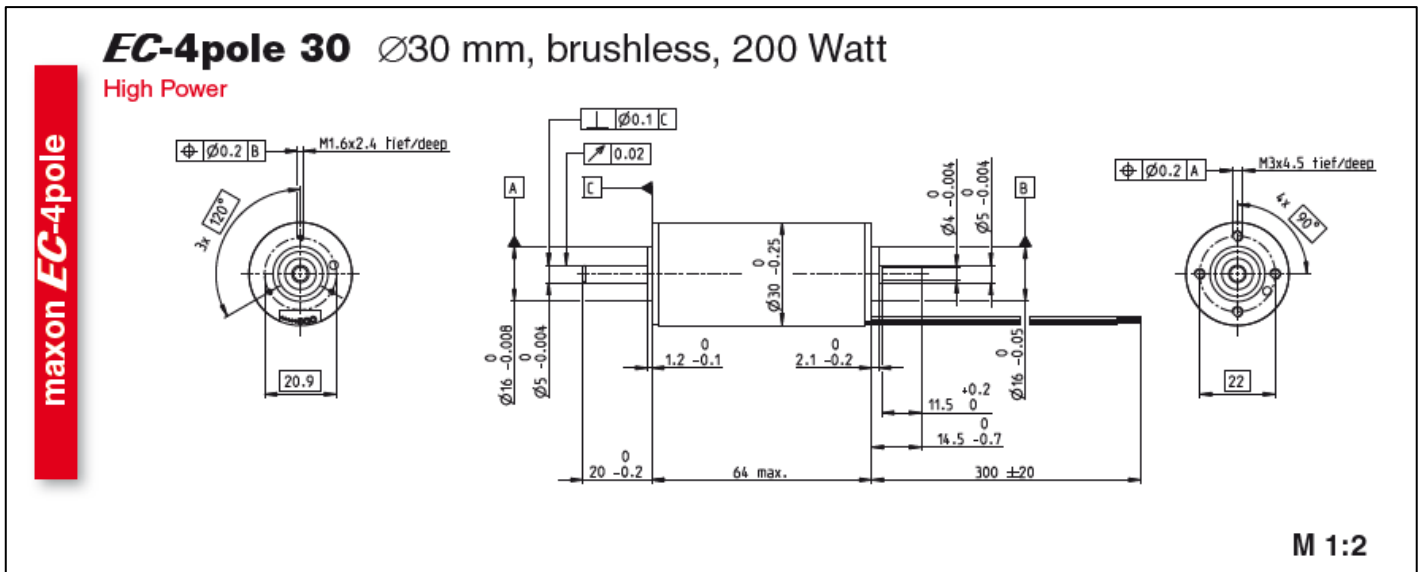


Figura A-11: Dimensiones del motor[30]

		Part Numbers		
		305013	305014	305015
<b>Motor Data</b>				
<b>Values at nominal voltage</b>				
1 Nominal voltage	V	24	36	48
2 No load speed	rpm	16700	16700	16500
3 No load current	mA	728	485	356
4 Nominal speed	rpm	15900	15900	15700
5 Nominal torque (max. continuous torque)	mNm	135	134	132
6 Nominal current (max. continuous current)	A	10.5	6.96	5.06
7 Stall torque	mNm	3220	3510	3430
8 Starting current	A	236	171	124
9 Max. efficiency	%	89	90	90
<b>Characteristics</b>				
10 Terminal resistance phase to phase	Ω	0.102	0.21	0.386
11 Terminal inductance phase to phase	mH	0.0163	0.0368	0.0653
12 Torque constant	mNm/A	13.6	20.5	27.6
13 Speed constant	rpm/V	700	466	346
14 Speed/torque gradient	rpm/mNm	5.21	4.78	4.83
15 Mechanical time constant	ms	1.82	1.67	1.69
16 Rotor inertia	gcm <sup>2</sup>	33.3	33.3	33.3

Figura A-12: Constantes eléctricas y mecánicas[30].



## A4.2 Driver ESCON 50/5

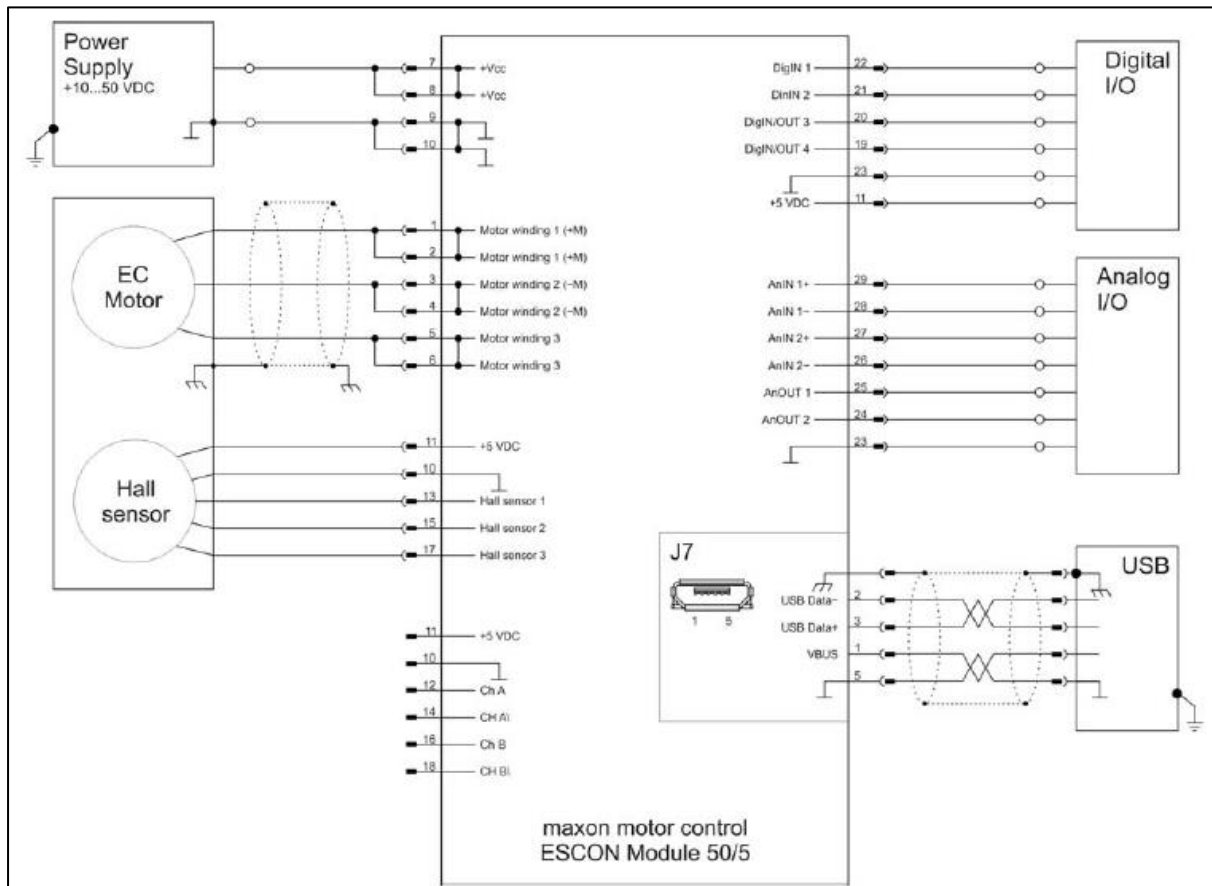


Figura A-15:Pines y conexiones [31].

### A5. Derivación de algoritmo de backpropagation para un MLP de 3 capas.

La presente derivación considera un MLP de 3 capas donde la función de activación en la capa de salida y en la oculta es la función identidad y tangente hiperbólica respectivamente (Ver Figura 2-9).

Sea  $U_o$  una señal ideal de control feedforward y se desea que la salida de la red neuronal  $U_{ff}$  sea tal que:

$$U_{ff} = U_o \quad (\text{A-1})$$

De esta forma la función de error que describe el rendimiento de la red neuronal con respecto a dicha señal ideal es:

$$E = \frac{(U_{ff} - U_o)^2}{2} \quad (\text{A-2})$$

$U_{ff}$  es función de las entradas “ $x$ ” de la red, los pesos sinápticos “ $w$ ” en la neurona de salida:

$$U_{ff} = p^{-1}(x, w) \quad (\text{A-3})$$

Donde “ $p^{-1}$ ” representa el modelo inverso de la planta que se desea controlar.

El método del gradiente descendiente ajusta el valor de los pesos en la dirección del gradiente de  $E(w)$  es decir que se debe derivar parcialmente (A-2) con respecto al valor de los pesos tal que:

$$\frac{\partial E}{\partial w} = \frac{\partial U_{ff}}{\partial w} (U_{ff} - U_o) \quad (\text{A-4})$$

Sin embargo, no es posible calcular el error ya que  $U_o$  se desconoce, no obstante, es posible utilizar  $U_{fb}$  como señal de error[32] ya que:

$$\begin{aligned} U_o &= U_{ff} + U_{fb} \\ -U_{fb} &= U_{ff} - U_o \end{aligned} \quad (\text{A-5})$$

Se tiene entonces reemplazando (A-5) en (A-4):

$$\frac{\partial E}{\partial w} = -\frac{\partial U_{ff}}{\partial w} * U_{fb} \quad (\text{A-6})$$

Ahora bien, es posible expresar (A-4) como:

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial h} * \frac{\partial h}{\partial w} \quad (\text{A-7})$$

Donde “ $h$ ” representa el potencial en la neurona de salida debido a la sumatoria de los valores de activación “ $a_o$ ” en cada neurona oculta por el peso sináptico “ $w$ ” asociado tal que:



$$h = \sum w * a_o \quad (A-8)$$

De esta forma reemplazando (A-8) en el segundo término de la derecha de (A-7):

$$\frac{\partial h}{\partial w} = \frac{\partial \sum w * a_o}{\partial w} = \sum \frac{\partial w * a_o}{\partial w} = a_o \quad (A-9)$$

Comparando (A-6) con (A-7) y utilizando el resultado obtenido en (A-9):

$$\frac{\partial E}{\partial w} = -\frac{\partial Uff}{\partial w} * Ufb = \frac{\partial E}{\partial h} * \frac{\partial h}{\partial w} = \frac{\partial E}{\partial h} * a_o$$

Ya que:

$$\frac{\partial Uff}{\partial w} = \frac{\partial h}{\partial w} = a_o$$

Y

$$Ufb = \frac{\partial E}{\partial h}$$

Se tiene:

$$\frac{\partial E}{\partial w} = -a_o * Ufb$$

$$-\frac{\partial E}{\partial w} = a_o * Ufb$$

Haciendo:

$$\delta_o = Ufb$$

Finalmente:

$$-\frac{\partial E}{\partial w} = a_o * \delta_o \quad (A-10)$$

Lo que nos indica que el error<sup>16</sup> en los pesos de la salida es equivalente al producto del valor de activación de cada neurona oculta por el valor de Ufb. Dicho resultado se utiliza en la ecuación (2-5) para la actualización del valor de los pesos sinápticos entre la capa de salida y la oculta.

Para el caso de los pesos sinápticos entre la capa de entrada y la oculta se sigue el mismo planteamiento, derivando el error con respecto a los valores “v” y los potenciales de activación “hh” de estos:

$$\frac{\partial E}{\partial v} = \frac{\partial E}{\partial hh} * \frac{\partial hh}{\partial v}$$

$$\frac{\partial E}{\partial v} = \frac{\partial E}{\partial h} * \frac{\partial h}{\partial hh} * \frac{\partial hh}{\partial v}$$

---

<sup>16</sup> El signo “-” indica que se va en la dirección donde el gradiente del error disminuye.

De (A-10) se tiene:

$$\frac{\partial E}{\partial v} = \delta_o * \frac{\partial h}{\partial hh} * \frac{\partial hh}{\partial v} \quad (\text{A-11})$$

Ahora analizando el segundo termino de (A-11) y utilizando (A-8):

$$\frac{\partial h}{\partial hh} = \frac{\partial \sum w * a_o}{\partial hh} = \sum \frac{\partial w * g(hh)}{\partial hh} = \sum w * g'(hh) \quad (\text{A-12})$$

Donde “g” representa la función de activación en las neuronas ocultas y su derivada es tal que:

$$a_o = g(hh) = \tanh(hh)$$

$$g'(hh) = (1 - \tanh^2(hh))$$

$$g'(hh) = (1 - a_o^2)$$

Reemplazando en (A-12):

$$\frac{\partial h}{\partial hh} = \sum w * (1 - a_o^2) \quad (\text{A-13})$$

Concentrándonos en el tercer término de (A-11):

$$\frac{\partial hh}{\partial v} = \frac{\partial \sum v * x}{\partial v} = \sum \frac{\partial v * x}{\partial v} = x \quad (\text{A-14})$$

Reemplazando (A-13) y (A-14) en (A-11):

$$\frac{\partial E}{\partial v} = (1 - a_o^2) \sum \delta_o * w * x$$

$$\frac{\partial E}{\partial v} = \delta_h * x$$

Lo que nos indica que el error<sup>17</sup> en los pesos entre la capa de entrada y la oculta es equivalente. Dicho resultado se utiliza en la ecuación (2-5) para la actualización del valor de los pesos sinápticos entre la capa de entrada y la oculta.

---

<sup>17</sup> El signo “-” indica que se va en la dirección donde el gradiente del error disminuye.

## A6. Carta de aceptación

 <p>MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD</p>	 <p>CSIC CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS</p> <p>INSTITUTO CAJAL</p>
<p>Madrid, 1 de Junio del 2016</p>	
<p>A quien corresponda:</p>	
<p>Con la presente informo que Mario Ríos Mora, cédula 115130872, estudiante de ingeniería Mecatrónica del Instituto Tecnológico de Costa Rica (Cartago, Costa Rica), ha sido aceptado para llevar a cabo su proyecto de graduación en el grupo de Neurorehabilitación del Instituto Cajal del Consejo Superior de Investigaciones Científicas (Madrid, España), DNI: Q2818002D. El estudiante será supervisado por el Prof. José Luis Pons Rovira y el Dr. José E. González Vargas, desde del 27 de Julio hasta el 12 de Diciembre del 2016.</p>	
 <p>Prof. José Luis Pons Rovira Investigador Principal del Grupo de Neuro-Rehabilitación Instituto Cajal Consejo Superior de Investigaciones Científicas (CSIC) jose.pons@csic.es</p>	

## **A7. Hoja de proyecto**

Datos del estudiante:

**Nombre:** Mario Alberto Rios Mora

**Cédula:** 115130872 **Carné ITCR:** 201281252

**Dirección de su residencia:** San Ramón de Tres Rios, Salón Bellavista 50 E, 350 S. Ultima casa a mano derecha

**Teléfono de Residencia:** 22 78 62 68

**Teléfono Celular:** 87 30 72 55

**Correo Electrónico:** [mrriosm1992@gmail.com](mailto:mrriosm1992@gmail.com)

Información del proyecto:

**Nombre del Proyecto:** Control en trayectoria multi-articular para una plataforma humanoide

**Información de la empresa:**

**Nombre:** Grupo de Neuro-Rehabilitación del Instituto Cajal, CSIC.

**Actividad Principal:** Investigación

**Zona:** Madrid, España

**Dirección:** Av. Doctor Arce, 37, 28002 Madrid, España.

**Teléfono:** +34915854754

**Información del encargado/asesor en la empresa/institución:**

**Nombre:** José E. González Vargas.

**Puesto que ocupa:** Investigador del Grupo de Neuro-Rehabilitación.

**Grado académico:** Doctor

**Teléfono** +34915854754

**Correo Electrónico:** je.gonzalez@csic.es