BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA ELECTRÓNICA

MAESTRÍA EN CIENCIAS DE LA ELECTRÓNICA
OPCIÓN EN AUTOMATIZACIÓN

"DESARROLLO DE UNA RED NEURONAL CONVOLUCIONAL PARA REALIZAR LA IDENTIFICACIÓN PARAMÉTRICA DE CUATRO PARÁMETROS DE UN ROBOT CARTESIANO DE TRES GRADOS DE LIBERTAD"

T E S I S

Presentada para obtener el título de:

Maestro en Ciencias de la Electrónica Opción en Automatización

Presenta:

Ing. Michelle Guerra Marín

Directores:

Dra. María Aurora Diozcora Vargas Treviño (FCE-BUAP)

Dr. Sergio Vergara Limon (FCE-BUAP)

Director externo:

Dr. Jesús López Gómez (DAIA-UJAT)

Julio 2024

*Becario CONAHCYT

BUAP

Resumen

En esta tesis, se ha desarrollado una red neuronal convolucional para la identificación paramétrica de un robot cartesiano de tres grados de libertad. Tradicionalmente, las metodologías convencionales de identificación paramétrica utilizan principalmente mínimos cuadrados para determinar los parámetros dinámicos, lo cual es laborioso debido a la necesidad de hallar una trayectoria óptima para una identificación precisa. Dada la eficacia demostrada de las redes neuronales convolucionales en tareas de clasificación y regresión de datos, se ha adoptado esta arquitectura en el presente trabajo. Aunque en la literatura existente se han encontrado estudios que utilizan redes neuronales artificiales en la identificación paramétrica, estos suelen emplearse como herramientas auxiliares y no como métodos directos para la determinación de parámetros. En esta investigación, para realizar la identificación paramétrica, se emplea una imagen generada a partir de las señales de posición, velocidad, aceleración y torque, utilizando el modelo dinámico del robot cartesiano como caso de estudio. Se propone una estructura de red neuronal convolucional específica para este caso, la cual es entrenada mediante la técnica de generación de imágenes y un conjunto de datos de simulación del robot experimental, utilizando el gradiente descendente del error. Se ha diseñado un algoritmo de identificación paramétrica que incorpora una métrica de similitud considerando señales tanto en el dominio del tiempo como en el de la frecuencia. Los resultados de las simulaciones muestran que los algoritmos pueden identificar los parámetros dinámicos con una similitud aproximada del 98 %. Además, los resultados experimentales de la identificación paramétrica revelan que el algoritmo diseñado es capaz de determinar los parámetros dinámicos de dos robots experimentales, superando el 90 % de similitud en el caso de estudio con 12 parámetros dinámicos para el robot cartesiano. La principal contribución de esta tesis es el desarrollo de una metodología innovadora de identificación paramétrica basada en una red neuronal convolucional, que permite determinar los parámetros dinámicos de un robot utilizando una imagen de entrada generada a partir de las señales de movimiento del robot, ofreciendo una solución eficaz y precisa para este tipo de problemas en el ámbito de la robótica.

Agradecimientos

Gracias, Dios, por el don de la vida. Me siento profundamente agradecido por el apoyo y la guía de tantas personas excepcionales que me han acompañado en este camino.

A mis asesores, la Dra. María Aurora Diozcora Vargas Treviño, el Dr. Sergio Vergara Limon y el Dr. Jesús López Gómez, merecen mi más sincero reconocimiento. Su dedicación incansable, sabiduría y aliento incondicional fueron el sostén que necesitaba para avanzar con confianza en esta travesía académica.

El Dr. Carlos Leopoldo Carreón Díaz de León merece una mención especial por ser una fuente invaluable de conocimiento y apoyo durante todo el proceso de investigación.

Mi gratitud se extiende también a todos los profesores cuya sabiduría moldeó mi camino académico. Su enseñanza no solo me proporcionó conocimientos, sino que también me inspiró a cultivar la curiosidad incansable y la búsqueda constante de la excelencia.

A mi jurado revisor, compuesto por la Dra. Olga Guadalupe Félix Beltrán, la Dra. Amparo Palomino Merino y el Dr. César Martínez Torres, les agradezco sinceramente por sus observaciones perspicaces, comentarios constructivos y preguntas que me desafiaron a crecer y mejorar en mi trabajo.

Quiero expresar mi más profundo agradecimiento a mi hermosa familia, cuyo apoyo incondicional ha sido fundamental para la culminación de este trabajo. A mi madre, Rosa Isela, fuente inagotable de amor y fortaleza. Tus sacrificios han sido mi inspiración y tu amor, mi refugio. Gracias por ser mi luz en la oscuridad, por cada sonrisa, cada abrazo, cada consejo. Sin tu amor y apoyo, este sueño no habría sido posible; a mis abuelos, Eneas y Amada †, gracias, por ser el faro que ilumina mi camino con amor y sabiduría. Su ejemplo y cariño son el tesoro más valioso de mi vida, a mi querido hermano, Emmanuel, a mi tía que me quiere mucho, Zeila, a mis primos, Xhunaxhi Juquila y Luis Carlos, por su compañía y estímulo en cada paso de este camino. Imposible no mencionar a mis apreciables tíos, Josefina y Gamaliel †. Además, a mi novia, Ximena, cuyo amor inagotable y compañía han sido mi fortaleza.

A la Benemérita Universidad Autónoma de Puebla y a la Facultad de Ciencias de la Electrónica, les agradezco por brindarme la oportunidad de perseguir mis sueños académicos y por proporcionar los recursos y la infraestructura necesarios para mi maestría.

Por último, pero no menos importante, agradezco a CONACYT por su generoso apoyo financiero, que ha hecho posible que muchos como yo persigan la excelencia académica y contribuyan al avance de la investigación y el desarrollo en nuestro país.

Dedicatoria

.

Dedico esta tesis a mi mamá Amada Marín Ruiz †, quien, aunque ahora se encuentra en el cielo, sigue siendo mi guía y mi luz en cada paso que doy. Su amor y enseñanzas perduran en mi corazón y su espíritu me acompaña siempre. Gracias, mami, por ser mi inspiración eterna.

Índice general

Índice de figuras

Introducción

La identificación paramétrica es una herramienta de gran utilidad debido a que permite diseñar nuevos modelos dinámicos y hallar sus parámetros, implementar control de trayectoria, estimación de fuerza, detección de fallas entre otras aplicaciones. Las metodologías convencionales de identificación paramétrica necesitan optimizar la trayectoria que realiza el robot para poder hallar los parámetros con mínimos cuadrados, sin embargo, cada nueva trayectoria necesita probarse en el robot real: optimizar esta trayectoria toma más tiempo que el algoritmo de mínimos cuadrados. Con la identificación paramétrica basada en la RCN no es necesario que las señales de torque del robot sean idénticas a las producidas por un modelo dinámico, solo es necesario que tengan una forma aproximadamente igual.

Varios estudios de identificación de robots, como [1], [2], [3], y [4], emplean acelerómetros, cámaras, funciones recursivas y resolutores para estimar la velocidad y aceleración de manipuladores robóticos. [5] presenta un teorema de convergencia para un algoritmo de identificación paramétrica. [6] investiga la variación térmica de un actuador harmonic drive, mientras que [7] informa sobre el juego mecánico de los engranes utilizados en actuadores, que incluyen máquinas de corriente directa [8] y actuadores lineales [9]. En [10] se emplea un sensor de fuerza para identificar el torque debido a parámetros dinámicos inerciales y determinar el torque de fricción sin un modelo previo. En [11] utiliza la identificación paramétrica en un brazo robot para detectar colisiones mediante un observador de momento de estado no lineal, inmune al ruido de los sensores de posición. [12] utiliza un software para mejorar la calidad de las señales de sensores y mejorar la identificación paramétrica en un robot de dos grados de libertad.

La señal a los controladores de servomotores se usa como medida indirecta del torque, pero también se emplea la corriente de actuadores para estimar el torque [13], [14]. Algunos algoritmos buscan reducir el ruido en las señales de posición, utilizando métodos como mínimos cuadrados y máxima verosimilitud [15]. En [16], se identifican parámetros de un robot de seis grados de libertad sin considerar la cuantificación de la señal de movimiento. Filtros pasa baja se aplican para disminuir distorsiones y estimar velocidad y aceleración, como se ve en estudios como [17], [18], [19], [20], [21] y [22]. [23] emplea un filtro promediador en las señales de posición y torque.

Más estudios analizados emplean el método de mínimos cuadrados para la identificación de robots, como se informa en [15], [18], [24], [25], [13], [26], [27], [28], [29].

En [30] compara mínimos cuadrados con un método indirecto de identificación paramétrica en un robot de seis grados, favoreciendo los resultados de mínimos cuadrados. [8] realiza una comparación entre mínimos cuadrados ponderados y un filtro de Kalman extendido para la identificación paramétrica. [14] utiliza mínimos cuadrados para identificar parámetros dinámicos en un robot 3-RPS mediante multicriterio de optimización. En [31], se describe la identificación paramétrica de un robot industrial con análisis de repetibilidad, observando cambios en la fricción según la temperatura de las uniones del robot.

En [32], se demuestra que desacoplar la mayoría de las ecuaciones del modelo dinámico permite obtener una identificación paramétrica más robusta en comparación con las metodologías tradicionales que utilizan mínimos cuadrados. En [19]

propone un nuevo modelo de fricción para predecir el torque de un robot manipulador ante cambios bruscos de movimiento, utilizando mínimos cuadrados para su identificación paramétrica y validación de su efectividad. En [33]), se realiza la identificación paramétrica de un robot con uniones prismáticas donde el modelo de fricción varía con la posición del mecanismo, utilizando mínimos cuadrados.

Esta investigación tendrá las siguientes aportaciones:

- El desarrollo de una red neuronal convolucional para la identificación paramétrica de cuatro parámetros de un robot cartesiano.
- La creación de un conjunto de datos de entrenamiento para la red neuronal convolucional utilizando el modelo dinámico del robot cartesiano.
- Una nueva forma de obtener parámetros de modelado, la cual puede ser extrapolado a una amplio conjunto de aplicaciones.

Esta investigación contribuirá al campo de conocimiento mediante la implementación de un algoritmo de identificación paramétrica basado en una red neuronal convolucional para un robot cartesiano utilizando un modelo dinámico cuyos parámetros puedan reconstruir las señales del robot experimental. El estudio es pertinente debido a que se ha encontrado en el estado del arte diversas aplicaciones actuales donde los parámetros dinámicos de los robots manipuladores son utilizados como base de otros estudios de control, entre otras aplicaciones. Actualmente las redes neuronales convolucionales son ampliamente utilizadas en la robótica y con esta investigación se aprovechará un área de oportunidad en la identificación de parámetros de robots cartesianos. Los productos esperados de este tema de tesis son:

- Un algoritmo de identificación paramétrica de un robot cartesiano basado en una red neuronal convolucional.
- Una presentación en un congreso internacional.

El presente trabajo de tesis incrementará la infraestructura de plataformas experimentales de control en la Facultad de Ciencias de la Electrónica, el proyecto se puede considerar de gran impacto para la imagen de nuestra institución ya que la infraestructura que se generaría en base al desarrollo propuesto junto con la infraestructura existente fortalecerá a la Maestría en Ciencias de la Electrónica, Opción Automatización. También elevara la productividad ya que se contaría con un prototipo institucional el que puede generar al menos una publicación en congreso internacional.

Objetivos

Los objetivos general y específicos de este trabajo de tesis se enuncian a continuación.

Objetivo general

Diseñar e implementar una red neuronal convolucional para que realice la identificación paramétrica de cuatro parámetros de un robot cartesiano de tres grados de libertad.

Objetivos particulares

1. Estudiar el modelo dinámico del robot de tres grados de libertad.
2. Estudiar los fundamentos de redes neuronales.
3. Diseñar una red neuronal convolucional para la obtención de cuatro parámetros del robot de 3 gdl.
4. Actualización de la eletrónica para mejorar el rendimiento del robot.
5. Implementar la red neuronal convolucional.
6. Realizar pruebas experimentales para evaluar el rendimiento de la red neuronal.
7. Comparar los resultados obtenidos con la red neuronal y los resultados del modelo dinámico.
8. Publicar los resultados.
9. Escritura de la tesis

Organización de la Tesis:

Capítulo 1: En este capítulo se ofrece una panorámica del sistema mecatrónico en estudio, abordando tanto sus elementos mecánicos clave como las especificaciones de su componente electrónico, que incluyen el firmware y el software. Se destaca especialmente el proceso de actualización llevado a cabo en la electrónica para potenciar el rendimiento del sistema, una medida necesaria para su integración exitosa en el programa de posgrado.

Capítulo 2: En el segundo capítulo se presenta la metodología utilizada para derivar el modelo dinámico del sistema. Se inicia con un análisis preliminar enfocado en un grado de libertad del robot, extendiéndolo posteriormente para abarcar los tres grados de libertad que componen el mecanismo. Asimismo, se detalla el proceso de obtención del modelo de fricción propuesto, el cual se incorporará al modelo dinámico para investigar sus efectos observados en el mecanismo.

Capítulo 3: En esta sección se expone la metodología de identificación propuesta, detallando la arquitectura de la red neuronal convolucional diseñada para el caso de estudio. Se incluye el modelo dinámico del robot utilizado y la técnica para generar imágenes a partir de señales como posición, velocidad, aceleración y torque. Además, se describe la creación del repositorio de entrenamiento y se explica el uso del gradiente descendente en las redes neuronales convolucionales. Por último, se presenta el algoritmo de identificación desarrollado para este estudio.

Capítulo 4: Este capítulo presenta un resumen de los métodos convencionales empleados para llevar a cabo la identificación paramétrica, destacando las peculiaridades operativas de cada uno. Asimismo, se ofrece una descripción detallada de la metodología de identificación propuesta, la cual se basa en el uso de los parámetros entrenados de la red neuronal, el modelo dinámico del sistema y los datos experimentales del robot obtenidos a través del firmware. Se abordan también los parámetros específicos a identificar y se detalla su caracterización en el contexto de la simulación.

Capítulo 5: En este capítulo se muestran los resultados de entrenamiento de las redes neuronales convolucionales desarrolladas, algunas imágenes de los repositorios de imágenes creados para cada robot y el costo computacional de la convolución utilizando distintas funciones en Matlab. Se muestran resultados de simulación para los dos algoritmos de identificación. Se describe brevemente la arquitectura de los robots experimentales utilizados y los resultados de identificación paramétrica experimentales.

Capítulo 1

Robot cartesiano de tres grados de libertad

Dentro de la Maestría en Ciencias de la Electrónica opción en Automatización (MCEA) de la Benemérita Universidad Autónoma de Puebla (BUAP) se han desarrollado diversos trabajos de complejidad destacable, dentro de estos se encuentra el cual fue desarrollado mecánicamente y descrito analíticamente para obtener su modelo dinámico. El robot posee un mecanismo basado en el control numérico por computadora (CNC) [34].

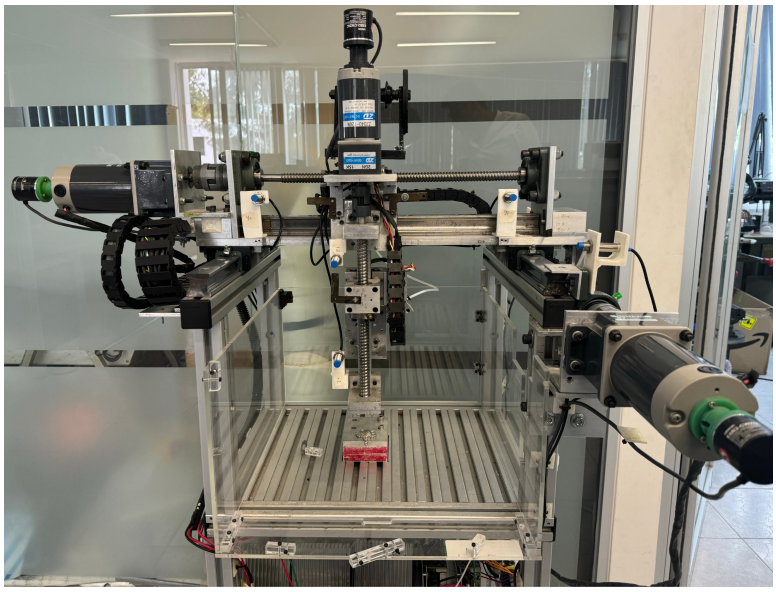


Figura 1.1: Robot cartesiano tres grados de libertad.

1.1. Diagrama bloques general del sistema

En la Figurea 1.2 se presenta el diagrama general del robot CNC de tres grados de libertad, configurado para operar como un escáner 3D.

La comunicación con la computadora se realiza mediante un módulo Wi-Fi, lo que facilita la interacción entre el usuario y el microprocesador. Esto permite al usuario enviar instrucciones al microprocesador, que simultáneamente procesa las señales recibidas de los sensores y envía señales de control en forma de generadores de ondas PWM y secuencias de pasos para el motor encargado de alimentar el material. Estas señales pasan a través de una etapa de acoplamiento antes de ser entregadas a los actuadores, lo que cierra los lazos de control.

La etapa de acoplamiento de señales incluye una sección de potencia para los motores DC, que utiliza puentes H en configuración Darlington para activar los motores en ambas direcciones, proporcionando una corriente máxima de 25A en modo de saturación y corte. Además, hay una etapa de acoplamiento para el extrusor de material, que consiste en un circuito de potencia con transistores NPN en configuración Darlington, que amplifica la corriente del microprocesador de 1mA a 2.5A, permitiendo el control de la resistencia del extrusor. De manera similar, existe una etapa de potencia para controlar la temperatura de la cama caliente, también utilizando transistores tipo NPN.

Finalmente, el alimentador de material está acoplado a una etapa de potencia compuesta por dos puentes H (uno para cada bobina), lo que permite controlar las dos bobinas del motor y, por tanto, regular su posición y velocidad.

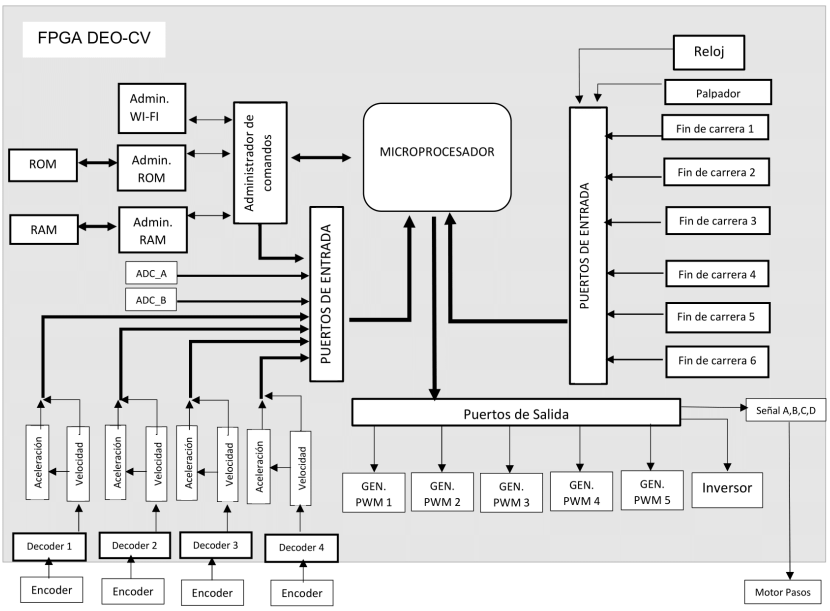


Figura 1.2: Diagrama a bloques del robot cartesiano (Figura 1.1).

1.2. Características mecánicas

El robot está fabricado en una estructura fabricada en acero 1018 con la característica de que cada grado de libertad está delimitado por rieles de alta precisión lo que aumentó considerablemente la precisión del funcionamiento del robot, ya que

dichos rieles se encuentran a su vez sobre una base rígida, que son los carros lineales. La transformación de movimiento rotacional a axial de los motores, se hace a través de tornillos embalados con sus respectivos componentes el cual siguen favoreciendo al funcionamiento del mecanismo. Cada motor está acoplado con un reductor que permita generar un torque de 3-4 Nm. Finalmente se tienen acoplados encoder giratorios lo cual permite determinar la posición actual del sistema, en la Tabla 1.1 se describen las características generales del robot.

Descripción	Requerimiento
Número de grados de Libertad	3(X,Y,Z)
Ejes X y Y	Desplazamiento: $X = 276$ mm Tornillo embalado de 12 mm Guía de alta precisión 32 mm Bloque para guía de 32 mm
Eje Z	Desplazamiento:145 mm Tornillo embalado de 12 mm Guía de alta precisión de 15 mm Bloque para guía de 15 mm
Tamaño (mm)	950 x 1470 x 860
Resolución	$\Delta x = 0.286\mu$m $\Delta y = 0.287\mu$m $\Delta z = 0.171\mu$
Velocidad	$v_x = 35$ mm/s $v_y = 25$ mm/s $v_z = 21$ mm/s
Software	Abierto
Conectividad	Wifi
Sistemas Operativos	Abiertos

Tabla 1.1: Caracteríticas generales del Robot Cartesiano.

Se describen los diversos elementos que componen el sistema y hacen posible la magnitud y precisión que posee el robot, esto posible a la mecánica del sistema, como lo son los rieles de alta precisión, los carros lineales de precisión, los tornillos embalados que sirven para la transformación de movimiento rotacional a lineal, los motores DC de uso robusto que cuentan con reductores para transformación de torque, los encoders que permiten la determinación de la posición de los motores, la herramienta utilizada, la comunicación realizada por WiFi mediante un módulo especializado y la tarjeta de implementación utilizada que posee un FPGA tipo Cyclone V FPGA 5CEBA4F23C7N.
Entre los demás componentes del sistema, se encuentran:

- **Riel de alta precisión.** Ambos son implementados en cada uno de los ejes del robot como medio para obtener un desplazamiento lineal con una fricción mínima. Al final de cada uno de estos carros, es donde se puede ubicar los sensores de final de carrera.

- **Tornillo embalado.** El robot posee un tornillo embalado que es responsable de la transformación del movimiento rotatorio del motor en un desplazamiento lineal. Se tiene un tornillo por cada uno de los ejes de movimiento del robot y el comportamiento interno de este tipo de tornillo se lleva a cabo por medio de balines internos que se desplazan por la tuerca del tornillo.
- **Interruptores de fin de carrera.** El robot tiene instalados en cada uno de los ejes sensores de fin de carrera que se adaptaron en su momento para evitar que los carros de alta precisión sobrepasaran los límites del área de trabajo.
- **Motores DC y reductor.** Motores DC y reductores El sistema mecánico emplea motores de uso robusto para los tres ejes del robot. Los motores implementados son diferentes para los ejes X-Y y el eje Z.
- **Encoders E6B2-CWZ6C.** Este modelo de encoders provee de 1000 pulsos por revolución y puede ser alimentado en un rango que va de 5-24 V con un consumo de corriente máximo de 80mA.
- **Módulo Wi-Fi.** Este módulo se trata de un modelo de Wi-Fi RN-XV que es empleado para establecer una comunicación entre el sistema mecatrónico y la computadora.
- **Tarjeta de implementación DE0-CV.** Esta tarjeta puede ser adquirida por medio de internet y tiene implementado un FPGA modelo Cyclone V 5CEBA4F23C7N, con elementos propios.
- FPGA Cyclone V 5CEBA4F23C7N
- 49,000 elementos lógicos programables
- 3080 Kbits de memoria embebida
- Memoria SDRAM de 64MB x 16 bits de bus de datos
- 4 PLLs fraccionales
- Memoria EPCS64
- 2x20 pines de propósito general (72 pines I/O, 2 pines de alimentación de 3.3VCD y 2 de 5VCD con sus respectivas tierras)
- 6 displays de 7 segmentos
- 10 LEDs
- 10 interruptores
- 4 pulsadores de propósito general
- 1 pulsador para el reset de la CP

Caracterización de un motor DC

Se describen los pasos implementados que permitieron la caracterización de los motores para que tuvieran un adecuado funcionamiento en cada eje:

1. Primero se debe determinar el torque máximo del motor, para lo cual se realiza una variación del voltaje de alimentación con que se conecta el motor (sin exceder el límite de voltaje de trabajo recomendado en la hoja de datos).

2. Realizar el cálculo correspondiente para obtener la mitad del torque máximo al que trabaja el motor al alimentarlo con una señal de PWM al 50 % del ciclo de trabajo. En orden de poder lograr este paso, es necesario contar con el programa que te genere la señal de variación de frecuencia de la señal de PWM, la cual debe ir desde los 50 Hz hasta la frecuencia determinada por el usuario.

3. Elsiguiente paso consiste en seleccionar las frecuencias o frecuencia que proporcione el valor que sea más cercano a la mitad del torque máximo que se calculó en el paso anterior.

4. Se debe probar cada una de las secuencias seleccionadas al ir variando el ciclo de trabajo del 10 % al 90 % de su rango. Esto se realiza sin variar la alimentación del motor, empleando para ello el voltaje del torque máximo. Cuando se varía el ciclo de trabajo, se debe determinar el torque del motor para cada variación del PWM y con los datos obtenidos graficar lo que corresponde a la respuesta del motor.

5. Al final, se deben revisar las gráficas obtenidas y analizarlas para determinar la ecuación de la curva característica. Con esto, se tiene que seleccionar la frecuencia que proporcione la respuesta más cercana posible a un comportamiento lineal para el motor.

L

% Ciclo de trabajo de PWM	Torque(Nm) - Motor X	Torque(Nm) - Motor Y
0	0	0
10	2.8616	3.3124
12	3.9494	4.2924
14	4.3708	5.1058
16	5.2528	5.8016
18	5.9976	6.3406
20	6.4288	6.8208
22	7.595	7.6832
24	8.1144	8.477
26	8.428	9.31
28	9.555	9.996
30	10.388	11.0152
32	10.8192	11.7992
34	11.6032	12.348
36	12.348	13.1

Tabla 1.2: Caracterización de los motores X, Y con sus correspondientes frecuencias seleccionadas.

% Ciclo de trabajo de PWM	Torque(Nm) - Motor Z
0	0
10	0.0616077
20	0.3435096
30	0.6571488
40	0.840105
50	1.250823
60	1.437513
70	1.8930366
80	2.072259
90	2.5688544

Tabla 1.3: Caracterización del motor Z con su correspondiente frecuencia seleccionada.

Se muestran las gráficas obtenidas por medio de medición de variables, donde es posible observar el comportamiento de los motores para cada eje.

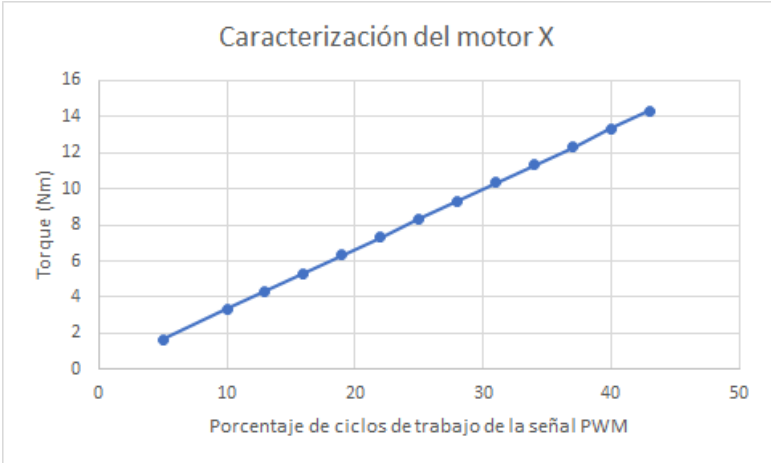


Figura 1.3: Curva característica del motor X.

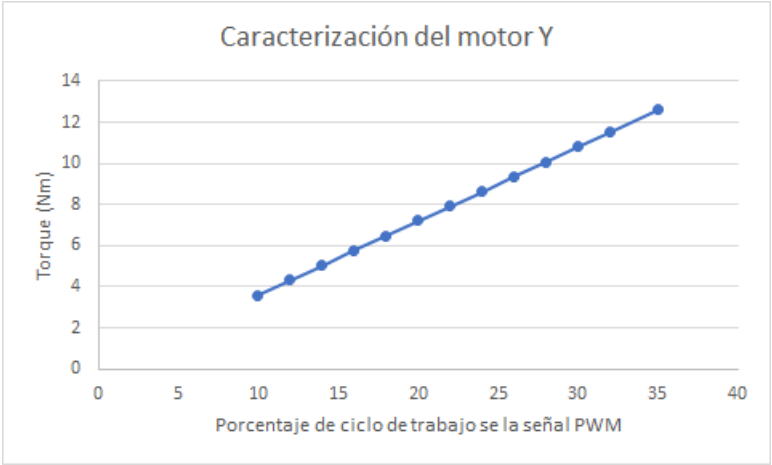


Figura 1.4: Curva característica del motor Y.

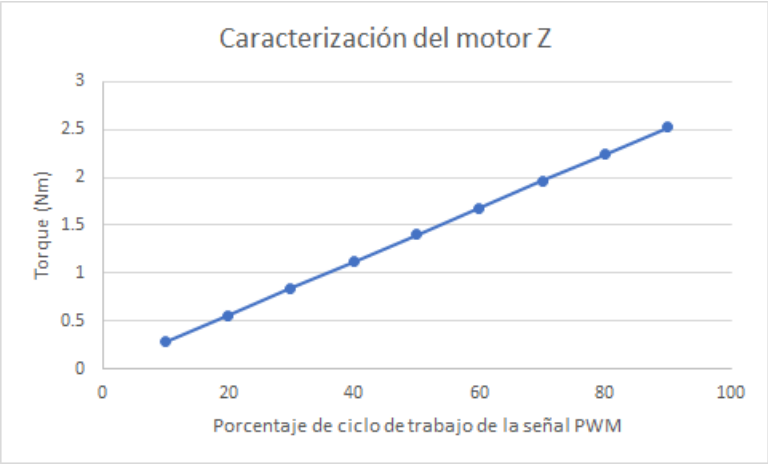


Figura 1.5: Curva característica del motor Z.

1.3. Principio de funcionamiento

Conjunto a la información previamente detallada, es posible observar que el generador PWM, el cual permite mandar una señal de control que tiene la forma de un pulso de ancho determinado, se vuelve sumamente importante para el funcionamiento del motor. Con la curva característica obtenida para el comportamiento del motor, es posible solicitar un torque y obtenerlo con la variación del ancho de pulso del PWM. Para lograr esto, se tiene desarrollado un firmware que genera la señal de PWM y que se encarga del control del ancho de pulso de este. La señal generada por este firmware es mandada de manera directa a una etapa de potencia que consiste en una serie de puentes H y que activan el motor. El firmware también se encuentra encargado del conteo que determinará la posición y la dirección de cada motor. Esto se logra con la implementación de un divisor de frecuencia, lo que se hace posible debido a que la frecuencia a la que trabajan los motores es menor a la frecuencia de trabajo del FPGA. Con esta parte, es necesario mencionar nuevamente al proceso de caracterización de los motores, que como se mencionó, permite obtener una frecuencia para cada motor que resulta la ideal para que el comportamiento de este resulte de manera similar al de un motor de transmisión directa.

Esto permite proporcionar una respuesta casi lineal que permite la manipulación y el control del movimiento, la velocidad y la dirección del motor de manera precisa. Es por estas razones que la etapa de caracterización es crucial para el correcto manejo del robot, puesto que permite simplificar el comportamiento de un motor (que generalmente se trata de una curva) para que la respuesta asemeje a la de una línea recta. Esto permite generar mejores y más precisos modelos dinámicos que representan el comportamiento del motor.

1.4. Sistema embebido

El sistema de control del robot es igual de importante que la parte mecánica del mismo, puesto que es lo que habilita al usuario para dar comandos al dispositivo. Para realizar el control del robot se empleó una tarjeta de desarrollo modelo DE0-CV con FPGA perteneciente a la familia Cyclone V, en la que se desarrolló un firmware embebido construido en el software Quartus II mediante el empleo del lenguaje AHDL.

Tanto el sistema de control, la parte mecánica y la obtención del modelo dinámico fue realizado en la MCEA de la BUAP. Este microprocesador se encuentra diseñado en base a una arquitectura MIPS (Microprocessor without Interlocked Pipeline Stages), y posee un procesador de 32 bits con arquitectura RISC (*Reduced Instruction Set Computer*) que cuenta con un banco de 32 registros de 32 bits. El firmware tiene un protocolo de comunicación Wi-Fi que permite mantener un enlace de datos entre una computadora y la tarjeta de adquisición de datos. A su vez, esto permite procesar las instrucciones recibidas de un usuario en lenguaje ensamblador y de esta forma realizar las correspondientes operaciones aritméticas con la finalidad de realizar el control del robot de manera eficiente y de acuerdo con las instrucciones, además los bloques necesarios para las mandar/recibir dichas instrucciones.

- **Administrador de Wi-Fi:** Permite al FPGA comunicarse con protocolo serial al módulo Wi-Fi.
- **Administrador de SDRAM:** Es un bloque de control de memoria que permite enviar comandos para escribir y leer datos de 32 bits en forma paralela. Se encarga de comunicarse con los datos que manda el administrador Wi-Fi enviados por la computadora.
- **Decodificador de señales:** Se encarga de procesar las señales de los encoder de los motores que controlan los ejes del motor. Los encoders incrementales de cuadratura para poder determinar la posición actual en cada eje, para interpretar las señales A y B de los encoders.
- **Convertidor para ADC:** El bloque ADC se encarga de obtener las señales de los ADC y determinar la temperatura actual de cada dispositivo.
- Interruptores de proximidad. Se encarga de leer los sensores de proximidad (fin de carrera) instalados en cada eje,
- **Generador PWM:** El firmware genera la señal de PWM y controla la velocidad de los motores.

El funcionamiento general del sistema estudiado se basa en la manipulación por parte del usuario a través de una interfaz de LabView donde es posible visualizar y modificar los parámetros del sistema, las posiciones deseadas para los motores y las ganancias de ajuste de los controles usados en cada eje del mecanismo. Esta interfaz permite también el envío del software de control hacia el microprocesador, lo que es posible con el uso de un protocolo de comunicación Wi-Fi proveniente de la computadora empleada para visualizar la interfaz. El robot establece comunicación con el sistema digital por medio de las funciones del FPGA de la tarjeta empleada (como se mencionó con anterioridad), la cual se encarga a su vez de ejecutar los seis algoritmos de control que emplea el robot CNC para funcionar: 3 lazos de control cerrado que corresponden a los ejes de movimiento.
La comunicación con la computadora que se establece por medio del módulo Wi-Fi habilita la comunicación entre el usuario y el microprocesador y permite que el usuario mande instrucciones a este, lo cual realiza al mismo tiempo que procesa las señales mandadas por los sensores, permitiendo regresar señales de control. En el acoplamiento de señales, se tienen etapas de potencia para los motores DC consistentes de puentes H en configuración Darlington que se encargan de activar los motores en ambas direcciones al proporcionar una corriente máxima de 25A en modo de saturación y corte. Para finalizar se tiene que el alimentador de material se encuentra acoplado con una etapa de potencia formada por dos puentes H (uno para cada bobina) que permite controlar las dos bobinas del motor y de esta forma regular su posición y velocidad.

1.5. Software de usuario

Es necesario saber la información que el microprocesador está mandando y a su vez mandar información a la máquina, es por eso que se tiene un interfaz usuario-máquina. Para el dispositivo de estudio, se tienen desarrollados dos software de usuario:

- Compilador de posiciones deseadas.
- Interfaz de usuario.

El **compilador de posiciones deseadas** es necesario para obtener las instrucciones para que la máquina construya una figura 3D y se generan por medio del procesamiento del diseño de una pieza 3D realizada en un software CAD (*computer assisted design*), lo que hace es traducir el diseño a instrucciones que el microprocesador pueda ejecutar. (Figura 1.6) La interfaz está desarrollado en LabVIEW, el programa solicita el archivo CAD, el cual lo compila y transforma a código G, después se compila para generar el código hexadecimal el cual produce las coordenadas para cada eje. La interfaz tiene la opción de generar un archivo para guardar las posiciones en un archivo de texto desde donde puede ser ingresado al otro software de usuario.

La segunda interfaz lleva acabo el control del robot y permite al usuario controlar la velocidad y posición de los 3 ejes (cada uno por separado),los controles de temperatura del extrusor, la cama caliente y el control de velocidad para el motor a pasos. El interfaz está implementada en LabVIEW, el software cuenta con 3 ventanas, en la Figura 1.6, muestra la interfaz en donde se puede descargar el archivo del código con las instrucciones necesarias para realizar el control del sistema y cerrar los lazos de control de la máquina.

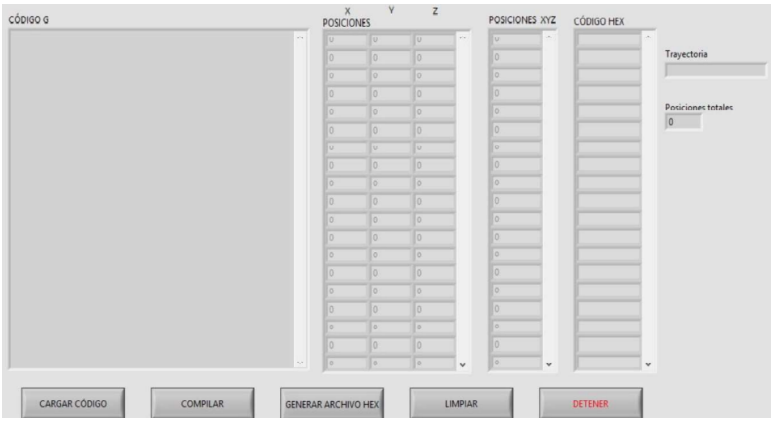


Figura 1.6: Interfaz gráfica para el compilador de posiciones deseadas.

En la pestaña 2 se realiza la descarga a la memoria RAM (Figura 1.7), del archivo con las posiciones deseadas para la realización de una pieza. Este archivo debe tener las instrucciones (coordenadas) en formato hexadecimal, lo que explica la importancia del primer software de usuario. Es importante mencionar que antes de realizar la descarga del archivo, la interfaz provee con la opción de asignar un offset para que al descargar los datos se empiecen a almacenar en una dirección determinada de la RAM.

Figura 1.7: Descarga de posiciones deseadas en la RAM.

La última pestaña es la ejecución del programa este a su vez contiene 3 sub-pestañas, la primera sub-pestaña (Figura 1.8) se encarga de de asignar los parámetros

de inicio (velocidades y temperaturas deseadas) para cada uno de los dispositivos controlados por el microprocesador. También permite asignar las ganancias K_s y K_p para los controles de lazo cerrado. Tiene un botón para iniciar o detener el programa.

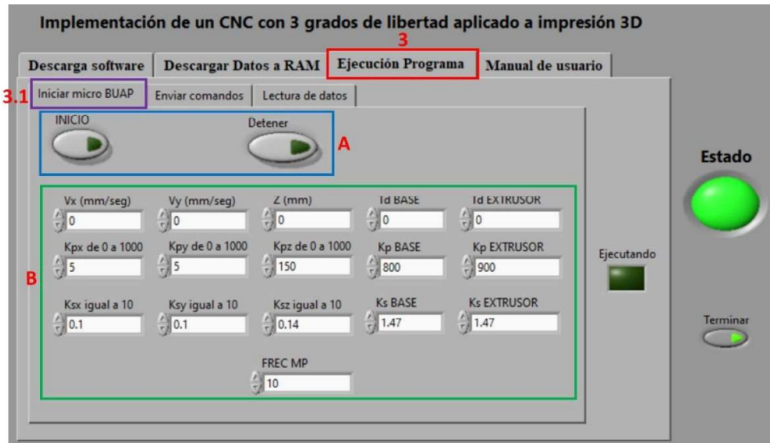


Figura 1.8: Pestaña de ejecución del programa: sub-pestaña de inicialización.

La segunda sub-pestaña, Enviar comandos (Figura 1.9), en este se mandan datos directos al microprocesador, mandar las posiciones y/o temperaturas deseadas en cada uno de los dispositivos controlados, variar las ganancias de los controles, etc.

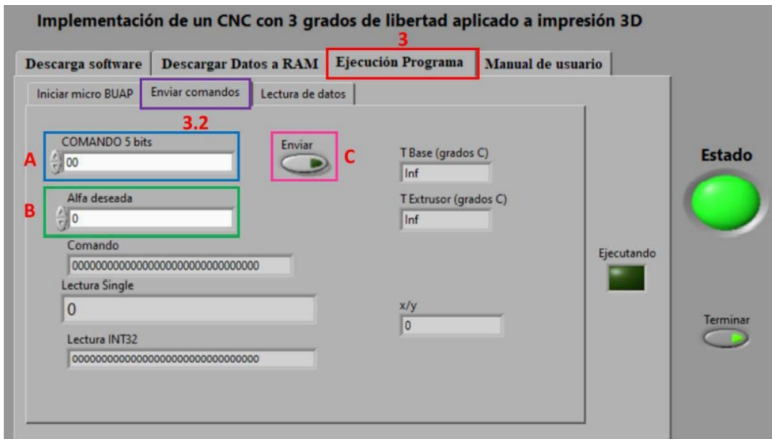


Figura 1.9: Sub-pestaña de envío de comandos.

La última sub-pestaña es la encargada de leer los datos que vienen del microprocesador (Figura 1.10), como la posición actual que viene de alguno de los ejes. Los datos son procesados por la interfaz, es por eso que tiene la opción de poder guardar los datos en un archivo con extensión .tex para posteriormente realizar un análisis y procesamiento de los mismos.

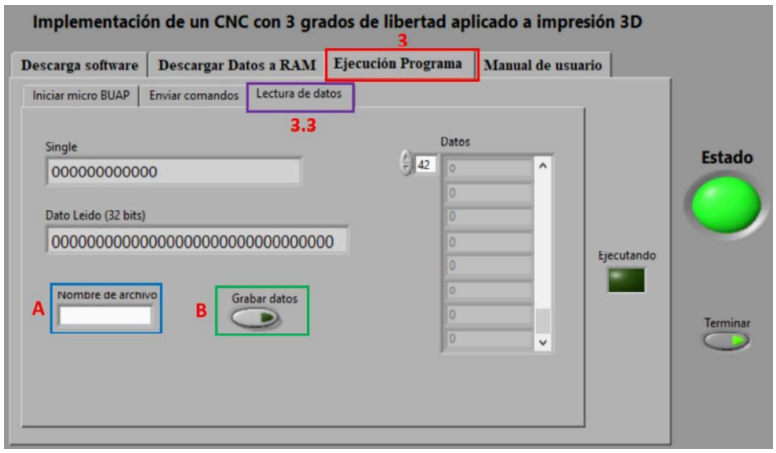


Figura 1.10: Sub-pestaña Lectura de datos.

Estado inicial

El estado de un prototipo implica que puede tener fallas en el diseño de este mismo, lo que ocasiona que el funcionamiento se vea afectado, y en este prototipo de un robot cartesiano de tres grados de libertado, no es la excepción, esto quiere decir que cuenta con ciertos elementos que presentan fallo en su funcionamiento independiente lo cual ocasione un fallo en todo el funcionamiento del robot. Dentro de los principales problemas que presenta el robot, es el diseño e implementación de la electrónica, ya que al contar con distintas tarjetas y estar interconectadas por cables tipo "jumper" provoca falsos en la comunicación e incluso la desconexión de estos, desafortunadamente el prototipo fue entregar sin un diagrama de conexiones o identificador de señales, por lo que el primer objetivo fue identificar cada una de las tarjetas que el robot contiene, para poder comenzar con la desconexión y así poder trabajar con los elementos que requiere atención de manera individual.

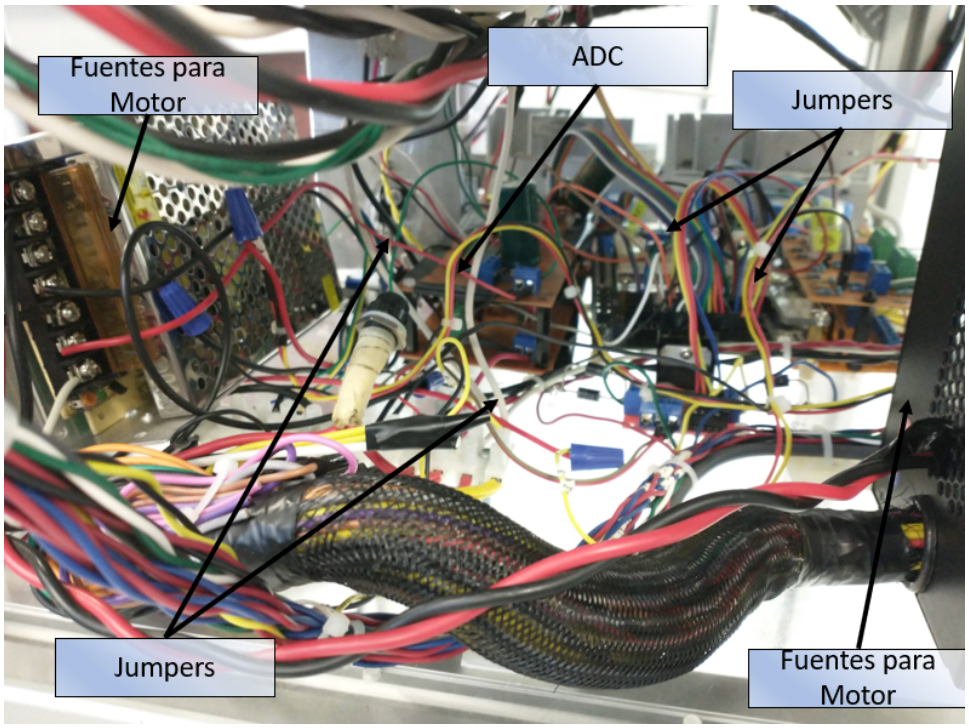


Figura 1.11: Cableado interno inicial (1).

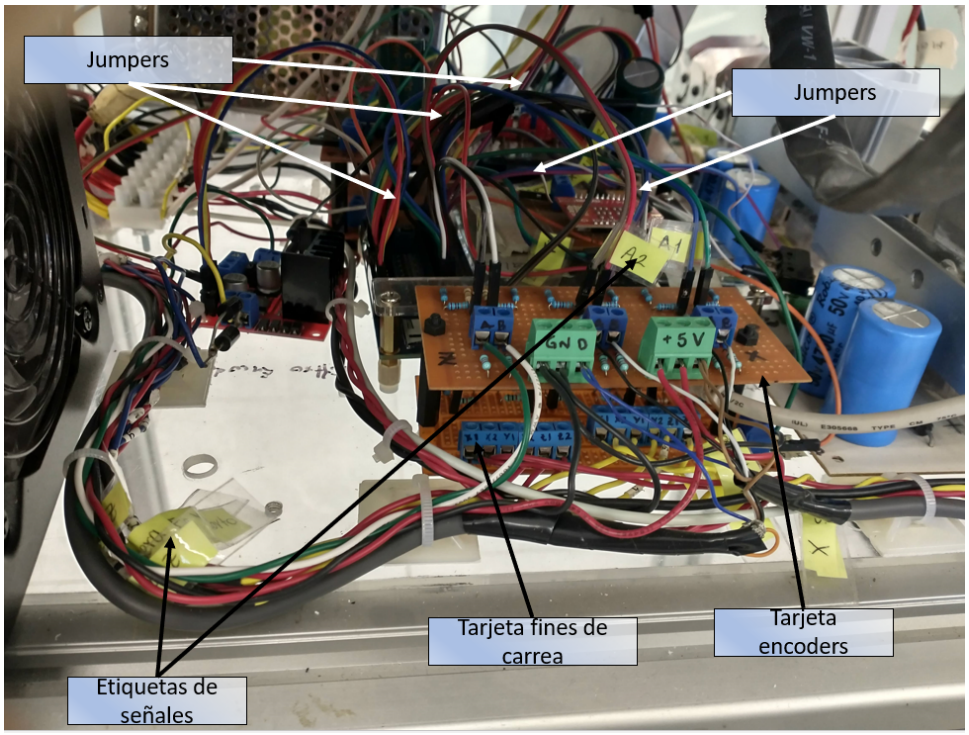


Figura 1.12: Cableado interno inicial (2).

Durante el análisis de la estructura y cableado inicial del robot, lo primero fue realizar la identificación de cada una de las tarjetas que componenen el sistema y el etiquetado de las conexiones.

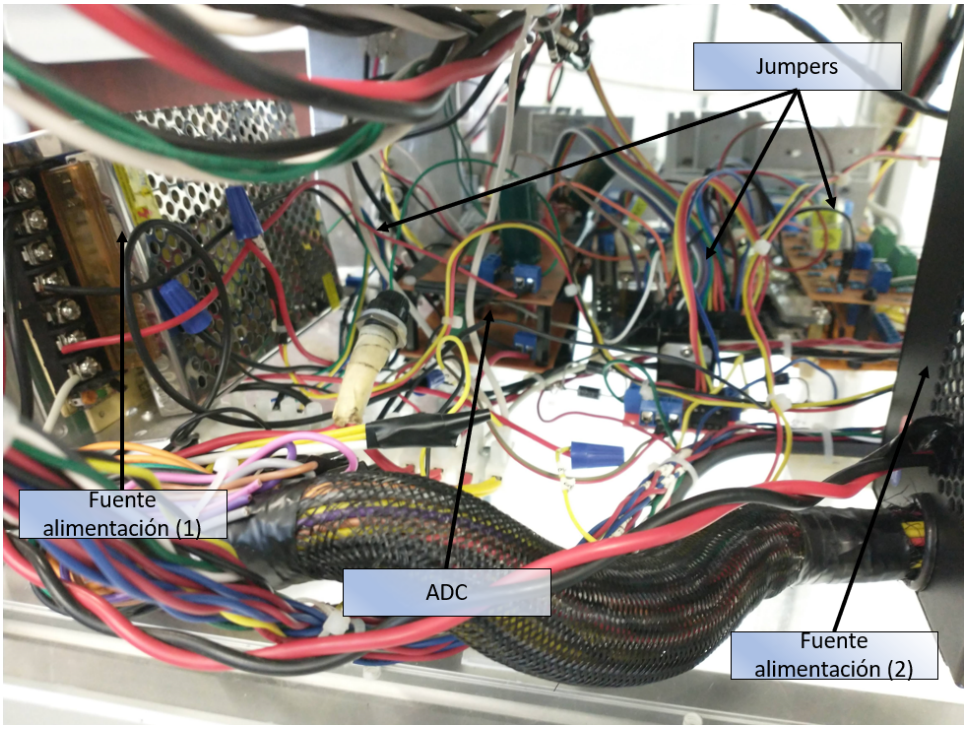


Figura 1.13: Cableado interno inicial (3).

El hecho de no contar con un diagrama de conexiones, aumenta el grado de dificultad para identificar las distintas conexiones que se encuentran en toda la parte electrónica del robot, respetando cada una de las señales involucradas en el funcionamiento de este. Lo primero, fue identificar la tarjeta que contiene un sistema de adquisición de datos basado en un FPGA, un sistema que utiliza sensores, transductores, amplificadores, convertidores tipo ADC y DAC, todo esto y más que permite procesar información de un sistema físico a una forma digitalizada. Para poder procesar toda esta información, son necesarias distintas etapas de acoplamiento de señales, como lo son:

- Amplificación
- Excitación
- Filtrado
- Multiplexado
- Aislamiento

En la sección 1.6 de Hardware, se identificaron ciertos bloques o circuitos que ayudan con el acoplamiento de las señales, y de igual manera, la falta de un diagrama inicial de conexiones implica que se tengan que identificar y señalar cada una de estas tarjetas y las conexiones que éstas implican, las tarjetas o bloques que se identificaron en este proceso, fueron:

- Tarjeta para la lectura de encoder.
- Tarjeta para la lectura de los fines de carrera.
- Tarjeta para la lectura de la temperatura ADC (cama caliente).
- Puente H para cada uno de los motores
- Etapa de potencia para el extrusor.
- Etapa de potencia para la cama caliente.
- Puente H para la alimentación del material.

Es por ello, que es de suma importancia identificar y etiquetar cada una de las señales que implica el sistema así como las tarjetas que este cuenta, por lo que se realiza la siguiente tabla con las señales involucradas en el robot.

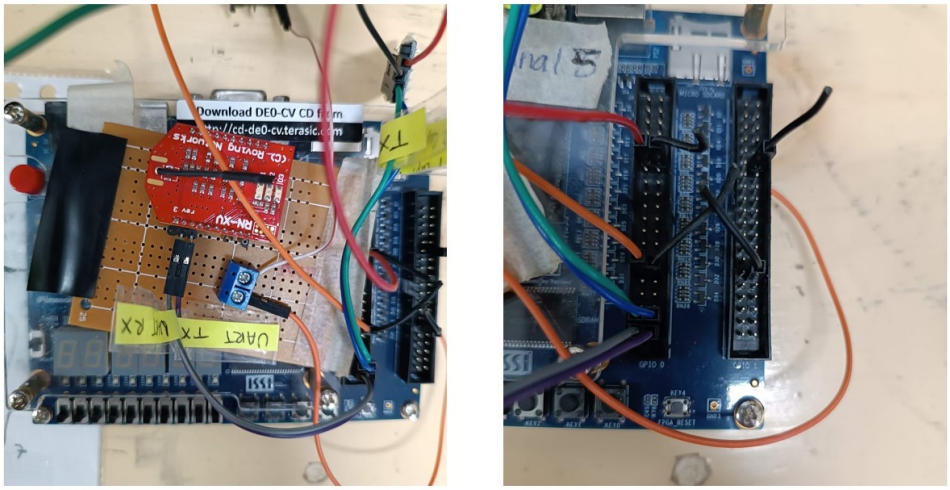


Figura 1.14: FPGA con módulo WiFi.

El conteo aproximado de cables se muestra en la siguiente tabla 1.4.

Elemento	Etiquetas	Número de cables
Fuente Conmutada 1	línea, Neutro, GND, 12 volts PH X	6
Fuente Conmutada 2	Línea, Neutro, GND, 12 volts PH Y	4
Fuente PC,	GND1,GND2, GND3, 12 volts PH Z, 12 volts regulador, -12 regulador, 5 volts Encoder, 5 volts FC,5 volts encoders, 5 volts PH (XYZ), 12 volts MP, 5 volts MP, 12 volts (2 ventiladores),	16
Fines de Carrera	X1,X2,Y1,Y2,Z1,Z2 (Rojos),X1,X2,Y1,Y2,Z1,Z2 (Amarillos),3.3 FPGA, GND.	14
Entrada FPGA Fines C.	FCX1,FCX2,FCY1,FCY2,FCZ1, FCZ2.	6
Encoder	Encoder Z(VCC, A, B, GND.), Encoder Y(VCC, A, B, GND.), Encoder X(VCC, A, B, GND.)	12
Entradas FPGA Encoder	EZA, EZB, EYA, EYB, EXA, EXB	6
ADCB (Extrusor)	5 volts, -5 volts, V1 (señal extrusor), V2 (señal extrusor), GND1,GND1. TIP 35 C (EBC).	9
Entradas FPGA ADCB (Extrusor)	B0,B1,B2,B3,B4,B5,B6,B7.	8
Reguladores LM2596	12 VOLTS, -12 VOLTS, GND1, Out 5 V, Out 1 -5 V.	6
ADCA (Cama caliente)	5 volts, V3 (señal Cama caliente), V4 (señal cama caliente), GND1, TIP 35 C (EBC).	7
Entradas FPGA ADCA (Cama caliente)	B0,B1,B2,B3,B4,B5,B6,B7.	8
Puente H eje X	5 volts, 12 volts, GND2, PWM3I, PWM3D, MX1, MX2	7
Puente H eje Y	5 volts, 12 volts, GND2, PWM4I, PWM4D, MY1, MY2	7
Puente H eje Z	5 volts, 12 volts, GND2, PWM5I, PWM5D, MZ1, MZ2	7
Salidas FPGA PWM motores	PWM3I, PWM3D, PWM4I, PWM4D, PWM5I, PWM5D	6
Alimentador Material (Motor pasos)	12 volts, GND3, 5 volts, MPA, MPB, MPC, MPD, B1A, B1C, B2B, B2D.	11
Salidas FPGA (Motor pasos)	MPA, MPB, MPC, MPD.	4
TIP 35 C Extrusor	Emisor, Base, Colector	3
TIP 35 C Cama caliente	Emisor, Base, Colector	3
Salidas FPGA PWM extrusor-cama caliente	PWME, PWMCC.	2
Modulo Wifly	RX,TX,VCC 5 VOLTS,GND.	4
	Total	**156**

Tabla 1.4: Número de conexiones para el robot cartesiano.

Es necesario mapear en el FPGA todos los puertos de entrada y salida para manipular el escáner con el robot cartesiano. La tarjeta de desarrollo que se va a utilizar es un FPGA modelo Cyclone V 5CEBA4F23C7N. El dispositivo completo cuenta con las siguientes características principales:

- FPGA Cyclone V 5CEBA4F23C7N
- 49,000 elementos lógicos programables
- 3080 K Bits de memoria embebida
- Memoria SD RAM de 64 MB x 16 bits de bus de datos
- 4 PLL fraccionales
- Memoria EPCS64
- 2x20 pines de propósito general (72 pines I/O, 2 pines de alimentación de 3.3 VCD y 2 de 5 VCD con sus respectivas tierras)
- 6 displays de 7 segmentos
- 10 LEDS
- 10 interruptores
- 4 pulsadores de propósito general
- 1 pulsador para el reset de la CPU

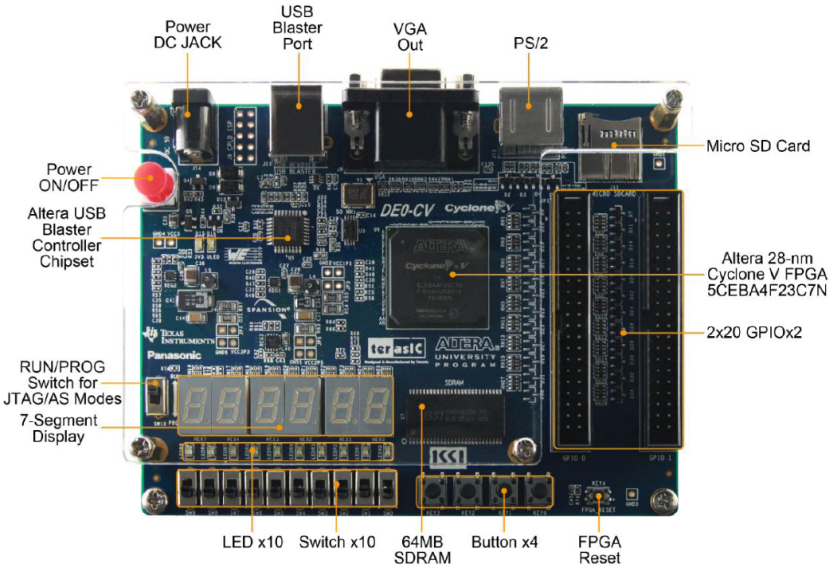


Figura 1.15: Tarjeta de desarrollo DE0-CV Cyclone V.

Como podemos apreciar en la Figura 1.15 tenemos 2 módulos de 20 pines, cada uno puede ser configurado como entrada o salida, es necesario mapear cada una de las señales para poder controlar el robot.

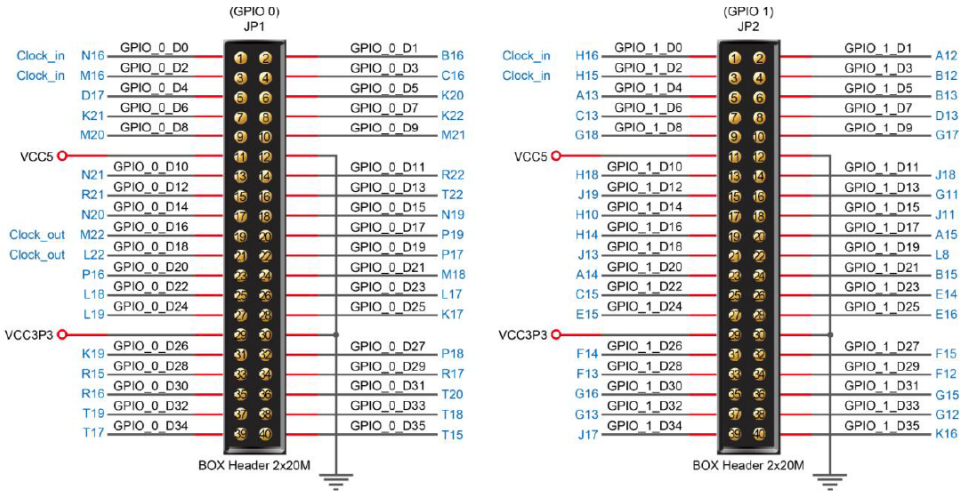


Figura 1.16: Pines del FPGA IN/OUT.

Fines de carrera

Dentro de los componentes electrónicos, se encuentra el final de carrera o sensor de contacto (también conocido como "interruptor de límite"), son dispositivos eléctricos, neumáticos o mecánicos situados al final del recorrido o de un elemento móvil, como por ejemplo una cinta transportadora, con el objetivo de enviar señales que puedan modificar el estado de un circuito. Internamente pueden contener interruptores normalmente abiertos (NA), cerrados (NC) o conmutadores dependiendo de la operación que cumplan al ser accionados, de ahí la gran variedad de finales de carrera que existen en mercado.

Ventajas e inconvenientes

Entre las ventajas encontramos la facilidad en la instalación, la robustez del sistema, es insensible a estados transitorios, trabaja a tensiones altas, debido a la inexistencia de imanes es inmune a la electricidad estática.
Los inconvenientes de este dispositivo son la velocidad de detección y la posibilidad de rebotes en el contacto, además depende de la fuerza de actuación. Los sensores actuales, son de tipo electromecánico el cual se encarga de detectar la posición de un elemento móvil mediante accionamiento mecánico. Además de ser los sensores más instalados alrededor del mundo, no dejan de ser sensores que necesitan estar en contacto físico con el objeto para detectar la llegada de un elemento móvil.

Como se menciona, la implementación de este tipo de sensores, implica inconvenientes como lo son la velocidad de detección puesto que el accionamiento mecánico conlleva un tiempo de retardo, además de la posibilidad de rebotes en el contacto puesto que depende de la fuerza de actuación.

Figura 1.17: Fin de carrera electromecánico.

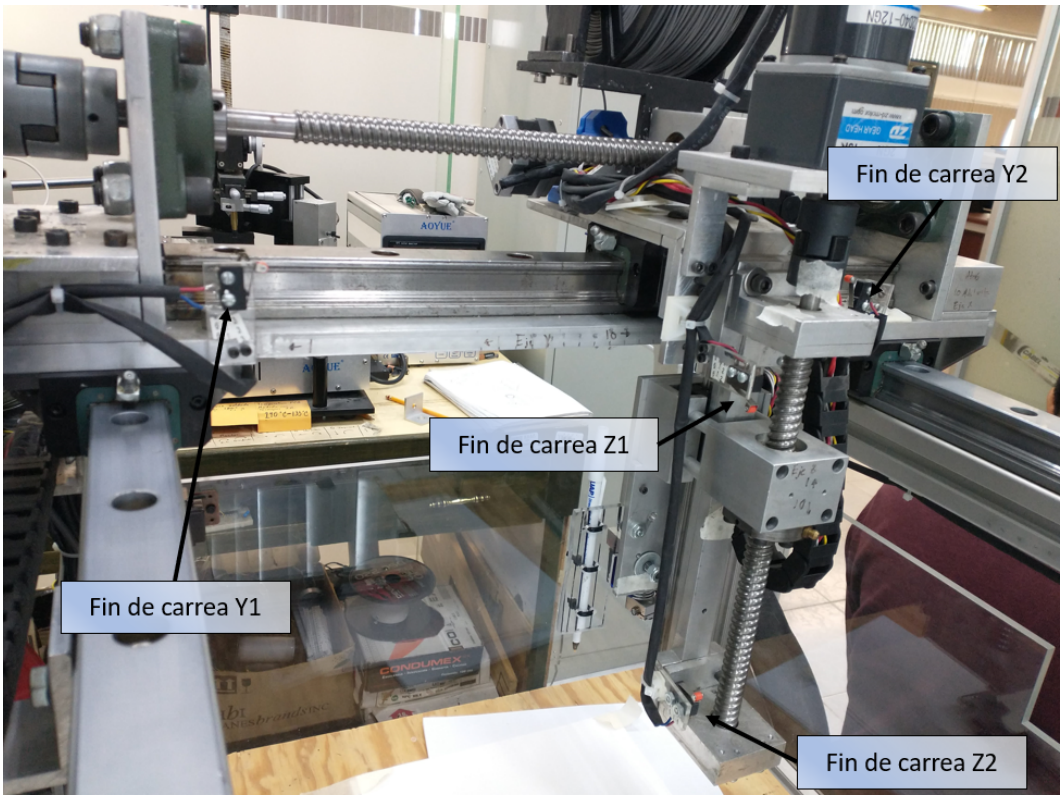


Figura 1.18: Fines de carrera Y y Z.

Para leer las señales provenientes de los encoder se utiliza una tarjeta la cual acopla la señal al FPGA. Los sensores fin de carrera identificados con su respectiva tarjeta, permitía el acoplamiento de las señales originadas por los sensores y que era comunicada a la tarjeta FPGA, pero dichos sensores utilizan voltajes de salida de 5 V y el FPGA admite un voltaje de entrada en sus terminales de 3.3 V, es por esto que la tarjeta identificada se encargaba del acoplamiento de dicha señal proveniente del sensor a la tarjeta.

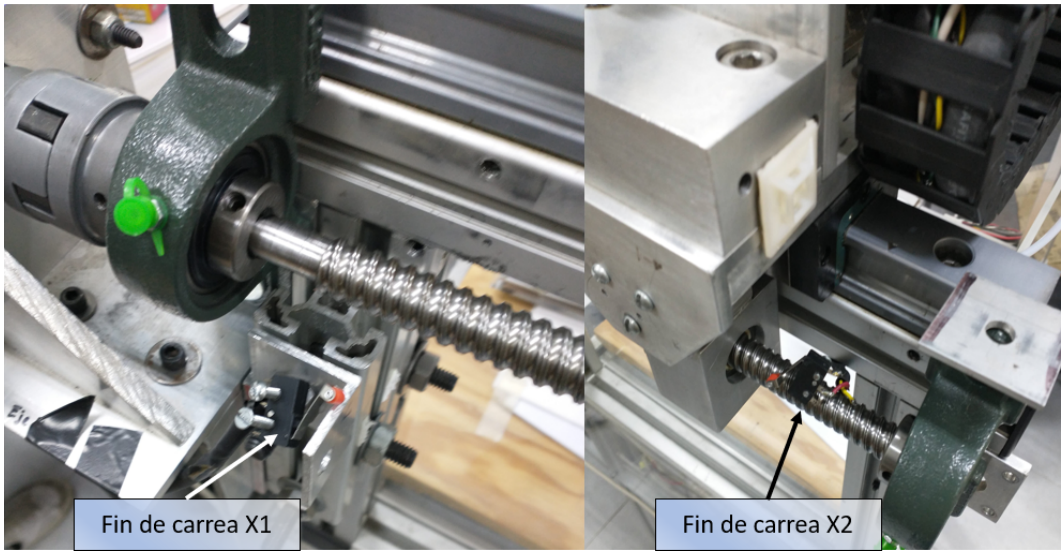


Figura 1.19: Fines de carrera X

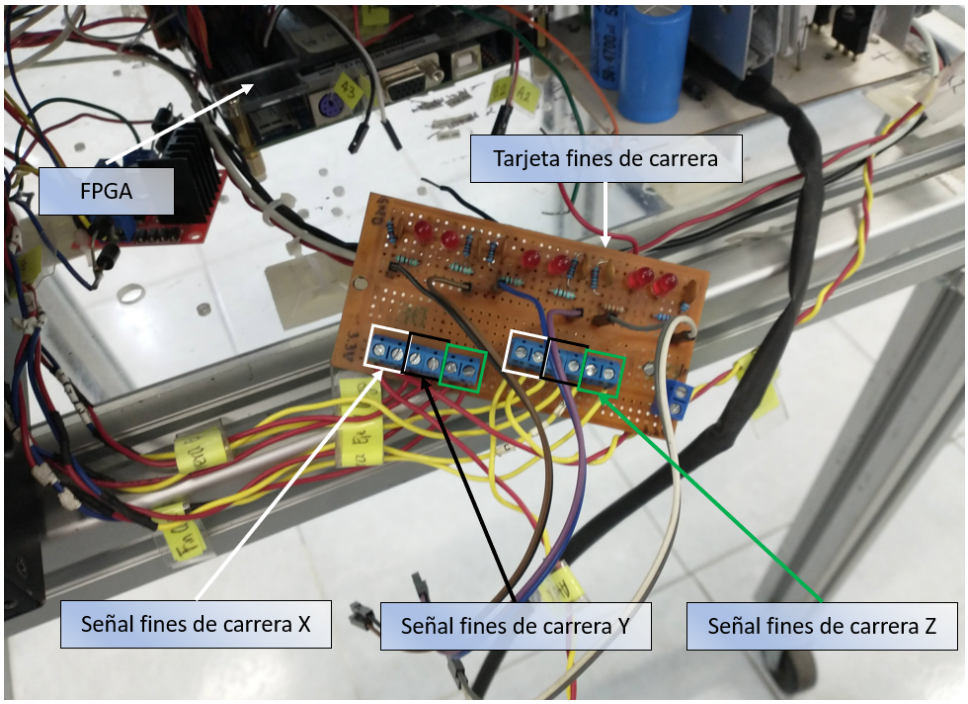


Figura 1.20: Fin de carrera tarjeta.

Encoders

El encoder utilizado perteneciente al modelo E6B2-CWZ6C, contiene 1000 pulsos por revolución, con un rango de alimentación de 5-24 V con un consumo de corriente que puede alcanzar los 80 mA. Este tipo de encoders de cuadratura produce 2 señales de salidas, las cuáles son denominadas A y B, gracias a estas señales es posible identificar la posición y velocidad, a demás con una adecuada interpretación de esta señal, se hace posible verificar el sentido de giro del motor. La tarjeta involucrada

Entrada	PIN FPGA
F. Carrera Z1	N20
F. Carrera Z2	N19
F. Carrera X1	N21
F. Carrera X2	T22
F. Carrera Y1	R21
F. Carrera Y2	N19

Tabla 1.5: Pines de conexión fines de carrera.

con el acoplamiento de estas señales permite la compatibilidad entre las magnitudes de las señales de salida del encoder con los pines de entrada para las señales del FPGA.

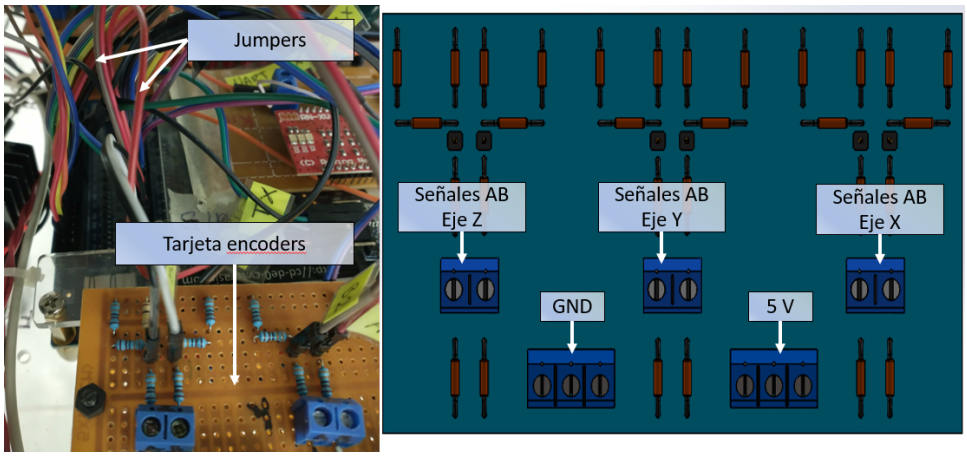


Figura 1.21: Tarjeta para encoders.

Encoder	PIN FPGA
Encoder - X (A)	M20
Encoder - X (B)	M21
Encoder - Y (A)	K21
Encoder - Y (B)	K22
Encoder - Z (A)	D17
Encoder - Z (B)	K20

Convertidor ADC

Este robot había sido implementado como una impresora 3D así que dentro de toda la electrónica se encuentran unas tarjetas la cuales se encargan de hacer la conversión analógica digital, proveniente de un extrusor y la cama caliente, estos convertidores tienen una resolución de 8 bits. Es decir, se tiene un total de 16 entradas que se conectan al FPGA. En la Figura 8 se aprecia que se tienen 2 tarjetas una sobre otra, la que se encuentra en la parte superior se encarga de controlar al extrusor.

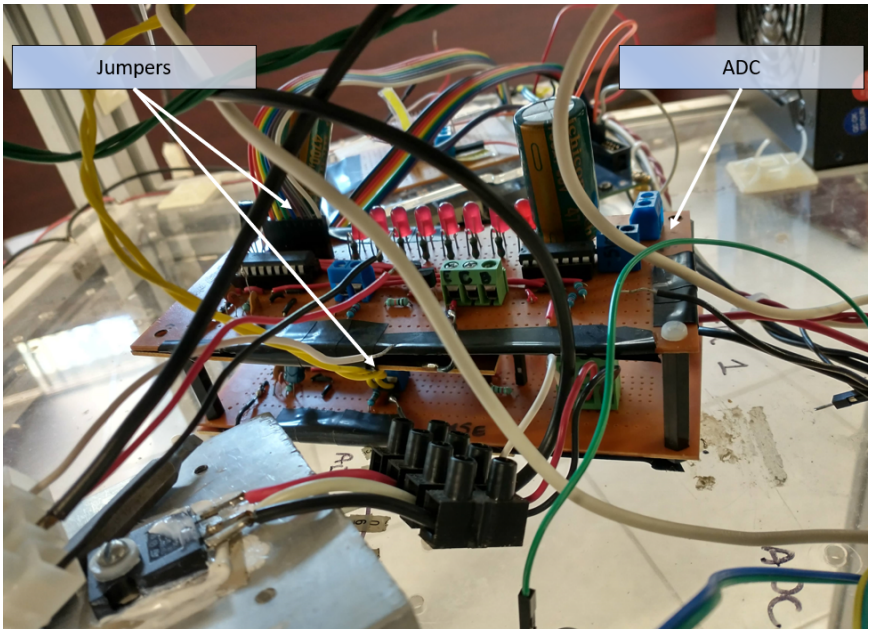


Figura 1.22: Tarjeta ADC.

Con la siguiente distribución de entradas:

Entrada	PIN FPGA
BO	J18
B1	G11
B2	J11
B3	A15
B4	L8
B5	B15
B6	E14
B7	E16

La otra tarjeta que se muestra por debajo en la figura 15, procesa las señales originadas de la cama caliente, con la misma configuración de BITS y entradas.

Puente H

Para poder activar los moto reductores se identificó la implementación de Puentes H en configuración Darlington, debiod a que cada motor en base a su hoja de datos

técnicos, consuma hasta 15 Amperes, los transistores implementados en esta tarjeta son los TIP35C (NPN) y TIP36C (PNP), donde cada uno de estos elementos tiene la capacidad de drenar una corriente hasta de 25 Amperes, se logró identificar el siguiente diagrama esquemático el cual describe la conexión que existe en cada una de estas tarjetas.

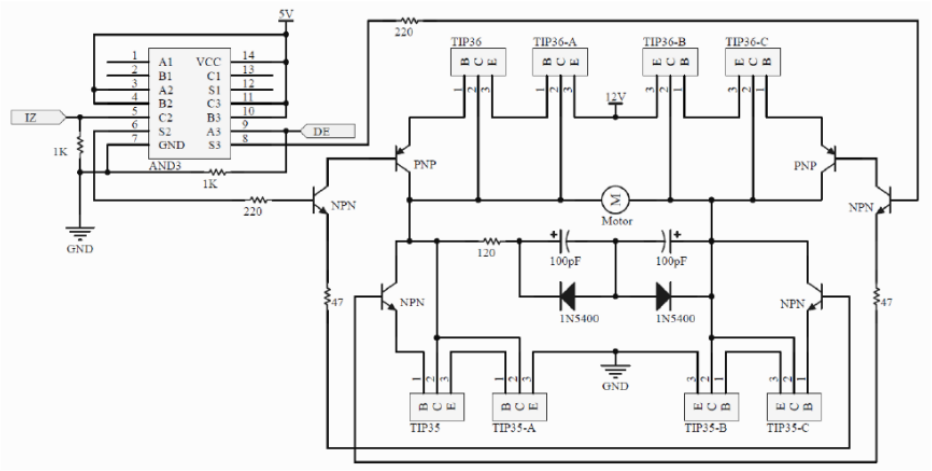


Figura 1.23: Diagrama - Puente H.

Estos Puentes H fueron diseñados e implementados en un PCB, presentan varios inconvenientes, al estar consumiendo una cantidad alta de corriente estos tienden a calentarse, sin embargo, el diseño de los disipadores de energía se queda corto en rendimiento, puesto que se presenta el problema de calentamiento lo que afecta el funcionamiento del robot en un lapso de trabajo de 4 horas, pues hasta ahora se han utilizado láminas de aluminio recicladas y pedazos de riel mal seleccionados, como principales disipadores de energía.

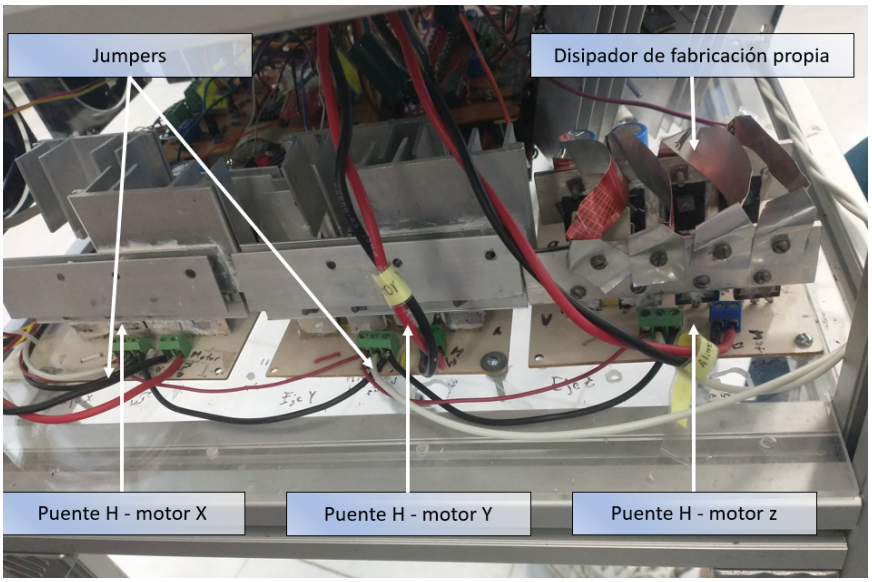


Figura 1.24: Cableado interno inicial.

Analizando las 3 tarjetas correspondiente a cada puente H de los motores, se hizo la siguiente identificación de señales propios a cada tarjeta.

Salida	Número entrada FPGA
X PWM3 Izquierda	P16
X PWM3 Derecha	M18
Y PWM4 Izquierda	L18
Y PWM4 Derecha	L17
Y PWM5 Izquierda	L10
Y PWM5 Derecha	K17

Tabla 1.6: Salidas para los Puentes H

Cama caliente y extrusor

Se tienen dos señales de PWM, uno de estos es para controlar la temperatura del extrusor y otro para controlar la temperatura de la cama caliente, este se conecta a 2 puentes H en configuración Darlington.

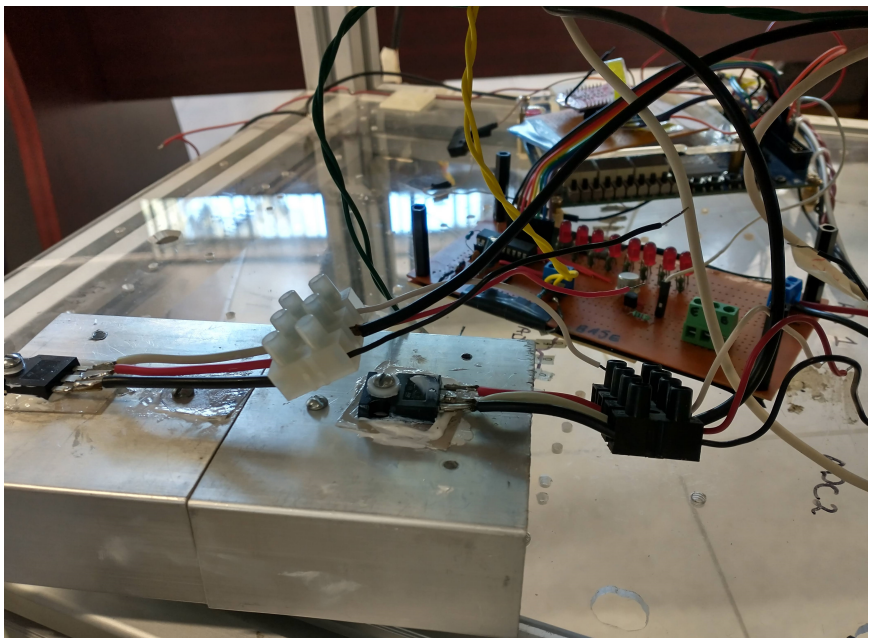


Figura 1.25: Tarjeta Extrusor - Cama caliente.

Salida	PIN FPGA
PWM Extrusor	F12
PWM cama caliente	F13

Cableado externo

El cableado del robot involucran distintos tipo de señales, como lo son los encoders, fines de carrera, alimentación de los motores, etapas de acoplamiento, etc. Esto presenta un problema, ya que existe una diversidad de señales y el tipo de estas, puesto que los motores consumen una corriente de hasta 15 A, esto implica que dichos cables generen un campo magnético de cierta magnitud y por Ley de inducción de Faraday induce a los cables por los que sólo conduce muy poca corriente, pequeña señal. En la siguiente figura podemos observar como todos los cables se encuentra entrelazados sin importar el tipo de estos. Esto provoca un ruido en las señales, lo que hace que el robot pierda en un determinando momento el valor de lectura de la posición de los datos del encoder.

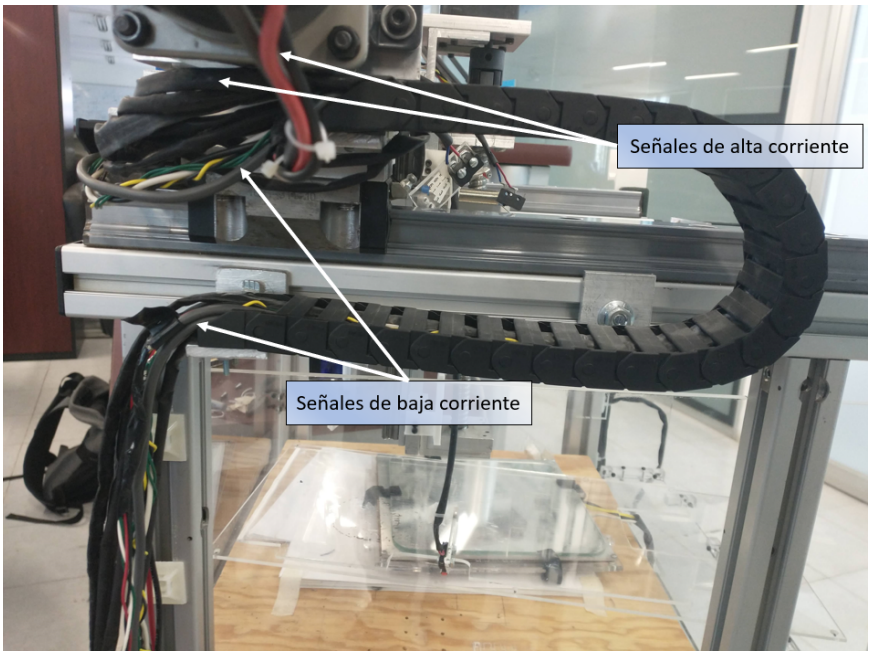


Figura 1.26: Cableado externo.

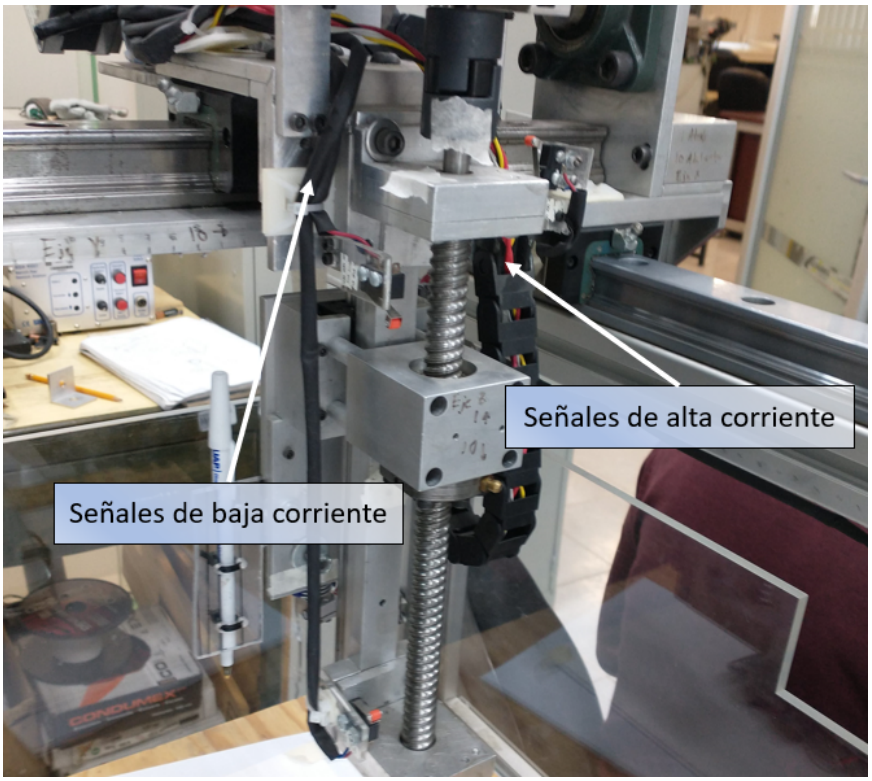


Figura 1.27: Cableado externo.

1.6. Actualización de la electrónica

Es claro que de primera instancia, el correcto funcionamiento del robot se ve afectado por su diseño electrónico y construcción mecánica, es por ello que, identificados los principales detalles con posibilidad de mejorar, fueron abordados, analizados y solucionados mediante una actualización en electrónica y hardware.

Modificación en la etapa de potencia

Durante el análisis de la tarjeta de los puentes H encargados de activar los motores, se probaron de manera individual cada una de estas tarjetas, comprobando los voltajes en las terminales, verificando el correcto funcionamiento de estas, se identificó que una de estas tarjetas presentaba una falla en su funcionamiento, puesto que al pasar minutos, los transistores comenzaban a sobre calentarse y afectaba el comportamiento de todo el circuito, se detectó un mal diseño en el circuito en cuanto a calidad se refiere.
Posterior al análisis de la tarjeta con el fallo, se hicieron las pruebas necesarias que permitieran identificar el error con el objetivo de arreglar este fallo, pero se encontró con un mal diseño y calidad en la tarjeta tanto en la implementación de los disipadores de calor como en el diseño de las pistas de conexión.

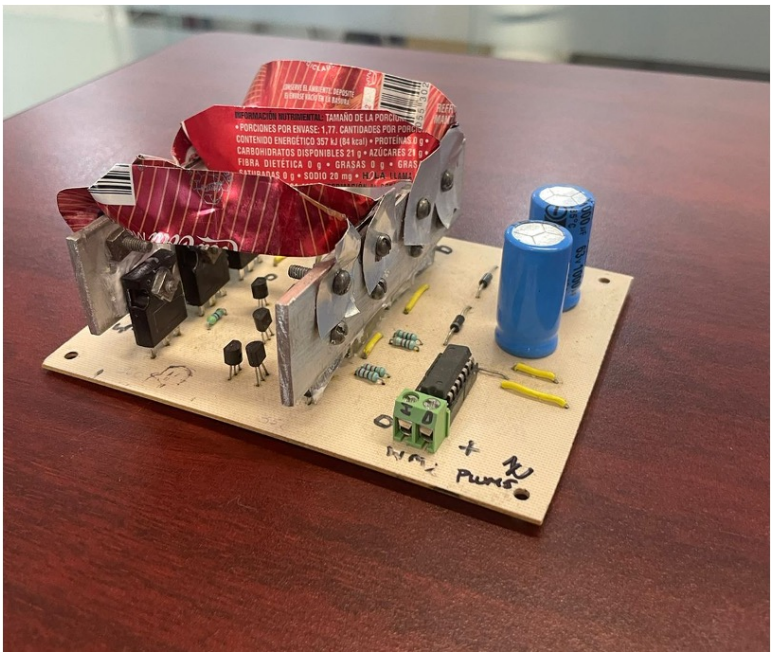


Figura 1.28: Puente H dañado.

De la Figura 1.28, se puede observar como los disipadores de calor utilizados eran láminas de aluminio reciclado, con un muy bajo rendimiento lo que producía sobre calentamiento después de un lapso de 3-4 horas de funcionamiento y producía errores en la implementación del robot, este problema propiciaba a que el valor de la posición del robot se perdiera durante el proceso y ejecución, de igual manera, un mal diseño e implementación en las pistas de conexión, era motivo para un fallo de esta magnitud.

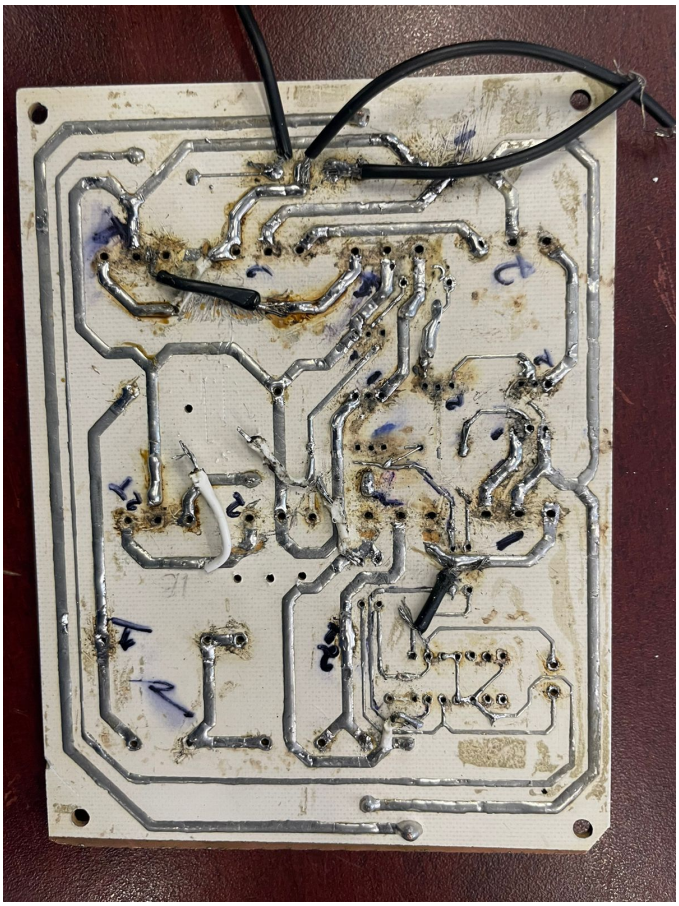


Figura 1.29: Pistas de conexión puente H dañado.

Por este motivo, se dio la iniciativa de diseñar una nueva tarjeta para replicar la tarjeta dañada. Se inició con el diseño de las pistas en NI Ultiboard siguiendo las conexiones de la figura 47, para su futura impresión y poder ser implementado en una nueva tarjeta PCB.

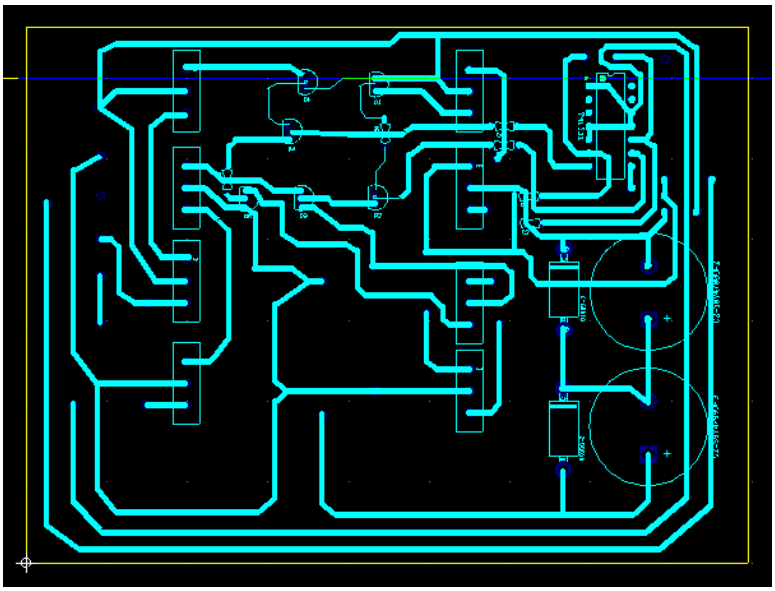


Figura 1.30: Diseño del nuevo puente H.

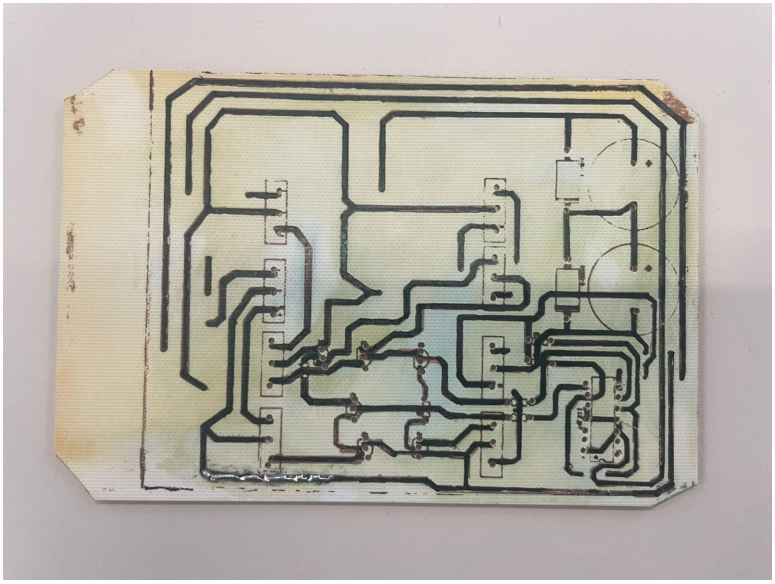


Figura 1.31: Puente H implementado.

Cambio de fines de carrera

Los finales de carrera unitarios inductivos son dispositivos activados sin contacto. Se utilizan como alternativa a los interruptores mecánicos. Las ventajas que ofrecen son que trabajan sin desgaste, son resistentes a los ambientes agresivos y ofrecen una vida de servicio prácticamente ilimitada.

Bajo esta premisa se optó por cambiar los sensores fin de carrera, para lo cual se optó por utilizar sensores fines de carrera de tipo inductivo, ya que estos no presentan los inconvenientes planteados para el primer tipo implementado, destacando que no es necesario el contacto físico para poder realizar la detección.

Figura 1.32: Fin de carrera inductivo.

El mínimo problema encontrado con la implementación de este nuevo sensor, es que la señal de salida presenta una magnitud de 5 V de los cuales nosotros necesitamos como máximo una respuesta de 3.3 V puesto que, este valor pertenece al rango de voltaje permitido para la lectura en la tarjeta FPGA Cyclone V 5CEBA4F23C7N implementada, para solucionar este obstáculo bastó con adecuar un circuito el cual permita la atenuación de la señal de salida del sensor para que corresponda a la señal de entrada de la tarjeta.

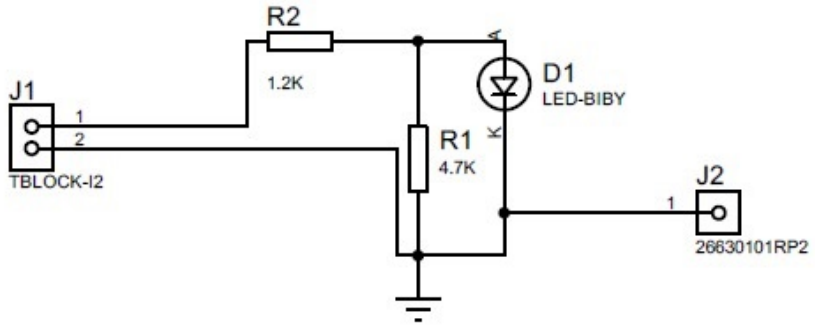


Figura 1.33: Circuito: Voltaje Fin de carrera - FPGA.

Instalar estos nuevos sensores en la estructura mecánica ya establecida del robot, fue un reto de primera instancia, pues como parte de la construcción no se había

considerado la opción de esta mejora y eso dificulta la instalación de los nuevos sensores, es por ello que, analizando la estructura e identificando posibles opciones de instalación, se diseñaron los siguientes soportes para así poder acomodar de manera adecuada estos nuevos dispositivos procurando el óptimo funcionamiento de estos.

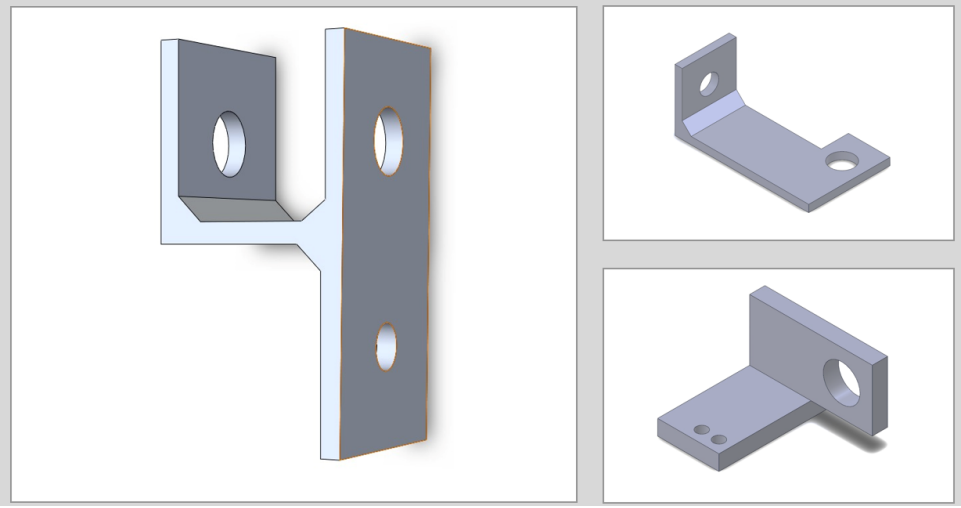


Figura 1.34: Soportes para fines de carrera inductivos.

Diseñados los soportes para los nuevos fines de carrera, se instalaron de manera que el funcionamiento de estos nuevos sensores sea lo más eficaz posible, fueron diseñados en SolidWorks e impresos con equipo perteneciente al programa de posgrados.

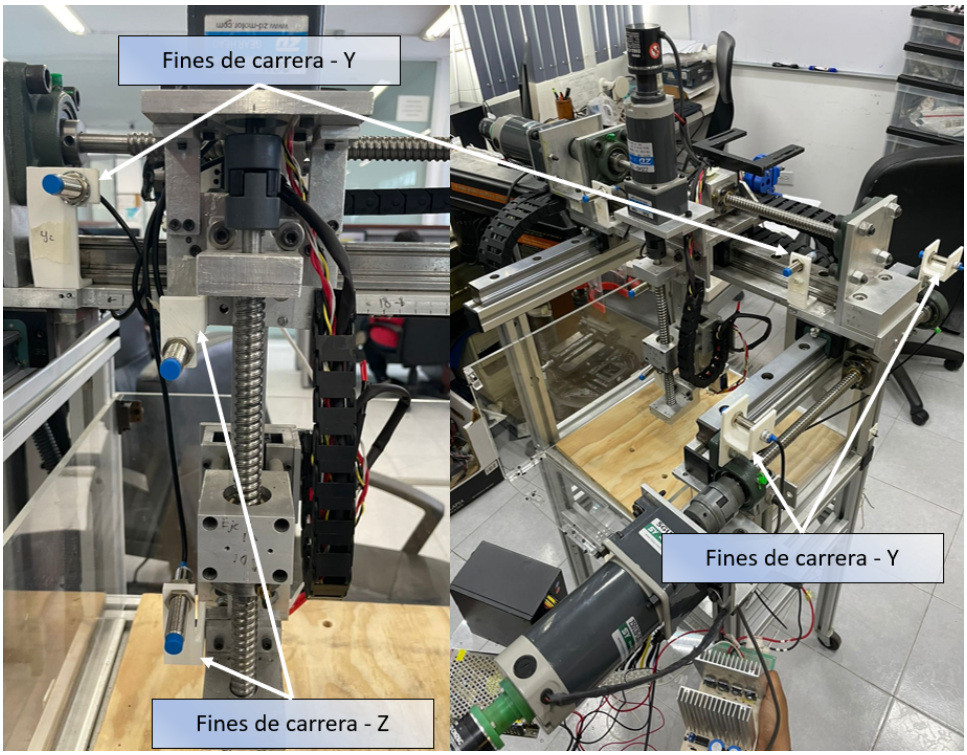


Figura 1.35: Nuevos fines de carrera instalados.

Aislamiento de señales

Dividir en dos vías diferentes las señales totales del sistema es de suma importancia, ya que existen elementos que funcionan con baja y alta corriente, el hecho de que estos cables estén en una misma vía puede interferir con las señales de baja corriente, puesto que los cables de alta corriente pueden interferir en el campo magnético de las otras líneas, y producir ruido en la lectura de las señales, es por eso que se separaron de manera adecuada estás líneas.

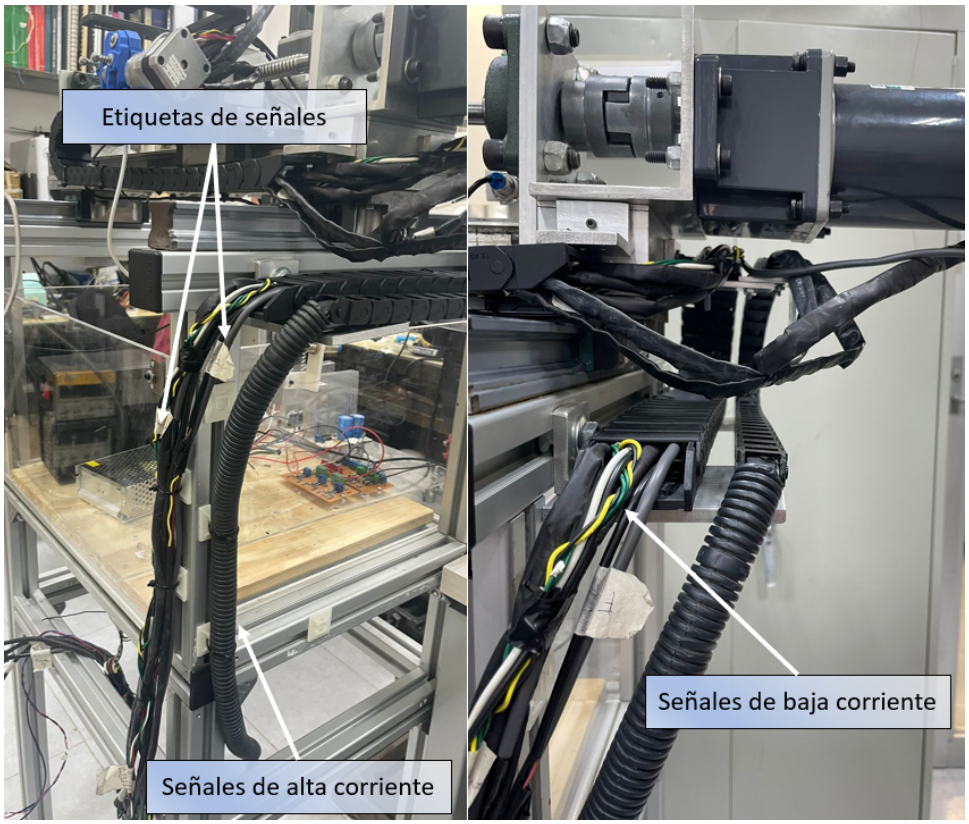


Figura 1.36: Aislamiento de señales externas.

Armado de electrónica y hardware

Teniendo todas las señales identificadas y los componentes electrónicos revisados y probados, se procedió a crear un objeto 3D para acomodar de manera apropiada para poder visualizar como sería la nueva organización de la electrónica.

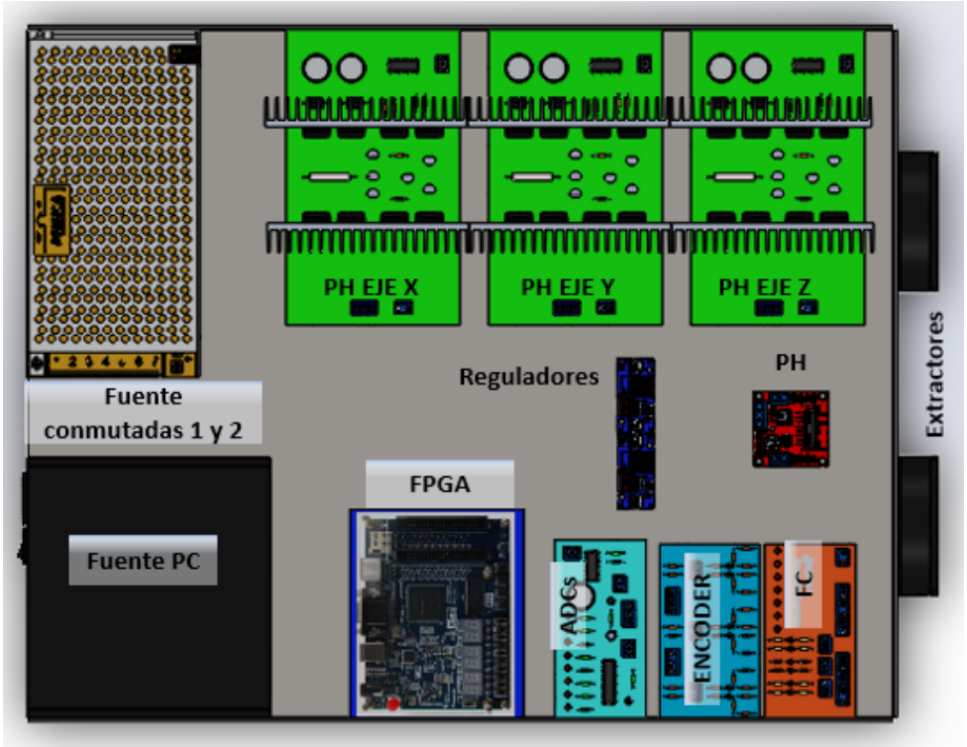


Figura 1.37: Organización de la electrónica.

1.7. Conclusiones

En este capítulo, se ha realizado una exhaustiva descripción de un robot cartesiano de tres grados de libertad, abarcando en detalle todas sus características mecánicas y funcionales. Aunque dicho robot fue construido previamente como parte de un proyecto en el sistema de posgrado de la facultad, se ha puesto de manifiesto que su funcionamiento se veía comprometido debido a limitaciones inherentes a su diseño y estructura.

El enfoque principal de este capítulo se centró en un estudio integral del sistema, tanto desde una perspectiva mecánica como electrónica. Se llevó a cabo una identificación minuciosa de cada componente del robot, analizando su contribución al funcionamiento global. Además, se sometieron todas las partes electrónicas a pruebas exhaustivas con el propósito de identificar posibles fallos y debilidades en el sistema.

Las actualizaciones realizadas a lo largo de este proceso se han dirigido específicamente a mejorar y optimizar el rendimiento del robot. Estas modificaciones tienen como objetivo final la creación de un sistema robusto y altamente eficiente que opere correctamente. Este enfoque integrado de análisis, diagnóstico y actualización es esencial para abordar las complicaciones previas y lograr un funcionamiento óptimo del robot cartesiano de tres grados de libertad, sentando las bases para futuras investigaciones y aplicaciones en este campo.

Capítulo 2

Modelo dinámico

Es importante determinar el modelo dinámico para realizar un control más apropiado del robot y que además presente un mejor desempeño, de igual manera, es muy útil para analizar su comportamiento con diferentes algoritmos de control.

2.1. Obtención del modelo dinámico

Ecuación de Euler-Lagrange

Las ecuaciones de Euler-Lagrange pueden ser obtenidas por el mismo lagrangiano del sistema:

$$\mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) = \mathcal{K}(\mathbf{q}, \dot{\mathbf{q}}) - \mathcal{U}(\mathbf{q}). \tag{2.1}$$

El lagrangiano de un sistema se considera como la diferencia entre la energía cinética $\mathcal{K}(\mathbf{q}, \dot{\mathbf{q}})$ menos la energía Potencial $\mathcal{U}(\mathbf{q})$, de la ecuación (2.1), se define la ecuación de movimiento de Euler-Lagrange.

$$\tau = \frac{d}{dt}\left[\frac{\partial \mathcal{L}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}}\right] - \frac{\partial \mathcal{L}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}}. \tag{2.2}$$

Aplicando la ecuación de Lagrange ya conocida, se obtiene la siguiente definición.

$$\frac{d}{dt}\left[\frac{\partial \mathcal{L}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}}\right] - \frac{\partial \mathcal{L}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}} + \frac{\partial \mathcal{D}t(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} = \frac{\partial \delta \mathcal{W}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} - F_s. \tag{2.3}$$

De la ecuación (2.3), $\mathcal{L}$ representa el Lagrangiano del sistema, $\mathcal{D}t$ es la energía disipativa del sistema, $\delta\mathcal{W}$ es el trabajo virtual, que son las fuerzas generalizadas del sistema. Empleando el método de parámetros agrupados, nos permite considerar todas las energías involucradas en el sistema, de esta manera, haciendo uso de las leyes de física, se puede obtener una representación del comportamiento de la energía de un sistema.

Esfuerzos Generalizados

Como podemos ver en la ecuación (2.4), todas las energías están igualadas a $\delta\mathcal{W}$, esto nos indica los trabajos realizados o fuerzas generalizadas del sistema. El trabajo es la fuerza que se aplica sobre un cuerpo para desplazarlo de un punto a otro, es decir, se desplace en dirección de la fuerza.

Normalmente, las fuentes que generan un tipo de esfuerzo se dan en términos de configuración de desplazamiento, no los desplazamientos generalizados. La transformación de coordenadas que relaciona la configuración de desplazamiento con el desplazamiento generalizado se utiliza en la expresión del trabajo virtual, para determinar los esfuerzos generalizados, el trabajo [35] presenta la siguiente tabla.

	Esfuerzo e	Flujo f	Desplazamiento q	Momento p
Traslación mecánica	Fuerza, F	Velocidad, v	Posición, x	Momento lineal, p
Rotación mecánica	torque, τ	velocidad angular, ω	ángulo, θ	momento angular, H
Eléctrico	voltaje, v	corriente, i	carga, q	enlace de flujo, λ
Fluidos	presión, P	tasa de flujo volumétrico, Q	volumen, V	momento de presión, Γ
Térmico	temperatura, T	tasa de flujo de entropía, $\dot{S}$	entropía, S	-

Tabla 2.1: Variables Fundamentales.

El trabajo virtual viene dado por la siguiente expresión:

$$\delta\mathcal{W} = \sum_{i=1}^{n} e_i \delta q_i, \tag{2.4}$$

Donde $\delta\mathcal{W}$ es el trabajo virtual, δq_i son los desplazamientos virtuales, e_i son los esfuerzos aplicados, a continuación se presentan los distintos trabajos virtuales realizados por distintos esfuerzos.

Disciplina		Trabajo Virtual
	$e\delta q$	$f\delta p$
Traslación mecánica	$F\delta x$	$v\delta p$
Rotación mecánica	$\tau\delta\theta$	$\omega\delta H$
Eléctrica	$v\delta q$	$i\delta\lambda$
Flujo	$P\delta V$	$W\delta\Gamma$
Térmico	$T\delta S$	Nulo

Tabla 2.2: Trabajos Virtuales.

2.2. Método de Parámetros Agrupados

El método de parámetros agrupados, permite y simplifica el análisis de un sistema, mediante la descripción de elementos que aproximan el comportamiento de los componentes distribuidos sobre el sistema. Aplicando las ecuaciones descritas y analizando la distribución de elementos a lo largo de un eje, se puede iniciar el análisis dinámico del sistema.

Figura 2.1: Estructura mecánica de un eje.

Empleando el método de parámetros agrupados, el cual considera la aportación de energía de todos los elementos conectados, los cuales se observan en la Figura 2.1. Iniciamos con la inclusión del motor, seguido de la caja de engranajes, el cople es el encargado de unir dos ejes, el de salida de la caja de engranes y el tornillo embalado, este último esta conectado a una tuerca la cual lo acopla a la mesa de trabajo. Realizando el esquema completo para cada elemento tenemos lo siguiente:

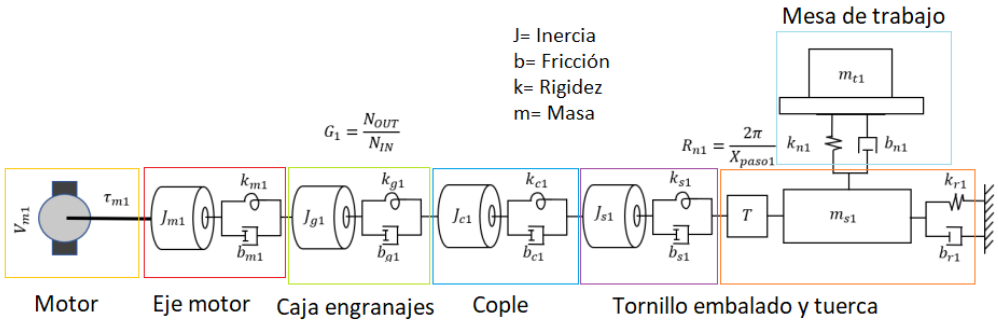


Figura 2.2: Estructura mecánica.

En la Figura 2.2 se observa la representación de cada elemento que afecta el comportamiento del robot, cada término descrito en este diagrama puede encontrarse descrito detalladamente en la tabla 2.3 y 2.4, para el caso en cuestión se trata de la representación esquemática del eje X, pero al tener una estructura similar en los 3 ejes, es posible generalizar el análisis dinámico.

Símbolo	Significado y Unidades
K_{m1}	Rigidez del eje del motor [Nm/rad]
K_{g1}	Rigidez del mecanismo de la caja de engranes [Nm/rad]
K_{c1}	Rigidez del cople [Nm/rad]
K_{s1}	Rigidez torsional del tornillo embalado [N/m]
K_{r1}	Rigidez axial del tornillo embalado [Nm/rad]
K_{n1}	Rigidez de la unión entre la tuerca y la mesa de trabajo [Nm/-rad]
λ_1	Parámetro de adaptación de rigidez [m/m]
b_{m1}	Coeficiente de fricción del eje del motor [Nms/rad]
b_{g1}	Coeficiente de fricción de la caja de engranes [Nms/rad]
b_{c1}	Coeficiente de fricción del cople [Nms/rad]
b_{s1}	Coeficiente de fricción torsional del tornillo embalado [Nms/-rad]
b_{n1}	Coeficiente de fricción de la mesa de trabajo [Nm/s]
b_{r1}	Coeficiente de fricción axial del tornillo embalado [Nm/s]
J_{m1}	Inercia del motor [Kgm^2]
J_{g1}	Inercia de la caja de engranes [kgm^2]
J_{c1}	Inercia del comple [kgm^2]
J_{s1}	Inercia del tornillo embalado [kgm^2]
m_{s1}	Masa del tornillo embalado y la tuerca [m]
m_{t1}	Masa de la mesa de trabajo [m]

Tabla 2.3: Parámetros del sistema.

Símbolo	Definición
θ_{m1}	Posición angular del eje del motor
θ_{g1}	Posición angular del eje de la caja de engranes
θ_{c1}	Posición angular del cople
θ_{s1}	Posición angular del tornillo embalado
X_{s1}	Posición axial del tornillo embalado
X_{t1}	Posición axial de la mesa de trabajo
$\dot{\theta}_{m1}$	Velocidad angular del motor
$\dot{\theta}_{g1}$	Velocidad angular de la caja de engranes
$\dot{\theta}_{c1}$	Velocidad angular del cople
$\dot{\theta}_{s1}$	Velocidad angular del tornillo embalado
$\dot{X}_{s1}$	Velocidad Axial del Tornillo embalado
$\dot{X}_{t1}$	Velocidad Axial de la mesa de trabajo

Tabla 2.4: Variables del sistema.

El robot analizado en cuestión, es de tipo cartesiano, así que la cinemática para el eje X está dada por la siguiente expresión.

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} x_{b1} + x_{xt1} \\ 0 \\ 0 \end{bmatrix}. \tag{2.5}$$

2.3. Modelo dinámico de un grado de libertad

Siguiendo este planteamiento y tomando en cuenta, como se mencionó con anterioridad, a la metodología de parámetros agrupados, es posible proponer una descripción de las ecuaciones que describen a las energías de nuestro eje, especialmente la energía cinética K, la energía potencial , la energía de disipación D y los trabajos virtuales $\delta\mathcal{W}$. Se obtienen las siguientes ecuaciones:

$$\begin{aligned} K_1(\varphi_1, \dot{\varphi}_1) &= \frac{1}{2}J_{m1}\dot{\theta}_{m1}^2 + \frac{1}{2}J_{g1}\dot{\theta}_{g1}^2 + \frac{1}{2}J_{c1}\dot{\theta}_{c1}^2 + \frac{1}{2}J_{s1}\dot{\theta}_{s1}^2 + \frac{1}{2}m_{s1}(\dot{X}_{s1}^2) + \frac{1}{2}m_{t1}(\dot{X}_{t1}^2) \\ U_1(\varphi_1) &= \frac{1}{2}K_{m_1}(\theta_{m1}G_1\theta_{g1})^2 + \frac{1}{2}K_{g1}(\theta_{g1} - \theta_{c1})^2 + \frac{1}{2}K_{c1}(\theta_{c1} - \theta_{s1})^2 + \\ &\quad \frac{1}{2}K_{s1}(\theta_{s1} - R_{n1}X_{s1})^2 + \frac{1}{2}k_{r1}X_{s1}^2 + \frac{1}{2}k_{t1}(X_{s1} - X_{t1})^2 \\ D_1(\varphi_1, \dot{\varphi}_1) &= \frac{1}{2}b_{m_1}(\dot{\theta}_{m1} - G_1\dot{\theta}_{g1})^2 + \frac{1}{2}b_{g1}(\dot{\theta}_{g1} - \dot{\theta}_{c1})^2 + \frac{1}{2}b_{c1}(\dot{\theta}_{c1} - \dot{\theta}_{s1})^2 + \\ &\quad \frac{1}{2}b_{s1}(\dot{\theta}_{s1} - R_{n1}\dot{X}_{s1})^2 + \frac{1}{2}b_{r1}\dot{X}_{s1}^2 + \frac{1}{2}b_{t1}(\dot{X}_{s1} - \dot{X}_{t1})^2 \\ \delta W_1(\varphi_1) &= \tau_{m1}\delta\theta_{m1} \end{aligned} \tag{2.6}$$

donde

$$\begin{aligned} R_{n1} &= \frac{X_{paso1}}{2\pi\phi_1} \\ \varphi_1 &= [\theta_{m1}, \theta_{g1}, \theta_{c1}, \theta_{b1}, X_{s1}, X_{t1}]^T \\ \dot{\varphi}_1 &= [\dot{\theta}_{m1}, \dot{\theta}_{g1}, \dot{\theta}_{c1}, \dot{\theta}_{b1}, \dot{X}_{s1}, \dot{X}_{t1}]^T \end{aligned} \tag{2.7}$$

Con las ecuaciones de energía descritas en (2.6) y empleando una vez más la ecuación de Lagrange, se pueden obtener las siguientes ecuaciones que describen el comportamiento de cada uno de los elementos asociados a lo largo de cada grado de libertad para el robot cartesiano, dicho sistema de ecuaciones se describe en [34]:

Ecuaciones para el eje X

$$\tau_{m1} = J_{m1}\ddot{\theta}_{m1} + k_{m1}(\lambda_1)(\theta_{m1} - G_1\theta_{g1}) + b_{m1}(\dot{\theta}_{m1} - G_1\dot{\theta}_{g1}) \tag{2.8}$$

$$0 = J_{g1}\ddot{\theta}_{g1} + b_{m1}G_1(G_1\dot{\theta}_{g1} - \dot{\theta}_{m1}) + k_{m1}G_1(G_1\theta_{g1} - \theta_{m1}) + k_{g1}(\theta_{g1} - \theta_{c1}) + b_{g1}(\dot{\theta}_{g1} - \dot{\theta}_{c1}) \tag{2.9}$$

$$0 = J_{c1}\ddot{\theta}_{c1} + k_{g1}(\theta_{c1} - \theta_{g1}) + k_{c1}(\theta_{c1} - \theta_{s1}) + b_{g1}(\dot{\theta}_{c1} - \dot{\theta}_{g1}) + b_{c1}(\dot{\theta}_{c1} - \dot{\theta}_{s1}) \tag{2.10}$$

$$0 = J_{s1}\ddot{\theta}_{s1} + k_{c1}(\theta_{s1} - \theta_{c1}) + k_{s1}(\theta_{s1} - R_{n1}X_{s1}) + b_{c1}(\dot{\theta}_{s1} - \dot{\theta}_{c1}) + b_{s1}(\dot{\theta}_{s1} - R_{n1}\dot{X}_{s1}) \tag{2.11}$$

$$0 = (m_{s1} + m_{s2} + m_{s3})\ddot{X}_{s1} + k_{s1}R_{n1}(R_{n1}X_{s1} - \theta_{s1}) + k_{r1}X_{s1} + k_{t1}(X_{s1} - X_{t1}) + b_{s1}R_{n1}(R_{n1}\dot{X}_{s1} - \dot{\theta}_{s1}) + b_{r1}\dot{X}_{s1} + b_{t1}(\dot{X}_{s1} - \dot{X}_{t1}) \tag{2.12}$$

$$0 = (m_{t1} + m_{t2} + m_{t3})\ddot{X}_{t1} + K_{t1}(X_{t1} - X_{s1}) + b_{t1}(\dot{X}_{t1} - \dot{X}_{s1}) + F_{s1} \tag{2.13}$$

Estas seis ecuaciones obtenidas corresponden al modelo dinámico de un grado de libertad del robot que se estudia. Es por esto, que cada ecuación posee un significado determinado y corresponde a la descripción del comportamiento de una parte del robot.

Descripción de las ecuaciones

- La ecuación (2.8) es responsable de describir el comportamiento del eje del motor y es en esta ecuación donde se plantea la entrada de energía del sistema en la forma del torque proporcionado por el sistema embebido del FPGA con su correspondiente etapa de potencia.
- La ecuación (2.9) corresponde a la descripción del comportamiento del eje de la caja de engranes, por lo que está directamente relacionada con la constante G_1 que corresponde a la transformación de energía que se tiene por efecto del funcionamiento de la caja de engranes.
- La ecuación (2.10) describe el comportamiento del cople que une a la caja de engranes y al tornillo embalado. En este punto es importante marcar que el comportamiento planteado para este elemento se considera como lineal.
- La ecuación (2.11) corresponde al comportamiento del tornillo embalado, haciendo énfasis en el comportamiento de su componente rotacional, por lo que su comportamiento no es tan diferente del comportamiento de los anteriores elementos.
- La ecuación (2.12) se encarga de describir el comportamiento del tornillo embalado en su componente axial, por lo que, contrario a la anterior ecuación, se encuentra estrechamente ligada con la constante del sistema R_{n1}, la cual

corresponde a la transformación efectuada por el mecanismo rosca-tuerca del tornillo.

- La ecuación (2.13) se corresponde a la descripción del comportamiento de la mesa de trabajo, la cual se encuentra acoplada a la tuerca del tornillo embalado y cuya masa corresponde al peso del eje Y (en el caso del eje X).

2.4. Modelo dinámico de tres grados de libertad

Como se mencionó anteriormente, la descripción de las ecuaciones 2.8 a 2.13 corresponden al modelo dinámico para 1 grado de libertad, es importante destacar que para describir los grados restantes, es posible generalizar estas ecuaciones, por lo que se obtiene el siguiente diagrama esquemático y sistema de ecuaciones.

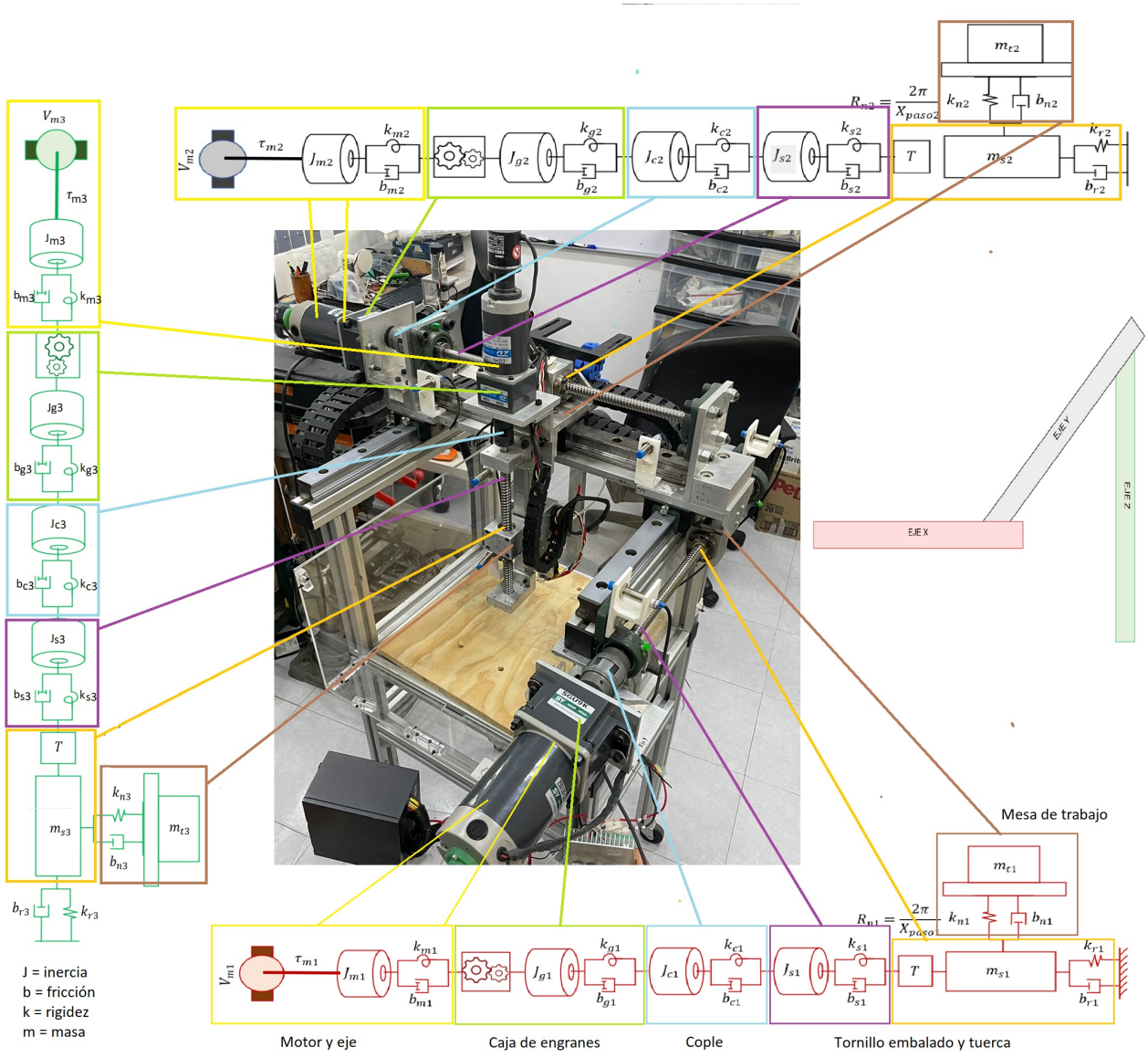


Figura 2.3: Estructura del robot cartesiano.

El robot como es de tipo cartesiano, así que la cinemática está dada por la siguiente expresión:

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} x_{s1} + x_{xt1} \\ 0 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} x_{s1} + x_{xt1} \\ x_{s2} + x_{xt2} \\ 0 \end{bmatrix} \qquad \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = \begin{bmatrix} x_{s1} + x_{xt1} \\ x_{s2} + x_{xt2} \\ x_{s3} + x_{xt3} \end{bmatrix} \tag{2.14}$$

De forma particular las ecuaciones de energía quedan de la siguiente manera:

$$\begin{aligned} K_i(\varphi_i, \dot{\varphi}_i) =& \frac{1}{2} J_{mi} \dot{\theta}_{mi}^2 + \frac{1}{2} J_{gi} \dot{\theta}_{gi}^2 + \frac{1}{2} J_{ci} \dot{\theta}_{ci}^2 + \frac{1}{2} J_{si} \dot{\theta}_{si}^2 + \frac{1}{2} m_{si} (\dot{X}_{si}^2) + \frac{1}{2} m_{ti} (\dot{X}_{ti}^2) \\ U_i(\varphi_i) =& \frac{1}{2} K_{m_i} (\theta_{mi} - G_i \theta_{gi})^i + \frac{1}{2} K_{gi} (\theta_{gi} - \theta_{ci})^2 + \frac{1}{2} K_{ci} (\theta_{ci} - \theta_{si})^2 + \frac{1}{2} K_{si} (\theta_{si} - R_{ni} X_{si})^2 + \\ & \frac{1}{2} k_{ri} X_{si}^2 + \frac{1}{2} k_{ti} (X_{si} - X_{ti})^2 \\ \mathcal{D}_i(\varphi_i, \dot{\varphi}_i) =& \frac{1}{2} b_{m_i} (\dot{\theta}_{mi} - G_i \dot{\theta}_{gi})^2 + \frac{1}{2} b_{gi} (\dot{\theta}_{gi} - \dot{\theta}_{ci})^2 + \frac{1}{2} b_{ci} (\dot{\theta}_{ci} - \dot{\theta}_{si})^2 + \frac{1}{2} b_{si} (\dot{\theta}_{si} - R_{ni} \dot{X}_{si})^2 + \\ & \frac{1}{2} b_{ri} \dot{X}_{si}^2 + \frac{1}{2} b_{ti} (\dot{X}_{si} - \dot{X}_{ti})^2 \\ \delta \mathcal{W}_i(\varphi_i) =& \tau_{mi} \delta \theta_{mi} \end{aligned} \tag{2.15}$$

Entonces las energías totales del sistema están dadas por las siguientes ecuaciones:

$$\mathcal{K}_T(\varphi_T, \dot{\varphi}_T) = \mathcal{K}_1(\varphi_1, \dot{\varphi}_1) + \mathcal{K}_2(\varphi_2, \dot{\varphi}_2) + \mathcal{K}_3(\varphi_3, \dot{\varphi}_3) \tag{2.16a}$$

$$\mathcal{U}_T(\varphi_T) = \mathcal{U}_1(\varphi_1) + \mathcal{U}_2(\varphi_2) + \mathcal{U}_3(\varphi_3) \tag{2.16b}$$

$$\mathcal{D}_T(\varphi_T, \dot{\varphi}_T) = \mathcal{D}_1(\varphi_1, \dot{\varphi}_1) + \mathcal{D}_2(\varphi_2, \dot{\varphi}_2) + \mathcal{D}_3(\varphi_3, \dot{\varphi}_3) \tag{2.16c}$$

$$\delta \mathcal{W}_T(\varphi_T) = \delta \mathcal{W}_1(\varphi_1) + \delta \mathcal{W}_2(\varphi_2) + \delta \mathcal{W}_3(\varphi_3) \tag{2.16d}$$

Con las ecuaciones generales de energía y aplicando la ecuación de Lagrange, se obtiene el sistema de ecuaciones que describe el comportamiento de cada uno de los componentes del robot para los tres grados de libertad.

$$\begin{aligned} \tau_{mi} =& J_{mi} \ddot{\theta}_{mi} + k_{mi}(\lambda_i)(\theta_{mi} - G_i \theta_{gi}) + b_{mi}(\dot{\theta}_{mi} - G_i \dot{\theta}_{gi}) \\ 0 =& J_{gi} \ddot{\theta}_{gi} + b_{mi} G_i (G_i \dot{\theta}_{gi} - \dot{\theta}_{mi}) + k_{mi} G_i (G_i \theta_{gi} - \theta_{mi}) + k_{gi}(\theta_{gi} - \theta_{ci}) + b_{gi}(\dot{\theta}_{gi} - \dot{\theta}_{ci}) \\ 0 =& J_{ci} \ddot{\theta}_{ci} + k_{gi}(\theta_{ci} - \theta_{gi}) + k_{ci}(\theta_{ci} - \theta_{si}) + b_{gi}(\dot{\theta}_{ci} - \dot{\theta}_{gi}) + b_{ci}(\dot{\theta}_{ci} - \dot{\theta}_{si}) \\ 0 =& J_{si} \ddot{\theta}_{si} + k_{ci}(\theta_{si} - \theta_{ci}) + k_{si}(\theta_{si} - R_{ni} X_{si}) + b_{ci}(\dot{\theta}_{si} - \dot{\theta}_{ci}) + b_{si}(\dot{\theta}_{si} - R_{ni} \dot{X}_{si}) \\ 0 =& (\sum_{k=1}^{3} m_{sk}) \ddot{X}_{si} + k_{si} R_{ni} (R_{ni} X_{si} - \theta_{si}) + k_{ri} X_{si} + k_{ti}(X_{si} - X_{ti}) \\ & + b_{si} R_{ni} (R_{ni} \dot{X}_{si} - \dot{\theta}_{si}) + b_{ri} \dot{X}_{si} + b_{ti}(\dot{X}_{si} - \dot{X}_{ti}) \\ 0 =& (\sum_{k=1}^{3} m_{tk}) \ddot{X}_{ti} + K_{ti}(X_{ti} - X_{si}) + b_{ti}(\dot{X}_{ti} - \dot{X}_{si}) + F_{si} \end{aligned}$$

para

$$i = 1, 2, 3$$

2.5. Representación de las variables de estado

Para hacer la representación se toman en consideración las ecuaciones (2.8) a (2.13), al tener los tres ejes la misma configuración mecánica, se puede generalizar el sistema en estados para un solo grado.

Elemento	Variable de estado
Motor	$x_1 = \theta_{m1}$ Posición Eje motor. $x_2 = \dot{\theta}_{m1}$ Velocidad Eje motor
Caja de Engranes	$x_3 = \theta_{g1}$ Posición Caja engranes. $x_4 = \dot{\theta}_{g1}$ Velocidad caja engranes
Cople	$x_5 = \theta_{c1}$ Posición Cople. $x_6 = \dot{\theta}_{c1}$ Velocidad Cople
Tornillo embalado	$x_7 = \theta_{s1}$ Posición Tornillo embalado. $x_8 = \dot{\theta}_{s1}$ Velocidad Tornillo embalado
Movimiento axial rosca Tuerca	$x_9 = X_{s1}$ Posición $x_{10} = \dot{X}_{s1}$ Velocidad Rosca Tuerca
Mesa de Trabajo	$x_{11} = X_{t1}$ Posición Eje motor. $x_{12} = \dot{X}_{t1}$ Velocidad Eje motor

Tabla 2.5: Variables de estado eje X.

$$
\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{U(t)}{J_{m1}} - \frac{K_{m1}}{J_{m1}}x_1 + \frac{K_{m1}}{J_{m1}}x_3 - \frac{b_{m1}}{J_{m1}}x_2 + \frac{b_{m1}G_1}{J_{m1}}x_4 \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= x_1\frac{K_{m1}G_1}{J_{g1}} + x_2\frac{b_{m1}G_1}{J_{g1}} - x_3\frac{(K_{m1}G_1^2 + K_{g1})}{J_{g1}} - x_4\frac{(b_{m1}G_1^2 + b_{g1})}{J_{g1}} \\
&\quad + x_5\frac{K_{g1}}{J_{g1}} + x_6\frac{b_{g1}}{J_{g1}} \\
\dot{x}_5 &= x_6 \\
\dot{x}_6 &= x_3\frac{K_{g1}}{J_{c1}} + x_4\frac{b_{g1}}{J_{c1}} - x_5\frac{(K_{g1} + K_{c1})}{J_{c1}} - x_6\frac{(b_{g1} + b_{c1})}{J_{c1}} + x_7\frac{K_{c1}}{J_{c1}} + x_8\frac{b_{c1}}{J_{c1}} \\
\dot{x}_7 &= x_8 \\
\dot{x}_8 &= x_5\frac{K_{c1}}{J_{s1}} + x_6\frac{b_{c1}}{J_{s1}} - x_7\frac{(K_{c1} + K_{s1})}{J_{s1}} - x_8\frac{(b_{c1} + b_{s1})}{J_{s1}} + x_9\frac{R_{n1}K_{s1}}{J_{s1}} + x_{10}\frac{R_{n1}b_{s1}}{J_{s1}} \\
\dot{x}_9 &= x_{10} \\
\dot{x}_{10} &= x_7\frac{R_{n1}K_{s1}}{m_{s1} + m_{s2} + m_{s3}} + x_8\frac{R_{n1}b_{s1}}{m_{s1} + m_{s2} + m_{s3}} - x_9\frac{(K_{s1}R_{n1}^2 + K_{r1} + K_{t1})}{m_{s1} + m_{s2} + m_{s3}} \\
&\quad - x_{10}\frac{(b_{s1}R_{n1}^2 + b_{r1} + b_{t1})}{m_{s1} + m_{s2} + m_{s3}} + x_{11}\frac{K_{t1}}{m_{s1} + m_{s2} + m_{s3}} + x_{12}\frac{b_{t1}}{m_{s1} + m_{s2} + m_{s3}} \\
\dot{x}_{11} &= x_{12} \\
\dot{x}_{12} &= x_9\frac{K_{t1}}{m_{t1}} - x_{11}\frac{K_{t1}}{m_{t1}} + x_{10}\frac{b_{t1}}{m_{t1}} - x_{12}\frac{b_{t1}}{m_{t1}}
\end{aligned}
\tag{2.17}
$$

Siguiendo la secuencia de asignación de las variables de estado para el eje X (Tabla 2.5) y las ecuaciones de esta asignación en 2.17, es posible hacer la misma asignación para los ejes restantes Y y Z.

Elemento	Variable de estado
Motor	$x_{13} = \theta_{m2}$ Posición Eje motor. $x_{14} = \dot{\theta}_{m2}$ Velocidad Eje motor
Caja de Engranes	$x_{15} = \theta_{g2}$ Posición Caja engranes. $x_{16} = \dot{\theta}_{g2}$ Velocidad caja engranes
Cople	$x_{17} = \theta_{c2}$ Posición Cople. $x_{18} = \dot{\theta}_{c2}$ Velocidad Cople
Tornillo embalado	$x_{19} = \theta_{s2}$ Posición Tornillo embalado. $x_{20} = \dot{\theta}_{s2}$ Velocidad Tornillo embalado
Movimiento axial rosca Tuerca	$x_{21} = X_{s2}$ Posición $x_{22} = \dot{X}_{s2}$ Velocidad Rosca Tuerca
Mesa de Trabajo	$x_{23} = X_{t2}$ Posición Eje motor. $x_{24} = \dot{X}_{t2}$ Velocidad Eje motor

Tabla 2.6: Variables de estado eje Y.

$$
\begin{aligned}
\dot{x}_{13} &= x_{14} \\
\dot{x}_{14} &= \frac{U(t)}{J_{m2}} - \frac{K_{m2}}{J_{m2}}x_{13} + \frac{K_{m2}}{J_{m2}}x_{15} - \frac{b_{m2}}{J_{m2}}x_{14} + \frac{b_{m2}G_1}{J_{m2}}x_{16} \\
\dot{x}_{15} &= x_{16} \\
\dot{x}_{16} &= x_{13}\frac{K_{m2}G_2}{J_{g2}} + x_{14}\frac{b_{m2}G_2}{J_{g2}} - x_{15}\frac{(K_{m2}G_2^2 + K_{g2})}{J_{g2}} - x_{16}\frac{(b_{m2}G_1^2 + b_{g2})}{J_{g2}} \\
&\quad + x_{17}\frac{K_{g2}}{J_{g2}} + x_{18}\frac{b_{g2}}{J_{g2}} \\
\dot{x}_{17} &= x_{18} \\
\dot{x}_{18} &= x_{15}\frac{K_{g2}}{J_{c2}} + x_{16}\frac{b_{g2}}{J_{c2}} - x_{17}\frac{(K_{g2} + K_{c2})}{J_{c2}} - x_{18}\frac{(b_{g2} + b_{c2})}{J_{c2}} + x_{19}\frac{K_{c2}}{J_{c2}} + x_{20}\frac{b_{c2}}{J_{c2}} \\
\dot{x}_{19} &= x_{20} \\
\dot{x}_{20} &= x_{17}\frac{K_{c2}}{J_{s2}} + x_{18}\frac{b_{c2}}{J_{s2}} - x_{19}\frac{(K_{c2} + K_{s2})}{J_{s2}} - x_{20}\frac{(b_{c2}b_{s2})}{J_{s2}} + x_{21}\frac{R_{n2}K_{s2}}{J_{s2}} + x_{22}\frac{R_{n2}b_{s2}}{J_{s2}} \\
\dot{x}_{21} &= x_{22} \\
\dot{x}_{22} &= x_{19}\frac{R_{n2}K_{s2}}{m_{s2}} + x_{20}\frac{R_{n2}b_{s2}}{m_{s2}} - x_{21}\frac{(K_{s2}R_{n2}^2 + K_{r2} + K_{t2})}{m_{s2}} - x_{22}\frac{(b_{s2}R_{n2}^2 + b_{r2} + b_{t2})}{m_{s2}} \\
&\quad + x_{23}\frac{K_{t2}}{m_{s2}} + x_{24}\frac{b_{t2}}{m_{s2}} \\
\dot{x}_{23} &= x_{24} \\
\dot{x}_{24} &= x_{21}\frac{K_{t2}}{m_{t2}} - x_{23}\frac{K_{t2}}{m_{t2}} + x_{22}\frac{b_{t2}}{m_{t2}} - x_{24}\frac{b_{t2}}{m_{t2}}
\end{aligned}
\tag{2.18}
$$

Asignación de las variables de estado para el eje Z.

Elemento	Variable de estado
Motor	$x_{25} = \theta_{m3}$ Posición Eje motor. $x_{26} = \dot{\theta}_{m3}$ Velocidad Eje motor
Caja de Engranes	$x_{27} = \theta_{g3}$ Posición Caja engranes. $x_{28} = \dot{\theta}_{g3}$ Velocidad caja engranes
Cople	$x_{29} = \theta_{c3}$ Posición Cople. $x_{30} = \dot{\theta}_{c3}$ Velocidad Cople
Tornillo embalado	$x_{31} = \theta_{s3}$ Posición Tornillo embalado. $x_{32} = \dot{\theta}_{s3}$ Velocidad Tornillo embalado
Movimiento axial rosca Tuerca	$x_{33} = X_{s3}$ Posición $x_{34} = \dot{X}_{s3}$ Velocidad Rosca Tuerca
Mesa de Trabajo	$x_{35} = X_{t3}$ Posición Eje motor. $x_{36} = \dot{X}_{t3}$ Velocidad Eje motor

Tabla 2.7: Variables de estado eje Z.

$$
\begin{aligned}
\dot{x}_{25} &= x_{14} \\
\dot{x}_{26} &= \frac{U(t)}{J_{m2}} - \frac{K_{m2}}{J_{m2}}x_{13} + \frac{K_{m2}}{J_{m2}}x_{15} - \frac{b_{m2}}{J_{m2}}x_{14} + \frac{b_{m2}G_1}{J_{m2}}x_{16} \\
\dot{x}_{27} &= x_{16} \\
\dot{x}_{28} &= x_{13}\frac{K_{m2}G_2}{J_{g2}} + x_{14}\frac{b_{m2}G_2}{J_{g2}} - x_{15}\frac{(K_{m2}G_2^2 + K_{g2})}{J_{g2}} - x_{16}\frac{(b_{m2}G_1^2 + b_{g2})}{J_{g2}} \\
&\quad + x_{17}\frac{K_{g2}}{J_{g2}} + x_{18}\frac{b_{g2}}{J_{g2}} \\
\dot{x}_{29} &= x_{18} \\
\dot{x}_{30} &= x_{15}\frac{K_{g2}}{J_{c2}} + x_{16}\frac{b_{g2}}{J_{c2}} - x_{17}\frac{(K_{g2} + K_{c2})}{J_{c2}} - x_{18}\frac{(b_{g2} + b_{c2})}{J_{c2}} + x_{19}\frac{K_{c2}}{J_{c2}} + x_{20}\frac{b_{c2}}{J_{c2}} \\
\dot{x}_{31} &= x_{20} \\
\dot{x}_{32} &= x_{17}\frac{K_{c2}}{J_{s2}} + x_{18}\frac{b_{c2}}{J_{s2}} - x_{19}\frac{(K_{c2} + K_{s2})}{J_{s2}} - x_{20}\frac{(b_{c2}b_{s2})}{J_{s2}} + x_{21}\frac{R_{n2}K_{s2}}{J_{s2}} + x_{22}\frac{R_{n2}b_{s2}}{J_{s2}} \\
\dot{x}_{33} &= x_{22} \\
\dot{x}_{34} &= x_{19}\frac{R_{n2}K_{s2}}{m_{s2}} + x_{20}\frac{R_{n2}b_{s2}}{m_{s2}} - x_{21}\frac{(K_{s2}R_{n2}^2 + K_{r2} + K_{t2})}{m_{s2}} - x_{22}\frac{(b_{s2}R_{n2}^2 + b_{r2} + b_{t2})}{m_{s2}} \\
&\quad + x_{23}\frac{K_{t2}}{m_{s2}} + x_{24}\frac{b_{t2}}{m_{s2}} \\
\dot{x}_{35} &= x_{24} \\
\dot{x}_{36} &= x_{33}\frac{K_{t3}}{m_{t3}} - x_{35}\frac{K_{t3}}{m_{t3}} + x_{34}\frac{b_{t3}}{m_{t3}} - x_{36}\frac{b_{t3}}{m_{t3}}
\end{aligned}
\tag{2.19}
$$

Obteniendo las ecuaciones de estado para cada grado de libertad del robo, es posible implementar dicho sistema de ecuaciones de estados en un algoritmo que permita simular el comportamiento de cada variable, siendo de elección el software MATLAB, el cual facilita la obtención de dichas gráficas mediante el uso de programación. Esto se desarrolla mediante el planteamiento de las ecuaciones como sistema ODE (Ecuaciones diferenciales ordinarias), el mecanismo de solución de estas ecuaciones permite resolver problemas de valores iniciales con una gran variedad de propiedades implementando el método de Runge-Kutta 4/5, una vez obtenida la solución numérica, es posible observar el comportamiento de las variables de estado, sea así, mediante gráficas de respuesta en tiempo.

2.6. Conclusiones

En este capítulo, se ha llevado a cabo un profundo análisis y estudio acerca de la metodología de obtención de un modelo dinámico para un robot cartesiano. Este enfoque ha sido desarrollado mediante la aplicación de la metodología de parámetros agrupados, una técnica que ha permitido modelar con precisión cada elemento mecánico que constituye el sistema robótico en cuestión. La utilización de la ecuación de Lagrange, una herramienta fundamental en la mecánica avanzada, ha desempeñado un papel crucial en la formulación del modelo dinámico.

No obstante, este proceso de modelado no se limitó únicamente a la consideración de los elementos mecánicos. Se fue un paso más allá al incluir los esfuerzos generalizados del sistema, lo que permitió obtener una representación más completa de las fuerzas y momentos que actúan en el robot durante su funcionamiento. Además, se tuvieron en cuenta las perturbaciones presentes en el entorno, ya que en la realidad, ningún sistema mecánico opera en un entorno perfectamente ideal.

Un objetivo destacado en este capítulo fue la definición de un sistema en variables de estado que facilitará la implementación de simulaciones detalladas del sistema. Estas simulaciones se convierten en una herramienta invaluable para comprender y analizar el comportamiento de cada variable de estado en diferentes condiciones de operación. Esto no solo contribuye a la validación del modelo, sino que también es esencial para optimizar el rendimiento del robot y garantizar su funcionamiento seguro y eficiente.

Es importante destacar que el enfoque desarrollado en este capítulo se generaliza para los tres grados de libertad que componen el robot cartesiano, lo que lo convierte en un marco de trabajo sólido y versátil aplicable a una amplia gama de configuraciones de robots de este tipo. En resumen, este capítulo sienta las bases para un profundo entendimiento del comportamiento dinámico de los robots cartesianos, permitiendo avances significativos en su diseño, control y operación.

Capítulo 3

Redes Neuronales

Un modelo es siempre una simplificación de la realidad. Por tanto, nunca será del todo correcto. Pese a ello, algunos modelos nos pueden resultar útiles en la práctica. En palabras de Manfred Eigen, Premio Nobel de Química, una teoría sólo tiene la alternativa de ser correcta o equivocada. Un modelo incluye una tercera posibilidad, que sea correcto pero irrelevante. El principal riesgo que corremos al intentar modelar nuestro cerebro al nivel de neuronas individuales es que nuestro modelo puede ser tan ineficiente como intentar descubrir los principios del vuelo analizando la microbiología de los pájaros. Por ello, resulta mucho más interesante analizar el comportamiento de las redes neuronales a nivel de estructuras neuroanatómicas. Es la perspectiva modular característica del deep learning [36].

Aunque las neuronas biológicas ocupan un espacio tridimensional en el cerebro de un animal, lo que les impone restricciones con respecto a su configuración e interconectividad, los modelos de neuronas artificiales que utilicemos no han de respetar las dimensiones físicas de una neurona real. Si queremos implementarlas en hardware, las técnicas de fabricación de dispositivos electrónicos que utilicemos posiblemente nos impondrán sus restricciones, en ocasiones más restrictivas que las existentes en el interior de un cerebro biológico. No en vano, la tecnología comercial de fabricación de circuitos integrados es, básicamente bidimensional [37].

Las redes neuronales son más que otra forma de emular ciertas características propias de los seres humanos, como la capacidad de memorizar y de asociar hechos, si se examinan con atención aquellos problemas que no pueden expresarse a través de un algoritmo, se observará que todos ellos tienen una característica en común: la experiencia. El hombre es capaz de resolver estas situaciones acudiendo a la experiencia propia. Así, parece claro que una forma de aproximarse al problema consiste en la construcción de sistemas que sean capaces de reproducir esta característica humana. En definitiva, las redes neuronales no son más que un modelo artificial y simplificado del cerebro humano, que es el ejemplo más perfecto del que disponemos para un sistema que es capaz de adquirir conocimiento a través de la experiencia.

Orígenes: el Neocognitrón de Fukushima

El origen de las RNC se remonta a los años 70, proveniente de los estudios de David Hubel y Torsten Wiesel sobre el cortex estriado, que es la primera parte del cortex visual que infiere en el procesamiento de la información visual, que proviene directamente del tálamo. Los trabajos de Hubel y Wiesel, responsables de recibir el Premio Nobel en 1981. Como primera descripción, formaron su "cubito de hielo", un modelo de procesamiento de la información visual en el córtex, en donde propusieron llamar "hipercolumna" a los pequeños fragmentos del córtex.

En la misma década en la que Hubel y Wiesel publicaron su modelo sobre una propuesta de organización, un investigador japonés Kuhiniko Fukushima, propuso un modelo de red neuronal artificial bioinspirado: el cognitrón [38]. Pocos años después, Fukushima publicaría una versión mejorar de su primer modelo, al cual llamó: neocognitrón [39], según la configuración de los modelos de esta época, análogamente, la señal captada por conos y bastones en la retina se transmite hasta el núcleo geniculado lateral en el tálamo. El funcionamiento de estos modelos por separados era de funcionamiento lineal, fueron estos las bases que le permitieron crear un primer modelo de lo que hoy en día sería una configuración convolucional [40], en este modelo, la red neuronal está compuesta por una retina, que sirve como capa de entrada y una jerarquía de conexiones entre los módulos o niveles, donde cada nivel estaba compuesto por dos capas, una de células simples (cognitrón) y otra de células compuestas (neocognitrón).

Como veremos más adelante, la arquitectura de las redes de Fukushima es muy similar a las de las redes convolutivas actuales: las células simples corresponderían a las capas convolutivas de las redes actuales, mientras que las células compuestas, desempeñarían un rol similar al de las capas de *pooling* o submuestreo. La principal diferencia entre ambos modelos reside en que los modelos de Fukushima carecían de un algoritmo de e adecuado. En el caso de las redes convolutivas, se utiliza la misma estrategia que en las redes multicapa convencionales: gradiente descendente y propagación de errores hacia atrás. En el caso del cognitrón o del neocognitrón, su mecanismo de "entrenamiento" es algo más laborioso. El término "entrenamiento" aparece entrecomillado porque la red, más que entrenarse como cualquier modelo de aprendizaje automático, se diseña manualmente en función de su objetivo.

3.1. Redes neuronales artificiales

Las Redes Neuronales fueron originalmente una simulación abstracta de los sistemas nerviosos biológicos, constituidos por un conjunto de unidades llamadas neuronas o nodos conectados unos con otros. El primer modelo de red neuronal fue propuesto en 1943 por McCulloch y Pitts en términos de un modelo computacional de actividad nerviosa. Este modelo era un modelo binario, donde cada neurona tenía un escalón o umbral prefijado, y sirvió de base para los modelos posteriores.

Las redes neuronales artificiales (RNA) surgen en los años 40 a raíz de la teoría neuronal de Ramón y Cajal de finales del siglo XIX, en plena época del desarrollo de las teorías biológicas del aprendizaje. Estas teorías se basan en el estudio de los fenómenos biológicos que se dan en el cerebro para que el aprendizaje tenga lugar. A raíz de la tendencia del ser humano a implementar de forma computacional muchas de las funciones realizadas por los seres vivos, las redes neuronales nacen con la intención de simular el funcionamiento de las neuronas del cerebro humano [41]. Las RNA han sido recientemente utilizadas para problemas de clasificación y segmentación de imágenes, regresión de datos, predicción de señales, aumento de resolución, entre otros problemas aplicados a la robótica, medicina, astronomía y economía. Las RNA pueden hallar soluciones prácticas a problemas donde otras metodologías convencionales no pueden alcanzar.

Ventajas de una red neuronal

Debido a su constitución y a sus fundamentos, las redes neuronales artificiales presentan un gran número de características semejantes a las del cerebro. Por ejemplo, son capaces de aprender de la experiencia, de generalizar de casos anteriores a nuevos casos, de abstraer características esenciales a partir de entradas que representan información irrelevante, etc. Esto hace que ofrezcan numerosas ventajas y que este tipo de tecnología se esté aplicando en múltiples áreas. Entre las ventajas se incluyen:

- **Aprendizaje Adaptativo.** Capacidad de aprender a realizar tareas basadas en un entrenamiento o en una experiencia inicial.
- **Auto-organización.** Una red neuronal puede crear su propia organización o representación de la información que recibe mediante una etapa de aprendizaje.
- **Tolerancia a fallos.** La destrucción parcial de una red conduce a una degradación de su estructura; sin embargo, algunas capacidades de la red se pueden retener, incluso sufriendo un gran daño.
- **Operación en tiempo real.** Los cómputos neuronales pueden ser realizados en paralelo; para esto se diseñan y fabrican máquinas con hardware especial para obtener esta capacidad.
- **Fácil inserción dentro de la tecnología existente.** Se pueden obtener chips especializados para redes neuronales que mejoran su capacidad en ciertas tareas. Ello facilitará la integración modular en los sistemas existentes.

Elementos básicos que componen una red neuronal

La neurona artificial consta de varias entradas x_i , una desviación b, una función de activación f y una salida y como se muestra en la figura 3.1-a y en la ecuación 3.1, donde w_i son los pesos sinápticos asociados a cada entrada. El perceptrón es el arreglo más sencillo de neuronas que combina las entradas denominadas como capa de entrada para obtener una capa de salida [5]. El perceptrón puede hallar soluciones no lineales si y solo si se utiliza una función de activación no lineal: Sigmoide, Rectificación lineal, tangente hiperbólica entre otras [36]. Las redes neuronales modernas constan de varias capas ocultas (ver Firua 3.1): cada una de ellas procesa la información de la capa de entrada para obtener un resultado deseado en la capa de salida.

$$y_j = f\left(\sum_{i=0}^{n-1} x_i w_{ij} + b_j\right) \tag{3.1}$$

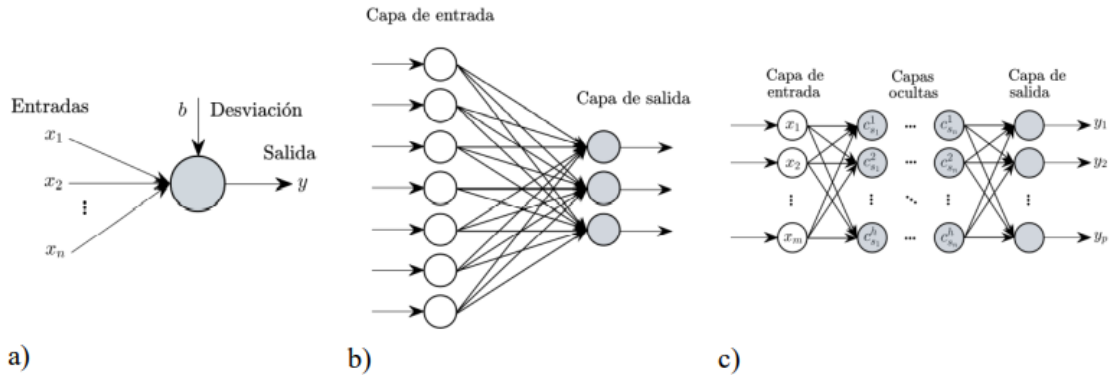


Figura 3.1: Partes de una red neuronal artificial: a) neurona con n entradas, una desviación b y una función de activación f, b) perceptrón simple, c) red neuronal artificial completamente conectada de varias capas ocultas.

Cuando se tiene una conexión de una sola capa de neuronas, se habla de una estructura tipo perceptrón neuronal. Los modelos de neuronas utilizados en redes neuronales artificiales combinan sus entradas usando pesos que modelan sus conexiones sinápticas y, a continuación, le aplican a la entrada neta de la neurona una función de activación o transferencia. La entrada neta de la neurona recoge el nivel de estímulo que la neurona recibe de sus entradas y es la función de activación la que determina cuál es la salida de la neurona. Conectando varias capas, se crea una red neuronal profunda que tiene una mayor capacidad para procesar información. En la Figura 3.2 se observa una red neuronal completamente conectada donde existe más de una capa intermedia. Una característica de esta arquitectura es que tiene conexiones completas: esto es, cada neurona de una capa c está interconectada con todas las neuronas de la capa c-1 (capa anterior). Las redes neuronales convolucionales comparten pesos sinápticos. Para el caso de las que utilizan imágenes de entrada, solo en regiones acotadas analizan información que en cada capa se van interconectando las regiones iniciales hasta llegar a una red neuronal completamente conectada que toma las características extraídas previamente (Figura 3.3).

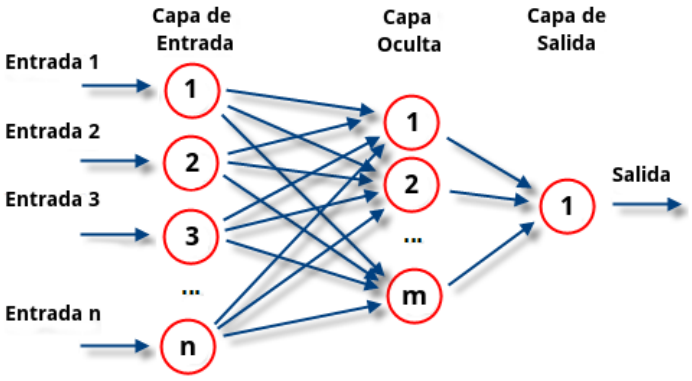


Figura 3.2: Red neuronal de múltiples capas.

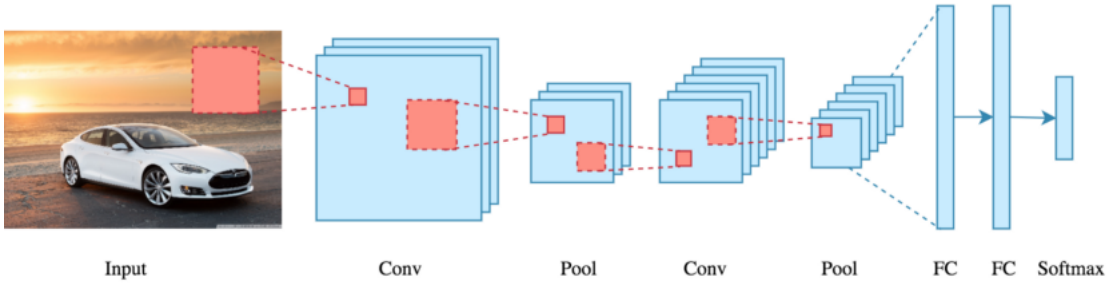


Figura 3.3: Imagen procesada por filtros convolucionales.

A los pesos sinápticos de una red convolucional se les denomina filtros ya que dada una entrada el resultado es una versión que puede resaltar bordes, difuminar áreas u otro tipo de filtro como se muestra en la Figura 3.4 A estas salidas procesadas por los filtros convolucionales se denominan mapas de características.

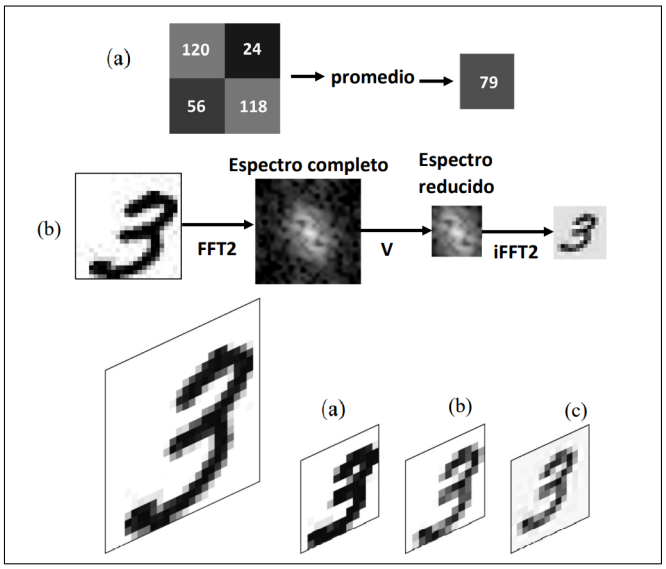


Figura 3.4: capas de reducción para redes convolucionales

Una red neuronal usada para clasificación de imágenes fue la red AlexNet [42]. Consiste en cinco capas convolucionales, tres capas de reducción y tres completamente conectadas (Figura 3.5). En la Figura 3.5 está una red neuronal convolucional para clasificar imágenes de una imagen de $n \times n$ píxeles [36]. Esta red neuronal utiliza capaz de reducción que son distintas a las mencionadas en la Figura 3.5.

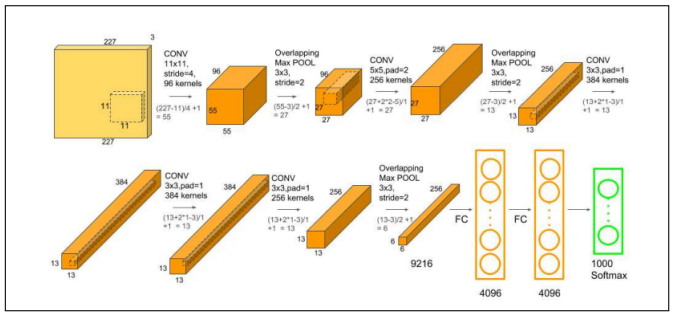


Figura 3.5: Red neuronal AlexNet para clasificación de imágenes.

Las redes neuronales convolucionales tienen diversas aplicaciones como hallar la posición y orientación de una cámara usando la información extraída de una red convolucional. Son usadas para detectar fallas mecánicas por medio de cámaras termográficas, para detectar edificios derrumbados en terremotos, para detectar hemorragias oculares en tomografías a color y para mejorar la compresión de imágenes (Figura 3.6), por mencionar solo algunas.

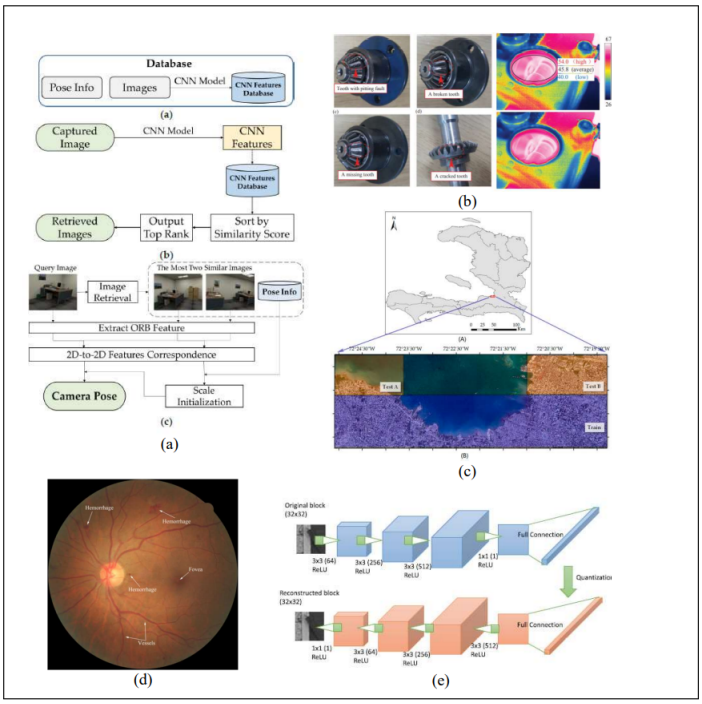


Figura 3.6: Aplicación de redes neuronales convolucionales.

Las redes neuronales se entrenan en dos formas: entrenamiento supervisado y no supervisado. El primer tipo se sabe que debe hacer la red neuronal ya que se muestran datos de entrada etiquetados con las salidas deseadas. Para la segunda forma, una red neuronal crea conjuntos de clasificación de los datos.

Para el entrenamiento supervisado, se utiliza el gradiente del error respecto a los pesos sinápticos y se va descendiendo con el algoritmo de propagación del error [36]. El problema cuando se entrena una red neuronal artificial es que se puede dar el caso en que solo memoriza y no es capaz de generalizar conocimiento. Para prevenir esto, comúnmente se utiliza técnicas para aumentar los datos de entrada, pero modificando su orientación, tamaño y modificando la entrada con filtros pero que esencialmente tienen la idea que se quiere clasificar o de los datos para hacer una regresión.

3.2. Redes neuronales convolucionales

Las *Redes Neuronales Convolucionales* hoy en día se han convertido en una de las técnicas más populares dentro del campo del aprendizaje automático, que es muy utilizada en el reconocimiento de imágenes. La capacidad de las RNC puede ser controlada variando sus parámetros de función. Elanálisis y procesamiento de imágenes fue uno de los primeros campos donde el *deep learning* comenzó con aportaciones más fuertes en las distintas áreas de implementación.

Evidentemente como su nombre lo indica, las RNC se basan en el uso la convolución, una operación matemática ampliamente conocida y utilizada en el área de procesamiento de imágenes, como se refirió anteriormente. Su éxito es gracias a la resolución de problemas de visión artificial, que hasta hace unos años no muy atrás, era un área intratable por la dificultad propia y las implicaciones que esto tenía, esto sirvió para volver a poner en de moda el uso de *redes neuronales* en la Inteligencia Artificial.

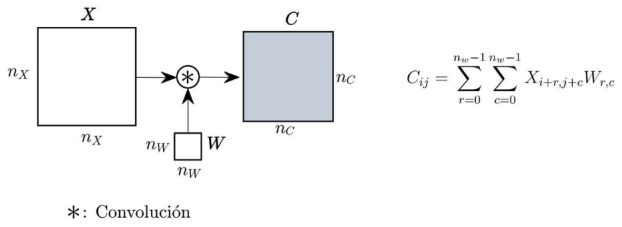
Una señal es, principio, cualquier cantidad física que puede además de ser detectable también es posible transmitir un mensaje, el caso más común las luces de un auto y los distintos mensajes que estas significan. De igual manera, en un contexto más familiar y sencillo, una señal x puede representar una cantidad que varía a lo largo del tiempo, por lo que se representa como $x(t)$. Continuando con funciones un poco más complejas que pueden ser bidireccionales, como lo es la representación de una imagen mediante la función $f(x, y)$ donde x e y corresponden a las coordenadas espaciales que hacen posible conocer el valor de la imagen en cualquier punto en particular.

Desarrollo de una red neuronal convolucional

Las redes neuronales convolucionales que se encargan de analizar imágenes mediante la operación de convolución en dos dimensiones, en la siguiente figura se puede observar que el símbolo $*$ indica dicha operación y sus entradas son un mapa X y un filtro W. Esta operación pondera cada elemento del filtro por una determinada región del mapa de entrada y se suma para obtener un mapa de salida C.

La operación de convolución sin salto está definida en la ecuación 3.2

$$C_{ij} = \sum_{r=0}^{n_w-1} \sum_{c=0}^{n_w-1} X_{i+r,j+c} W_{r,c} \tag{3.2}$$



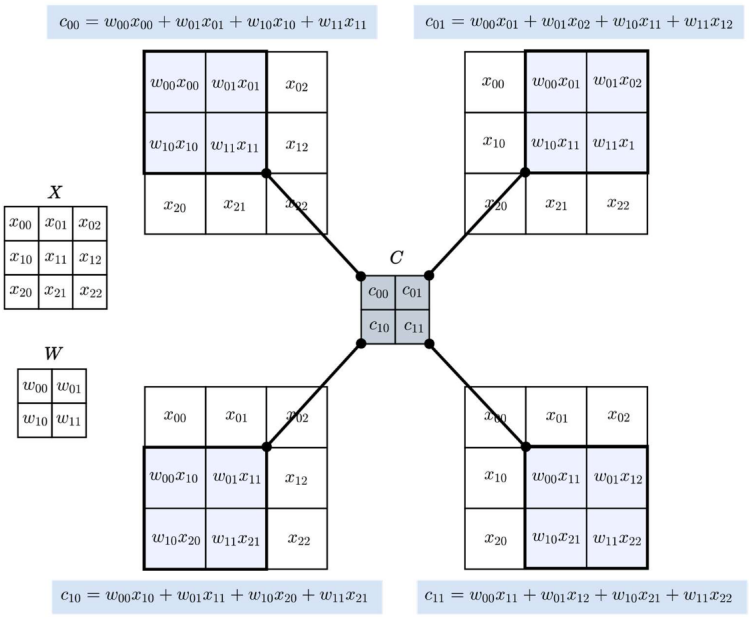


Figura 3.7: Operación de convolución en dos dimensiones utilizando un filtro de 2x2.

La dimensión del mapa de salida en una capa convolucional es igual a $N_x - N_w + 1$ [43] considerando que la imagen de entrada tiene el mismo número de renglones y columnas, al igual que el filtro W. Utilizando la operación de convolución en dos dimensiones como se aprecia en la figura 3.7, se puede construir una capa convolucional que incluya una desviación y una función de activación no lineal, tal como se muestra en la Figura 3.8.

La funciones ReLu y sigmoide utilizadas en [43] se muestran en la Figura 3.8. Una capa convolucional puede tener un solo mapa de entrada X y varios mapas de salida Y. La siguiente figura muestra que para cada mapara de salida Y se necesita añadir un filtro convolucional diferente con características propias. Además, se deben añadir constantes de desviación diferente. La primera sección de la imagen, resaltada en azul, muestra la representación que adopta para una capa de entrada y M salidas.

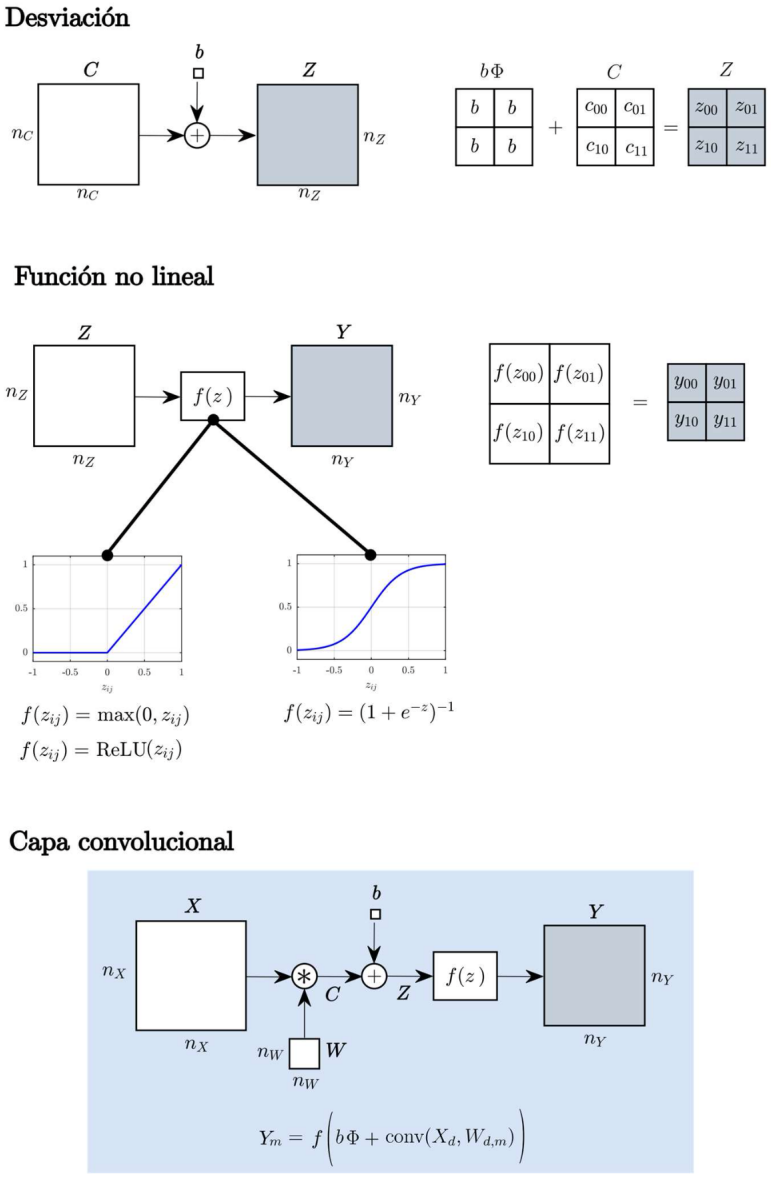


Figura 3.8: Elementos de una capa convolucional de una entrada y una salida.

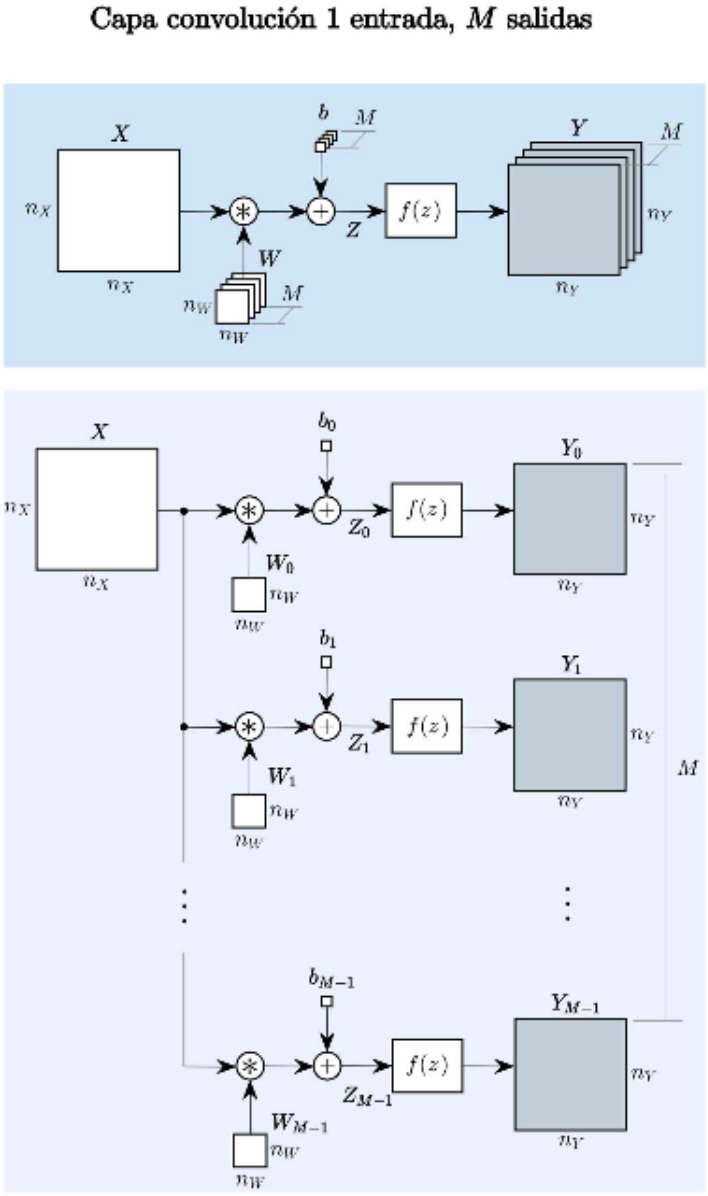


Figura 3.9: Capa convolucional de una entrada y M salidas.

No se descarta la posibilidad donde se tienen varios mapas de entrada X, pero sólo un mapa de salida Y. Dicha configuración es útil cundo se desea reducir el número de mapas en una red neuronal convolucional. En la figura 3.10 se observa la representación utilizada para una capa convolucional de D entradas y una sola salida.

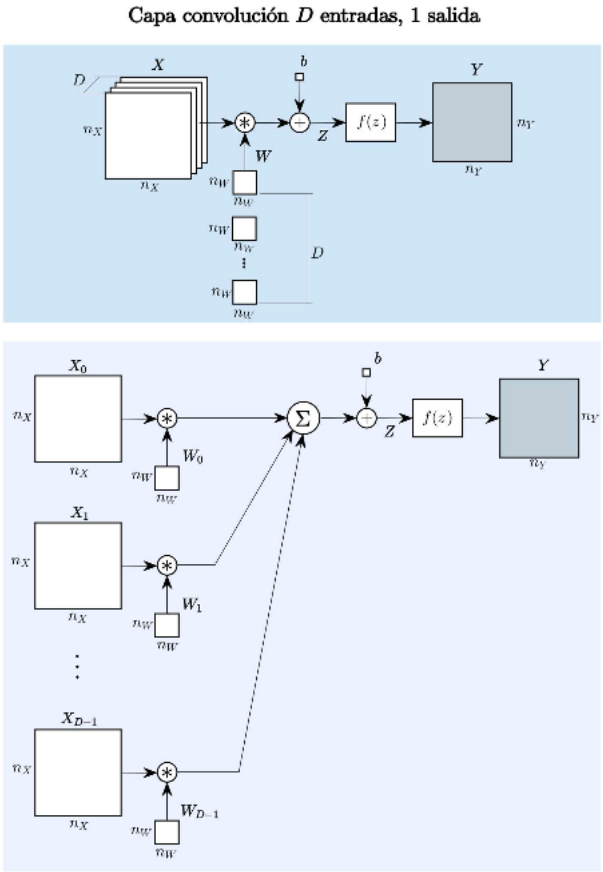


Figura 3.10: Capa convolucional de D entradas y una salida.

Es importante resaltar que al igual que la capa de una sola entrada y M salidas mostradas en la figura 3.9, es necesario utilizar un filtro W para cada entrada. Sin embargo, solo se utiliza una desviación ya que solo se tiene una de salida.

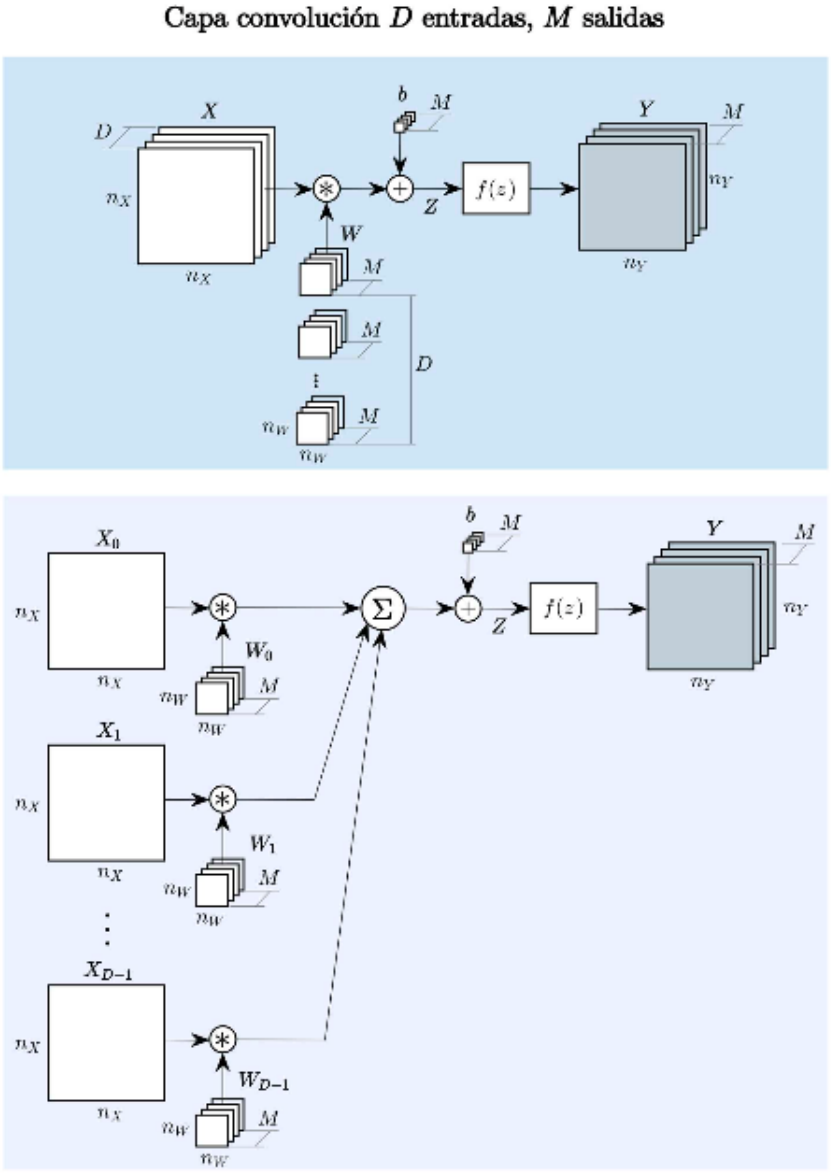


Figura 3.11: Capa convolucional de M entradas y salidas.

Finalmente, existe el caso cuando existen múltiples mapas de entrada y de salida. En la figura 3.11 se observa la representación utilizada para una capa convolucional de D entradas y M salidas. Observe que la cantidad de filtro W es igual a la multiplicación del número de entradas por el número de salidas [36].

3.3. Propuesta de la red neuronal convolucional

La red neuronal que se plantea desarrollar, presenta por lo menos 3 capas de convolución pero el número de estas se definirá conforme a las pruebas que se vayan realizando, estas capas se encargarán de recorrer todos los datos numéricos que representar los pixeles, pero en vez de realizarlo de manera independiente, lo realiza de manera grupal. Para tener una representación visual, imaginemos que tenemos como entrada a la primer capa convolucional, una imagen de 100 por 100 pixeles, y lo que hace la convolución es tomar un grupo de celdas cercanas, puede ser el caso de 5 por 5, y así recorrer de manera secuencial, hasta que se le aplique a toda la capa de entrada, el tamaño de este grupo está definido por un parámetro que se llama filtro o kernel. Ahora, cada uno de estos grupos de pixeles seleccionados se va a multiplicar por otra matriz, a esta operación se le llama "aplicarle un filtro", esto quiere decir que se implementará una función matemática, la cual será definida de manera conveniente conforme a lo que se desea encontrar.

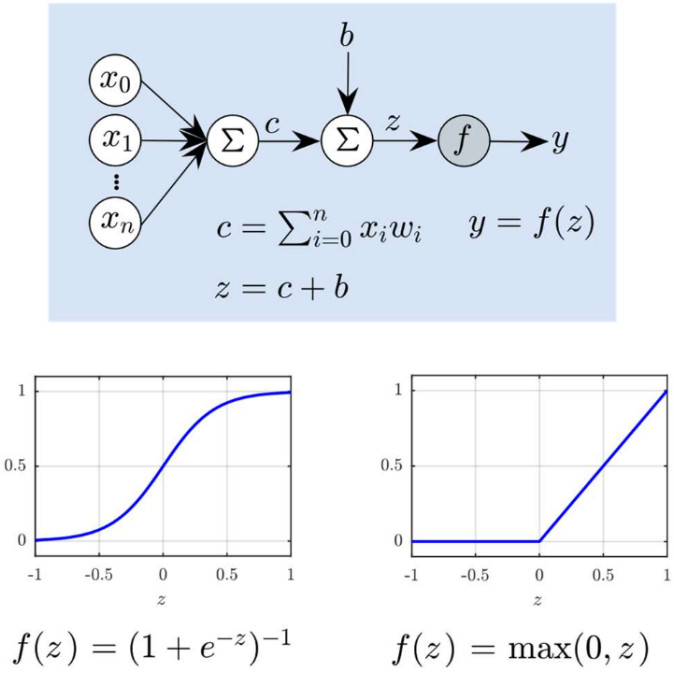


Figura 3.12: Estructura de una neurona.

En la metodología de identificación estudiada [43], las red neuronal convolucional, utiliza después de las capas convolucionales un conjunto de capas completamente conectadas. La figura 3.12 muestra las partes que componen una neurona artificial con su respectiva función de activación. Las capas completamente conectadas y las capas convolucionales se utiliza la función de activación ReLu y la función sigmoide.

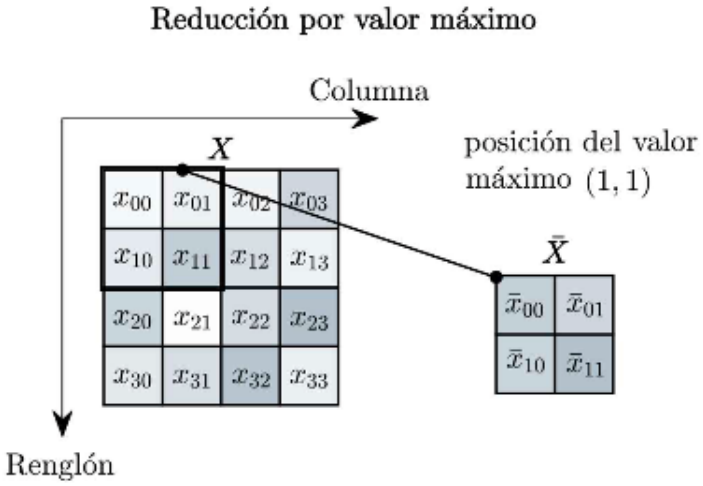


Figura 3.13: Construcción de una capa por reducción de valor máximo.

Este trabajo de investigación, se utilizará la reducción de mapas de características (imagen) entre capas convolucionales con la técnica del valor máximo. La figura 3.13 es posible apreciar como la reducción de un mapa de entrada de 4×4 a uno de 2×2 mediante un filtro que determina el valor máximo de la selección. Se describe el desarrollo de una red neuronal convolucional para el caso de estudio. La identificación del robot cartesiano de tres grados de libertad, permite proponer una red neuronal convolucional con dos imágenes de entrada y cuatro de salidas. En esta primera capa

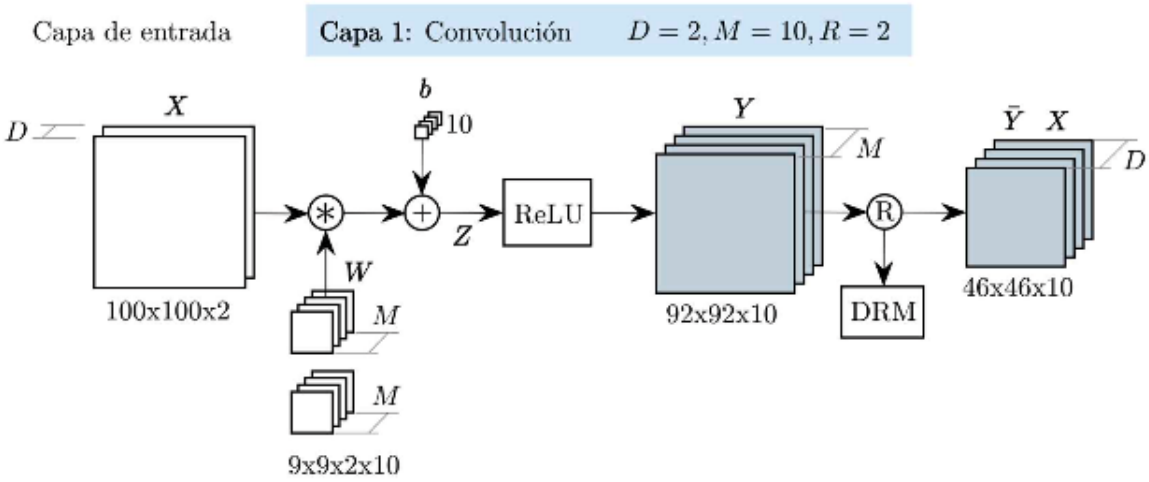


Figura 3.14: Primera capa para la CNN.

convolucional se han propuesto 10 filtros para la primera capa convolucional de 9×9 y se utiliza una reducción por valor máximo de 2×2, en la figura 3.14 está la primera de la red neuronal convolucional, se puede observar que al tener dos imágenes de entrada por lo tanto la cantidad del filtro es de 2×10. Para el entrenamiento de la red neuronal convolucional se utiliza la dirección de los valores máximos guardados en un vector DRM.

Para poder implementar el entrenamiento con el gradiente descendente se utilia el algoritmo de propagación de error hacia atrás, tal como se describe en [36], implementar el vector DRM ayuda a propagar el gradiente por las direcciones del valor máximo.

La figura 3.15 muestra la segunda capa convolucional para el caso de estudio, se

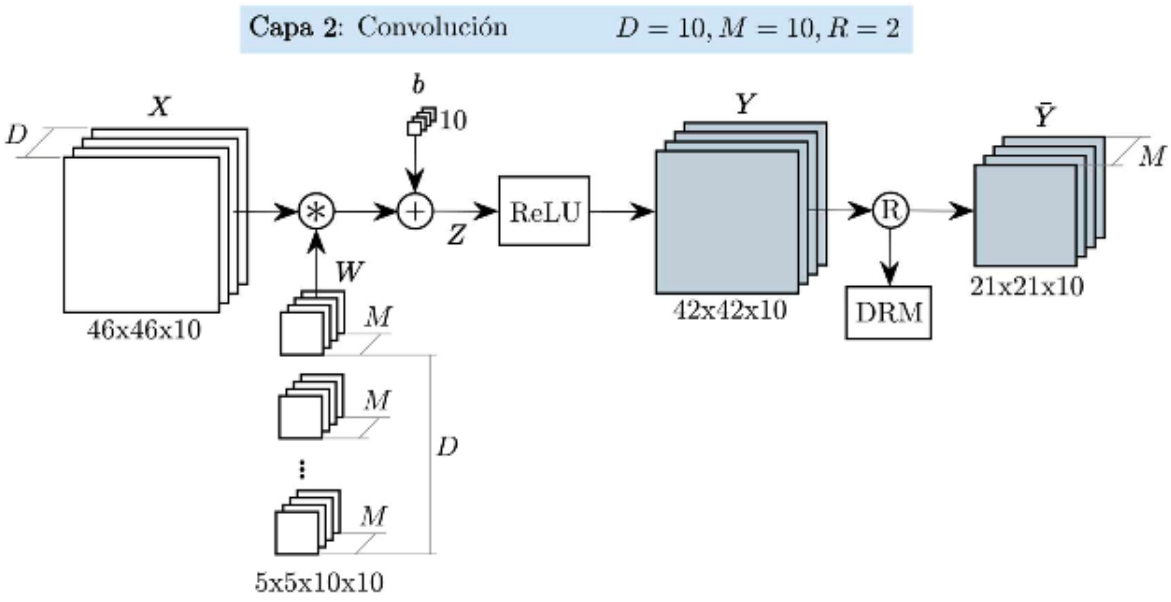


Figura 3.15: Segunda capa para la CNN.

puede observar que el mapa de entrada es de $46 \times 46 \times 10$ y el tamaño de los filtros convolucionales es de 5×5, de igual manera se utilizará la función ReLu se encargada de la activación de esta capa, el tamaño de los filtros convolucionales ha sido definido manualmente de acuerdo al comportamiento de las redes neuronales. La tercera capa convolucional que es apreciable en la figura 3.16. En esta capa no se implementa una capa de reducción y su utiliza un filtro de 3×3 de tamaño.

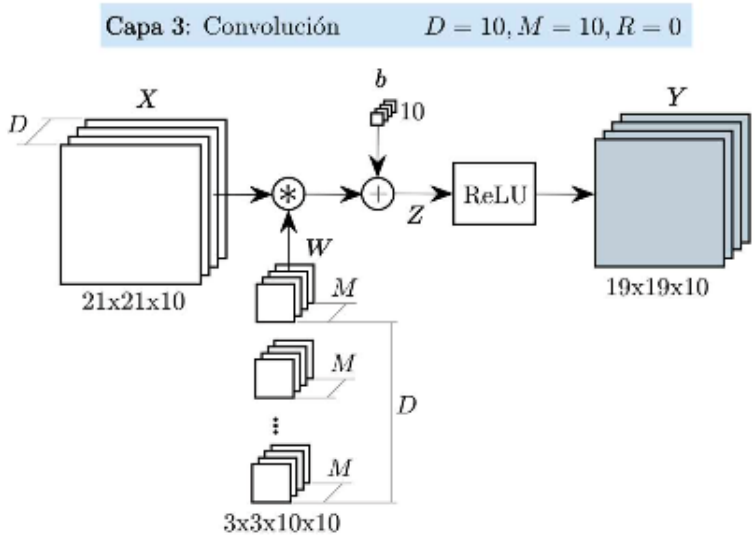


Figura 3.16: Tercera capa para la CNN.

Observe que la primera capa empieza con un filtro convolucional de tamaño 9×9, la segunda es de 5×5 y finalmente la tercera es de 3×3. En la metodología estudiada se justifica la razón de las dimensiones de los filtros, ya que se ha reportado que se puede analizar una imagen con una red neuronal convolucional enfocándose primera en áreas grandes y paulatinamente ir reduciendo el área para poder analizar pequeños detalles en la imagen [43].

La cuarta capa se trata de una capa completamente conectada como se muestra en la figura 3.17. Para este caso se hace la transformación de los mapas de salidas de la capa 3 a un vector de 3610 elementos. Esta capa tiene 100 neuronas de salidas y de igual forma se utiliza la función de activación ReLu. La quinta capa de la figura 3.18 también se trata de una capa completamente conectada y utiliza 100 neuronas de salidas, así como la función de activación ReLu.

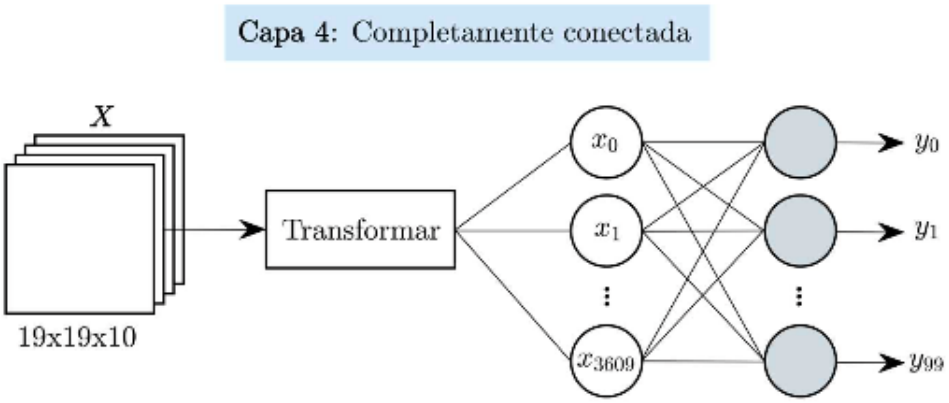


Figura 3.17: Cuarta capa para la CNN.

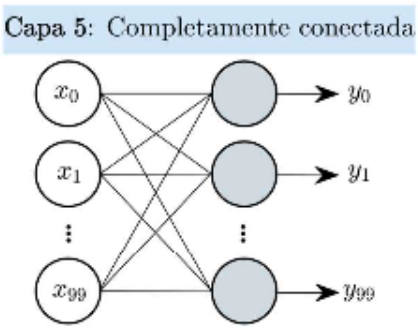


Figura 3.18: Quinta capa para la CNN.

Finalmente, la capa de salida de la figura 3.19 para la estructura de la red propuesta, se utiliza una red neuronal convolucional de cuatro salidas. En esta capa se utiliza la función sigmoide ya que de acuerdo a la literatura [36], es la que mejor responde para la extracción del residuo paramétrico.

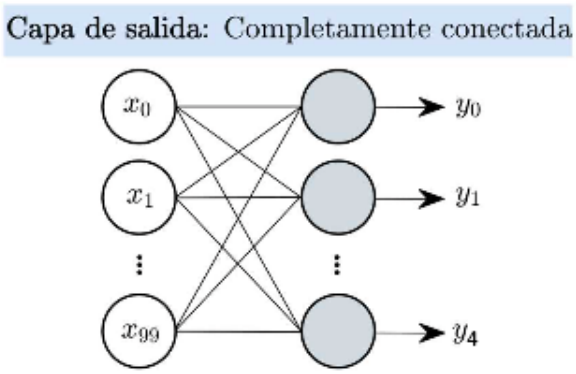


Figura 3.19: Capa de salida para la CNN.

En la figura 3.20 se observa la red neuronal convolucional diseñada para el caso de estudio. La ejecución de las redes neuronales se lleva a cabo desde la capa de entrada hasta la capa de salida. En la Tabla 3.1 se pueden apreciar las dimensiones de la red neuronal convolucional. El número total de neuronas se determina multiplicando los mapas de salida antes de las capas de reducción más el número de neuronas de las capas completamente conectadas. Los pesos sinápticos son calculados por la multiplicación de los filtros convolucionales más las desviaciones y se añade los pesos de las capas completamente conectadas [36].

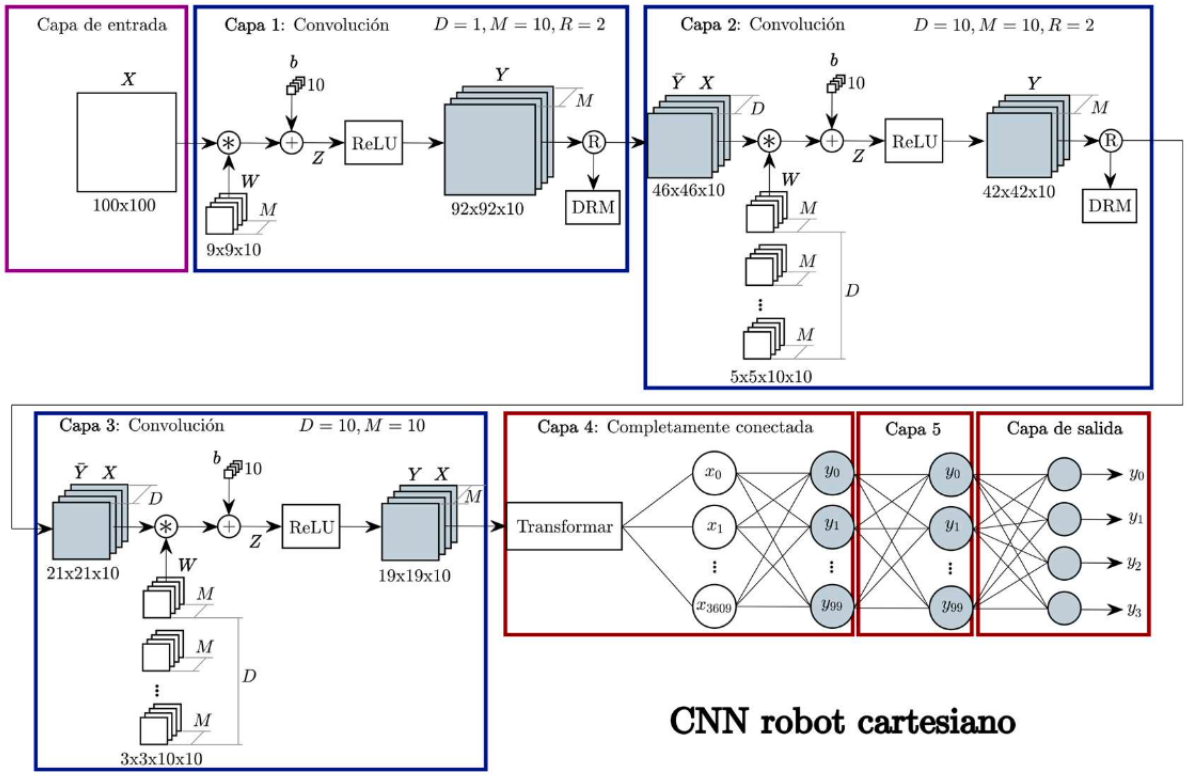


Figura 3.20: Red neuronal convolucional para el robot cartesiano.

	C1	C2	C3	C4	C5	C6
Tamaño entrada	100×100	$46 \times 46 \times 10$	$21 \times 21 \times 10$	3610×1	100×1	100×1
Tamaño salida	$46 \times 46 \times 10$	$21 \times 21 \times 10$	$19 \times 19 \times 10$	100×1	100×1	4×1
Filtros/pesos	$9 \times 9 \times 1 \times 10$	$5 \times 5 \times 10$	$3 \times 3 \times 10$	3610×100	100×100	100×4
Neuronas	84,600	17,640	3610	100	100	4
Pesos	820	2,510	910	361,100	10,100	404

Tabla 3.1: Dimensión de la red neuronal convolucional propuesta para la identificación paramétrica del robot cartesiano de 3 gdl.

El total de neuronas para la red neuronal empleada para realizar la identificación paramétrica de un robot cartesiano de tres grados de libertad es de 106,054, mientras que el número total de pesos sinápticos es de 375,844.

3.4. Entrenamiento de una red neuronal convolucional con el gradiente descendente

Para lograr que las redes neuronales convolucionales capturen el residuo paramétrico presente en las imágenes de entrada, es necesario someterlas a un proceso de entrenamiento utilizando conjuntos de imágenes preexistentes. Durante este proceso, se emplea el método de gradiente descendente del error, siguiendo las pautas descritas en las obras de Chapra y Canale (1999) y Berzal (2018) [36], con el fin de ajustar los pesos sinápticos **W** y las desviaciones **B** El conjunto de parámetros ω engloba tanto los filtros convolucionales como los pesos relacionados con las capas completamente conectadas en la red neuronal. El esquema general del proceso de entrenamiento de las redes neuronales convolucionales se representa en la Figura 3.21. Inicialmente, se extraen las imágenes Ajunto con sus correspondientes etiquetas E de los repositorios de entrenamiento. Estos repositorios para cada caso de estudio se componen de un 50 % de imágenes del tipo A_p y un 50 % de imágenes del tipo A_n. Cada imagen extraída se introduce en la red neuronal convolucional, que aún no ha sido entrenada, y se obtiene su salida para su posterior comparación con la etiqueta correspondiente a la imagen original. A continuación, se calcula una función de costo basada en el error, la cual se propaga desde la capa de salida hasta la capa inicial de la red. Posteriormente, se procede a actualizar los pesos W y las desviaciones B mediante la regla delta, y este proceso se repite un total de N veces.

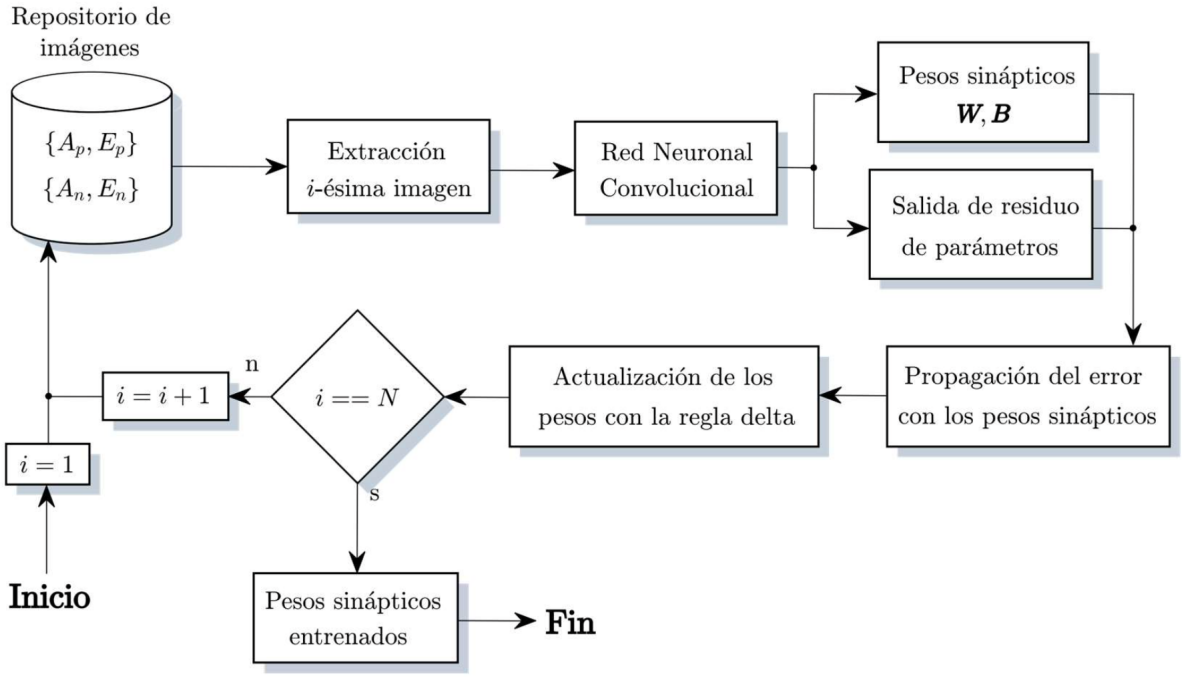


Figura 3.21: Entrenamiento de redes neuronales convolucionales.

Con el objetivo de aplicar de manera efectiva el método del gradiente descendente para entrenar redes neuronales convolucionales, se recurre al algoritmo de retropropagación, como se describe en el trabajo de Berzal (2018). Este algoritmo se encarga de propagar el error que se registra entre la salida generada por la red y la salida deseada. En primer lugar, se establece una función de costo del error E, que se caracteriza por ser siempre un valor positivo. En el contexto de esta investigación, se emplea la ecuación 3.3 como la expresión representativa de dicha función:

$$E = \frac{1}{2}\sum_{j=1}^{N}(y_{jd} - y_j)^2 \tag{3.3}$$

donde y es el vector de salidas de las redes neuronales convolucionales, y_d es el vector de salidas deseadas, j indica la dirección de las salidas y_d y N es el número de salidas de la red. Para poder modificar los pesos sinápticos W y las desviaciones B se utiliza el gradiente de la función de costo con respecto a los pesos de la red. Para las capas de salida de la red de la figura 3.20, el gradiente del error es calculado utilizando la función de costo E, los pesos sinápticos de esta capa con la ecuación 3.4:

$$\begin{aligned} \frac{\partial E}{\partial w_{i,j}^c} &= \left(\frac{\partial E}{\partial y_j^c}\right)\left(\frac{\partial y_j^c}{\partial w_{i,j}^c}\right) \\ y_j^c &= f(z_j^c) \\ z_j^c &= \sum_{i=1}^{M} x_i^c w_{i,j}^c + b_j^c \end{aligned} \tag{3.4}$$

donde c indica la última capa, M son las salidas de la capa c, f es la función de activación de la capa c, w^c y b^c son pesos sinápticos de la capa c. Considerando que $\partial z_j^c / \partial w_{i,j}^c = x_i$, se utiliza la regla delta de la ecuación 3.5 para ajustar los pesos sinápticos w^c de la capa de salida c; para ajustar las desviaciones se considera que la entrada es:

$$\begin{aligned} \delta_j^c &= \left(\frac{\partial E}{\partial y_j^c}\right)\left(\frac{\partial y_j^c}{\partial z_j^c}\right) \\ \Delta w_{i,j}^c &= -\eta \delta_j^c x_i^c \\ \Delta b_j^c &= -\eta \delta_j^c \end{aligned} \tag{3.5}$$

de la ecuación 3.5 η es la taza de aprendizaje. Para ajustar los pesos de la capa $C-1$ se sustituye el término $\frac{\partial E}{\partial y_j^c}$ por la propagación de δ_j^c de la capa C que es ponderada por los pesos sinápticos w^c de la capa C (ecuación 4.14). En este caso las desviaciones no propagan el error ya que su entrada es 1 y no se encuentra conectada a otra parte de la red neuronal.

$$\begin{aligned} \delta_j^{c-1} &= \left(\frac{\partial y_j^c}{\partial z_j^c}\right) \sum_{p=1}^{N} \delta_p^c w_{j,p}^c \\ \Delta w_{i,j}^{c-1} &= -\eta \delta_j^{c-1} x_i^{c-1} \\ \Delta b_j^{c-1} &= -\eta \delta_j^{c-1} \end{aligned} \tag{3.6}$$

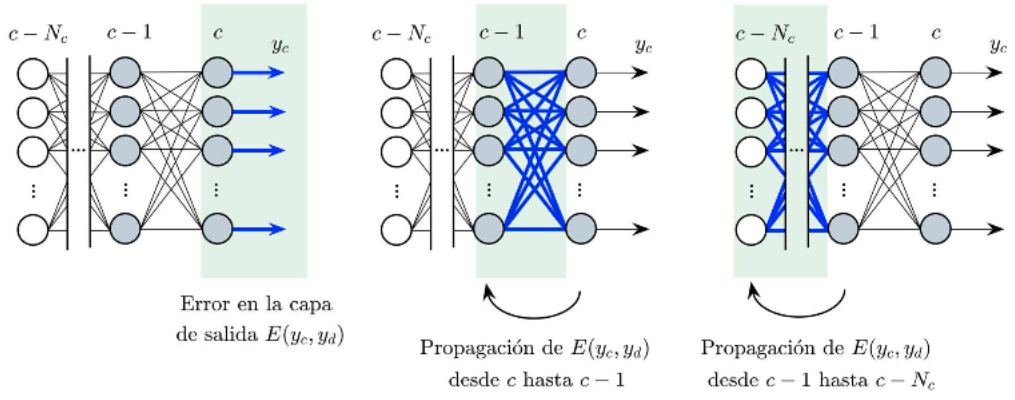


Figura 3.22: Propagación del error entra capas completamente conectadas.

En la representación gráfica proporcionada en la Figura 3.22, se evidencia la transferencia del error desde la capa de salida hacia las capas ocultas mediante la utilización de los pesos sinápticos como factores de ponderación. Las ecuaciones 3.6 se aplican de manera recurrente en todas las capas completamente interconectadas.

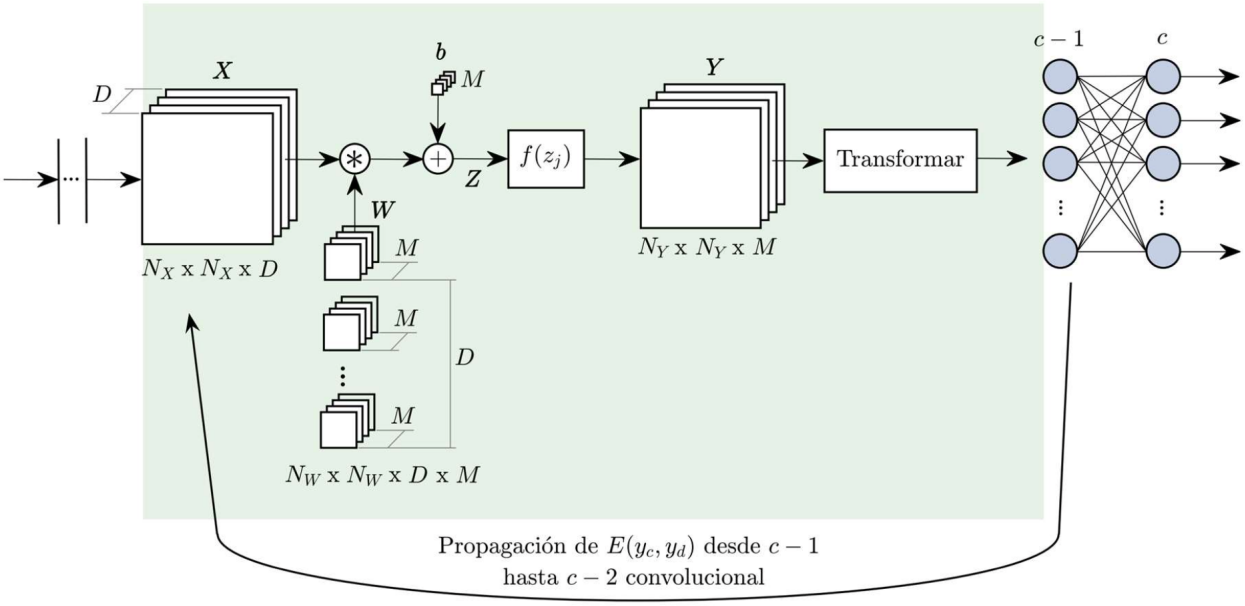


Figura 3.23: Propagación del error entra capas completamente conectadas hacia las capas convolucionales.

Con el fin de difundir el error desde las capas totalmente conectadas hacia las capas convolucionales, se realiza una adaptación del delta correspondiente a la capa completamente conectada $c-1$ de manera que concuerde con la estructura del mapa de características Y. Luego, este delta adaptado se multiplica por la derivada de la función de activación con respecto a su argumento, tal como se describe en la ecuación 3.7 y se ilustra en la Figura 3.23.

$$\begin{aligned} \phi_j^{c-2} &= \sum_{p=1}^{N} \delta_p^{c-1} w_{j,p}^{c-1} \\ \Phi_m^{c-2} &= \text{Mapa}\left(\phi_j^{c.2}\right), \qquad \forall j \in \{1,2,...,N\} \\ \delta_m^{c-2} &= \left(\frac{\partial Y_m^{c-1}}{\partial Z_j^{c-1}}\right) \Phi_m^{c-2} \end{aligned} \tag{3.7}$$

En las ecuaciones anteriores, la función Mapa convierte el vector $\phi_j^{c.2}$ a un formato que se asocie con la forma del mapa Y de la capa convolucional mostrada en la Figura 3.23. Para derivar las fórmulas de actualización de los filtros, comenzamos con un escenario ilustrativo que involucra un mapa de entrada de 3×3 y un filtro de 2×2, como se muestra en la figura 3.24. En este contexto, las salidas se determinan mediante la variable delta, que a su vez influye en la actualización de los parámetros $\mathbf{W}$ y $\mathbf{b}$. La figura 4.31 exhibe las conexiones ponderadas por ω_{11}, ω_{12}, ω_{21} y ω_{22} para el ejemplo mencionado en la figura 3.24. Para calcular ΔW, sumamos las contribuciones de los valores δ correspondientes tanto en las salidas como en las entradas $x_{i,j}$.

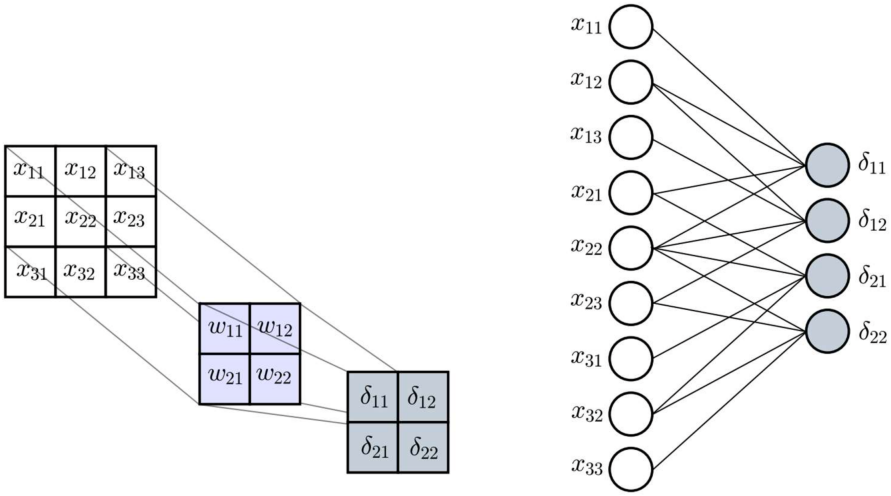


Figura 3.24: Capa convolucional para determinar las ecuaciones de actualización de los filtros convolucionales.

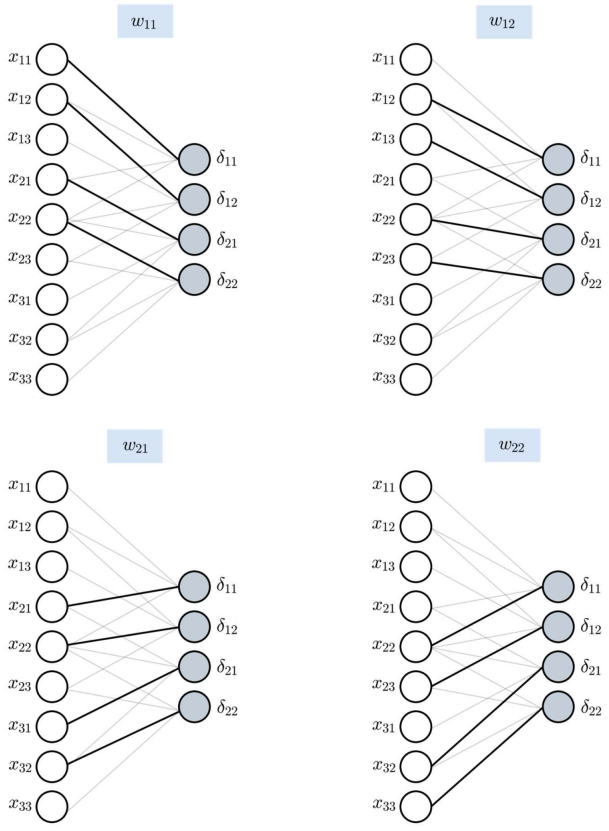


Figura 3.25: Conexiones ponderadas por un filtro convolucional.

$$
\begin{aligned}
\Delta w_{11} &= -\eta\left(\delta_{11}x_{11} + \delta_{12}x_{12} + \delta_{21}x_{21} + \delta_{22}x_{22}\right) \\
\Delta w_{12} &= -\eta\left(\delta_{11}x_{12} + \delta_{12}x_{13} + \delta_{21}x_{21} + \delta_{22}x_{23}\right) \\
\Delta w_{21} &= -\eta\left(\delta_{11}x_{21} + \delta_{12}x_{22} + \delta_{21}x_{31} + \delta_{22}x_{32}\right) \\
\Delta w_{22} &= -\eta\left(\delta_{11}x_{22} + \delta_{12}x_{23} + \delta_{21}x_{32} + \delta_{22}x_{33}\right) \\
\Delta W &= -\eta \mathrm{conv}(X, \delta)
\end{aligned}
\tag{3.8}
$$

Aplicando la regla delta, representada por la ecuación 3.5, podemos deducir que el incremento correspondiente al filtro convolucional presente en la Figura 3.24 se describe mediante la ecuación 3.8 . Al extender esta conceptualización más allá del ejemplo específico de la Figura 3.24, la actualización de los filtros convolucionales se generaliza utilizando la ecuación 3.9. Las desviaciones son determinadas mediante la suma de los valores delta.

$$
\begin{aligned}
\Delta W_{d,m}^{c-2} &= -\eta \mathrm{conv} \\
\Delta b_{m}^{c-2} &= -\eta \sum_{u=1}^{s}\sum_{v=1}^{s} \delta_{m}^{c-2}(u,v)
\end{aligned}
\tag{3.9}
$$

La transmisión del error entre las capas de reducción se lleva a cabo mediante las direcciones DRM mencionadas en la sección 3.7. En la Figura 3.26 se puede apreciar un ejemplo de un mapa de entrada de 8×8 que se reduce utilizando la operación de valor máximo con una ventana de 2×2. La flecha coloreada en azul indica el valor máximo de una región en color cian, el cual se coloca en el mapa reducido. Este valor máximo se determina durante el proceso de alimentar una imagen a la red neuronal y ejecutarla capa por capa hacia adelante. Por otro lado, la flecha de color rojo representa la dirección de retropropagación del error. Es importante destacar que en el estado del arte [43], se emplea la función de valor máximo para reducir los mapas de características y, en consecuencia, la propagación del error siempre debe seguir la dirección del valor máximo (el vector DRM almacena las direcciones del valor máximo en cada capa de reducción).

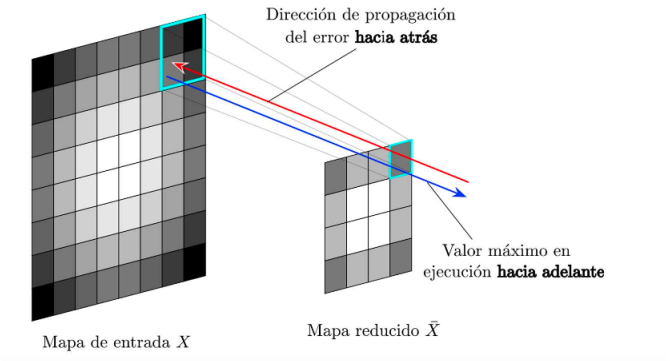


Figura 3.26: Propagación del error en las capas de reducción.

Para llevar a cabo la transmisión del error entre las capas convolucionales, se realiza una operación de suma que tiene en cuenta la contribución de los valores δ ponderados por los filtros correspondientes. La Figura 3.27 ilustra un ejemplo similar al presentado en la Figura 3.24, pero con las entradas identificadas como ϕ, tal como se describe en la ecuación 3.7. A través de la aplicación de la ecuación 3.6, se derivan las ecuaciones que rigen la propagación del error entre las distintas capas convolucionales.

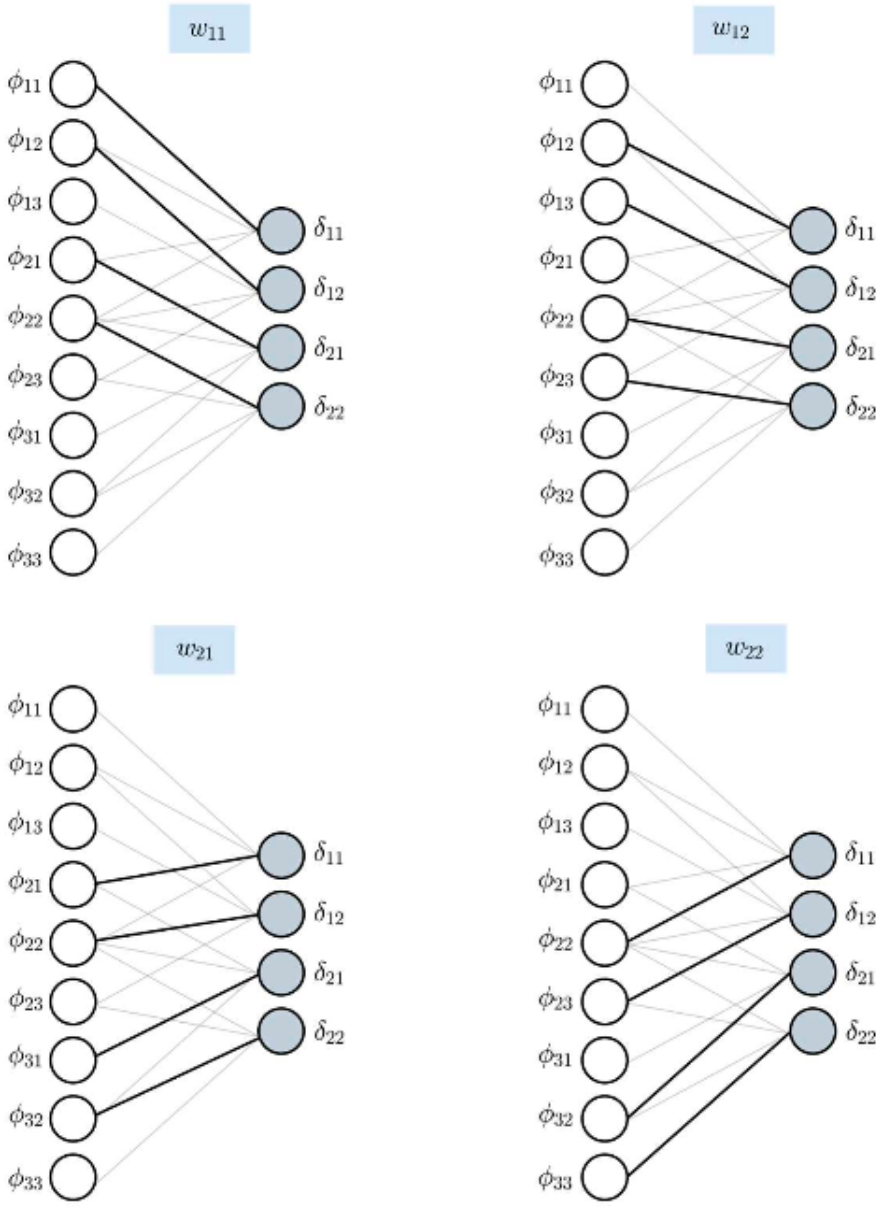


Figura 3.27: Propagación del error en las convolucionales.

$$
\begin{aligned}
\phi_{11} &= \delta_{11} w_{11} \\
\phi_{12} &= \delta_{12} w_{11} + \delta_{11} w_{12} \\
\phi_{13} &= \delta_{12} w_{12} \\
\phi_{21} &= \delta_{21} w_{11} + \delta_{11} w_{21} \\
\phi_{22} &= \delta_{22} w_{11} + \delta_{21} w_{12} + \delta_{12} w_{21} + \delta_{11} w_{22} \\
\phi_{23} &= \delta_{22} w_{12} + \delta_{12} w_{22} \\
\phi_{31} &= \delta_{21} w_{21} \\
\phi_{32} &= \delta_{22} w_{21} + \delta_{21} w_{22} \\
\phi_{33} &= \delta_{22} w_{22}
\end{aligned}
\tag{3.10}
$$

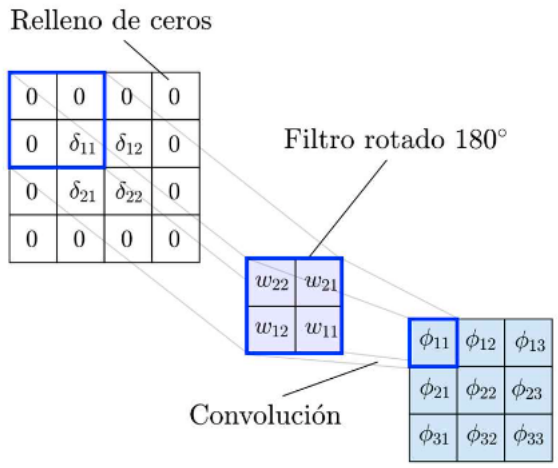


Figura 3.28: Convolución para la propagación del error en las capas convolucionales.

Se procede a reorganizar las ecuaciones 3.10 con el fin de aplicar la convolución en dos dimensiones, como se ejemplifica en la Figura 3.28. En esta instancia, la entrada se extiende con ceros en los bordes y el filtro convolucional se somete a una rotación; el resultado es la propagación ponderada por el filtro entre las capas convolucionales. Esta operación, representada en la figura 3.28, es conocida como convolución completa. Al generalizar esta operación para las capas convolucionales, se obtiene la ecuación 3.11. Este proceso tiene un importante alcance en la aplicación de técnicas de procesamiento de señales y visión por computadora, y su análisis reviste relevancia en contextos científicos y de ingeniería.

$$
\Phi_d = \sum_{m=1}^{M} \text{full-conv}(\delta_m, W_{d,m}) \tag{3.11}
$$

3.5. Generación del repositorio de imágenes con las señales del robot cartesiano

En esta sección, se propone la idea de generar una imagen utilizando las señales del robot (posición, velocidad, aceleración y torque) junto con un conjunto de parámetros dinámicos de entrada para realizar la identificación paramétrica del robot. Para crear la imagen se utiliza el modelo dinámico del robot cartesiano, en las ecuaciones (3.12), (3.13) y (3.14) se encuentran las ecuación del torque correspondiente a cada grado de libertad del robot.

$$J_{m1}\ddot{\theta}_{m1} + k_{m1}(\lambda_1)(\theta_{m1} - G_1\theta_{g1}) + b_{m1}(\dot{\theta}_{m1} - G_1\dot{\theta_{g1}}) = \tau_{m1} \tag{3.12}$$

$$J_{m2}\ddot{\theta}_{m2} + k_{m2}(\lambda_2)(\theta_{m2} - G_2\theta_{g2}) + b_{m2}(\dot{\theta}_{m2} - G_2\dot{\theta_{g2}}) = \tau_{m2} \tag{3.13}$$

$$J_{m3}\ddot{\theta}_{m3} + k_{m3}(\lambda_3)(\theta_{m3} - G_3\theta_{g3}) + b_{m3}(\dot{\theta}_{m3} - G_3\dot{\theta_{g3}}) = \tau_{m3} \tag{3.14}$$

donde θ_{m1}, θ_{m2} y θ_{m3} son las posiciones en el eje x, y, z respectivamente, J_{m1}, J_{m2}, J_{m3} son los momentos de inercia, b_{m1}, b_{m2}, b_{m3} son los coeficientes de fricción viscosa, k_{m1}, k_{m2}, k_{m3} son los coeficientes de la fricción de Coulomb. Con los modelos de las ecuaciones descritos, se hizo el estudio de técnicas de transformación de señales a imagen. La técnica estudiada [43], propone emplear dos vectores sean ψ y α que son denominadas funciones generadoras, por teoría de vectores, se sabe que al multiplicar un vector por la transpuesta de otro vector , se obtiene una matriz, para el caso en cuestión, una imagen llamada A. En la figura 3.29 se observan los vectores ψ y α que ambos contienen N_r elementos propios de las ecuaciones ya descritas. Observe como la imagen es generada por la multiplicación de cada componente de los vectores.

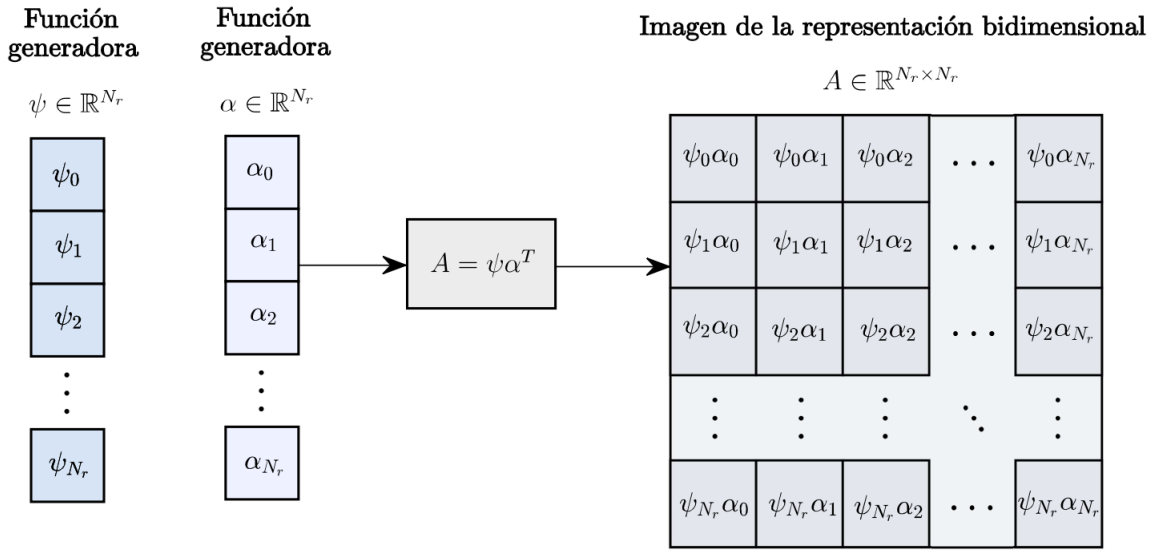


Figura 3.29: Generación de una imagen a partir de dos vectores α y ψ.

Para el robot cartesiano de tres grados de libertad se utilizan las ecuaciones 3.15. Las datos de entrada son las señales de velocidad $\dot{q}_x, \dot{q}_y, \dot{q}_z$, torque τ_x, τ_x, τ_x y el conjunto de parámetros dinámicos de entrada $\hat{\beta} = \{\hat{J}_x, \hat{J}_y, \hat{J}_z, \hat{b}_x, \hat{b}_y, \hat{b}_z, \hat{k}_x, \hat{k}_y, \hat{k}_z, \hat{\lambda}_x, \hat{\lambda}_y, \hat{\lambda}_z, \}$. Es importante destacar que cuando los parámetros de entrada $\hat{\beta}$ sean los correctos, la imagen producida por la ecuación 3.15 será $A = \tau\tau^T$.

$$
\begin{aligned}
\psi_x &= 2\tau_x - \hat{J}_{mx}\ddot{\theta}_{mx} + \hat{k}_{mx}(\hat{\lambda}_x)(\theta_{mx} - G_1\theta_{gx}) + \hat{b}_{mx}(\dot{\theta}_{mx} - G_1\dot{\theta_{gx}}) \\
\alpha_x &= \tau_x \\
\psi_y &= 2\tau_y - \hat{J}_{my}\ddot{\theta}_{my} + \hat{k}_{my}\hat{\lambda}_y(\theta_{my} - G_1\theta_{gy}) + \hat{b}_{my}(\dot{\theta}_{my} - G_1\dot{\theta_{gy}}) \\
\alpha_y &= \tau_y \\
\psi_z &= 2\tau_z - \hat{J}_{mz}\ddot{\theta}_{mz} + \hat{k}_{mz}\hat{\lambda}_z(\theta_{mz} - G_1\theta_{gz}) + \hat{b}_{mz}(\dot{\theta}_{mz} - G_1\dot{\theta_{gz}}) \\
\alpha_z &= \tau_z \\
A &= \psi\alpha^T \\
Z &= \frac{A - min(A)}{max(A) - min(A)}
\end{aligned}
\tag{3.15}
$$

Nótese que las imágenes Z en las ecuaciones (3.15), son las versiones normalizadas de las imágenes A. Las redes neuronales convolucionales tienen un mejor desempeño si los datos de entrada están en un rango fijo [44]. La dimensión de la imagen Z es de $N_r \times N_r$. Para poder estimar la velocidad y la aceleración con los datos obtenidos de la posición del enconder para cada grado de libertad, se utiliza la serie de Taylor de la señal de posición del robot y se utiliza una aproximación de la derivada como

se muestra en la ecuación 3.16

$$\begin{aligned} q_i &= q_{i-1} + \frac{d}{dt}q_{i-1}h + ... + \frac{d^n}{dt^n}q_{i-1}h^n \\ \frac{d}{dt}q_{i-1} &\approx \frac{q_i - q_{i-1}}{h} \end{aligned} \tag{3.16}$$

Debido a que la posición es adquirida mediante un encoder incremental, la derivada de la ecuación 3.16 contendrá pulsos y no es útil para la identificación paramétrica [3] pero, es posible utilizar un filtro pasa bajas para eliminar el ruido numérico inducido por el encoder. En este objetivo, se utiliza la transformada discreta del coseno de la ecuación 3.17 [45] para filtrar las señales.

$$\begin{aligned} X(k) &= \sqrt{\frac{2}{N}} \sum_{i=1}^{N} a(k)x(i) \left[\frac{\pi}{2N}(2i-1)(k-1)\right] \\ a(k) &= \frac{1}{\sqrt{1+\delta(k-1)}} \\ \delta(z) &= 1 \text{ sí y sólo sí } z = 0 \end{aligned} \tag{3.17}$$

Las señales del robot contendrán información repetitiva a lo largo del tiempo que se ejecute el sistema. Por lo tanto, se realiza el estudio de una metodología para el submuestreo de señales utilizando la transformada discreta del coseno [43] donde el autor asegura que dicha técnica de submuestreo presenta ventajas sobre otras metodologías ya existente en el estado del arte.

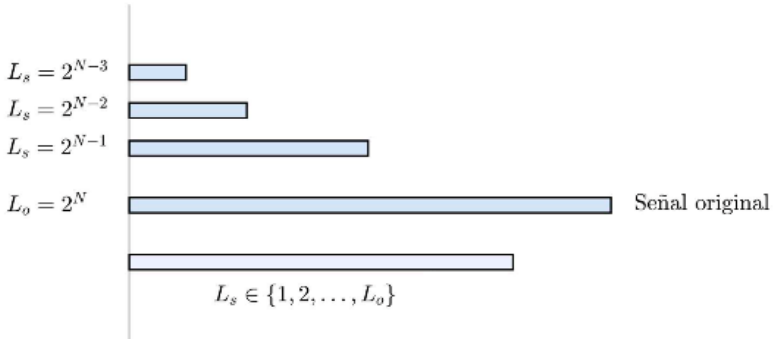


Figura 3.30: Técnica de submuestreo estudiada para la identificación paramétrica.

En la figura 3.30 se puede observar una señal de longitud L_o. Asumiendo que L_0 es igual a 2^N, esto quiere decir que la señal se puede submuestrear sobre "2^{N-1}, 2^{N-2} hasta 1. Desafortunadamente con las técnicas comunmente utilizadas, no puede ser posible submuestrear la señal con una longitud L_s entre "2^N y 2^{N-1}. De la metodología en cuestión, utilizar la transformada discreta del coseno, sí es posible submuestrear la señal a cualquier longitud deseada, utilizando está técnica, L_0 no queda sujeta a ser potencia de dos y puede tomar cualquier valor entero.

$$\begin{aligned} X &= \text{dct}_{\text{II}}(x) \\ X_s &= X(1), X(2), ..., X(L_s) \\ x_n &= \text{idct}_{\text{II}}(X_s) \\ x_s &= x_n \frac{\max(x)}{\max(x_n)} \end{aligned} \tag{3.18}$$

El proceso de submuestrear una señal x de longitud L_0 empleando la transformada discreta del coseno comienza con determinar el espectro de frecuencia X de la señal

x. Posterior a esto se debe recortar el espectro X hasta una longitud deseada L_s, teniendo ya el espectro de frecuencia recortado se debe convertir a tiempo con la transformada discreta del coseno para poder obtener x_n. Por último, se ajusta el rango de x_n para finalmente obtener una señal submuestreada x_s con logitud L_s. Esta metodología está limitada por la frecuencia máxima del espectro recortado, de ser el caso en donde la señal supere este límite, se debe aplicar el filtro pasa bajas que permita obtener un mejor resultado en el submuestreo.

Algoritmo para generación de imágenes

En esta sección se revisa la metodología a seguir para la generación de imágenes a partir de las señales emitidas por un robot cartesiano. Las señales de entrada comprenden datos clave, incluyendo la posición, velocidad, aceleración y torque del robot, junto con un conjunto de parámetros aleatorios que se utilizarán en el proceso de reconstrucción del torque a partir de las señales proporcionadas. Este algoritmo se encargará de realizar una serie de cálculos y transformaciones matemáticas para representar gráficamente la información recopilada. El proceso de generación de imágenes permitirá una mejor comprensión y análisis de la operación, así como crear una base de datos que será esencial para realizar la identificación paramétrica del robot cartesiano.

Algorithm 1 Algoritmo para generar imágnes a partir de las señales del robot cartesiano.

Input: Posición q, torque τ, número de iteraciones N.

$\dot{q} \leftarrow \mathbf{dct2dev}(q, h, c_f)$
$\ddot{q} \leftarrow \mathbf{dct2dev}(\dot{q}, h, c_f)$
$\{q_s, \dot{q}_s, \ddot{q}_s, \tau_s\} \leftarrow \mathbf{dct2dev}(q, \dot{q}, \ddot{q}, \tau, 100)$
for $k = 1 : N$ **do**
  $\boldsymbol{\beta} \leftarrow \mathbf{rand}(1, 14)$
  $\tau_r \leftarrow \mathbf{Dynmod}(\ddot{q}, \dot{q}, \boldsymbol{\beta})$
  $\psi \leftarrow 2 * \tau - \tau_r$
  $\alpha \leftarrow \tau$
  $A \leftarrow \psi * \alpha$
  $A_{p,n} \leftarrow \mathbf{norm}(A)$
  $L_{p,n} \leftarrow \beta_r - \beta$
end for

Output: Imágenes $A_{p,n}$ con etiquetas $L_{p,n}$.

El algoritmo 1 describe un proceso que utiliza la señal de posición proveniente de un encoder para aproximar la velocidad y aceleración de un robot. Para redimensionar las señales a una longitud deseada, se emplea la transformada discreta del coseno. Luego, se inicia un ciclo for en el cual se generan conjuntos de datos aleatorios. Estos conjuntos se utilizan para reconstruir una señal de torque mediante las señales provenientes y estimadas del robot. Las variables ψ y α se asignan funciones

adecuadas que ayudan a obtener un residuo paramédico durante el entrenamiento de una red neuronal. Este residuo se guarda en una etiqueta para su posterior análisis o referencia.

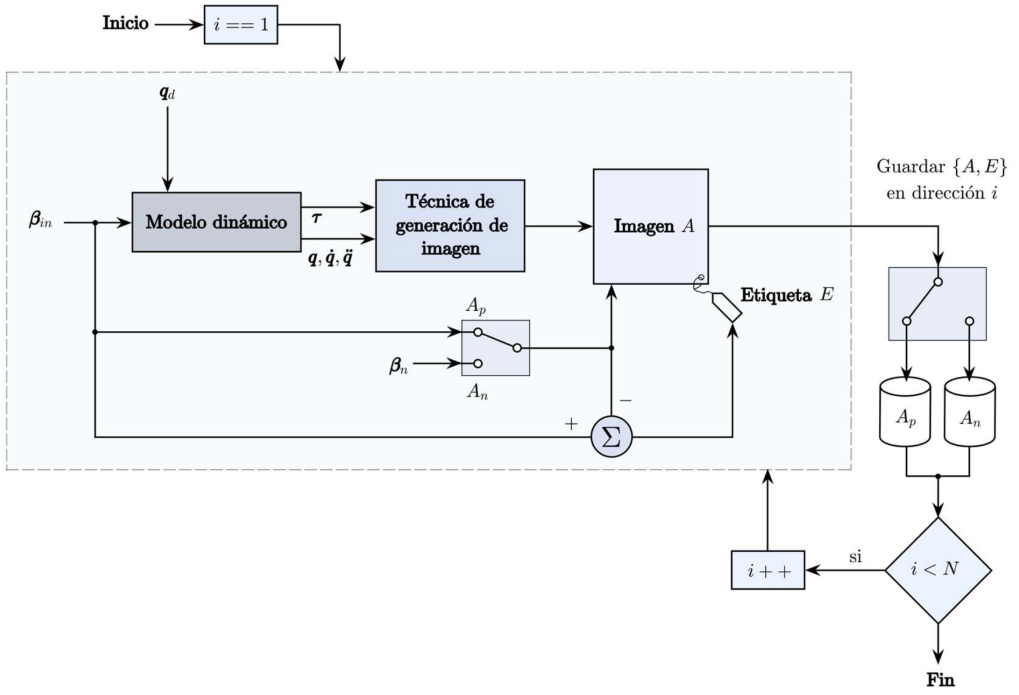


Figura 3.31: Diagrama para la generación del repositorio de imágenes.

3.6. Conclusiones

En este capítulo se ha establecido una estructura de red neuronal eficiente para llevar a cabo la identificación paramétrica del robot, así como se ha definido un algoritmo y técnica para la generación de imágenes a partir de señales cruciales como posición, velocidad, aceleración y torque. La implementación y explicación detallada de cada capa en la red neuronal convolucional respaldan la solidez de la propuesta.

El proceso de entrenamiento de la red mediante el método de Backpropagation, utilizando las imágenes generadas por el robot, fue ejecutado de manera rigurosa. El análisis de los comportamientos durante el entrenamiento reveló resultados alentadores, demostrando la convergencia de las señales de entrenamiento y test hacia un mismo punto a lo largo del tiempo. Este indicador sugirió que la estructura propuesta tenía el potencial de funcionar de manera efectiva en la aplicación deseada. En suma, los resultados obtenidos en este avance de tesis proporcionaron una base sólida para el desarrollo futuro de sistemas robóticos con capacidades de identificación paramétrica y generación de imágenes, abriendo nuevas perspectivas en la mejora de la comprensión y control de robots cartesianos de tres grados de libertad.

Capítulo 4

Identificación paramétrica

La estructura matemática del modelo dinámico de un robot manipulador incluye parámetros como los centros de gravedad, centros de masa, momentos de inercia y coeficientes de fricción. En muchos casos, estos parámetros son desconocidos, especialmente en robots comerciales donde el fabricante no proporciona sus valores nominales. Aunque existen herramientas de la teoría de control, como esquemas adaptables y controladores robustos, que pueden manejar errores en los parámetros dinámicos, tener un conocimiento preciso de estos parámetros es esencial para la mayoría de los enfoques basados en el modelo dinámico de un robot.

La necesidad de abordar la identificación de parámetros ha dado lugar al desarrollo de diversos métodos de identificación que resultan valiosos para estimar los parámetros dinámicos de un robot, especialmente en situaciones en las que medirlos directamente presenta desafíos. No obstante, la complejidad no lineal inherente al modelo dinámico de los robots manipuladores hace que la tarea de identificación paramétrica sea un desafío significativo [46].

Es de particular relevancia para los esquemas de identificación paramétrica y de control adaptable expresar el modelo dinámico no lineal del robot manipulador como el producto de una matriz de regresión compuesta de funciones no lineales (dependientes de la posición, velocidad y aceleración) y un vector de parámetros costantes dependientes de masas, momentos de inercia, distancias a centros de masa, coeficientes de fricción.

Uno de los enfoques de identificación paramétrica ampliamente adoptados en la comunidad científica es el método de mínimos cuadrados. Este método se enmarca en la filosofía de identificación híbrida, donde el modelo de regresión se establece en un contexto de tiempo continuo, mientras que la identificación se realiza mediante un estimador recursivo denominado mínimos cuadrados. Este método es ampliamente reconocido en la literatura debido a su simplicidad y su propiedad de recursividad, lo que lo convierte en una opción atractiva para su aplicación en diversas aplicaciones.

4.1. Técnicas de identificación paramétrica

La identificación paramédica en el contexto de robots manipuladores se refiere al proceso de determinar los parámetros físicos y dinámicos de un robot, lo que es

esencial para su control y operación eficiente. Este proceso es fundamental para garantizar un funcionamiento seguro y preciso del robot en aplicaciones que requieren manipulación de objetos, como en la industria manufacturera, la atención médica y la automatización de tareas domésticas.

La identificación paramédica implica la obtención de parámetros tales como las masas, las longitudes de los eslabones, los coeficientes de fricción, los momentos de inercia y otros valores que describen la dinámica del robot. Estos parámetros son necesarios para modelar matemáticamente el comportamiento del robot, lo que permite diseñar algoritmos de control eficientes.

Mínimos cuadrados

El método de mínimos cuadrados se erige como una técnica fundamental en la estimación de parámetros. Su premisa radica en la minimización de la suma de los cuadrados de las discrepancias entre los valores observados y los predichos. Esta metodología es aplicable a una amplia gama de contextos de estudio en diversas disciplinas científicas. Su aplicación resulta especialmente sencilla cuando el modelo exhibe linealidad en los parámetros. A continuación, consideremos el siguiente modelo de regresión [47].

$$\hat{y}(k) = \varphi(k)^T \theta \tag{4.1}$$

donde $\hat{y}(k)$ representa la variable de salida calculada, $\varphi(k)$ denota un vector $p \times 1$ del sistema compuesto por elementos completamente definidos, y θ es el vector $p \times 1$ de parámetros desconocidos. La indexación del modelo por la variable k, que varía en N y representa el tiempo de muestreo, es fundamental. El objetivo primordial consiste en determinar los parámetros de manera que la salida del modelo, representada por la ecuación 4.1, se aproxime óptimamente a la variable medida $y(k)$, conforme al criterio de mínimos cuadrados.

Variables instrumentales

El método de variables instrumentales se emplea cuando no se pretende modelar las perturbaciones. Su estructura es análoga a la de mínimos cuadrados, pero sustituyendo el vector $\Phi^T(N)$ por otro $\tilde{\Phi}^T(N)$ que no guarda correlación con $\epsilon(k)$. En su forma recursiva, suele iniciarse con los parámetros estimados por mínimos cuadrados, ya que una selección inapropiada de los parámetros iniciales puede resultar en una identificación defectuosa. Este enfoque tiende a arrojar mejores resultados que el algoritmo de mínimos cuadrados cuando hay alguna forma de correlación entre el ruido y la salida del sistema [8].

4.2. Identificación paramétrica en base a inteligencia artificial

En la figura 4.1 es posible observar el diagrama general de la metodología propuesta de identificación paramétrica para el caso de estudio del robot tipo CNC de tres grados de libertad. Para que el algoritmo de identificación funcione, el robot cartesiano a identificar debe seguir una trayectoria predefinida deseada q_d.

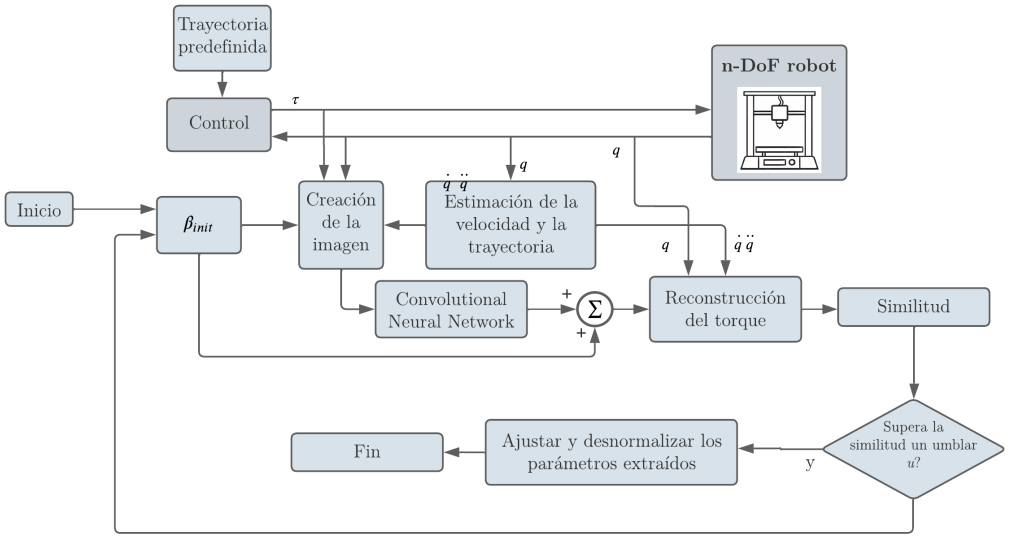


Figura 4.1: Diagrama general de la metodología propuesta de identificación paramétrica.

Una vez seguida la trayectoria definida con un error de posición mínimo, el siguiente paso requerido para la identificación, es estimar la velocidad y aceleración con las muestras de la señal de posición. Se utiliza un filtro pasa bajas basado en la transformada discreta del coseno estudiada en [45] que ayuda a reducir el error numérico debido a la resolución de los encoders en el robot cartesiano tipo CNC. Con las señales de posición $\mathbf{q}$, velocidad $\dot{\mathbf{q}}$, aceleración $\ddot{\mathbf{q}}$ y torque $\boldsymbol{\tau}$, el algoritmo de identificación inicia con la inicialización alatoria de un conjunto de parámetros dinámicos β_{init} que son utilizados para generar una imagen con las señales obtenidas y estimadas del robot.

La idea principal de esta imagen es que cuando los parámetros con la que se construye corresponden con el torque, posición, velocidad y aceleración, así esta imagen contendrá de manera aproximada el cuadrado de la señal de torque del robot. En el otro caso, la imagen contendrá información completamente distinta y es en este caso donde se utilizará la red neuronal convolucional para hallar los parámetros dinámicos del robot. La red neuronal convolucional analiza las imágenes creadas y devuelve el residuo que existe entre los parámetros que coinciden con las señal es de posición, velocidad, aceleración y torque, y los parámetros β_{init}. La CNN se diseñara de manera que la última capa, sea la de salida, utilice una función sigmoide, por lo tanto su salida está limitada de 0 a 1. Sin embargo, el residuo paramétrico tendrá

valores positivos y negativos. Para poder recuperar este residuio, se utiliza la función de activación inversa, dicha recuperación se realizará sumando los parémtros β_{init} al residuo paramétrico que entregará la red neuronal convolucional.

Una vez extraídos los parámetros dinámicos, estos se utilizan para reconstruir el torque utilizando el modelo dinámico, donde son utlizadas las señales de posición, velocidad y aceleración. Reconstruida la señal de torque τ_ry el torque de entrada τ, donde se pretende utilizar una métrica de evaluación en tiempo y frecuencia. Si la similitud entre ambas señales superar un umbral, entonces los parámetros dinámicos obtenidos son ajustados a los nivlees de torque de entrada y es aquí donde el algoritmo finaliza. Para una situación contraria, se reinician los parámetros β_{init}. Estos pasos son descritos en la figura 4.1.

4.3. Parámetros a identificar

Los robots manipuladores son examinados mediante un modelo dinámico, el cual se constituye como un conjunto de ecuaciones diferenciales respecto al tiempo t. Este modelo dinámico, representado por la función f, se compone de expresiones matemáticas, pudiendo ser tanto lineales como no lineales, que describen las variables de posición, velocidad y aceleración, junto con una entrada u, y finalmente, un conjunto de constantes o parámetros dinámicos β.

Las técnicas convencionales empleadas en la identificación paramétrica se encuentran restringidas por la disponibilidad limitada de datos provenientes del movimiento del robot, dado que la mayoría de los robots están equipados únicamente con sensores de posición. Esta limitación implica que tales técnicas exigen una captura exhaustiva del movimiento, lo cual incide significativamente en el tiempo requerido para la identificación paramétrica, tornándolo considerablemente elevado. En virtud de las limitaciones inherentes a la capacidad de medición de los robots y del prolongado proceso asociado a la identificación paramétrica conforme a las metodologías convencionales, surge la necesidad imperiosa de desarrollar una nueva metodología capaz de operar eficazmente con datos incompletos y, al mismo tiempo, de identificar los parámetros dinámicos en un intervalo temporal de computación razonable, parar este trabajo de investigación, se realizará la identificación paramétrica de cuatro parámetros pertenecientes al modelo dinámico (2.4) del robot cartesiano de tres grados de libertad, descritos por:

$$\tau_{mi} = J_{mi}\ddot{\theta}_{mi} + k_{mi}(\lambda_i)(\theta_{mi} - G_i\theta_{gi}) + b_{mi}(\dot{\theta}_{mi} - G_i\dot{\theta}_{gi}) \tag{4.2}$$

para

$$i = 1, 2, 3$$

En el marco de la presente investigación, se prevé la obtención de los valores correspondientes a diversos parámetros a través de la implementación de la red neuronal propuesta para esta metodología. A continuación, se detallan cada uno de estos parámetros, acompañados de las unidades pertinentes y una descripción de su significado y funcionalidad en el contexto del estudio [48].

Símbolo	Significado	Unidades
J_{mi}	Inercia del motor	[Kgm^2]
k_{mi}	Rigidez del eje del motor	[Nm/rad]
λ_i	Parámetro de adaptación de rigidez	[m/m]
b_{mi}	Coeficiente de fricción del eje del motor	[Nms/rad]

Tabla 4.1: Parámetros a identificar.

La fricción constituye un fenómeno inherente en la interacción entre dos cuerpos en contacto, caracterizado por la presencia de fuerzas con componentes tangenciales en su región de contacto que resisten el movimiento relativo entre ellos. La gestión de la fricción adquiere relevancia significativa en el ámbito del control de sistemas mecánicos, dado que su influencia puede generar desviaciones en el seguimiento de trayectorias, la aparición de ciclos límite, vibraciones y otras complicaciones que inciden directamente en el rendimiento del control de movimiento. Para abordar eficazmente los efectos de la fricción, resulta crucial contar con modelos dinámicos precisos que describan este fenómeno de manera efectiva.

Uno de los modelos más comúnmente empleados y objeto de estudio es el denominado modelo de fricción viscosa y de Coulomb. Este modelo, aunque simple en su formulación, captura dos componentes fundamentales presentes en otros modelos de fricción. Por un lado, contempla la fricción viscosa, la cual es directamente proporcional a la velocidad de desplazamiento. Por otro lado, incluye un término discontinuo, caracterizado por ser constante y depender del signo de la velocidad, que refleja la fricción de Coulomb [49], se representa mediante la ecuación (4.3) y su comportamiento es apreciable en la Figura 4.2.

$$F(\dot{\theta}) = f_v\dot{\theta} + f_c \tanh(\lambda, \dot{\theta}) \tag{4.3}$$

donde f_v denota el coeficiente de fricción viscosa, f_c representa el coeficiente de fricción de Coulomb, λ ayuda a acondicionar el comportamiento de la función tanh, y $\dot{\theta}$ describe la velocidad del sistema.

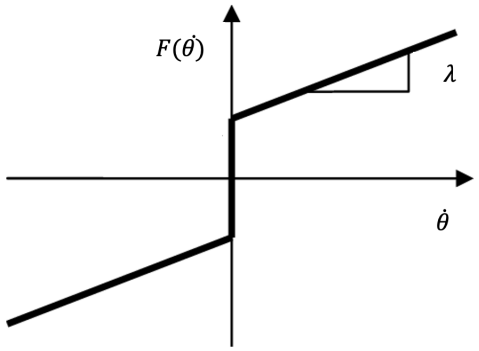


Figura 4.2: Modelo de fricción viscosa y Coulomb.

4.4. Conclusiones

En este capítulo, se ha explorado la importancia de la identificación paramétrica en el contexto del modelado dinámico de robots manipuladores. Dichos modelos encapsulan una serie de parámetros físicos que describen su comportamiento, incluyendo longitud de eslabones, posición del centro de gravedad, masas, momentos de inercia y coeficientes de fricción.

La falta de acceso a los parámetros físicos de los robots comerciales representa un desafío, ya que los fabricantes rara vez proporcionan información detallada sobre la estructura interna de los robots. Ante esta limitación, se recurre a algoritmos de control adaptables y robustos para mitigar las incertidumbres asociadas con estos parámetros.

Por otra parte, se ha destacado la influencia significativa de la fricción en la respuesta de los sistemas mecánicos. Aunque existen numerosos estudios sobre este fenómeno, es importante reconocer la singularidad de cada mecanismo, lo que hace que el estudio de la fricción en un sistema específico sea altamente especializado. La comprensión de los diversos factores que influyen en la fricción resulta crucial para analizar y optimizar el rendimiento de los sistemas mecánicos de interés.

Capítulo 5

Resultados experimentales

En este capítulo se exponen los resultados derivados del caso de estudio abordado. Se inicia con una descripción detallada de las especificaciones del equipo informático empleado en la implementación de los algoritmos. A continuación, se presentan ejemplos representativos de las imágenes generadas mediante las ecuaciones detalladas en el capítulo 3. Se analizan y visualizan las curvas de entrenamiento y validación de las dos redes neuronales empleadas en el estudio. Se exhiben los espectros de frecuencia correspondientes a los filtros convolucionales, así como los pesos sinápticos asociados a las capas completamente conectadas. Para la validación del algoritmo, se recurre a datos simulados basados en el modelo dinámico del robot objeto de estudio. Se ofrece una breve descripción del robot experimental, seguida de la presentación de los resultados obtenidos en la identificación paramétrica experimental del caso de estudio.

5.1. Generación del repositorio de imágenes

En la Figura 5.1, se presentan imágenes derivadas de las señales del robot, evidenciando detalladamente las características intrínsecas de su comportamiento dinámico. En particular, se distingue una representación visual de tipo A_n, donde los parámetros dinámicos del sistema exhiben una notoria concordancia con las señales de posición, velocidad, aceleración y torque. Este análisis gráfico permite una inspección minuciosa de la relación entre las variables cinemáticas y dinámicas, proporcionando una perspectiva visual esclarecedora sobre el desempeño del robot en términos de su movimiento y las fuerzas ejercidas.

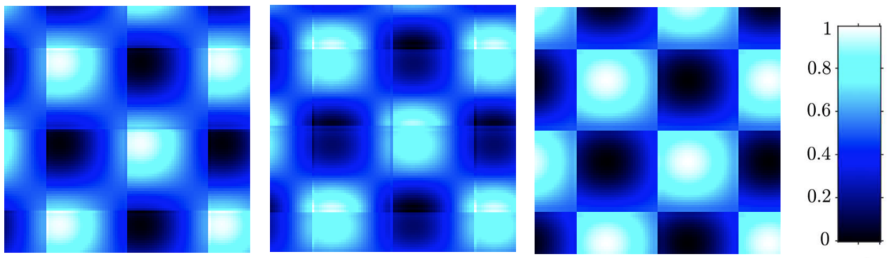


Figura 5.1: Imágenes tipo A_n generadas a partir de las señales del robot.

A continuación se presenta una tabla con las propiedades de una imagen tipo An, incluyendo parámetros dinámicos generados pseudoaleatoriamente para su implementación en el modelo dinámico, y las funciones responsables de generar la imagen para la red neuronal. También se muestra una gráfica en tiempo de las dos funciones implementadas y el espacio utilizado en la memoria del sistema, con un total de 1,000,000 imágenes ocupando 15 GB.

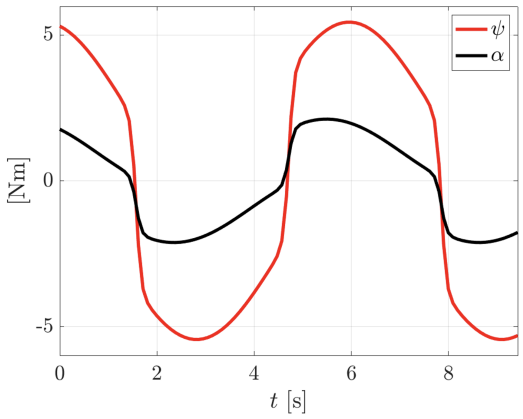
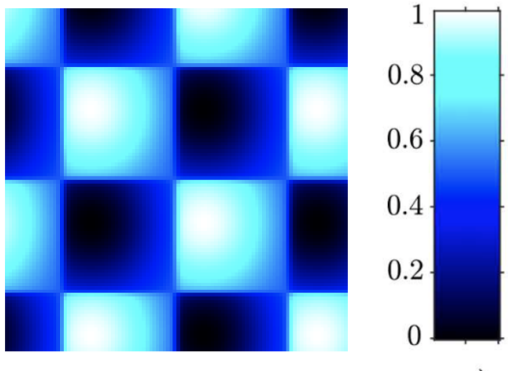
<table>
<tr><th colspan="2">Imagen tipo A_n</th></tr>
<tr><th>Parámetros dinámicos</th><th>Funciones generadoras</th></tr>
<tr><td>$J_m = 0.6704Kg * m^2$
$k_m = 0.7356Nm/rad$
$b_m = 0.8196Nms/rad$
$\lambda = 25.7926m/m$</td><td>$\psi_x = 2\tau_x - \hat{J}_{mx}\ddot{\theta}_{mx} + \hat{k}_{mx}(\hat{\lambda}_x)(\theta_{mx} - G_1\theta_{gx}) +$
$\hat{b}_{mx}(\dot{\theta}_{mx} - G_1\dot{\theta_{gx}})$
$\alpha_x = \tau_x$
$Z = \psi\alpha^T$</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>Espacio utilizado en memoria: 15 Gb</td><td>Indice: 1730 de 1000000</td></tr>
</table>

Tabla 5.1: Propiedades de una imágenes tipo A_n

En la Figura 5.2, se visualiza una representación A_p que utiliza las mismas señales que la Figura 5.1; no obstante, en este escenario, se advierte que los parámetros dinámicos exhiben discrepancias con las señales del robot. La comparación entre las imágenes Ap y An revela notables diferencias, aunque algunas características compartidas persisten.

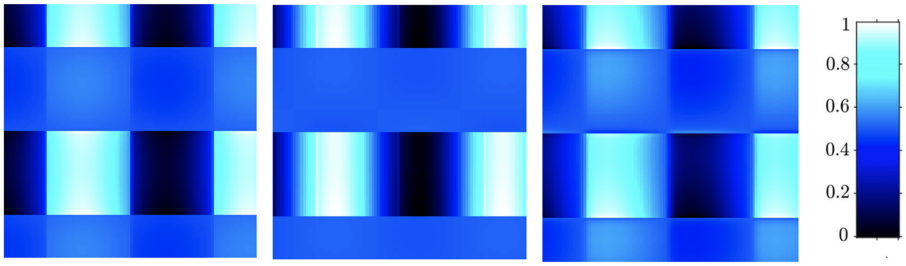


Figura 5.2: Imágenes tipo A_p generadas a partir de las señales del robot.

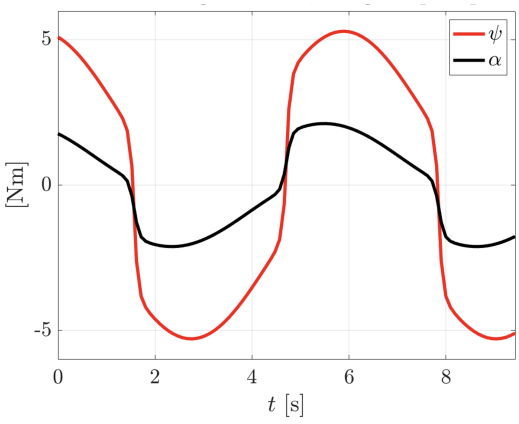
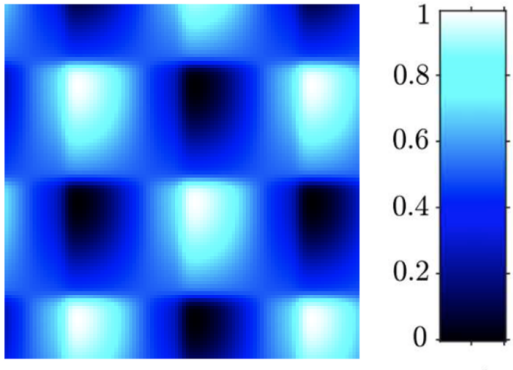
<table>
<tr><th colspan="2">Imagen tipo A_p</th></tr>
<tr><th>Parámetros dinámicos</th><th>Funciones generadoras</th></tr>
<tr><td>$J_m = 0.8285Kg * m^2$
$k_m = 0.8134Nm/rad$
$b_m = 0.9551Nms/rad$
$\lambda = 9.8598m/m$</td><td>$\psi_x = 2\tau_x - \hat{J}_{mx}\ddot{\theta}_{mx} + \hat{k}_{mx}(\hat{\lambda}_x)(\theta_{mx} - G_1\theta_{gx}) +$
$\hat{b}_{mx}(\dot{\theta}_{mx} - G_1\dot{\theta_{gx}})$
$\alpha_x = \tau_x$
$Z = \psi\alpha^T$</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>Espacio utilizado en memoria: 15 Gb</td><td>Indice: 1730 de 1000000</td></tr>
</table>

Tabla 5.2: Propiedades de una imágenes tipo A_n

En la Figura **??**, se presenta el valor absoluto de la resta entre las imágenes Ap y An. Estas disparidades constituyen la información que la red neuronal convolucional asimila, sirviendo como base para la determinación de los residuos paramétricos. Este proceso ilustra cómo la red aprende a discernir y cuantificar las variaciones entre las representaciones generadas y las señales fundamentales, contribuyendo así a la mejora de la precisión en la modelización dinámica del sistema.

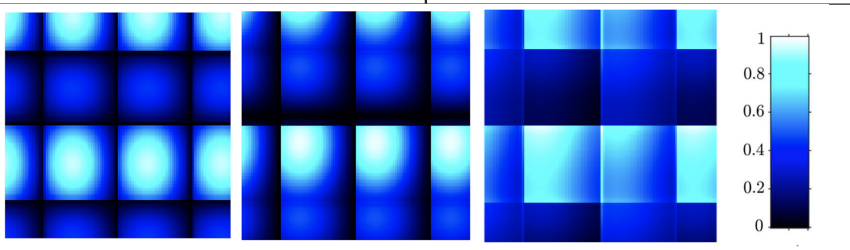
Residuo de imágenes tipo A_p y A_p	
Parámetros dinámicos	**Funciones generadoras**
Imagen A_n $J_m = 0.8285Kg * m^2$ $k_m = 0.8134Nm/rad$ $b_m = 0.9551Nms/rad$ $\lambda = 9.8598m/m$ Imagen A_p $J_m = 0.6704Kg * m^2$ $k_m = 0.7356Nm/rad$ $b_m = 0.8196Nms/rad$ $\lambda = 25.7926m/m$	$\psi_x = 2\tau_x - \hat{J}_{mx}\ddot{\theta}_{mx} +$ $\hat{k}_{mx}(\hat{\lambda}_x)(\theta_{mx} - G_1\theta_{gx}) +$ $\hat{b}_{mx}(\dot{\theta}_{mx} - G_1\dot{\theta_{gx}})$ $\alpha_x = \tau_x$ $Z = \psi\alpha^T$
	
Espacio utilizado en memoria: 15 Gb	**Indice:** 1730 de 1000000

Tabla 5.3: Propiedades de una imágenes tipo $A_n - A_p$.

5.2. Resultados del entrenamiento

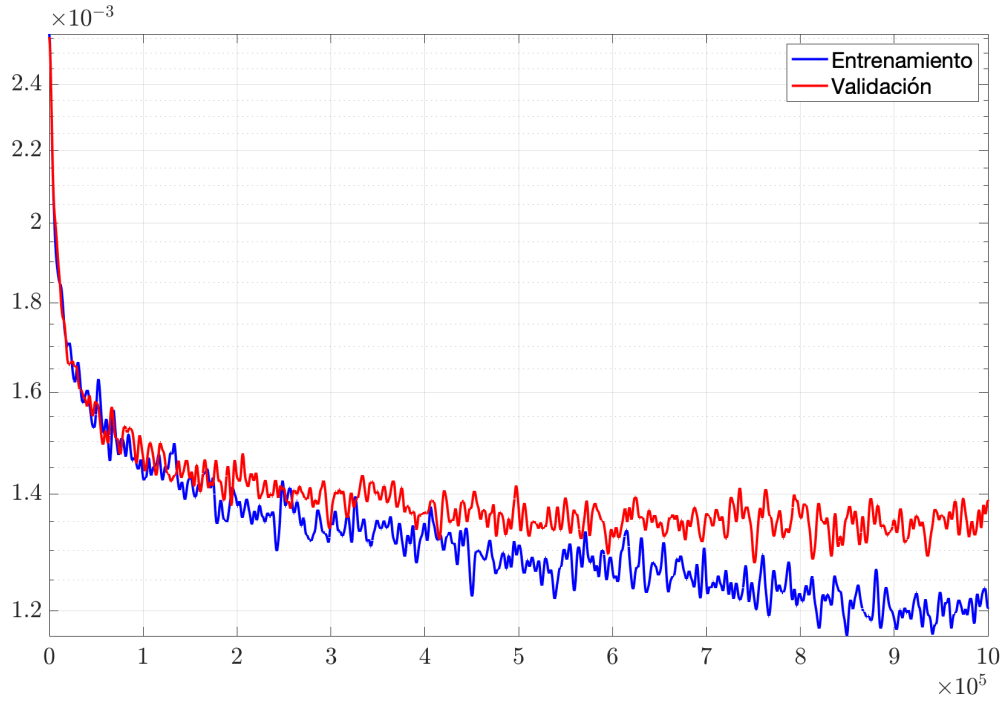


Figura 5.3: Curva de entrenamiento para la red neuronal convolucional (3.20).

Las red neuronal convolucional para la identificación paramétrica del robot cartesiano ha sido entrenada con el gradiente descendente del error. Se ha utilizado el algoritmo de propagación hacía atrás del error para ajustar los pesos sinápticos de la red propuesta en 3.20. Se ha utilizado la función de costo de la ecuación (3.3) para la red neuronal convolucional. En la figura 5.3 se observa la evolución de la función de costo del error para el entrenamiento de la red neuronal convolucional para el robot cartesiano. Se han realizado 500,000 iteraciones de entrenamiento. La gráfica muestra que la función de costo del error para el conjunto de entrenamiento está por debajo de la función de costo para el conjunto de prueba. Esto es debido a que las redes neuronales convolucionales generalizan el conocimiento con las que han sido entrenadas dando como consecuencia una ligera diferencia en las curvas de entrenamiento [36].

Tiempos de ejecución				
Equipo	Cantidad	Operación	Tiempo/Iteración	Tiempo total
Personal	Generar imágenes	1,000,000	15 min.	≈28.5 años
	Entrenamiento	1,000,000	-	= 7 hrs y 37 min
Supercómputo	Generar imágenes	1,000,000	-	= 9 hrs y 12 min.
	Entrenamiento	1,000,000	-	-

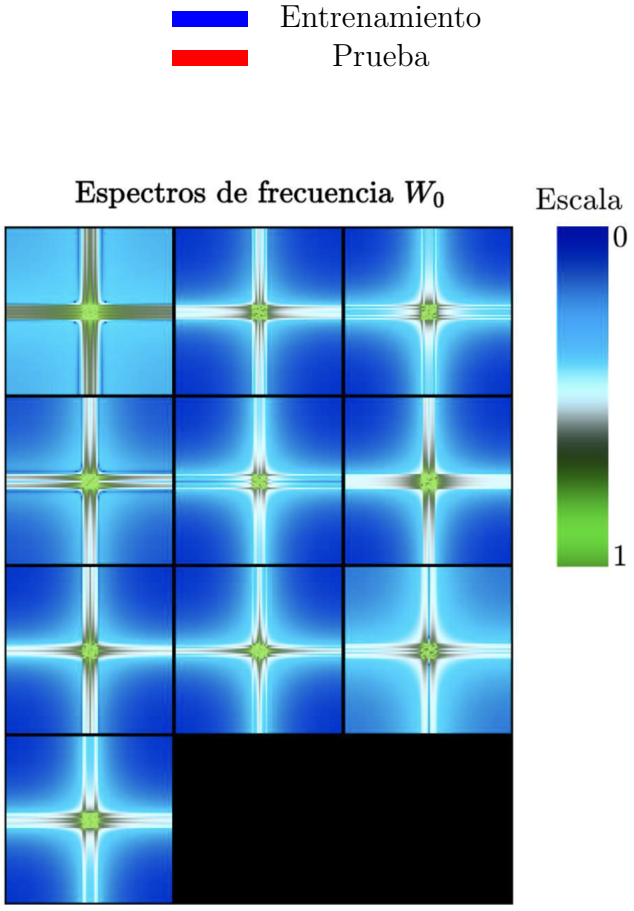


Figura 5.4: Filtros convolucionales de la primera capa convolucional.

En una arquitectura de red neuronal convolucional (CNN), los filtros convolucionales juegan un papel crucial en la extracción de características de las imágenes de entrada. En la primera capa de una CNN, estos filtros suelen presentar una concentración mayor en la magnitud en el centro del filtro. Esta característica se traduce en que los valores de los pesos del filtro son más altos en el centro y disminuyen gradualmente hacia los bordes [36].

Esta disposición de los pesos del filtro permite que la red neuronal convolucional sea más sensible a las características principales o prominentes de la imagen, que a menudo se encuentran en el centro de la ventana de convolución. Esto ayuda a la red a capturar detalles importantes y patrones relevantes en las imágenes de entrada desde el principio del proceso de convolución.

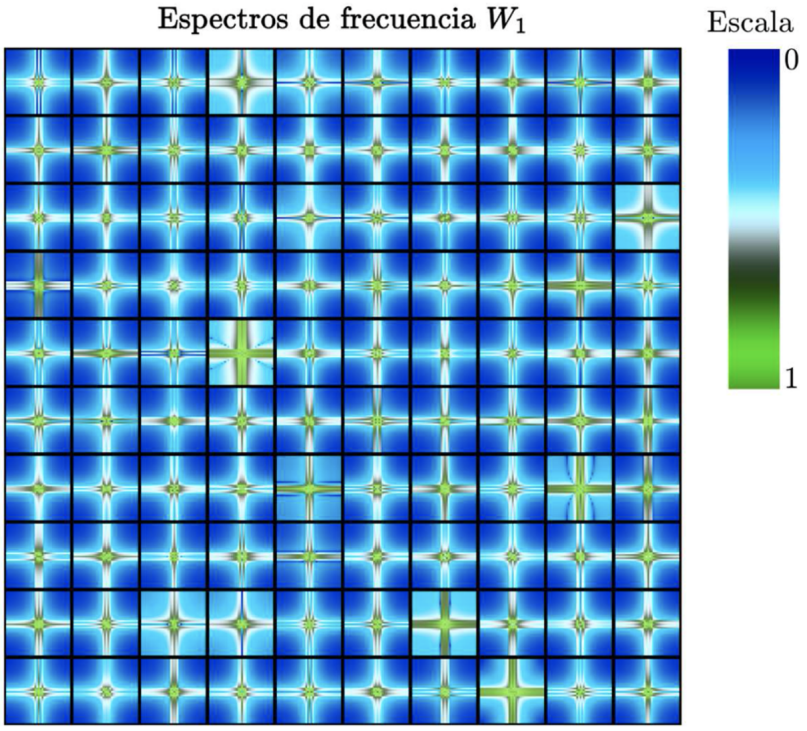


Figura 5.5: Filtros convolucionales de la segunda capa convolucional.

Los filtros convolucionales de la red realizan la extracción de características de la imagen de entrada para el escenario de estudio. Estas características pueden incluir elementos como líneas, bordes, esquinas, sombras y gradientes, según lo señalado por Li et al. (2021). En la Figura 5.5, se presentan los espectros normalizados de frecuencia para la segunda capa convolucional , obtenidos mediante la transformada discreta de Fourier en dos dimensiones según [50], para los filtros de la red convolucional aplicada al robot cartesiano. Es evidente que cada espectro exhibe una forma única, lo cual se atribuye a que cada filtro convolucional extrae una característica específica de la imagen de entrada.

De igual manera, la figura 5.6 presenta los filtros convolucionales para la tercera capa de la red, donde la referencia a la "misma inclinación de los valores de los residuos paramétricos encontrados en las imágenes residuales entre Ap y An" sugiere una relación con el concepto de aprendizaje residual en redes neuronales. Las redes neuronales convolucionales (CNN) introducen conexiones de salto (skip connections) para facilitar el flujo de información a través de las capas de la red. Estas conexiones permiten que las redes aprendan las diferencias (residuos) entre las características extraídas en las capas intermedias y las características deseadas 3. La mención de la inclinación de los valores de los residuos paramétricos podría indicar una similitud en la disposición de los pesos del filtro con los residuos paramétricos encontrados en las conexiones residuales, lo que podría contribuir a una mejor integración de la información a lo largo de la red [41].

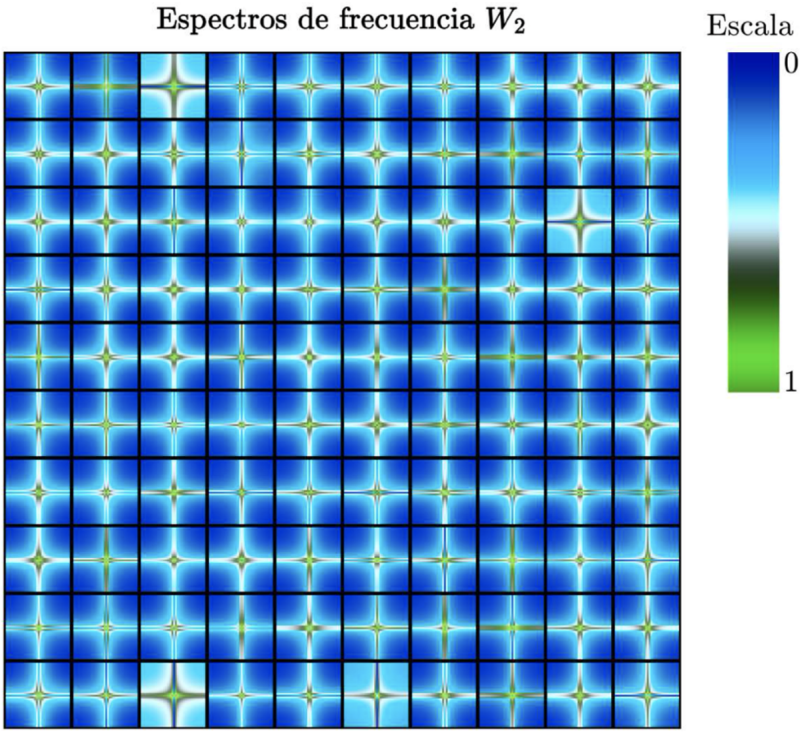


Figura 5.6: Filtros convolucionales de la segunda capa convolucional.

Los pesos sinápticos de una red neuronal convolucional desempeñan un papel fundamental en el procesamiento de las características extraídas por los filtros convolucionales, y su patrón de distribución puede recordar al comportamiento de las señales en el cerebro humano. Estos pesos son esenciales para la transformación de la información de entrada en representaciones significativas que permiten a la red realizar tareas específicas, como la clasificación de imágenes o la detección de objetos.

La figura 5.7 ilustra los datos de entrada (3610) y las neuronas de salida (100) correspondientes a la cuarta capa de la red neuronal convolucional representada en la figura 3.20. En esta capa, los pesos sinápticos procesan la información para generar un residuo paramétrico que captura las características más relevantes de la entrada. Esta etapa es crucial para la extracción de características discriminativas que permiten a la red distinguir entre diferentes clases de objetos o patrones en los datos de entrada.

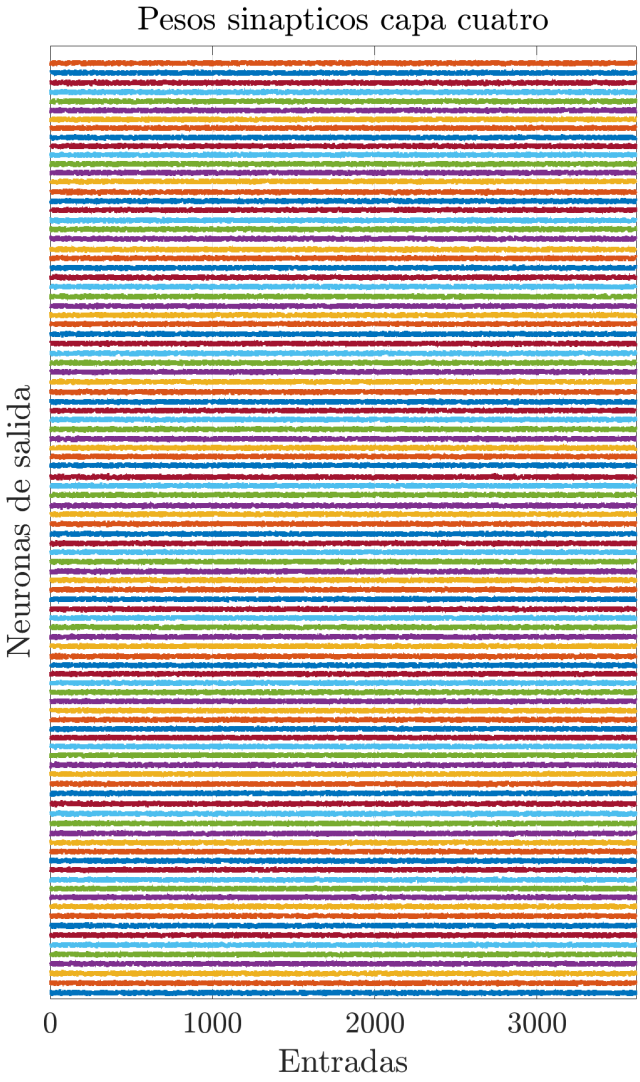


Figura 5.7: Pesos sinápticos de la cuarta capa convolucional.

Al avanzar en la red, en la imagen 5.8 se observa cómo los datos de salida de la cuarta capa se convierten en los datos de entrada de la quinta capa. Esta transición refleja la naturaleza secuencial del procesamiento en una CNN, donde las características extraídas en capas anteriores se utilizan como entrada para capas posteriores, permitiendo una representación cada vez más abstracta y compleja de la información.

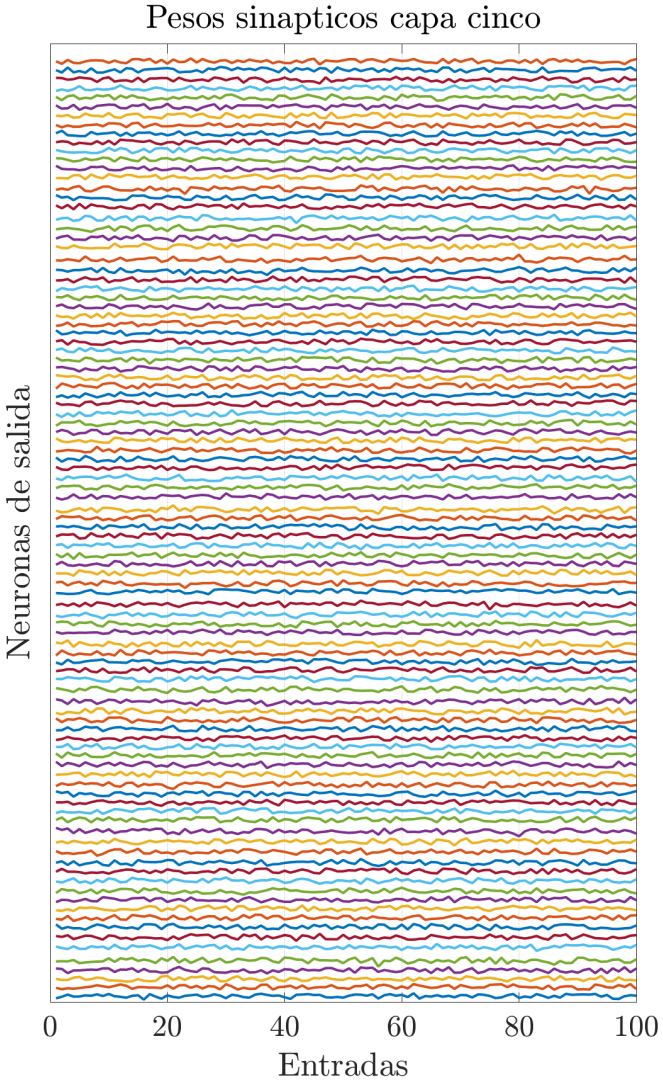


Figura 5.8: Pesos sinápticos de la quinta capa convolucional.

Finalmente, en la capa 6 (imagen 5.9), los datos de entrada son los generados por la capa anterior, mientras que las neuronas de salida representan los parámetros que la red intenta identificar. Esta última etapa del procesamiento refina aún más la información para producir resultados específicos, como la clasificación de objetos en una imagen o la predicción de valores de salida.

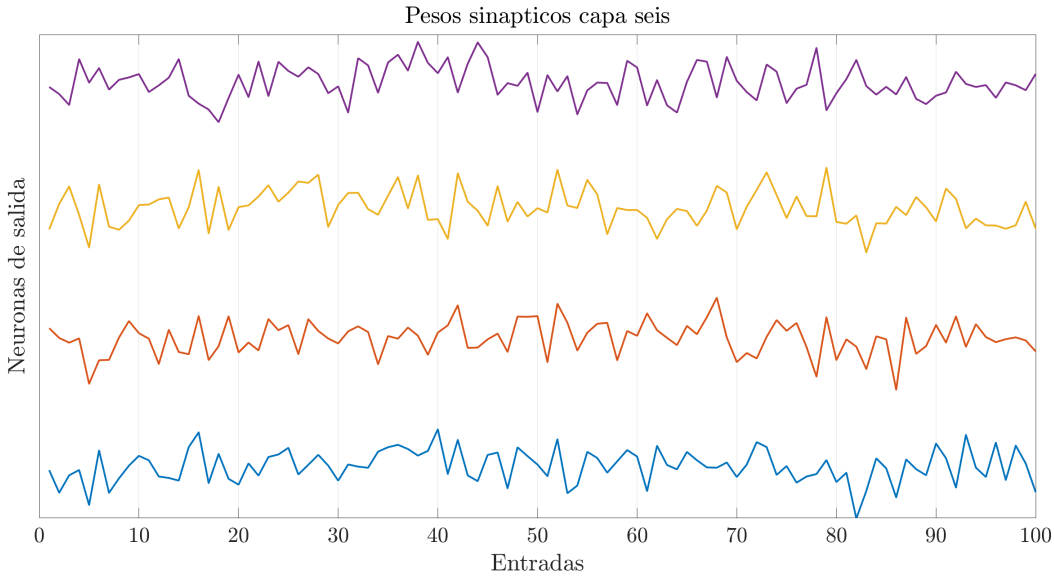


Figura 5.9: Pesos sinápticos de la capa de salida.

La similitud entre los pesos sinápticos registrados en una capa de salida de una red neuronal convolucional y las señales obtenidas en un encefalograma sugiere una interesante analogía entre el procesamiento de información en las redes neuronales artificiales y el funcionamiento del cerebro humano. Esta similitud puede explicarse por la capacidad de las redes neuronales convolucionales para aprender y representar características complejas de los datos de entrada de manera jerárquica y distribuida, de manera similar a cómo el cerebro procesa y codifica la información. Los pesos sinápticos en una red neuronal convolucional actúan como conexiones ponderadas entre las neuronas, determinando cómo la información se propaga a través de la red y se transforma en representaciones más abstractas y discriminativas a medida que avanza el procesamiento. Esta analogía subraya la utilidad de las redes neuronales convolucionales como modelos computacionales para comprender y simular procesos cognitivos complejos, así como la inspiración que el cerebro humano proporciona para el diseño de algoritmos de aprendizaje automático más eficientes y efectivos.

5.3. Resultados de la identificación paramétrica

Para verificar la eficacia del algoritmo de identificación paramétrica, se emplearon datos simulados correspondientes al comportamiento del robot cartesiano. La Figura 5.10 muestra dos señales de torque simuladas, las cuales fueron generadas para el caso de estudio utilizando la trayectoria delineada en las ecuaciones 3.12. Se destaca la proximidad entre los torques reconstruidos mediante los parámetros obtenidos y los torques de entrada simulados.

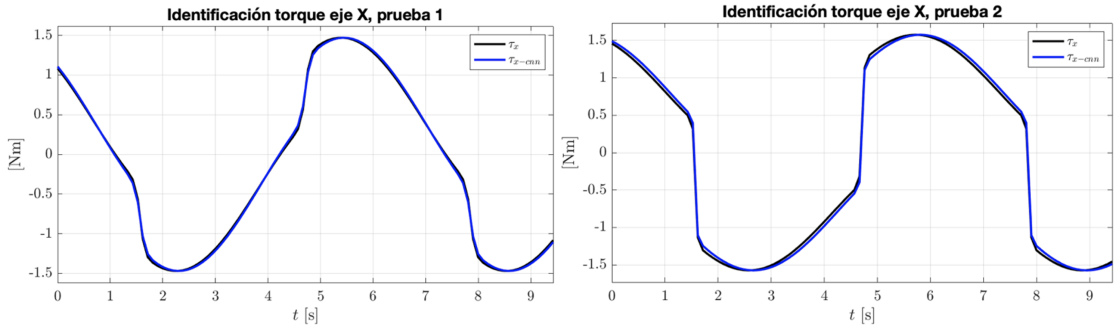


Figura 5.10: Resultados de la simulación de identificación paramétrica primer eje.

En la Tabla 5.2 se presentan los parámetros dinámicos empleados en las simulaciones, los parámetros estimados mediante el algoritmo, así como las métricas de evaluación correspondientes. Para cada una de las seis simulaciones, se observa que la métrica de evaluación supera el 98 %. Esto indica un alto grado de precisión en la identificación paramétrica realizada por el algoritmo, validando su eficacia en la estimación de los parámetros dinámicos del robot cartesiano.

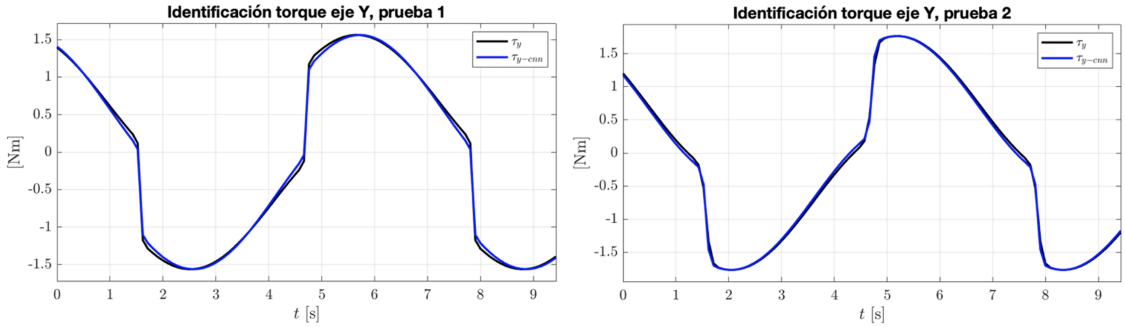


Figura 5.11: Resultados de la simulación de identificación paramétrica segundo eje.

Estos resultados demuestran la capacidad del algoritmo de identificación paramétrica para capturar con precisión el comportamiento dinámico del robot cartesiano a partir de datos simulados. La proximidad entre los torques reconstruidos y los torques de entrada simulados, así como las altas métricas de evaluación obtenidas, respaldan la utilidad y fiabilidad del enfoque propuesto en la identificación de los parámetros del sistema.

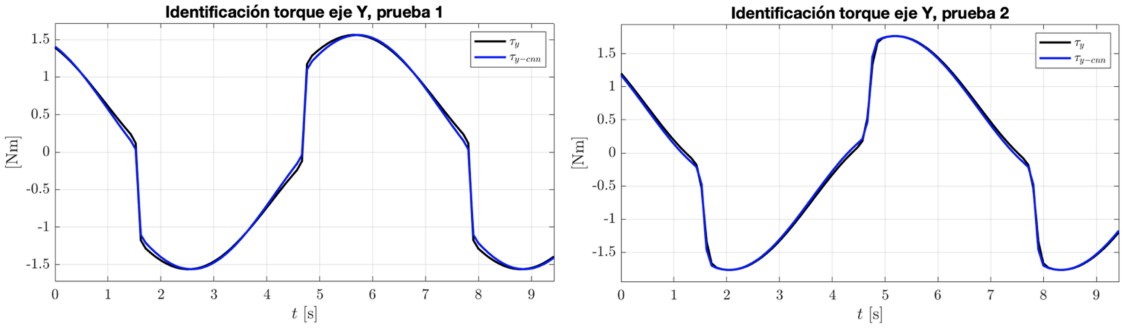


Figura 5.12: Resultados de la simulación de identificación paramétrica segundo eje.

Parameters	S1	$S1_{cnn}$	S2	$S2_{cnn}$
J_{mx} $[Kgm^2]$	0.2215	0.1678	0.0622	0.1192
J_{my} $[Kgm^2]$	0.3490	0.4372	0.449	0.8467
J_{mz} $[Kgm^2]$	0.4658	0.5122	0.0606	0.1318
b_{mx} $[Nms/rad]$	0.3590	0.355	0.4647	0.4732
b_{my} $[Nms/rad]$	0.9001	0.7632	0.3321	0.2466
b_{mz} $[Nms/rad]$	0.1041	0.0951	0.4147	0.4297
k_{mx} $[Nm/rad]$	0.0746	0.1035	0.1109	0.0918
k_{my} $[Nm/rad]$	0.0438	0.1304	0.1701	0.1051
k_{mz} $[Nm/rad]$	0.8034	0.7598	0.9683	0.9376
λ_{mx} $[m/m]$	29.3055	29.1544	5.7120	12.8462
λ_{my} $[m/m]$	29.0571	20.8791	11.4388	11.2913
λ_{mz} $[m/m]$	15.8142	17.7470	27.7764	36.1140

Tabla 5.4: Resultados numéricos de la identificación paramétrica.

Similitude	S1	S2
$\boldsymbol{\tau_x}$	98.17 %	98.74 %
$\boldsymbol{\tau_y}$	98.47 %	98.58 %
$\boldsymbol{\tau_z}$	99.61 %	98.51 %

Tabla 5.5: Parámetros dinámicos de los resultados de simulación.

Tras analizar los resultados, se destaca la notable coincidencia entre las señales de torque reconstruidas y las obtenidas durante las pruebas numéricas. Este hallazgo subraya la capacidad del algoritmo de identificación, respaldado por la arquitectura de la CNN propuesta 3.20, para capturar y modelar de manera efectiva el comportamiento dinámico del robot cartesiano en diversas condiciones de operación. La comparación entre las señales reales y las reconstruidas valida la eficacia del algoritmo propuesto, evidenciando su potencial para aplicaciones en el control y la optimización de sistemas robóticos.

5.4. Resultados experimentales

El algoritmo de identificación paramétrica propuesto se implementó en un entorno experimental mediante la utilización de un robot compuesto por un sistema cartesiano y un brazo robótico. El sistema incluyó distintos motores para los ejes X-Y y el eje Z. Específicamente, para los ejes X-Y se emplearon dos motores equipados con reductoras del modelo Z5D120-12, operados a una tensión de 12V, capaces de alcanzar una velocidad de 3100 rpm y generar un par entre 3 y 4 Nm. Las reductoras asociadas a estos motores corresponden al modelo 5GU9K. Respecto al eje Z, se implementó un motor del modelo Z4D40-12GN-30S, también alimentado a 12V, junto con una reductora del modelo 4GN15K.

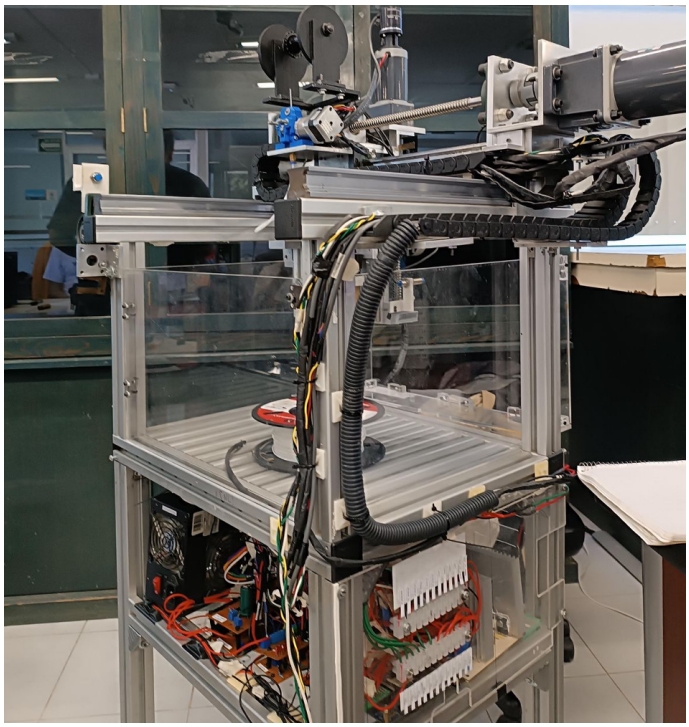


Figura 5.13: Robot cartesiano experimental.

Para controlar el robot cartesiano, se emplearon las tarjetas FPGA Altera DE0-cv y DE0-nano. El diagrama de bloques del sistema embebido utilizado con este propósito se muestra en la Figura 5.12. Para implementar la derivada proporcional (DP) clásica [51]. Se utilizó un módulo para decodificar las señales del codificador y determinar la posición de los motores del robot. El control de los motores de corriente continua se efectuó a través de un puente H de transistores BJT, según lo propuesto por Boylestad [52]. La alimentación del robot se llevó a cabo mediante una fuente de 12 voltios. Los datos experimentales se transmitieron a un ordenador personal (PC) mediante el software Labview, para su posterior análisis en el contexto de la identificación paramétrica. Además, dicho software se encargó de cargar el programa para el microprocesador integrado en la FPGA. La comunicación entre la FPGA y el PC se estableció mediante un módulo wifi.

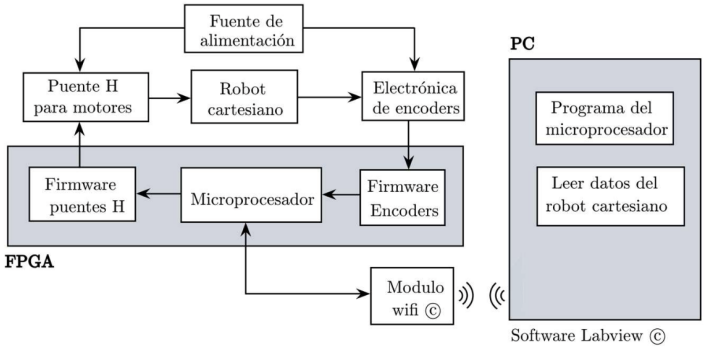


Figura 5.14: Diagrama a bloques del sistema embebido.

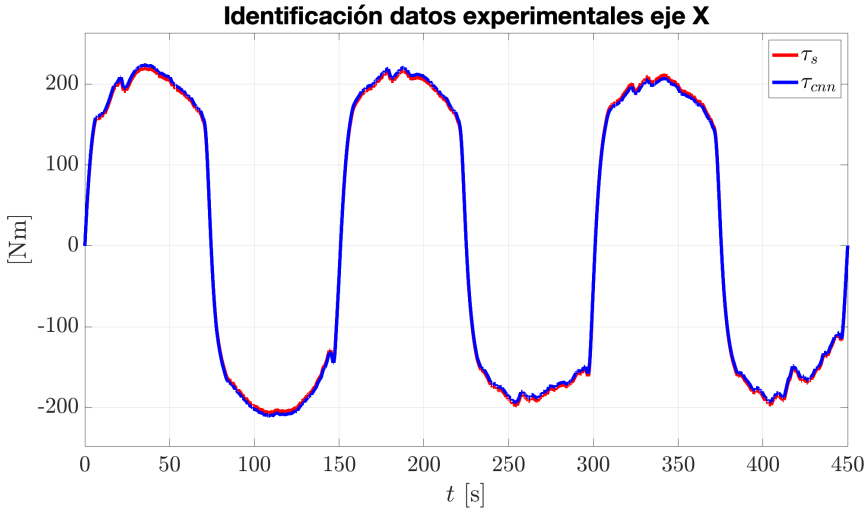
Los resultados muestran que los valores reconstruidos utilizando los parámetros identificados son cercanos a los observados experimentalmente, como se detalla en la Tabla 5.6. Sin embargo, se observa una pequeña discrepancia entre los valores reconstruidos y los observados, lo cual puede atribuirse a varios factores. Uno de ellos es la descripción del modelo dinámico subyacente, que puede no capturar completamente todas las características del sistema en cuestión. La estructura de la red neuronal es crucial para la precisión de la identificación paramétrica. Factores como el diseño de la red, incluyendo el número de capas y neuronas, junto con la función de activación, influyen en su capacidad para modelar la relación entre entradas y salidas del sistema.

Parameters $[\beta]$	X-axis	Y-axis	Z-axis
$\hat{J}_m$ $[Kgm^2]$	0.7046	0.1715	0.0710
$\hat{b}_m$ $[Kgm^2]$	0.1927	0.6423	0.9471
$\hat{k}_m$ $[Kgm^2]$	0.6054	0.8516	0.8674
$\hat{\lambda}_m$ $[Kgm^2]$	15.6895	20.7276	5.1856
metrev	98.13 %	98.92 %	99.67 %

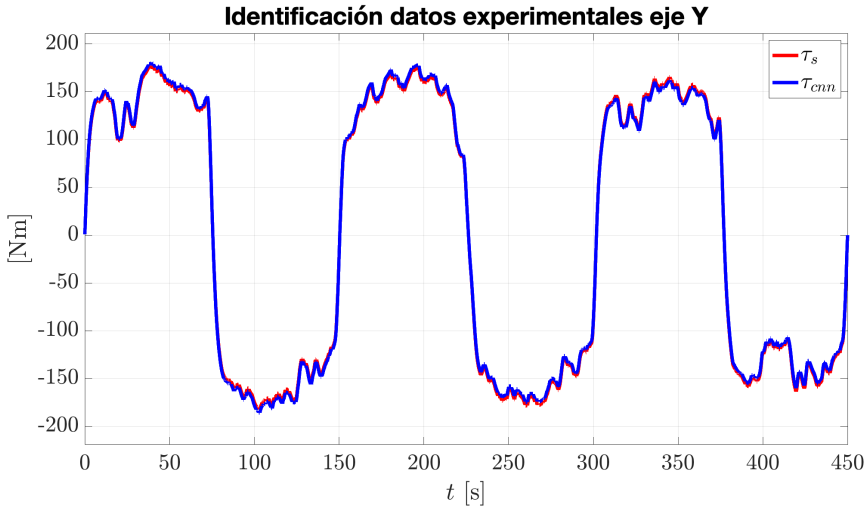
Tabla 5.6: Parámetros dinámicos experimentales de la identificación paramétrica del robot cartesiano.

La tabla 5.6 muestra los valores de similitud con la métrica estudiada en el capítulo 3. La similitud de los torques reconstruidos con el algoritmo 2 tiene un valor mínimo 98.13 % en relación a los datos experimentales del robot. El valor máximo de similitu para el algoritmo es de 99.67 %.

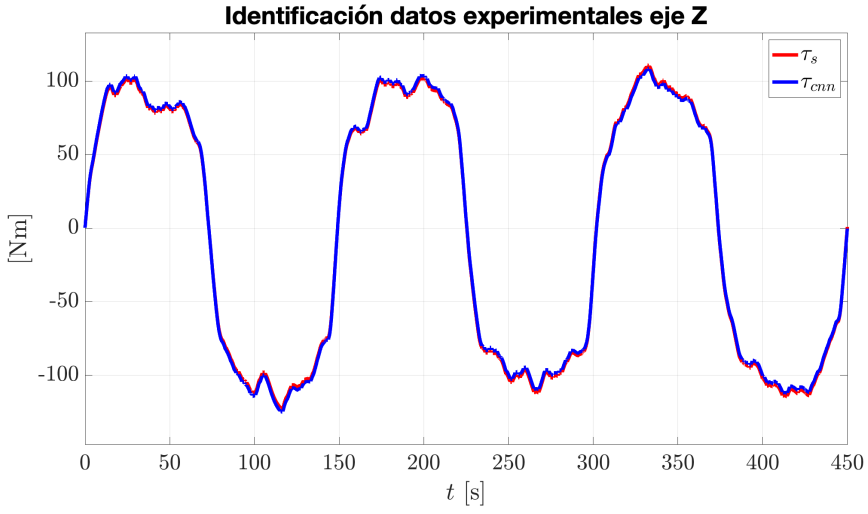
En la figura 5.15 se presentan los resultados derivados del proceso de identificación paramétrica del robot cartesiano de tres grados de libertad, empleando el algoritmo 1. En esta instancia, se ha empleado el torque adquirido experimentalmente como punto de comparación para evaluar la eficacia del algoritmo propuesto. Se representan los torques reconstruidos a partir de los parámetros derivados del algoritmo 1 en tonalidad azul, mientras que en tono rojo se exhibe el torque experimentalmente del robot.



(a)



(b)



(c)

Figura 5.15: Resultados de la identificación paramétrica con valores experimentales.

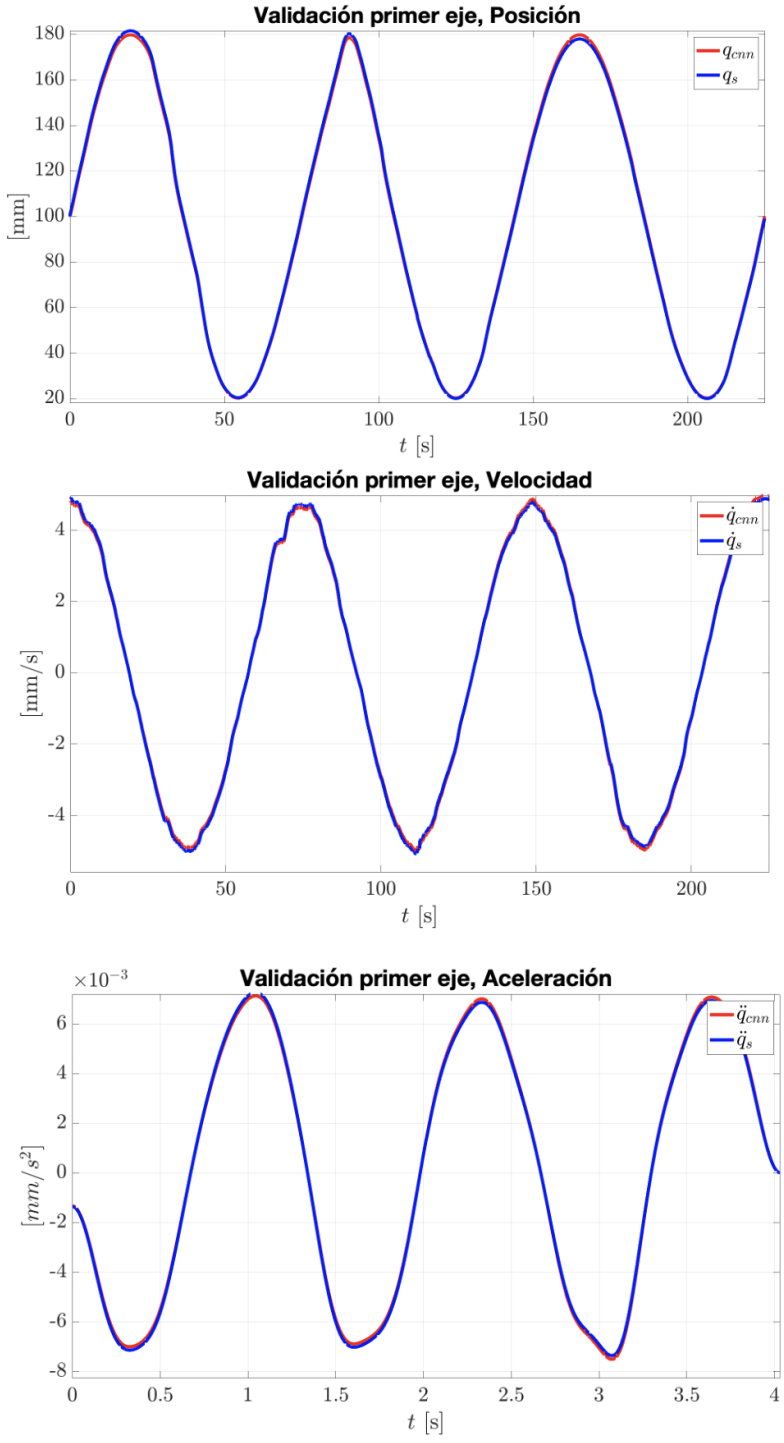


Figura 5.16: Trayectorias de validación para el robot cartesiano, primer eje.

La Figura 5.16 ilustra una trayectoria senoidal utilizada para el robot cartesiano. Las señales correspondientes a posición, velocidad y aceleración representadas en dicha figura, reflejan el desempeño del robot experimental bajo el control proporcional-derivativo (PD) implementado en el sistema embebido, tal como se muestra en la Figura 5.14.

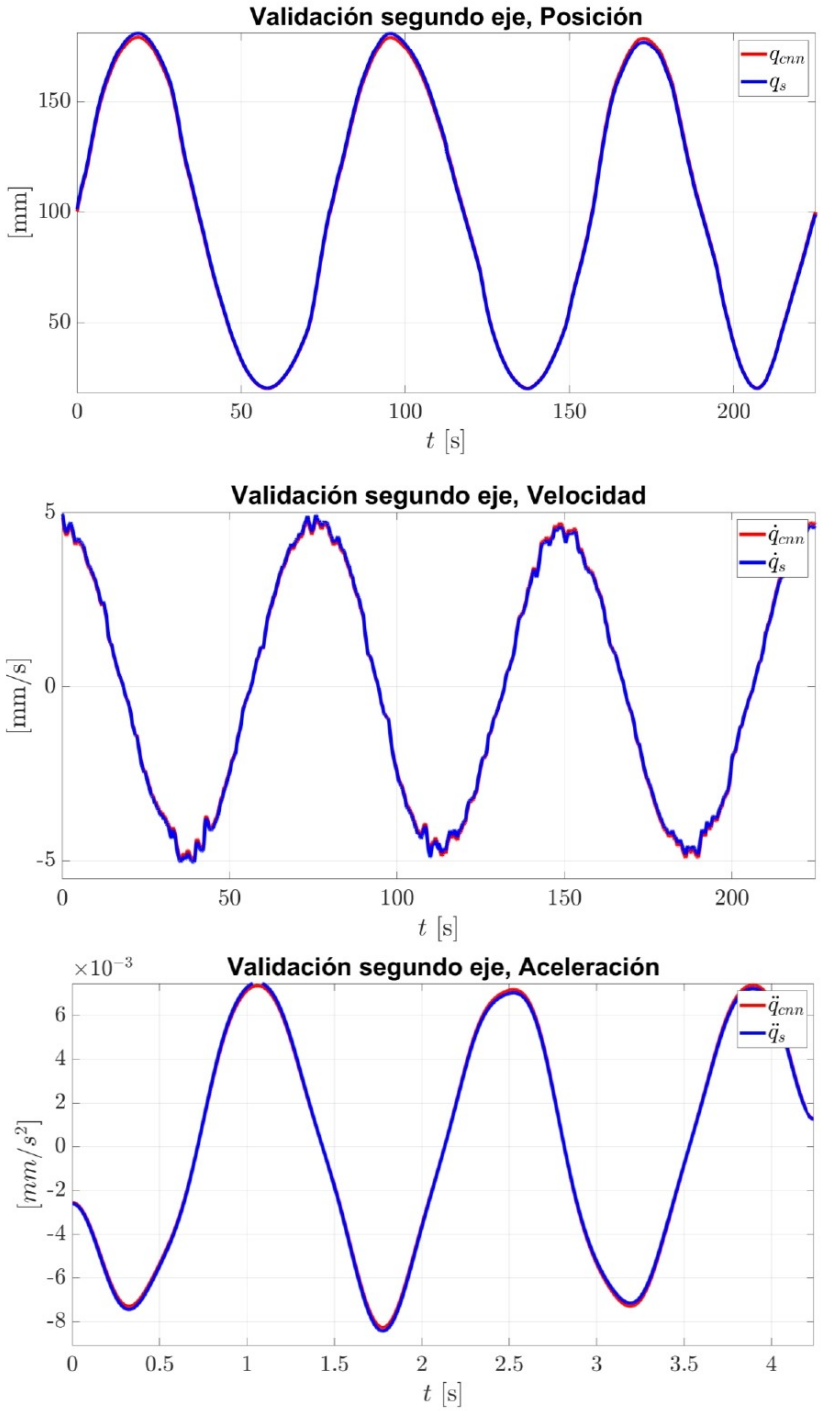


Figura 5.17: Trayectorias de validación para el robot cartesiano, segundo eje.

Además, es importante destacar que las trayectorias experimentadas por los otros dos ejes del robot cartesiano también se presentan en la misma figura. Dichas trayectorias, al igual que la mencionada anteriormente, se implementan como trayectorias senoidales y se evalúan bajo el mismo esquema de control PD.

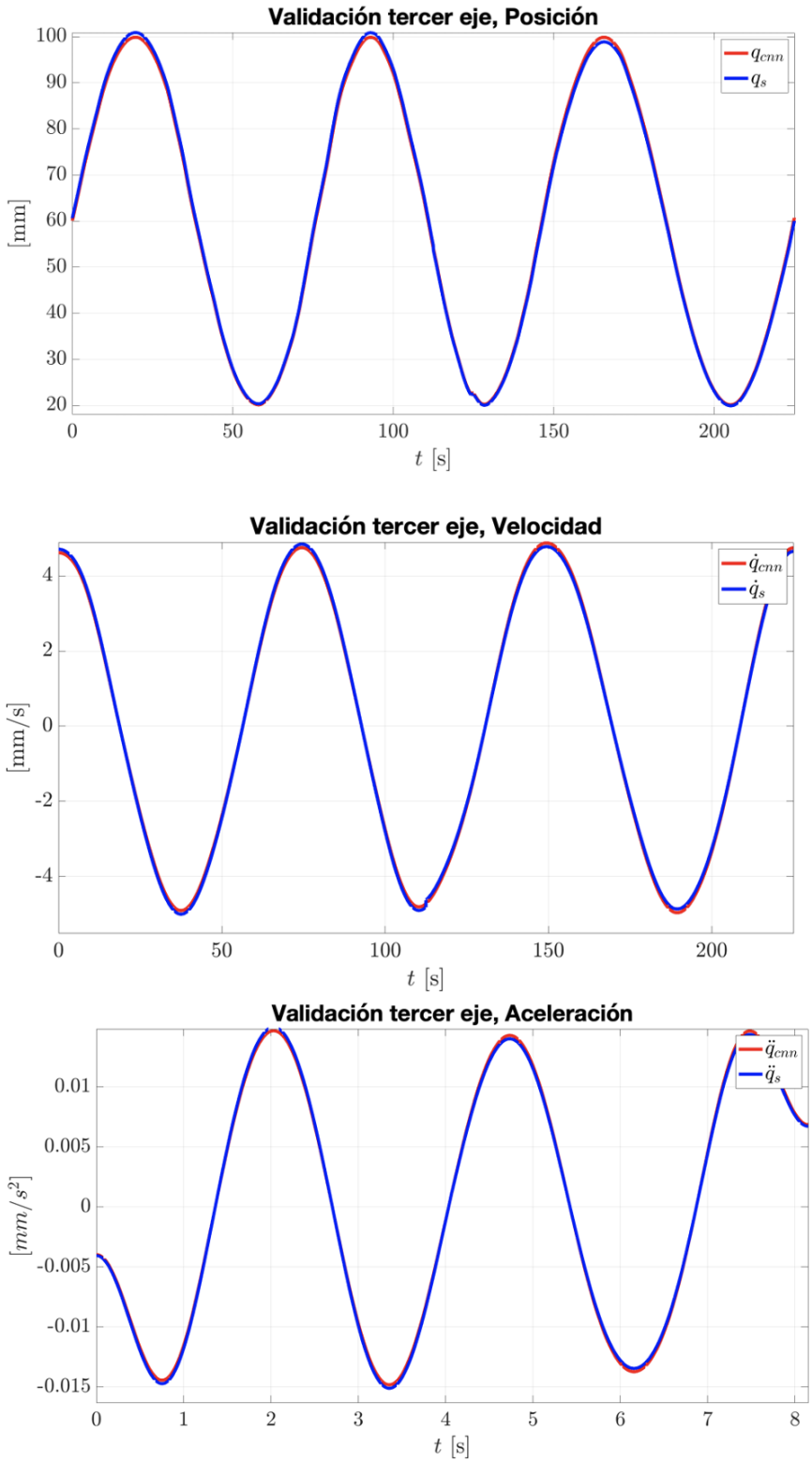


Figura 5.18: Trayectorias de validación para el robot cartesiano, tercer eje.

5.5. Conclusiones

En este capítulo, hemos explorado la aplicación de una red neuronal convolucional para la identificación paramétrica de un robot cartesiano, utilizando simulaciones del modelo dinámico y datos experimentales. Los resultados revelaron una alta similitud entre los torques simulados y reconstruidos, demostrando la eficacia del algoritmo de identificación paramétrica en condiciones controladas. Sin embargo, se observaron disparidades entre los torques reconstruidos y experimentales, atribuibles a las simplificaciones inherentes a los modelos dinámicos utilizados.

Esta discrepancia subraya la necesidad de mejorar los modelos dinámicos de los robots para aumentar la precisión de los algoritmos de identificación paramétrica. A pesar de estas diferencias, se demostró que las redes neuronales convolucionales pueden extraer parámetros dinámicos con una precisión notable, reconstruyendo los torques experimentales con más del 95 % de similitud. Este hallazgo resalta el potencial de las técnicas de aprendizaje automático en la identificación paramétrica de sistemas robóticos, aunque señala la importancia de continuar refinando los modelos subyacentes para una precisión aún mayor en la predicción de comportamientos dinámicos.

Conclusiones generales

En función de los resultados obtenidos en este trabajo de tesis podemos decir que los objetivos planteados se han cumplido.

El contar con un modelo matemático que represente el comportamiento físico del sistema nos permitió realizar simulaciones bajo múltiples condiciones variando los parámetros de la ecuación obtenida.

El modelo matemático o también conocido como modelo dinámico resulto ser no lineal por lo que el tiempo de simulación para un caso dado fue entre 15 y 20 minutos en una PC actual. Por lo que el tiempo para realizar el millón de simulaciones requerido para entrenar la red neuronal se llevaría aproximadamente 28.5 años. Este problema se solucionó pidiendo acceso al Laboratorio Nacional de Super Computo de la BUAP, lo que permitió generar el repositorio o base de datos de un millón de imágenes en un tiempo de aproximadamente 8 horas.

Encontrar una estructura de red neuronal que obtuviera los parámetros del modelo dinámico a partir de los datos experimentales no fue fácil, ya que se estudiaron tres estructuras diferentes: en la primera, la curva de validación no convergía con el entrenamiento, la segunda si parecía converger, pero el error era alrededor de uno y en la tercera propuesta el error que se alcanzo fue del orden de diez a la menos tres. Este error comprende la suma de los errores de los 4 parámetros identificados por la red neuronal.

Con la obtención de los 4 parámetros del modelo dinámico se modificó el control de trayectoria y se compenso en la ecuación de control la fricción de coulomb y viscosa, lo cual produjo una coincidencia del alrededor del 99 % entre los resultados de la simulación y los resultados experimentales.

De la conclusión anterior obtenemos que nuestro modelo dinámico emula el comportamiento real del robot en un 99 %.

De acuerdo a los resultados podemos decir que el uso de la inteligencia artificial en la identificación paramétrica tiene mucho futuro ya que nos permite alcanzar tan precisión en el modelo dinámico que permite conocer el comportamiento real del robot en diferentes situaciones, por lo que trabajos futuros se podría conocer la situación del robot en tiempo real e identificar fallas automáticamente.

Bibliografía

[1] Baron L. & Birglen L. Qin, Z. A new approach to the dynamic parameter identification of robotic manipulators. pages 28(4), 539–547, 2010.

[2] O. Bingul, Z. & Karahan. Dynamic identification of staubli rx-60 robot using pso and ls methods. pages 38(4), 4136– 4149., 2011.

[3] A. Curkovic Hace. Accurate fpga-based velocity measurement with an incremental encoder by a fast generalized divisionless mt-type algorithm. 2018.

[4] R. Celikel. Ann based angle tracking technique for shaft resolver. page 148., 2019.

[5] W. Yang S. Peng Z. Li, F. Liu and J. Zhou. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Trans. Neural Netw. Learn. Syst.*, 2021.

[6] Rosenlund O. S. Brandt D. & Zhang X. Madsen, E. Comprehensive modeling and identification of nonlinear joint dynamics for collaborative industrial robot manipulators. page 101., 2020.

[7] Wu J. Liu C. & Xiong Z. Han, Y. An iterative approach for accurate dynamic model identification of industrial robots. 2020.

[8] P. Gautier, M. & Poignet. Extended kalman filtering and weighted least squares dynamic identification of robot. 2001.

[9] Teng L.-T. Chen C.-Y. & Chang C.-K. Lin, F.-J. Robust rbfn control for linear induction motor drive using fpga. 2009.

[10] Wang L. & ZhenHua X. JianHua, W. Identification of robot dynamic model and joint frictions using a baseplate force sensor. 2022.

[11] Li Y. Zhu M. Xu Z.-& Mu D. Li, Y. A nonlinear momentum observer for sensorless robot collision detection under model uncertainties. 2021.

[12] Chihi I. & Kamavuako E. N. Sidhom, L. Software sensor to enhance online parametric identification for nonlinear closed-loop systems for robotic applications. 2021.

[13] Zhang C. Hu T. Wang T.-Chen Q. & Chen-C. Ni, H. A dynamic parameter identification method of industrial robots considering joint elasticity. 2019.

[14] Iriarte X. Mata V. & Ros J. Diaz-Rodriguez, M. On the experiment design for direct dynamic parameter identification of parallel robots. 2009.

[15] Ganseman C. Tukel D. DeSchutter J. & VanBrussel H. Swevers, J. Optimal robot excitation and identification. 1997.

[16] Janot A. Young P. C. & Carrillo F. Brunot, M. An improved instrumental variable method for industrial robot model identification. 2018.

[17] Verdonck W. & De Schutter J. Swevers, J. Dynamic model identification for industrial robots - integrated experiment design and parameter estimation. 2007.

[18] N. Jin, J. & Gans. Parameter identification for industrial robots with a fast and robust trajectory design approach. 2015.

[19] Li Q. Fang L. Han B. & Zhang H. Liu, G. A new joint friction model for parameter identification and sensor-less hand guiding in industrial robots. 2020.

[20] A. Afrough, M. & Abu Hanieh. Identification of dynamic parameters and friction coefficients: of a robot with planar serial kinemtic linkage. 2019.

[21] Gondokaryono R. Munawar A. & Fischer G. S. Wang, Y. A convex optimization-based dynamic model identification package for the da vinci research kit. 2019.

[22] Mata V. Valera A. & Page A. Diaz-Rodriguez, M. A methodology for dynamic parameters identification of 3-dof parallel robots in terms of relevant parameters. 2010.

[23] Jiang M. Cao Y. Hua D. Wu H. Ding Y.-& Chen B. Jiang, S. A typical dynamic parameter identification method of 6-degree-of-freedom industrial robot. 2017.

[24] Navarro B. Bonnet V. Fraisse P. Crosnier A. & Venture-G. Katsumata, T. Optimal exciting motion for fast robot identification. application to contact painting tasks with estimated external forces. 2019.

[25] M. Briot, S. & Gautier. Global identification of joint drive gains and dynamic parameters of parallel robots. 2015.

[26] J. Urrea, C. & Pascal. Design and validation of a dynamic parameter identification model for industrial manipulator robots. 2021.

[27] A. Janot. A separable instrumental variable method for robot identification. 2021.

[28] Huang Y. Guo X. Zhou S. & Jia L. Chen, J. Parameter identification and adaptive compliant control of rehabilitation exoskeleton based on multiple sensors. 2020.

[29] Wang S. Jing F. & Tan M. Zhang, S. Parameter estimation survey for multi-joint robot dynamic calibration case study. 2019.

[30] Mata V. & Valero F. Benimeli, F. A comparison between direct and indirect dynamic parameter identification methods in industrial robots. 2006.

[31] Pagani R. Beschi M. & Legnani G. Hao, L. Dynamic and friction parameters of an industrial robot: Identification, comparison and repetitiveness analysis. 2021.

[32] Do H. Choi T. Park J. & Cheong J. Jung, D. Robust parameter estimation of robot manipulators using torque separation technique. 2021.

[33] B. Rosyid, A. & El-Khasawneh. Identification of the dynamic parameters of a parallel kinematics mechanism with prismatic joints by considering varying friction. 2020.

[34] M García López. *Control de seguimiento de Trayectoria y contorneado basado en el modelo de fricción para un sistema CNC de feed-drive.* Tesis de Maestría,Benemérita Universidad Autónoma de Puebla, 2022.

[35] Brian C. Fabien. Analytical system dynamics: Modeling and simulation. 2009.

[36] F. Berzal. *Redes Neuronales and Deep Learning.* Independently published,, 1 edition, 2018.

[37] R. C. Gonzales and R. E. Woods. *Digital Image Processing.* Prentice Hall, 2008, 3 edition, 2008.

[38] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 20(3):121–136, 1975.

[39] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.

[40] Kunihiko Fukushima. A neural network for visual pattern recognition. *Computer*, 21(3):65–75, 1988.

[41] Sotelo J. A. L. Bravo, E. F. C. *Una aproximación a las redes neuronales artificiales.* Universidad del Valle. Programa Editorial, 1 edition, 2009.

[42] I. Sutskever A. Krizhevsky and G. E. Hinton. *ImageNet Classification with Deep Convolutional Neural Networks.* Commun. ACM, 1 edition, 2017.

[43] Carreon D. De León. *Desarrollo de una red neuronal convolucikonal para la identificación paramétrica de un robot.* Benemérita Universidad Autónoma de Puebla, Doctorate's tesis, 1 edition, 2023.

[44] J. X. Gu D. Zhen M. Xu Y. Li and A. Ball. An evaluation of gearbox condition monitoring using infrared therm. *Sensors*, 19(9):1–16, 2019.

[45] C. Fralick Chen, W-H Smith. A fast computational algorithm for the discrete cosine transform. 1977.

[46] Fernando R. Cortés. Robótica. control de robots manipuladores. 2011.

[47] K.J. str"om and B. Wit-tenmark. *Adaptive Control*. Dover Books on Electrical Engineering. Dover Publications, 2008.

[48] Ing Marín, María Vargas-Treviño, Sergio Vergara-Limon, Jesús López Gómez, Carlos L. C. Diaz D.L., and Amparo Merino. Desarrollo de una red neuronal convolucional para realizar la identificación paramétrica de un robot cartesiano de 3 grados de libertad. 03 2024.

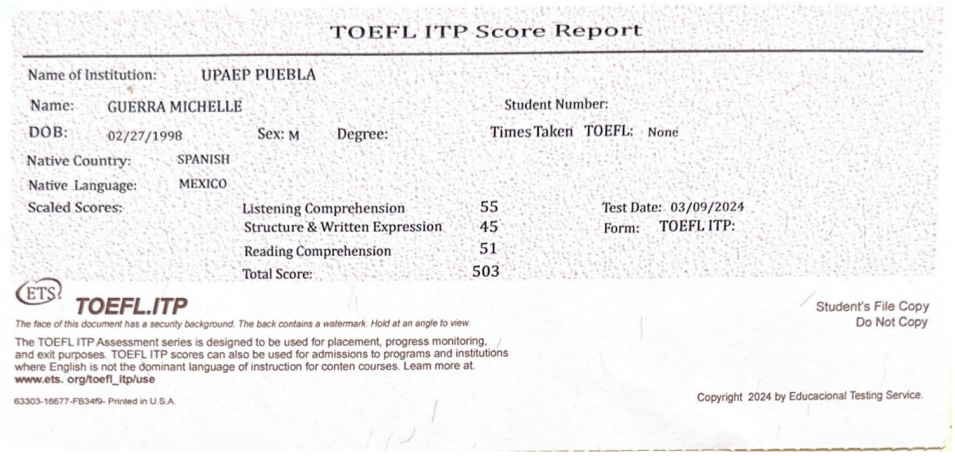
[49] W. Marilyn. *Computers as Components Principles of Embedded Computing System Design*. 5th Edition MK, 2022.

[50] J. Huang and A. Rosendo. Variable stiffness object recognition with a cnn-bayes classifier on a soft gripper. *Soft Robot*, pages 1–12, 2022.

[51] K. Ogata. *INGENIERIA DE CONTROL MODERNA.* MADRID: PRENTICE HALL HISPANOAMERICANA., 3 edition, 1998.

[52] R. L. BOYLESTAD. *Electrónica: teoría de circuitos y dispositivos electrónicos*. Pearson Educación, 10 edition, 2009.

Anexos



TOEFL ITP Score Report

Name of Institution: UPAEP PUEBLA

Name: GUERRA MICHELLE Student Number:

DOB: 02/27/1998 Sex: M Degree: Times Taken TOEFL: None

Native Country: SPANISH

Native Language: MEXICO

Scaled Scores:			
	Listening Comprehension	55	Test Date: 03/09/2024
	Structure & Written Expression	45	Form: TOEFL ITP:
	Reading Comprehension	51	
	Total Score:	503	

ETS TOEFL.ITP

The face of this document has a security background. The back contains a watermark. Hold at an angle to view.

The TOEFL ITP Assessment series is designed to be used for placement, progress monitoring, and exit purposes. TOEFL ITP scores can also be used for admissions to programs and institutions where English is not the dominant language of instruction for conten courses. Learn more at. **www.ets. org/toefl_itp/use**

63303-16677-FB34f9- Printed in U.S.A.

Student's File Copy
Do Not Copy

Desarrollo de una Red Neuronal Convolucional para Realizar la Identificación Paramétrica de un Robot Cartesiano de 3 Grados de Libertad

Ing. Michelle Guerra Marín[1], Dra. María Aurora Diozcora Vargas Treviño[2], Dr. Sergio Vergara Limon[3], Dr. Jesús López Gómez[4], Dr. Carlos Leopoldo Carreón Díaz de León[5] y Dra. Dora Amparo Palomino Merino[6]

***Resumen*—** En el presente trabajo, se presenta el desarrollo de una arquitectura de red neuronal convolucional diseñada con el objetivo específico de realizar la identificación paramétrica de un robot cartesiano de tres grados de libertad, el cual fue concebido y se encuentra instalado en las instalaciones del programa de Maestría en Ciencias de la Electrónica Opción en Automatización de la Benemérita Universidad Autónoma de Puebla (BUAP). El modelo dinámico de dicho robot ha sido obtenido mediante la metodología de parámetros agrupados, utilizando la ecuación de Lagrange. Se proporciona una descripción detallada de la configuración específica de la red neuronal, abordando la disposición y funcionalidad de sus capas. Además, se examina minuciosamente el método de entrenamiento que será implementado con el fin de optimizar los parámetros de la red para la identificación paramétrica.

***Palabras clave*—**Red neuronal, robótica, modelo dinámico, parámetros dinámicos, identificación paramétrica.

Introducción

Los robots manipuladores son formalmente descritos mediante ecuaciones diferenciales ordinarias, conocidas como modelo dinámico. Para que dicho modelo dinámico pueda asociarse de manera precisa con un robot real, es esencial que las constantes o parámetros dinámicos estén en concordancia con las propiedades físicas inherentes al robot en cuestión. Sin embargo, la medición de estos parámetros plantea un desafío, ya que el robot debe encontrarse en movimiento, dado que la mayoría de estos parámetros están vinculados a la velocidad y aceleración del robot. El proceso de extracción de parámetros de un robot se conoce como identificación paramétrica, y consiste en medir los parámetros utilizando las señales de movimiento y los datos provenientes de los actuadores disponibles. Dado que el modelo de un robot está definido por ecuaciones matemáticas, la tendencia en la identificación de los parámetros dinámicos implica el diseño de algoritmos basados en estas ecuaciones (C. Carreón, 2022).

Es importante señalar que, para el correcto funcionamiento del algoritmo de identificación, es necesario disponer de muestras de todas las variables contempladas en el modelo, tales como posiciones, velocidades y aceleraciones. No obstante, la tecnología de medición del movimiento de los robots presenta limitaciones, ya que normalmente solo permite la medición precisa de la posición mediante un encoder. Por consiguiente, las señales de velocidad y aceleración deben estimarse a partir de la información de posición disponible. Dentro del área del control de robots, los parámetros dinámicos desempeñan un papel crucial al compensar el modelo del robot para lograr una trayectoria de movimiento deseada. La ausencia de estos parámetros dinámicos complica la tarea de ajustar los sistemas de control, haciendo que dependan en gran medida de la experiencia para encontrar las constantes del controlador que posibiliten alcanzar una trayectoria deseada con el menor error posible.

Se presenta una metodología basada en redes neuronales para realizar la identificación paramétrica de un robot cartesiano de tres grados de libertad. El trabajo se organiza en secciones: revisión del estado del arte, metodología estudiada, los resultados y finalmente las conclusiones del artículo.

[1] Ing. Michelle Guerra Marín es Alumno de la Maestría en Ciencias de la Electrónica opción en Automatización en la Benemérita Universidad Autónoma de Puebla, Puebla, México. michelle.guerram@alumno.buap.mx **(autor corresponsal)**
[2] La Dra. María Aurora Diozcora Vargas Treviño es Profesora de la Maestría en Ciencias de la Electrónica Opción en Automatización en la Benemérita Universidad Autónoma de Puebla, Puebla, México. aurora.vargas@correo.buap.mx
[3] El Dr. Sergio Vergara Limon es Profesor de la Maestría en Ciencias de la Electrónica Opción en Automatización en la Benemérita Universidad Autónomas de Puebla. sergio.vergara@correo.buap.mx
[4] El Dr. Jesús López Gómez es Profesor de la División Académica de Ingeniería y Arquitectura, en la Universidad Juárez Autónoma de Tabasco (DAIA-UJAT), Tabasco, México. jlgo6599@docente.ujat.mx
[5] El Dr. Carlos Leopoldo Carreón Díaz de León es Profesor de la Facultad de Ciencias de la Computación en la Beneméritca Universidad Autónoma de Puebla, Puebla, México. carlos.carreon@correo.buap.mx
[6] La Dra. Amparo Dora Palomino Merino es Profesora de la Mestría en Ciencias de la Electrónica Opción en Automatización en la Benemérita Universidad Autónoma de Puebla, Puebla, México. amparo.palomino@correo.buap.mx

ACADEMIA JOURNALS
OPUS PRO SCIENTIA ET STUDIUM

Congreso Internacional de Investigación
Academia Journals **Los Mochis 2024**
Conectando Mentes, Innovando Horizontes

CONGRESO INTERNACIONAL DE INVESTIGACIÓN ACADEMIA JOURNALS LOS MOCHIS 2024

Conectando Mentes, Innovando Horizontes

CERTIFICADO

otorgado a

Ing. Michelle Guerra Marín
Dra. María Aurora Diozcora Vargas Treviño
Dr. Sergio Vergara Limón
Dr. Jesús López Gómez
Dr. Carlos Leopoldo Carreón Díaz de León
Dra. Amparo Dora Palomino Merino

por su artículo titulado

Desarrollo de una Red Neuronal Convolucional para Realizar la Identificación Paramétrica de un Robot Cartesiano de 3 Grados de Libertad

(Artículo No. MCH013)

La ponencia correspondiente fue presentada en el Congreso Internacional de Investigación Academia Journals Los Mochis 2024, *Conectando Mentes, Innovando Horizontes,* desarrollado los 26 y 27 de febrero de 2024, con sede en el Centro de Innovación y Educación Los Mochis. El evento fue presentado en colaboración con eminentes investigadores de Los Mochis. El artículo está incluido en las siguientes publicaciones: (1) en el portal de Internet AcademiaJournals.com, con ISSN 1946-5351 online, Vol. 16, No. 01, 2024 e indexación en la base de datos *Fuente Académica Plus de EBSCOHOST,* Massachusetts, Estados Unidos y (2) en libros ebook digitales compilados por área temática, con números ISBN online*. Se tiene acceso libre a todas las publicaciones del congreso en el portal de internet de Academia Journals.

Los organizadores del congreso reconocen la participación de los autores en el congreso, agradeciendo sus contribuciones.

DR. RAFAEL MORAS, P.E.
Director General
Academia Journals

CONGRESO INTERNACIONAL DE INVESTIGACIÓN ACADEMIA JOURNALS LOS MOCHIS 2024

Conectando Mentes e Innovando Horizontes en las Ciencias Administrativas – Los Mochis 2024 979-8-89020-057-0
Conectando Mentes e Innovando Horizontes en las Ciencias de la Salud – Los Mochis 2024 979-8-89020-055-6
Conectando Mentes e Innovando Horizontes en las Ciencias Naturales – Los Mochis 2024 979-8-89020-056-3
Conectando Mentes e Innovando Horizontes en las Ciencias de la Educación – Los Mochis 2024 979-8-89020-058-7
Conectando Mentes e Innovando Horizontes en las Humanidades y Ciencias Sociales – Los Mochis 2024 979-8-89020-059-4
Conectando Mentes e Innovando Horizontes en las Ingenierías – Los Mochis 2024 979-8-89020-060-0