ESTANCIA DE INVESTIGACIÓN EN CIRCUITOS EMBEBIDOS Y SISTEMAS EN CHIP APLICADOS A ROBÓTICA Y CONTROL.

Autor

Mario Andrés Pastrana Triana

Director

Cesar Augusto Peña Cortes

Dr.

Codirector

Sergio Andres Pertuz Méndez

M.Sc.

Universidad de Pamplona

Facultad de Ingenierías y Arquitectura

Departamento de Ingenierías Mecánica, Mecatrónica, e Industrial

Programa de Ingeniería Mecatrónica

Pamplona, Norte de Santander – Colombia

2019

DEDICATORIA

Para mi madre Maria del Rosario Triana Diaz y mi padre Floresmiro Pastrana Medina.

Bueno también para mi hermano Juan Pablo Pastrana Triana.

AGRADECIMIENTOS

Primeramente, agradecer a Dios todo poderos y a la Santísima virgen María por guiar mis pasos al momento de realizar este trabajo de investigación, a mis padres Floresmiro Pastrana Medina y María del Rosario Triana Díaz, a mi hermano Juan Pablo Pastrana Triana, a la Universidad de Pamplona por patrocinar la estancia de investigación, ya que sin ese dinero no hubiera sido posible poder viajar y trabajar en Brasil, a Sergio Andres Pertuz Mendez, el cual con su paciencia y ayuda me asesoró en este trabajo de investigación y de una manera muy especial a todos mis profesores, sin los cuales nunca hubiera podido llegar hasta esta instancia.

Me gustaría agradecer a cada profesor que estuvo con migo en este proceso de formación, de manera especial al profesor Pablo A. Santafé, el cual con su trabajo y dedicación me enseño el valor de la puntualidad, al profesor Diego Armando Mejía Bugallo, el cual a lo largo de la carrera nos inspiró para ir más allá de los problemas planteados en los libros, al profesor Cesar Augusto Peña Cortés, por mostrarme ese maravilloso mundo que es la investigación y por brindarme acompañamiento a lo largo de mi proceso universitario y por último al profesor Javier Adolfo Corredor, el cual despertó en mi uno de los gusto más espectaculares de toda la carrera, el gusto por el “CONTROL”.

De igual forma agradecer a todos mis camaradas de universidad, los que estuvieron en esos anocheceres y en esas Albas, de manera especial a mi primer mejor amigo, Juan Manuel Calderón Agudelo, el cual fue mi primer apoyo, mi primer compañero de estudios, a Sergio Daniel Cardona, con el cual trabajamos hombro a hombro en todo este proceso universitario, a mi primo Gabriel Triana Osorio, con el cual trabajamos arduamente para poder llegar hasta esta instancia y por ultimo a Jaime Yesid Contreras Chaparro, gracias por siempre estar en esos momentos de dificultad, por ser mi compañero de proyectos y por ser un gran amigo, gracias por siempre acompañarme en mis locos proyectos y por siempre estar conmigo hasta el final.

Resumen

En esta monografía se describe los trabajos y obligaciones que se realizaron en la estancia de investigación, las cuales consistieron primeramente en apoyar en el diseño de una placa PCB, la cual será utilizada para la creación de un pequeño servomotor este será implementado en la construcción de una mano robótica capaza de realizar tareas de manipulación, esta placa fue diseñada en el software Eagle, de autodesk, el cual proporciona una licencia gratuita por tres años, además de tener una buena compatibilidad con el software Solidworks.

Durante el transcurso de toda la pasantía se asistió a la disciplina de “Algoritmos bioinspirados para ingeniería”, con la cual se logró aportar en la realización de un artículo científico, el cual contiene diferentes tipos de algoritmos bioinspirados para realizar la sintonización de un controlador de impedancia. Los algoritmos base que se trabajaron fueron PSO (Particle Swarm Optimization), y DE (Diferential evolution), aplicando métodos de diversidad artificial para mejorar la eficiencia de los algoritmos base, se comparan los algoritmos aplicando análisis estadísticos, testes de hipótesis y la respuesta del controlador de cada uno.

Por último, se trabajó en el área de programación de ARM en circuitos en chips, para lo cual se tomó una disciplina llamada “Sistemas Digitales”, con esta se adquirieron conocimientos en el área de lenguaje de descripción de hardware llamado VHDL (del inglés “VHSIC Hardware Description Language"), utilizado para sintetizar circuitos en FPGAs (tanto de los fabricantes de Xilins, como de Altera).

El trabajo realizado en este campo permitió el desarrollo de un “módulo master i2c”, este módulo permite realizar la configuración y lectura de datos de un conversor analógico digital (ADS1115) desde una FPGA.

Abstract

This monograph describes the works and obligations that were carried out in the research stay, which consisted first in supporting the design of a PCB board, which will be used for the creation of a small servomotor. This will be implemented in the construction of a robotic hand able to carry out manipulation tasks; this board was designed in software Eagle, of Autodesk, which provides a free license for three years, in addition to having good compatibility with Solidworks software.

During the course of the entire internship, we attended the discipline of "Bioinspired Algorithms for Engineering", with which we were able to contribute in the realization of a scientific article, which contains different types of bioinspired algorithms to perform the tuning of a controller. Impedance. The base algorithms that were worked were PSO (Particle Swarm Optimization), and DE (Diferential evolution), applying artificial diversity methods to improve the efficiency of the base algorithms, the algorithms are compared applying statistical analysis, hypothesis tests and the response of the controller of each one.

Finally, we worked in the area of programming ARM circuits in chips, for which a discipline called "Digital Systems" was taken, with this acquired knowledge in the area of hardware description language called VHDL (from English " VHSIC Hardware Description Language "), used to synthesize circuits in FPGAs (both of the manufacturers of Xilins, as of Altera).

The work carried out in this field allowed the development of a "master i2c module", this module allows configuring and reading data from a digital analog converter (ADS1115) from an FPGA.

Tabla de Contenido

Índice de Figuras:

Índice de tablas

1 Introducción

1.1 Planteamiento del problema

Actualmente en la Universidad de Pamplona no se ofrece un curso relacionado con algoritmos bionispirados, los cuales poseen un gran campo de aplicación (como por ejemplo en la sintonización de controladores, en redes neuronales) y de investigación (permitiendo la creación de nuevos convenios con diferentes Universidades para realizar trabajos en conjunto), esto genera que la universidad quede un poco rezagada en esta área del control inteligente. Además de estas técnicas de control también se precisa de manera urgente trabajar con nuevas tecnologías de fabricación de circuitos impresos (PCB), ya que actualmente las máquinas se encuentran dañadas y se recurre a técnicas antiguas como el quemado de placas de circuito impreso, generando una contaminación considerable (ya que por lo general el ácido utilizado se descarta por la tubería de agua corriente) y una calidad regular. ¿Podremos realmente aprender a trabajar con algoritmos bioinspirados para sintonizar controladores y con esto permitir que nuestra universidad este a la vanguardia en el área de control? Y ¿Se logrará como estudiantes adquirir nuevos conocimientos para la fabricación de PCBs implementando nuevas tecnologías?

Por otro lado, en la Universidad de Pamplona poco se trabaja con placas que poseen arquitectura ARM (FPGA) en el programa de Ingeniería Mecatrónica, esto conlleva a que el programa quede enfrascado en la utilización de tecnologías simples como Arduino, Tiva, etc. ¿Será que como estudiantes podremos incentivar el uso de nuevas tecnologías vanguardistas en la Universidad de Pamplona tal como FPGAs, las cuales son utilizadas en el ámbito industrial y científico (como en la NASA)?

1.2 Justificación

Con la presente pasantía de investigación se obtuvieron conocimientos básico-Medio en el área de ARM en Sistemas en chip (Programación de FPGA), sentando una base sólida para que los estudiantes de la Universidad de Pamplona continúen con investigaciones semejantes, generando competitividad internacional por parte de los estudiantes del programa de Ingeniería Mecatrónica, permitiéndole a los mismos realizar intercambios a otras universidades (como por ejemplo a la Universidad Nacional de Brasilia).

Por otro lado, los conocimientos adquiridos en el área de algoritmos bioinspirados, permitieron la inserción de un nuevo campo de investigación en el área de control inteligente (ya que los algoritmos bioinspirados pueden ser utilizados en la sintonización de controladores), esto conlleva a que la universidad diversifique aún más sus áreas de investigación logrando competitividad internacional

Por último, el trabajo realizado en la fabricación de circuitos impresos (PCB) permitirá contribuir en la fabricación del prototipo de un pequeño servomotor.

1.3 Delimitación

Objetivo general

- Participar como pasante de investigación apoyando actividades para el desarrollo de un proyecto sobre el uso de un algortimo bioinpirado para sintonizar un controlador de impedancia y realizar pruebas estadísticas para comparar los respectivos resultados.

Objetivos específicos

- Adquirir conocimientos en programación ARM (c++) en Sistemas en Chip (ZYNQ7-Soc)
- Apoyar el desarrollo de placas electrónicas PCB, aplicando técnicas modernas empleadas en la Universidad de Brasilia.
- Acompañar al investigador al impartir la disciplina de algoritmos bioinspirados para ingeniería
- Realizar las actividades asignadas para colaborar en el desarrollo del proyecto sobre el uso de un algoritmo bioinspirado para sintonizar un controlador de impedancia.
- Participar en la redacción de un artículo científico para publicar en la conferencia sobre sistemas bioinspirados (CEC 2019) o similares.

1.4 Metodología

Las metodologías utilizadas en esta monografía son tres, la primera para el diseño de circuitos impresos, la segundo para el trabajo de algoritmos bioinspirados para ingeniería y la tercera para la realización del módulo master i2c.

- La primera metodología se utilizó para el diseño de una placa de circuito impreso, para esto se utilizó el software Eagle en su versión 9.1.0, la realización del circuito supuso una investigación a priori de los componentes que se utilizarían para realizar las respectivas conexiones, además de saber que tanta intensidad de corriente pasaba por cada una de las pistas, para que de esta manera se calculará el grosor de las mismas mediante la utilización de recursos disponibles en páginas web tal como se muestra en la sección tres.
- La segunda metodología s utilizó para la realización del trabajo de algoritmos bioinpirados aplicados a ingeniería, en donde se utilizó el software de Matlab en su versión 2018Ra(las licencias de Matlab los proporcionó la Universidad de Brasilia), en el cual se realizaron los trabajos de optimización (minimización) del error ponderado para un controlador de impedancia, la función para realizar el trabajo de minimización ya estaba lista al momento de utilizar los algoritmos, los algoritmos utilizados para realizar la sintonización del controlador de impedancia fueron el OPSO y el ODE, los resultados se validaron valiéndose de un análisis estadístico, el teste de hipótesis el cual rechazo la hipótesis nula, permitiendo comparar los

algoritmos utilizando el valor de la media aritmética y las respuestas de los controladores de impedancia.

- La última metodología se utilizó para la creación del módulo master i2c, para ello se trabajó con el software de Vivado en su versión 2018.1, se utilizó máquinas de estado para su realización, la cual se simuló (en el mismo software de Vivado).

1.5 Estructura del documento

Esta monografía está estructurada de la siguiente forma, en el capítulo dos se exponen los laboratorios y equipos con los cuales contaba la Universidad nacional de Brasilia, en el tercero los fundamentos teóricos que respaldan el trabajo de investigación, al igual que sus trabajos más recientes, en el capítulo cuatro se expone el diseño y fabricación de la PCB, en el capítulo cinco se muestra el trabajo realizado con los algoritmos bioinspirados en la sintonización de un controlador de impedancia, en el capítulo seis se observa el trabajo realizado en el área de Circuitos on-Chip, por último se aprecia la discusión de los resultados, las conclusiones a la cual la investigación llega, los trabajos futuros y la respectiva bibliografía.

2 Informe de actividades desarrolladas en la estancia de investigación

La estancia de investigación se dio en la Universidad nacional de Brasilia (UnB), en Brasilia Brasil, esta estancia duró 4 meses, comenzando el 8 de agosto del 2018 y terminando el 12 de diciembre del 2018.

Estando en la UnB se entablo contacto con el profesor Sergio Pertuz Mendez, y con los profesores Daniel Muñoz Arboleda, Carlos Humberto Llanos Quintero y Ricardo Pezzuol Jacobi, estos dos últimos encargados de la asignatura de "*Sistemas Digitais Reconfiguráveis e Aplicações* " y los tres primeros encargados de la asignatura de "*Sistemas Bioinspirados Aplicados a Engenharia*", con estos profesores se resolvieron dudas para la realización de cada una de las actividades propuestas, su ayuda y orientación fueron esenciales para la correcta finalización de la estancia de investigación.

Las asignaturas a las cuales se presentó el estudiante eran de maestría por lo cual los profesores propusieron que, si él quería continuar con la maestría, trabajará duro para pasarlas y con eso conseguir la respectiva homologación, cada metería poseía cuatro créditos (se tienen que hacer 24 créditos para realizar la maestría). Adicionalmente me ofrecieron una beca la cual era de 1200 R, aproximadamente un millón de pesos colombianos, la cual se iniciará en agosto

Además, cabe decir que las materias de maestría se pasaron exitosamente logrando buenas calificaciones (en la de algoritmos bioinspirados se logró tener una nota equivalente a 4,28), esto conlleva a la homologación de las dos asignaturas al momento de realizar la maestría. Las notas se muestran a continuación.

NotaFinal

Nota Final SBIE 2018-2

Matricula	Listas			Apresentação Algoritmo	Projeto Final			Nota Final	Menção
	Lista 1	Lista 2	Lista 3		Desenv	Analise resul	Apresentacao		
18/0061739	5.15	5.57	7.70	8.00	9.00	10.00	10.00	7.83	MS
14/0056289	3.86	6.90	5.50	7.50	9.00	10.00	8.00	7.16	MS
	5.53	8.12	7.85	7.50	8.00	7.00	8.00	7.42	MS
17/0196330	7.38	9.00	8.85	6.50	9.00	8.00	7.00	7.96	MS
*	8.18	9.10	8.20	8.50	10.00	8.00	8.00	8.56	MS
	2.48	7.82	6.80	9.00	8.00	5.00	5.00	6.32	MM
	0.00	0.00	0.00	6.50	0.00	0.00	0.00	0.98	II
	9.00	9.75	8.60	8.50	9.00	10.00	9.00	9.11	SS

NOTA: 06.*(MediaListas e Apresentacao) + 0.4*Projeto

Figura 1: Notas de la asignatura de algoritmos bioinspirados para ingeniería.

2.1 Laboratorio y equipos

La UnB poseía varios laboratorios de robótica, automatización y manufactura, uno de los laboratorios al cual se obtuvo acceso fue el que poseía tres robots, los cuales eran: uno serial, uno paralelo y un robot humanoide.

Estos robots estuvieron a disposición del estudiante en todo momento para realizar trabajos de control, lamentablemente debido a los objetivos específicos de las pasantías de investigación no se utilizaron.

Los robots de este primer laboratorio se encuentran a continuación.

Figura 2:Robot paralelo ABB

Fuente: Autor

Este robot paralelo se utilizaba para labores de investigación en el área de maestría y pregrado, este robot se encontraba en un salón junto a los robots Nao y antropomórfico ABB.

Figura 3: Robot Nao

Fuente: Autor.

El robot Nao se encuentra en la UnB para desarrollar trabajos de investigación, este cuenta con su propio software para realizar la labor de programación.

Figura 4: Robot antropomórfico ABB

Fuente: Autor.

Este robot contaba con su propio módulo para realizar la parte de control, además de una cámara ubicada en el efector final, la cual permite trabajar en el área de visión artificial.

Figura 5: Estudiante pasante en el laboratorio de la UnB

Fuente: Autor.

Adicionalmente se podía utilizar otro laboratorio de nombre Graco, en donde se tenía a disposición plantas didácticas de automatización de procesos industriales, brazos robótico soldadores, máquinas CNC para fresado entre otros. Estas máquinas son expuestas a continuación.

Figura 6:Planta didáctica de automatización

Fuente: Autor

Esta planta didáctica de automatización de procesos industriales permitía a los estudiantes trabajar en el área de PLCs.

Figura 7: Plata didáctica de automatización

Fuente: Autor

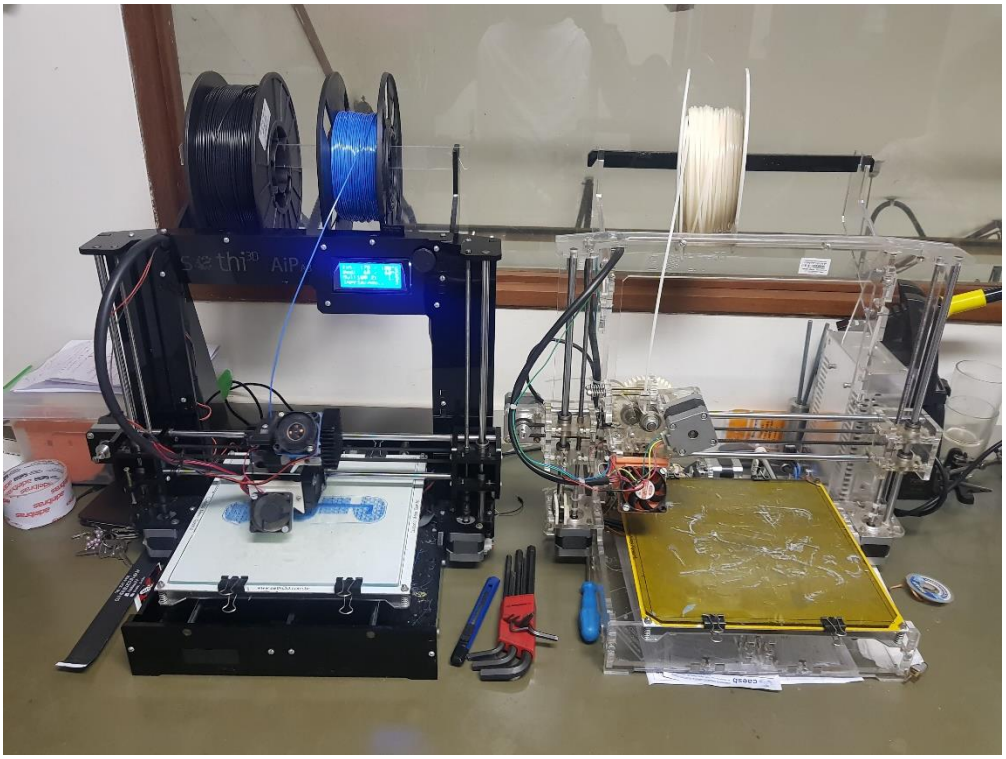

Figura 8: Impresoras 3D

Fuente: Autor

Estas impresoras 3D, son de una empresa Brasilera, estuvieron siempre a disposición para el diseño y construcción de piezas en PLA y ABS.

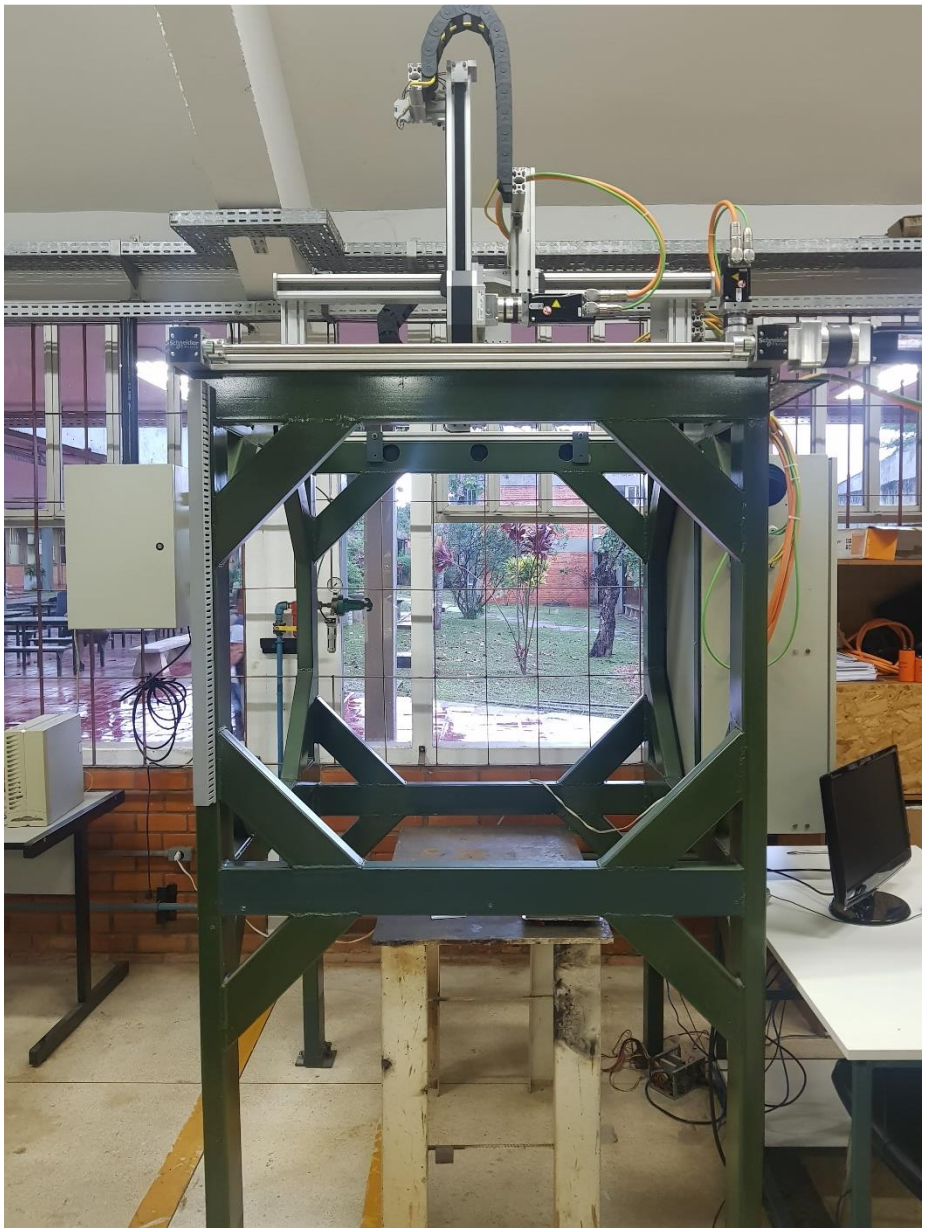

Figura 9: Máquina CNC

Fuente: Autor

Figura 10: Brazo de soldadura

Fuente: Autor

Este brazo para soldadura, se utiliza para realizar ciertos trabajos en el área de la metalurgia donde se precisa de una buena precisión, estaba a disposición de los estudiantes de pregrado o maestría que quisieran realizar pruebas con él.

Figura 11: Brazo robótico para fresado

Fuente: Autor

Este brazo se utilizaba para realizar tareas de fresado, el cual poseía una broca en su efector final además de tener una configuración antropomórfica.

Figura 12: Máquina CNC para fresado

Fuente: Autor

Esta máquina permite realizar trabajos en el área de fresado o grabado mediante control numérico computacional.

Figura 13: Máquina CNC

Fuente: Autor

Figura 14: Laboratorio Graco UnB

Fuente: Autor

Por último, se observa el laboratorio de forma general, donde se pueden apreciar ciertos equipos que anteriormente fueron descritos.

Las actividades que se realizaron en esta estancia de investigación fueron:

2.2 Trabajos relacionados con ARM(c++) en ZINQ7-SOC.

Esta actividad se realizó a los largos de los 4 meses ya que fueron necesarios adquirir conocimientos en el área de síntesis de circuitos y programación en VHDL, para lo cual se asistió a la asignatura de “Sistemas Digitais Reconfiguraveis e Aplicações”, esto debido a que la Universidad de Pamplona no transmite estos conocimientos en las aulas de aprendizaje.

Para esta tarea se comenzó con conocimientos básicos, como realizar sumadores, restadores, realizar memorias RAM, hasta llegar a trabajar máquinas de estado, este último conocimiento fue esencial para la realización de la actividad propuesta desde el comienzo de la pasantía, la cual fue la realización de un módulo en FPGA capaz de configurar el módulo ADS 1115 el cual es un conversor analógico digital.

Este conversor se utilizará en una mano robótica biomimética para calcular la cantidad de corriente que pasa por cada uno de los dedos.

2.3 Desarrollo de placas electrónicas PCB.

Esta actividad fue una las primeras que fueron asignadas (se asignó al día siguiente de la llegada a la UnB el 9 de agosto del 2018), la cual consistía en el diseño de una placa de circuito impreso, la cual se realizó en el software de Eagle de Autodesk el cual proporcionaba una licencia gratuita de 3 años para estudiantes, la cual fue aprovechada para la realización de esta tarea.

Esta placa de circuito impreso se implementará en la creación de un servo motor pequeño, esto con el fin de mejorar una mano robótica bioinspirada con motores propios.

Esta tarea finalizó el día 20 de septiembre del 2018, con lo cual los diseños fueron enviados a china para su respectiva fabricación, esto debido a que la UnB no contaba con la maquinaría necesaria para su creación (los grosores de las pistas eran muy pequeñas y las máquinas no tenían la resolución necesaria).

Las maquinas con las que cuenta la UnB son:

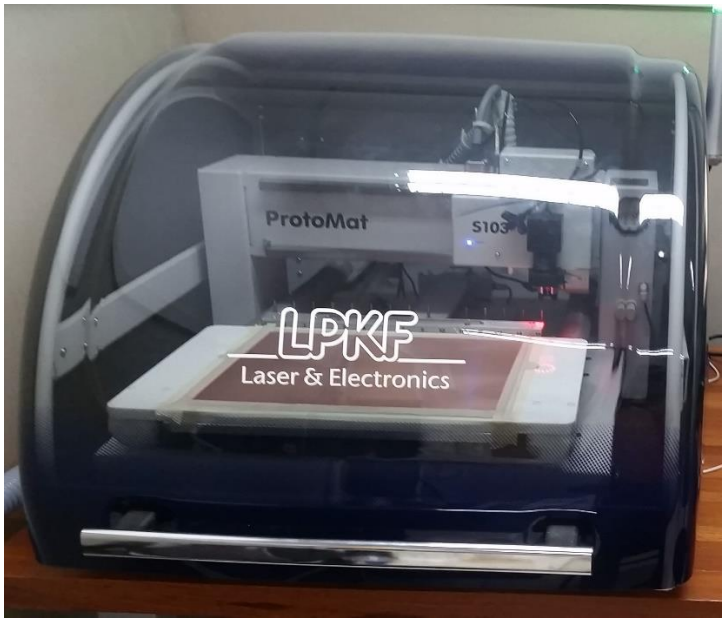

Figura 15: Máquina para el ruteo de placas de circuito impreso.

Fuente: Autor.

Esta máquina permite realizar las pistas en las placas PCB, lastimosamente la máxima resolución que se alcanzó al utilizarla fue de apenas 0.2mm, y el diseño requería que fueran de 0.1mm, por eso se optó por enviar las placas para ser fabricadas en china.

Figura 16: Máquina para la colocación de los componentes electrónicos en la PCB.

Fuente: Autor.

Esta máquina sirve para posicionar los componentes con una precisión milimétrica, la cual debido al mecanismo que posee permite desplazar los componentes dentro de la PCB en el orden de los milímetros, logrando una buena exactitud al momento de organizar los componentes. Además, la pantalla que posee permite ampliar el rengo de visión, observando con una mejor resolución donde quedaran los componentes.

Figura 17:Maquina utilizada para soldar los componentes electrónicos a la PCB.

Fuente: Autor.

Una vez se tienen los componentes electrónicos ubicados en la placa PCB se dispone a su respectiva soldadura, para eso se utiliza el ProtoFlow, el cual es básicamente un horno que se calienta fundiendo el estaño a la placa de cirucitos impresos.

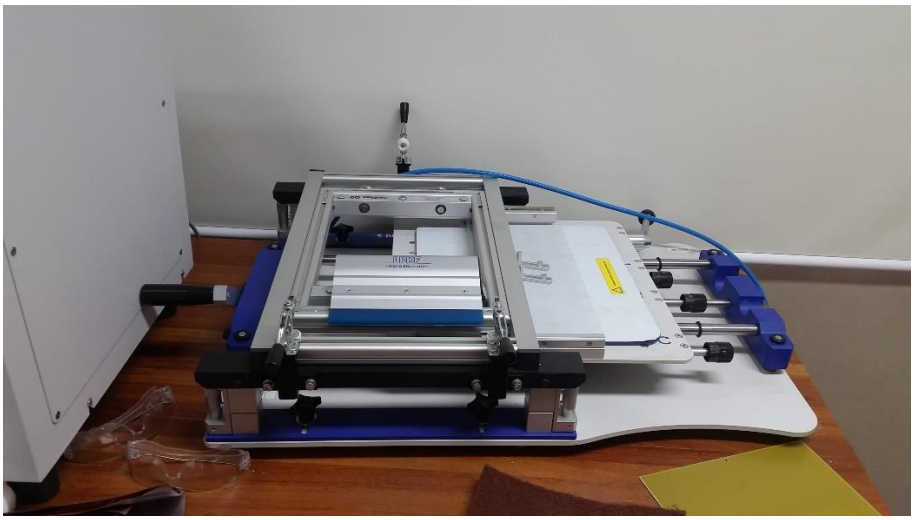

Figura 18:Máquina utilizada para la capa anti óxido de las PCB.

Fuente: Autor.

Por ultimo esta máquina añade una capa antioxidante a la PCB por donde se encuentra el cobre, esto con el fin de evitar el deterio de la misma al pasar los años.

2.4 Asignatura de algoritmos bioinspirados para ingeniería.

Esta asignatura se tomó al comienzo del semestre 2018-2, con la cual se adquirieron conocimientos en el área de algoritmos bioinspirados, los cuales se pueden implementar en diferentes aplicaciones que pueden ser sintonizaciones optimas de controladores clásicos tal como el PID o para ajustar pesos sinápticos en una red neuronal.

Ha esta asignatura se asistía una vez por semana con una intensidad horaria de cuatro horas, esta asignatura sirvió para la realización del proyecto final, el cual consistía en comparar los resultados de varios algoritmos al momento de realizar la sintonización del controlador de impedancia.

En esta asignatura se aprendió a implementar varios algoritmos de optimización tal como: la optimización por enjambre de partículas, (Particle Swarm Optimization en inglés), optimización por colonias de abejas (Artificial Bee Colony), el algoritmo de optimización de Evolución Diferencial (Differential Evolution en inglés), y el algoritmo de las ballenas, además de implementar diversas técnicas para realizar estos algoritmos más optimos y eficientes tal como el aprendizaje por oposición o las técnicas de repulsión y atracción.

Ademas de aprender optimización mediante algoritmos bioinspirados, se adquirieron conocimientos en el área de análisis estadísticos, estos conocimientos son de gran importancia al momento de realizar comparaciones, en este caso se utilizaron para la comparación de varios algoritmos bioinspirados logrando escoger un algoritmo ganador, este análisis se puede aplicar a cualquier comparación los cuales pueden ser la comparación de sensores para escoger el mejor.

Los conocimientos en análisis estadísticos fueron: conocer de primera mano si los datos correspondían a una distribución normal, esto se realizaba mediante el test de Kolmogorov-Smirnov el cual se implementaba directamente en MATLAB, seguidamente dependiendo del test de normalidad se utilzaban test paramétricos o no paramétricos. El test paramétrico era el de la ANOVA el cual si el valor de la probabilidad era menor al 0,001 se podía afirmar con un 99,999% de seguridad que las muestras venían de diferentes poblaciones logrando con esto las comparaciones de las mismas mediante su mediana (se utilizaba la mediana ya que era menos propensa a variaciones en comparación de la media).

2.5 Sintonización del controlador de impedancia y comparación estadística de los diferentes algoritmos utilizados.

El trabajo final de la asignatura de “Algoritmos bioinspirados para ingeniería”, fue la realización de varias sintonizaciones de un controlador del cual se hablará más adelante en el marco teórico.

En esta aplicación se utilizaron dos algoritmos canónicos los cuales fueron PSO (Particle Swarm Optimization) y DE (Differential Evolution), a estos algoritmos canónicos se les aplico la técnica de aprendizaje por oposición, creando por consiguiente dos algoritmos que evitaban de una manera más óptima la convergencia prematura, estos algoritmos fueron OPSO y ODE.

De cada uno de los algoritmos se realizaron 20 experimentos para poder ejecutar la comparación estadística, estos 20 experimentos se realizaron en un servidor de la Universidad de Brasilia el cual poseía buenas características para la correcta implementación de los algoritmos (poseía 32 gigas de memoria RAM), aun así con estas características cada algoritmo demoro cerca de una semana en realizar los 20 experimentos, finalizando los experimentos a finales de diciembre del año 2018.

Una vez obtenidos los 80 experimentos se dispuso a realizar el análisis estadístico, para este análisis primero se corroboro la normalidad de las muestras mediante el test de Kolmogorov-Smirnov, seguidamente se utilizó el test de la Anova para corroborar que las muestras provenían de diferentes poblaciones, logrando con esto por realizar las respectivas comparaciones. Con esto se logró concluir que con los parámetros que se implementaron los cuatro algoritmos el más eficiente resulto ser el ODE.

2.6 Realización del artículo científico.

Una vez terminado el proyecto de sintonización del controlador de impedancia mediante algoritmos bioinspirados, los resultados obtenidos fueron utilizados para la construcción de un artículo científico, el cual se escribió en inglés y español, este archivo es un anexo a esta pasantía de investigación.

3 Marco teórico

A continuación, se presenta la base teórica que respalda la pasantía de investigación

3.1 Placas de circuito impreso (PCB)

Las placas de circuito impreso (del inglés Printed Ciruit Board), son comúnmente utilizados en el ámbito de la electrónica, por lo general están hechas de resina de epoxy, fibra de vidrio (en la parte del aislante) [1]y de cobre.

Para realizar la fabricación de las placas de circuito impreso, se precisa primeramente realizar su respectivo diseño en algún software de computadora (el cual puede ser Proteus, Eagle, etc), posteriormente el diseño puede ser impreso para realizar el proceso de quemado de la baquela (el cual es un método antiguo y ambientalmente desagradable).

La alternativa al procedimiento anterior es trabajar con robots cartesianos para crear los caminos de la placa de circuito impreso mediante la utilización de brocas o lazer, estas máquinas utilizan el control numérico computacional [2], el cual básicamente le dice a la herramienta la trayectoria que se desea realizar.

3.1.1 Software Eagle

Autodesk EAGLE es un software de automatización de diseño electrónico(EDA), permitiendo que los diseñadores de placas de circuito impreso (PCB) conecten a la perfección diagramas esquemáticos, ubicación de componentes, enrutamiento de PCB, además la empresa Autodesk proporciona una licencia de estudiante de 3 años de manera gratuita. [3]

Este software permite exportar la placa terminada en un archivo con extensión STEP, permitiendo su utilización en diferentes softwares de diseño CAD, para que en trabajos futuros se realice el diseño CAD del motor con la placa PCB y sus respectivos componentes electrónicos y mecánicos.

Adicionalmente la facilidad de conversión a código G para la utilización en la maquina LPKF (el robot cartesiano donde se fabrican las placas PCB en la universidad de Brasilia), hacen de este software el más idóneo para trabajar.

3.2 Algoritmos Bioinspirados.

Son algoritmos que se basan en la inteligencia colectiva de algunos animales tales como cardúmenes de peses, parvadas de aves, colonias de hormigas, colonias de abejas, etc. O en la evolución de las especies.

3.2.1 Inteligencia Colectiva.

La inteligencia colectiva es básicamente la que emerge cuando varios agentes simples se juntan para un propósito específico [4], cabe destacar que una agente es cualquier cosa que pueda percibir su entorno por medio de sensores e interactuar con este por medio de actuadores [5] ,con esta definición se puede deducir que los agentes por si solos son bastante ineficientes, pero al juntar varios de los mismos

se logra generar una inteligencia capaz de sobrevivir a depredadores (tal como el cardumen de peces), de recolectar comida (tal como hormigas o abejas), y crear arquitecturas inmensas tales como una colonia de abejas o de hormigas.

Las metaheurísticas se conocen como métodos diseñados para la solución de problemas difíciles de optimización combinatoria [6]. Las metahurísticas que son ampliamente utilizadas en los estudios de optimización son: los algoritmos genéticos, optimización basada en colonias, la inteligencia de enjambre, entre otros.

3.2.2 Algoritmo PSO

PSO (del inglés Particle Swarm Optimization), es un algoritmo basado en la mejora de la población, desarrollado por Eberhart y Kennedy en 1995, este algoritmo fue inspirado en el comportamiento de los enjambres de pájaros y peces [7].

Esta optimización consiste en inicializar las partículas o soluciones de una manera aleatoria, las cuales tienen una velocidad (ellas no poseen ni masa ni volumen), permitiendo que las partículas recorran el espacio de búsqueda, procurando la solución óptima.

Cada partícula guarda la mejor solución a la cual ella ha llegado hasta el momento, guardándolo en una variable que puede ser, p_{mejor}. Adicionalmente el algoritmo procura otro valor, llamando g_{mejor}, el cual es el valor de la mejor posición de todo el enjambre de partículas (la mejor solución del algoritmo).

El movimiento realizado en el espacio de búsqueda es dado por la ecuación de velocidad, la cual es la ecuación 1 y la actualización de las posiciones es dado por la ecuación 2, la cual depende directamente de la velocidad.

Con el paso de las iteraciones el valor de velocidad decrece logrando con esto que las partículas lleguen al punto g_{mejor}, permitiendo la convergencia del algoritmo [7].

Las ecuaciones que se utilizan para implementar el algoritmo de PSO clásico son:

$$V_i^{(t+1)} = w * V_i^{\,t} + C1 * \overbrace{rand_1 * \left(pmejor_i - X_i^{\,t}\right)}^{Componente\ cognitivo\ ind} + C2 * \overbrace{rand_2 * (gmejor - X_i^{(t)})}^{Componente\ social} \quad (1)$$

$$X_i^{(t+1)} = X_i^{(t)} + V_i^{(t+1)} (2)$$

Donde X_i y V_i son la posición y la velocidad de la partícula i, p_{mejor} y g_{mejor}, son la mejor posición para la partícula i_{th} y la mejor posición global encontrada por todo el enjambre de partículas, $rand_1$ y $rand_2$ son números generados aleatoriamente con un rango de [0,1], por último $C1$ y $C2$, son constantes que multiplican el componente cognitivo y el componente social [8].

Cabe destacar que hay una restricción en $C1$ y $C2$, la cual es $C1 + C2 > 4$ [9]. Adicionalmente "w" es un factor de inercia propuesto por Shi and Eberhart [10], este factor de inercia no aparece en la primera versión del algoritmo PSO.

El seudocódigo quedará de la siguiente manera:

Donde S es el tamaño de la población, N es la dimensionalidad del problema, x_{max}, x_{min}, son los límites de la función custom, v_{max}es la velocidad máxima alcanzada por cada partícula, Max_{iter}, es el máximo de iteraciones que se ejecuta el algoritmo, threshold es el error aceptable para detener la optimización.

Algoritmo 1 Seudocódigo para el algoritmo PSO

1: **función** PSO-básico(*S,N,c1,c2,*Max_{iter}*, threshold*)
2: Inicializa el enjambre con X_{max} y con X_{min}
3: *iter*=1;
4: **Mientras** *iter* < Max_{iter}
5: **para i de** 1 **hasta** S
6: **sí** $f(x_k) \leq f(x_{ik})$ **entonces**
7: $y_{ik} \leftarrow x_k$**;**
8: $Limit_k = 0$;
9: **De lo contrario**
10: $Limit_k = Limit_k + 1$;
11: **Fin sí**
12: calcula y_s usando el *S* fitness valor $f(y_{ik})$;
13: **para i de** 1 **hasta** S
14: **para j de** 1 **hasta** N
15: $v_{ij}^{(t+1)} \leftarrow wv_{ij}^{(t)} + c_1U_{1j}(y_{ij}^{(t)} - x_{ij}^{(t)}) + c_2U_{2j}(y_{sj}^{(t)} - x_{ij}^{(t)})$;
16: $x_{ij}^{(t+1)} \leftarrow x_{ij}^{(t)} + v_{ij}^{(t+1)}$;
17: **Fin para;**
18: **Fin para;**
19: *iter=iter+1*
20: **Fin mientras;**
21: **Retorna** Posición de la mejor partícula fit x y su valor $f(x)$
22:**Fin de la función**

Figura 19: Algoritmo PSO clásico.

Fuente:autor

Algunos trabajos realizados con el algoritmo PSO son por ejemplo, la utilización del mismo para predecir la sobreexplotación el cual es un factor importante en el campo de minería y construcción de túneles [11], en otro trabajo se utilizan para determinar los parámetros de tres modelos diferentes de condensadores suministrados por voltajes con distorsiones armónicas [12], además estos algoritmos también se utilizan para hallar las constantes optimas en un controlador, en [13] lo utilizan para encontrar las constantes ideales en un controlador PI implementado en una turbina eólica, además de ser utilizado para encontrar los pesos sinápticos en redes neuronales.

3.2.3 Algoritmo DE

El algoritmo DE (del inglés Differential Evolution), es un potente algoritmo evolutivo el cual fue desarrollador en el marco del algoritmo genético e inspirado por el método Nelder-Mead Simplex [14]. Es un algoritmo relativamente simple pero poderoso, basado en la población para la optimización de problemas. El crea nuevos individuos mediante la evolución de los mismos utilizando mutaciones [15]**.**

En el algoritmo DE los vectores actuales $\vec{X}$, son llamados de vectores objetivos, los cuales se utilizan para generar nuevos vectores, los cuales se llaman vectores ruidosos $\vec{V}$, mediante la utilización de la ecuación tres.

$$\vec{V} = \vec{X}_{r3} + F * \left(\vec{X}_{r2} - \vec{X}_{r1}\right) \text{ (3)}$$

Donde $Xr1$ es el primer individuo, $Xr2$ el segundo individuo, $Xr3$ el tercer individuo, los individuos son diferentes entre sí, y F es el factor de mutación el cual está en el rango [0 2].

Con esto se busca hacer un vector mutante $\vec{U}$, a partir de una operación de crossover (ver la ecuación número cuatro), sí el vector mutante tiene un mejor valor en la función Custom que el vector objetivo, este primero se sobre escribe en el vector objetivo, realizando una operación de selección donde predomina el mejor.

$$\vec{U} = \begin{cases} \vec{V}, & Rand() \leq C \\ \vec{X}, & Rand() > C \end{cases} \qquad \text{(4)}$$

Donde C es la probabilidad de realizar el crossover y $Rand()$ es un número aleatorio comprendido entre [0 1].

En el algoritmo DE primeramente se inicializan los parámetros de número de población M, total de dimensiones N , la constante de mutación o constante de peso diferencial F, el factor de crossover C, además del máximo de iteraciones $iter_{max}$.

El seudocódigo queda de la siguiente manera:

Algoritmo 2 Seudocódigo para el algoritmo DE

```
1: función DE-básico (M,C,F,range,f)
2:     x ← aleatorio (range, M);
3:     fit_x ← f(x);
4:     Mientras no se cumplan los criterios de parada Hacer
5:         para i de 1 hasta M
6:             v_i ← Mutación (x_i, F);
7:             u_i ← Crossover (x_i, v_i, CR);
8:         Fin para;
9:         fit_u ← f(u);
```

10: **Para i de** 1 **hasta** M
11: **Sí** $fit_u(i) < fit_x(i)$ **Entonces**
12: $x_i \leftarrow u_i$;
13: $Limit_k = 0$;
14: **De lo contrario**
15: $x_i \leftarrow x_i$;
16: $Limit_k = Limit_k + 1$;
17 **Fin sí;**
18: **Fin para;**
19: **Fin mientras;**
20: **Retorna** Posición de la mejor partícula x y su valor en $f(x)$

Figura 20: Algoritmo DE clásico

Fuente: Autor

Algunos trabajos con este algoritmo de evolución diferencial ayudaron a optimizar el consumo de energía de las redes de acceso inalámbrico en Gante, Bélgica [16], también utilizado para el área de control tal como en [17]**,** y en la optimización de horarios de trenes como en [18].

3.3 Diversidad Artificial

El uso de la diversidad artificial, permiten alternar los algoritmos entre dos etapas, exploración y explotación [19], logrando que una mayor probabilidad de alcanzar el óptimo global en la función Objetivo la fórmula para calcular la diversidad de presenta a continuación.

$$Div = \frac{1}{|S||L|}\sum_{i=1}^{S}\sqrt{\sum_{j=1}^{N}(pij - Pj)^2} \quad (5)$$

Donde S es el número de partículas o el tamaño de la población, N es la dimensionalidad del problema, L es la diagonal de todas las dimensiones, p_{ij} es la partícula i en la dimensión j, y p_j es el promedio de la dimensión en la cual se esta hallando la diversidad [20]**.**

3.3.1 Técnica de Oposición

Básicamente esta técnica permite realizar una búsqueda en el lugar opuesto donde se encuentra, siempre cuando la solución no mejore (cierto número de iteraciones) y aun no se ha alcanzado el treeshold especificado, lo cual evita una convergencia prematura en un mínimo local, la ecuación que describe este método es la siguiente:

$$X = a + b - X \quad (6)$$

Donde a y b son los límites menores y mayores de nuestra función Custom, y X es la posición de la partícula [21].

3.4 Hardware reconfigurable

El hardware reconfigurable proporciona un buen desempeño entre ieficiencia en la implementación y flexibilidad, esto es porque el hardware reconfigurable combina la capacidad de programar después de su fabricación con el estilo de computación espacial (paralelo), de programaciones específicas utilizando circuitos (ASICs), el cual es más eficiente en comparación a la computación temporal (secuencial) [22].

3.4.1 FPGAs(Field Programable Gate Array)

Es un dispositivo semiconductor en el cual su función puede definirse después de su fabricación, el dispositivo FPGA permite programar características y funciones del producto, al igual que reconfigurar su hardware para ser implementado en funciones específicas, incluso después de ser instalado en el campo de acción [23].

En general la arquitectura de la FPGA es descrita en la Figura 4 la cual se ha simplificado, en la figura se tiene la simplificación del modelo virtex 5 de la empresa Xilin donde los bloques lógicos programables son llamadas CLB, los CLB están hechos de dos placas de cilicio (uno de tipo L, y otro de tipo L o M) [24].

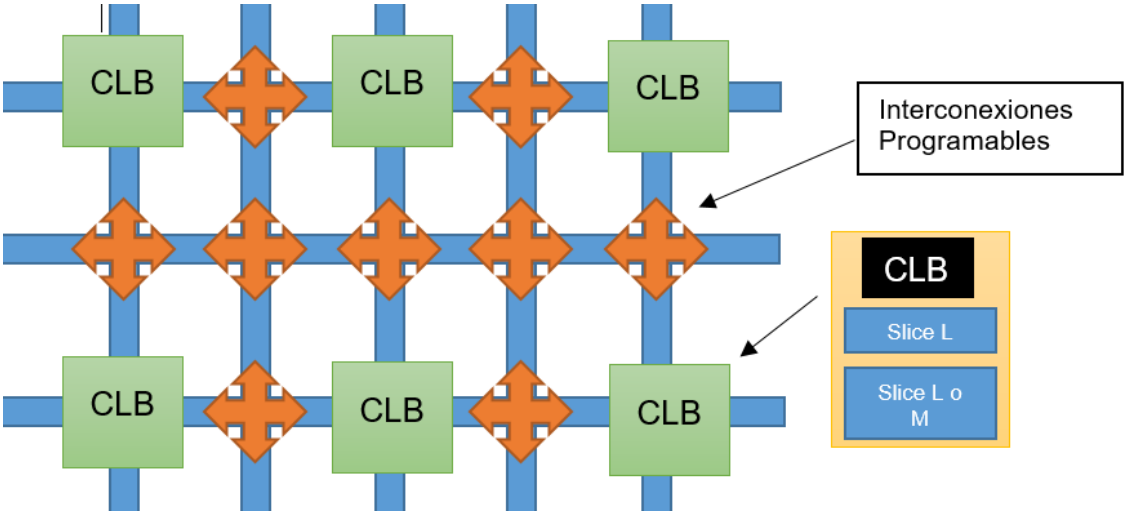

Figura 21: Estructura interna simplificada de una FPGA.

(Fuente: Autor)

Las FPGAs presentar ciertas ventajas al ser reconfigurables, bastantes robustas, trabajar en paralelo lo cual las hace más rápidas que los microcontroladores, actualmente su precio ha disminuido lo cual las hace de bastante fácil acceso.

Aunque también presentan ciertas desventajas las cuales son, un consumo más elevado de potencia [25]**.**

Algunas de las ultimas investigaciones con FPGAs, han sido implementar en ellas controladores [26], redes neuronales [27], o utilizadas en el internet de las cosas IoT [28]**.**

3.5 Controlador de impedancia.

El controlador de impedancia es uno de los más utilizados para el control dinámico, ya que permite convertir el sistema dinámico en un sistema amortiguado de masa, resorte de segundo orden con una rigidez de amortiguación y masa deseada [29].

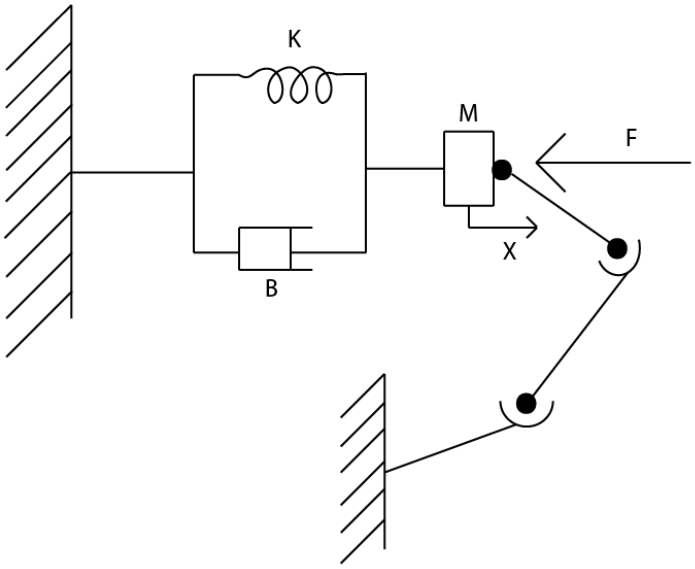

Figura 22: Sistema masa-resorte-amortiguador que representa el comportamiento del control por impedancia

(Fuente: Pertuz,2016)

La fuerza que actúa sobre el sistema masa-resorte-amortiguador se puede expresar de la siguiente manera:

$$\Delta F = M_D \Delta \ddot{X} + B_D \Delta \dot{X} + K_D \Delta X \quad (8)$$

Donde M_D, B_D y K_D son los parámetros objetivos de la impedancia [30], donde M_D representa el valor de la masa, B_D es el coeficiente viscoso del amortiguador, $K_D$es la constante del resorte, X es la posición del robot, y $\dot{X}$, $\ddot{X}$ son las derivadas de X [31]**.**

Algunos trabajos en este campo son los siguientes, en [32] el control de impedancia fue utilizado para regular la fuerza utilizada en una pinza al momento de tomar un objeto, y en [33] se utilizada el control por impedancia para controlar las articulaciones en un robot con patas, por último, en [34] se utiliza un controlador de impedancia para guiar la trayectoria de una grúa hidráulica.

3.6 Descripción de las actividades

En esta sección se describen las actividades realizadas en la estancia de investigación y el grado de cumplimiento de alcanzarón.

3.6.1 Desarrollo de placas electrónicas PCB

En esta actividad el objetivo era construir una placa PCB, la cual sirviera para la creación de un servomotor, esto con el fin de construir servomotores pequeños para aplicarlos en el mejoramiento de una mano robótica.

Esta actividad se desarrolló hasta el diseño de la placa PCB, debido a la complejidad de la misma la construcción fue realizada en china. El diseño y los componentes utilizados se describen en la sección 3.

3.6.2 Sintonización del controlador de impedancia utilizando algoritmos bioinspirados.

Este trabajo acoplo varias actividades, las cuales fueron, asistir a la asignatura de “sistemas bioinspirados para ingeniería” y el trabajo de sintonización para el controlador de impedancia, de este trabajo se obtuvo un artículo científico el cual será postulado para congresos internacionales.

Esta actividad se logró completar satisfactoriamente, logrando la óptima sintonización del controlador de impedancia mediante la utilización de algoritmos bioinspirados, adicionalmente para que el artículo tuviera un mayor impacto se decidió utilizar cuatro algoritmos bioinspirados diferentes para lograr una comparación entre los mismos, encontrando el algoritmo más eficiente.

Los algoritmos que fueron comparados son PSO (Particle Swarm Optimization), DE (Differential Evolution), y OPSO (aplicando la técnica de oposición) y ODE.

3.6.3 Sistemas en Chip

El trabajo esperado en esta área era la realización de un circuito capaz de realizar la configuración de un ADS 1115 de Texas instrument, para lo cual se tomó la asignatura de “Sistemas digitales”, en esta actividad se logró realizar la comunicación de manera simulada, no se alcanzó a implementar físicamente, esta tarea se detalla de una mejor manera en el capítulo 5.

Para corroborar que todas las actividades se realizaron de forma correcta en anexos se tiene una carta donde se indica que todos los objetivos fueron realizados de forma satisfactoria.

4 Trabajo en sistemas en chip

El trabajo realizado en el área de Soc (Sistem on chip), fue primeramente asistir a la disciplina de “Sistemas Digitales”, en esta disciplina se adquirieron conocimientos en el ámbito de programación en VHDL. Estos conocimientos son fundamentales para la realización de cualquier proyecto en Soc.

El proyecto desarrollado a continuación es el proyecto final de la disciplina, el cual contribuyo para realizar la comunicación mediante i2c de una FPGA con un conversor analógico digital.

El trabajo realizado en sistemas en chip fue la creación de un módulo master capaz de realizar la configuración y lectura de un conversor analógico digital (ADS 1115) de Texas instrument mediante comunicación i2c.

Este trabajo se dividió en dos etapas, la primera es la configuración del conversor analógico digital (ADS 1115) y la segunda su respectiva lectura.

4.1 Especificaciones del módulo ADS 1115 para su configuración y lectura.

4.1.1 Parámetros de configuración del ADS 1115

El módulo ADS 1115 de Texas instrument es un conversor analógico digital. El cual se configura enviando un vector de 16 bits especificando las configuraciones que queramos correspondientes.

Bit	Campo	Descripción
15	OS	**Estado operativo o inicio de conversión de disparo.** Este bit determina el estado operacional del dispositivo. Cuando escribe 0 No tiene efecto 1 Comienza una conversión simple (Cuando está en estado de apagado)
14:12	MUX[2:0]	**Configuración de entrada multiplexada.** Estos bits configuran la entrada del multiplexor. 000 : AINP= AIN0and AINN= AIN1(Predeterminado) 001 : AINP= AIN0and AINN= AIN3 010 : AINP= AIN1and AINN= AIN3 011 : AINP= AIN2and AINN= AIN3 100 : AINP= AIN0and AINN= GND 101 : AINP= AIN1and AINN= GND 110 : AINP= AIN2and AINN= GND 111 : AINP= AIN3and AINN= GND

11:9	PGA[2:0]	**Configuración de la ganancia del amplificador programable.** Estos bits configuran el FSR del amplificador de ganancia programable. 000: FSR= ±6.144V (1) 001: FSR= ±4.096V (1) 010: FSR= ±2.048V (predeterminado) 011: FSR= ±1.024V 100: FSR= ±0.512V 101: FSR= ±0.256V 110: FSR= ±0.256V 111: FSR= ±0.256V
8	MODE	**Modo de operación del dispositivo.** Este bit controla el modo de operación del dispositivo 0: Modo de conversión continua 1: Modo de disparo único o estado de apagado (predeterminado)
7:5	DR[2:0]	**Velocidad de datos.** Este bits ajusta la velocidad de los datos. 000 : 8 SPS 001 : 16 SPS 010 : 32 SPS 011 : 64 SPS 100 : 128 SPS(Predeterminado) 101 : 250 SPS 110 : 475 SPS 111 : 860 SPS SPS(sample per second)
4	COMP_MODE	**Modo de comparación.** Este bit configura el modo de funcionamiento del comparador. 0: Comparador tradicional (Predeterminado). 1: Comparador de ventana
3	COMP_POL	**Comparador de polaridad.** Este bit controla la polaridad del pin ALERT / RDY 0: Activa en bajo 1: Activa en alto
2	COMP_LAT	**Comparador Latching.** Este bit controla si el pin ALERT / RDY se engancha después de ser confirmado o se borra una vez que las conversiones están dentro del margen de los valores de umbral superior e inferior

		0: comparador de nonlatch. El pin ALERT / RDY no se engancha cuando se afirma (predeterminado). 1: comparador de enclavamiento. El pin ALERT / RDY declarado permanece enganchado hasta que el maestro lee los datos de conversión o el maestro envía una respuesta de alerta SMBus apropiada
1:0	COMP_QUE[1:0]	Estos bits realizan dos funciones. Cuando se establece en 11, el comparador se desactiva y el pin ALERT / RDY se establece en un estado de alta impedancia. Cuando se establece en cualquier otro valor, el pin ALERT / RDY y la función de comparación están habilitados, y el valor establecido determina el número de conversiones sucesivas que exceden el umbral superior o inferior requerido antes de afirmar el pin ALERT / RDY. Estos bits no cumplen ninguna función en el ADS1113 00: Afirmar después de una conversión 01: Afirmar después de dos conversiones 10: Afirmar después de cuatro conversiones 11: Deshabilite el comparador y configure el pin ALERT / RDY en alta impedancia (predeterminado)

Tabla 1: Parámetros de configuración del módulo ADS

Fuente (Texas Instrument)

La configuración del módulo ADS posee ciertos parámetros importantes, por ejemplo se desea capturar el mayor número de muestras posibles por segundo, por lo tanto en la variable DR será de 111, el canal de lectura que para este trabajo será el canal cero con respecto a tierra (la variable CH será de 100), el bit más significativo tendrá un valor de 1 indicando que se desea realizar una conversión simple, la variable PGA se deja predeterminada (010) al igual que la variable MODE(1), COMP_MODE(0), COMP_LAT(0), COMP_POL(0) y COMP_QUE(11).

La configuración que se satisface lo anteriormente dicho es(comenzando por el bit número 15)

$$Vec_{conf} = 1100010111100011$$

Esta configuración fue determinada por el profesor Sergio Andrés Pertuz Méndez.

4.1.2 Especificaciones para la configuración del ADS 1115.

Primeramente, se precisa saber cómo se realiza la escritura en el módulo ADS para poder realizar la respectiva configuración del mismo.

La forma para realizar la escritura en el módulo ADS se da por la siguiente figura, la cual indica como los datos se deben enviar.

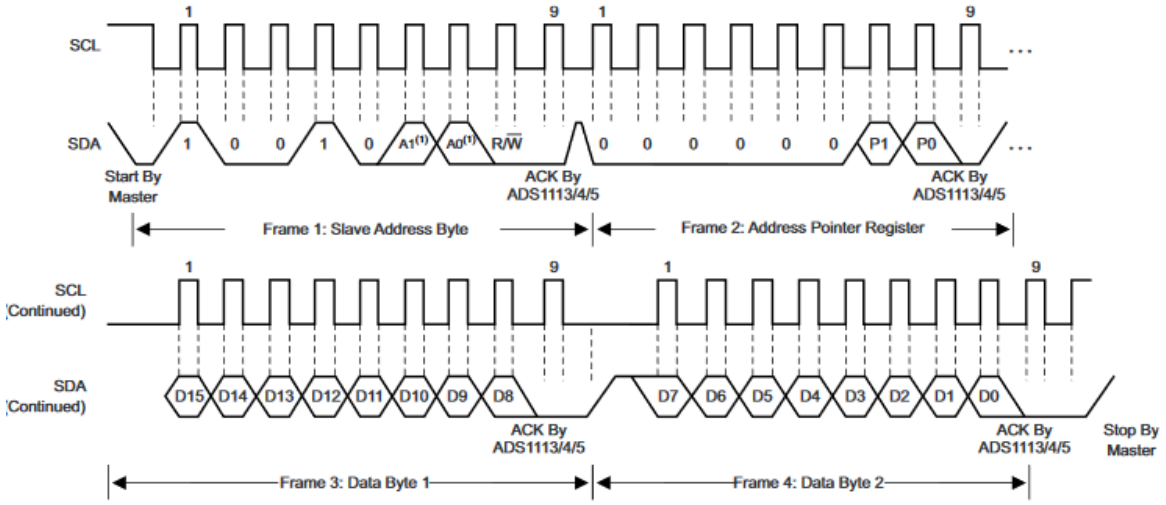

Figura 23: Secuencia para la escritura en el ADS 1115

Fuente (Texas Instrument)

En este trabajo el valor de $A1\ y\ A0$ es igual a cero, para poder conectar el pin ADDR a GND (esto se puede observar en el datasheet del módulo ADS115).

Primeramente, para realizar la configuración del ADS se envía el vector de dirección del módulo esclavo (en este caso el módulo esclavo es el módulo ADS), junto con la variable $R/\overline{W}$ la cual tendrá el valor de cero (esto indica que se va a escribir en el esclavo), se espera la confirmación de recibido del dispositivo esclavo (en la figura 16 se muestra como ACK by ADS115). Esta confirmación es de suma importancia, ya que indica que el módulo esclavo recibe los datos que enviamos desde el módulo master (módulo que se creará).

Seguidamente se envía el vector P(P es un vector de dos posiciones), el cual se utilizará para indicarle al módulo esclavo que se va a realizar la configuración inicial(el valor de P para la configuración es de 01). Igualmente se espera la confirmación del módulo esclavo.

Por último, se envían los valores del vector de configuración en dos momentos, primero se envía del valor 15 al valor 8 del vector configuración, se espera la confirmación del esclavo y se envía la última cadena de bits (de la posición 7 a la posición 0) esto se da por que la comunicación i2c solo permite él envió de 8 bits.

Al finalizar el módulo dará una señal de confirmación, de esta manera ya se habrá configurado el ADS 1115 y se pasará a la siguiente etapa (etapa de lectura del módulo ADS).

4.1.3 Especificaciones para la lectura del ADS 1115

Terminada la configuración del ADS 1115 se procede a la etapa de lectura del mismo, esta etapa de lectura está dada por la siguiente figura:

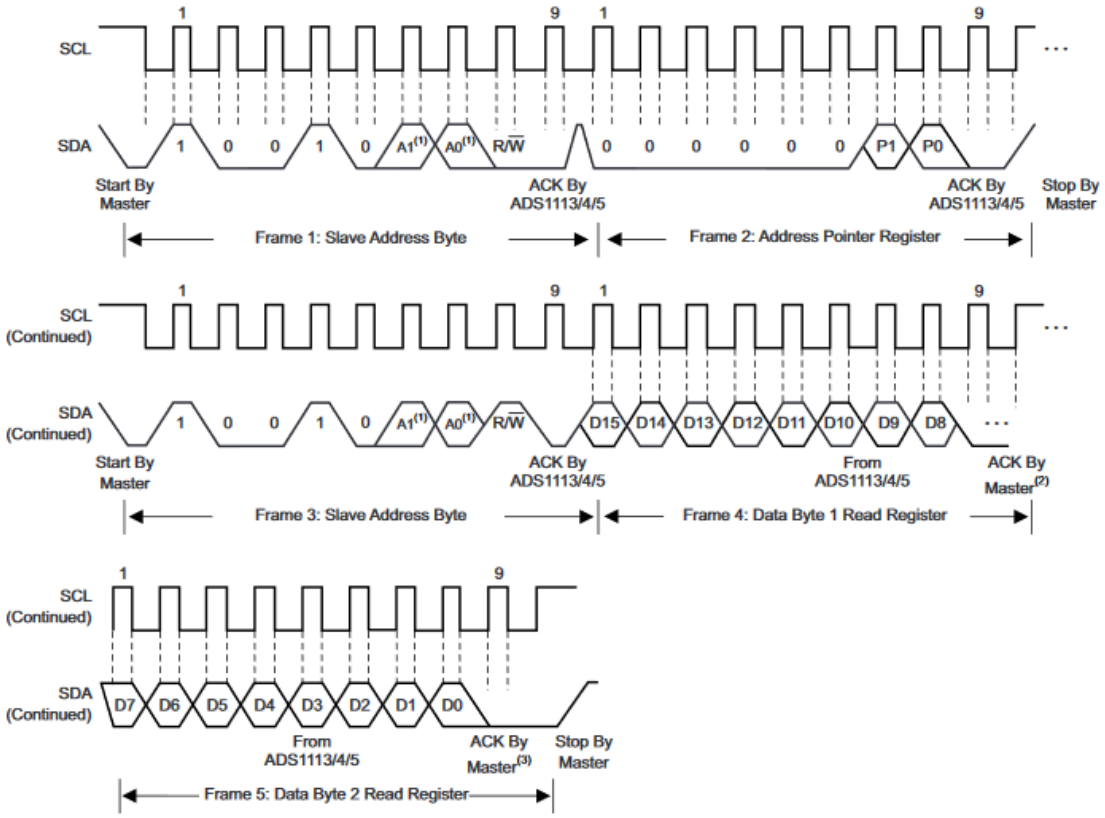

Figura 24: Secuencia para la lectura ADS 1115

Fuente (Texas Instrument)

Cabe aclarar que para pasar del modo configuración al modo lectura se tendrá que esperar unos nanosegundos para que la configuración del ADS surta efecto.

Una vez se realiza la respectiva espera se pasa a realizar la lectura del módulo esclavo ADS, para esto primeramente se envía el vector dirección(incluyendo los valores de $A0$ y $A1$ los cuales al igual que en el modo configuración poseen un valor de cero) junto con la variable $R/\overline{W}$, en este caso el valor de ella es de cero (ya que vamos a escribir los valores del vector P en el módulo esclavo).

Seguidamente se procede a enviar el vector P el cual para realizar la lectura del módulo esclavo tendrá un valor de 00, indicándole al esclavo que se realizaran trabajos de lectura.

Una vez terminado se procede a enviar nuevamente el vector dirección, con la variación de que esta vez la variable $R/\overline{W}$ será de valor uno logrando recibir la lectura del módulo esclavo.

Cabe aclarar que el ADS 1115 envía como valor de lectura un vector de 16 posiciones, el cual es recibido en dos momentos, primero recibe 8 bits, posteriormente el módulo master envía una señal de reconocimiento (en la figura 17 se muestra como ACK by Master), esto indica que se desea recibir la última

cadena de bits, si no se desea los últimos datos el master no envía la señal de reconocimiento.

Por ultimo cabe decir que para recibir un dato del módulo esclavo se realiza las dos etapas (de configuración y de lectura), esto con el fin de alternar los canales que se desean leer (esto se realiza cambiando el valor del vector MUX).

4.2 Diseño del módulo master

Este módulo se creó en el software de Vivado perteneciente a Xilins, el cual permite programar en VHDL el cual es utilizado para describir circuitos (Hardware), sintetizarlos, realizar simulaciones y posteriormente ser utilizados en una FPGA.

El módulo master tiene ciertas especificaciones dadas por el instructor, las cuales son, cada lectura de datos solo se podrá realizar si una señal de start es proporcionada, los valores de lectura se guardarán en un vector de cuatro posiciones, las señales de entrada serán:

- Reloj
- Reset
- Start
- Vector de configuración
- Valor de lectura del ADS 1115
- Busy

La entrada de Busy es del módulo de comunicaciones i2c el cual sirve para comunicar el módulo master con el ADS, este busy indica si el módulo i2c se encuentra desocupado (listo para enviar o recibir datos).

Las señales de salida del módulo master son:

- Enable i2c
- Dirección del ADS
- RW
- Datos a ser escritos en el ADS
- Ready
- Vector con los valores de lectura

La variable *Enable i2c* se utiliza para activar el módulo de comunicaciones i2c, ya que solo al momento de realizar la comunicación con el ADS se precisa que este este activo.

La programación de descripción de hardware requirió la implementación de una máquina de estados para poder realizar el envió de datos de una manera más fácil. Cabe decir que estos conocimientos se adquirieron asistiendo a la disciplina de Sistemas digitales impartidas en la Universidad de Brasilia.

4.2.1 Máquina de estados del módulo master.

La máquina de estados que posee el módulo master es de tipo Moore, en la cual la salida no depende de las entradas sino exclusivamente del estado en el cual se encuentra, en la siguiente figura se muestra el diagrama de estados utilizados.

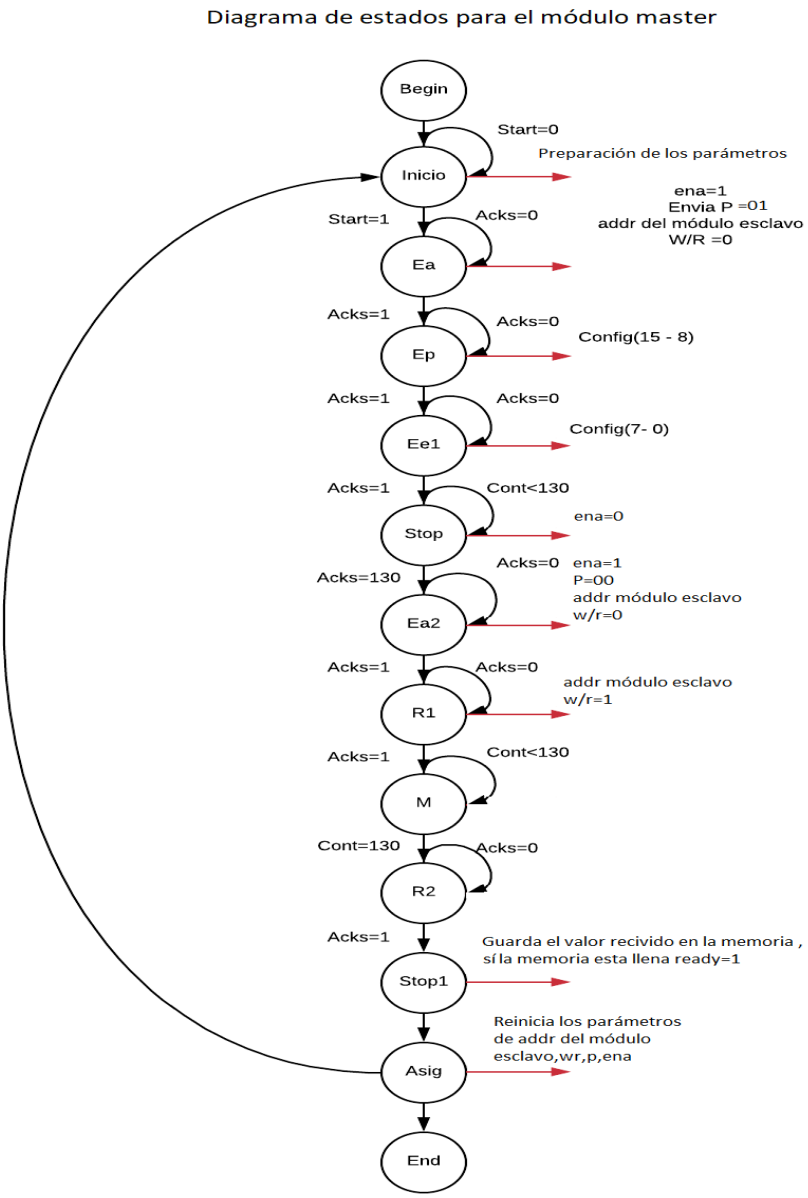

Figura 25: Diagrama de estados del módulo master

Fuente(Autor)

Esta máquina de estados posee doce estados en donde la idea es realizar paso por paso el procedimiento para la configuración del módulo ADS 1115.

En el primer estado (*start*) la máquina se encuentra esperando que la variable *start* sea uno, esta variable es la que da inicio a todo el proceso, si la variable es cero el programa se quedará en este estado.

Si llega el valor de uno a la variable *star,* pasa al siguiente estado el cual es *Ea,* en este estado envía la variable *ena* con el valor de uno, indicando que la comunicación va a comenzar, seguidamente enviar el valor de la varible *P* en este estado es de

01, lo que indica que va a configurar el módulo, además de la dirección del módulo esclavo y el respectivo bit de escritura (*W/R=0*).

En los dos siguientes estados se envían dos cadenas de ocho bits, esto con el fin de realizar la configuración del módulo esclavo de (mirar el Vec_{conf}).

Seguidamente se realiza una pausa y esto con el fin de que el módulo se configure correctamente. Una vez el módulo queda configurado se procede a realizar la captura de datos que el posee, para esto son los estados siguientes.

En el estado *Ea2* se envían los respectivos parámetros para proceder a realizar una lectura del módulo ADS 1115, los cuales son *P* con valor de 00, la dirección del módulo esclavo ADS y la variable *W/R* en cero, esto con el fin de escribir en el módulo el valor de *P* (ya que con esta configuración el módulo ADS puede ser leído).

En *R1*, el módulo ADS envía los primeros ocho bits de lectura (de un total de dieciséis), en el siguiente estado (*M*), el módulo master envía un bit de *ackm* indicando que se desea recibir los últimos ocho datos, posteriormente en *R2,* se reciben estos últimos bits por parte del ADS.

Estos datos son guardados en una memoria cuando se pasa al estado *Stop1*, por último se reinician todas las variables (en el estado *Asig*),y se procede nuevamente a realizar todo el proceso(siempre y cuando el valor de *Start* sea uno).

4.2.2 Conexiones del módulo master con el esclavo y el módulo i2c

Las conexiones del módulo master con el ADS 1115 y el módulo i2c queda de la siguiente manera:

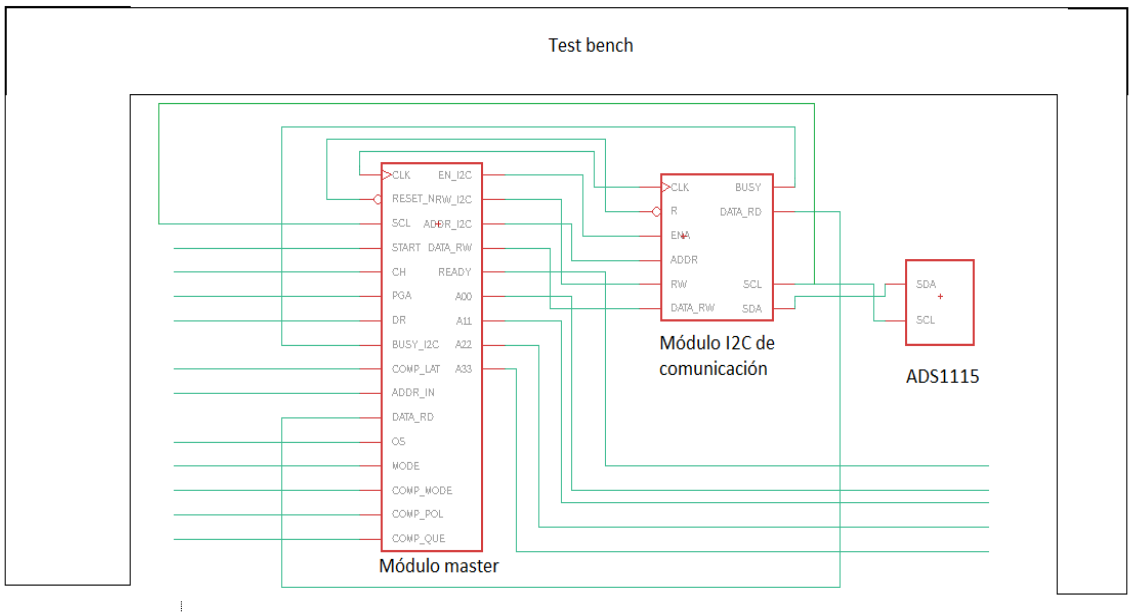

Figura 26: Circuito DUT (Device Under Test)

Fuente(Autor)

El test bench es el archivo utilizado para realizar la simulación de los circuitos, en este archivo se encuentran los valores del vector configuración, los cuales una vez llegan al módulo master son concatenados para generar el vector configuración de 16 posiciones.

4.2.3 Timing Analysis.

Para realizar la simulación primeramente se necesita la frecuencia máxima a la cual nuestros circuitos pueden funcionar, el módulo master funciona hasta máximo con una frecuencia de 250MHz (el reloj debe tener un periodo mayor a 4ns) esto es un resultado que resulta al aplicar el *TimeQuest Timing Analysis.* Este análisis se realizó en el software de Quartus en su versión 13 ya que es relativamente sencilla su implementación, la figura 48 muestra el resultado del módulo master:

Slow 1200mV 85C Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	300.39 MHz	250.0 MHz	clk	limit due to minimum period restriction (max I/O toggle rate)
2	1319.26 MHz	844.59 MHz	state.ea	limit due to hold check

Figura 27: Timing Analysis del módulo master

Fuente (Autor)

En el circuito del módulo i2c la frecuencia máxima de trabajo fue de 250 MHz (un periodo máximo de 4ns), se utilizo el mismo software que para el reloj del módulo master, la respuesta del *TimeQuest Timing Analysis* es mostrado en la siguiente figura 24.

Slow 1200mV 85C Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	297.8 MHz	250.0 MHz	clk	limit due to minimum period restriction (max I/O toggle rate)

Figura 28: Timing analysis del módulo i2c

Fuente(Autor)

4.2.4 Simulación del DUT

Con base en estos resultados se optó por una frecuencia de reloj de 50ns para la simulación de los circuitos, el envió de datos se realizará siempre que el reloj del SCL (este reloj pertenece al módulo de comunicaciones i2c) este en la borda de subida . La simulación de estos tres circuitos se puede observar en la figura 25.

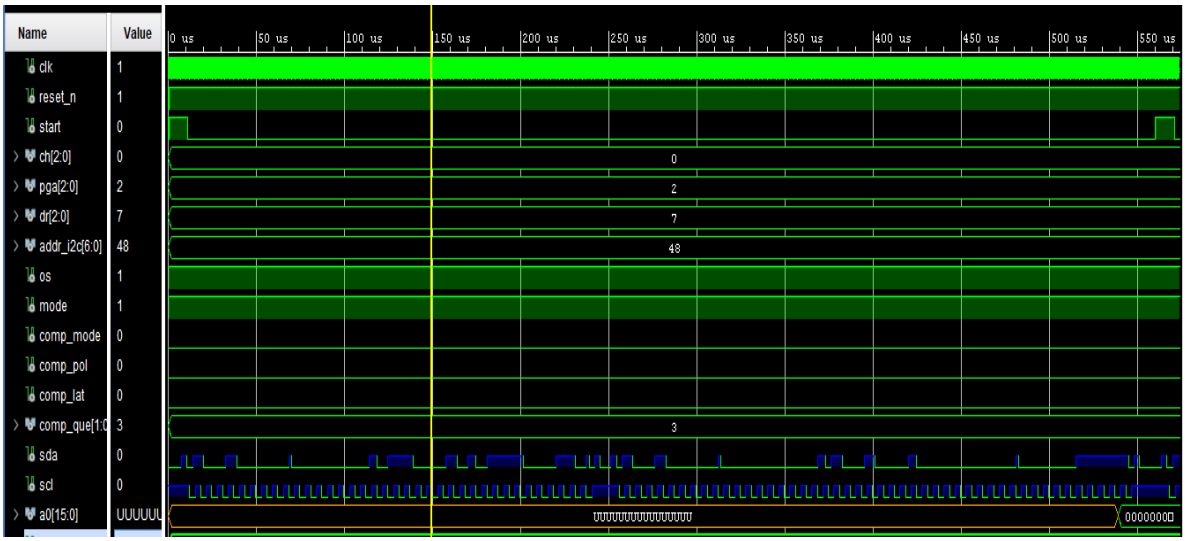

Figura 29: Simulación del circuito DUT

Fuente(Autor)

Según la simulación el tiempo necesario para capturar un dato del conversor ADS 1115 es de 539.085μs, en esta simulación la señal principal que debemos observa es la de SDA, ya que por esta señal es la comunicación entre el módulo master y el conversor ADS.

El primer pico de la señal es simplemente una confirmación diciendo que el módulo de comunicaciones i2c está listo para comenzar la comunicación, seguidamente se tiene el valor de 10010000, el cual corresponde con la dirección del ADS con valores de $A0 = 0$ y $A1 = 0$, además del valor de $R/\overline{W}$ el cual posee un valor de cero.

Seguidamente se tiene la confirmación del módulo esclavo en el tiempo 68.685 μs, seguidamente se envía el vector de ceros y P, esto se repite tal como se describió en las especificaciones para la configuración y comunicación.

El valor de la variable a0 que se observa en el tiempo 539.085μs es el valor enviado por parte del módulo ADS, este valor se almacenará en una memoria FIFO de cuatro posiciones, cuando la memoria está llena el primer valor es eliminado para poder almacenar el ultimo.

Cabe destacar que como el conversor posee cuatro canales para la lectura de datos se poseen cuatro memorias que almacenan cada uno de estos datos del conversor ADS, la utilización de estas memorias dependerá del canal que se desee leer, esta selección está dada por la variable CH, la cual es una entrada del módulo master.

5 Diseño PCB

La primera tarea consistió en el diseño de la placa de circuito impreso, esta se realizó en el software de EAGLE (de la empresa AUTODESK), la placa será usada para la creación de un servo motor. Los componentes utilizados fueron los siguientes:

Componentes	Cantidad Unitaria
Convertidor analógico digital (ads1115)	1
Controlador para el motor (DRV8830)	1
Sensor de corriente INA 240	1
Regular a cinco volts slvs 178b	1
Sensor de posición angular Murata	1
R1 de 0.2Ω a 0.1W encapsulado 0805	1
R2 de 1kΩ a 1/8 W encapsulado 1206	1
R3 de 1kΩ a 1/8 W encapsulado 1206	1
R4 de 100mΩ a 1/4 W encapsulado 1210	1
R5 de 1kΩ de 1/8 W encapsulado 1206	1
R6 de 330Ω de 1/8W encapsulado 1206	1
C1 de 10µF a 1/8W encapsulado 1206	1
C2 de 0.1µF a 1/8W encapsulado 1206	1
C3 de 1µ a 1/8W encapsulado 1206	1
C4 de 1µ a 1/8W encapsulado 1206	1
C5 de 1µ a 1/8W encapsulado 1206	1
D1 de 1.5A encapsulado SOD-128-2	1
LED1 1.7V a 2mA encapsulado 1206	1
1 Micro conector de 2 posiciones	1
Micro conector de 4 posiciones	2
Micro dipswitch de 2 posiciones	1
Micro dipswitch de 4 posiciones	1

Tabla 2:Componentes para la creación de la PCB

los primeros cuatro de la familia de Texas instrument, el quinto de la compañía Murata.

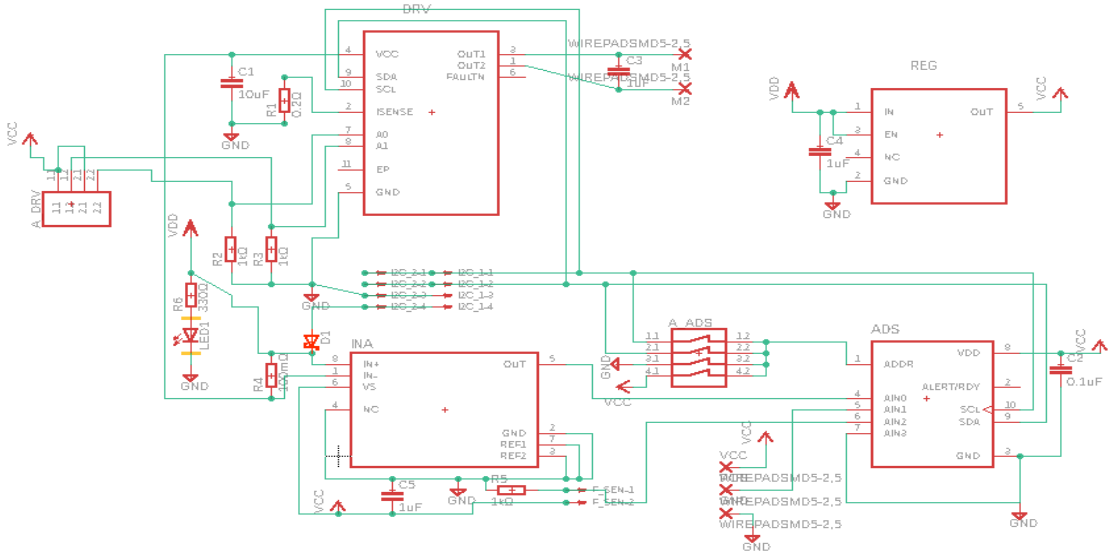

Figura 30: Esquemático de la PCB

(Fuente: Autor)

Las conexiones se realizaron siguiendo las instrucciones de los respectivos datasheet, adicionando un led indicador el cual enciende cuando el motor se encuentra con alimentación.

Para la creación de las conexiones de la placa en EAGLE se tomaron ciertos criterios de diseño como por ejemplo el tamaño de las vías las cuales depende de la cantidad de corriente que circulará por la misma, para este cálculo el orientador de Brasil proporciono una página de internet la cual realiza automáticamente el cálculo del grosor de las vías(https://www.7pcb.com/trace-width-calculator.php), dando los siguientes resultados:

Para las vías que van conectadas al motor se utilizó un grosor de 50 mil (11,3 es era lo recomendado) con una reducción hacia el DRV 8830, esto se realizó por recomendación del orientador para una mejor conductividad, al igual que tener un factor de seguridad alto.

La vía de VCC se realizó con un grosor de 24 mil (11,3 mil es lo recomendado), se dejó también un cierto rango de seguridad para su buen funcionamiento. Por último se realizaron las vías de señal las cuales tienen un grosor de 8 mil, soportando hasta 623 mA.

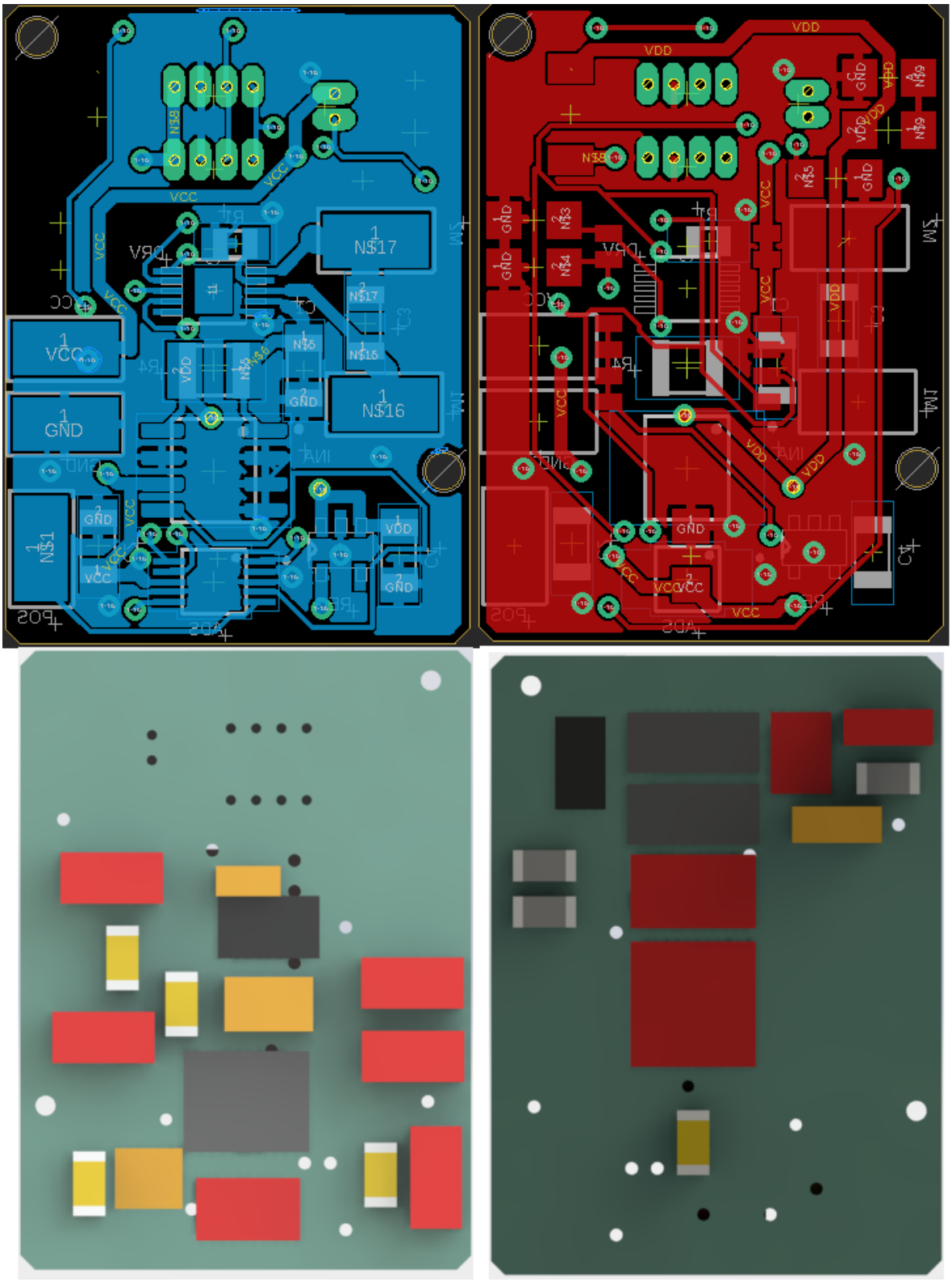

Figura 31: Diseño terminado de la PCB

(Fuente: Autor)

6 Implementación de optimización bioinspirada

El controlador de impedancia se utilizará en una mano robótica la cual básicamente agarra objetos con una determinada fuerza/torque, el controlador de impedancia realizará el control dinámico del robot, tal como se observó en la sección 2.5.

6.1 Controlador de impedancia

Como se comentó en la sección de marco teórico la dinámica de un robot es controlada por la Eq. 8, en el trabajo de [31], se utilizó la integración discreta para la implementación del controlador de impedancia, su esquema es representado a través de estos tres pasos:

- Discretiza la Eq. 8 del sistema representado de la siguiente manera:

$$\ddot{X}(K) = M^{-1}(\Delta T(k) - k\Delta_x(k) - B\Delta\dot{x}(k)) \quad (9)$$

  Donde la fuerza de la Eq. 8, es sustituida por el error del torque medido en el momento k ($\Delta T(k)$, $\Delta_x(k)$ es el error del movimiento y $\Delta\dot{x}(k)$ es el error de la derivada del movimiento

- El resultado de función anterior $\ddot{X}$ representa la aceleración de la junta, la cual es integrada para encontrar la velocidad de la misma $\dot{X}$

$$\dot{X} = \int_{k-1}^{k} \ddot{X}\, dx \quad (10)$$

- El paso anterior se repite para hallar la posición de la junta X

$$X = \int_{k-1}^{k} \dot{X}\, dx \quad (11)$$

x en [31] es la posición o el desplazamiento deseado del robot controlado.

La representación esquemática de método de integración discreta del control de impedancia es mostrada en la figura 7.

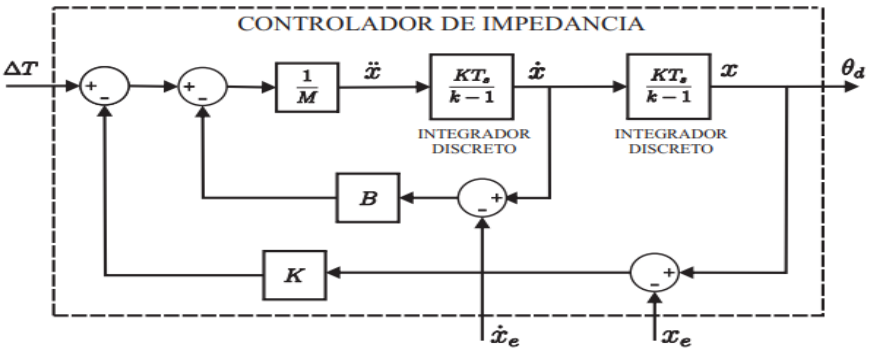

Figura 32: Esquema de control

Fuente(Pertuz,2016)

Donde T_s es el periodo de la actualización del controlador, y x_e es el movimiento medido y $\dot{x}_e$ es su deriva.

El esquema de la figura anterior es implementado en el sistema de control del dedo robótico, ilustrado en la Fig. 8, el bloque de controlador de impedancia tiene como dato de entrada el error del torque ($\Delta T = T_d - T_e$) y los valores estimados de posición y velocidad (θ y $\dot{\theta}$).

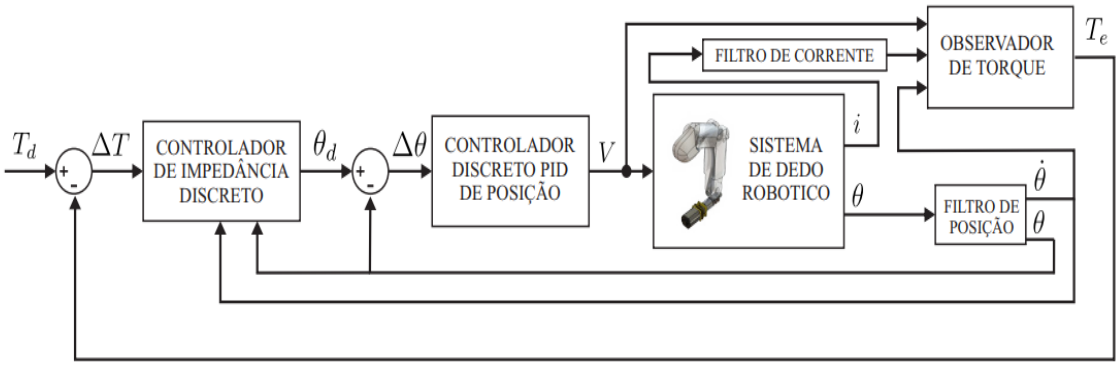

Figura 33: Esquema de control propuesto

Fuente (Pertuz,2016)

6.2 Sintonización del controlador de impedancia.

El algoritmo PSO (ver la sección 3.2.2) es un algoritmo de optimización el cual implementa una búsqueda heurística, con el menor esfuerzo posible, simulando el comportamiento que tienen los enjambres cuando procuran alimento.

En el algoritmo DE (ver sección 3.2.3), es un algoritmo que se basa en la población, generando agentes nuevos aplicando mutaciones para crear una mejor población y con esto lograr alcanzar la optimización deseada (en este caso optimizar los parámetros del controlador de impedancia para encontrar el menor error ponderado posible).

El uso de algoritmos bioinspirados para la sintonización de controladores es una aplicación muy común tal como se observó en [13] y en [17], este problema de optimización se puede resolver con diferentes tipos de funciones Custom(o función objetivo), para este trabajo se utilizó la siguiente función objetivo, la cual se presenta de manera más detalla en [31], la función objetivo es:

$$\boldsymbol{f_{custom} = 0.15 * t_e + 0.45 * MSE + 0.15 * Sobrepico + 0.25 * t_o} \quad (12)$$

Donde t_e es el tiempo de estabilización del controlador, t_o el tiempo que se demora en ser aplicado el setpoint y el MSE de la respuesta del controlador, comparado con la del setpoint, estos valores de cada variable fueron determinados de manera empírica en [31].

Resumiendo, en esta implementación se desea realizar un trabajo de minimización (ya que en algoritmos bioinspirados se puede trabajar maximizando o minimizando funciones), de la función Objetivo (función Custom) la cual es el error ponderado de la mano robótica, encontrando los respectivos parámetro de $K, B\ y\ M$ que llevan a esta minimización, además de realizar la comparación de los algoritmos mediante análisis estadístico, teste de hipótesis y las repuestas de los controladores de impedancia.

La implementación de los algoritmos bioinspirados para la minimización de la función Objetivo posee unas generalidades que son, el tamaño de la población es 12 individuos, 80 iteraciones por experimento, la inicialización de todos los algoritmos es de manera aleatoria, las variables de decisión son $K, B\ y\ M$, generando que se trabaje en tres dimensiones.

Dado la naturaleza de la función Objetivo (la cual posee mínimos locales) y el problema que poseen los algoritmos PSO y DE (convergencia prematura) se añadió una técnica de diversificación, la cual es la oposición, esta permite al algoritmo invertir las partículas si no mejora la salida de la función objetivo después de un determinado número de iteraciones. En la siguiente tabla se resumen las condiciones de los algoritmos.

Parámetros	Valor
Número de partículas para OPSO y ODE	12
Dimensiones (variables de decisión B, K, M)	3
Límite inferior de búsqueda	$[1x10^{-4}\ \ 1x10^{-4}\ \ 1x10^{-4}]$
Límite superior de búsqueda	[300 30 10]
Límite de iteraciones	80
Límite de experimentos	20
Coeficiente cognitivo C1 de OPSO	3.05
Coeficiente social C2 de OPSO	1.05
Wmax	0.9
Wmin	0.08
Paso de decrecimiento	0.01025
Factor de mutación ODE	1,25

Valor de Crossover ODE	0,75
Límite de iteraciones para la oposición	20
Treshold	0.1

Tabla 3: Parámetros generales de los algoritmos bioinspirados y de la función objetivo.

Fuente(Autor)

El treshold en estos experimentos se utiliza para observar cuantas veces el algoritmo logra obtener un valor aceptable y no para detener el algoritmo. Se utilizó el software de MATLAB en su versión 2015Ra la cual, la Universidad de Brasilia poseía la licencia.

6.3 Sintonización con PSO

En la primera sintonización se utilizó el algoritmo PSO canónico el cual no posee modificaciones (el mismo algoritmo del marco teórico). Los resultados de la convergencia de los experimentos se observan en la figura 16:

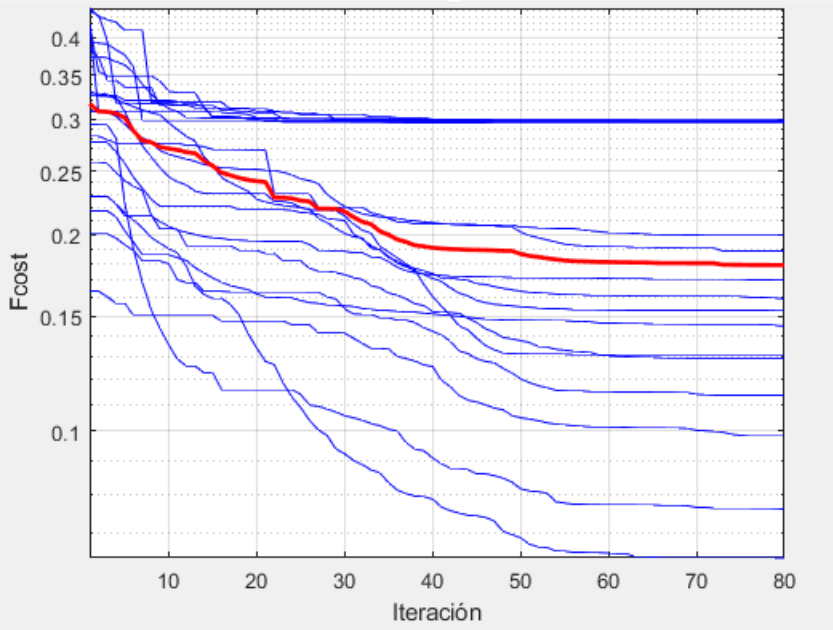

Figura 34: Curvas de convergencia para el algoritmo PSO

Fuente(Autor)

En la gráfica de curvas de convergencia en color azul se puede apreciar los 20 experimentos y en color rojo se puede apreciar la mediana de los 20 experimentos, esta última se utilizará para compararse con los demás algoritmos bioinspirados.

En estas curvas se puede observar como el error ponderado (valor de la función objetivo) va disminuyendo hasta llegar a valores por debajo del treshold (valores

aceptables para la implementación), los valores que quedan por encima del treshold son descartados para su utilización.

Además, se utilizó la escala logarítmica para el eje y, logrando una mejor apreciación de la convergencia de cada uno de los experimentos.

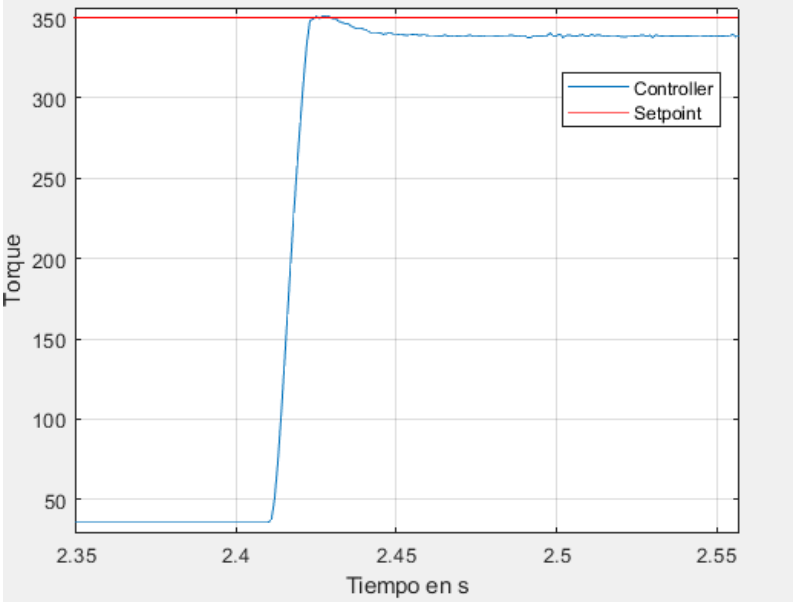

Figura 35: Respuesta del controlador con el algoritmo de PSO

Fuente(Autor)

De la figura 17 se pueden extraer varios aspectos importantes para evaluar el desempeño del algoritmo al momento de la sintonización del controlador, el primero es que posee un error en estado estable de 3.2%, lo cual indica que la sintonización no fue acertada (la idea es que la acción del controlador permita llegar al setpoint), además las variables halladas de $M, B\ y\ K$ son $12.4672, 5.1057, 0.0030$ respectivamente (el error en estado estable se da por que la constante M posee un valor relativamente grande)

6.4 Sintonización con DE

En la segunda sintonización se utilizó el algoritmo DE canónico el cual no posee ninguna modificación (el mismo algoritmo del marco teórico). Los resultados de la convergencia de los experimentos se observan en la figura 18:

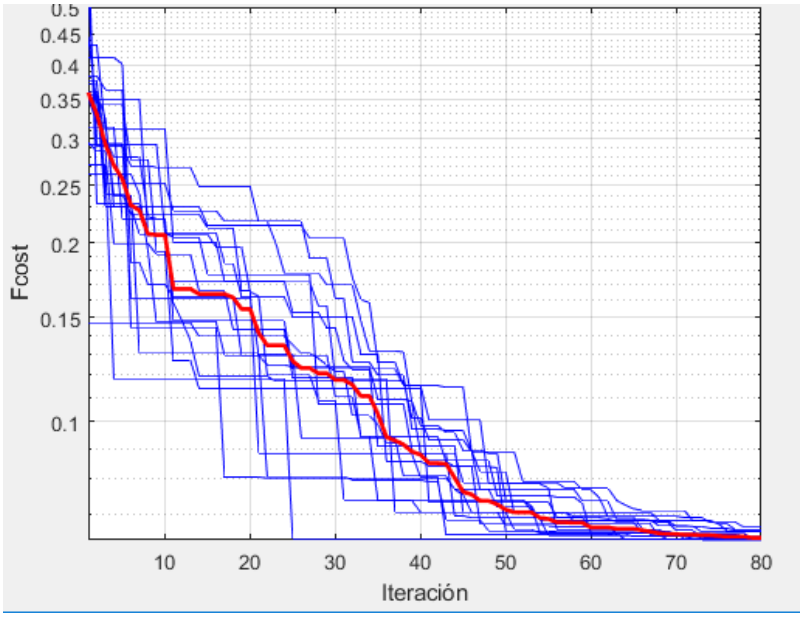

Figura 36: Curvas de convergencias para los experimentos con el algoritmo DE

Fuente(Autor)

Al utilizar este algoritmo, se observa que todos los experimentos llegan a tener valores menores a 0.1, una característica bien importante ya que nuestro valor de error aceptable (o treshold) es de 0.1.

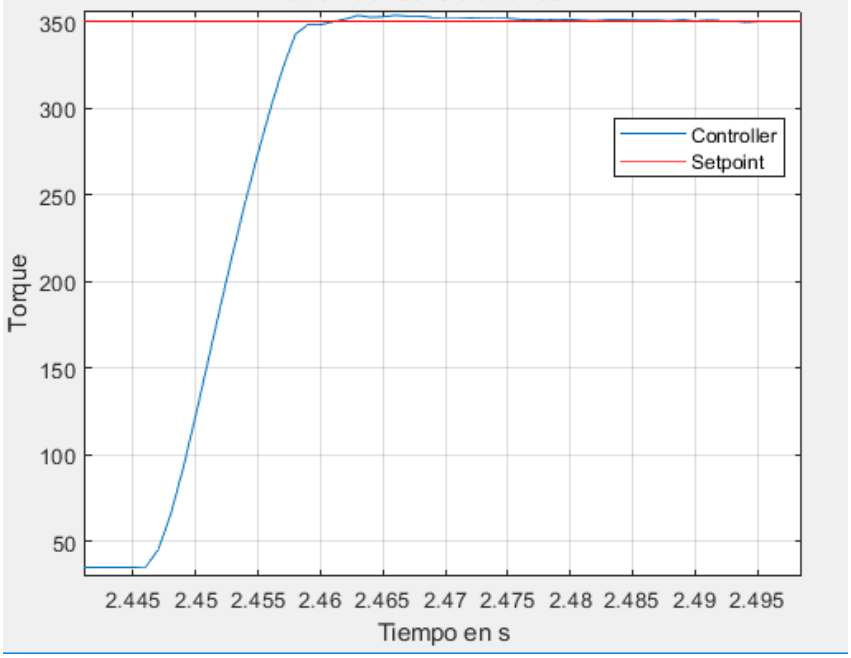

Figura 37: Respuesta del controlador con el mejor valor de los experimentos del algoritmo DE

Fuente(Autor)

Para la respuesta utilizando el algoritmo DE no se presentó error en estado estable, con valor de M igual a 0.00010 B igual a 5.3825 y K 0.0028

6.5 Sintonización con OPSO

Para la sintonización del tercer controlador, el PSO clásico (o canónico) se modificó para evitar caer en una convergencia prematura mediante la utilización del algoritmo de oposición, el cual se aplica cuando la solución no mejora después de 20 iteraciones, el pseudocódigo del OPSO es el siguiente:

Algoritmo 3 Seudocódigo para el algoritmo OPSO

1: **función** OPSO-básico(*S,N,c1,c2,*Max_{iter}*, threshold,Lim*)
2: Inicializa el enjambre con X_{max} y con X_{min}
3: *iter*=1;
4: **Mientras** *iter* < Max_{iter}
5: **para i de** 1 **hasta** S
6: **sí** $f(x_k) \leq f(x_{ik})$ **entonces**
7: $y_{ik} \leftarrow x_k$**;**
8: $Limit_k = 0$;
9: **De lo contrario**
10: $Limit_k = Limit_k + 1$;
11: **Fin sí**
12: calcula y_s usando el *S* fitness valor $f(y_{ik})$;
13: **para i de** 1 **hasta** S
14: **para j de** 1 **hasta** N
15: $v_{ij}^{(t+1)} \leftarrow wv_{ij}^{(t)} + c_1U_{1j}(y_{ij}^{(t)} - x_{ij}^{(t)}) + c_2U_{2j}(y_{sj}^{(t)} - x_{ij}^{(t)})$;
16: $x_{ij}^{(t+1)} \leftarrow x_{ij}^{(t)} + v_{ij}^{(t+1)}$;
17: **Fin para;**
18: **Fin para;**
19: **Para** K **de** 1 **hasta** N
20: **Para** i **de** 1 **hasta** S
21: **Sí** $Limit_k = Lim$ **entonces**
22: $x_{ik} = X_{max} + X_{min} - x_{ik}$**;**
23: **Fin sí;**
24: **Fin para;**
25: **Fin para;**
26: *iter=iter+1*
27: **Fin mientras;**
28: **Retorna** Posición de la mejor partícula fit x y su valor $f(x)$
29:**Fin de la función**

Figura 38: Pseudocódigo de OPSO

Fuente(Autor)

Para la implementación de la oposición se utiliza la ecuación número seis, esta ecuación se utiliza con cada uno de los agentes participes del enjambre, los resultados obtenidos se aprecian a continuación:

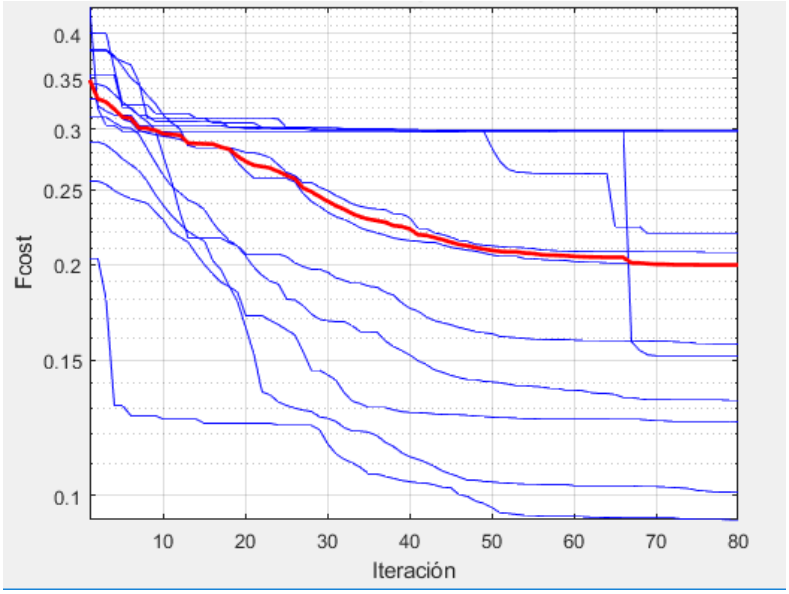

Figura 39: Curvas de convergencia para el algoritmo de OPSO

Fuente(Autor)

Todos los valores finales del error llegan a un rango de 0.2991 a 0.0929. La gráfica de torque con respecto al tiempo del mejor valor mínimo se muestra a continuación:

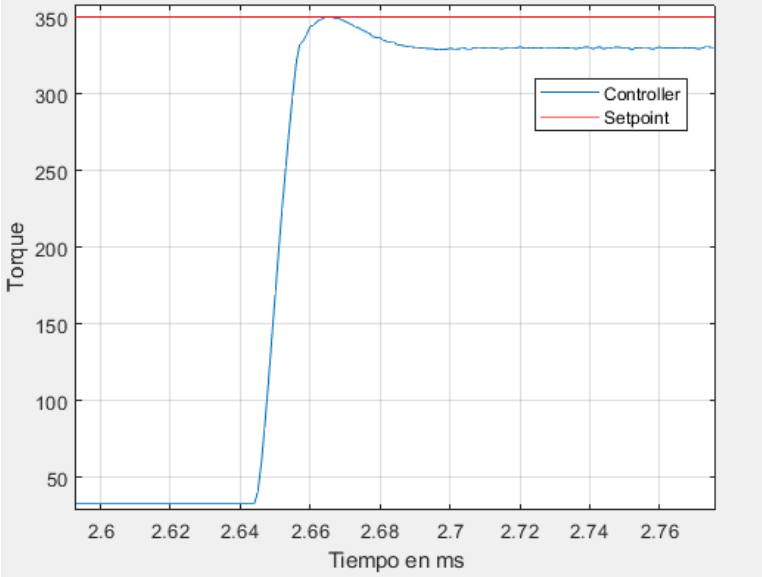

Figura 40: Respuesta del controlador de impedancia

Fuente(Autor)

En la gráfica se puede observar que en la respuesta del controlador no se tiene sobre pico, pero si un error en estado estacionario el cual es de 6,36%, tiene un tiempo pico de 21ms, un tiempo de asentamiento de 56 ms, no posee sobre pico, un tiempo de retardo de 6,78 ms y un tiempo de subida de 9,68 ms

Los valores de las constantes halladas $M, B\ y\ k$ son, $22{,}4603, 5{,}5517$y $0{,}0178$ respectivamente.

6.6 Sintonización con ODE

Al igual que el OPSO la oposición se implementa después de 20 iteraciones sin obtener mejoras. El pseudocódigo con la oposición queda de la siguiente manera:

Algoritmo 2 Seudocódigo para el algoritmo DE

1: **función** DE-básico (*M,C,F,range,f,Lim*)
2: $x \leftarrow aleatorio\ (range, M)$;
3: $fit_x \leftarrow f(x)$;
4: **Mientras** no se cumplan los criterios de parada **Hacer**
5: **para i de** 1 **hasta** M
6: $v_i \leftarrow \boldsymbol{Mutación}\ (x_i, F)$;
7: $u_i \leftarrow \boldsymbol{Crossover}\ (x_i, v_i, CR)$;
8: **Fin para;**
9: $fit_u \leftarrow f(u)$;
10: **Para i de** 1 **hasta** M
11: **Sí** $fit_u(i) < fit_x(i)$ **Entonces**
12: $x_i \leftarrow\ u_i$;
13: $Limit_k = 0$;
14: **De lo contrario**
15: $x_i \leftarrow\ x_i$;
16: $Limit_k = Limit_k + 1$;
17 **Fin sí;**
18: **Fin para;**
19: **Para** K **de** 1 **hasta** N
20: **Para** i **de** 1 **hasta** S
21: **Sí** $Limit_k = Lim$ **entonces**
22: $x_{ik} = X_{max} + X_{min} - x_{ik}$;
23: **Fin sí;**
24: **Fin para;**
25: **Fin para;**
26: **Fin mientras;**
27: **Retorna** Posición de la mejor partícula x y su valor en $f(x)$

Figura 41: Pseudocódigo del algoritmo ODE.

Fuente (Autor)

Las curvas de convergencia de cada uno de los experimentos obtenidos se aprecian a continuación.

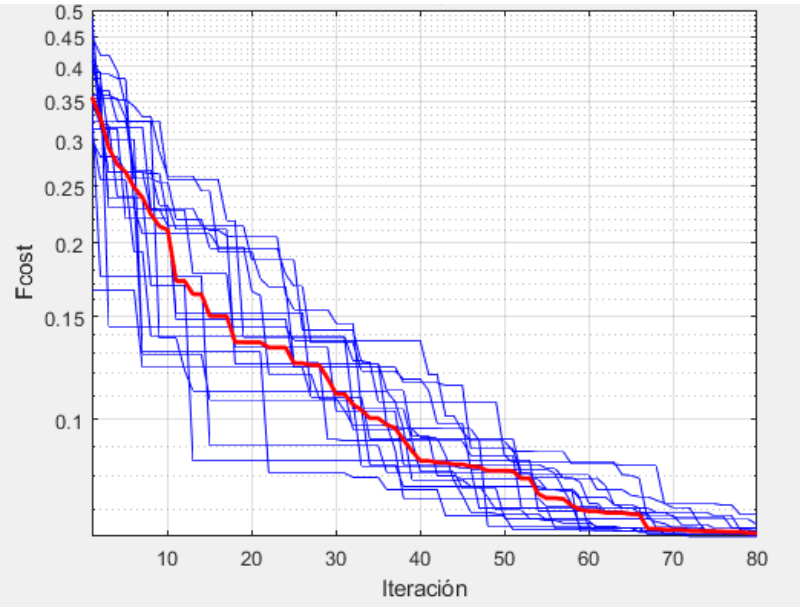

Figura 42: Curvas de convergencia de los experimentos

Fuente(Autor)

De igual forma en estas curvas se puede observar si nuestro algoritmo logró reducir el error hasta un valor aceptable (valor por debajo del treshold), a simple vista se puede observar que varios experimentos obtienen un valor de error deseado para ser implementado en el controlador de impedancia.

Todas las curvas convergen a un rango de 0.0632 a 0.0688 (claro está que el mejor valor es el de 0.0632).la gráfica de torque con respecto al tiempo del mejor valor mínimo es la siguiente:

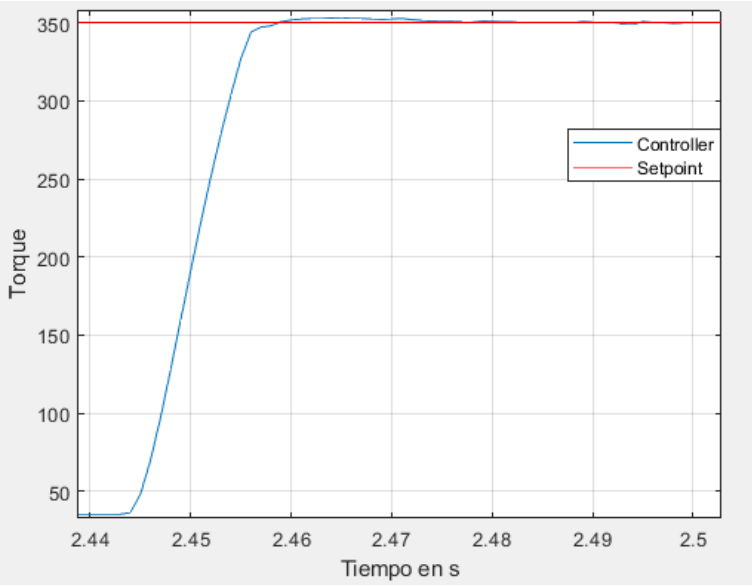

Figura 43: Respuesta del controlador de impedancia

Fuente(Autor)

En la gráfica se puede apreciar que la respuesta del controlador posee un sobre pico de 0,95%, un tiempo pico de 20ms, un tiempo de asentamiento de 36 ms, un tiempo de retardo de 7 ms y un tiempo de subida de 9 ms.

Por último, se sobrepusieron las diferentes gráficas de los controladores junto con el setpoin para poder tener una mejor apreciación de los valores de respuesta del controlador y poder realizar una comparación gráfica de las mismas.

Los valores de las constantes halladas $M, B\ y\ k$ son, $1x10^{-4}, 5.3770\ y\ 0.0028$ respectivamente.

6.7 Resumen de las sintonizaciones

Por último, se realizó la comparación gráfica del mejor valor de cada algoritmo con su respectiva mediana, esto con el fin de determinar visualmente cual algoritmo había tenido en un mejor rendimiento en comparación a los demás.

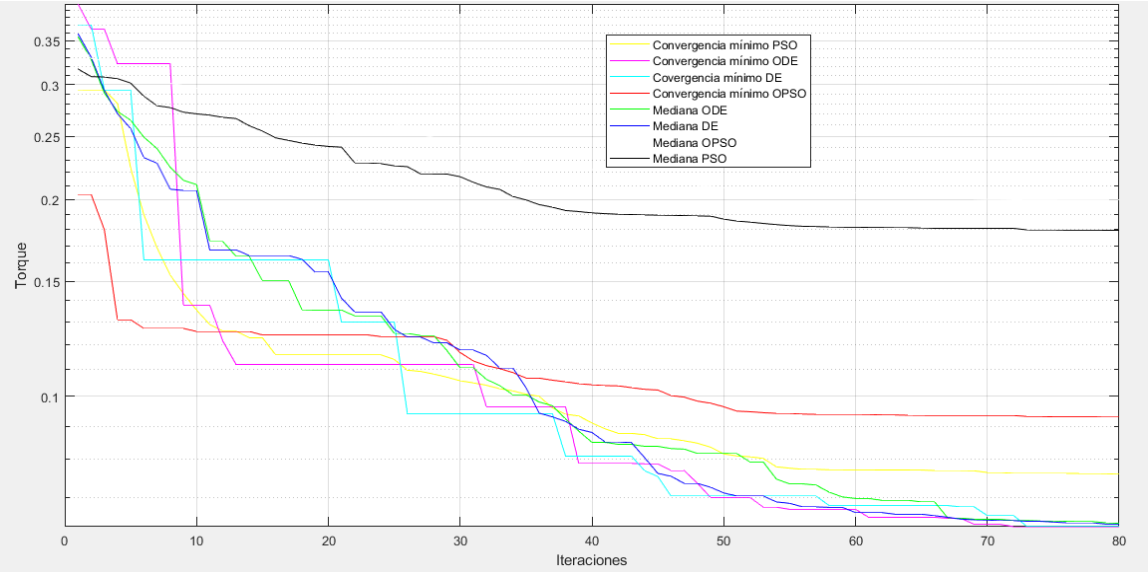

Figura 44: Curvas de convergencia de los mínimos y medias de cada algoritmo

Fuente(Autor)

De igual forma la respuesta del sistema mediante la utilización de los diferentes algoritmos fue graficada, pudiéndose realizar una comparación visual y lograr determinar cuál algoritmo realizó la mejor sintonización.

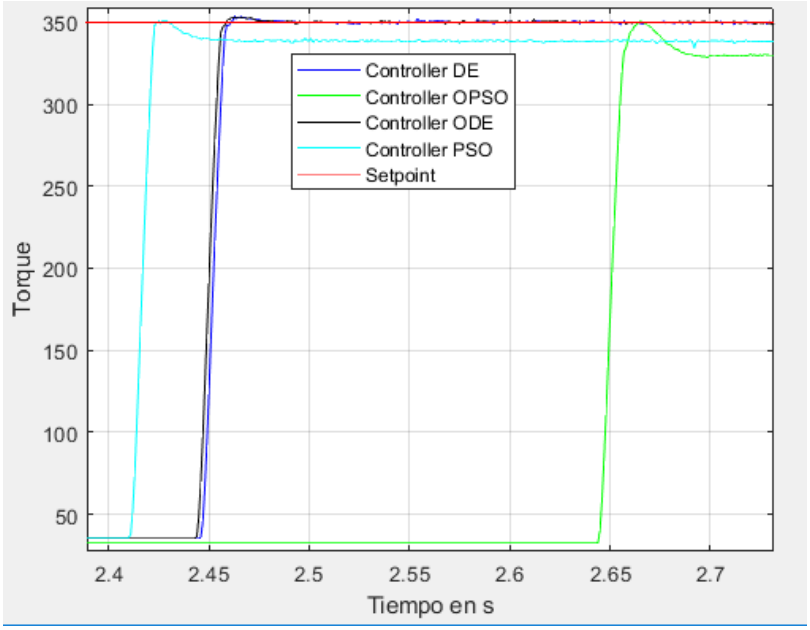

Figura 45: Respuestas de los controladores

Fuente(Autor)

Por último, se graficó la posición del dedo para cada una de las sintonizaciones, las cuales se observan en la figura 21:

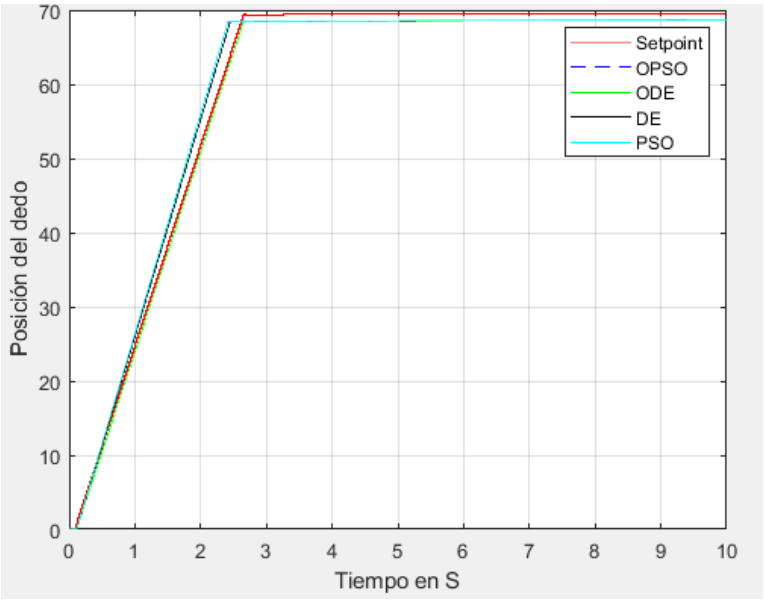

Figura 46: Posición del dedo con cada uno de los algoritmos bioinspirados

Fuente(Autor)

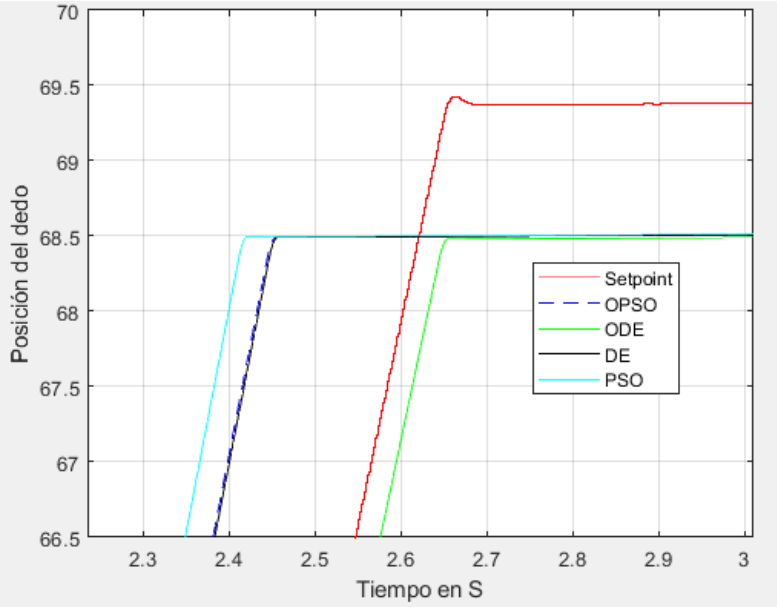

Figura 47: Ampliación de la imagen anterior.

Fuente(Autor)

Los valores de los resultados de la gráfica de posición del dedo se ven reflejados en la siguiente tabla, donde el valor del setpoint es de $69{,}37°$:

Algoritmo	Valor en °	Error en estado estacionario (%)
PSO	68,5	1,2541
DE	68,49	1,2685
OPSO	68,49	1,2685
ODE	68,48	1,2829

Tabla 4: Valores de la gráfica de posiciones

Fuente(Autor)

En la siguiente tabla se observa el resumen de los datos estadístico obtenidos, donde G es el total de experimentos que alcanzaron a llegar por debajo del treshold (el cual es de $0{,}1$).

Algoritmo	Media	Mediana	Mínimo	Máximo	G
ODE	0.0644	0.0639	0.0632	0.0688	20
OPSO	0.1989	0.1997	0.0929	0.2991	1
DE	0.0641	0.0637	0.0632	0.0664	20
PSO	0.2005	0.1795	0.0760	0.2991	3

Tabla 5: Datos estadísticos de los algoritmos

Fuente (Autor)

Además, se posee una tabla de resumen con los datos relevantes de las respuestas de los controladores.

Algoritmo	Tiempo pico	Tiempo asentamiento	Sobre-pico	Tiempo de retardo	Error en estado estacionario
ODE	20ms	36ms	0,95%	7ms	No
OPSO	21ms	56ms	0	6,7756ms	Si
PSO	17ms	41ms	0,31%	6,5ms	Si
DE	21ms	55ms	1%	5ms	No

Tabla 6: Resultados de la respuesta de los controladores de impedancia.

Fuente(Autor)

6.8 Análisis estadístico

Para el análisis estadístico primeramente se corroboraron si los resultados de los 80 experimentos poseen una distribución normal o no, para esto se utiliza el test de Kolmogorov-Smirnov. [35]

La normalidad de las muestras permite la utilización del teste ANOVA, este teste permite conocer si los datos provienen de poblaciones diferentes, esto es de suma importancia, ya que si los datos provinieran de las mismas poblaciones no se podrían comparar (no se puede comparar migajas de un mismo pan).

La tabla del test ANOVA proporcionada por al implementarse es la siguiente:

Source	SS	Df	MS	F	Prob>F
Colums	0.3672	3	0.1224	35.54	1.8411e-14
Error	0.2617	76	0.0034		
Total	0.6289	79			

Tabla 7: Resultado de la Anova proporcionada por Matlab

Fuente (Autor)

De la tabla anterior el valor que permite conocer si las muestras son de diferentes populaciones es el valor de $Prob > F$, el cual si es menor a $0{,}05$ podemos decir con un 95% de certeza que las muestras son de poblaciones diferentes, como el valor de probabilidad es de 1.8411^{-14} podemos afirmar con un 99% de probabilidad que las muestras son de diferentes poblaciones.

Además de la tabla de ANOVA se pueden observar cómo están distribuidos los datos.

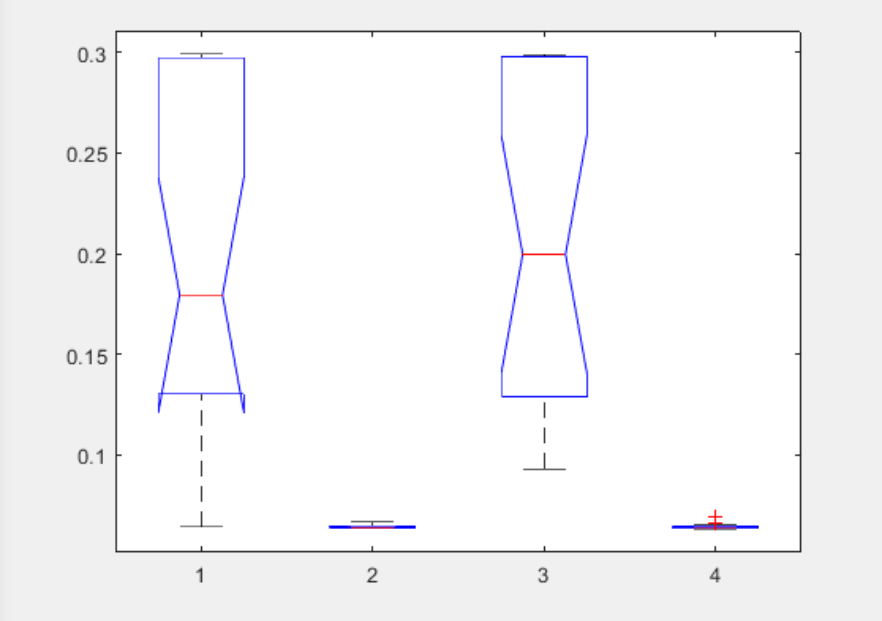

Figura 48: Distribución de los datos de los experimentos PSO, DE, OPSO, ODE

Fuente (Autor)

Donde el primer grupo son los datos del PSO, lo cual corrobora perfectamente con las curvas de convergencia, las cuales llegan a valores muy distantes , por esta razón los datos se muestran separados, este comportamiento se observa también en el grupo tres, el cual pertenece al OPSO, en contraposición se tiene los datos de DE (el cual es el grupo dos) donde el rango de convergencia era más pequeño por consiguiente los datos se muestran más cercanos, de igual forma para el ODE que es el grupo cuatro. La línea roja es la mediana de los datos.

Con la información suministrada por el algoritmo ANOVA se puede concluir que las muestras son de diferentes poblaciones, permitiendo compararlas entre ellas.

Si se compara utilizando su media se da por ganadora el algoritmo de DE el cual posee la menor mediana, pero si lo que interesa es la respuesta del controlador (son valores más relevantes), el mejor algoritmo sería el ODE, el cual posee el menor tiempo pico 21 ms, el menor tiempo de asentamiento $36\,ms$ y el menor sobre pico el cual es de 0,95% , además que no posee error en estado estacionario. Solamente pierde con el algoritmo DE canónico en el tiempo de subida el cual es de 7 ms y el del DE es de 5 ms

7 Conclusiones

Los trabajos realizados en la Universidad de Brasilia fueron satisfactorios, se logró cumplir con todos los objetivos establecidos, los cuales fueron en tres áreas:

- Ayudar en la fabricación de circuitos impresos.
- Cooperar en la sintonización de un controlador de impedancia para una mano robótica
- Colaborar en el desarrollo de Sistemas en Chip (ZYNQ7-SoC).

En la primera debido a la resolución de las máquinas de fabricación de circuitos impresos de la Universidad de Brasilia solo se logró ayudar hasta la etapa de diseño de la placa, esta se mandó a fabricar en china y se importó a Brasil en el mes de diciembre del año 2018 para su uso en la creación del motor.

La segunda área fue la más importante, ya que de ello se logró colaborar para la redacción del artículo científico, esta implico la asistencia a la disciplina llamada sistemas bioinspirados para ingeniería, además de la utilización de ordenadores, de ciertas características específicas para la implementación de estos algoritmos, entre las que se destaca una buena memoria RAM (las cuales oscilaba entre 16 y 32 Gb). Este acompañamiento implico la utilización de varias herramientas adicionales como, por ejemplo, el simulador del doctorando Sergio Pertuz para lograr llegar al objetivo de realizar la sintonización del controlador mediante el simulador de la mano robótica en el tiempo de la estancia de investigación, ya que la creación de este simulador fácilmente se abría demorado más de 2 meses.

Los resultados obtenidos de este trabajo se utilizaron para la redacción de un artículo científico, el cual será postulado para congresos internacionales en el área de control.

Por último se realizaron trabajos en el área de Sistemas on chip, los cuales consistieron en la creación de un módulo maestro para la configuración de convertidor ADS a través de comunicación I2C, para la culminación de esta pasantía de investigación se tomó otra disciplina llamada “Sistemas Digitales”, en los cuales se adquirieron los conocimientos necesarios para terminar el trabajo de sistemas on chip, este último trabajo sirvió para la colaboración del mejoramiento a la mano robótica del doctorando Sergio Pertuz, con lo cual se busca hacer una versión más sofisticada de la mano robótica que se posee actualmente.

Por último, es una buena idea realizar estancia de investigación sí se desea continuar con la maestría, ya que permite obtener becas (tal como se logró en este trabajo) además de poder homologar materias de la misma (ya que a medida que se está haciendo el trabajo de grado se van tomando materias las cuales sirven para la maestría).

8 Bibliografía

[1] L. Rocchetti, A. Amato, y F. Beolchini, «Printed circuit board recycling : A patent review», *J. Clean. Prod.*, vol. 178, pp. 814-832, 2018.

[2] S. H. Ahmed, N. Razzaq, Z. Malik, U. Qadeer, I. Sarfraz, y A. Sharif, «Design & Fabrication of MATLAB Based Solar Powered CNC Machine», pp. 265-268, 2017.

[3] [en línea] Disponible: https://www.autodesk.com/products/eagle/overview.

[4] R. Eberhart y J. Kennedy, «A New Optimizer Using Particle Swarm Theory», pp. 39-43.

[5] S. Russell y P. Norving, *Artificial Intelligence*, 2.ª ed. New Jersey: Pearson Education, 1996.

[6] C. Eugenia y T. Torres, «Inteligencia colectiva: enfoque para el análisis de redes», *Estud. Gerenciales*, vol. 30, n.º 132, pp. 259-266, 2014.

[7] J. Kennedy y R. Eberhart, «Particle Swarm Optimization», pp. 1942-1948, 1995.

[8] Hui Wang, Hui Li, Yong Liu, Changhe Li and Sanyou Zeng, "Opposition-based particle swarm algorithm with cauchy mutation," *2007 IEEE Congress on Evolutionary Computation*, Singapore, 2007, pp. 4750-4756.

[9] M. Clerc y J. Kennedy, «The Particle Swarm — Explosion , Stability , and Convergence in a Multidimensional Complex Space», vol. 6, n.º 1, pp. 58-73, 2002.

[10] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, Anchorage, AK, USA, 1998, pp. 69-73.

[11] A. Mottahedi, F. Sereshki, y M. Ataei, «Overbreak prediction in underground excavations using hybrid ANFIS-PSO model», *Tunn. Undergr. Space Technol.*, vol. 80, n.º April 2017, pp. 1-9, 2018.

[12] G. Spavieri, R. T. M. Ferreira, R. A. S. Fernandes, G. G. Lage, D. Barbosa, y M. Oleskovicz, «Particle Swarm Optimization-based approach for parameterization of power capacitor models fed by harmonic voltages», *Appl. Soft Comput. J.*, vol. 56, pp. 55-64, 2017.

[13] R. Laina, F. E. Lamzouri, E. Boufounas, y A. El, «ScienceDirect ScienceDirect The First International Conference On Intelligent Computing in Data Sciences Intelligent control of a DFIG wind turbine using a PSO evolutionary algorithm», *Procedia Comput. Sci.*, vol. 127, pp. 471-480, 2018.

[14] R. Storn y K. Price, «Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces», pp. 1-12, 1995.

[15] X. Zhang, J. Chen, B. Xin, y H. Fang, *Online Path Planning for UAV Using an Improved Differential Evolution Algorithm*, vol. 44, n.º 1. IFAC, 2011.

[16] S. K. Goudos, M. Deruyck, D. Plets, L. Martens, y W. Joseph, «Optimization of Power Consumption in Wireless Access Networks Using Differential Evolution with Eigenvector Based Crossover Operator».

[17] Z. Juan, «Impulsive Control Method Based on Improved Differential Evolution Algorithm», pp. 3777-3781, 2011.

[18] Q. Sheng, Z. Lei-shan, y Y. Yi-xiang, «A clonal selection based differential evolution algorithm for double-track railway train schedule optimization», pp. 155-158, 2010.

[19] R. K. Ursem, «Diversity-Guided Evolutionary Algorithms».

[20] J. Riget, J. S. Vesterstrøm, A. Universitet, N. Munkegade, y D.-A. C, «A Diversity-Guided Particle Swarm Optimizer – the ARPSO», 2002.

[21] L. Han, «A Novel Opposition-Based Particle Swarm Optimization for Noisy Problems», n.º Icnc, pp. 0-5, 2007.

[22]N. S. Voros y K. Masselos, *System Level Desing of Reconfigurable Systems-on-Chip*, 1.ª ed. Netherlands: Springer, 2005.

[23] A. Moore y R. Wilson, *FPGAs FOR DUMMIES*, 2.ª ed. New Jersey: Jhon Wiley & Sons, Inc, 2017.

[24] V. A. Pedroni, *Digital ELECTRONICS AND DESING WITH VHDL*, 1.ª ed. United States: Elsevier Inc., 2008.

[25] S. Hauck y A. Dehon, *RECONFIGURABLE COMPUTING*, 1.ª ed. United States: Elsevier Inc., 2008.

[26] H. Gürsoy y M. Ö. Efe, «CONTROL CONTROL SYSTEM SYSTEM IMPLEMENTATION IMPLEMENTATION ON ON AN FPGA PLATFORM PLATFORM», *IFAC-Pap.*, vol. 49, n.º 25, pp. 425-430, 2016.

[27] Z. Hajduk, «Neurocomputing Reconfigurable FPGA implementation of neural networks», *Neurocomputing*, vol. 308, pp. 227-234, 2018.

[28] M. S. Kavitha y P. Rangarajan, «ScienceDirect An optimized reconfigurable algorithm for FPGA architecture oriented IoT applications», *Cogn. Syst. Res.*, vol. 52, pp. 335-341, 2018.

[29] K. Ba, B. Yu, Z. Gao, Q. Zhu, G. Ma, y X. Kong, «An improved force-based impedance control method for the HDU of legged robots», *ISA Trans.*, 2018.

[30] B. Heinrichs, N. Sepehri, y A. B. Thornton-trump, «and AB.», pp. 46-52, 2000.

[31] F. D. E. Tecnologia, «Dissertação de mestrado sistema embarcado baseado em arquiteturas reconfiguráveis do controle dinâmico de uma mão robótica sintonizado com algoritmos bioinspirados», 2017.

[32] Q. Xu, «Robust Impedance Control of a Compliant Microgripper for High-Speed Position / Force Regulation», vol. 62, n.º 2, pp. 1201-1209, 2015.

[33] C. Semini *et al.*, «Towards versatile legged robots through active impedance control», *Int. J. Robot. Res.*, vol. 34, pp. 1003-1020, 2015.

[34]J. Wu, Z. Yang, y D. Wu, «Impedance control of secondary regulated hydraulic crane in the water entry», *Ocean Eng.*, vol. 169, n.º 51679101, pp. 134-143, 2018.

[35]R. Wilcox, “Kolmogorov-Smirnov Test,” in SpringerReference. Berlin/Heidelberg: Springer-Verlag, 1998, pp. 83–90.