UNIVERSIDAD SANTO TOMÁS

CONTROL DE VOLTAJE DE SALIDA DE UN CONVERTIDOR FORWARD APLICANDO REDES NEURONALES ARTIFICIALES

Realizado por

Brian David Noriega Nonsoque

Proyecto de Grado presentado en cumplimiento del requisito
para optar por el grado de Ingeniería Electrónica

Grupo de Investigación GED (Grupo de Estudio y Desarrollo en Robótica)
Facultad de Ingeniería Electrónica
División de Ingenierías

Julio de 2022

CONTROL DE VOLTAJE DE SALIDA DE UN CONVERTIDOR FORWARD APLICANDO REDES NEURONALES ARTIFICIALES

Realizado por

Brian David Noriega Nonsoque

Proyecto de Grado presentado en cumplimiento del requisito
para optar por el grado de Ingeniería Electrónica

Dirigido por

José Guillermo Guarnizo Marín

Co-Dirigido por

Jhon Fredy Bayona Navarro

GED
Facultad de Ingeniería Electrónica
División de Ingenierías

Julio de 2022

DECLARACIÓN DE AUTORÍA

Yo, Brian David Noriega Nonsoque identificado con cédula de ciudadanía numero 1.233.908.002 de nacionalidad colombiana, estudiante de la Universidad Santo Tomás del programa de ingeniería electrónica, declaro que en el trabajo titulado: **CONTROL DEL VOLTAJE DE SALIDA DE UN CONVERTIDOR FORWARD APLICANDO REDES NEURONALES ARTIFICIALES**, se ha desarrollado de manera íntegra, asimismo, se identifican y citan las fuentes originales en las que se basa la información contenida en el documento. Ninguno de los datos presentes ha sido plagiado, inventado o manipulado a conveniencia. En virtud de esta declaración, me responsabilizo del contenido, autenticidad y alcance del presente proyecto.

Bogotá, Junio de 2022

DEDICATORIA

El presente trabajo está dedicado a mi madre Nelly Nonsoque y mi padre Vladimir Noriega, quienes me han apoyado en todo momento, dándome ánimos y fortaleza para seguir adelante.
A mis hermanas Kate Alexandra y Larissa Sofia, por la confianza que han puesto en mi.
A Maria Cruz por todo el apoyo y consejos que me ha dado en toda esta travesía.

AGRADAMIENTOS

Agradezco el desarrollo de este proyecto principalmente a mi director de tesis Jose Guillermo Guarnizo Marín junto con el co-director Jhon Fredy Bayona Navarro, quienes me acompañaron y apoyaron en gran parte del desarrollo de este proyecto, haciendo posible su culminación. Agradezco a mi familia y seres queridos por el apoyo que me dieron. A mis compañeros Juan Pablo Romero y David Paez, quienes fueron gran apoyo para este trabajo, junto a diversos compañeros con los que tuve el placer de compartir diferentes experiencias, en especial a Michael Gutiérrez, y a todos los docentes de la universidad Santo Tomás con los que tuve el placer de aprender de sus enseñanzas.

Contenido

Lista de tablas

Lista de figuras

Resumen

En el siguiente trabajo de grado presenta una propuesta del diseño y desarrollo de un convertidor tipo Forward junto con un controlador, siendo el controlador usado como control principal hacia el convertidor, basado en redes neuronales artificiales. Para esto se realizó una serie de investigaciones las cuales dan como resultado un diseño electrónico detallado del convertidor Forward junto el proceso, que llevó la realización de la red neuronal para implementarlo al convertidor. En la cual su programación fue ejecutada en el software de MATLAB/SIMULINK. Para la implementación en físico se efectúan las modificaciones en el algoritmo en la DSP C2000-F28069M LaunchPad y en el convertidor Forward diseñado capaz de aportar una potencia de salida de 120W a 12V, con un voltaje de entrada de 220V.

Introducción

En el contexto del control de dispositivos electrónicos de potencia, las técnicas más usadas por los desarrolladores son las técnicas de control lineal [1], [2], y con frecuencia se usa el popular control proporcional integral derivativo (PID) [3], [4], ya que tiene un excelente rendimiento en condiciones específicas de funcionamiento y la simplicidad en el momento de su implementación. En general, estas soluciones son fiables, capaces de seguir correctamente una referencia, ya sea de tensión o de corriente.

El método de control digital para convertidores CC-CC atrae mucha atención, ya que puede realizar un control flexible, y un proceso de diseño fácil y rápido en comparación con el control analógico [5]. Sin embargo, el control digital requiere tiempo para realizar muchos cálculos; esto provoca un retraso en el proceso de control [6].

Los controladores digitales requieren de un modelamiento matemático muy preciso [7] por lo que requiere de un adecuado proceso de análisis, para eludir este planteamiento, en los últimos años ha habido un interés creciente en importar herramientas del dominio del aprendizaje automático a los controles, como optimizar el rendimiento de un controlador con búsquedas aleatorias y muchas iteraciones. Además, el propósito de usar redes neuronales para reemplazar modelos dinámicos, o reducir las demandas de otro modelo de control predictivo (MPC) [8].

En la industria de la electrónica de potencia, existe una tendencia hacia frecuencias de conmutación altas, ya que ayudan a reducir el tamaño de los componentes magnéticos en las fuentes de alimentación. El convertidor Forward es una topología prometedora debido a su simplicidad y robustez. Este convertidor tiene aplicaciones en voltajes bajos y corriente alta, no experimenta problemas de recuperación inversa [9], [10].

Las redes neuronales no son muy comunes en el control de estos dispositivos a pesar de su alta capacidad de aproximación [11], por lo que este trabajo propone una estrategia de control basada en redes neuronales artificiales, el cual actuará como único control aplicado a el convertidor

Forward, además, para comprobar su funcionamiento y efectividad, se le aplicarán diversas alteraciones, tanto al control como a el convertidor.

Capítulo 1

Planteamiento del problema

Los convertidores electrónicos de potencia CC/CC se utilizan como dispositivos, sistemas, circuitos, etc., para extraer y ajustar la potencia de las diferentes fuentes de energía y transferirla a la carga[12],así como tener la cualidad de suprimir las interferencias del sistema, tales como los ruidos y perturbaciones desconocidas en la carga que varían con el tiempo[13] además de tener la propiedad de lograr una transformación eficiente de la energía eléctrica y ser flexible en su uso[14], por esto son ampliamente usados en la industria. Pese a su recurrente uso, todavía presenta posibles problemas, como el mal dimensionamiento de los elementos del circuito, e inclusive una frecuencia de conmutación baja. Las consecuencias de esto pueden ser una respuesta transitoria lenta como la inestabilidad en estado permanente[15].

Hoy en día, gracias a la modernización, el sistema de control del convertidor a menudo sufre de cambios significativos a diferencia de la topología de los convertidores, que no han tenido cambios significativos desde la década de 1980[16]. El convertidor requiere de un controlador idóneo, ya que de lo contrario puede provocar respuestas no deseadas que afecten el desempeño del sistema [15]. Existen varios métodos de control que se pueden aplicar a los convertidores de potencia, algunos de estos son los compensadores de voltaje o corriente, histéresis, modo deslizante, y el habitual PID, que a su vez se pueden agrupar en métodos aplicables, que van desde el controlador clásico hasta los inteligentes. El controlador clásico es un método que requiere de una identificación de un modelo de la planta a controlar, que puede alterarse con el pasar del tiempo por lo que se debe realizar una re-identificación del modelo. Frente a los controladores clásicos, aparece el control inteligente. Este método en cierta medida usa el conocimiento de un experto en control de procesos sobre el objeto a controlar, además, no requiere de un modelo de referencia explícito, sin embargo, tienen que enfrentarse a la obtención de datos fuera de operación[16][17].

Los convertidores CC/CC pueden funcionar en modo de conducción continua o en modo de conducción discontinua dependiendo de la carga de resistencia, el tamaño del inductor y la frecuencia de conmutación[18]. Los diseñadores para evitar este inconveniente, seleccionan sólo un modo de conducción en el cual deberá operar todo el tiempo, esto se logra ajustando los parámetros o desarrollando sistemas insensibles al cambio de carga[19]. El ajuste estos parámetros implica que el control debe ser diseñado con excelentes propiedades dinámicas y de alta estabilidad en un rango de cargas[14], aparte la estabilidad del controlador básicamente está asegurada alrededor del pequeño vecindario del punto de equilibrio[20].

En cuanto al desarrollo de un controlador inmune al cambio de carga se requiere de la aplicación de técnicas robustas, pero estas técnicas exigen de una búsqueda exhaustiva de constantes de control y en muchas ocasiones estas resultan de un orden elevado[19]. De igual forma sucede con los controladores no lineales, como lo es el controlador de modo deslizante, el cual requiere de extensos y complejos métodos para dar con el controlador deseado[21].

Por otra parte, durante la crisis mundial producida por el COVID-19, la población se encuentra en un periodo extraordinario, lo que ha llevado a que los fabricantes y proveedores de productos de todo el mundo a tomar medidas extraordinarias. En esfuerzo por hacer su parte para contrarrestar la falta de suministros médicos que salvan vidas; los fabricantes de aviones y otros fabricantes industriales pesados están produciendo dispositivos médicos. La enfermedad COVID-19, la cual ataca los pulmones de las personas y, en algunos casos, compromete su capacidad de respirar, requiere de respiradores artificiales o ventiladores que suministran aire a los pulmones para mantener vivos a estos pacientes.

Los dispositivos médicos eléctricos requieren de una fuente de alimentación con requisitos estrictos de confiabilidad y seguridad, debido a su naturaleza de alto riesgo. Debido a esto y su alta demanda hace que haya un gran déficit de estos; ningún país posee los suficientes para afrontar este problema, y no se puede acelerar la producción por la complejidad del aparato[22][23].

Por lo que las instituciones de investigación están proponiendo prototipos de dispositivos médicos para solventar el problema de forma esencial, pero está siendo un reto el desarrollo de una fuente de alimentación que cumpla con los estrictos requisitos y ser capaces de aguantar los diferentes cambios de potencia que se requieren para mantener el correcto funcionamiento de la misma.

Dado el anterior planteamiento del problema se formula la siguiente pregunta de investigación:

¿Cómo la aplicación de un control basado en redes neuronales artificiales sobre un convertidor tipo Forward afecta a su respuesta de salida en el modo de conducción continua y discontinua?

Capítulo 2

Justificación

Durante los últimos años, el campo de la electrónica de potencia ha tenido un enorme crecimiento, principalmente en la investigación de los convertidores CC/CC conmutados controlables. Esto es debido a su amplio uso en aplicaciones eléctricas y electrónicas, que requieren de dispositivos que acondicionen un voltaje de salida[24][25]. En otras palabras, el voltaje de salida del convertidor CC/CC debe fijarse en un valor constante en estado estacionario y en estado transitorio cuando se cambia el voltaje de suministro y/o la corriente de carga.

El convertidor Forward es un convertidor CC/CC que se usa para lograr una transformación de energía eficiente y posee un aislamiento galvánico del circuito[26]. Es la topología más prometedora para conversiones de bajo voltaje y alta corriente de hasta unos pocos cientos de vatios debido a su simplicidad y robustez[9]. Sin embargo, se requiere de una estrategia de control, capaz de proporcionar buen desempeño en el estado transitorio enfocado en la velocidad de respuesta frente a un cambio extremo de carga.

Se buscan estrategias de control alternativas a las tradicionales, en conjunto con el procesamiento computacional, por lo que se han propuesto varios métodos en inteligencia computacional que sean capaces de autocorregirse, como lo son: el controlador difuso, el control robusto, el control inteligente y métodos combinados[27]. Se aplicará una red neuronal artificial como controlador ya que estas son versátiles y adaptativas a un sistema, sin necesidad de obtener su ecuación matemática, por lo que puede ser usado como apoyo de otro sistema de control o tomar el rol principal, y presenta características como la variación de los parámetros de sintonización de manera más rápida y converger a la respuesta deseada más cercana posible, por lo que puede ser un método especializado con ciertas ventajas.

Existen trabajos de control basados en redes neuronales en el estado del arte, pero ahora las redes neuronales se usan en muchas aplicaciones de procesamiento de señales. Las redes neuronales se consideran uno de los enfoques con mayor perspectiva, ya que son capaces para aproximar casi cualquier tipo de funciones como resultado del proceso de capacitación fuera de línea y en línea utilizando solo estadísticas del funcionamiento del objeto de control. Otra ventaja es que posee las redes neuronales artificiales es que son adecuadas en la interacción con otros métodos de control para eliminar las deficiencias propias del método y se utilizan para el filtrado adaptativo de señales, en la identificación de parámetros del sistema, en el diseño de observadores, así como controladores independientes, lo cual es necesario para el correcto control de los convertidores CC/CC[16]. Otra de las ventajas es que pueden usarse en modelos no lineales y proporcionan una nueva alternativa a la regresión logística que se usa para la predicción del resultado de una variable categórica[28]. Hay que considerar el mantener una buena eficiencia de conversión de energía en un amplio rango de carga, y el hecho de la variación de carga en el convertidor, hace que este opere en dos modos, el continuo y el discontinuo [29][30]. Operar en modo continuo significa que la carga del inductor nunca llegará a cero durante el ciclo de conmutación, esto ocurre gracias a la presencia de una carga, en algunos casos cuando la carga es muy baja, carga del inductor llega a 0, lo que convierte al sistema en no lineal. Para evitar esto [19], en algunos diseños se incluye una resistencia de sangrado, pero el ahorro de energía es un esfuerzo para controlar el calentamiento global (uno de los principales problemas ambientales) por lo que los dispositivos deben mejorar día a día en ahorrar energía eléctrica en las redes eléctricas, por lo tanto no se debe considerar el uso de esta resistencia.

Las ventajas que surgen a partir de la implementación de redes neuronales artificiales al sistema de control de un convertidor CC/CC son que no realizan ninguna suposición sobre la disposición de los datos, también son apropiados para la adaptación con otros métodos de control y como controladores independientes; evitan el cálculo de derivadas en las matrices jacobianas ya que se desarrollan a partir de la dinámica del sistema. Como resultado se puede mejorar la fiabilidad del controlador, así como la mejora de su respuesta en tiempo real y la calidad del voltaje de salida[31]. En comparación con un control PID, la sintonización para un sistema que requiere una alta precisión se convierte en una operación muy compleja ya que son muy sensibles al ruido, mientras que la red neuronal puede lograr una gran precisión con el entrenamiento.

Para el desarrollo de un controlador para los convertidores CC/CC hay que tener en cuenta varios factores que pueden afectar el rendimiento, como lo son: la no linealidad del circuito debido a elementos no lineales utilizados en el convertidor, las perdidas en los elementos del sistema, la reducción de interferencia electromagnética (EMI) y la estabilidad del circuito en

estados transitorios, con cambios de voltaje de entrada del convertidor y cambios de carga, estabilidad del convertidor bajo un amplio rango de control[32]. Todos estos factores se pueden obviar con el uso de un control por redes neuronales [33].

Debido a el alcance que puede presentar este convertidor controlado, se va a utilizar para alimentar un respirador artificial para el tratamiento de personas con la enfermedad COVID-19, este respirador cuenta con dos turbinas que funcionan a 60 vatios, el cual requiere de un sistema de alimentación que responda rápidamente a la variación de carga desde 0 % a 100 %, por lo que se debe observar su adaptación a los estados transitorios del sistema y la rapidez de la respuesta requerida. El respirador está en desarrollo para solventar el déficit de respiradores que hay en el mundo, ya que esta enfermedad es de alto contagio y la construcción de respiradores no puede responder a la demanda.

El propósito de este estudio es desarrollar un sistema de control mediante redes neuronales para un convertidor Forward, el cual alimente una turbina de un respirador artificial de las unidades de cuidados intensivos.

Capítulo 3

Impacto social

El control por redes neuronales para convertidores de potencia, se usan para mejorar la eficiencia de muchos sistemas. Uno de estos sistemas que puede resultar beneficiado son los motores, un motor de velocidad que impulsa una carga de bomba centrífuga que se alimenta desde un panel fotovoltaico [34], este motor debe producir la suficiente torsión para arrancar la bomba y llevarla a una velocidad de operación adecuada, si un motor carece de suficiente torsión para operar una bomba, la bomba puede no arrancar o solo puede funcionar a una velocidad reducida. Esto puede servir a un tanque hídrico que requiera un suministro de líquido constante y estable, la bomba centrífuga es actualmente la máquina más utilizada para bombear líquidos en general, por lo que la aplicabilidad de este control puede ser bien aceptada.

Otra aplicación para los motores es la regularización de la velocidad, la variación de la carga que pueda tener un motor se ve reflejada en variaciones de voltaje del convertidor, un control por redes neuronales permite la mantener la velocidad, la respuesta al cambio de carga será rápida y eficiente [35].

Por otra parte, existen aplicaciones para la gestión y estabilidad de una Micro red de corriente continua donde comúnmente los voltajes nominales son de 12V y 42V, donde las cargas que pueden tener son baterías, ultra condensadores y pilas de combustible. En la entrada del convertidor puede estar conectado a un generador de energía renovable, como lo pueden ser energía solar, eólica o marina, con altos niveles de penetración, la salida del convertidor será un bus de voltaje, que son aplicables a vehículos eléctricos, sistemas de carga de baterías y almacenamiento de energía, también se puede aplicar a sistemas de distribución en hogares, establecimientos e industrias que requieran de una línea común de corriente continua [36][37].

Los ventiladores mecánicos requieren de un suministro de energía que pueda responder rápidamente a los cambios de carga que pueda presentar en el encendido y apagado de la máquina. Ya que la disponibilidad de ventiladores se volvió en un indicador estándar para evaluar la preparación del sistema de salud contra la pandemia[38].

Capítulo 4

Objetivos

4.1. Objetivo general

Diseñar e implementar un controlador mediante el uso de redes neuronales artificiales para un convertidor Forward, que presenta variaciones abruptas en la carga, con el fin de mantener el voltaje de salida constante.

4.2. Objetivo específico

- Implementar en simulación un modelo conmutado del convertidor Forward que alimenta dos motores brushless con voltaje 12V y potencia 60W cada uno, mediante el software MATLAB.
- Determinar un algoritmo y arquitectura de control adecuado que permita mantener el voltaje de salida del convertidor, que involucre redes neuronales artificiales.
- Implementar el algoritmo de control en simulación y comprobar el comportamiento del convertidor con variaciones en la carga.
- Implementar el algoritmo de control basado en redes neuronales en un convertidor Forward que alimenta energía a dos turbinas de un ventilador.
- Comparar y validar los resultados obtenidos por el controlador propuesto frente a un controlador sintonizado por PID, para contrastar las diferencias.

Capítulo 5

Marco teórico

5.1. Topología convertidor Forward

En la figura 1 se muestra una topología típica de convertidor Forward. Esta topología se usa normalmente para potencias por debajo de $200W$ y voltajes de entrada desde $60V$ hasta $200V$. Si se usa con un voltaje de entrada mayor a $200V$, el transistor Q_1 tiene problemas para lidiar con voltajes tan altos, y si se usa con potencias mayores a $200W$ la corriente de la entrada primaria se vuelve muy alta, incluso si el voltaje de entrada está al máximo mencionado[39]. A diferencia del convertidor Push-pull, que sí puede operar con potencias más altas, el convertidor Forward es más económico en costo y tamaño.

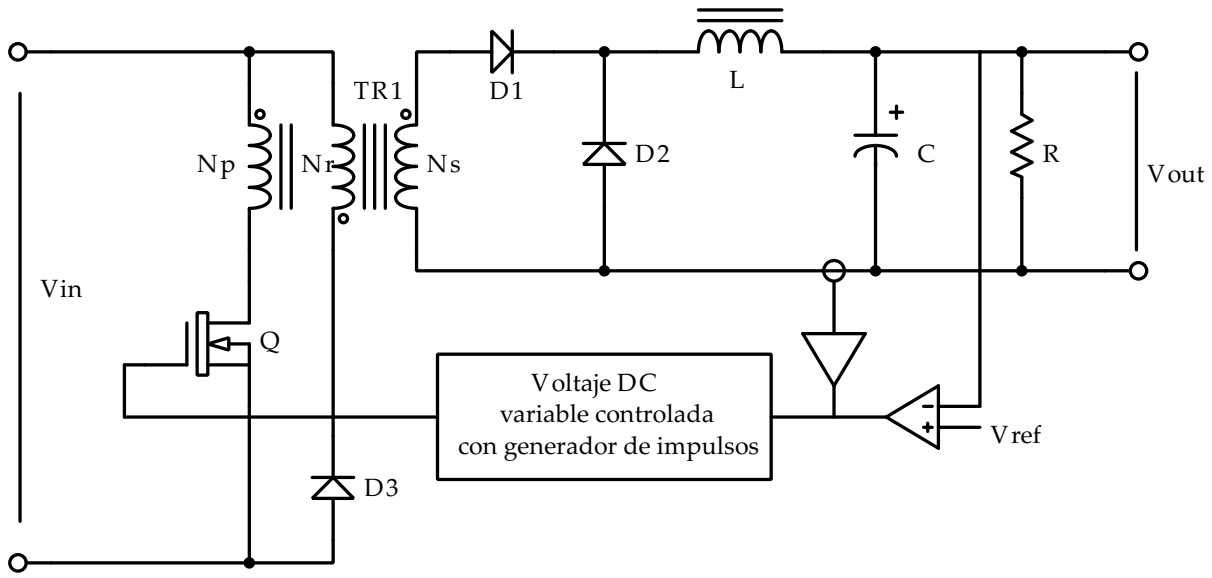


FIGURA 1: Topología basica convertidor Forward. Fuente: Autor

En la figura 2 se ven diferentes formas de onda importantes para la construcción del convertidor Forward. En 2(a) se muestra la frecuencia de conmutación que controla el transistor Q_1, en el lado secundario, cuando el Q_1 se apaga, el diodo D_2 recoge la corriente del inductor [39]. La corriente en el diodo luego desciende en rampa 2(c). La corriente del inductor es la corriente de la corriente secundaria cuando Q_1 está encendido más la corriente del diodo D_2 cuando esta apagado, la suma de esto, genera una rampa, la cual el centro de esta es igual a la corriente de salida, como se ve en 2(b). A través del diodo D_3 cruza una corriente magnetizante, la cual es muy pequeña, como se muestra en 2(d), siendo casi despreciable, lo que hace que el diodo D_3 se elija considerando solamente la frecuencia de conmutación y el voltaje de bloqueo. En 2(e) se muestra el voltaje entre Drain y Source en el Mosfet, la cual es una característica importante al momento de seleccionar un transistor capaz de soportar las condiciones requerida.

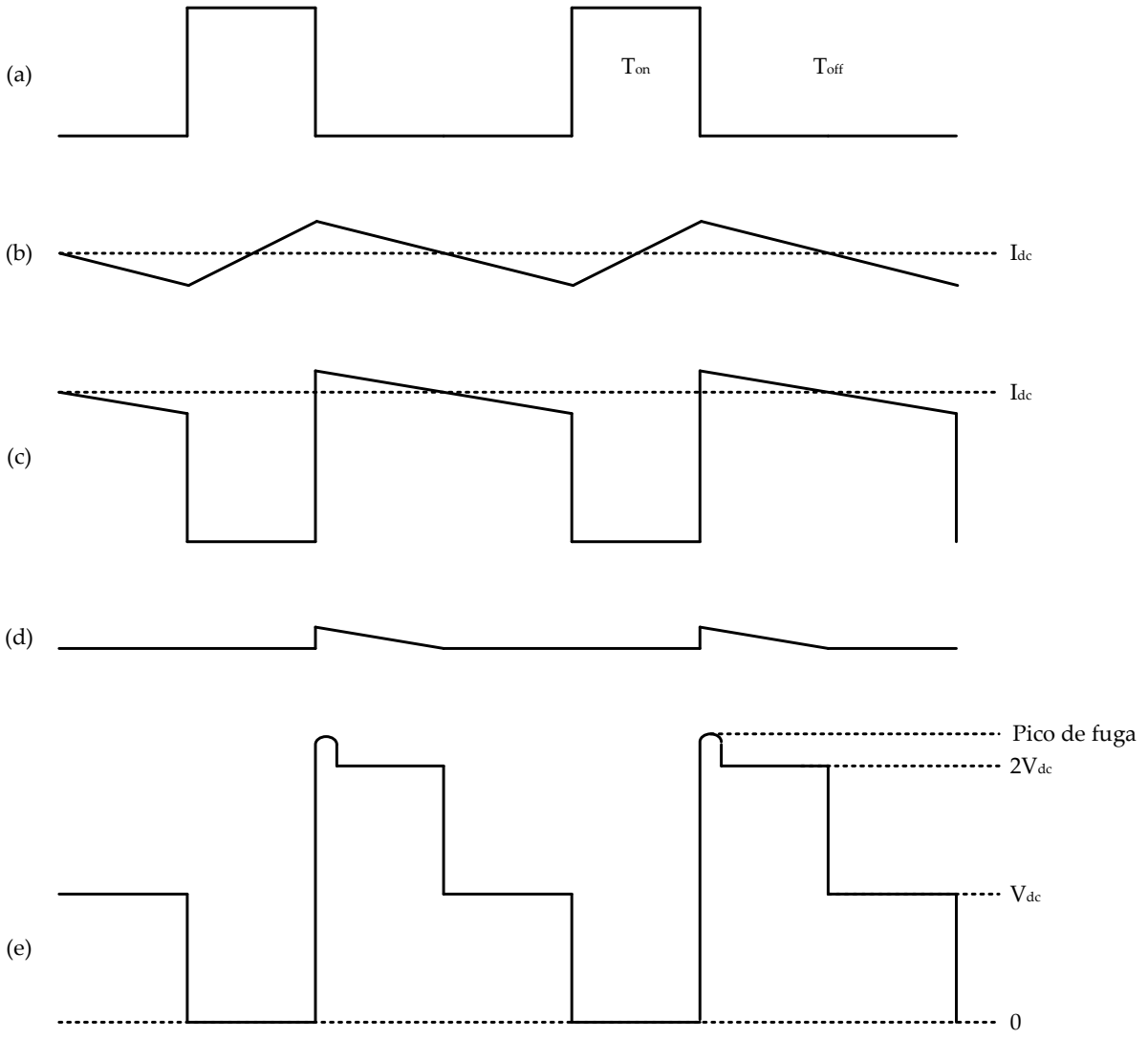


FIGURA 2: Formas de onda de corriente y voltaje Fuente: Autor

Considerando la figura 1, en transistor Q_1 está controlado por un generador de impulsos, que lo hace cambiar entre regiones de funcionamiento, de corte ("Apagado" T_{off}) a saturación ("Encendido" T_{on}). Cuando Q_1 este encendido, se energiza el bobinado N_p magnetizando el bobinado N_r y N_s. El diodo D_3 no le permitirá al bobinado N_r distribuir su energía, mientras que el bobinado N_s hace que el diodo D_1 se encienda y el diodo D_2 se apague y la energía fluya hacia el filtro LC, cargando el inductor L_1 y alimentando la carga R_1. Cuando el transistor Q_1 este apagado, la energía almacenada en el bobinado N_r fluirá a través del diodo D_3, provocando una magnetización en N_p y N_s pero de forma inversa, ayudando a que estos bobinado regresen a su posición original [40]. Como el bobinado N_s esta invertido, los diodos D_1 y D_2 cambian de

estado a apagado y encendido respectivamente, por consecuencia hace que la energía almacenada en el inductor L_1 fluya hacia el condensador C_1 y la resistencia R_1 y regresando de vuelta por D_2. Suponiendo que los voltajes de N_p y N_r son iguales y las perdidas en el diodo D_3 son despreciables, el voltaje que cae entre Drain y Source del transistor $Q1$ es de $2V_{in}$ como se ve en la figura 2(e).

Si la bobina N_r no logra regresar al bobinado N_p a su estado original, desencadenará un estado de saturación lo que hará que eventualmente el transistor Q_1 se destruya [39].

Suponiendo que la caída de voltaje para los diodos D_1 y D_2 es V_D y teniendo en cuenta el comportamiento del convertidor Forward descrito anteriormente, en un tiempo T_{on} en el ánodo del diodo D_1 hay un voltaje positivo, y durante un tiempo T_{off} es un voltaje negativo [39]. El filtro LC promedia esta forma de onda por lo que se deduce que el voltaje de salida V_{out} es como se muestra en la ecuacion 5.1.

$$V_{out} = \left[\left((V_{in} - 1)\,\frac{N_s}{N_p}\right) - V_d\right]\frac{T_{on}}{T} \tag{5.1}$$

5.1.1. Diseño del transformador de potencia

5.1.1.1. Cálculo de vueltas primarias

El número de vueltas primarias se calcula a partir de la ecuación 5.2, donde N_p es el numero de vueltas del bobinado primario, V es el voltaje del bobinado, D el ancho del pulso, T_s el tiempo de conmutación en microsegundos ΔB, diferencia de densidad de flujo en teslas, A_e es el área efectiva del núcleo en mm^2.

$$N_p = \frac{V_{in}\,D\,T_s}{A_e\,\Delta B} \tag{5.2}$$

5.1.1.2. Cálculo de vueltas secundarias

El número de vueltas secundarias se calcula a partir de la ecuación 5.1, de esta se despeja N_s , como se muestra en la ecuación 5.3.

$$N_s = \frac{V_{out}\, N_p}{V_{in}\, \frac{T_{on}}{T}} \tag{5.3}$$

5.1.1.3. Selección de tamaño del cable primario

Suponga una eficiencia del 80 % de la potencia de salida con respecto a la potencia de entrada, entonces $P_{out} = 0{,}8P_{in}$ o $P_{in} = 1{,}25P_{out}$. Todas estas corrientes secundarias se reflejan en la primaria por sus relaciones de espiras y, por lo tanto, el pulso de corriente primaria es una forma de onda de rampa en un escalón único de ancho $0{,}8T/2$. Solo hay un pulso de este tipo por período ya que se trata de un circuito de un solo transistor. El ciclo de trabajo de este pulso primario es entonces $(0{,}8T/2)/T$ o 0.4. Esta rampa en un escalón se puede aproximar mediante un pulso "flat-topped" I_{pft} cuya amplitud es el centro de la rampa. El valor promedio de esta corriente es de 0.4 I_{pft}, en la ecuación 5.4 se muestra el valor de I_{pft}.

$$P_{in} = 1{,}25\ P_{out} = V_{in}(0{,}4I_{pft})\ o\ I_{pft} = \frac{3{,}13P_{out}}{V_{out}} \tag{5.4}$$

Con la corriente primaria "flat-topped". Esta corriente fluye durante un máximo del 80 % de medio periodo, por lo que su ciclo útil máximo es de 0.4 y tomando en cuenta la ecuación, la corriente primaria RMS se muestra en la ecuación 5.5.

$$\begin{aligned} I_{rms\ (primario)} &= I_{pft}\sqrt{\frac{T_{on}}{T}} \\ I_{rms(primario)} &= \frac{3{,}125\ P_{out}}{V_{in}}\sqrt{0{,}4} = \frac{1{,}97\ P_{out}}{V_{in}} \end{aligned} \tag{5.5}$$

Si la densidad de corriente se escoge 3 amperios por milímetro cuadrado $\left(J = 3\frac{A}{\mathrm{mm}^2}\right)$ el área transversal del cable en mm^2 será calculado por la ecuación 5.6.

$$S_a = \frac{0{,}659\ P_{out}}{V_{in}} \tag{5.6}$$

5.1.1.4. Selección de tamaño del cable secundario

En la parte (b) figura 2 se ve que la corriente secundaria tiene la forma de una rampa. La amplitud en el centro de la rampa es igual a la corriente de salida DC promedio. Por lo tanto, la corriente secundaria "flat-topped" tiene una amplitud de I_{dc}, además de un ciclo de trabajo de $(0{,}8T/2)/T$ o 0.4, la corriente secundaria RMS se muestra en la ecuación 5.7.

$$I_{rms(secundario)} = I_{dc}\sqrt{0{,}4} \tag{5.7}$$

Si la densidad de corriente se escoge 3 amperios por milímetro cuadrado $\left(J = 3\frac{A}{\mathrm{mm}^2}\right)$ el área transversal del cable en mm^2 será calculado por la ecuación 5.8.

$$S_a = 0{,}211\ I_{dc} \tag{5.8}$$

5.1.1.5. Cálculo de vueltas y selección de tamaño del cable restauración

Generalmente el número de vueltas del bobinado de restauración N_r y el bobinado principal N_p son iguales, ya que el voltaje de reinicio es igual al voltaje establecido y el tiempo de reinicio es igual al tiempo establecido. Para N_r = N_p, el tiempo máximo de "encendido" de Q1 que ocurre con el voltaje de entrada de CC mínimo se elige como $0{,}8T/2$ para asegurar que el núcleo se reinicie antes del inicio del siguiente período; $Ton + Tr$ es entonces 0.8T.

El devanado de restauración solo transporta corriente magnetizante y esta corriente es tan pequeña que el bobinado de restauración frecuentemente es número 30 AWG o menor.

5.1.1.6. Diseño inductor de salida

El modo discontinuo en un inductor ocurre cuando la rampa de corriente cae a cero. Dado que la corriente de salida DC es el valor central de la rampa, el modo discontinuo se produce a una corriente mínima I_{dc}.

Calcular el mínimo de vueltas que se pueden enrollar en un núcleo para dar la inductancia requerida, está dada por la ecuación 5.9. Donde N_{min} es el número mínimo de vueltas, L es la

inductancia requerida en henrios, I_{max} es la corriente máxima en amperios y B_{max} es la máxima densidad de flujo en teslas.

$$N_{min} = \frac{L\ I_{max}\ 10^4}{A_e\ B_{max}} \tag{5.9}$$

Para evitar que el núcleo se sature en condiciones de corriente continua se requiere un Gap de aire la longitud aproximada del Gap está dada por la ecuación 5.10. Donde ℓ_g es el gap en mm, $\mu_0 = 4\pi \times 10^{-7}$, $\mu_r = 1$, N el número de vueltas.

$$\ell_g = \frac{\mu_r \mu_0 N^2 A_e 10^2}{L} \tag{5.10}$$

Determinar el número óptimo de vueltas que se enrollaran en el núcleo está dada por la ecuación 5.11.

$$N = \sqrt{\frac{L\ \ell_g}{\mu_0\ A_e}} \tag{5.11}$$

En el diseño de bobinas, los criterios para seleccionar el tamaño del cable difieren de los utilizados para los transformadores, debido a la mayor componente de corriente continua en las bobinas. Esto significa que la pérdida de cobre normalmente excede la pérdida del núcleo por un amplio margen. Dado que la corriente de ondulación suele ser bastante pequeña, los efectos superficiales y de proximidad normalmente no son un problema tan grande como lo son en los diseños de transformadores. Por lo tanto, para una pérdida de cobre mínima, se debe maximizar el tamaño del cable. Para hacer esto, el espacio de bobinado disponible debe llenarse completamente con el número requerido de vueltas usando un tamaño de alambre que llene completamente la bobina [39].

Capítulo 6

Estado del arte

Una red neuronal artificial es un sistema dinámico, compuesto por múltiples unidades de procesamiento simples, que se conectan entre sí, estas tienen la capacidad de aprender a aproximar procesos a partir de muestras. Dependiendo de la conexión entre las unidades, pueden aproximar procesos no lineales arbitrarios de sistemas complejos[41]. En los últimos años, ha habido muchas investigaciones sobre la aplicación de redes neuronales sobre el control, y de esto se han derivado varios algoritmos diferentes de redes neuronales, como lo son el control de desacoplamiento de redes neuronales PID, métodos de controla basados en redes neuronales BP y métodos de control basado en redes neuronales difusas.

El control PID es un mecanismo de control que utiliza un lazo de retroalimentación, y es ampliamente utilizado en los sistemas de control, este intenta minimizar el error entre una variable medida y una referencia. Cuando un proceso tiene múltiples lazos de realimentación de control, estos pueden alterase entre sí, por lo que es necesario el uso de un sistema que aísle las variables a controlar y los lazos de realimentación. A esto se le conoce como control de desacoplamiento. Mediante la combinación entre el control PID convencional y las redes neuronales (PIDNN) no se requiere describir un modelo matemático, y asimismo se utiliza para eliminar el acoplamiento entre los bucles. Para optimizar los pesos de conexión y evitar que durante el entrenamiento de la red neuronal caiga en un óptimo local, se combina con un algoritmo de optimización de enjambre de partículas (PSO) [42] y evolución diferencial (DE)[43].

En los procesos multivariables, se realizan investigaciones mediante el uso de controladores desacoplados en regiones de optimización, en el artículo [44], se propone un controlador PIDNN con optimización extrema de población multivariable (PEO-MPIDNN) y realizan una comparación entre este y otros métodos como el algoritmo genético codificado real (RCGA).

Una aplicación para el control de desacoplamiento PIDNN es el proceso de fermentación de enzimas biológicas marinas[45], este es un sistema variable en el tiempo, no lineal, incierto y multivariable. Gracias a el componente neuronal, se logra obtener un modelo del proceso de fermentación, luego se lleva a cabo un control de lazo cerrado obteniendo alta precisión en el control.

La red neuronal back-propagation, es un algoritmo con múltiples capas de entrada donde los datos fluyen hacia adelante, para luego propagar el error de cada capa hacia atrás. Usa un algoritmo de gradiente descendiente para minimizar el error cuadrático medio entre el valor de la salida de la red y un valor patrón [46].

Este algoritmo es altamente utilizado por su confiabilidad; ejemplos de sus aplicaciones es el uso de la monitorización de la temperatura del aire y la radiación solar [47], donde se desarrolla un modelo de red neuronal para corregir errores en los datos, entre el error de temperatura y el de radiación solar. En el artículo [48], se desarrolla un modelo de red neuronal back-propagation basado en características Meta-Paht, esto para lograr realizar enlaces entre redes heterogéneas, con el fin de lograr configurar redes de acuerdo con la información observada. Por ejemplo la recomendaciones de amigos y ubicaciones. Para una investigación geológica, se desarrolla un algoritmo capaz de predecir los parámetros geotécnicos, en base a esto destacar los parámetros más relevantes y las distribuciones de estratos y sedimentos[49].

Los convertidores de conmutación CC/CC pueden considerarse los sistemas de electrónica de potencia más usados debido a su alta eficiencia y voltaje de salida ajustable de forma flexible. Se utilizan en el suministro de energía para una variedad de componentes electrónicos con énfasis en la estabilidad del voltaje de salida, incluso con fluctuaciones en el voltaje de suministro de entrada y la carga del convertidor. Su diseño e implementación a menudo dependen de los parámetros como los voltajes y corrientes de entrada y salida, además de la topología que se desea utilizar.

Los sistemas difusos junto con las redes neuronales son sistemas dinámicos no lineales, los cuales se caracterizan por ser controladores sin modelo y se usan comúnmente para tratar problemas inciertos. Este tipo de redes posee un control mediante el aprendizaje de impedancia para un robot restringido[50], quien está sujeto a la dinámica de un sistema desconocido. Este dúo posee ventajas y desventajas como por ejemplo, los sistemas difusos utilizan patrones de pensamientos antropomórficos para extraer el conocimiento, el cual no presenta la necesidad de obtener conocimiento previo si tiene una cantidad suficiente de datos, y el razonamiento simple pero carece de capacidades de autoadaptación y autoaprendizaje, lo que conlleva un tiempo de aprendizaje demasiado largo[51].

El rendimiento del seguimiento que da esta red neuronal, está garantizado bajo una condición de incertidumbre y restricciones de estado.

Se hace alusión a la maniobra de cambio de carril, ya que sirve como ejemplo, explicando que este cambio de carril irrazonables pueden provocar colisiones graves y los consiguientes retrasos en el tráfico. La predicción de alta precisión de la intención de cambiar de carril es útil para mejorar la seguridad en la conducción. Teniendo esto en cuenta el modelo de predicción posee dos partes red neuronal difusa basada en la inferencia difusa de Takagi-Sugeno [52], la cual adopta un estimador de mínimos cuadrados (LSE) mejorado para optimizar los parámetros; mientras hay un algoritmo de aprendizaje adaptativo para actualizar las funciones de membresía y la base de reglas. Los resultados de la predicción muestran que el método propuesto es capaz de seguir con precisión los patrones de ángulo de dirección. Por último un análisis de sensibilidad indica los ángulos de rumbo y aceleración del vehículo, los cuales son factores importantes para la predicción del comportamiento del cambio de carril.

El artículo [53] nos presenta el uso de la técnica de control de predicción del ancho de impulsos el cual es comúnmente utilizado en convertidores tipo AC-DC con corrección de factor de potencia en topología boost, quienes han dado resultados satisfactorios, apartándose de los resultados que utilizaron las leyes clásicas de control. Lamentablemente esta técnica ha sido poco explorada para otros tipos de convertidores como los anteriormente mencionados. Dentro de las características que presenta este tipo de técnica es el ciclo de trabajo, el cual está predicho cuando procesa los valores promedio de las variables de entrada y el estado de convertido, utilizando únicamente un total de nueve multiplicaciones, presentando una menor complejidad en su implementación..

Por otro lado, el artículo [54] y [55] con un tema en común el convertidor de fuente de voltaje (VSC), el cual normalmente se utiliza en sistemas de corriente de alto voltaje, así como en sistemas de generación distribuida y sistemas de transmisión de alta tensión en corriente continua, donde un VSC puede ser usado como inversor o rectificador. Este tipo de sistemas permite el control independiente así como eficaz en la velocidad de control y precisión, del flujo de potencia activa o reactiva, utilizando diversas técnicas. Presentan comportamientos no lineales, múltiples entradas y múltiples salidas, permitiendo el uso de controladores no lineales para un comportamiento adecuado.

Se evidencia en el artículo [56] un trabajo que plantea soluciones de uso de controles neuronales en problemas de diversos sistemas, presentando un bosquejo general de la forma en que fueron tratados problemas en la realización del proyecto.

Lo más relevante en el artículo [57] es su descripción de redes neuronales, las cuales han sido implementadas en diferentes aplicaciones de sistemas de control automático, utilizando convertidores de tensión continua. En la cual se ha hecho énfasis en un mayor uso en convertidores tipo Buck para el control de voltaje, y un desconocimiento en el uso de la técnica para el control de corriente en un dispositivo de una carga nominal.[55]

En 2019, en el artículo "Comparision Between Clasical and Modern Methods for Optimal Control of Buck DC/DC Converter"[58], se presenta una investigación de una comparación entre los métodos de control de un convertidor reductor, realizado con la red neuronal NARMA y el regulador PI, se calcula y examina un modelo matemático con control realizado con la red neuronal NARMA y el regulador PI. Se compararon las señales de el voltaje de salida del condensador y la corriente de la fuente de alimentación y se demuestra como ambos métodos son apropiados para el control de convertidores reductor y elevador. La diferencia entre ambos métodos es mínima, tomando en cuentas el sobrepaso máximo y el tiempo de establecimiento, siendo la red neuronal la de mayor rapidez y menor sobrepaso, además de que ninguno de los dos métodos tiene un error de estado estacionario.

En 2019, en el artículo "Artificial Intelligence Aided Automated Design for Reliability of Power Electronic Systems"[59], se propone un nuevo método para el diseño de sistemas electrónicos de potencia realizada mediante el uso de inteligencia artificial. Este documento establece una relación funcional entre los parámetros de diseño y las métricas de confiabilidad, y los utiliza como base para un diseño óptimo. Además, modela el convertidor considerando los parámetros de diseño como la frecuencia de conmutación y las condiciones de operación como lo son la temperatura ambiente e irradiación, para esto se entrena una red neuronal artificial dedicada a los datos experimentales. Se entrena una segunda neurona con los resultados que arroje la primera neurona justo a las herramientas de diseño de convertidor estándar. Esto se realiza con el fin de mostrar cómo encontrar el equilibrio optimo entre la confiabilidad y el tamaño del filtro de salida en el sistema con respecto a varias restricciones de diseño.

En 2019, en el artículo "Transient Response Improvement of Repetitive-trained Neural Network Controlled DC-DC Converter with Overcompensation Suppression"[60], se estudia un método para la mejora de la respuesta transitoria de un convertidor CC/CC, mediante un control de red neuronal, se enfoca en mejorar la sobrecompensación que causa el control neuronal. Esta sobrecompensación surge por la predicción no lineal que es principalmente la predicción de un retraso de tiempo del sistema convertidor y el cálculo del control. La idea principal del estudio es buscar un método de supresión de la sobrecompensación, modificando el control por red neuronal.

Capítulo 7

Metodología

7.1. Diagrama de flujo

A continuación, se presenta un diagrama de flujo en el cual representa las actividades realizadas para el proyecto detalladamente a través del tiempo. Teniendo en cuenta los objetivos específicos como los puntos de referencia, además se tuvo en consideración la cronología de las actividades y detallando los puntos clave del desarrollo de cada una de las fases.

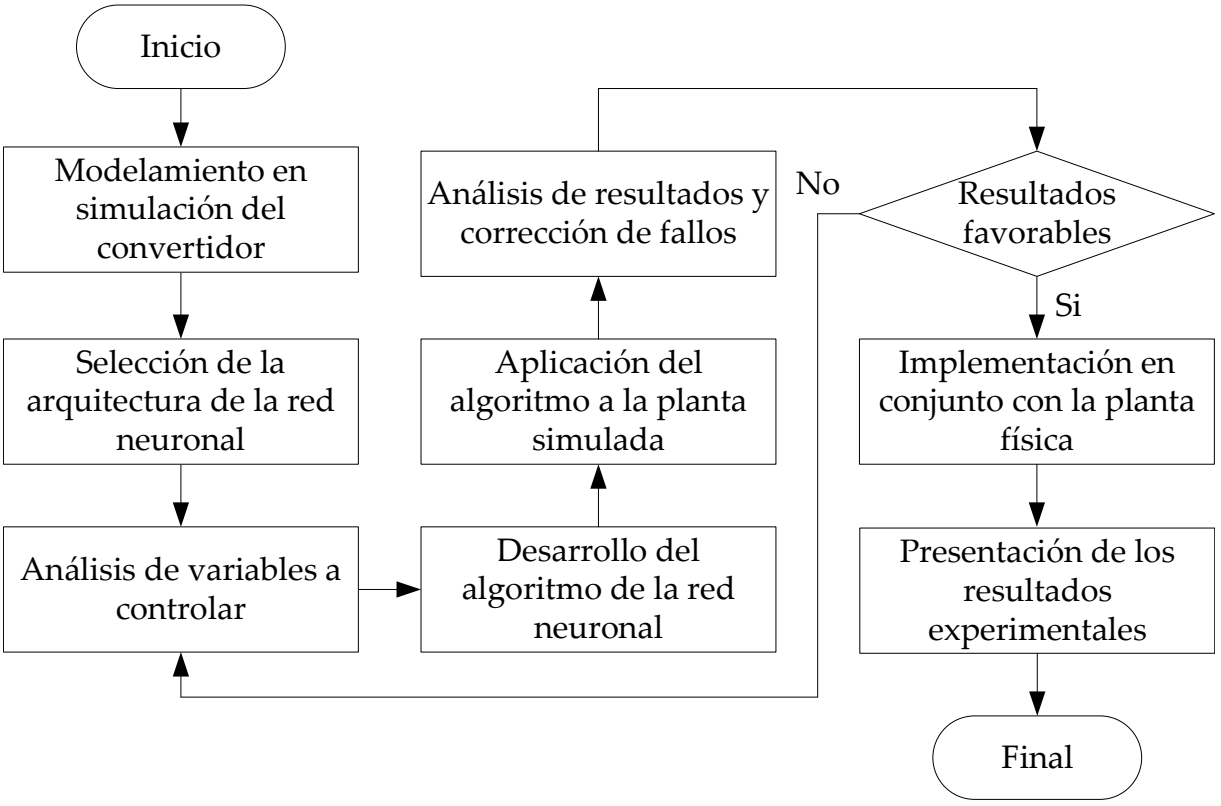


FIGURA 3: Diagrama de flujo de actividades realizadas. Fuente: Autor

7.1.1. Modelamiento en simulación del convertidor

Se elabora un modelo típico de convertidor Forward en el software MATLAB/ SIMULINK, este se hace considerando las pérdidas en los elementos, como lo son la resistencia de encendido intrínseca en los mosfet, la resistencia DC en el inductor, el voltaje de encendido de los diodos, etc. Esta simulación se realiza con el fin de mostrar los posibles resultados en las variaciones que puede llevar el modificar una variable y el comportamiento del convertidor ante los cambios.

7.1.2. Selección de la arquitectura de la red neuronal

Para el desarrollo del controlador se implementaron redes neuronales, estas tienen varias arquitecturas, métodos de aprendizaje y algoritmos, además, la red al ejecutarse y entrenarse simultáneamente no puede ser de un gran peso computacional. Se elige una arquitectura de red neuronal recurrente, esta arquitectura, las neuronas de la capa oculta tienen autoconexiones, por lo que reciben la activación de la capa anterior y su propio valor de activación anterior.

7.1.3. Análisis de variables a controlar

Se debe mirar qué variables son las más importantes a monitorear para poderlas usar como datos para la red neuronal y esta pueda analizar el convertidor de la mejor manera, la variable más típica para este caso de control sería el voltaje del condensador, y adicionalmente se puede escoger la corriente que cruza a través del inductor.

7.1.4. Desarrollo del algoritmo de la red neuronal

Al igual que el paso anterior, es importante mirar las variables pertinentes a monitorear para poderlas usar y analizar como datos para la red neuronal para el convertidor, en el desarrollo se realizó una primera versión de la red neuronal, luego se rediseñó y se mejoró la inicialización de los pesos de la velocidad de cálculo por época.

7.1.5. Aplicación del algoritmo a la planta simulada

Se realizan pruebas del algoritmo como controlador principal para el convertidor Forward simulado, esto mediante un bloque de función programable, estos resultados se muestran para recibir una aprobación de correcto funcionamiento para comenzar con la implementación física.

7.1.6. Análisis de resultados y corrección de fallos

A partir de los resultados extraídos de las anteriores pruebas, se verifican y modifican los fallos que se presentaron. Para recibir nuevamente una aprobación del correcto funcionamiento para continuar con la implementación física.

7.1.7. Implementación en conjunto con la planta física

Una vez superada la etapa de simulación se empieza con el modelamiento de la planta física, con la ayuda del software Proteus, se realiza el modelo esquemático, con todos los posibles elementos para el correcto funcionamiento del convertidor Forward, después con el mismo software se realiza el modelamiento de la PCB, siguiendo minuciosamente el esquemático anteriormente realizado. Una vez finalizado se imprime, y se prepara para soldar los componentes. Para la implementación del controlador, se usa la DSP C2000-F28069M, y con la supervisión del codirector Jhon Bayona se realizan las respectivas pruebas.

7.1.8. Presentación de los resultados experimentales

Una vez terminadas las respectivas pruebas sobre el diseño en simulación y en planta física, se concluye y presenta los resultados del proyecto.

Capítulo 8

Diseño

8.1. Control PID

Con el fin de implementar un controlador PID, debe existir un modelo matemático lineal que represente el comportamiento dinámico y estacionario del convertidor Forward. El convertidor Forward mostrado en la figura 1, para su análisis se obtienen dos circuitos equivalentes al tiempo de encendido y de apagado del MOSFET empleado como interruptor, se eligen la corriente del inductor L y el voltaje del condensador C como las variables de estado (x) como se ve en la ecuación 8.1.

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} i_L \\ v_{OUT} \end{bmatrix} \tag{8.1}$$

Durante el primer intervalo de tiempo, es cuando el MOSFET está encendido, por lo que presenta una resistencia de encendido R_{on}, al igual que una resistencia R_l en el inductor, los diodos D_1 y D_3 al encontrarse polarizados directamente, se presenta el voltaje de conducción mientras que D_2 esta polarizado inversamente, como muestra la figura 4.

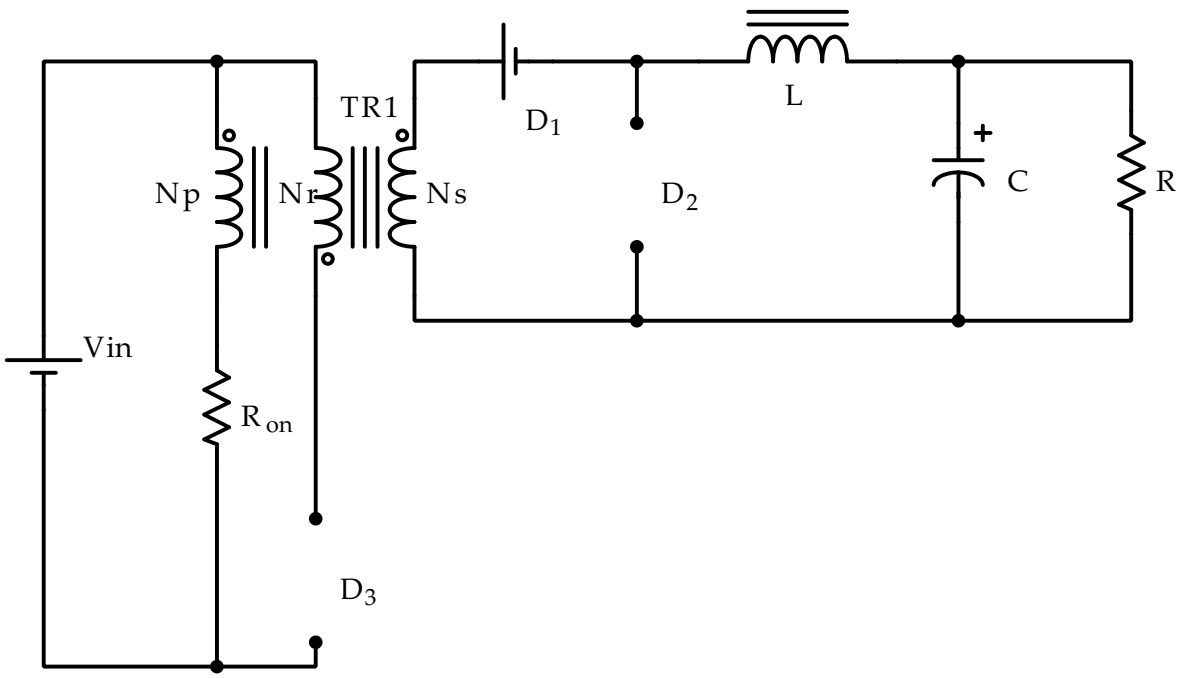


FIGURA 4: Circuito intervalo Mosfet encendido. Fuente: Autor

Se analiza el circuito de la figura 4, y se obtienen las ecuaciones 8.2 que muestran la corriente en el inductor y el voltaje en el condensador.

$$\begin{aligned} \dot{x}_1 &= \frac{1}{L}\left(\frac{N_s}{N_p}\left(V_{in} + \frac{N_s}{N_p}x_1 R_{Q1}\right)\right) - \frac{v_D + x_2}{L} \\ \dot{x}_2 &= \frac{1}{C}\left(x_1 + \frac{x_2}{R}\right) \end{aligned} \tag{8.2}$$

El circuito correspondiente al segundo intervalo de tiempo, cuando el MOSFET está apagado, es mostrado en la figura 5. Ahora no hay resistencia de encendido R_{on} y los diodos D_1 y D_3 están polarizados inversamente y el D_2 esta polarizado directamente.

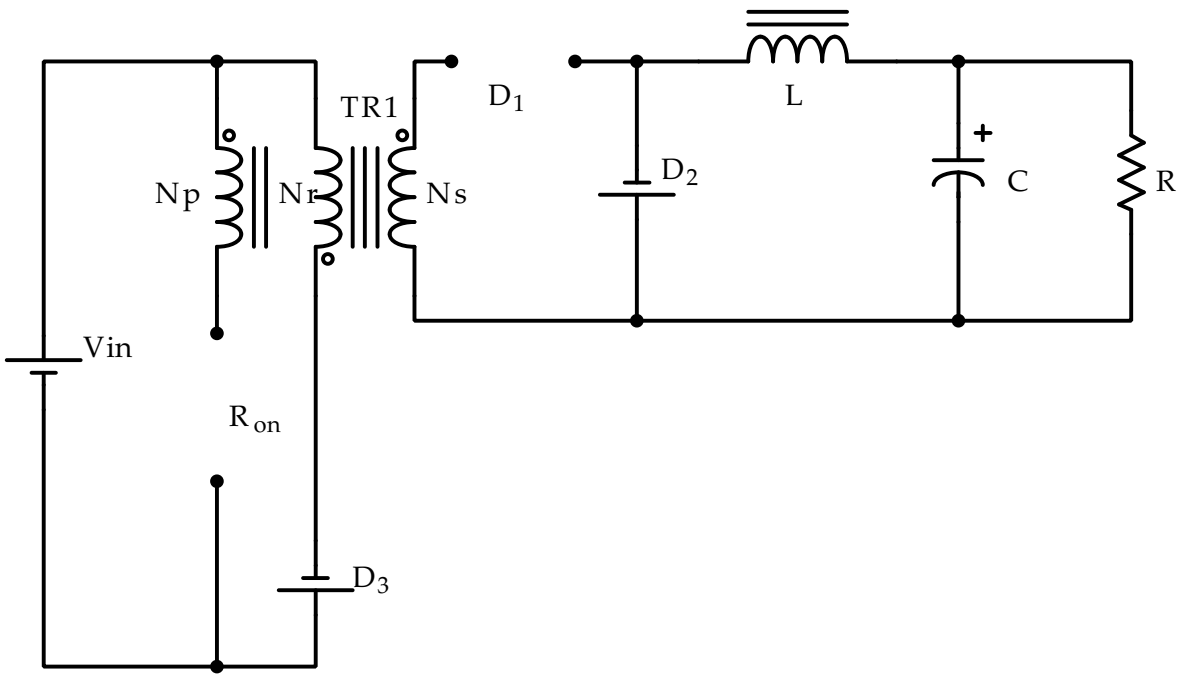


FIGURA 5: Circuito intervalo Mosfet encendido. Fuente: Autor

Se analiza el circuito de la figura 5, y se obtienen las ecuaciones 8.3 que muestran la corriente en el inductor y el voltaje en el condensador.

$$\begin{aligned} \dot{x}_1 &= -\frac{v_D + x_2}{L} \\ \dot{x}_2 &= \frac{1}{C}\left(x_1 + \frac{x_2}{R}\right) \end{aligned} \tag{8.3}$$

Se toma el promedio entre los dos intervalos de tiempo con las ecuaciones 8.2 y 8.3, por lo que da como resultado la ecuación 8.4.

$$\begin{aligned} \dot{x}_1 &= \frac{N_s\left(N_s\, x_1\, R_{Q1} + N_p\, V_{in}\right)}{L\, {N_p}^2} d - \frac{v_D + x_2}{L} \\ \dot{x}_2 &= \frac{1}{C}\left(x_1 + \frac{x_2}{R}\right) \end{aligned} \tag{8.4}$$

Con las ecuaciones 8.4, se plantea obtener un modelo lineal del convertidor Forward, la ecuación 8.5 se emplea para el análisis de espacio de estados, tomando como entrada la variación del ciclo útil, y como salida el voltaje del condensador C.

$$G(s) = C(sI - A)^{-1} B \tag{8.5}$$

El análisis produce la ecuación 8.6, que es la función de transferencia característica del convertidor Forward.

$$G(s) = \frac{b_0}{s^2 + a_1 s + a_0} \tag{8.6}$$

En la ecuación 8.7 se muestran la ampliación de los términos b_0, a_0 y a_1 de la ecuación 8.6.

$$\begin{aligned} b_0 &= \frac{N_s\, N_p\, V_{in}\, R \;-\; {N_p}^2\, R_{Q1}\, v_D}{{N_p}^2\, R\, C\, L \;-\; {N_s}^2 R_{Q1}\, C\, L\, d} \\ a_0 &= \frac{x_2}{R\, L\, C} - \frac{{N_s}^2\, R_{Q1}\, d}{{N_p}^2\, R\, L\, C} \\ a_1 &= \frac{L}{R\, L\, C} - \frac{{N_s}^2\, R_{Q1}\, d}{{N_p}^2\, L} \end{aligned} \tag{8.7}$$

Haciendo uso de esta ecuación de transferencia y la ayuda del software MATLAB, específicamente la herramienta Sisotool por el método de auto tuner PID, se calculan las constantes para la implementación del PID.

8.2. Control por red neuronal artificial

8.2.1. Descripción del algoritmo

Para este proyecto se propone un controlador compuesto únicamente por la red neuronal, el cual tiene como característica que aprende las propiedades y dinámicas de la planta mientras simultáneamente la controla, a esto se le conoce como control en línea.

Cada neurona tiene 3 componentes; el peso, que da valor a la salida dependiendo de la entrada, que para este proyecto se elige una función de activación $Tanh$, un delta que representa el error de la neurona y la salida.

El entrenamiento de la red tiene un periodo de 20μs, que coincide con el periodo del PWM y cada entrenamiento genera un nuevo ciclo útil para el convertidor y así mismo un error con respecto a la referencia.

Para la inicialización de los pesos esta debe ser adecuada para un mejor desempeño en los resultados del entrenamiento, según [61], para una función de activación $Tanh$, los pesos se inicializan con un rango aleatorio según la ecuación 8.8, donde n es el número de conexiones entrantes de la capa anterior.

$$U\left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right] \tag{8.8}$$

La estructura de la red neuronal está compuesta por tres capas: la capa de entrada, las capas ocultas y la capa de salida.

La primera parte de la estructura es la capa de entrada, en la cual se sitúan las variables que entrenan la red, para este controlador se usarán voltaje del condensador, la corriente del inductor, la referencia del voltaje, el error entre el voltaje y la referencia, también el ciclo útil resultado de una iteración anterior de la red neuronal; estas entradas se miden cada 10μs. Cada variable de la entrada de la red neuronal tiene un historial de sus valores anteriores, donde el voltaje, la corriente y el error tiene el mismo tamaño del historial; la realimentación del ciclo útil tendrá un tamaño máximo de la mitad del historial, y finalmente la referencia, que solo dispondrá de su valor actual.

Dentro de las probabilidades puede darse el caso que el sensor de corriente o de voltaje lean un valor con ruido; para solucionar esto se les aplica un filtro promediado, como el que se ve en la ecuación 8.9, con los últimos dos valores.

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n} \tag{8.9}$$

La segundo componente de la estructura de la red neuronal son las capas ocultas, quienes se compone de varias neuronas, que como se había dicho anteriormente tienen una autoconexión que hace de memoria de término corto

Finalmente, la capa de salida, consta de una sola salida, que es el ciclo útil que permite al convertidor generar un voltaje de salida que seguirá una referencia.

La red neuronal, adicionalmente tiene una arquitectura similar a la de una red neuronal recurrente, lo que significa que cada neurona de la capa oculta está alimentada por las neuronas que la preceden y por su propia salida. Al usar un control basado en redes neuronales, no es necesario el cálculo de una función matemática que describa el convertidor, sino que esta va aprendiendo a controlarla aprendiendo a través de los datos que se le suministren.

La ventaja de un control en línea es que le permite a la red neuronal adaptarse a cambios que sucedan en ejecución. Lamentablemente, dentro de sus desventajas se presenta la falta de conocimiento al inicio de su funcionamiento.

A la iniciar el controlador tendrá un valor de ciclo útil fijo en 0.04, esto con el fin de que cuando en la entrada del convertidor haya como mínimo 180V, a la salida se verá reflejado 1v, el cual será medido por el sensor de voltaje e iniciara con el entrenamiento y ejecución de la red neuronal.

El proceso de aprendizaje de la red consta de dos etapas: La primera es la propagación hacia adelante 6, la cual es donde se genera el ciclo útil como este se aplica a la planta.

Esta etapa inicia con la propagación de las entradas a través de las capas ocultas, para cada neurona para calcular su salida, se usa la ecuación 8.10 y si es una neurona de capa oculta, se deberá multiplicar el peso, por el resultado de las salidas de la capa de entrada, por su propia salida de una iteración anterior. Al resultado de la multiplicación se le aplica la función $Tanh$, y asi hasta la capa de salida.

$$\mathrm{output}_i = \frac{e^{(\mathrm{weights}_i \bullet \mathrm{output}_{i-1})} - e^{-(\mathrm{weights}_i \bullet \mathrm{output}_{i-1})}}{e^{(\mathrm{weights}_i \bullet \mathrm{output}_{i-1})} + e^{-(\mathrm{weights}_i \bullet \mathrm{output}_{i-1})}} \tag{8.10}$$

El valor obtenido se aplica al convertidor para obtener un nuevo voltaje de salida, que se utiliza para calcular la diferencia entre el voltaje deseado 8.11.

$$error = V_{ref} - V_o \tag{8.11}$$

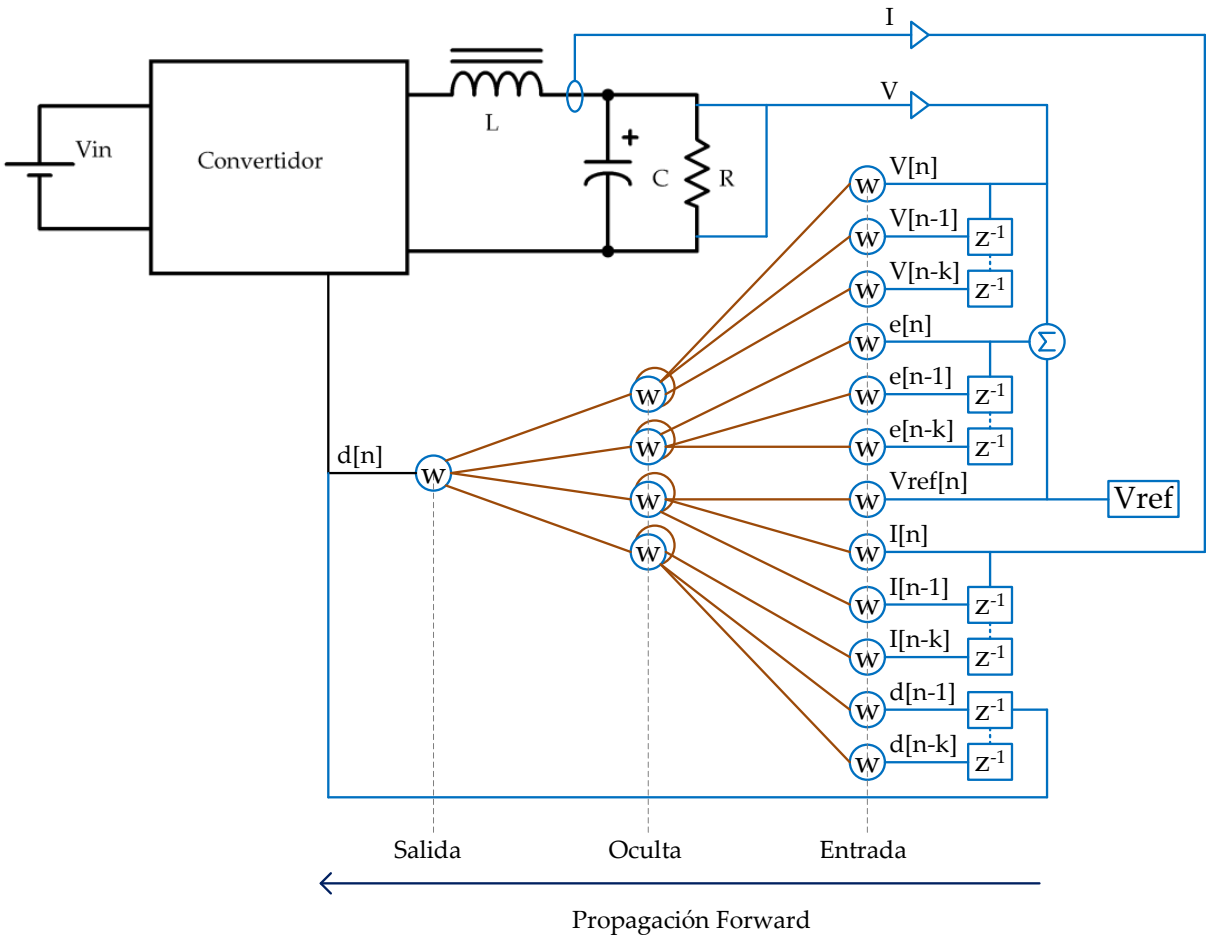


FIGURA 6: Algoritmo de propagación Forward con convertidor. Fuente: Autor

Y la segunda es la propagación hacia atrás, la figura 7 explica el proceso de entrenamiento de la red neuronal, ,a corrección de los pesos de las neuronas.

En esta etapa, el delta de error de cada neurona se calcula en base al error calculado previamente, y al delta de la capa de salida, este delta sera igual que el error entre el voltaje obtenido y la referencia 8.12.

$$delta_{end} = error \tag{8.12}$$

Para las siguientes capas, el delta se calcula a partir del delta anterior haciendo uso de la ecuaicon 8.13:

$$delta_i = \left(weights_i \cdot output_{i-1}\right) \cdot \left(weights_{i+1} \cdot delta_{i+1}\right) \tag{8.13}$$

La ecuación 8.14 se usa para actualizar los pesos de cada perceptrón en la dirección inversa, comenzando desde la capa de salida y continuando hasta la capa de entrada.

$$weights_i\ (t+1) = weights_i\ (t) + (\alpha\ delta a_i\ (t+1) + \beta(delta_i\ (t) - delta_i\ (t-1)))\ \ output_{(i-1)}\ (t) \tag{8.14}$$

Donde $alfa$ es la tasa de aprendizaje. Cuanto mayor sea la tasa de aprendizaje, más rápido será el proceso de aprendizaje, ya que los pesos se modifican más en cada iteración. Por otro lado, esta tasa puede causar oscilaciones. Para reducir esta oscilación, agregamos un término β (inercia). La inercia hace que la tasa de aprendizaje disminuya cuando la función comienza a fluctuar y aumenta cuando la función empieza a ser estable estable.

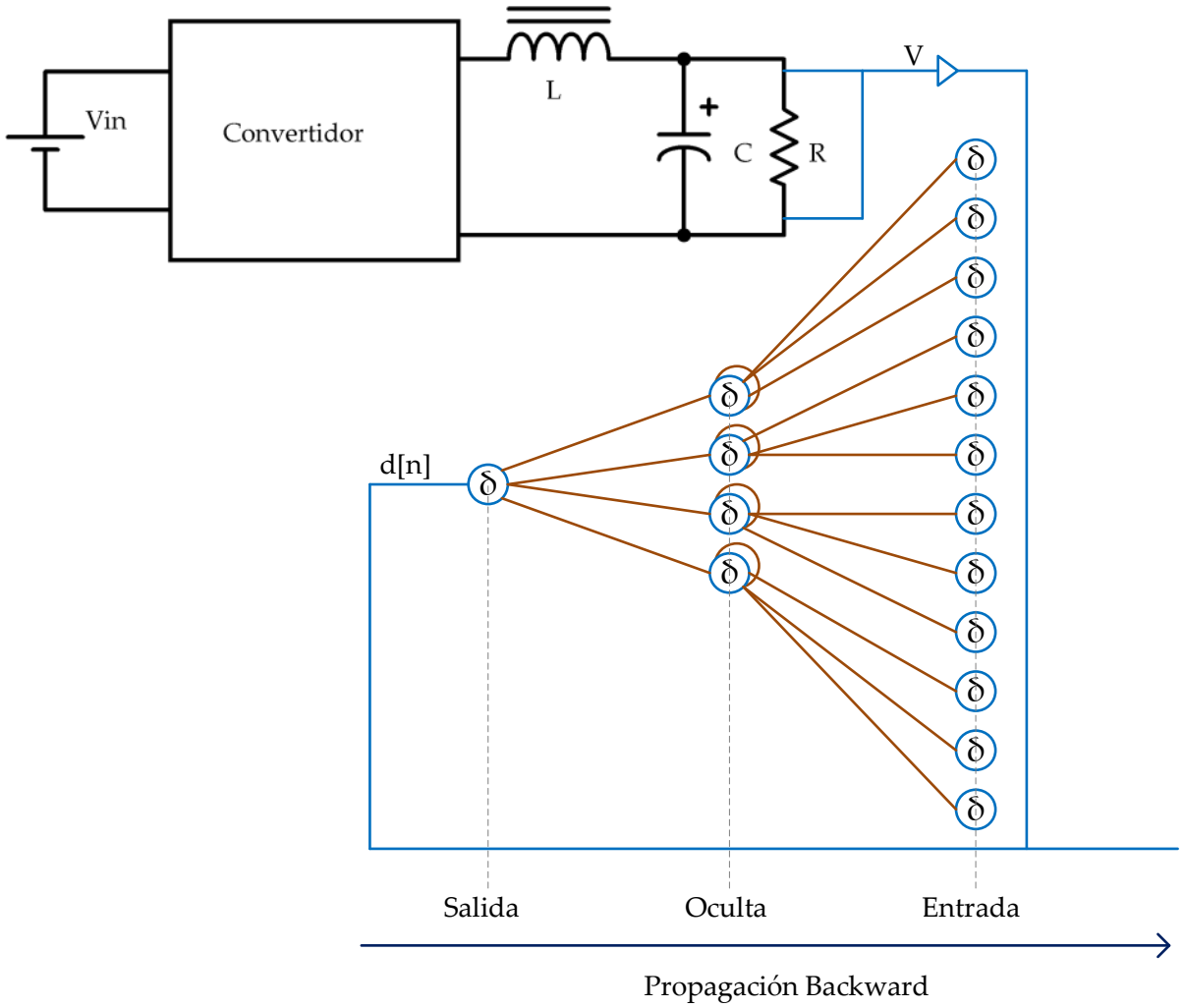


FIGURA 7: Algoritmo de propagación Backward con convertidor. Fuente: Autor

El pseudocódigo que se muestra en 1, es una síntesis de la idea principal que tiene el algoritmo de la red neuronal.

Algorithm 1 Neural Network Algorithm

Initialize Parameters: Epoch, LearnRate, InertiaRate
$weights \leftarrow random\left(-\frac{1}{\sqrt{n_{inputs}}}, \frac{1}{\sqrt{n_{inputs}}}\right)$ ▷ Connecting input to hidden
$deltas \leftarrow 0$
$outpurts \leftarrow 0$
$weights \leftarrow random\left(-\frac{1}{\sqrt{n_{hidden}}}, \frac{1}{\sqrt{n_{hidden}}}\right)$ ▷ Connecting hidden to output
$deltas \leftarrow 0$
$outpurts \leftarrow 0$
for $epoch \leftarrow 1, n$ **do**
$inputs \leftarrow [V_{out}, I_l, error, \delta]$
$activation_{input} \leftarrow activation + weights_{input} * inputs$ ▷ Forward propagation
$output_{input} \leftarrow Tanh(activation_{input})$
$activation_{hidden} \leftarrow activation_{hidden} + weights_{hidden} * output_{input} * output_{hidden_{old}}$
$output_{hidden} \leftarrow Tanh(activation_{hidden})$
$output_{hidden_{old}} \leftarrow output_{hidden}$
$error \leftarrow V_{ref} - V_{out}$
$delta_{output} \leftarrow error$ ▷ Backward propagation
$delta_{hidden} \leftarrow weights_{hidden} * outputs_{input} * weights_{outpus} * delta_{output}$
$delta_{input} \leftarrow weights_{input} * outputs_{input} * weights_{hidden} * delta_{hidden}$
$know_1 \leftarrow LearnRate * delta$
$know_2 \leftarrow InertiaRate * (delta - delta_{old})$
$weights \leftarrow weights + (know_1 + know_2) * output$
$DuttyCycle \leftarrow output_{hidden}$
end for

8.2.2. Implementación en MATLAB/SIMULINK

Por medio del software Matlab Simulink, se modela un convertidor forward, este modelo incluye al convertidor Forward y el muestreo de diferentes variables con el fin de monitorear el comportamiento del convertidor como se muestra en la figura 8.

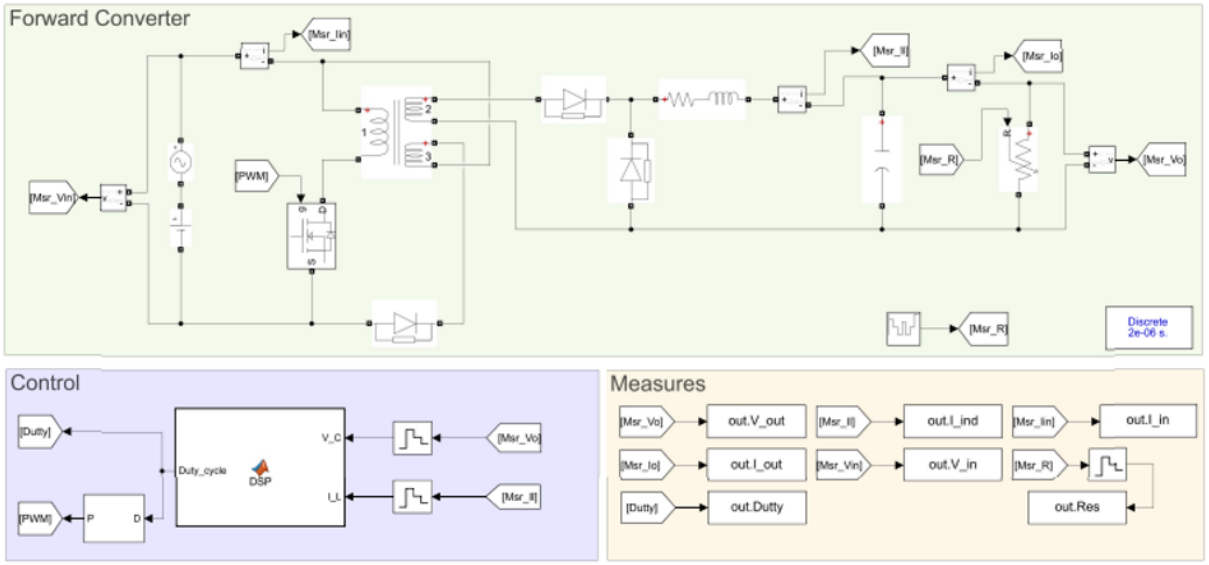


FIGURA 8: Simulación MATLAB/SIMULINK – Convertidor y Controlador. Fuente: Autor

Con el fin de comprobar la capacidad del controlador de seguir el voltaje de referencia, el convertidor se somete a cambios de carga repentinos, esta carga puede variar entre 9 diferentes valores de resistencia. El entrenamiento de la red neuronal, al ser continua, cada cambio en la carga representa un conjunto de datos útiles para el entrenamiento de esta, además estos cambios de carga están configurados para que ocurran después de que el controlador logre estabilizar el convertidor.

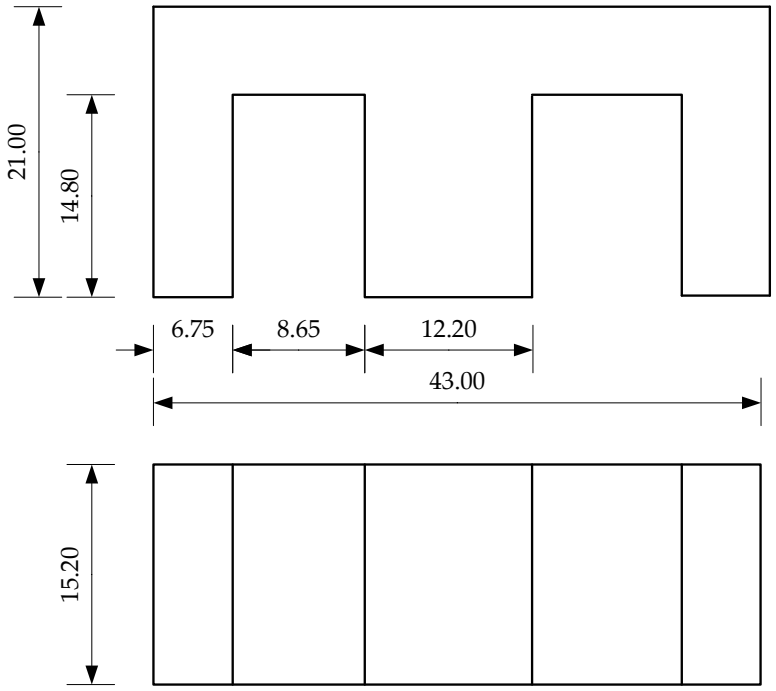
El controlador por red neuronal tiene la finalidad de mantener el voltaje de salida V_{out} en 12V independientemente de el valor de la carga actual, lo que conlleva a que el controlador aumente o disminuya el ciclo útil, para lograr un correcto seguimiento de la corriente.

8.3. Planta física

8.3.1. Elementos pasivos

8.3.1.1. Inductor y Transformador

Para la construcción del inductor y el transformador se usa un núcleo de ferrita tipo E, las características de este núcleo se ven en la figura 9.



Propiedades magnéticas		
Parámetro	**Valor**	**Unidad**
Volumen efectivo	17300	mm^2
Longitud efectiva	97	mm
Área efectiva	178	mm^3

FIGURA 9: Nucleo de ferrita E42/21/15. Fuente: Autor

Para el diseño del transformador hay de calcular el número de vueltas del bobinado primario, secundario y de restauración además de el calibre del cable adecuado para cada bobinado para que pueda tolerar el paso de corriente. Los parámetros que se toman en cuenta son los indicados en la tabla 1.

V_{DC}	P_O	I_{DC}	D
220V	120W	10A	0.3

ΔB	T_s	A_e	J
0.16T	20μs	178mm^3	3A/mm^2

TABLA 1: Parametros Transformador. Fuente: Autor

Para calcular el número de vueltas de los bobinados se usan las ecuaciones:

Primario 5.2, dando como resultado la ecuación 8.15.

$$N_p = \frac{220\ 0{,}4\ 20\mu}{184\mu\ 0{,}16} = 59{,}78 \approx 60 vueltas \tag{8.15}$$

Secundario 5.3, dando como resultado la ecuación 8.16.

$$N_s = \frac{12\ 60}{220\ 0{,}33} = 9{,}91 \approx 10 vueltas \tag{8.16}$$

El de restauración emplea el mismo número de vueltas que el primario.

Para calcular el calibre de los bobinados se usan las ecuaciones:

Primario 5.6, dando como resultado la ecuación 8.17.

$$S_a = \frac{0{,}659\ 120}{220} = 0{,}36\text{mm}^2 \tag{8.17}$$

El cable número 22 AWG es el más próximo a tener esta área transversal.

Secundario 5.8, dando como resultado la ecuación 8.18.

$$S_a = 0{,}21110 = 2{,}11\text{mm}^2 \tag{8.18}$$

El cable número 14 AWG es el más próximo a tener esta área transversal.

El bobinado de restauración debido a que maneja corrientes muy pequeñas, se escoge un calibre número 28 AWG para facilitar la manipulación de este.

Para el diseño de la bobina se toma en cuenta el I_{DC} de la tabla 1, y que tenga una inductancia de 200μH. Se debe calcular el Gap necesario entre los núcleos de ferrita con la ecuación 5.10 y el número de vueltas, esto con la ecuación 5.11, dando como resultado la ecuación 8.19:

$$N = \sqrt{\frac{200\mu\ 2{,}5m}{4\pi \times 10^{-7}\ 180\mu}} = 47 vueltas \tag{8.19}$$

8.3.1.2. Carga resistiva

Para diseñar la carga resistiva se debe tener en cuenta que debe soportar una potencia aproximada a los $120W$, con un voltaje de $12V$. Aplicando la ley de ohm, se puede determinar que la resistencia debe ser de 1.2Ω.

Para ver el comportamiento del convertidor al cambio de carga, se decide construir un banco de resistencias. Este se realiza por etapas que permitirán cambiar el valor de la resistencia de carga mediante el accionamiento de interruptores, cada etapa tiene 3 resistencias de 33Ω a $10W$ en paralelo. En la tabla 2 se muestran las variaciones de resistencia y potencia por cada etapa proporcionada por el banco de resistencias a un voltaje de $12V$.

Etapa	Resistencia[Ω]	Potencia[W]
1	11	13.1
2	5.5	26.2
3	3.7	38.9
4	2.8	51.4
5	2.2	65.5
6	1.8	80
7	1.6	90
8	1.4	102.9
9	1.2	120

TABLA 2: Etapas banco de resistencias. Fuente: Autor

Este banco de resistencias genera una alta temperatura, por lo cual, para lograr una disipación de calor, se rodean de una caja metálica que hace la función de disipador de calor y ventiladores que generan un flujo de aire. En la figura 10 se demuestra el resultado del banco de resistencias.

FIGURA 10: Banco de resistencias de 1.2Ω a 11Ω. Fuente: Autor

8.3.2. Circuito esquemático

Para la construcción del convertidor Forward, se inicia creando el diagrama eléctrico, el cual ayudará a la creación del circuito impreso. En este se toman en cuenta todos los componentes que serán utilizados para el convertidor Forward.

Este diagrama se divide en cuatro secciones: El convertidor Forward, el driver del Mosfet, las fuentes de alimentación y la adquisición de señales.

En la figura 11 se muestra el convertidor Forward que será implementado físicamente.

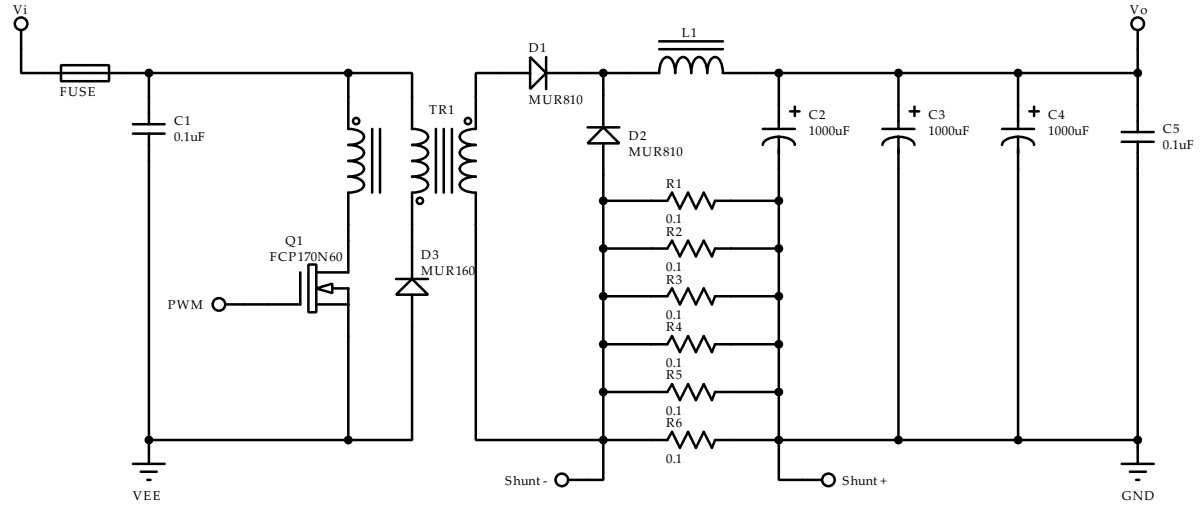


FIGURA 11: Diagrama convertidor Forward. Fuente: Autor

El controlador del Mosfet se muestra en la figura 12, se usa el Optoacoplador HCPL-3120, para la conmutación del led, se le aplica un voltaje de encendido de 3.3V y para el apagado se le aplica un voltaje de -1.7V, esto con la ayuda el transistor Mosfet 2N700. En el lado del IGBT se acompaña con una resistencia de 10 Ω para lograr una salida de corriente de 1.5A, esto con el fin de lograr una mayor pendiente en la señal de PWM, además de una resistencia de 10k para evitar disparos falsos.

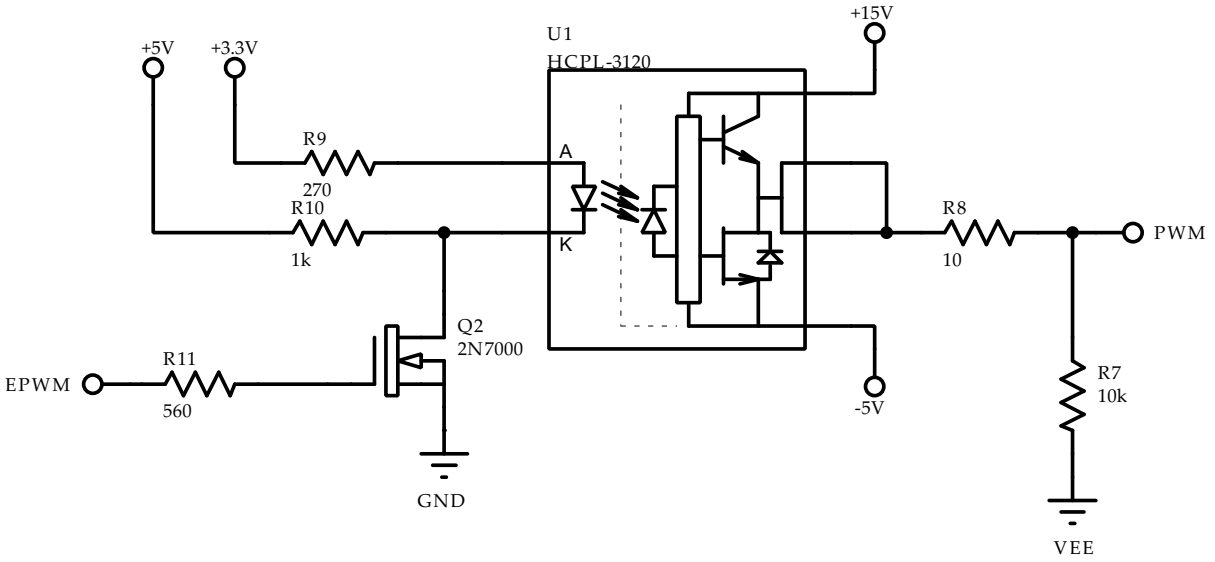


FIGURA 12: Diagrama Controlador Mosfet. Fuente: Autor

Para realizar la obtención de las señales deseadas, las cuales son el voltaje del condensador C2 y el voltaje de la resistencia R1, se usa el amplificador operacional TLV2374. Para muestrear el voltaje del condensador C2 se toma en cuenta que el voltaje máximo a registrar es de 18V, por lo que el amplificador operacional se diseña en modo diferencial y se reduce el voltaje a un 18 %. Para muestrear el voltaje de la resistencia R1, el voltaje máximo a medir es de 0.25V, se configura el amplificador operacional en modo diferencial y se amplifica el voltaje al 1300 %, la configuración utiliza se muestra en la figura 13.

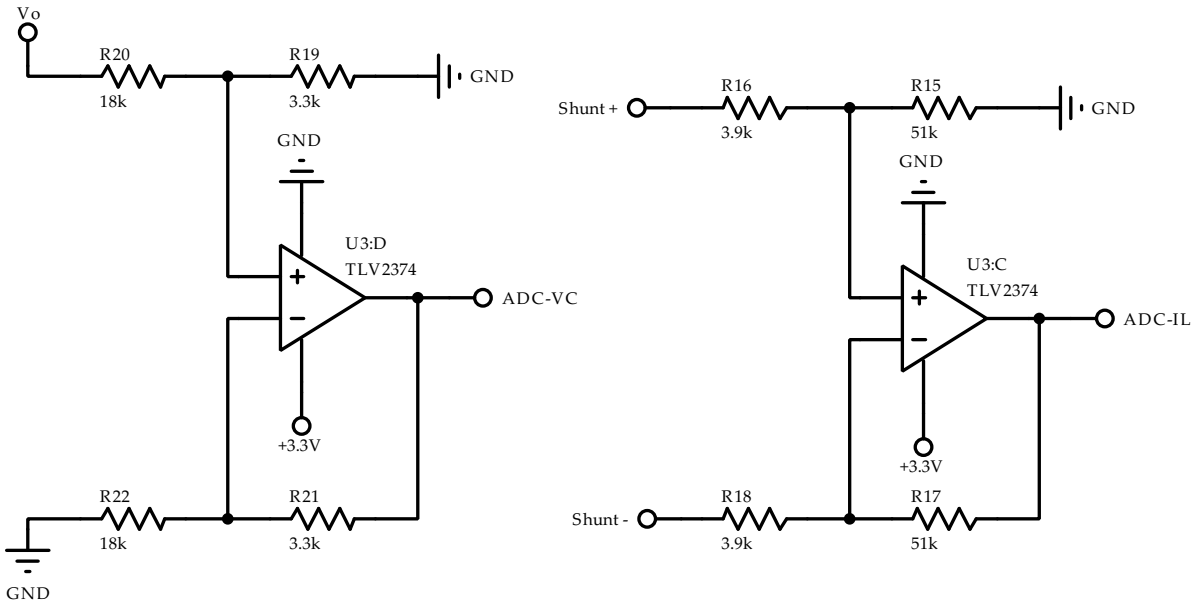


FIGURA 13: Diagrama Aplificador Operacional. Fuente: Autor

Las fuentes de alimentación se muestran en la figura 14, se usan fuentes pequeñas de 5V para la alimentación de los circuitos de control. El circuito controlador del Mosfet se alimenta con voltajes de -5V y 15V, esto para asegurar que el Mosfet asegure los estados de corte y saturación. El amplificador operacional se alimenta con 3.3V por lo que la fuente de 5V se transforma a 3.3V con un regulador de voltaje LM1117. Adicionalmente, se adicionan leds para indicar la correcta conexión entre las fuentes y el circuito.

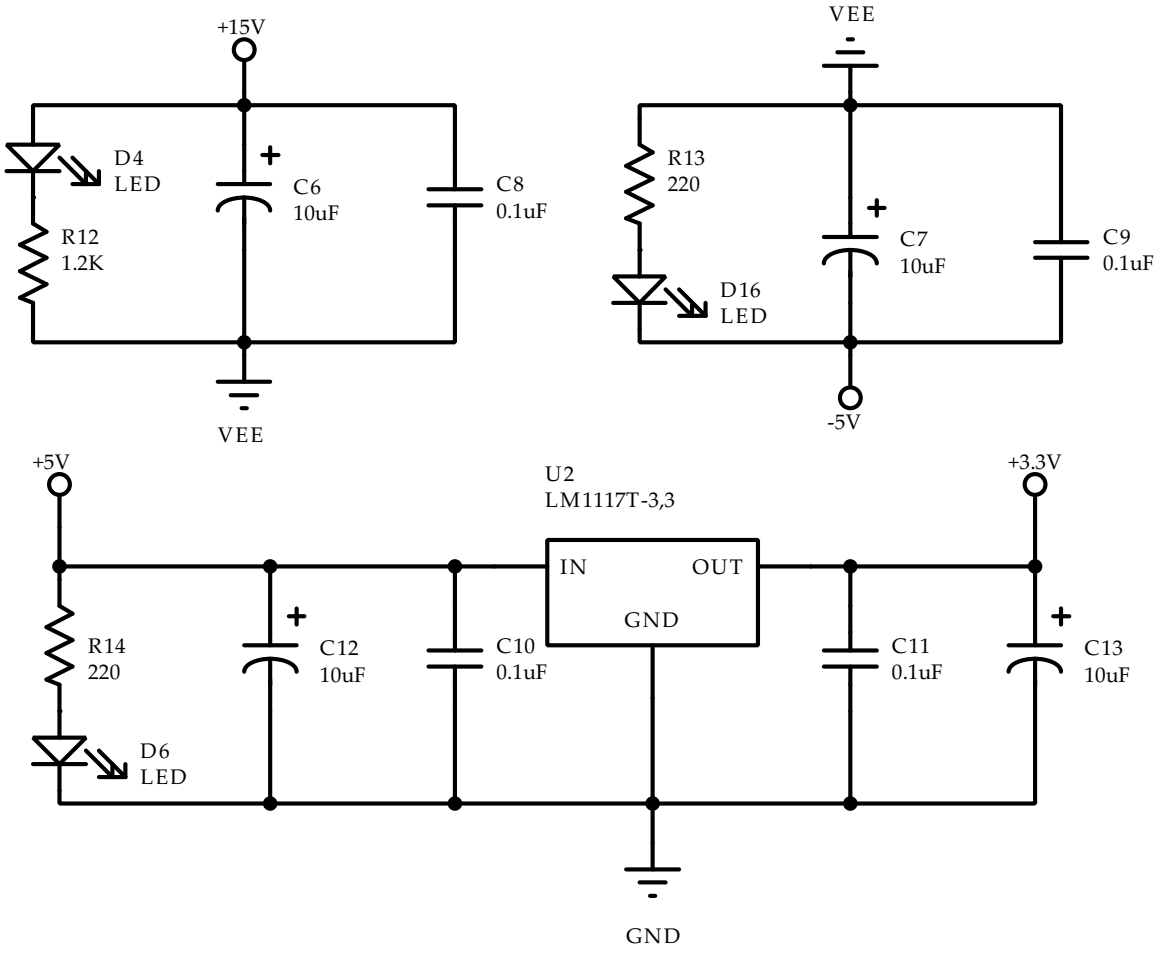


FIGURA 14: Diagrama fuentes de voltaje. Fuente: Autor

8.3.3. PCB

8.3.3.1. Listado de materiales

En la tabla 3 se listan todos los elementos usados para la implementación del convertidor, algunas características importantes sobre estos, y la cantidad de cada elemento, exceptuando la resistencia de carga, la bobina y el transformador.

Elemento	Referencia	Caracteristicas	Cantidad	Fabricante
Transistor	FCP170N60	$R_{DS_{(ON)}}$: 150 mΩ V_{DSS}: 600V V_{GSS}: ±20V I_D: 14A	1	On Semiconductor
Transistor	2N7000	$R_{DS_{(ON)}}$: 1.8 Ω V_{DSS}: 60V V_{GSS}: ±20V I_D: 200mA	1	Onsemi
Diodo	MUR810G	V_{RRM}: 100V $I_{F(AV)}$: 8A v_F: 0.895V	2	On Semiconductor
Diodo	MUR160G	V_{RRM}: 600V $I_{F(AV)}$: 1A v_F: 0.710V	1	On Semiconductor
Gate Drive	HCPL-3120	I_O: 2.5A	1	AVAGO Tech
OPAMP	TLV2374	BW: 3 MHz	1	Texas Instruments
Regulador	LM1117	V_{OUT}: 3.3V	1	Texas Instruments
Condensador	1000μF	V: 25V	3	Standard
Condensador	0.1μF	V: 250V	1	Standard
Condensador	0.1μF	V: 100V	1	Standard
Condensador	0.1μF	V: 50V	4	Standard
Condensador	10μF	V: 25V	4	Standard
Resistencia	0.1Ω	P: 1W	6	Standard
Resistencia	3.9kΩ	P: 0.25W	2	Standard
Resistencia	51kΩ	P: 0.25W	2	Standard
Resistencia	3.3kΩ	P: 0.25W	2	Standard
Resistencia	18kΩ	P: 0.25W	2	Standard
Resistencia	10kΩ	P: 0.25W	1	Standard
Resistencia	1kΩ	P: 0.25W	2	Standard
Resistencia	220Ω	P: 0.25W	4	Standard
Resistencia	10Ω	P: 0.25W	1	Standard
Fusible	-	I: 1A	1	Standard
Led	-	Color: Rojo	3	Standard

TABLA 3: Listado de materiales. Fuente: Autor

8.3.3.2. Montaje real

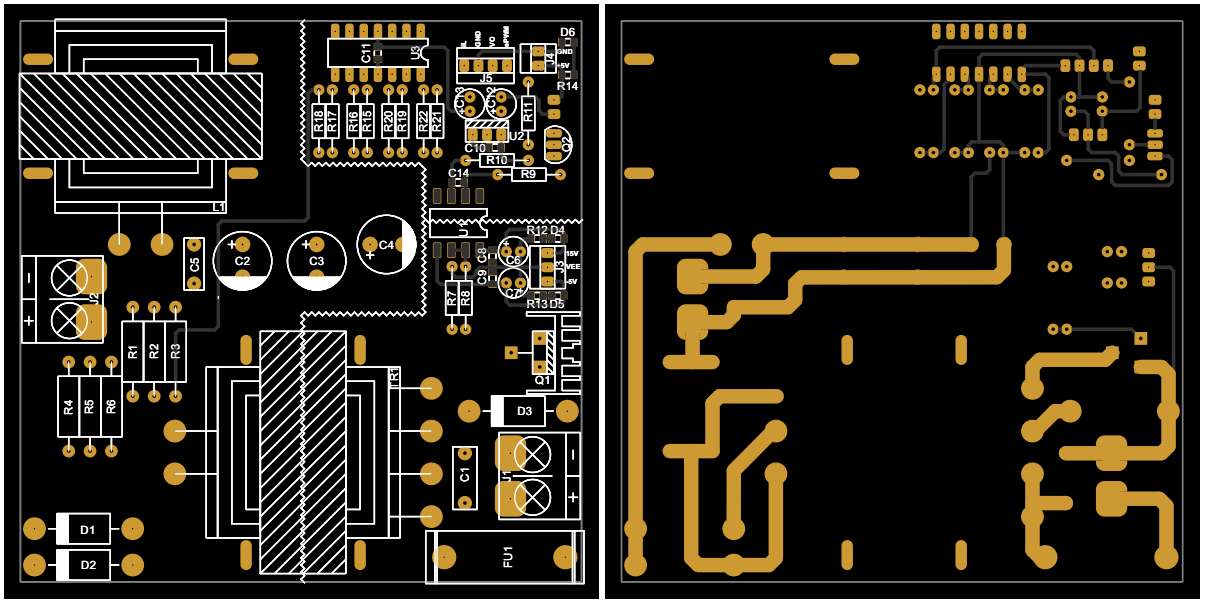


FIGURA 15: Visor Gerver, capa superior, capa inferior . Fuente: Autor

En la figura 15 se observa la versión final del diseño de la capa superior e inferior del circuito impreso, las medidas de la PCB son de 100mm por 100mm. A la PCB se le conecta la DSP C2000-F28069M, la cual cumple la función de implementar el sistema de control. También se conecta el banco de carga, las fuentes de alimentación y el circuito que le proporcionara el voltaje necesario de entrada, en la figura 16 se muestra el circuito ensamblado.

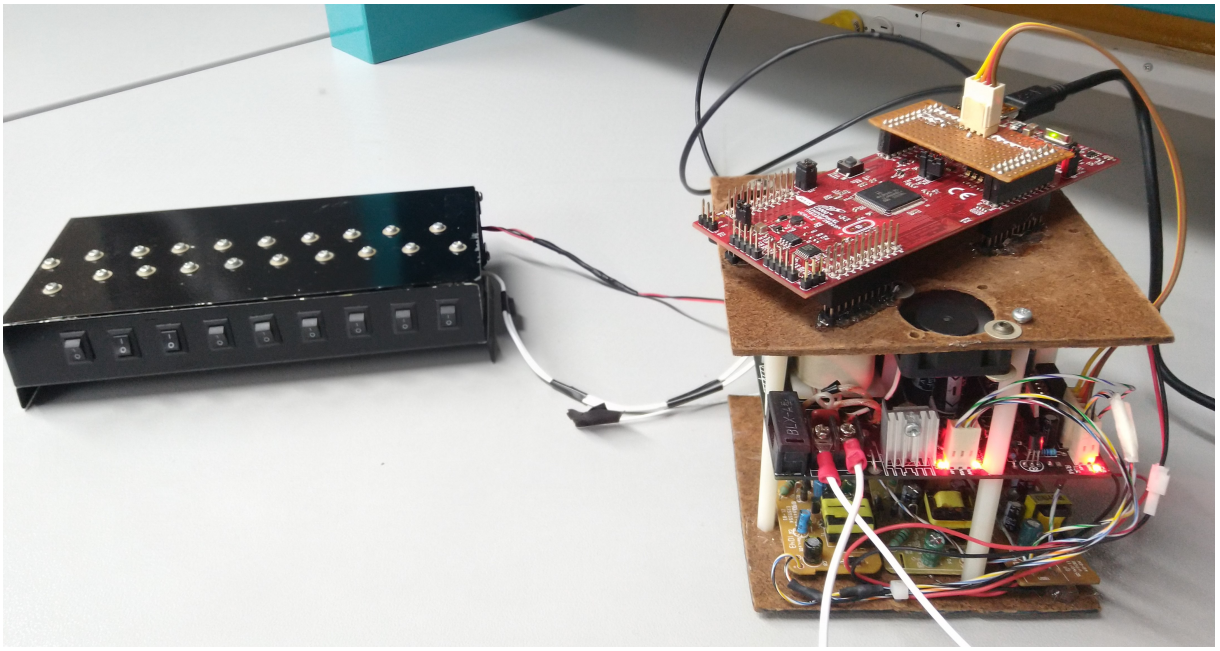


FIGURA 16: Implementacion real de la planta y el sistema de control. Fuente: Autor

En la figura 16 al lado derecho se ve el convertidor Forward, este modelo consta de 3 pisos, siendo el primero el inferior este se compone de 5 fuentes de voltajes de 5 voltios que alimentan los elementos de control, el segundo piso, el central, es la pcd diseñada, el convertidor Forward en sí, y el piso superior consta de los ventiladores para refrigerar los diodos y el mosfet de potencia, también de el DSP C2000-F28069M. Desde el lado izquierdo se ve el banco de resistencias conectado a la salida del convertidor.

Capítulo 9

Resultados

9.1. Simulaciones

9.1.1. Controlador sintonizado por métodos comunes

A partir de la ecuación de transferencia 8.7 y tomando en cuenta los valores de la tabla 3, se obtiene la ecuación 9.1.

$$\frac{61{,}19 \times 10^6}{s^2 + 269{,}44s + 1{,}664 \times 10^6} \tag{9.1}$$

Con la función de transferencia 9.1, se calcula las constantes proporcional, integral y derivativa, con la ayuda de MATLAB, Sisotool. Dando como resultado $K_p = 0{,}3361$, $K_i = 437{,}3178$, $K_d = 89{,}509 \times 10^{-6}$, $N = 4{,}3 \times 10^6$

Se aplican estos valores a al convertidor Forward en simulación. Los resultados se muestran en la figura 17.

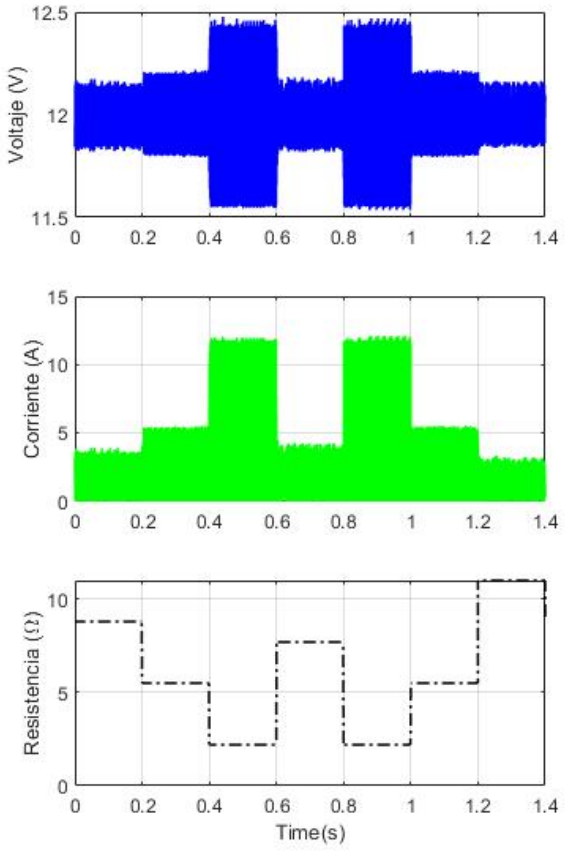


FIGURA 17: Simulación PID: Voltaje, corriente y cambio de carga. Fuente: Autor

Al aplicar este controlador en simulación muestra un alto componente de oscilación, y aparte logra seguir el voltaje de referencia, pero las oscilaciones muestran un error de ± 0.5V.

9.1.2. Convertidor controlado por red neuronal artificial

El controlador por redes neuronales, para este proyecto, no hace uso de otras topologías de convertidores como el PID, y además, este controlador se sintoniza continuamente dependiendo su entrenamiento de los cambios en el voltaje de salida que se puedan presentar, ya sea por un cambio en la carga, una variación en el voltaje de entrada o demás factores.

Se realizan varias simulaciones, variando el número de neuronas que hay en la capa oculta, que toma valores desde 4, sumando 4 neuronas por simulación hasta sumar 40, también se varía el valor de la taza de aprendizaje tomando valores desde 50μ hasta 950μ sumando 100μ por

simulación, y finalmente con variaciones en la taza de inercia, tomando valores desde 0 con incrementos de 0.15 hasta 1.35. Esto da un total de 1000 simulaciones diferentes.

Entre estas simulaciones se escojen 3 diferentes, la que mejor error cuadrático tubo durante su entrenamiento, y adicionalmente 2 escogidas de forma aleatoria. Las propiedades de estas simulaciones escogidas se ven en la tabla 4.

Parámetro	Valor	Valor	Valor
Simulación	1	2	3
Limites de inicialización	[-0.288, 0.288]	[-0.189, 0.189]	[-0.158, 0.158]
Taza de aprendizaje (l)	50×10^{-6}	150×10^{-6}	250×10^{-6}
Taza de inercia (m)	0.3	0.6	1.5
numero de neuronas ocultas	12	28	40

TABLA 4: Parámetros algoritmo. Fuente: Autor

Los resultados de la primera simulación se ven en la figura 18

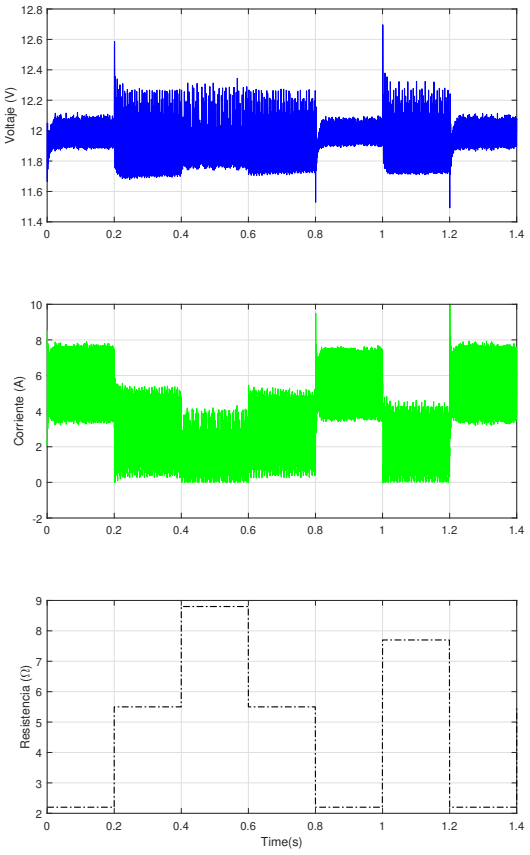


FIGURA 18: Simulación 1: Voltaje, corriente y cambio de carga. Fuente: Autor

Los resultados de la primera simulación se ven en la figura 19.

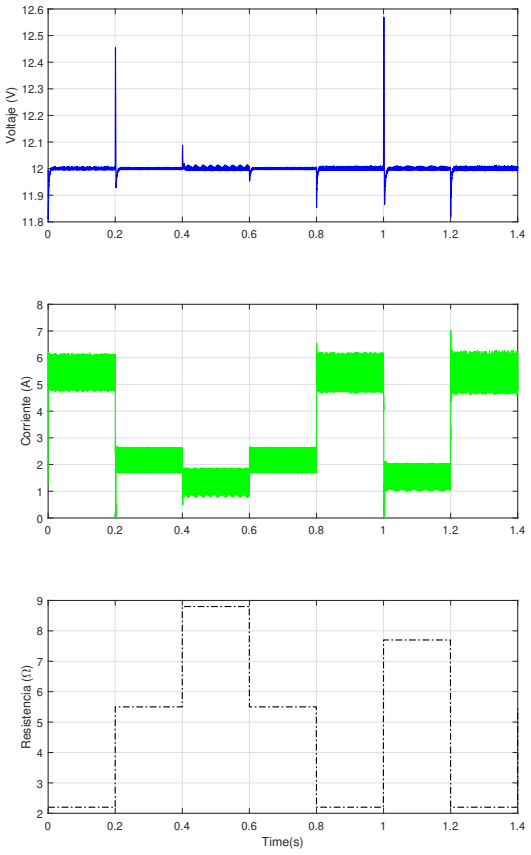


FIGURA 19: Simulación 2: Voltaje, corriente y cambio de carga. Fuente: Autor

Los resultados de la primera simulación se ven en la figura 20

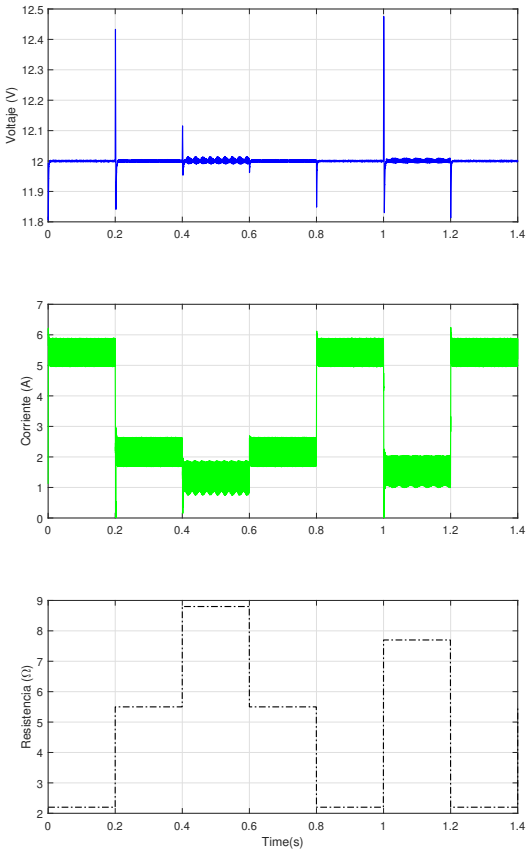


FIGURA 20: Simulación 3: Voltaje, corriente y cambio de carga. Fuente: Autor

Los resultados anteriores se ve el voltaje de salida, la corriente de el inductor y la carga, y como se observa, los tres son capases de seguir el voltaje, pero presentan algunos presentan demasiada oscilación, y esto es lo que se busca evitar.

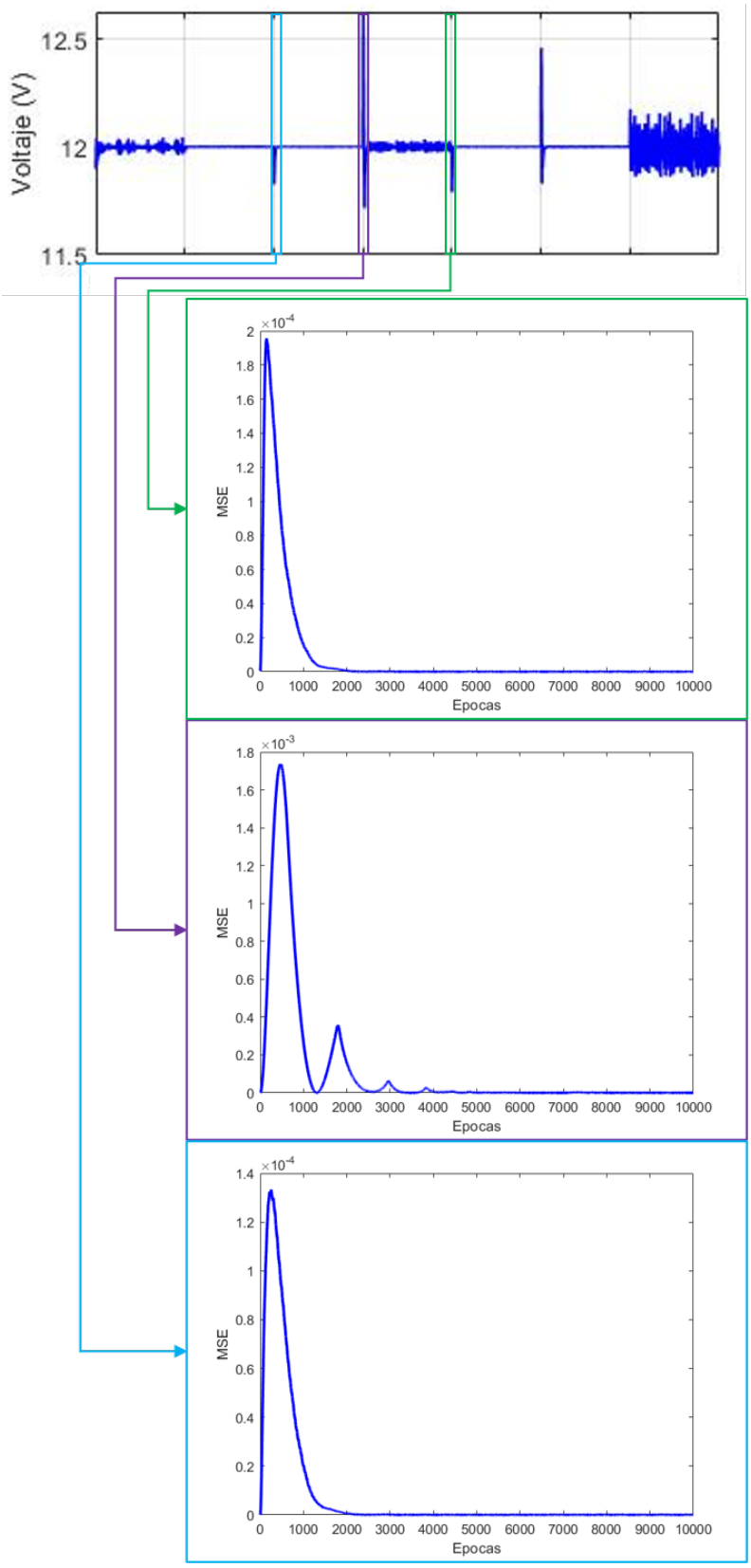


FIGURA 21: Error cuadrático medio vs Épocas en simulación. Fuente: Autor

Para demostrar el entrenamiento que realiza la red neuronal en cada época de entrenamiento, se toma como ejemplo la simulación de la figura 20, de la cual se extraen 3 instantes de tiempo donde cambia la carga, surgiendo la figura 21. Esta figura muestra como del cambio de carga surge una perturbación en el entrenamiento de la red neuronal, y como esta es capaz de reducir el error cuadrático medio y regulando nuevamente el voltaje de salida en aproximadamente 1000 épocas de entrenamiento.

9.2. Resultados de la planta física

Se implementa los controladores en la tarjeta C2000-F28069M de Texas Instruments, se procede a incluir los valores del controlador PID obtenidos por el método de sintonización auto tuner PID del software de MATLAB, Sisotool y también se incluyen los valores de entrenamiento del controlador por redes neuronales. Para la lectura de datos se hace uso de un osciloscopio Keysight technologies U1620A, con este se hacen las diferentes mediciones de voltaje y corriente del convertidor.

De la planta física se obtienen varias señales que se consideraron pertinente mostrar. Una de estas señales, es el voltaje que hay en el Mosfet Q_1 11, en la figura 22, se pueden observar dos señales, la inferior es el voltaje que hay entre Gate y Source (V_{gs}), este es un PWM al 10 % de entre -5V a 15V que controla el Mosfet, la superior es el voltaje que hay entre Drain y Source (V_{ds}),

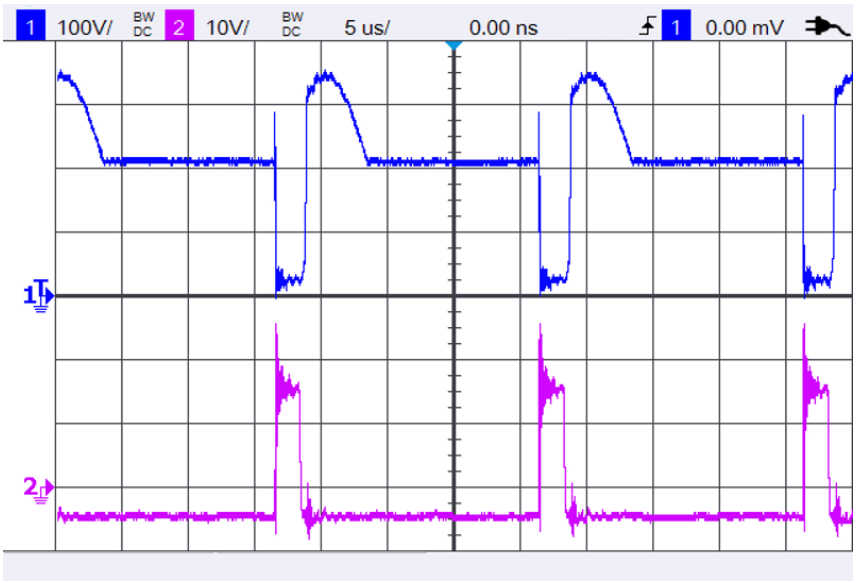


FIGURA 22: Voltaje Mosfet Drain-Source y Gate-Source 10 %. Fuente: Autor

Como se observa, en un tiempo T_{on} este voltaje se vuelve 0 y en un tiempo T_{off}, llega a hasta los 400V, esto debido a que en el tiempo T_{on} la bobina principal tenia un voltaje de 200V, y cuando pasa a T_{off}, el voltaje se vuelve – 200V, lo que se ve reflejado en el Mosfet como 2*200. Este se mantiene en este estado mientras el bobinado principal restaura su estado original y luego se estabiliza en 200V, este comportamiento también se puede ver en la figura 23, que representa las mismas señales de (V_{gs}) y (V_{ds}) per esta vez un PWM del 30

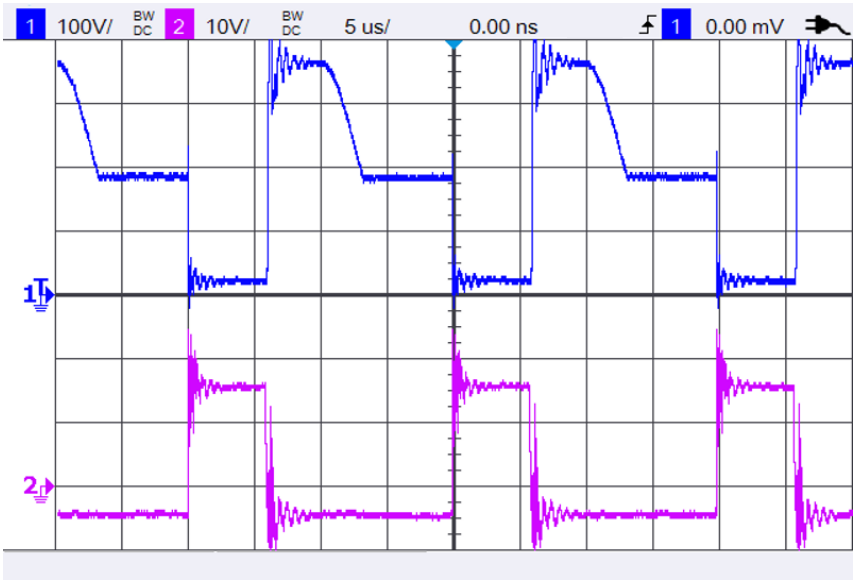


FIGURA 23: Voltaje Mosfet Drain-Source y Gate-Source 30 %. Fuente: Autor

La figura 24 muestra el voltaje de las resistencias de Shunt (R_1, R_2, R_3, R_4, R_5, R_6) 11, este voltaje representa la forma de onda de la corriente que cruza a través del inductor L_1, y con la ayuda de la ley de Ohm, se calcula que tiene en promedio 6.25A cruzando por ella.

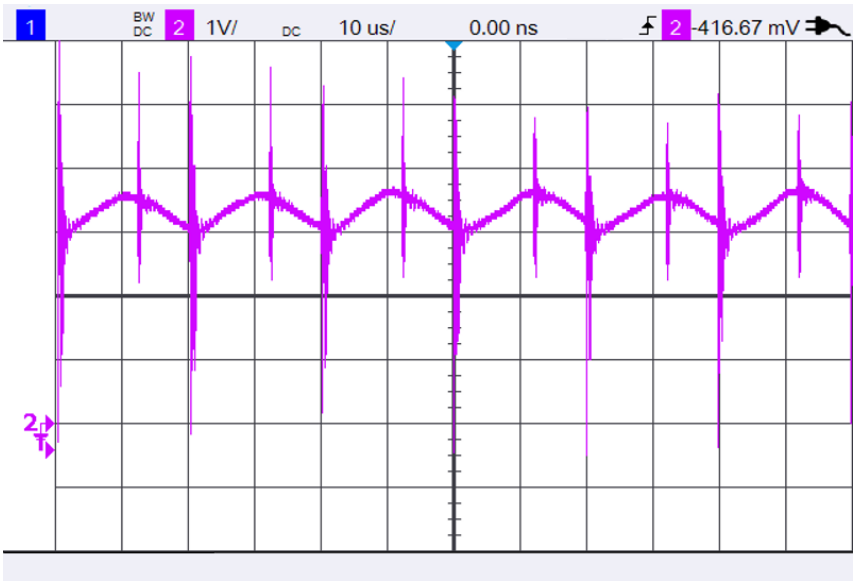


FIGURA 24: Forma onda de corriente en el inductor. Fuente: Autor

La figura 25 muestra el voltaje y la corriente que hay en la estrada del convertidor, de esta se puede deducir que el convertidor esta a plena carga, con una potencia promedio de 116W.

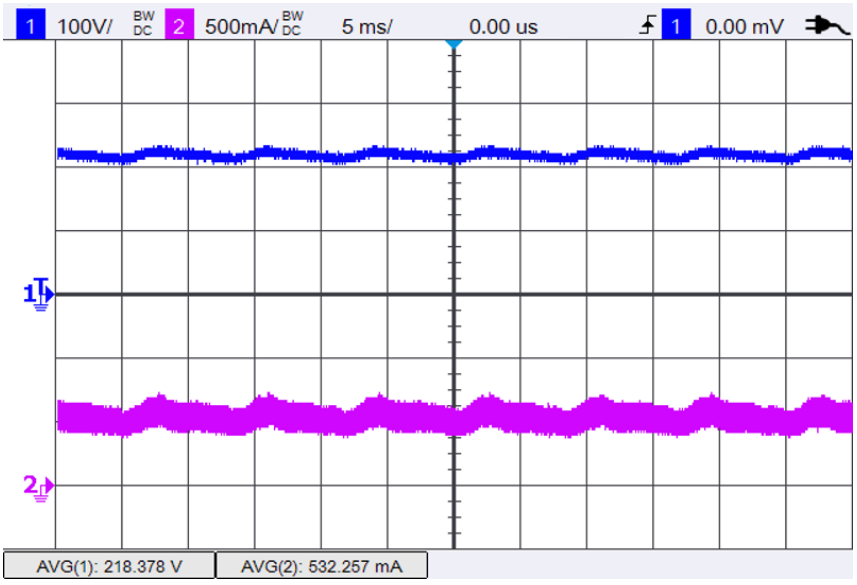


FIGURA 25: Voltaje y corriente de entrada del convertidor. Fuente: Autor

9.2.1. Controlador sintonizado por métodos comunes

Para la implementación del controlador PID sobre el convertidor, primero se debe sintonizar para que realice un correcto control, para esto se utiliza la herramienta de Matlab, Sisotool, esta es una herramienta gráfica que permite el análisis de sistemas.

Para esto se requiere una función de transferencia en lazo abierto, esta fue calculada con anterioridad y se muestra en la ecuación 9.1, la cual se digita en la herramienta, está por defecto realiza un lazo cerrado con control proporcional, representada por la letra k, para el sistema y muestra gráficamente el lugar geométrico de los polos para diferentes valores de k. Además de la respuesta al escalón, junto con su respectivo tiempo de establecimiento, error de estado estacionario y sobrepico máximo.

Luego para sintonizarlo, se usa el mecanismo de PID Autotuning, el cual usa un compensador PID calculado por métodos robustos en respuesta al tiempo. Este arrojará automáticamente los valores de K_p, K_i, K_d y N, con la mejor respuesta temporal que sea capaz de calcular.

El resultado es estos valores fueron $K_p = 0{,}3361$, $K_i = 437{,}3178$, $K_d = 89{,}509 \times 10^{-6}$, $N = 4{,}3 \times 10^{6}$. Estos son implementados en la DSP y el resultado se muestra en la figura 26

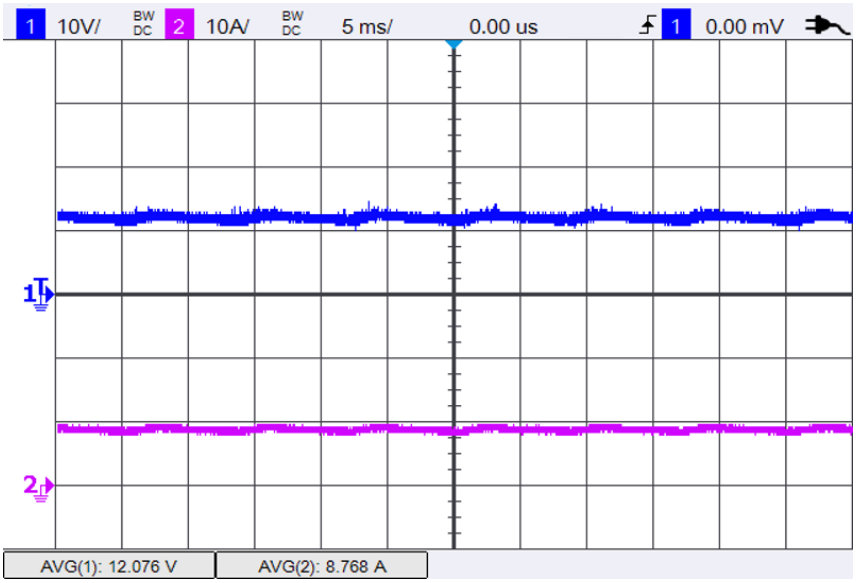


FIGURA 26: Voltaje y corriente de salida, controlado por PID. Fuente: Autor

La figura 26 representa la forma de onda del voltaje y la corriente de salida que tiene el convertidor, muestra como es capaz de controlar el voltaje, con un voltaje promedio de 12.07V y una

corriente de 8.7A, lo que da una potencia promedio de 105.8W, esto supone que el control PID es capaz de controlar la planta sin mayor problema.

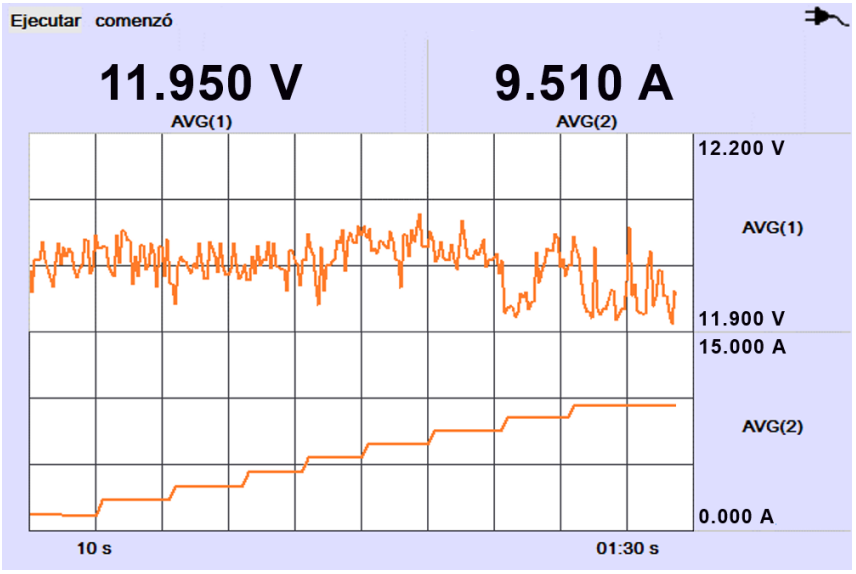


FIGURA 27: Voltaje y corriente de salida, controlado por PID (Variación de carga).
Fuente: Autor

Se hace una prueba se control, variando la carga cada 10 segundos. el resultado de esta prueba se evidencia en la figura 27. Se muestra una variación a la salida, pero su error con cada cambio no aumenta significativamente.

9.2.2. Convertidor controlado por red neuronal artificial

Para la implementación del control por redes neuronales se define los parámetros de entrenamiento, los cuales son el coeficiente de aprendizaje y el coeficiente de inercia, para esta prueba se experimentan con algunos valores al azar, y finalmente se opta por los valores de 500μ, y 0.5 como coeficiente de inercia. Cada época se ejecuta cada 20μ, y con cada época se calcula un nuevo error cuadrático entre la referencia y el voltaje obtenido en la salida.

Las entradas de la red neuronal son el voltaje de salida el cual es medido sobre el banco de resistencias, la corriente del inductor, medido sobre la resistencia de shunt, el error generado entre la referencia y el voltaje medido. Estas entradas de la red neuronal tienen 30 tiempos en atraso, más 15 atrasos en tiempo del ciclo útil, esto da un total de 105 datos de entrada, en la capa oculta tiene 35 neuronas y la capa de salida, una neurona, que es el ciclo útil proporcionado al controlador del Mosfet.

Con estos parámetros mencionados anteriormente, se realizan tres pruebas con cargas diferentes, las cuales son 11Ω, 2.7Ω y 1.2Ω.

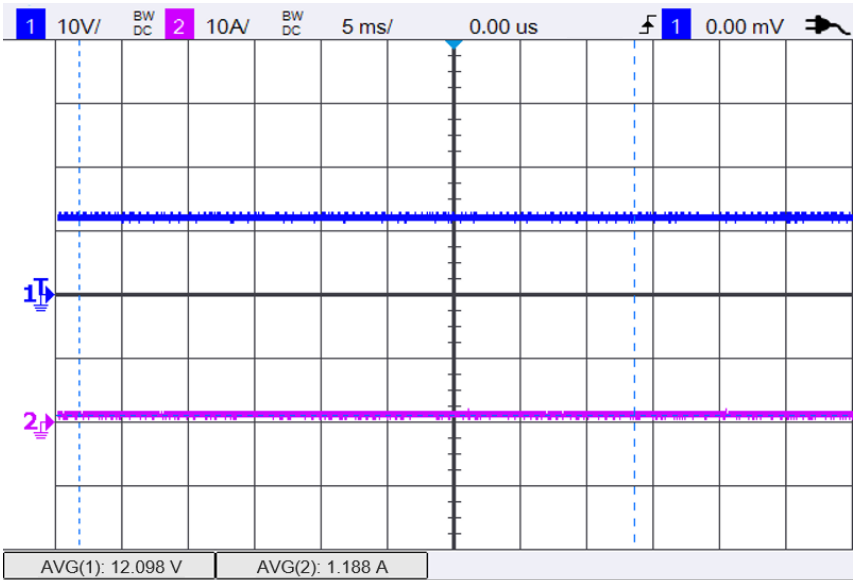


FIGURA 28: Voltaje y corriente de salida, controlado por Red Neuronal (Carga Baja). Fuente: Autor

La primera prueba es utilizando una carga baja (11Ω), y los resultados se ven en la figura 28. Obteniendo una potencia de 14.3W, con un error de estado estacionario de 0.089V.

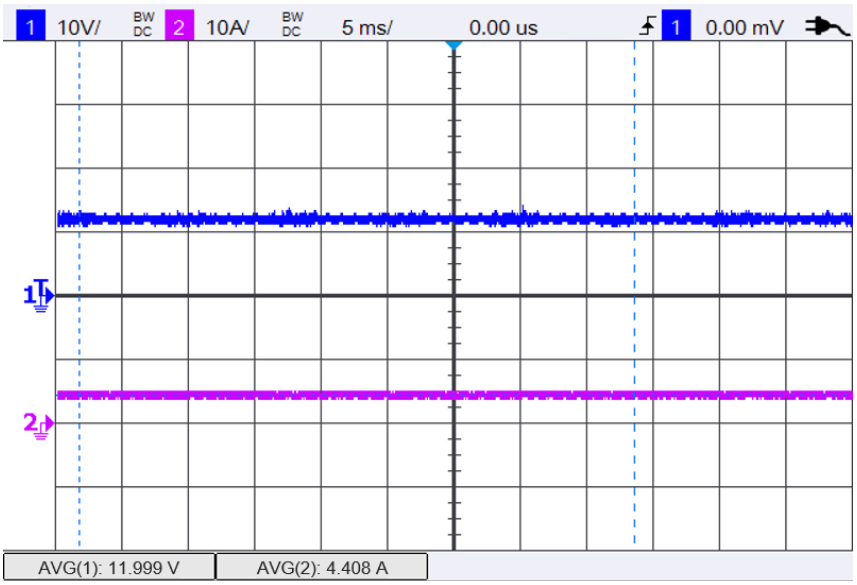


FIGURA 29: Voltaje y corriente de salida, controlado por Red Neuronal (Carga Media). Fuente: Autor

La segunda prueba es utilizando una carga media (2,75Ω), y los resultados se ven en la figura 29. Obteniendo una potencia de 52.8W, con un error de estado estacionario de 0.001V.

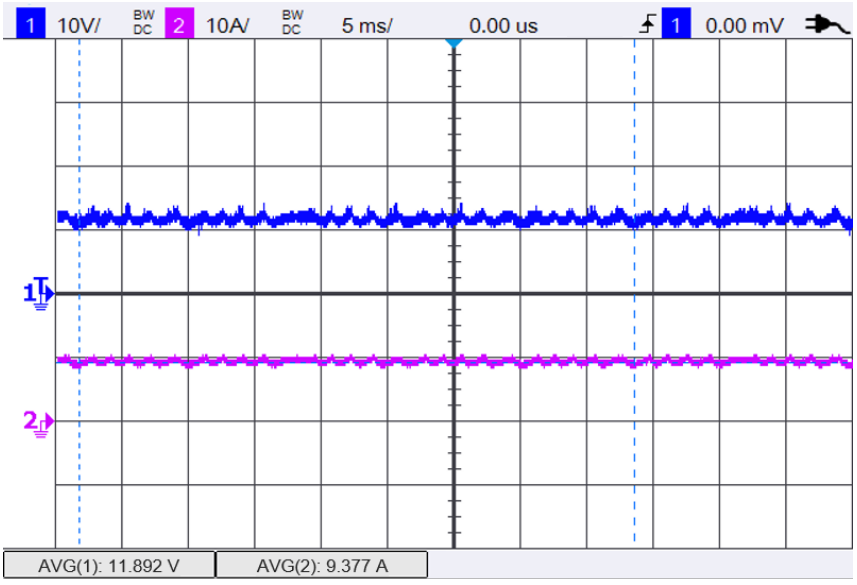


FIGURA 30: Voltaje y corriente de salida, controlado por Red Neuronal (Carga Completa). Fuente: Autor

La tercera prueba es utilizando una carga alta, es decir $1{,}2\Omega$ y los resultados se ven en la figura 30. Obteniendo una potencia de 111.5W, con un error de estado estacionario de 0.108V.

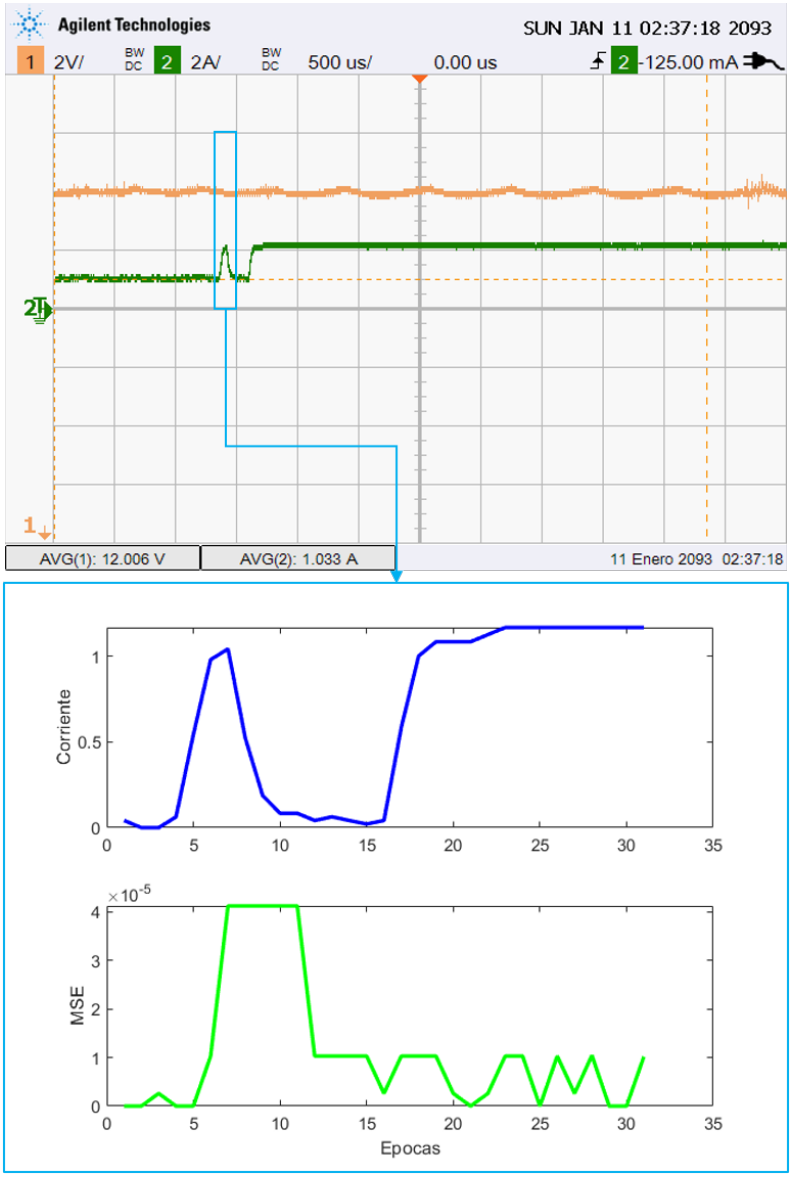


FIGURA 31: Error cuadrático medio vs Épocas en planta física. Fuente: Autor

De igual forma que en la simulación, para demostrar el entrenamiento que realiza la red neuronal, se realiza un cambio en la carga y se toma una una muestra en ese momento. En la figura 31 se demuestra el cambio de la carga con el aumento en la corriente de 0.5A a 1.033A. La figura

muestra como en el momento que hay un cambio en la carga, el error de entrenamiento aumenta, pero con el pasar de las épocas, logra el reducir nuevamente el error cuadrático medio y por consiguiente estabilizar el voltaje a 12V.

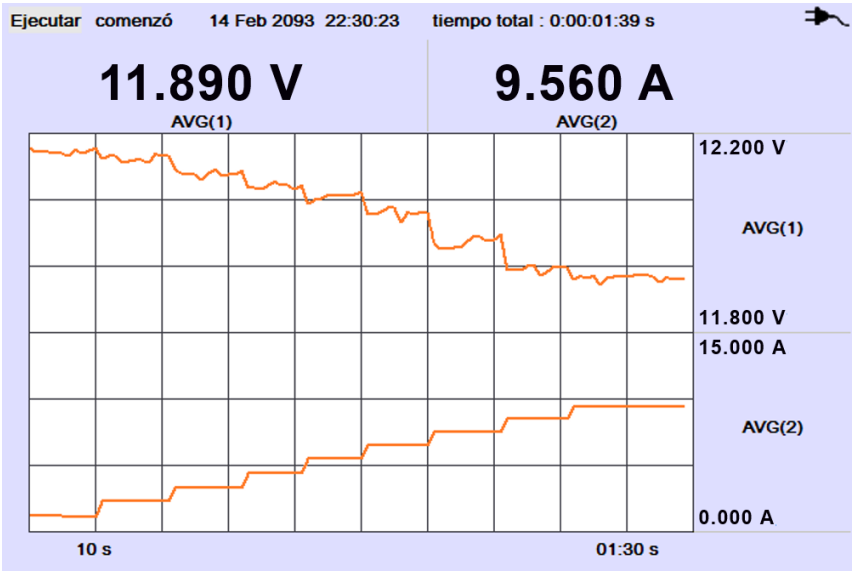


FIGURA 32: Voltaje y corriente de salida, controlado por Red Neuronal (Variación de carga). Fuente: Autor

Finalmente se hace una prueba variando la carga cada 10 segundos, empezando con una resistencia baja (11Ω) y aumentando la carga hasta (1,2Ω), las etapas de la carga se pueden ver en la tabla 2. El resultado de estas prueba se puede observar en la figura 32.

9.2.3. Discusión

A Continuación se presenta una comparación de los voltajes de salida de el convertidor Forward controlados tanto por el controlador PID y el controlador neuronal.

Para comprobar el nivel de control de cada uno de estos controladores, se aumenta la carga cada 10 segundos; teniendo un total de 9 cargas, que varían entre los 11Ω hasta los 1,2Ω

Se mide el error de estado estacionario; con cada variación de carga, el error cuadrático del entrenamiento

Los datos se obtienen de las figuras 27 y 32.

Carga		Control PID		Control neuronal	
Tiempo de variación	resistencia [Ω]	Error cuadrático	Error de estado estacionario [V]	Error cuadrático	Error de estado estacionario [V]
1	11	$0{,}38 \times 10^{-3}$	$0{,}22 \times 10^{-3}$	$25{,}6 \times 10^{-3}$	$-160{,}76 \times 10^{-3}$
2	5,5	$0{,}68 \times 10^{-3}$	$4{,}15 \times 10^{-3}$	$21{,}5 \times 10^{-3}$	$-146{,}49 \times 10^{-3}$
3	3,6	$0{,}20 \times 10^{-3}$	$6{,}97 \times 10^{-3}$	$13{,}11 \times 10^{-3}$	$-114{,}12 \times 10^{-3}$
4	2,7	$0{,}54 \times 10^{-3}$	$8{,}32 \times 10^{-3}$	$7{,}5 \times 10^{-3}$	$-85{,}39 \times 10^{-3}$
5	2,2	$0{,}89 \times 10^{-3}$	$-23{,}17 \times 10^{-3}$	$3{,}7 \times 10^{-3}$	$-58{,}32 \times 10^{-3}$
6	1,8	$0{,}71 \times 10^{-3}$	$-17{,}89 \times 10^{-3}$	$0{,}92 \times 10^{-3}$	$1{,}81 \times 10^{-3}$
7	1,5	$2{,}01 \times 10^{-3}$	$28{,}90 \times 10^{-3}$	$3{,}34 \times 10^{-3}$	$49{,}00 \times 10^{-3}$
8	1,3	$2{,}42 \times 10^{-3}$	$29{,}36 \times 10^{-3}$	$7{,}61 \times 10^{-3}$	$86{,}61 \times 10^{-3}$
9	1,2	$2{,}91 \times 10^{-3}$	$46{,}05 \times 10^{-3}$	$8{,}27 \times 10^{-3}$	$90{,}85 \times 10^{-3}$

TABLA 5: Comparativa control PID y Neuronal. Fuente: Autor

De la tabla 5 se pueden discutir:

Si se comparan el error cuadrático de los dos controladores, el del controlador PID en cada uno de los cambios de carga es menor que el controlador por redes neuronales.

Si solo nos fijamos en el error cuadrático del controlador PID este va aumentando con el aumento de la carga, empezando con un error de $0{,}38 \times 10^{-3}$ hasta alcanzar $2{,}91 \times 10^{-3}$, mientras que el error cuadrático del controlador neuronal disminuye con el aumento de la carga, empezando con un error de $25{,}67 \times 10^{-3}$ disminuyendo hasta $8{,}27 \times 10^{-3}$.

Con respecto al error de estado estacionario, el control PID se mantiene con un menor error en comparación con el control neuronal, manteniéndose cerca de la referencia con un error máximo de 0.046V, mientras que el control neuronal comienza con un error negativo, cruza por cero y termina en un valor positivo iniciando en 0.16V hasta los 0.90V.

Conclusiones

Se diseñó, construyó e implementó un convertidor Forward, tanto en MATLAB/SIMULINK como en planta físico, junto con un banco de cargas, además de tomar como referencia un voltaje de 12V, logra proveer una potencia desde los 13W hasta los 118W.

En el convertidor se implementa en primera instancia un controlador PID, el cual en la simulación y en la planta real tienen un comportamiento similar, ambos logran un seguimiento del voltaje de referencia de 12V, y ante los cambios de carga logra estabilizarse, sin embargo en simulación tiene un componente oscilatorio que mantiene un error de estado estacionario de ± 0.5V, mientras en la planta real el error es de ± 0.1V

Posteriormente se desarrolla e implementa un controlador basado en redes neuronales, el cual usa una arquitectura recurrente, el cual se caracteriza por sus neuronas interiores, quienes tienen autoconexiones, lo que le permite tener una memoria de sus resultados pasados. Los resultados en simulación muestra cómo corrige el error de estado estacionario a 0, y vuelve a seguir la referencia después de un cambio en la carga, reduciendo el sobre impulso. El coeficiente de inercia proporciona un ajuste para reducir las oscilaciones que consigue una mejora tanto en la velocidad de estabilización.

En la planta real, logra mantener un voltaje cercano a los 12V, con un error de ± 0.2V. y el sobreimpulso es reducido.

Al comparar los resultados de este controlador contra los del PID, se muestra como no logra superarlo como se ve en la tabla 5.

La inicialización de los pesos de las redes neuronales es un factor fundamental al momento del rápido entrenamiento de la red, así como la correcta limitación de los parámetros para proteger el convertidor en caso de un entrenamiento irregular.

El resultado de que la red neuronal sea capaz de controlar el convertidor sin necesidad de conocer el modelo matemático es satisfactorio, sin embargo tiene un margen de funcionamiento,

ya que si ocurre una alteración extremadamente brusca en los parámetros, ya sean cambios en el voltaje o en la carga, lo podría sacar de funcionamiento.

Trabajo futuro

Para futuras implementaciones y modificaciones a este trabajo, se plantea explorar su capacidad de controlar otro tipo de plantas, para comprobar la estabilidad que pueda proporcionarle al seguimiento de la referencia. Implementandolo en una variedad de plantas que sean desde simples y lineales hasta plantas mucho más complejas y no lineales

Otra futura implementación puede ser vista en la mejora del controlador presentado en este trabajo para que sea capaz de controlar de una manera más efectiva los cambios de carga y las variaciones del voltaje de entrada, incluyendo más componentes de deep learning, con algoritmos más avanzado y complejos

Otro uso puede ser para comprobar y comparar el comportamiento de este controlador neuronal contra diferentes tipos de controles, tanto lineales y no lineales, sobre una misma planta, registrar los resultados, y relacionar sus datos. Así como Explorar la posibilidad de implementar este controlador en conjunto con otros controladores, como podría ser los controladores PID o los de lógica difusa.

Productos

Este proyecto tiene como producto el artículo titulado “Design and Simulation of a Voltage Control Based on Neural Networks”, presentado en la conferencia “IEEE Colombian Conference on Automatic Control (CCAC)” y publicado en la librería IEEE Xplore.

Junto a los estudiantes Juan Pablo Romero Camacho y David Páez Ramírez, se tuvo como resultado la implementación de un controlador basado en redes neuronales en simulación, el algoritmo de redes neuronales tiene implementada una arquitectura de perceptrones multicapa. Este controlador es aplicado a un convertidor Forward. Hace uso de los coeficientes de aprendizaje y de inercia, el coeficiente de aprendizaje define la velocidad de la actualización de los pesos de la red, mientras el coeficiente de inercia, dependiendo de las oscilaciones en el error de entrenamiento, ajusta el coeficiente de aprendizaje, logran un entrenamiento más estable.

Como resultado, logra el seguimiento del voltaje de referencia, consiguiendo que el error de estado estacionario sea 0, sin embargo, el sobre impulso generado por un cambio de carga abrupto no logra corregirse en gran medida.

El aporte que tuvo al este proyecto se dio los inicios de este trabajo, en el cual se utiliza un perceptrón multicapa del coeficiente de inercia que dio unos primeros resultados favorables en la simulación.

Adicionalmente, se realizó una colaboración en el artículo titulado “Bio-inspired PSO technique applied to PID sintonization for powerfactor correction in a Boost Converter”, presentado en la conferencia “IEEE Colombian Conference on Automatic Control (CCAC)” y publicado en la librería IEEE Xplore. Junto a los estudiantes Juan Pablo Romero Camacho y David Páez Ramírez, el cual trata sobre la sintonización de un controlador PID mediante un algoritmo de optimización de enjambre de partículas múltiples. Este controlador es implementado en convertidor Boost con corrección de factor de potencia.

La implementación de varios enjambres, cada uno con un método de búsqueda y optimización diferentes, quienes superaron los óptimos locales, logran un correcto control del voltaje de salida además de lograr una corrección del factor de potencia mayor al 0.98.

El aporte que tuvo al este proyecto fue el empoderamiento de nuevos conocimientos en temas como son la programación y el estado del arte que se presentan alrededor de los controladores digitales junto el aprendizaje automático.

Bibliografía

[1] Hisham M Soliman y RS Al Abri. «Saturated Digital Control for Regional Pole Placement». En: *2017 9th IEEE-GCC Conference and Exhibition (GCCCE)*. IEEE. 2017, págs. 1-6.

[2] Xuanying Shao, Jian Hu y Zhongtian Zhao. «Stabilisation strategy based on feedback linearisation for DC microgrid with multi-converter». En: *The Journal of Engineering* 2019.16 (2019), págs. 1802-1806.

[3] Saman A Gorji y col. «Topologies and control schemes of bidirectional DC–DC power converters: An overview». En: *IEEE Access* 7 (2019), págs. 117997-118019.

[4] Jesus Aguila-Leon y col. «Particle Swarm Optimization, Genetic Algorithm and Grey Wolf Optimizer Algorithms Performance Comparative for a DC-DC Boost Converter PID Controller». En: *Advances in Science, Technology and Engineering Systems Journal* 6.1 (2021), págs. 619-625.

[5] Hidenori Maruta y col. «Improved transient response for wide input range of DC-DC converter with neural network based digital controller». En: *2017 19th European Conference on Power Electronics and Applications (EPE'17 ECCE Europe)*. IEEE. 2017, P-1.

[6] Pedro Melin y col. «Study of the Open-Source Arduino DUE Board as Digital Control Platform for Three-Phase Power Converters». En: *IEEE Access* 10 (2021), págs. 7574-7587.

[7] Airan Frances y col. «Modeling electronic power converters in smart DC microgrids—An overview». En: *IEEE Transactions on Smart Grid* 9.6 (2017), págs. 6274-6287.

[8] Tamer Kamel, Yevgen Biletskiy y Liuchen Chang. «Capacitor aging detection for the DC filters in the power electronic converters using ANFIS algorithm». En: *2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE)*. IEEE. 2015, págs. 663-668.

[9] Ravindra Janga y Sushama Malaji. «Performance evaluation of active clamp forward converter with fuzzy logic controller». En: *2017 International Conference on Intelligent Computing and Control (I2C2)*. IEEE. 2017, págs. 1-6.

[10] Ju-Young Lee, Chang-Min Lee y Sang-Kyoo Han. «Two-Switch Reset Winding Forward Converter with Low Input Current Ripple». En: *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*. IEEE. 2018, págs. 1543-1549.

[11] Irina A Belova, Miroslav V Martinovich y Vladimir A Skolota. «Neural network model of the solar battery». En: *2017 18th International Conference of Young Specialists on Micro/-Nanotechnologies and Electron Devices (EDM)*. IEEE. 2017, págs. 417-421.

[12] S Kumaravel, A Sivaprasad y S Ashok. «A fuzzy controller for a DC/DC converter to integrate different power characteristic sources». En: *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*. IEEE. 2016, págs. 550-555.

[13] Babita Panda y Bhagabat Panda. «Non-Linear Sliding Mode Control of Three Phase AC-DC PWM Converters». En: *International Journal of Application or Innovation in Engineering & Management (IJAIEM)* (2013).

[14] Martin Lešo y col. «Survey of control methods for DC-DC converters». En: *Acta Electrotechnica et Informatica* 18.3 (2018), págs. 41-46.

[15] Pradeep Shenoy y Anthony Fagnani. «Common Mistakes in DC/DC Converters and How to Fix Them». En: *Power Supply Design Seminar* (2018).

[16] Anton Glushchenko y Vladislav Petrov. «Adaptive Control System Based on Neural Tuner of DC Drive with Sinamics DCM». En: *2019 XXI International Conference Complex Systems: Control and Modeling Problems (CSCMP)*. IEEE. 2019, págs. 117-120.

[17] Meng Jia, Zhuochao Sun y Liter Siek. «A Novel Zero-Voltage-Detector for Buck Converter in Discontinuous Conduction Mode (DCM)». En: *2018 IEEE 4th Southern Power Electronics Conference (SPEC)*. IEEE. 2018, págs. 1-4.

[18] S-C Tan y col. «Special family of PWM-based sliding-mode voltage controllers for basic DC-DC converters in discontinuous conduction mode». En: *IET Electric Power Applications* 1.1 (2007), págs. 64-74.

[19] Carlos Olalla, Ramon Leyva y Abdelali El Aroudi. «QFT design for current-mode PWM buck converters operating in continuous and discontinuous conduction modes». En: *IECON 2006-32nd Annual Conference on IEEE Industrial Electronics*. IEEE. 2006, págs. 1828-1833.

[20] Babita Panda y col. «A comparative study of pi and fuzzy controllers for solar powered dc-dc boost converter». En: *2015 International Conference on Computational Intelligence and Networks*. IEEE. 2015, págs. 47-51.

[21] Ayoub Jebri y col. «Robust Adaptive Neuronal Controller for Exoskeletons with Sliding-Mode». En: *Neurocomputing* (2020).

[22] Sarah Kliff y col. *There Aren't Enough Ventilators to Cope With the Coronavirus*. Mar. de 2020. URL: https://www.nytimes.com/2020/03/18/business/coronavirus-ventilator-shortage.html.

[23] Arturo Wallace. *Coronavirus: cómo funcionan los respiradores y por qué la desesperada carrera por fabricar más es clave en la batalla contra covid-19*. Mar. de 2020. URL: https://www.bbc.com/mundo/noticias-52060716.

[24] Nubia Ilia Ponce de León Puig y col. «An Adaptive Predictive control scheme with dynamic Hysteresis Modulation applied to a DC-DC buck converter». En: *ISA transactions* (2020).

[25] Mengting Zhang y col. «Digital LQR steady-state optimal control with feedforward for nonminimum phase boost DC-DC converter». En: *2016 Chinese Control and Decision Conference (CCDC)*. IEEE. 2016, págs. 384-389.

[26] E Sudeep y col. «Design and implementation of current mode controlled 150W miniature forward converter for defence application». En: *2016 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES)*. IEEE. 2016, págs. 1-6.

[27] Amir Sharifian, Samaneh Fathi Sasansara y Alireza Agah Balgori. «A new control method based on type-2 fuzzy neural PI controller to improve dynamic performance of a half-bridge DC–DC converter». En: *Neurocomputing* 214 (2016), págs. 718-728.

[28] JA Ganeswari y R Kiranmayi. «Performance improvement for DC boost converter with fuzzy controller». En: *2018 2nd International Conference on Inventive Systems and Control (ICISC)*. IEEE. 2018, págs. 358-362.

[29] Ricardo Madeira y col. «Live Demonstration: An Automated Test Bench for an 130nm SC DC-DC Converter». En: *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2018, págs. 1-1.

[30] Reinhard Jaschke. «Conduction losses in dc/dc-converters as buckboost/boostbuck synchronous rectifier types». En: *2007 Compatibility in Power Electronics*. IEEE. 2007, págs. 1-10.

[31] Hao Quan y col. «A survey of computational intelligence techniques for wind power uncertainty quantification in smart grids». En: *IEEE Transactions on Neural Networks and Learning Systems* (2019).

[32] Robert W Erickson y Dragan Maksimovic. *Fundamentals of power electronics*. Springer Science & Business Media, 2007.

[33] T Arunkumari, V Indragandhi y S Sreejith. «Topologies of a DC–DC Converter for Microgrid Application and Implementation of Parallel Quadratic Boost Converter». En: *Advances in Smart Grid and Renewable Energy*. Springer, 2018, págs. 633-644.

[34] Ahmed Hussein y col. «The dynamic performance of photovoltaic supplied dc motor fed from DC-DC converter and controlled by neural networks». En: *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*. Vol. 1. IEEE. 2002, págs. 607-612.

[35] Chen Lanping, Ma Zhenghua y Duan Soulin. «Adaptive speed controller design based on backstepping for DC motor system with parameter uncertainties». En: *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*. Vol. 2. IEEE. 2009, págs. 140-144.

[36] Li Wang y Mi Sa-Nguyen Thi. «Stability enhancement of large-scale integration of wind, solar, and marine-current power generation fed to an SG-based power system through an LCC-HVDC link». En: *IEEE Transactions on Sustainable Energy* 5.1 (2013), págs. 160-170.

[37] Yi-Hung Liao. «A novel reduced switching loss bidirectional AC/DC converter PWM strategy with feedforward control for grid-tied microgrid systems». En: *IEEE Transactions on Power Electronics* 29.3 (2013), págs. 1500-1513.

[38] Chequeo COVID-Colombia. *Uso de respiradores artificiales en casos críticos de COVID-19*. Mayo de 2020. URL: https://uniandes.edu.co/es/noticias/salud-y-medicina/uso-respiradores-artificiales-casos-criticos-covid19.

[39] Abraham Pressman. *Switching power supply design*. McGraw-Hill Education, 2009.

[40] Afshin Balal, Shah Rukh y Shahab Balali. «Designing a Dual Active Transformer DC-DC Forward Converter for DC Micro-Grid Applications Using LTSPICE». En: *International Journal of Applied Engineering Research* 16.4 (2021), págs. 327-331.

[41] MJ Willis y col. «Artificial neural networks in process engineering». En: *IEE Proceedings D (Control Theory and Applications)*. Vol. 138. 3. IET. 1991, págs. 256-266.

[42] Juan P Romero y col. «Bio-inspired PSO technique applied to PID sintonization for powerfactor correction in a Boost Converter». En: *2021 IEEE 5th Colombian Conference on Automatic Control (CCAC)*. IEEE. 2021, págs. 139-144.

[43] Hong-Tao Ye y Zhen-Qiang Li. «PID neural network decoupling control based on hybrid particle swarm optimization and differential evolution». En: *International Journal of Automation and Computing* 17.6 (2020), págs. 867-872.

[44] Guo-Qiang Zeng y col. «Adaptive population extremal optimization-based PID neural network for multivariable nonlinear control systems». En: *Swarm and evolutionary computation* 44 (2019), págs. 320-334.

[45] Xiang Lin Zhu y col. «Decoupling control based on fuzzy neural-network inverse system in marine biological enzyme fermentation process». En: *IEEE Access* 6 (2018), págs. 36168-36175.

[46] Brian Noriega y col. «Design and Simulation of a Voltage Control Based on Neural Networks». En: *2021 IEEE 5th Colombian Conference on Automatic Control (CCAC)*. IEEE. 2021, págs. 133-138.

[47] Baowei Wang y col. «Temperature error correction based on BP neural network in meteorological wireless sensor network». En: *International Journal of Sensor Networks* 23.4 (2017), págs. 265-278.

[48] Ji-chao Li y col. «A link prediction method for heterogeneous networks based on BP neural network». En: *Physica A: Statistical Mechanics and its Applications* 495 (2018), págs. 1-17.

[49] Kai Cui y Xiang Jing. «Research on prediction model of geotechnical parameters based on BP neural network». En: *Neural Computing and Applications* 31.12 (2019), págs. 8205-8215.

[50] Wei He y Yiting Dong. «Adaptive fuzzy neural network control for a constrained robot using impedance learning». En: *IEEE transactions on neural networks and learning systems* 29.4 (2017), págs. 1174-1186.

[51] Paulo Vitor de Campos Souza. «Fuzzy neural networks and neuro-fuzzy networks: A review the main techniques and applications used in the literature». En: *Applied soft computing* 92 (2020), pág. 106275.

[52] Jinjun Tang y col. «Lane-changes prediction based on adaptive fuzzy neural network». En: *Expert Systems with Applications* 91 (2018), págs. 452-463.

[53] Jhon Fredy Bayona, Jose Guarnizo y Nancy Gelvez. «Pulse Width Prediction Control Technique Applied to a Half-Bridge Boost». En: *Tecciencia* 13.25 (2018), págs. 47-54.

[54] José Guillermo Guarnizo Marin, Nelson Díaz Aldana, Rodríguez Trujillo y col. «Diseño e implementación de un Controlador Neuronal Inverso aplicado a un Conversor VSC para el control de la potencia activa y potencia reactiva, basado en regiones de trabajo». En: *Revista Facultad de Ingeniería Universidad de Antioquia* 72 (2014), págs. 9-19.

[55] Nelson Díaz Aldcmcí, César Leonardo Trujillo y José Guillermo Guarnizo. «Active and reactive power flow regulation for a grid connected vsc based on fuzzy controllers». En: *Revista Facultad de Ingeniería Universidad de Antioquia* 66 (2013), págs. 118-130.

[56] Jose Guillermo Guarnizo Marin, Juan Carlos Lopez Rodriguez y col. «Análisis diseño e implementación de un control neuronal de un conversor DC-DC». B.S. thesis. Universidad Distrital Francisco José de Caldas.

[57] Guarnizo M José Guillermo, Guacaneme M Javier Antonio y Trujillo R Cesar Leonardo. «General inverse neural current control for buck converter». En: *Novel Algorithms and Techniques in Telecommunications, Automation and Industrial Electronics*. Springer, 2008, págs. 117-122.

[58] Gergana VACHEVA, Nikolay HINOV y Bogdan GILEV. «Comparision Between Clasical and Modern Methods for Optimal Control of Buck DC/DC Converter». En: *2019 16th Conference on Electrical Machines, Drives and Power Systems (ELMA)*. IEEE. 2019, págs. 1-4.

[59] Tomislav Dragičević, Patrick Wheeler y Frede Blaabjerg. «Artificial intelligence aided automated design for reliability of power electronic systems». En: *IEEE Transactions on Power Electronics* 34.8 (2018), págs. 7161-7171.

[60] Hidenori Maruta y Daiki Hoshino. «Transient response improvement of repetitive-trained neural network controlled dc-dc converter with overcompensation suppression». En: *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*. Vol. 1. IEEE. 2019, págs. 2088-2093.

[61] Xavier Glorot y Yoshua Bengio. «Understanding the difficulty of training deep feedforward neural networks». En: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop y Conference Proceedings. 2010, págs. 249-256.

Apéndice A

Código en MATLAB SIMULINK

```
function Duty_cycle = DSP(V_C, I_L)
    coder.extrinsic('evalin', 'assignin')
    % Initialize values - start
    rates = zeros(1,3);
    rates = evalin('base','rates');
    Vref = 12;
    len_data = round(8);
    n_epoch = 1;
    l_rate = rates(1);
    m_rate = rates(2);
    % Initialize values - end
    % Create dataset - start
    dataset = zeros(1,len_data*7/2+1);
    dataset = evalin('base','dataset');
    n_inputs = length(dataset(1,:)) - 1;
    n_hidden = rates(3);
    n_outputs = 1;
    % Create dataset - end
    % Create network - start
    network = cell(2,3);
        %Layer 1 : input to hidden
    network{1,1} = (2*rand(n_inputs+1,n_hidden)-1)/sqrt(n_inputs);
%Weights
    network{1,2} = zeros(1,n_hidden);
%Outputs
    network{1,3} = zeros(1,n_hidden);
%Deltas
```

```
        %Layer 2 : hidden to output
    network{2,1} = (2*rand(n_hidden+1,n_outputs)-1)/sqrt(n_hidden);
%Weights
    network{2,2} = zeros(1,n_outputs);
%Outputs
    network{2,3} = zeros(1,n_outputs);
%Deltas
    network = evalin('base','network');
    % Create network - end
    % Normalize dataset - start
    dataset(1:len_data*7/2-1) = dataset(2:len_data*7/2);
    dataset(len_data*1) = normalize_dataset(V_C,[0;24]);
    dataset(len_data*2) = normalize_dataset(I_L,[0;10]);
    dataset(len_data*3) = normalize_dataset(Vref - V_C,[-12;12]);
    dataset(len_data*7/2) = network{end,2};
    dataset(end) = 0.5;
    % Normalize dataset - end
    % Main - start
    for epoch = 1:n_epoch
        old_network = network;
        for row = 1:length(dataset(:,1))
            % forward propagate - start
            inputs = dataset(row,:);
            for neuron = 1:length(network{1,1}(1,:))
                weights = network{1,1}(:,neuron)';
                activation = weights(end);
                for i = 1:length(weights)-1
                    activation = activation + weights(i)*inputs(i);
                end
                network{1,2}(neuron) = (exp(activation)-exp(-
activation))/(exp(activation)+exp(-activation));
            end
            inputs = network{1,2};
            for neuron = 1:length(network{2,1}(1,:))
                weights = network{2,1}(:,neuron)';
                activation = weights(end);
                for i = 1:length(weights)-1
                    activation = activation + weights(i)*inputs(i);
                end
                network{2,2}(neuron) = (exp(activation)-exp(-
activation))/(exp(activation)+exp(-activation));
            end
            % forward propagate - end
```

```
            expected = dataset(row,end);
            % backward propagate error - start
            errors = zeros(1,length(network{2,1}(1,:)));
            for j = 1:length(network{2,1}(1,:))
                errors(j) = dataset(len_data) - expected(j);
            end
            for j = 1:length(network{2,1}(1,:))
                network{2,3}(j) = errors(j) * (1 -
network{2,2}(j)*network{2,2}(j));
            end
            errors = zeros(1,length(network{1,1}(1,:)));
            for j = 1:length(network{1,1}(1,:))
                error = 0.0;
                for neuron = 1:length(network{1+1,1}(1,:))
                    error = error + (network{1+1,1}(j,neuron) *
network{1+1,3}(neuron));
                end
                errors(j) = error;
            end
            for j = 1:length(network{1,1}(1,:))
                network{1,3}(j) = errors(j) * (1 -
network{1,2}(j)*network{1,2}(j));
            end
            % backward propagate error - end
            % update weights - start
            inputs = dataset(1:end-1);
            for neuron = 1:length(network{1,1}(1,:))
                for j = 1:length(inputs)
                    network{1,1}(j,neuron) = network{1,1}(j,neuron) - (
l_rate * network{1,3}(neuron) + l_rate*m_rate * old_network{1,3}(neuron)
) * inputs(j);
                end
                network{1,1}(end,neuron) = network{1,1}(end,neuron) - (
l_rate * network{1,3}(neuron) + l_rate*m_rate * old_network{1,3}(neuron)
);
            end
            inputs = network{2 - 1,2};
            for neuron = 1:length(network{2,1}(1,:))
                for j = 1:length(inputs)
                    network{2,1}(j,neuron) = network{2,1}(j,neuron) - (
l_rate * network{2,3}(neuron) + l_rate*m_rate * old_network{2,3}(neuron)
) * inputs(j);
                end
```

```
                network{2,1}(end,neuron) = network{2,1}(end,neuron) - (
l_rate * network{2,3}(neuron) + l_rate*m_rate * old_network{2,3}(neuron)
);
            end
            % update weights - end
        end
    end
    % Main - end
    assignin('base','dataset',dataset);
    assignin('base','network',network);
    Duty_cycle = network{2,2};
end
function dataset = normalize_dataset(dataset,minmax)
    for i = 1:length(dataset(1,:))
        dataset(:,i) = (dataset(:,i) - minmax(1))/(minmax(2) -
minmax(1));
    end
end
```

```
%% Limpieza
clear all
close all
clc
%% Creacion Network
Time = 7e4;
TimeSim = Time*20e-6;
Resistencia = 11*[0.8 0.5 0.2 0.7 0.2 0.5]';
len_data = 8;
dataset = zeros(1,len_data*7/2+1);
n_inputs = length(dataset(1,:)) - 1;
n_outputs = 1;

n_hidden = 40;
l_rate = 250e-6;
m_rate = 1.5;

rates = zeros(1,3);
rates(1) = l_rate;
rates(2) = m_rate;
rates(3) = n_hidden;

expected = ones(Time*10+1,1)*12;
%network = initialize_network(n_inputs, n_hidden, n_outputs);

%% Entrenamiento Network
out = sim('Converter.slx','SimulationMode','normal');
V_out = out.get('V_out');
I_out = out.get('I_out');
I_in = out.get('V_in');
V_in = out.get('V_in');
R_load = out.get('Res');
I_ind = out.get('I_ind');
D = out.get('Dutty');

X = (0:2e-6:TimeSim)';

figure(1)
subplot(3,1,1)
plot(X(1:7e4*10+1),V_out((Time-7e4)*10+1:end),'LineStyle','-
','LineWidth',1,'Color','b')
ylabel('Voltaje (V)')
grid on
```

```
subplot(3,1,2)
plot(X(1:7e4*10+1),I_ind((Time-7e4)*10+1:end),'LineStyle','-
','LineWidth',1,'Color','g')
ylabel('Corriente (A)')
grid on
subplot(3,1,3)
plot(X(1:7e4*10+1),R_load((Time-7e4)*10+1:end),'LineStyle','-
.','LineWidth',1,'Color','k')
ylabel('Resistencia (\Omega)')

xlabel('Time(s)')
grid on

Error03 = 1/length(V_out)*sum((V_out-expected).^2);
%% 1. Initialize Network
function network = initialize_network(n_inputs, n_hidden, n_outputs)
    network = cell(2,3);% cell(N_LAYERS,3)
    %Layer 1 : input to hidden
    network{1,1} = (2*rand(n_inputs+1,n_hidden)-1)/sqrt(n_inputs);
%Weights
    network{1,2} = zeros(1,n_hidden);
%Outputs
    network{1,3} = zeros(1,n_hidden);
%Deltas

    %Layer 2 : hidden to hidden
    %network{2,1} = (2*rand(n_hidden+1,n_hidden)-1)/sqrt(n_hidden);
%Weights
    %network{2,2} = zeros(1,n_hidden);
%Outputs
    %network{2,3} = zeros(1,n_hidden);
%Deltas

    %Layer 3 : hidden to output
    network{2,1} = (2*rand(n_hidden+1,n_outputs)-1)/sqrt(n_hidden);
%Weights
    network{2,2} = zeros(1,n_outputs);
%Outputs
    network{2,3} = zeros(1,n_outputs);
%Deltas
end
```

Apéndice B

Código en Code Composer Studio

```
#include "DSP28x_Project.h"
#include "F2806x_EPwm_defines.h"
#include "F2806x_Cla.h"

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

#define n_inputs 105
#define n_hidden 35
#define n_outputs 1

struct Layer
{
    double weights[106][35];
    double outputs[35];
    double deltas[35];
};

__interrupt void adc_isr(void);
void EPwm2_Config();
float NeuralControl();
float PIDControl();

float normalize_dataset(float result, float min, float max);
```

```
float f_rand();

Uint16 Vout;
Uint16 Il;

struct Layer network[2];
struct Layer old_network[2];

Uint16 V_ref = 2730;
Uint16 n_epoch = 1;
float32 loop[12];
float32 l_rate = 0.0005;
float32 m_rate = 0.5;
float32 dutty_cycle = 0.;
float32 dataset[n_inputs + 1];

int contador = 0;
float BPWM = 0.0;

void main(void)
{
    InitSysCtrl();
    DINT;
    InitPieCtrl();
    IER = 0x0000;
    IFR = 0x0000;
    InitPieVectTable();
    EALLOW;
    PieVectTable.ADCINT1 = &adc_isr;
    EDIS;
    InitAdc();
    AdcOffsetSelfCal();

    PieCtrlRegs.PIEIER1.bit.INTx1 = 1;
    IER |= M_INT1;
    EINT;
    ERTM;

    EALLOW;
    AdcRegs.ADCCTL2.bit.ADCNONOVERLAP = 1;
    AdcRegs.ADCCTL1.bit.INTPULSEPOS = 1;
```

```
AdcRegs.INTSEL1N2.bit.INT1E = 1;
AdcRegs.INTSEL1N2.bit.INT1CONT = 0;
AdcRegs.INTSEL1N2.bit.INT1SEL = 1;

AdcRegs.ADCSOC0CTL.bit.CHSEL = 0X9;
AdcRegs.ADCSOC1CTL.bit.CHSEL = 0X2;


AdcRegs.ADCSOC0CTL.bit.TRIGSEL = 5;
AdcRegs.ADCSOC1CTL.bit.TRIGSEL = 5;

AdcRegs.ADCSOC0CTL.bit.ACQPS = 6;
AdcRegs.ADCSOC1CTL.bit.ACQPS = 6;
EDIS;

EPwm1Regs.ETSEL.bit.SOCAEN = 1;
EPwm1Regs.ETSEL.bit.SOCASEL = 4;
EPwm1Regs.ETPS.bit.SOCAPRD = 1;
EPwm1Regs.CMPA.half.CMPA = 1;
EPwm1Regs.TBPRD = 900;
EPwm1Regs.TBCTL.bit.CTRMODE = 0;

srand((unsigned) time( NULL));

int i = 0;
int j = 0;

for (i = 0; i < n_inputs + 1; i++)
{
    dataset[i] = 0.;
}
for (i = 0; i < n_hidden; i++)
{
    for (j = 0; j < n_inputs + 1; j++)
    {
        network[0].weights[j][i] = (2. * f_rand() - 1.) / sqrt(n_inputs);
    }
    network[0].outputs[i] = 0.;
    network[0].deltas[i] = 0.;
}
```

```
    for (i = 0; i < n_outputs; i++)
    {
        for (j = 0; j < n_hidden + 1; j++)
        {
            network[1].weights[j][i] = (2. * f_rand() - 1.) / sqrt(n_hidden);
        }
        network[1].outputs[i] = 0.;
        network[1].deltas[i] = 0.;
    }

    EPwm2_Config();

    for (;;)
        ;
}

__interrupt void adc_isr(void)
{
    Vout = (0.25*AdcResult.ADCRESULT0)+((1-0.25)*Vout);
    Il = (0.25*AdcResult.ADCRESULT1)+((1-0.25)*Il);

    contador++;
    if (contador >= 2)
    {
        contador = 0;
        BPWM = NeuralControl();
        //BPWM = PIDControl();

        EPwm2Regs.CMPA.half.CMPA = 1800 * (1-BPWM);
    }

    AdcRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;
    PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;

    return;
}

void EPwm2_Config()
{
    InitEPwm2Gpio();
    EALLOW;
```

```
    SysCtrlRegs.PCLKCR0.bit.TBCLKSYNC = 1;
    EDIS;

    EPwm2Regs.TBCTL.bit.PRDLD = TB_IMMEDIATE;
    EPwm2Regs.TBPRD = 1800;
    EPwm2Regs.CMPA.half.CMPA = 1800*(1-0.3);
    EPwm2Regs.CMPA.half.CMPAHR = (1 << 8);
    EPwm2Regs.CMPB = 1;
    EPwm2Regs.TBPHS.all = 0;
    EPwm2Regs.TBCTR = 0;

    EPwm2Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
    EPwm2Regs.TBCTL.bit.PHSEN = TB_DISABLE;
    EPwm2Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
    EPwm2Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
    EPwm2Regs.TBCTL.bit.CLKDIV = TB_DIV1;

    EPwm2Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
    EPwm2Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;
    EPwm2Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
    EPwm2Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;

    EPwm2Regs.AQCTLA.bit.ZRO = AQ_CLEAR;
    EPwm2Regs.AQCTLA.bit.CAU = AQ_SET;
    EPwm2Regs.AQCTLB.bit.ZRO = AQ_CLEAR;
    EPwm2Regs.AQCTLB.bit.CBU = AQ_SET;
}

float PIDControl()
{
    //loop[0]  =>  Vout
    //loop[1]  =>  error(n)
    //loop[2]  =>  error(n-1)
    //loop[3]  =>  proportional(n)
    //loop[4]  =>  integral(n)
    //loop[5]  =>  integral(n-1)
    //loop[6]  =>  derivative(n)
    //loop[7]  =>  derivative (n-1)
    //loop[8]  =>  u
    //loop[9]  =>  ul
    //loop[10] =>  eawv(n-1)
```

```
    //loop[11] =>  eawv(n-2)

    loop[0] = (0.004396*(float32)Vout);
    loop[1] = 10 - loop[0];
    loop[3] = 0.08*loop[1];
    loop[4] = loop[5]+0.00064*(loop[1]+loop[2])-loop[10]-loop[11];
    loop[6] = 0.3443*loop[7]+0.0023*(loop[1]-loop[2]);
    loop[8] = loop[3]+loop[4];//+loop[6];
    if(loop[8] < 0.05){loop[9] = 0.05;}
    else if(loop[8] > 0.40){loop[9] = 0.40;}
    else{loop[9] = loop[8];}
    loop[11] = loop[10];
    loop[10] = loop[8] - loop[9];
    loop[2] = loop[1];
    loop[5] = loop[4];
    loop[7] = loop[6];
    return loop[9];
}

float NeuralControl()
{
    int i = 0;
    int j = 0;
    int neuron = 0;

    for (i = 0; i < n_inputs; i++)
    {
        dataset[i] = dataset[i + 1];
    }
    dataset[7] = normalize_dataset(Vout, 0., 4095.);
    dataset[15] = normalize_dataset(Il, 0., 4095.);
    dataset[23] = normalize_dataset(V_ref - Vout, V_ref - 0, V_ref - 4095);
    dataset[27] = network[1].outputs[0];
    dataset[28] = normalize_dataset(V_ref, 0., 4095.);

    old_network[0] = network[0];
    old_network[1] = network[1];

    double inputs[n_inputs + 1];
    for (i = 0; i < n_inputs + 1; i++)
    {
```

```
        inputs[i] = dataset[i];
    }

    for (neuron = 0; neuron < n_hidden; neuron++)
    {
        double weights[n_inputs + 1];
        for (i = 0; i < n_inputs + 1; i++)
        {
            weights[i] = network[0].weights[i][neuron];
        }
        double activation = weights[n_inputs];
        for (i = 0; i < n_inputs; i++)
        {
            activation = activation + weights[i] * inputs[i];
        }
        network[0].outputs[neuron] = (exp(activation) - exp(-activation))
                / (exp(activation) + exp(-activation));
    }
    for (i = 0; i < n_hidden; i++)
    {
        inputs[i] = network[0].outputs[i];
    }
    for (neuron = 0; neuron < n_outputs; neuron++)
    {
        double weights[n_hidden + 1];
        for (i = 0; i < n_hidden + 1; i++)
        {
            weights[i] = network[1].weights[i][neuron];
        }
        double activation = weights[n_hidden];
        for (i = 0; i < n_hidden; i++)
        {
            activation = activation + weights[i] * inputs[i];
        }
        network[1].outputs[neuron] = (exp(activation) - exp(-activation))
                / (exp(activation) + exp(-activation));
    }
    double expected = dataset[n_inputs];

    double errors[n_hidden];
```

```
    for (j = 0; j < n_outputs; j++)
    {
        errors[j] = dataset[7] - expected;
    }
    for (j = 0; j < n_outputs; j++)
    {
        network[1].deltas[j] = errors[j]
                * (1. - network[1].outputs[j] * network[1].outputs[j]);
    }
    for (j = 0; j < n_hidden; j++)
    {
        double error = 0.;
        for (neuron = 0; neuron < n_outputs; neuron++)
        {
            error =
                    error
                            + (network[1].weights[j][neuron]
                                    * network[1].deltas[neuron]);
        }
        errors[j] = error;
    }
    for (j = 0; j < n_hidden; j++)
    {
        network[0].deltas[j] = errors[j]
                * (1. - network[0].outputs[j] * network[0].outputs[j]);
    }
    for (i = 0; i < n_outputs; i++)
    {
        inputs[i] = dataset[n_inputs];
    }
    for (neuron = 0; neuron < n_hidden; neuron++)
    {
        for (j = 0; j < n_outputs; j++)
        {
            network[0].weights[j][neuron] = network[0].weights[j][neuron]
                    - (l_rate * network[0].deltas[neuron]
                            + l_rate * m_rate * old_network[0].deltas[neuron])
                            * inputs[j];
        }
        network[0].weights[n_outputs - 1][neuron] =
network[0].weights[n_outputs
```

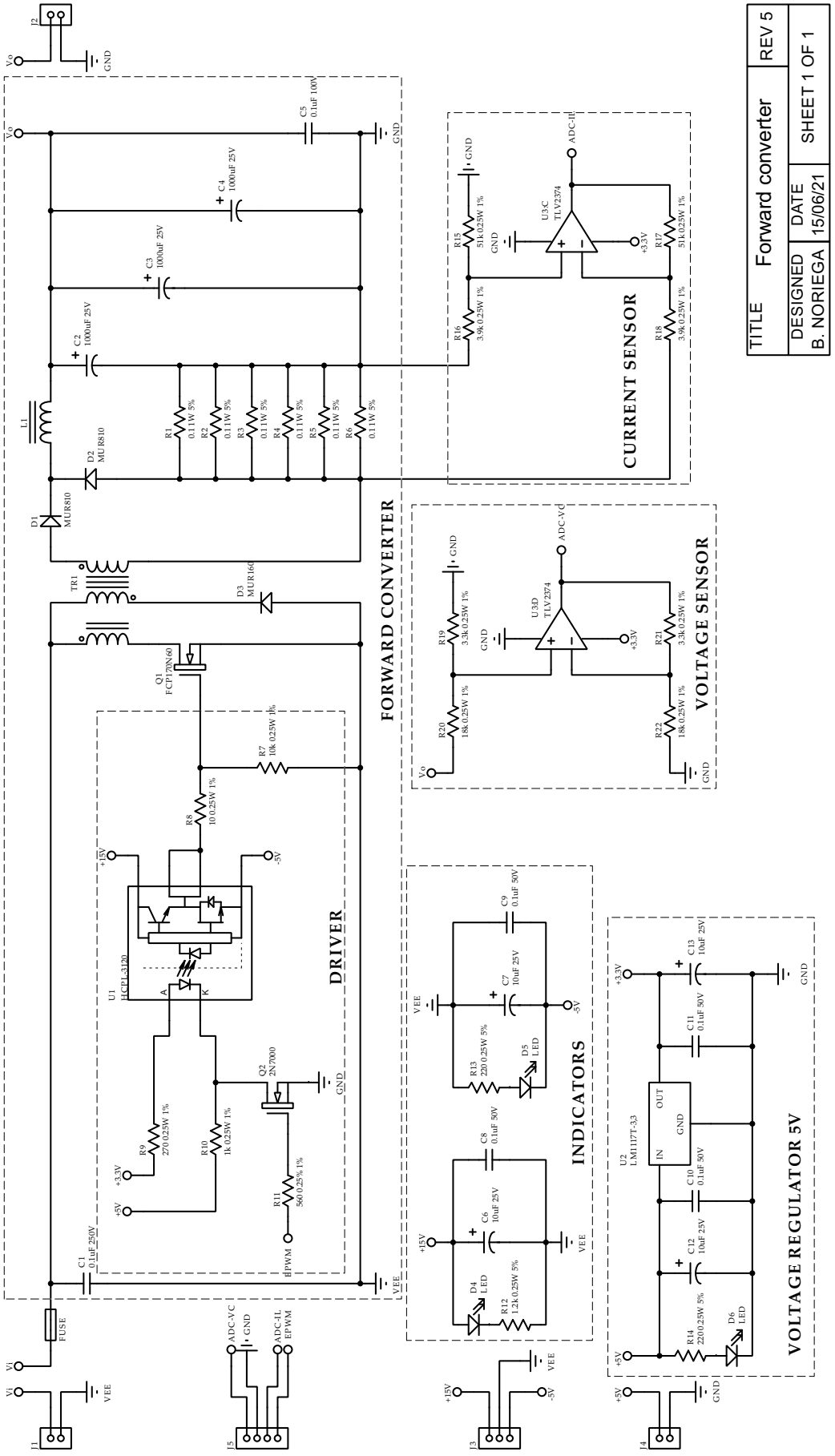
```
                - 1][neuron]
                - (l_rate * network[0].deltas[neuron]
                        + l_rate * m_rate * old_network[0].deltas[neuron]);
    }

    for (i = 0; i < n_hidden; i++)
    {
        inputs[i] = network[0].outputs[i];
    }
    for (neuron = 0; neuron < n_outputs; neuron++)
    {
        for (j = 0; j < n_hidden; j++)
        {
            network[1].weights[j][neuron] = network[1].weights[j][neuron]
                    - (l_rate * network[1].deltas[neuron]
                            + l_rate * m_rate * old_network[1].deltas[neuron])
                            * inputs[j];
        }
        network[1].weights[n_hidden - 1][neuron] = network[1].weights[n_hidden
                - 1][neuron]
                - (l_rate * network[1].deltas[neuron]
                        + l_rate * m_rate * old_network[1].deltas[neuron]);
    }
    if(network[1].outputs[0] > 0.40){network[1].outputs[0] = 0.40;}
    else if(network[1].outputs[0] < 0.05){network[1].outputs[0] = 0.05;}
    return network[1].outputs[0];
}

float normalize_dataset(float dato, float min, float max)
{
    return (dato - min) / (max - min);
}
float f_rand()
{
    return (float) rand() / RAND_MAX;
}
```

Apéndice C

Plano esquemático del convertidor


FORWARD CONVERTER
DRIVER
CURRENT SENSOR
VOLTAGE SENSOR
INDICATORS
VOLTAGE REGULATOR 5V
TITLE
Forward converter
REV 5
DESIGNED
B. NORIEGA
DATE
15/06/21
SHEET 1 OF 1