SINTONIZACIÓN AUTOMÁTICA DE CONTROLADOR PID IMPLEMENTANDO UN ALGORITMO BIOINSPIRADO PARA UN CONVERTIDOR DC-DC BIDIRECCIONAL

ALVARO DAVID CALLEJAS LOPEZ

UNIVERSIDAD SANTO TOMÁS
INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C.
2023

SINTONIZACIÓN AUTOMÁTICA DE CONTROLADOR PID IMPLEMENTANDO UN ALGORITMO BIOINSPIRADO PARA UN CONVERTIDOR DC-DC BIDIRECCIONAL

ALVARO DAVID CALLEJAS LOPEZ

Proyecto de grado presentado como requisito para optar al título de
INGENIERO ELECTRÓNICO

DIRECTOR: CARLOS JAVIER MOJICA CASALLAS.
CODIRECTOR: JOSÉ GUILLERMO GUARNIZO MARÍN.

UNIVERSIDAD SANTO TOMÁS
INGENIERIA ELECTRÓNICA
BOGOTÁ D.C.
2023

AGRADECIMIENTOS

En primer lugar agradezco a mis padres por el apoyo que me brindaron en todo momento y especialmente en las etapas más difíciles ya que son la inspiración y la razón por la cual puede culminar mi proyecto y carrera.

Aquellos profesores que con su conocimiento y dedicación fueron mi guía para formarme como profesional.

Índice general

Resumen **9**

Introducción **10**

1. Planteamiento del problema **11**

2. Justificación **13**

3. Antecedentes **15**

4. Objetivos **21**
4.1. Objetivo General 21
4.2. Objetivos Específicos 21

5. Marco Teórico **22**
5.1. Convertidores conmutados DC/DC 22
5.1.1. Convertidor Buck 23
5.1.2. Convertidor Boost 24
5.2. Convertidor DC/DC buck-boost bidireccional no inversor . 25
5.2.1. Operación en modo Buck 26
5.2.2. Operación en modo Boost 27
5.3. Control PID 28
5.4. Sintonización de controladores PID 29
5.4.1. Ziegler-Nichols 30
5.4.2. Sintonización por mejor ganancia (Good Gain) . . . 30

5.5. Algoritmo bioinspirados 32
5.5.1. Algoritmos evolutivos [AE] 33
5.5.2. Optimización de colonias de hormigas 34
5.5.3. Optimización de enjambre de partículas [PSO] . . . 35
5.5.4. Algoritmo de selección clonal 35

6. Desarrollo Metodológico **39**
6.1. Investigación 39
6.2. Diseño y simulación del convertidor 39
6.3. Diseño del algoritmo bioinspirado 39
6.4. Simulación del algoritmo en conjunto al convertidor y ajustes 40
6.5. Implementación del controlador obtenido en la planta real . 40

7. Diseño **41**
7.1. Diseño del convertidor bidireccional 41
7.2. Diseño prototipo Buck-Boost bidireccional 42
7.3. Diseño de la interface y ley de control en LabVIEW 44

8. Resultados **48**
8.1. Comprobación del convertidor 48
8.2. Sintonización de PID mediante algoritmo clonal 49
8.3. Ejecución del algoritmo en modo Buck 52
8.4. Ejecución del algoritmo en modo Boost 56
8.5. Pruebas en el prototipo 58

9. Discusión **63**

10. Trabajo Futuros **66**

A. Anexos **72**
A.1. Codigo implementado en Matlab: 72
A.2. PCB convertidor Buck-Boost Bidireccional 81
A.3. Circuito esquemático 82

Índice de tablas

5.1. Acción de las ganancias del controlador PID. 29
5.2. Tabla Ziegler-Nichols lazo cerrado. 30

7.1. Especificaciones de diseño convertidor. 41
7.2. Valores componentes modo Buck. 42
7.3. Valores componentes modo Boost. 42

8.1. Ganacias por generación sintonicacion Buck, mutación 40 %. 52
8.2. Ganancias por generación sintonización Boost. 58

Índice de figuras

5.1. Topología convertidor Buck. 24
5.2. Topología convertidor Boost. 25
5.3. Buck-Boost bidireccional no inversor. 26
5.4. Diagrama de bloques control PID. 29
5.5. Sintonización Mejor Ganancia. 31
5.6. Estructura general algoritmo bioinspirado. 33
5.7. Algotirmo colonia de hormigas. 34
5.8. Algoritmo enjambré de partículas. 35
5.9. Diagrama de flujo algoritmo clonal para reconocimiento de patrones 36
5.10. Diagrama de flujo algoritmo clonal para la optimización . . 36
5.11. Principio de selección clonal. 38

7.1. MOSFET dual NVMFD5C650NL. 43
7.2. Esquema driver iso5451 con salida bipolar. 43
7.3. Amplificador operacional aislado AMC1200. 44
7.4. Interface de usuario en LabVIEW. 45
7.5. Diagrama PWM LabVIEW - tarjeta myRIO. 45
7.6. Adquisición de señales - tarjeta myRIO. 46
7.7. Ley de control PID - tarjeta myRIO. 46
7.8. Planta experimental. 47

8.1. Salida convertidor modo Buck en lazo abierto, Simulink. . . 48
8.2. Salida convertidor modo Boost en lazo abierto, Simulink. . 49

8.3. Error cuadrático con un porcentaje de mutación de 10 %. . 53
8.4. Error cuadrático con un porcentaje de mutación de 20 %. . 54
8.5. Error cuadrático con un porcentaje de mutación de 30 %. . 55
8.6. Error cuadrático - con un porcentaje de mutación del 40 %. 56
8.7. Comportamiento ante cambio de cargas de las ganancias obtenidas por el algoritmo. 57
8.8. Error cuadrático - con un porcentaje de mutación del 40 %. 57
8.9. Comportamiento ante cambio de cargas de las ganancias del algoritmo. 58
8.10. Cambio de carga del 100 % al 50 % modo Buck. 59
8.11. Cambio de carga del 50 % al 100 % modo Buck. 59
8.12. Cambio de carga del 100 % al 50 % modo Boost. 60
8.13. Cambio de carga del 50 % al 100 % modo Boost. 60
8.14. Cambio de carga del 50 % al 100 % modo Buck - Mejor ganancia. 61
8.15. Cambio de carga del 100 % al 50 % modo Buck - Mejor ganancia 61
8.16. Cambio de carga del 50 % al 100 % modo Boost - Mejor ganancia. 62
8.17. Cambio de carga del 100 % al 50 % modo Boost - Mejor ganancia. 62

A.1. Bottom PCB convertidos 81
A.2. Top PCB convertidos 81
A.3. Circuito convertidor Buck-Boost Bidireccional 82
A.4. Circuito driver de los MOSFETS 82
A.5. Circuito Sensores de voltaje 83

Resumen

En el presente documento se muestra el proceso de sintonización de los parámetros de un control PID que regule el voltaje de salida en un convertidor DC-DC Bidireccional. Se realiza una investigación sobre métodos implementados para la sintonización automática de controladores PID y se buscan distintos tipos de algoritmos Bioinspirados que permitan dar una solución al problema.

Se realizan simulaciones para poner en práctica el algoritmo inmune artificial que se basa en la selección clonal y que se ejecuta junto a la planta en el programa de MATLAB y Simulink, para la sintonización de los dos modos operación del convertidor DC-DC Bidireccional. En el algoritmo implementado, las ganancias del controlador PID son representadas como la población de antígenos. Se propone escoger una población inicial de 30 antígenos que toman un valor preliminar entre un rango de cero a uno. La función objetivo a minimizar está representada por el error cuadrático de su sobrepaso máximo, tiempo de establecimiento y del error de estado de estacionario que se obtienen del voltaje de salida.

La población obtenida por el algoritmo logra minimizar el error para cada operación, por lo que se realiza la comprobación de las ganancias conseguidas en un prototipo de la planta.

Introducción

Los convertidores conmutados son fuentes de energía con amplia alternativa y topologías como las bidireccionales, que satisfacen diversas necesidades en la industria automotriz o de microrred para la carga y descarga de baterías, las técnicas de control varían según el modelo del sistema, la robustez y el tipo de uso que se busque en el convertidor. El control PID es sin duda el más utilizado, ya que presenta una implementación simple, pero no mucha robustez debido a que la sintonización de este muchas veces es de prueba y error.

Como solución se plantea el uso de un algoritmo bioinspirado para la sintonización de las constantes de dicho controlador. Este documento abarca la implementación de un algoritmo de selección clonal que utiliza el principio del sistema inmune para lograr sintonizar un controlador PID para el voltaje de salida de un convertidor bidireccional en sus dos modos de operación Buck y Boost, esto minimizando la función objetivo que se relaciona con el error del parámetro a controlar, de manera que se realizan clonaciones y mutaciones durante un ciclo hasta encontrar los valores que minimicen el error.

Capítulo 1

Planteamiento del problema

Es evidente el uso de fuentes que suministran energía para el funcionamiento de los sistemas electrónicos, siendo utilizados para este trabajo las fuentes conmutadas "convertidores DC-DC". El problema aquí, conlleva a mantener una estabilidad en la salida de la fuente con un control de realimentación. Atendiendo a las variaciones de los parámetros y a las perturbaciones propias que este tipo de sistemas presentan, hace el trabajo de sintonizar el control PID más complicado [1].

La sintonización del control PID es ampliamente estudiada, y hoy en día los nuevos métodos dan una gran capacidad de adaptar este tipo de control para soportar parámetros de no linealidad y de perturbaciones en el sistema. Pero estos no poseen una respuesta deseada con un control PID mientras la sintonización del mismo no sea la adecuada, en consecuencia no se logra un control eficiente para sistemas inciertos, acoplados y no lineales [2] [3]. Por lo tanto, se buscan nuevos métodos que permitan la modificación dinámica de los parámetros de un controlador PID, y que en experimentos realizados han demostrado buenos resultados sin el conocimiento del modelo matemático de la planta [4].

El diseño de un control PID puede presentar errores, ya que la sintonización del PID con los métodos tradicionales puede generar

problemas como error de estado estacionario, sobrepaso máximos altos y tiempos de establecimientos largos o inestables que toman cierto tiempo en ajustarse, o simplemente no se logra mejorar [5]. Distintos investigadores se centran principalmente en mejorar y facilitar los métodos para la obtención de los parámetros PID y obtener un óptimo desempeño de estos. Por ejemplo, el Método de Curva de Atenuación, el Método de ziegler-Nichols, el Método de Configuración Óptima de ISTE, el Método de Configuración Rápida y el Método de Relé que permiten la obtención de los parámetros para el control PID. Pero estos métodos obtienen los parámetros mediante prueba y error, de experimentos en ejecución o fórmulas empíricas y de la variable controlada [3], haciendo que el proceso de sintonía del control sea en muchos casos tedioso y lento.

Partiendo de los problemas antes presentados se formula la siguiente pregunta de investigación.

¿Cómo implementar un algoritmo basado en técnicas bioinspiradas de manera que permita sintonizar los parámetros apropiados para un controlador PID con el fin de mejorar el desempeño de un convertidor DC-DC bidireccional?

Capítulo 2

Justificación

Desde finales de la década de los años 40 los sistemas electrónicos se han utilizado ampliamente en favor de la sociedad, una prueba de esto es el desarrollo y el amplio uso que se le dan a estos sistemas en industrias como la farmacéutica, alimenticia, automotriz, agrícola, tecnología de consumo y entre otros campos emergentes como las energías renovables. Estos sistemas o circuitos electrónicos tienen en común el uso de fuentes o convertidores conmutados, que en esencia son circuitos que permiten suplir la alimentación de todo un sistema, y es aquí donde la necesidad de controlar este tipo de fuentes es altamente requerido y crítico, siendo el controlador PID uno de los más implementados.

El campo de la teoría de control tiene como objetivo principal el estudio, análisis y diseño de controladores que permiten el correcto funcionamiento comúnmente de la variable de salida de un sistema o planta, que la compara con un punto de operación requerido según la aplicación. Para esto se han desarrollado diversas estructuras de control que poseen sus propias características que las diferencian entre sí y que pueden favorecer o no su implementación. Uno de estos sistemas es el control proporcional, integral y derivativo más conocido como PID [6]. Este es muy utilizado en la industria por la robustez que brinda a la variable de control en estado estacionario y permanente, la

implementación de este controlador toma fuerza por su fácil integración en términos del hardware como del software del sistema [7].

Las nuevas herramientas emergentes permiten que los parámetros de los controladores PID tradicionales sean capaces de sintonizarse en tiempo real utilizando la captación, almacenamiento y supervisión de las señales de la planta que sean necesarias por el controlador y empleando métodos como la respuesta al escalón o de dominio de la frecuencia permitiendo la correcta sintonización del proceso [8]. Otro punto a favor de los procesos de sintonización automática utilizados en los controladores es la poca influencia del factor humano en la configuración, selección, ejecución o supervisión del proceso, ya que se utilizan técnicas o algoritmos especificados para este proceso y en particular en los ciclos iterativos posteriores [9].

A partir de lo anterior se hace pertinente la implementación de algoritmos de aprendizaje de máquina que permita una adecuada sintonización de los parámetros de un controlador PID con el fin de mejorar el desempeño del sistema. Siendo dentro de estos algoritmos las técnicas bioinspiradas una alternativa, que aplicado a un sistema de potencia como lo es un convertidor DC-DC bidireccional permite validar el desempeño de un controlador sintonizado con técnicas aquí propuestas.

Capítulo 3

Antecedentes

La necesidad de mejorar la sintonización de los controladores PID, comprende métodos como lógica difusa, algoritmos genéticos, algoritmos evolutivos, redes neuronales, sistemas inmunes artificiales (AIS) entre otros.

Se ha implementado lógica difusa como método para ajustar los parámetros de un controlador PID de manera dinámica. Con este método se plantean dos principales objetivos, identificar el modelo matemático del sistema (fase y magnitud) implementando un procedimiento FFT (Transformada Rápida de Fourier), que se complementa con un proceso para diagnosticar las características dinámicas del sistema basado en reglas de lógica difusa. Estos dos métodos en consecuencia permiten establecer los parámetros más adecuados para el control PID [4]. Otra propuesta es el uso de un controlador PID difuso de orden fraccional (FOFPID) para controlar un sistema servo rotativo. En donde se realiza una comparación de su rendimiento con un control PID tradicional y un PID de orden fraccional (FOPID). Los autores evaluaron su rendimiento por medio de su sobrepaso máximo, tiempo de establecimiento y tiempo de subida utilizando la respuesta al escalón [10]. Otra implementación es el uso de un control PD+I difuso para mejorar el estacionamiento automático en paralelo de automóviles para disminuir el error generado

entre la posición real y la posición objetivo [11].

El uso de algoritmos genéticos (GA) es otra técnica para el ajuste de las ganancias en un control PID utilizado en sistemas de segundo y tercer orden. Este método comienza seleccionando un rango de la población inicial utilizando la fórmula de Ziegler-Nichols para evitar que el sistema sea inestable, luego de esto el algoritmo se encarga de optimizar generación tras generación los coeficientes del control PID, el método de sintonización (GA-PID) demuestra una mejora en la respuesta al escalón del sistema comparándolo con métodos como el de Ziegler-Nichols o lógica difusa [12]. Otros trabajos proponen una estrategia distribuida y con un ajuste adaptativo, de manera que se utiliza un control PID y una red neuronal artificial Adaline (AANN) para la sintonización del sistema [13].

Otra idea es usar un estabilizador de sistema de potencia (PSS) que utiliza un controlador PID para amortiguar las oscilaciones de baja frecuencia en una máquina síncrona. Se propone ajustar los parámetros del PSS utilizando la ley de adaptación y estimar los parámetros con un identificador de mínimos cuadráticos recursivos (RLS) para estimar los parámetros del PID en tiempo real. [14].

También se han presentado sintonización automática y robusta en los parámetros de un controlador PID con un sistema interno de segundo orden ajustado por el método adaptativo de un modelo parcial para controladores de primer y segundo orden. Usando un método de búsqueda secuencial ISE para la sintonización automática [15].

El continuo desarrollo de metodologías para mejorar la sintonización de los controladores PID ha incrementado, esto ya que se han demostrado lo útil que es la implementación de métodos de auto sintonización de los parámetros de control. Como en el uso de un control auto ajustable Ziegler-Nichols PID (AZNPID) que ajusta los parámetros

automáticamente para una planta de transferencia de calor, comparando el rendimiento obtenido con el método tradicional Ziegler-Nichols PID (ZNPID), teniendo como gran diferencia principal el ajuste automático en el tiempo a través de un bucle, esto hace que la respuesta del sistema mejore como el tiempo de establecimiento del mismo [16]. O la comparación de la sintonización empírica de Ziegler-Nichols y la sintonización PID modificada de Ziegler-Nichols aplicadas a un motor de corriente directa, comparando los métodos en función de la respuesta de salida, el tiempo de establecimiento y el sobrepaso máximo, demostrando en simulaciones que el método modificado de Ziegler-Nichols da mejores resultados que la técnica tradicional para controlar la velocidad del motor [5].

Otra aplicación del ajuste automático de los controladores PID es el control adaptativo de vehículos submarinos autónomos (AUV), donde el objetivo es poder adaptar el error que se genera al utilizar diferentes canales de control y sus errores variables en el tiempo, primero se utiliza el método de dinámica de fluidas computacional (CFD) para construir un modelo hidrodinámico que simula el efecto de acople para poder implementar el control adaptativo. En esta implementación, el PID que se adquiere demostró un mejor desempeño que un control PID ajustado de forma manual a la hora de realizar el acople de las señales de error [17].

Los autores de [18] muestran buenos resultados en la implementación de varios controladores como la ubicación de polos, PID y LQR. El método de comparación para de estos controladores fue en un péndulo invertido de un robot móvil de dos ruedas y utilizando CSA como método de sintonización, un modelo de optimización inmune artificial, que los autores compararon con otros cinco métodos de optimización para ver sus efectos de optimización.

Los métodos de sintonización por medio de AIS para el controlador PID

se han aplicado en modelos a escala de helicópteros, ya que el sistema presenta una mayor sensibilidad en sus entradas, por lo tanto, se utiliza la capacidad meta-heurística del sistema inmune artificial para la búsqueda del valor optimizado en los parámetros de ganancia PID. Mostrando resultados de gran rendimiento con una respuesta de seguimiento en menos de cuarenta (40) iteraciones [19]. Los algoritmos inmunes forman parte del ajuste de los controladores PID, con funciones como la diversidad, la computación distribuida, la adaptación y la función de autosupervisión, aplicando las ganancias del controlador para los valores de aptitud del algoritmo inmune para incrementar el rechazo de perturbaciones del sistema que se ajusta a una respuesta requerida comparando los resultados con respuestas basadas en FNN y algoritmos genéticos [20].

El control de retraso de tiempo (TDC) es un método que permite mantener un buen rendimiento en la carga de la batería o en el suministro de la energía requerida sin pérdidas significativas de energía en convertidores DC-DC bidireccionales[21].

Es eventual que en los convertidores DC-DC bidireccionales los métodos más comunes de control que se utilizan son el ya mencionado PID tradicional utilizado para el control de la planta de carga y descarga de las baterías de un sistema de turbinas eólicas [22]. También se implementan las variantes de los PID para los convertidores bidireccionales aplicados a vehículos eléctricos (EV) implementando un control PI y PID para la activación de los IGBT del convertidor [23], demostrando que el uso de un PI genera una curva de salida menos ruidosa en la planta. Se ha comprobado que el convertidor bidireccional DC-DC controlado por un PI obteniendo una tensión de salida estable por la topología de control PI, el control PI logra un error de estado estacionario bajo en el convertidor [24], pero este tipo de control requiere de una sintonización por lo que es habitual el desarrollo del modelo

promediado del convertidor y posteriormente se simulan las formas de ondas del modelo matemático y compararlas con las del circuito para verificar la precisión del modelo con el cual realizar el controlador [25].

Otra implementación de algoritmos bioinpirados es el algoritmo de optimización de Grey Wolves (GWO), que es comparable a la optimización por enjambre de partículas (PSO) y colonia de hormigas. En este los autores utilizaron el optimizador de Grey Wolves para mejorar el desempeño de un controlador de un MPPT y comparándolo con métodos tradicionales, los autores lograron obtener un 6 % más de eficiencia con este algoritmo [26].

En [27] realizaron una comparación entre diversos algoritmos bioinspirados para la sintonización de los parámetros de un controlador PI y también de un controlador PID utilizados para regular el voltaje en un convertidor tipo Buck-Boost, las técnicas de optimización utilizadas fueron el algoritmo genético (AG), optimización de enjambre de partículas (PSO), optimización de hormiga león (ALO), algoritmo de optimización de ballenas (WOA) y optimización por GreyWolves (GWO). Para comparar los rendimientos de los algoritmos, los autores compararon el comportamiento transitorio de los valores óptimos obtenidos por cada algoritmo y controlador con cambios en la carga del sistema, demostrando que el convertidor más adecuado es el PID y el algoritmo AG es la mejor técnica para optimizar el control PID en un convertidor Buck-Boost.

Los autores de [28] implementan los algoritmos bioinspirados de optimización de enjambre de partículas (PSO) y algoritmo genético (GA), utilizados para ajustar los parámetros de un controlador Gaussiano adaptativo PID (GAPID), en su investigación utilizaron un convertidor reductor Buck para este estudio y determinaron que ambos algoritmos son capases de sintonizar el controlador (GAPID) donde este demuestra mejoras en comparación con un control PID clásico (47 % para el GA y

52 % para PSO), destacando el algoritmo PSO implementador por su velocidad de procesamiento al converger con tan solo 20 iteraciones.

Capítulo 4

Objetivos

4.1. Objetivo General

Implementar un algoritmo bioinspirado que permita la sintonización automática de un controlador PID aplicado a un conversor DC-DC bidireccional, para el control del voltaje de salida.

4.2. Objetivos Específicos

- Diseñar e implementar un convertidor DC-DC bidireccional de 12 V a 24 V a 100 W.
- Diseñar e implementar un algoritmo de sintonización automático de PID, basado en sistemas inmunes artificiales, que permita sintonizar los parámetros del controlador, a partir de su respuesta en el tiempo.
- Determinar una métrica que permita la comparación a partir de figuras de mérito basadas en la respuesta del tiempo, con el fin de evaluar el desempeño del controlador obtenido.
- Evaluar el desempeño del algoritmo de sintonización del PID implementado, comparando el desempeño del controlador antes y después de la sintonización, con otro obtenido a partir de la técnica de Ziegler-Nichols.

Capítulo 5

Marco Teórico

5.1. Convertidores conmutados DC/DC

En electrónica de potencia, si se parte de la idea de almacenar energía para la conversión DC/DC (corriente continua a corriente continua), es indispensable el uso de semiconductores que permitan la posibilidad de tener un voltaje menor o mayor según sea su voltaje de entrada variando la topología de estos convertidores. Siendo utilizados como elementos indispensables para este fin el uso de transistores MOSFETs, diodos, inductores y capacitores, que varían en cantidad y ubicación según la topología.

Los convertidores conmutados, a diferencia de las fuentes lineales, tienen una mayor eficiencia en su transformación de la energía, pero presentan la generación de ruido producto de la conmutación en el MOSFET.

Siendo las configuraciones más básicas el convertidor Buck y el convertidor Boost.

5.1.1. Convertidor Buck

La topología del convertidor Buck obtiene a su salida una tensión continúa menor que a su entrada en función del ciclo útil de trabajo 5.4 que se le aplique al interruptor de entrada MOSFET o IGBT, también consta de un filtro pasa bajos LC en su salida y un diodo de conmutación rápida que permite el cambio en la corriente del inductor según el estado del interruptor. El diagrama de circuito para el convertidor reductor se muestra en la figura 5.1

La relación del voltaje de salida con respecto al de entrada se obtiene siempre que la corriente en la bobina sea positiva y según el tiempo en el que el interruptor se encuentre cerrado y abierto, de tal forma el periodo de conmutaciones T, el interruptor se encuentra cerrado en un tiempo DT y abierto en con un tiempo de (1-D)T.

$$\Delta iL(abierto) = -\frac{Vo}{L} * (1 - D) * T \tag{5.1}$$

$$\Delta iL(cerrado) = \frac{(Vi - Vo)}{L} * DT \tag{5.2}$$

Teniendo en cuenta que la operación del convertidor es en régimen permanente y la corriente en el inductor no tiene que variar al principio y final del ciclo de trabajo, por lo que subaración neta tendrá que ser cero tenemos que:

$$\Delta iL(cerrado) + \Delta iL(abierto) = 0 \tag{5.3}$$

Utilizando las ecuaciones 5.3 y 5.1 se obtiene que la relación de voltaje en el convertidor Buck sin contemplar las pérdidas en el sistema es:

$$Vo = Vi * D \tag{5.4}$$

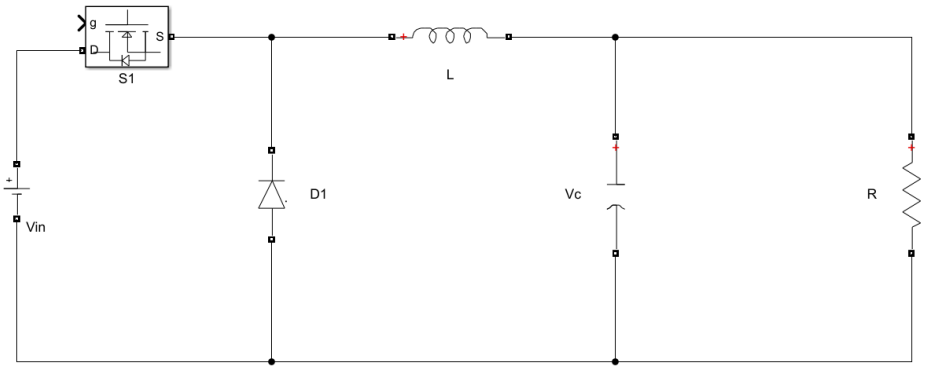

Figura 5.1. Topología convertidor Buck.
(Fuente: Autor)

5.1.2. Convertidor Boost

La topología del convertidor Boost obtiene a su salida una tensión continua mayor que a su entrada en función de su ciclo útil de trabajo 5.7. La entrada a este convertidor puede ser de cualquier fuente de DC como panel solar, baterías, etc. El diagrama de circuito para el convertidor elevador se muestra en la Figura 5.2

Teniendo presente que el convertidor funciona en régimen permanente, podemos obtener la variación de la corriente en el inductor cuando el interruptor se encuentra encendido y apagado.

$$\Delta iL(abierto) = -\frac{(Vi - Vo) * (1 - D) * T}{L} \tag{5.5}$$

$$\Delta iL(cerrado) = \frac{Vi * DT}{L} \tag{5.6}$$

Utilizando las equiaciones 5.6 y 5.5 obtenemos que la relación de voltaje en el convertidor Buck sin contemplar las pérdidas en el sistema:

$$Vo = \frac{Vi}{(1-D)} \tag{5.7}$$

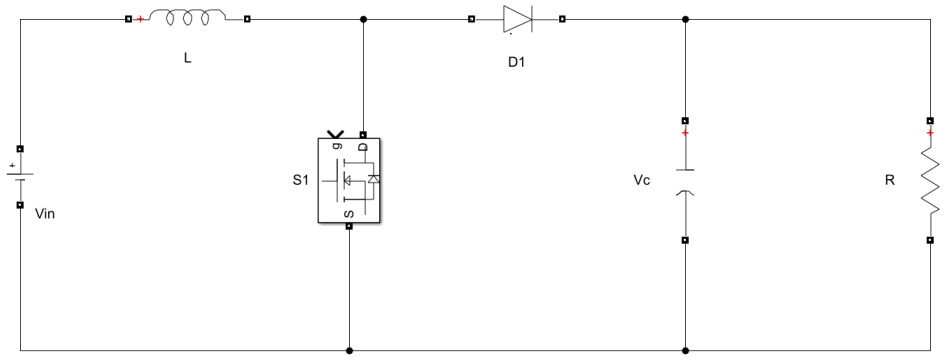

Figura 5.2. Topología convertidor Boost.
(Fuente: Autor)

5.2. Convertidor DC/DC buck-boost bidireccional no inversor

El uso de convertidores bidireccionales permite el flujo de potencia en dos direcciones, ideal para administrar la energía de un sistema fotovoltaico o de un generador DC para la carga de baterías. La estructura del convertidor Buck-Boost bidireccional no inversor de medio puente permite configurar dos modos de trabajo que dependen del flujo de la potencia, modo reductor y elevador, como se muestra en la Figura 5.3.

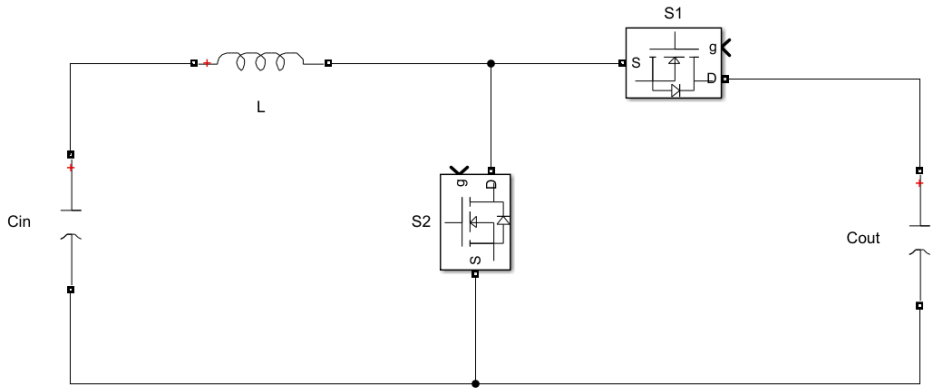

Figura 5.3. Buck-Boost bidireccional no inversor.
(Fuente: Autor)

Para realizar el diseño del convertidor de la figura 5.3 se considera que el circuito opera en régimen permanente, la corriente presente en la bobina L es estable y siempre es positiva, los valores de los condensadores son muy grandes y la tensión de salida sea en cualquier modo de operación se mantiene constante a un voltaje Vo, el periodo de conmutación T es la suma del tiempo en el que el interruptor está cerrado DT y del tiempo restante en el que se encuentre abierto $(1 - D)T$, se asumen componentes ideales para el análisis y se plantean las ecuaciones de estado en cada tiempo de conmutación para los dos modos de operación Buck y Boost.

5.2.1. Operación en modo Buck

Para diseñar eel convertidor, primero se define el ciclo de trabajo (D) que permite definir los componentes del circuito, D se puede calcular despejándolo de la relación del voltaje de salida y entrada de la ecuación 5.4

$$D = \frac{Vo}{Vin} \tag{5.8}$$

Se define la corriente de salida del convertidor $Iout$.

$$Iout = \frac{W}{Vout} \tag{5.9}$$

Se calculan las variaciones en la corriente del inductor y la tensión en el condensador de salida.

$$\Delta V = Vout * 0{,}01 \tag{5.10}$$

$$\Delta I = Iout * 0{,}18 \tag{5.11}$$

En donde se espera una variación del 1 % para la tensión de salida y 18 % para la corriente en el inductor.

Se definen los valores para los elementos del convertidor. La resistencia de salida,

$$R = \frac{Vout^2}{W} \tag{5.12}$$

La inductancia,

$$L = \frac{Vin * (1 - D) * D}{\Delta I * Fsw} \tag{5.13}$$

El capacitor de salida,

$$Cout = \frac{Vin * (1 - D) * D}{8 * L * \Delta Vout * Fsw^2} \tag{5.14}$$

5.2.2. Operación en modo Boost

Para el diseño del convertidor en este modo de operación, al igual que el modo buck se define el ciclo de trabajo para poder definir sus

componentes. De la ecuación 5.7 optenemos,

$$D = 1 - \frac{Vint}{Vout} \tag{5.15}$$

Se calculan las variaciones en la corriente del inductor y la tensión en el condensador de salida con las ecuaciones 5.10 y 5.11 respectivamente, pero esperando una variación de 36 % para la corriente en el inductor.

Se definen los valores para los elementos del convertidor. La inductancia,

$$L = \frac{Vin * (1 - D)}{\Delta I * Fsw} \tag{5.16}$$

El capacitor de salida,

$$Cout = \frac{Vin * D}{\Delta Vout * (1 - D) * Fsw * R} \tag{5.17}$$

5.3. Control PID

Los controladores PID, han demostrado su robustez en diversas aplicaciones por su simpleza en su implementación y la eficiente solución a muchos problemas de control en la industria desde su invención en 1910 y por el método de sintonización de ziegler-Nichols en 1942.

El controlador PID presenta una estructura tipo (SISO), como se muestra en la Figura 5.4 con una entrada y una salida y con un grado de libertad.

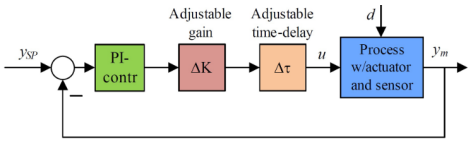

Figura 5.4. Diagrama de bloques control PID.
Fuente [29]

Un controlador PID implica el uso de tres acciones, proporcional(P), integral(I) y derivativa(D), de estas ganancias se derivan las familias del controlador PID que se denominan P, I, PI, PD y PID.

$$u(t) = Kpe(t) + Ki\int_0^T e(t) * dt + Kd\frac{\mathrm{de(t)}}{\mathrm{d}t} \tag{5.18}$$

Siendo la parte proporcional quien otorga una acción proporcional a la señal de error, la parte integral reduce el error de estado estacionario mediante el integrador con su baja frecuencia y el derivador mejoró la respuesta transitoria del sistema con un diferenciador. En la tabla 5.1 se muestra una comparativa de cada en la acción de cada ganancia en el controlador PID.

Respuesta	Sobre Paso	t Establecimiento	Error
Aumentando Kp	Aumenta	Aumenta Poco	Disminuye
Aumentando Ki	Aumenta	Aumenta	Gran Disminución
Aumentando Kd	Disminuye	Disminuye	Cambio bajo

Tabla 5.1. Acción de las ganancias del controlador PID.
(Fuente: Autor)

5.4. Sintonización de controladores PID

Existen varios métodos para la sintonización de un controlador PID, entre estos se encuentran los métodos de Ziegler-Nichols o el método de mejor

ganancia, ambos métodos de lazo cerrado.

5.4.1. Ziegler-Nichols

El método de lazo cerrado de Ziegler-Nichols. Propone inicialmente establecer en cero la ganancia integral y derivativa e ir incrementando poco a poco la ganancia proporcional del controlador. Esto hasta que el sistema comience a oscilar con una amplitud constante en el tiempo, pero sin llegar a la inestabilidad. Este método se puede realizar en plantas de segundo orden o superior.

Al conseguir estas oscilaciones con amplitud constante la ganancia proporcional obtenida se denomina Ganancia Crítica: Ku y a partir de la respuesta se calcula el Periodo Crítico: Pu, con estos parámetros se pueden encontrar las ganancias del controlador con la tabla 5.2.

Controlador	Kp	Ti	Td
P	$0.5Ku$	∞	0
PI	$0.45Ku$	$\frac{1}{1,2}\ Kp$	0
PID	$0.6Ku$	$0.5Pu$	$0.125Pu$

Tabla 5.2. Tabla Ziegler-Nichols lazo cerrado.
(Fuente: Autor)

5.4.2. Sintonización por mejor ganancia (Good Gain)

El método de mejor ganancia, al igual que el método de Ziegler-Nichols, es un método experimental que se puede utilizar en procesos sin tener conocimiento del proceso a controlar, este método no requiere que el lazo de control entre en oscilaciones durante la sintonización.

Con el método de mejor ganancia se propone establecer las ganancias Kp, Ki y Td inicialmente en cerro e ir incrementando la ganancia proporcional

poco a poco hasta llegar a obtener una buena estabilidad en la variable de salida, como se muestra en la figura 5.5.

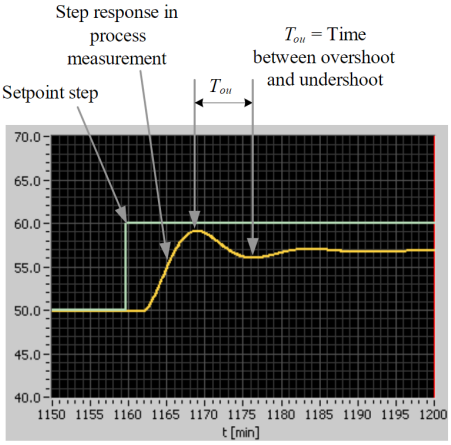

Figura 5.5. Sintonización Mejor Ganancia.
Fuente [29]

Luego de obtener una buena estabilidad se obtienen las ganancias del controlador PID en donde, $KpGG$ es la ganancia que estabiliza el sistema y Tou es el tiempo desde el sobrepaso máximo y el siguiente paso mínimo,

$$Kp = 0{,}8 * KpGG \tag{5.19}$$

$$Ti = 1{,}5 * Tou \tag{5.20}$$

$$Td = \frac{Ti}{4} \tag{5.21}$$

5.5. Algoritmo bioinspirados

Los algoritmos bioinspirados se basan en replicar el comportamiento aleatorio de la naturaleza. Estos utilizan este comportamiento a la hora de afrontar problemas de optimización. Los algoritmos bioinspirados buscan optimizar o maximizar ciertas funciones objetivo, en donde los algoritmos deterministas no son eficientes. Además, los algoritmos bioinspirados a menudo presentan una estructura paralela y adaptativa[30].

En los algoritmos bioinspirados se pueden encontrar la siguiente estructura en común tal cual se presenta en la Figura 5.6:

- **Población:** Conjuto de datos a ser solución.
- **Mecanismo de evolución:** metodo por el cual se modifica la poblacion.
- **Mecanismo de desempeño:** método por el cual se asigna un valor a cada elemento de la población (solución), normalmente la función a optimizar.
- **Condición de parada:** procedimiento que controla si se ha encontrado una solución o se ha alcanzado un número de operaciones determinado con anterioridad.

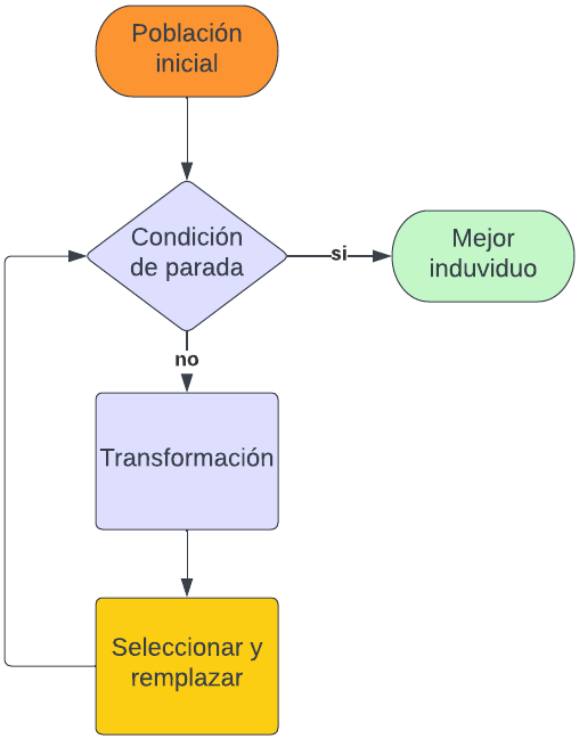

Figura 5.6. Estructura general algoritmo bioinspirado.
(Fuente: Autor)

Ningún algoritmo es mejor para resolver todos los problemas de optimización como tal, ya que el rendimiento de estos dependen de las características del problema, por lo que también hay variedad de algoritmos que pueden llegar a una optimización en menor tiempo.

5.5.1. Algoritmos evolutivos [AE]

Estos son una familia de algoritmos bioinspirados que replican el proceso de evolución natural de las especies, para resolver problemas de optimización. Los algoritmos evolutivos parten de un conjunto inicial aleatorio de candidatos para solucionar un problema. Estos candidatos

son sometidos a un proceso de selección y modificación, en donde son favorecidos los individuos más aptos. Cada ciclo de selección y modificación corresponde a una generación.

A diferencia de las otras técnicas de optimización, los algoritmos evolutivos cuentan con una amplia gama de soluciones tentativas que mediante la aplicación de algunas variaciones es posible encontrar soluciones óptimas y globales [31].

5.5.2. Optimización de colonias de hormigas

Los algoritmos de optimización de colonia de hormigas se encuentran inspirados en el comportamiento de hormigas reales, en donde estas encuentran la ruta más corta hacia su alimento mediante la comunicación indirecta entre ellas llamada caminos de feromonas, figura 5.7. La idea principal del algoritmo es modelar el problema como la búsqueda de una ruta mínima de costo en un gráfico. Por sí solas estas hormigas artificiales no son capaces de encontrar los mejores caminos, siendo necesaria la cooperación de la colonia [32]

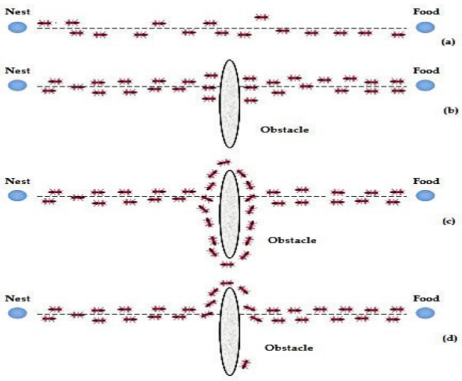

Figura 5.7. Algotirmo colonia de hormigas.
Fuente [33]

5.5.3. Optimización de enjambre de partículas [PSO]

La optimización de enjambre de partículas (PSO) fue diseñada por Kennedy y Eberhart. Este método imita el comportamiento de un enjambre Figura 5.8, como la formación de bancos de peces y las bandadas de aves. Por ejemplo, un pájaro puede encontrar comida y compartir esta información con otros. Por lo tanto, las aves acuden en masa a esta posición o región prometedora en busca de alimento.

El PSO imita el comportamiento de las aves mediante el uso de una población llamada enjambre. El miembro del grupo se denomina como una partícula. Cada partícula encuentra una solución al problema. De este modo, se comparte la posición de la experiencia o la información y las partículas actualizan la posición y buscan la solución ellas mismos [34].

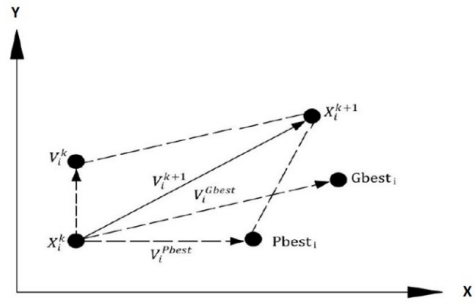

Figura 5.8. Algoritmo enjambré de partículas.
Fuente[6]

5.5.4. Algoritmo de selección clonal

Este algoritmo se aplica para el reconocimiento de patrones y para la optimización de funciones. En donde se emplea al antígeno como el análogo, el problema a resolver y los anticuerpos como las posibles soluciones Figura 5.11. La naturaleza del algoritmo permite que este evite caer en un óptimo local debido a la cantidad de conjuntos potenciales a ser una posible solución. Uno de los inconvenientes que posee este

algoritmo es su elevado coste de cómputo, que se puede solucionar reduciendo el número de la población pero afectando el tiempo de ejecución en generaciones o iteraciones del algoritmo.[35]

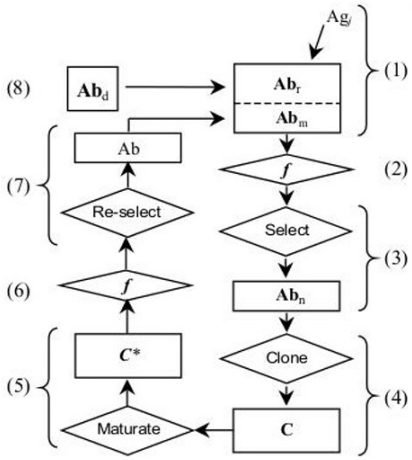

Figura 5.9. Diagrama de flujo algoritmo clonal para reconocimiento de patrones

Fuente [36]

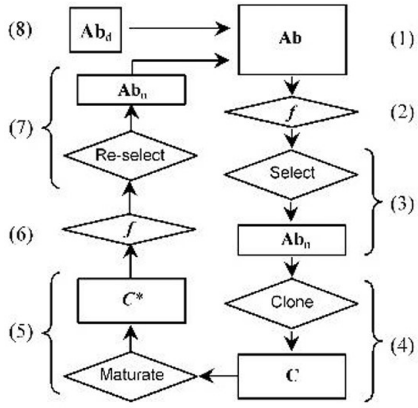

Figura 5.10. Diagrama de flujo algoritmo clonal para la optimización

Fuente [36]

Este algoritmo hace uso de los siguientes operadores genéticos:

Generación: creación de los anticuerpos iniciales por los linfocitos B.

Reconocimiento: esto determina si el antígeno detectado ha sido reconocido previamente, al poseer una similitud con un antígeno detectado previamente o simplemente es totalmente nuevo.

Evaluación: comportamiento empleado para la categorización de los anticuerpos en su desempeño.

Selección: Los anticuerpos con mayor capacidad de detectar el antígeno.

Clonación: es un aumento de la población al replicar los anticuerpos elegidos dentro de la población con mejor desempeño.

Mutación: es una perturbación genética aleatoria que permite explorar diversas zonas de búsqueda y diversificar la población con nuevos individuos.

Memoria: reserva de los mejores individuos y su información.

Eliminación: es la exclusión de los anticuerpos de menor capacidad.

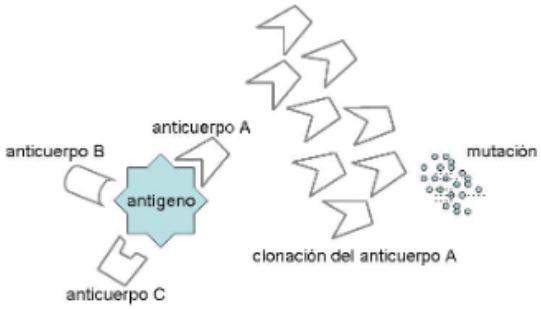

Figura 5.11. Principio de selección clonal.
Fuente [35]

El algoritmo de selección clonal para la optimización de función objetivo presenta los siguientes pasos característico:

1. Población inicial de anticuerpos, Ab, con tamaño N.
2. Evaluación de los anticuerpos con la función de desempeño (f).
3. Selección de los anticuerpos que presenten una mayorr afinidad (Abn).
4. Clonación de los anticuerpos con mayor afinidad (C).
5. Mutación inversamente proporcional a su afinidad ($C*$), A B.
6. Evaluar de la nueva función con la función de desempeño ($f*$).
7. Selección de anticuerpos que presentan un valor óptimo para hacer parte de la población de memoria.
8. El resto de anticuerpos serán reemplazados o excluidos de la nueva población.

Capítulo 6

Desarrollo Metodológico

6.1. Investigación

Se realizó una investigación de los diferentes tipos de algoritmos bioinspirados para la optimización de problemas y poder implementarlo en la sintonización de un controlador PID, también se investigó sobre convertidores DC-DC Bidireccionales y el uso que estos tienen en la industria.

6.2. Diseño y simulación del convertidor

Una vez que se escogió el convertidor Bidireccional a implementar se diseña bajo los parámetros establecidos en los objetivos, para esto se optó por establecer dos formas de funcionamiento del convertidor modo Buck y Boost con la finalidad de facilitar su diseño y obtener los valores de sus componentes, posterior a esto se realizó una comprobación del diseño en Simulink.

6.3. Diseño del algoritmo bioinspirado

Una vez definido el algoritmo, este se desarrolló en el entorno de MATLAB de forma que se pudiera escoger el tamaño y tipo de la

población (ganancias del control PID), el número de generaciones, la función objetivo a minimizar, el porcentaje de mutación y su condición de parada. En este punto también se implementó la comunicación entre Simulink y MATLAB para obtener el comportamiento de la población sobre el convertidor.

6.4. Simulación del algoritmo en conjunto al convertidor y ajustes

Una vez determinados los parámetros iniciales del algoritmo, se ejecuta junto con el convertidor y evaluar su comportamiento para minimizar la función objetivo, mostrando las ganancias obtenidas por cada generación junto a su error, con el fin de medir su desempeño y poder realizar los ajustes requeridos en el algoritmo y que se consigan los valores deseados.

6.5. Implementación del controlador obtenido en la planta real

Luego de encontrar las ganancias por medio del algoritmo, se procede con la implementación en el prototipo de la planta, el cual se diseñó por el autor para ser utilizada en conjunto con la placa myRIO y su FPGA, para implementar el controlador PID y comprobar el comportamiento en lazo cerrado de la planta ante perturbaciones en la carga de tal forma que se pueda comparar el resultado del algoritmo con otro método de sintonización.

Capítulo 7

Diseño

7.1. Diseño del convertidor bidireccional

Partiendo de la investigación, se realiza el diseño del convertidor bidireccional a implementar mencionado en el capítulo 5.2 para que cumpla con los objetivos planteados y presentes en la siguiente tabla 7.1,

Variable	Valor
V1	12V
V2	24V
W	100W
Fsw	30000

Tabla 7.1. Especificaciones de diseño convertidor.
(Fuente: Autor)

En donde V1 y V2 son las entradas de voltaje en cada modo de operación Boost y Buck respectivamente, W la potencia máxima de salida y Fsw la frecuencia de conmutación implementada.

Utilizando las ecuaciones 5.13, 5.14, 5.16y 5.17 obtenemos los valores de las tablas 7.2 y 7.3 según el modo de operación,

Componente	Valor
R	1.44Ω
L	$130uH$
Cout	$50uF$

Tabla 7.2. Valores componentes modo Buck.
(Fuente: Autor)

Componente	Valor
R	5.76Ω
L	$130uH$
Cout	$280uF$

Tabla 7.3. Valores componentes modo Boost.
(Fuente: Autor)

7.2. Diseño prototipo Buck-Boost bidireccional

Se diseña un prototipo PCB del circuito Buck-Boost Bidireccional. El circuito presenta tres etapas importantes. En primer lugar, el convertidor bidireccional DC-DC no aislado como se muestra en la figura A.3, para este se implementó un transistor MOSFET dual de canal N de potencia con diodos en antiparalelo (NVMFD5C650NL) para permitir la bidireccionalidad de la planta, este presenta un voltaje Drain-Source de 60 V y una corriente Drain máxima de 111A.

$V_{(BR)DSS}$	$R_{DS(ON)}$ MAX	I_D MAX
60 V	4.2 mΩ @ 10 V	111 A
	5.8 mΩ @ 4.5 V	

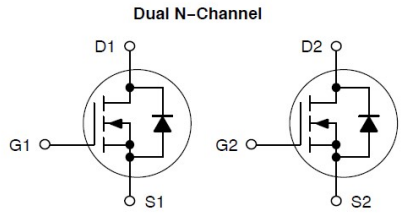

Figura 7.1. MOSFET dual NVMFD5C650NL.
(Fuente: hoja de datos)

En segundo lugar, se diseñó el circuito de conmutación de los MOSFETs implementando el driver iso5451 como se muestra en la figura A.4, este driver se implementó debido al aislamiento que permite entre los voltajes digitales de control y voltajes de potencia como medida de protección entre estas dos partes, también por las diferentes protecciones que maneja en para el MOSFET.

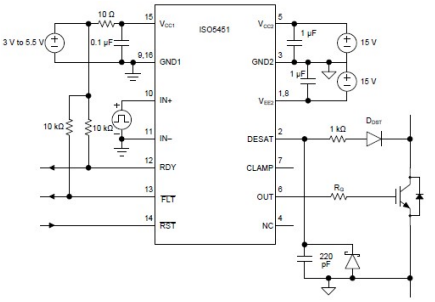

Figura 7.2. Esquema driver iso5451 con salida bipolar.
(Fuente: hoja de datos)

En la figura 7.2 proporcionada por el fabricante, se observa que la alimentación de la etapa de potencia en el driver entre los pines Vcc, Vee y GND2 corresponde a una alimentación bipolar de +15 V y -15 V, esta es una medida para evitar falsos disparos del MOSFET a causa de ruido

que se pueda generar, para esto se utilizó la fuente aislada TEL 8-2423WI con voltaje de entrada de 12 V y salida bipolar de 15 V.

Por último, se diseñó una etapa de senado de voltaje implementando el integrado AMC1200, un amplificador diferencial aislado que permite separar la etapa de potencia con la de control, este integrado realiza la medición con gran precisión, la ganancia por cada 250mV es de 2 V con un nivel DC de 1.29V. La señal de salida se adapta a un nivel de 0 V a 5 V con un amplificador "rail to rail" en modo diferencial, como se muestra en la figura A.5.

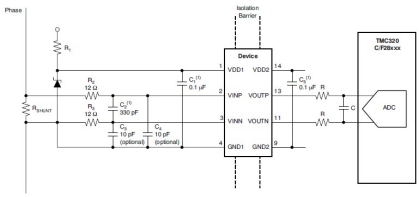

Figura 7.3. Amplificador operacional aislado AMC1200.
(Fuente: hoja de datos)

7.3. Diseño de la interface y ley de control en LabVIEW

Se decide realizar una interface gráfica que permita la interacción con el control PID implementado en la tarjeta myRIO, utilizando para ello el programa de LabVIEW tanto para programar la tarjeta como realizar la interface gráfica.

Esta interface permite seleccionar el modo de operación del convertidor Buck-Boost bidireccional, ya sea para operación elevadora o reductor se puede introducir los parámetros Kp, Ti y Td del controlador PID. De igual manera, en esta interface se pueden observar los valores de voltaje de entrada, salida y corriente en el inductor en tiempo real Figura 7.4.

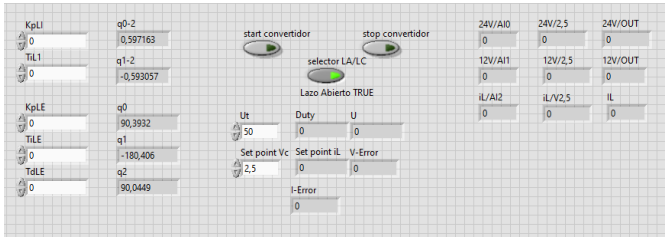

Figura 7.4. Interface de usuario en LabVIEW.
(Fuente: Autor)

Se decide utilizar la FPGA de la tarjeta myRIO para implementar el control PID digital obtenido con el algoritmo CLONALG, para esto primero fue necesario implementar un PWM con una frecuencia de 30KHz, que se implementa utilizando un TIME LOOP que utiliza el reloj de la FPGA para realizar las operaciones en un tick que equivale a 25ns Figura 7.5.

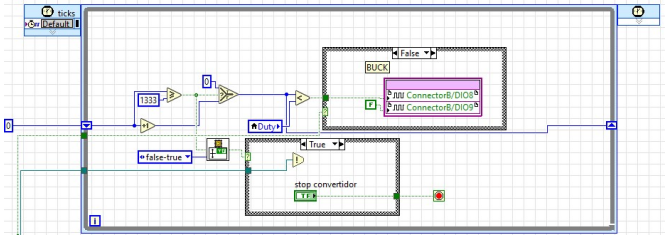

Figura 7.5. Diagrama PWM LabVIEW - tarjeta myRIO.
(Fuente: Autor)

Una vez implementado el PWM, se realiza el código para la adquisición de las señales de los sensores y posteriormente la ley de control, utilizando un Flat Sequense que permite en LabVIEW ejecutar código de forma secuencial Figura 7.6. La adquisición de las señales se realiza en la mitad de la conmutación para evitar el ruido en la lectura de las variables por los efectos de la conmutación del convertidor.

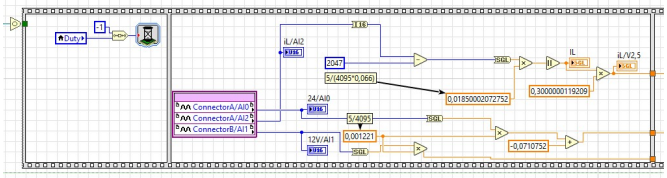

Figura 7.6. Adquisición de señales - tarjeta myRIO.
(Fuente: Autor)

Esta parte de código se ejecuta una vez cada 33 microsegundos, esto se logra utilizando ocurrencias en el código, que permite la sincronización de módulos en LabVIEW. De esta forma, cada vez que el contador del PWM se reinicie, se ejecuta esta parte del código figura 7.6, junto con la ley de control figura 7.7.

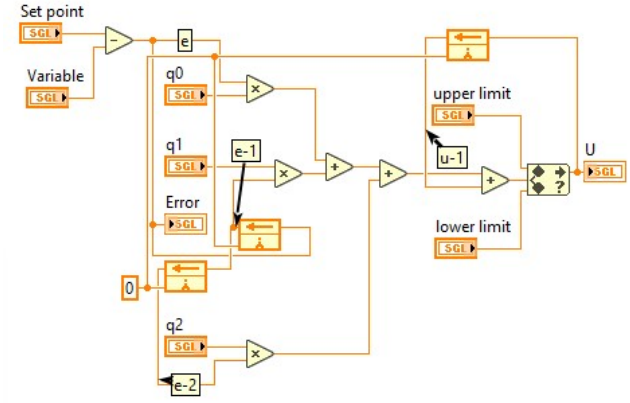

Figura 7.7. Ley de control PID - tarjeta myRIO.
(Fuente: Autor)

Ya programada la interface gráfica y la tarjeta myRIO solo queda verificar el funcionamiento de la ley de control y los parámetros obtenidos en cada modo de operación por el algoritmo clonal, para ello se utilizó el prototipo del convertidor Bidireccional Figura 7.8

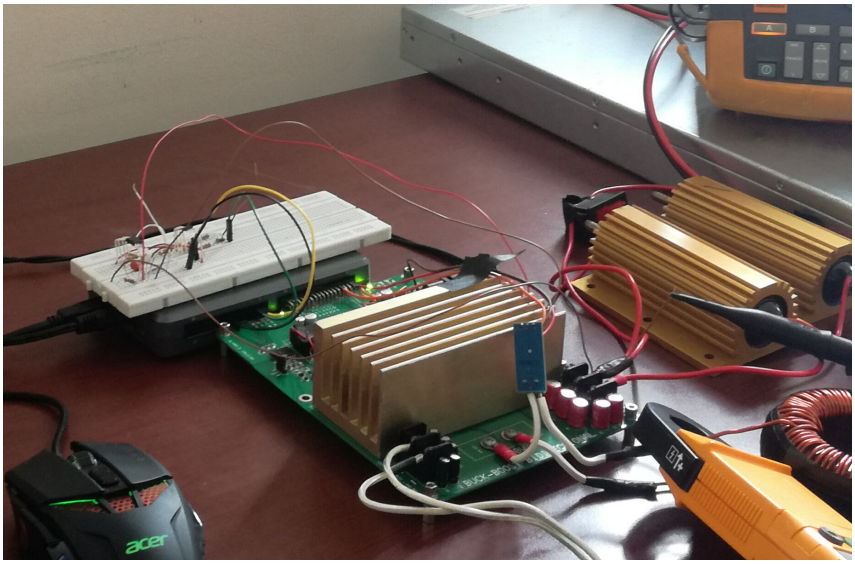

Figura 7.8. Planta experimental.
(Fuente: Autor)

Capítulo 8

Resultados

8.1. Comprobación del convertidor

Mediante el programa Simulink se realizó las simulaciones del convertidor para comprobar su correcto funcionamiento en lazo abierto.

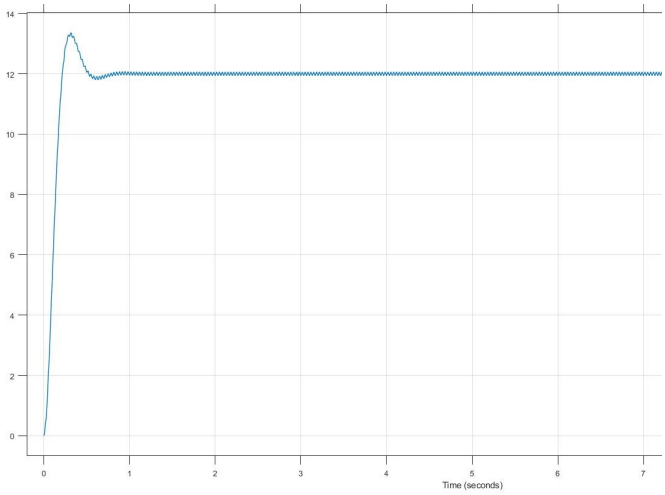

Figura 8.1. Salida convertidor modo Buck en lazo abierto, Simulink.

(Fuente: Autor)

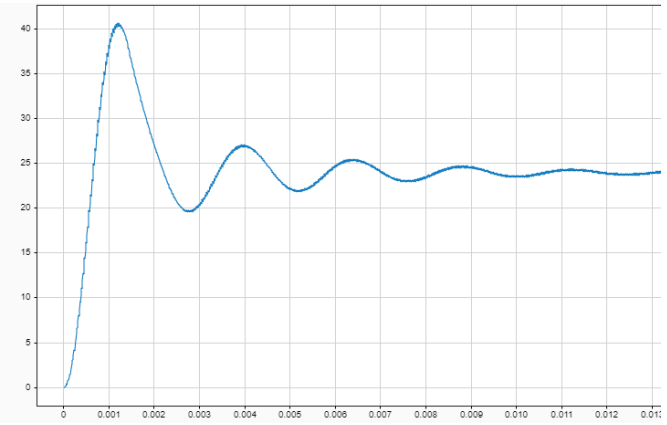

Figura 8.2. Salida convertidor modo Boost en lazo abierto, Simulink.

(Fuente: Autor)

En las figuras 8.1 y 8.2 se observa las salidas del convertidor en sus dos modos de operación y que los valores de los componentes pasivos satisfacen el punto de operación deseado. Se puede observar de en modo reductor el voltaje de salida corresponde a 12V, presenta un sobrepaso del 13,5 % y un tiempo de establecimiento de 600 ms. De igual forma, en el modo de operación elevador presenta 24V en su salida, un sobrepaso del 66 % y un tiempo de establecimiento de 9 ms

8.2. Sintonización de PID mediante algoritmo clonal

La sintonización utilizada para los controladores PID de ambos modos de funcionamiento se realizó de manera Off-line implementando el algoritmo de selección CLONALG A.1 y Figura 5.10 que se enfoca en la optimizacion de minimizar el error de la funcion objetivo que califica el desempeño de cada anticuerpo, definida como:

$$f(x) = \sqrt{K1e_{mp}^2 + K2e_{ts}^2 + K3e_{ess}^2} \tag{8.1}$$

Donde e_{mp}, e_{ts}, e_{ess} representan la presión de los antígenos para cumplir con los parámetros del diseño y (K_1, K_2, K_3) son constantes que se determinan heurísticamente y se encargan de dar mayor o menor importancia a las figuras de mérito al definir la aptitud de los antígenos,

$$e_{Mp} = \frac{M_{pD} - M_P}{M_{pD}} \tag{8.2}$$

$$e_{ts} = \frac{t_{sD} - t_s}{t_{sD}} \tag{8.3}$$

$$e_{ess} = \frac{e_{ssD} - e_{ss}}{e_{ssD}} \tag{8.4}$$

En donde los valores (M_p, t_s, e_{ss}) son los valores de sobrepaso máximo, tiempo de establecimiento y error de estado estacionario, extraídos del voltage de salida de la planta, (M_{pD}, t_{sD}, e_{ssD}) son los valores a los que se desean llegar y se establecen al inicio del algoritmo.

Al iniciar, la población inicial se establece en treinta individuos, en donde las ganancias (antígenos) del controlador PID en la primera generación toman un valor aleatorio entre cero y uno, cada uno de los antígenos es evaluado por la función objetivo de la ecuación 8.1 de esta primera generación se escoge el 30 % de la población con el mejor desempeño o aptitud para ser clonados en razón de la ecuación 8.5.

$$K_n = K + rand(-K, K) * N \tag{8.5}$$

Luego de realizar la clonación y mutación de los antígenos, se evalúan estos nuevos individuos con la función objetivo 8.1 para medir el desempeño de esta población clonada y mutada, en caso de que algún individuo satisfaga la función se terminará la ejecución del algoritmo, en caso contrario el 10 % de los mejores clones se escogen para una nueva generación y se completa la nueva población con un rango aleatorio entre los valores máximos y mínimos del 10 % de los clones durante cincuenta generaciones. Cómo se representa en la figura 5.9

En caso de no converger el algoritmo se realizarán los ajustes necesarios para llegar a una solución, siendo necesario modificar el valor de mutación de cada antígeno o las constantes (K1, K2, K3)

Una vez entendido el funcionamiento del algoritmo para la sintonización del controlador PID, se procede a ejecutar el algoritmo en conjunto a la planta simulada en Simulink, se ejecuta la sintonización implementando un controlador independiente para cada modo de operación.

Para esto se establecen los errores deseados para cada modo de operación, en donde el error de estado estacionario no debe ser mayor al 2 %, el sobrepaso máximo debe ser inferior al 10 % del valor final y el tiempo de establecimiento tendrá que ser menor o igual a la obtenida en lazo abierto como se muestra en las figuras 8.1 y 8.2.

8.3. Ejecución del algoritmo en modo Buck

Para comenzar, se decide iniciar la simulación con un porcentaje de mutación del 10 % e ir aumentando su valor en las simulaciones posteriores si es necesario.

En la figura 8.3 se puede observar que el porcentaje de mutación del 10 % no es suficiente para que el algoritmo converja, por lo que se decide aumentar este valor al 20 % llegando a reducir el error en 25 generaciones como se muestra en la figura 8.4, si se aumenta este porcentaje de mutación se observa que el algoritmo logra converge en menor cantidad de generaciones Figuras 8.5 y 8.8.

Geneación	Kp	Ti	Td
1	0,5247	0,8045	0,2450
2	0,5247	0,7445	0,1250
3	0,8493	1,334	1,878e-04
4	1,3486	0,253	2,4055e-05
5	2,7513	0,2805	4,0895e-05
6	4,6773	0,2806	6,9521e-05
7	7,9561	0,28064	6,9521e-05
8	10,5173	0,04779	6,9521e-05
9	9,8768	0,010338	2,843e-6

Tabla 8.1. Ganacias por generación sintonicacion Buck, mutación 40 %.
(Fuente: Autor)

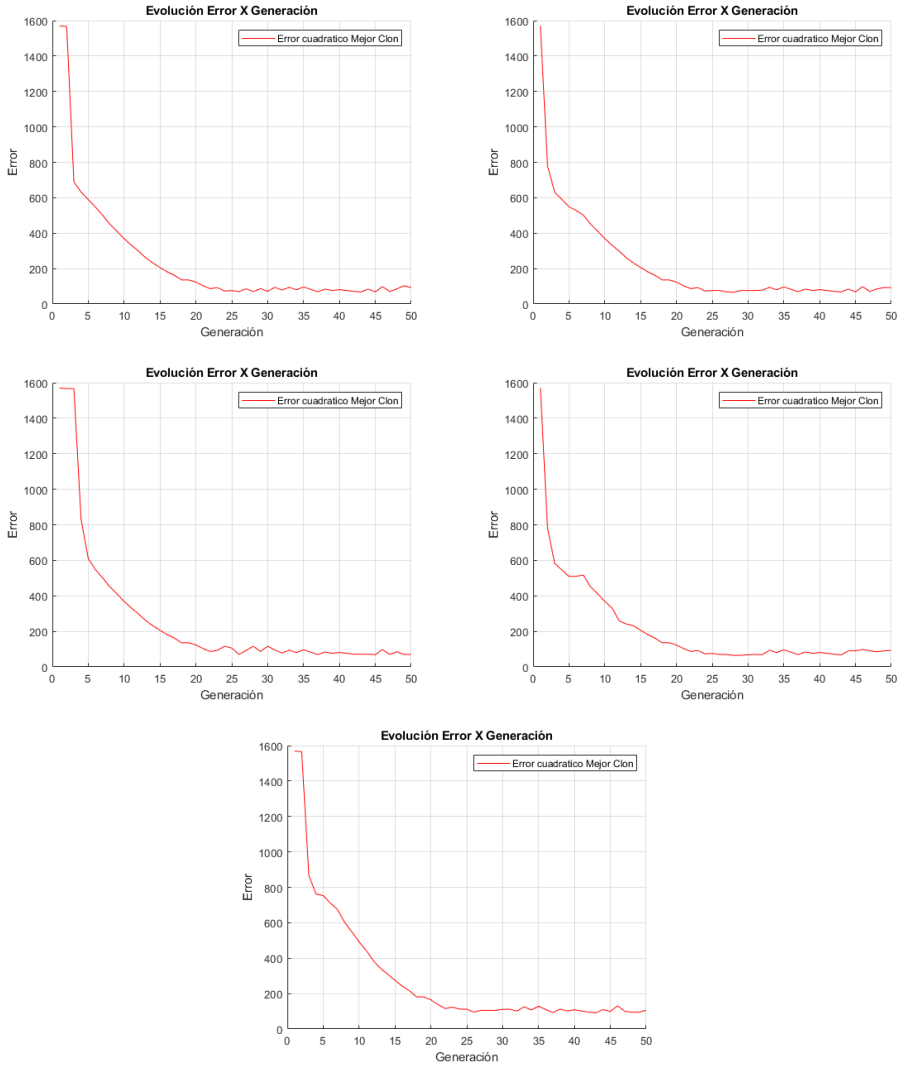

Figura 8.3. Error cuadrático con un porcentaje de mutación de 10 %.

(Fuente: Autor)

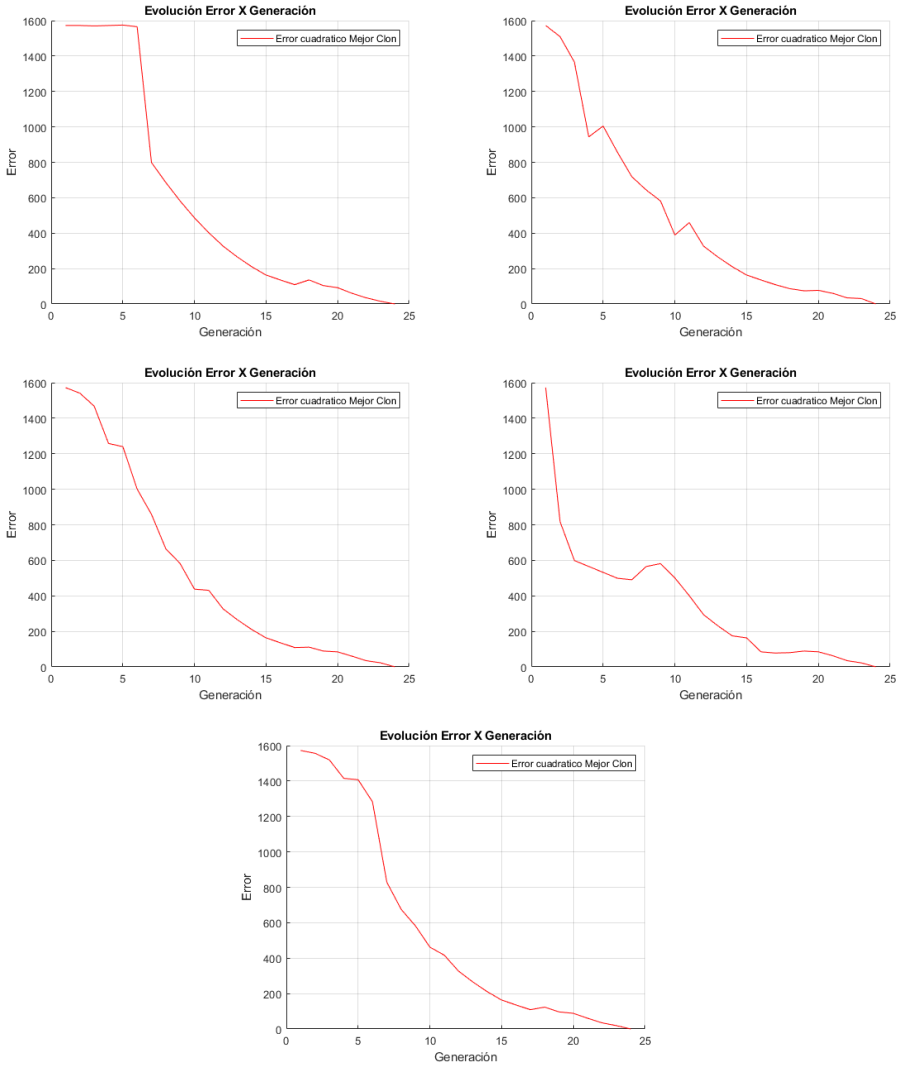

Figura 8.4. Error cuadrático con un porcentaje de mutación de 20 %.

(Fuente: Autor)

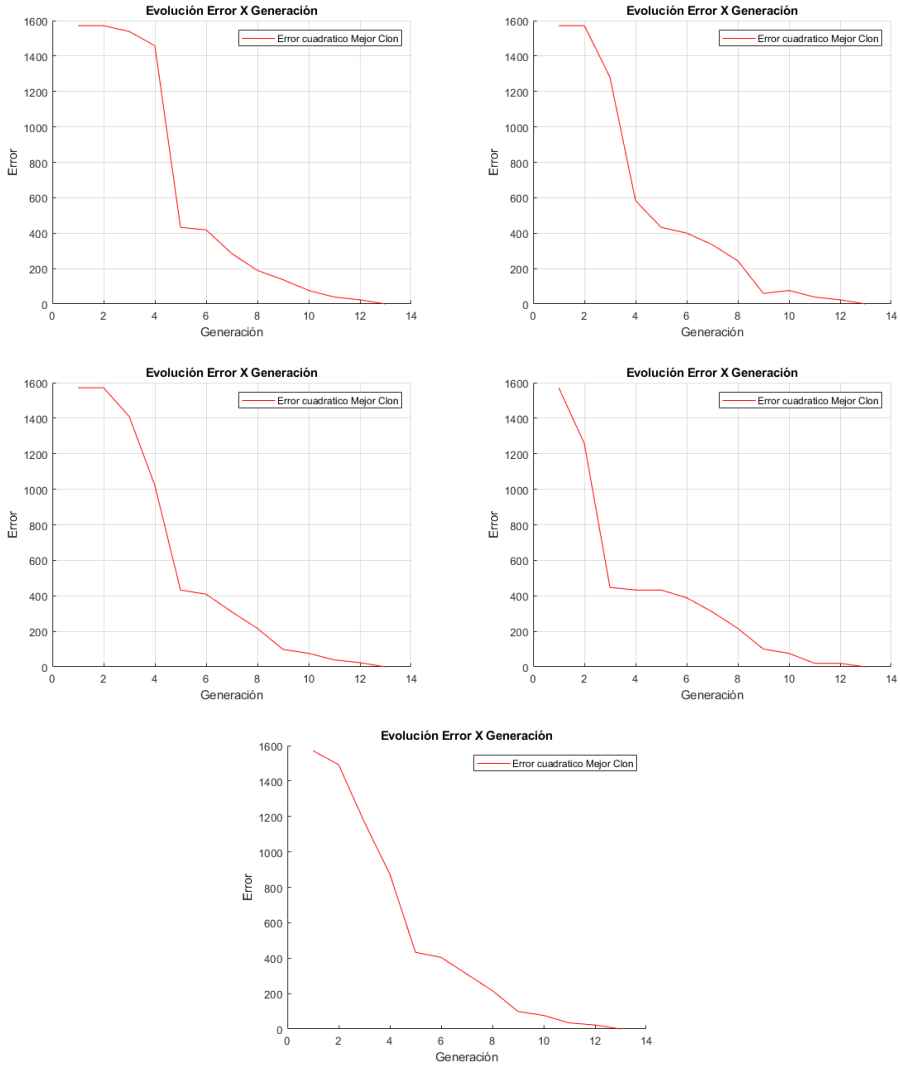

Figura 8.5. Error cuadrático con un porcentaje de mutación de 30 %.

(Fuente: Autor)

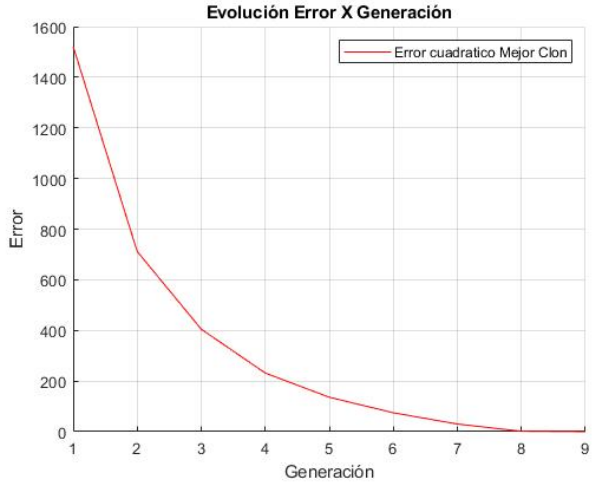

Figura 8.6. Error cuadrático - con un porcentaje de mutación del 40 %.

(Fuente: Autor)

En la tabla 8.1 se muestran el comportamiento de las mejores ganancias desde la primera generación hasta que el algoritmo llega a converger en la novena generación. La Figura 8.7 muestra el comportamiento de la planta en simulación con las ganancias de PID obtenidas.

8.4. Ejecución del algoritmo en modo Boost

Para realizar la sintonización del convertidor en modo Boost fue necesario implementar un lazo de control en cascada, realizando un control PI interno para un lazo de corriente y un lazo de control externo PID para el voltaje. Esto se realiza porque el modelo del convertidor en modo Boost con respecto al voltaje de salida es de fase no mínima, dificultando el control de voltaje para un controlador lineal tipo PID.

Ya que el algoritmo de sintonización converge con un porcentaje de mutación del 40 % en el convertidor en modo Buck, se decide utilizar este mismo valor para realizar la sintonización del controlador PID de la

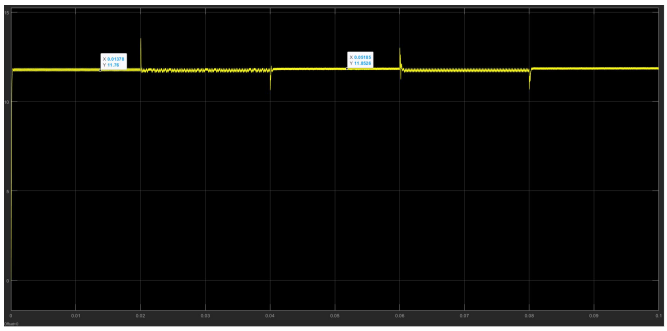

Figura 8.7. Comportamiento ante cambio de cargas de las ganancias obtenidas por el algoritmo.
(Fuente: Autor)

planta en modo Boost Figura 8.8.

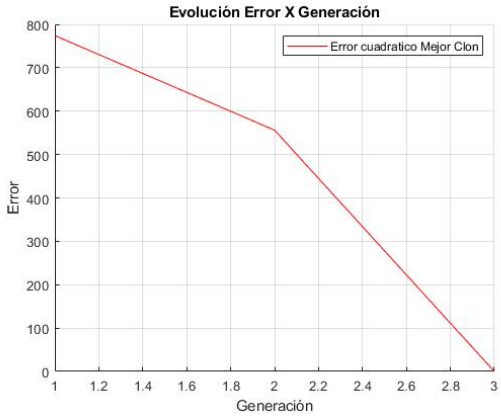

Figura 8.8. Error cuadrático - con un porcentaje de mutación del 40 %.
(Fuente: Autor)

En la tabla 8.2 se muestra los valores de las ganancias del controlador PID, desde la primera hasta la última generación. La Figura 8.9 muestras el comportamiento de las ganancias obtenidas por el algoritmo.

Geneación	Kp	Ti	Td
1	0.927	0.068	0.547
2	1.858	0.0412	0.7671
3	1,735718	0,0005768	0,000179

Tabla 8.2. Ganancias por generación sintonización Boost.
(Fuente: Autor)

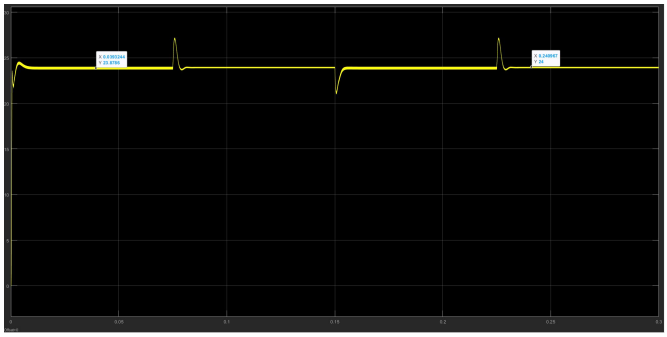

Figura 8.9. Comportamiento ante cambio de cargas de las ganancias del algoritmo.
(Fuente: Autor)

En la figura 8.8 se observa que el algoritmo converge en tres décadas.

8.5. Pruebas en el prototipo

Luego de realizar la comprobación del funcionamiento de las ganancias obtenidas por los algoritmos figuras 8.7 y 8.9 se procede a realizar las pruebas de dichas ganancias en el prototipo de la planta. Como se había mencionado en capítulos anteriores, se utilizó la tarjeta myRIO para implementar el control digital, esto con el fin de poder realizar un programa en LabVIEW que permita la interacción con el usuario y poder cambiar tanto el modo de operación, como las ganancias del controlador Figura 7.4.

Para las pruebas del modo Buck de las figuras 8.10 y 8.11 se utilizaron las ganancias $Kp = 9{,}876801$ $Ti = 0{,}010338$ $Td = 0{,}000028$

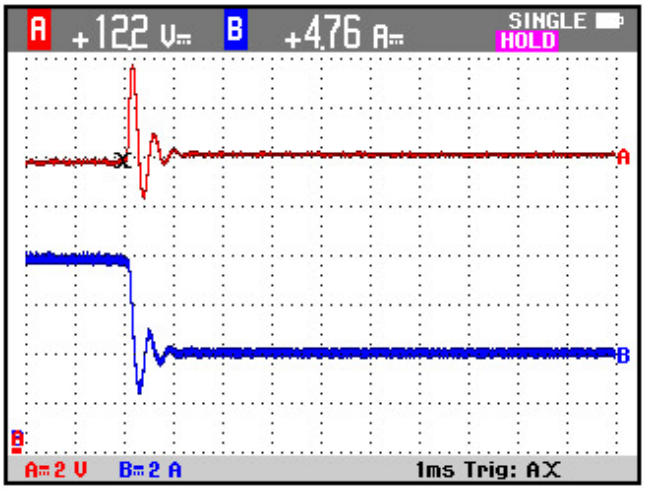

Figura 8.10. Cambio de carga del 100 % al 50 % modo Buck.
(Fuente: Autor)

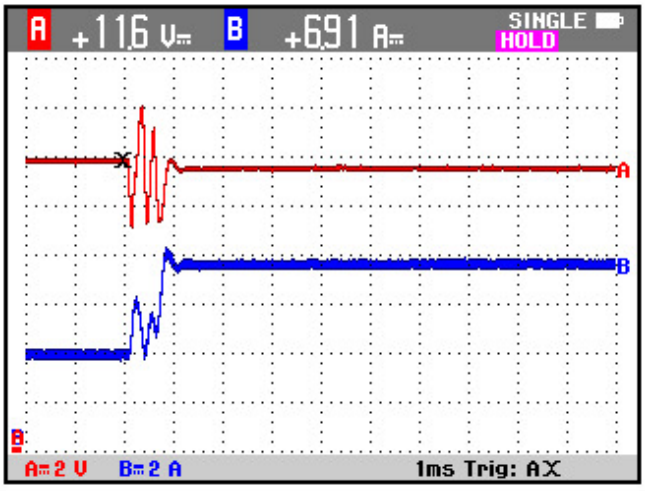

Figura 8.11. Cambio de carga del 50 % al 100 % modo Buck.
(Fuente: Autor)

Para las pruebas del modo Boost de las figuras 8.12 y 8.13 se utilizaron las ganancias $Kp = 1{,}735718$ $Ti = 0{,}0005768$ $Td = 0{,}000179$ para el lazo de voltaje y $Kp = 0{,}511477$ $Ti = 0{,}001428$ para el lazo de corriente.

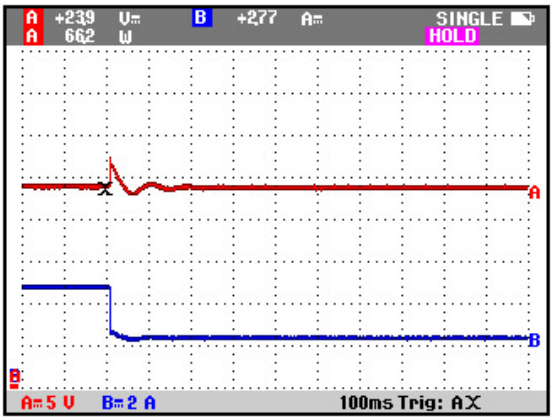

Figura 8.12. Cambio de carga del 100 % al 50 % modo Boost.
(Fuente: Autor)

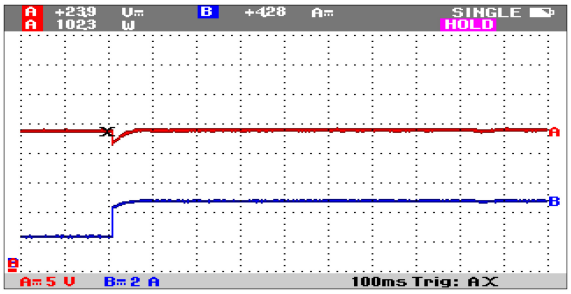

Figura 8.13. Cambio de carga del 50 % al 100 % modo Boost.
(Fuente: Autor)

Para el modo Boost fue necesario implementar el sensor de voltaje fuera de la PCB, ya que esta presenta ruidos por la conmutación que afectan la señal en el pin del ADC cuando se llegaba a un voltaje de salida superior a los 19 voltios.

Para la sintonización de comparación se utilizó el método de Mejor

Ganancia explicada previamente, esto es debido a que la sintonización por ziegler-Nichols por el método de resonancia no es compatible con sistemas de primer o segundo orden, por los que el sistema no logra oscilar en ningún modo de funcionamiento.

Las ganancias utilizadas por el método de Mejor ganancia en el modo Buck son $Kp = 0{,}0128$ $Ti = 0{,}00035850$ $Td = 0{,}000089625$ Figuras 8.14 y 8.15.

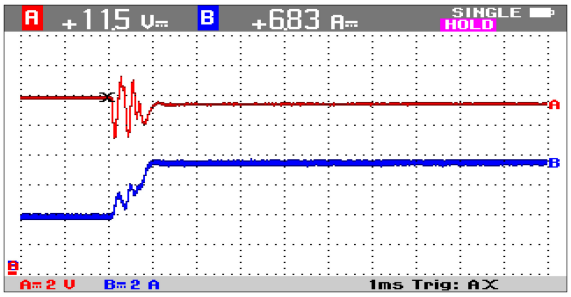

Figura 8.14. Cambio de carga del 50 % al 100 % modo Buck - Mejor ganancia.

(Fuente: Autor)

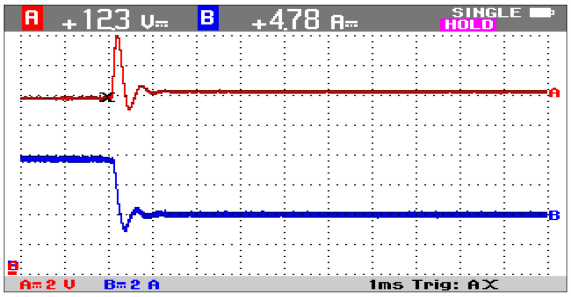

Figura 8.15. Cambio de carga del 100 % al 50 % modo Buck - Mejor ganancia

(Fuente: Autor)

Las ganancias utilizadas por el método de Mejor ganancia en el modo Boost son $Kp = 1{,}6000$ $Ti = 0{,}0019$ $Td = 0{,}00048750$ para el lazo de voltaje y $Kp = 0{,}0144$ $Ti = 0{,}0008445$ para el lazo de corriente Figuras 8.16 y 8.17.

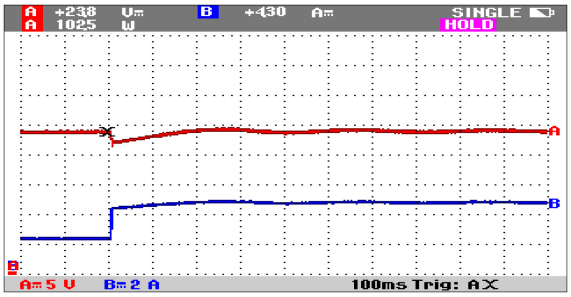

Figura 8.16. Cambio de carga del 50 % al 100 % modo Boost - Mejor ganancia.

(Fuente: Autor)

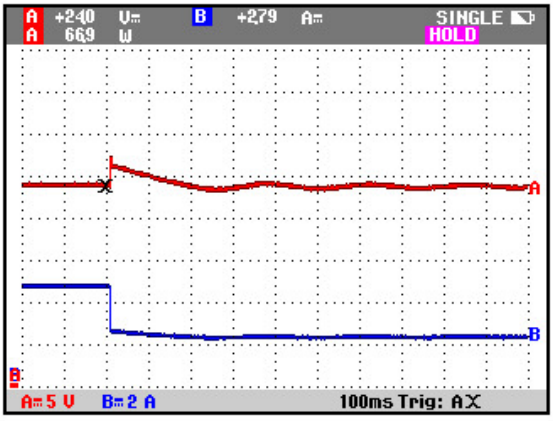

Figura 8.17. Cambio de carga del 100 % al 50 % modo Boost - Mejor ganancia.

(Fuente: Autor)

Capítulo 9

Discusión

El implementar este convertidor DC-DC bidireccional Buck Boost de medio puente permitió una facilidad en su diseño, ya que este se pudo realizar con los requisitos de funcionamiento como dos modos de operación independientes, pero compartiendo los valores de la tabla 7.1.

Al momento de realizar la sintonización del controlador PID por medio del algoritmo clonal se puede ver que el uso de una población aleatoria como primera generación no es impedimento para que el algoritmo llegue a una solución por lo menos en una sintonización Off-Line, se podría mirar si este factor no cambia en una sintonización online.

Las primeras simulaciones que se hicieron para sintonizar el controlador PID en el modo Buck muestran que un valor muy bajo en el porcentaje de mutación de un 10 % no es capaz de hacer que el algoritmo llegue a una solución, esto cambia al momento de incrementar este porcentaje de mutación entre valores del 20 % al 40 %, ya que se muestra que con un 20 % el algoritmo si logra una solución al converger en 25 generaciones y que mejora incrementando la mutación hasta un 40 %, que disminuye el número de generaciones necesarias para llegar a una solución con un total de 9 generaciones. Al momento de realizar la sintonización del controlador PID con el otro modo de funcionamiento Boost se realiza directamente

con un porcentaje de mutación del 40 % llegando a una solución en tan solo tres generaciones, esto corrobora el funcionamiento del algorítmico en simulaciones, cabe aclarar que para el modo Boost se utiliza un lazo interno de control PI para la corriente en la bobina ya que la topología del convertidor Boost hace que su sistema sea de fase no mínima dificultando el control de voltaje para un controlador lineal como el PID.

Al realizar las comprobaciones de las ganancias obtenidas en las simulaciones en el prototipo se ve una clara similitud con sus simulaciones en el comportamiento ante el cambió en la carga y se evidencia que los controladores logran seguir la referencia. Al comparar con la sintonización de mejor ganancia vemos que el modo Buck presenta una similitud en la respuesta, en el modo Boost el controlador sintonizado por el algoritmo clonal presenta una mejora en su tiempo de establecimiento en comparación con el método de mejor ganancia.

Conclusiones

El algoritmo es capaz converger como se observa en la figura 8.4, cumpliendo los parámetros de diseño del controlador PID especificados para la salida de voltaje en los dos modos de operación, pero se puede evidenciar que el algoritmo puede caer en un mínimo local si el porcentaje mutación no es el adecuado como se observa en la figura 8.3, esto se puede solucionar aumentando el porcentaje de mutación.

En este desarrollo también se puede evidenciar que el rango de la población inicial utilizado en el algoritmo para la optimización de este problema no es decisivo a la hora de llegar a converger el algoritmo. También se evidencia una similitud en el anticuerpo que minimiza la función objetivo con los valores del método de sintonía de comparación, esto indica que al utilizar estos valores como rangos acotados para generar la población inicial, el número de generaciones para que converja el algoritmo se verá drásticamente reducida.

Teniendo presente que la sintonización del algoritmo como la del método de comparación son sintonizaciones Offline, la implementación en el prototipo presenta un comportamiento muy similar a la simulación, pero teniendo un mejor comportamiento el control obtenido con el algoritmo en el modo Boost, siendo más rápido en el tiempo de establecimiento obtenido con el algoritmo, ya que es uno de los parámetros que conforman la función objetivo como parámetro de comprobación.

Capítulo 10

Trabajo Futuros

Como trabajo futuro se busca implementar la sintonización online del algoritmo, para esto hay que tener en cuenta los rangos de la población inicial ya que los rangos grandes y aleatorios pueden afectar variables del sistema que no se tienen en cuenta y que en simulación no presentan mayor problema como la corriente que pueden ocasionar daños en la planta en una implementación real.

Otro punto a futuro es la implementación de un solo controlador que permita el cambio en el flujo de la corriente del inductor cuando el sistema lo requiera, por lo que se deberá tener en cuenta el uso de un simulador de una microred.

También se busca mejorar el diseño del PCB para el prototipo de la planta, ya que este presentó mucho ruido en la conmutación, generando daños en las señales de los sensores.

Bibliografía

[1] S. Mumtaz, S. Ali, S. Ahmad, L. Khan, S. Z. Hassan, and T. Kamal, "Energy management and control of plug-in hybrid electric vehicle charging stations in a grid-connected hybrid power system," *Energies*, vol. 10, no. 11, 2017.

[2] W. A. Shutnan and T. Y. Abdalla, "Artificial immune system based optimal fractional order pid control scheme for path tracking of robot manipulator," in *2018 International Conference on Advance of Sustainable Engineering and its Application (ICASEA)*, pp. 19–24, 2018.

[3] Q. Meng and T. Liu, "Study on immune pid control method of an in-wheel motor used in an electric car," in *2017 36th Chinese Control Conference (CCC)*, pp. 9554–9559, 2017.

[4] R. Ferreiro Garcia and F. Perez Castelo, "Fuzzy adaptive pid controller using frequency techniques," in *1996 IEEE International Conference on Systems, Man and Cybernetics. Information Intelligence and Systems (Cat. No.96CH35929)*, vol. 4, pp. 2581–2585 vol.4, 1996.

[5] P. M. Meshram and R. G. Kanojiya, "Tuning of pid controller using ziegler-nichols method for speed control of dc motor," in *IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012)*, pp. 117–122, 2012.

[6] H. Yazgan, F. Yener, S. Soysal, and A. Gür, "Comparison performances of pso and ga to tuning pid controller for the dc motor," *Sakarya University Journal of Science*, pp. 1–1, 04 2019.

[7] A. Saleem, H. Soliman, S. al ratrout, and M. Mesbah, "Design of a fractional order pid controller with application to an induction motor drive," *TURKISH JOURNAL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCES*, vol. 26, pp. 2768–2778, 09 2018.

[8] J. Portillo, M. Marcos, and D. Orive, "Auto-tuner: Tuning commercial pid controllers on demand in industrial environments," in *1999 European Control Conference (ECC)*, pp. 747–752, 1999.

[9] P. S. Malik, S. S. Gawas, I. A. Patel, N. P. Parsekar, A. A. Parab, and S. S. Parkar, "Transient response improvement of dc to dc converter by using auto-tuned pid controller," in *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pp. 546–549, 2018.

[10] R. Bhimte, K. Bhole, and P. Shah, "Fractional order fuzzy pid controller for a rotary servo system," in *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 538–542, 2018.

[11] J. Ghaisari, M. Danesh, and N. Naderi Samani, "Parallel parking of a car-like mobile robot based on the p-domain path tracking controllers," *IET Control Theory & Applications*, vol. 10, 02 2016.

[12] L. Fan and E. M. Joo, "Design for auto-tuning pid controller based on genetic algorithms," in *2009 4th IEEE Conference on Industrial Electronics and Applications*, pp. 1924–1928, 2009.

[13] L. R. Dos Santos, F. R. Durand, A. Goedtel, and T. Abrao, "Auto-tuning pid distributed power control for next-generation passive optical networks," *Journal of Optical Communications and Networking*, vol. 10, no. 10, pp. D110–D125, 2018.

[14] G. Kasilingam and J. Pasupuleti, "Auto tuning of pid controller of a synchronous machine connected to a linear and non linear load," in *2014 IEEE International Conference on Power and Energy (PECon)*, pp. 71–76, 2014.

[15] M. Katoh and A. Fujiwara, "Auto tuning on robust parameters of a pid controller for a 2nd-order adjusted system with one changeable parameter," in *SICE Annual Conference 2007*, pp. 1624–1630, 2007.

[16] A. K. Sukede and J. Arora, "Auto tuning of pid controller," in *2015 International Conference on Industrial Instrumentation and Control (ICIC)*, pp. 1459–1462, 2015.

[17] X.-Y. Liu, Y.-P. Li, S.-X. Yan, and X.-S. Feng, "Adaptive attitude controller design of autonomous underwater vehicle focus on decoupling," in *2017 IEEE Underwater Technology (UT)*, pp. 1–7, 2017.

[18] Y. Olmez, G. O. Koca, and Z. H. Akpolat, "Clonal selection algorithm based control for two-wheeled self-balancing mobile robot," *Simulation Modelling Practice and Theory*, vol. 118, 2022.

[19] S. F. Toha, A. F. A. Rahim, H. Mansor, and R. Akmeliawati, "Pid tuned artificial immune system hover control for lab-scaled helicopter system," in *2015 10th Asian Control Conference (ASCC)*, pp. 1–4, 2015.

[20] D. H. Kim and J. H. Cho, "Robust tuning for disturbance rejection of pid controller using evolutionary algorithm," in *IEEE Annual Meeting of the Fuzzy Information, 2004. Processing NAFIPS '04.*, vol. 1, pp. 248–253 Vol.1, 2004.

[21] D.-H. Yu, Y.-X. Wang, and Y. B. Kim, "Robust control for bi-directional dc/dc converter for increase of battery state of charge," in *2013 CACS International Automatic Control Conference (CACS)*, pp. 527–531, 2013.

[22] R. P. Eviningsih, R. I. Putri, M. Pujiantara, A. Priyadi, and M. H. Purnomo, "Controlled bidirectional converter using pid for charging battery in the stand-alone wind turbine system with modified p amp;o to obtain mppt," in *2017 International Conference on Green Energy and Applications (ICGEA)*, pp. 69–73, 2017.

[23] R. Das and M. A. UddinChowdhury, "Pi controlled bi-directional dc-dc converter (bdddc) and highly efficient boost converter for electric vehicles," in *2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*, pp. 1–5, 2016.

[24] R. Das, H. Rashid, and I. U. Ahmed, "A comparative analysis of pi and pid controlled bidirectional dc-dc converter with conventional bidirectional dc-dc converter," in *2017 3rd International Conference on Electrical Information and Communication Technology (EICT)*, pp. 1–6, 2017.

[25] H. Riazmontazer, J. Moghani, M. Taheri, and H. Bayat, "Averaged modeling of a new bi-directional dc-dc converter," in *2011 2nd Power Electronics, Drive Systems and Technologies Conference*, pp. 574–580, 2011.

[26] J. Aguila-Leon, C. Vargas-Salgado, C. Chiñas-Palacios, and D. Díaz-Bello, "Solar photovoltaic maximum power point tracking controller optimization using grey wolf optimizer: A performance comparison between bio-inspired and traditional algorithms," *Expert Systems with Applications*, vol. 211, 2023.

[27] K. P. Vittal, S. Bhanja, and A. Keshri, "Comparative study of pi, pid controller for buck-boost converter tuned by bio-inspired optimization techniques," pp. 219–224, Institute of Electrical and Electronics Engineers Inc., 2021.

[28] E. D. P. Puchta, H. V. Siqueira, and M. D. S. Kaster, "Optimization tools based on metaheuristics for performance enhancement in a

gaussian adaptive pid controller," *IEEE Transactions on Cybernetics*, vol. 50, pp. 1185–1194, 3 2020.

[29] B. Popadic, B. Dumnic, D. Milicevic, V. Katic, and Z. Corba, "Tuning methods for pi controller - comparison on a highly modular drive," 2013.

[30] K. K. Mishra, S. Tiwari, and A. K. Misra, "A bio inspired algorithm for solving optimization problems," pp. 653–659, 2011.

[31] Z. Tang, "Comparison between hierarchical distributed evolutionary algorithms and general distributed evolutionary algorithms," pp. 2158–2161, Institute of Electrical and Electronics Engineers Inc., 2013.

[32] R. Jangra and R. Kait, "Analysis and comparison among ant system; ant colony system and max-min ant system with different parameters setting," Institute of Electrical and Electronics Engineers Inc., 7 2017.

[33] K. Loubna, B. Bachir, and Z. Izeddine, "Optimal digital iir filter design using ant colony optimization," pp. 1–5, Institute of Electrical and Electronics Engineers Inc., 5 2018.

[34] M. Anantathanavit and M. A. Munlin, "Radius particle swarm optimization," pp. 126–130, 2013.

[35] C. D. Investigación, E. N. Computación, J. Carlos, H. Lozada, F. Hiram, C. Castro, D. Hind, and T. México, "Instituto politÉcnico nacional sistema inmune artificial con población reducida para optimización numérica," 2011.

[36] L. C. Payo, "Implementación de algoritmo de scan-matching basado en clonalg," 7 2015.

Apéndice A

Anexos

A.1. Codigo implementado en Matlab:

```
% ALGORITMO CLONAL
clc,clear
%
   ------------------------------------------------------------------------



%CONFIGURACION CONVERTIDOR
Convertidor="Buck";
%Convertidor="Boost";

if Convertidor=="Buck"
    Vout = 12;
    Ts = 1/30000;
end
if Convertidor=="Boost"
    IL = 8.3;
    Vout=24;
    Ts = 1/30000;
end
```

```
%
   ------------------------------------------------------------------------


%CLONALG
N_Poblacion=30;
N_Iteraciones=50;
Elementos=3;    %-Kp
                %-Ti
                %-Td

Poblacion_Actual=CrearPoblacion(N_Poblacion,
   Elementos,0,1);
N_Mejores=round(0.3*N_Poblacion);

N_Poblacionc=0;
for x=1:N_Mejores
    N_Poblacionc=N_Poblacionc+round((0.5*
       N_Poblacion)/x);
end
N_Mejoresc=round(0.3*N_Poblacionc);
P=0;
CD_Decempeno=0;

for Generacion=1:N_Iteraciones

    fprintf('Generaci n actual = %d\n\n',
       Generacion);

    for x=1:N_Poblacion
        Desempeno(x)=fitness(Convertidor,Vout,
```

```
        Poblacion_Actual(:,x),"no");
end

[Mejores_Desempenos,Indices_Ordenados]=sort(
   Desempeno);

%POBLACION ORDENADA DE - A +
   ----------------------------------------

PA_Ordenada=Poblacion_Actual(:,
   Indices_Ordenados);

Referencia=0;
for x=1:N_Mejores
   N_Clones=round((0.5*N_Poblacion)/x);
   for y=Referencia+1:N_Clones+Referencia
        for z=1:Elementos



                if rand<=0.4  %--> probabilidad
                    de mutaci n PM
                    if rand>=0.5
                        Clones(z,y)=PA_Ordenada
                           (z,x)+((PA_Ordenada(
                           z,x))*0.4); %-->
                           porcentaje de
                           mutaci n
                    else
                        Clones(z,y)=PA_Ordenada
                           (z,x)-((PA_Ordenada(
                           z,x))*0.4); %-->
```

```
                                porcentaje de
                                mutaci n
                        end
                    else
                        Clones(z,y)=PA_Ordenada(z,x
                            );
                    end

            end
        end
        Referencia=Referencia+N_Clones;
    end

    %FITNESS-POBLACION-CLONADA
       ---------------------------------------------


    for x=1:N_Poblacionc
        [Desempenoc(x),EEss(x),ETs(x),EMp(x)]=
           fitness(Convertidor,Vout, Clones(:,x),"
           no");
    end

    [Mejores_Desempenosc,Indices_Ordenadosc]=sort(
       Desempenoc);

    %POBLACION-CLONALG-ORDENADA
       ---------------------------------------------
    PoplacionClonalg_Ordenada=Clones(:,
       Indices_Ordenadosc);

    fprintf('Mejor desempe o = %f\n\n',
```

```
    Mejores_Desempenosc(1));
     fprintf('Kp = %f\n',
        PoplacionClonalg_Ordenada(1,1));
     fprintf('Ti = %f\n',
        PoplacionClonalg_Ordenada(2,1));
     fprintf('Td = %f\n\n',
        PoplacionClonalg_Ordenada(3,1));

     fprintf('EEss = %f\n',EEss(
        Indices_Ordenadosc(1)));
     fprintf('ETs = %f\n',ETs(Indices_Ordenadosc
        (1)));
     fprintf('EMp = %f\n\n',EMp(
        Indices_Ordenadosc(1)));
 %
    ------------------------------------------------------

for x=1:N_Mejoresc
     N_Poblacion1(:,x)=PoplacionClonalg_Ordenada
        (:,x);
 end
 Minimo=[min(N_Poblacion1(1,:)) min(N_Poblacion1
    (2,:)) min(N_Poblacion1(3,:))];
 Maximo=[max(N_Poblacion1(1,:)) max(N_Poblacion1
    (2,:)) max(N_Poblacion1(3,:))];

 for x=1:N_Poblacion-N_Mejoresc
     for y=1:Elementos
         N_Poblacion2(y,x)=0+(Maximo(y)-0).*rand
            ;
     end
 end
```

```
    Poblacion_Actual=[N_Poblacion1 N_Poblacion2];

    Mejores_DesempenosGc(Generacion,:)=[
       Mejores_Desempenosc(1)
       PoplacionClonalg_Ordenada(1,1)
       PoplacionClonalg_Ordenada(2,1)
       PoplacionClonalg_Ordenada(3,1)];
    Error_CuadraticoMedioGc(Generacion)=sqrt(sum(
       Desempenoc.^2)/N_Poblacionc);

    if Mejores_Desempenosc(1)==0
            break
    end
end

[X1 Y1]=size(Mejores_DesempenosGc);
[X2 Y2]=size(Error_CuadraticoMedioGc);

title('Evoluci n Error X Generaci n');
xlabel('Generaci n');
ylabel('Error');
hold on
plot(1:X1,Mejores_DesempenosGc(:,1),'r');
legend('Error cuadratico Mejor Clon')
grid


% FUNCION FITNESS -------

function [Fitness,EEss,ETs,EMp,Ts] = fitness(
   converter,Vout,G,grafica)

    assignin('base','Kp',G(1));
```

```
    assignin('base','Ti',G(2));
    assignin('base','Td',G(3));

    kEMp=1/3;
    kEEss=1/3;
    kETs=1/3;

    Ess_D=0.02;
    Mp_D=0.1;

    if converter=="Buck"
        Time=10e-3;
        assignin('base','Time',Time);
        Out = sim('Buck_LC.slx');

        Ts_La =  6e-04;
        Ts_D = Ts_La/2;    %%Ts deseado
    end
    if converter=="Boost"
        Time=30e-3;
        assignin('base','Time',Time);
        Out = sim('Boost_LC.slx');

        Ts_La = 0.04548;
        Ts_D = Ts_La/3;      %%Ts deseado
    end

    Vout_sim = Out.yout.getElement('Vout');
    VoutY=Vout_sim.Values.Data;
    VoutX=Vout_sim.Values.Time;

% Sobre paso maximo Sm
```

```
    [SmMax,IndexSmMax]=max(VoutY);

    SizeVout=size(VoutY);
    Vpromedio=0;
    contador=0;

    for x=SizeVout:-1:(size(VoutY)/2)
        Vpromedio=Vpromedio+VoutY(x);
        contador=contador+1;
    end

    Vpromedio=(Vpromedio/contador);

        SmX=VoutX(IndexSmMax);
        SmY=VoutY(IndexSmMax);
        Mp=abs((SmY-Vpromedio)/Vpromedio);

        if Mp<Mp_D
            EMp=0;
        elseif Mp>Mp_D
            EMp=abs((Mp_D-Mp)/Mp_D);
        end

%%----------------------------------------
    %tiempo de estableciomiento
    Voutmax=Vpromedio+(Vpromedio*0.02); % 0.02 ;
       corriente 0.15
    Voutmin=Vpromedio-(Vpromedio*0.02); % 0.02 ;
       corriente 0.15

    for x=SizeVout:-1:1
```

```
    if VoutY(x)>Voutmax||VoutY(x)<Voutmin
        IndexTs=x;
        Ts=VoutX(x);
        if Ts<Ts_D
            ETs=0;
        elseif Ts>Ts_D
            ETs=abs((Ts-Ts_D)/Ts_D);
        end
        break
    end
end


%-----------------------------
% Error de esta estacioneario

Ess=abs((Vout-Vpromedio)/Vout);

if Ess<Ess_D
    EEss=0;
elseif Ess>Ess_D
    EEss=abs((Ess_D-Ess)/Ess_D);
end
%--------------------
%funcion objetivo

Fitness=sqrt((kEMp*(EMp)^2)+(kEEss*(EEss)^2)+(
   kETs*(ETs)^2));
Fitness=Fitness*100;
```

A.2. PCB convertidor Buck-Boost Bidireccional

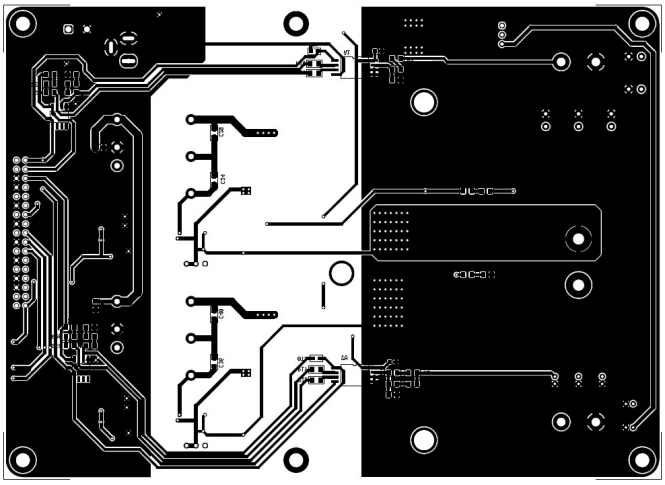

Figura A.1. Bottom PCB convertidos

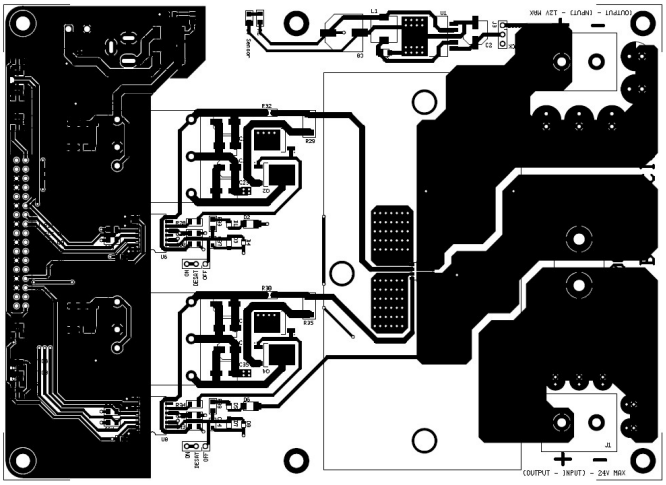

Figura A.2. Top PCB convertidos

A.3. Circuito esquemático

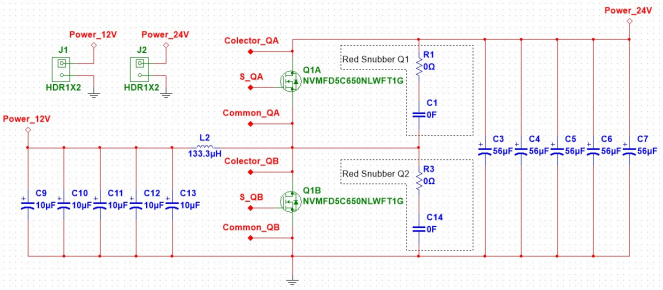

Figura A.3. Circuito convertidor Buck-Boost Bidireccional

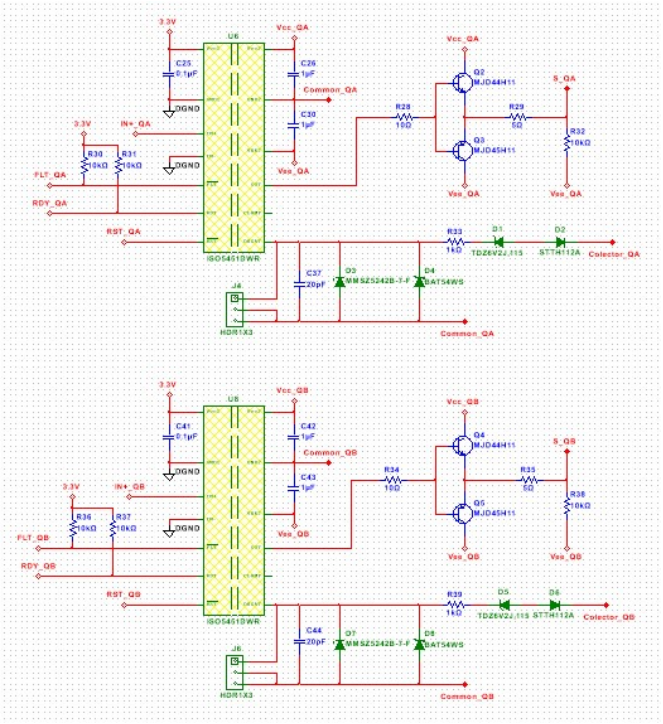

Figura A.4. Circuito driver de los MOSFETS

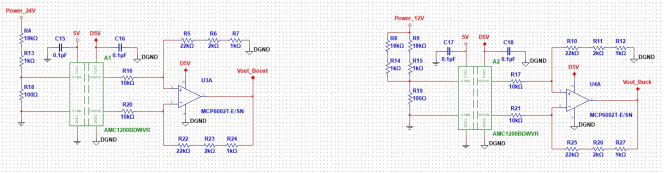

Figura A.5. Circuito Sensores de voltaje